THE EFFECTIVENESS OF PSO IN SOLVING GROUPS OF CONVEX AND NON-CONVEX FUNCTIONS

BLERINA BOÇI[1], AURORA SIMONI[2]

[1]Department of Mathematics, Faculty of Information Technology, "Aleksandër Moisiu" University, Durrës Albania

[2]Department of Applied Mathematics, Faculty of Natural Science, University of Tirana, Albania

email: blerinaboci@yahoo.com

Abstract

This study investigates the effectiveness of Particle Swarm Optimization (PSO) in solving groups of convex and non-convex functions for global optimization problems. PSO is a popular nature inspired optimization algorithm known for its simplicity and efficiency. We evaluate its performance in tackling a variety of optimization problems, including both convex and non-convex objective functions. Global optimization problems, which involve the search for optimal solutions over a wide range of possible inputs, are ubiquitous in various fields, including engineering, economics, and science. Particle Swarm Optimization (PSO), and its technique, has gained attention for its simplicity and remarkable performance in various applications. This research aims to investigate the effectiveness of PSO in addressing global optimization problems that encompass groups of both convex and non-convex functions.

Key words: *PSO, algorithm, convex, non-convex, parameters.*

Përmbledhje

Ky studim hulumton efikasitetin e algoritmit (PSO) në zgjidhjen e grupeve të funksioneve konvekse dhe jo-konvekse për probleme të optimizimit global. PSO është një algoritëm i njohur për optimizim, i inspiruar nga natyra dhe i njohur për thjeshtësinë dhe efikasitetin e tij. Ne vlerësojmë performancën e tij në adresimin e një sërë problemesh optimizimi, duke përfshirë funksione objektive të të dy llojeve, konvekse dhe jo-konvekse. Problemet e optimizimit global, që përfshijnë kërkimin e zgjidhjeve optimale në një gamë të gjerë të mundshme të të dhënave, janë të shpeshta në fusha të ndryshme, përfshirë inxhinieri, ekonomi dhe shkencë. Algoritmi PSO dhe teknika e tij, ka tërhequr *vëmendjen për thjeshtësinë dhe performancën e tij të jashtëzakonshme në aplikime të ndryshme. Ky kërkim synon të hulumtojë efikasitetin e PSO në adresimin e problemeve të optimizimit global që përfshijnë grupe të të dy llojeve të funksioneve, konvekse dhe jo-konvekse.*

Fjalë kyçe: *PSO, funksion, konveks, jo konveks, parametra.*

Introduction

Finding global optima in optimization problems is indeed a complex and challenging task, especially when dealing with non-convex objective functions. In this context, optimization algorithms play a critical role in providing efficient and effective solutions (Heitzinger, 2022). The objective of this study is to evaluate the performance of PSO in solving global optimization problems with convex and non-convex functions (Boyd, 2004), assessing its ability to efficiently locate global optima in well-behaved landscapes. By exploring how PSO performs in global optimization problems involving both convex and non-convex functions, this research contributes to a deeper understanding of the algorithm's applicability in real-world scenarios (Trehan, Singh 2020), (Yangand, Liu 2019). The findings of this study are expected to offer valuable insights into the role of PSO in solving global optimization problems and its potential to uncover global optima across a spectrum of objective function landscapes.

Particle Swarm Optimization

PSO is a stochastic search algorithm inspired by the social-psychological behavior of biological entities in nature (developed by Erberhart and Kennedy 1995), when they are in search of culture resources (bird flocking or fishing). Similar to the paradigm of "Swarm Intelligence" (Dorigo, 1999), PSO abstracts this reality through correspondences in one table each, thus defining a mathematical model that tries to simulate the dynamics of it. searching for resources, individually and thanks to the influence of the herd. The dynamic adaptation of these entities is reflected mathematically through the use of some stochastic elements, which will favor the movement towards better areas, of things in the environment in the past, and will open up the optimal convert.

A. Basic Concept of Particle Swarm Optimization

PSO is inspired by the social behavior of birds and fish and has similarities to other evolutionary algorithms in its approach to finding optimal solutions through the exploration and exploitation of the search space. Particles fly around the multidimensional search space. During flight, each particle adjusts its position according to its own experience and according to experience of a neighboring particle, making use of the best position encountered by itself and its neighbor.

The position and velocity of the i-th particle are represented as the vectors

$X_i = (x_{i1}, x_{i2}, \dots, x_{id})$ and $V_i = (v_{i1}, v_{i2}, \dots, v_{id})$ respectively.

The previous best position of the i-th particle is recorded and represented as $Pbest_i$. The index of best particle among all the particles in the group is represented by the $Gbest_d$.

The modified velocity of each particle can be calculated using the current velocity and the distance from $Pbest_{id}$ to $Gbest_d$ as shown below:

$$V_{id}^{k+1} = w * V_{id}^{k} + C_1 * rand_1(\,) * \left(Pbest_{id} - X_{id}^{k}\right) + C_2 * rand_2(\,) * \left(Gbest_d - X_{id}^{k}\right)$$

$i = 1,2,\dots,N_p \qquad d = 1,2,\dots,\ Ng.$ (1)

$rand_1()$, $rand_2()$ are uniform random numbers in the range [0,1] (Kennedy et al., 1995).

As we see in (1) by combining the components discussed above, a new velocity vector is obtained, which orients the position of the particle towards a new position X_{id}^{k+1} in the search space.

The updated velocity can be used to change the position of each particle in the swarm as:

$X_{id}^{k+1} = X_{id}^{k} + V_{id}^{k+1}$ (2)

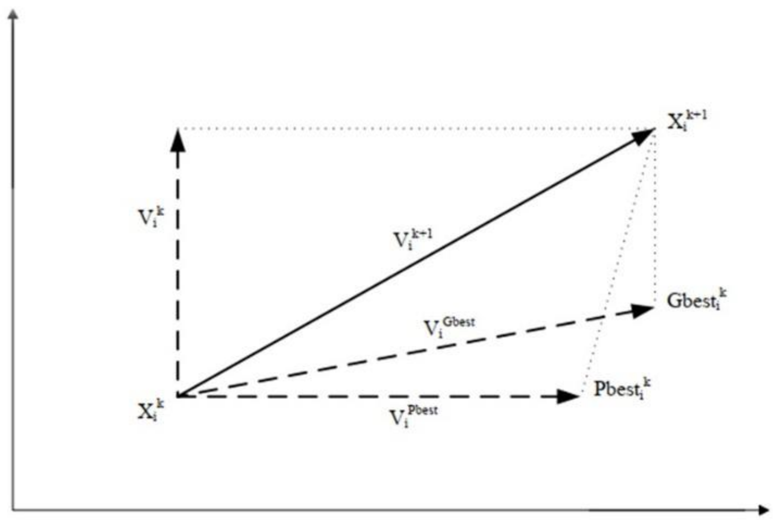


Figure 1. Graph of a particle's position update using current speed V_{id}^{k+1}

In general, the inertia weight w is set according to the following equation (3), (Xin et al., 2009).

B. LDW-PSO

The inertia weight w determines the impact of the particle's current velocity on its future velocity. It balances exploration and exploitation in the search space. A higher inertia weight allows particles to maintain their current velocities, promoting exploration, while a lower inertia weight enables particles to focus more on exploiting promising areas.

In LDW-PSO, w is given as

$$w = \frac{(w_{max} - w_{min})(iter_{max} - iter)}{iter_{max}} + w_{min} \qquad (3)$$

where $iter_{max}$ donates the maximum number of allowable iterations, $iter$ represents the current iteration times, and w_{max} and w_{min} are the initial and final values of the inertia weight, respectively. The performance of LDW-PSO (Bansal et al., 2011), can be improved significantly when w_{max}=0.9 and w_{min}=0.4.

Over the years, PSO has undergone modifications or hybrid methods with PSO have been employed, by adjusting the inertial weight or tuning the parameters (Barrera, et al., 2007. Shi et al., 1998). To propose an enhanced strategy for users employing PSO with Linear Decreasing Inertia Weight, a series of experiments have been systematically conducted across eight distinct test problems. These experiments aim to optimize the performance of the PSO algorithm by analyzing its behavior and effectiveness under various conditions. The linear decreasing inertia weight is a common variant in PSO, where the inertia weight decreases linearly over iterations.

Benchmarks

Eight commonly used benchmarks were adopted to evaluate the performance of algorithm. Convex functions are characterized by their single global optima and smooth landscapes, while non-convex functions exhibit multiple local optima, discontinuities, and irregular surfaces (Adorio, 2005). Understanding how PSO performs in these distinct function categories is crucial for its practical utility in solving real-world problems where the landscape of the objective function may vary from highly structured to highly irregular.

Convex functions:

1. Sphere function:

$$f_1(x) = \sum_{i=1}^{n} x_i$$

2. Booth function:

$$f_2(x) = (x_1 + 2x_2 - 7)^2 + (2x_1 + x_2 - 5)^2$$

3. Matyas function:

$$f_3(x) = 0.26(x_1^2 + x_2^2) - 0.48x_1x_2$$

4. Three hump Camel Back function:

$$f_4(x) = 2x_1^2 - 1.05x_1^4 + \frac{1}{6}x_1^6 + x_1x_2 + x_2^2$$

Non convex functions:

5. Rosenbrock function:

$$f_5(x) = \sum_{i=1}^{n-1}(100(x_{i+1} - x_i^2)^2 + (1 - x_i)^2)$$

6. Bird function:

$$f_6(x) = (x_1 - x_2)^2 + sinx_1 \exp(1 - \cos x_2)^2 + cosx_2 \exp(1 - \sin x_1)^2$$

7. Mishra 04 function:

$$f_7(x) = \sqrt{\left| sin\sqrt{|x_1^2 + x_2^2|} \right|} + 0.01(x_1 + x_2)$$

8. Rastrigin function

$$f_8(x) = 10n + \sum_{i=1}^{n}(x_i^2 - 10\cos(2pi\, x_i))$$

As previously discussed, eight different test optimization functions are used for experiments. Table 1 presents these functions, along with their dimensions and the range of the search space.

Table 1. Dimensions and the range of the search space of test functions.

Function	Dimension	Range
f_1	4	(-20,20)
f_2	2	(-10,10)
f_3	2	(-10,10)
f_4	2	(-10,10)
f_5	4	(-20,20)
f_6	2	$(-2\pi,2\pi)$
f_7	2	(-10,10)
f_8	5	(-10,10)

Experimental result and discussion

For implementing these eight functions, a MATLAB code has been developed.

The population size was set at 50. The symmetric initialization method was used. Each case was tested 10 times on each benchmark. The optimization results for 1000, 3000 and 5000 iterations were recorded, respectively with the condition $V_{max} = X_{max}$. In each case, the values of acceleration parameters C_1 and C_2, the inertia weight, the dimension of the problem, and other relevant parameters are indicated.

In table 2, we have given the results, using PSO with C_1=C_2=2 using the inertia weight

$$w = \frac{(w_{max}-w_{min})(iter_{max}-iter)}{iter_{max}} + w_{min}$$

Table 2. The optimization results for 1000, 3000 and 5000 iterations for each function when C_1=C_2=2 and inertia weight given in (3).

Func	BestFun 1000 Iterations	BestRun 1000 Iterations	BestFun 3000 Iterations	BestRun 3000 Iterations	BestFun 5000 Iterations	BestRun 5000 Iterations
f_1	9.8588e-20	9	1.6383e-17	3	0.0038	9
f_2	0	3	0	6	0	7
f_3	0	8	2.2846e-07	7	0	10
f_4	0	3	0	8	8.0183e-08	5
f_5	1.8209e-16	9	0	8	0	6
f_6	-106.7877	5	-106.7877	5	-106.7877	4
f_7	-0.1761	3	-0.1753	9	-0.1759	3
f_8	0	2	7.2680	2	9.2781	6

For tables 3 and 4, the population sizes of 50, 30, and 20 were chosen for 1000 iterations because we observed that increasing the number of iterations to 3000 and 5000 did not significantly impact performance. In table 3, the results are shown with C_1=C_2=2.01 and $w = 1$.

Table 4 presents the results obtained with C_1=C_2=2.05 and $w = 1$ for the same iterations.

Table 3. The optimization results for population size 50, 30, 20 and 1000 iterations are considered for each function when C_1=C_2=2.01 and $w = 1$.

Func	BestFun Popu. Size 50	BestRun Popu. Size 50	BestFun Popu. Size 30	BestRun Popu. Size 30	BestFun Popu. Size 20	BestRun Popu. Size 20
f_1	1.8887	8	0.2480	4	0.0794	4

f_2	0.0019	3	0.0197	1	89	1
f_3	4.1113e-05	4	0.0007	2	0.0005	10
f_4	0.0034	6	0.0042	3	0	3
f_5	0.0181	1	0.1213	3	0.0297	2
f_6	-106.7611	3	-106.6362	5	-106.7199	2
f_7	-0.1716	3	0.2259	1	0.2259	1
f_8	13.6706	8	11	9	119	9

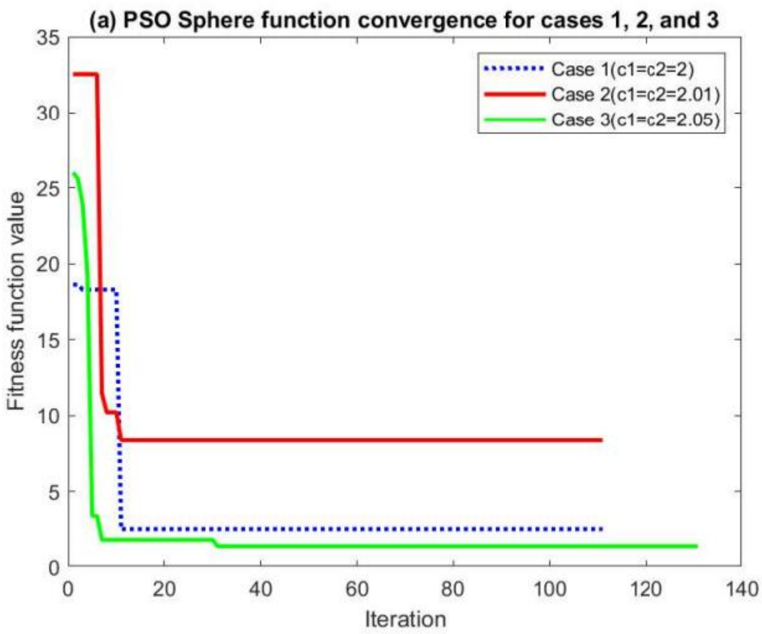
In table 4 we have given the results obtained in case were we take the accelerated coefficients C_1=C_2=2.05 and the inertia weight $w = 1$.

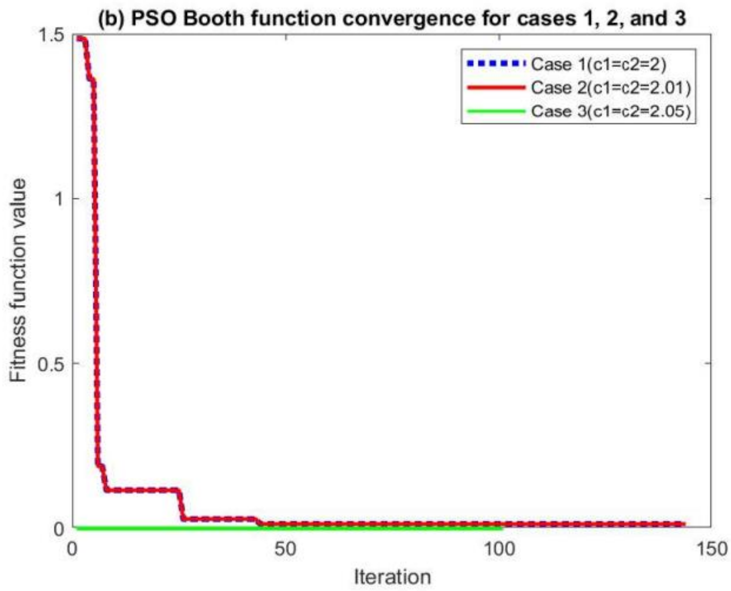
Table 4. The optimization results for population size 50, 30, 20 and 1000 iterations are considered for each function when C_1=C_2=2.05 and $w = 1$.

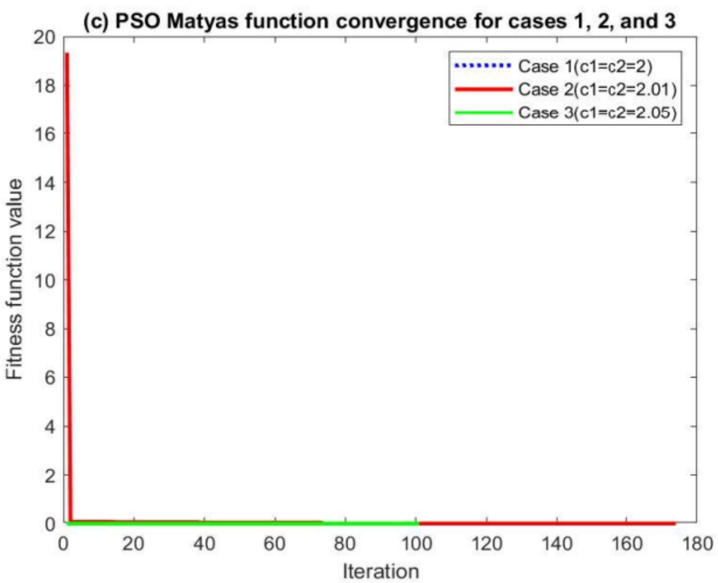
Func	BestFun Popu. Size 50	BestRun Popu. Size 50	BestFun Popu. Size 30	BestRun Popu. Size 30	BestFun Popu. Size 20	BestRun Popu. Size 20
f_1	1.3836	7	0.4734	2	0.0056	1
f_2	0	7	0.0012	7	0	3
f_3	0	4	0	7	0.0005	1
f_4	0.0026	3	0	6	0	1
f_5	0.0761	9	0.2029	2	0.3528	10
f_6	-106.7473	6	123.3693	1	123.3693	1
f_7	-0.1718	5	0.6189	1	0.6189	1
f_8	16	7	10	3	12	3

Below are the eight graphs representing case 1, case 2 and case 3, with a chosen population size of 50 and 1000 iterations.

For each case (linearly decreasing inertia weight with C_1=C_2=2 and C_1=C_2==2.01, C_1=C_2=2.05 with $w = 1$), the PSO algorithm will be influenced differently in terms of exploration and exploitation. The behavior of PSO is influenced by these parameters, and the algorithm will try to balance the exploration of the search space and the exploitation of promising regions**.**




(b) PSO Booth function convergence for cases 1, 2, and 3
Case 1(c1=c2=2)
Case 2(c1=c2=2.01)
Case 3(c1=c2=2.05)
Fitness function value
Iteration


(c) PSO Matyas function convergence for cases 1, 2, and 3
Case 1(c1=c2=2)
Case 2(c1=c2=2.01)
Case 3(c1=c2=2.05)
Fitness function value
Iteration

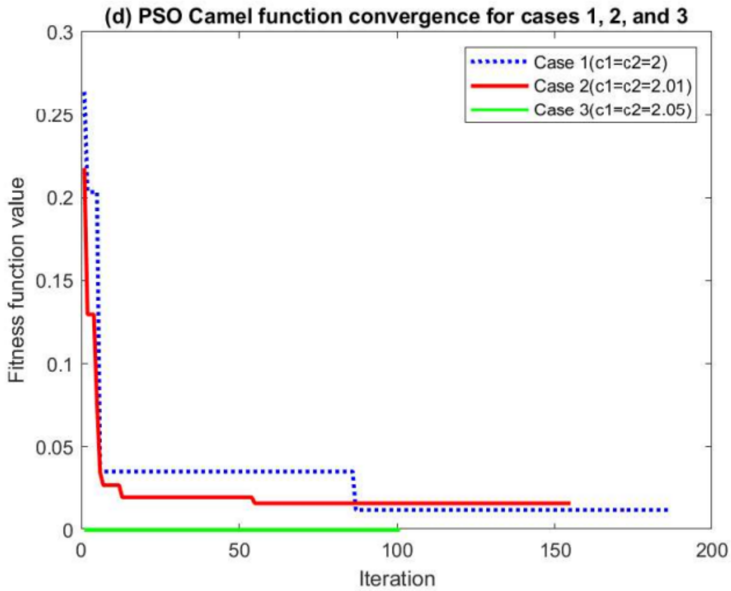
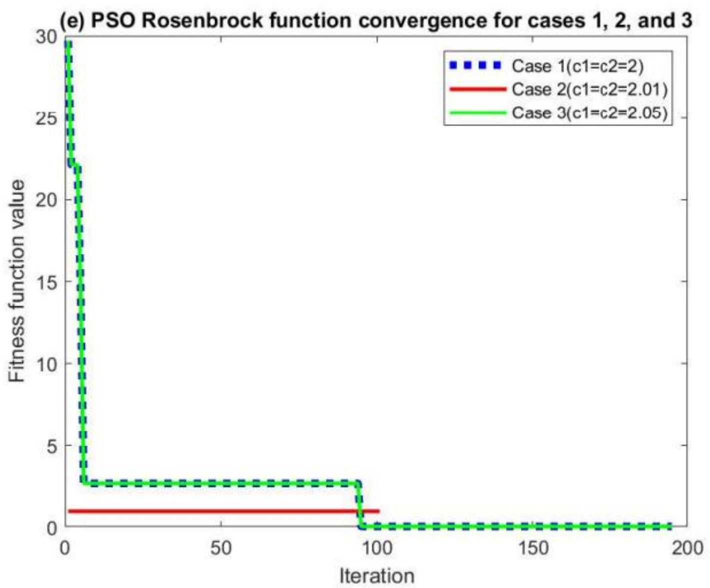
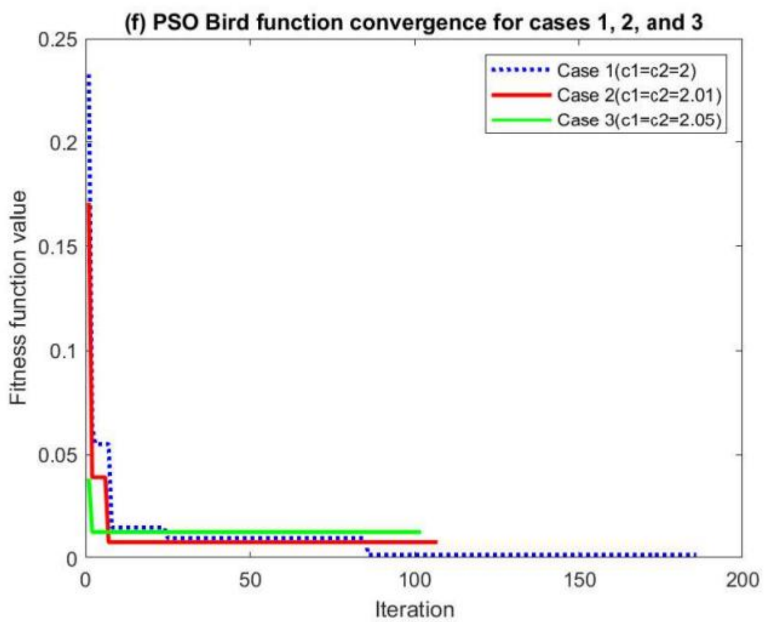


Figure 2. PSO convergence characteristics case 1, 2, 3 convex functions

The case 1 for the function in (a) is faster in the early iterations for a population of 50 compared to cases 2 and 3, for 1000 iterations. For function (b), we have a similar behavior for case 1 and case 2, for the first 150 iterations. In (c), (d) we have a similar behavior for both cases 2 and 3, emphasizing that LDW has better convergence.

For non-convex functions, we have the graphs.


(e) PSO Rosenbrock function convergence for cases 1, 2, and 3
Case 1(c1=c2=2)
Case 2(c1=c2=2.01)
Case 3(c1=c2=2.05)
Fitness function value
Iteration
0
5
10
15
20
25
30
0
50
100
150
200


(f) PSO Bird function convergence for cases 1, 2, and 3
Case 1(c1=c2=2)
Case 2(c1=c2=2.01)
Case 3(c1=c2=2.05)
Fitness function value
Iteration
0
0.05
0.1
0.15
0.2
0.25
0
50
100
150
200

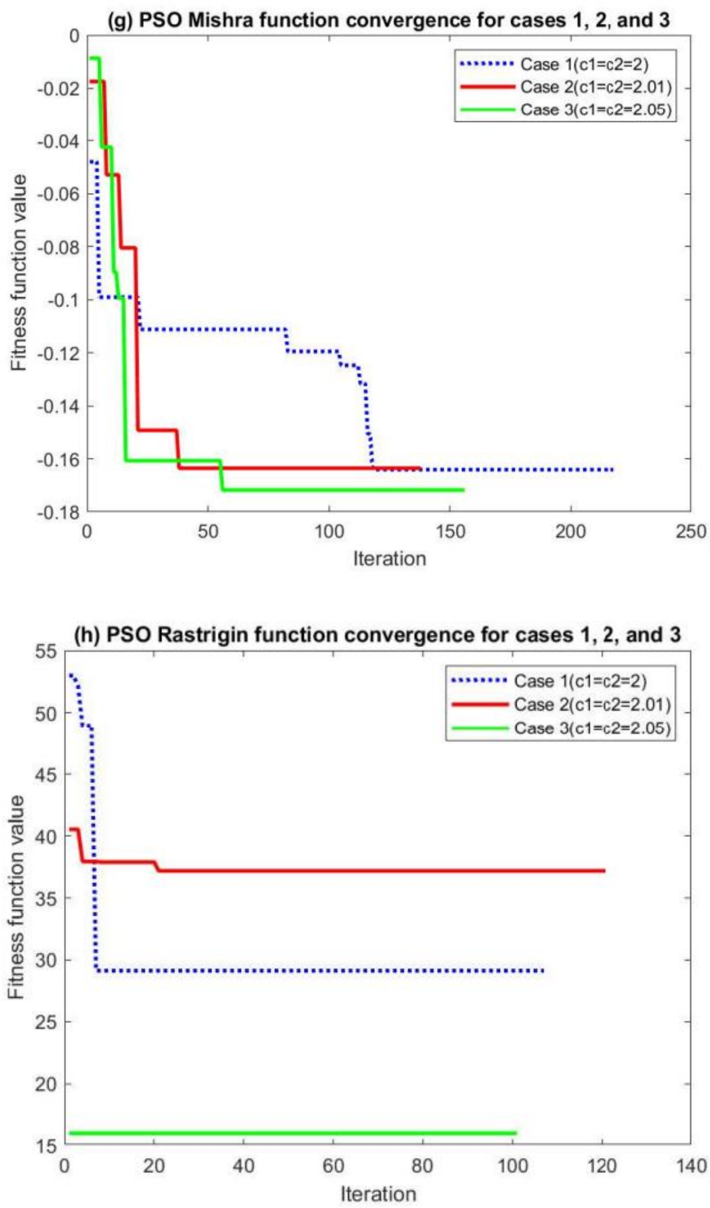


Figure 3. PSO convergence characteristics case 1, 2, 3 non-convex functions

In case (e) where we observe that cases 1 and 2 behave similarly for the first 100 iterations, in function (f), all three cases exhibit similar behavior in the first 100 iterations. However, for function (g), after 50 iterations, we observe a consistent behavior between cases 2 and 3. Finally, for function (h), in the first 100 iterations, case 3 dominates over case 2, but it is emphasized that LDW, i.e., case 1, converges faster than the other two. This result is also based on tables 2, 3, and 4

Conclusions

Through these experiments, we observed that the algorithm's behavior varied significantly for different functions, both convex and non-convex. The results indicate that the algorithm exhibited distinct characteristics in obtaining optimal solutions based on the nature of the underlying function.

For a convex function, the optimization problem is relatively straightforward because there is only one global minimum. PSO works well in such cases. Non-convex functions have multiple local minima, and finding the global minimum is more challenging.

The results demonstrate that PSO exhibits strong performance in solving convex functions, showcasing its efficiency in smooth and well-behaved landscapes. Moreover, the study reveals its adaptability to non-convex functions, indicating the algorithm’s potential in handling complex, irregular optimization spaces.

The findings presented in this research contribute to a better understanding of PSO’s applicability and limitations when dealing with different function types, providing valuable insights for optimization practitioners and researchers. We observed that adjusting the parameters can yield different performances in the PSO algorithm. Ultimately, this analysis helps shed light on the broader role of PSO in solving real-world optimization challenges with varying degrees of convexity.

Our future work will involve interpreting the Bayesian Optimization (BO) results of the optimization methods used to solve these test functions.

References

C. Heitzinger, Global Optimization 2022. Global Optimization | SpringerLink

D. Tian. Particle Swarm Optimization with Chaotic Maps and Gaussian Mutation for Function Optimization. http://dx.doi.org/10.14257/ijgdc.2015.8.4.12

D. Trehan, P. Singh, Non-Convex Optimization. Non-Convex Optimization: A Review | IEEE Conference Publication | IEEE Xplore

E. P. Adorio, U.P. Diliman. MVF - Multivariate Test Functions Library in C for Unconstrained Global Optimization. Revised January 14, 2005.

G. Chen, X. Huang. Natural Exponential Inertia Weight Strategy in Particle Swarm Optimization. Natural Exponential Inertia Weight Strategy in Particle Swarm Optimization | IEEE Conference Publication | IEEE Xplore

J. Barrera, O. Á. Bajo, J. J. Flores, C. A. Coello. Limiting the Velocity in the Particle Swarm Optimization Algorithm. https://www.researchgate.net/publication/316491502

J. C. Bansal, P. K. S. M. Saraswat. Inertia Weight Strategies in Particle Swarm Optimization. Inertia Weight strategies in Particle Swarm Optimization | IEEE Conference Publication | IEEE Xplore

J. J. Liang, A. K. Qin. Comprehensive learning particle swarm optimizer for global optimization of multimodal functions. http://dx.doi

J. Xin, G. Chen, Y. Hai. A Particle Swarm Optimizer with Multi-Stage Linearly-Decreasing Inertia Weight. https://www.researchgate.net/publication/221186791

K. Kenedy and Russel C. Eberhart, "Particle Swarm Optimization'' Particle swarm optimization | IEEE Conference Publication | IEEE Xplore

M. Clerc, J. Kennedy. The Particle Swarm—Explosion, Stability, and Convergence in a Multidimensional Complex Space. The particle swarm - explosion, stability, and convergence in a multidimensional complex space | IEEE Journals & Magazine | IEEE Xplore

M. Jia, L. Li, B. Xiong, L. Feng, W. Cheng, W. F .Dong. Multi-Parameter Auto-Tuning Algorithm for Mass Spectrometer Based on Improved Particle Swarm Optimization. https://www.mdpi.com/journal/bioengineering

M. N. Alan. Codes in MATLAB for Training Artificial Neural Network using Particle Swarm Optimization. https. https://www.researchgate.net/publication/305325563

M. Sudhanshu. Some new test functions for global optimization and performance of repulsive particle swarm method.

N. Chopra, G. Kumar, Sh. Mehta. Hybrid GWO-PSO Algorithm for Solving Convex Economic. E-ISSN: 2321-9637 Available online at www.ijrat.org

R. F. Malik, Th. A. Rahman, S. Z. M. Hashim, R. Ngah. International Journal of Computer Science and Security, Volume (1): Issue (2) 35 New Particle Swarm Optimizer with Sigmoid Increasing Inertia Weight. https://www.researchgate.net/publication/4184592

R. Jamal, B. Men, N. Habib Khan, M. A. Z. Raja, Y. Muhammad. Application of Shannon Entropy Implementation in to a Novel Fractional Particle Swarm Optimization Gravitational Search Algorithm (FPSOGSA) for Optimal Reactive Power Dispatch Problem. IEEE. January 6, 2021.

R. Yangand and X. Liu, Comparison of Convex Optimization-Based Approaches to Solve Nonconvex Optimal Control Problems 2019. www.researchgate.net/publication/330209591

S. Boyd, Convex Optimization 2004. Convex Optimization – Boyd and Vandenberghe (stanford.edu)

Shi, Y. Eberhart, R. Parameter selection in particle swarm optimization | SpringerLink.

Y Shi, R Eberhart. Empirical study of particle swarm optimization. Empirical study of particle swarm optimization | IEEE Conference Publication | IEEE Xplore