Zhang and Zhang *EURASIP Journal on Wireless Communications and Networking* (2017) 2017:7
DOI 10.1186/s13638-016-0802-2

EURASIP Journal on Wireless Communications and Networking

RESEARCH Open Access

A non-revisiting artificial bee colony algorithm for phased array synthesis

Xin Zhang[1,2] and Xiu Zhang[1,2*]

Abstract

An effective non-revisiting artificial bee colony (NrABC) algorithm based on the paradigm of artificial bee colony (ABC) is developed in this paper. NrABC is applied to tackle the synthesis of phased linear arrays. Pros and cons of NrABC is provided along with a comparison to standard ABC. Binary space partitioning tree structure is used to record history evolutionary information. Non-revisiting scheme assures NrABC keeping good diversity of population. Moreover, scout bee stage is discarded in NrABC which also removes an algorithmic parameter of standard ABC. Three phased array synthesis examples are employed to study the performance of NrABC. It turns out that under the same experimental configuration, NrABC outperforms standard ABC in terms of both solution quality and reliability in repeated runs.

Keywords: Antenna pattern synthesis, Phased arrays, Artificial bee colony, Non-revisiting, Numerical optimization

1 Introduction

In remote sensing systems of radar networks [1, 2], phased arrays play an important role to ensure the quality of services (QoS) under certain communication requirements [3–5]. The synthesis of phased arrays is known to be a nonlinear programming problem [6], which is a multimodal and high dimensional problem. Recently, many researchers pay attention to create powerful algorithms for dealing with the synthesis of phased arrays. These kinds of algorithms are mainly heuristic-based methods since traditional optimization methods or analytical methods are not suitable. It turns out that evolutionary computing methods present good performance in handling phased array design problems [7, 8].

Phased arrays are basic components in communication networks including radar networks, commercial communication networks, military networks, and mobile communication networks. They are capable to enlarge the capacity of network systems. Usually, the synthesis of phased array means to obtain certain radiation pattern sufficing to special conditions. Interfering signals are suppressed during synthesis process by setting nulls on array pattern along interfering signals. At the same time, the direction of the main beam is kept facing desired signal. The degradation of sidelobe level of array pattern becomes a severe question if sidelobe region was set null. This problem would be worse if the number of setting nulls increased and close to the main beam [9, 10]. Thus, the synthesis of phased arrays is highly difficult, and effective and reliable algorithms which can find promising solutions are urgently needed.

Evolutionary computing methods are comprised of evolutionary algorithms (EAs) and swarm intelligence (SI) algorithms. Generally speaking, evolutionary algorithms include genetic algorithm [11], evolutionary strategies [12], evolutionary programming, and differential evolution [13]. EAs are featured by simulating the evolution of genetic process of livings. Mutation and crossover operations are necessary operators in these algorithms. The typical paradigms of swarm intelligence algorithms are particle swarm optimization [14], artificial bee colony [15], neighborhood search optimization [16], and brain storm optimization [17]. SI algorithms are featured by emulating the social or foraging behaviors of swarms such as flocks, ants, fish schools, etc. They usually do not involve crossover operation. Although EAs and SI are inspired from different nature background, their positive combinations are able to result effective algorithms, which are recently classified to memetic computing [18].

Amongst these methods, genetic algorithm and particle swarm optimization have been widely used in antenna

*Correspondence: zhang210@126.com
[1]Tianjin Key Laboratory of Wireless Mobile Communications and Power Transmission, Tianjin Normal University, Tianjin, China
[2]College of Electronic and Communication Engineering, Tianjin Normal University, Tianjin, China

designs [6, 19, 20]. Recently, covariance matrix adaptation evolutionary strategy [21] and differential evolution [22] are applied to synthesize antenna array patterns. Although good performance is attained by these algorithms, synthesis of antenna arrays is still a challenge due to the rapid development of communication networks. Moreover, it is also meaningful to create more powerful optimization algorithms so that problems could be solved in shorter time than existing methods.

Artificial bee colony (ABC) is a SI paradigm emulating foraging behavior of honey bees [15]. This algorithm is comprised of employed bees, onlooker bees, and scout bees, i.e., three bee groups. There exist other bees in nature, though only these three are simulated in this algorithm. Many studies show that this algorithm is a powerful and efficient one for handling multimodal problems [23, 24]. Although its variants have been applied to deal with antenna designs [25, 26], they often cost a large number of function evaluations to attain desirable solutions. Thus, it is necessary to further enhance the powerfulness of this algorithm.

It is observed that a great number of function evaluations is needed for artificial bee colony algorithm to obtain a satisfactory solution. Accordingly, a great number of cycles or generations is evolved during optimization process. It is well known that population diversity in EAs and SI algorithms heavily reduces once algorithms are evolved in a large number of generations. This means solutions tend to be homogeneous or most genetic information becomes identical. Thus, such algorithms could not produce diverse variations and then lose the capability of refining fitness of solutions. In ABC, scout bees is responsible to diversify solutions for having a wider search area. However, recent study shows that this method is not effective to speed up convergence rate of ABC [27]. In this paper, a non-revisiting scheme is used to keep population diversity substituting for scout bee stage in ABC. The idea of non-revisiting scheme is to restrict an algorithm from revisiting an already searched place. For one thing, it avoids revisiting and repeated solution evaluation. For another thing, it memorizes all visited places by algorithm so as to let the search of algorithm focus on unknown places with high uncertainty. Besides non-revisiting scheme, three bee groups in ABC are also changed. As non-revisiting scheme is apt to guide the search directions of an algorithm, employed and scout bees are eliminated from standard ABC algorithm. Hence, only onlooker bee stage is kept in the resulting algorithm. The new algorithm is named as non-revisiting artificial bee colony (NrABC). NrABC is then applied to tackle phased array design problems. Numerical experiment is conducted studying NrABC and standard ABC. The results are discussed and analyzed at the end of the paper.

The paper is organized as follows. Section 2 reviews the synthesis of phased array problem and related works. Section 3 gives standard ABC and the proposed NrABC algorithm. Section 4 reports numerical simulation results and discussions. Section 5 concludes the paper.

2 Problem overview and related works

As above mentioned, phased array synthesis is the application to be faced in this paper. The target of phased array is to find the best weights satisfying predefined far-field sidelobe requirement. One side of far-field sidelobe includes a 60-dB notch [19]. In case the approximate direction to an interference source is determined, it is desired that a notched far-field pattern would be attained. A linear phased antenna array is considered which consists of one hundred half-wavelength spaced radiators. Denote far-field radiation pattern as $FF(\theta)$ at radial angle θ. $FF(\theta)$ is expressed by the following equation:

$$FF(\theta) = EP(\theta) \cdot AF(\theta)\,, \tag{1}$$

where $EP(\theta)$ is the voltage element pattern and $AF(\theta)$ is the array factor. Their equation is given as follows:

$$EP(\theta) = \sqrt{\cos^{1.2}(\theta)}\,, \tag{2}$$

$$AF(\theta) = \sum_{n=1}^{N} A_n e^{j2\pi n(d/\lambda)\sin(\theta)}\,, \tag{3}$$

where $AF(\theta)$ stands for array factor and $EP(\theta)$ is voltage element pattern. The number of radiators is denoted as N and is set to 100. Decision variables are complex element weights A_n $(n = 1, 2, \ldots, N)$. Hence, problem dimension is $D = N$. The space between elements normalized by wavelength is $d/\lambda = 1/2$. Clearly, decision variables are continuous parameters to be determined by an optimization algorithm.

Objective function of this problem can be defined to minimize the sum of squares of the difference between excess far-field magnitude and specified sidelobe envelope. This objective function penalizes sidelobes above envelope, and noting is performed if sidelobes is below the given specification [19]. The following equation shows the objective function:

$$f(A_n) = \sum |\text{magnitude} - \text{threshold}|^2\,. \tag{4}$$

From this equation, a pattern point provides a penalty value to $f(A_n)$ if it locates above predefined threshold. Similarly, objective function value can be computed given both an upper threshold and a lower threshold. It is observed that the objective function is a nonlinear one and could not be tackled by analytical algorithms. This model is interesting and useful in practice. This is because users do not concern how to arrange sidelobes. Instead, it is

required by users that the amount of sidelobes should be below some level.

To solve the problem, an algorithm can directly optimize weights of radiating elements. To evaluate a candidate solution, a far-field pattern of 512 points is computed which are evenly spaced in sine space. Sine space means the sine of far-field angle. Because such equally spaced pattern points usually produces more uniform spreading points than using angle directly. In this problem model, the lower the cost, the fitter the antenna array is distributed.

Boeringer and Werner were two of the pioneers applying evolutionary computing approaches to design phased arrays [19]. Bataineh and Ababneh used particle swarm optimization (PSO) to synthesize aperiodic linear antenna array with good results obtained as in the literature [28]. Li et al. proposed an improved PSO for electromagnetic applications with better results obtained compared with PSO and genetic algorithm (GA) [6]. Abu-Al-Nadi et al. used array polynomial method and PSO to design linear phased arrays [9]. Simulation results show that this method could reduce more than half parameters of synthesis problems compared with conventional methods. Mahmoud et al. combined a modified PSO and method of moment to realize beamforming of designing a smart antenna array [29]. Simulated on a planar uniform circular array with 30 elements of half wave dipoles, the algorithm outperformed standard PSO and several improved PSO algorithms [29].

Li et al. propose a modified DE which consists of a best random mutation and a randomized local search [22]. The proposed algorithm is then applied to deal with a linear array and a cylindrical conformal array designs. Results show that it is better than other benchmarked algorithms in pattern synthesis. Rocca et al. use GA to design wideband phased arrays [20]. The synthesis of phased arrays is achieved by using polyomino-shaped subarrays and by optimizing tile orientations and positions. Their proposed GA method presents good performance in dealing with such problems. Dhaliwal and Pattnaik compare GA, PSO, and bacterial foraging optimization algorithms for Sierpinski gasket fractal antenna design [30]. Simulation results show that all three algorithms outperform conventional nonlinear programming methods. None of the three algorithms could completely beat the others based on the results. From the viewpoint of both minimum error and computational time, PSO is recognized as the best model for this kind of problems [30].

On the other hand, researchers continue working on nonlinear programming algorithms for tackling phased array designs. Yang et al. propose a fast space-time adaptive processing (STAP) method based on projection approximation subspace tracking with sparse constraint [31]. This method is able to exploit the low rank character of clutter covariance matrix. Experiment is conducted on two real airborne phased array radar data sets. Results show that the proposed method outperforms existing STAP methods without sparse constraint. Khan et al. propose a compressed sensing technique for antenna array diagnosis [32]. They propose to hybridize iterative reweighted least squares and separable surrogate functional algorithms. Simulation results show that the hybrid algorithm is more accurate to diagnose defective sensors compared with separate algorithms. Alotaibi et al. propose a switch phased array method to improve security of antennas at physical layer [33]. This may guide a new direction for phased array designs.

3 Optimization algorithms

This section introduces standard ABC and the proposed NrABC algorithm.

3.1 Standard artificial bee colony algorithm

The paradigm of standard ABC can be expressed in Fig. 1, where *Np* denotes population size. That is also the number of food sources (solutions). As in step 1, ABC starts by a set of randomly generated solutions in given search space.

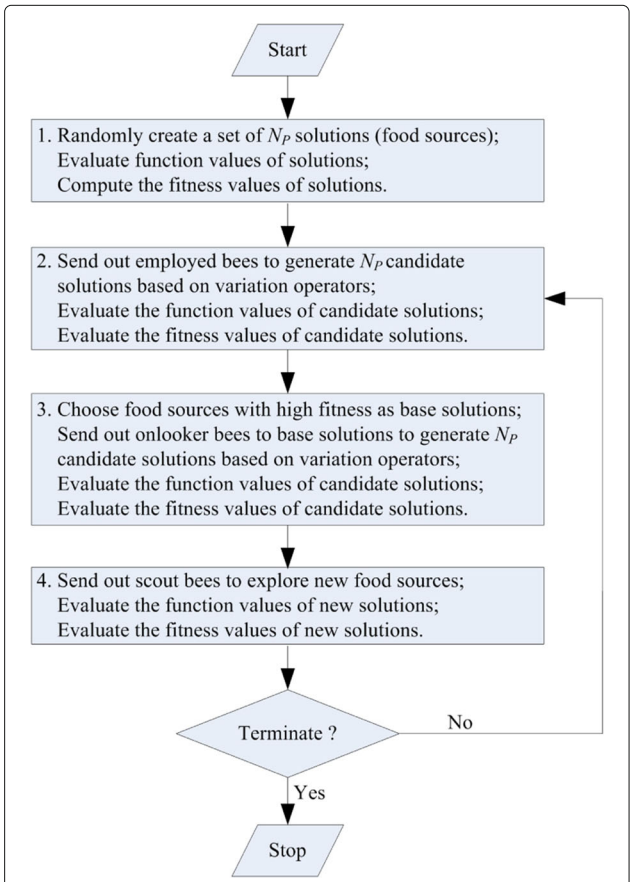

Fig. 1 Flow chart of standard artificial bee colony algorithm

Then the main cycle of standard ABC is executed. That is, step 2, step 3, and step 4. They correspond to employed bee stage, onlooker bee stage, and scout bee stage.

Standard ABC starts by a set of randomly generated solutions in given search space. Then the main cycle of standard ABC is executed. That is employed bee stage, onlooker bee stage, and scout bee stage. Variation operator is responsible for modifying current solutions which are usually called parent solutions in EAs. The variation in ABC is a one dimension perturbation as follows:

$$x_{t+1,j}=\begin{cases} x_{t,j}+\phi_{t,j1}(x_{t,j}-x_{r1,j}) & \text{if } j=j1 \\ x_{t,j} & \text{otherwise} \end{cases}, j=1,2,\cdots,D, \tag{5}$$

where $\phi_{t,j1}$ is a real number randomly generated between -1 and 1, and $j1$ is a random integer between 1 and D. $\mathbf{x}_{r1}$ is restricted to be a different solution from $\mathbf{x}_t$. In standard ABC, employed bee stage and onlooker bee stage use the same variation operator as in (5).

At each stage of a cycle, the number of variation trials is recorded. If a found solution is fitter than its parent, its associated trial count is reset to 0; otherwise, its associated trial count is add by 1. In this way, trial count for each solution is a non-negative integer. Such integer values indicate the evolutionary stage of each solution, and the algorithm. If trial count is above certain threshold, it is reasonable to assume that ABC stagnates or traps into local optimum if global optima have not been found. In this case, scout bees are sent out for exploring new food sources in replace of the one having trial count above threshold. In standard ABC, the search equation of scout bee is as follows:

$$x_{t+1,j}=x_j^{\min}+\phi_{t,j}(x_j^{\max}-x_j^{\min}),\ j=1,2,\cdots,D, \tag{6}$$

where $x_j^{\min}$ and $x_j^{\max}$ ($j=1,2,\cdots,D$) are the lower and upper boundary of decision variables, respectively.

Survivor selection step of standard ABC algorithm is a greedy selection technique. Candidate solution $\mathbf{x}_{t+1}$ competes with its parent $\mathbf{x}_t$. The winner after comparison survives and the other is discarded. For ABC, global best solution is also recorded at the end of each cycle so that it can be returned once the algorithm terminates. To avoid best so far solution is discarded in scout bee stage. Elitism can be implemented with size 1. That is best so far solution would not be replaced by randomly created solutions.

3.2 Non-revisiting artificial bee colony algorithm

Many researchers have aware that an evolutionary algorithm (EA) may revisit an already searched place, especially when the population of EA converges or most solutions in population becomes homogenous [34]. Yuen and Chow proposed to use binary space partitioning (BSP) tree to integrate with GA [35]. BSP tree stores all evaluated solutions by GA. They also modified mutation operator of GA and make it adaptively performed with BSP tree. As said in [34], non-revisiting scheme is superior to an algorithm without this scheme. Moreover, the modified algorithm makes sure good diversity of population through duplicate removal. Furthermore, non-revisiting scheme naturally results in adaptive mutation method without additional algorithmic parameter.

Note that non-revisiting scheme has such good characters; here, it is integrated with standard ABC, called NrABC. First, let depict BSP tree structure. There are many ways to store history search information of ABC. BSP tree is taken as it has been demonstrated to be powerful in several paradigms of EAs and SI approaches. The structure is illustrated in Fig. 2. BSP tree structure splits search space into many small regions based on evaluated solutions. As shown in Fig. 2, each region corresponds to an existing solution. Each tree node contains various search information of its associated solution such as time, fitness, function value, region size, and so on.

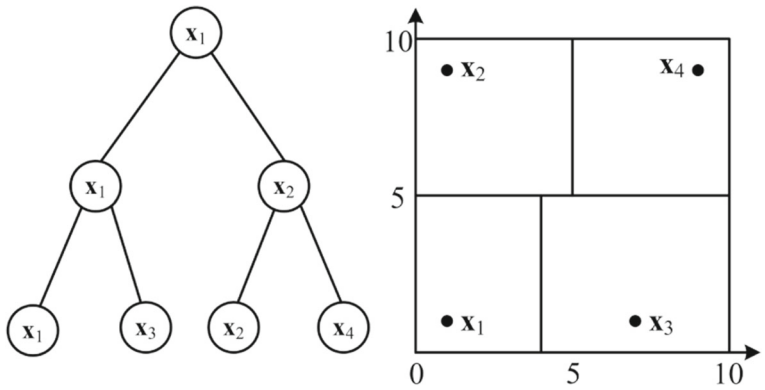

Fig. 2 An example of binary space partitioning tree

Clearly, an overhead of non-revisiting scheme is the cost of memory. On the one hand, it cost a lot of computer memory than an algorithm without such scheme, whereas computational complexity does not increase. On the other hand, due to the archive of whole search history, many search methods could be produced based on history information. For engineering design applications, memory cost of non-revisiting scheme is not a serious problem.

In NrABC, to alleviate the above overhead, employed bee and scout bee stages are removed from standard ABC, and only onlooker bee stage is kept. The procedures of the NrABC algorithm are presented in Algorithm 1. It can bee seen that NrABC is more concise than standard ABC. Because non-revisiting scheme checks if a revisit happens before each evaluation, revisiting an existing solution is avoided. In case a revisit is detected, the associated solution is replaced by a randomly created one. The probability of producing the same solution is very low due to uniform random distribution. Thus, based on non-revisiting scheme, premature convergence and trapping in local optima is naturally detected and resolved. Note that in the cycle of NrABC, the best so far solution has chances to be discarded if a revisit comes across the same as the best so far solution. This is allowed as all solutions are memorized in BSP tree. BSP tree is able to provide necessary information for post-processing.

Algorithm 1 Procedures of the non-revisiting artificial bee colony algorithm

Require: $\boldsymbol{f}(\cdot), D, \mathbf{x}^{min}, \mathbf{x}^{max}, Np, feval = 0$
Ensure: Optimal solution found by the algorithm
1: Randomly create Np solutions between $\mathbf{x}^{min}$ and $\mathbf{x}^{max}$, $feval = feval + Np$;
2: Evaluate function values and fitness values of initial solutions;
3: **repeat**
4: **repeat**
5: Choose a food source as base solution depending on its fitness;
6: Produce $\mathbf{x}_{t+1}$ based on (5);
7: **if** $\mathbf{x}_{t+1}$ is a revisit **then**
8: Produce $\mathbf{x}_{t+1}$ based on (6);
9: **end if**
10: Evaluate function value and fitness value of $\mathbf{x}_{t+1}$, $feval = feval + 1$;
11: Greedy selection between $\mathbf{x}_{t+1}$ and $\mathbf{x}_t$;
12: Record $\mathbf{x}_{t+1}$ and related information in BSP tree;
13: **until** Np onlooker bees are sent out
14: Memorize the best so far solution;
15: **until** Termination criteria are met

Table 1 Configuration of standard ABC and PADE algorithms

Algorithm	Parameters
ABC	$Np = 30$, limit $= 100$
NrABC	$Np = 30$

From the viewpoint of algorithmic parameters, standard ABC contains population size Np and consecutive non-improved evaluation trials *limit*. While NrABC removes scout bee stage, the parameter *limit* is also removed. Only parameter Np has to be set by users before executing NrABC. Thus, the NrABC algorithm reduces the burden of algorithmic parameter setting.

4 Numerical experiment

In this section, the proposed NrABC algorithm is applied to deal with phased array design problem.

4.1 Experimental setting

Simulation configuration is described in the following. Three examples of phased array design are used to study how NrABC performs in handling different types of problems. Example 1 is amplitude-only synthesis of phased arrays. The sidelobes of example 1 is −30 dB with a −60 dB notch. Problem dimension D is 100 as 100 element factors is specified in this example. The range of amplitude values is between 0 and 1 for example 1; constant phase is used for building an amplitude-only case. Example 2 is a simple modification of example 1. Amplitude weights are restricted to be Taylor weights. The range of phase values is between 0 and $\pi/2$. Example 3 is a complex case with both amplitude and phase weights as parameters. Problem dimension D = 200 in example 3. The range of amplitude values is [0,1], and the range of phase values is [0,π].

Standard ABC and the proposed NrABC are applied to tackle the above three examples. The configurations of the test algorithms are shown in Table 1. Clearly, the NrABC algorithm contains less algorithmic parameters than standard ABC algorithm. In standard ABC, *limit* is often set

Table 2 Optimal function values found by the test algorithms for the three examples

Problem	Algorithm	Min	Med	Max	Mean	std
Example 1	ABC	−15.2937	−12.1810	−5.9880	−11.1377	2.9022
Example 1	NrABC	−16.9162	−15.8933	−10.0833	−15.4967	1.7260
Example 2	ABC	−19.2429	−18.0636	−16.8978	−18.0208	0.5926
Example 2	NrABC	−19.3504	−18.5705	−17.9930	−18.5763	0.4130
Example 3	ABC	8.8439	9.5340	9.7779	9.4790	0.2254
Example 3	NrABC	8.8165	9.2605	9.5634	9.2580	0.2043

Over 25 independent runs, min denotes the minimum objective function values overall 25 runs. Similarly, med, max, and mean are, respectively, the median, maximum, and mean values; std denotes standard deviation of function values over 25 runs

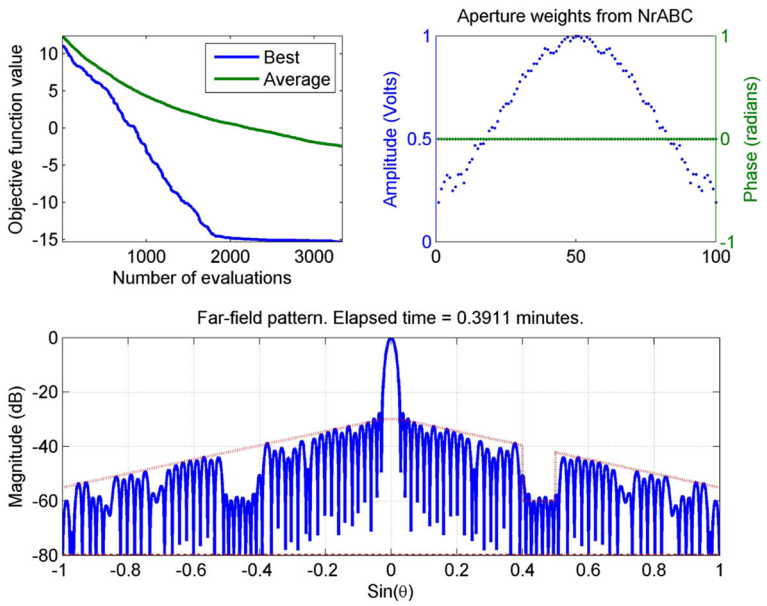

Fig. 3 The best result for the synthesis of amplitude-only phased array using NrABC

to $0.5NpD$. As D is very large in our examples, this setting needs too many cycles to covering the usefulness of scout bees. Hence, it is set to a fixed number in our experiment. Moreover, $Np = 30$ is proper for large problem dimension.

Termination condition is a fixed cost one. The maximum number of function evaluations (MFE) is set to 1e5 for all three examples, i.e., $MFE = 100{,}000$. Each algorithm is independently run 25 times for each example to gain an average performance. The synthesis of phased arrays and algorithms are implemented in MATLAB, and executed on a personal computer with a 4-core 2.50 GHz CPU and 4 GB of memory. This could provide a fair comparison environment for the test algorithms.

4.2 Simulation results

The optimal objective function values attained by each algorithm is shown in Table 2. This table contains the statistics of results for all three examples.

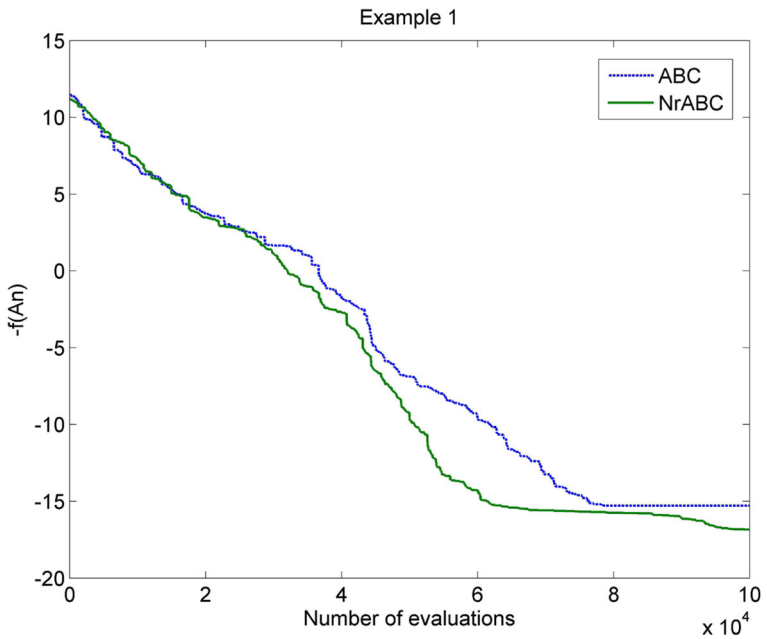

Fig. 4 Convergence graph of NrABC and ABC for example 1

Example 1: this case is constrained under the condition that output distribution is symmetrical. The best result over 25 runs is plotted in Fig. 3. The upper left graph shows the convergence curve of the NrABC algorithm, where "best" curve means the minimum function value in population and "average" curve is the average function value of solutions in population. The gap between "best" curve and "average" curve gradually increases along with the number of function evaluations. This indicates that NrABC is able to keep a large population diversity while it can also find good solutions. The upper right graph shows the ultimate aperture weights obtained from NrABC. For example 1, horizontal dotted line is phase weights as it is kept constant in this example. In Fig. 3, the bottom graph shows the far-field pattern. The dotted line and dash-dot line are, respectively, the upper and lower targets of sidelobe. Clearly, the curve obtained by NrABC well lies between the constraints, even for the deep notch. In short, NrABC presents good performance in solving example 1.

Figure 4 presents the convergence graph of the NrABC and ABC algorithms. It shows the best trial of each algorithm over 25 runs. Initially, the curves of both algorithms are intertwined. After 4000 function evaluations (FEs), their gap becomes larger, which means that old solution revisiting happens and non-revisiting scheme starts working and forces the algorithm search more diverse than ABC. Hence, the convergence graphs of both algorithms become close at 80,000 FEs. Hence, our NrABC algorithm explores more than standard ABC, and finally locates better solution than standard ABC.

Example 2: this case does not have the symmetry constraint of output distribution. The best result over 25 runs is plotted in Fig. 5. Amplitude weights are kept constant but are produced by Taylor weights. Similar to the results in example 1, NrABC also presents good performance in this case. Observed from the upper left convergence graph, "best" convergence curve sharply drops down in initial stage; it becomes gentle from 800 function evaluations (FEs) to 1800 FEs. The curve continues to drop down since 1800 FEs. This stage indicates that the algorithm traps in local optima, though it walks out the neighborhood of local optima after a search period.

Example 3: this case is constrained under the condition that output distribution is a symmetrical flat-top beam. The best result over 25 runs is plotted in Fig. 6. Seen from the bottom graph, a flat-top beam is desired under the constraint, whereas the curve obtained by NrABC is not as desirable as in examples 1 and 2. Note that example 3 is a much harder case than examples 1 and 2, as the number of decision variables of example 3 is twice of that of examples 1 and 2. The results obtained by NrABC is also good in this sense.

5 Conclusions

The main advantage of phased arrays is its capability to construct nearly arbitrary far-field pattern. This character is attained by well tuning of the amplitude and phase features of aperture. It is realized by phase shifters and attenuators of all radiators. Although phased arrays has such kind characteristics, it is difficult to attain a satisfying far-field pattern; it is also hard if such pattern is

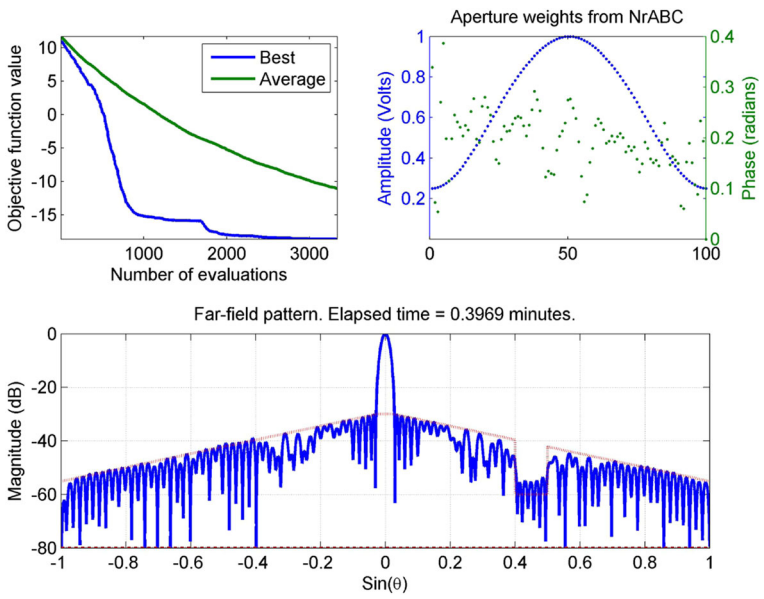

Fig. 5 The best result for the synthesis of phase-only phased array using NrABC

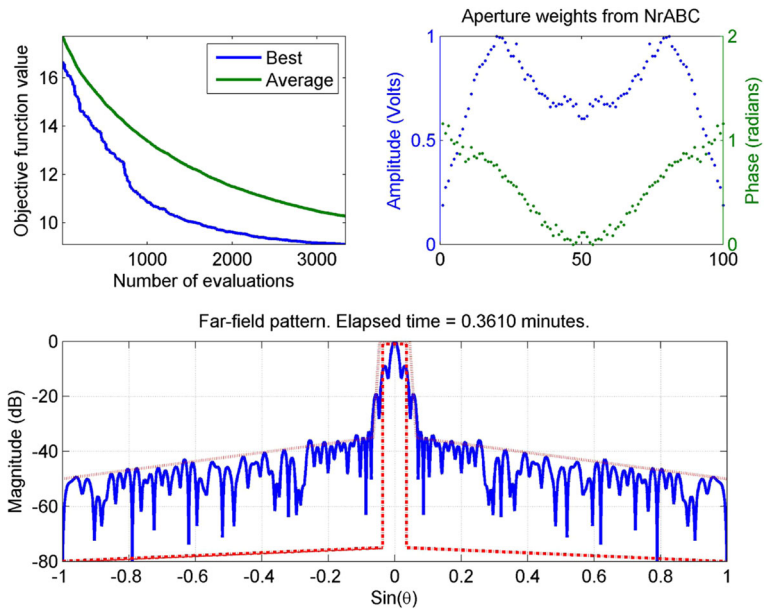

Fig. 6 The best result for the synthesis of complex phased array using NrABC

reachable with a given aperture [19]. Thus, metaheuristic approaches are suitable to deal with this kind of problems. Although convergence to global optimum could not be guaranteed by metaheuristic approaches, they can find promising solutions given a fixed-cost condition. This paper attempts to tackle synthesis of phased arrays based on a recently proposed paradigm—artificial bee colony (ABC).

The main contribution of this paper is to propose a non-revisiting artificial bee colony (NrABC) algorithm for tackling the synthesis of phased array problem. NrABC uses a non-revisiting scheme to record history evolution information of the algorithm. The information then assists the search of the algorithm. Exploitation is conducted by NrABC if a revisit is not detected; otherwise, exploration search is performed. With this scheme, the algorithm is able to keep population diversity along with evolutionary process. Moreover, employed bee stage and scout bee stage are removed from standard ABC, which makes NrABC more concise while it still keep the essence of ABC. Furthermore, parameter *limit* in standard ABC is not used any more in NrABC. Thus, NrABC contains less algorithmic parameter than standard ABC, which saves the efforts of users and makes the algorithm easier to use. Simulated on three examples, the NrABC algorithm presents good performance in both convergence process and solution quality. However, the performance of the proposed algorithm degrades when problem dimension $D = 200$. This issue would be investigated in the future.

Acknowledgements
This research was supported in part by the National Science Foundation of China (61603275, 61601329), and the Applied Basic Research Program of Tianjin (15JCYBJC51500, 15JCYBJC52300).

Competing interests
The authors declare that they have no competing interests.

Received: 21 September 2016 Accepted: 19 December 2016
Published online: 04 January 2017

References

1. Q Liang, Radar sensor wireless channel modeling in foliage environment: UWB versus narrowband. IEEE Sensors J. **11**(6), 1448–1457 (2011)
2. T Jiang, W Zang, C Zhao, J Shi, An energy consumption optimized clustering algorithm for radar sensor networks based on an ant colony algorithm. EURASIP J. Wirel. Commun. Netw. **2010**, 627253 (2010)
3. HK Min, MS Song, I Song, JH Lim, A frequency-sharing weather radar network system using pulse compression and sidelobe suppression. EURASIP J. Wirel. Commun. Netw. **2016**, 100 (2016)
4. Q Liang, Situation understanding based on heterogeneous sensor networks and human-inspired favor weak fuzzy logic system. IEEE Syst. J. **5**(2), 156–163 (2011)
5. Q Liang, X Cheng, S Huang, D Chen, Opportunistic sensing in wireless sensor networks: theory and applications. IEEE Trans. Comput. **63**(8), 2002–2010 (2014)
6. W Li, X Shi, Y Hei, An improved particle swarm optimization algorithm for pattern synthesis of phased arrays. Prog. Electromagn. Res. **82**, 319–332 (2008)
7. B Yan, S Wang, Y Bao, Research on an improved parallel algorithm based on user's requirement in cognitive radar network. EURASIP J. Wirel. Commun. Netw. **2015**, 47 (2015)
8. L Jin, Y Li, C Zhao, Z Wei, B Li, J Shi, Cascading polar coding and lt coding for radar and sonar networks. EURASIP J. Wirel. Commun. Netw. **2016**, 254 (2016)
9. DI Abu-Al-Nadi, TH Ismail, H Al-Tous, MJ Mismar, Design of linear phased array for interference suppression using array polynomial method and particle swarm optimization. Wirel. Pers. Commun. **63**(2), 501–513 (2012)

10. D Zhou, Orthogonal maximum margin projection subspace for radar target HRRP recognition. EURASIP J. Wirel. Commun. Netw. **2016**, 72 (2016)
11. C Ergun, K Hacioglu, Multiuser detection using a genetic algorithm in CDMA communications systems. IEEE Trans. Commun. **48**(8), 1374–1383 (2000)
12. HG Beyer, HP Schwefel, Evolution strategies—a comprehensive introduction. Nat. Comput. **1**(1), 3–52 (2002)
13. R Storn, K Price, Differential evolution—a simple and efficient heuristic for global optimization over continuous spaces. J. Global Optim. **11**(4), 341–359 (1997)
14. J Kennedy, RC Eberhart, in *Proc. IEEE Int. Conf. Neural Networks,* vol. 4. Particle swarm optimization (IEEE, Perth, Australia, 1995), pp. 1942–1948
15. D Karaboga, B Basturk, A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithms. J. Glob. Optim. **39**(3), 459–471 (2007)
16. Z Wu, TWS Chow, Neighborhood field for cooperative optimization. Soft. Comput. **17**(17), 819–834 (2013)
17. Y Shi, *Brain Storm Optimization Algorithm*. (Y Tan, Y Shi, Y Chai, G Wang, eds.) (Springer, Berlin, Heidelberg, 2011), pp. 303–309
18. F Neri, C Cotta, Memetic algorithms and memetic computing optimization: a literature review. Swarm Evol. Comput. **2**, 1–14 (2012)
19. DW Boeringer, DH Werner, Particle swarm optimization versus genetic algorithms for phased array synthesis. IEEE Trans. Antennas Propag. **52**(3), 771–779 (2004)
20. P Rocca, RJ Mailloux, G Toso, GA-based optimization of irregular subarray layouts for wideband phased arrays design. IEEE Antennas Wirel. Propag. Lett. **14**, 131–134 (2015)
21. MD Gregory, Z Bayraktar, DH Werner, Fast optimization of electromagnetic design problems using the covariance matrix adaptation evolutionary strategy. IEEE Trans. Antennas Propag. **59**(4), 1275–1285 (2011)
22. X Li, WT Li, XW Shi, J Yang, JF Yu, Modified differential evolution algorithm for pattern synthesis of antenna arrays. Prog. Electromagn. Res. **137**(137), 371–388 (2013)
23. X Zhang, SY Yuen, Improving artificial bee colony with one-position inheritance mechanism. Memet. Comput. **5**(3), 187–211 (2013)
24. C Ozturk, E Hancer, D Karaboga, A novel binary artificial bee colony algorithm based on genetic operators. Inf. Sci. **297**, 154–170 (2015)
25. SK Goudos, K Siakavara, JN Sahalos, Novel spiral antenna design using artificial bee colony optimization for UHF RFID applications. IEEE Antennas Wirel. Propag. Lett. **13**(13), 528–531 (2014)
26. X Zhang, X Zhang, A novel artificial bee colony algorithm for radar polyphase code and antenna array designs. EURASIP J. Wirel. Commun. Netw. **2016**(1), 40 (2016)
27. A Banharnsakun, T Achalakul, B Sirinaovakul, The best-so-far selection in artificial bee colony algorithm. Appl. Soft Comput. **11**(2), 2888–2901 (2011)
28. MH Bataineh, JI Ababneh, Synthesis of aperiodic linear phased antenna arrays using particle swarm optimization. Electromagnetics. **26**(26), 531–541 (2006)
29. KR Mahmoud, MI El-Adawy, SMM Ibrahem, R Bansal, SH Zainud-Deen, MPSO-MOM: a hybrid modified particle swarm optimization and method of moment algorithm for smart antenna synthesis. Electromagnetics. **28**(6), 411–426 (2008)
30. BS Dhaliwal, SS Pattnaik, Performance comparison of bio-inspired optimization algorithms for sierpinski gasket fractal antenna design. Neural Comput. & Applic. **27**(3), 585–592 (2016)
31. X Yang, Y Sun, T Zeng, T Long, Fast STAP method based on PAST with sparse constraint for airborne phased array radar. IEEE Trans. Signal Process. **64**(17), 4550–4561 (2016)
32. SU Khan, IM Qureshi, H Haider, F Zaman, B Shoaib, Diagnosis of faulty sensors in phased array radar using compressed sensing and hybrid IRLS-SSF algorithm. Wirel. Pers. Commun. **91**(1), 383–402 (2016). doi:10.1007/s11277-016-3466-7
33. NN Alotaibi, KA Hamdi, Switched phased-array transmission architecture for secure millimeter-wave wireless communication. IEEE Trans. Commun. **64**(3), 1303–1312 (2016)
34. SY Yuen, KC Chi, A genetic algorithm that adaptively mutates and never revisits. IEEE Trans. Evol. Comput. **13**(2), 454–472 (2009)
35. SY Yuen, CK Chow, in *IEEE Congress on Evolutionary Computation*. A non-revisiting genetic algorithm (IEEE, Singapore, 2007), pp. 4583–4590