DELHI TECHNOLOGICAL UNIVERSITY

(Formerly Delhi College of Engineering) Shahbad Daulatpur, Bawana Road, Delhi 110042

NUMERICAL AND ENGINEERING OPTIMISATION METHOD, (MC-201)

PROJECT REPORT

ON

PARTICLE SWAMP OPTIMISATION METHOD

SUBMITTED BY:-

MOHAMMAD YASA, 2K20/EE/164

MAYANK, 2K20/EE/160

SUBMITTED TO:-

DR. RADHESHAYM SAHA

CANDIDATE'S DECLARATION

We, (Mohammad Yasa 2K20/EE/164 and Mayank (2K20/EE/160) students of B.Tech hereby declare that the project Dissertation titled "PARTICLE SWAMP OPTIMISATION METHOD" which is submitted by us to Professor Radheshyam Saha, Delhi Technological University, Delhi in partial fulfilment of the requirement for the Mid-term Examination Component of Bachelor of Technology, is original and not copied from any source without proper citation.

Place: Delhi
Date: 2 November 2021

Mohammad Yasa, 2K20/EE/164
Mayank, 2K20/EE/160

CERTIFICATE

I hereby certify that the project dissertation titled "PARTICLE SWAMP OPTIMISATION METHOD" which is submitted by Mohammad Yasa (2K20/EE/164) and Mayank, (2K20/EE/160) Delhi Technological University, Delhi in complete fulfilment of the requirement for the Mid-term Examination Component of the Bachelor of Technology, is a record of the project work carried out by the students under my supervision.

Place: Delhi	Dr. Radheshyam Saha
Date: 2 November 2021	Professor

ACKNOWLEDGEMENT

In performing our innovative project, we had to take the help and guideline of some respected persons, who deserve our greatest gratitude. The completion of this project gives us much pleasure. We would like to show our gratitude to Professor Radheshyam Saha, Mentor for project. Giving us a good guideline for report throughout numerous consultations. We would also like to extend our deepest gratitude to all those who have directly and indirectly guided us in writing this assignment.

Many people, our classmates and team members itself, have made valuable comment suggestions on this proposal which gave us an inspiration to improve our project. We thank all the people for their help directly and indirectly to complete our project.

CONTENTS

INTRODUCTION

AN OVERVIEW

In the early of 1990s, several studies regarding the social behaviour of animal groups were developed. These studies showed that some animals belonging to a certain group, that is, birds and fishes, are able to share information among their group, and such capability confers these animals a great survival advantage. Inspired by these works, Kennedy and Eberhart proposed in 1995 the PSO algorithm, a metaheuristic algorithm that is appropriate to optimize nonlinear continuous functions. The author derived the algorithm inspired by the concept of swarm intelligence, often seen in animal groups, such as flocks and shoals.

Actually, the mechanical engineering optimization can be expressed as continuity interval constraint optimization. To investigate this problem, some traditional gradient methodologies were investigated such as the penalty function and Lagrange multiplier. Although the theory of these methods is impeccable, the objective function and the constraint condition must be differentiable. However, the constraint function and objective function are nondifferentiable and discontinuous implicit functions in practical engineering.

Consequently, a new optimization method was developed to study this problem, which plays an important role for the development of engineering optimization design.

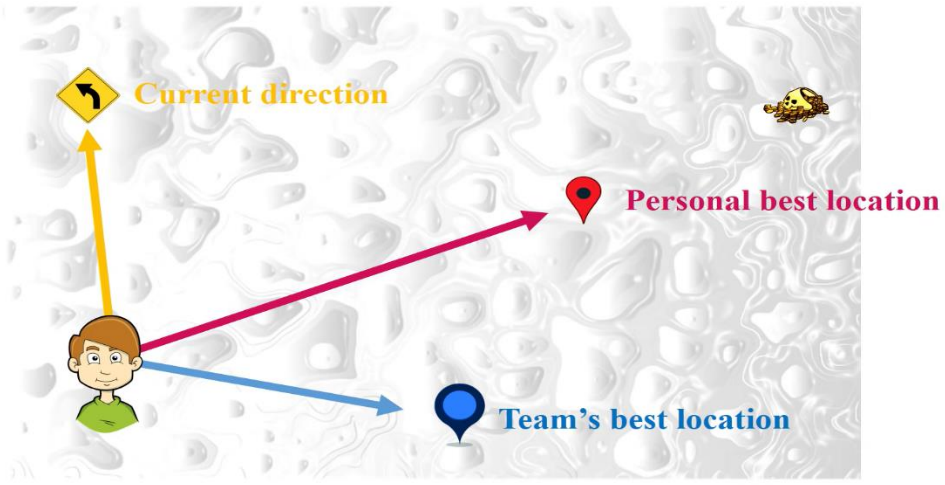
the PSO algorithm indicate that some constrained points are not located in the feasible region but outside the feasible region; however, these excluded constrained points are closer to the boundary than the points located in the feasible region.

HOW DOES IT WORK ?

PSO search strategy
Bob
Anthony
Jennifer

Before understanding the mathematics of this algorithm, let’s analyse an analogy, the given above figure depicts a treasure hidden in a mountainous region and a team of 3 people are set to search, and the team decides to use the PSO technique to look for the treasure, for this they have to follow the PSO rules.

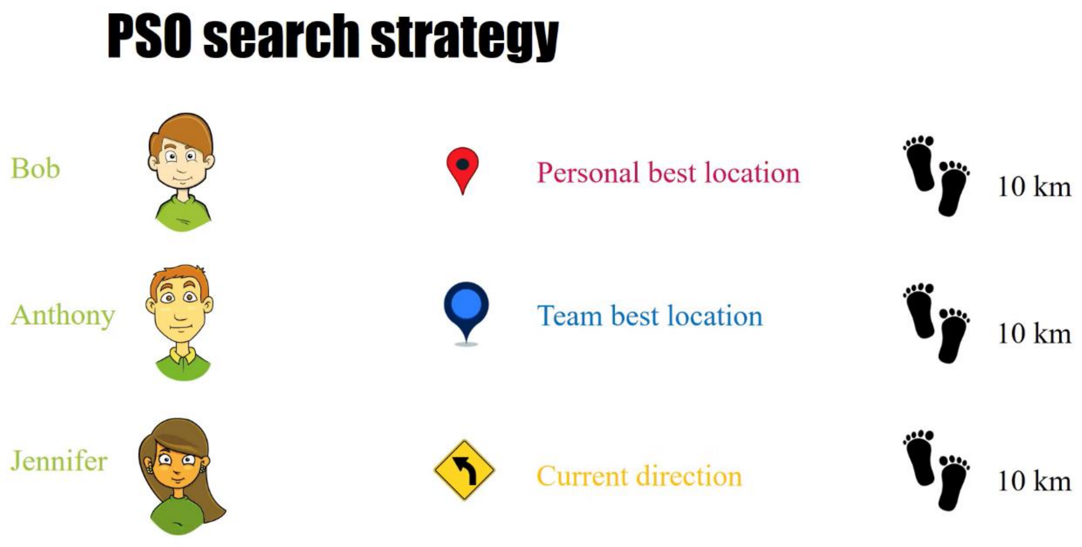
PSO search strategy



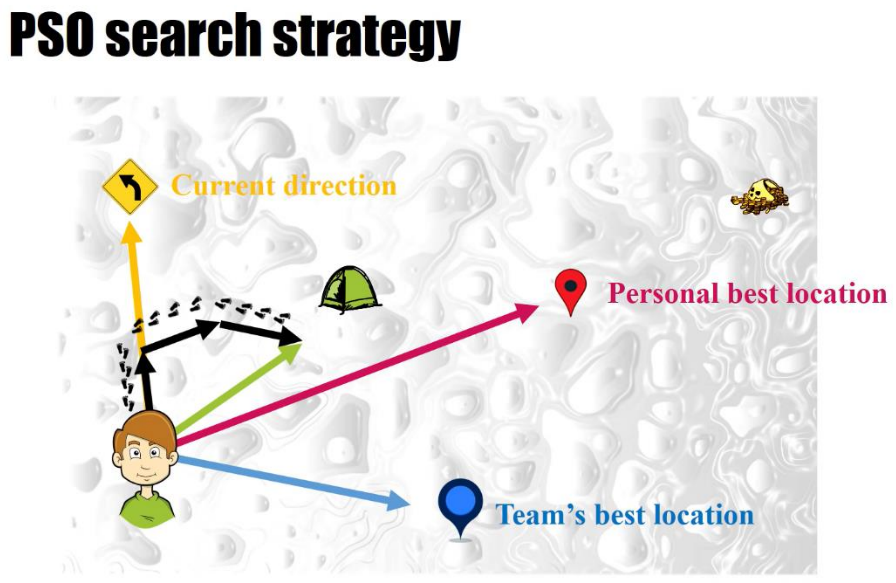
1:- Each team member needs to evaluate the journey visited so far, which is the location with the minimum elevation.

2:- Before any movement they need to communicate to be able to find out the updated position that the entire team has found so far.

3:- Each team member has to know his/her current travel direction, as they will need it to find the next possible location of further journey.

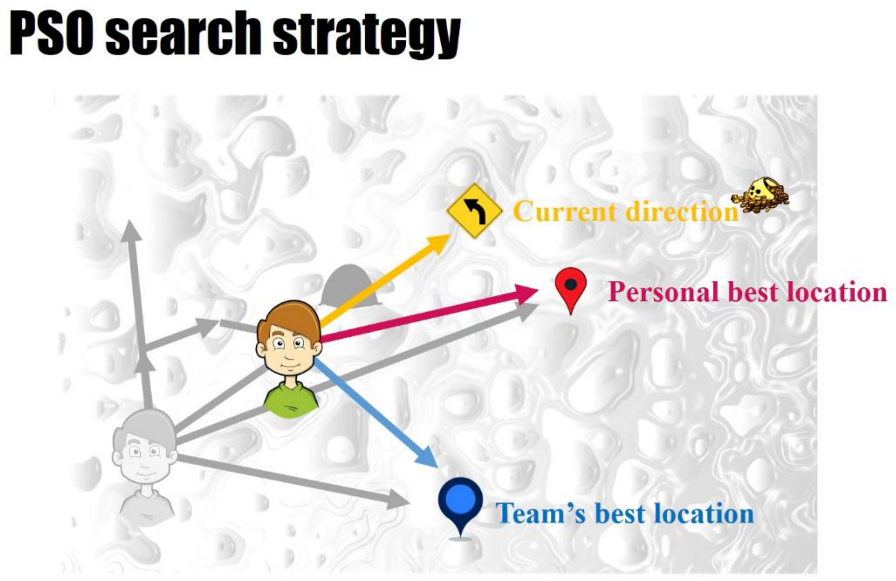


Every day we may assume for instance that each team member walks 10 km in the current location, the same distance is covered by their individual best direction and the same distance in the direction decided mutually by all team members.



For instance, Bob after completing his in total 30Km journey with his personal best direction, team's best direction and the current direction, as so travelled by each member Bob finally calls of the day and sets up a camp in the region as shown in the above figure. Now, this position where the camp has been setup could be a better or a worse position than the best value found by the entire team.

If this position is better than the personal best position, Bob needs to update his record before he starts the journey the next day, and if it is better than the team's best position he again needs to update his record and inform the team members, but if this new position is not better then Bob does not have to take any action.



Assuming that the position was not better, henceforth Bob has to travel again in the 3 locations' directions i.e. 10 km each again, with slight changes in the directions as observed in the above picture.

PSO search strategy

If in the similar manner Bob travels and sets up camps for 5 consecutive days, his positions at the end of the day would look something like this.

We come up with an equation to comprehend these arbitrary random locations.

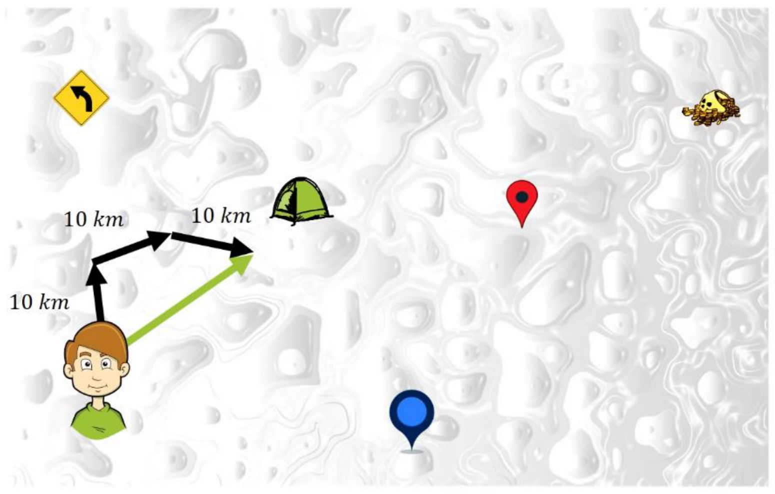
So, varying the random component of each team member -

2 X **r** X **10** ***km***

Where are is a random position in the interval say, [0,1].

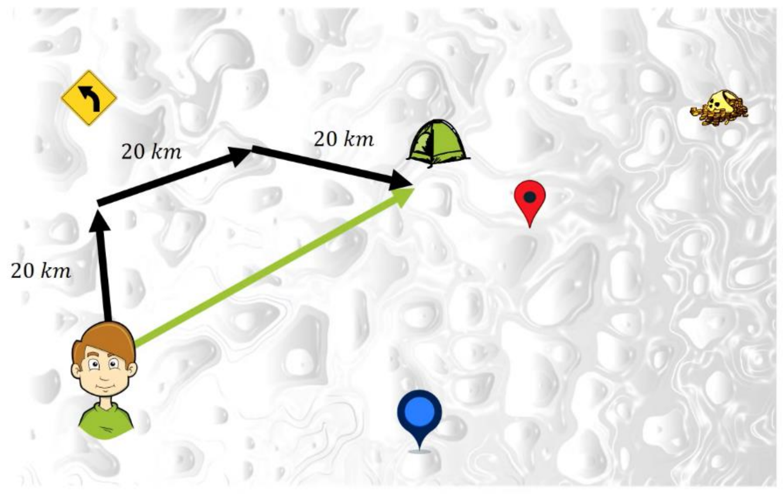
So, we understand that this random component varied the walking distance from 0 to 20 km, impacting the next position every day.

PSO search strategy



$2 \times r \times 10\ km$
$r\ in\ [0,1]$

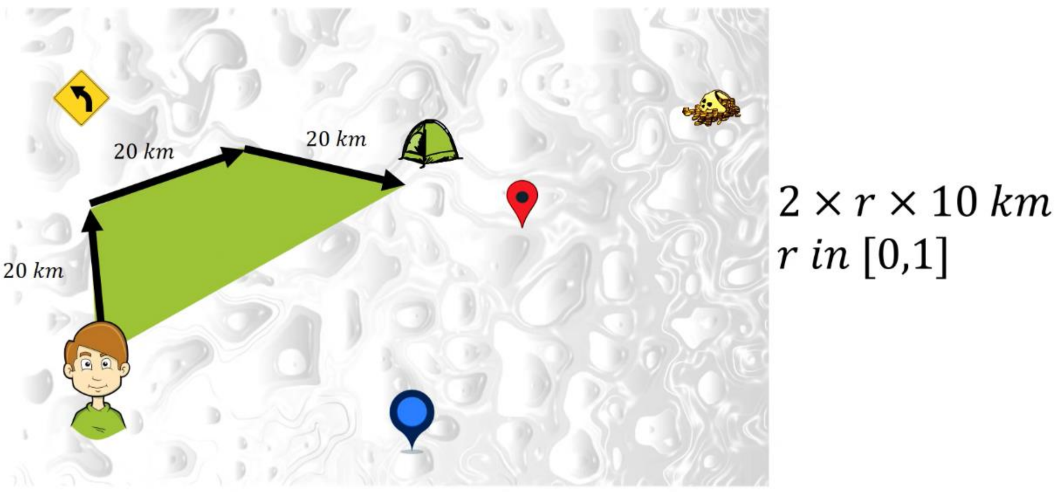
PSO search strategy



$2 \times r \times 10\ km$
$r\ in\ [0,1]$

In the above pictures we can observe the difference between how when Bob first sleeps at night after walking 10 km in each direction, and when Bob travels 20 km in each direction every day.

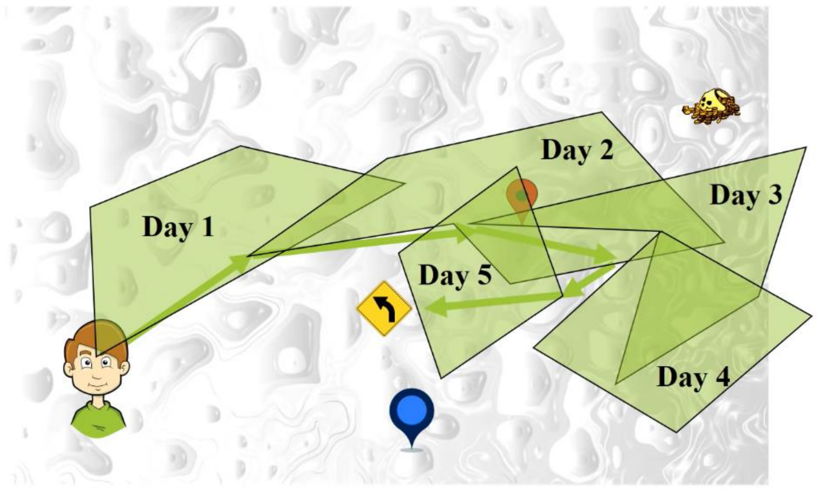
PSO search strategy



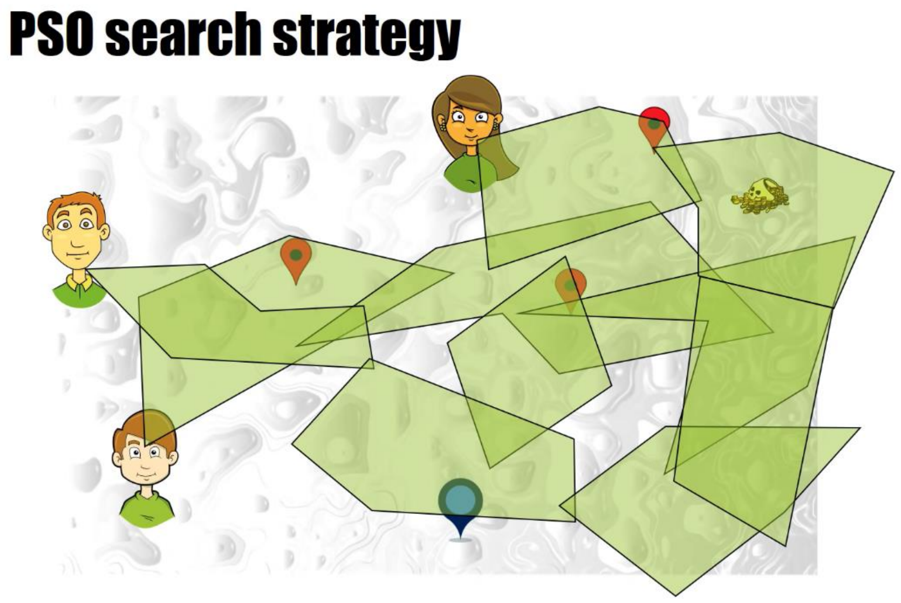
$2 \times r \times 10\ km$
$r\ in\ [0,1]$

Depending on the value of ‘r’ we may assume that Bob might be in the region highlighted in the above picture, meaning he could sleep anywhere in the region.

PSO search strategy



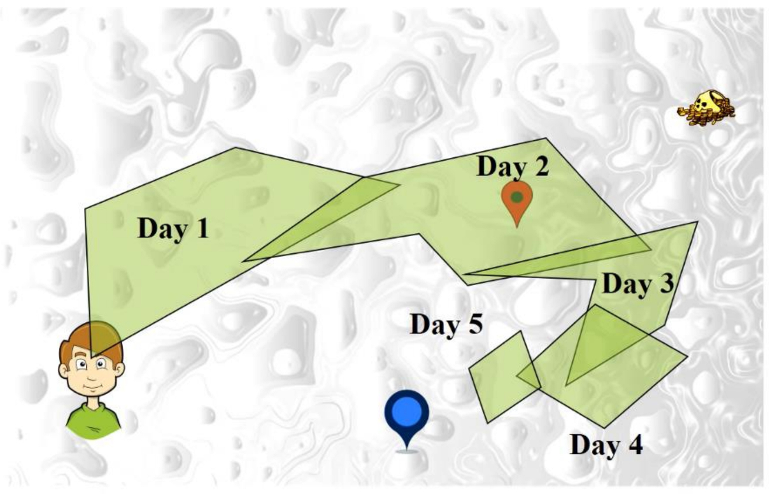
Having the possibility of Bob sleeping anywhere in the highlighted area, we may assume that the above highlighted regions should be the possible areas where Bob would have travelled 20 km in each directions, and would have continued to travel depending on the new positions he spent the nights and camped in the above enclosed regions, hence the above picture gives us a blueprint of his positions and successive location he is travelling. Imposing the same concept on all 3 members and their positions we can sketch their positions and supposed directions as follows :-



After analysing all the above details and assumptions of positions & directions of their journey a question arises which is , How can this method guaranty the discovery of the treasure or it’s exact location?

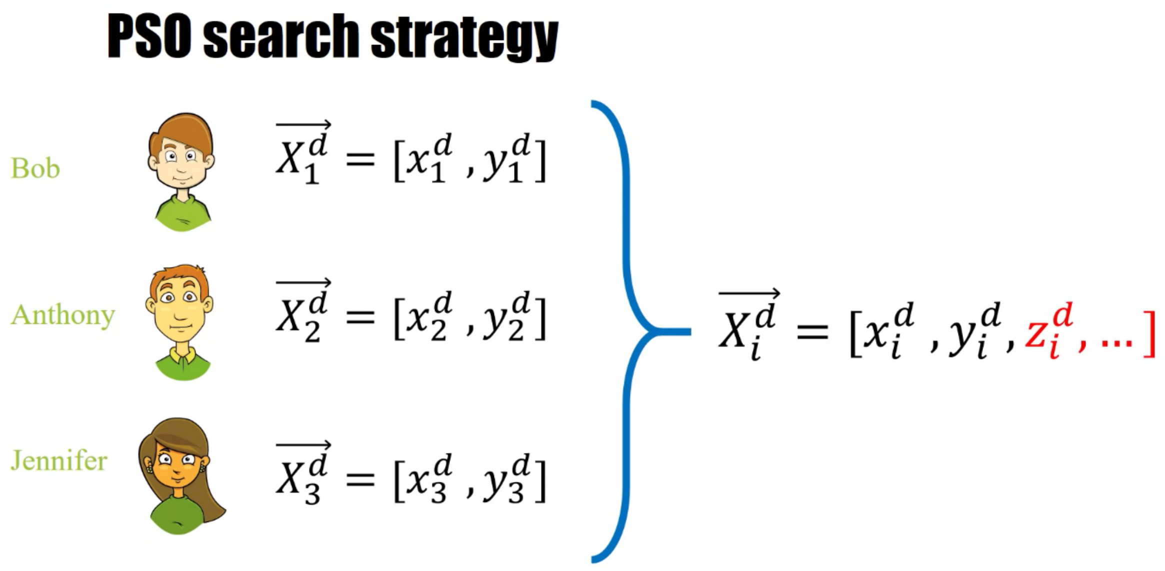
Generally speaking, as the team manages to select the best positions and search around them, they highly increase the possibility of finding the treasure, as in the picture we observe that at some point the team will end up digging in the position where the treasure is buried.

PSO search strategy



As we understand that the speed of walking and the number of days are directly proportional to each other, we also understand that with each day the region of search keeps getting local rather than global. This is exactly how the PSO algorithm search.

As we have now understood the concept of PSO and the theory behind it, let’s understand the mathematics behind it.


PSO search strategy
Bob
Anthony
Jennifer
$\overrightarrow{X_1^d} = [x_1^d, y_1^d]$
$\overrightarrow{X_2^d} = [x_2^d, y_2^d]$
$\overrightarrow{X_3^d} = [x_3^d, y_3^d]$
$\overrightarrow{X_i^d} = [x_i^d, y_i^d, z_i^d, \ldots]$

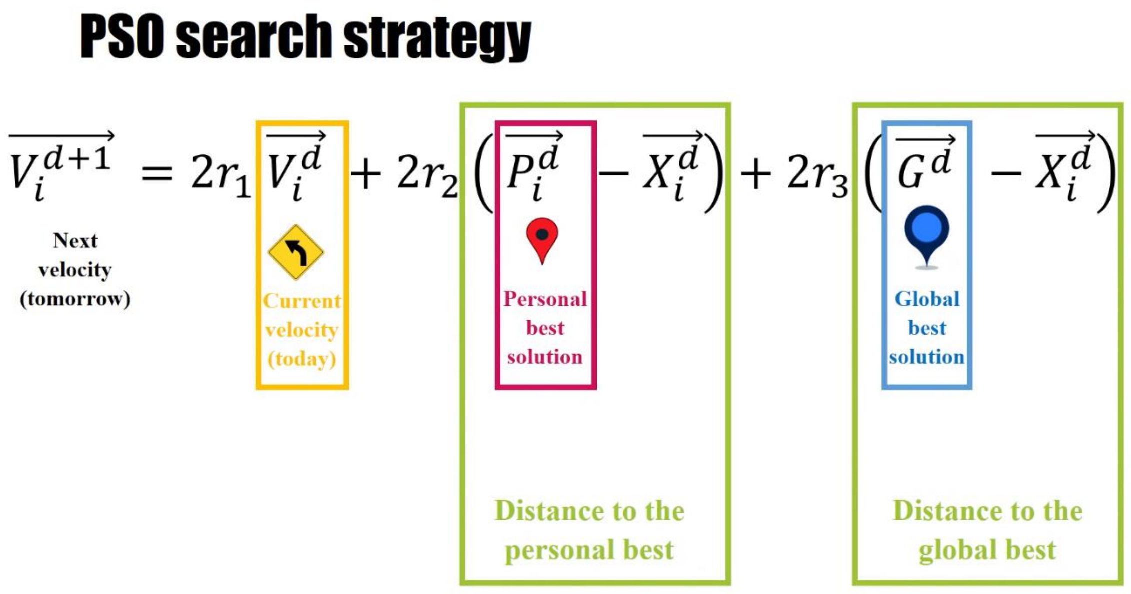
As we can observe that the position of each member is defined with pair of x and y, i.e. their coordinates, stored in a vector X. In the form of $\mathbf{X_i^d}$. Where –

d:- The number of days.

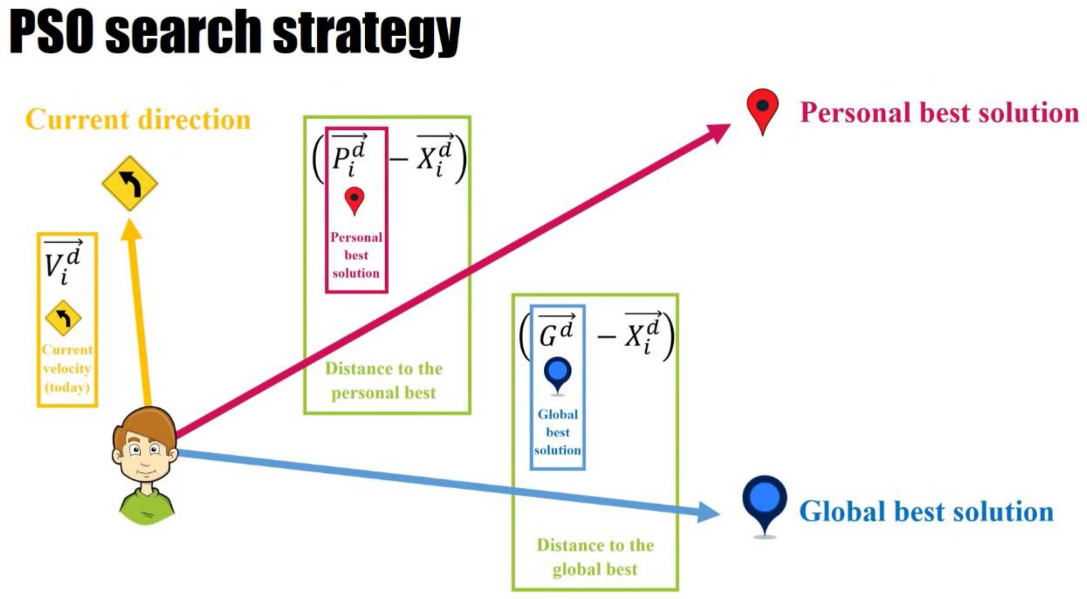
i:- Position vector of the i^{th} team member.

Also, if we move into N dimensional search space we can add more variables in the vector X.

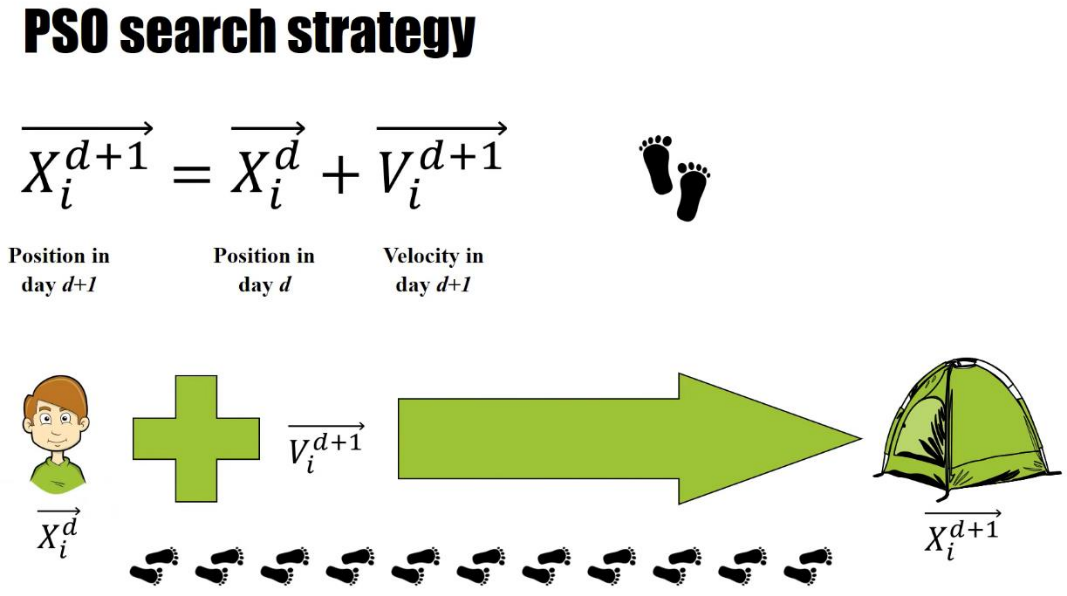
As we understood the mathematical terminology of position and its identification, we need to move in the next step i.e. the movements, this is where velocity vector comes into play.


PSO search strategy
$\overrightarrow{V_i^{d+1}} = 2r_1 \overrightarrow{V_i^d} + 2r_2 \left(\overrightarrow{P_i^d} - \overrightarrow{X_i^d}\right) + 2r_3 \left(\overrightarrow{G^d} - \overrightarrow{X_i^d}\right)$
Next velocity (tomorrow)
Current velocity (today)
Personal best solution
Global best solution
Distance to the personal best
Distance to the global best

The LHS shows the velocity vector of the next day, which is defined by 3 components, the current velocity, personal best and tendency towards the team’s best. The second and third component are calculates by finding the distance between the personal best and the distance between the global best, each component is multiplied by 2 times ‘r’, to randomly increase or decrease its impact.


PSO search strategy
Current direction
$\overrightarrow{V_i^d}$
Current velocity (today)
$\left(\overrightarrow{P_i^d} - \overrightarrow{X_i^d}\right)$
Personal best solution
Distance to the personal best
Personal best solution
$\left(\overrightarrow{G^d} - \overrightarrow{X_i^d}\right)$
Global best solution
Distance to the global best
Global best solution

As we proceed to calculate the velocity of the next day, the position can be calculated as well


PSO search strategy
$\overrightarrow{X_i^{d+1}} = \overrightarrow{X_i^d} + \overrightarrow{V_i^{d+1}}$
Position in day *d+1*
Position in day *d*
Velocity in day *d+1*
$\overrightarrow{V_i^{d+1}}$
$\overrightarrow{X_i^d}$
$\overrightarrow{X_i^{d+1}}$

In PSO algorithm every solution of the given problem is considered to be a particle, which is able to move in such landscape.

In order to operate the position of each particle 2 vectors are considered which are position and velocity vectors respectively. Although the position vector remains the same , there are changes in velocity vectors.

Mathematical model of PSO

$$\overrightarrow{X_i^{t+1}} = \overrightarrow{X_i^{t}} + \overrightarrow{V_i^{t+1}}$$

$$\overrightarrow{V_i^{t+1}} = \underbrace{w\overrightarrow{V_i^{t}}}_{\text{Inertia}} + \underbrace{c_1 r_1 \left(\overrightarrow{P_i^{t}} - \overrightarrow{X_i^{t}}\right)}_{\text{Cognitive component}} + \underbrace{c_2 r_2 \left(\overrightarrow{G^{t}} - \overrightarrow{X_i^{t}}\right)}_{\text{Social component}}$$

Where,

W:- Inertia of the particle multiplied by the velocity

t:- Time duration

The first component is called Inertia as it maintains the current velocity and movement direction.

The second component is called cognitive component or individual component, because in each particle is considered as the distance between its personal best and current position.

The last component is called Social component as the particle calculates the distance between its current position and the personal best position of the entire swarm. Also the variable G in the third component does not have a transcript as we have one base solution in the swarm for all the particles.

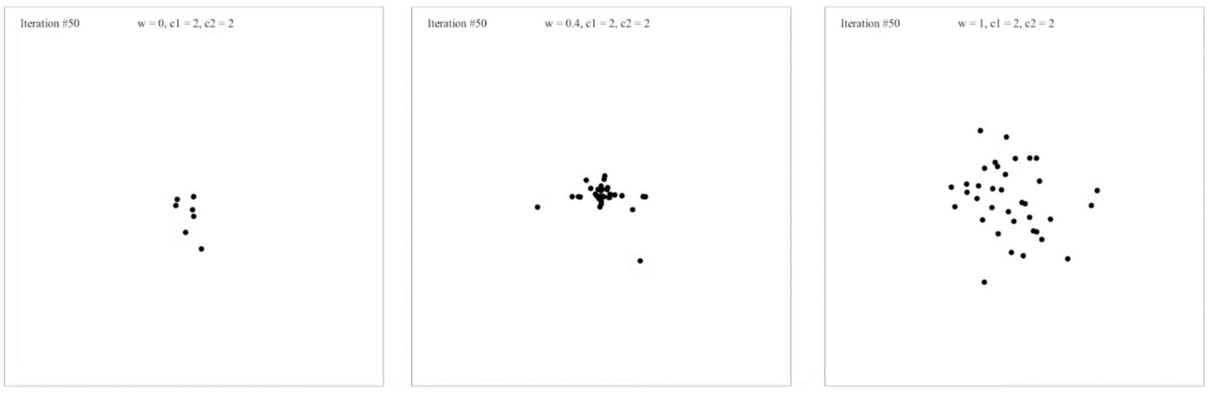
Also, the impact of cognitive and social components on the particle can be changed by tuning the coefficients c_1 and c_2.

Now, one must wonder ,what about the particle’s weight, doesn’t it have any role?

Well, this parameter tunes exploration and exploitation!

Here in PSO exploration is increased and exploitation is decreased, being proportional to the number of iterations, better understanding this factor by considering the fact that the region decreased as the number of days of travel increased.

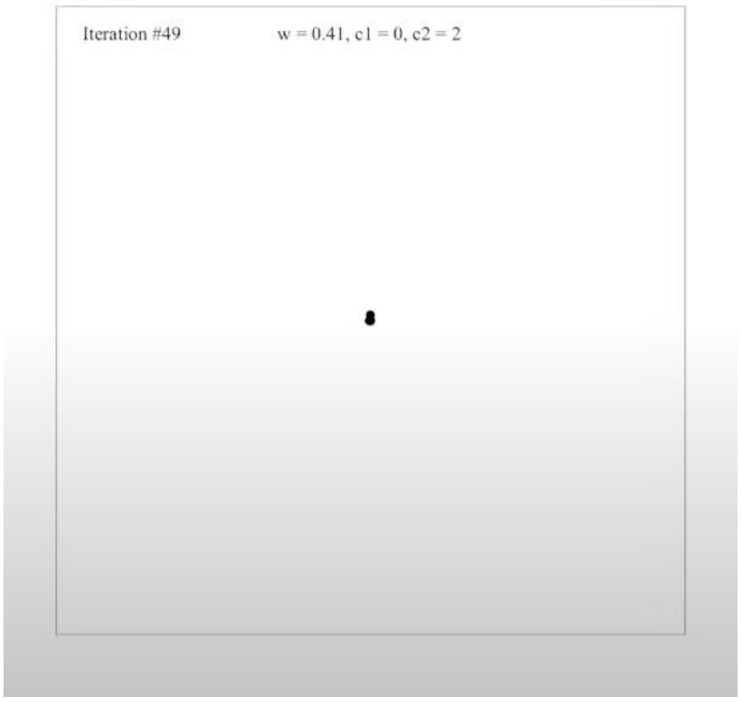
PARAMETERS’ IMPACT



In the above-mentioned figure it was observed that the exploration was at the lowest level when the inertia equals to 0, and it was maximum when the inertia was equal to , however balancing the parameters we can balance the exploration as well as the exploitation, which is an extremely useful mechanism when searching for the unknown global optimum of a real world problem.

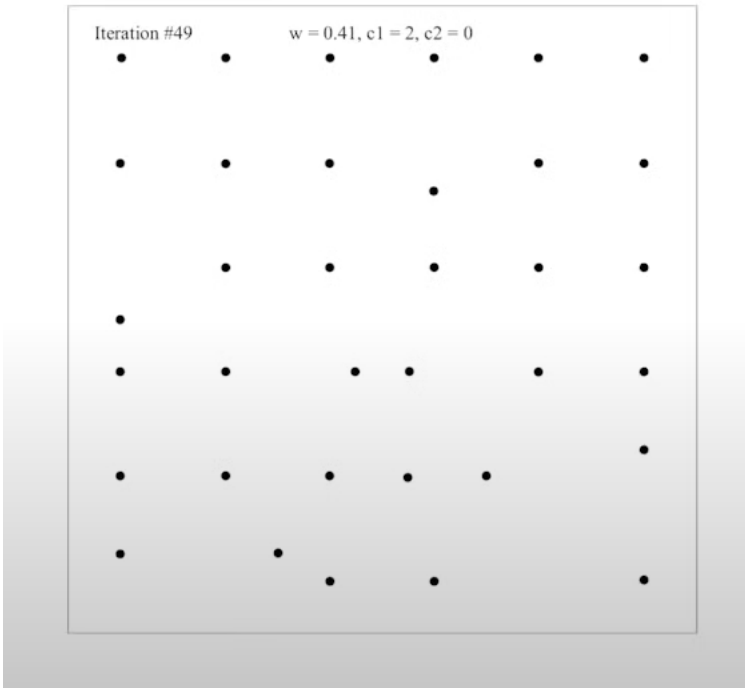
For $c_1 = 0$ & $c_2 = 2$

In this case we would observe that there is no individual component and every particle follows the best solution so far.



For $c_1 = 2$ & $c_2 = 0$

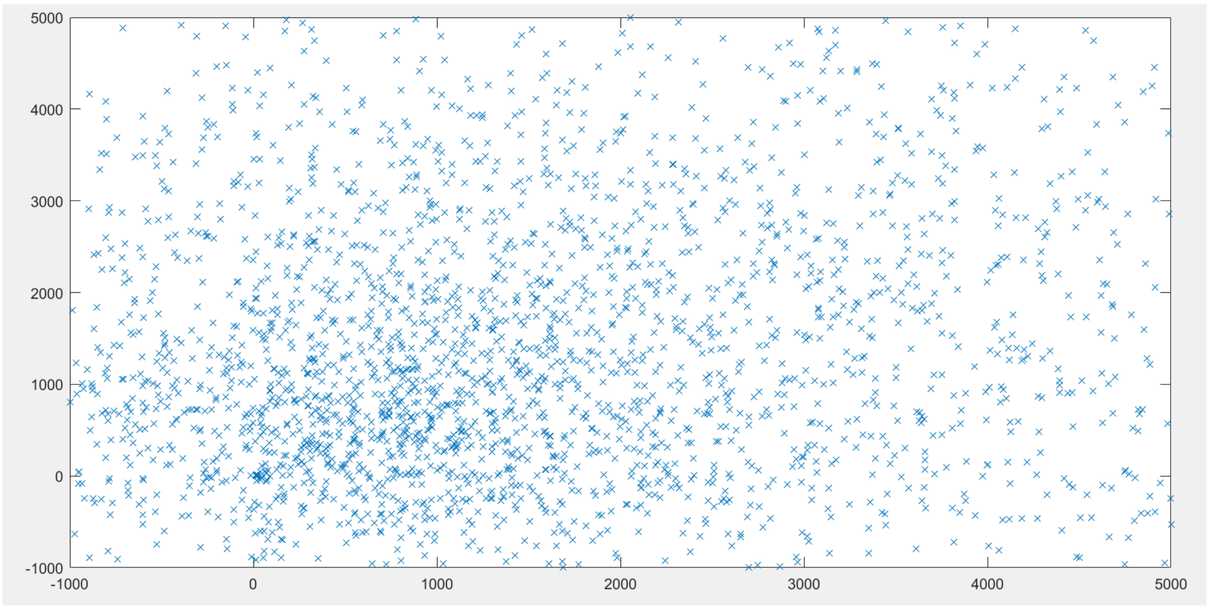
In this case every part of our searches one part of the search landscape, which simply means there is no exchange of information between the particles, as they don't need to know where the global base is.



MATLAB CODE

```
%% Particle Swarm Optimization Simulation
% Find minimum of  the objective function
%% Initialization
clear
clc
iterations = 1000;
inertia = 1.0;
correction_factor = 2.0;
swarms = 5000;
% ---- initial swarm position -----
swarm=zeros(5000,7);
step = 1;
for i = 1 : 5000
swarm(step, 1:7) = i;
step = step + 1;
end
swarm(:, 7) = 1000;        % Greater than maximum possible value
swarm(:, 5) = 0;           % initial velocity
swarm(:, 6) = 0;           % initial velocity
%% Iterations
for iter = 1 : iterations

    %-- position of Swarms ---
    for i = 1 : swarms
        swarm(i, 1) = swarm(i, 1) + swarm(i, 5)/1.2  ;  %update u position
        swarm(i, 2) = swarm(i, 2) + swarm(i, 6)/1.2 ;    %update v position
        u = swarm(i, 1);
        v = swarm(i, 2);

        value = (u - 20)^2 + (v - 10)^2;          %Objective function
        if value < swarm(i, 7)             % Always True
            swarm(i, 3) = swarm(i, 1);      % update best position of u,
            swarm(i, 4) = swarm(i, 2);      % update best postions of v,
            swarm(i, 7) = value;            % best updated minimum value
        end
    end
    [temp, gbest] = min(swarm(:, 7));         % gbest position

    %--- updating velocity vectors
    for i = 1 : swarms
        swarm(i, 5) = rand*inertia*swarm(i, 5) +
correction_factor*rand*(swarm(i, 3)...
            - swarm(i, 1)) + correction_factor*rand*(swarm(gbest, 3) -
swarm(i, 1));   % u velocity parameters
        swarm(i, 6) = rand*inertia*swarm(i, 6) +
correction_factor*rand*(swarm(i, 4)...
            - swarm(i, 2)) + correction_factor*rand*(swarm(gbest, 4) -
swarm(i, 2));   % v velocity parameters
    end

    %% Plotting the swarm
    clf
    plot(swarm(:, 1), swarm(:, 2), 'x')   % drawing swarm movements
    axis([-1000 5000 -1000 5000])
pause(.1)
end
```

OUTPUT OF THE PROGRAM



THE END