

Evolutionary Computation for Single and Multi-Objective Optimization
Dr. Deepak Sharma
Department of Mechanical Engineering
Indian Institute of Technology, Guwahati

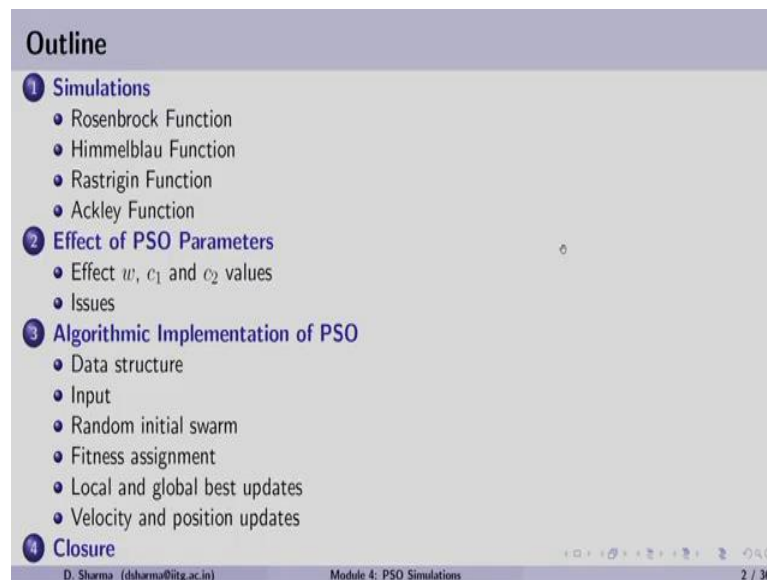
Module – 04

Lecture – 09

Simulations and Algorithmic Implementation of Particle Swarm Optimization

Welcome to the session on Simulations and Algorithmic Implementation of Particle Swarm Optimization. This session includes the simulations of PSO on four problems for which we know the optima.

(Refer Slide Time: 00:54)



The slide displays an 'Outline' section with a purple header. It lists four main topics, each with a numbered icon and a list of sub-points. The first topic is 'Simulations' with sub-points for Rosenbrock, Himmelblau, Rastrigin, and Ackley functions. The second is 'Effect of PSO Parameters' with sub-points for parameter effects and issues. The third is 'Algorithmic Implementation of PSO' with sub-points for data structure, input, random initial swarm, fitness assignment, local/global best updates, and velocity/position updates. The fourth is 'Closure'. The footer includes the presenter's name, email, module title, and slide number.

Outline	
1	Simulations <ul style="list-style-type: none">• Rosenbrock Function• Himmelblau Function• Rastrigin Function• Ackley Function
2	Effect of PSO Parameters <ul style="list-style-type: none">• Effect w, c_1 and c_2 values• Issues
3	Algorithmic Implementation of PSO <ul style="list-style-type: none">• Data structure• Input• Random initial swarm• Fitness assignment• Local and global best updates• Velocity and position updates
4	Closure

D. Sharma (dsharma@iitg.ac.in) Module 4: PSO Simulations 2 / 36

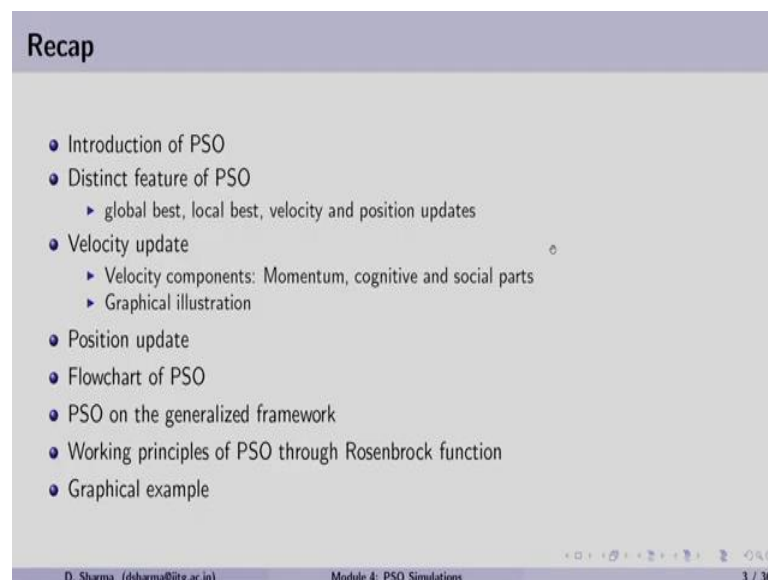
We will start with Rosenbrock Function followed by, we will solve Himmelblau Function, then we have a Rastrigin Function and Ackleys Function. After performing the simulations we will also see the effect of the parameters with PSO. So, as we know there are three main important parameters which are omega that is multiplied with the velocity and there are two constant; c_1 and c_2 which are also a part of the velocity in PSO.

We will also discuss the issues with PSO and at the last we will understand the algorithmic implementation of PSO. And we will go through the data structures and various functions using which we can make a code or source code for a particle swarm optimization and at the end we will conclude this session. So, before we start the simulation, let us have a recap.

So, in this last session we have gone through the particle swarm optimization from there we know that PSO is based on artificial life and evolutionary computing. Artificial life in a sense, that it is mimicking the flocking of the birds. So, the birds are moving, they are searching the food in a cooperative way.

Similarly, since there are many birds or say particles are involved, those particles are evaluated, compared and the new particles are generated or their positions are updated therefore, it has both the features of artificial life and evolutionary computing. Thereafter, we found that there are three distinct feature with PSO.

(Refer Slide Time: 02:53)



These three distinct features include; the global best of the swarm, local best of each particle and the velocity and position update. As we can see in the slide that in the velocity update we have three parts; one is called momentum part, another is called cognitive and the third is called social part.

In the momentum part we include the previous velocity, in the cognitive part we include the difference of the current position of the particle with its local best. And in the social part we include the difference between the current position and the global best of the swarm.

So, all these parts are combined together to make a velocity components. We understood this velocity component as a graphical, as a addition of the vectors through an graphical

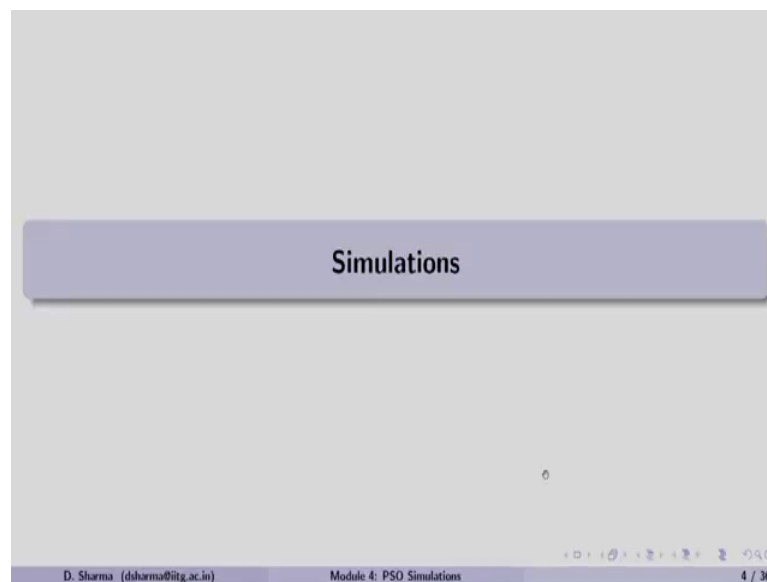
example. The position update is simple in PSO which includes the current position plus the updated velocity. We understood particle swarm optimization through flowchart in which we have one standard loop of generation and one loop on the number of particles.

So, in the number of particles we update the position as well as we update the velocity followed by the position. We also fit our particle swarm optimization on the generalized framework of easy techniques. Thereafter, we understood the working principles of particle swarm optimization through Rosenbrock function.

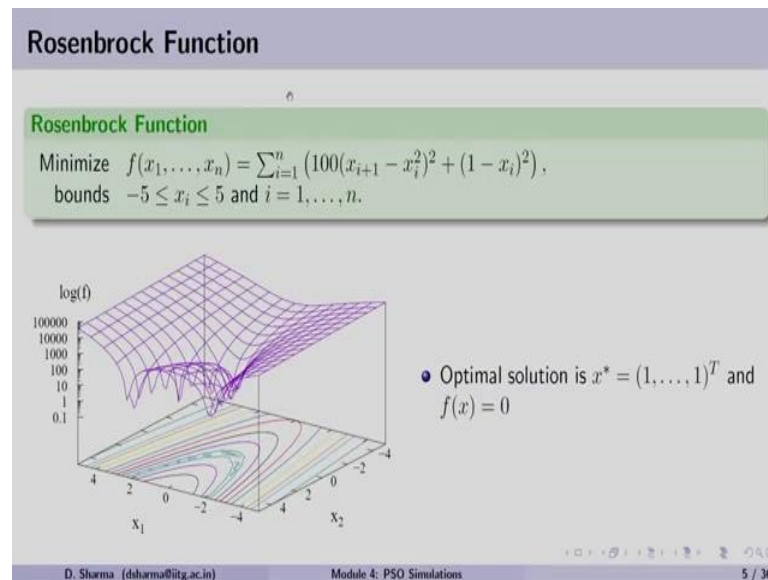
So, in that case we run for two number of generations to understand how we have found the local best and the global best for the particle and the swarm. And using those component we updated the velocity followed by, we updated the position.

The same example, which we solved for two generations, we showed the graphical illustration in which we can find that how these particles are moving. Their local best are improving and finally, the global best is also improving to get the optimum solution for the given problem.

(Refer Slide Time: 05:26)



(Refer Slide Time: 05:31)



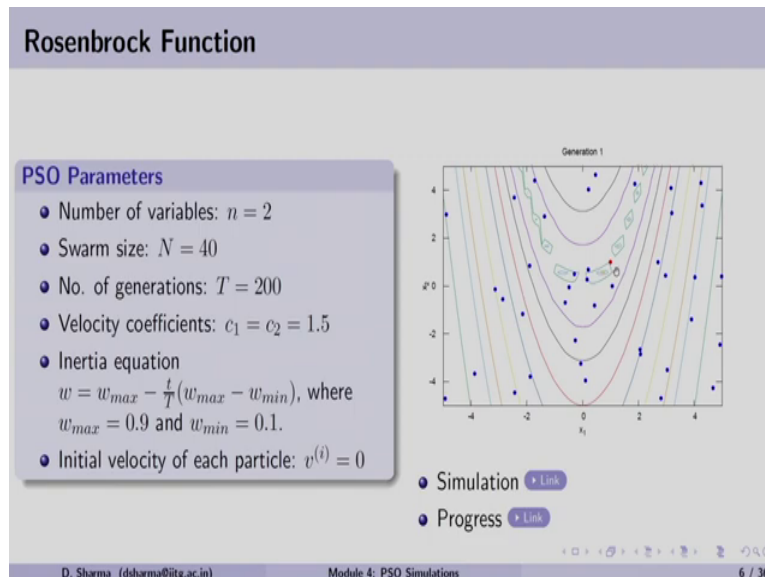
Now, let us start with the simulations now. In this simulation we will start with the Rosenbrock function as you can see here. Now, this Rosenbrock function is a scalable function which we can write in terms of n number of variable. So, the objective function is written on the top and the variable bound is given as minus 5 to plus 5.

$$\text{Minimize } f(x_1, \dots, x_n) = \sum_{i=1}^n (100(x_{i+1} - x_i^2)^2 + (1 - x_i)^2)$$

$$\text{bounds } -5 \leq x_i \leq 5 \text{ and } i = 1, \dots, n$$

If we show this Rosenbrock function for 2 variables as a 2 variables now, in this case the third axis is drawn as the logarithmic of the function. Now, looking at the surface of the Rosenbrock function and the contours which are drawn in x 1 and x 2 plane we can see there are lot of local optima. However, as we as it is mentioned on the right hand side, the global optima is lying at 1 1 1 1 and the function value is 0 at the global optima.

(Refer Slide Time: 06:32)

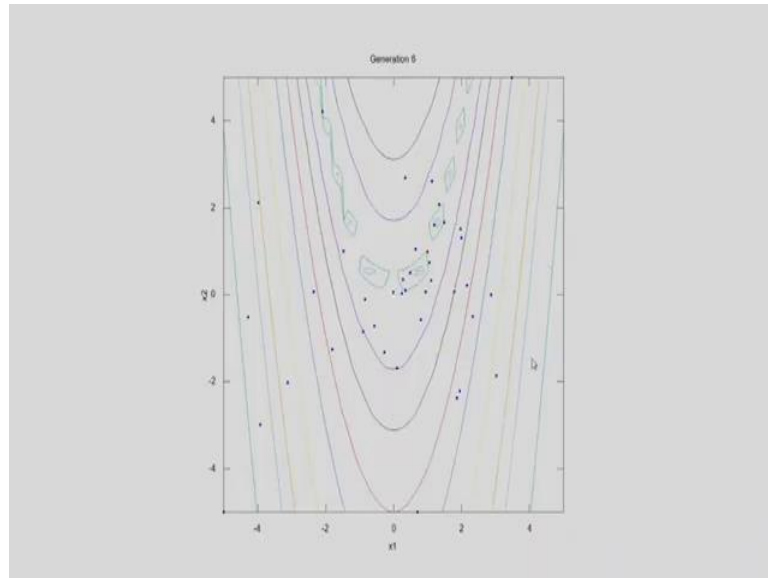


So, let us have the first simulation on Rosenbrock function. In this particular simulation we have fixed the number of variable equals to 2, swarm size 40, number of generation is 200, the two velocity come constants which are c_1 and c_2 , those are both of them are fixed to 1.5.

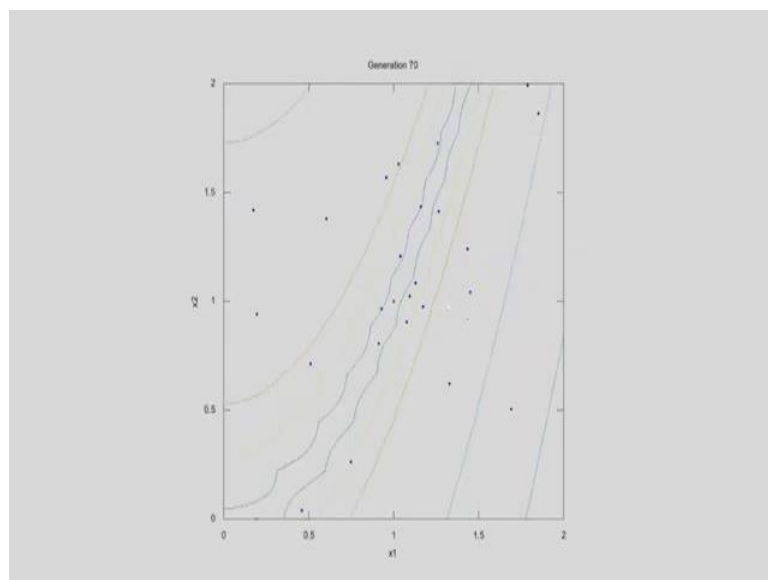
Now, here instead of having a constant ω we have included an inertia equation which says that ω equals to $\omega_{max} - \frac{t}{T}(\omega_{max} - \omega_{min})$. This particular function will help us to take the ω value large at the beginning say 0.9 and this ω value keep on reducing when the number of generation will be increasing.

So, this means that initially the velocity will be having a component, significant component in the velocity update and in the later stage this component will be reduced. The initial velocity of each particle is taken as 0. So, on the right hand side we can see that the solutions are distributed in x_1 and x_2 plane and contours of the Rosenbrock function is are shown in this figure, so this is called initial swarm. So, let us see how this PSO work here.

(Refer Slide Time: 08:05)



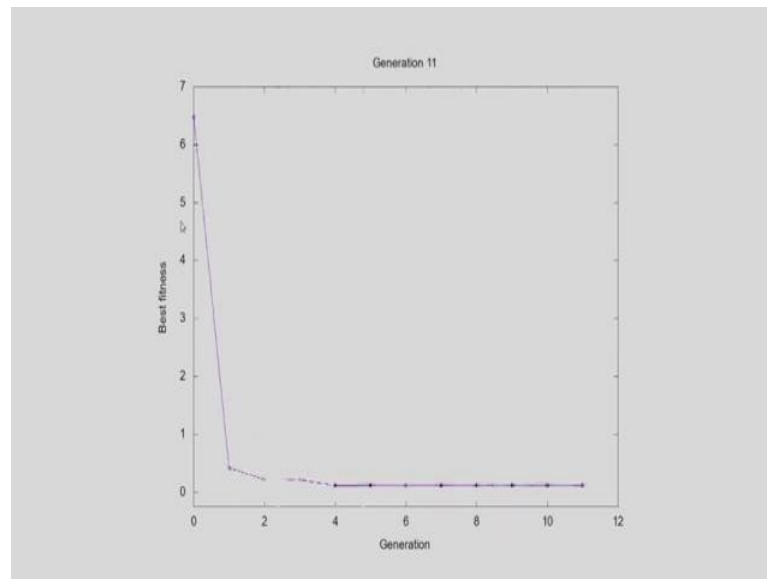
(Refer Slide Time: 08:12)



So, in this particular simulation the blue dots are the particles position, which are getting updated in every generation. As you can see that few particles I have, are coming closer to the global best which is the red dot here.

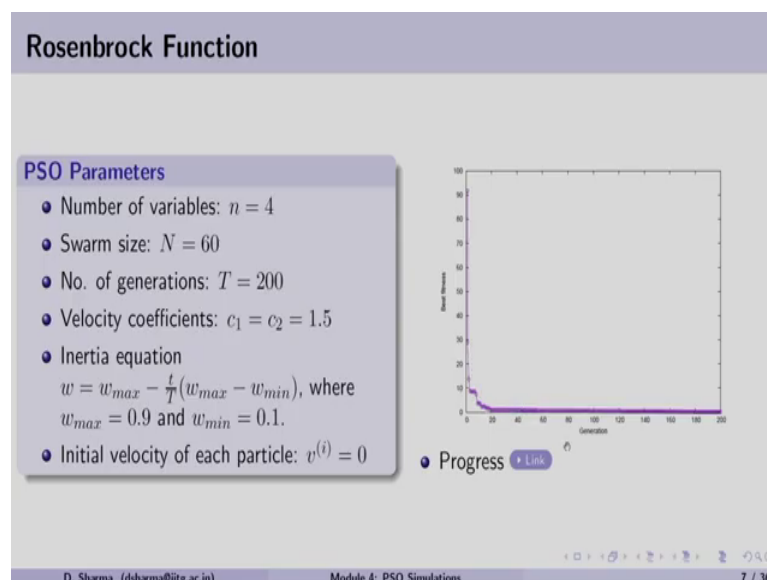
So, as soon as any of the close goes closer to this red dot, the local best as well as the global best for the swarm gets changed. And that is why in the beginning the particles are randomly moving in x_1 and x_2 plane and after few number of generations the particles gets attracted, because the global best is on the optimum solution.

(Refer Slide Time: 08:56)



If we look the progress here, the progress we will see that the y axis is written as a best fitness and the x axis is written as the generation. So, if y best fitness suggest that we are printing the value of global best, as we can see that initially it is started at close to 6.5, drastically the fitness is reduced and even in 10 generations, we are very close to the optima the fitness is keep on improving. And close to 65 generation we have found the optima of this problem using PSO.

(Refer Slide Time: 09:41)

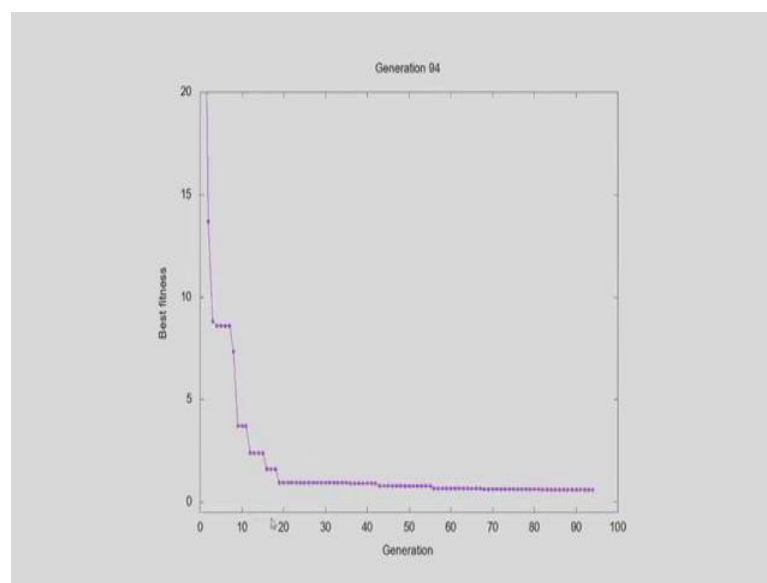


Let us move to the second simulation here. Here, the number of variable now, it is increased to 4 and at the same time the swarm size is taken as 60. So, this also has been increased with respect to the previous simulation study. The number of generation is kept

200, the velocity coefficients are same as c_1 and c_2 equals to 1.5, we are using the same inertia equation and finally, the initial velocity we start with 0.

So, since it is the 4 variable, we will see how the best global best is progressing with the generation. As you can see from the plot on the right hand side, it is started a little more than 90 and then drastically the fitness of the solution or the global best is improving and then it has reached to the optima after few generation. So, let us see how the progress is going through.

(Refer Slide Time: 10:48)



Now, in this particular simulation as we can see there is a drastic improvement in the performance in the initial generation and close to 20, it is we are near to the optima and with the generation the fitness value is keep on improving. So, the best fitness means the fitness of the global best is keep on improving.

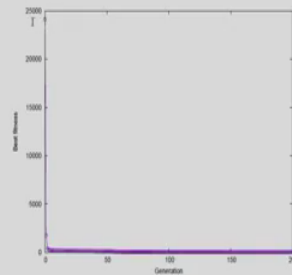
So, let us see one more time, the how this progress is going on. So, as you can see there is a little improvement in the fitness value of the global best and it is quite close to 0 at 200 generation. Meaning that; if we are going to run this for little more number of generation, then PSO can find the optimum solution for n equals to 4 number of variables for Rosenbrock function.

(Refer Slide Time: 11:46)

Rosenbrock Function

PSO Parameters

- Number of variables: $n = 10$
- Swarm size: $N = 60$
- No. of generations: $T = 200$
- Velocity coefficients: $c_1 = c_2 = 1.5$
- Inertia equation
 $w = w_{max} - \frac{t}{T}(w_{max} - w_{min})$, where
 $w_{max} = 0.9$ and $w_{min} = 0.1$.
- Initial velocity of each particle: $v^{(i)} = 0$

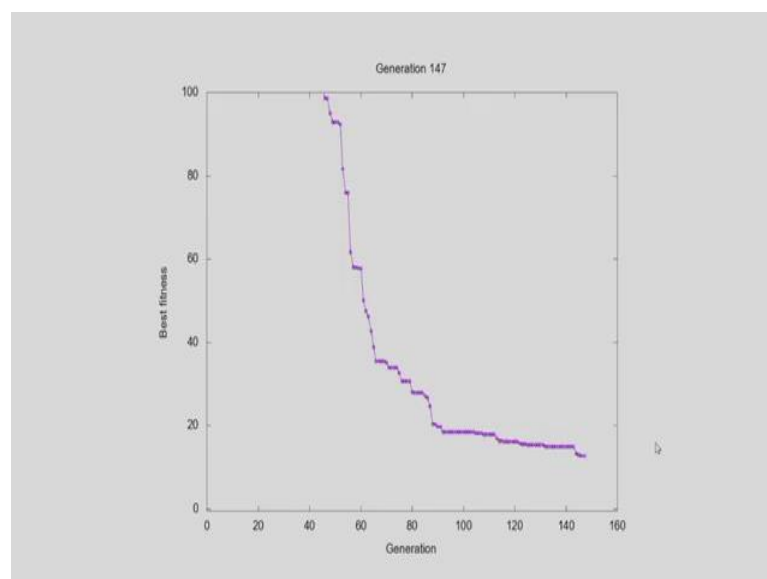


• Progress [Link](#)

Now, let us move to the third simulation here. In this case we have taken number of variable as n equals to 10, swarm size is kept as 60, the number of generation is 200, and the velocity c_1 and c_2 is velocity coefficients are 1.5. We are using the same inertia equation and our initial velocity of each particle is again 0. So, since it is a 10 variable problem, we will see how our global best particle is improving.

Now, looking at the figure on the right hand side, we can see that the global best started from 25000 value which is a big number and it reaches close to the 0 after few generation.

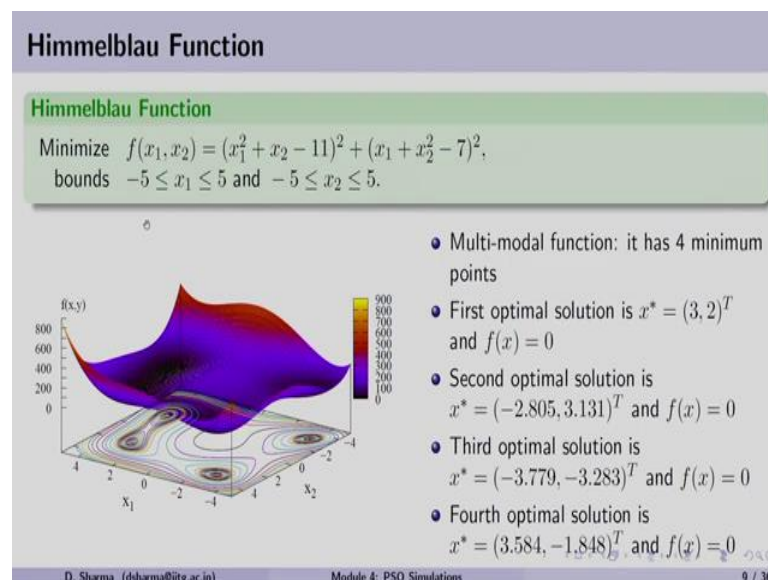
(Refer Slide Time: 12:40)



So, let us see the simulation now. As you can see here the fitness is improving drastically in the initial generation. If we zoom the y axis from 10 to 0, we can see that the global best or the best fitness in the swarm is keep on improving with number of generation.

And although after 200 number of a generation we are not at the optimum solution, but using a limited number of population and number of a generation we have reached quite close to the global optimum. So, if we increase the number of increase the number of swarm size and the number of generation PSO can find the optimum solution for n equals to 10 Rosenbrock function.

(Refer Slide Time: 13:34)



Now, come to the second function which is a Himmelblau function. The, we want to minimize this function as it is written on the top and both the variables are lying between minus 5 to plus 5.

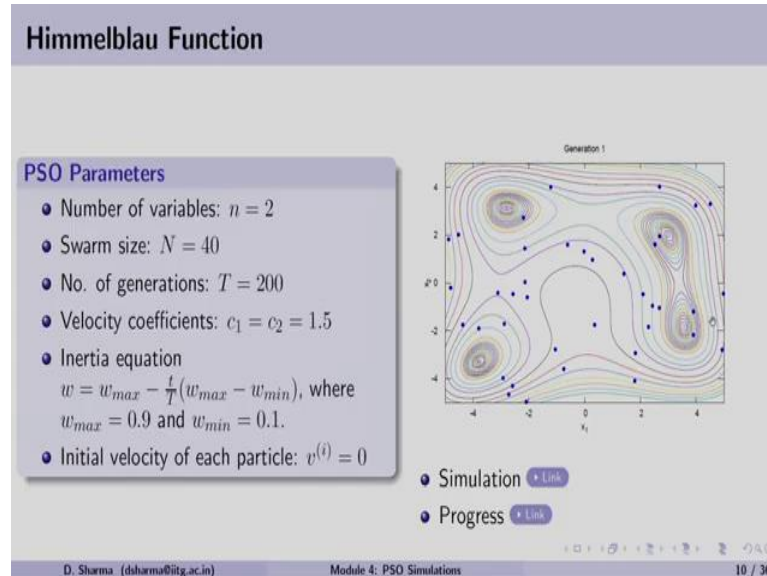
$$\text{Minimize } f(x_1, x_2) = (x_1^2 + x_2 - 11)^2 + (x_1 + x_2^2 - 7)^2$$

$$\text{bounds } -5 \leq x_1 \leq 5 \text{ and } -5 \leq x_2 \leq 5$$

Looking at the surface of the Himmelblau function as well as if we see the contours on x_1 and x_2 plane we can see that this particular function is multi modal function. So, multi modal function as we know that we can have different optimum solution. So, looking at

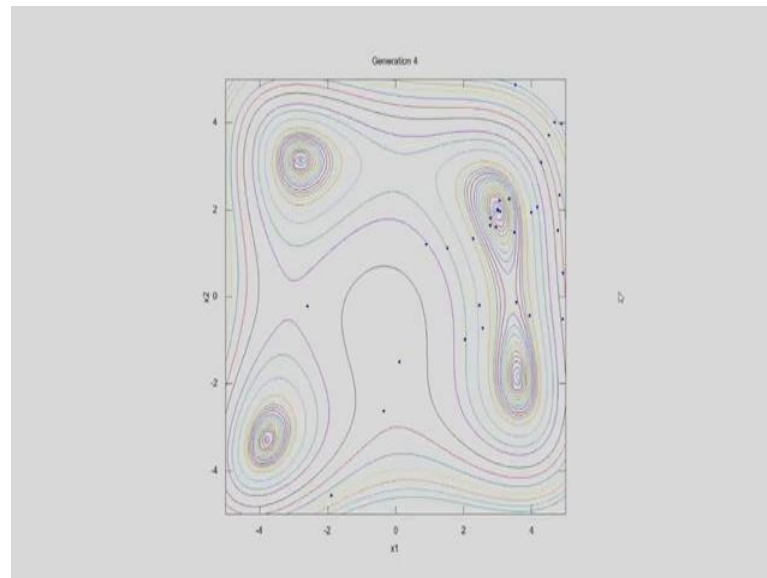
the right hand side we have four optimum solution for this Himmelblau function and every point the function value is 0.

(Refer Slide Time: 14:21)



So, let us solve this problem using particle swarm optimization. So, since here we have a number of variable equals to 2, swarm size is kept 40, number of generation is 200, c_1 c_2 are 1.5, inertia equation as we have defined earlier, and the initial velocity is again kept 0. On the right hand side figure, we can see that our solutions or the particles are randomly distributed so this is called random swarm and let us see how this particular swarm approaches towards the optimum solution.

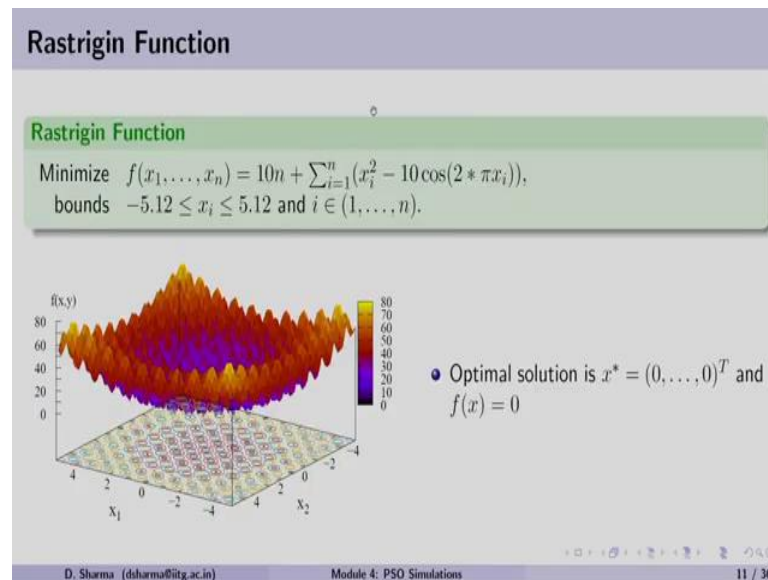
(Refer Slide Time: 14:58)



Now, as you know that any blue point goes close to the optimum solution, our global best solution gets updated. So, these blue points in this simulation signifies the position of the particle in every generation and since the particle has already reached to the global optimum solution or one of the optimum solution, the other particles are getting attracted towards this optimum solution.

Second observation you can see that the particles are getting attracted to one of the optimum solution, but the three solutions, other three solutions we cannot find it. It is only, because the PSO which we have made it to find the global solution not for multi modal. So, we have to make appropriate change with particle swarm optimization and then it can work for multi modal functions.

(Refer Slide Time: 16:01)



Coming to the third function, which is Rastrigin function. Now, Rastrigin function is again the scalable function. As you can see that the function can be written in terms of n number of variable. We want to minimize the function and the function is given on the top. The variable bound is given as minus 5.12 to 5.12 plus 5.12.

$$\text{Minimize } f(x_1, \dots, x_n) = 10n + \sum_{i=1}^n (x_i^2 - 10 \cos(2 * \pi x_i))$$

$$\text{bounds } -5.12 \leq x_i \leq 5.12 \text{ and } i \in (1, \dots, n)$$

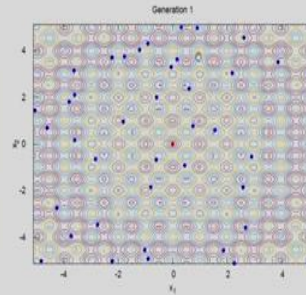
Now, looking at the surface on the left hand side as well as the contour on x 1 and x 2 plane we can see that there are so many local optima's. And therefore, restoring function is always difficult to solve, but it has just one global optima which is at the origin as you can see on the right hand side and the function value is 0.

(Refer Slide Time: 16:54)

Rastrigin Function

PSO Parameters

- Number of variables: $n = 2$
- Swarm size: $N = 40$
- No. of generations: $T = 200$
- Velocity coefficients: $c_1 = c_2 = 1.5$
- Inertia equation
 $w = w_{max} - \frac{t}{T}(w_{max} - w_{min})$, where
 $w_{max} = 0.9$ and $w_{min} = 0.1$.
- Initial velocity of each particle: $v^{(i)} = 0$



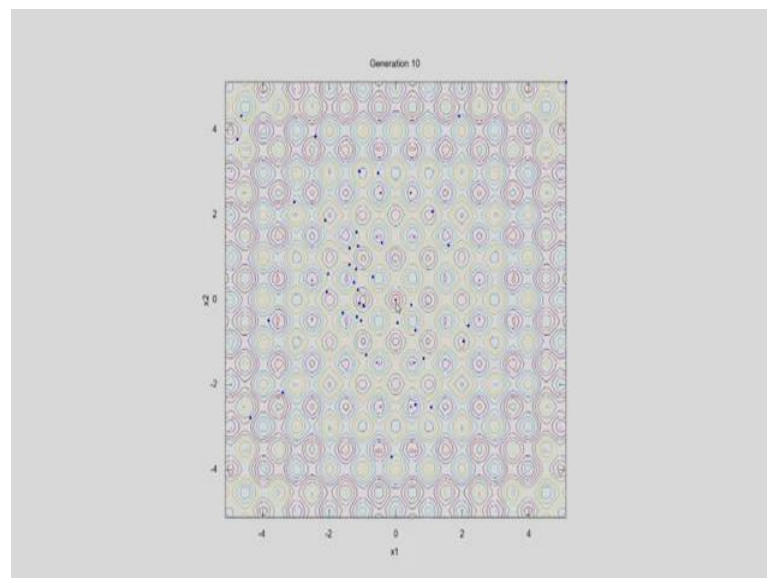
• Simulation [Link](#)

• Progress [Link](#)

So, let us solve Rastrigin function using PSO we have taken the first case as n equals to 2, swarm size is equals to 40, number of generation equals to 200, c_1 c_2 equal to 1.5, the inertia equation is the same and the initial velocity is kept 0.

Now, the figure on the right hand side we can see that the solutions are distributed in x_1 and x_2 plane. Since, it is the initial swarm so that is why these solutions are distributed randomly in x_1 and x_2 plane. Now, let us see how this PSO going to solve Rastrigin function.

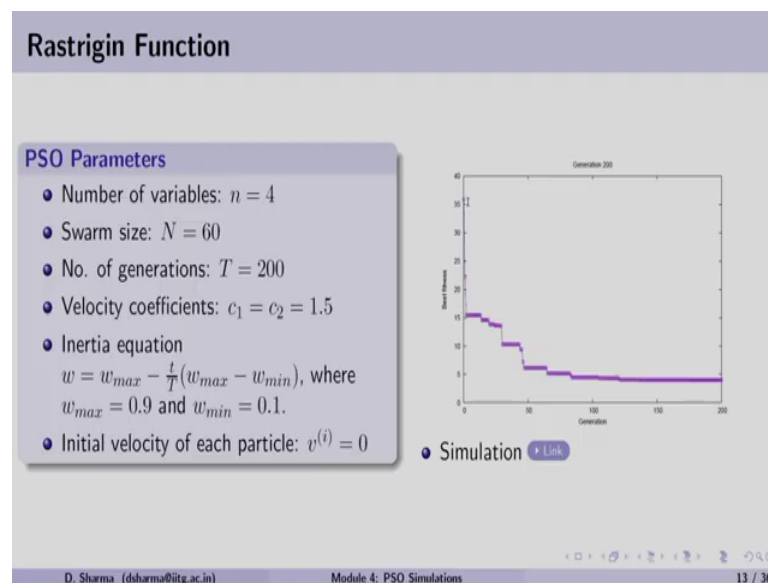
(Refer Slide Time: 17:42)



Now, the blue dots as you can see these are the particle positions and as soon as the particle they are improving, the global best and local best are improving. And as you can see the global best is already at the optima, the other particles are converged converging to the global optima which is shown in the red color. So, this is these are the properties that as soon as our global best is converged to the optima or the global optima, other particles will get attracted towards this optimal solution.

Let us see the progress of the Rastrigin function here for 2 variable. Again, the best fitness is the fitness of the global best in the swarm in every generation. As you can see it is started close to 11, it is keep on improving till 50 generation and then there is a there is no improvement, why? Because we have already reached to the optima using PSO, so that is why you can see that PSO is quite efficient that even close to the 50 number of generation, the optima is found by PSO.

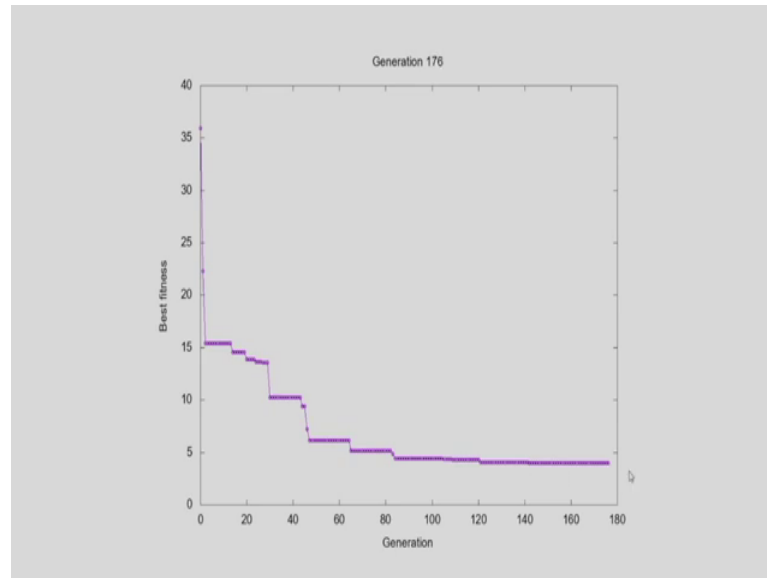
(Refer Slide Time: 18:59)



Now, let us move to the another simulation here. In this case the Rastrigin problem is solved using n equals to 4 meaning 4 number of variables; the swarm size is increased to 60, number of generation is 200, c_1 and c_2 is 1.5, and inertia equation remains the same and our initial velocity we have kept it 0. Since, it is a 4 variable problem. We will show the best fitness the progress of the for the Rastrigin function.

Now, as you can see that the best fitness is starting little more than 35 which keeps on improving with the number of generations and it has found a solution close to the fitness value 5. So, let us see how the solutions are progressing.

(Refer Slide Time: 19:56)



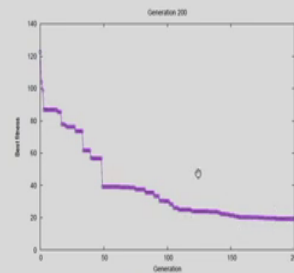
So, in this particular progress we can see that in every generation there is a improvement in the fitness of the global best which is the best fitness of the swarm which keep on improving. As you can see that after 120 generations the fitness is already less than 5 and thereafter it can, it is terminated based on the number of generation. If we allow PSO for more number of generation and also with large number of swarm size, then this PSO can solve this particular problem which is for n equals to 4 number of variable.

(Refer Slide Time: 20:39)

Rastrigin Function

PSO Parameters

- Number of variables: $n = 10$
- Swarm size: $N = 60$
- No. of generations: $T = 200$
- Velocity coefficients: $c_1 = c_2 = 1.5$
- Inertia equation
 $w = w_{max} - \frac{t}{T}(w_{max} - w_{min})$, where
 $w_{max} = 0.9$ and $w_{min} = 0.1$.
- Initial velocity of each particle: $v^{(i)} = 0$

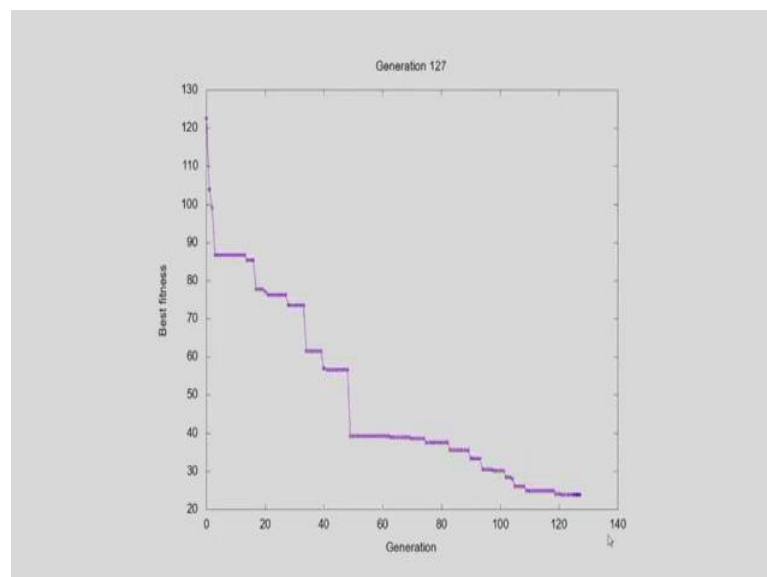


• Simulation [Link](#)

Coming to the Rastrigin problem with n equals to 10. The number of variable is increased to 10 now, swarm size is 60, number of generation is 200, velocity coefficients are 1.5, inertia equation remains the same and the velocity initial velocity we kept it 0.

Now, since it is a 10 variable problem, we will show you the progress of the best fitness with respect to the generation. As you can see on the right hand side the fitness, the best fitness is keep on improving and then close to 150 generation the improvement is less.

(Refer Slide Time: 21:25)

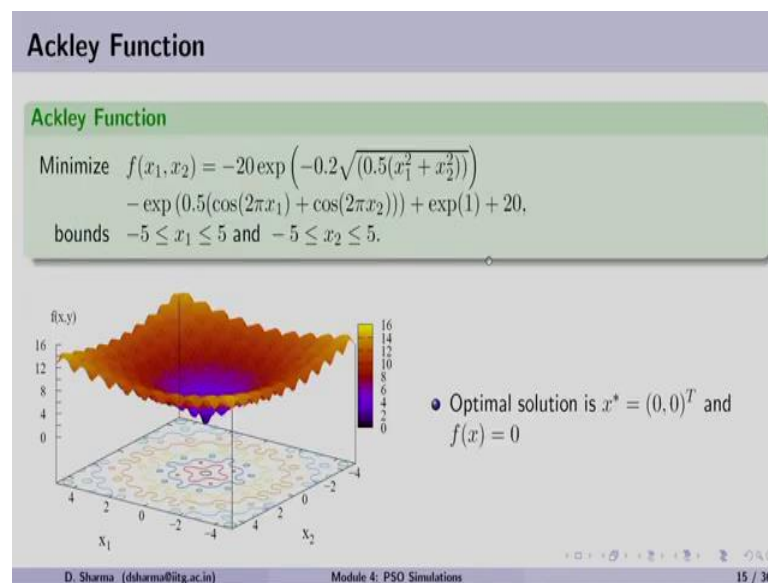


So, let us see the simulation now. So, it is started somewhere close to 123 and you can see that the fitness, the best fitness in the swarm is keep on improving and still it is not reached

to the 20 as you can see on the y axis. And thereafter, close to 180 it has converged little less than fitness value less than 20.

Meaning that although PSO has not converged in 200 generation, but if we allow PSO for more number of generations and for a larger swarm size, we, this PSO can converge to the optimum solution for n equals to 10 variable for Rastrigin function.

(Refer Slide Time: 22:12)

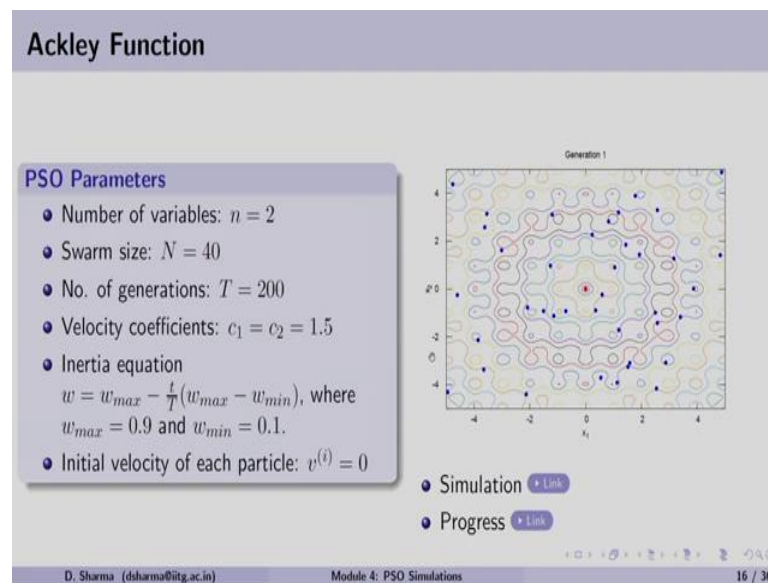


Now, let us come to the last function that is Ackley's function on the top we can see that we want to minimize this function and it involve exponential as well as cos term. So, that is why this particular problem is difficult to solve. Both the variables are lying between minus 5 to plus 5.

$$\begin{aligned} \text{Minimize } f(x_1, x_2) &= -20 \exp \left(-0.2 \sqrt{0.5(x_1^2 + x_2^2)} \right) \\ &- \exp \left(0.5(\cos(2\pi x_1) + \cos(2\pi x_2)) \right) + \exp(1) + 20 \\ \text{bounds } &\& -5 \leq x_1 \leq 5 \text{ and } -5 \leq x_2 \leq 5 \end{aligned}$$

Now, looking at the figure on the left hand side, you can see the surface as well as the contours on x_1 and x_2 plane. We can see that this particular function has lot many local optima's, but the global optima as you can see on the right hand side it is lying at the origin and the function value or the optimum function value for the Ackley function is 0.

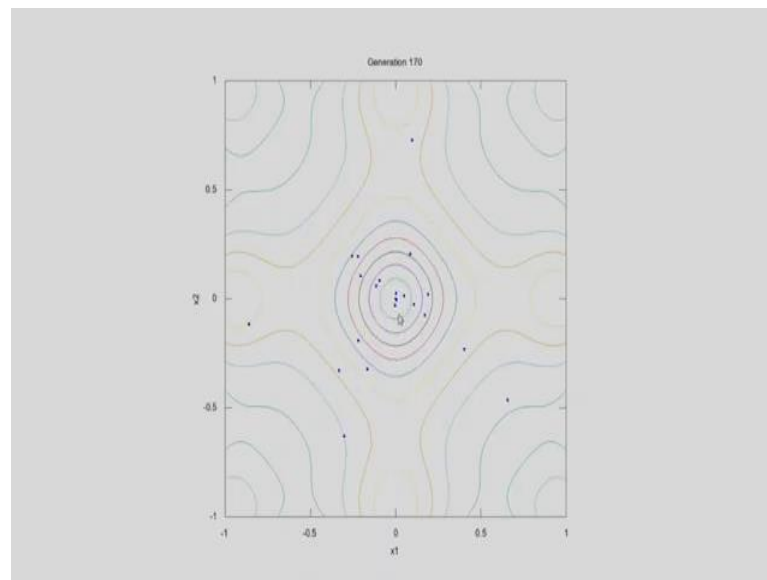
(Refer Slide Time: 23:02)



So, this is a 2 variable problem the swarm size is kept 40, number of generation is 200, c_1 and c_2 are 1.5, inertia equation remains the same, and initial velocity is kept 0. So, you can see that in our all experiments or simulations we have started with the 0 velocity.

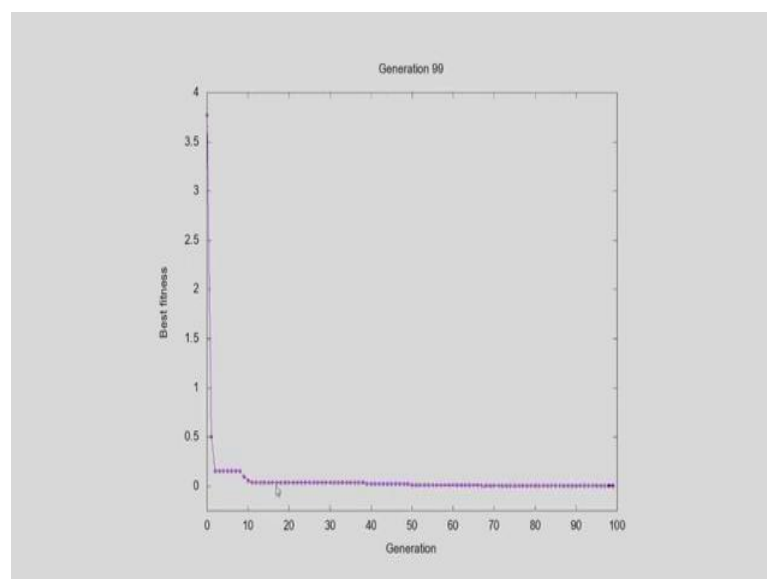
On the right hand side figure, we can see that the solutions are distributed throughout the x_1 and x_2 plane and since it is a generation 1 meaning that we started with the random swarm here. So, let us see how is the simulation, how PSO is solving this particular problem.

(Refer Slide Time: 23:49)



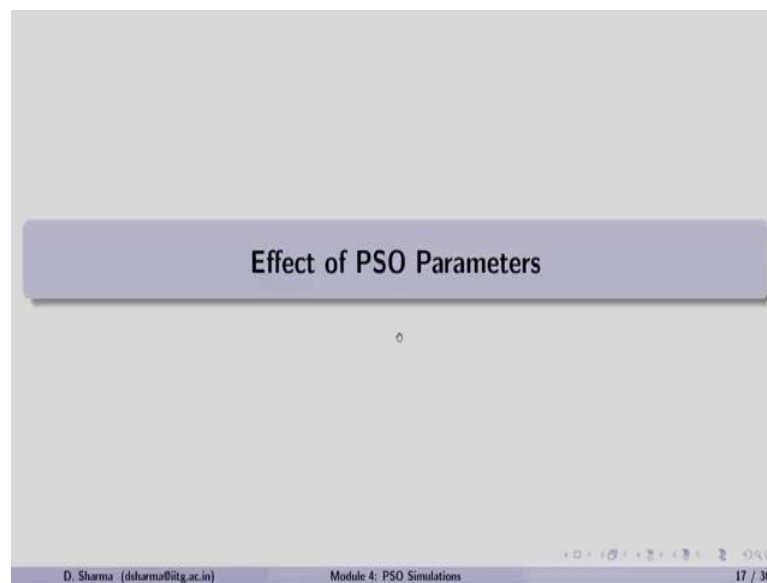
In this particular problem you can see the blue dots here, these blue dots these are the particle positions as and when the particle any of the particle is reached to the optimum solution, its local best as well as the global best of the swarm is improved. And once the global best reaches to the red dot which is the optima other particles are getting attracted towards the global optima. So, in this simulation we can see the solution, many solutions are converging towards the global optima.

(Refer Slide Time: 24:27)



Let us see the progress of the best fitness yeah, best fitness with respect to the number of generations here. As we can see close to 10, we are already we are already close to the optima somewhere just before 40 generation and or at 45 generation PSO is found an optimum solution for the given problem. So, one more time we can see that close to 45 number of a generation, there is a little improvement you can see here and that takes PSO to the global optimum solution for the Ackleys function.

(Refer Slide Time: 25:04)



Now, till now what we have gone through is the simulation of PSO and we know that for 2 variable problems PSO is quite efficient to solve the problem, but as and when we are increasing the number of variable for Rosenbrock and Rastrigin function. It allows or it indicates that, we have to increase the swarm size and allow PSO for more number of a generation to find the global optimum solution for the given problem.

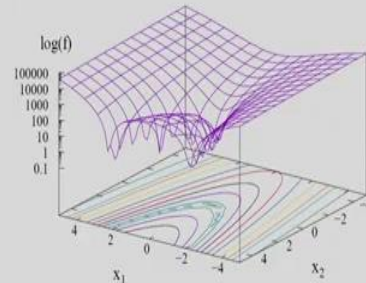
Now, as we know in the in the PSO, this velocity component has three parts and all these three parts have constants and those constants are to be decided by the users. So, such as we have to decide what should be the omega? What should be c_1 and a c_2 ? Now, we will understand what are those, what are the effects of these constant parameter if you take some different values.

(Refer Slide Time: 26:12)

Rosenbrock Function

Rosenbrock Function

Minimize $f(x_1, \dots, x_n) = \sum_{i=1}^n (100(x_{i+1} - x_i^2)^2 + (1 - x_i)^2)$,
 bounds $-5 \leq x_i \leq 5$ and $i = 1, \dots, n$.



- Optimal solution is $x^* = (1, \dots, 1)^T$ and $f(x) = 0$

So, let us begin our experiment simulation again on the Rosenbrock function. Since, it has a multi, it has many local optima's and one global optima.

$$\text{Minimize } f(x_1, \dots, x_n) = \sum_{i=1}^n (100(x_{i+1} - x_i^2)^2 + (1 - x_i)^2)$$

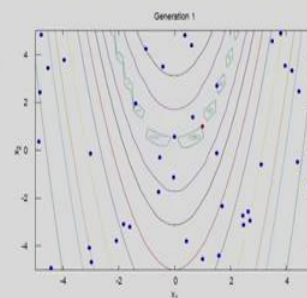
$$\text{bounds } -5 \leq x_i \leq 5 \text{ and } i = 1, \dots, n$$

(Refer Slide Time: 26:19)

Large w

PSO Parameters

- Number of variables: $n = 2$
- Swarm size: $N = 40$
- No. of generations: $T = 200$
- Velocity coefficients: $c_1 = c_2 = 1.5$
- Inertia equation $w = 5.0$.
- Initial velocity of each particle: $v^{(i)} = 0$



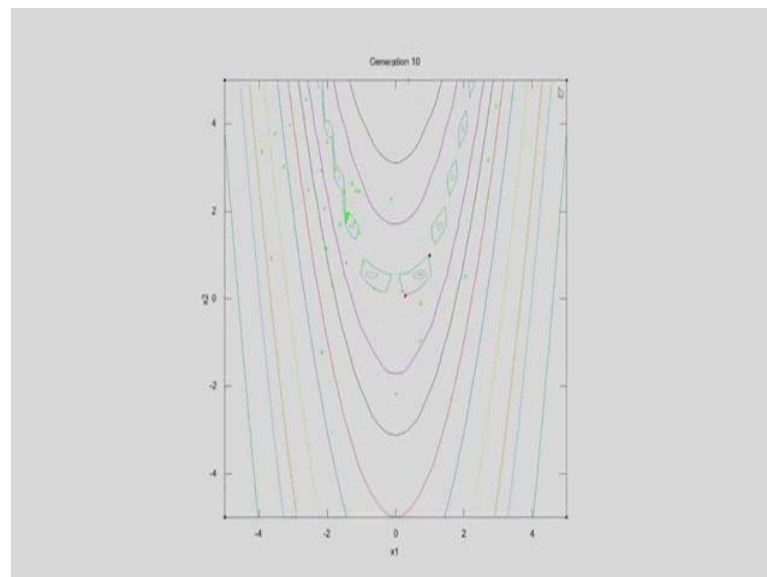
- Simulation [Link](#)
- Progress [Link](#)

- Velocities increase over time
- Swarm diverges
- Particles sometimes fail to change direction toward more promising regions

Now, in this particular effect we will take, we will do the simulation for a large value of ω as you can see on the top. For this particular simulation we kept number of variable 2, swarm size is 40, number of generation is 200, velocity components are coefficients are c_1 and c_2 which are 1.5.

Now, look at the inertia. So, although I have written inertia equation, but the constant we kept it ω constant and we took this value 5 which is a very large value and we kept our initial velocity is 0. Now, looking on the right hand side figure, these are the particles distributed in the initial swarm and let us see the simulations.

(Refer Slide Time: 27:12)

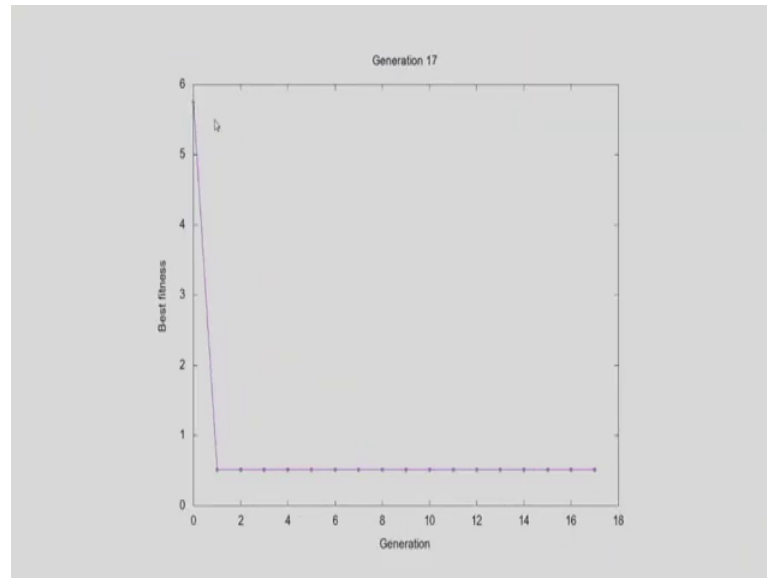


In this simulation there are different kinds of dots. So, the green dots represents the local best of each particle. This red dot represents the global, the position of the global best and the black dot represents the global optima of the given problem. Now, here you cannot see the blue points why; because if you look at the corners, all the blue points are settled at the corners.

So, these are the particle positions in every generation why it is happening so? It is, because the ω component is so large that the velocity component is taking so high and which takes the current position of the particle out of the bound and therefore, there is no improvement in the local best. As you can see the green dots are not moving as well as this red dot is also not moving which is the global best.

So, this means that if we keep omega value large, then the particles are going out of the bound and there is no improvement in the fitness of, best fitness of the PSO. Now, let us see the progress here, for the current example.

(Refer Slide Time: 28:31)



Now, it is started with the 6, as we know the velocity omega component is so high, then it says that the it is taking the particle position out of the bound. And therefore, there is no improvement in the fitness, because we are not performing any search using PSO with the large value of omega.

So, what we can conclude here is; that velocities of each particle increased over the time why; because omega is 5 thereafter and, because of that this swarm diverges. And because of this all the particles are going out of the bound and we are not able to update the local best and accordingly, the global best.

Moreover; as you can see at the last, the particles sometimes fail to change direction toward more promising region. It is only, because we are not performing search in x_1 and x_2 plane, because most of the time or all the time the solutions are going out of the bound.

(Refer Slide Time: 29:42)

Effect of $c_1 > c_2$

PSO Parameters

- Number of variables: $n = 2$
- Swarm size: $N = 40$
- No. of generations: $T = 200$
- Velocity coefficients: $c_1 = 2.5, c_2 = 0.5$
- $w = w_{max} - \frac{t}{T}(w_{max} - w_{min})$, where $w_{max} = 0.9$ and $w_{min} = 0.1$.
- Initial velocity of each particle: $v^{(i)} = 0$

Simulation [Link](#)

Progress [Link](#)

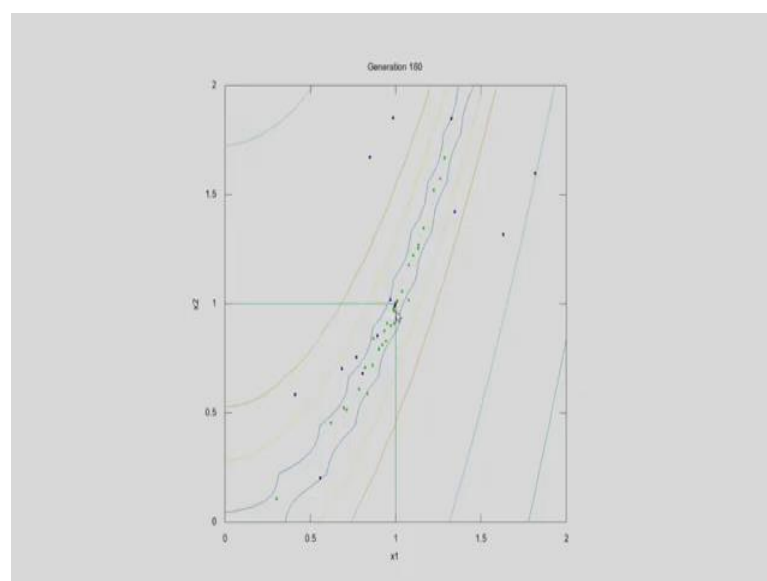
It can be useful for multi-modal optimization problems.

D. Sharma (dsharma@itg.ac.in) Module 4: PSO Simulations 20 / 36

Now, let us consider the another effect where we are keeping larger value of a c_1 as compared to c_2 . So, as you can see on the top we will see the effect of c_1 greater than c_2 . Again, to understand the simulation we have kept n equals to 2 means 2 variable Rosenbrock function, swarm size is again 40, number of generation is 200.

Now, look at the c_1 value. Now, c_1 is 2.5 and c_2 is 0.5. So, c_1 is relatively larger than the c_2 value, ω is changing with our equation and the initial velocity of the particle is kept 0. On the right hand side, we can see the x_1 and x_2 plane. In this case the particles are randomly distributed as you can see. Now, let us see the simulations.

(Refer Slide Time: 30:39)

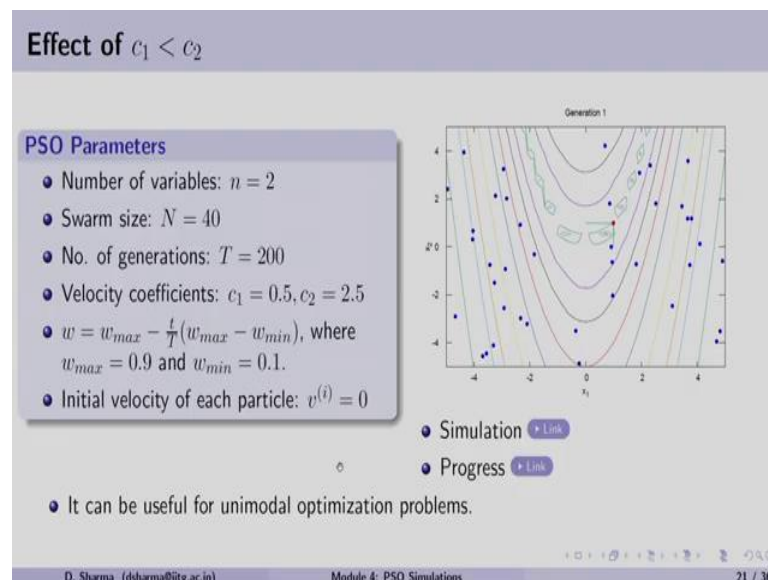


Now, here as you can see black and the blue and the green dots; the blue dots represent the particle positions, green dots represent the local best, red dot represents the global best which is now finally, converged to the optimum solution. Now, as you can see as and when the particle is updated the local best is updated, the global best is also updated and finally, this global best is converged to the optimum solution.

If we look at the progress here, it is started close to the 5 best fitness and even in 7 number of a generation we got the optimum solution for the given problem when we are keeping c_1 greater than c_2 . Although, the effect of c_1 and a c_2 is not prominent here, but as we know that if we keep c_1 value large as you can see here, this means that we are giving more emphasis to the cognitive part as compared to the social part.

Now, from the simulations we can say that it can be useful for multi modal optimization problem. It is only, because since the each particle we are giving emphasis to the cognitive part. So, in this case a particle which may reach to the other optimum solution in a multi modal problem then it is going to preserve its position, because we are giving more emphasis to the cognitive part. So, this kind of small change may help PSO to solve multi modal problems.

(Refer Slide Time: 32:23)

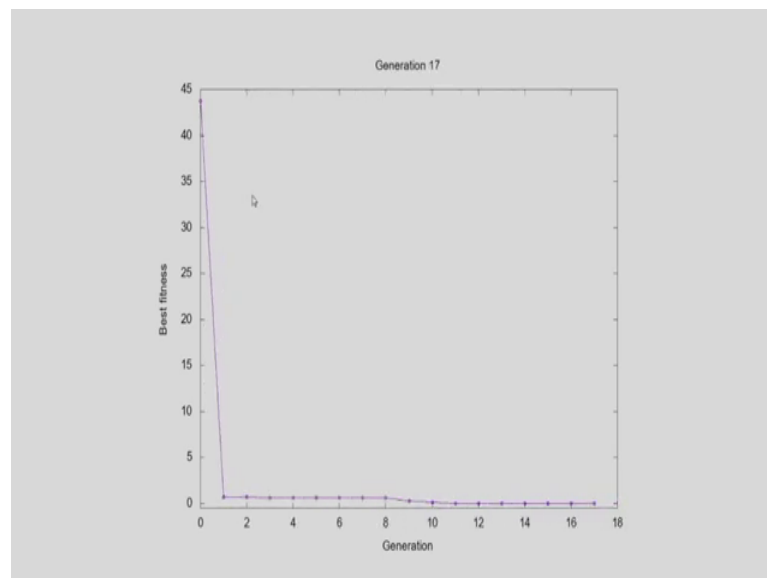


Now, let us come to the another simulation where c_1 is smaller than c_2 , the number of variable is 2, swarm size is 40, number of generation is 200. Now, look at the value of c_1 and a c_2 , c_1 is 0.5 and c_2 is 2.5.

So, in this case c_2 is taken more than c_1 and we are taking the same inertia equation and initial velocity of the particle is 0. Now, on the right hand side we can see that the particles are randomly distributed in x_1 and x_2 plane and let us see how PSO solve this particular problem when c_2 is greater than c_1 . Meaning; we are giving more emphasis to the social part as compared to the cognitive part.

Now, again the blue dots are blue dots represent the position of the particle, green dots represent the local best of each particle, and the red dot represents the global best. As you can see that as soon as the local best is updated of each particle and the global best is also updating and taking the solution close to the optimum solution here. So, in this case see the red dot, it is converging to the optimum solution here.

(Refer Slide Time: 33:59)

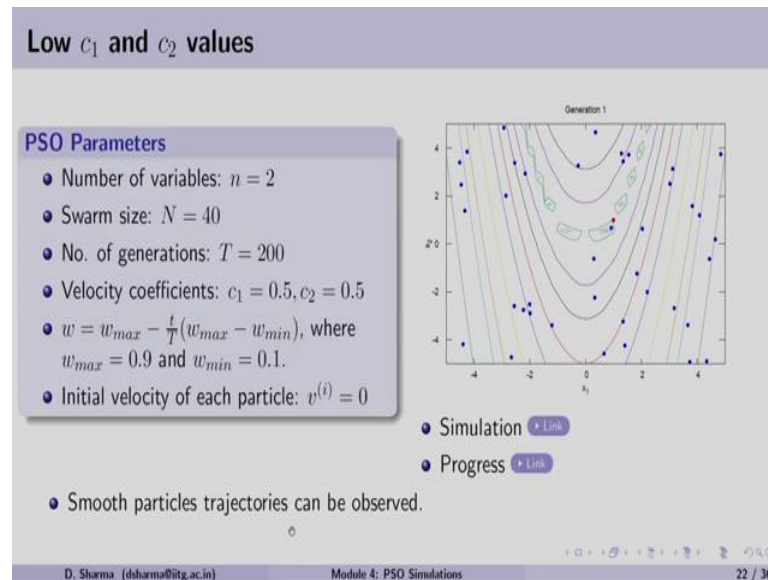


Now, look at the progress now in this particular case the best fitness is started from 45 and drastically it reduces its fitness close to 10 and thereafter since, we have already reached to the global optima. So, that is why we can see the straight line. So, in 10 number of a generation we have reached it.

It means that, if we compare this simulation with the previous simulation what we found that since we have increased c_2 value. So, we are giving more emphasis to the social part where the particle will be attracted more towards the global best. And since one of the particle has improved and the global best is reaches, close to the optimum solution rest of the solutions are also converging towards the optimal solution.

Such kind of setting, when we are keeping c_2 greater than c_1 that can be useful for unimodal optimization problem. Because as and when any optimum solution is found, PSO will take all the particles towards this global optimal solution.

(Refer Slide Time: 35:12)

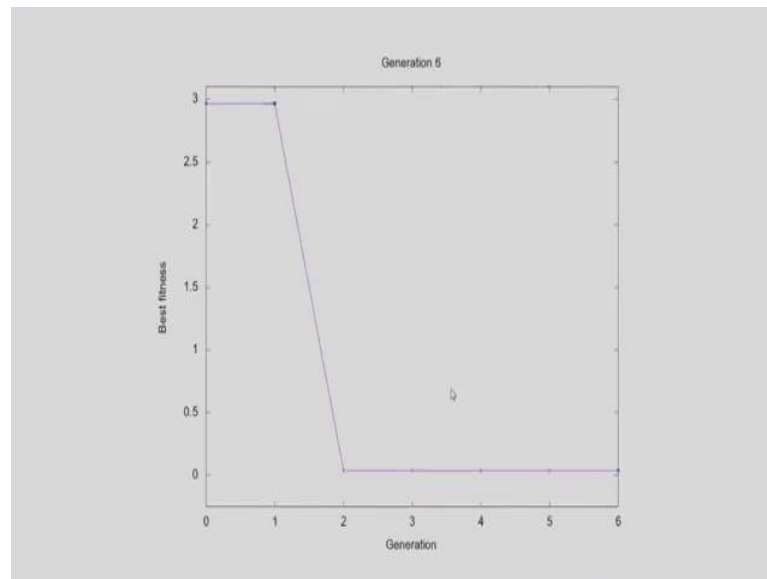


Now, let us take the another effect where c_1 and c_2 values are kept low. So, in this case n is equals to 2, number of swarm the swarm size is 40, number of generation is 200, look at the c_1 and c_2 value which are 0.5 and 0.5 which is relatively small values, omega equation or the inertia equation remains the same, initial velocity we kept it 0. On the right hand side, we can see the particles are distributed in x_1 and x_2 plane randomly.

Now, let us see the simulation here. Now, again green blue dots represent; so, the blue dots represent the position of the particle, green dots represent the local best, and red dot represents the global best.

So, we can see that when we are keeping small. So, we have these particles or the green dots are converging close to the optimum solution and therefore, the global optimum or the global best of the swarm has converged to the optimum solution.

(Refer Slide Time: 36:28)



Look at the progress now. In this case we started with the 3, the best fitness and in early generation we are already close to the optima. So, close to say 14 number of a generation, this particular setting has taken PSO to the global optimum solution. So, as you can see once we reached it so we have a straight line along the, which is parallel to the x axis.

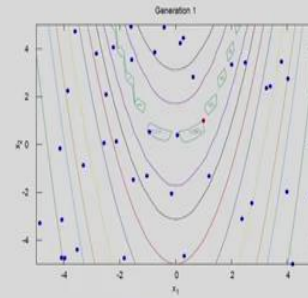
Now, in this case when we are keeping low value of a c_1 and a c_2 that allows smooth particle trajectories, as we also observe that the particles are not moving abruptly, they are moving slowly towards the optimum solution. And since, we have given low value to both to the cognitive part and the social part. These particles are in a cooperative way moving towards the optimum solution and reach to the global optimum solution for the Rosenbrock function.

(Refer Slide Time: 37:31)

Large c_1 and c_2 values

PSO Parameters

- Number of variables: $n = 2$
- Swarm size: $N = 40$
- No. of generations: $T = 200$
- Velocity coefficients: $c_1 = 5.0, c_2 = 5.0$
- $w = w_{max} - \frac{t}{T}(w_{max} - w_{min})$, where $w_{max} = 0.9$ and $w_{min} = 0.1$.
- Initial velocity of each particle: $v^{(i)} = 0$



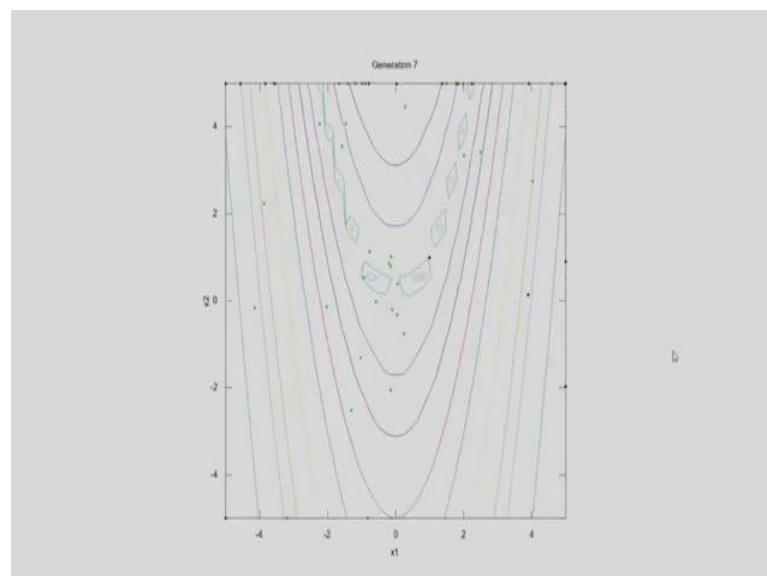
• Simulation [Link](#)

• Progress [Link](#)

- It supports large acceleration to particles but with abrupt movement.

Let us come to the large value of a c_1 and a c_2 value. So, in this case our simulation setup remains the same, where number of variable is 2, swarm size is 40, number of generation is 200, coefficient of c_1 and a c_2 so we are keeping a very large value say 5 and a 5 for c_1 and a c_2 , omega is we are using the same equation and our initial velocity of particle is 0. On the right hand side figure, we can see that the solutions are randomly distributed in x_1 and x_2 plane. Let us see how PSO will solve this problem.

(Refer Slide Time: 38:15)

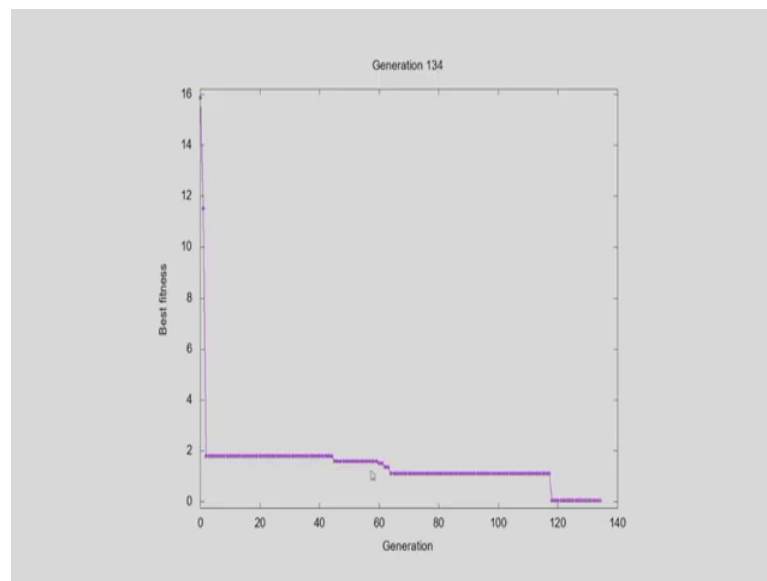


As you can see there are drastic changes in the velocity component, because of the c_1 and a c_2 . And because of that the particles are also moving abruptly and therefore, there are

only few particles which are moving inside it, trying to improve the local best as well as the global best.

Currently as you can see the global best is randomly moving from one position to another it is only, because the function has so many local optima's. So, in this case let us see where this red dot is converged at 200. So, it is already close to the black point in 200 generation.

(Refer Slide Time: 39:02)



Let us run, let us see the progress of the best fitness here. In this case we started with 16 and then drastically the fitness is improved to 2 and it is keep on improving. Now, you can see that the improvement is not continuous it is only, because the c_1 and c_2 values are large that are making a large velocity component here. And, because of the large velocity component the position is also changed drastically.

Many of the particles are going out of the bound and very few particles are searching inside the region and that is why it takes many generations to converge to the optimum solution. Now, from this particular simulation as we can understand that if we are keeping c_1 and c_2 value large, it supports large acceleration to the particle, but abrupt movement.

And that can be evident from our simulation as well that the large acceleration may help to move the particle from one position to the another one, but that can take our particle out of the bound on either x_1 or x_2 or both.

(Refer Slide Time: 40:18)

The slide is titled "Issues" and contains the following content:

- We can observe a potential dangerous property of PSO
 - ▶ when $x_i^{(t)} = p_{(i,lb)}^{(t)} = p_{gb}^{(t)}$
 - ▶ The velocity update depends only on $wv_i^{(t)}$
 - ▶ If this condition persists for a number of generations, $wv_i^{(t)} \rightarrow 0$
- There are certain potential problems with PSO
 - ▶ **Infeasible solutions:** Particles leave the search boundaries frequently.
 - ▶ **Wasted search effort:** Particles are pulled back into the feasible space or on the boundary.
 - ▶ **Incorrect swarm diversity calculations:** As particles move outside of the search boundaries, diversity increases.

At the bottom of the slide, there is a footer with the text "D. Sharma (dsharma@itg.ac.in)", "Module 4: PSO Simulations", and "24 / 36".

Now, let us discuss the issues with PSO. In this one the first potential dangerous property of a PSO is when as you can see in the slide that the current position of the particle is the local best as well as it is the global best.

So, in this case the velocity component the cognitive part as well as the social part are 0. So, the velocity update will depend only on the multiple of omega v_i and with the number of a generation what we will realize that omega v_i will be tending towards 0. So, overall the if this situation happen then the particle will not be moving and exploring the search space to find the optimum solution for the given problem.

The second problem or the issue is that we have so many infeasible solution. Meaning that; the particles leave the search boundaries very frequently. This n once they go out of the bound what we do is; we pulled back them into the feasible search space or on the boundary. So, this means that the effort which we are making to update the velocity as well as the local as well as the current position of the particle all these calculation are getting wasted.

Third is this kind of particles when they are going out of the bound we thought that it is the, it is the diversity of the swarm, but when we are keeping them on the boundary or in any or in the feasible search space basically that indicates the wrong impression about the diversity in the swarm.

So, these are the two potential issues with PSO. After understanding the behavior of PSO on different problems as well as looking at the simulations for different values of ω c_1 and c_2 . Now, let us understand how we can implement PSO. So, now, we will discuss the algorithmic implementation of particle swarm optimization.

As we have understood this particular algorithmic implementation is independent of any programming language. Once we understood it, we can use any of the programming language such as C, C plus plus, Java, MATLAB or Python. So, any of the language we can use it and we can make our own PSO code for solving the problems. So, let us start with the algorithmic implementation.

(Refer Slide Time: 43:12)

Algorithmic Implementation of PSO

Algorithm 1 Generalized Framework

1. Solution representation % Genetics
2. Input: $t := 1$ (Generation counter), Maximum allowed generation = T , etc.
3. Initialize random swarm $(P(t))$; % Swarm
4. Evaluate $(P(t))$; % Evaluate objective, constraints and assign fitness
5. while $t \leq T$ do
6. Update $p_{(i,lb)}$ of each particle (i) and find $p_{gb}^{(t)}$; % New step
7. for $(i = 1; i \leq N, i++)$ do % For each particle i
8. Update velocity $(v_i^{(t+1)})$; % Variation
9. Update position $(x_i^{(t+1)})$;
10. Evaluate $(x_i^{(t+1)})$;
11. end for
12. $t := t + 1$;
13. end while

D. Sharma (dsharma@iitg.ac.in) Module 4: PSO Simulations 26 / 36

So, as we know that this is the generalized framework which we discussed with the PSO. In this case we start with the solution representation in the step number 1 which is in this as PSO is used for real parameters so the variables are real in number. In step 2 we give certain input to the algorithm.

So, as of now we have included two, but there are more input parameters which are needed to run the PSO. In step 3 we initialize this one as we can observe from our simulations. The particles are randomly distributed in the variable space.

As of now we are currently saying that it is P . So, the swarm is represented by P . Once the swarm is generated we have to calculate this particular swarm here. So, when we are

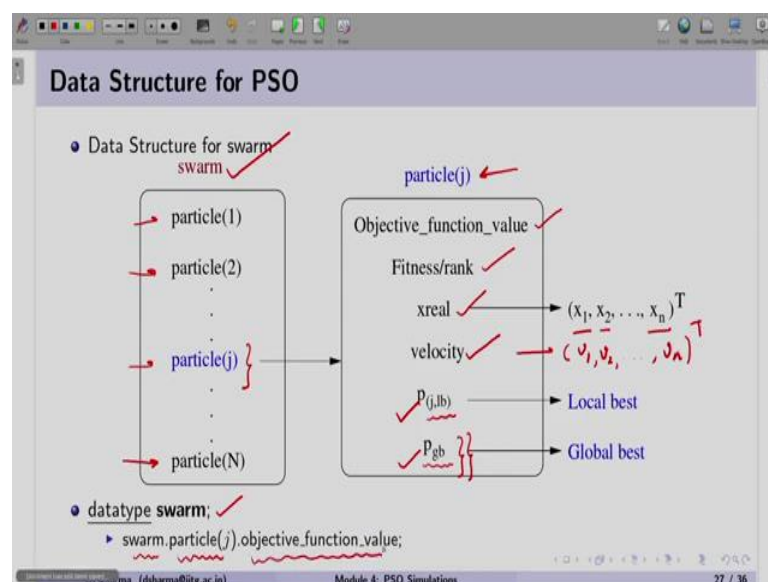
evaluating it this means that we will be calculating the objective function constraint and finally, we have to assign the fitness.

Now, from in the step 5 we are in the standard loop of number of generation and step 6 which is new as we discussed earlier. In this particular step, we have to update the local best of each particle as well as we have to update the global best.

So, that is why we are writing this as our new step. Now, in this PSO we do this update with respect to the each variable, so that is why in step number 7 you can see we have a for loop starting from the first variable to the last variable it should not be variable. We are starting from first particle to the last particle and then first we will be updating the velocity component using the equation which we have gone through earlier and we are updating the position.

Since, the position is updated in step 10 we have to evaluate, basically we have to evaluate the new position and assign the fitness. Once it is done we will increase the counter of number of generation by 1 and we keep on moving in this particular loop till the termination condition is not satisfied. So, as you can see that there are so many functions which are involved here. So, let us understand them one by one.

(Refer Slide Time: 45:55)



We will start with the data structure of PSO. Similar to our earlier implementation as you can see we have a swarm here and this particular swarm consist of various particles. So,

the size of the swarm is N . So, therefore, we can see there are N particles. So, let us take a typical particle say j and in this particular particle j what are the values we can save it? That is the objective function value, the fitness value, we have to keep the value of the of each variable.

Now, you can see x_1 , x_2 and x_n . So, we are saving the values as a column vector. It is for representation purpose now, we also have a velocity component. So, this data structure should have the option of velocity. So, velocity means that it is going to have say v_1 , v_2 and similarly, v_n which is the same size as the number of variable.

We should also save the local best and the global best. So, the local best of the particle is needed why; because we are going to use it in the velocity update. Similarly, the global best of the swarm is also should also be stored with the particle, because we are going to use with the in the velocity update.

In some implementation this global best is saved explicitly, but we can save in our data structure just for the so that we can use it easily, but our memory size will increase. So, at the bottom as you can see that the data type is swarm and if we have to say store the value or update. So, we have a swarm then we will be calling as a particle j and then this objective function value will be storing the function value.

(Refer Slide Time: 47:55)

Input to PSO

Algorithm 2 Input

- 1: Swarm size: N
- 2: Number of generations: T
- 3: Number of real variables: n
- 4: **for** ($j = 1; j \leq n; j++$) **do**
- 5: Lower and upper bounds on x_j that are $x_j^{(L)}$ and $x_j^{(U)}$
- 6: **end for**
- 7: Other parameters: w, c_1, c_2

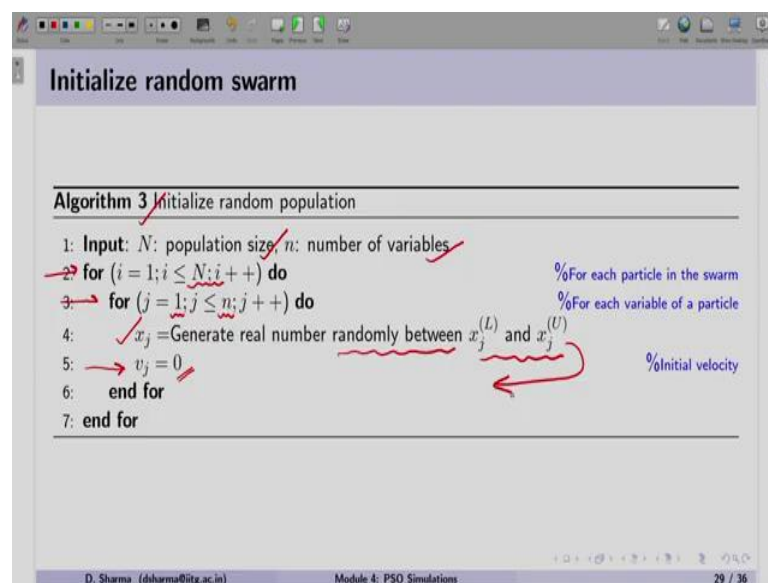
%For each variable

D. Sharma (dsharma@iitg.ac.in) Module 4: PSO Simulations 28 / 36

Once we when we have to start PSO, we have to gave certain input to our algorithm. So, in this case what we need is the swarm size, the maximum number of a generation the number of variable. As soon as we know how many variables are there then we have to decide the lower and the upper bound as you can see in the step number 5.

And there are some other parameters as you can see on the step number 7 that we have to fix omega c 1 and c 2. So, in our simulations we have taken omega as an equation. So, in that case we have to save say omega max and omega min for PSO.

(Refer Slide Time: 48:43)



Now, let us start with the initializing the random swarm. So, in this case as you can see in Algorithm number 3, we need input as what is the population's population size or a swarm size and what is the number of variable. So, the first loop will start for all the particles in the swarm, as you can see for the loop i in the step number 2 and step number 3 is the loop which is on the number of variable.

So, for every particle we will be running a loop for number of variable our main point is we have to generate the x_j which is the current position of the particle, that we generate randomly between the lower and the upper limit of the of this particular variable. Step number 5 shows that we have to keep the velocity component 0.

So, as we know that in our implementation we have taken 0, but if we want to have some random values, we can exactly copy the same step here, if we do not want to start with a 0

value. So, that way we can start with the random values of each variable as well as the random values of the velocity or we can keep it 0.

(Refer Slide Time: 50:05)

Evaluate Particle

Algorithm 4 Evaluate Population

1: **Input:** particle (j) ✓

2: **Evaluate** $f(x_j)$ ✓ %Extract $x_j = (x_1, \dots, x_n)^T$ from the data structure of a particle(j)

- Assign fitness same as the function value
- $\text{swarm.particle}(j).\text{objective_function_value} = f(x_1, \dots, x_n);$
- $\text{swarm.particle}(j).\text{fitness} = \text{swarm.particle}(j).\text{objective_function_value};$

D. Sharma (dsharma@itg.ac.in) Module 4: PSO Simulations 30 / 36

Now, let us come to the evaluate particle which is the simplest in all the functions where the input to the Algorithm 4 is particle j . So, this particular x_j as we know, this x_j includes a column vector of a variable. So, once we are including the value in the objective function we can calculate.

So, as of now since our problems are we have taken unconstrained problems. So, we are calculating only objective function here. Now, here our assumption is that the fitness is the same as the function value, so the fitness what we calculated in the step 2 in Algorithm 4 that will become the fitness here.

So, here you can see that we have a swarm, we have a particle j and then the objective function value is updated with the help of $x_1 \times 2 \times 3$ values. Once we calculated it then we can update this fitness at the last line with the objective function value here.

(Refer Slide Time: 51:10)

Local best update of particle (j)

Algorithm 5 Local best update of particle (j)

```
1: Input: particle ( $j$ )
2: if ( $t == 1$ ) then                                     %Only for the first generation
3:    $p_{(j,lb)} = x_j$ ;
4: else
5:   if ( $f(x_j) < f(p_{(j,lb)})$ ) then                     %Update when the fitness of particle is better
6:      $p_{(j,lb)} = x_j$ ;
7:   end if
8: end if
```

D. Sharma (dsharma@iitg.ac.in) Module 4: PSO Simulations 31 / 36

Now, once we evaluate the swarm or each particle, we are in the standard loop of generation. Inside it our first task is to find out the local best of each particle. So, let us see how we can do it. In this Algorithm 5 this is made basically to update the local best of each particle. So, the input to this is the particle. So, every time a one particle is going here. Suppose, now here in this step number 2 you remember that t is our generation counter if t is equals to 1 this means it is the first generation.

So, remembering the hand calculation in the first generation the current position itself is the local best of the particle. This is exactly the same thing we have done in step number 3 where the local best of the particle j is the current position. If it is not the first generation then in step number 5 as you can see we have to compare the fitness of the current position of the particle and the fitness of the local best.

In this case if the fitness of the current position is better than the fitness of the local best, we will update the local best otherwise we will not change the local best. So, this is as per our definition of the local best we can create such kind of algorithmic implementation.

(Refer Slide Time: 52:45)

Global best of swarm

Algorithm 6 Global best of swarm

```
1: Input:  $P(t)$ : swarm,  $N$ : size of swarm
2:  $p_{gb} = p_{1,lb}$  %Initialization of the global best with the local best of the first particle
3: for ( $j = 2; j \leq N; j++$ ) do
4:   if ( $f(p_{j,lb}) < f(p_{gb})$ ) then %Update when the fitness of the local best of particle is better
5:      $p_{gb} = p_{j,lb}$ 
6:   end if
7: end for
```

D. Sharma (dsharma@iitg.ac.in) Module 4: PSO Simulations 32 / 36

Coming to the global best of the swarm; for the global best of the swarm we know all of the particles. So, that is why we are taking swarm as our input and the size of the swarm N is also needed.

So, what we are doing just for our simplification; at step number 1 we are assigning that the global best is equals to the local best of the first particle why; because in because this particular value of the global best the fitness value that based on that we will keep on updating this value.

So, let us see the step number 3, we run now here, it is important to note that we are running from the second particle to the last particle and thereafter in the step number 4 we are checking the condition whether the fitness of the local best of the particle j is smaller than the fitness of the global best. If yes, then the global best of the swarm is updated, if not we will not do it. So, at the end of this function we will get the global best of the swarm.

(Refer Slide Time: 54:01)

Velocity and Position Updates

Algorithm 7 Velocity and position updates of each particle (j)

- 1: Input: particle (j) and constant parameters
- 2: Update velocity of particle (j) using

$$v_j = wv_j + c_1r_1(p_{j,b} - x_j) + c_2r_2(p_{gb} - x_j)$$
- 3: Update position of particle (j) using

$$x_j = x_j + v_j$$

D. Sharma (dsharma@iitg.ac.in) Module 4: PSO Simulations 33 / 36

Now, coming to the velocity and the position update, as we know once we updated the local and a global best then only we can update our velocity. Looking at the Algorithm number 7 here so the input to this function is say particle j and we have some constant parameter as you remember we have ω , c_1 and c_2 .

So, the velocity is updated. As you can see this is the velocity component we are adding ωv_j , then we have this cognitive part, and then we have this social part. So, these three component for every particle is updated. So, since it is the simple operation. So, I am showing you in a equation form. Only it will involve one more for loop on the number of variable, because we are going variable wise and keep on updating the each component of velocity.

Now, once the velocity is updated as you can see in step number 3 the position is updated. So, this is the vector addition and this can be done simply. Again, in this particular step we need a for loop for a number of variable and each component of the position is added with the updated velocity to get the updated position. Since, it is simple so we are writing in a very direct form as a vector addition.

(Refer Slide Time: 55:30)

The image shows a presentation slide titled "Copy Particle". Below the title, it lists "Algorithm 8 Copy Particle" with seven numbered steps. The steps are: 1: Input: particle 1, particle 2; 2: Copy objective function value of particle 1 to particle 2; 3: Copy fitness/rank of particle 1 to particle 2; 4: Copy x_j of particle 1 to x_j of particle 2; 5: Copy v_j of particle 1 to v_j of particle 2; 6: Copy $p_{(1,lb)}$ of particle 1 to $p_{(2,lb)}$ of particle 2; 7: Copy p_{gb} of particle 1 to p_{gb} of particle 2. Below the list, there is a bullet point that says "Copy the complete data structure". The slide has a footer with the text "D. Sharma (dsharma@iitg.ac.in) Module 4: PSO Simulations 34 / 36".

Copy Particle

Algorithm 8 Copy Particle

- 1: **Input:** particle 1, particle 2
- 2: Copy objective function value of particle 1 to particle 2
- 3: Copy fitness/rank of particle 1 to particle 2
- 4: Copy x_j of particle 1 to x_j of particle 2
- 5: Copy v_j of particle 1 to v_j of particle 2
- 6: Copy $p_{(1,lb)}$ of particle 1 to $p_{(2,lb)}$ of particle 2
- 7: Copy p_{gb} of particle 1 to p_{gb} of particle 2

- Copy the complete data structure

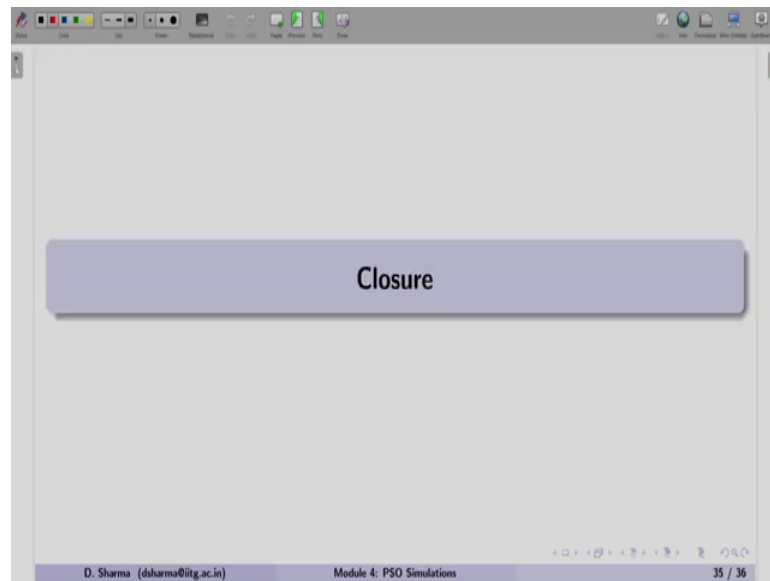
D. Sharma (dsharma@iitg.ac.in) Module 4: PSO Simulations 34 / 36

Now, at the last we have to copy. Now, this copy is required, because when we have identified for example, we have identified that our local best is to be updated or our global best has to be updated. So, in this case what we are doing is we are copying. So, when we copy our main idea is we have to copy the complete data structure that we have made it at the beginning.

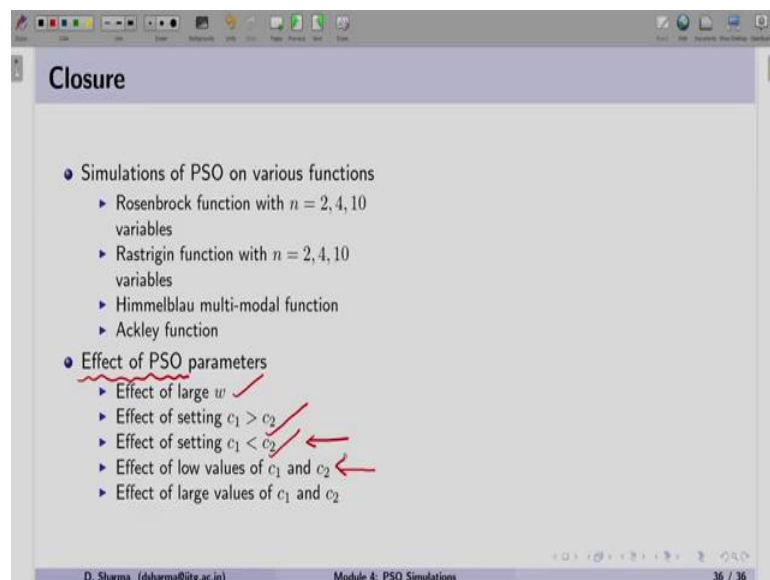
So, as you can see in the Algorithm 8 the input to the copy is particle 1 and a particle 2 here, we are assuming that the data from particle 1 is copied to the particle 2. So, step two says that let us copy the objective function, then we have to copy the fitness from particle 1 to particle 2, then we have to copy the full vector of particle 1 to particle 2. We have to copy the velocity as well, we have to copy the local best, we have to copy the global best.

So, as per our data structures as you can see at the bottom, we have to copy the complete data structure. This is important, because when we are updating a solution this copy will be needed. So, simple function can be made it to copy the data from particle 1 to particle 2.

(Refer Slide Time: 56:58)



(Refer Slide Time: 57:02)



So, now we have come to the closure of this session. So, in this session we started our discussion on the simulation of a PSO. Here, we have solved four problems. As you can see the Rosenbrock function is solved using PSO in which we have taken different set of number of variable.

So, the simulations were performed for 2 variables, 4 variables and 10 variables. Similarly, for the Rastrigin function as well we did perform the simulation of PSO for 2, 4, and 10 number of variables.

Himmelblau function is a multi modal function and Ackley was a difficult problem or a function to solve, because it has so many local optima's. So, from this simulation we found that PSO is able to solve all the problems having two number of variables, but when we have to increase the number of variable. In this case we have to increase the population size or the swarm size as well as the number of generation and then PSO can able to find the global optimum for the problem.

Once we have performed the simulation then we talked about or we have seen the effect of PSO parameters. So, we started with the large value of omega. As we remember if we keep a very large value of omega then the velocity component is going out of the bound which has impact on the position and that is why the particles are also going out of the bound. So, in that case we were not getting any particles within the x_1 and x_2 range that is updating the local best as well as the global best.

The second effect we have seen with respect to c_1 greater than c_2 or c_2 greater than c_1 . In that case what we found is that when we are keeping c_1 more than c_2 then we are emphasizing on the cognitive part. Meaning that; in this case the local best is emphasized more as compared to the global best.

So, the particles are free to move and after few number of a generation PSO was able to find the optimum solution, but in the second case when c_2 was more than c_1 we are giving more emphasis to the social part. This means that particles will be attracted towards the global best. So, in this scenario such value of c_1 and a c_2 can be useful for say multi modal or unimodal kind of optimization problem.

Then we have seen the effect of low value of c_1 and a c_2 and we have seen that there were smooth trajectories of the velocity. And that makes the position or the current position of the particle inside the bound, slowly the local best were updating and finally, we get the optimum solution in few generations.

In the last effect we have taken large value of a c_1 and a c_2 . As we understood that the large value of a c_1 and a c_2 taking the velocity component out of the bound, eventually the position of the particle is also going out of the bound and that is why the algorithm took more number of simulations to converge. So, the acceleration is fast, but the abrupt changes making our PSO little slower for the Rosenbrock function. At the last we discussed about the implementation.

We started discussing the data structure of implementing the PSO then we talk about what could be the important input to the PSO. How we can generate the random number, random initial swarm, then the fitness evaluation. We have a dedicated function for the local update, local position or the local best of each particle, we also have update on the global best global best of the swarm.

Once we find it out, with a simple function we can update the velocity of each particle and similarly the position and finally, we have understood the copy. So, that copy was essential to copy the data from one particle to another. So, with these remarks and simulations and understanding the behavior of PSO, I conclude this session.

Thank you.