Particle swarm optimization

(A meta-heuristic approach for solving optimization problem)

A dissertation submitted for mini-project

by

Kuldeep Gautam

(2023RMA06)

Under the guidance of

Dr. Shubham Gupta

(Assistant professor)

Department of mathematics

Motilal Nehru National Institute of Technology,

Prayagraj (Allahabad), U.P (211004)

Undertaking

I, at this moment, declare that the work presented in this mini-project report entitled "Particle Swarm Optimization (A meta-heuristic approach for solving optimization problem)" submitted to the Department of Mathematics, Motilal Nehru National Institute of Technology, Allahabad (Prayagraj), for the partial fulfillment of the Ph.D. degree in Mathematics. I further declare that this work has not been the basis for awarding any other degree, diploma, or title elsewhere.

Date: 29-12-2023

(Kuldeep Gautam)

(2023RMA06)

Certificate

It is certified that the work in the mini-project entitled "Particle Swarm Optimization (A meta-heuristic approach for solving optimization problem)" by “Kuldeep Gautam” has been carried out under my supervision. This work has not yet been submitted elsewhere for the degree.

Dr. Shubham Gupta
(Assistant Professor)

Department of Mathematics

Motilal Nehru National Institute of Technology, Allahabad

Prayagraj, India.

December, 2023

Acknowledgment

I wish to express my sincere gratitude to my mini-project guide, Dr. Shubham Gupta, Assistant Professor, Department of Mathematics, Motilal Nehru National Institute of Technology, Allahabad (Prayagraj), for his valuable guidance and motivation, without which, it would not have been possible to complete this mini-project. Working with Dr. Shubham Gupta has been a worthwhile experience. He has the immense capability of understanding the student and also knows how to train somebody to get some good work from him. I am also grateful to him for his elaborate and detailed discussion on the subject and for providing sufficient help, and I would like to continue my work under his guidance.

I also thank all faculty members of the Department of Mathematics, MNNIT Allahabad (Prayagraj) for their support, encouragement, and help when it was needed.

Kuldeep Gautam

(2023RMA06)

1. Optimization

Optimization is a process of finding best solution from a given set of alternative solutions. In optimization, optima referred to as either minima (for minimization problem) or maxima (for maximization problem).

Optimization techniques are used in various fields such as science, engineering, economics, software industry, manufacturing, transportation, etc. Therefore, developing efficient and realistic computational techniques are desired in the current real-world.

Mathematically, a generalized form of the single objective optimization problem is stated as follows:

$$\text{Max/Min } f(X) \quad (1)$$

$$\text{s.t.} \quad g_j(\mathrm{X}) \leq 0;\ j = 1, 2, \ldots\ldots, J \quad (2)$$

$$h_k(\mathrm{X}) = 0;\ \mathrm{k} = 1, 2, \ldots\ldots, K \quad (3)$$

$$l_i \leq x_i \leq u_i,\ i = 1, 2, \ldots\ldots, D \quad (4)$$

$$l_i \leq x_i \leq u_i,\ i = 1, 2, \ldots\ldots, D$$

where $\mathrm{X} = (x_1, x_2, \ldots\ldots, x_D) \epsilon R^D$

and $f, g_1, g_2, \ldots\ldots g_j, h_1, h_2, \ldots\ldots h_k$ are real valued functions. The given function $f(X)$ is called the 'objective function' of the problem. $h_k(\mathrm{x}) = 0$; for k =1, 2,…………...,K is known as equality constraints. $g_j(\mathrm{x}) \leq 0$; for j = 1,2,…………………......,J is known as inequality constraints. It is necessary to keep in mind during the calculation of independent variables x_1, x_2,......... x_D, which optimize the objective function, need not break the restrictions on given constraints. The variables x_i's are known as 'decision variables. l_i's are the lower bounds, and u_i's is the upper bound of the decision variables x_i.A decision variable x = (x_1, x_2,.................., x_D) which satisfies all the constraints is called a 'feasible solution'. A feasible solution that optimizes the objective function is called a feasible optimal solution.

Based on constraints there are two type of optimization problem namely, unconstrained and constrained optimization problems. An unconstrained optimization problem is a problem where the objective function is to be optimized over a set of variables without any constraints, but bound constraints may be included. Whereas constrained optimization problem is a problem where the objective function is to be optimized over a set of variables with equality or inequality constraint(s).

1.1 Local and global optima

Let D denote the feasible region of the problem that satisfies all the constraints of an optimization problem. Then, for a minimization problem, if for $\bar{x} \in D$ there exists a neighbourhood $N_\in(\bar{x})$ around $\bar{x}$ such that $f(\bar{x}) \leq f(x)$ for each $x \in D \cap N_\in(\bar{x})$, then $\bar{x}$ is known as a ‘local minimum solution’. However, if, $\bar{x} \in D$ and $f(\bar{x}) \leq f(x)$ for all $x \in D$ then $\bar{x}$ is known as a ‘global minimum solution’ of the optimization problem.

1.2 Deterministic algorithm

A deterministic algorithm is an algorithm that is purely determined by its inputs, where no randomness is involved in the algorithms. A deterministic algorithm is a type of algorithm in which the results of every algorithm is uniquely defined. That why these algorithms will always come up with the same result as given the same inputs. A deterministic algorithm always has a single outcome, i.e., its input always results in the same output. They perform a fixed number of steps and always finishes with an accept or reject state with the same result. Linear search, binary search and bubble sort are some examples of deterministic algorithms.

1.3 Non-deterministic method

A non-deterministic algorithm is one where the same input can lead to multiple possible outcomes. Unlike deterministic algorithms, these types of algorithms do not follow a single clear path through execution. One common form of non-deterministic algorithm involves making random choices during the computation. These algorithms often use randomness to explore different possibilities or search spaces efficiently. They can explore various paths or possibilities, simultaneously. Simulated Annealing and genetic algorithms are some examples of non-deterministic algorithm.

1.4 Heuristics

Heuristics are problem dependent procedure to solve a given problem. Heuristics use practical techniques and shortcuts to produce solutions that may or may not be optimal, but those solutions are sufficient in a given limited time frame. A heuristic method solves a problem faster than the deterministic methods. These methods are used to find the approximate solution of a problem. Heuristics are used in situations in which there is a requirement for a short-term solution. When a complex problem facing difficulties with limited resources of time, heuristics can help to make quick decisions using different search procedures. The heuristic method might not always provide us with a solution, but it may produce satisfactory solution to a given problem.

Based on context, different heuristic methods can correlate with the problem's scope. The most common heuristic methods are - trial and error, guesswork, the process of elimination, and historical data analysis. These methods involve simple information of problem that is most appropriate. They can include representative, affect, and availability heuristics.

1.5 Meta-heuristics algorithm

In computer science and mathematical optimization, a metaheuristic is a higher-level procedure or heuristic designed to find, generate, tune, or select a heuristic that may provide a sufficiently good solution to an optimization problem, especially with incomplete or imperfect information or limited computation capacity. Metaheuristic algorithms are a type of algorithm that are used to solve optimization problems. These algorithms are used when either deterministic algorithms cannot be applied or difficult to use. Compared to optimization algorithms and iterative methods, metaheuristics do not guarantee that a globally optimal solution can be found on some class of problems.

These are some properties that characterize most metaheuristics:

- Metaheuristics are strategies that guide the search process.
- The goal is to efficiently explore the search space in order to find near–optimal solutions.
- Metaheuristic algorithms are approximate the solution and usually non-deterministic.
- Metaheuristics are not problem-specific.

1.6 Nature-inspired optimization algorithms:

Nature-inspired optimization algorithms are the collection of optimization algorithms that are inspired by natural processes or phenomena, such as biological, physical and chemical systems. These algorithms draw inspiration from the behaviour of living organisms or physical processes in nature to solve complex optimization problems. These algorithms are meta-heuristic in nature. Some of the meta-heuristic, nature inspired optimization algorithms are as Genetic algorithm, Particle Swarm Optimization algorithm, Artificial Bee Colony algorithm, Cuckoo Search algorithm, Bat Algorithm, Firefly Algorithm, Immune Algorithm, Gray Wolf Optimizer, and so on. They are used to find the global optimum solution of a problem by exploring a large search space.

Here are some popular nature-inspired optimization algorithms:

(a) Particle swarm optimization (PSO)

- Inspiration: Food foraging behaviour of social creatures (referred to as particle in PSO)
- Procedure: Particles (representing potential solutions) move through the search space, adjusting their positions based on their own experience and the experience of elite particle.

(b) Ant colony optimization (ACO)

- Inspiration: Foraging behavior of ants.
- Procedure: Ants deposit pheromones on paths while searching for food. The pheromone trails influence the future directions of other ants, leading to the discovery of the shortest path

(c) Grey Wolf Optimizer (GWO)

- Inspiration: Leadership hierarchy and hunting behavior of grey wolves.
- Procedure: All the wolfs follow the positions of alpha, beta, and delta wolves guide the search to explore near-to-optima solution.

(d) Firefly Algorithm (FA)

- Inspiration: Flashing behaviour of fireflies.
- Procedure: Fireflies attract each other by flashing light signals. The algorithm simulates the flashing behaviour to optimize problems, with brighter fireflies representing better solutions.

These algorithms have been applied to various optimization problems in engineering, finance, machine learning, and more. The choice of a particular algorithm depends on the nature of the optimization problem and the application's specific requirements.

2. Particle swarm optimization (PSO)

2.1 Introduction

PSO is a nature-inspired optimization algorithm that comes under the category of swarm intelligence-based algorithms. This algorithm was developed by *Kennedy and Russell Eberhart* [1]. The inspiration to develop the algorithm comes from the food foraging behaviour of social creatures in nature such as bird flocking and fish schooling or insect swarming. PSO iteratively tries to improve the quality of particle using its search mechanism. It is a population-based optimization algorithm, where the population consists of candidate solutions known as particles.

2.2 Framework for PSO

Many species are social and form swarms (flocks) for some reason. Flocks may be of different sizes, occur in different weather and may even be composed of other species that can work well together in a group.

Foraging: E.O. Wilson said that in their theory, at least one member of the group can benefit from the discoveries and previous experiences of all other members of the group during the food search. So, more species take advantage of other species' discoveries about the food's location.

There are some rules for species flocking

1. Safe Wandering: When species move, they are not allowed to collide with each other and with obstacles.
2. Dispersion: Each species necessarily maintains a minimum distance from the other species.
3. Aggregation: Each species also maintains a maximum distance from the other species.
4. Homing: All species necessarily have the potential to find a food source or their home.

2.3 Particle swarm optimization process

In the PSO, collection of particles are known as swarm.

During the movement in search space, our search is influenced by two types of learning by the particles. Every particle learns from other particles and knows from its own experience during movement. Learning from others is noted as *social learning*, and learning from itself may *be called cognitive learning*. As a result, from cognitive learning, the particle stores in its memory the best solution visited by the particle itself, called $pbest$. As a consequence of social learning, the particle stores in its memory the best solution seen by any particle of the swarm, which we call $gbest$.

The change in direction and magnitude of a particle is decided by a factor called velocity. Velocity is the rate of change of position to time. But as in PSO, time is iteration, and then the velocity is defined for PSO, the rate of change of position to the iteration. As our iteration increases by unity, the velocity (v) and the position (x) have the same dimensions.

Let an n –dimensional search space; the k^{th} particle of the swarm at $step\ time\ t$ is represented by an n –dimensional vector X_k^t. The velocity vector of this particle at time step t is represented by n–dimensional vector $V(x,t)$. The previously best-visited position of the k^{th} particle at time step t is denoted by the n–dimensional vector P (k, t). "g" is the index of the best particle in the swarm. The velocity update equation updates the velocity of the kth particle.

$V(x,t)$

- x_k^t

Velocity Update Equation:

$$v_{id}^{t+1} = v_{id}^{t} + c_1 * r_1 * \left(x_{pi}^{t} - x_{id}^{t}\right) + c_2 * r_2 * \left(x_{gi}^{t} - x_{id}^{t}\right) \quad (1)$$

Position Update Equation:

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

where – represents the dimension.,m represents the particle

k = 1,2, 3……

d = 1,2, 3……, D

index.

n - Total number of particles in the swarm.

c_1, c_2 - Acceleration coefficients(constants).
r_1, r_2 - Random numbers in the range [0,1].

Equations (1) and (2) show that every particle dimension is updated independently from the others. The only link between the dimensions of the search space is the location of the best positions found, so for $gbest$ and $pbest$.

$Note$: The above two equations represent the basic version of the PSO algorithm.

2.4 Terms in update equation

In the velocity update equation, there are three terms.

1. The previous velocity ' v ' this velocity term reminds us in which direction we are moving previously. The velocity term prevents particles from drastically changing direction.
2. The second terms of the equation are known as the cognitive component. This component helps the particle attract toward its personal best position and thus prohibits itself from wandering.

   Here, it should be noted that $(p_{id} - x_{id})$ is a vector whose direction is from x_{id} to p_{id}, which results in the attraction of the current position towards the particle's best position.
3. Our third term $(p_{gi} - x_{id})$ of the equation is called the social component and is responsible for sharing information throughout the swarm. Because of this term, a particle is attracted towards the best particle of the swarm, i.e. Each particle learns from others.

   Note: Acceleration coefficient c_1 regulates the maximum step size in the direction of the local best position of that particle, while social acceleration coefficient c_2 regulates the maximum step size in the order of the global best particle.

2.5 Particle swarm optimization parameters

The convergence speed and the ability to find the optimal solution of any papulation-based algorithm are mainly influenced by the choice of parameter involved in the population algorithm. No rule recommends we follow a general way during the selection of parameters. However, theoretical or experimental studies tell us the generic parameter value range. Similarly, in other population-based search algorithms, tuning parameters for a generic form of PSO has been challenging. That is why stochastic factors r_1 and r_2 in the search procedure, the basic version of PSO, benefit from having fewer parameters.

One radical parameter is swarm size, often set empirically based on the number of decision variables in the problem. In general, 20-50 particles are recommended.

Other parameters are the accelerating factor c_1 and c_2. These parameters decide the step size to use for the next iteration. This means that c_1 and c_2 determine the speed of particles. For the basic version of PSO, c_1 and c_2 are the same and equal to two. With this choice, the particle's speed increases without control, which is good for faster convergence rate but harmful for better exploitation of the search space. If we choose c_1 and c_2, both are positive and equal; then particles will attract towards the average of $pbest$ and $gbest$. If we decide c_1 is greater than c_2, it will benefit multimodal problems, while c_2 is less than c_1 and will benefit unimodal problems. If we choose small values of c_1 and c_2, it will provide smooth particle trajectories during the search procedure, while larger values of c_1 and c_2 will be responsible for abrupt movements with more acceleration.

The stopping criterion is also a parameter not only for PSO but also for any population-based meta-heuristic algorithm. The known stopping criteria are generally based on a maximum number of function evaluations or iterations, which are proportional to the time taken by the algorithm and acceptable error. A more efficient stopping criteria is based on the available search capacity of the algorithm. Search should be stopped if an algorithm does not improve the solution with a significant amount up to a certain number of iterations.

2.6 Mathematical Model

Steps: -

The following step involves into the PSO

1. Initialization of algorithm constants.
2. Initialize the solution from the solution space (initial values for position and velocity).
3. Calculate the fitness of each entity(particle).
4. Updating local and global bests ($pbest$ and $gbest$).
5. We are updating the velocity and position of each particle.
6. Come back to step third and repeat until the termination condition.

2.7 Pseudo-code:

Create and initialize a D-dimensional swarm, S and corresponding velocity vectors;

{For t=1 to the maximum bound on the number of iterations,

do

{for i =1 to S do

{For d = 1 to D, do

Apply the velocity update equation

Apply position update equation

}

Compute fitness of updated position;

If needed, update historical information for $pbest$ and $gbest$;

}

Terminate if $gbest$ meets problem requirements}

4. Numerical example:

$$\boldsymbol{Min} \ \ f(X) = x_1^2 + x_2^2$$

$$\boldsymbol{s.t.} \ x_1, x_2 \in [-5,5]$$

To apply the PSO on the above problem, we have fixed following setting:

We will fix the

- **(a)** $swarm\ size(N)$ = 5,
- **(b)** parameter c_1= c_2= 2,
- **(c)** lower bound (l_b) = -5,
- **(d)** upper bound (u_b) = 5.

Initialization ($\boldsymbol{t}$) = 0

For 1st particle $\boldsymbol{x_1}^{(\mathbf{0})} = (\boldsymbol{x_{11}}^{(\mathbf{0})}, \boldsymbol{x_{12}}^{(\mathbf{0})})$:

$x_{11}{}^{(0)} = l_b + rand\,() * (u_b - l_b)$

$= -5 + 0.97 * \{5 - (-5)\} = 4.7000$

$x_{12}{}^{(0)} = -5 + 0.4718 * 10$

$= -0.2824$

$\Rightarrow \mathbf{x_1}^{(\mathbf{0})} = (\mathbf{4.7000}, \mathbf{-0.2824})$

For 2nd particle $\boldsymbol{x_2}^{(\mathbf{0})} = (\boldsymbol{x_{21}}^{(\mathbf{0})}, \boldsymbol{x_{22}}^{(\mathbf{0})})$:

$x_{21}{}^{(0)} = -5 + 0.9273 * 10$

$= 4.2727$

$x_{22}{}^{(0)} = -5 + 0.9107 * 10$

$= 4.1074$

$\Rightarrow \boldsymbol{x_2}^{(\mathbf{0})} = (\mathbf{4.2727}, \mathbf{4.1074})$

For 3rd particle $\boldsymbol{x_3}^{(\mathbf{0})} = (\boldsymbol{x_{31}}^{(\mathbf{0})}, \boldsymbol{x_{32}}^{(\mathbf{0})})$:

$x_{31}{}^{(0)} = -5 + 0.4858 * 10$

$= -0.1422$

$x_{32}{}^{(0)} = -5 + 0.79 * 10$

$= 2.9010$

$\Rightarrow\ \boldsymbol{x_3}^{(\mathbf{0})} = (-\mathbf{0.1422}, \mathbf{2.9010})$

For 4th particle $\boldsymbol{x_4} = (\boldsymbol{x_{41}}, \boldsymbol{x_{42}})$**:**

$x_{41}{}^{(0)}$ = -5 $+\ 0.8113 * 10$

$= 3.1128$

$x_{42}{}^{(0)} = -5 + 0.9495 * 10$

= 4.4949

$\Rightarrow\ \boldsymbol{x_4}^{(\mathbf{0})} = (\mathbf{3.1128}, \mathbf{4.4949})$

For 5th particle $\boldsymbol{x_5} = (\boldsymbol{x_{51}}, \boldsymbol{x_{52}})$**:**

$x_{51}{}^{(0)} = -5 + 0.1018 * 10$

$= -3.9822$

$x_{52}{}^{(0)} = -5 + 0.6655 * 10$

= 1.6545

$\Rightarrow \boldsymbol{x_5}^{(\mathbf{0})} = (-\mathbf{3.9822}, \mathbf{1.6545})$

In this way we have initialized all the particles of the PSO. Positions of all the particles along with their fitness values are provided in Table 1. From this table it can be seen that the particle $\boldsymbol{x_3}^{(\mathbf{0})} = (-\mathbf{0.1422}, \mathbf{2.9010})$ is a $gbest$ solution and its fitness value is 8.4360.

Table 1. Particles of the PSO at iteration $(t) = 0$ with their fitness values.

Particles	Positions		Fitness
$\boldsymbol{x_1}$	4.7000	-0.2824	22.1697
$\boldsymbol{x_2}$	4.2727	4.1074	35.1267
$\boldsymbol{x_3}$	**-0.1422**	**2.9010**	**8.4360**
$\boldsymbol{x_4}$	3.1128	4.4949	29.8936
$\boldsymbol{x_5}$	-3.9822	1.6545	18.5953

In the PSO, we initialize velocity vectors corresponding to all the particle in the range $(-v_{max}, +v_{max})$. These velocities are provided in Table 2.

A,b,c ,d hattao and colun

Table 2. Velocities of particles at iteration $(t) = 0$.

Particle velocity	Velocities	
$\boldsymbol{v_1}$	4.0079	-2.1150
$\boldsymbol{v_2}$	-3.8302	0.4468
$\boldsymbol{v_3}$	4.2338	4.6852
$\boldsymbol{v_4}$	1.3336	4.4489
$\boldsymbol{v_5}$	-4.0246	-3.3239

For initial iteration, $pbest$ position corresponding to all the particles will be same as their position. Now, we will apply the PSO search rule iteratively to update the particles.

Iteration (t) = 1

For 1st particle $\boldsymbol{x_1^{(0)} = (x_{11}, x_{12})}$:

$\boldsymbol{x_1^{(0)} = (4.7000, -0.2824)}$

$pbest = (4.7000, -0.2824)$

$gbest = (-0.1422, 2.9010)$

We will apply velocity update equation for v_{11}

$$\boldsymbol{v_{id}^{t+1} = v_{id}^{t} + c_1 r_1 \left(x_{pi}^{t} - x_{id}^{t}\right) + c_2 r_2 \left(x_{gi}^{t} - x_{id}^{t}\right)}$$

$v_{11}^{(1)}$ = 4.0079 + 2* 0.34 (4.7000 – 4.7000) + 2* 0.86* (-0.1422 – 4.7000)

= -4.3207 (accepted)

Now we will apply position update equation:

$\boldsymbol{x_{11}^{1} = x_{11}^{(0)} + v_{11}^{(0)}}$

$= 4.7000 - 4.3207.$

= 0.3793. (accepted)

For $\boldsymbol{x_{12}}$

$v_{12}^{(1)} = -2.1150 + 2 * 0.47 * (-0.2824 + 0.2824) + 2 * 0.91 * (2.9010 + 0.2824)$

= 3.6788 (accepted)

$x_{12}^{(1)} = -0.2824 + 3.6788$

$=$ 3.3964 (accepted)

$\Rightarrow\ \boldsymbol{x_1^{(1)}} = (\mathbf{0.3793, 3.3964})$

For 2nd particle $\mathbf{x_2^{(0)}} = (\mathbf{4.2727, 4.1074})$:

$pbest = (4.2727, 4.1074)$

$gbest = (-0.1422, 2.9010)$

$$\boldsymbol{v_{21}^{(1)} = v_{21} + c_1 * r_1 * (x_{p1} - x_{21}) + c_2 * r_2 * (x_{g1} - x_{21})}$$

$v_{21}^{(1)}$ = -3.8302 + 2* 0.34* (4.2727–4.2727) + 2* 0.86* (-0.1422 – 4.2727)

= -11.4238 (not − accepted)

Re- initializing $v_{21}^{(1)}$

$v_{21}^{(1)}$ = -5

$x_{21}^{(1)}$ = 4.2727 – 5

= -0.7273 (accepted)

For $x_{22}^{(0)}$

$v_{22}^{(1)} = 0.4468 + 2 * 0.12 * (4.1074 - 4.1074) + 2 * 0.06 * (2.9010 - 4.1074)$

= 0.3020

$x_{22}^{(1)}$ = 4.1074 + 0.3020

= 4.4094 (accepted)

$\Rightarrow\ \mathbf{x_2^{(1)}} = (\mathbf{-0.7273, 4.4094})$

For 3rd particle $\mathbf{x_3^{(0)}} = (\mathbf{-0.1422, 2.9010})$:

$pbest = (-0.1422, 2.9010)$

$gbest = (-0.1422, 2.9010)$

$v_{31}^{(1)} = v_{31} + c_1 * r_1 * (x_{p1} - x_{31}) + c_2 * r_2 * (x_{g1} - x_{31})$

$= 4.2338 + 2 * 0.69 * (-0.1422 + 0.1422) + 2 * 0.34 * (-0.1422 + 0.1422)$

= 4.2338

$x_{31}^{(1)} = -0.1422 + 4.2338$

= 4.0916

For $x_{32}^{(0)}$

$v_{32}^{(1)} \quad = 4.6852 + 2 * 0.42 * (2.9010 - 2.9010) + 2 * 0.21 * (2.9010 - 2.9010)$

$v_{32}^{(1)}$ = 4.6852

$x_{32}^{(1)}$ = $2.9010 + 4.6852 = 7.5862$ (not-accepted)

Re-initializing $x_{32}^{(1)}$ by choosing random value from the range [0, 1]

$x_{32}^{(1)} \quad = l_b + rand\,() * (u_b - l_b)$

$= -5 + 0.31 * (5 + 5)$

= -1.9

$\Rightarrow \boldsymbol{x_3^{(1)}} \quad = (\mathbf{4.0916, -1.9})$

For 4th particle $\boldsymbol{x_4^{(0)}}$ = $(\mathbf{3.1128, 4.4949})$:

$pbest = (3.1128, 4.4949)$

$gbest = (-0.1422, 2.9010)$

$v_{41}^{(1)}$ = $1.3336 + 2 * 0.18 * (3.1128 - 3.1128) + 2 * 0.23 * (-0.1422 - 3.1128)$

= -0.1637

$x_{41}^{(1)}$ = 2.9491 (accepted)

For $x_{42}^{(0)}$

$v_{42}^{(1)}$ = $4.4489 + 2 * 0.61 * (4.4949 - 4.4949) + 2 * 0.94 * (2.9010 - 4.4949)$

= -0.0366

$x_{42}^{(1)}$ = 4.4583 (accepted)

$\Rightarrow \boldsymbol{x_4^{(1)}} = (\mathbf{2.9491, 4.4583})$

For 5th particle $\boldsymbol{x_{51}^{(0)}} = (\mathbf{-3.9822, 1.6545})$:

$pbest = (-3.9822, 1.6545)$

$gbest = (-0.1422, 2.9010)$

$v_{51}^{(1)} = -4.0246 + 2 * 0.09 * (-3.9822 + 3.9822) + 2 * 0.39 * (-0.1422 + 3.9822)$

= -1.0294

$x_{51}^{(1)}$ = -5.0116 (not accepted)

Re-initializing x'_{51}

$x_{51}^{(1)} = -5 + 0.89 * 10$

$x_{51}^{(1)} = 3.9$

For $x_{51}^{(0)}$

$v_{52}^{(1)} = -3.3239 + 2 * 0.65 * (1.6545 - 1.6545) + 2 * 0.10 * (2.9010 - 1.6545)$

= -3.0746

$x_{52}^{(1)}$ = 1.6545 – 3.0746

= -1.4201 ($accepted$)

$\Rightarrow \boldsymbol{x_5^{(1)} = (3.9, -1.4201)}$

Table 3. Updated velocities and particles position along with their fitness values are provided in Table 3. And Table 4. respectively.

Particle velocity	Velocities	
$\boldsymbol{v_1}$	-4.3207	3.6788
$\boldsymbol{v_2}$	-5.0000	0.3020
$\boldsymbol{v_3}$	4.2338	4.6852
$\boldsymbol{v_4}$	-0.1637	-0.0366
$\boldsymbol{v_5}$	-1.0294	-3.0746

Table 4. Particles of the PSO at iteration $(t) = 1$ with their fitness values.

Particles	Positions		Fitness value
$\boldsymbol{x_1}$	0.3793	3.3964	11.6794
$\boldsymbol{x_2}$	-0.7273	4.4094	19.9718
$\boldsymbol{x_3}$	4.0916	-1.9000	20.3512
$\boldsymbol{x_4}$	2.9491	4.4583	28.5736
$\boldsymbol{x_5}$	3.9000	-1.4201	17.2267

After comparison of Tables 1. and 4. it can be seen that the particle $\boldsymbol{x_3^{(0)}}$ = (-0.1422,2.9010) is $gbest$ with fitness value 8.4360.

To get the $pbest$ value of each particle, we need to compare Table 1.and 4.

Second iteration (t) = 2:

For 1st particle $\boldsymbol{x_1^{(1)} = (0.3793, 3.3964)}$:

$pbest = (0.3793, 3.3964)$

$gbest = (-0.1422, 2.9010)$

Velocity updating equation:

$v_{11}^{(2)} = v_{11} + c_1 * r_1 * \left(x_{p1} - x_{11}\right) + c_1 * r_1 * (x_{g1} - x_{11})$

$= -4.3207 + 2 * 0.18 * (0.3793 - 0.3793) + 2 * 0.68 * (-0.1422 - 0.3793)$

$= -5.0299$ (not accepted)

Re -initialization on the boundary of velocity.

$v_{11}^{(2)}$ = -5

$x_{11}^{(2)}$ = 0.3793 – 5

$= -4.6207$ (accepted)

$v_{12}^{(2)} = 3.6788 + 2 * 0.47(3.3964 - 3.3964) + 2 * 0.23 * (2.9010 - 3.3964)$

= 3.4509

$x_{12}^{(2)}$ = 3.3964 + 3.4509 = 6.8473 (not accepted)

Re-initialization of $x_{12}^{(2)}$

$x_{12}^{(2)} = -5 + 10 * (0.531)$

$x_{12}^{(2)}$ = 0.3100

$\Rightarrow \mathbf{x_1^{(2)}} = (\mathbf{-4.6207, 0.3100})$

For 2nd particle $x_2^{(1)}$= ($\mathbf{-0.7273, 4.4094}$):

$pbest = (-0.7273, 4.4094)$

$gbest = (-0.1422, 2.9010)$

$v_{21}^{(2)} = -5 + 2 * 0.97 * (-0.7273 + 0.7273) + 2 * 0.32 * (-0.1422 + 0.7273)$

$= -4.6255$

$x_{21}^{(2)} = -0.7273 - 4.6255$

$= -5.3528$ (not accepted)

Re-initializing $x_{21}^{(2)}$

$x_{21}^{(2)} = -5 + 0.18 * 10$

= -3.2

$v_{22}^{(2)} = 0.3020 + 2 * 0.61 * (4.4094 - 4.4094) + 2 * 0.11 * (2.9010 - 4.4094)$

= -0.0298

$x_{22}^{(2)} = 4.4094 - 0.0298$

= 4.3796

$\Rightarrow \boldsymbol{x_2^{(2)}} = (\mathbf{-3.2, 4.3796})$

For 3rd particle $\mathbf{x_3^{(1)}} = (\mathbf{4.0916, -1.9})$:

$pbest = (-0.1422, 2.9010)$

$gbest = (-0.1422, 2.9010)$

$v_{31}^{(2)} = 4.2338 + 2 * 0.45 * (-0.1422 - 4.0916) + 2 * 0.17 * (-0.1422 - 4.0916)$

$v_{31}^{(2)} = 3.2177$ (accepted)

$x_{31}^{(2)}$ = 4.0916 + 3.2177 = 7.3093 (not accepted)

Re-initializing $x_{31}^{(2)}$

$x_{31}^{(2)} = -5 + 0.23 * 10$

= -2.7

$v_{32}^{(2)} = 4.6852 + 2 * 0.39 * (2.9010 + 1.9) + 2 * 0.09 * (2.9010 + 1.9)$

$= 9.2942$ (not accepted)

Reducing the value of v'_{32} on the boundary

$v_{32}^{(2)}$ = 5

$x_{32}^{(2)} = -1.9 + 5 = 3.1$(accepted)

$\Rightarrow \boldsymbol{x_3^{(2)}} = (\mathbf{-2.7, 3.1})$

For 4th particle $\mathbf{x_4^{(1)}} = (\mathbf{2.9491, 4.4583})$:

$pbest$ = (2.9491,4.4583)

$gbest = (-0.1422, 2.9010)$

$v_{41}^{(2)} = -0.1637 + 2 * 0.77 * (2.9491 - 2.9491) + 2 * 0.27 * (-0.1422 - 2.9491)$

$= -0.1637 - 1.6693 = -1.8330$

$x_{41}^{(2)}$ = 2.9491 – 1.8330

$= 1.1161$ (accepted)

$v_{42}^{(2)} = -0.0366 + 2 * 0.46 * (4.4583 - 4.4583) + 2 * 0.21 * (2.9010 - 4.4583)$

$= -0.6907$

$x_{42}^{(2)} = 4.4583 - 0.6907$

$= 3.7676$

$\Rightarrow \boldsymbol{x_4^{(2)}} = (\mathbf{1.1161, 3.7676})$

For 5th particle $\mathbf{x_5^{(1)}} = (\mathbf{3.9, -1.4201})$:

$pbest = (3.9, -1.4201)$

$gbest = (-0.1422, 2.9010)$

$v_{51}^{(2)} = -1.0294 + 2 * 0.51 * (3.9 - 3.9) + 2 * 0.16 * (-0.1422 - 3.9)$

$= -2.3229$

$x_{51}^{(2)} = 3.9 - 2.3229 = 1.5771$ (accepted)

$v_{52}^{(2)} = -3.0746 + 2 * 0.28 * (-1.4201 + 1.4201) + 2 * 0.54 * (2.9010 + 1.4201)$

$= 1.5922$

$x_{52}^{(2)} = -1.4201 + 1.5922$

$= 0.1721$ (accepted)

$\Rightarrow \boldsymbol{x_5^{(2)}} = (\mathbf{1.5771, 0.1721})$

Table 5. Updated velocities and particles position along with their fitness values are provided in Table 5. And Table 6. respectively

Particles	Velocities	
$\boldsymbol{v_1}$	-5.0000	3.4509
$\boldsymbol{v_2}$	-4.6255	-0.0298
$\boldsymbol{v_3}$	3.2177	5.0000
$\boldsymbol{v_4}$	-1.8330	-0.6907
$\boldsymbol{v_5}$	-2.3229	1.5922

Table 6. Particles of the PSO at iteration $(t) = 2$ with their fitness values.

Particles	Positions		Fitness value
$\boldsymbol{x_1}$	-4.6207	0.3100	21.4470
$\boldsymbol{x_2}$	-3.2000	4.3796	24.0041
$\boldsymbol{x_3}$	-2.7000	3.1000	16.9000
$\boldsymbol{x_4}$	1.1161	3.7676	15.4405
$\boldsymbol{x_5}$	**1.5771**	**0.1721**	**2.5169**

After the second iteration we can seen the final $gbest$ value is $\boldsymbol{x_5^{(2)}} = (\mathbf{1.5771}, \mathbf{0.1721})$ and its fitness value is 2.5169.

4. Modified Particle Swarm Optimization

The particle swarm optimizer has been found to be robust and fast in solving nonlinear, non-differential, multi-modal problems. but it is still in its infancy

As we know the PSO equation

Velocity Update Equation:

$$v_{id}^{t+1} = v_{id}^{t} + c_1 r_1 \left(x_{pi}^{t} - x_{id}^{t}\right) + c_2 r_2 \left(x_{gi}^{t} - x_{id}^{t}\right) \tag{1}$$

Position Update Equation:

$$x_{id}^{t+1} = x_{id}^{t} + v_{id}^{t+1} \tag{2}$$

The right-hand side of the equation (1) consists of three parts:

The first part is the previous velocity of the particle, the second and third parts are ones contributing to the change of the velocity of a particle. without these two parts, the particles will keep on flying at the current speed in the same direction until they hit the boundary. PSO will not find an acceptable solution unless there are acceptable solutions on their flying trajectories, but that is rare case.

On the other-hand in equation (1) without the first part, the flying particles velocities are only determined by their current positions and their best position. The velocity is itself is memoryless. Assume at the beginning the particle i has the best global position, then the particle i will be flying at the velocity ‘0’(Zero), that is it will keep still until another particle takes over the global best position. At the same time, each other particle will be flying toward its weighted-centroid of its own best position and global best position of the population. A recommended choice for constant $c_1$and c_2 is integer ‘2’, Since it on average makes the weights for “social” and “cognitive” parts to be ‘1’, Under this condition the particles statistically contract to the new global best position. Therefore, it can be imagined that the search process for PSO without first part is a process where the search process statistically shrinks throughout the iterations. There is only one chance for PSO to find the solution is that when the global optimum is within the initial search space. The final solution is heavily dependent on the initial population.

On the other hand, by adding the first part, the particles have tendency to expand the search space, that is, they have the ability to explore the new area. So, they more likely have a global search ability by adding the first part. Both the local search and global will benefit solving some kinds of problems. There is a trade-off between the global and local search. For different problems, there should be different balances between the local search ability and global search ability.

4.1 Exploration and exploitation:

Exploration and exploitation are two key aspects in optimization algorithms, including Particle Swarm Optimization (PSO). These terms describe the balance between discovering new regions in the solution space (exploration) and refining current solutions (exploitation).

(a). Exploration:

- Definition: Exploration involves searching the solution space broadly to discover new, potentially better solutions.
- Effect in PSO: High exploration in PSO means that particles are making larger and more diverse movements in the search space, which can help escape local optima and discover new promising areas.

(b). Exploitation:

- Definition: Exploitation involves focusing on known promising regions to refine and improve solutions.
- Effect in PSO: High exploitation in PSO means that particles are converging toward the best-known solutions, refining their positions based on local and global information.

The exploration-exploitation trade-off is crucial in optimization, as too much focus on exploration might lead to a slow convergence, while too much exploitation may result in premature convergence to suboptimal solutions.

4.2 Inertia Weight in PSO:

The inertia weight in PSO is a parameter that controls the impact of the particle's previous velocity on its future velocity. It is a key factor in balancing exploration and exploitation. The equation governing the update of particle velocity in PSO is typically given by,

$$v_{id}^{t+1} = w * v_{id}^{t} + c_1 * r_1 * \left(x_{pi}^{t} - x_{id}^{t}\right) + c_2 * r_2 * \left(x_{gi}^{t} - x_{id}^{t}\right) \tag{3}$$

Where:

- v_{id}^{t} - Velocity of particle i at iteration t
- x_{gi}^{t} - $gbest$ position of particle i at iteration t.
- x_{pi}^{t}- $pbest$ position of particle i at iteration t.
- w - Inertia weight,
- c_1 ,c_2- Acceleration coefficients,
- r_1, r_2 - Random numbers in the range [0, 1].

the role of the inertia weight is to balance the impact of the particle's current velocity with its historical velocity. The inertia weight is typically initialized and updated during the optimization process. Its effect on exploration and exploitation is as follows:

- High inertia (exploration): A higher inertia weight allows particles to retain more of their previous velocities, enabling them to explore the solution space more extensively. This promotes exploration by allowing particles to make larger jumps.
- Low inertia (exploitation): A lower inertia weight places more emphasis on the current velocity and tends to make particles converge toward the best-known solutions. This promotes exploitation by focusing on refining the search around known promising regions.

The choice of the inertia weight is problem-dependent, and finding the right balance is essential. Adaptive or time-varying inertia weight strategies are often employed to dynamically adjust the exploration-exploitation balance during the optimization process.

Iteration (t) = 1

For 1th particle $\mathbf{x}_1^{(0)}$ = $(\mathbf{4.7000}, \mathbf{-0.2824})$:

$pbest = (4.7000, -0.2824)$

$gbest = (-0.1422, 2.9010)$

$v_{11}^{(1)} = w * v_{11} + c_1 * r_1 * (x_{p1} - x_{11}) + c_1 * r_1 * (x_{g1} - x_{11})$

$= 0.8995 * 4.0079 + 2 * 0.34 * (4.7 - 4.7) + 2 * 0.86 * (-0.1422 - 4.7)$

= -4.7235

$x_{11}^{(1)} = -0.0235$ (accepted)

$v_{12}^{(1)} = 0.8995 * (-2.11500) + 2 * 0.47 * (-0.2824 - 0.2824) + 2 * 0.91 * (2.9010 + 0.2824)$

$= 3.8914$ (accepted)

$x_{12}^{(1)} = -0.2824 + 3.8914 = 3.6090$ (accepted)

$\Rightarrow \boldsymbol{x}_1^{(1)} = (\mathbf{-0.0235}, \mathbf{3.6090})$

For 2nd particle $\mathbf{x}_2^{(0)}$ = (4.2727,4.1074):

$pbest = (4.2727, 4.1074)$

$gbest = (-0.1422, 2.9010)$

$v_{21}^{(1)} = w * v_{21} + c_1 * r_1 * (x_{p1} - x_{21}) + c_1 * r_1 * (x_{g1} - x_{21})$

$v_{21}^{(1)}$ $= 0.8995 * (-3.8302) + 2 * 0.34 * (4.2727 - 4.2727) + 2 * 0.86 * (-0.1422 - 4.2727)$

$= -11.0389$ (not accepted)

Choosing the value of v'_{21} boundary of velocities.

$v_{21}^{(1)}$ = -5

$x_{21}^{(1)}$ = 4.2727-5 = -0.7273

$v_{22}^{(1)} = w * v_{22} + c_1 * r_1 * \left(x_{p2} - x_{22}\right) + c_1 * r_1 * (x_{g2} - x_{22})$

$= 0.8995 * 0.4468 + 2 * 0.12 * (4.1074 - 4.1074) + 2 * 0.06 * (2.9010 - 4.1074)$

$= 0.2571$(accepted)

$x_{22}^{(1)} = 4.1074 + 0.2571$

$x_{22}^{(1)} = 4.3645$(accepted)

$\Rightarrow \boldsymbol{x_2^{(1)}} = (\mathbf{-0.7273, 4.3645})$

For 3rd particle $\mathbf{x_3^{(0)}}$ = (-0.1422,2.9010):

$pbest = (-0.1422,2.9010)$

$gbest = (-0.1422,2.9010)$

$v_{31}^{(1)} = 0.8995 * 4.2338 + 2 * 0.69 * (-0.1422 + 0.1422) + 2 * 0.34 * (-0.1422 + 0.1422)$

= 3.8083

$x_{31}^{(1)} = -0.1422 + 3.8083 = 3.6661$ (accepted)

$v_{32}^{(1)} = 0.8995 * 4.6852 + 2 * 0.42 * (2.9010 - 2.9010) + 2 * 0.21 * (2.9010 - 2.9010)$

= 4.2143

$x_{32}^{(1)}$ = 2.9010 + 4.2143

= 7.1153 (not accepted)

Re-initialization of $x_{32}^{(1)}$

$x_{32}^{(1)} = -5 + 10 * (0.19)$

$x_{32}^{(1)}$ = -3.10

$\Rightarrow \boldsymbol{x_3^{(1)}} = (\mathbf{3.6661, -3.10})$

For 3rd particle $x_4^{(0)}$= $(\mathbf{3.1128}, \mathbf{4.4949})$:

$pbest \ = \ (3.1128,4.4949)$

$gbest \ = \ (-0.1422,2.9010)$

$v_{41}^{(1)}$ = $0.8995 * 1.3336 \ + \ 2 * \ 0.18 * (3.1128 - 3.1128) \ + \ 2 * 0.23 * (-0.1422 - 3.1128)$

= 1.196 – 1.4973

= -0.3013

$x_{41}^{(1)}$ = 3.1120 – 0.3013

$x_{41}^{(1)}$ = 2.8107(accepted)

$v_{42}^{(1)}$ = $0.8995 * 4.4489 \ + \ 2 * 0.61 * (4.4949 - 4.4949) \ + \ 2 * 0.94 * (2.9010 - 4.4949)$

= 1.0053

$x_{42}^{(1)}$ = $4.4949 \ + \ 1.0053 \ = \ 5.5002$ (not accepted)

Re-initializing $x_{42}^{(1)}$

$x_{42}^{(1)}$ = $-5 \ + \ 10 * 0.89$

$x_{42}^{(1)}$ = 3.9

$\Rightarrow \boldsymbol{x_4^{(1)}}$ = $(\mathbf{2.8107}, \mathbf{3.9})$

For 5th particle $x_5^{(0)}$= $(-\mathbf{3.9822}, \mathbf{1.6545})$:

$pbest \ = \ (-3.9822,1.6545)$

$gbest \ = \ (-0.1422,2.9010)$

$v_{51}^{(1)} = w * \ v_{51} + \ c_1 * r_1 * \ \left(x_{p1} - \ x_{51} \ \right) + \ c_1 * r_1 * (\ x_{g1} - \ x_{51})$

$= \ 0.8995 * (-4.0246) \ + \ 2 * 0.09 * (-3.9822 + 3.9822) \ + \ 2 * 0.39 * (-0.1422 + 3.9822)$

= -0.6249

$x_{51}^{(1)}$ = -3.9822 – 0.6249

= -4.6071 (accepted)

$v_{52}^{(1)}$ = $0.8995 * (-3.3239) \ + \ 2 * 0.65 * (1.6545 - \ 1.6545) \ + \ 2 * 0.10 * \ (2.9010 - 1.6545)$

= -2.7405

$x_{52}^{(1)}$ = 1.6545 -2.7405

= -1.0860 (accepted)

$\Rightarrow \mathbf{x}_5^{(1)} = (\mathbf{-4.6071, -1.086})$

Table 7. Updated velocities and particles position along with their fitness values are provided in Table 7. And Table 8. respectively.

Particles velocity	**velocities**	
1.	-4.7235	3.8914
2.	-5	0.2571
3.	3.8083	4.2143
4.	-0.3013	1.0053
5.	-0.6249	-2.7405

Table no 8. Particles of the PSO at iteration $(t) = 1$ with their fitness values.

Particles	**Positions**		**Fitness value**
$\boldsymbol{x_1}$	-0.0235	3.6090	13.0254
$\boldsymbol{x_2}$	-0.7273	4.3645	19.5778
$\boldsymbol{x_3}$	3.6661	-3.10	23.0129
$\boldsymbol{x_4}$	2.8107	3.9	23.1100
$\boldsymbol{x_5}$	-4.6071	-1.086	22.4048

After comparison of Tables 1. and 8. it can be seen that the particle $\boldsymbol{x_3^{(0)}}$ = (-0.1422,2.9010) is $gbest$ with fitness value 8.4360.

To get the $pbest$ value of each particle, we need to compare Table 1.and 8.

Iteration (t) = 2

For 1[th] particle $\mathbf{x}_1^{(1)} = (-\mathbf{0.0235}, \mathbf{3.6090})$:

$pbest = (-0.0235{,}3.6090)$

$gbest = (-0.1422{,}2.9010)$

$v_{id}^{t+1} = v_{id}^t + c_1 r_1 (x_{pi}^t - x_{id}^t) + c_2 r_2 (x_{gi}^t - x_{id}^t)$

$v_{11}^{(2)} = 0.8890 * (-4.7235) + 2 * 0.18 * (-0.0235 + 0.0235) + 2 * 0.68 * (-0.1422 + 0.0235)$

$v_{11}^{(2)} = -4.3606$ (accepted)

$x_{11}^{(2)}$ = -0.0235 -4.3606

$= -4.3841$ (accepted)

$v_{12}^{(2)} = 0.8890 * 3.8914 + 2 * 0.47 * (3.6090 - 3.6090) + 2 * 0.23 * (2.9010 - 3.6090)$

$= 3.4595 - 0.3257$

$x_{12}^{(2)} = 3.6090 + 3.1338$

$x_{12}^{(2)} = 6.7428$ (not accepted)

Re-initializing $x_{12}^{(2)}$

$x_{12}^{(2)} = -5 + 0.39 * 10 = -1.1$

$\Rightarrow \boldsymbol{x}_1^{(2)} = (-\mathbf{4.3841}, -\mathbf{1.1})$

For 2[nd] particle $\mathbf{x}_2^{(1)} = (-\mathbf{0.7273}, \mathbf{4.3645})$:

$pbest = (-0.7273, 4.3645)$

$gbest = (-0.1422, 2.9010)$

$v_{21}^{(2)} = 0.8890 * (-5) + 2 * 0.97 * (-0.7273 + 0.7273) + 2 * 0.32 * (-0.1422 + 0.7273)$

= -4.0705

$x_{21}^{(2)}$ = -0.7273 – 4.0705

= -4.7978

$v_{22}^{(2)} = 0.8890 * 0.2571 + 2 * 0.61 * (4.3645 - 4.3645) + 2 * 0.11 * (2.9010 - 4.3645)$

= -0.0940

$x_{22}^{(2)}$ = 4.3645 – 0.0940

= 4.2705

⇒ $\boldsymbol{x_2^{(2)}}$= (-4.7978 ,4.2705)

For 3nd particle $\boldsymbol{x_3^{(1)}}$ = (3.6610, -3.10):

$pbest = (-0.1422,2.9010)$

$gbest = (-0.1422,2.9010)$

$v_{31}^{(2)} = 0.8890 * 3.8083 + 2 * 0.45 * (-0.1422 - 3.6610) + 2 * 0.17 * (-0.1422 - 3.6610)$

= -1.3374

$x_{31}^{(2)}$ = 3.6610-1.3374

= 2.3236 (accepted)

$v_{32}^{(2)} = 0.8890 * 4.2143 + 2 * 0.39 * (2.9010 + 3.10) + 2 * 0.09 * (2.9010 + 3.10)$

$v_{32}^{(2)}$ = 9.5075 (*not accepted*)

Re-initializing $v_{32}^{(2)}$ = +5

$x_{32}^{(2)} = 2.5010 + 5 = 7.5010$ (not accepted)

Re-initializing $x_{32}^{(2)}$

$x_{32}^{(2)} = -5 + 0.79 * 10 = 2.9$

⇒ $\boldsymbol{x_3^{(2)}}$ = $(\mathbf{2.3236}, \mathbf{2.9})$

For 4th particle $\boldsymbol{x_4^{(1)}}$= $(\mathbf{2.8107}, \mathbf{3.90})$:

$pbest = (2.8107,3.90)$

$gbest = (-0.1422,2.9010)$

$v_{41}^{(2)} = 0.8890 * (-0.3013) + 2 * 0.42 * (2.8107 - 2.8107) + 2 * 0.16 * (-0.1422 - 2.8107)$

= -1.2128

$x_{41}^{(2)}$ = 2.8107 – 1.2128

= -1.5979

$v_{42}^{(2)} = 0.8890 * (1.0053) + 2 * 0.46 * (3.90 - 3.90) + 2 * 0.07 * (2.9010 - 3.90)$

= 1.0336

$x_{42}^{(2)}$ = 1.0079 +1.0336

= 2.0415

$\Rightarrow \boldsymbol{x_4^{(2)}} = (\mathbf{-1.5979, 2.0415})$

For 5th particle $x_5^{(1)}$= $(\mathbf{-4.6071, -1.0860})$:

$pbest = (-3.9822,1.6545)$

$gbest = (-0.1422,2.9010)$

$$v_{51}^{(2)} = 0.8890 * (-0.6249) + 2 * 0.12 * (-3.9822 + 4.6071) + 2 * 0.42 * (-0.1422 + 4.6071)$$

= 3.3449

$x_{51}^{(2)}$ = -1.2622 (accepted)

$v_{52}^{(2)}$ = w* $v'_{52} + c_1 * r_1 * \left(x'_{p1} - x'_{52}\right) + c_2 * r_2 * \left(x'_{g1} - x'_{52}\right)$

$$= 0.8890 * (-2.7405) + 2 * 0.11 * (1.6545 + 1.0860) + 2 * 0.50 * (2.9010 + 1.0860)$$

= 2.1536

$x_{52}^{(2)}$ = -1.0860 + 2.1536

= 1.0676

$\Rightarrow \boldsymbol{x_5^{(2)}} = (\mathbf{-1.2622, 1.0676})$

Table 9. Updated velocities and particles position along with their fitness values are provided in Table 9. And Table 10. respectively.

Particle velocity	**Velocities**	
1.	-4.3606	3.1338
2.	-4.0705	1.2201
3.	-1.3374	5.0000
4.	-1.2128	1.0336
5.	3.3449	2.1536

Table 10. Particles of the PSO at iteration $(t) = 2$ with their fitness values.

Particles	**Positions**		**Fitness value**
1.	-4.3841	-1.1	20.4303

2.	-4.7978	4.2705	41.2561
3.	2.3236	2.9	13.8091
4.	-1.5979	2.0415	6.7210
5.	-1.2622	1.0676	2.7329

After the second iteration we can see the final $gbest$ value is $\boldsymbol{x}_{\mathbf{5}}^{(\mathbf{2})} = (\mathbf{-1.2622}, \mathbf{2.7329})$ and its fitness value is 2.7329.

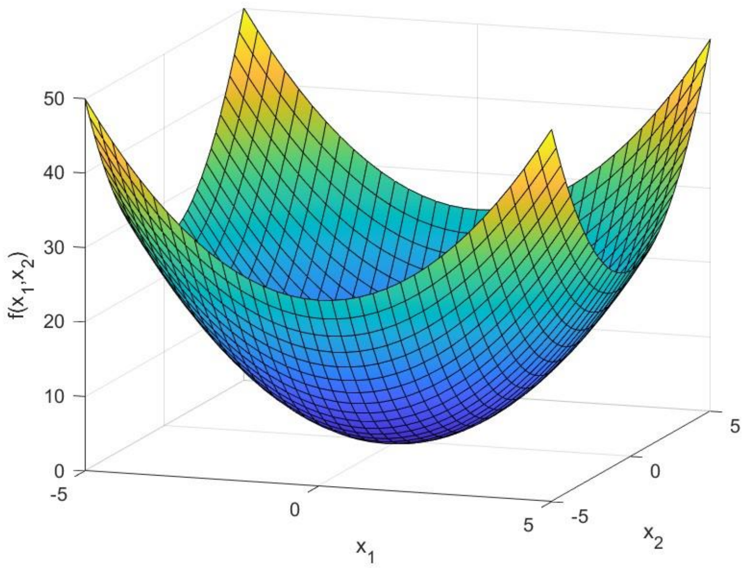
Implementation of PSO and modified PSO in MATLAB

We have considered the following optimization problem with sphere function to analyse the search performance of the PSO and modified PSO.

$$Min\ f(X) = \sum_{i=1}^{D} x_i^2$$

$$s.t. -5 \leq x_i \leq 5$$

Two-dimensional plot of the sphere function is shown in Figure, which also indicate that the minima lie at (0,0).



Parameter settings:

1. Swarm size (N) = 50
2. $c_1, c_2 = 2$
3. $w = wMax - l.* \; ((wMax - wMin) \; / \; Max\_iter)$

where, $wMax = 0.9$, $wMin = 0.2$

4. $Max \; iteration = 200$

Table 12. Numerical results produced by PSO and modified PSO

Dimension (D)	PSO	Modified PSO
$D = 2$	1.3708E-05	1.9852E-33
$D = 10$	1.4641E+00	3.8850E-12
$D = 20$	3.6012E+00	6.2518E-05
$D = 50$	3.4629E+01	4.0704E-01

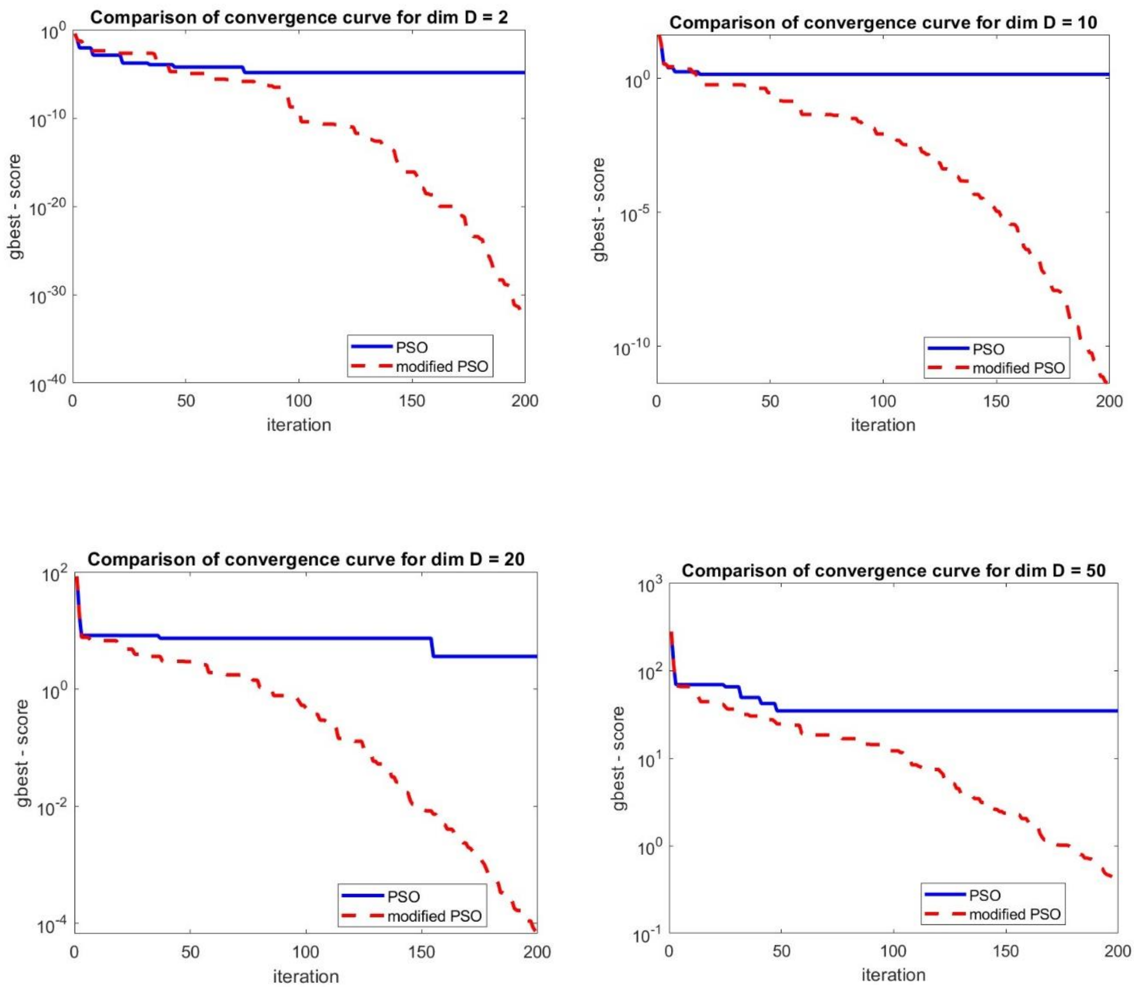


Fig. Convergence curves for dimension $D = 2, 10, 20, 50$ using PSO and modified PSO.

5. Concluding remarks and future research plan:

In this mini-project report, I have learned basic concepts of optimization like local and global optima, deterministic and non-deterministic algorithms, and heuristic and meta-heuristic algorithms, and nature inspired optimization algorithms. I will focus on nature inspired optimization algorithms as my Ph.D. research work. With the aim of this objective, I have started with learning of basic understanding of the PSO and one of its modified variants called modified PSO in this mini project. I have done hand-calculations on a simple example of two-dimensional sphere function with search space range $[-5,5]^D$. I have also implemented PSO and $mPSO$ on this optimization problem on MATLAB with $D = 2,10,20,$ and $50,$ and analyzed their performance using numerical results and convergence behavior.

In the future research work of my Ph.D., I will explore the literature review on other nature inspired algorithms and will continue my research by developing their advanced versions. These extensions will be done based on their performance on standard benchmark optimization problem and real-world applications.

References:

1. Lange, K. (2013). *Optimization* (Vol. 95). Springer Science & Business Media.
2. Osman, I. H., & Kelly, J. P. (1996). Meta-heuristics: an overview. *Meta-heuristics: Theory and applications*, 1-21.
3. Bansal, J. C. (2019). Particle swarm optimization. *Evolutionary and swarm intelligence algorithms*, 11-23.
4. Dorigo, M., Birattari, M., & Stutzle, T. (2006). Ant colony optimization. *IEEE computational intelligence magazine*, *1*(4), 28-39.

5. Emary, E., Zawbaa, H. M., Grosan, C., & Hassenian, A. E. (2015). Feature subset selection approach by gray-wolf optimization. In *Afro-European Conference for Industrial Advancement: Proceedings of the First International Afro-European Conference for Industrial Advancement AECIA 2014* (pp. 1-13). Springer International Publishing.

6. Tian, D., & Shi, Z. (2018). MPSO: Modified particle swarm optimization and its applications. *Swarm and evolutionary computation*, *41*, 49-68.