Hindawi
International Journal of Antennas and Propagation
Volume 2019, Article ID 4583268, 17 pages
https://doi.org/10.1155/2019/4583268

Research Article

Synthesis of Sparse Concentric Ring Arrays Based on Fitness-Associated Differential Evolution Algorithm

Xujian Wang, Minli Yao, Fenggan Zhang, and Dingcheng Dai

Xi'an Research Institute of High Technology, Xi'an 710025, China

Correspondence should be addressed to Xujian Wang; wxj_903@163.com

Received 2 July 2019; Revised 3 September 2019; Accepted 21 November 2019; Published 21 December 2019

Academic Editor: N. Nasimuddin

In this paper, fitness-associated differential evolution (FITDE) algorithm is proposed and applied to the synthesis of sparse concentric ring arrays under constraint conditions, whose goal is to reduce peak sidelobe level. In unmodified differential evolution (DE) algorithm, crossover probability is constant and remains unchanged during the whole optimization process, resulting in the negative effect on the population diversity and convergence speed. Therefore, FITDE is proposed where crossover probability can change according to certain information. Firstly, the population fitness variance is introduced to the traditional differential evolution algorithm to adjust the constant crossover probability dynamically. The fitness variance in the earlier iterations is relatively large. Under this circumstance, the corresponding crossover probability shall be small to speed up the exploration process. As the iteration progresses, the fitness variance becomes small on the whole and the crossover probability should be set large to enrich population diversity. Thereby, we construct three variation strategies of crossover probability according to the above changing trend. Secondly, FITDE is tested on benchmark functions, and the best one of the three strategies is determined according to the test results. Finally, sparse concentric ring arrays are optimized using FITDE, of which the results are compared with reference algorithms. The optimization results manifest the advantageous effectiveness of FITDE.

1. Introduction

Antenna arrays are extensively applied both in the military field and the civilian field, such as radar, sonar, wireless commutation, and deep space exploration. Synthesis of antenna arrays is an essential research area which has vast prospect in practical applications. Concentric ring arrays (CRAs), as one kind of antenna arrays, have also aroused great interest in the past few years, especially for the one with uniform excitations but unequal elements spacing and different ring radii. Compared with the equally spaced arrays, the unequally spaced arrays have an ocean of merits. Under the condition of the same array aperture, the unequally spaced arrays can reduce the elements number and therefore save the cost. Under the condition of the same elements number, the array aperture can be lengthened, and therefore, the main lobe will be narrowed, improving the spatial resolution and enhancing the directivity of the beam. Besides the above advantages, for CRAs, they have an outstanding characteristic of pattern symmetry and can achieve omnidirectional scanning [1–5].

Generally, the unequally spaced arrays can be classified into two categories. One is to "close" part of array elements on the basis of equally spaced array, which is called thinned arrays. The other is sparse arrays, which can be obtained by arranging elements randomly under the premise of satisfying the constraint conditions, and those conditions usually include the elements number, the array aperture, and the minimum elements spacing. Compared with thinned arrays, sparse arrays have more degrees of freedom and better performance. The pattern function of the antenna array is a complex exponential function of the elements position, which is restricted by several constraints, causing that the optimization problems in the array synthesis are often highly nonlinear with multidimensional variables, with the fact that existing numerical analysis optimization methods are no longer suitable. In recent years, population-based stochastic methods are proven to be effective for these problems, such

as genetic algorithm (GA) [6–11], particle swarm optimization (PSO) [12–14], differential evolution algorithm (DE) [15–20], invasive weed optimization (IWO) [21], and cuckoo search (CS) algorithm [22–24]. Besides, Bayesian compressive sensing (BCS) [25–31] and matrix pencil method (MPM) [32–34] have also aroused great interest.

As an effective stochastic method, DE has the advantages of simple operation, rich population diversity, and strong global search ability and has broad application space in the optimization of antenna arrays. To improve the performance of DE, numerous modified variants have appeared. Ghosh and Das [15] proposed the differential evolution with global and local neighbourhoods (DEGL) algorithm. DEGL defines a neighbour range centering on the i-th individual and then generates a local donor vector and a global donor vector based on the neighbour range and the whole population, respectively, rather than only a global donor vector. Then the actual donor vector is generated by merging the local and global donor vectors through a scaling weight w, which can control the balance between the exploration and exploitation capabilities. Mandal et al. [17] developed a hybrid mutation strategy to prevent the algorithm from stagnating at any local optima. Besides, an efficient classic local search technique called Solis–Wet's algorithm is introduced to the above process to improve the search capability. Kerim and Suad [18] divided the whole population into several subpopulations and applied the method to the null synthesis of time-modulated circular antenna arrays. A modified DE algorithm which utilized the strategies of best of random mutation and randomized local search was put forward in [19], making a good balance between convergence speed and population diversity. Dai et al. [20] came up with a new strategy based on DE and Lévy flight (DELF) to update populations, in which the Lévy flight can help jump from a local optimum, while DE can search a global optimal solution efficiently.

In this paper, CRAs are optimized based on our proposed fitness-associated differential evolution (FITDE) algorithm. Firstly, we introduce the concept of population fitness variance to adjust crossover probability dynamically which remains unchanged in traditional DE and construct three crossover probability variation strategies according to the relationship between population fitness variance and crossover probability. Secondly, FITDE is tested on benchmark functions, and the test results prove the advantageous effectiveness of FITDE. Furthermore, the best one of the three changing strategies is also determined through the results. Finally, FITDE is applied to optimize three concentric ring arrays under different conditions. Simulation results show that our method can reduce the peak sidelobe level of the sparse arrays and obtain better optimization performance.

2. Sparse Concentric Ring Arrays

Figure 1 demonstrates the geometry of CRA, which has m rings with the same centre but different radii. It is noted that the element fixed in the centre is not included in m. r_m and N_m denote the radius of the m-th ring (where the centre, $r_0 = 0$) and the number of elements on this ring, respectively. φ_{mn} represents the azimuth angle of the n-th element on the m-th ring.

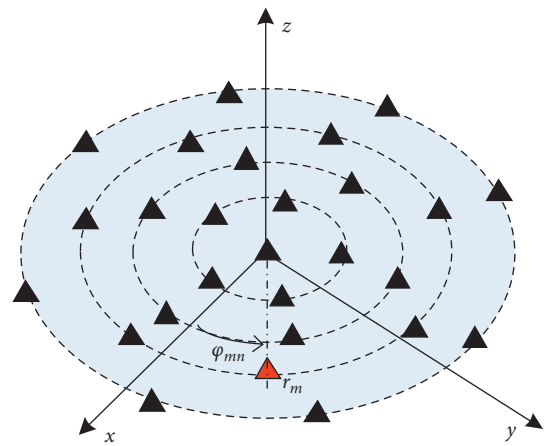

Figure 1: Geometry of CRA, where r_m is the radius of the m-th ring and φ_{mn} is the azimuth angle of the n-th element on the m-th ring.

Theoretically, array factor (AF) for CRAs can be calculated as

$$\mathrm{AF}(u,v) = 1 + \sum_{m=1}^{M}\sum_{n=1}^{N_m} I_{mn}\exp\left[jkr_m\left(\cos\varphi_{mn}u + \sin\varphi_{mn}v\right)\right], \tag{1}$$

where $k = 2\pi/\lambda$ is the wave number and λ is the wavelength. $(r_m\cos\varphi_{mn}, r_m\sin\varphi_{mn})$ is the coordinate of the n-th element on the m-th ring. $u = \sin\theta\cos\varphi_{mn}$, and $v = \sin\theta\sin\varphi_{mn}$. I_{mn} represents the excitation current of the n-th element on the m-th ring. It should be noted that for CRAs which are excited uniformly, the amplitude and phase of I_{mn} are the same, which are often set as 1 for convenience.

For uniform CRAs, the spacing between any two adjacent rings is the same, and the spacing between any two adjacent elements on each ring is also the same. Figure 2(a) shows a CRA with a ring spacing of 0.6λ and elements spacing of 0.5λ, and Figure 2(b) shows the radiation pattern of this CRA. As can be seen from Figure 2(b), the radiation pattern of uniform CRAs has an excellent symmetrical feature, which is exactly the superiority of CRAs to linear arrays and planar arrays. When synthesizing concentric ring arrays, if element are arranged completely randomly, the symmetrical feature of CRAs will be destroyed. Therefore, we pay more attention to the arrays which can maintain CRAs' symmetrical feature, as are usually discussed in most references. To avoid destroying the symmetry of the radiation pattern of CRAs, elements on the same ring are equally distributed; that is to say, the spacing of two adjacent elements on the same ring is the same.

For CRAs, the optimization goal can be expressed as

$$f(\mathbf{R},\mathbf{D}) = \max_{u,v\in\text{sidelobe}}\left|\frac{\mathrm{AF}(u,v)}{\mathrm{AF}_{\max}}\right|, \tag{2}$$

where $\mathrm{AF}_{\max}$ represents the peak level of the main lobe and $\mathrm{AF}(u,v)$ is the array factor when (u,v) falls into sidelobe.

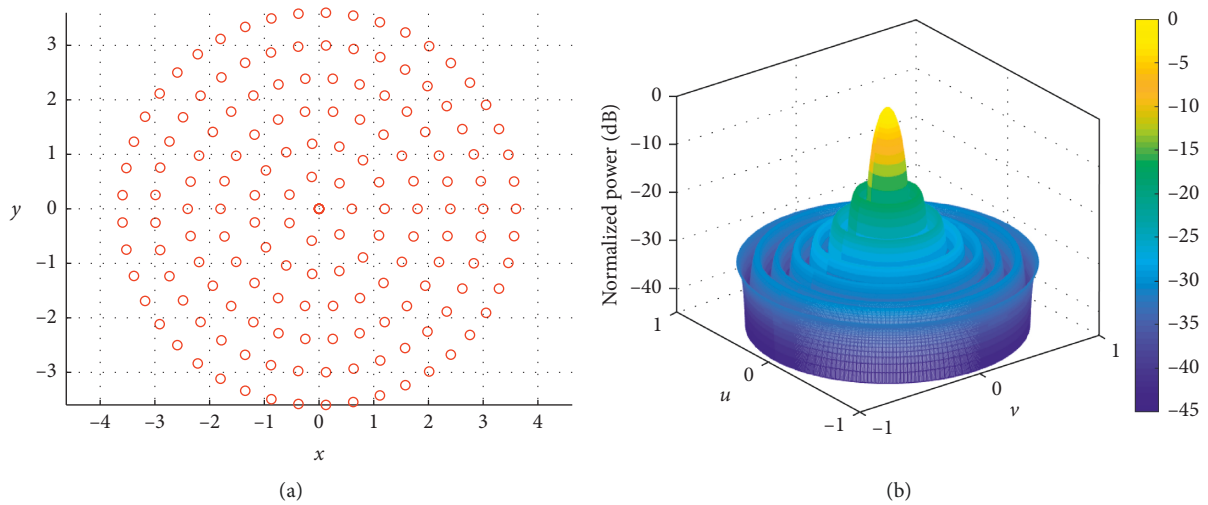

Figure 2: Uniform CRA. (a) Geometry of the CRA, of which ring spacing is 0.6λ and elements spacing is 0.5λ. (b) Radiation pattern, which has an excellent symmetrical feature.

$\mathbf{R} = [r_1, r_2, \ldots, r_M]$, and $\mathbf{D} = [d_1, d_2, \ldots, d_M]$. $d_m (m = 1, 2, \ldots, M)$ denotes the distance along circular arc between two adjacent elements. For simplifying calculation, d_m can represent straight-line distance between two elements approximately. Therefore, the optimization model of CRAs can be expressed as

$$\left.\begin{aligned} &\min\{f(\mathbf{R}, \mathbf{D})\} \\ &\text{s.t.} \quad r_0 = 0, r_M = L \\ &r_1 \geq d_c, r_i - r_j \geq d_c, 1 \leq j < i \leq M \\ &d_m \geq d_c, m = 1, 2, \ldots, M \end{aligned}\right\}, \tag{3}$$

where d_c is the minimum ring spacing and the minimum elements spacing, which is often set as 0.5λ.

In order to satisfy the constraint of the minimum ring spacing, we reduce the radii search scope from $[0, L]$ to $[0, \mathrm{SP}]$, $\mathrm{SP} = L - M \times d_c$. Generate $M - 1$ random numbers from $[0, \mathrm{SP}]$, sort them from smallest to largest as $\mathbf{X} = [x_1, x_2, \ldots, x_{M-1}]$, and then optimize $\mathbf{X}$ using our proposed method. The radius of the ring can be calculated by the following formula:

$$\mathbf{R} = \begin{bmatrix} r_1 \\ r_2 \\ \vdots \\ r_{M-1} \\ r_M \end{bmatrix} = \begin{bmatrix} dc \\ 2dc \\ \vdots \\ (M-1)dc \\ L \end{bmatrix} + \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_{M-1} \\ 0 \end{bmatrix}. \tag{4}$$

Ring spacing between r_1 and r_0 is

$$\Delta r_1 = r_1 - 0 = d_c + x_1. \tag{5}$$

Ring spacing between r_i and $r_{i-1} (1 < i < M)$ is

$$\begin{aligned} \Delta r_i &= r_i - r_{i-1} \\ &= i d_c + x_i - [(i-1)d_c + x_{i-1}] \\ &= d_c + (x_i - x_{i-1}). \end{aligned} \tag{6}$$

Ring spacing between r_M and r_{M-1} is

$$\begin{aligned} \Delta r_M &= r_M - r_{M-1} \\ &= L - [(M-1)d_c + x_{M-1}] \\ &= d_c + (L - M d_c) - x_{M-1} \\ &= d_c + \mathrm{SP} - x_{M-1}. \end{aligned} \tag{7}$$

$x_i (1 \leq i \leq M)$ are all generated from $[0, \mathrm{SP}]$, and $x_i \geq x_{i-1}$; therefore, $d_c + x_1$ in (5), $d_c + (x_i - x_{i-1})$ in (6), and $d_c + \mathrm{SP} - x_{M-1}$ in (7) are all bigger than or equal to d_c, which can make ring radii generated according to (4) satisfy the minimum spacing constraint.

3. Fitness-Associated Differential Evolution Algorithm

3.1. Crossover Probability Associated with Fitness. Crossover probability is an important parameter in the DE algorithm. The larger the crossover probability is, the more information the new individual will inherit from the mutant individual, which is beneficial to global exploration. The smaller the crossover probability is, the less information the new individual will inherit from the mutant individual and the more the information from parents, which is beneficial to local search. In the traditional DE algorithm, crossover probability remains unchanged in the whole evolution process and cannot be adjusted according to evolution process, doing harm to the population diversity and convergence speed. On the one hand, if crossover probability is still big when individuals in the population are pretty dispersed, the convergence speed will be slowed down. On the other hand, if crossover probability is still small when the distribution of individuals is relatively concentrated, the population diversity will be reduced, and the algorithm is likely to fail to get global optimum. In order to solve this problem, fitness-associated differential evolution (FITDE) is

proposed, which adjusts crossover probability according to the change of population fitness in the process of evolution.

Definition 1. Let NP denote population size, f_i denote the fitness of i-th individual, and f_{avg} denote average fitness of the current population. Then the fitness variance of population can be defined as

$$s = \frac{1}{\text{NP}} \sum_{i=1}^{\text{NP}} \left| \frac{f_i - f_{\text{avg}}}{\max\left\{\left|f_i - f_{\text{avg}}\right|\right\}} \right|^2, \tag{8}$$

where $\max\{|f_i - f_{\text{avg}}|\}$ is the added scaling factor on the basis of mathematical definition of variance, which can guarantee s to fall into the range $[0, 1]$.

s reflects the aggregation degree of individuals in the population. The larger s is, the more dispersed the individuals are, and the population is in the stage of random search. On the contrary, the smaller s is, the more concentrated the individuals are, and the population is in the stage of local search stage. Therefore, for crossover probability associated with population fitness, when s is small, crossover probability should be defined as a large value to break up the individual clustering phenomenon and enrich the diversity of the population, and when s is large, crossover probability should be defined as a small value to speed up the convergence.

Definition 2. According to the above analysis, there ought to be a negative correlation between crossover probability and population fitness variance. Thereby, we construct three changing strategies for crossover probability, which are marked as CR1, CR2, and CR3. Figure 3 depicts the changing curves of crossover probability along with population fitness variance:

$$\begin{aligned} \text{CR1} &= (\text{CR}_{\min} - \text{CR}_{\max}) \cdot s^2 + \text{CR}_{\max}\,(a), \\ \text{CR2} &= (\text{CR}_{\min} - \text{CR}_{\max}) \cdot s + \text{CR}_{\max}\,(b), \\ \text{CR3} &= (\text{CR}_{\max} - \text{CR}_{\min}) \cdot (s-1)^2 + \text{CR}_{\min}\,(c). \end{aligned} \tag{9}$$

3.2. Flow of FITDE

Step 1. Initialize parameters and generate initial population.

Set values for population size NP, dimension D, scaling factor F, maximum iterations G, minimum crossover probability $\text{CR}_{\min}$, maximum crossover probability $\text{CR}_{\max}$, and search scope $[l^0, u^0]$. Generate NP individuals within search scope randomly and calculate the population fitness.

Step 2. Determine crossover probability.

Calculate fitness variance and crossover probability according to (8) and (9).

Step 3. Do mutation, crossover, and selection operations according to the following (10), (11), and (12), respectively:

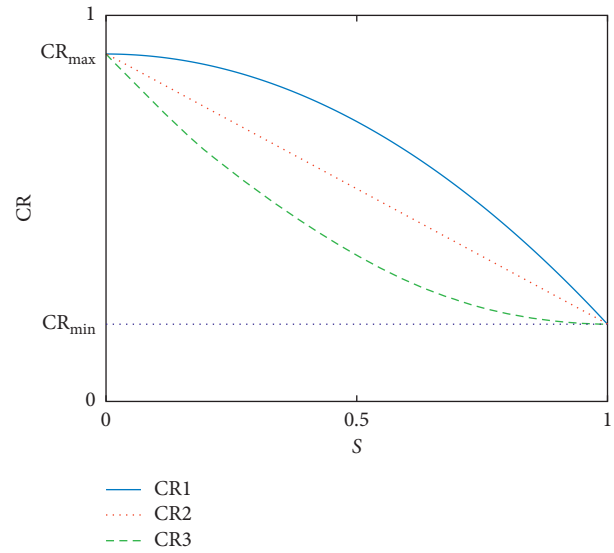

Figure 3: Changing curves of crossover probability. CR1 is the part of a convex quadratic curve, CR2 is the part of a linear curve, and CR3 is the part of a concave quadratic curve.

$$x_m = x_{r_1}^t + F \times \left(x_{r_2}^t - x_{r_3}^t\right), \tag{10}$$

$$x_{\text{new},j} = \begin{cases} x_{mj}, & \text{rand}_j \le \text{CR}, \\ x_{ij}, & \text{rand}_j > \text{CR}, \end{cases} \tag{11}$$

$$x_i^{t+1} = \begin{cases} x_{\text{new}}, & f(x_{\text{new}}) \le f(x_i^t), \\ x_i^t, & f(x_{\text{new}}) > f(x_i^t), \end{cases} \tag{12}$$

where r_1, r_2, and r_3 are different individuals generated randomly from the population. x_m is on behalf of mutant individual. x_{new} represents the individual after crossover operation. x_i^{t+1} stands for the individual passed on to the next generation after differential evolution.

Step 4. End the algorithm if the terminal condition is satisfied. Otherwise, let $t = t + 1$ and skip to Step 2.

Pseudocode of FITDE is summarized in Algorithm 1, and the flowchart of FITDE is presented in Figure 4.

Compared with traditional DE algorithm, although FITDE adds two calculation steps, namely, the calculation of fitness variance and the calculation of crossover probability, they are not so sophisticated and will not add much calculation burden to the method. Moreover, FITDE can make full use of the information in the evolutionary process, that is, the fitness value, to adjust the evolutionary process dynamically, which can improve the information utilization rate and algorithm efficiency, and reduce the redundancy and waste.

3.3. Benchmark Functions Test. We excerpt 10 benchmark functions from [35] to test the performance of FITDE. They are listed in Table 1. The parameters in the test are set as shown in Table 2.

```
Initialize a population of NP individuals x_i, i = 1, 2, ..., NP
for all x_i do
   Calculate fitness f(x_i)
end for
while t ≤ G do
   Determine CR according to population fitness variance
   Generate mutant individual x_m
   for each component j (j = 1, 2, ..., D) of x_i and x_m do
      if rand < CR
         x_new,j = x_mj
      else
         x_new,j = x_ij
      end if
   end for
   if f(x_new) < f(x_i) then
      Replace x_i by x_new; replace f(x_i) by f(x_new)
   end if
end while
```

Algorithm 1: Pseudocode of FITDE.

To test the performance of FITDE under different dimensions, we conduct two groups of experiments with $D = 50$ and $D = 100$. Four methods, DE, CR1-DE, CR2-DE, and CR3-DE all run 50 times independently. To demonstrate the performance of FITDE, we compare FITDE with DELF, PSO, IWO, HS, and CS. For the sake of the comparison, the following three criteria are adopted: (1) mean error, where the error is defined as the difference between the result of an independent run and the global optimum which has been known accurately; (2) standard deviation of error; and (3) percentage of successful runs (PSR), where the reference value is set as 10^{-12}. One run is regarded as a successful one if its result is better than or equal to 10^{-12}. Test results are summarized in Table 3, where the best outcomes are marked in bold font.

We introduce state variables 0 and 1 to mark the test results and take the mean error of DE as the comparison benchmark. If the mean error of a method is better than or equal to that of DE, the state of this method is denoted as 1; otherwise, it is 0. States of CR1-DE, CR2-DE, and CR3-DE are shown in Table 4. At the same time, we also compare different changing strategies of crossover probability in Table 4 and mark the best one of three strategies CR1-DE, CR2-DE, and CR3-DE that has better results with "∗." If there is no mark, it indicates that there is no difference among the three methods in terms of the mean error value.

As can be seen from Table 3, among DE, CR1-DE, CR2-DE, and CR3-DE, when $D = 50$, CR1-DE, where crossover probability changes according to (9(a)), has 9 better optimization results than DE for all functions other than f_4. CR2-DE has 6 better results than DE, and CR3-DE has 7 better results than DE. In addition, for f_1, f_2, f_6, f_9, and f_{10}, when the reference value is set as 10^{-12}, PSR of four methods are all 100%, while for other functions, PSR are all 0, meaning that the optimization results of four methods are all worse than 10^{-12}. When $D = 100$, CR1-DE, CR2-DE, and CR3-DE have 7, 8, and 8 better results than DE, respectively. In terms of PSR, due to the increase of dimension, four methods' PSR for f_2 and f_9 decrease to various degrees, although PSR for f_1, f_6, and f_{10} are still 100%. Optimization results of DE and CR3-DE for f_2 and f_9 are all worse than 10^{-12}. However, CR1-DE and CR2-DE can still achieve 100% under the given reference value for f_2. For f_9, PSR of CR1-DE and CR2-DE decrease to 88% and 96%, respectively.

When compared with other optimization algorithms, including DELF, PSO, IWO, HS, and CS, FITDE (CR1-DE, CR2-DE, and CR3-DE) also demonstrates its good performance. In terms of mean error, when $D = 50$, FITDE has 9 better results than reference algorithms, with only the result of f_4 worse than reference algorithms. The best result of f_4 is obtained using DELF. When $D = 100$, FITDE has 7 better results than reference algorithms except for f_3, f_4, and f_9, and the best results of these three functions are acquired using IWO, CS, and DELF, respectively.

As can be seen from Table 4, for three different crossover probability strategies, when $D = 50$, the optimization results of CR1-DE for f_1, f_2, f_3, f_7, and f_8 are superior to those of CR2-DE and CR3-DE. When $D = 100$, the optimization results of CR1-DE for f_1, f_2, f_3, f_6, f_7, and f_8 are superior to those of CR2-DE and CR3-DE. That is to say, the overall performance of CR1-DE is better than those of CR2-DE and CR3-DE. This is because, in the early stage of evolution, population fitness variance is large, and the crossover probability calculated according to crossover probability changing strategies is relatively small. However, scattered state of individuals in the early stage of evolution is a normal random search phenomenon, and large fitness variance should be allowed in this stage. At the initial stage, the crossover probability calculated by CR1 is larger than that calculated by CR2 and CR3, which can ensure the normal random search of the population. If crossover probability is set according to CR2 and CR3, it may make the population stagnated at the initial stage of evolution and unable to carry out the subsequent search process. In the later stage of evolution, the larger

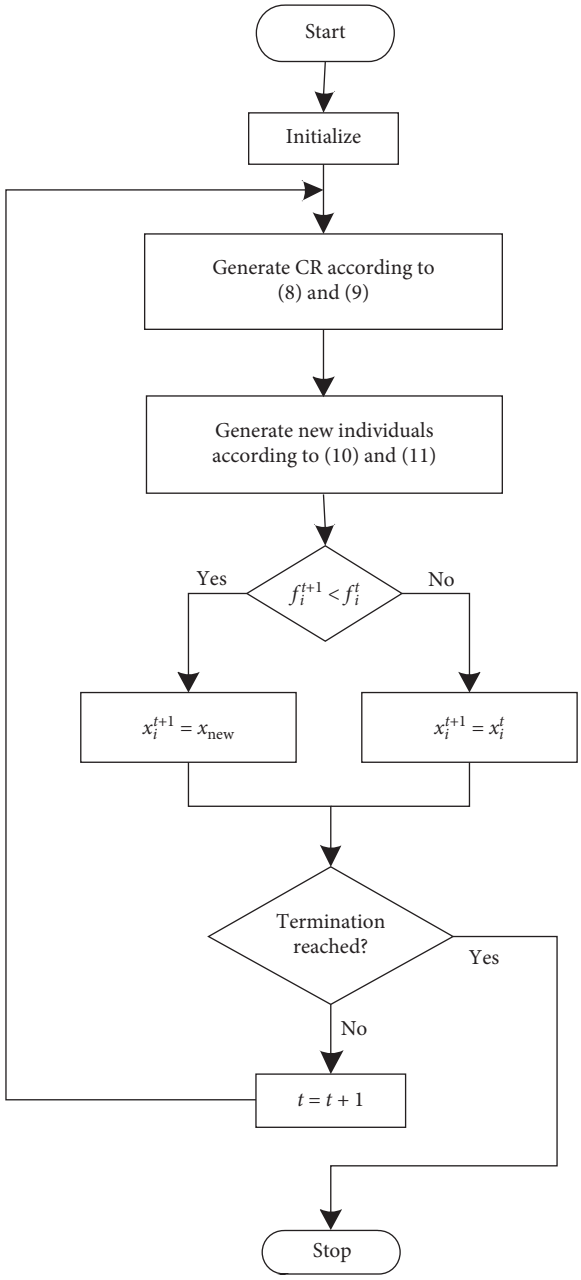

Figure 4: Flowchart of FITDE.

Table 1: Ten benchmark functions.

Function		Range of x	Min
f_1	$\sum_{i=1}^{n} x_i^2$	$[-100, 100]$	0
f_2	$\sum_{i=1}^{n} \lvert x_i^2 \rvert + \prod_{i=1}^{n} x_i$	$[-10, 10]$	0
f_3	$\sum_{i=1}^{n} (\sum_{j=1}^{i} x_j)^2$	$[-100, 100]$	0
f_4	$\max\{\lvert x_i \rvert\}$	$[-100, 100]$	0
f_5	$\sum_{i=1}^{D-1} [100 (x_{i+1} - x_1^2)^2 + (x_i - 1)^2]$	$[-30, 30]$	0
f_6	$\sum_{i=1}^{D} (\lvert x_i + 0.5 \rvert)^2$	$[-100, 100]$	0
f_7	$\sum_{i=1}^{D} i x_i^4 + \text{random}[0, 1)$	$[-1.28, 1.28]$	0
f_8	$\sum_{i=1}^{D} [x_i^2 - 10 \cos(2\pi x_i) + 10]$	$[-5.12, 5.12]$	0
f_9	$-20e^{-0.2\sqrt{1/D \sum_{i=1}^{D} x_i^2}} - e^{1/D \sum_{i=1}^{D} \cos(2\pi x_i)} + 20 + e$	$[-32, 32]$	0
f_{10}	$1/100 \sum_{i=1}^{D} x_i^2 - \prod_{i=1}^{D} \cos(x_i/\sqrt{i}) + 1$	$[-600, 600]$	0

Table 2: Parameters in the test.

	CR	NP	D	G	F
FITDE	[0.2, 0.9]	100	50/100	10000	0.5
DE	0.5				

Table 3: Test results for ten benchmark functions at $D = 50$ and $D = 100$.

		$D = 50$			$D = 100$		
		Mean	SD	PSR (%)	Mean	SD	PSR (%)
f_1	DE	3.52×10^{-64}	6.76×10^{-128}	100	2.00×10^{-14}	5.29×10^{-29}	100
	CR1-DE	$\mathbf{5.63 \times 10^{-76}}$	$\mathbf{1.46 \times 10^{-150}}$	100	$\mathbf{1.16 \times 10^{-40}}$	$\mathbf{2.09 \times 10^{-80}}$	100
	CR2-DE	1.04×10^{-56}	2.74×10^{-112}	100	7.59×10^{-26}	4.05×10^{-51}	100
	CR3-DE	2.79×10^{-50}	6.18×10^{-100}	100	1.68×10^{-16}	1.67×10^{-32}	100
	DELF	4.90×10^{-44}	1.05×10^{-87}	100	8.90×10^{-21}	1.97×10^{-41}	100
	PSO	6.72×10^{-58}	6.49×10^{-96}	100	3.03×10^{-27}	8.16×10^{-52}	100
	IWO	2.20×10^{-54}	3.92×10^{-90}	100	1.07×10^{-32}	5.50×10^{-45}	100
	HS	3.71×10^{-67}	6.09×10^{-102}	100	6.23×10^{-30}	3.61×10^{-36}	100
	CS	2.05×10^{-62}	2.46×10^{-118}	100	1.36×10^{-23}	9.82×10^{-34}	100
f_2	DE	4.60×10^{-37}	6.36×10^{-74}	100	8.42×10^{-09}	2.00×10^{-18}	0
	CR1-DE	$\mathbf{5.49 \times 10^{-38}}$	$\mathbf{5.17 \times 10^{-75}}$	100	$\mathbf{3.80 \times 10^{-21}}$	$\mathbf{1.98 \times 10^{-41}}$	100
	CR2-DE	3.38×10^{-29}	1.39×10^{-57}	100	1.73×10^{-13}	1.73×10^{-26}	100
	CR3-DE	1.36×10^{-26}	1.74×10^{-52}	100	4.38×10^{-09}	3.73×10^{-18}	0
	DELF	3.00×10^{-24}	1.24×10^{-48}	100	7.09×10^{-11}	2.68×10^{-22}	0
	PSO	6.47×10^{-20}	2.56×10^{-51}	100	3.06×10^{-13}	8.57×10^{-26}	100
	IWO	5.03×10^{-31}	6.61×10^{-62}	100	8.42×10^{-14}	1.38×10^{-30}	100
	HS	1.83×10^{-32}	5.77×10^{-61}	100	2.44×10^{-15}	3.65×10^{-12}	92
	CS	2.41×10^{-37}	4.01×10^{-53}	100	3.74×10^{-12}	1.90×10^{-08}	83
f_3	DE	$7.46 \times 10^{+04}$	$4.06 \times 10^{+07}$	0	$3.40 \times 10^{+05}$	$5.80 \times 10^{+08}$	0
	CR1-DE	$\mathbf{1.21 \times 10^{-02}}$	1.37×10^{-04}	0	$1.16 \times 10^{+04}$	$1.78 \times 10^{+07}$	0
	CR2-DE	$3.37 \times 10^{+03}$	$2.53 \times 10^{+06}$	0	$2.22 \times 10^{+05}$	$9.18 \times 10^{+08}$	0
	CR3-DE	$4.36 \times 10^{+04}$	$7.83 \times 10^{+07}$	0	$3.00 \times 10^{+05}$	$6.54 \times 10^{+08}$	0
	DELF	$2.66 \times 10^{+01}$	$3.31 \times 10^{+01}$	0	$1.60 \times 10^{+02}$	$8.12 \times 10^{+02}$	0
	PSO	$1.02 \times 10^{+04}$	$4.38 \times 10^{+07}$	0	$4.55 \times 10^{+04}$	$2.68 \times 10^{+08}$	0
	IWO	1.56×10^{-02}	$\mathbf{5.59 \times 10^{-08}}$	0	$\mathbf{3.24 \times 10^{+00}}$	$\mathbf{1.01 \times 10^{+00}}$	0
	HS	$2.02 \times 10^{+04}$	$1.90 \times 10^{+07}$	0	$2.45 \times 10^{+05}$	$5.61 \times 10^{+08}$	0
	CS	2.27×10^{-02}	5.22×10^{-04}	0	$5.72 \times 10^{+01}$	$9.53 \times 10^{+01}$	0
f_4	DE	7.14×10^{-03}	3.70×10^{-06}	0	$8.65 \times 10^{+01}$	$2.16 \times 10^{+01}$	0
	CR1-DE	$7.65 \times 10^{+00}$	$1.64 \times 10^{+01}$	0	$9.32 \times 10^{+01}$	$1.09 \times 10^{+02}$	0
	CR2-DE	9.68×10^{-01}	$2.23 \times 10^{+00}$	0	$9.21 \times 10^{+01}$	$1.71 \times 10^{+02}$	0
	CR3-DE	1.75×10^{-03}	7.96×10^{-05}	0	$9.09 \times 10^{+01}$	$2.45 \times 10^{+02}$	0
	DELF	$\mathbf{6.34 \times 10^{-05}}$	$\mathbf{1.95 \times 10^{-10}}$	0	1.38×10^{-01}	4.50×10^{-03}	0
	PSO	$1.12 \times 10^{+01}$	$3.94 \times 10^{+00}$	0	$1.56 \times 10^{+01}$	$4.59 \times 10^{+00}$	0
	IWO	5.30×10^{-03}	1.38×10^{-07}	0	1.22×10^{-01}	1.28×10^{-01}	0
	HS	$9.19 \times 10^{+00}$	4.99×10^{-01}	0	$4.19 \times 10^{+01}$	7.51×10^{-01}	0
	CS	1.64×10^{-01}	5.71×10^{-04}	0	$\mathbf{1.67 \times 10^{+00}}$	$\mathbf{5.24 \times 10^{-02}}$	0
f_5	DE	$2.09 \times 10^{+01}$	5.66×10^{-01}	0	$8.94 \times 10^{+01}$	$\mathbf{1.27 \times 10^{-01}}$	0
	CR1-DE	1.59×10^{-01}	6.23×10^{-01}	0	$1.01 \times 10^{+02}$	$1.25 \times 10^{+03}$	0
	CR2-DE	$\mathbf{8.34 \times 10^{-04}}$	$\mathbf{1.13 \times 10^{-06}}$	0	$\mathbf{7.86 \times 10^{+01}}$	$2.13 \times 10^{+02}$	0
	CR3-DE	$1.05 \times 10^{+00}$	3.33×10^{-01}	0	$8.31 \times 10^{+01}$	$8.24 \times 10^{+01}$	0
	DELF	$3.43 \times 10^{+01}$	$2.07 \times 10^{+00}$	0	$8.79 \times 10^{+01}$	3.16×10^{-01}	0
	PSO	$2.93 \times 10^{+01}$	$3.36 \times 10^{+01}$	0	$2.17 \times 10^{+05}$	$1.74 \times 10^{+10}$	0
	IWO	$5.93 \times 10^{+01}$	$2.93 \times 10^{+03}$	0	$1.32 \times 10^{+02}$	$7.34 \times 10^{+03}$	0
	HS	$1.36 \times 10^{+03}$	$1.56 \times 10^{+05}$	0	$3.77 \times 10^{+06}$	$2.67 \times 10^{+11}$	0
	CS	$4.71 \times 10^{+01}$	$1.45 \times 10^{+00}$	0	$2.13 \times 10^{+02}$	$5.33 \times 10^{+02}$	0

Table 3: Continued.

		$D = 50$			$D = 100$		
		Mean	SD	PSR (%)	Mean	SD	PSR (%)
f_6	DE	0	0	100	1.99×10^{-14}	6.35×10^{-29}	100
	CR1-DE	0	0	100	$\mathbf{4.73 \times 10^{-32}}$	$\mathbf{1.84 \times 10^{-63}}$	100
	CR2-DE	0	0	100	1.07×10^{-25}	1.62×10^{-50}	100
	CR3-DE	0	0	100	2.19×10^{-16}	3.94×10^{-32}	100
	DELF	0	0	100	8.56×10^{-21}	1.27×10^{-41}	100
	PSO	0	0	100	3.10×10^{-17}	9.39×10^{-35}	100
	IWO	0	0	100	1.09×10^{-20}	4.10×10^{-25}	100
	HS	0	0	100	6.14×10^{-24}	2.85×10^{-27}	100
	CS	0	0	100	1.48×10^{-21}	1.37×10^{-22}	100
f_7	DE	8.03×10^{-03}	1.16×10^{-06}	0	7.22×10^{-02}	5.75×10^{-05}	0
	CR1-DE	$\mathbf{3.42 \times 10^{-03}}$	$\mathbf{7.39 \times 10^{-07}}$	0	$\mathbf{1.65 \times 10^{-02}}$	1.77×10^{-05}	0
	CR2-DE	5.53×10^{-03}	1.55×10^{-06}	0	1.71×10^{-02}	1.26×10^{-05}	0
	CR3-DE	7.85×10^{-03}	2.84×10^{-06}	0	3.58×10^{-02}	4.64×10^{-05}	0
	DELF	7.01×10^{-03}	2.14×10^{-06}	0	1.89×10^{-02}	$\mathbf{1.06 \times 10^{-05}}$	0
	PSO	2.06×10^{-01}	2.79×10^{-02}	0	1.23×10^{-01}	3.95×10^{-01}	0
	IWO	5.11×10^{-03}	2.50×10^{-06}	0	2.20×10^{-02}	2.27×10^{-05}	0
	HS	6.99×10^{-02}	1.81×10^{-04}	0	$5.20 \times 10^{+00}$	4.85×10^{-01}	0
	CS	1.10×10^{-02}	9.37×10^{-06}	0	8.05×10^{-02}	4.31×10^{-04}	0
f_8	DE	$2.40 \times 10^{+02}$	$1.24 \times 10^{+02}$	0	$7.66 \times 10^{+02}$	$4.52 \times 10^{+02}$	0
	CR1-DE	$\mathbf{8.57 \times 10^{+01}}$	$2.38 \times 10^{+03}$	0	$\mathbf{2.54 \times 10^{+02}}$	$3.08 \times 10^{+04}$	0
	CR2-DE	$2.56 \times 10^{+02}$	$1.05 \times 10^{+03}$	0	$7.17 \times 10^{+02}$	$2.07 \times 10^{+03}$	0
	CR3-DE	$2.99 \times 10^{+02}$	$3.33 \times 10^{+02}$	0	$7.96 \times 10^{+02}$	$8.25 \times 10^{+02}$	0
	DELF	$2.65 \times 10^{+02}$	$1.51 \times 10^{+01}$	0	$1.43 \times 10^{+03}$	$1.30 \times 10^{+02}$	0
	PSO	$1.68 \times 10^{+02}$	9.18×10^{-02}	0	$4.99 \times 10^{+02}$	$2.31 \times 10^{+03}$	0
	IWO	$3.55 \times 10^{+02}$	$1.46 \times 10^{+03}$	0	$7.52 \times 10^{+02}$	$3.83 \times 10^{+03}$	0
	HS	$5.72 \times 10^{+02}$	$\mathbf{1.60 \times 10^{+00}}$	0	$1.13 \times 10^{+03}$	$\mathbf{5.46 \times 10^{+01}}$	0
	CS	$2.14 \times 10^{+02}$	$4.05 \times 10^{+02}$	0	$6.22 \times 10^{+02}$	$1.13 \times 10^{+03}$	0
f_9	DE	7.99×10^{-15}	0	100	2.81×10^{-08}	1.49×10^{-17}	0
	CR1-DE	7.57×10^{-15}	1.36×10^{-30}	100	$2.36 \times 10^{+00}$	$4.18 \times 10^{+01}$	88
	CR2-DE	$\mathbf{7.43 \times 10^{-15}}$	1.73×10^{-30}	100	7.98×10^{-01}	$1.56 \times 10^{+01}$	96
	CR3-DE	7.78×10^{-15}	$\mathbf{7.26 \times 10^{-31}}$	100	5.28×10^{-10}	5.31×10^{-20}	0
	DELF	7.64×10^{-15}	1.16×10^{-30}	100	$\mathbf{2.20 \times 10^{-11}}$	$\mathbf{3.32 \times 10^{-23}}$	0
	PSO	7.82×10^{-15}	1.85×10^{-30}	100	$9.24 \times 10^{+00}$	9.53×10^{-01}	83
	IWO	1.79×10^{-14}	2.09×10^{-25}	100	1.93×10^{-01}	2.34×10^{-02}	0
	HS	2.11×10^{-14}	5.54×10^{-30}	100	$9.35 \times 10^{+00}$	8.35×10^{-02}	0
	CS	2.37×10^{-14}	9.89×10^{-25}	100	4.37×10^{-01}	6.17×10^{-03}	0
f_{10}	DE	0	0	100	1.12×10^{-14}	1.26×10^{-29}	100
	CR1-DE	0	0	100	1.15×10^{-16}	4.83×10^{-34}	100
	CR2-DE	0	0	100	$\mathbf{1.11 \times 10^{-16}}$	**0**	100
	CR3-DE	0	0	100	1.67×10^{-16}	3.14×10^{-33}	100
	DELF	0	0	100	1.12×10^{-16}	0	100
	PSO	0	0	100	2.55×10^{-16}	5.54×10^{-33}	100
	IWO	0	0	100	2.88×10^{-13}	1.90×10^{-34}	100
	HS	0	0	100	5.43×10^{-15}	1.87×10^{-28}	100
	CS	0	0	100	5.32×10^{-13}	1.37×10^{-16}	100

crossover probability calculated by CR1, rather than small values calculated by CR2 and CR3, can increase diversity of the population and obtain better search results.

Besides the consideration of getting better results for modified crossover probability in FITDE, it should also be able to speed up the convergence speed. To give an intuitive presentation, we drew convergence curves of ten benchmark functions and made comparisons of DE and three strategies in terms of convergence speed. Figures 5 and 6 are the convergence curves when $D = 50$ and $D = 100$, respectively.

As can be seen from Figure 5, for f_4, the result of CR1-DE is worse than the other three methods, but the convergence speed of CR1-DE is faster; for f_8, within the range of the given iterations, none of the four methods can get a satisfying result, but the result of CR1-DE is still better than the other three methods. Except f_4 and f_8, for the other eight functions, the convergence speed of CR1-DE is the fastest among the four methods. As can be seen from Figure 6, for f_3, f_4, and f_8, within the range of the given iterations, none of the four methods can get a pleasurable result. Except those

Table 4: States of three changing strategies according to their testing performances.

Method	$D = 50$										$D = 100$									
	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8	f_9	f_{10}	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8	f_9	f_{10}
CR1-DE	1∗	1∗	1∗	0	1	1	1∗	1∗	1	1	1∗	1∗	1∗	0	0	1∗	1∗	1∗	0	1
CR2-DE	0	0	1	0	1∗	1	1	0	1∗	1	1	1	1	0	1∗	1	1	1	0	1∗
CR3-DE	0	0	1	1∗	1	1	1	0	1	1	1	1	1	0∗	1	1	1	0	1∗	1

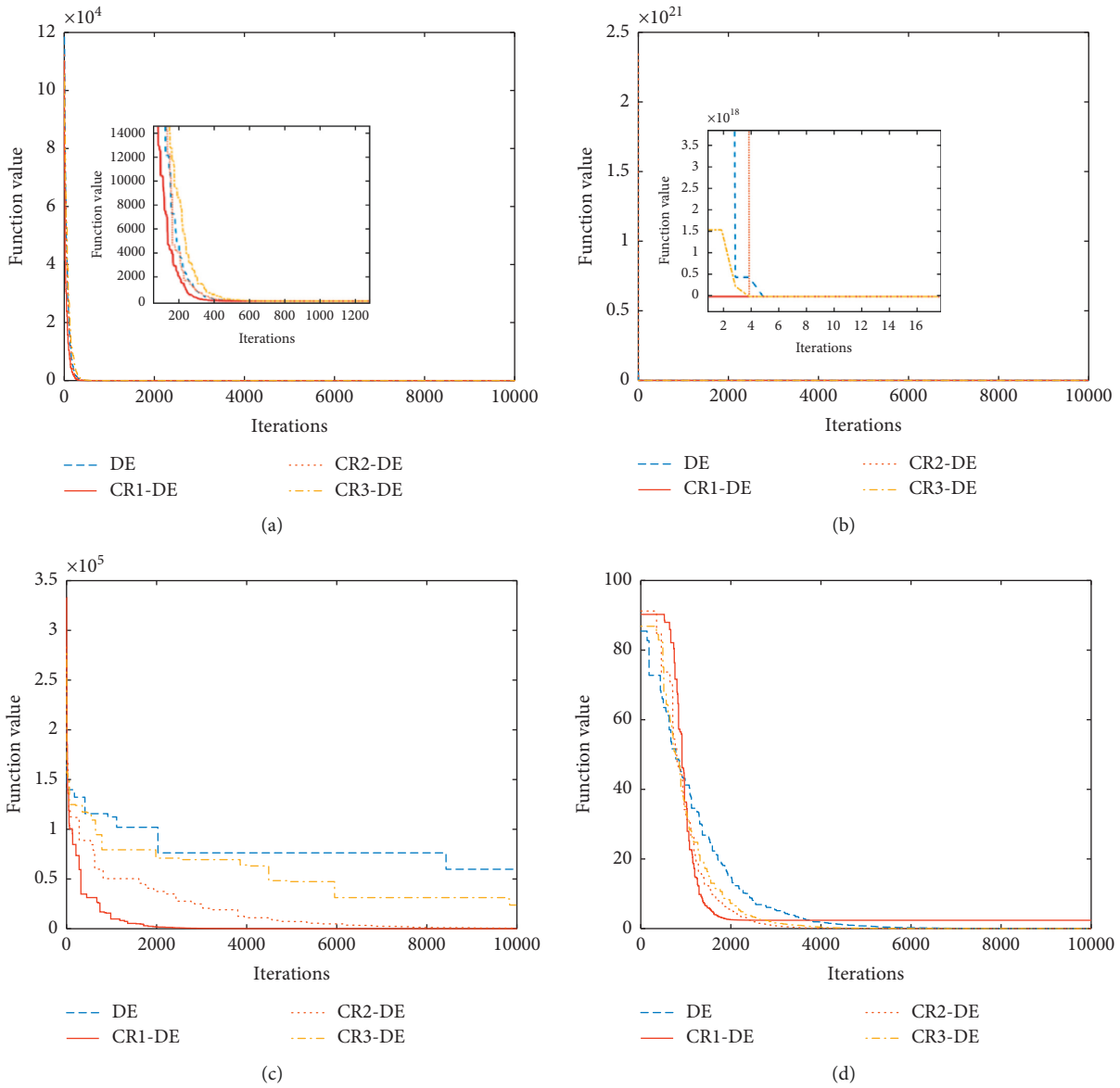

Figure 5: Continued.

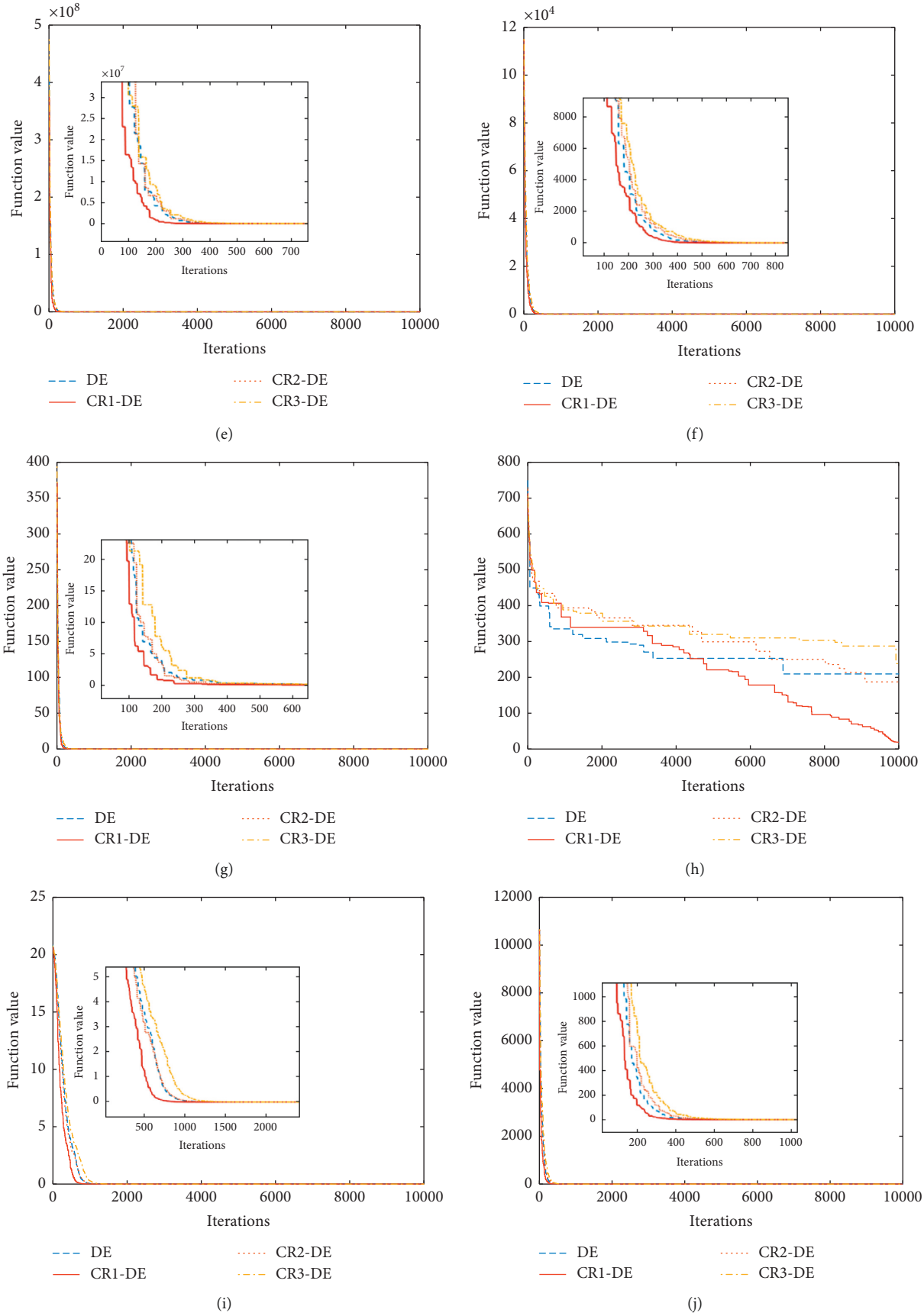

Figure 5: Convergence curves of f_1–f_{10} when $D = 50$ (if there are two figures in one subgraph, the smaller one represents partial enlarged drawing of the bigger one for clear presentation). (a) f_1, (b) f_2, (c) f_3, (d) f_4, (e) f_5, (f) f_6, (g) f_7, (h) f_8, (i) f_9, and (j) f_{10}.

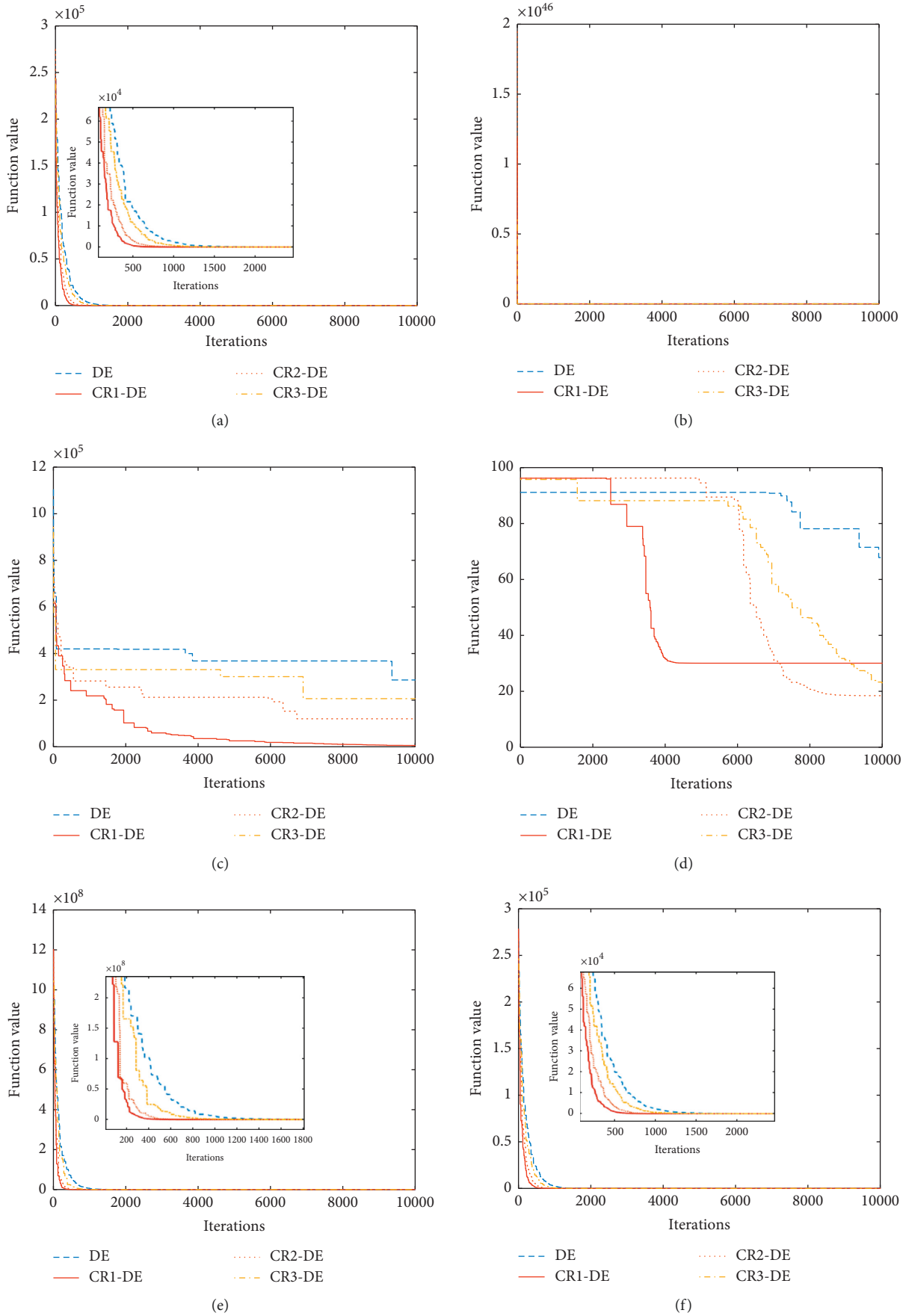

Figure 6: Continued.

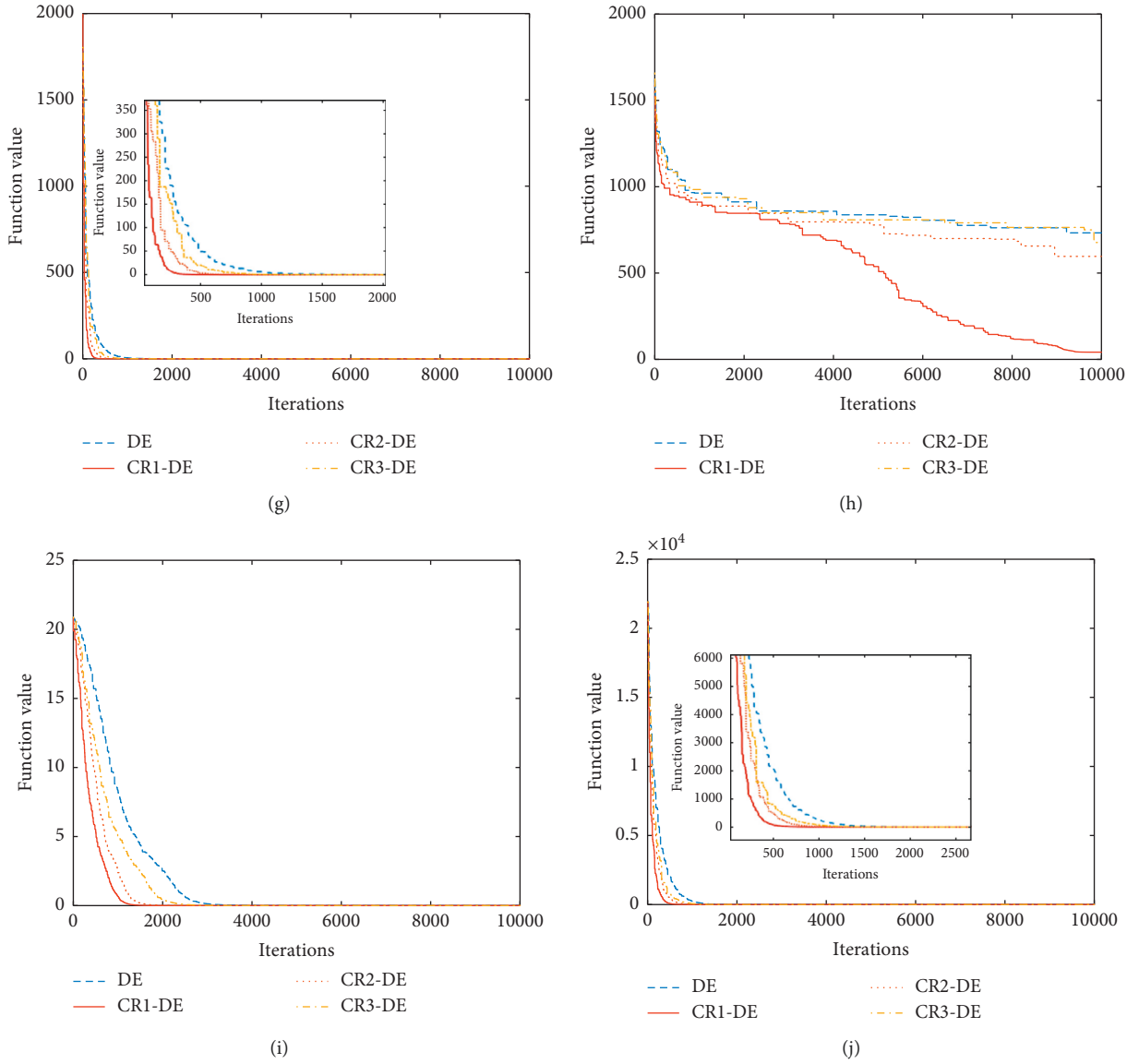

Figure 6: Convergence curves of f_1–f_{10} when $D = 100$ (if there are two figures in one subgraph, the smaller one represents partial enlarged drawing of the bigger one for clear presentation). (a) f_1, (b) f_2, (c) f_3, (d) f_4, (e) f_5, (f) f_6, (g) f_7, (h) f_8, (i) f_9, and (j) f_{10}.

three functions, for the other seven functions, the convergence speed of CR1-DE is the fastest among the four methods, and DE has the slowest convergence speed.

The analyses of the above test data and results demonstrate that FITDE can improve the performance of the traditional DE algorithm, not only in optimization results but also in convergence speed. In addition, the conclusion that the first changing strategy of crossover probability is better than the other two changing strategies can also be drawn.

4. Synthesis of Sparse Concentric Ring Arrays

In this section, we optimize three different kinds of sparse concentric arrays and compare simulation results with HGA in [1], DRGA in [11], and MGA in [12]. Example 1 only optimizes ring spacing, Example 2 only optimizes elements spacing on each ring, and Example 3 optimizes ring spacing and elements spacing at the same time. Reference [1] has these three examples, [11] deals with a sparse CRA with 9 rings, and [12] does not make individual optimizations of ring spacing or elements spacing, so we compare FITDE with HGA in Example 1, with HGA and DRGA in Example 2, and with HGA and MGA in Example 3.

Parameters in examples are set as follows: population size NP = 50, iteration number $G = 200$, scaling factor $F = 0.5$, $\mathrm{CR}_{\min} = 0.3$, and $\mathrm{CR}_{\max} = 0.95$. Each example runs 50 times independently. All simulations are conducted using MATLAB R2018b on a computer with CPU of 4.0 GHz and RAM of 8.0 GB.

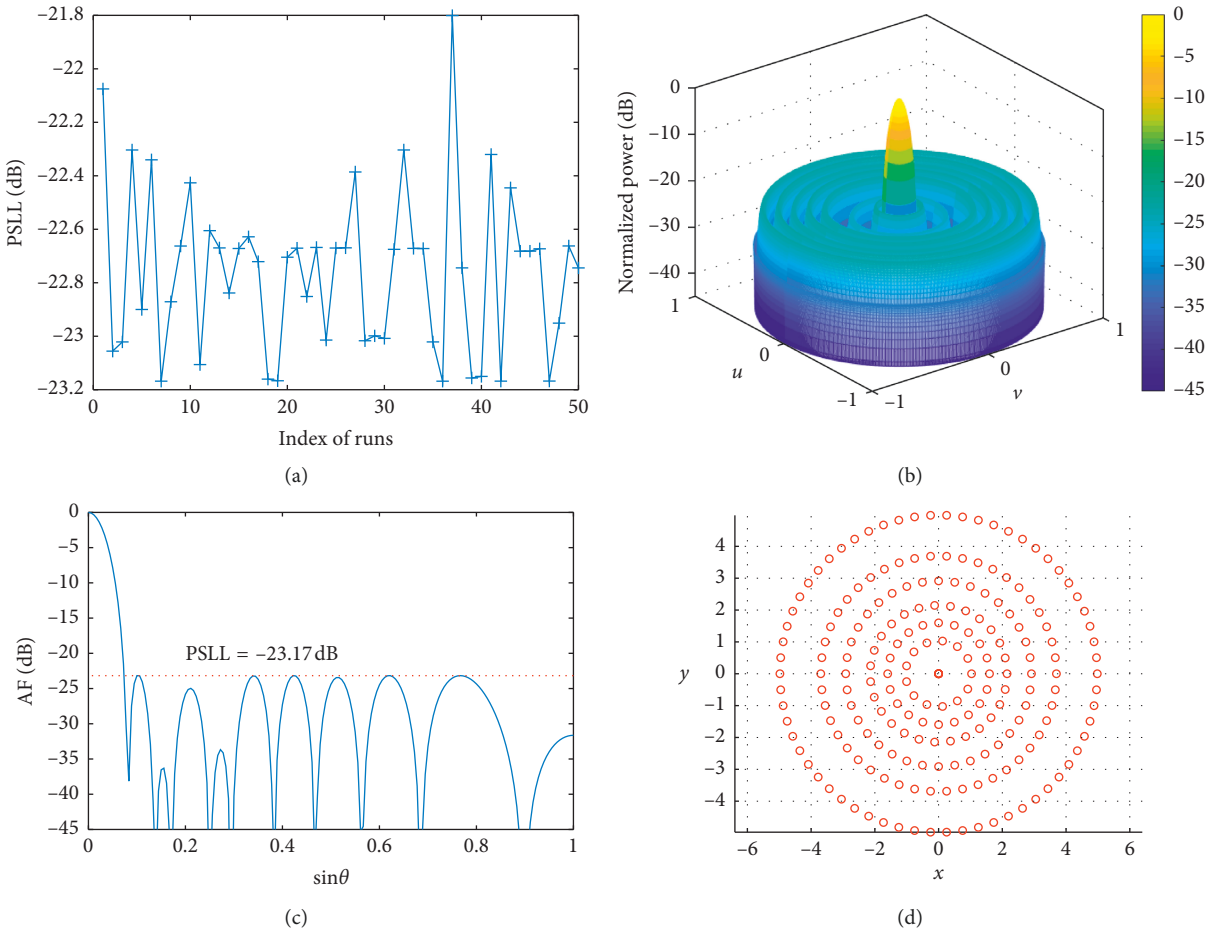

Figure 7: Result of Example 1. (a) PSLL of 50 independent runs. The best PSLL is −23.17 dB, the average PSLL is −22.76 dB, and the variance is 0.095. (b) is the best radiation pattern of all planes, characteristic of excellent symmetry. (c) is the sectional pattern of one single plane. (d) is the optimal distribution of elements.

Table 5: Results of Example 1

$L(\lambda)$	Method	PSLL (dB)	N_e	Ring	1	2	3	4	5	6
4.98	FITDE	−23.17	205	$r_m(\lambda)$	1.03	1.60	2.15	2.91	3.70	4.98
				N_m	13	20	27	36	46	62
	HGA	−22.94	201	$r_m(\lambda)$	1.00	1.59	2.14	2.88	3.66	4.98
				N_m	12	19	26	36	45	62

Example 1. In this example, we deal with a CRA with $L = 4.98\lambda$ and $M = 6$, and only the ring spacing is regarded as the optimization variable. The elements spacing on each ring is the minimum spacing $d_c = 0.5\lambda$, and elements are equally distributed on the ring. Therefore, when the radius of each ring is determined, the number of elements on the ring N_m is also determined:

$$N_m = \left[\frac{2\pi r_m}{d_c}\right], \tag{13}$$

where $[\cdot$ is the maximum integer not greater than $\cdot$.

Results of 50 independent runs are shown in Figure 7(a). The best PSLL is −23.17 dB, which is 0.23 dB lower than HGA best PSLL. The average PSLL of 50 runs is −22.76 dB, and the variance is 0.095. The corresponding best ring radii, the number of elements on each ring, and the comparison with HGA method are concluded in Table 5. Radiation pattern, sectional pattern, and distribution of elements are displayed in Figures 7(b)–7(d).

Example 2. We consider a CRA with $L = 4.5\lambda$ and $M = 9$ in this example and optimize only the elements spacing. The spacing between two adjacent rings is $d_c = 0.5\lambda$. Elements spacing on each ring is restricted to $[0.5\lambda, \lambda]$. The lower limit 0.5λ is to satisfy the constraint condition of the minimum elements spacing, and the upper limit λ is to avoid the

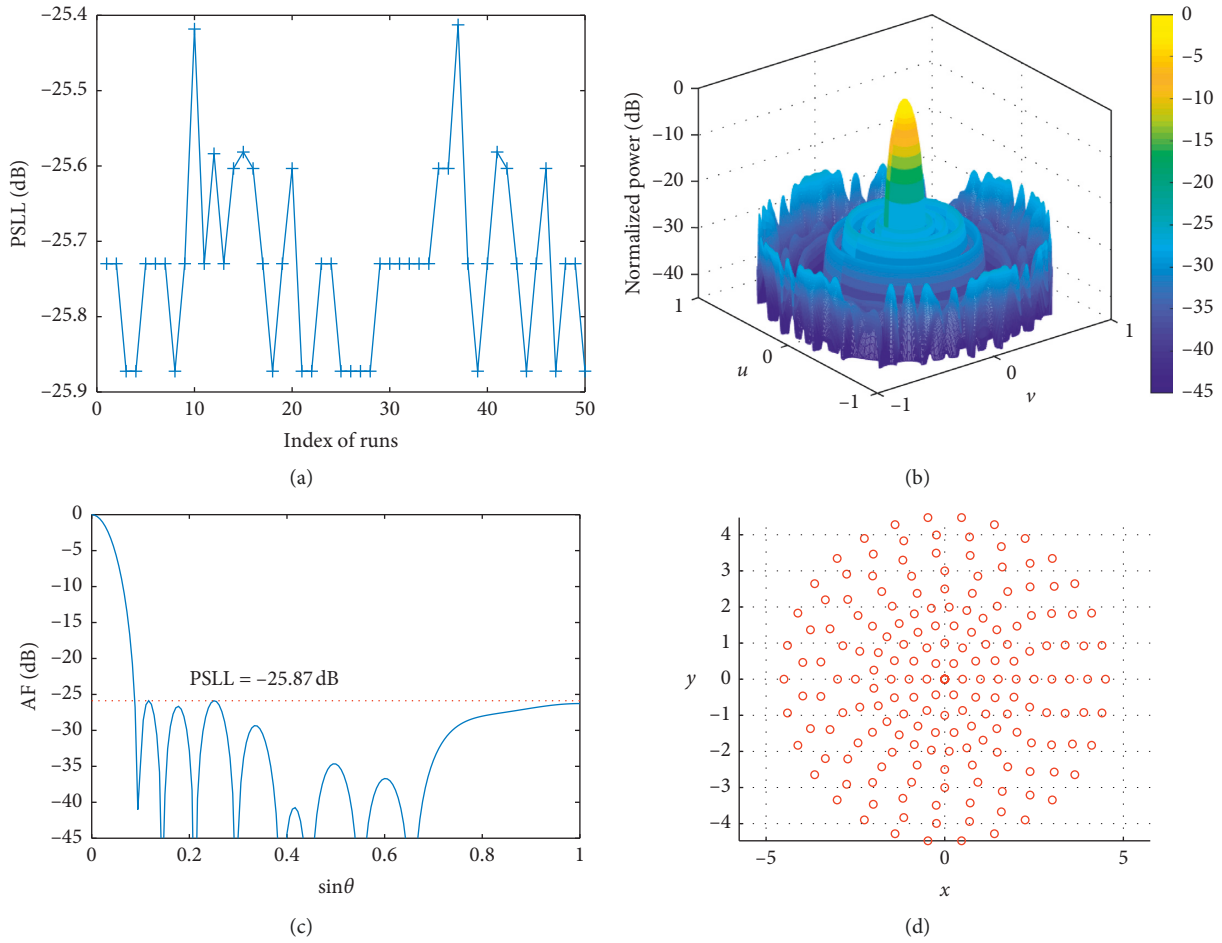

Figure 8: Result of Example 2. (a) PSLL of 50 independent runs in Example 2. The best PSLL is −25.87 dB, the average PSLL is −25.73 dB, and the variance is 0.014. (b) is the best radiation pattern of all planes, characteristic of excellent symmetry. (c) is the sectional pattern of one single plane. (d) is the optimal distribution of elements.

Table 6: Results of Example 2.

$L(\lambda)$	Method	PSLL (dB)	N_e	$r_m(\lambda)$								
				0.5	1	1.5	2	2.5	3	3.5	4	4.5
4.5	FITDE	−25.87	182	6	12	18	25	20	20	23	27	30
	HGA	−25.58	183	6	12	18	25	17	23	22	27	32
	DRGA	−23.74	185	6	12	18	25	19	22	26	26	30

occurrence of the gating lobe due to excessive spacing between elements.

Figure 8(a) exhibits the results of 50 independent runs. The best PSLL is −25.87 dB, which is 0.29 dB lower than HGA best PSLL and 2.13 dB lower than DRGA best PSLL. Apart from better PSLL, the sparse CRA optimized by FITDE in this example saves one and three elements, respectively, compared with HGA and DRGA. The average PSLL of 50 runs is −25.73 dB, and the variance is 0.014. Table 6 gives the corresponding best elements spacing on each ring, the number of elements on each ring and the comparisons with HGA and DRGA. Figures 8(b)–8(d) plot the radiation pattern, sectional pattern, and distribution of elements.

Example 3. A CRA characterized by $L = 4.7\lambda$ and $M = 6$ is involved in Example 3, and the ring spacing and the elements spacing are both considered as optimization variables. Figure 9(a) is the curve of PSLL of 50 independent runs. The best PSLL is −28.58 dB, which is 0.76 dB lower than HGA best PSLL, and 0.25 dB lower than MGA best PSLL. Besides lower PSLL, the elements number of the sparse CRA optimized by FITDE in this example is one less than that of HGA and MGA. The average PSLL of 50 runs is −28.17 dB, and the variance is 0.084.

Among the 50 independent runs, the radius of each ring corresponding to the optimal result, the number of elements on each ring, and the comparisons with HGA and MGA are listed in Table 7.

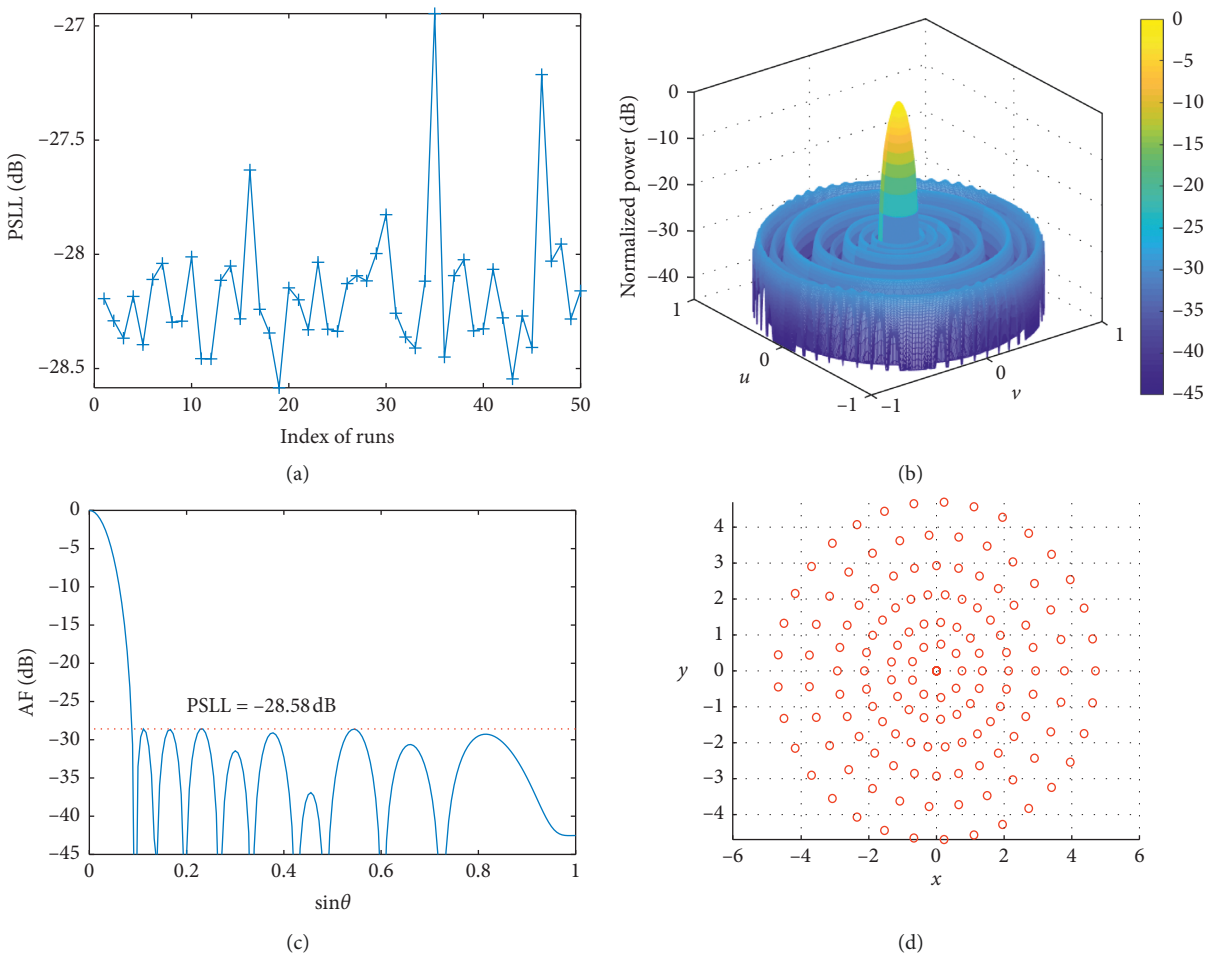

Figure 9: Result of Example 3. (a) PSLL of 50 independent runs in Example 3. The best PSLL is −28.58 dB, the average PSLL is −28.17 dB, and the variance is 0.084. (b) is the best radiation pattern of all planes, characteristic of excellent symmetry. (c) is the sectional pattern of one single plane. (d) is the optimal distribution of elements.

Table 7: Results of Example 3.

$L(\lambda)$	Method	PSLL (dB)	N_e	Ring	1	2	3	4	5	6
4.7	FITDE	−28.58	141	$r_m(\lambda)$	0.75	1.35	2.13	2.93	3.78	4.70
				N_m	9	17	26	28	27	33
	HGA	−27.82	142	$r_m(\lambda)$	0.76	1.36	2.09	2.99	3.78	4.70
				N_m	9	17	25	31	26	33
	MGA	−28.33	142	$r_m(\lambda)$	0.74	1.32	2.10	2.93	3.79	4.70
				N_m	9	16	26	30	27	33

The best radiation pattern, sectional pattern, and distribution of elements are portrayed in Figure 9(b)–9(d).

From the above results, FITDE can get lower PSLL. However, the advantages of FITDE consist not only in reducing PSLL but also in reducing the complexity of the optimization method and saving running time. In fact, crossover and mutation operations in reference algorithms became complex after modification, which led to low calculation efficiency and long running time, while for FITDE, the algorithm structure is simple with high calculation efficiency and short running time. Therefore, we also make the comparison of running time between FITDE and reference algorithms, as is concluded in Table 8. As can be seen from the table, the running time advantage of FITDE is quite prominent.

Table 8: Comparisons of running time (h: hour; min: minute).

Example	1	2	3
FITDE	51 min	52 min	43 min
HGA	3 h 7 min	2 h 29 min	57 min
DRGA	—	3 h 30 min	—
MGA	—	—	70 min

5. Conclusions

Aiming at solving the problem that crossover probability is unchanged in traditional DE algorithm, we propose fitness-associated DE (FITDE) algorithm and introduce population fitness variance to the DE algorithm, which can adjust crossover probability dynamically. Three crossover probability changing strategies are constructed and tested by benchmark functions. Test results manifest that FITDE can improve the performance of the traditional DE algorithm, and the best one of the three crossover probability changing strategies is determined. Sparse concentric ring arrays are optimized using FITDE, and three groups of optimization simulation examples show that FITDE can enrich the population diversity, accelerate the convergence speed, and reduce the peak sidelobe level of concentric ring array effectively.

Data Availability

No data were used to support this study.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work was supported by the National Natural Science Foundation of China under Grants 61179004 and 61179005.

References

[1] R. L. Haupt, "Optimized element spacing for low sidelobe concentric ring arrays," *IEEE Transactions on Antennas and Propagation*, vol. 56, no. 1, pp. 266–268, 2008.

[2] Y. Jiang, S. Zhang, Q. Guo, and M. Li, "Synthesis of uniformly excited concentric ring arrays using the improved integer GA," *IEEE Antennas and Wireless Propagation Letters*, vol. 15, pp. 1124–1127, 2016.

[3] X. W. Zhao, Q. S. Yang, and Y. H. Zhang, "A hybrid method for the optimal synthesis of 3-D patterns of sparse concentric ring arrays," *IEEE Transactions on Antennas and Propagation*, vol. 64, no. 2, pp. 515–524, Feb. 2016.

[4] Y. Jiang, S. Zhang, Q. Guo, and X. Luan, "A hybrid strategy based on weighting density and genetic algorithm for the synthesis of uniformly weighted concentric ring arrays," *IEEE Antennas and Wireless Propagation Letters*, vol. 16, pp. 186–189, 2017.

[5] Z. Medina, A. Reyna, M. A. Panduro, and O. Elizarraras, "Dual-band performance evaluation of time-modulated circular geometry array with microstrip-fed slot antennas," *IEEE Access*, vol. 7, pp. 28625–28634, 2019.

[6] R. L. Haupt, "Thinned arrays using genetic algorithms," *IEEE Transactions on Antennas and Propagation*, vol. 42, no. 7, pp. 993–999, 1994.

[7] K. S. Chen, Z. S. He, and C. L. Han, "A modified real GA for the sparse linear array synthesis with multiple constraints," *IEEE Transactions on Antennas and Propagation*, vol. 54, no. 7, pp. 2169–2173, 2006.

[8] C. Ling, Z. L. Yu, W. Ser, and W. Cen, "Linear aperiodic array synthesis using an improved genetic algorithm," *IEEE Transactions on Antennas and Propagation*, vol. 60, no. 2, pp. 895–902, 2012.

[9] B. Yu, K.-S. Chen, P. Zhu et al., "An optimum method of sparse concentric rings array based on dimensionality reduction," *Journal of Electronics and Information Technology*, vol. 36, no. 2, pp. 476–481, 2014.

[10] K.-S. Chen, Y.-Y. Zhu, X.-L. Ni, and H. Chen, "Low sidelobe sparse concentric ring arrays optimization using modified GA," *International Journal of Antennas and Propagation*, vol. 2015, Article ID 147247, 5 pages, 2015.

[11] K. S. Chen, H. Chen, L. Wang, and H. Wu, "Modified real GA for the synthesis of sparse planar circular arrays," *IEEE Antennas and Wireless Propagation Letters*, vol. 15, pp. 274–277, 2016.

[12] N. Pathak, G. K. Mahanti, S. K. Singh, J. K. Mishra, and A. Chakraborty, "Synthesis of thinned planar circular array antennas using modified particle swarm optimization," *Progress in Electromagnetics Research Letters*, vol. 12, pp. 87–97, 2009.

[13] Z. D. Zaharis, S. K. Goudos, and T. V. Yioultsis, "Application of Boolean PSO with adaptive velocity mutation to the design of optimal linear antenna arrays excited by uniform amplitude current distribution," *Journal of Electromagnetic Waves and Applications*, vol. 25, no. 10, pp. 1422–1436, 2011.

[14] R. Bhattacharya, T. K. Bhattacharyya, and R. Garg, "Position mutated hierarchical particle swarm optimization and its application in synthesis of unequally spaced antenna arrays," *IEEE Transactions on Antennas and Propagation*, vol. 60, no. 7, pp. 3174–3181, 2012.

[15] P. Ghosh and S. Das, "Synthesis of thinned planar concentric circular antenna arrays-A differential evolutionary approach," *Progress in Electromagnetics Research B*, vol. 29, pp. 63–82, 2011.

[16] X. Li and M. Yin, "Design of a reconfigurable antenna array with discrete phase shifters using differential evolution algorithm," *Progress in Electromagnetics Research B*, vol. 31, pp. 29–43, 2011.

[17] A. Mandal, H. Zafar, S. Das, and A. V. Vasilakos, "Efficient circular array synthesis with a memetic differential evolution algorithm," *Progress in Electromagnetics Research B*, vol. 38, pp. 367–385, 2012.

[18] G. Kerim and B. Suad, "Null synthesis of time-modulated circular antenna arrays using an improved differential evolution algorithm," *IEEE Antennas and Wireless Propagation Letters*, vol. 12, pp. 817–820, 2013.

[19] X. Li, W.-T. Li, X.-W. Shi, J. Yang, and J.-F. Yu, "Modified differential evolution algorithm for pattern synthesis of antenna arrays," *Progress in Electromagnetics Research*, vol. 137, pp. 371–388, 2013.

[20] D. Dai, M. Yao, H. Ma, W. Jin, and F. Zhang, "An effective approach for the synthesis of uniformly excited large linear sparse array," *IEEE Antennas and Wireless Propagation Letters*, vol. 17, no. 3, pp. 377–380, 2018.

[21] Y.-Y. Bai, S. Xiao, C. Liu, and B.-Z. Wang, "A hybrid IWO-PSO algorithm for pattern synthesis of conformal phased array," *IEEE Transactions on Antennas and Propagation*, vol. 61, no. 4, pp. 2328–2332, 2013.

[22] X. Li, S. Ma, and G. Yang, "Synthesis of difference patterns for monopulse antennas by an improved cuckoo search algorithm," *IEEE Antennas and Wireless Propagation Letters*, vol. 16, pp. 141–144, 2016.

[23] S. Liang, T. Feng, and G. Sun, "Sidelobe level suppression for linear and circular antenna arrays via the cuckoo search–

chicken swarm optimization algorithm," *IET Microwaves, Antennas & Propagation*, vol. 11, no. 2, pp. 209–218, 2017.

[24] G. Sun, Y. H. Liu, Z. Y. Chen, S. Liang, A. Wang, and Y. Zhang, "Radiation beam pattern synthesis of concentric circular antenna arrays using hybrid approach based on cuckoo search," *IEEE Transactions on Antennas and Propagation*, vol. 66, no. 9, pp. 4563–4576, 2018.

[25] G. Oliveri and A. Massa, "Bayesian compressive sampling for pattern synthesis with maximally sparse non-uniform linear arrays," *IEEE Transactions on Antennas and Propagation*, vol. 59, no. 2, pp. 467–481, 2011.

[26] G. Oliveri, M. Carlin, and A. Massa, "Complex-weight sparse linear array synthesis by Bayesian compressive sampling," *IEEE Transactions on Antennas and Propagation*, vol. 60, no. 5, pp. 2309–2326, 2012.

[27] F. Viani, G. Oliveri, and A. Massa, "Compressive sensing pattern matching techniques for synthesizing planar sparse arrays," *EEE Transactions on Antennas and Propagation*, vol. 61, no. 9, pp. 4577–4587, 2013.

[28] M. Carlin, G. Oliveri, and A. Massa, "Hybrid BCS-Deterministic approach for sparse concentric ring isophoric arrays," *IEEE Transactions on Antennas and Propagation*, vol. 63, no. 1, pp. 378–383, 2015.

[29] B. Zhang, W. Liu, and X. Gou, "Compressive sensing based sparse antenna array design for directional modulation," *IET Microwaves, Antennas & Propagation*, vol. 11, no. 5, pp. 634–641, 2017.

[30] C. Yan, P. Yang, Z. Xing, and S. Y. Huang, "Synthesis of planar sparse arrays with minimum spacing constraint," *IEEE Antennas and Wireless Propagation Letters*, vol. 17, no. 6, pp. 1095–1098, 2018.

[31] Z. X. Huang and Y. J. Cheng, "Near-field pattern synthesis for sparse focusing antenna arrays based on Bayesian compressive sensing and convex optimization," *IEEE Transactions on Antennas and Propagation*, vol. 66, no. 10, pp. 5249–5257, 2018.

[32] C. Zhu, X. Li, Y. Liu, L. Liu, and Q. H. Liu, "An extended generalized matrix pencil method to synthesize multiple-pattern frequency-invariant linear arrays," *IEEE Antennas and Wireless Propagation Letters*, vol. 16, pp. 2311–2315, 2017.

[33] H. Shen, B. Wang, and X. Li, "Shaped-beam pattern synthesis of sparse linear arrays using the unitary matrix pencil method," *IEEE Antennas and Wireless Propagation Letters*, vol. 16, pp. 1098–1101, 2017.

[34] P. Gu, G. Wang, Z. Fan, and R. Chen, "Efficient unitary matrix pencil method for synthesizing wideband frequency patterns of sparse linear arrays," *IET Microwaves, Antennas & Propagation*, vol. 12, no. 12, pp. 1871–1876, 2018.

[35] X. T. Li, J. N. Wang, and M. H. Yin, "Enhancing the performance of cuckoo search algorithm using orthogonal learning method," *Neural Computing and Applications*, vol. 24, no. 6, pp. 1233–1247, 2014.