Hindawi
International Journal of Antennas and Propagation
Volume 2020, Article ID 9483971, 12 pages
https://doi.org/10.1155/2020/9483971

Research Article

A Modified Wolf Pack Algorithm for Multiconstrained Sparse Linear Array Synthesis

Ting Wang,[1,2] **Ke-Wen Xia**,[1] **Hai-Lin Tang**,[2] **Su-Wei Zhang**,[3] **and Mukase Sandrine**[1]

[1]*School of Electronic Information Engineering, Hebei University of Technology, Tianjin 300000, China*
[2]*People's Liberation Army Air Force 93756, Tianjin 300000, China*
[3]*Yichang Testing Technology Research Institute, No. 58, Shengli Third Road, Yichang City, Hubei Province, China*

Correspondence should be addressed to Ke-Wen Xia; kwxia@hebut.edu.cn

Received 2 July 2019; Revised 28 November 2019; Accepted 3 December 2019; Published 13 January 2020

Academic Editor: Ana Alejos

Copyright © 2020 Ting Wang et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

The aim of the research is to propose a new optimization method for the multiconstrained optimization of sparse linear arrays (including the constraints of the number of elements, the aperture of arrays, and the minimum distance between adjacent elements). The new method is a modified wolf pack optimization algorithm based on the quantum theory. In the new method, wolves are coded by Bloch spherical coordinates of quantum bits, updated by quantum revolving gates, and selectively adaptively mutated when performing poorly. Because of the three-coordinate characteristics of the sphere, the number of global optimum solutions is greatly expanded and ultimately can be searched with a higher probability. Selective mutation enhances the robustness of the algorithm and improves the search speed. Furthermore, because the size of each dimension of Bloch spherical coordinates is always [−1, 1], the variables transformed by solution space must satisfy the constraints of the aperture of arrays and the minimum distance between adjacent elements, which effectively avoids infallible solutions in the process of updating and mutating the position of the wolf group, reduces the judgment steps, and improves the efficiency of optimization. The validity and robustness of the proposed method are verified by the simulation of two typical examples, and the optimization efficiency of the proposed method is higher than the existing methods.

1. Introduction

Since the 1960s, the sparse array has been widely studied for its high target resolution and low cost. It has been successfully applied in the fields of interference array in radio astronomy, high-frequency ground radar, and satellite receiving antenna against environmental interference [1–4]. Pattern synthesis as the key technology of the array antenna has been playing an important role in antijamming, interception rate, and parameter estimation [5–9].

The purpose of antenna pattern synthesis is to determine some parameters of the array antenna so that certain radiation characteristics of the array meet the design requirements [10]. These parameters include array element number, array element spacing, array element excitation amplitude, and phase coefficient. Compared with uniform array synthesis, the optimization of nonuniform array placement has always been a difficult problem. To solve this problem, many synthesis methods have been proposed, such as dynamic programming [11], fractional Legendre transform [12], simulated annealing [13], particle swarm optimization [14–16], and genetic algorithm [17–21].

In the literature [7], nonuniform arrays are divided into two categories. One is the sparse array based on a grid, in which the spacing of elements is only an integer multiple of half wavelength. [11, 13, 14]. The other is the sparse array of antenna elements distributed randomly in a certain aperture. In recent years, more attention has been paid to the second approach, which not only has more freedom but also can obtain lower peak side-lobe level under the same array element number and array aperture.

Because the genetic algorithm is suitable for nonlinear optimization problems, it has been widely used in antenna

design and optimization in recent years [17–21]. However, the problem of optimal placement of the sparse array is very complex. When the genetic algorithm is used to optimize the element position of the sparse array, sometimes it is difficult to obtain satisfactory solution in limited time. The main reasons are as follows: (1) the basic genetic algorithm has the disadvantages of slow convergence speed and easy to fall into local optimum when dealing with complex problems; (2) a sparse array spacing optimization is a multiconstrained optimization problem with constraints on the number of elements, aperture, and spacing of elements. In the operation of the optimization algorithm, the existence of infeasible solutions will seriously affect the optimization efficiency of the algorithm. Aiming at the first problem, the improved genetic algorithm is used in the literature [17, 18, 20] to optimize the sparse array element spacing and amplitude weighting. In reference [15, 16], an improved particle swarm optimization algorithm is used to obtain a better numerical solution. In order to solve the second problem, a new optimization model is proposed in reference [17]. By introducing intermediate variables and designing matrix transformations, generalized crossover operators, and mutation operators to deal with constraints, the infeasible solutions of gene recombination and mutation are effectively avoided under multiconstraints. In [18], the constraints of element spacing are separated from genetic operations by complex mapping criteria, and the purpose of avoiding the occurrence of infeasible solutions is also achieved.

Wolf pack algorithm (WPA) is a new swarm intelligence optimization algorithm proposed by Yang in 2007, which imitates wolf predation behavior and prey allocation. Verified by typical test functions, the WPA has been proved to have better global optimization ability and faster search speed than the GA in complex nonlinear optimization problems and has achieved good results in sensor optimal placement [22] and hydropower station optimal dispatch [23]. However, the application of the WPA in array synthesis has not been discussed at home and abroad. This paper takes it as a research topic.

In recent years, quantum computing has attracted wide attention for its excellent performance. Many scholars have integrated the quantum theory with intelligent optimization algorithms and proposed many efficient quantum evolutionary algorithms [24–28]. Inspired by this, we introduced quantum Bloch spherical coordinates into the WPA and proposed a modified quantum wolf pack algorithm (MQWPA) based on Bloch spherical coordinates. Then, the new algorithm is applied to the multiconstraint synthesis of sparse linear arrays. By means of intermediate variables, the constrained optimization problem of the element spacing is successfully transformed into a nonconstrained optimization problem. At the same time, because the size of each dimension of Bloch spherical coordinates is always [−1, 1], the distance variable obtained by solution space transformation can satisfy the constraints of the aperture of arrays and the minimum distance between adjacent elements. Therefore, wolves will not exceed the feasible solution space in the process of location updating and mutation. The whole algorithm no longer needs to judge the feasible solution and improves the optimization efficiency. Compared with the existing literature methods, this paper presents a new scheme for the multiconstraint synthesis of one-dimensional sparse linear array, which has higher optimization efficiency.

2. The Problem Formulation

Pattern synthesis is a complex optimization problem, which determines the position, amplitude, and phase of the antenna array elements according to the shape or performance index of the given beam pattern. In engineering design, in order to reduce the mutual coupling between the elements and keep the narrow main lobe width, some constraints are imposed on the aperture of the array and the spacing of the elements. At this time, pattern synthesis becomes a multiconstrained, multivariable, and nonlinear global optimization problem [1–8].

In this paper, the problem of multiconstraint synthesis of the symmetric one-dimensional sparse linear array is discussed. The array structure is shown in Figure 1.

The number of elements of the symmetrical sparse linear array is $2N+1$ ($N \in Z^+$), and all elements are the same and have no directivity. I_n and d_n represent the excitation and position of the n-th antenna element, respectively, and $\theta \in [0, \pi]$ is the scanning angle. Because the array has a symmetrical structure, we can set the position of the central element of the array as the reference origin and then $d_0 = 0$ and $d_n = -d_n$. At this time, the pattern function of the array is expressed as

$$E(u) = \sum_{n=-N}^{N} I_n \exp(jkud_n), \tag{1}$$

where k equals $2\pi/\lambda$ and u equals $\cos\theta$.

Because arrays are symmetrical, we only need to consider the structure of arrays when $n \in [0, N]$. When the aperture of the sparse array is constant, the distance between adjacent elements is not less than d_c and the peak side-lobe level (PSLL) is the lowest; the sparse array synthesis optimization model can be expressed as

$$\begin{cases} \min & \{\mathrm{PSLL}(d_1, d_2, \ldots, d_{N-1})\} \\ & d_i - d_j \geq d_c > 0 \\ \text{s.t.} & i, j \in Z, 0 \leq j < i \leq N \\ & d_0 = 0; d_N = L. \end{cases} \tag{2}$$

For the above optimization problems, when the intelligent algorithm is used to optimize, the penalty function is usually introduced into the fitness function. By punishing the infeasible solution, the whole optimization problem evolves towards the feasible solution. This method is applicable in the case of few and simple constraints, while it is difficult for complex constrained optimization problems. For this reason, this paper adopts the method of the literature [17]. In this method, the intermediate variables are used to replace the element spacing and the minimum spacing constraint problem is transformed into a nonminimum spacing constraint problem. The specific steps are as follows.

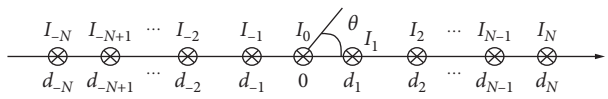


Figure 1: Geometric structure of a sparse linear array.

Divide d_i into two parts, x_i and $(i-1)d_c$. Then, we can get

$$\begin{aligned} D &= \begin{bmatrix} d_1 \\ d_2 \\ d_3 \\ \vdots \\ d_{N-2} \\ d_{N-1} \end{bmatrix} = \begin{bmatrix} x_1 + d_c \\ x_{2+} 2d_c \\ x_3 + 3d_c \\ \vdots \\ x_{N-2} + (N-2)d_c \\ x_{N-1} + (N-1)d_c \end{bmatrix} \\ &= \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_{N-2} \\ \vdots \\ x_{N-1} \end{bmatrix} + \begin{bmatrix} d_c \\ 2d_c \\ 3d_c \\ \vdots \\ (N-2)d_c \\ (N-1)d_c \end{bmatrix} = X + \begin{bmatrix} d_c \\ 2d_c \\ 3d_c \\ \vdots \\ (N-2)d_c \\ (N-1)d_c \end{bmatrix}. \end{aligned} \tag{3}$$

In order to ensure that the spacing between adjacent elements satisfies the following conditions,

$$\min\left\{d_i - d_j\right\} \geq d_c, \quad 0 \leq j < i \leq N. \tag{4}$$

The following relationships must be established:

$$0 \leq x_1 \leq x_2 \leq x_3 \leq \cdots \leq x_N \leq [L - Nd_c]. \tag{5}$$

Through the above operations, the problem of element spacing constraints is transformed into an unconstrained optimization problem of x_i on interval $[L - Nd_c]$. Then, the array synthesis optimization model can be transformed into the following form:

$$\begin{cases} \min\limits_{X}\{\text{PSLL}\}, \\ X = \left\{(x_1, x_2, \cdots, x_N) \,\middle|\, x_1 \leq x_2 \leq x_3 \leq \cdots \leq x_N \in [0, L - Nd_c]\right\}. \end{cases} \tag{6}$$

3. Proposed Method

3.1. Modified Wolf Pack Algorithm. Wolf pack algorithm (WPA) is a new group search method for simulating the social hierarchy mechanism and predatory behavior of wolves in nature. The heuristic optimization of the WPA is mainly realized by simulating three kinds of wolf swarm intelligence behaviors: wandering, summoning, and besieging, as well as the rules of leading wolf generation and the mechanism of wolf swarm renewal. The algorithm changes the position of wolves with the change of the operator and measures the position of individual wolves by the value of an objective function. The algorithm has excellent computational robustness and global search ability [22].

The WPA is more efficient in optimizing the extreme of typical test functions than the GA and PSO [22]. However, it is still easy to fall into local optimum in multidimensional nonlinear problems. In order to improve the global optimization ability of the algorithm, this paper introduces quantum Bloch spherical coordinates into the WPA and then proposes the MQWPA. The new algorithm uses three-dimensional Bloch spherical coordinates to encode wolves and carries out stagnation detection to selectively mutate wolves that do not perform well. The new algorithm significantly improves the diversity of population, avoids premature convergence, and achieves a good balance in convergence speed and accuracy through selective mutation. The basic principle of the MQWPA is given below.

3.1.1. Initialization of Quantum Wolves. In the spherical coordinates of Bloch, a point P can be determined by two angles θ and φ. See Figure 2.

Qubits with Bloch spherical coordinates are expressed as $\left|\varphi\right> = \left[\cos\varphi \sin\theta \sin\varphi \sin\theta \cos\theta\right]^T$. In the MQWPA, the initial wolves were encoded by Bloch spherical coordinates of qubits, which were coded as follows:

$$P_i = \begin{vmatrix} \cos\varphi_{i1}\sin\theta_{i1} & \cos\varphi_{i2}\sin\theta_{i2} & \cdots & \cos\varphi_{in}\sin\theta_{in} \\ \sin\varphi_{i1}\sin\theta_{i1} & \sin\varphi_{i2}\sin\theta_{i2} & \cdots & \sin\varphi_{in}\sin\theta_{in} \\ \cos\theta_{i1} & \cos\theta_{i2} & \cdots & \cos\theta_{in} \end{vmatrix}, \tag{7}$$

where φ_{ij} equals $(2\pi \times \text{rand})$ and θ_{ij} equals $(\pi \times \text{rand})$. rand are the random numbers between $(0, 1)$, $i = 1, 2, \ldots, m$, $j = 1, 2, \ldots, n$, m are the size of the wolf pack, and n is the dimension of the function to be optimized.

3.1.2. Solution Space Transformation. In the MQWPA, each one-dimensional traversal space of wolves in Bloch coordinates is $[-1, 1]$. In order to calculate the adaptive value of wolves, the solution space transformation is required to map the three coordinate positions occupied by each wolf from the unit space to the solution space of the optimization problem. Assuming that the Bloch coordinate of the i-th wolf in the j-th qubit is $\begin{bmatrix} x_{ij} & y_{ij} & z_{ij} \end{bmatrix}^T$, the corresponding solution space variable is

$$\begin{cases} X_{ix}^j = 0.5\left[b_j\left(1 + x_{ij}\right) + a_j\left(1 - x_{ij}\right)\right], \\ X_{iy}^j = 0.5\left[b_j\left(1 + y_{ij}\right) + a_j\left(1 - y_{ij}\right)\right], \\ X_{iz}^j = 0.5\left[b_j\left(1 + z_{ij}\right) + a_j\left(1 - z_{ij}\right)\right], \end{cases} \tag{8}$$

where $[a_j, b_j]$ is represented as the value range of the j qubit, $i = 1, 2, \ldots, m$ and $j = 1, 2, \ldots, n$.

From the above transformation relations, we can find that each solution of the optimization problem will correspond to three feasible solutions in Bloch coordinates.

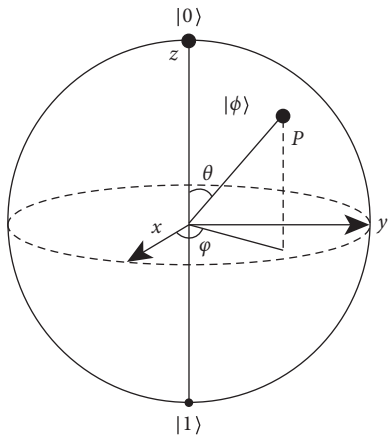


Figure 2: Bloch sphere representation of quantum bits.

According to the proof in [25], we know that this coding method can significantly increase the diversity of the population, expand the number of global optimal solutions, and improve the probability of finding global optimal solutions. Therefore, compared with the standard WPA, the MQWPA must have better global optimization ability.

3.1.3. Quantum Bit Status Update. In the MQWPA, the corresponding qubits of wolves will also perform three kinds of intelligent behaviors, namely, wandering, attacking, and siege. In these processes, if a wolf in the population is less adaptive than the leader wolf, then the leader wolf will be replaced by the wolf.

The position of the leader wolf in the process of wandering and attacking is

$$P_{il} = \left\{\cos\varphi_{il,1}\sin\theta_{il,1},\ \cos\varphi_{il,2}\sin\theta_{il,2},\ \ldots,\ \cos\varphi_{il,n}\sin\theta_{il,n}\right\}. \tag{9}$$

The location of the food is

$$P_g = \left\{\cos\varphi_{g,1}\sin\theta_{g,1},\ \cos\varphi_{g,2}\sin\theta_{g,2},\ \ldots,\ \cos\varphi_{g,n}\sin\theta_{g,n}\right\}. \tag{10}$$

Based on the above assumptions, the update behavior of wolves can be described as follows:

① When the i-th wolf is a wolf detector, the updating formula of quantum bit amplitude and angular increment in the j-dimensional space is as follows:

$$\begin{cases} \theta_{i,j}(t+1) = \theta_{i,j}(t) + \sin\left(2\pi\times\dfrac{a_1}{h}\right)\times \text{step}_a^j \\ \quad + \lambda\cdot\text{step}_c^j\left|\theta_{g,j} - \theta_{i,j}\right|, \\ \varphi_{i,j}(t+1) = \varphi_{i,j}(t) + \sin\left(2\pi\times\dfrac{a_1}{h}\right)\times \text{step}_a^j \\ \quad + \lambda\ \text{step}_c^j\cdot\left|\varphi_{g,j} - \varphi_{i,j}\right|, \\ \theta_{i,j}(t+1) = \theta_{i,j}(t) + \eta\theta_{i,j}(t+1), \\ \varphi_{i,j}(t+1) = \varphi_{i,j}(t) + \eta\varphi_{i,j}(t+1). \end{cases} \tag{11}$$

② When the i-th wolf is a fierce wolf, the updating formula of quantum bit amplitude and angular increment in the j-dimensional space is as follows:

$$\begin{cases} \theta_{i,j}(t+1) = \theta_{i,j}(t) + \dfrac{\theta_{il,j} - \theta_{i,j}}{\left|\theta_{il,j} - \theta_{i,j}\right|}\times \text{step}_b^j \\ \quad + \lambda\cdot\text{step}_c^j\left|\theta_{g,j} - \theta_{i,j}\right|, \\ \varphi_{i,j}(t+1) = \varphi_{i,j}(t) + \dfrac{\varphi_{il,j} - \varphi_{i,j}}{\left|\varphi_{il,j} - \varphi_{i,j}\right|}\times \text{step}_b^j \\ \quad + \lambda\cdot\text{step}_c^j\left|\varphi_{g,j} - \varphi_{i,j}\right|, \\ \theta_{i,j}(t+1) = \theta_{i,j}(t) + \eta\theta_{i,j}(t+1), \\ \varphi_{i,j}(t+1) = \varphi_{i,j}(t) + \eta\varphi_{i,j}(t+1), \end{cases} \tag{12}$$

where t means the number of iterations. step_a means the walking step of wolves. step_b means the running step of wolves. step_c means the attack step of wolves when they perform siege. $i = 1, 2, \ldots, m$ and $j = 1, 2, \ldots n$. $a1 = 1, 2, \ldots, h$. h denotes the number of exploring directions for wolves. λ is a random number with a uniform distribution between $[-1, 1]$. $\eta = 0.7298$ is a compression factor. The compression factor is used to accelerate population convergence.

③ In the new algorithm, the wolf swarm position is moved by a quantum revolving gate. The quantum bit phase of the wolf pack position is updated by the following quantum revolving gate U:

$$U = \begin{bmatrix} \cos\varphi\cos\theta & -\sin\varphi\cos\theta & \sin\theta\cos(\varphi+\varphi) \\ \sin\varphi\cos\theta & \cos\varphi\cos\theta & \sin\theta\sin(\varphi+\varphi) \\ -\sin\theta & -\tan(\varphi/2)\sin\theta & \cos\theta \end{bmatrix}, \tag{13}$$

$$U\begin{bmatrix} \cos\varphi\sin\theta \\ \sin\varphi\sin\theta \\ \cos\theta \end{bmatrix} = \begin{bmatrix} \cos(\varphi+\varphi)\sin(\theta+\theta) \\ \sin(\varphi+\varphi)\sin(\theta+\theta) \\ \cos(\theta+\theta) \end{bmatrix}. \tag{14}$$

3.1.4. Quantum Bit State Selective Variation. Through analysis, we find that each time the algorithm randomly selects a certain number (r) of poorly performing wolves to eliminate them and then generates the same number of new wolves as normal distribution to keep the population number m constant. This scheme prevents the convergence stagnation near the local optimal solution to some extent, but it also increases the computational complexity. Because the wolves are initialized randomly, the phenomenon of convergence and stagnation in wolf search is not certain. Also, we can detect whether the algorithm is in convergence stagnation by certain means. When the test results are positive, it is known from the

literature [16] that it is a good solution to increase population diversity through mutation. Through variation, the wolves can jump out of the local best and try to move closer to the overall best (food). When the test result is negative, the original wolf information is maintained and the next search process is continued without mutation. In this way, the global search ability of the algorithm can be guaranteed and the search speed of the algorithm can be improved.

In response to the above issues, this article proposes two improvements:

① In the iterative process of the algorithm, the convergence stagnation detection mechanism is introduced. The detection formula is as follows:

$$\begin{cases} f_{\text{avg}} = \dfrac{\sum_{i=1}^{m} \text{fit}(P_{il})}{m}, \\ \\ f_g = \text{fit}\left(P_g\right), \end{cases} \tag{15}$$

where f_g denotes the fitness of the head wolf and f_{avg} denotes the average best fitness of the individual wolf pack. If the value of f_{avg}/f_g still tends to 1 after t_0 ($t_0 = 20$) iterations and the algorithm does not terminate, the algorithm is considered to be stagnant.

② When the algorithm is stagnating, the wolves are selectively mutated. According to the previous analysis, the probability of variation in the wolves should be proportional to the distance between wolves and the leader. The distance corresponds exactly to the wolves' fitness. In the minima optimization, the smaller the adaptation value, the smaller the distance. So, we mutate according to the level of wolves' fitness. The greater the wolf's fitness value, the higher the probability of its mutation. This will not only maintain the diversity of the population but also increase the iterative efficiency.

Firstly, we find out the different mutation probabilities of wolves. The mutation probability of the k-th wolf is as follows:

$$\rho_k = \left| \frac{f(k,1) + f(k,2) + f(k,3)}{3 * \text{fit}\left(P_g\right)} \right|, \tag{16}$$

where m is the size of the wolf pack, and the three fitness values of the k-th wolf are $f(k,1)$, $f(k,2)$, and $f(k,3)$, $k = 1, 2, \ldots, m$. Each wolf is assigned a random number Rand between $(0,1)$. If the random number is less than the probability of variation calculated by the wolf, the wolf is mutated. On the contrary, the wolf population remains in its original state.

In the MQWPA, mutation operation is realized by constructing operator V:

$$V = \begin{bmatrix} 0 & \cot\theta & 0 \\ \cot\theta & 0 & 0 \\ 0 & 0 & \tan\theta \end{bmatrix}, \tag{17}$$

$$V \begin{bmatrix} \cos\varphi \sin\theta \\ \sin\varphi \sin\theta \\ \cos\theta \end{bmatrix} = \begin{bmatrix} \cos\left(\dfrac{\pi}{2} - \varphi\right)\sin\left(\dfrac{\pi}{2} - \theta\right) \\ \sin\left(\dfrac{\pi}{2} - \varphi\right)\sin\left(\dfrac{\pi}{2} - \theta\right) \\ \cos\left(\dfrac{\pi}{2} - \theta\right) \end{bmatrix}. \tag{18}$$

From the above formula, it is known that the variation is actually a larger rotation of the qubit along the Bloch sphere and the rotation angles are $\Delta\theta_{ij} = \pi/2 - 2\theta_{ij}$ and $\Delta\varphi_{ij} = \pi/2 - 2\varphi_{ij}$. Using the quantum Hadamard gate to perform the mutation operation on the rotation angle can effectively improve the population diversity and avoid premature convergence.

The implementation steps of the improved new algorithm can be summarized as follows:

Step 1: initialize the wolves based on the quantum Bloch coordinate.

Step 2: calculate the fitness of wolves and select the leader wolves.

Step 3: intelligent search of wolves, including roaming, running, and siege.

Step 4: update the leader wolf position; then, according to the stasis test results, the wolves with poor performance were selectively mutated.

Step 5: determine whether the optimization accuracy requirement or the maximum number of iterations is met. If reached, output the three coordinate positions of the leader wolf and compare the corresponding adaptive values of the three coordinates. The best coordinates are the optimal solution of the problem; otherwise, Step 2 is carried out.

The specific flow chart is shown in Figure 3.

3.2. MQWPA for Antenna Pattern Synthesis. Through the extremum optimization experiments of various typical test functions, we find that the MQWPA is very suitable for solving complex nonlinear optimization problems. Therefore, this paper applies the MQWPA to the multiconstraint synthesis of the symmetric one-dimensional sparse linear array, hoping to get good results.

3.2.1. Wolf Colony Initialization and Fitness Function Creation. In order to optimize the pattern synthesis model represented by formula (6), the wolf population with the number of individuals M and dimension N is used as the intermediate population. Wolves are coded by quantum bit Bloch spherical coordinates in the following way:

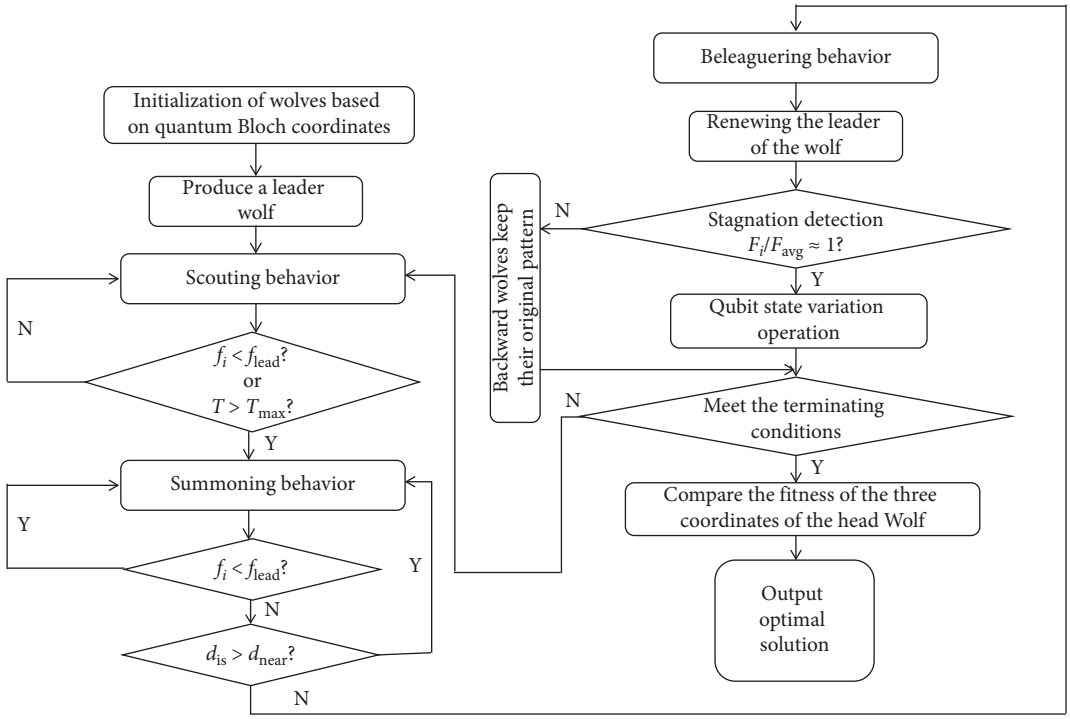


Figure 3: MQWPA flow chart.

$$P_i = \begin{vmatrix} \cos\varphi_{i1}\sin\theta_{i1} & \cos\varphi_{i2}\sin\theta_{i2} & \cdots & \cos\varphi_{iN}\sin\theta_{iN} \\ \sin\varphi_{i1}\sin\theta_{i1} & \sin\varphi_{i2}\sin\theta_{i2} & \cdots & \sin\varphi_{iN}\sin\theta_{iN} \\ \cos\theta_{i1} & \cos\theta_{i2} & \cdots & \cos\theta_{iN} \end{vmatrix}, \tag{19}$$

where φ_{ij} equals $(2\pi \times \text{rand})$ and θ_{ij} equals $(\pi \times \text{rand})$. rand are the random numbers between $(0, 1)$, $i = 1, 2, \ldots, M$ and $j = 1, 2, \ldots, N$. M is the size of the wolf pack, and N is the dimension of the function to be optimized.

The initial wolves are randomly distributed on the Bloch sphere, and the traversal space of each wolf is [−1, 1]. According to the previous analysis, we know that each wolf corresponds to three approximate solutions in the feasible solution space of the optimization problem. In the multi-constraint synthesis of symmetric one-dimensional sparse linear arrays, the search region of the elements is

$$\text{SP} = L - 2d_c - (N-2)d_c = L - Nd_c. \tag{20}$$

The Bloch coordinate of the j-th qubit on the i-th wolf is $\begin{bmatrix} x_{ij} & y_{ij} & z_{ij} \end{bmatrix}^T$. According to (13), three approximate solutions in the corresponding feasible solution space can be obtained as follows:

$$\begin{cases} X_{ix}^j = \left(1 + x_{ij}\right)\dfrac{(L - Nd_c)}{2}, \\ X_{iy}^j = \left(1 + y_{ij}\right)\dfrac{(L - Nd_c)}{2}, \\ X_{iz}^j = \left(1 + z_{ij}\right)\dfrac{(L - Nd_c)}{2}. \end{cases} \tag{21}$$

We call the above three approximate solutions X, Y, and Z, respectively. That is to say, after initialization of wolves, the corresponding three feasible solution matrices in the feasible solution space are as follows:

$$X = \left[\overline{X_{ix}^j}, \overline{X_{2x}^j}, \ldots, \overline{X_{Mx}^j}\right] = \begin{bmatrix} X_{ix}^1 & X_{2x}^1 & \cdots & X_{Mx}^1 \\ X_{1x}^2 & X_{2x}^2 & \cdots & X_{Mx}^2 \\ \cdots & \cdots & \cdots & \cdots \\ X_{1x}^N & X_{2x}^N & \cdots & X_{Mx}^N \end{bmatrix}, \tag{22}$$

$$Y = \left[\overline{X_{iy}^j}, \overline{X_{2y}^j}, \cdots, \overline{X_{My}^j}\right] = \begin{bmatrix} X_{iy}^1 & X_{2y}^1 & \cdots & X_{My}^1 \\ X_{1y}^2 & X_{2y}^2 & \cdots & X_{My}^2 \\ \cdots & \cdots & \cdots & \cdots \\ X_{1y}^N & X_{2y}^N & \cdots & X_{My}^N \end{bmatrix}, \tag{23}$$

$$Z = \left[\overline{X_{iz}^j}, \overline{X_{2z}^j}, \ldots, \overline{X_{Mz}^j}\right] = \begin{bmatrix} X_{iz}^1 & X_{2z}^1 & \cdots & X_{Mz}^1 \\ X_{1z}^2 & X_{2z}^2 & \cdots & X_{Mz}^2 \\ \cdots & \cdots & \cdots & \cdots \\ X_{1z}^N & X_{2z}^N & \cdots & X_{Mz}^N \end{bmatrix}. \tag{24}$$

In order to ensure that the spacing between adjacent elements is larger than the minimum spacing, it is necessary to rank the variables of each dimension intermediate solution of wolves and then add $d_c, 2d_c, \ldots, (N-1)d_c$. The actual distance intervals DX, DY, and DZ, respectively, are obtained. The analysis shows that the elements in DX, DY, and DZ all meet the condition of restriction that the distance between adjacent elements is not less than that of d_c.

Since the aim of linear array synthesis is to obtain the lowest peak side-lobe level (PSLL), the fitness function is constructed as follows:

$$\text{fitness}\left(d_{-N},\ldots,d_0,\ldots,d_N\right)=\left|\frac{FF_{\max}}{\max\left[E\left(u\right)\right]}\right|=\left|\frac{FF_{\max}}{\max\left[\sum_{n=1}^{N}2\cos\left(kud_n\right)+1\right]}\right|. \tag{25}$$

The larger the fitness of wolves, the lower the PSLL obtained by array synthesis. $\text{FF}_{\max}$ denotes the peak value of the main lobe of the array pattern, and u can only be selected in the side-lobe region of the pattern. The corresponding fitness functions of DX, DY, and DZ are FX, FY, and FZ, respectively. The best value of the three is chosen as the optimal result of this iteration.

3.2.2. Predatory Behavior of Wolves. The wolves generated in the MQWPA are heuristically searched through four main predatory behaviors: wandering, summoning, sieging, and competitive renewal. The location of wolves is updated according to the rule of (11)–(14). Then, the solution space is transformed and the fitness is calculated, and the selective variation of wolves is carried out through stagnation detection.

Because the size of each dimension is $[-1, 1]$, that is, $x_{ij} \in [-1, 1]$, $y_{ij} \in [-1, 1]$, and $z_{ij} \in [-1, 1]$, the wolves are updated according to (11)–(14), and then the following results can be obtained by solving space transformation:

$$\begin{cases} 0 \le X_{ix}^{j} \le L - Nd_c, \\ 0 \le X_{iy}^{j} \le L - Nd_c, \\ 0 \le X_{iz}^{j} \le L - Nd_c. \end{cases} \tag{26}$$

The variables of each dimension intermediate solution are sorted by size and added with $d_c, 2d_c, \ldots, (N-1)d_c$. From our analysis, we can see that the variable of element spacing must satisfy the multiconstraint condition that the aperture is L and the distance between adjacent elements is not less than d_c. It can be seen that due to the characteristics of Bloch spherical coordinates, the infeasible solutions of wolves are effectively prevented in the updating process, the steps of judgment are reduced, and the efficiency of optimization is improved.

3.2.3. Selective Variation. In order to avoid falling into local optimum, the stagnation detection mechanism is introduced in the process of algorithm optimization. When the algorithm stagnates, a selective variation of wolves is detected. The variation mode is that the quantum bit rotates along the Bloch sphere, and the rotation amplitude is large, which increases the population diversity better.

4. Simulation Examples

The fractional-order Legendre transformation method proposed in [12] achieves the multiconstrained thin-line linear array with the goal of reducing the PSLL. The literature [17] also obtained good optimization results using the MGA (modified genetic algorithm). In recent years, some new efficient heuristic algorithms have been proposed and have achieved good results in complex optimization problems, such as the MQPSO (modified quantum particle swarm optimization) proposed in [26]. The algorithm designs a new location and dynamic parameter update strategy, which greatly improves the algorithm search efficiency and has an excellent ability to deal with nonlinear optimization problems. Then, in order to verify the validity and robustness of the MQWPA, it is compared with the MGA [17] and MQPSO in two typical examples. The simulation environment of the example is as follows: Lenovo G460, CPU: Intel (R) Core (TM) i3-380M 2.53 GHz, memory: 2.00 GB, hard disk capacity: 500 GB, software platform: Windows 7 (32-bit operating system), and simulation software: MATLAB 2012.

4.1. Example 1. Linear arrays with 17 elements have a central symmetry and an aperture of 9.744λ. All elements are omnidirectional with equal amplitude, and the beam direction is 0°. Sparse synthesis of arrays is needed. The requirements for the new array are as follows: the aperture is unchanged, the distance between adjacent elements is greater than half wavelength, and the side-lobe level of the pattern is the lowest.

The analysis shows that, in this case $N = 8$ and $L = 4.872\lambda$. This paper uses the MQWPA to optimize. The basic parameters are as follows: population size $M = 50$, maximum iteration steps 300, wolf detection ratio factor $\alpha = 4$, maximum number of strolls $T_{\max} = 20$, distance determination factor $\omega \in [100\ \ 1000]$, step size factor $S \in [100\ \ 1600]$, and update ratio factor $\beta = 6$. In order to avoid falling into local optimum, the stagnation detection mechanism was adopted and the wolves were mutated selectively.

In order to verify the robustness of the proposed method, 10 simulation experiments were carried out independently and randomly for all three algorithms.

Table 1 lists the optimization results of the three algorithms. It can be seen from the table that the optimal array PSLL obtained by the MQWPA is the lowest, which is 0.1224 dB and 0.1014 dB lower than that obtained by the MGA and MQPSO, respectively.

Figures 4–6 are the optimal convergence curve, the worst convergence curve, and the average convergence curve of the three algorithms in 10 simulation experiments. From the three figures, we can see that the best PSLL of the array obtained from 10 simulation experiments of the MQWPA is −19.9194 dB and the worst PSLL is −19.7991 dB. The difference between them is only 0.1203 dB. However, the best PSLL obtained by the MQPSO algorithm is 0.2701 dB lower than the worst PSLL. The best PSLL obtained by the MGA is

Table 1: Comparison of the best results of three algorithms for optimizing example 1 (unit: λ).

Element spacing	MGA	MQPSO	MQWPA
$d_1 - d_0$	0.5002	0.5000	0.5001
$d_2 - d_1$	0.5002	0.5000	0.5002
$d_3 - d_2$	0.5009	0.5000	0.5000
$d_4 - d_3$	0.5245	0.5000	0.5001
$d_5 - d_4$	0.5738	0.5700	0.5321
$d_6 - d_5$	0.7066	0.7071	0.7389
$d_7 - d_6$	0.7868	0.7592	0.7526
$d_8 - d_7$	0.7792	0.7941	0.8101
PSLL	−19.7970 dB	−19.8180 dB	−19.9194 dB
Operation hours	623s	451s	244s

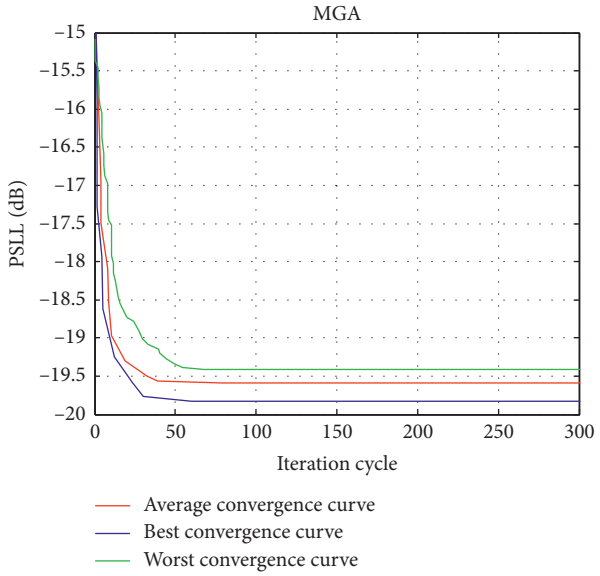


Figure 4: Convergence curve of fitness function. MGA for Example 1 optimization.

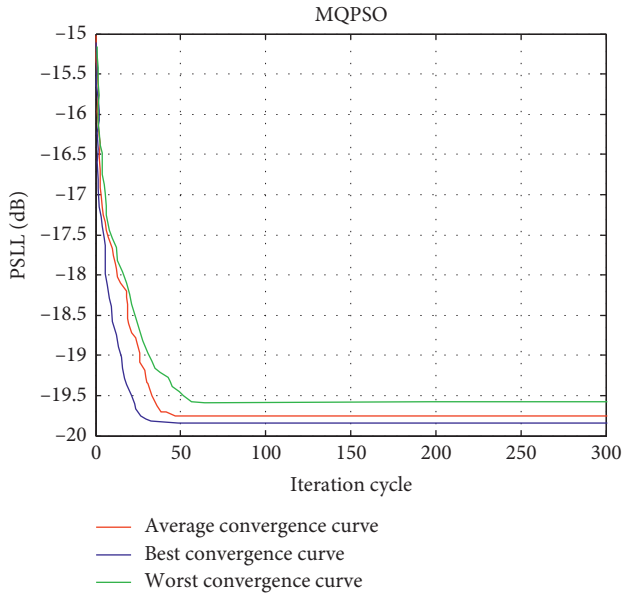


Figure 5: Convergence curve of fitness function. MQPSO for Example 1 optimization.

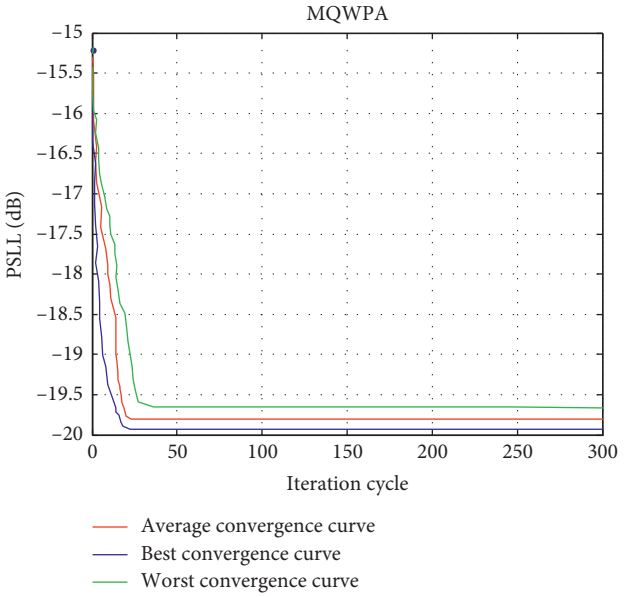


Figure 6: Convergence curve of fitness function. MQWPA for Example 1 optimization.

0.4330 dB lower than the worst PSLL. Obviously, the MQWPA has the best numerical stability. Moreover, from the above fitness function convergence curve, we can see that the MGA converges around 70 generations and the MQPSO algorithm converges around 50 generations. The proposed MQWPA has converged in about 25 generations. The MQWPA takes 244 seconds, far less than the other two algorithms. It is obvious that the MQWPA has both good global optimization ability and high optimization efficiency in Case 1 optimization.

Although the MQWPA has excellent optimization performance, the algorithm involves relatively many parameters. After many simulations, we found that the distance judgment factor ω and the step factor S of the algorithm have a great influence on the performance of the algorithm in the antenna pattern synthesis. Therefore, this paper mainly analyzes these two parameters.

Keep the other parameters unchanged, set $S = \{100, 200, 400, 600, 800, 1000, 1200, 1400, 1600\}$, when $\omega = \{100, 200, 300, 400, 500, 600, 700, 800, 900, 1000\}$, and continue to optimize example 1 with the MQWPA. Each combination of parameters was optimized 100 times and then averaged. The statistical results are shown in Figure 7.

It can be clearly seen from Figure 7 that when $\omega = 500$ and $S = 800$, the optimization effect of the MQWPA is the best and the PSLL of the array is the lowest.

Our analysis believes that the distance determination factor w is an important parameter to control the change from the running state to the siege behavior of the fierce wolves. When w is increased (less than 500), the d_{near} decreases. The fierce wolves will turn into siege when they are closer to the leader wolf. At this time, only a small solution region near the leader wolf will be finely searched, which promotes the fast convergence of the algorithm. Finally, the average number of iterations is reduced and the

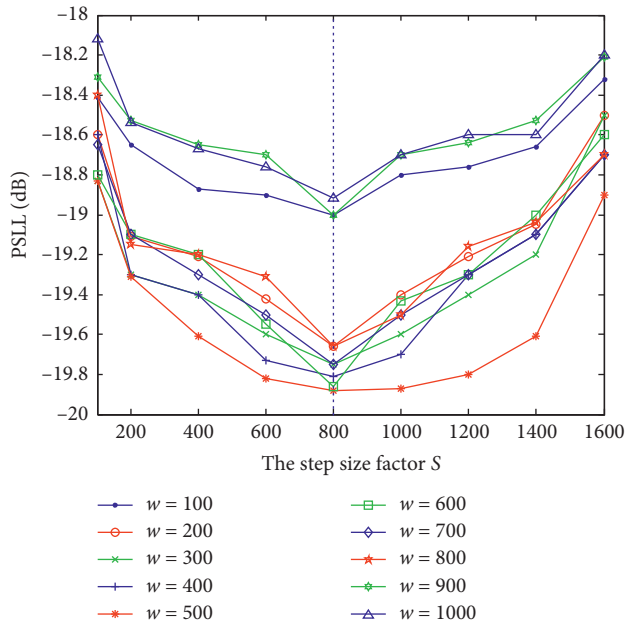


Figure 7: Effect of parameters S and ω on the optimization of Example 1.

Table 2: Comparison of the best results of three algorithms for optimizing example 2 (unit: λ).

Element spacing	MGA	MQPSO	MQWPA
$d_1 - d_0$	0.5024	0.5000	0.5000
$d_2 - d_1$	0.5000	0.5000	0.5000
$d_3 - d_2$	0.5000	0.5000	0.5000
$d_4 - d_3$	0.5008	0.5000	0.5000
$d_5 - d_4$	0.5003	0.5000	0.5000
$d_6 - d_5$	0.5001	0.5000	0.5000
$d_7 - d_6$	0.5045	0.5000	0.5000
$d_8 - d_7$	0.5703	0.5000	0.5000
$d_9 - d_8$	0.5369	0.5210	0.5000
$d_{10} - d_9$	0.5194	0.6451	0.5000
$d_{11} - d_{10}$	0.5868	0.6671	0.5000
$d_{12} - d_{11}$	0.5765	0.6870	0.5233
$d_{13} - d_{12}$	0.7737	0.7143	0.7436
$d_{14} - d_{13}$	0.7045	0.7230	0.8092
$d_{15} - d_{14}$	1.0065	0.8330	0.8403
$d_{16} - d_{15}$	0.8806	0.7451	0.9130
$d_{17} - d_{16}$	0.8293	0.7492	0.9635
$d_{18} - d_{17}$	0.5054	0.5110	0.5002
PSLL	−20.5620 dB	−20.7781 dB	−21.2134 dB
Operation hours	1214 s	901 s	540 s

optimization accuracy is improved. However, if w continues to increase (greater than 500), the decision distance d_{near} will become too small. It makes it difficult for the fierce wolves to turn into the siege behavior, and the number of iterations increases and the optimization precision decreases. When w increases to a certain degree ($\omega = 1000$), only a small number of fierce wolves turn into the siege. Because of the lack of a fine search for a good solution domain, the optimization effect is degraded, resulting in multiple poor results in 100 optimization calculations.

Step size factor Sreflects the fineness of wolf search in solution space. The search fineness increases with the increase of S. From the point of convergence accuracy, when the value of S is [100, 800], increasing S is beneficial to improve the convergence accuracy of the algorithm. When S is greater than 800, as S continues to increase, the step size of wolves moving in the solution domain will become too small. The algorithm cannot traverse the good solution domain effectively, which results in a decrease of convergence accuracy and the increase of the PSLL for the array.

Through the above analysis, it can be seen that the MQWPA has a wide range of adaptability to parameters w and S in pattern synthesis. The final performance of the MQWPA will not be greatly affected by the slight change of parameters, which shows that the parameter selection of the MQWPA is not difficult and the algorithm has good robustness. In order to get the best antenna pattern synthesis effect, we take $w = 500$ and $S = 800$ in the following examples.

4.2. Example 2. The number of elements is 37. For a linear array with symmetrical center, the aperture is 21.996λ. All elements are omnidirectional with equal amplitude, and the beam direction is 0°. It is required to synthesize the array, keep the aperture unchanged, the distance between adjacent elements is greater than half wavelength, and the side-lobe level of the pattern is the lowest.

Because of the symmetrical distribution, there are 18 elements in the half-aperture. The MQWPA is used to optimize $\omega = 500$ and $S = 800$, and the remaining parameters are consistent with Example 1. Ten independent random simulation experiments were also carried out, and the best results of 10 optimizations for the three algorithms were shown in Table 2. The convergence curves of fitness functions in the optimization process of the three algorithms are shown in Figures 8–10.

In the 10 independent experiments optimized by the MQWPA, the best PSLL is −21.2134 dB, the average PSLL is −21.1140 dB, and the worst PSLL is −21.0591 dB. It can be seen that the 10 arrays obtained by the MQWPA are better than those optimized by the MGA and MQPSO.

From Figure 10, it can be seen that the best value of PSLL obtained from 10 MQWPA simulation tests is only 0.1543 dB lower than the worst value. However, the best PSLL of MQPSO 10 simulation tests is 0.8512 dB lower than the worst value and the best PSLL of MGA10 simulation tests is 0.3871 dB lower than the worst value. Obviously, the MQWPA has the best numerical stability. In addition, the MGA converges around 100 generations in the optimization process and the MQPSO algorithm converges around 60 generations. The proposed MQWPA has converged in about 40 generations. The MQWPA takes 540 seconds, far less than the other two algorithms. It is obvious that, among the three algorithms, the MQWPA can not only optimize the lowest PSLL but also has the best convergence speed. It has a good advantage in the pattern synthesis of multiconstrained sparse linear array.

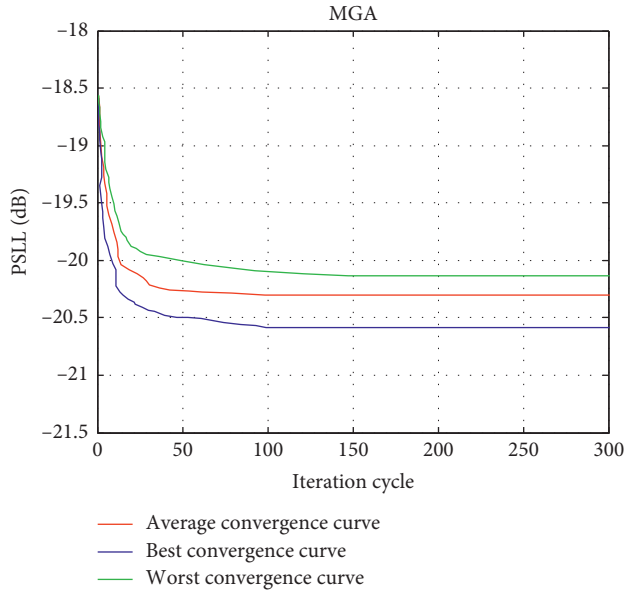


Figure 8: Convergence curve of fitness function. MGA for Example 2 optimization.

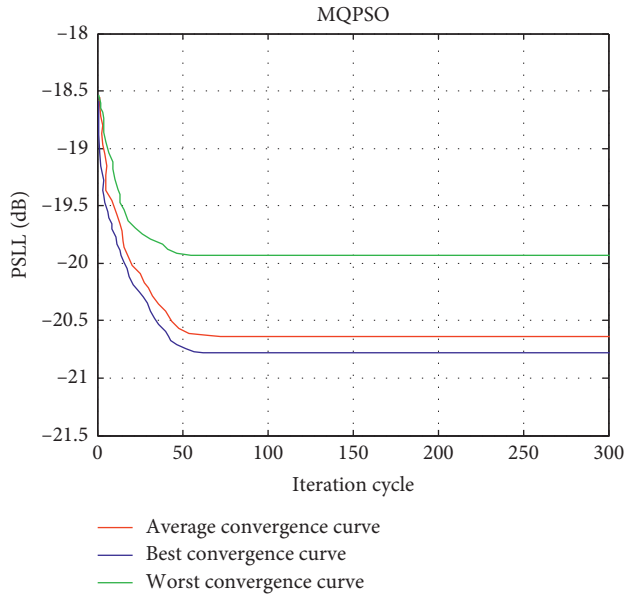


Figure 9: Convergence curve of fitness function. MQPSO for Example 2 optimization.

Through the simulation of the above typical examples, we find that the MQWPA can synthesize the multiconstrained sparse array robustly and efficiently.

The coding method based on Bloch spherical coordinates greatly expands the number of global optimal solutions and improves the probability of obtaining global optimal solutions. Selective mutation enhances the robustness of the algorithm and improves the search efficiency.

At the same time, because the size of each dimension of Bloch spherical coordinates is always [−1, 1], the variables transformed by solution space must satisfy the constraints of aperture and minimum element spacing. It effectively avoids

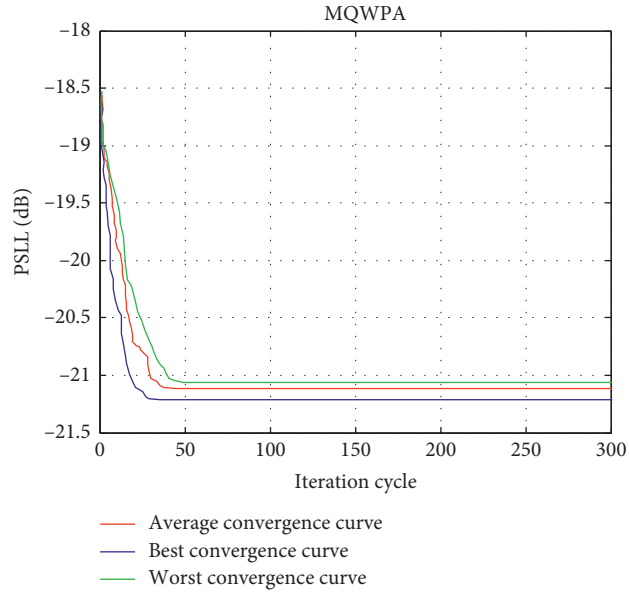


Figure 10: Convergence curve of fitness function. MQWPA for Example 2 optimization.

the infeasible solution in the algorithm iteration, reduces the judgment steps, and improves the optimization speed. Therefore, compared with the MGA in [17] and MQPSO in [26], the algorithm proposed in this paper has better accuracy and speed in pattern synthesis of multiconstrained sparse linear arrays.

5. Conclusion

In this paper, the MQWPA based on quantum Bloch spherical coordinates is proposed for the multiconstrained optimization of sparse linear arrays (including the constraints of the number of elements, the aperture of arrays, and the minimum distance between adjacent elements). Multiconstrained sparse array synthesis is not only a nonlinear optimization problem but also a multiconstrained problem. Constraint condition judgment is needed for each iteration update. It takes a lot of time and is difficult to be optimized. The MQWPA adopts a new encoding method, which not only expands the number of global optimal solutions and improves the probability of obtaining global optimal solutions but also effectively avoids the occurrence of infeasible solutions in iteration, reduces the steps of judging multiconstraint conditions, and improves the optimization efficiency. Several independent simulation experiments show that the MQWPA performs well in solving the multiconstrained optimization problem of sparse linear arrays. It not only has high convergence accuracy and fast convergence speed but also has good numerical stability.

In the future, we can proceed from two aspects. One is to further analyze the intelligent search strategy, mutation operator, and range of important parameters of the WPA and continue to tap the optimization potential of the algorithm. On the other hand, the MQWPA is applied to more antenna designs, such as circular array, plane array, and conformal array. In short, the new algorithm has a good

ability to improve continually, as well as a wide range of applications, and has a good value of promotion.

Data Availability

The data used to support the findings of this study are included within the article. For the simulation environment, we use the following platforms: the microcomputer type Lenovo G460, Intel (R) Core (™) i3-380-m 2.53 GHz CPU, 2.00 GB memory, 500 GB hard drive. For the software platform, we use Windows 7 (32 bit operating system) and the simulation software MATLAB 2012. If there are other data problems, you can contact the corresponding author.

Conflicts of Interest

The authors declare that there are no conflicts of interests regarding the publication of this article.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (no. U1813222), Tianjin Natural Science Foundation (no. 18JCYBJC16500), and Key Research and Development Project from Hebei Province (no. 19210404D).

References

[1] H. Hu, J. Du, C. Ye, and X. Li, "Ultrasonic phased array sparse-TFM imaging based on sparse array optimization and new edge-directed interpolation," *Sensors*, vol. 18, no. 6, p. 1830, 2018.

[2] D. Dai, M. Yao, H. Ma, W. Jin, and F. Zhang, "An asymmetric mapping method for the synthesis of sparse planar arrays," *IEEE Antennas and Wireless Propagation Letters*, vol. 17, no. 1, pp. 70–73, 2018.

[3] L.-J. Li, B.-H. Wang, and C.-H. Xia, "Synthesis of sparse conformal array antennas pattern," *Acta Electronica Sinica*, vol. 45, pp. 104–111, 2017.

[4] K. 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, 2015.

[5] Y. Ma, S. Yang, Y. Chen, S.-W. Qu, and J. Hu, "Pattern synthesis of four dimension irregular antenna arrays based on maximum entropy model," *IEEE Transactions on Antennas and Propagation*, vol. 67, no. 5, pp. 3048–3057, 2019.

[6] X. Fan, J. Liang, and H. C. So, "Beampattern synthesis with minimal dynamic range ratio," *Signal Processing*, vol. 152, pp. 411–416, 2018.

[7] X. Zhang, Z. He, B. Liao, X. Zhang, and W. Peng, "Pattern synthesis for arbitrary arrays via weight vector orthogonal decomposition," *IEEE Transactions on Signal Processing*, vol. 66, no. 5, pp. 1286–1299, 2017.

[8] S. R. Wu, "A versatile pattern synthesis algorithm for circular antenna array," *International Journal of RF & Microwave Computer-Aided Engineering*, vol. 6, Article ID e21254, , 2018.

[9] Z. Wang, Y. Song, T. Mu, and Z. Ahmad, "A short-range range-angle dependent beampattern synthesis by frequency diverse array," *IEEE Access*, vol. 6, no. 99, pp. 22664–22669, 2018.

[10] T. Wang, K.-W. Xia, and N. Lu, "Pattern synthesis for sparse arrays by compressed sensing and low-rank matrix recovery methods," *International Journal of Antennas and Propagation*, vol. 2018, Article ID 6403269, 11 pages, 2018.

[11] M. Skolnik, G. Nemhauser, and J. Sherman, "Dynamic programming applied to unequally spaced arrays," *IEEE Transactions on Antennas and Propagation*, vol. 12, no. 1, pp. 35–43, 1964.

[12] B. P. Kumar and G. R. Branner, "Design of unequally spaced arrays for performance improvement," *IEEE Transactions on Antennas and Propagation*, vol. 47, no. 3, pp. 511–523, 1999.

[13] V. Murino, A. Trucco, and C. S. Regazzoni, "Synthesis of unequally spaced arrays by simulated annealing," *IEEE Transactions on Signal Processing*, vol. 44, no. 1, pp. 119–122, 1996.

[14] J. W. Hooker and R. K. Arora, "Optimal thinning levels in linear arrays," *IEEE Antennas and Wireless Propagation Letters*, vol. 9, no. 1, pp. 771–774, 2010.

[15] T. H. Ismail and Z. M. Hamici, "Array pattern synthesis using digital phase control by quantized particle swarm optimization," *IEEE Transactions on Antennas and Propagation*, vol. 58, no. 6, pp. 2142–2145, 2010.

[16] Y.-J. Won, K. H. Lee, and J.-H. Lee, "Performance improvement of spaceborne SAR using antenna pattern synthesis based on quantum-behaved particle swarm optimization," *International Journal of Antennas and Propagation*, vol. 2017, Article ID 6928970, 12 pages, 2017.

[17] K. Chen, Z. He, and C. 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.

[18] J. Weimin, L. Zhiqiang, Y. Minli, Z. Peng, and Z. Jianxun, "A method of antenna synthesis for multi-constrained sparse array," *Journal of Electronics*, vol. 41, no. 5, pp. 926–930, 2013.

[19] D. F. Li and Z. L. gong, "Application of genetic algorithms in the pattern synthesis of ultra-low sidelobe linear array antenna," *Acta Electronica Sinica*, vol. 31, no. 1, pp. 82–84, 2013.

[20] W.-P Liao and F.-L Chu, "Array pattern synthesis with null steering using genetic algorithms by controlling only the current amplitudes," *International Journal of Electronics*, vol. 86, no. 4, pp. 445–457, 1999.

[21] A. Lommi, A. Massa, E. Storti, and A. Trucco, "Sidelobe reduction in sparse linear arrays by genetic algorithms," *Microwave and Optical Technology Letters*, vol. 32, no. 3, pp. 194–196, 2002.

[22] C. Yang, X. Tu, and J. Chen, "Algorithm of marriage in honey bees optimization based on the wolf pack search," in *Proceedings of the International Conference on Intelligent Pervasive Computing IEEE Computer Society*, Seattle, WA, USA, July 2007.

[23] J. Wang, Y. Jia, and Q. Y. Xiao, "Wolf pack algorithm optimizes at the water station application in scheduling," *Water Technologies, Science and Technology*, vol. 35, no. 3, pp. 1–4, 2015.

[24] J. Shan and J. Zhigang, "Multi target convergence node coverage algorithm based on quantum Wolf gang evolution," *Journal of Electronics and Information*, vol. 39, no. 5, pp. 1178–1184, 2017.

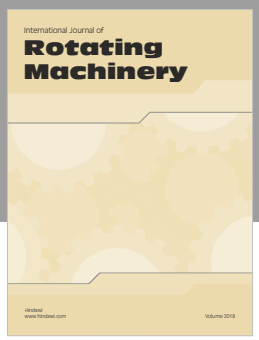
[25] P. Li and S. Li, "Quantum-inspired evolutionary algorithm for continuous space optimization based on Bloch coordinates of qubits," *Neurocomputing*, vol. 72, no. 1–3, pp. 581–591, 2008.

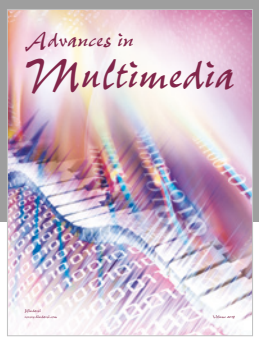
[26] O. U. Rehman, S. U. Rehman, S. Tu et al., "A quantum particle swarm optimization method with fitness selection

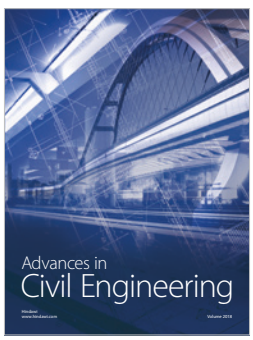
methodology for electromagnetic inverse problems," *IEEE Access*, vol. 6, pp. 63155–63163, 2018.

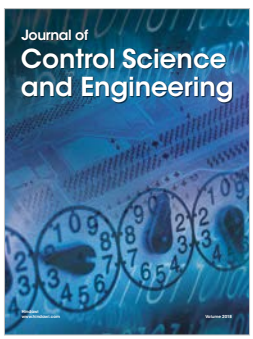
[27] H. Gao and R. Zhang, "Improved real-coded quantum evolutionary algorithms and its application on parameter estimation," *Kongzhi yu Juece/Control and Decision*, vol. 26, no. 3, pp. 418–422, 2011.

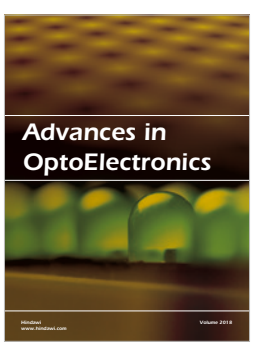
[28] O. Ur Rehman, S. Tu, S. Khan, H. Khan, and S. Yang, "A modified quantum particle swarm optimizer applied to optimization design of electromagnetic devices," *International Journal of Applied Electromagnetics and Mechanics*, vol. 58, no. 3, pp. 347–357, 2018.

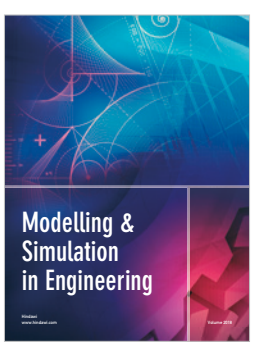


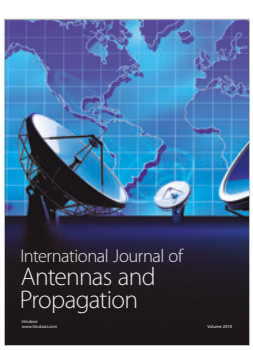


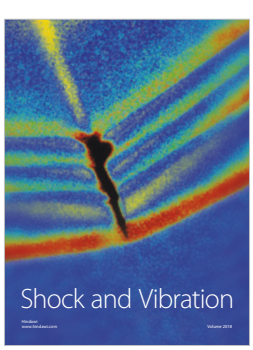


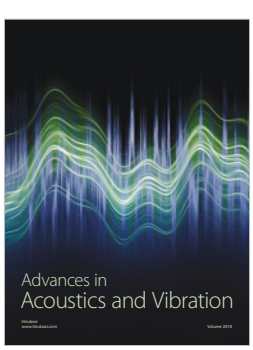












Hindawi

Submit your manuscripts at
www.hindawi.com