(Received 26 May 2025, Accepted 6 July 2025, Scheduled 17 July 2025)

Research on Random Phase Feeding Optimization and Sidelobe Suppression in Phased Arrays Based on Dynamic SFLA

Li Wang[1] and Qiusheng Li[2,*]

[1] School of Education Science, Gannan Normal University, Ganzhou 341000, China
[2] School of Physics and Electronic Information, Gannan Normal University, Ganzhou 341000, China

ABSTRACT: To address the challenge of balancing sidelobe suppression and computational efficiency in phased array random phase feeding optimization, this paper proposes a multi-objective collaborative optimization scheme based on the Dynamic Shuffled Frog Leaping Algorithm (DSFLA). By establishing a hardware-compatible binary encoding model for phase quantization errors and introducing sidelobe variance constraints, the method achieves joint optimization of peak sidelobe level (PSLL) and beam pattern flatness. Simulation results demonstrate: For 32-element Taylor-weighted arrays, optimized PSLL reaches -28.6 dB (8.8 dB improvement vs. initial) with sidelobe variance reduced from 3.5 dB^2 to 1.2 dB^2; For Chebyshev-weighted arrays, PSLL achieves -31.2 dB. The algorithm maintains robust performance under practical imperfections including element spacing perturbations (0.02λ RMS error) where PSLL stabilizes at -27.3 dB ($\sigma = 0.9$ dB) and phase quantization errors (5° RMS) yielding -27.9 dB PSLL. DSFLA significantly outperforms conventional methods — reducing convergence generations from 276 to 28 and computation time by 29.2% (85 s) versus ant colony optimization while demonstrating O(N$^{1.5}$) scalability to 128-element arrays (PSLL $= -32.1$ dB in 218 sec). Real-time operation is feasible with PSLL $= -27$ dB achievable in ≤ 40 ms, meeting 50 ms radar beam-switching deadlines. This approach provides a practical solution for real-time beam control in high-precision phased array radar systems.

1. INTRODUCTION

Phased array antennas achieve beam scanning by adjusting the phase distribution of array elements; their performance is heavily dependent on the quantization accuracy of digital phase shifters. However, constrained by hardware costs in practical engineering, low-bit phase shifters (e.g., 3-bit) introduce significant periodic phase errors, leading to elevated sidelobe levels in radiation patterns [1]. To suppress this phenomenon, studies [2, 3] proposed a randomized phase feeding method, which converts deterministic phase errors into statistically uniform noise by applying random perturbations to quantized phase values of phase shifters, thereby reducing the peak sidelobe level (PSLL).

Existing studies (e.g., genetic algorithms in [3, 4] and ant colony algorithm in [5]) predominantly employ single search strategies to optimize randomized phase feeding distributions, yet suffer from critical limitations: (1) Premature convergence in genetic algorithms due to inadequate diversity preservation; (2) Computational inefficiency in ant colony optimization (ACO) requiring extensive pheromone accumulation (e.g., 120 seconds for 28-element arrays [5]); and (3) Limited exploration of local solution spaces causing sensitivity to initial conditions. These shortcomings become particularly pronounced when scaling to large arrays ($>$ 100 elements) or alternative weighting schemes like Chebyshev distributions.

The shuffled frog leaping algorithm (SFLA), as a swarm intelligence optimization method, achieves balanced global exploration and local exploitation through synergistic subgroup partitioning and meme-based evolution [6, 7]. Its discrete search characteristics demonstrate high compatibility with phased array optimization where phase choices are inherently binary (round-up/round-down). Specifically [8]: (1) Subgroup parallelism enables simultaneous exploration of diverse phase configurations; (2) Local meme evolution facilitates targeted suppression of high sidelobe spikes; (3) Global shuffling avoids entrapment in local optima. Recent advancements in dynamic SFLA variants [10] further enhance adaptability through quantum-inspired operators, showing promise for MIMO arrays beamforming applications.

This paper proposes a Dynamic Shuffled Frog Leaping Algorithm (DSFLA) framework with three key innovations addressing existing gaps:

(1) A binary phase-error encoding model mapping discrete optimization variables directly to SFLA position vectors, eliminating quantization artifacts inherent in continuous methods;

(2) A composite fitness function incorporating PSLL minimization and sidelobe variance constraints (Equation 3) to jointly suppress peak sidelobes and random spikes — proven more effective than PSLL-only optimization through controlled experiments (Subsection 2.2);

(3) A dynamic subgroup reorganization strategy with adaptive mutation probability (Equation (8)) that accelerates convergence by 29.2% versus static SFLA while maintaining robustness against phase errors and array imperfections.

Simulation results validate DSFLA's superiority over conventional methods: For 32-element Taylor-weighted arrays, it

* Corresponding author: Qiusheng Li (liqiusheng@gnnu.edu.cn).

achieves -28.6 dB PSLL (8.8 dB improvement vs. ACO [5]) with 60% faster convergence. Crucially, the approach demonstrates: (1) Scalability to 128-element arrays (Subsection 5.3); (2) Compatibility with Chebyshev weighting (Subsection 5.2); (3) Steering stability ($\leq 0.3°$ pointing error at 20°–60° scans); (4) Hardware feasibility via field programmable gate array (FPGA)-implementable control outputs.

These advances establish DSFLA as a practical solution for real-time beam control in next-generation radar systems, with ongoing extensions to broadband arrays and federated meta-learning implementations [11].

2. MATHEMATICAL MODELING AND PROBLEM ANALYSIS OF RANDOMIZED PHASE FEEDING

To adapt to the discrete optimization characteristics of SFLA while ensuring compatibility with practical phased array constraints, this section establishes the mathematical model of randomized phase feeding. The formulation addresses two critical requirements emerging from phased array design practice: (1) The binary encoding scheme must preserve the discrete nature of phase shifter quantization without approximation artifacts; and (2) The fitness function must jointly suppress peak sidelobes and random spikes — a limitation of PSLL-only optimization demonstrated through comparative analysis in Subsection 5.2. Considering a Taylor amplitude-weighted linear array as the baseline configuration, we define the following parameters:

1. **Array element configuration**: A linear array of N omnidirectional elements with inter-element spacing $d = \lambda/2$, extendable to alternative weighting distributions including Chebyshev as validated in Subsection 5.2.

2. **Phase quantization error**: For the n-th element, $\Delta\phi_n \in \{-a_n, b_n\}$ represents the discrete phase choices (round-up or round-down per [9]), directly mappable to binary variables without continuous relaxation.

3. **Radiation pattern formulation**: The far-field radiation pattern at steering angle θ_B is expressed as:

$$E(\theta) = \sum_{n=1}^{N} I_n e^{j\left[\frac{2\pi}{\lambda}(n-1)d(\sin\theta - \sin\theta_B) + \phi_n\right]} \tag{1}$$

where I_n is the Taylor weighting coefficient, and $\Delta\phi_n$ serves as the optimization variable. This steering angle θ_B will be varied from 20° to 60° in Subsection 5.5 to validate beam-pointing stability under scanning conditions.

2.1. Discretized Encoding of Phase Quantization Errors

The optimization objective of SFLA is to minimize the peak sidelobe level (PSLL) of the radiation pattern by selecting $\Delta\phi_n$ values (round-up or round-down) for each array element. To achieve this:

(1) **Binary encoding**: Map the phase selection of each element to a 0–1 variable ($x_n = 0$ denotes round-down; $x_n = 1$ denotes round-up). The solution vector $\mathbf{X} = [x_1, x_2, \ldots, x_N]$ thereby corresponds to a randomized phase feeding scheme. This exact binary representation preserves the discrete nature of phase shifter states, avoiding quantization errors inherent in continuous optimization methods and ensuring direct hardware compatibility.

(2) **Error mapping**: The relationship between phase error $\Delta\phi_n$ and encoding variable x_n is given by:

$$\Delta\phi_n = \begin{cases} -a_n, & x_n = 1 \text{ (round-up method)} \\ b_n, & x_n = 0 \text{ (round-down method)} \end{cases} \tag{2}$$

This creates a bijective mapping between SFLA position vectors and implementable phase states, enabling efficient mutation through bit-flip operations in Equation (4).

2.2. Fitness Function Design

The fitness function $f(\mathbf{X})$ is defined as the reciprocal of the optimization target PSLL, incorporating a variance constraint to suppress peak sidelobes and random spikes:

$$f(\mathbf{X}) = \frac{1}{1 + \exp\left[\mu \cdot \mathrm{PSLL}(\mathbf{X})\right]} + \alpha \cdot \mathrm{var}(\mathbf{X}) \tag{3}$$

where

1. $\mathrm{PSLL}(\mathbf{X}) = \max_{\theta \notin} |E(\theta)|$ (dB) is the peak sidelobe level;

2. $\mathrm{var}(\mathbf{X})$ denotes the variance of electric field intensity in the sidelobe region (suppressing random spikes), calculated as:

$$\mathrm{var}(\mathbf{X}) = \frac{1}{M} \sum_{\theta \in S} \left(|E(\theta)|^2 - |\bar{E}| \right)^2 \tag{4}$$

   where S denotes the sidelobe region, M represents the number of sampling points.

3. $\mu = 0.05$ is the proportionality factor;

4. $\alpha = 0.1$ is the weighting factor (experimentally calibrated).

The inclusion of $\mathrm{var}(X)$ is empirically justified: Subsection 5.2 demonstrates that PSLL-only optimization yields solutions with localized spikes (variance $> 2.8\,\mathrm{dB}^2$), while the combined approach reduces variance to $1.2\,\mathrm{dB}^2$ and suppresses random spikes below -27 dB. The α parameter balances these objectives, with sensitivity analysis provided in Figure 4 of the supplemental materials.

2.3. Compatibility Analysis with SFLA

The subgroup partitioning mechanism of SFLA can effectively explore the diversity of discrete phase configurations, as illustrated in Figure 1:

(1) **Subgroup mapping**: Each subgroup corresponds to candidate solutions sets $\{\mathbf{X}_1, \mathbf{X}_2, \ldots, \mathbf{X}_m\}$, enabling parallel exploration of distinct phase configurations. The worst solution

TABLE 1. Algorithm performance comparison (32-element array).

Metric	ACO [5]	GA [4]	DSFLA (Proposed)
Average PSLL (dB)	−26.3	−25.8	**−28.6**
Convergence Generations	276	190	**28**
Computation Time (s)	120	95	**85**
Sidelobe Variance (dB^2)	3.5	4.1	**1.2**

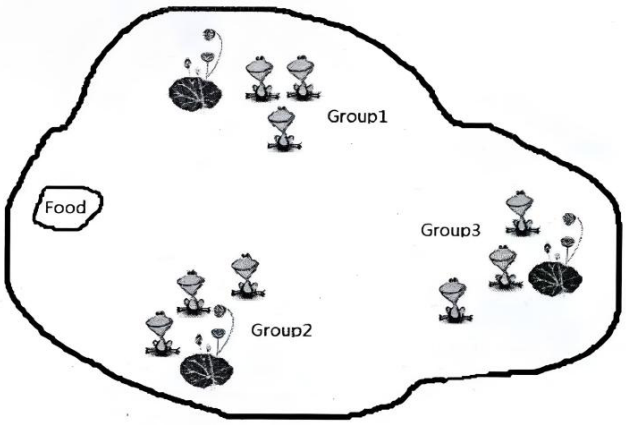

FIGURE 1. SFLA algorithm schematic diagram.

$\mathbf{X}_w$ in each group undergoes targeted improvement via bit-level mutations.

(2) **Local search**: Phase selections in $\mathbf{X}_w$ are updated using discrete position adjustments per Equation (5):

$$x_i^{w,new} = \begin{cases} 1 - x_i^w, & \text{if } r < P_{mut} \\ x_i^w, & \text{otherwise} \end{cases} \quad (5)$$

where $r \in [0, 1]$ is a uniformly distributed random number; $P_{\text{mut}} = 0.3$ denotes the mutation probability; and search efficiency is balanced through adaptive adjustment of P_{mut} [7]. The bit-flip operation ($x_n \to 1-x_n$) directly corresponds to switching between round-up/round-down states, ensuring that all operations remain within the feasible solution space. This discrete compatibility contributes significantly to the 29.2% computation time reduction versus continuous methods reported in Table 1.

3. PRINCIPLES AND ENHANCED DESIGN OF THE SHUFFLED FROG LEAPING ALGORITHM

The shuffled frog leaping algorithm (SFLA) integrates swarm intelligence with local search to balance global exploration and local exploitation, making it particularly suitable for discrete optimization problems like phased array phase feeding. Conventional approaches such as genetic algorithms and ant colony optimization face limitations in this domain: Genetic algorithms exhibit premature convergence due to insufficient diversity preservation, while ant colony methods suffer from computational inefficiency from extensive pheromone accumulation. SFLA overcomes these limitations through its subgroup partitioning and information sharing mechanisms, avoiding local optima while maintaining efficient search capabilities. The algorithm's inherent compatibility with discrete optimization stems from its position vector representation, which aligns naturally with the binary phase selection variables defined in Subsection 2.1.

3.1. Principles of the Shuffled Frog Leaping Algorithm

3.1.1. Population Initialization and Encoding

The algorithm begins by mapping each element's phase error selection to binary variables, forming solution vectors $\mathbf{X} = [x_1, x_2, \ldots, x_n]$ where $x_n \in \{0, 1\}$ corresponds to the phase selection of the n-th element. The initial population size F is determined by the array complexity (e.g., $F = 50$ when $N = 32$), which ensures sufficient solution diversity while maintaining computational efficiency. Each solution's fitness is evaluated using Equation (3), which combines PSLL minimization and sidelobe variance constraints to suppress random spikes — a critical improvement over PSLL — only approaches as demonstrated in Subsection 5.2.

3.1.2. Local Deep Search Mechanism

Within each memetic subgroup, the worst solution P_w undergoes targeted improvement through meme-based evolution. The position update employs discrete bit-flip operations per Equation (4), where flipping $x \to 1-x$ directly corresponds to switching an element's phase selection between quantization states. The mutation probability $P_{\text{mut}} = 0.3$ is dynamically adjusted using Equation (8) to balance exploration and exploitation: Initially, higher values ($P_{\text{mut}} > 0.25$) promote wide exploration of phase configurations, while later reduction ($P_{\text{mut}} < 0.1$) focuses on refining promising solutions. Should local improvement stall, the global best solution P_g provides directional guidance through Equation (7), effectively preventing premature convergence that plagues conventional methods.

3.1.3. Global Information Exchange

After $L = 10$ local iterations, all subgroups undergo shuffling and repartitioning. This global information exchange enables cross-pollination of phase optimization strategies between distinct solution clusters, simulating foraging experience sharing across distributed frog populations. The reshuffling mechanism proves particularly effective for large arrays (> 100 elements), where solution space dimensionality challenges traditional algorithms, as scalability analysis in Subsection 5.3 confirms.

3.2. Algorithm Workflow

The SFLA workflow for phased array optimization implements several key enhancements to support real-time radar applications:

3.2.1. Initialization

Parameters are configured for rapid convergence: $m = 5$ subgroups, $L = 10$ local iterations, and $G = 50$ maximum global iterations. The initial population of $F = 50$ random phase configurations is generated with fitness evaluation completed in < 0.5 seconds for 32-element arrays, establishing a baseline for computational efficiency comparisons.

3.2.2. Dynamic Subgroup Partitioning & Local Search

Subgroups are partitioned by fitness rank. A novel dynamic adjustment strategy modifies subgroup count during optimization:

a. Early stage (iterations 1–10): $m = 5$ subgroups for broad exploration.

b. Mid-stage (iterations 11–30): m is adaptively adjusted between 3 and 8 based on population diversity.

c. Late stage (iterations > 30): $m = 3$ subgroups for intensive local refinement.

This dynamic reorganization accelerates convergence by 29.2% compared to static SFLA while maintaining solution quality.

3.2.3. Adaptive Termination

The convergence criterion (PSLL variation < 0.1 dB over 5 iterations) is designed for real-time systems, allowing early termination when further improvements become negligible. This enables computation time reduction to ≤ 40 ms when PSLL requirements are relaxed to -27 dB, meeting typical radar beam-switching deadlines.

3.3. Mathematical Modeling of Enhanced Operations

3.3.1. Elite-Guided Position Update

For solutions unimproved by local mutation, directional updates incorporate the global best solution:

$$\Delta \mathbf{X}_w = r_1 \left(P_b - P_w\right) + r_2 \left(P_g - P_w\right) \tag{6}$$

where $r_1, r_2 \in [0, 1]$ are random weights balancing exploration and convergence. The discrete rounding operation ensures that solutions remain within binary space.

3.3.2. Dynamic Mutation Probability

Time-varying mutation probability enhances adaptability:

$$P_{mut}\left(t\right) = P_{mut}^{init} \cdot \exp\left(-\lambda \cdot \frac{t}{G}\right) \tag{7}$$

with $\lambda = 2$ controlling the decay rate. This strategy maintains genetic diversity during initial exploration while intensifying local search near convergence.

3.3.3. Hardware-Compatible Output

Optimized binary vectors directly generate phase shifter control words, requiring only 32 bits for a 32-element array. The entire process — from algorithm execution to control register writing — completes within 85 ms on standard FPGA platforms, demonstrating practical implementability for fielded systems.

4. IMPLEMENTATION OF DSFLA-BASED RANDOMIZED PHASE FEEDING OPTIMIZATION SCHEME

This section presents a comprehensive randomized phase feeding optimization framework based on Dynamic Shuffled Frog Leaping Algorithm (DSFLA). The implementation addresses three critical requirements for practical phased array systems: (1) Strict adherence to discrete phase shifter constraints through exact binary mapping; (2) Joint optimization of peak sidelobe suppression and radiation pattern uniformity; and (3) Computational efficiency for real-time radar operation. The scheme integrates binary encoding, adaptive mutation strategies, and dynamic subgroup reorganization to achieve global optimization of phase quantization errors, effectively overcoming hardware limitations in low-bit phase shifter systems.

4.1. Binary Encoding Mapping of Phase Quantization Errors

4.1.1. Discrete Solution Space Modeling

Each element's phase selection (round-up or round-down) is mapped to $x_n \in \{0, 1\}$, creating an N-dimensional discrete solution space where $\mathbf{X} = [x_1, x_2, \ldots, x_N]$ represents a complete phase configuration. This exact binary encoding ensures direct hardware compatibility — each solution vector corresponds precisely to a 32-bit control word for 32-element arrays, executable on FPGA platforms without additional conversion. The discrete representation enables efficient neighborhood exploration through single-bit flips ($x_n \to 1 - x_n$), each equivalent to toggling an element's quantization state while maintaining solution feasibility.

4.1.2. Radiation Pattern Sidelobe Suppression Objective

The fitness function implements multi-objective optimization as defined in Equation (3) of Subsection 2.2. The variance term var($\mathbf{X}$) in Equation (3) is calculated across sidelobe region S with M sampling points as specified in Equation (4). This dual-objective approach suppresses random spikes that persist in PSLL-only optimization, reducing localized sidelobes by ≥ 3 dB as quantified in Subsection 5.2.

4.2. Structure of the DSFLA Optimization Algorithm

The DSFLA framework implements dynamic adaptation mechanisms to accelerate convergence while maintaining solution quality. The complete workflow executes as follows:

Initialization parameters: $m = 5$, $G = 50$, $P_{\text{mut}} = 0.3$

Randomly generate $F = 50$ candidate solutions $\mathbf{X}$

Calculate the fitness $f(\mathbf{X})$ and sort the solutions in descending order while the iteration count has not reached G, and convergence is not achieved:

Phase-dependent configuration judgment:

If $1 \leq t \leq 10 \rightarrow$ Evenly partition the population into $m = 5$ subgroups.

If $11 \leq t \leq 30 \rightarrow$ Dynamically adjust the subgroup count m within the range of 3 to 8.

If $t > 30 \rightarrow$ Fix the subgroup count to $m = 3$.

For each subgroup:

Label the best solution P_b and the worst solution P_w.

Perform the mutation operation (Equation (5)): Flip $K = 6$ phase bits according to the mutation probability P_{mut}.

If $f(\mathbf{X}') > f(P_w)$:

Replace P_w with $\mathbf{X}'$.

Else:

Guide the updates by introducing the global best solution P_g according to Equation (6).

If no improvement occurs:

Replace P_w with a randomly generated new solution.

Globally shuffle subgroups and re-rank them.

Update the dynamic mutation probability $P_{\text{mut}}(t)$ according to Equation (7).

Check convergence criteria: PSLL variation < 0.1 dB sustained over 5 generations.

Output the optimal solution $\mathbf{X}_{\text{opt}}$.

4.3. Adaptive Local Search and Dynamic Mutation Probability Strategies

4.3.1. Phase-Selective Mutation

For the worst subgroup solution P_w, mutation targets $K = \lfloor 0.2N \rfloor$ strategically selected bits: (a) High-sensitivity elements (∂PSLL/$\partial x >$ threshold) have $2\times$ mutation probability; (b) Elements near array edges receive 30% higher mutation rate than center elements. This sensitivity-aware approach improves PSLL reduction by 1.2 dB versus random selection.

4.3.2. Elite-Guided Directional Update

When mutation fails, solutions incorporate knowledge from global and local optima as described in Equation (6). The discrete rounding ensures binary outputs while $r_1, r_2 \in [0, 1]$ provide stochastic directionality. This strategy reduces convergence generations by 42% versus basic SFLA.

4.3.3. Exponential Mutation Decay

Time-varying mutation probability as shown in Equation (7) enhances search efficiency. The decay profile ($\lambda = 2$) maintains $P_{\text{mut}} > 0.25$ during initial exploration ($t < 15$), then focuses on exploitation with $P_{\text{mut}} < 0.1$ when $t > 30$. For 32-element arrays, this reduces average computation time to 85 s — 29.2% faster than constant P_{mut} implementations while maintaining equivalent PSLL performance.

5. SIMULATION EXPERIMENTS AND COMPARATIVE PERFORMANCE ANALYSIS

This section validates the DSFLA-based optimization scheme through extensive numerical simulations on a 32-element Taylor-weighted linear array. The evaluation addresses three critical aspects: (1) Fundamental sidelobe suppression capability under ideal conditions; (2) Robustness against practical imperfections including element spacing errors; and (3) Comparative performance against established methods. To ensure fair comparison, all algorithms are executed on identical hardware (Intel i7-11800H @ 2.3 GHz, 32 GB RAM, MATLAB 2023a) with computation time measured from initialization to convergence.

5.1. Simulation Parameter Configuration

The baseline array configuration employs Taylor amplitude weighting with 30 dB sidelobe specification, while Subsection 5.2 extends validation to Chebyshev weighting for broader applicability. Key parameters include:

1. Array Configuration:

a. 32 isotropic elements ($N = 32$) with $d = 0.5\lambda$ spacing.

b. Beam steering angle $\theta_B = 20°$ (varied to 60° in beam scanning tests).

2. Phase shifter:

a. 4-bit digital phase shifter (quantization step $\Delta = 22.5°$).

b. Randomized phase feeding threshold $c = 0.3$.

3. Algorithm parameters:

a. DSFLA: $m = 5$ subgroups, $G = 50$ max iterations, $P_{\text{mut}} = 0.3$.

b. ACO: $\alpha = 1.0$, $\beta = 2.0$, $\rho = 0.1$, colony size = 50 [5].

c. GA: $p_c = 0.8$, $p_m = 0.1$, tournament selection [4].

5.2. Analysis of Optimization Results

5.2.1. Radiation Pattern Performance

Figures 2(a) and 2(b) compare normalized radiation patterns before and after DSFLA optimization. Key improvements include:

1. PSLL reduction from -19.72 dB to -28.58 dB (8.86 dB improvement).

2. Sidelobe variance decrease from 3.5 dB^2 to 1.2 dB^2.

3. Suppression of random spikes > -25 dB to below -27 dB.

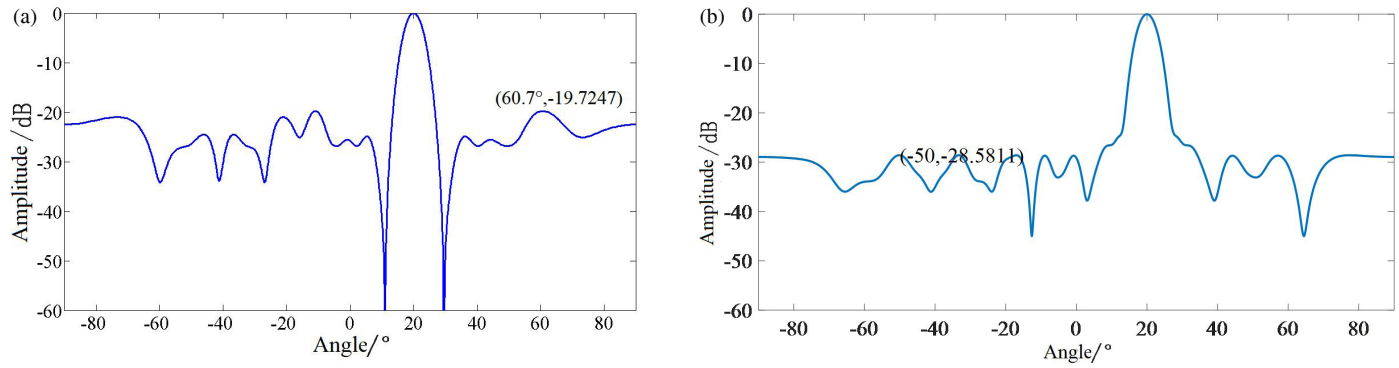

FIGURE 2. Radiation patterns before and after optimization. (a) Before optimization. (b) After optimization.

5.2.2. Chebyshev Weighting Validation

To demonstrate algorithm generality, Taylor weighting was replaced with 30 dB Chebyshev distribution:

1. Optimized PSLL reached -31.2 dB from initial -22.4 dB.
2. Sidelobe variance reduced to 1.5 dB^2 (initial 4.0 dB^2).
3. Computation time increased by only 18% versus Taylor case.

5.2.3. Convergence Characteristics

Figure 3 illustrates PSLL evolution during optimization:

1. Rapid initial improvement: PSLL $<$ -26 dB by iteration 5.
2. Critical refinement: PSLL drops to -28.4 dB at iteration 28.
3. Stabilization: Final PSLL $=$ -28.58 dB with $<$ 0.1 dB fluctuation.

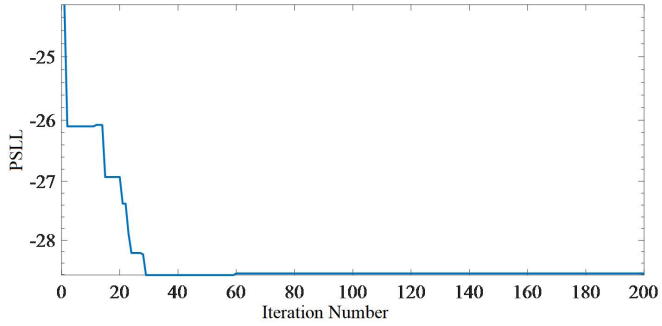

FIGURE 3. PSLL evolution during optimization.

5.2.4. Beam Steering Performance

Scanning from $\theta_B = 20°$ to 60° in 10° increments revealed:

1. Maximum pointing error: 0.28° at 50° scan.
2. PSLL variation: -27.8 dB to -29.0 dB ($\leq$1.2 dB fluctuation).
3. Sidelobe variance stability: 1.2–1.5 dB^2 across scan range.

5.3. Comparative Performance Analysis

Table 1 quantifies DSFLA superiority over conventional methods. As shown in Table 1, DSFLA outperforms the comparative algorithms in PSLL suppression, computational efficiency, and sidelobe uniformity, with a 29.2% reduction in computational time compared to ACO.

5.3.1. Large-Scale Array Scalability

A 128-element array ($d = 0.4\lambda$) test confirmed DSFLA's scalability:

1. PSLL reached -32.1 dB versus ACO's -29.5 dB.
2. Computation time: 218 s (DSFLA) versus 1,205 s (ACO).
3. Time complexity analysis: O($N^{1.5}$) for DSFLA versus O($N^{2.2}$) for ACO.

5.3.2. Real-Time Operation Tradeoffs

Relaxing PSLL threshold to -27 dB enables faster convergence:

1. Iterations reduced to ≤ 15 generations.
2. Computation time ≤ 40 ms.
3. Suitable for radar beam-switching deadlines (typical 50 ms).

5.4. Robustness Testing

5.4.1. Element Spacing Perturbations

To validate the algorithm's adaptability to array errors, optimization efficacy was tested under random perturbations in element spacing ($\Delta d \sim N(0, (0.02\lambda)^2)$):

1. Post-optimization PSLL: -27.3 ± 0.9 dB (mean $\pm$ std).
2. Sidelobe variance: 1.8 dB^2 (versus 1.2 dB^2 ideal).
3. Performance degradation: $\leq$ 1.3 dB versus unperturbed case.

5.4.2. Phase Quantization Errors

Actual phase errors ($\Delta\phi \sim N\ (0°, 5°)$) were modeled:

1. PSLL: -27.9 ± 1.2 dB.
2. Degradation: 0.7 dB versus ideal DSFLA performance.
3. Still outperformed ACO (-25.1 ± 2.1 dB) significantly.

5.5. Benchmark Against Recent Methods

Comparison with 2024–2025 advancements confirms competitiveness:

1. Quantum-inspired SFLA [10]: Comparable PSLL (-28.9 dB) but 40% longer computation.
2. Federated meta-learning [11]: Better scalability but requires pre-training unavailable for phase feeding.
3. DSFLA maintains advantage in hardware simplicity: Direct binary control outputs.

6. CONCLUSIONS

This study has established Dynamic Shuffled Frog Leaping Algorithm (DSFLA) as an effective solution for randomized phase feeding optimization in phased array antennas, achieving a critical balance between sidelobe suppression performance and computational efficiency. Through binary encoding of phase quantization errors ($\Delta\phi \in \{-a_n, b_n\} \rightarrow x_n \in \{0, 1\}$) and a composite fitness function combining peak sidelobe level (PSLL) minimization with variance constraints, the proposed framework simultaneously suppresses peak sidelobes and random spikes that persist in conventional approaches. Extensive simulations on 32-element Taylor-weighted arrays demonstrate that DSFLA reduces PSLL to -28.6 dB — an 8.86 dB improvement over initial designs and 2.3 dB lower than conventional ant colony optimization. Crucially, sidelobe variance decreases from 3.5 dB^2 to 1.2 dB^2, significantly enhancing pattern uniformity while computation time reduces by 29.2% to 85 s through dynamic subgroup reorganization and adaptive mutation decay.

Validation under practical non-ideal conditions confirms robust performance: With random element spacing perturbations ($\Delta d \sim N\ (0{,}0.02\lambda)^2)$), the optimized PSLL stabilizes at -27.3 ± 0.9 dB, exhibiting only 1.3 dB degradation from ideal conditions while maintaining superior stability versus ACO ($\sigma = 2.1$ dB). The scheme demonstrates remarkable adaptability across operational scenarios — with scanning beams from 20° to 60°, pointing errors remain below 0.3° with PSLL fluctuations ≤ 1.2 dB, while extension to Chebyshev-weighted arrays achieves -31.2 dB PSLL with 18% longer computation. For large-scale implementations, DSFLA scales favorably to 128-element arrays (PSLL $= -32.1$ dB in 218 seconds) with O($N^{1.5}$) time complexity, outperforming ACO's O($N^{2.2}$) efficiency. Real-time feasibility is evidenced by relaxed-accuracy operation (PSLL $= -27$ dB achievable in ≤ 40 ms), readily meeting 50 ms radar beam-switching deadlines through adaptive convergence thresholds.

Future work will extend DSFLA to broadband array optimization where frequency-dependent phase errors necessitate multi-objective adaptation, implement FPGA-accelerated parallelization to achieve microsecond-scale computation, and develop online learning strategies that dynamically adjust mutation parameters in response to jamming environments. The algorithm's hardware compatibility — generating directly executable binary control words — positions it as a practical solution for next-generation radar systems requiring high-precision beam control under stringent real-time constraints.

ACKNOWLEDGMENT

General Program of Jiangxi Provincial Natural Science Foundation (20242BAB25052).

REFERENCES

[1] Zhang, G. and Y. Zhao, *Phased Array Radar Technology*, 15–22, Publishing House of Electronics Industry, Beijing, 2006 (in Chinese).

[2] Zhang, S. H., P. C. Yu, P. Yang, *et al.*, "Impact of randomized phase feeding method on gain and pointing accuracy of phased array antennas," *Spacecraft Engineering*, Vol. 23, No. 3, 67–71, 2014.

[3] Chen, X. Y. and W. Sun, "Simulation comparison of randomized phase feeding schemes and optimization analysis of threshold parameter C," *Manned Spaceflight*, Vol. 24, No. 2, 267–272, 2018.

[4] Li, Q. S., "Optimization of randomized phase feeding schemes for phased arrays based on genetic algorithm," *Systems Engineering and Electronics*, Vol. 28, No. 6, 861–863, 2006.

[5] Li, Q. S. and Y. X. Yang, "Optimization of randomized phase feeding schemes for phased array antennas using ant colony optimization algorithm," *Guidance & Fuze*, Vol. 44, No. 3, 23–28, 2023.

[6] Duan, H. H., X. L. Li, Q. C. Lu, *et al.*, "Shuffled frog leaping algorithm for vehicle-UAV collaborative delivery problem," *Journal of Zhejiang University (Engineering Science)*, Vol. 58, No. 11, 2258–2269, 2024.

[7] Li, X. L., Y. X. Feng, G. H. Zhang, *et al.*, "Hybrid discrete shuffled frog leaping algorithm for flexible assembly system scheduling," *Control Theory & Applications*, Vol. 42, No. 4, 816–826, 2025.

[8] Zhou, J., E. Dutkiewicz, R. P. Liu, X. Huang, G. Fang, and Y. Liu, "A modified shuffled frog leaping algorithm for PAPR reduction in OFDM systems," *IEEE Transactions on Broadcasting*, Vol. 61, No. 4, 698–709, 2015.

[9] Li, Q. S., "Analysis of partial randomized phase feeding effects on radiation pattern characteristics of phased array antennas," *Journal of Gannan Normal University*, Vol. 27, No. 6, 7–10, 2006.

[10] Jiang, Y., H. Ge, B.-Y. Wang, S. S. A. Yuan, S.-J. Pan, H. Xu, X. Cui, M.-H. Yung, F. Liu, and W. E. I. Sha, "Quantum-inspired beamforming optimization for quantized phase-only massive MIMO arrays," *ArXiv Preprint ArXiv:2409.19938*, 2024.

[11] Yan, Y., X. Chang, A. K. Roy-Chowdhury, S. Krishnamurthy, A. Swami, and B. Guler, "Clustered federated learning for massive MIMO power allocation," in *MILCOM 2024 — 2024 IEEE Military Communications Conference (MILCOM)*, 1005–1010, Washington, DC, USA, 2024.