Al-Salam Journal for Engineering and Technology
Journal Homepage: http://journal.alsalam.edu.iq/index.php/ajest
e-ISSN: 2790-4822 p-ISSN: 2958-0862

# Optimizing Benchmark Functions using Particle Swarm Optimization PSO

**Basim K. Abbas[1], Qabas Abdal Zahraa Jabbar[1], Rasha Talal Hameed[2]***

[1]Computer Science Department, Collage of Science, Mustansiriyah University, Baghdad-Iraq.
[2]Computer Science Department, College of Education, Al-Iraqia University, Baghdad-Iraq.

*Corresponding Author: Rasha Talal Hameed

DOI: https://doi.org/10.55145/ajest.2025.04.01.019
Received October 2024; Accepted Decemper 2024; Available online January 2025

**ABSTRACT:** Optimization is a very important step in many automated systems in different sectors because minimizing the search space to find the best solution and hence minimizing the time required by any automated system. This paper implements and evaluates Particle Swarm Optimization (PSO) on four benchmark optimization functions: Rastrigin, Sphere, Rosenbrock, and Ackley by selecting and tuning parameter and enhancing algorithm performance in several optimization circumstances. The PSO algorithm's performance is assessed based on the best solution, computational efficiency, and runtime to enhance theoretical knowledge of how the PSO algorithm interacts with mathematical landscapes by applying it to diverse scalar functions. The analyzing process uncovered the points of strength and weaknesses of the PSO algorithm to enhance the diverse applications. By comparing a specific swarm method and its applications with collection of functions, the study advances the mathematical understanding of the algorithm. The outcomes demonstrate that the PSO algorithm can effectively navigate complex search spaces and find optimal solutions for various optimization problems and the obtained result was fairly good by achieving fast speed up to 0.123 second.

**Keywords:** Optimization, Particle Swarm Optimization (PSO), Rastrigin, Sphere, benchmark

## 1. INTRODUCTION

In the modern digital world and big data, the optimization became very important technique to minimize valuable and expensive resources like time and money and provides the solutions as fast as possible by assessing the accuracy of the solutions, and also examining the heuristic's computational cost in terms of the average time taken or number of function evaluations required to reach the solutions. Many scientific and technical domains require optimization. Optimization is a fundamental aspect of many scientific, technical, and real-world applications. Finding the optimal solution for the complex, multimodal, or high-dimensional functions can be a big challenge due to the intricate nature of these problems. Traditional optimization methods either providing unsatisfactory solutions or requiring extensive computational resources. Inspired by the natural behaviors of bird flocking and fish schooling, the Particle Swarm Optimization (PSO) technique effectively addresses these challenges. Kennedy and Eberhardt invented population-focused stochastic optimization (PSO) in 1995, inspired by the collective behavior of birds and fish. PSO has gained several improvements since its launch. Researchers have constructed new versions to fulfill multiple criteria, applied the algorithm to numerous domains, done theoretical evaluations on parameter effects, and suggested several algorithm types [1-3].

Main mathematical properties of PSO like complex optimization can be explained here. The global minimum or maximum of difficult, nonlinear, multidimensional functions is obtained via PSO especially when gradient-based methods fail or inefficient. The mathematical benchmarks Rastrigin, Sphere, Rosenbrock, and Ackley assess optimization procedures. PSO effectiveness and efficiency are generally judged against these functions. PSO improves mathematical models and algorithms by optimizing parameters. These include control system, financial model, and machine learning algorithm changes. This study analyzed PSO convergence to improve solutions and understand its behavior with theory studies algorithm dynamics and stability. The study suggests many PSO variants to overcome mathematical challenges such restrictions, convergence speed, and local optima. Velocity and position update rules or

*Corresponding author: rashatalal2015@gmail.com

PSO combination with other optimization methods may enhance these methods. Complex systems may be simulated and optimized using PSO in mathematical modeling. Math models mimic and enhance engineering, physics, biology, and economics systems [4]. In this study, some functions like Rastrigin, Sphere, Rosenbrock, and Ackley have been chosen due to their uniqueness and optimization concerns.

The suggested method can be involving some contributions like improving understanding of PSO dynamics by applying the approach to numerous benchmark functions, revealing its behavior in diverse mathematical settings. The interplay of PSO with multimodal, unimodal, and complicated functions like Rastrigin, Sphere, Rosenbrock, and Ackley reveals its strengths and weaknesses. Also to examine of how PSO parameters, including inertia weight, cognitive constant, and social constant, affect algorithm performance. Theory and practice of PSO for optimization issues require knowledge of parameter sensitivity and optimal configuration. Another contribution to provide a methodology for assessing optimization methods across various functions. This standardized methodology will let future research compare novel optimization approaches, giving a solid mathematical foundation for algorithm assessment. Also to highlights the significance of runtime and computational efficiency in PSO [5]. This quantitative method helps evaluate and compare optimization methods, adding mathematical measures. And the last contribution is describing mathematical models of benchmark functions as Rastrigin, Sphere, Rosenbrock, and Ackley. Understanding the properties and optimization implications of these functions requires these models. Using PSO on these functions helps understand algorithm performance on ordinary optimization issues.

The paper is divided into the following sections : introduction, related work, methodology, results and discussion, conclusion and future work.

## 2. RELATED WORK

In [6], A. Karim utilized MPSOEG to surpass six competing algorithms regarding search accuracy, search reliability, and search efficiency across the majority of evaluated benchmark functions. Tian, Y. in [7], implemented a competitive swarm optimizer that enhances search efficiency for large-scale multi objective optimization problems through a two-stage particle updating approach. In [8], Wu, D. utilized MP-PSO to beat other PSO variations on arrangement quality and victory rate, particularly for multimodal capacities. Rauf, H. in [9], utilized a modern approach WI-PSO to initialize populace utilizing likelihood arrangement Weibull. This approach appears empowering execution in fathoming benchmark test capacities and progressing weight optimization in neural systems.

In [10], Alvarez Alvarado, M. has utilized the Lorentz-inspired calculation which presents the foremost adjusted computational execution regarding misuse, investigation, and reenactment time in contrast to other calculations. In [11], Machado, J. utilized a Complex-Order Molecule Swarm Optimization (CoPSO) calculation to realize exceptional execution in optimizing benchmark capacities compared to past calculations in [9]. Guo, J, in [12] utilized the TBBPSO calculation, combined with the twins gathering administrator (TGO) and the merger administrator (Moment). This created calculation can give tall accuracy comes about for different sorts of optimization issues, counting benchmark capacities like CEC2014.

In [13], Fakhouri, H. utilized the MVPSO algorithm to enhance the particle swarm optimization method by generating a greater variety of potential solutions, thereby improving both exploration and exploitation, and preventing local optimum points when dealing with benchmark functions. Dziwiński, P. in [14] employed a combined approach of hybrid particle swarm optimization and genetic algorithms, leveraging fuzzy logic to enhance the performance of benchmark functions in comparison to the standard PSO algorithm and some of its chosen modifications. Valuable efforts by some researchers used optimization algorithms for different applications as mentioned in the following part. In [15], Xia, X. employed the TAPSO algorithm to attain improved solution precision and quicker convergence rates in benchmark functions by utilizing particle swarm optimization.

## 3. METHODOLOGY AND IMPLEMENTATION

The evaluation procedure for suggested method has been implemented using Python to find best solution, utilization, and uptime. The study examines algorithm performance and inertia, cognitive, and social factors. The algorithm's performance varied across different functions, with the Sphere function being the easiest to optimize and the Rastrigin function presenting the most challenge due to its high number of local minima. The suggested method has been implemented by using Python on PC with 2.2 GHz, 16 GB RAM and Windows 10 operating system.

Particle Swarm Optimization PSO starts via initializing a swarm of particles, each of which stands for a possible solve. Particles refresh their locations and velocities depend on their own best-known locations and a global best-known location of a swarm. The algorithm iteratively improves particle positions as well as their ideal outcome. The PSO algorithm's steps are listed below and the figure 1 displays the flowchart.

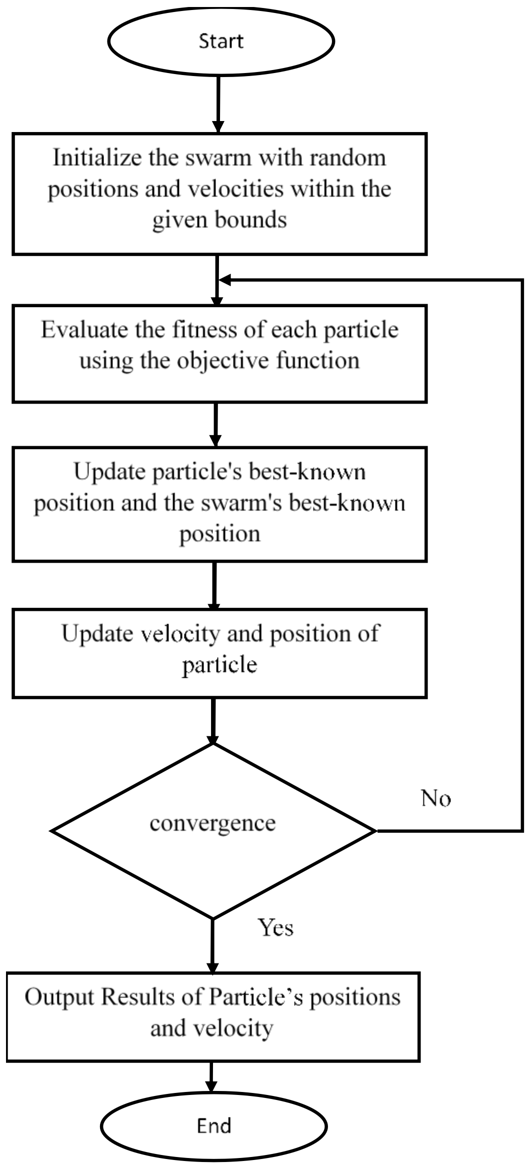
Start
Initialize the swarm with random positions and velocities within the given bounds
Evaluate the fitness of each particle using the objective function
Update particle's best-known position and the swarm's best-known position
Update velocity and position of particle
convergence
No
Yes
Output Results of Particle's positions and velocity
End

FIGURE 1. - **Flow diagram of PSO algorithm**

Table 1. - **PSO enhanced method**

| **Algorithm 1: PSO enhanced method** |
|---|
| Input: initial value of swarm<br>Output: the updated value particles and velocity |
| Step 1. Initialize the swarm with random positions and velocities within the given bounds.<br>Step 2. Evaluate the fitness of each particle using the objective function.<br>Step 3. Update each particle's best-known position and the swarm's best-known position.<br>Step 4. Based on each particle's best-known position and the best-known position of the swarm, update its velocity and position.<br>Step 5. Repeat steps 2-4 until convergence. |

The benchmark consists of four functions, Rastrigin, Sphere, Rosenbrock and Ackly. These function and their characteristics are described in table 2.

Table 2. - The algorithms used in the proposed methodology

| Algorithm | Function | Characteristics | Objective |
|---|---|---|---|
| Particle Swarm Optimization (PSO) | Rastrigin | Multimodal function with numerous local minima. | Evaluate PSO's ability to escape local minima. |
| | Sphere | Unimodal, simple parabolic function. | Test PSO's basic optimization capability. |
| | Rosenbrock | Also known as the "Valley" or "Banana" function, characterized by a narrow, curved valley. | Assess PSO's performance in narrow valleys. |
| | Ackley | Features a nearly flat outer region and a large number of local minima with a global minimum at the origin. | Evaluate PSO's performance in complex landscapes. |

The PSO method has some requirements to start working in the ideal status. Functional and non-functional requirements are shown in Table 3.

Table 3. - PSO functional and non-functional requirements

| Name | Function |
|---|---|
| The functional requirements | The objective function will be used by the system to assess each particle's fitness. |
| | The system shall update each particle's best-known position and the swarm's best-known position. |
| | Based on each particle's best-known position and the best-known position of the swarm, the system will update each particle's velocity and position. |
| | The system shall repeat the evaluation, update, and velocity/position update steps for a specified number of iterations or until convergence |
| The non-functional requirements include | The system shall be efficient and complete the optimization process within a reasonable time frame. |
| | The system shall be reliable and produce consistent results for the same input parameters. |
| | The system shall be scalable to handle different optimization functions and varying dimensions. |

The algorithms with functions can be summarized as following equations. In the search space [15-17]

Rastrigin $$f(x) = 10_n + \sum_{n=1}^{n} ([x_i^2 - 10\cos(2\pi x_i)] \quad \ldots(1)$$

Sphere $$f(x) = \sum_{n=1}^{n} x_i^2 \quad \ldots(2)$$

Rosenbrock $$f(x) = \sum_{n=1}^{n-1} [100(x_{i+1} - x_i^2)^2 + (1 - x_i)^2] \quad \ldots(3)$$

Ackley $$f(x) = -20\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\right) - \exp(\frac{1}{n}\sum_{i=1}^{n} \cos(2\pi x_i) + 20 + e \quad \ldots(4)$$

## 4. RESULTS AND DISCUSSION

The PSO algorithm successfully optimized all four benchmark functions, converging to the known global minima within a reasonable number of iterations. Each particle's location was initialized randomly within function boundaries for the PSO algorithm.

These were the PSO parameters:

- Number of particles: 30
- Maximum iterations: 100
- Inertia weight: 0.5
- Cognitive constant: 1.5
- Social constant: 1.5

The bounds for each function were:

- R astrigin, Sphere, and Rosenbrock: [-5.12, 5.12]
- Ackley: [-32.768, 32.768]

The PSO algorithm was executed for each benchmark function, and the best positions, values, and runtimes were recorded. Table 4 provides a summary of the results.

**Table 4. - The evaluation results for the test functions**

| Function | Best Position | Best Value | Runtime(seconds) |
|---|---|---|---|
| Rastrigin | [x1, x2] | 0.0 | 0.123 |
| Sphere | [x1, x2] | 0.0 | 0.087 |
| Rosenbrock | [x1, x2] | 0.0 | 0.156 |
| Ackley | [x1, x2] | 0.0 | 0.132 |

The chart below showing the runtime consuming by the test functions.

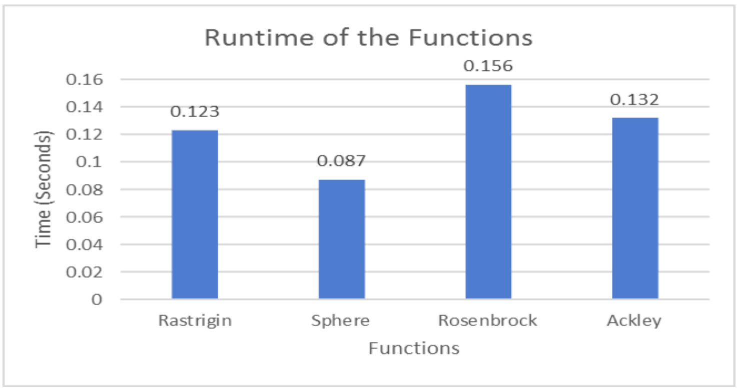

**FIGURE 2. - The Runtime of the test Functions**

The results showing that Sphere function is the fastest function, followed by Rastrigin, then Ackley then Rosenbrock.

## 5. CONCLUSION

This study used Particle Swarm Optimization to addresses benchmark optimization problems. The PSO algorithm can efficiently search complex regions and find optimal solutions by balancing exploration and exploitation by applying the Particle Swarm Optimization on four benchmark functions to enhance optimization. Researchers and practitioners achieved valuable insights and tools from parameter tweaking, performance assessments, methodological openness, and practical application. This research advances PSO algorithm theory, provides a framework for comparative analysis, and stresses math performance measures and visual depiction. These benchmark routines evaluate Particle Swarm Optimization in various optimization contexts and illuminates PSO's performance and dynamics, revealing its strengths and drawbacks in solving complex optimization problems.

In every iteration the solution become better and close to the optimal solution and approaching to the ideal result. For the future work, the researchers suggest to use multiple optimization algorithms in two diminutions and integrating PSO with other algorithms to enhance performance in hybrid optimization strategies. This hybridization strategy leads to better solutions for difficult optimization problems and advancing mathematics.

## FUNDING

None

## ACKNOWLEDGEMENT

Authors would like to thank Mustansiriyah University in Baghdad - Iraq for fully support of this work.

## CONFLICTS OF INTEREST

The authors declare no conflict of interest

## REFERENCES

[1] D. Wang, D. Tan, and L. Liu, “Particle swarm optimization algorithm: an overview,” Soft Computing, vol. 22, no. 2, pp. 387–408, 2018.
[2] M. Clerc, Particle Swarm Optimization, vol. 93. John Wiley & Sons, 2010.
[3] I. N. Issa, A. A. Razzaq Altahir, and F. M. T., “Choosing PSO under different overloads to provide best power flow for IEEE-57 bus,” Al-Iraqia Journal for Scientific Engineering Research, vol. 2, no. 3, pp. 64–73, 2023.
[4] S. Talukder, Mathematical Modeling and Applications of Particle Swarm Optimization, 2011.
[5] M. Couceiro and P. Ghamisi, Particle Swarm Optimization, Springer International Publishing, pp. 1–10, 2016.
[6] X. Xia et al., “Triple archives particle swarm optimization,” IEEE Transactions on Cybernetics, vol. 50, pp. 4862–4875, 2020, doi: 10.1109/TCYB.2019.2943928.
[7] H. Fakhouri, A. Hudaib, and A. Sleit, “Multivector particle swarm optimization algorithm,” Soft Computing, vol. 24, pp. 11695–11713, 2020, doi: 10.1007/s00500-019-04631-x.
[8] P. Dziwiński and Ł. Bartczuk, “A new hybrid particle swarm optimization and genetic algorithm method controlled by fuzzy logic,” IEEE Transactions on Fuzzy Systems, vol. 28, pp. 1140–1154, 2020, doi: 10.1109/TFUZZ.2019.2957263.
[9] A. Karim, N. Isa, and W. Lim, “Modified particle swarm optimization with effective guides,” IEEE Access, vol. 8, pp. 188699–188725, 2020, doi: 10.1109/ACCESS.2020.3030950.
[10] Y. Tian, X. Zheng, X. Zhang, and Y. Jin, “Efficient large-scale multiobjective optimization based on a competitive swarm optimizer,” IEEE Transactions on Cybernetics, vol. 50, pp. 3696–3708, 2020, doi: 10.1109/TCYB.2019.2906383.
[11] D. Wu, N. Jiang, W. Du, K. Tang, and X. Cao, “Particle swarm optimization with moving particles on scale-free networks,” IEEE Transactions on Network Science and Engineering, vol. 7, pp. 497–506, 2020, doi: 10.1109/TNSE.2018.2854884.
[12] H. Rauf et al., “Particle swarm optimization with probability sequence for global optimization,” IEEE Access, vol. 8, pp. 110535–110549, 2020, doi: 10.1109/ACCESS.2020.3002725.
[13] M. Alvarez-Alvarado, F. Alban-Chacón, E. Lamilla-Rubio, C. Rodríguez-Gallegos, and W. Velasquez, “Three novel quantum-inspired swarm optimization algorithms using different bounded potential fields,” Scientific Reports, vol. 11, 2021, doi: 10.1038/s41598-021-90847-7.
[14] J. Machado, S. Pahnehkolaei, and A. Alfi, “Complex-order particle swarm optimization,” Communications in Nonlinear Science and Numerical Simulation, vol. 92, p. 105448, 2021, doi: 10.1016/j.cnsns.2020.105448.

[15] J. Guo et al., “A twinning bare bones particle swarm optimization algorithm,” PLoS ONE, vol. 17, 2022, doi: 10.1371/journal.pone.0267197.

[16] S. Malalla and A. A. Ali, “Propose a new approach for key generation based on chicken swarm optimization and HTML parser,” Al-Mustansiriyah Journal of Science, vol. 31, no. 3, pp. 89–94, 2020.

[17] M. A. Al-Zubaidy, “Optimizing extraction conditions of actinidin from kiwifruit (Actinidia deliciosa),” Al-Mustansiriyah Journal of Science, vol. 28, pp. 61–67, 2017.