applied sciences

MDPI

Article

A Phase Angle-Modulated Bat Algorithm with Application to Antenna Topology Optimization

Jian Dong [1], Zhiyu Wang [1] and Jinjun Mo [2,*]

[1] School of Computer Science and Engineering, Central South University, Changsha 410083, China; dongjian@csu.edu.cn (J.D.); zhiyu_1997@csu.edu.cn (Z.W.)
[2] School of Information and Communication, Guilin University of Electronic Technology, Guilin 541004, China
* Correspondence: jinjunmo@guet.edu.cn

Abstract: This paper proposes a phase angle-modulated bat algorithm (P-AMBA) for high-dimensional binary optimization. The idea was to reduce the optimization time by introducing angle modulation technology to reduce the optimization dimensions. Different from the original angle-modulated bat algorithm (AMBA), the control of the trigonometric generating function cosine wave is by introducing new parameters, thereby improving the perturbation ability of the function curve near the *x*-axis. P-AMBA can explore more 0/1 solutions, and it has advantages in optimizing convergence speed and global search capabilities. The numerical results of the 0–1 knapsack problem tests show that P-AMBA is superior to the contrast algorithms on optimization ability and optimization time. Finally, the experimental result of a compact dual-band planar monopole antenna design showed the effectiveness of P-AMBA in engineering applications.

Keywords: bat algorithm; binary optimization; angle modulation; P-AMBA; high-dimensional optimization

check for updates

Citation: Dong, J.; Wang, Z.; Mo, J. A Phase Angle-Modulated Bat Algorithm with Application to Antenna Topology Optimization. *Appl. Sci.* **2021**, *11*, 2243. https://doi.org/10.3390/app11052243

Academic Editor: Emanuel Guariglia

Received: 11 January 2021
Accepted: 25 February 2021
Published: 3 March 2021

1. Introduction

With the rapid development of modern society, big data have become a research hotspot and a large number of data sets need to be optimized. High-dimensional optimization problems exist in many application scenarios, such as complex neural network weight optimization [1], text mining and analysis [2], sensor networking [3], vehicle routing [4], and antenna topology optimization [5]. To solve such optimization problems, meta-heuristic algorithms have been proposed as a reliable solution [6]. As an improvement of heuristic algorithms, meta-heuristic algorithms are the combination of random algorithms and local search algorithms. Meta-heuristic algorithms can find the optimal solution to the problem under limited time and space conditions. A large number of meta-heuristic algorithms have been proposed including the genetic algorithm (GA) [7], particle swarm optimization (PSO) [8], differential evolution (DE) [9], artificial bee colony (ABC) [10], the dragonfly algorithm (DA) [11], the fireworks algorithm (FA) [12], brain storm optimization (BSO) [13], harmony search (HS) [14], moth search (MS) [15], and the bat algorithm (BA) [16].

The BA, proposed by Yang in 2010, is a meta-heuristic algorithm developed according to the principle of bat echolocation [16]. Compared with other meta-heuristic algorithms, BA has two characteristics: (i) frequency tuning can facilitate the control of the optimization direction of the algorithm, and (ii) dynamic conversion of global search and local search can effectively avoid premature convergence of the algorithm. The advantages of BA are its simplicity, fewer parameters, robustness, and ease of implementation. Due to its superiority, BA has been applied in various fields, such as wireless sensor network deployment in the 3D environment [17], data clustering [18], the low-carbon job shop scheduling problem [19], optimization of reactive power dispatch [20], and the blood vessel segmentation problem [21]. BA was first proposed to address the optimization problem of the continuous real search space. In 2014, Yang proposed a binary bat algorithm (BBA) [22] for discrete optimization problems. BBA maps continuous variables to discrete variables by introducing a transfer function. Similar operations are used in the binary particle swarm optimization algorithm (BPSO) [23] and the binary dragonfly algorithm (BDA) [11]. However, as the problem dimension increases, the BBA exploration capacity may become poor, and the global search capability will decrease [24]. Some researchers have also made improvements for solving the problem of insufficient optimization performance of traditional binary optimization algorithms. Xue et al. proposed a binary fireworks algorithm (BFWA) [25] by introducing firework explosion, mutation explosion, and the selection of locations and showed its better convergence compared with GA, BPSO, and the binary cuckoo search algorithm (BCS). In [26], a binary multi-scale quantum harmonic oscillator algorithm (BMQHOA) with a genetic operator was proposed. The framework of BMQHOA consists of three nested phases including energy level stabilization, energy level decline, and scale adjustment, and it is superior to the BBA, BDA, and BPSO on computational accuracy, convergence capability, and stability. In 2015, Zakaria et al. proposed an angle-modulated bat algorithm called AMBA [27]. The introduction of the angle modulation technology can achieve the rapid optimization and convergence by reducing the dimensions of variables. Furthermore, to improve the optimization ability of AMBA, some researchers proposed the amplitude angle-modulated bat algorithm (A-AMBA) [28] by adding one more parameter to control the amplitude of the sine wave in the generating function.

Recently, meta-heuristic algorithms have found their applications in antenna designs because of their good global optimization ability. The traditional antenna optimization by meta-heuristic algorithms is to firstly determine the shape of the antenna based on empirical research and then to only optimize the dimensions of the given geometric parameters. This method not only highly depends on antenna designers' experiences but is also not capable of designing more unexpected antenna geometries. In this paper, the topology optimization method was used in small antenna designs, which are necessary for wearable or portable devices [29,30]. Antenna topology optimization first appeared in [31] for creating more unexpected resonant structures based on GA. The antenna topology optimization method works by discretizing the design space into several pixels, and each pixel is filled with air or conductors. This feature enables pixels to make full use of the space in a small volume, which is also a key reason why it can be used in antenna miniaturization design [32,33]. In designing radio frequency identification (RFID) tag antennas [34], microstrip patch antennas [8], and Multi-input Multi-output (MIMO) antennas [35], researchers have shown great interest in this method. However, antenna topology optimization often faces the problem of the "dimensionality curse", i.e., the design space exponentially expands with the number of design variables (i.e., the number of pixels).

Inspired by AMBA, this paper proposes a new binary bat algorithm, named phase angle-modulated bat algorithm (P-AMBA), to solve high-dimensional binary optimization problems. By applying a six-dimensional trigonometric function, the proposed algorithm can greatly reduce the dimension of optimization variables and achieve a better convergence speed and global search capability than the basic BBA and AMBA. Herein, numerical tests of the knapsack problem and a design example of a compact dual-band planar monopole antenna topology optimization verify the superiority of P-AMBA in convergence speed and optimization ability.

The rest of this paper is arranged as follows. The standard BA and BBA are elaborated in Section 2. In Section 3, the AMBA and P-AMBA are explained in detail. Section 4 uses 12 0–1 knapsack problems to carry out numerical verification experiments and a design example of a compact dual-band planar monopole antenna to demonstrate the effectiveness of the proposed P-AMBA. Finally, Section 5 concludes the paper and provides suggestions for future studies.

2. Background

2.1. The Bat Algorithm

The BA is a heuristic search algorithm that was proposed by Yang in 2010 based on swarm intelligence [16]. It is an effective method to search for the global optimal solution. It simulates bats' behavior of using ultrasound to detect and locate obstacles or prey, as shown in Figure 1. The bionic principle of BA is to map the population of individual bats to the *n*-dimensional problem space and then to simulate the optimization process. The fitness function value of the solution is used to measure the advantages and disadvantages of the bat's position. The survival process of the most suitable individual is similar to the iterative process of replacing the poorer feasible solutions with better feasible solutions in the optimization and search process. Compared with other algorithms, BA is superior in terms of accuracy and effectiveness, and it does not have many parameters that need to be adjusted.

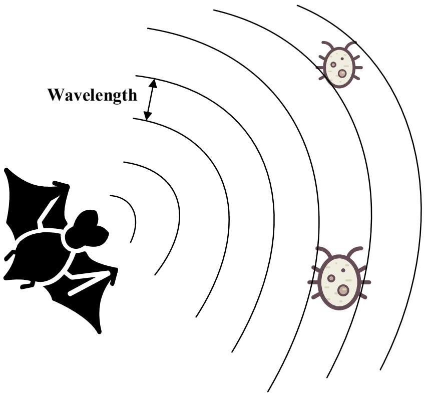


Figure 1. Schematic diagram of the bat's preying behavior.

In the BA, the optimization parameter is *n*-dimensional, and the number of bat populations is *n*. Each artificial bat has a frequency vector $X_i^k(i = 1,2,3,\ldots,n, k = 1,2,3,\ldots,N)$, a velocity vector $V_i^k(i = 1,2,3,\ldots,n, k = 1,2,3,\ldots,N)$, and a position vector $F_i(i = 1,2,3,\ldots,n)$, which are updated at iteration *t* as Equations (1)–(3):

$$V_i{}^k(t+1) = V_i{}^k(t) + (X_i{}^k(t) - X_{gbest})F_i \tag{1}$$

$$X_i{}^k(t+1) = X_i{}^k(t) + V_i{}^k(t+1) \tag{2}$$

where $X_i^k(t)$ and $V_i^k(t)$ indicate the *k*-th dimension position and velocity of the *i*-th bat at iteration t, respectively. X_{gbest} is the best solution thus far, and F_i indicates the frequency of the *i*-th bat that is updated in each course of iterations as follows:

$$F_{\mathrm{i}} = F_{\min} + (F_{\max} - F_{\min})\beta \tag{3}$$

where β is the random number uniformly distributed in [0,1]. According to Equations (1) and (2), different frequencies can improve the diversification of artificial bats and reach the optimal solution.

To improve the local search capability, each bat has a local search strategy. When $rand > r$, the new position $X_i^k(t)_{new}$ changes with a random walk:

$$X_i^k(t)_{new} = X_i^k(t)_{old} + \varepsilon A_i(t) \tag{4}$$

where $\varepsilon \in [-1,1]$, $rand \in [0,1]$, r is pulse emission rate, and $A_i(t)$ is the loudness value of all particles at iteration t.

As a result of the introduction of pulse emission rate (r) and loudness (A), the bat algorithm can be seen as a balance between particle swarm search and intensive local search. With the update of the two parameters, the optimization enters the exploration stage from the exploitation state. For each iteration of the algorithm, the pulse emission rate (r) and loudness (A) are updated as follows:

$$A_i(t+1) = \alpha A_i(t) \tag{5}$$

$$r_i(t+1) = r_i(0)[1 - \exp(-\gamma t)] \tag{6}$$

where α and γ are constants, with α being similar to the cooling factor in the simulated annealing (SA) [36]. The loudness and pulse emission rate are updated in each iteration until the artificial bat finds the optimal solution or reaches the maximum number of iterations.

For any $\alpha > 0; \gamma < 1$:

$$A_i(t) \to 0\,,\ r_i(t) \to r_i(0)\,,\ \text{as } t \to \infty \tag{7}$$

The pseudo-code of the bat algorithm is shown in Algorithm 1:

Algorithm 1 Pseudo-code of bat algorithm

```
Initialize artificial bat population $X_i$ ($i$ = 1, 2, …, $n$) and $V_i$;
Define pulse frequency $F_i$
Initialize pulse rate $r_i$ and the loudness $A_i$
While ($t$ < Max iteration)
  Generate new solutions by adjusting frequency, updating velocities and positions
[Equations (1) to (3)]
if ($rand > r_i$) then
      Select a solution among the best solutions randomly
      Generate a local solution around the selected best solution
end if
Generate a new solution by flying randomly;
if ($rand < A_i$ &$f(x_i) < f(X_{gbest})$) then
Accept the new solutions
Increase $r_i$ and reduce $A_i$
end if
Rank the bats and find the current $X_{gbest}$;
end while
```

2.2. Binary Bat Algorithm

The emergence of the bat algorithm was as a means to solve the continuous numerical optimization problem and to find the necessary 0/1 string in the feature selection problem. Additionally, the BBA introduces the transfer function to change the position update equation [22]. The velocity and frequency of the BBA are defined in continuous space. Compared with BA, BBA was additionally modified as follows:

- A transformation function was added to map a continuous position vector to a discrete bit vector.
- The random walk Equation (4) that improves the local search ability was deleted, as it is not suitable for the binary optimization algorithm.

In the BBA, using Equation (8) as the transfer function, the probability of the individual velocity of the BBA population is mapped to its position vector element flip. The greater the absolute value of the independent variable speed, the greater the dependent variable result, and the greater the probability of element flipping. The location update rules are as follows:

$$f(v_i^k(t)) = |\frac{2}{\pi}\arctan(\frac{2}{\pi}v_i^k(t))| \tag{8}$$

$$X_i^k(t+1) = \begin{cases} \left(X_i^k(t)\right)^{-1}, & rand \leq f(v_i^k(t+1)) \\ X_i^k(t), & rand > f(v_i^k(t+1)) \end{cases} \tag{9}$$

where $X_i^k(t)$ and $V_i^k(t)$ indicate the k-th dimension position and velocity of the i-th bat at iteration t, and $\left(X_i^k(t)\right)^{-1}$ indicates the complement of $X_i^k(t)$.

3. Phase Angle Modulated Bat Algorithm (P-AMBA)

3.1. AMBA

Even though traditional BBA can be optimized in the binary search space, the optimization time is longer, and the optimization ability is limited [24]. Thus, the idea of angle modulation was introduced. The AMBA is an improvement of the BBA [22], and its core idea is to introduce a trigonometric function as a bit vector generator to map the real-valued solutions into binary ones. This method was first used as the angle-modulated (AM) technique for communication signal processing [37–39].

This angle modulation idea uses a compound *sin*/*cos* generating function:

$$g(x) = sin(2\pi(x-a) \times b \times cos(2\pi(x-a) \times c)) + d \tag{10}$$

where x is a sequence of equal intervals, the number of which needs to be determined by the number of optimized dimensions, the coefficient a represents the horizontal shift of the function; b represents the maximum frequency of the sine wave; c represents the frequency of the cosine wave (which changes the rate at which the frequency of the sine wave changes); and d represents the vertical shift of the function. The default value of the four parameters in the generating function is a = 0, b = 1, c = 0.5, and d = 0, and a function diagram is shown in Figure 2.

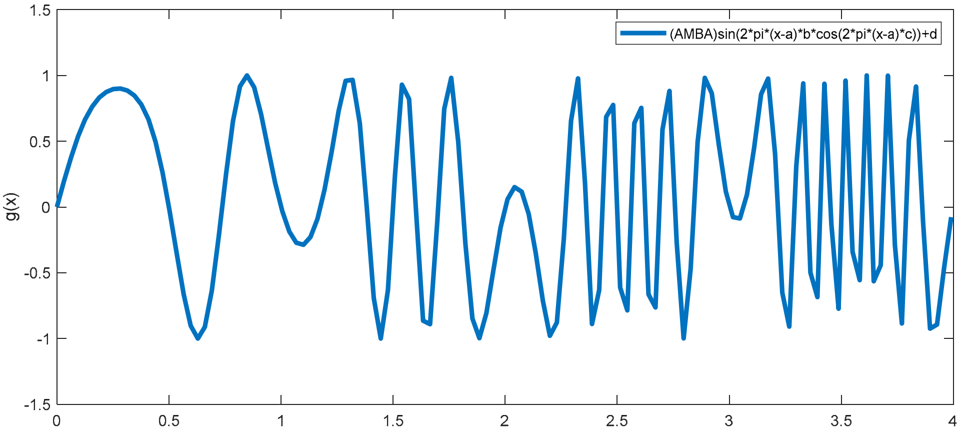


Figure 2. Generating function plotted with default values (a = 0, b = 1, c = 0.5, and d = 0).

After introducing the angle modulation strategy, a high-dimensional binary search space can be obtained by optimizing the four-dimensional continuous search space. The updated range of the four variables a, b, c, and d is as follows:

$$X = (a, b, c, d) \in [-1, 1]. \tag{11}$$

The change from a high-dimensional search space to four-dimensional search space greatly reduces the optimization time under the premise of ensuring optimization accuracy.

3.2. P-AMBA

In AMBA, after each iteration of the four-dimensional parameters, the richness of the discrete search space mapped by the trigonometric generating function is not enough to meet the optimization requirements. To maximize the impact of the parameter changes on the generating function, the P-AMBA proposed in this paper adds a phase adjustment to the cosine wave and an amplitude adjustment to the sine wave. P-AMBA can increase the global search ability of the optimization process to a certain extent. The generating function is described as follows:

$$g(x) = e \times sin(2\pi(x-a) \times b \times cos(2\pi(x-a) \times c) + g) + d. \tag{12}$$

When x takes the number of equal intervals from 1 to n dimensions, the function g(x) will generate the corresponding value. By converting $g(x)(x = 0, 1, 2, \cdots, n)$ into a 0/1 bit string through the conversion of Equation (12), we can achieve a mapping relationship. The rule of the conversion function is to take 0 when $g(x) \leq 0$ and to take 1 when $g(x) > 0$.

$$g(x) = \begin{cases} 0 & g(x) \leq 0 \\ 1 & g(x) > 0 \end{cases} \quad (x = 0, 1, 2, \cdots, n) \tag{13}$$

The original generating function lacks the control of the amplitude of the sine wave, so that the update of the vertical displacement d has less influence on the function curve. To make up for the lack of amplitude, the parameter e that controls the amplitude of the sine wave is added. Similarly, the cosine wave lacks phase adjustment. Adding a parameter g after the cosine wave can increase the fluctuation ability of the function near the zero points. The improved function adds two parameters, so the individual bat in the bat algorithm is six-dimensional:

$$\mathrm{X} = (a, b, c, d, e, g) \in [-1, 1]. \tag{14}$$

A comparison of the figures for the generating function between P-AMBA and AMBA is represented in Figure 3. It can be seen from the figure that when a parameter e is added, the parameter d that controls the phase of the entire function has a greater impact on the up and down positions of the function. This can better control the position of the function and obtain the desired 0/1 solution set. At the same time, by adding the parameter g to control the phase of the cosine wave, the function is more complicated. Each iterative update of the value of the six-dimensional parameter will result in a greater change in the overall mapping relationship of the function, and it will further enrich the exploration of possible solutions. P-AMBA has more parameters than BBA and AMBA, which may increase the complexity of the algorithm in the optimization process. However, the operation of adding new parameters can significantly improve the optimization ability in the high-dimensional binary search space.

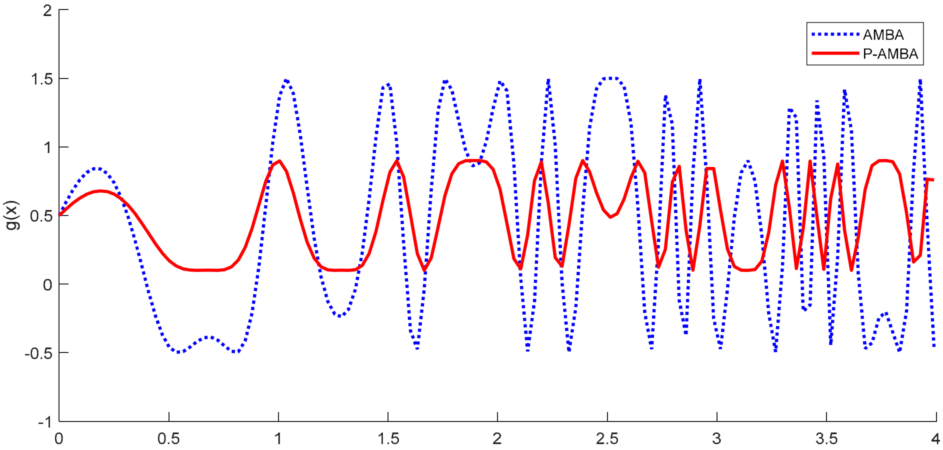


Figure 3. Generating function plotted with default values (a = 0, b = 0.5, c = 0.8, d = 0.5, e = 0.4, and g = 0.2).

After introducing the idea of angle modulation, the optimization process is essentially a continuous optimization process applied under binary demand. The key step is to substitute six parameters into the trigonometric generating function to generate an n-dimensional bit string and then to calculate the fitness function. The pseudo-code of the P-AMBA is shown in Algorithm 2. To better demonstrate the mapping process, the principle through which P-AMBA obtains discrete n-dimensional search space results by optimizing the continuous six-dimensional search space is shown in Figure 4.

Algorithm 2 Pseudo-code of P-AMBA

```
Initialize $a, b, c, d, e, g$, and $\boldsymbol{V}_i = 0$ in the continuous bat algorithm
Define pulse frequency $F_i$Initialize pulse rates $r_i$ and the loudness $A_i$
While (t < Max iteration)
    Generate new solutions by adjusting frequency, updating velocities and positions [Equations (1) to (3)]
    Calculate the output value $g(\boldsymbol{x})$ using Equation (12) to generate $n$-dimensional bit stringsUpdate the position vectors using Equation (13)
    if ($rand > r_i$)
        Select a solution among the best solutions randomly
        Change some of the dimensions of the position vector with some of the dimensions of $\boldsymbol{X}_{\text{gbest}}$
    end if
    Generate a new solution by flying randomly
    if ($rand < A_i$ & $f(\boldsymbol{x}_i) < f(\boldsymbol{X}_{gbest})$)
        Accept the new solutions
        Increase $r_i$ and reduce $A_i$
    end if
    Rank the bats and find the current $\boldsymbol{X}_{gbest}$;
end while
```

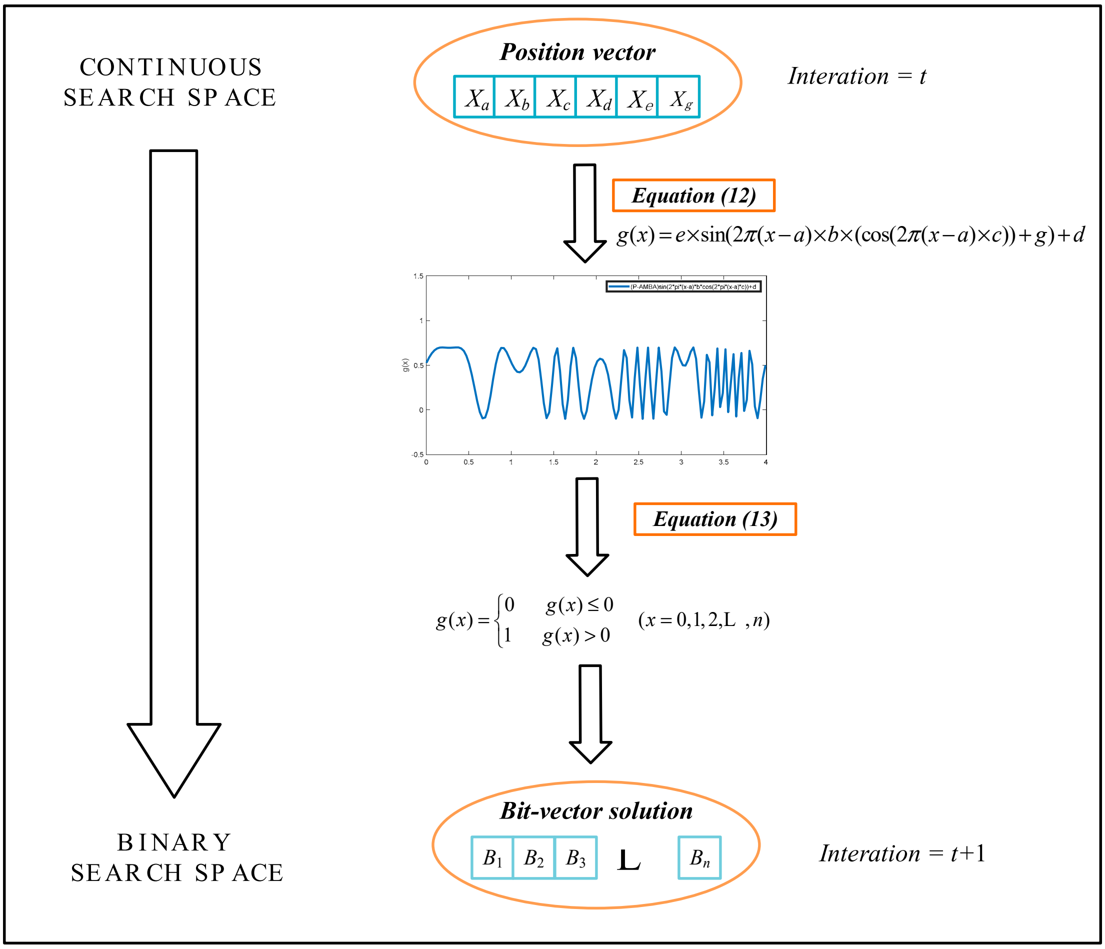

CONTINUOUS SEARCH SPACE
Position vector
X_a X_b X_c X_d X_e X_g
Interation = t
Equation (12)
$g(x)=e\times\sin(2\pi(x-a)\times b\times(\cos(2\pi(x-a)\times c))+g)+d$
(P-AMBA)sin(2*pi*(x-a)*b*cos(2*pi*(x-a)*c))+d
g(x)
1.5
1
0.5
0
-0.5
0
0.5
1
1.5
2
2.5
3
3.5
4
Equation (13)
$g(x)=\begin{cases}0 & g(x)\le 0\\ 1 & g(x)>0\end{cases}\quad (x=0,1,2,\text{L}\ \ ,n)$
Bit-vector solution
B_1 B_2 B_3 L B_n
Interation = t+1
BINARY SEARCH SPACE

Figure 4. The process of mapping continuous search space to discrete search space.

4. Experimental Results and Discussions

The processor used for the simulation experiment environment was an Intel(R) Core (TM) i5-8265U 1.80 GHz (with 8.0 GB RAM and a 64-bit Windows10 operating system), and the simulation software was Matlab 2016a and HFSS 15.

4.1. Zero-One Knapsack Problems

The 0–1 knapsack problem is a common and typical combinatorial optimization non-deterministic Polynomial complete (NP-C) problem, which is often used for numerical testing of binary optimization algorithms. The knapsack problem was first proposed by Dantzig in 1957 [40,41]. Classical 0–1 knapsack problems can be described as follows:

$$\begin{aligned} &\max f(p) = \sum_{i=1}^{N} p_i x_i \\ &s.t. = \begin{cases} f(w) = \sum\limits_{i=1}^{N} w_i x_i \leq C \\ x_i = 0, 1 \, (i = 1, 2, \cdots, N) \end{cases} \end{aligned} \tag{15}$$

where p_i and w_i represent the profit and weight of the individual items i, N is the number of individuals, and x represents the decision variable of the i-th item. If the item i is packed, x takes 1; otherwise, x takes 0. The backpack problem can be described as follows: On the premise of ensuring that the total weight $f(w)$ of the objects in the backpack does not exceed the maximum capacity C that the backpack can bear, we find the distribution strategy with the highest total value $f(p)$ of the items. The profit and weight of the items are positive.

In this paper, 12 different dimensional test examples of the knapsack problem were selected to compare P-AMBA with other optimization algorithms. The data of the classic knapsack problem test cases k1–k5 are shown in Table 1, where 'D' represents the dimension of the backpack, that is, how many items are there. 'Parameter (w, p, C)' represents the weight and value of the item and the maximum load capacity of the backpack, respectively, while 'Opt' is the optimal value for this instance. Meanwhile, cases k6–k12 are high-scale knapsack problems. The item data are generated by a random generator to compare the application of several optimization algorithms in a high-dimensional environment. The weight $w_i(i = 1, 2, \cdots, N) \in [5, 20]$, and the profit $p_i(i = 1, 2, \cdots, N) \in [50, 100]$ [42]. The relevant data of these randomly generated backpack items are shown in Table 2. The 'Total values' denotes the total values of all items. In the k6–k12 cases, the formula $\frac{3}{4}\sum_{i=1}^{N} w_i$ was used to calculate the total capacity of items in the backpack.

For the example test of the knapsack problem, the classic BPSO [23], the traditional BBA [22], the AMBA [27], and the A-AMBA [28] (the latter two also used the idea of angle modulation) were selected for comparison with the proposed P-AMBA. The relevant parameter settings of the optimization algorithms in the verification tests are shown in Table 3. The parameter range is set according to [28], which is suitable for the knapsack problem. In order to objectively compare the optimization capabilities of different optimization algorithms, the parameter settings of this paper are all initial settings. The population size was set to 30, and the number of iterations was set to 500. To avoid accidental deviations, the algorithm was run 30 times independently for each function. Table 4 shows the test results of the five optimization algorithms after the optimization, including the maximum, minimum, average, standard deviation, and optimization time. In the table, 'No.' and 'Alg.' represent the test examples and the optimization algorithm types of the knapsack problem, respectively. 'Best' represents the maximum value of the 30 population optimal values, which is the optimal value that the optimization algorithm can find in the optimization process. 'Worst' represents the worst value of the optimal value of the 30 populations. 'Mean' is the average optimal value of the 30 optimization results. 'Std' is the standard deviation, reflecting the degree of dispersion of the 30 optimization results. 'Time' is the average time consumed for each optimization, reflecting the optimization speed of the optimization algorithm. Moreover, the average value of the iterative data of the examples k7–k12 in 30 optimization tests was selected, and the iterative convergence of the five algorithms was compared, as shown in Figure 5.

Table 1. Five typical test cases of 0–1 knapsack problems.

No.	D	Parameter (*w*, *p*, *C*)	Opt

k_1	10	*w* = (95, 4, 60, 32, 23, 72, 80, 62, 65, 46);
		p = (55, 10, 47, 5, 4, 50, 8, 61, 85, 87);
		C = 269
			295

k_2	20	*w* = (92, 4, 43, 83, 84, 68, 92, 82, 6, 44, 32, 18, 56, 83, 25, 96, 70, 48, 14, 58);
		p = (44, 46, 90, 72, 91, 40, 75, 35, 8, 54, 78, 40, 77, 15, 61, 17, 75, 29, 75, 63);
		C = 878
			1024

k_3	50	*w* = (80, 82, 85, 70, 72, 70, 66, 50, 55, 25, 50, 55, 40, 48, 59, 32, 22, 60, 30, 32, 40, 38, 35, 32, 25, 28, 30, 22, 50, 30, 45, 30, 60, 50, 20, 65, 20, 25, 30, 10, 20, 25, 15, 10, 10, 10, 4, 4, 2, 1);
		p = (220, 208, 198, 192, 180, 180, 165, 162, 160, 158, 155, 130, 125, 122, 120, 118, 115, 110, 105, 101, 100, 100, 98, 96, 95, 90, 88, 82, 80, 77, 75, 73, 72, 70, 69, 66, 65, 63, 60, 58, 56, 50, 30, 20, 15, 10, 8, 5, 3, 1);
		C = 1000
			3103

k_4	80	*w* = (40, 27, 5, 21, 51, 16, 42, 18, 52, 28, 57, 34, 44, 43, 52, 55, 53, 42, 47, 56, 57, 44, 16, 2, 12, 9, 40, 23, 56, 3, 39,16, 54, 36, 52, 5, 53, 48, 23, 47, 41, 49, 22, 42, 10, 16, 53, 58, 40, 1, 43, 56, 40, 32, 44, 35, 37, 45, 52, 56, 40, 2, 23,49, 50, 26, 11, 35, 32, 34, 58, 6, 52, 26, 31, 23, 4, 52, 53, 19);
		p = (199, 194, 193, 191, 189, 178, 174, 169, 164, 164, 161, 158, 157, 154, 152, 152, 149, 142, 131, 125, 124, 124, 124, 122, 119, 116, 114, 113, 111, 110, 109, 100, 97, 94, 91, 82, 82, 81, 80, 80, 80, 79, 77, 76, 74, 72, 71, 70, 69,68, 65, 65, 61, 56, 55, 54, 53, 47, 47, 46, 41, 36, 34, 32, 32,30, 29, 29, 26, 25, 23, 22, 20, 11, 10, 9, 5, 4, 3, 1);
		C = 1173
			5183

k_5	100	*w* = (54, 95, 36, 18, 4, 71, 83, 16, 27, 84, 88, 45, 94, 64, 14, 80, 4, 23, 75, 36, 90, 20, 77, 32, 58, 6, 14, 86, 84, 59, 71, 21, 30, 22, 96, 49, 81, 48, 37, 28, 6, 84, 19, 55, 88, 38, 51, 52, 79, 55, 70, 53, 64, 99, 61, 86, 1, 64, 32, 60, 42, 45, 34, 22, 49, 37, 33, 1, 78, 43, 85, 24, 96, 32, 99, 57, 23, 8, 10, 74, 59, 89, 95, 40, 46, 65, 6, 89, 84, 83, 6, 19, 45, 59, 26, 13, 8, 26, 5, 9);
		p = (297, 295, 293, 292, 291, 289, 284, 284, 283, 283, 281, 280, 279, 277, 276, 275, 273,264, 260, 257, 250, 236, 236, 235, 235, 233, 232, 232, 228, 218, 217, 214, 211, 208, 205, 204, 203, 201, 196, 194, 193, 193, 192, 191, 190, 187, 187, 184, 184, 184, 181, 179, 176, 173, 172, 171, 160, 128, 123, 114, 113, 107, 105, 101, 100, 100, 99, 98, 97, 94, 94, 93, 91, 80, 74, 73, 72, 63, 63, 62, 61, 60, 56, 53, 52, 50, 48, 46, 40, 40, 35, 28, 22, 22, 18, 15, 12, 11, 6, 5);
		C = 3818
			15,170

Table 2. Randomly generating seven 0–1 knapsack problems.

No.	D	*C*	Total Values
k_6	200	1948.5	15,132
k_7	300	2793.5	22,498
k_8	500	4863.5	37,519
k_9	800	7440.5	59,791
k_{10}	1000	9543.5	75,603
k_{11}	1200	11,267	90,291
k_{12}	1500	14,335	111,466

Table 3. Initial parameters for phase angle-modulated bat algorithm (P-AMBA), amplitude angle-modulated bat algorithm (A-AMBA), angle-modulated bat algorithm (AMBA), binary bat algorithm (BBA), and binary partical swarm algorithm (BPSO) in the knapsack problem.

Algorithm	**Parameters**	**Value**
BPSO	Population size	30
	C_1, C_2	2
	W	Decrease linearly from 0.4 to 1.2
	Max iteration	500
	Max velocity	2
	Stopping criterion	Max iteration
BBA	Population size	30
	F_{min}	0
	F_{max}	2
	A	0.25
	r	0.5
	ε	$[-1, 1]$
	α	0.9
	γ	0.9
	Max iteration	500
	Stopping criterion	Max iteration
AMBA	Population size	30
	Artificial bat (a,b,c,d)	$[-1, 1]$
	F_{min}	0
	F_{max}	2
	A	0.25
	r	0.5
	ε	$[-1, 1]$
	α	0.9
	γ	0.9
	Max iteration	500
	Stopping criterion	Max iteration
A-AMBA	Population size	30
	Artificial bat (a,b,c,d,e)	$[-1, 1]$
	F_{min}	0
	F_{max}	2
	A	0.25
	r	0.5
	ε	$[-1, 1]$
	α	0.9
	γ	0.9
	Max iteration	500
	Stopping criterion	Max iteration
P-AMBA	Population size	30
	Artificial bat (a,b,c,d,e,g)	$[-1, 1]$
	F_{min}	0
	F_{max}	2
	A	0.25
	r	0.5
	ε	$[-1, 1]$
	α	0.9
	γ	0.9
	Max iteration	500
	Stopping criterion	Max iteration

Table 4. Comparison among five algorithms for solving 0–1 knapsack problems with different scales.

No.	Alg.	Best	Worst	Mean	std	Time	D/Opt
k_1	BPSO	**295.00**	**295.00**	**295.00**	**0.00**	**0.0105**	D = 10 Opt = 295
	BBA	**295.00**	**295.00**	**295.00**	**0.00**	0.0520	
	AMBA	**295.00**	**295.00**	**295.00**	**0.00**	0.0534	
	A-AMBA	**295.00**	**295.00**	**295.00**	**0.00**	0.0553	
	P-AMBA	**295.00**	**295.00**	**295.00**	**0.00**	0.0579	
k_2	BPSO	**1024.00**	860.00	949.15	42.48	**0.0149**	D = 20 Opt = 1024
	BBA	**1024.00**	**1013.00**	1021.45	**3.72**	0.0756	
	AMBA	**1024.00**	1010.00	1021.70	3.96	0.0654	
	A-AMBA	**1024.00**	995.00	1019.30	7.66	0.0672	
	P-AMBA	**1024.00**	1004.00	**1022.05**	5.60	0.0706	
k_3	BPSO	2854.00	2494.00	2699.60	92.71	**0.0335**	D = 50 Opt = 3103
	BBA	2976.00	2303.00	2827.60	158.28	0.1510	
	AMBA	2956.00	2840.00	2884.13	**29.81**	0.1062	
	A-AMBA	2961.00	2813.00	2882.07	30.75	0.1086	
	P-AMBA	**2989.00**	**2818.00**	**2885.40**	36.24	0.1115	
k_4	BPSO	3739.00	2786.00	3337.37	**252.55**	**0.0526**	D = 80 Opt = 5183
	BBA	4597.00	1349.72	3687.52	669.77	0.2205	
	AMBA	**4780.00**	3618.00	4295.47	406.05	0.1436	
	A-AMBA	**4780.00**	3734.00	4306.57	405.36	0.1483	
	P-AMBA	**4780.00**	**3768.00**	**4408.03**	425.51	0.1516	
k_5	BPSO	13,683.00	11,873.00	13,042.40	473.75	**0.0589**	D = 100
	BBA	14,867.00	12,406.00	13,113.93	551.59	0.2662	
	AMBA	14,869.00	13,564.00	14,501.40	509.06	0.1620	
	A-AMBA	14,869.00	13,595.00	14,439.67	531.16	0.1676	
	P-AMBA	**14,904.00**	**13,673.00**	**14,555.10**	**452.43**	0.1705	
k_6	BPSO	12,675.61	11,242.36	11,871.12	387.33	0.1138	D = 200
	BBA	12,797.70	11,406.46	11,974.79	332.13	0.5090	
	AMBA	12,873.84	11,215.95	12,085.26	379.63	0.2936	
	A-AMBA	12,806.88	11,409.21	12,100.93	374.24	0.2988	
	P-AMBA	**12,877.02**	**11,547.41**	**12,126.24**	**312.29**	0.3092	
k_7	BPSO	18,244.04	15,668.21	17,089.19	509.09	**0.2976**	D = 300
	BBA	17,887.51	16,364.22	17,082.09	415.63	0.7407	
	AMBA	17,814.18	16,651.69	17,284.45	**304.44**	0.4052	
	A-AMBA	18,005.78	16,585.54	17,280.36	332.41	0.4202	
	P-AMBA	**18,252.20**	**16,672.67**	**17,331.96**	368.11	0.4317	
k_8	BPSO	30,724.89	28,159.65	29,457.28	570.25	**0.4615**	D = 500
	BBA	30,503.68	28,795.94	29,564.48	**437.66**	1.2226	
	AMBA	30,873.78	28,933.24	29,822.73	514.35	0.6480	
	A-AMBA	30,787.96	28,877.36	29,776.48	459.65	0.6684	
	P-AMBA	**31,238.10**	**28,992.05**	**29,981.75**	687.97	0.6849	
k_9	BPSO	46,563.54	42,342.29	44,851.12	922.30	**0.4957**	D = 800
	BBA	46,289.64	42,371.40	44,867.11	778.18	1.9901	
	AMBA	46,589.03	44,034.71	45,488.08	658.94	1.0144	
	A-AMBA	47,204.08	**44,355.01**	45,419.85	**592.17**	1.0372	
	P-AMBA	**47,597.69**	43,959.02	**45,607.85**	736.12	1.0652	
k_{10}	BPSO	59,561.44	55,885.05	57,496.69	893.97	**0.6484**	D = 1000
	BBA	59,617.02	56,103.77	57,677.75	855.63	2.4772	
	AMBA	59,423.89	56,924.37	58,365.76	719.56	1.2627	
	A-AMBA	59,682.09	56,896.22	58,236.85	733.81	1.2869	
	P-AMBA	**59,867.14**	**57,408.02**	**58,409.60**	**660.54**	1.3196	

Table 4. *Cont.*

No.	Alg.	Best	Worst	Mean	std	Time	D/Opt
k_{11}	BPSO	69,899.29	65,823.05	67,998.33	960.01	**0.7627**	D = 1200
	BBA	70,880.25	65,925.82	68,010.64	931.82	2.9429	
	AMBA	70,142.52	66,377.51	68,827.49	815.61	1.5098	
	A-AMBA	70,001.73	67,426.10	68,670.86	**685.35**	1.5353	
	P-AMBA	**71,349.91**	**67,815.82**	**68,984.79**	870.12	1.5722	
k_{12}	BPSO	88,635.44	84,595.14	86,473.41	1108.52	**0.9212**	D = 1500
	BBA	89,195.16	84,320.84	86,486.67	1350.48	3.6654	
	AMBA	89,304.33	84,346.38	87,014.87	1138.13	1.8777	
	A-AMBA	88,575.85	**85,092.06**	86,678.28	**979.82**	1.9048	
	P-AMBA	**89,408.59**	84,120.74	**87,074.01**	1115.51	1.9664	

[1] Bold data is the optimal result.

The optimal, worst, and mean values in Table 4 show that the higher the dimensionality of the optimization problem, the more prominent the optimization ability of P-AMBA. By comparing the optimization time of the five algorithms, although the BPSO algorithm took less time, it did not perform as well as the other algorithms on high-dimensional problems. With the introduction of angle modulation technology, the time consumption of the bat algorithms was greatly reduced. In a high-dimensional environment, the time consumption of the P-AMBA algorithm was only half that of the traditional binary bat algorithm.

The comparison of the convergence curves in Figure 5 obviously shows that P-AMBA has a strong ability to optimize high-dimensional problems. After 500 iterations, a better solution set of optimal values was found by P-AMBA in comparison to the other optimization algorithms. Different from AMBA and A-AMBA, in P-AMBA, in addition to the effect of the amplitude amplifying the vertical displacement coefficient, new parameters were also introduced to adjust the phase of the cosine wave, thereby further affecting the disturbance of the function and adding more 0/1 combination possibilities.

4.2. Compact Dual-Band Planar Monopole Antenna Design

Antennas are widely used in mobile communications, radar, and satellite communications as they play an important role in the transmitting and receiving of signals. Since antenna performance determine the quality of a wireless communication system and different antenna configurations have different antenna performances, the optimization of the antenna configurations has attracted a lot of attention. Traditional antenna optimization starts from a given initial geometric structure and only adjusts the size of the relevant geometric parameters. However, this kind of optimization method is not conducive to exploring the possibility of novel antenna structures. The topology optimization used in this paper, also called pixel optimization, is a technique that can automatically generate an antenna topology suitable for the design requirements in a given design area. It divides the design space into several pixels represented by a matrix with “1” (conductor) and “0” (air). Through some stochastic optimization techniques, we can find the optimal pixel distribution mode (i.e., the optimal antenna topology) that satisfies the design requirements.

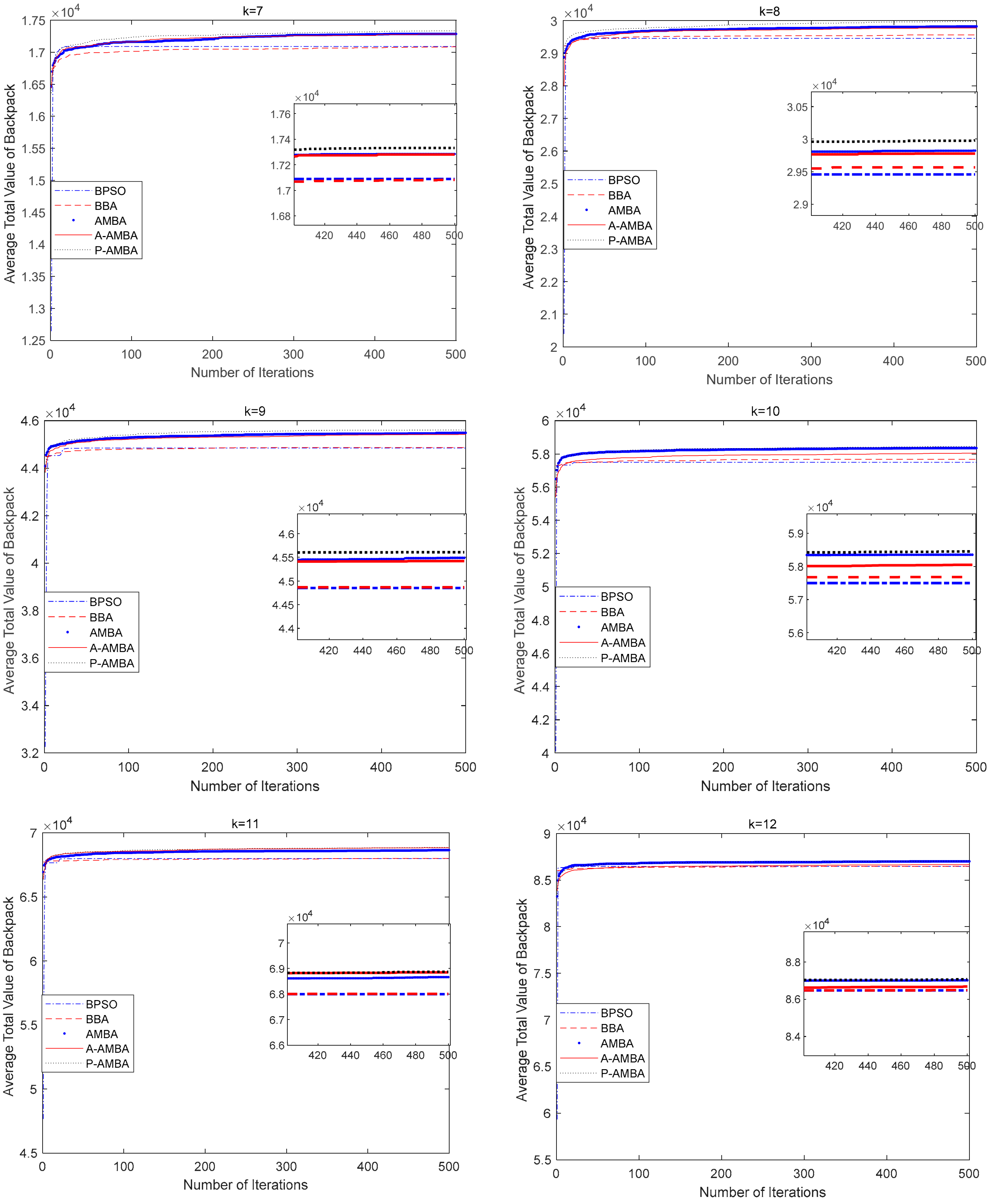


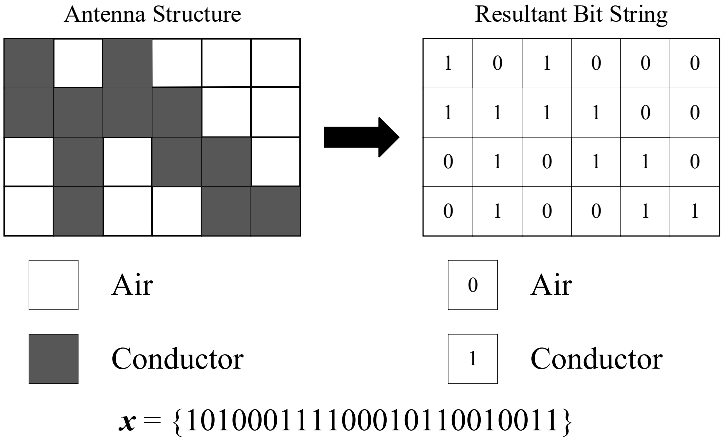
Figure 5. Average convergence curves of BPSO, BBA, AMBA, A-AMBA, and P-AMBA on some selected large-scale problems over 30 independent runs.

4.2.1. Antenna Topology Optimization Problem Formulation

In antenna topology optimization, the design area is divided into pixel blocks of the same size, and air and conductor are filled in each pixel block according to the binary mapping relationship. Therefore, the problem of antenna topology designs can be expressed as a binary optimization problem:

$$\begin{aligned} &find\ x^* = \arg\min_x f(x) \\ &s.t. x \in \{0,1\}^n \end{aligned} \tag{16}$$

where $\boldsymbol{x} = [x_1, x_2, \ldots, x_n]$ is an n-dimensional binary design variable, representing a binary code mapped from the specific antenna geometric structure; $\boldsymbol{x}^*$ is the obtained optimal antenna design result; $f(\boldsymbol{x})$ is the objective function of the antenna design, such as reflection coefficients, gain, or efficiency. Figure 6 shows the mapping process of the antenna structure. If the pixel is filled with the conductor, it is mapped as 1; otherwise, it is mapped as 0.



1	0	1	0	0	0
1	1	1	1	0	0
0	1	0	1	1	0
0	1	0	0	1	1

$\boldsymbol{x} = \{101000111100010110010011\}$

Figure 6. The relationship between the topology structure of the antenna and its corresponding binary code.

4.2.2. Antenna Topology Optimization Design

Multi-band antennas meet the needs of different communication standards with only one device. The advantage of a multi-band device is that it has adaptability and scalability, and the number of frequency bands can differ. Therefore, multi-band antennas are very popular in the field of antenna optimization designs.

To verify the effectiveness of the P-AMBA, herein, we present a design example of a dual-band planar monopole antenna and compare it with the BPSO [23], BBA [22], AMBA [27], and A-AMBA [28] optimization algorithms. The initial antenna geometry is given in Figure 7, where the left and right pictures are the top views and side views, respectively. It consists of a 14.6 × 12.8 mm rectangular patch and a 16.8 × 4 mm rectangular ground plane. The antenna is fed by a 50Ω microstrip line and printed on a 16.8 × 24.6 mm FR4 substrate with a 1.6 mm thickness, a 4.4 permittivity, and a 0.02 loss tangent. The antenna was designed to achieve the reflection coefficients lower than −10 dB (i.e., $S_{11} < -10$ dB) in the two frequency bands of 2.3−2.5 GHz and 5.1−5.9 GHz, covering the entire 2.4/5.2/5.8 GHz WLAN bands. The fitness function can be expressed as:

$$\begin{aligned} &Min\ F = \frac{1}{m}\sum_{i=1}^{m}(Q(f_i)) \\ &s.t.\ Q(f_i) = \begin{cases} -10, S_{11}(f_i) \le -10 \\ S_{11}(f_i), S_{11}(f_i) > -10 \end{cases} \end{aligned} \tag{17}$$

where f_i denotes the i th sampling frequencies within the given operating bands; $S_{11}(f_i)$ is the reflection coefficient of sample f_i; m is the total number of samples; and F is the fitness value.

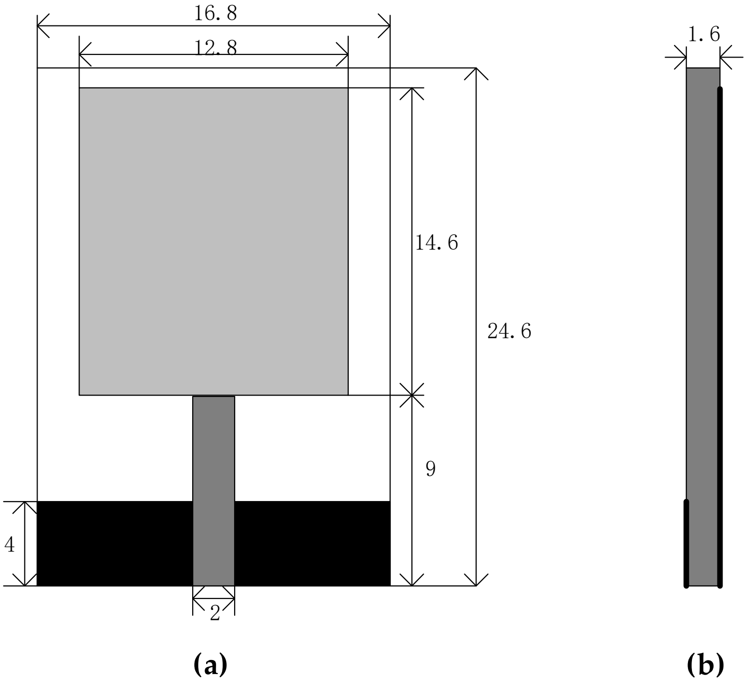
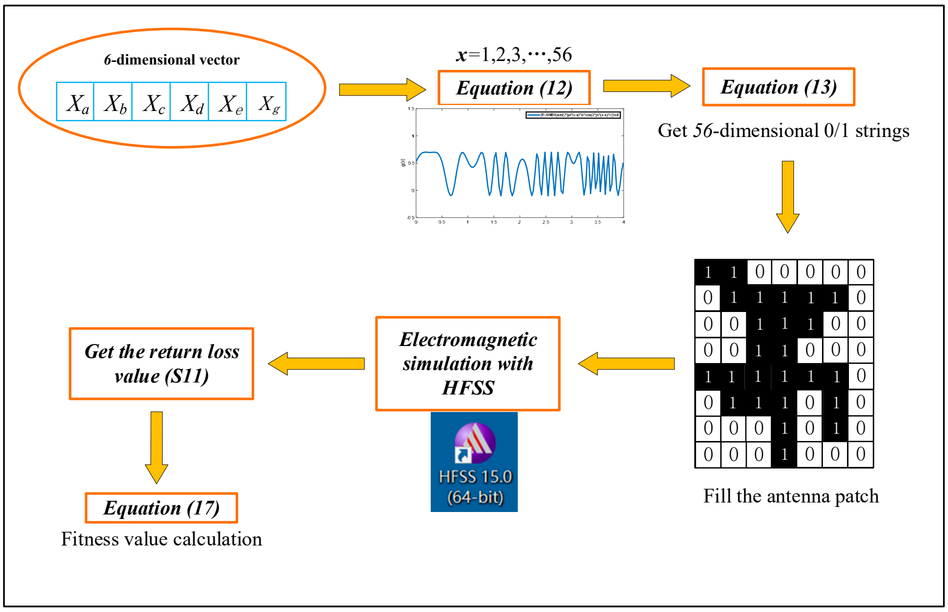


Figure 7. The top view (**a**) and the side view (**b**) of the initial antenna geometry (unit: mm).

The initial antenna setting discretized the rectangular patch into 8 × 7 pixels, that is, 56 pixels. The size of each pixel was 2 × 2 mm. To ensure that the two sub patches that were only in contact at the opposite corners could pass the current, each pixel was overlapped at the edge with an overlapping width of 0.2 mm [43]. In the algorithm parameter settings, the population size was set as 10, and the maximum iteration number was set as 30. To ensure that the antenna has enough metal sheets to achieve radiation performance, the curve of the angle modulation function was controlled above the y-axis as much as possible, so the initial range of the parameters d and g was set within $[-0.5, 1]$. The setting range of the other parameters of the algorithm is the same as the test data of the above knapsack problem. After testing, these parameter ranges can be well applied to antenna designs.

A schematic diagram of the application of phase angle-modulated technology in antenna optimization is shown in Figure 8. By optimizing the six-dimensional vector, the binary data of the 8 × 7 matrix corresponding to the antenna model could be mapped. Then, the HFSS was called for electromagnetic simulation, and the obtained return loss values were calculated as the fitness values. Finally, an iterative update of the optimization algorithm was performed.

A comparison of the convergence performance of the five optimization algorithms is shown in Figure 9, where the abscissa and the ordinate represent the number of iterations and the fitness value, respectively. It can be seen that the P-AMBA proposed in this paper achieved a good balance between exploitation and exploration, showing a strong optimization ability in the early stage of iteration and not falling into the local optimum prematurely. The final antenna topology and S_{11} results optimized by the P-AMBA are shown in Figure 10. The obtained antenna structure met the design objective requirements of the dual-band operation.



6-dimensional vector

X_a X_b X_c X_d X_e X_g

x=1,2,3,⋯,56

Equation (12)

Equation (13)

Get *56*-dimensional 0/1 strings

1	1	0	0	0	0	0
0	1	1	1	1	1	0
0	0	1	1	1	0	0
0	0	1	1	0	0	0
1	1	1	1	1	1	0
0	1	1	1	0	1	0
0	0	0	1	0	1	0
0	0	0	1	0	0	0

Fill the antenna patch

Electromagnetic simulation with HFSS

HFSS 15.0 (64-bit)

Get the return loss value (S11)

Equation (17)

Fitness value calculation

Figure 8. Schematic diagram of phase angle-modulated technology applied to antenna topology optimization.

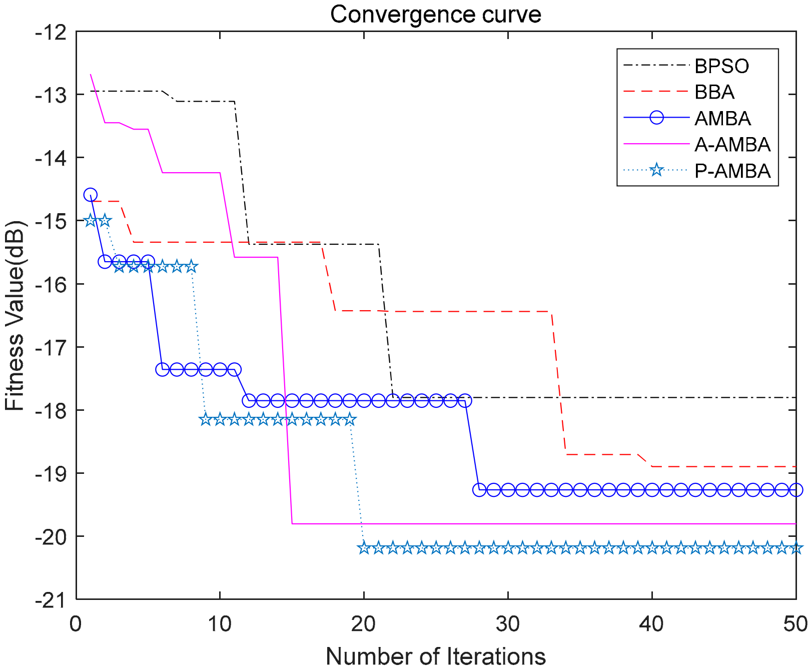


Figure 9. Comparison of the convergence curves of BPSO, BBA, AMBA, A-AMBA, and P-AMBA.

The antenna topology design is different from the traditional antenna structure design. It starts from a simple initial geometry and exhibits superior electrical performance with a compact size. This section used a design example of a dual-band planar monopole antenna to confirm the advantages of P-AMBA in global optimization capabilities and the convergence rate. The results showed that the antenna structure designed by this method is novel and can be used for antenna designs in limited surrounding environment or unexplored areas.

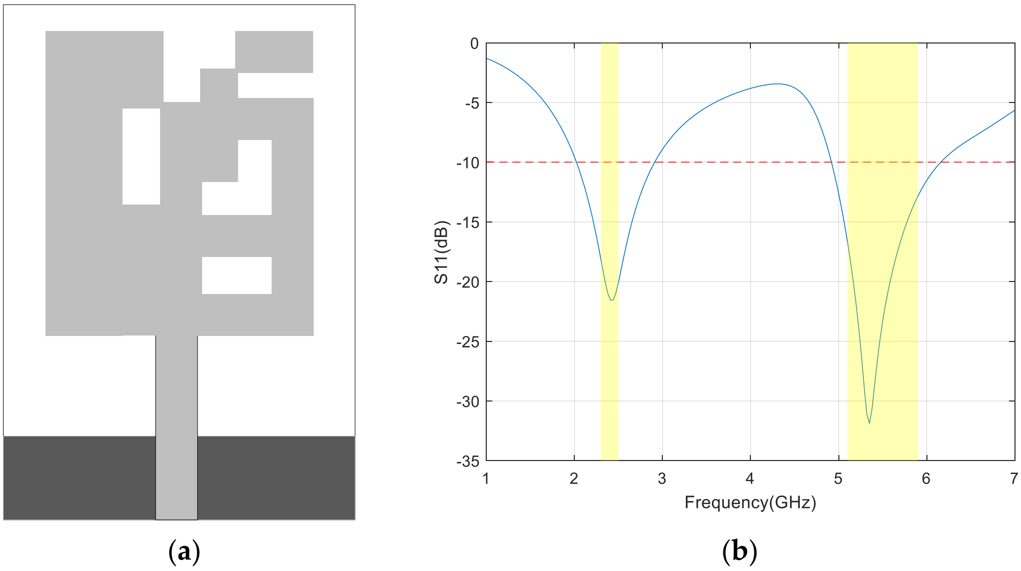


Figure 10. Final geometry (**a**) and S_{11} curve (**b**) of the final antenna.

5. Conclusions

This paper proposed an improved binary bat algorithm for antenna topology designs. The idea was to reduce the optimization time by introducing angle modulation technology to reduce the optimization dimensions. Compared with other optimization algorithms, it also has advantages in global search capability and convergence speed. The knapsack problem was used to verify the optimization ability of P-AMBA in high-dimensional problems. The results showed that the higher the dimension, the more obvious the advantage of this algorithm. Finally, through a topology design example of a compact dual-band planar monopole antenna, the superiority of the proposed P-AMBA in the optimization of high-dimensional discrete search space was verified.

In the antenna optimization example, since much time was spent on the antenna electromagnetic simulation process, the advantages of the algorithm in terms of optimization time were not obvious. Subsequent research will aim at neural networks-based antenna designs by introducing surrogate models instead of electromagnetic simulation tools to achieve fast antenna topology optimization. Moreover, a multi-objective antenna design will also be the task of follow-up research.

Author Contributions: Conceptualization, J.D. and Z.W.; methodology, J.D. and Z.W.; software, Z.W. and J.M.; validation, Z.W. and J.M.; investigation, Z.W.; resources, J.D.; data curation, J.D. and Z.W.; writing—original draft preparation, Z.W.; writing—review and editing, J.D. and J.M.; visualization, Z.W.; supervision, J.D. and Z.W.; project administration, J.M.; funding acquisition, J.D. and J.M. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded in part by the National Natural Science Foundation of China, grant number 61801521, 61471368, and 61971450, in part by the Natural Science Foundation of Hunan Province, grant number 2018JJ2533, and in part by the Fundamental Research Funds for the Central Universities, grant number 2018gczd014 and 20190038020050.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data is contained within the article.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Han, H.-G.; Lu, W.; Hou, Y.; Qiao, J.-F. An Adaptive-PSO-Based Self-Organizing RBF Neural Network. *IEEE Trans. Neural Networks Learn. Syst.* **2016**, *29*, 104–117. [CrossRef] [PubMed]
2. Śmieja, M.; Hajto, K.; Tabor, J. Efficient mixture model for clustering of sparse high dimensional binary data. *Data Min. Knowl. Discov.* **2019**, *33*, 1583–1624. [CrossRef]
3. Varshney, K.R.; Willsky, A.S. Linear Dimensionality Reduction for Margin-Based Classification: High-Dimensional Data and Sensor Networks. *IEEE Trans. Signal Process.* **2011**, *59*, 2496–2512. [CrossRef]
4. Guo, Y.-N.; Cheng, J.; Luo, S.; Gong, D.-W.; Xue, Y. Robust Dynamic Multi-Objective Vehicle Routing Optimization Method. *IEEE/ACM Trans. Comput. Biol. Bioinform.* **2018**, *15*, 1891–1903. [CrossRef] [PubMed]
5. Lu, D.; Wang, L.; Yang, E.; Wang, G. Design of High-Isolation Wideband Dual-Polarized Compact MIMO Antennas with Multiobjective Optimization. *IEEE Trans. Antennas Propag.* **2017**, *66*, 1522–1527. [CrossRef]
6. Faulin, J. Metaheuristics: From Design to Implementation. *Interfaces* **2012**, *42*, 414–415.
7. Kabir, M.; Shahjahan; Murase, K. A new local search based hybrid genetic algorithm for feature selection. *Neurocomputing* **2011**, *74*, 2914–2928. [CrossRef]
8. Minasian, A.A.; Bird, T.S. Particle Swarm Optimization of Microstrip Antennas for Wireless Communication Systems. *IEEE Trans. Antennas Propag.* **2013**, *61*, 6214–6217. [CrossRef]
9. Zorarpacı, E.; Özel, S.A. A hybrid approach of differential evolution and artificial bee colony for feature selection. *Expert Syst. Appl.* **2016**, *62*, 91–103. [CrossRef]
10. Kashan, M.H.; Nahavandi, N.; Kashan, A.H. DisABC: A new artificial bee colony algorithm for binary optimization. *Appl. Soft Comput.* **2012**, *12*, 342–352. [CrossRef]
11. Mirjalili, S. Dragonfly algorithm: A new meta-heuristic optimization technique for solving single-objective, discrete, and multi-objective problems. *Neural Comput. Appl.* **2016**, *27*, 1053–1073. [CrossRef]
12. Tan, Y.; Zhu, Y. Fireworks Algorithm for Optimization. *Constr. Side-Channel Anal. Secur. Des.* **2010**, *6145*, 355–364. [CrossRef]
13. Shi, Y. Brain Storm Optimization Algorithm. *Constr. Side-Channel Anal. Secur. Des.* **2011**, *6728*, 303–309. [CrossRef]
14. Geem, Z.W.; Kim, J.H.; Loganathan, G. A New Heuristic Optimization Algorithm: Harmony Search. *Simulation* **2001**, *76*, 60–68. [CrossRef]
15. Wang, G.-G. Moth search algorithm: A bio-inspired metaheuristic algorithm for global optimization problems. *Memetic Comput.* **2018**, *10*, 151–164. [CrossRef]
16. Yang, X.-S. A New Metaheuristic Bat-Inspired Algorithm. In *Nature Inspired Cooperative Strategies for Optimization (NICSO 2010)*; Springer: Berlin/Heidelberg, Germany, 2010; pp. 65–74. [CrossRef]
17. Ng, C.K.; Wu, C.H.; Ip, W.H.; Yung, K.L. A Smart Bat Algorithm for Wireless Sensor Network Deployment in 3-D Environment. *IEEE Commun. Lett.* **2018**, *22*, 2120–2123. [CrossRef]
18. Zhu, L.F.; Wang, J.S.; Wang, H.Y.; Guo, S.S.; Guo, M.W.; Xie, W. Data Clustering Method Based on Improved Bat Algorithm With Six Convergence Factors and Local Search Operators. *IEEE Access* **2020**, *8*, 80536–80560. [CrossRef]
19. Lu, Y.; Jiang, T. Bi-Population Based Discrete Bat Algorithm for the Low-Carbon Job Shop Scheduling Problem. *IEEE Access* **2019**, *7*, 14513–14522. [CrossRef]
20. Mugemanyi, S.; Qu, Z.; Rugema, F.X.; Dong, Y.; Bananeza, C.; Wang, L. Optimal Reactive Power Dispatch Using Chaotic Bat Algorithm. *IEEE Access* **2020**, *8*, 65830–65867. [CrossRef]
21. Sathananthavathi, V.; Indumathi, G. BAT algorithm inspired retinal blood vessel segmentation. *IET Image Process.* **2018**, *12*, 2075–2083. [CrossRef]
22. Mirjalili, S.; Mirjalili, S.M.; Yang, X.-S. Binary bat algorithm. *Neural Comput. Appl.* **2014**, *25*, 663–681. [CrossRef]
23. Hembecker, F.; Lopes, H.S.; Godoy, W. Particle Swarm Optimization for the Multidimensional Knapsack Problem. In *International Conference on Adaptive and Natural Computing Algorithms*; Springer: Berlin/Heidelberg, Germany, 2007; pp. 358–365. [CrossRef]
24. Guo, S.-S.; Wang, J.-S.; Ma, X.-X. Improved Bat Algorithm Based on Multipopulation Strategy of Island Model for Solving Global Function Optimization Problem. *Comput. Intell. Neurosci.* **2019**. [CrossRef]
25. Xue, J.; Xiao, J.; Zhu, J. Binary Fireworks Algorithm for 0-1 Knapsack Problem. In Proceedings of the 2019 International Conference on Artificial Intelligence and Advanced Manufacturing (AIAM), Dublin, Ireland, 16–18 October 2019; pp. 218–222. [CrossRef]
26. Huang, Y.; Wang, P.; Li, J.; Chen, X.; Li, T. A Binary Multi-Scale Quantum Harmonic Oscillator Algorithm for 0–1 Knapsack Problem With Genetic Operator. *IEEE Access* **2019**, *7*, 137251–137265. [CrossRef]
27. Dahi, Z.A.E.M.; Mezioud, C.; Draa, A. Binary Bat Algorithm: On The Efficiency of Mapping Functions When Handling Binary Problems Using Continuous-variable-based Metaheuristics. *New Trends Nonlinear Control Theory* **2015**, *456*, 3–14. [CrossRef]
28. Huang, X.; Li, P.; Pu, Y. Amplitude Angle Modulated Bat Algorithm With Application to Zero-One Knapsack Problem. *IEEE Access* **2019**, *7*, 27957–27969. [CrossRef]
29. Le, T.T.; Yun, T.-Y. Miniaturization of a Dual-Band Wearable Antenna for WBAN Applications. *IEEE Antennas Wirel. Propag. Lett.* **2020**, *19*, 1452–1456. [CrossRef]
30. Zheng, K.-K.; Chu, Q.-X. A Small Symmetric-Slit-Shaped and Annular Slotted BeiDou Antenna With Stable Phase Center. *IEEE Antennas Wirel. Propag. Lett.* **2017**, *17*, 146–149. [CrossRef]
31. Delabie, C.; Villegas, M.; Picon, O. Creation of new shapes for resonant microstrip structures by means of genetic algorithms. *Electron. Lett.* **1997**, *33*, 1509. [CrossRef]

32. Guariglia, E. Entropy and Fractal Antennas. *Entropy* **2016**, *18*, 84. [CrossRef]
33. Guariglia, E. Primality, Fractality, and Image Analysis. *Entropy* **2019**, *21*, 304. [CrossRef]
34. Tao, Y.; Zang, X.; Wang, C.; Yang, C. Design of a fragment-type UHF RFID tag integrated into printed circuit board. *Microw. Opt. Technol. Lett.* **2019**, *61*, 676–681. [CrossRef]
35. Li, Q.-Q.; Chu, Q.-X.; Chang, Y.-L. Design of Compact High-Isolation MIMO Antenna With Multiobjective Mixed Optimization Algorithm. *IEEE Antennas Wirel. Propag. Lett.* **2020**, *19*, 1306–1310. [CrossRef]
36. Brooks, S.P.; Morgan, B.J.T. Optimization Using Simulated Annealing. *J. R. Stat. Soc. Ser. D* **1995**, *44*, 241–257. [CrossRef]
37. Proakis, J.G.; Salehi, M.; Zhou, N.; Li, X. *Communication Systems Engineering*; Prentice Hall: Upper Saddle River, NJ, USA, 1994; Volume 2.
38. Pampara, G.; Franken, N.; Engelbrecht, A. Combining Particle Swarm Optimisation with angle modulation to solve binary problems. *IEEE Congr. Evol. Comput.* **2005**, 89–96. [CrossRef]
39. Leonard, B.J.; Engelbrecht, A.P.; Cleghorn, C.W. Critical considerations on angle modulated particle swarm optimisers. *Swarm Intell.* **2015**, *9*, 291–314. [CrossRef]
40. Dantzig, G.B. Discrete-Variable Extremum Problems. *Oper. Res.* **1957**, *5*, 266–288. [CrossRef]
41. Zhang, S.; Liu, S. A Discrete Improved Artificial Bee Colony Algorithm for 0–1 Knapsack Problem. *IEEE Access* **2019**, *7*, 104982–104991. [CrossRef]
42. Li, Y.; He, Y.; Liu, X.; Guo, X.; Li, Z. A novel discrete whale optimization algorithm for solving knapsack problems. *Appl. Intell.* **2020**, *50*, 3350–3366. [CrossRef]
43. Ohira, M.; Deguchi, H.; Tsuji, M.; Shigesawa, H. Multiband Single-Layer Frequency Selective Surface Designed by Combination of Genetic Algorithm and Geometry-Refinement Technique. *IEEE Trans. Antennas Propag.* **2004**, *52*, 2925–2931. [CrossRef]