PARTICLE SWARM OPTIMIZATION

A DISSERTATION

Submitted in Partial Fulfillment of Requirements
For the award of the degree
of

Master of Science
In
Applied Mathematics

Submitted by:

Hansraj
Roll No. 2K21/MSCMAT/19

and

Bijesh Yadav
Roll No. 2K21/MSCMAT/09

Under the supervision of
Prof. Anjana Gupta

DEPARTMENT OF APPLIED MATHEMATICS
DELHI TECHNOLOGICAL UNIVERSITY
(Formerly Delhi College of Engineering)
Bawana Road, Delhi-110042

MAY,2023

DELHI TECHNOLOGICAL UNIVERSITY
(Formerly Delhi College of Engineering)
Bawana Road, Delhi-110042

CANDIDATE'S DECLARATION

I, **Hansraj (2K21/MSCMAT/19)** and **Bijesh Yadav (2K21/MSCMAT/09)** students of M.Sc in Applied Mathematics at Delhi Technological University, hereby declare that the submitted Dissertation titled "**Particle Swarm Optimization**" is original and has not been copied from any sources without proper citation. This work has not been used to obtain any other degree, diploma, associateship, fellowship, or recognition previously.

Place: Delhi
Date:

HANSRAJ
2K21/MSCMAT/19

BIJESH YADAV
2K21/MSCMAT/09

Prof. Anjana Gupta

SUPERVISOR

Department of Applied Mathematics
Delhi Technological University
Bawana Road, Delhi-110042

DEPARTMENT OF APPLIED MATHEMATICS

DELHI TECHNOLOGICAL UNIVERSITY

(Formerly Delhi College of Engineering)

Bawana Road, Delhi-110042

CERTIFICATE

I certify that the Project Dissertation titled "Particle Swarm Optimization" submitted by Hansraj (2K21/MSCMAT/19) and Bijesh Yadav (2K21/MSCMAT/09) in partial fulfillment of the requirement for the award of the degree of Master of Science from the Department of Applied Mathematics at Delhi Technological University, Delhi, is a record of the project work carried out by the students under my supervision. To the best of my knowledge, this work has not been submitted in part or full for any degree or diploma at this university or elsewhere.

Place: Delhi
Date:

Prof. Anjana Gupta

SUPERVISOR

Department of Applied Mathematics
Delhi Technological University
Bawana Road, Delhi-110042

ACKNOWLEDGEMENT

I would to express my sincere thanks and appreciation to my guide, Prof. Anjana Gupta, for his patient guidance, constant encouragement, endless support during my study.

I would be like to thank the Department of Applied Mathematics of Delhi Technological University (Formerly Delhi College of Engineering) for providing a productive scientific environment.

HANSRAJ
2K21/MSCMAT/19

BIJESH YADAV
2K21/MSCMAT/09

ABSTRACT

An optimisation algorithm based on the behaviors of social organisms is known as particle swarm optimizatio (PSO).It represents a set of potential answers to an optimisation issue as a swarm of moving particles in the parameter space. The performance of the particles is guided by their own performance and the performance of their neighbors, leading to an optimized solution. This thesis presents a study of the impact of boundary conditions on the performance of Particle Swarm Optimization (PSO) through the use of the invisible wall technique. The convergence behaviors of PSO are analyzed and its application to discrete-valued problems and multi-objective optimization problems are discussed. Additionally, practical applications of PSO are explored. We are solved linear programming problems, transportation problem using Particle Swarm Optimization and applying on a Data Set.

Keywords: Particle Swarm Optimization (PSO), Swarm Intelligence, Current Position, Current Velocity, individual perfect or (Pbest), Global Best (Gbest), Multi-objective.

Contents

List of Figures

List of Tables

Abbreviations

- **Z :** A component that is being maximised or minimised. A scalar number is output after receiving a vector as input.
- $V_{i,j}^k$ **:** The velocity vector of particle i in dimension j at time k.
- $X_{i,j}^k$ **:** The position vector of particle i in dimension j at time k.
- $Pbest_{i,j}^k$ **:** The personal best position of particle i in dimension j obtained from initialization through time k.
- $Gbest_{i,j}^k$ **:** The global best position of particle i in dimension j obtained from initialization through time k.
- c_1 **,** c_2 **:** Positive acceleration constants which are used to level the contribution of the cognitive and social components respectively.
- r_1 **,** r_2 **:** Random numbers from uniform distribution $U(0,1)$ at time t.
- $\omega(t)$ **:** Inertia Weight.
- **n :** The swarm size or number of particles.
- **D :** The maximum number of dimensions.
- **N :** Total number of iterations.

Chapter 1

Preliminaries

1.1 Introduction

Dr. Kennedy and Dr. Eberhart. first proposed the based on populations probabilistic search approach known as Particle Swarm Optimization (PSO) in 1995 [1]. PSO's fundamental concept was influenced by how creatures interact with one another, such as when flocks of birds or schools of fish, and it provides an alternative approach to solving non-linear optimization problems. PSO depends on a group interaction method observed in animals such as birds and insects when they search for food or migrate. The algorithm simulates the sharing of individual information among group members to identify the positive direction in a search space. If one member of the group finds a positive direction, the others will quickly follow, reflecting the behavior seen in natural social systems [2].

The PSO algorithm uses the concept of a swarm and particles to solve optimization problems by simulating the behavior of animals. Every particle in the population represents a potential solution and traverses the search space, starting from a random location and moving in random directions. The particles remember their best past locations and those of their neighbors, and constantly modified their location and velocity based on the best locations found by the entire population. The particles communicate and transmit favorable positions to each other. The search process continues until the swarm converges towards the maximum of the fitness function $g : \mathcal{R}^n \to \mathcal{R}$.

The PSO algorithm is growing in popularity due to its ease of implementation and ability to converge quickly on a practical solution to optimization problems. In comparison to other optimization techniques, it is quicker, less expensive, and more effective. Additionally, PSO only has a few parameters that may be changed. PSO is a great tool for solving optimization problems because of this. Non-convex, continuous, discrete, integer variable problems are a good fit for PSO [2].

1.2 Swarm Intelligence

Swarm Intelligence (SI) is a paradigm for problem solving inspired by the collective behavior of decentralized, self-organized systems, both in nature and artificial systems. Ant communities, fish schooling, bird flocking, and bee swarming are a few examples of natural SI. Multi-robot systems and data analysis and optimisation software are examples of artificial SI systems. Two of the most effective swarm intelligence methods are PSO and Ant Colony Optimization (ACO). In PSO, each particle, which represents a potential solution, moves through the multidimensional search space and modifies its position in response to its own and its peers' experiences, ultimately leading the entire swarm to converge towards an optimal solution [3].

Chapter 2

Background

2.1 Optimization

Optimization is the technique of finding the perfect answer for a question, considering given constraints and objectives. It can involve either minimization or maximization of a specific metric, as the two tasks are mathematically equivalent by taking the additive inverse of the function. Optimization plays a critical role in various industries and professions, as decision-makers are often required to make choices that minimize effort or maximize benefits. It is a crucial aspect of problem-solving and decision-making in many fields i.e, management, engineering, finance and more.

Based on the characteristics of the objective function, optimization problems can be divided into two major groups: linear optimization problems and non-linear optimization problems. Linear optimization problems involve linear relationships between variables and constraints, making them easier to solve compared to non-linear optimization problems. Non-linear optimization problems, on the other hand, are generally more complex and difficult to solve due to the non-linear relationships between variables and constraints. Optimization difficulties are divided into the following categories based on the characteristics of the problem:

2.1.1 Constrained Optimization

Constrained optimization is a type of mathematical optimization problem where the objective function is subject to a set of constraints. The goal of constrained optimization is to obtained the optimal value of the objective function, subject to the constraints.

The constraints can be of different types, such as equality constraints, inequality constraints, or a combination of both. Equality constraints are conditions that must be satisfied exactly, while inequality constraints impose limits on the values that the decision variables can take.

A constrained maximisation problem's standard type [4] is described as follows:

$$\begin{aligned} &\underset{z}{\text{Maximize}} && k(z) \\ &\text{subject to} && l_i(z) = c_i,\ i = 1, \dots n. \\ & && m_j(x) \geq d_j,\ j = 1, \dots m. \end{aligned} \tag{2.1}$$

where z is vector of decision's variable, $k(z)$ is objective function that needs to be improved.

2.1.2 Unconstrained Optimization

Unconstrained optimisation refers to a class of mathematical optimisation problems where there are no restrictions on the values of the decision factors and where the objective function must be optimised. Finding the ideal values for the decision factors that maximise or minimise the objective function is the aim of unconstrained optimisation.

The decision variables are the variables that are being optimized, and the objective function is the function that is being optimized. In unconstrained optimization [4], the objective function can be of any form, including linear, quadratic, nonlinear, or even discontinuous functions.Many fields of study, including physics, engineering, machine learning, and finance, face unconstrained optimisation problems. Examples of unconstrained optimization problems include parameter estimation, function fitting, and model selection.

$$\underset{z}{\text{Minimize}} \quad g(z) \ \ ,\ z \in\ \mathcal{R}^n \tag{2.2}$$

where n is dimension of z.

2.1.3 Dynamic Optimization

Dynamic optimization is a type of optimization problem where the decision variables are determined over time. In dynamic optimization [4], the objective is to optimize a function that changes over time, subject to constraints that also vary with time. This type of optimization is used to solve problems where the decision-making process evolves over time, such as in financial planning, environmental management, or control systems.

$$\begin{aligned} &\underset{x}{\text{Minimize}} && g(y, \omega(t)), \quad y = (y_1, y_2, ...y_n)\,, \quad \omega(t) = (\omega_1(t), \omega_2(t), ...\omega_n(t)) \\ &\text{subject to} && k_m(y) \leq 0,\ m = 1, \dots n. \\ & && h_m(y) = 0,\ m = 1, \dots n. \end{aligned} \tag{2.3}$$

where $y(t)$ is the optimal value determined at step t and $\omega(t)$ is a vector of time-dependent objective function control parameters.

2.2 Global Optimization

A global minimum is defined as z^* such that

$$g(z^*) \leq g(z) \qquad \forall\, z \, \in \, S \tag{2.4}$$

where S is the search space and $S = \mathcal{R}^n$.

2.3 Local Optimization

A local minimum is defined as z_L^* of the region L such that

$$g(z_L^*) \leq g(z) \qquad \forall\, z \, \in \, L \tag{2.5}$$

where $L \subseteq \mathcal{R}^n$.

2.3.1 Example

Consider a function $f(x) = x^4 - 12x^3 + 47x^2 - 75x + 10$

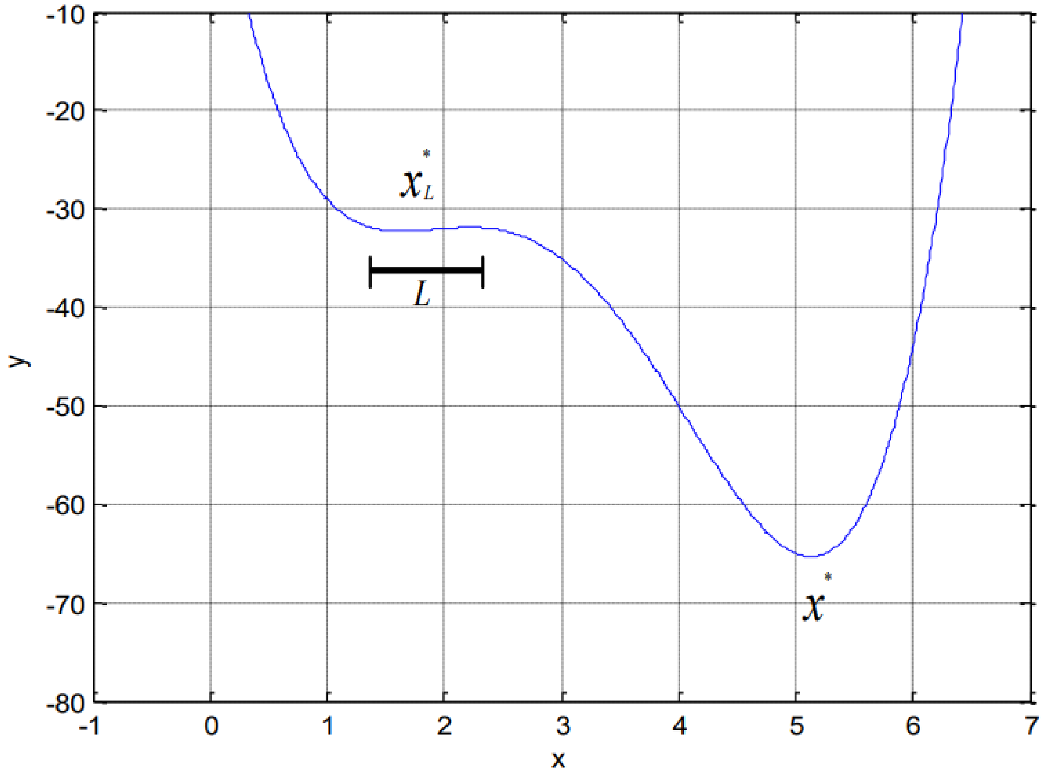


Figure 2.1: Explanation of local minimum and global minimum

2.4 Uniform Distribution

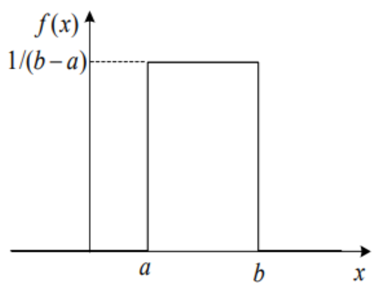
The uniform distribution is a probability distribution that is defined on a finite interval and assigns equal probability density to each point within that interval. In other words, the probability of any point in the interval is the same. $U\left(a,b\right)$, where a and b ar the distribution's lowest and greatest values, respectively, defines it.

The probability density function (PDF) of a continuous uniform distribution on the interval $[a,b]$ is :

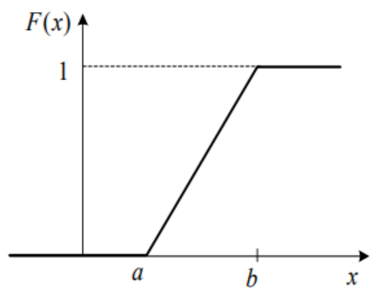
$$f\left(x\right)=\begin{cases}0 & for\ \ x\ <\ a\\ \frac{1}{b-a} & for\ \ a\ \leq\ x\ \leq\ b\\ 0 & for\ \ x\ >\ b\end{cases}\tag{2.6}$$

and the cummulative distribution function (CDF) are:

$$F\left(x\right)=\begin{cases}0 & for\ \ x\ <\ a\\ \frac{x-a}{b-a} & for\ \ a\ \leq\ x\ \leq\ b\\ 0 & for\ \ x\ >\ b\end{cases}\tag{2.7}$$



Uniform PDF



Uniform CDF

The term **standard uniform distribution** refers to $U\left(0,1\right)$.

Chapter 3

Particle Swarm Optimization

The algorithm of finding the most suitable option from a group of alternatives according to one or more user-specified factors is known as optimisation [5]. This is typically done mathematically by representing the objective as a parameterized function f that depends on D parameters. The problem with optimization's is finding at the parameter values that maximize the objective function g. The objective function is also known as the "fitness function" and the optimization process involves finding the values that lead to the maximum of the fitness function. The focus will be on maximizing the function in the following:

$$\begin{aligned} \text{Given} \;\; & g : \mathbb{R}^D \rightarrow \mathbb{R} \\ \text{Find} \;\; & \mathbf{z}_{opt} | g\left(\mathbf{x}_{opt}\right) \geq g(z) \quad \forall z \in \mathbb{R}^D \end{aligned} \tag{3.1}$$

The search (or parameter) space is the D-dimensional domain of the function $\mathbb{R}^D$ and each of its points, denoted by the vector of coordinates $\mathbf{z}$ represents possible solutions to the problems, with $\mathbf{z}_{opt}$ being the best option i.e, the one that maximizes g [6].
In context of optimization, the social behaviour of fish groups and bird flocks served as the basis for Particle Swarm Optimization (PSO). According to its own velocity and input from other particles in the swarm, each particle in PSO is treated as a point in an N-dimensional space and has its position adjusted. The following details are used to change the particle's position [5, 6]:

- The particle's present location
- The particle's speed at the moment
- The difference between a particle's present location and its most well-known location (Pbest)
- The distance between the present location and the swarm's overall best-known position (Gbest).

3.1 PSO Algorithm

In PSO, Consider a population (swarm) size of N with position vector $\mathbf{X}_i{}^k = [x_1, x_2, x_3, ..., x_n]^T$ where T is transpose, and velocity vector $\mathbf{V}_i{}^k = [v_1, v_2, v_3, ..., v_n]^T$ each of the i particles that make up it at k iterations. The following equation states how the number j affects these vectors:

$$\mathbf{X}_{i,j}^{k+1} = \mathbf{X}_{i,j}^{k} + \mathbf{V}_{i,j}^{k+1} \tag{3.2}$$

where k and $k+1$ indicate two additional iterations of the algorithms and v_i is the vector containing the velocity components of the i-th particle. The three terms that make up the velocity vectors, which control how the following particles travel through the search space: the first term, defined as inertia or momentum, maintains track of the previous flow direction to prevent a quick direction change by the particle; the second term,known as the cognitive component, which explains why particles have a tendency to revert to previously determined optimal locations; the last term, known as the social component, indicates a particle's tendency to migrate to the optimal location for the entire swarm (depending on whether a global or partial PSO is used, or of a small area around the particle) [6]. These factors lead to the following definition of the i-th particle's velocity:

$$V_{i,j}^{k+1} = V_{i,j}^{k} + c_1.r_1^k \left(pbest_{i,j}^k - X_{i,j}^k\right) + c_2.r_2^k \left(gbest_j^k - X_{i,j}^k\right) \tag{3.3}$$

The terms "personal best" (pbest) and "global best" (gbest) are used to denote the best position a particle has attained thus far and the best position attained by the complete swarm respectively in particle swarm optimisation (PSO). The size of the steps the particle takes towards its individual and collective best places is determined by two constants the "cognitive coefficient" (c_1) and "social coefficient" (c_2) The optimisation process uses the random matrices R_1 and R_2 to create a stochastic effect on the velocity update [6]. The velocity of a particle, which dictates its subsequent movement in the search space, is updated using these coefficients and matrices.

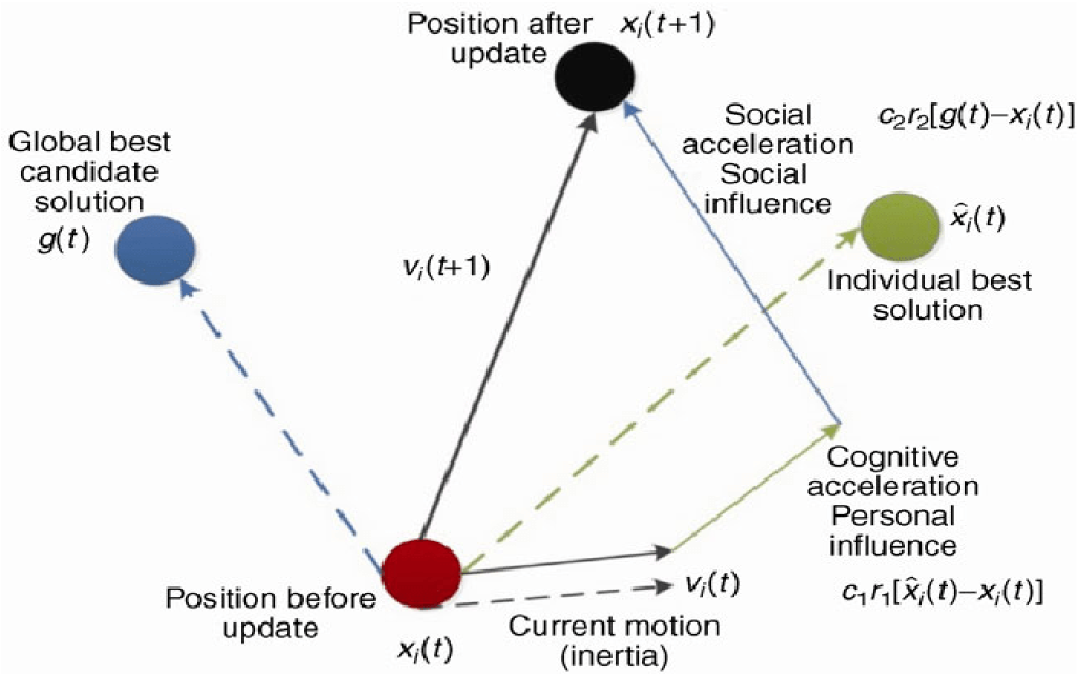


Figure 3.1: Implemanting the PSO

Let's assume $k = t$ in above figure:

The following equation can be used to mathematically describe how the particle's velocity changes:

$$V_{i,j}^{k+1} = \omega . V_{i,j}^{k} + c_1 . r_1^k \left(pbest_{i,j}^k - X_{i,j}^k \right) + c_2 . r_2^k \left(gbest_j^k - X_{i,j}^k \right) \tag{3.4}$$

3.1.1 Steps of Algorithm

1. Initialization
   (a) For each particle i in a swarm population size P.
      i. Initialize X_i randomly.
      ii. Initialize V_i randomly.
      iii. Evaluate fitness value $g\left(X_i\right)$.
      iv. Initialize $pbest_i$ with the help of X_i.
   (b) Initialize $gbest$ with the help of X_i with the best fitness
2. Repeat untill the stopping criteria satisfied.
   (a) For each particle i:
      i. Update X_i^k and V_i^k according to (3.2) and (3.4)
      ii. Evaluate fitness $g\left(X_i^k\right)$.
      iii. $pbest_i \leftarrow X_i^k$ if $g\left(pbest_i\right) < g\left(X_i^k\right)$
      iv. $gbest_i \leftarrow X_i^k$ if $g\left(gbest_i\right) < g\left(X_i^k\right)$

3.2 Flowchart

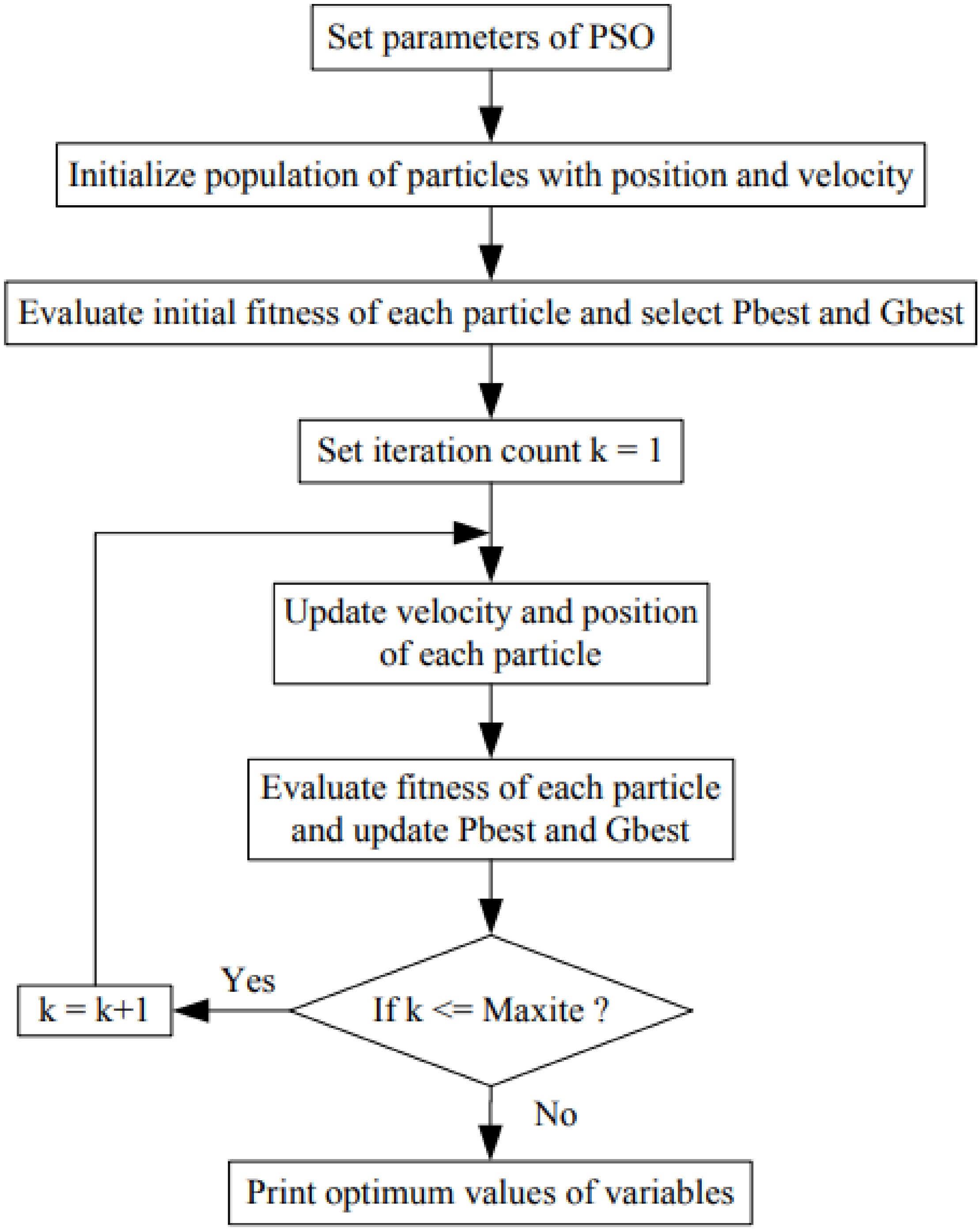


Figure 3.2: Flowchart of PSO [7]

3.3 Acceleration Constant C_1 and C_2

As from Eq 3.3, The amounts by which the particles move in the same way as the individual and global best particle are determined by the acceleration constants c_1 and c_2, adjusting the relative contributions of the social and cognitive aspects [8] or [9]. A number of authors have examined how these coefficients affect the trajectory of the particles and the algorithm's convergence properties, and their findings demonstrate that as the higher acceleration constants, the frequency of the particle's oscillation around the optimum increases while smaller values produce sinusoidal patterns. It has been demonstrated that the following situations:

$c_1 = c_2 = 2$

3.4 Inertial Weight Factor

Some authors advise using a combination of $\omega_{max} = 0.9$ and $\omega_{min} = 0.4$ for the best performance. Implementations of linearly reduced inertial weight have demonstrated that it provides very excellent results in many real-world applications. Overall, Bansal et al.'s [10] comparison of a set of common optimisation functions demonstrates that chaotic reduced inertia weights are the best fit (resulting in the lowest error mean in a set of 30 repeated simulations) while stochastic inertial weights are better if faster convergence is desired. However, the methods that result in the lowest error are linear and constant decreasing inertial weighting.

Table 3.1: Inertia weight dynamic adjustment methods

Strategy	Inertia weight
Constant weight of inertia	$\omega(t) = \omega = \text{const}$
Random weight of inertia	$\omega(t) = 0.5 + \frac{r}{2} \quad r \sim (0,1)$
Reducing inertia weight linearly	$\omega(t) = \omega_{max} - \frac{\omega_{max}-\omega_{min}}{t_{max}} t$
Chaotic random inertia weight	$\omega(t) = 0.5r_1 + 0.5z$ $z = 4r_2(1-r_2) with r_1, r_2 \sim U(0,1)$

3.5 Linear Programming Problem

$$\begin{aligned}
\underset{x}{\text{Maximize}} \quad & Z = X_1 + X_2 \\
\text{subject to} \quad & X_1 + X_2 \leq 8 \\
& 2X_1 + X_2 \leq 10 \\
& X_1, X_2 \geq 0
\end{aligned}$$

Parameter are set to be

- Population size = 3
- c_1 and $c_2 = 2$
- dimension of problem = 2
- The random numbers r_1 and r_2 range from 0 to 1.

The movement of particles are

$$V_{i,j}^{k+1} = \omega.V_{i,j}^{k} + c_1.r_1^k \left(pbest_{i,j}^k - X_{i,j}^k\right) + c_2.r_2^k \left(gbest_j^k - X_{i,j}^k\right)$$

$$X_{i,j}^{k+1} = X_{i,j}^{k} + V_{i,j}^{k+1}$$

Calculate initial position

$$x = l + r\left(u - l\right)$$

LB = [0 0]
UB = [5 8]

$$\begin{aligned}
X_1 &= 0 + 0.12(5-0) = 0.6 \\
&= 0 + 0.1(5-0) = 0.5 \\
&= 0 + 0.2(5-0) = 1
\end{aligned}$$

$$\begin{aligned}
X_2 &= 0 + 0.80(8-0) = 6.4 \\
&= 0 + 0.9(8-0) = 7.2 \\
&= 0 + 0.7(8-0) = 5.6
\end{aligned}$$

Table 3.2: Initial Position

	X_1	X_2	$Max\ Z = X_1 + 2X_2$
1	0.6	6.4	13.4
2	0.5	7.2	14.9
3	1	5.6	12.2

$$Gbest = [0.5 \quad 7.2]$$

let initial velocity $V_i = 0$

Iteration 1: Calculate new velocity and position from eqn (3.2) and (3.4)

$$\begin{aligned} V_{11} &= 0.9 \times 0 + 2 \times 0.1(0.6 - 0.6) + 2 \times 0.4(0.5 - 0.6) \\ &= 0 + 0 + (-0.08) \\ &= -0.08 \end{aligned}$$

$$\begin{aligned} V_{12} &= 0.9 \times 0 + 2 \times 0.1(6.4 - 6.4) + 2 \times 0.4(7.2 - 6.4) \\ &= 0 + 0 + 0.64 \\ &= 0.64 \end{aligned}$$

$$\begin{aligned} V_{21} &= 0.9 \times 0 + 2 \times 0.1(0.5 - 0.5) + 2 \times 0.4(0.5 - 0.5) \\ &= 0 + 0 + 0 \\ &= 0 \end{aligned}$$

$$\begin{aligned} V_{22} &= 0.9 \times 0 + 2 \times 0.1(7.2 - 7.2) + 2 \times 0.4(7.2 - 7.2) \\ &= 0 + 0 + 0 \\ &= 0 \end{aligned}$$

$$\begin{aligned} V_{31} &= 0.90 \times 0 + 2 \times 0.1(1.0 - 1.0) + 2 \times 0.4(0.5 - 1.0) \\ &= 0 + 0 + (-0.4) \\ &= -0.4 \end{aligned}$$

$$\begin{aligned} V_{32} &= 0.90 \times 0 + 2 \times 0.1(5.6 - 5.6) + 2 \times 0.4(7.2 - 5.6) \\ &= 0 + 0 + 1.28 \\ &= 1.28 \end{aligned}$$

Updated velocity

Table 3.3: Updated Velocity

	V_1	V_2
1	-0.08	0.64
2	0	0
3	-0.4	1.28

Now, update position = updated velocity + previous position
Pbest =

Table 3.4: Updated Position

	X_1	X_2	$Max\ Z = X_1 + 2X_2$
1	0.52	7.04	14.6
2	0.5	7.2	14.9
3	0.6	6.88	14.36

Now fitness value same 14.9 then, Gbest remain same

$$Gbest = [0.5 \quad 7.2]$$

Iteration 2: Again, calculate new velocity and position

$$\begin{aligned}
V_{11} &= 0.9 \times (-0.08) + 2 \times 0.1(0.52 - 0.52) + 2 \times 0.4(0.5 - 0.52)\\
&= -0.072 + 0 + (-0.016)\\
&= -0.088\\
V_{12} &= 0.90 \times 0.64 + 2 \times 0.10(7.04 - 7.04) + 2 \times 0.4(7.20 - 7.04)\\
&= 0.576 + 0 + 0.128\\
&= 0.704\\
V_{21} &= 0.9 \times 0 + 2 \times 0.1(0.5 - 0.5) + 2 \times 0.4(0.5 - 0.5)\\
&= 0 + 0 + 0\\
&= 0\\
V_{22} &= 0.9 \times 0 + 2 \times 0.1(7.2 - 7.2) + 2 \times 0.4(7.2 - 7.2)\\
&= 0 + 0 + 0\\
&= 0\\
V_{31} &= 0.9 \times (-0.04) + 2 \times 0.1(0.6 - 0.6) + 2 \times 0.4(0.5 - 0.6)\\
&= -0.36 + 0 + (-0.08)\\
&= -0.44\\
V_{32} &= 0.9 \times 1.28 + 2 \times 0.1(6.88 - 6.88) + 2 \times 0.4(7.2 - 6.88)\\
&= 1.152 + 0 + 0.256\\
&= 1.408
\end{aligned}$$

Updated velocity

Table 3.5: Updated Velocity

	V_1	V_2
1	-0.088	0.704
2	0	0
3	-0.44	1.408

Now, update position = updated velocity + previous position

1.1408 + 6.88 = 8.288 $\notin$ (0,8) then 8
Pbest =

Table 3.6: Updated Position

	X_1	X_2	$Max\ Z = X_1 + 2X_2$
1	0.432	7.74	15.91
2	0.5	7.2	14.9
3	0.16	8	16.16

Now fitness value 16.16 > 14.9 then update Gbest

$$Gbest = [0.16 \quad 8]$$

Iteration 3: Calculate new velocity and position

$$\begin{aligned}
V_{11} &= 0.9 \times (-0.088) + 2 \times 0.1(0.432 - 0.432) + 2 \times 0.4(0.16 - 0.432) \\
&= -0.072 + 0 - 0.2176 \\
&= -0.2968 \\
V_{12} &= 0.9 \times 0.704 + 2 \times 0.1(7.74 - 7.74) + 2 \times 0.4(8 - 7.74) \\
&= 0.633 + 0 + 0.208 \\
&= 0.841 \\
V_{21} &= 0.9 \times 0 + 2 \times 0.1(0.5 - 0.5) + 2 \times 0.4(0.16 - 0.5) \\
&= 0 + 0 - 0.272 \\
&= -0.272 \\
V_{22} &= 0.9 \times 0 + 2 \times 0.1(7.2 - 7.2) + 2 \times 0.4(8 - 7.2) \\
&= 0 + 0 + 0.64 \\
&= 0.64
\end{aligned}$$

$$\begin{aligned}
V_{31} &= 0.9 \times (-0.44) + 2 \times 0.1(0.16 - 0.16) + 2 \times 0.4(0.16 - 0.16) \\
&= -0.396 + 0 + 0 \\
&= -0.396 \\
V_{32} &= 0.9 \times 1.408 + 2 \times 0.1(8 - 8) + 2 \times 0.4(8 - 8) \\
&= 1.2672 + 0 + 0 \\
&= 1.2672
\end{aligned}$$

Updated velocity

Table 3.7: Updated Velocity

	V_1	V_2
1	-0.296	0.841
2	-0.272	0.64
3	-0.396	1.267

Now, update position = updated velocity + previous position

0.16 + (-0.396) = -0.236 $\notin$ (0,5) then 0

7.74 + 0.841 = 8.581 $\notin$ (0,8) then 8

Table 3.8: Updated Position

	X_1	X_2	$Max\ Z = X_1 + 2X_2$
1	0.136	8	16.136
2	0.228	7.8	15.82
3	0	8	16

Pbest =

Table 3.9: Pbest

	X_1	X_2	$Max\ Z = X_1 + 2X_2$
1	0.136	8	16.136
2	0.228	7.8	15.82
3	0.16	8	16.16

then update Gbest

$$Gbest = [0.136 \quad 8]$$

Iteration 4: Calculate new velocity and position

$$
\begin{aligned}
V_{11} &= 0.9 \times (-0.296) + 2 \times 0.1(0.136 - 0.136) + 2 \times 0.4(0.136 - 0.136) \\
&= -0.2664 + 0 + 0 \\
&= -0.2664 \\
V_{12} &= 0.9 \times 0.841 + 2 \times 0.1(8 - 8) + 2 \times 0.4(8 - 8) \\
&= 0.7569 + 0 + 0 \\
&= 0.7569 \\
V_{21} &= 0.9 \times (-0.272) + 2 \times 0.1(0.228 - 0.228) + 2 \times 0.4(0.136 - 0.228) \\
&= -0.3184 \\
V_{22} &= 0.9 \times 0.64 + 2 \times 0.1(7.8 - 7.8) + 2 \times 0.4(8 - 7.5) \\
&= 0.736 \\
V_{31} &= 0.9 \times (-0.396) + 2 \times 0.1(0 - 0.16) + 2 \times 0.4(0.136 - 0.16) \\
&= -0.2476 \\
V_{32} &= 0.9 \times 1.267 + 2 \times 0.1(8 - 8) + 2 \times 0.4(8 - 8) \\
&= 1.1403 + 0 + 0 \\
&= 1.1403
\end{aligned}
$$

Updated velocity

Table 3.10: Updated Velocity

	V_1	V_2
1	-0.266	0.7569
2	-0.318	0.73
3	-0.247	1.140

Table 3.11: Updated Position

	X_1	X_2	$Max\ Z = X_1 + 2X_2$
1	0	8	16
2	0	8	16
3	0	8	16

then,

$$|f(x)_{prev} - f(x)| = 0$$

$$|16.13 - 16| = 0$$

$$0.13 \geq 0$$

then by particle swarm optimization

$X_1 = 0$, $X_2 = 8$, $Z = 16$

3.5.1 Computational Result:

```
**********************************************************
Final Results-----------------------------

bestfun =

   -16


bestrun =

     1


best_variables =

     0     8

**********************************************************
Elapsed time is 0.249822 seconds.
```

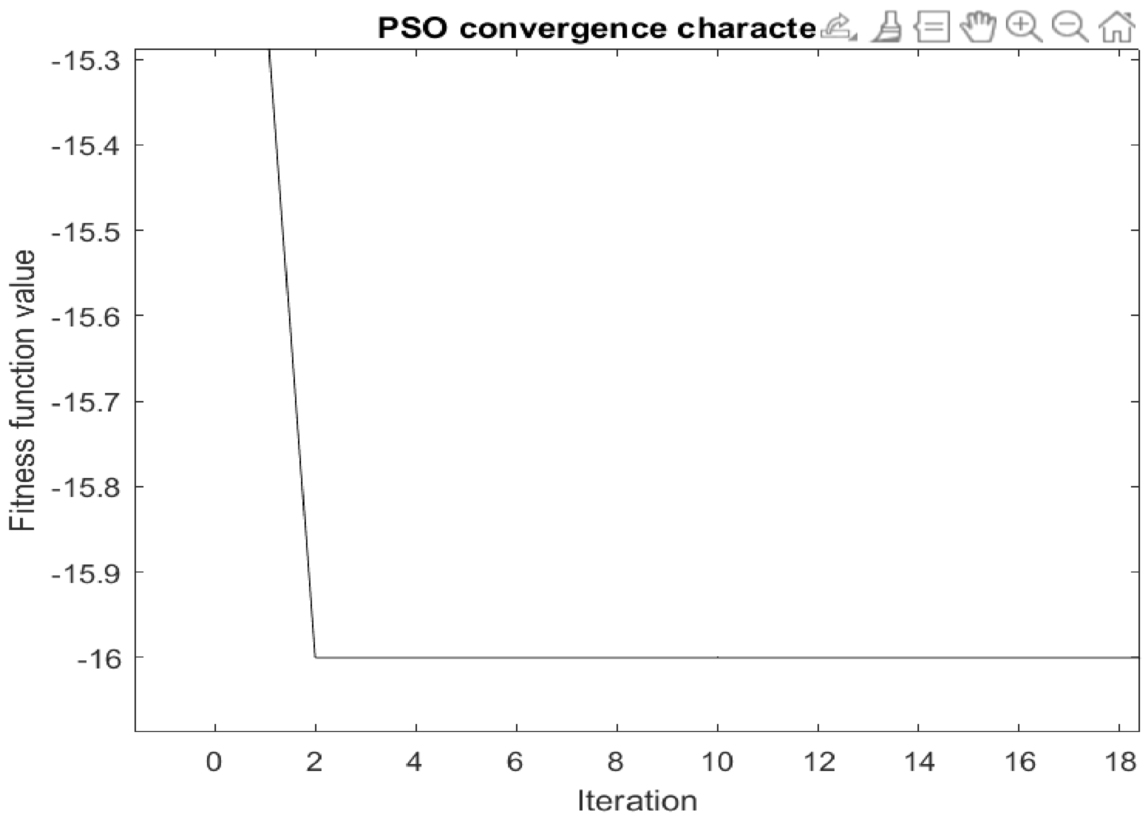


Figure 3.3: Convergence Characterstics [11]

Chapter 4

Transportation Problem

			a_i
	1	1	4
	2	1	6
b_j	5	5	

$$\sum a_i = \sum b_j = 10$$

4.1 Example

Maximize profit

$$Max\ \ Z \ = \ C_{11}X_{11} + C_{12}X_{12} + C_{21}X_{21} + C_{22}X_{22}$$

Subject to

$$X_{11} + X_{12} = 4$$
$$X_{21} + X_{22} = 6$$
$$X_{11} + X_{21} = 5$$
$$X_{12} + X_{22} = 5$$

then

$$Max\ \ Z \ = \ X_1 + X_2 + 2X_3 + X_4$$

Subject to

$$X_1 + X_2 = 4$$
$$X_3 + X_4 = 6$$
$$X_1 + X_3 = 5$$
$$X_2 + X_4 = 5$$
$$X_1, X_2, X_3, X_4 \geq 0$$

Parameter are set to be

- Population size = 3
- c_1 and c_2 = 2
- dimension of problem = 4
- The random numbers r_1 and r_2 range from 0 to 1.

The movement of particles are

$$V_{i,j}^{k+1} = \omega.V_{i,j}^{k} + c_1.r_1^k \left(pbest_{i,j}^k - X_{i,j}^k\right) + c_2.r_2^k \left(gbest_j^k - X_{i,j}^k\right)$$

$$X_{i,j}^{k+1} = X_{i,j}^{k} + V_{i,j}^{k+1}$$

Calculate initial position

$$x = l + r\,(u - l)$$

Table 4.1: Initial Position

	X_1	X_2	X_3	X_4	$Max\ Z = X_1 + X_2 + 2X_3 + X_4$
1	1.2	3.6	4	0.92	13.72
2	0.75	3.2	4.5	0.8	13.75
3	1.6	3.4	3.5	0.7	9.2

$$Gbest \;=\; [0.75\;\; 3.2\;\; 4.5\;\; 0.8]$$

initial velocity = 0.1 × initial position

Table 4.2: Initial velocity

	V_1	V_2	V_3	V_4
1	0.12	0.36	0.4	0.092
2	0.075	0.32	0.45	0.08
3	0.16	0.34	0.35	0.07

Iteration 1: Calculate new velocity and position from eqn (3.2) and (3.4)

$$
\begin{aligned}
V_{11} &= 0.9 \times 0.12 + 2 \times rand()(1.2 - 1.2) + 2 \times rand()(0.75 - 1.2) \\
&= -0.5984 \\
V_{12} &= 0.9 \times 0.36 + 2 \times rand()(3.6 - 3.6) + 2 \times rand()(3.2 - 3.6) \\
&= 0.244 \\
V_{13} &= 0.9 \times 0.4 + 2 \times rand()(4 - 4) + 2 \times rand()(4.5 - 4) \\
&= 1.1740 \\
V_{14} &= 0.9 \times 0.092 + 2 \times rand()(0.92 - 0.92) + 2 \times rand()(0.8 - 0.92) \\
&= 0.0588 \\
V_{21} &= 0.9 \times 0.075 + 2 \times rand()(0.75 - 0.75) + 2 \times rand()(0.75 - 0.75) \\
&= 0.0675 \\
V_{22} &= 0.9 \times 0.32 = 0.288 \\
V_{23} &= 0.9 \times 0.45 = 0.405 \\
V_{24} &= 0.9 \times 0.092 = 0.072 \\
V_{31} &= 0.9 \times 0.16 + 2 \times rand()(1.6 - 1.6) + 2 \times rand()(0.75 - 1.6) \\
&= -0.8703 \\
V_{32} &= 0.9 \times 0.34 + 2 \times rand()(3.4 - 3.4) + 2 \times rand()(3.2 - 3.4) \\
&= 0.0252 \\
V_{33} &= 0.9 \times 0.35 + 2 \times rand()(3.5 - 3.5) + 2 \times rand()(4.5 - 3.5) \\
&= 1.1789 \\
V_{34} &= 0.9 \times 0.07 + 2 \times rand()(0.7 - 0.7) + 2 \times rand()(0.8 - 0.7) \\
&= 0.1716
\end{aligned}
$$

Table 4.3: Updated velocity

	V_1	V_2	V_3	V_4
1	-0.5984	0.244	1.1740	0.0588
2	0.0675	0.288	0.405	0.072
3	-0.8703	0.0252	1.1789	0.1716

Updated Position

Table 4.4: Updated Position

	X_1	X_2	X_3	X_4	$Max\ Z = X_1 + X_2 + 2X_3 + X_4$
1	0.601	3.844	5	0.978	15.42
2	0.817	3.488	4.905	0.872	14.98
3	0.729	3.425	5	0.871	15.02

$$Gbest = [0.601 \;\; 3.844 \;\; 5 \;\; 0.978]$$

Iteration 2: Similarly, we update new velocity and position

$$
\begin{aligned}
V_{11} &= 0.7 \times (-0.5984) = -0.4188 \\
V_{12} &= 0.7 \times 0.244 = 0.1708 \\
V_{13} &= 0.7 \times 1.1740 = 0.821 \\
V_{14} &= 0.7 \times 0.0588 = 0.0411 \\
V_{21} &= 0.7 \times 0.0675 + 2 \times rand()(0.817 - 0.817) + 2 \times rand()(0.601 - 0.817) \\
&= -0.1687 \\
V_{22} &= 0.7 \times 0.288 + 2 \times rand()(3.488 - 3.488) + 2 \times rand()(3.844 - 3.488) \\
&= 0.3312 \\
V_{23} &= 0.7 \times 0.405 + 2 \times rand()(4.905 - 4.905) + 2 \times rand()(5 - 4.905) \\
&= 0.3716 \\
V_{24} &= 0.7 \times 0.072 + 2 \times rand()(0.872 - 0.872) + 2 \times rand()(0.978 - 0.872) \\
&= 0.2444 \\
V_{31} &= 0.7 \times (-0.8703) + 2 \times rand()(0.729 - 0.729) + 2 \times rand()(0.601 - 0.729) \\
&= -0.6096 \\
V_{32} &= 0.7 \times 0.02552 + 2 \times rand()(3.42 - 3.425) + 2 \times rand()(3.844 - 3.425) \\
&= 0.1923 \\
V_{33} &= 0.7 \times 1.7849 + 2 \times rand()(5 - 5) + 2 \times rand()(5 - 5) \\
&= 1.249 \\
V_{34} &= 0.7 \times 0.1716 + 2 \times rand()(0.871 - 0.871) + 2 \times rand()(0.978 - 0.871) \\
&= 0.1474
\end{aligned}
$$

Table 4.5: Updated velocity

	V_1	V_2	V_3	V_4
1	-0.4188	0.1708	0.821	0.0411
2	-0.1687	0.3312	0.3716	0.244
3	-0.6096	0.1923	1.249	0.1474

Updated Position

Table 4.6: Updated Position

	X_1	X_2	X_3	X_4	$Max\ Z = X_1 + X_2 + 2X_3 + X_4$
1	0.183	4	5	1	15.183
2	0.649	3.819	5	1	15.46
3	0.12	3.617	5	1	14.73

$$Gbest \;=\; [0.649\;\; 3.819\;\; 5\;\; 1]$$

Pbest =

Table 4.7: Pbest

	X_1	X_2	X_3	X_4
1	0.601	3.844	5	0.978
2	0.649	3.819	5	1
3	0.729	3.425	5	0.871

Iteration 3: Similarly, we update new velocity and position

then, $\omega \;=\; 0.9 - 0.3 \;=\; 0.6$

$$\begin{aligned}
V_{11} &= 0.6 \times (-0.4188) + 2 \times rand()(0.601 - 0.183) + 2 \times rand()(0.649 - 0.183) \\
&= -0.2316 \\
V_{12} &= 0.6 \times 0.1708 + 2 \times rand()(3.844 - 4) + 2 \times rand()(3.819 - 4) \\
&= -0.2905 \\
V_{13} &= 0.6 \times 0.821 + 2 \times rand()(5 - 5) + 2 \times rand()(5 - 5) \\
&= 0.492 \\
V_{14} &= 0.6 \times 0.0411 + 2 \times rand()(0.9708 - 1) + 2 \times rand()(1 - 1) \\
&= 0.02407 \\
V_{21} &= 0.6 \times (-0.1687) = -0.1012 \\
V_{22} &= 0.6 \times 0.3312 = 0.1986 \\
V_{23} &= 0.6 \times 0.3716 = 0.2226 \\
V_{24} &= 0.6 \times 0.244 = 0.1464
\end{aligned}$$

$$\begin{aligned}
V_{31} &= 0.6 \times (-0.6096) + 2 \times rand()(0.729 - 0.12) + 2 \times rand()(0.649 - 0.12) \\
&= -0.2333 \\
V_{32} &= 0.6 \times 0.1923 + 2 \times rand()(3.425 - 3.617) + 2 \times rand()(3.819 - 3.617) \\
&= 0.3204 \\
V_{33} &= 0.6 \times 1.249 + 2 \times rand()(5 - 5) + 2 \times rand()(5 - 5) \\
&= 0.749 \\
V_{34} &= 0.6 \times 0.1474 + 2 \times rand()(0.871 - 1) + 2 \times rand()(1 - 1) \\
&= 0.0262
\end{aligned}$$

Table 4.8: Updated velocity

	V_1	V_2	V_3	V_4
1	-0.2316	-0.2905	0.492	0.0247
2	-0.1012	0.1986	0.2226	0.1464
3	-0.2333	0.3204	0.749	0.0262

Updated Position

Table 4.9: Updated Position

	X_1	X_2	X_3	X_4	$Max\ Z = X_1 + X_2 + 2X_3 + X_4$
1	0	3.709	5	1	14.709
2	0.548	4	5	1	15.54
3	0.496	3.745	5	1	15.24

$$Gbest = [0.548 \;\; 4 \;\; 5 \;\; 1]$$

Pbest =

Table 4.10: Pbest

	X_1	X_2	X_3	X_4
1	0.183	4	5	1
2	0.548	4	5	1
3	0.496	3.745	5	1

Iteration 4: We update new velocity and position

then, $\omega \ = \ 0.9 - 0.4 \ = \ 0.5$

$$
\begin{aligned}
V_{11} &= 0.5 \times (-0.2316) + 2 \times rand()(0.183 - 0) + 2 \times rand()(0.548 - 0) \\
&= -0.11082 \\
V_{12} &= 0.5 \times (-0.2905) + 2 \times rand()(4 - 3.709) + 2 \times rand()(4 - 3.709) \\
&= 0.2675 \\
V_{13} &= 0.5 \times 0.492 + 2 \times rand()(5 - 5) + 2 \times rand()(5 - 5) \\
&= 0.492 \\
V_{14} &= 0.5 \times 0.02407 + 2 \times rand()(1 - 1) + 2 \times rand()(1 - 1) \\
&= 0.012 \\
V_{21} &= 0.5 \times (-0.1012) = -0.0506 \\
V_{22} &= 0.5 \times 0.1986 = 0.0993 \\
V_{23} &= 0.5 \times 0.2226 = 0.1112 \\
V_{24} &= 0.5 \times 0.1464 = 0.0732 \\
V_{31} &= 0.5 \times (-0.2333) + 2 \times rand()(0.496 - 0.496) + 2 \times rand()(0.548 - 0.496) \\
&= -0.0607 \\
V_{32} &= 0.5 \times 0.3204 + 2 \times rand()(3.745 - 3.745) + 2 \times rand()(4 - 3.745) \\
&= 0.3375 \\
V_{33} &= 0.5 \times 0.749 + 2 \times rand()(5 - 5) + 2 \times rand()(5 - 5) \\
&= 0.3745 \\
V_{34} &= 0.5 \times 0.0262 + 2 \times rand()(1 - 1) + 2 \times rand()(1 - 1) \\
&= 0.0131
\end{aligned}
$$

Table 4.11: Updated velocity

	V_1	V_2	V_3	V_4
1	-0.1108	0.295	0.246	0.012
2	-0.0506	0.0993	0.1113	0.0732
3	-0.0607	0.3375	0.3745	0.0131

Updated Position

Table 4.12: Updated Position

	X_1	X_2	X_3	X_4	$Max\ Z = X_1 + X_2 + 2X_3 + X_4$
1	0	4	5	1	15
2	0.497	4	5	1	15.49
3	0.435	4	5	1	15.43

$$Gbest \;=\; [0.548 \;\; 4 \;\; 5 \;\; 1]$$

Pbest =

Table 4.13: Pbest

	X_1	X_2	X_3	X_4
1	0	4	5	1
2	0.548	4	5	1
3	0.435	4	5	1

Iteration 5: We update new velocity and position

then, $\omega \;=\; 0.9 - 0.5 \;=\; 0.4$

$$\begin{aligned}
V_{11} &= 0.4 \times (-0.11082) + 2 \times rand()(0-0) + 2 \times rand()(0.548 - 0) \\
&= -0.0363 \\
V_{12} &= 0.4 \times 0.295 + 2 \times rand()(4-4) + 2 \times rand()(4-4) \\
&= 0.118 \\
V_{13} &= 0.4 \times 0.246 + 2 \times rand()(5-5) + 2 \times rand()(5-5) \\
&= 0.0984 \\
V_{14} &= 0.4 \times 0.012 = 0.0048 \\
V_{21} &= 0.4 \times (-0.0506) + 2 \times rand()(0.548 - 0.497) + 2 \times (0.548 - 0.497) \\
&= -0.0179 \\
V_{22} &= 0.4 \times (0.0993) + 2 \times rand()(4-4) + 2 \times (4-4) \\
&= 0.0397 \\
V_{23} &= 0.4 \times (0.1113) + 2 \times rand()(5-5) + 2 \times (5-5) \\
&= 0.0445 \\
V_{24} &= 0.4 \times 0.0732 = 0.029
\end{aligned}$$

$$
\begin{aligned}
V_{31} &= 0.4 \times (-0.0607) + 2 \times rand()(0.435 - 0.435) + 2 \times rand()(0.548 - 0.435) \\
&= -0.0213 \\
V_{32} &= 0.4 \times 0.3745 = 0.1498 \\
V_{33} &= 0.4 \times 0.3375 = 0.135 \\
V_{34} &= 0.4 \times 0.0131 = 0.0052
\end{aligned}
$$

Table 4.14: Updated velocity

	V_1	V_2	V_3	V_4
1	-0.0363	0.118	0.0984	0.0048
2	-0.0179	0.0397	0.0445	0.029
3	-0.0213	0.1498	0.135	0.0052

Updated Position

Table 4.15: Updated Position

	X_1	X_2	X_3	X_4	$Max\ Z = X_1 + X_2 + 2X_3 + X_4$
1	0	4	5	1	15
2	0.530	4	5	1	15.53
3	0.413	4	5	1	15.41

$$Gbest = [0.53 \;\; 4 \;\; 5 \;\; 1]$$

then,

$$|f(x)_{prev} - f(x)| = 0$$

$$|15.54 - 15.53| = 0$$

$$0.01 \geq 0$$

then by particle swarm optimization

$X_1 = 0.53$, $X_2 = 4$, $X_3 = 5$, $X_4 = 1$, $Z = 15.53$

4.1.1 MATLAB Result:

```
Final Results----------------------------

bestfun =

   -15


bestrun =

     6


best_variables =

     0     4     5     1

*******************************************************
```

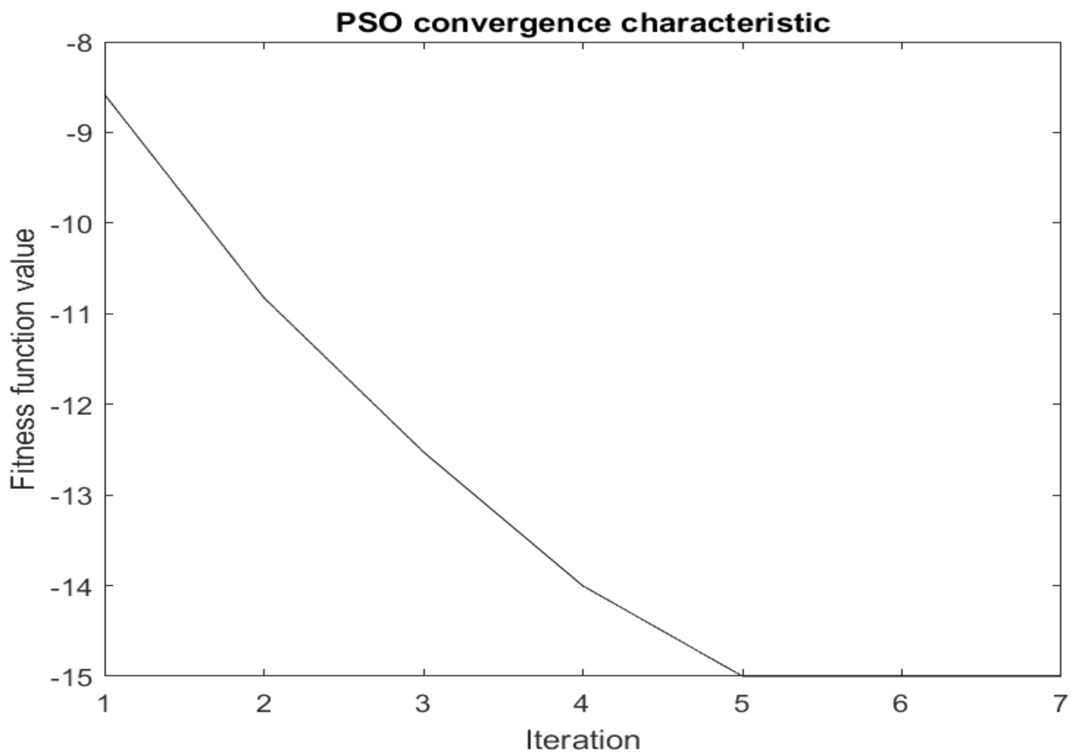


Figure 4.1: PSO Convergence Characterstics [11]

Chapter 5

Breast Cancer Diagnostic Data Set

Table 5.1: Data Set

	id	radius_mean	texture_mean	perimeter_mean	area_mean	smthnes_mean
0	842302	17.99	10.38	122.80	1001.0	0.11840
1	842517	20.57	17.77	132.90	1326.0	0.08474
2	84300903	19.69	21.25	130.00	1203.0	0.10960
3	84348301	11.42	20.38	77.58	386.1	0.14250
4	84358402	20.29	14.34	135.10	1297.0	0.10030
...	...	...	...	...	...	...
564	926424	21.56	22.39	142.00	1479.0	0.11100
565	926682	20.13	28.25	131.20	1261.0	0.09780
566	926954	16.60	28.08	108.30	858.1	0.08455
567	927241	20.60	29.33	140.10	1265.0	0.11780
568	92751	7.76	24.54	47.92	181.0	0.05263

	id	compactness_mean	concavity_mean	concave pt mn	symtry_mean	...
0	842302	0.27760	0.30010	0.14710	0.2419	...
1	842517	0.07864	0.08690	0.07017	0.1812	...
2	84300903	0.15990	0.19740	0.12790	0.2069	...
3	84348301	0.28390	0.24140	0.10520	0.2597	...
4	84358402	0.13280	0.19800	0.10430	0.1809	...
...	...	...	...	...	...	...
564	926424	0.11590	0.24390	0.13890	0.1726	...
565	926682	0.10340	0.14400	0.09791	0.1752	...
566	926954	0.10230	0.09251	0.05302	0.1590	...
567	927241	0.27700	0.35140	0.15200	0.2397	...
568	92751	0.04362	0.00000	0.00000	0.1587	...

	rds_worst	text_worst	prmtr_worst	area_worst	...	concty_worst	..	frctl_dim_wrt
0	25.380	17.33	184.60	2019.0	...	0.7119	..	0.11890
1	24.990	23.41	158.80	1956.0	...	0.2416	..	0.08902
2	23.570	25.53	152.50	1709.0	...	0.4504	..	0.08758
3	14.910	26.50	98.87	567.7	...	0.6869	..	0.17300
4	22.540	16.67	152.20	1575.0	...	0.4000	..	0.07678
...	...	...	...	...	...	...	..	...
564	25.450	26.40	166.10	2027.0	...	0.4107	..	0.07115
565	23.690	38.25	155.00	1731.0	...	0.3215	..	0.06637
566	18.980	34.12	126.70	1124.0	...	0.3403	..	0.07820
567	25.740	39.42	184.60	1821.0	...	0.9387	..	0.12400
568	9.456	30.37	59.16	268.6	...	0.0000	..	0.07039

The Wisconsin Diagnostic Breast Cancer (WDBC) dataset is a collection of medical data on breast cancer patients. It includes measurements from fine needle aspiration (FNA) tests of breast masses, such as the radius, texture, perimeter, area, smoothness, compactness, concavity, symmetry, and fractal dimension. These measurements are used to determine whether a tumor is benign (non-cancerous) or malignant (cancerous).

The dataset also includes information on the patient's age and menopausal status. This dataset has been widely used in research to develop machine learning models to predict whether a tumor is benign or malignant based on these measurements [12].

The attributes of the digital picture of a fine needle aspirate (FNA) of a breast mass include the

- ID Number
- Diagnosis, which is either Malignant (M) or Benign (B)

The picture also has ten real-valued features which describe the characteristics of the visible cell nuclei. These features include the

- radius
- texture
- perimeter
- area
- smoothness
- compactness
- concavity

- concave points
- symmetry
- fractal dimension
- Length of Data set = 569

These features can help in the diagnosis of the breast mass by providing information about the characteristics of the cell nuclei in the picture.

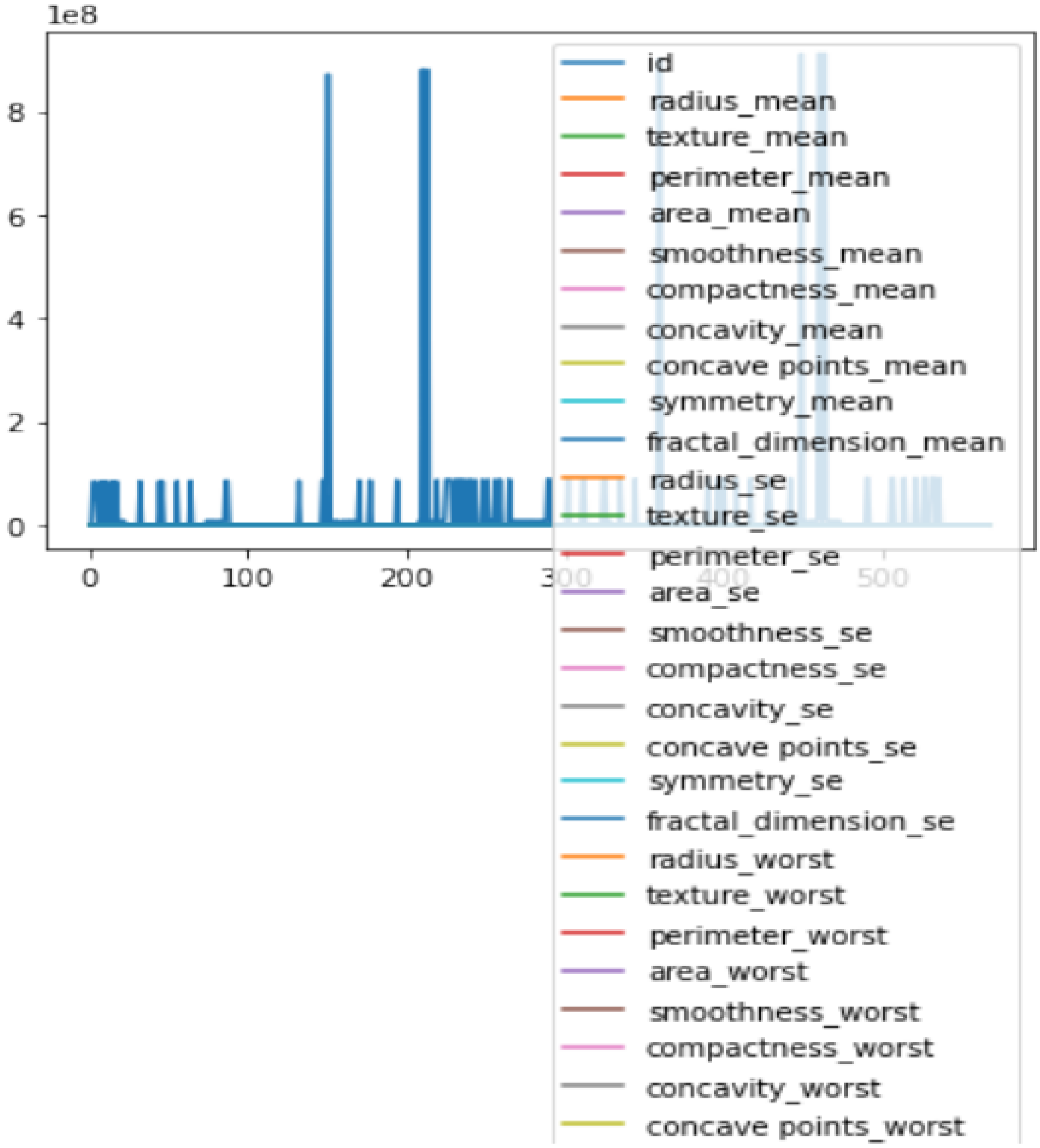


Figure 5.1: Plot of breast cancer diagnostic data set

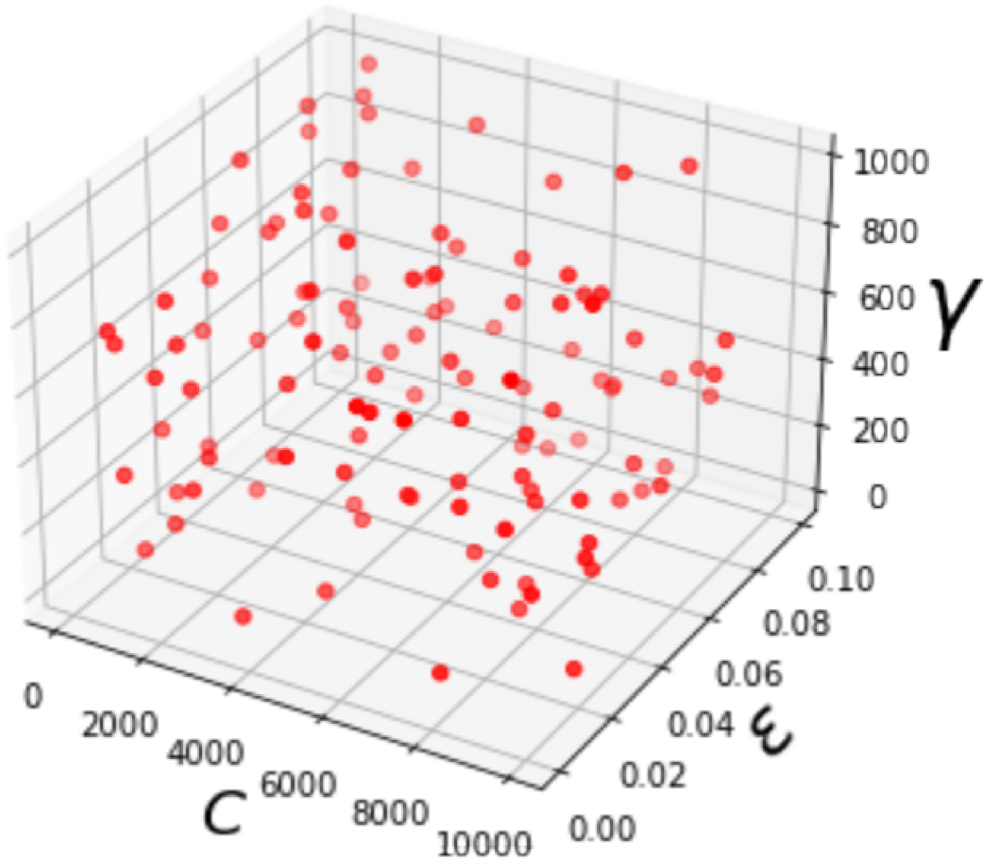


Figure 5.2: DataSet Plot with Particles

5.1 Result:

- Group best configuration found: [33.44793 0.0355742 0.001]
- Regressor: $C = 33.44793$ $\epsilon = 0.0355742$ $\gamma = 0.001$

Table 5.2: Function Value with inertia 1

Iteration	Function Value	Iteration	Function Value	Iter	Funct Value
0	0.026010	33	0.025310	66	0.025310
1	0.026010	34	0.025310	67	0.025310
2	0.026010	35	0.025310	68	0.025310
3	0.025740	36	0.025310	69	0.025310
4	0.025702	37	0.025310	70	0.025310
5	0.025702	38	0.025310	71	0.025310
6	0.025702	39	0.025310	72	0.025310
7	0.025582	40	0.025310	73	0.025310
8	0.025582	41	0.025310	74	0.025310
9	0.025582	42	0.025310	75	0.025310
10	0.025582	43	0.025310	76	0.025310
11	0.025389	44	0.025310	77	0.025310
12	0.025373	45	0.025310	78	0.025310
13	0.025357	46	0.025310	79	0.025310
14	0.025333	47	0.025310	80	0.025310
15	0.025333	48	0.025310	81	0.025310
16	0.025333	49	0.025310	82	0.025310
17	0.025333	50	0.025310	83	0.025310
18	0.025333	51	0.025310	84	0.025310
19	0.025333	52	0.025310	85	0.025308
20	0.025327	53	0.025310	86	0.025308
21	0.025327	54	0.025310	87	0.025308
22	0.025316	55	0.025310	88	0.025308
23	0.025316	56	0.025310	90	0.025308
24	0.025316	57	0.025310	91	0.025308
25	0.025316	58	0.025310	92	0.025308
26	0.025316	59	0.025310	93	0.025308
27	0.025316	60	0.025310	94	0.025308
28	0.025316	61	0.025310	95	0.025308
29	0.025310	62	0.025310	96	0.025308
30	0.025310	63	0.025310	97	0.025308
31	0.025310	64	0.025310	98	0.025308
32	0.025310	65	0.025310	99	0.025308

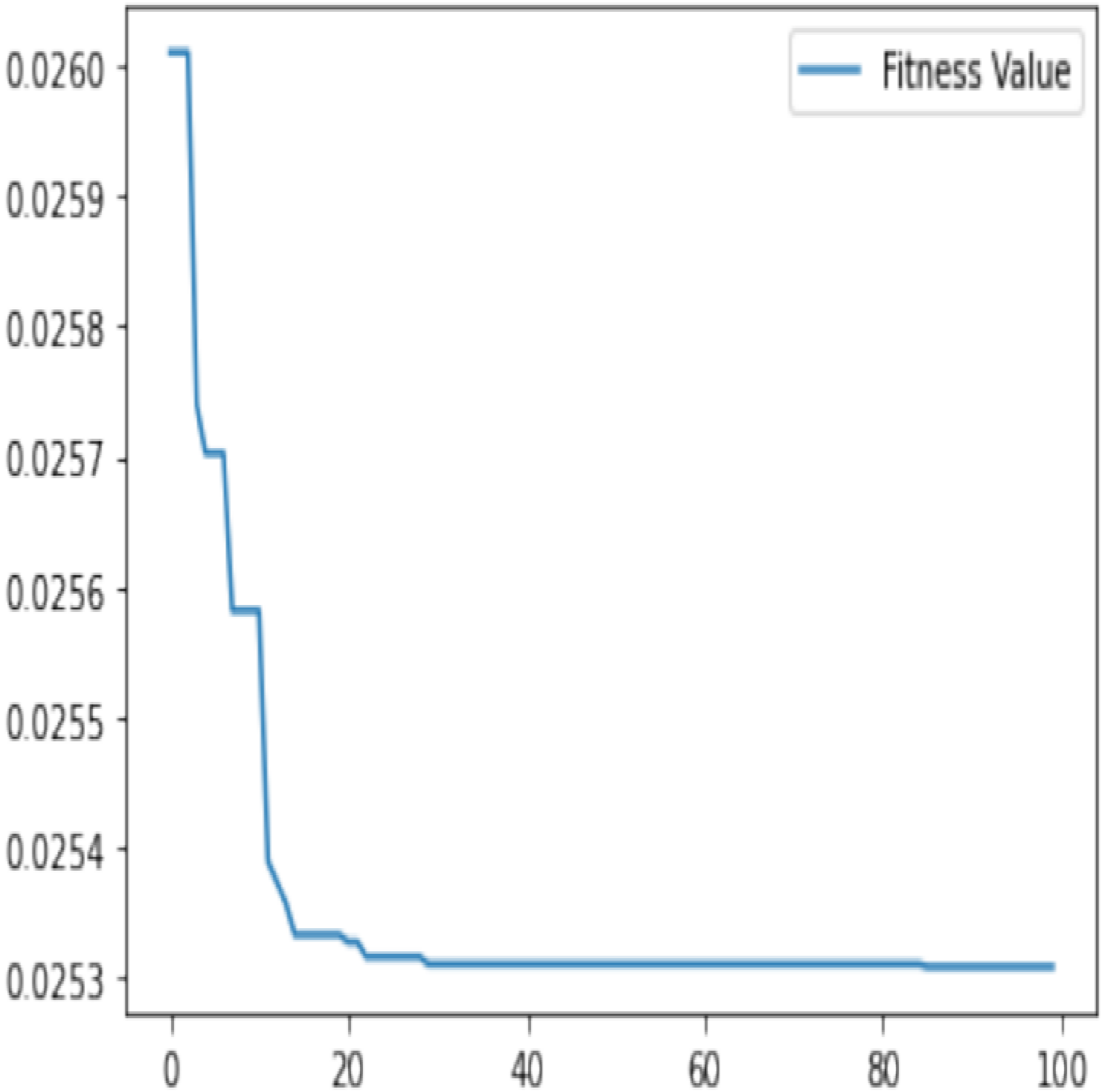


Figure 5.3: Best Regressor fitness value

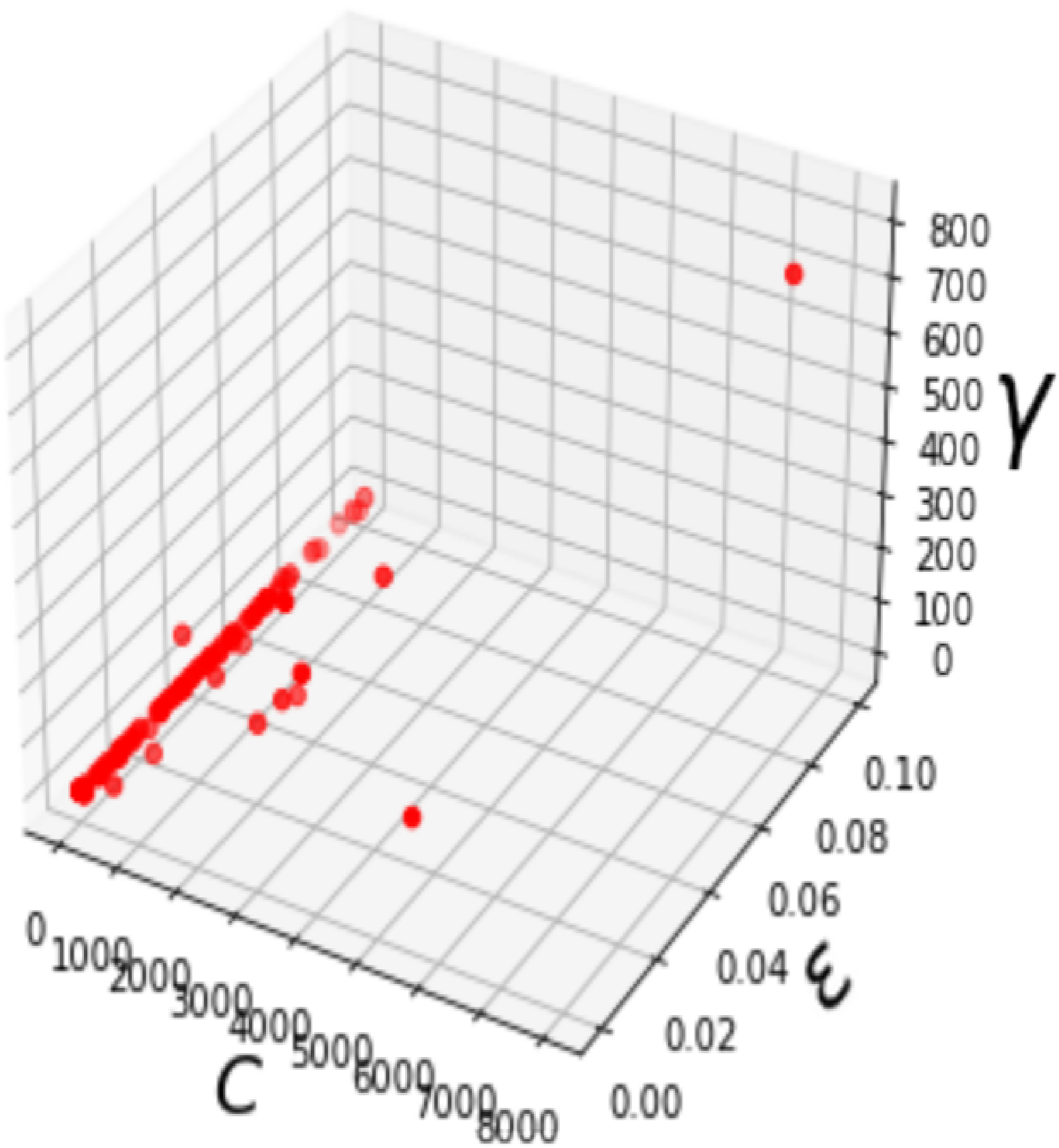


Figure 5.4: Prediction

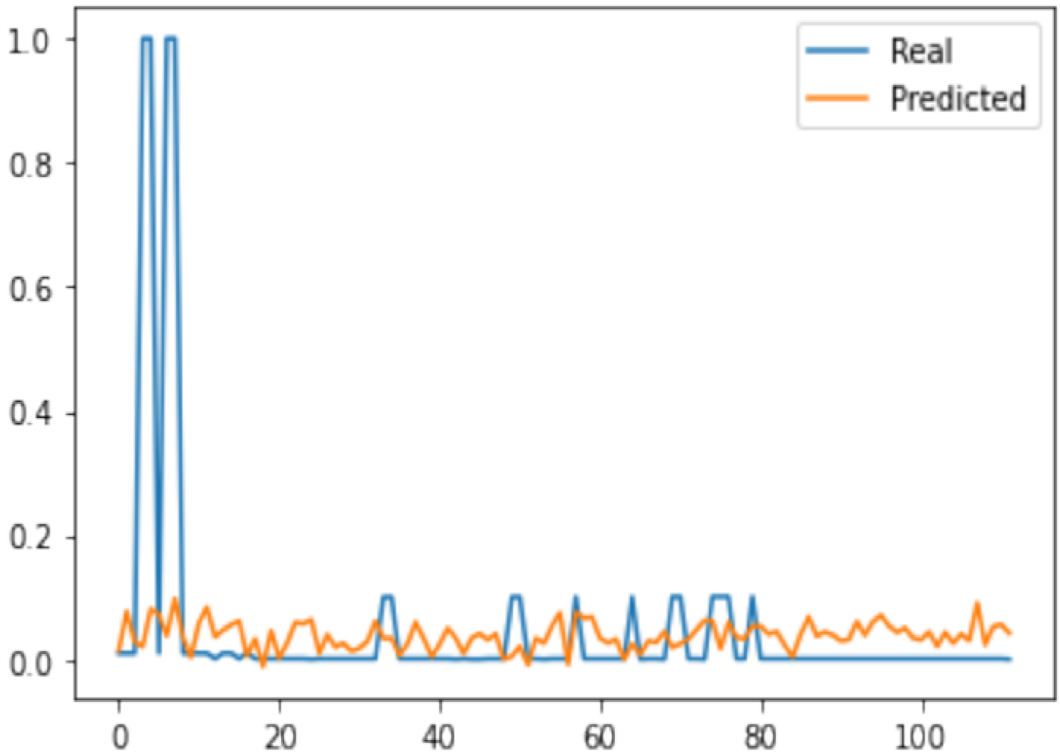


Figure 5.5: Prediction with population best value found

- Mean Squared error for the test set: 0.033581
- Predictions Average: 0.037820
- Predictions Median: 0.034948

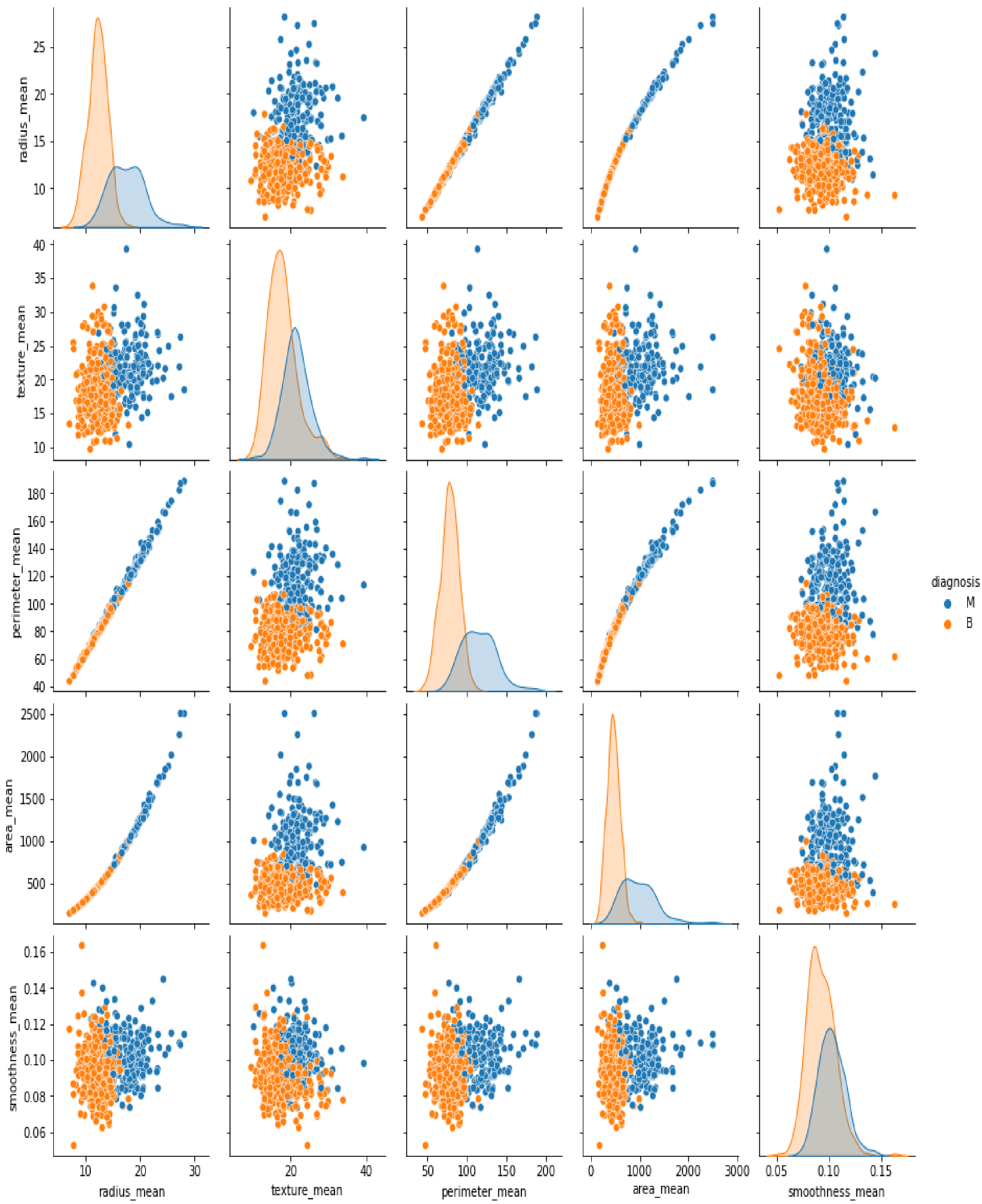


Figure 5.6: Data Set Plot

Table 5.3: Function Value with inertia 0.4

Iteration	Function Value	Iteration	Function Value	Iter	Funct Value
0	0.026010	33	0.025982	66	0.025982
1	0.026010	34	0.025982	67	0.025982
2	0.026010	35	0.025982	68	0.025982
3	0.026004	36	0.025982	69	0.025982
4	0.026004	37	0.025982	70	0.025982
5	0.026004	38	0.025982	71	0.025982
6	0.026004	39	0.025982	72	0.025982
7	0.026004	40	0.025982	73	0.025982
8	0.026004	41	0.025982	74	0.025982
9	0.026004	42	0.025982	75	0.025982
10	0.026003	43	0.025982	76	0.025982
11	0.026003	44	0.025982	77	0.025982
12	0.026003	45	0.025982	78	0.025982
13	0.026003	46	0.025982	79	0.025982
14	0.026003	47	0.025982	80	0.025982
15	0.025984	48	0.025982	81	0.025982
16	0.025984	49	0.025982	82	0.025982
17	0.025984	50	0.025982	83	0.025982
18	0.025983	51	0.025982	84	0.025982
19	0.025983	52	0.025982	85	0.025982
20	0.025982	53	0.025982	86	0.025982
21	0.025982	54	0.025982	87	0.025982
22	0.025982	55	0.025982	88	0.025982
23	0.025982	56	0.025982	90	0.025982
24	0.025982	57	0.025982	91	0.025982
25	0.025982	58	0.025982	92	0.025982
26	0.025982	59	0.025982	93	0.025982
27	0.025982	60	0.025982	94	0.025982
28	0.025982	61	0.025982	95	0.025982
29	0.025982	62	0.025982	96	0.025982
30	0.025982	63	0.025982	97	0.025982
31	0.025982	64	0.025982	98	0.025982
32	0.025982	65	0.025982	99	0.025982

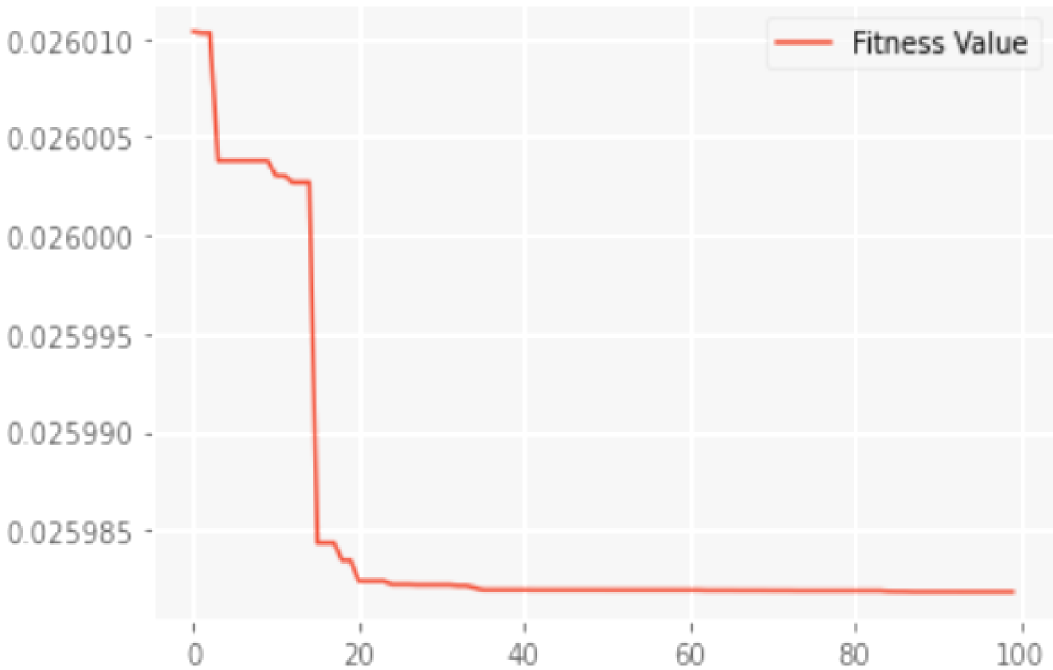


Figure 5.7: Data Set Plot

- Group best configuration found: [7656.83658 0.0157675 0.43472]
- Regressor: $C = 7656.836$ $\epsilon = 0.0157675$ $\gamma = 0.43472$

Chapter 6

Applications of PSO

Different application fields exist for the Particle Swarm Optimization technique.

The first application of particle swarm optimisation in the actual world was made by Kennedy and Eberhart in 1995. It concerned with neural network training and was disclosed along with the method. PSO has been successfully used in a wide range of applications, including those in telecommunications, system management, data mining, power systems, design, combinatorial optimization, signal processing, network training, and many more. PSO algorithms have since been developed to solve constrained problems, multi-objective optimisation problems, problems with dynamically changing landscapes, and problems that call for multiple solutions, whereas the original PSO algorithm was primarily used to solve unconstrained, single-objective optimisation problems. Here are a few PSO applications [13]:

- **Engineering design optimization:** PSO can be apply to develop the design parameters of complex systems, such as mechanical, electrical, or chemical systems. The aim is to determine the optimal values of the design parameters that increase or decrease a certain objective function, such as cost, performance, or reliability.
- **Image and signal processing:** PSO can be apply to develop the parameters of image and signal processing algorithms, such as picture segmentation, edge detection, and denoising. The aim is to determine the optimal values of the algorithm parameters that produce the best quality of the processed image or signal.
- **Machine learning:** PSO can be apply to develop the parameters of machine learning algorithms, such as neural networks, support vector machines, and decision trees. The aim is to determine the optimal values of the algorithm parameters that maximize the accuracy or minimize the error of the trained model.
- **Robotics:** PSO can be apply to develop the control parameters of robotic systems, such as trajectory planning, motion control, and obstacle avoidance. The aim is to determine the optimal values of the control parameters that minimize the energy consumption, reduce the collision risk, or increase the performance of the robot.

- **Financial modeling:** PSO can be used to optimize the portfolio allocation in financial modeling, such as stock trading and risk management. The aim is to determine the optimal allocation of the investment among different assets that maximizes the return or minimizes the risk.
- **Function optimization:** PSO can be apply to develop the optimal solution to mathematical functions with multiple variables. The aim is to determine the input variables that produce the lowest or highest value of the function.

6.1 Advantages and Disadvantages of PSO

Despite having some drawbacks, it is claimed that the PSO algorithm is one of the most effective ways to solve non-smooth global optimization issues. The following is a discussion of PSO's benefits and drawbacks:

6.1.1 Benefits of PSO Method:

1. Particle Swarm Optimization (PSO) is a derivative free method.
2. It is simple to execute, making it appropriate for use in both engineering and scientific study.
3. Compared to other optimization methods, it has fewer factors and their influence on the solutions is minimum.
4. The PSO algorithm's computation is very smooth.
5. There are some methods that guarantee convergence and make it simple and quick to compute the problem's optimal value.
6. Compared to other optimisation techniques, PSO is less depending on a set of initial points.
7. PSO can reach the ideal conclusion fast, especially for high-dimensional problems, as it is based on the collective behavior of a swarm of particles. This makes it a suitable choice for real-time applications where quick solutions are required.
8. PSO can be used to solve a wide range of optimization problems, including continuous, discrete, and multimodal problems.

6.1.2 Drawbacks of PSO Method:

While Particle Swarm Optimization (PSO) has several advantages, it also has some limitations and disadvantages that should be considered when using it for optimization problems. Here are some of the main drawbacks of PSO:

1. The partial optimism of the PSO method impairs the control of its speed and trajectory.
2. PSO may converge to a local optimum, especially if the swarm size is small or the search space is complex. This can result in suboptimal solutions and prevent the algorithm from finding the global optimum.
3. PSO has several parameters that need to be set, i.e, the weight of inertia , acceleration coefficients, and maximum velocity. The selection of these parameters can affect PSO effectiveness, and selecting the best values can be difficult.
4. Due to the fact that the computational complexity rises with the number of particles and dimensions, PSO can become computationally costly for large-scale issues. Due to this, it may not be feasible for issues involving numerous factors or constraints.

Chapter 7

Conclusion

The fundamental Particle Swarm Optimization algorithm, the geometrical and mathematical justification of PSO, the movement and velocity update of the particles in the search space, the acceleration factors, and the neighbourhood topologies of the particles were all covered in Chapter 3 of this thesis.

In chapter 3 we are also discussed linear programming problem using partcle swarm optimization,i.e, inertia weight, velocity, position and convergence characterstics of LPP. In chapter 4 we discussed about Transportation problem using PSO and determine its convergence characterstics using MATLAB(Matrix Laboratory).

In chapter 5 we consider a datset of breast cancer diagnosis and applying particle swarm optimization by SVM(Support Vector Machine) to get a convergence rate. In last we have some application of particle swarm optimization.

Bibliography

[1] Kennedy, J., Eberhart, R. (1995). Particle swarm optimization. Proceedings of ICNN'95 - International Conference on Neural Networks. https://doi.org/10.1109/icnn.1995.488968

[2] Talukder, S. (2011). Mathematical Modelling and Applications of Particle Swarm Optimization https://www.diva-portal.org/smash/get/diva2:829959/FULLTEXT01.pdf

[3] M. Dorigo and M. Birattar, 2007 "Swarm intelligence," in Scholarpedia

[4] Rao, S. S. (1996). Engineering optimization : theory and practice. Wiley.

[5] Fletcher, R. (2013). Practical Methods of Optimization. In Google Books. John Wiley Sons.

[6] Marini, F., Walczak, B. (2015). Particle swarm optimization (PSO). A tutorial. Chemometrics and Intelligent Laboratory Systems, https://doi.org/10.1016/j.chemolab.2015.08.020

[7] Albin, R. (n.d.). Particle Swarm Optimization: Algorithm and its Codes in MATLAB. Www.academia.edu. https://www.academia.edu/25008607/Particle Swarm Optimization Algorithm and its Codes in MATLAB

[8] E. Ozcan and C.K. Mohan C.H. Dagli, M. Akay, A.L. Buczak, O. Ersoy, B.R. Bernandez (Eds.) 1998, Intelligent Engineering Systems Through Artificial Neural Networks, Proceedings of the Artificial Neural Networks in Engineering Conference (ANNIE 98), vol. 8, American Society of Mechanical Engineering (ASME).

[9] Eberhart, R. C., Shi, Y., Kennedy, J. (2001). Swarm Intelligence. In Google Books. Elsevier.

[10] Bansal, J. C., Singh, P. K., Saraswat, M., Verma, A., Jadon, S. S., Abraham, A. (2011). Inertia Weight strategies in Particle Swarm Optimization. 2011 Third World Congress on Nature and Biologically Inspired Computing. https://doi.org/10.1109/nabic.2011.6089659

[11] Cuevas, E. and Rodríguez, A. (2020) “Particle Swarm Optimization (PSO) algorithm,” Metaheuristic Computation with MATLAB®, pp. 159–181. Available at: https://doi.org/10.1201/9781003006312-6.

[12] Learning, U.C.I.M. (2016) Breast cancer wisconsin (diagnostic) data set, Kaggle. Available at: https://www.kaggle.com/datasets/uciml/breast-cancer-wisconsin-data

[13] Riccardo Poli,(2008) ”Review Article-Analysis of the Publications on the Applications of Particle Swarm Optimisation,” Journal of Artificial Evolution and Applications.