ISSN: 2414-1895 DOI: 10.6919/ICJE.202512_11(12).0020

Function Optimization of Two Improved Particle Swarm Algorithms

Xiaoyue Cao, Chonglin Gu*, Dan Luo, Hongze Fu

Liaoning Institute of Science and Technology, Benxi, Liaoning 117004, China

*Corresponding author: Chonglin Gu (Email: 1971103922@qq.com)

Abstract

Aiming at the problem that in the later stage of the traditional Particle Swarm Optimization (PSO) algorithm, the particle velocity is dominated by the cognitive and social components, leading to a decrease in velocity and inertial effect, and making it difficult for particles to escape from the local optimal solution, this paper proposes two improved particle swarm optimization algorithms, namely the Adaptive Weight Particle Swarm Optimization (IPSO) and the Particle Swarm Optimization Combined with Chicken Swarm Optimization (PSOCSO). The specific strategies are as follows: First, a specific crossover operation is introduced to improve the global search efficiency of the solution, which effectively enhances the global search ability and convergence speed of the algorithm. Second, the PSO algorithm is combined with the Chicken swarm Optimization (CSO) to avoid the situation, where particles gather at the local optimal value prematurely and thus lose the ability to explore the entire search space. Through experimental simulations, a comparative analysis of the algorithm performance is conducted from three aspects: solution accuracy, convergence speed, and problem dimension. The results show that the optimized Adaptive Weight Particle Swarm Optimization (IPSO) algorithm exhibits significant advantages in performance compared with the other three contrastive particle swarm optimization algorithms and has stronger robustness.

Keywords

Particle Swarm Optimization (PSO); Adaptive Weight; Differential Evolution (DE) Algorithm; Chicken Swarm Optimization (CSO).

1. Introduction

Particle swarm optimization (PSO), proposed by Professors Kennedy and Eberhart in 1995, is a stochastic evolutionary optimization algorithm based on swarm intelligence [1]. Due to its advantages of requiring fewer parameters to be adjusted, fast convergence speed, and high precision, the PSO algorithm has been quickly accepted and applied in fields such as path planning [7-9], parameter optimization [10], and power systems [5, 6]. After Monika and Sehrawat H proposed the standard particle swarm algorithm, many scholars have conducted theoretical research and improvements on it: Moazami D and Esmaeilbeigi M [2] studied the convergence of the particle swarm algorithm using contraction factors, making the algorithm converge more stably to the optimal solution during the search. Twumasi E [3] et al utilized adaptive inertia weights and learning factors to enhance the algorithm's ability to solve complex optimization problems. Some scholars have combined chaos theory with the improved particle swarm algorithm in an attempt to improve the algorithm's global search ability and optimization efficiency [4], others have integrated quantum mechanics principles

ISSN: 2414-1895 DOI: 10.6919/ICJE.202512_11(12).0020

[5] to change the method of updating particles, thereby achieving a balanced distribution of particles in the search space.

However, for complex multi-objective function optimization problems, the current improved algorithms have some deficiencies in terms of solution efficiency and accuracy. The first algorithm in this paper introduces an adaptive inertia weight, which adjusts the inertia weight according to the stage of the algorithm; the second algorithm (PSOCSO) incorporates the behavior of the chicken swarm optimization (CSO) algorithm, and is "injected" into the PSO particle positions at fixed intervals, which is equivalent to periodically injecting new search momentum into the PSO population, increasing the diversity of particles. The third algorithm (PSOCSOEOBLHM) adds an elite opposition-based strategy to prevent individuals from falling into suboptimal solutions too early, thereby enhancing the flexibility of the entire solution process and the ability to escape local optima traps.

2. Basic Particle Swarm Optimization

First, the PSO algorithm initializes a swarm of particles $X_i = (x_{i1}, x_{i2}, \cdots x_{in})$. During each iteration, each particle updates its own velocity and position based on the personal best (Pbest, the optimal solution historically found by the individual particle) and the global best (gbest, the optimal solution historically found by the entire particle swarm)[11-12]. The update formulas are shown in Equations (1) and (2):

$$v_{id}^{t+1} = w * v_{id}^{t} + c_1 * rand_1 * \left(pbest_{ij}^{t} - x_{ij}^{t}\right) + c_2 * rand_2 * \left(gbest_{i}^{t} - x_{ij}^{t}\right) \quad (1)$$

$$x_{id}^{t+1} = x_{id}^{t} + v_{id}^{t+1} \quad (2)$$

w is the inertia weight, which represents the degree to which a particle inherits its current velocity. A larger w enhances the global search capability, while a smaller w strengthens the local search capability. c_1, c_2 denotes the velocity coefficients, where c_1 is the "cognitive component" (representing the particle's ability to learn from its own historical experience), and c_2 is the "social component" (representing the particle's ability to learn from the collective experience of the swarm). $rand_1$ and $rand_2$ are random numbers uniformly distributed within the interval [0, 1]. t is the current iteration number, and v_{id} is the velocity of the particle at iteration[13-14].

The main problems of the PSO algorithm are that the search speed slows down and the algorithm tends to get trapped in local optima as the number of iterations increases. Among the parameters, w is a critical factor controlling the particle velocity and global search performance, while the learning factors reflect the information exchange between particles and balance the algorithm's capabilities of global exploration and local exploitation. Therefore, improving the inertia weight and learning factors is of great significance for optimizing the PSO algorithm[15-16].

3. Improved Particle Swarm Optimization Algorithm

3.1 IPSO Algorithm

3.1.1 Improvement of Inertia Weight

A method for adaptively changing the inertia weight is proposed. The update formula of the inertia weight is shown in Equation (3).

ISSN: 2414-1895 DOI: 10.6919/ICJE.202512_11(12).0020

$$w^t = w_{max}(1 - \sqrt{\frac{t}{t_{max}}}) \tag{3}$$

In Equation (3),t is the current number of evolutions, t_{max} is the maximum number of evolutions, and w is the maximum inertia weight. Initially, wis at its maximum. As the number of iterations increases, the inertia weight gradually decreases nonlinearly. This nonlinear dynamic adjustment makes the inertia weight related to the particles. It automatically adjusts its value according to the change in the number of particle iterations, thereby improving the intelligence of the particle search process and greatly enhancing the convergence speed of the algorithm.

3.1.2 Introduction of Crossover and Mutation Strategies

In view of the problems of reduced population diversity and proneness to local optimal solutions after introducing nonlinear elements, a crossover mechanism from the Differential Evolution (DE) algorithm is incorporated to enhance the global search performance of the algorithm.

When F is relatively large, the Differential Evolution (DE) algorithm can maintain good population diversity, making it suitable for the initial stage of search. When F is relatively small, the algorithm can achieve fast convergence. In this way, the mutated individual$v_i^t$is obtained.

The crossover operation is as follows: the trial individual $u_i^t = (u_{i1}^t, u_{i2}^t, \cdots, u_{iD}^t)^T$is generated from the mutated individual $v_i^t$and the parent individual x_i^t.

$$u_{ij}^t = \begin{cases} v_{ij}^t & \text{if } rand[0\ \ 1] \le CR \quad \text{or} \quad j = j\_rand \\ x_{ij}^t & \text{if } rand[0\ \ 1] > CR \quad \text{and} \quad j \ne j\_rand \end{cases} \tag{4}$$

$rand$is a random number uniformly distributed in the interval [0, 1].CR is a constant within the range [0, 1], known as the crossover rate. A larger CR's value increases the probability of crossover occurring.j_rand is a randomly selected integer in the range [1, D]. This ensures that the trial individual $u_i$inherits at least one component from the mutated individua v_i.The above mutation and crossover operations are collectively referred to as reproduction operations.

Candidate solutions are generated by combining the mutation strategy of PSO, and the specific update formulas are as follows:

$$x_{id} = x_{r_1 j} + F \bullet (x_{r_2 j} - x_{r_3 j}),\ j = 1, 2, \cdots, D \tag{5}$$

Among them, $i = 1, 2, \cdots, N,$ and $r_1, r_2, r_3 \in \{1, 2, \cdots, N\}$ are three randomly selected individuals. $r_1 \ne r_2 \ne r_3 \ne 1$and F are mutation factors within the range of [0,2], and their values are set to 0.5 here.

3.1.3 Algorithm Flow Description

Step1: Construct a set of m particles, each assigned a randomly generated initial position and velocity;

Step2: Evaluate the fitness of each particle;

Step3: For each particle, compare its fitness with the best position achieved so far. If the former is better, update the particle's current best position record pbest;

Step4: For each particle, compare its fitness value against the best positions achieved by all particles. If superior, adopt it as the current best position gbest;

Step5: When the random number rand is less than the set mutation rate, the system adjusts particle positions using a crossover strategy; otherwise, it performs optimization operations on particle velocity and position according to traditional particle swarm optimization rules.

Step6: If termination conditions are not met, proceed to Step2.

ISSN: 2414-1895 DOI: 10.6919/ICJE.202512_11(12).0020

3.2 PSO-CSO Hybrid Algorithm (PSOCSO)

3.2.1 Chicken Swarms Optimization (CSO) Component

Role Assignment (Based on Fitness Ranking):

Roosters: Top 15%;

Hens: 70%;

Chicks: Remaining 15%;

Role-Specific Behavior:

Roosters: Adjust positions via perturbations with exponential decay, influenced by other roosters;

Hens: Update positions based on their own led roosters and another randomly selected flock member;

Chicks: Follow "Hens," moving toward the hen's position.

3.2.2 Fusion Steps

1) Reassign roles and randomly match hen leaders with chick mothers;

2) Apply CSO behavior formulas to update positions for roosters, hens, and chicks respectively;

3) Perform out-of-boundary corrections for all individual positions;

4) Recalculate fitness values and update individual and global optima. Combining swarm intelligence algorithms:

PSO ensures diversity in overall search and global exploration capability, while CSO enhances local search and inter-individual coordination through role-based behavior and leadership mechanisms.

CSO behaviors are periodically "injected" into PSO particle positions at fixed generation intervals. This periodically infuses the PSO population with new search momentum and structured exploration strategies, improving the algorithm's search performance and convergence quality while preventing local optima traps.

Algorithm Description:

Role Allocation & Matching: Within the CSO submodule, population roles are randomly reassigned. Hen leaders and chick mothers are randomly matched from the hen population to form a hierarchical collaborative structure.

Position Updates: Positions are updated for Roosters (global exploration role), Hens (group collaboration role), and Chicks (follow-and-learn role) based on CSO behavior formulas. Roosters expand the search range through competitive mechanisms, Hens guide group collaboration via leadership, and Chicks perform local fine-search by learning from mother hens.

Boundary Correction: Perform boundary checks on all entities (including PSO particles and CSO role entities). If positions exceed the solution space, correct them according to preset rules to ensure search validity.

Fitness Evaluation and Extreme Value Update: Recalculate the fitness values of all entities. Compare and update the PSO's existing best individual solution (pBest) and global best solution (gBest), while retaining high-quality solutions generated through role collaboration in the CSO.

3.2.3 PSOCSO Algorithm Flow

Step1: Set parameters such as population size, iteration count, and solution space boundaries; randomly generate initial positions for PSO particles and CSO roles (Rooster, Hen, Chick);

Step2: Hen group processing: randomly select some hens as leaders and chick mothers to form a hierarchical structure: Hen Leader → Hen → Chick;

Step3: Position update:

3.1: Rooster (global exploration) updates position via competitive mechanism: expands search range;

$$\text{Formula:} \quad X_R^{t+1} = X_R^t + \alpha \cdot \left(X_{gBest}^t + X_R^t\right) + \beta \cdot RD \tag{6}$$

ISSN: 2414-1895 DOI: 10.6919/ICJE.202512_11(12).0020

Equation (6) represents a parameter related to hen following behavior in the chicken flock algorithm. It controls the step size or weight governing a hen's movement toward the rooster and its own historical optimal position. This parameter determines the extent to which hens are influenced by the rooster and their own experience during the search process; Typically associated with the rooster's position update, it may serve as a modulation factor controlling the amplitude by which the rooster adjusts its current position based on its historical optimal position and the global optimal position, reflecting the rooster's ability to explore new regions within the search space.

3.2: Hen (Group Collaboration), guided by a leadership mechanism, hens update positions by following the leader;

$$\text{Formula:} \qquad X_H^{t+1} = X_R^t + \gamma \cdot (X_{pBest}^t + X_H^t) \tag{7}$$

Equation (7) serves as the weighting coefficient for different components within the fitness function. For instance, in hybrid algorithms, the fitness function may combine features of PSO and CSO, adjusting the contribution ratios of each component to coordinate the optimization directions of both algorithms.

3.3: Chicks (Follow-Learning), learning local search from the mother hen;

$$\text{Formula:} \qquad X_C^{t+1} = X_H^t + \delta \cdot (X_H^t - X_{HR}^t) \tag{8}$$

Equation (8) represents a parameter related to chick behavior in the flock algorithm, controlling the degree to which chicks adjust based on the mother hen's position and their own location. This reflects the chicks' learning and following capabilities within the group.

Step 4: Boundary Correction. Check if any individual's position exceeds the solution space boundary. If so, correct according to rules;

Step5: Fitness Evaluation and Extreme Value Update. Calculate the fitness values for all individuals. For the PSO component: Update the individual best solution (pBest) and global best solution (gBest). For the CSO component: Retain high-quality solutions generated through role collaboration.

Step6: Termination Condition Check. If the maximum iteration count is reached or convergence conditions are satisfied, terminate the algorithm. Otherwise, return to Step2 to continue iteration.

4. Function Selection for Testing

To comprehensively and accurately evaluate the performance of the improved particle swarm optimization algorithm in function optimization, this paper selects four standard functions for testing. These functions encompass unimodal, multimodal, and high-dimensional cases, examining the number and characteristics of their global and local optima to reveal profound correlations between complexity and dimensionality.

The four classic multimodal test functions are as follows:

The first objective function, the Ackley function:

$$f_1(x) = -20\left(\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\right) - exp\left(\frac{1}{n}\sum_{i=1}^{n}\cos(2\pi x_i)\right) \tag{9}$$

ISSN: 2414-1895 DOI: 10.6919/ICJE.202512_11(12).0020

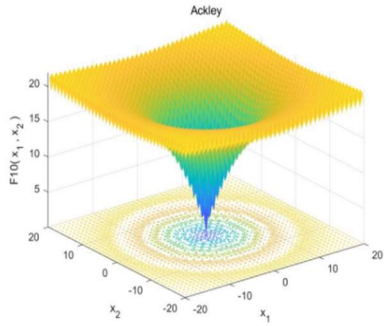

Figure 1. Simulated image of the Ackley function

The second objective function, the Griewanck function:

$$f_2(x) = \frac{1}{400}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} cos\left(\frac{x_i}{\sqrt{i}}\right) + 1 \tag{10}$$

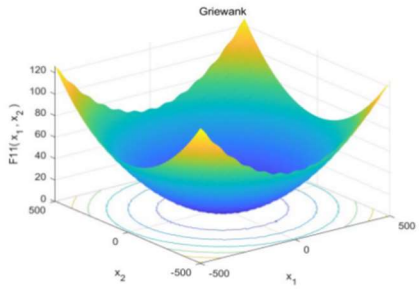

Figure 2. Simulated image of the Griewanck function

The third objective function: Schaffer function:

$$f_3(x) = 0.5 - \frac{\left(sin\sqrt{x_1^2+x_2^2}\right)^2 - 0.5}{\left(1+0.001(x_1^2+x_2^2)\right)} \tag{11}$$

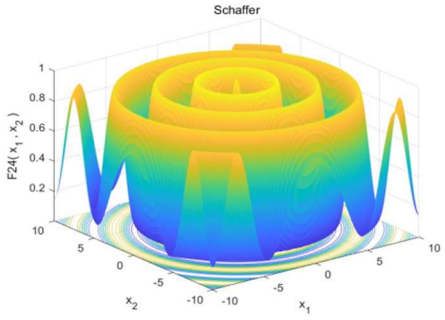

Figure 3. Simulated image of the Schaffer function

Fourth objective function: Sphere function:

$$f_4(x) = \sum_{i=1}^{10} x_i^2 \tag{12}$$

ISSN: 2414-1895 DOI: 10.6919/ICJE.202512_11(12).0020

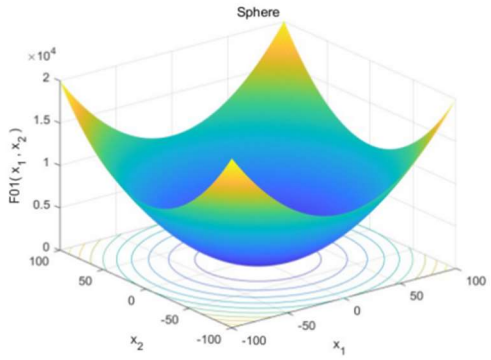

Figure 4. Simulated image of the Sphere function

5. Simulation Testing and Results Analysis

MATLAB simulations were conducted to compare the standard PSO algorithm, the particle swarm optimization (PSO) algorithm fused with the chicken swarm optimization (CSO) algorithm (PSOCSO), the adaptive inertia weight particle swarm optimization (IPSO) algorithm, and the elite backward strategy enhanced and hybrid mutation strategy-improved fused chicken swarm particle swarm optimization algorithm (PSOCSOEOBLHM). All algorithms employed a population size of 30. The inertia factor was set to 0.5 for the standard PSO algorithm. For the IPSO algorithm, the inertia factor decreased nonlinearly from 1.05 to 0.05. Each algorithm underwent a maximum of 2000 iterations. Each algorithm was randomly run 10 times on the test functions. The following figures present the evolutionary convergence curves for the four functions.

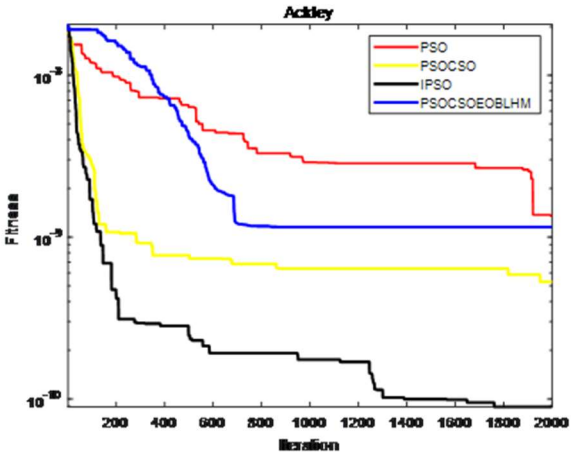

Figure 5. Evolutionary Convergence Curve of the Function

Figure 5 illustrates the iterative convergence performance of different particle swarm optimization algorithms on the Ackley function. From the curve trends, the PSOCSOEOBLHM algorithm exhibits a relatively rapid decline in fitness, achieving lower fitness values at earlier iteration counts; followed by the IPSO algorithm; the PSO and PSOCSO algorithms exhibit a relatively gradual decline in fitness, maintaining higher fitness values at the same iteration count. The IPSO algorithm found a local optimum at iteration 600 and a global optimum at iteration 1200; PSOCSO found the optimal solution at iteration 400, but its solution value did not match the performance achieved by IPSO; PSOCSOEOBLHM found the global optimum around iteration 700. This figure demonstrates that the IPSO algorithm exhibits superior convergence speed and optimization accuracy compared to other particle swarm algorithms in the Ackley function optimization task.

ISSN: 2414-1895 DOI: 10.6919/ICJE.202512_11(12).0020

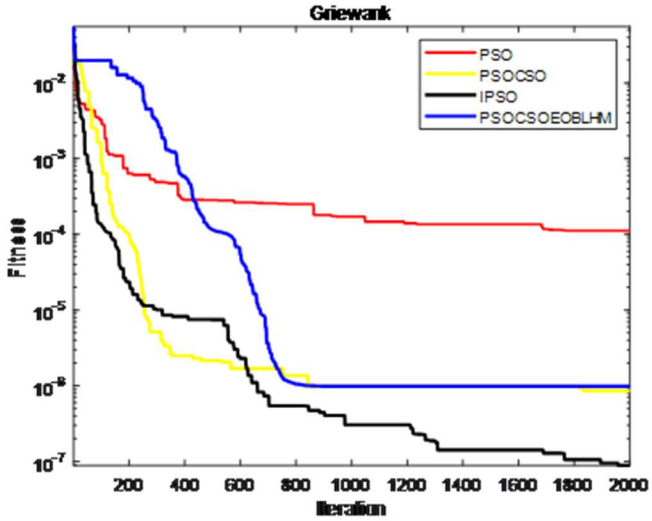

Figure 6. Evolutionary Convergence Curve of Function

Figure 6 illustrates the iterative convergence of different particle swarm optimization algorithms on the Griewank function. As the number of iterations increases, the fitness values corresponding to each algorithm exhibit a downward trend. Among them, the PSO algorithm's fitness value decreases relatively slowly, stabilizing at a higher value around iteration 400; the PSOCSO algorithm's fitness value decreases at a slightly faster rate, stabilizing at a certain value around iteration 900; The IPSO algorithm exhibits a more pronounced decrease, reaching a lower value than the previous two at iteration 1000. The PSOCSOEOBLHM algorithm shows the fastest rate of decline, rapidly decreasing throughout the iterations and stabilizing at the lowest value by iteration 800.

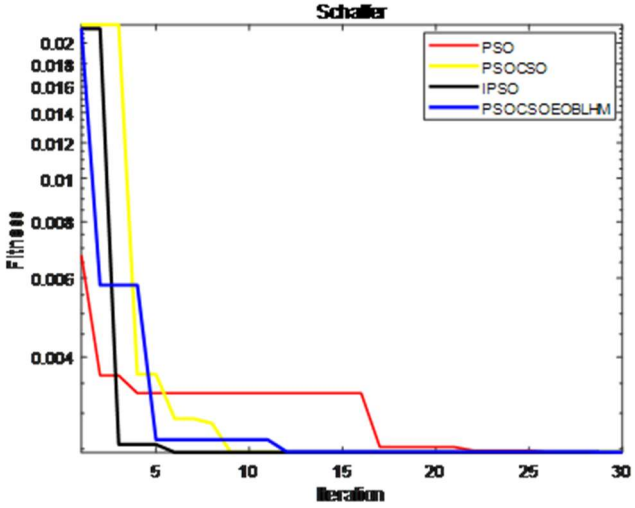

Figure 7. Evolutionary Convergence Curve of Function

In Figure 7, it is evident that the IPSO algorithm exhibits a rapid decline in fitness during the early stages, stabilizing at a low value after approximately 10 iterations, demonstrating superior convergence speed and accuracy. In contrast, the PSO algorithm converges relatively later, approaching stability only in the later stages of iteration. PSOCSQ and PSOCSOEOBLHM exhibit convergence performance intermediate between the two. Overall, this reflects the divergent convergence characteristics of different improved particle swarm optimization algorithms when optimizing the f3 function. IPSO demonstrates superior optimization efficiency and effectiveness in this test scenario.

ISSN: 2414-1895 DOI: 10.6919/ICJE.202512_11(12).0020

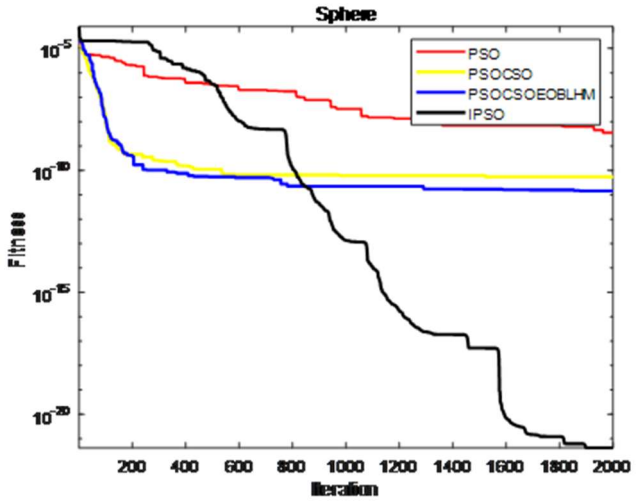

Figure 8. Evolutionary Curve of Function

Figure 8 displays the iterative convergence curves on the Sphere function. As the iteration count increases from 0 to 2000, different curves exhibit distinct trends, reflecting changes in the fitness values of corresponding algorithms on the Sphere function. Notably, the IPSO algorithm demonstrates a rapid decline in fitness during later iterations, yet achieves relatively superior optimization performance compared to other algorithms. These curves enable a comparative analysis of the optimization capabilities across different algorithms.

6. Conclusion

The experimental results clearly demonstrate that the improved particle swarm optimization algorithm achieves higher fitness values faster than the traditional version under identical computational resources, requiring fewer iterations and exhibiting superior optimization performance. Regarding search accuracy, it effectively avoids getting stuck in local optima and more accurately locates global optima. During the optimization of the Schaffer function and the Griewanck function, the results converged upon by the improved particle swarm algorithm were closer to the theoretical optimal values. In terms of global search capability, the improved particle swarm algorithm performs well in optimizing high-dimensional complex functions, capable of searching within complex solution spaces to find relatively high-quality solutions.

References

[1] Eberhart, R.C. and Kennedy, J. (1995) A New Optimizer Using Particle Swarm Theory. Proceedings of the Sixth International Symposium on Micro Machine and Human Science, Nagoya, 39-43.

[2] Monika, Sehrawat H. Multi-parametric and priority driven particle swarm (MPPPSO) optimized task scheduling approach for improving performance of fog computing system[J]. Progress in Artificial Intelligence, 2025, (prepublish): 1-18.

[3] Moazami D, Esmaeilbeigi M. Enhanced stability and accuracy in solving nonlinear Fredholm integral equations using hybrid radial kernels and particle swarm optimization[J]. Computational and Applied Mathematics, 2024, 44(1): 78-78.

[4] Twumasi E, Frimpong A E, II P K N, et al. A novel improvement of particle swarm optimization using an improved velocity update function based on local best murmuration particle[J]. Journal of Electrical Systems and Information Technology, 2024, 11(1): 42-42.

[5] J. Kennedy and R. Eberhart. Particle swarm optimization. In Proceedings of the IEEE International Conference on Neural Networks, pages , Piscataway, 1995.

[6] Purnamaasri S W H, Dennis W, Mohsin M. Abstract 4340723: Comparative Study of Coronary Artery Disease (CAD) Prediction: Conventional QRISK3 vs. Enhanced Machine Learning Models Combined

ISSN: 2414-1895 DOI: 10.6919/ICJE.202512_11(12).0020

with Particle Swarm Optimization (PSO) Algorithm[J]. Circulation, 2025,152(Suppl_3): A4340723-A4340723.

[7] Rajeswari S, Rathika P. Refining Hyperparameters of SVM for Enhancing Waterbody Classification in Satellite Imagery[J]. Journal of the Indian Society of Remote Sensing, 2025, (prepublish): 1-17.

[8] Crnkić A, Kapić Z. Simulation-based optimization of capital greenhouse gas emissions in water distribution systems using Particle Swarm Optimization[J]. IOP Conference Series: Materials Science and Engineering,2025,1339(1):012009-012009.

[9] J T A, R S, Chandramani V P. Optimization of fourth order noise transfer function using PSO algorithm for delta sigma modulator[J]. Integration, 2026,106102539-102539.

[10] Pathan M, Annapurna K. Particle Swarm Optimization–Based Multimetric Route Optimization Towards Quality of Service Enhancement in Vehicular Ad Hoc Networks[J]. International Journal of Communication Systems,2025,38(16):e70266-e70266.

[11] EL-Qasery M, Abbou A, Laamim M, et al. Comparative analysis of GA and PSO algorithms for optimal cost management in on-grid microgrid energy systems with PV-battery integration[J]. Global Energy Interconnection, 2025, 8(04):572-580.

[12] Moazami D, Esmaeilbeigi M, Akbari T. Hybrid radial kernels for solving weakly singular Fredholm integral equations: Balancing accuracy and stability in meshless methods[J]. Journal of Computational and Applied Mathematics,2026,473116848-116848.

[13] Deevi P D, Kodadi S,Allur S N, et al.Improvement and application of particle swarm optimization algorithm[J]. Intelligent Decision Technologies,2025,19(4):2347-2367.

[14] Kabemba M A, Mutombo K, Waters E K. A Predictive Geometallurgical Framework for Flotation Kinetics in Complexes Platinum Group Metal Orebodies: Mode of Occurrence-Based Modification of the Kelsall Model Using Particle Swarm Optimization[J]. Minerals, 2025,15(7):701-701.

[15] Min G, So J. Energy-Efficient Deployment Simulator of UAV-Mounted Base Stations Under Dynamic Weather Conditions[J]. Sensors, 2025,25(12):3648-3648.

[16] Asabere P, Sekyere F, Ayambire P, et al.Optimal capacitor bank placement and sizing using particle swarm optimization for power loss minimization in distribution network[J]. Journal of Engineering Research,2025,13(2):1307-1315.