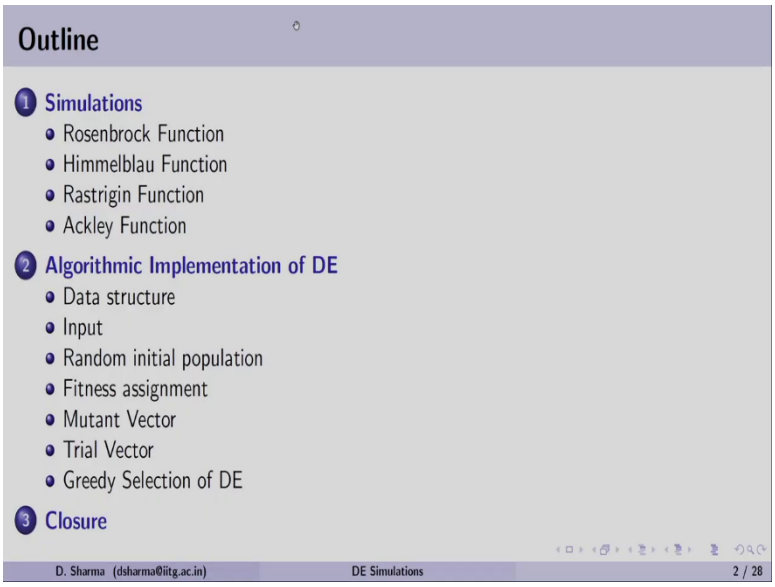
Evolutionary Computation for Single and Multi-Objective Optimization

Dr. Deepak Sharma
Department of Mechanical Engineering
Indian Institute of Technology, Guwahati

Lecture – 11
Simulations and Algorithmic Implementation of Differential Evolution

Welcome to this session. In this session, we will be looking at the Simulation of differential evolution followed by the Algorithmic Implementation of DE.

(Refer Slide Time: 00:50)



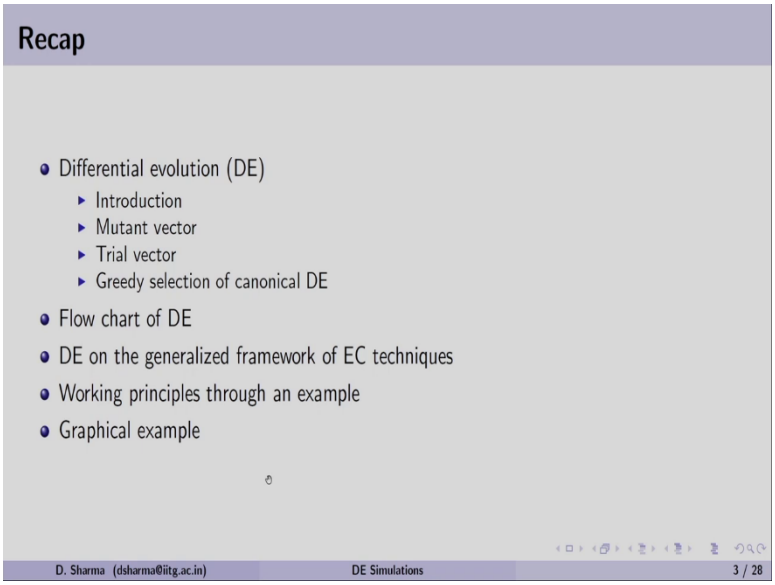
So, the content of this session includes that we will be showing simulation on four mathematical problems that include; Rosenbrock Function, Rastrigin Function, Himmelblau Function and Ackley Function. So, we will see how differential evolution is going to solve such kind of a problem. In the second part, we will be going through the algorithmic implementation as we can see here.

We will discuss the data structure first followed by the various inputs required by differential evolution. Thereafter we will generate initial population, then fitness assignment. As we know that we are going to create two types of vectors called mutant vector and trial vector for each target vector.

So, we will see how algorithmic representation using that, we can create these two vectors. Thereafter we will show the greedy selection of DE in the algorithmic form and at the end

we will conclude this session. So, before we start our simulation, let us recap our previous session.

(Refer Slide Time: 02:04)



We have gone through the differential evolution, the introduction about the algorithm and we found that the idea of differential evolution is borrowed from evolutionary computation technique and simplex search method. See evolutionary computation techniques uses multiple solutions and therefore, differential evolution is developed using multiple vectors. So, those vectors are improved generation by generations; they are compared and finally, we preserve the good one.

Similarly, in simplex search method as we use vector operations to search the to search the variable space. Similar to the same concept in differential evolution, we also use vector operations to generate new kinds of vectors. As we know differential evolution includes three types of vectors. So, the solution is represented or it is called as target vector and for each target vector we generate mutant vector.

And the mutant vector is the difference between the two vector multiplied by the scaling factor f and that is added to the third vector. All these vectors, we generally pick randomly and using this mutant vector and the target vector, we create trial vector. So, the mutant vector operation is referred as a mutation and the trial vector is generated using crossover.

So, as you might have realized that the trial vector is similar to of multi point crossover where we are exchanging the values of the decision variable between the mutant vector and the target vector. And thereafter we discuss the greedy selection of a DE that was proposed in the original paper of DE; differential evolution. Thereafter this DE differential evolution, we discussed with the help of a flowchart.

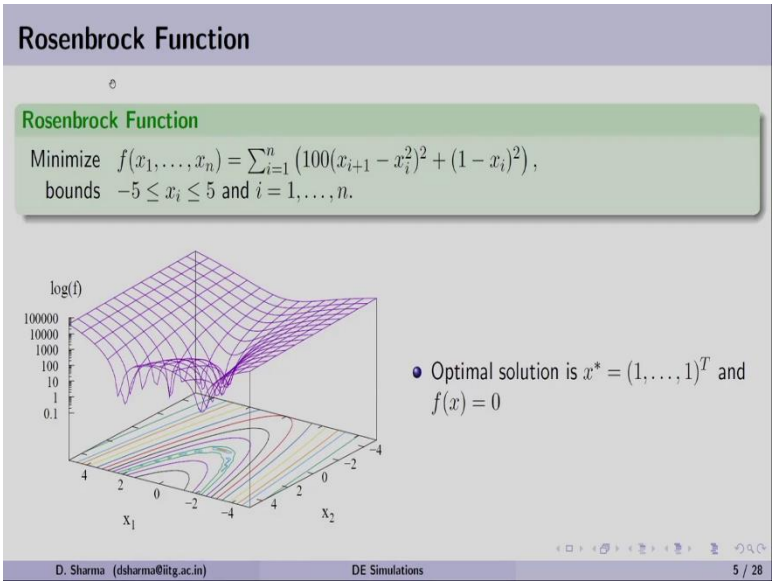
So, in that flowchart which was similar to the; flowchart as we are discussing for other evolutionary computing techniques. In this case, inside the loop or the decision loop for the number of generation, we also have another decision loop for the number of vectors in the population. So, for every vector or a target vector, we generate mutant vector, then we create a trial vector and finally, we select the best.

Similarly, since we are putting or we are making we are we are converting all these evolutionary computing technique in a generalized framework, differential evolution was also fitted into that framework. So, it was easy because it consists of upper loop for the generation and the lower loop for the number of vectors in the population. And then we understood the working principle of differential evolution using the Rosenbrock function.

So, we perform the hand calculation for one generation in which we generated the initial population followed by assigning the fitness. So, since in this particular case, we use the function value as a fitness so, we kept them same. Thereafter we generated the mutant vector followed by the trial vector and finally, we get the new mutant vectors or we get the new target vector for the next generation.

So, the whole hand calculation suggested us that generation by generation, the vectors in differential evolution will improve and finally, it will converge to the optimum solution. For the same example, we have shown the graphical illustration of Rosenbrock function and they are we understood how these vectors are generated and moving from one generation to the another generation. Now, let us begin the simulations here. Now the simulation we are starting with a problem called Rosenbrock function.

(Refer Slide Time: 06:41)



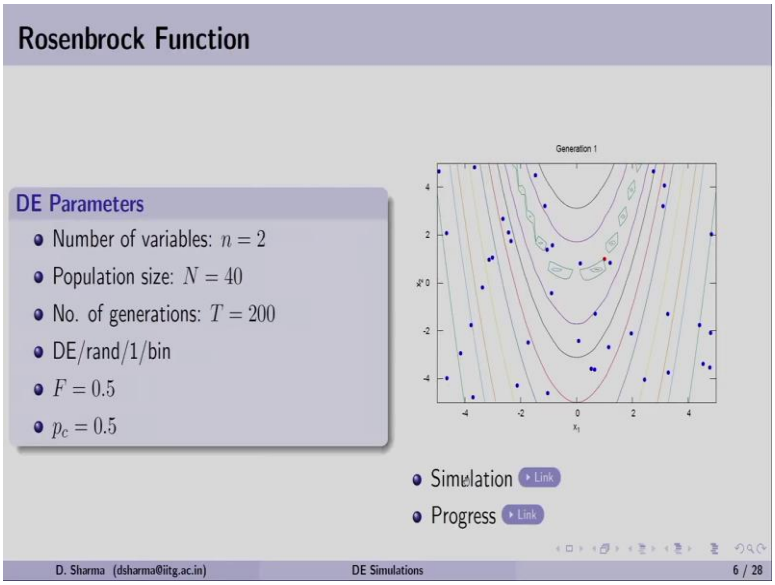
In this Rosenbrock function as you can see here; with this Rosenbrock function is written for n number of variables. So, this means that it is an a scaling function. We want to minimize the Rosenbrock function and as you can see the equation on the right hand side the variable bound so, for each variable should lie between minus 5 to plus 5. For a two variable case as you can see a we have a x 1 and x 2 plane and we have a third axis that is logarithmic of the function.

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

$$bounds\ -5\le x_i\le 5\ and\ i=1,\ldots,n.$$

Since we are since we have solved this particular problem in our previous session, we can see from the surface as well as from the contours on the x 1 and x 2 plane that this particular problem has lot of local optimum solutions. However on the right hand side, we can see the global optimum solution for this particular problem is lying at 1, 1, 1, 1. Meaning that all decision variable should have a value 1. The function value at that particular point is 0.

(Refer Slide Time: 07:58)



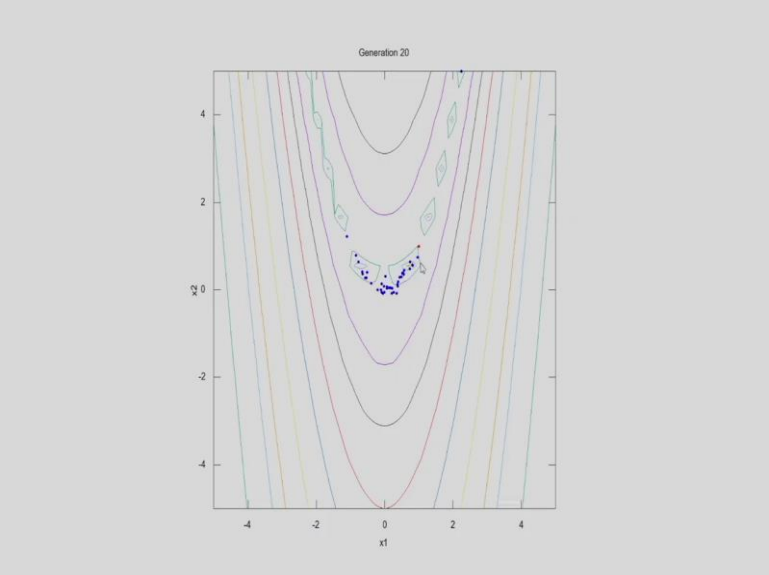
So, let us see our first simulation here. In this case the parameters for the DE, we have consider 2 number of variables. So, here we are choosing the number of variable for Rosenbrock function as 2, the population size is kept small that is 40, the number of generation recapped 200.

The variant of DE is considered as DE r a n d rand 1 and bin. So, as we understood because this particular variant we have discussed. So, they the same variant we are going to use it for solving the problem. The scaling factor F is 0.5 and the crossover rate is 0.5.

Let us look on the right hand side. Now the right hand side on generation 1, this figure shows the blue points which are the randomly generated points in the in the initial population and the red dot is the optimum solution for the given problem.

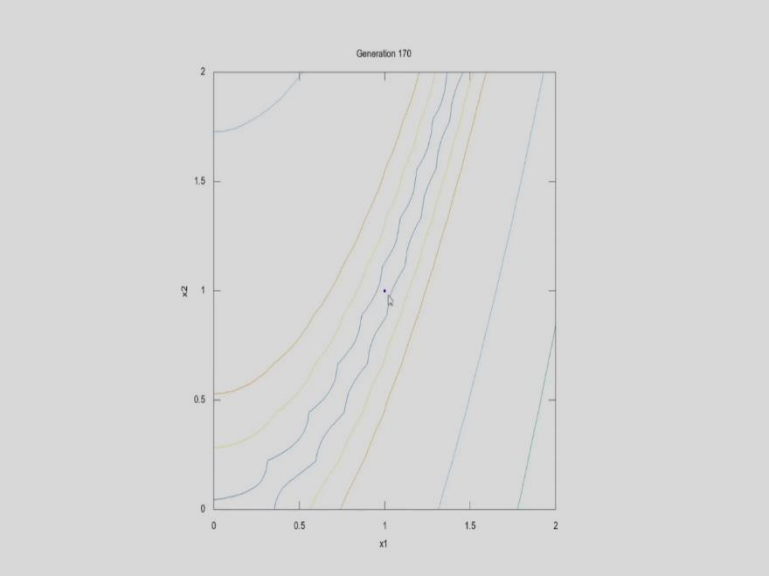
As you can see since evolutionary computation technique or DE algorithm uses multiple solutions and when we are generating random points in a x 1 and x 2 plane, some points can be close to the optimum as you can see in this particular figure. Now, let us see if we run this DE how it is going to solve our problem.

(Refer Slide Time: 09:34)



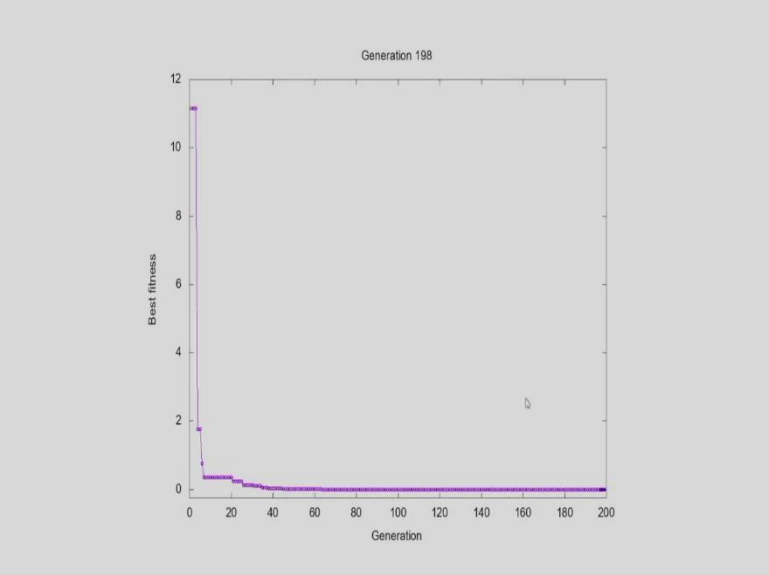
Now looking at this particular simulation here, all the blue dots are the target vectors in every generation and close to say 90 generations all the solutions have already converged.

(Refer Slide Time: 09:41)



So, one more time you can see the solutions are keep on moving towards the local as well as global optimum solutions and after some time, all the vectors in differential evolution converge to the optimum solution.

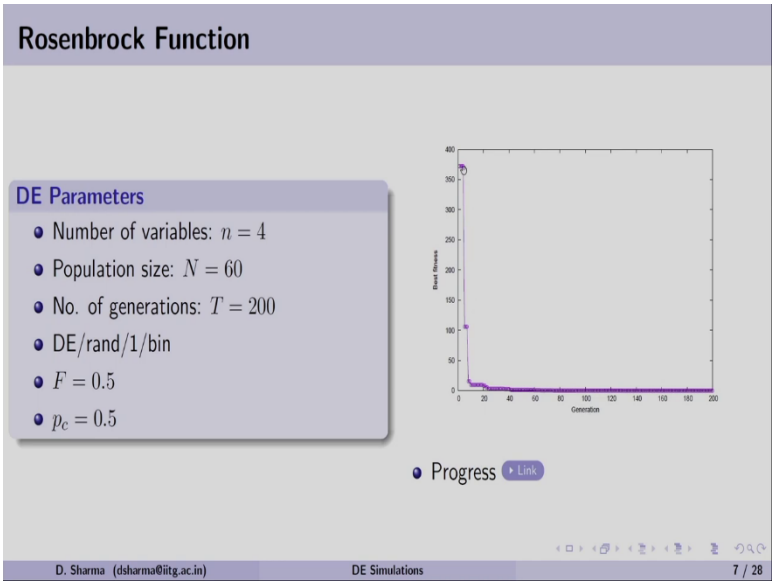
(Refer Slide Time: 10:12)



Now, we will see the progress. In this particular progress, we will see the generation versus fitness. So, the best fitness we get it in every generation is plotted here. As you can see the simulation is started with the best fitness around say 11 and quickly it has improved the solutions and close to 30 generations, we are already close to the optimum.

So, let us look at one look one more time. So, as we can see that close to 35 number for generation, differential evolution was able to find the optimum solution for the given problem and thereafter we have this flat curve.

(Refer Slide Time: 10:58)

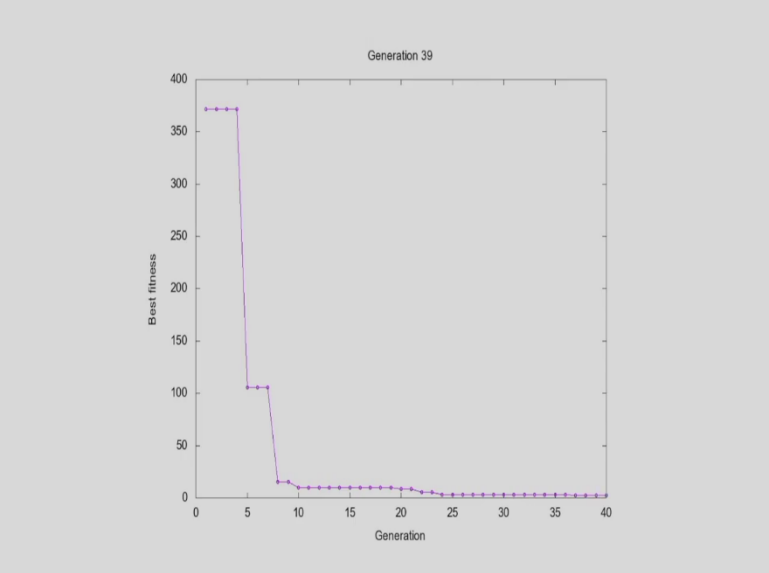


Since two-variable problem is relatively easy, now let us see what is the performance of DE on number of variable if we take number of variable as 4. Now we have kept population size 60 which is little more than the previous case.

Number of a generation is kept 200, same variant of DE is used here, scaling factor is again 0.5 and the crossover rate is 0.5. Now, as we know that the problem which we are solving is 4 variables. So, we cannot show the simulation in a variable space. So, here we will be looking at the progress.

Now if we see the figure on the right hand side, the best fitness in every generation. So, here you can see that more than 350 or close to 370, the best solution was there and then in very few generations say after 20 generation, it is already close to the minimum point. Let us see how the best solution is approaching towards the optimum solution.

(Refer Slide Time: 12:07)

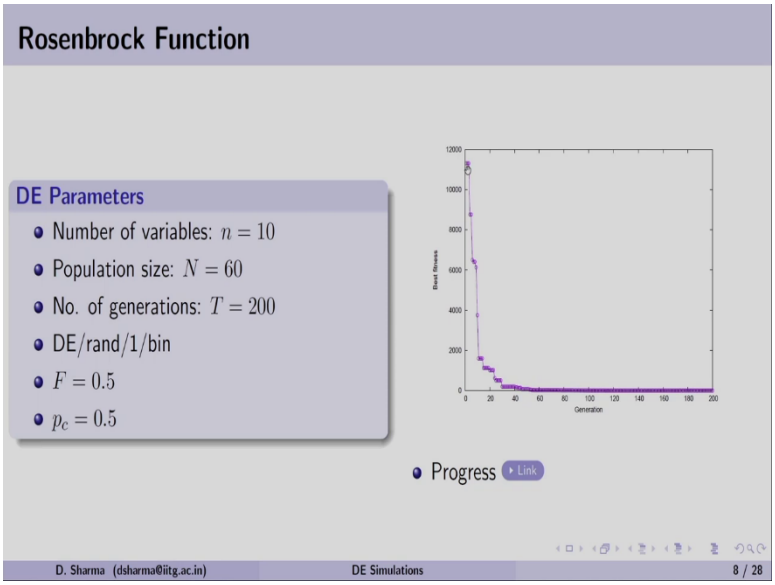


So, as you can see there is a drastic improvement in the fitness and if we look at the in the zoom part, the we are already close to the minimum point and yes somewhere close to 130 generations, we have reach to the optima. So, let us look one more time. We are already close to the optimum solution now and in the zoom if we make a zoom in the y axis here close to 130 generations, we reach to the minimum solution. Yes, you can see here.

So, we can see that differential evolution is able to solve such kind of a problem where you have many local optimum solutions even we have increased the number of variable,

DE was able to solve that problem with limited number of population size and limited number of generation.

(Refer Slide Time: 13:11)

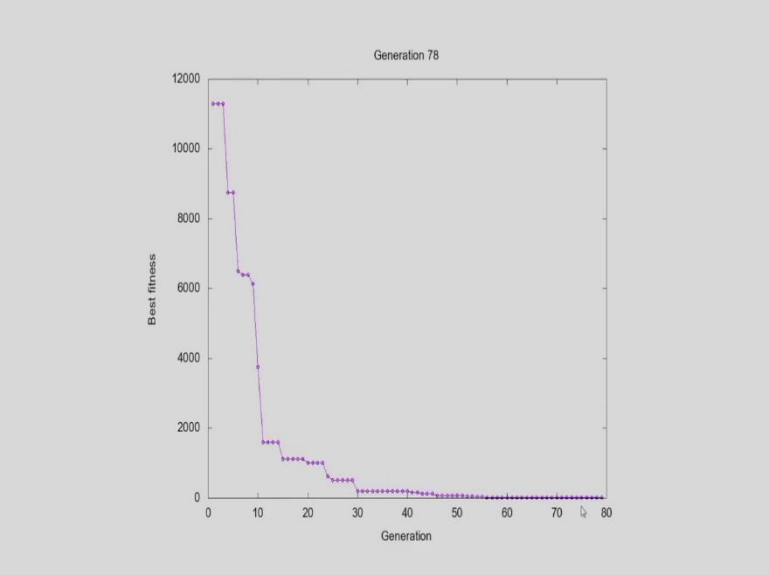


Now, suppose if we are going to increase the variable this means that if we increase it; the problem complexity will increase. So, let us see what is the performance of differential evolution here to solve this problem. So, the Rosenbrock function as you can see here, we are taking number of variable as 10, population size 60 and the number of generation 200. So, we have kept the same population size and a generation.

Let us see in this limited number can we solve this problem the variant of DE remains the same F is 0.5 and crossover rate is also 0.5. Now, since it is a 10 variable problem, we can see the progress of DE when we will be plotting the best fitness versus generation.

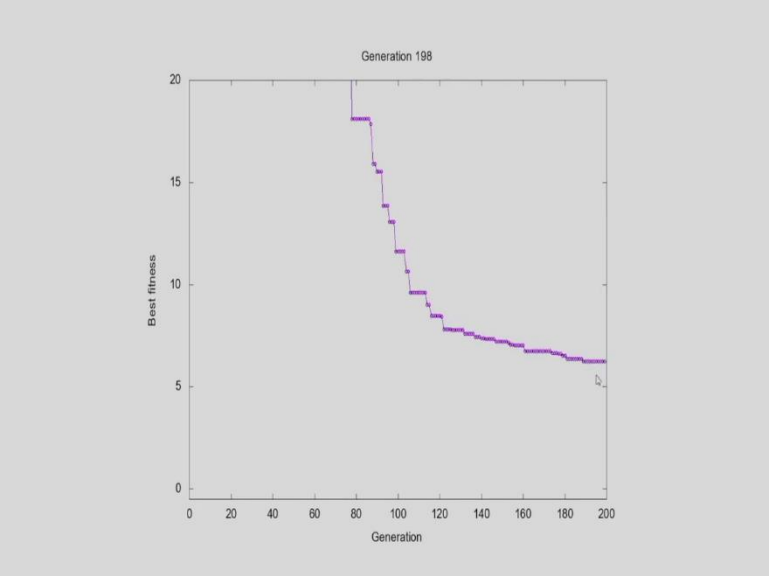
As you can see the best fitness in early generation it is almost close to 1100 and drastically it is increasing the it is improving the performance and somewhere close to 50 generations, we are going towards the optimum solution.

(Refer Slide Time: 14:23)



So, let us see how much this particular algorithm is able to solve this problem yes. So, we started with a very large value, we keep on improving the fitness.

(Refer Slide Time: 14:31)

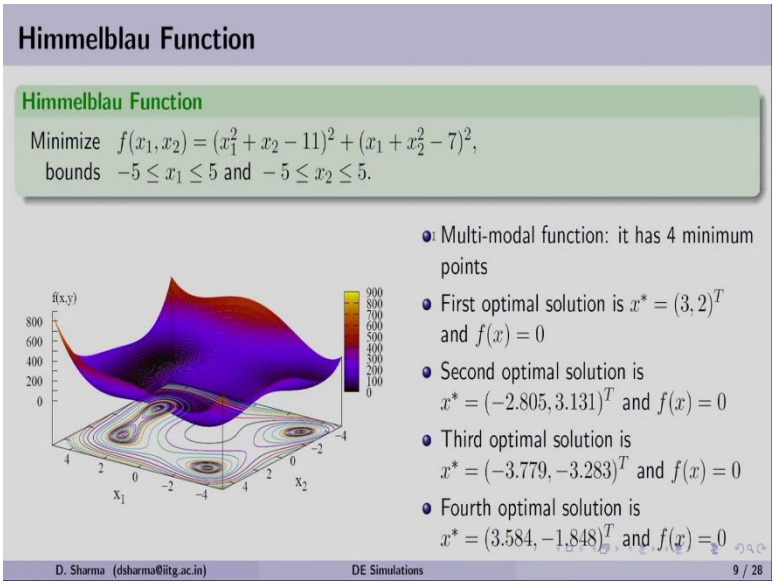


Now, let us look at the zoom version here. So, in this we have made this particular y axis between 0 to 20. You can see with the generation, the fitness is keep on improving slowly and slowly.

So, here if we look at the progress of differential evolution although we have not converged to the optimum solution, but we allow this algorithm for more number of a generation looking at the trend of reducing the fitness; definitely differential evolution can find an

optimum solution even for a number of variable or a large number of a variable that is 10 for Rosenbrock function.

(Refer Slide Time: 15:17)



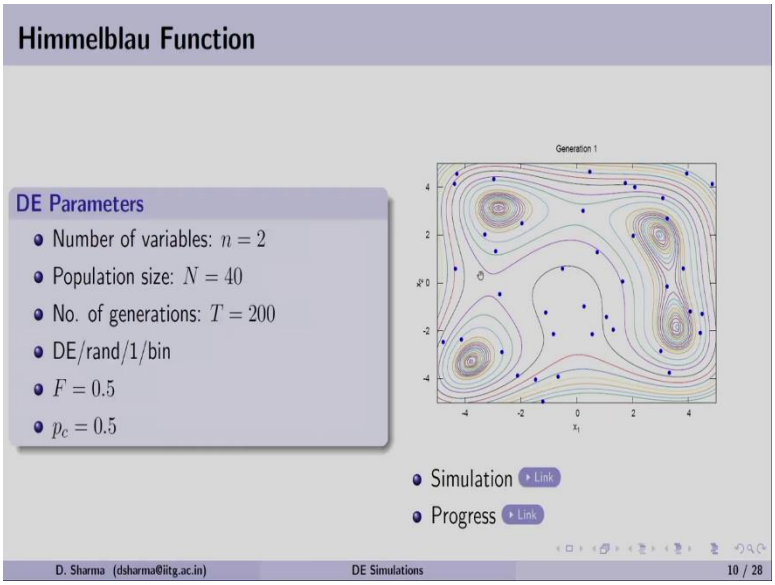
Now, coming to the Himmelblau function. Himmelblau function as we know it is a 2 variable function. So, looking at the function here, we want to minimize the function and on the right hand side, we can see the equation. Both the variables we have considered between minus 5 to plus 5. Look at the look at the surface of the Himmelblau function. So, x 1 and x 2 plane is given to us and we have a third axis which is the function value.

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

$$bounds\ -5 \le x_1 \le 5\ and\ -5 \le x_2 \le 5$$

Now, in this case, you can see that this particular surface says that we are going to have multiple optimum solution. Now looking at the contours here, we can find that this Himmelblau function in the given range of x 1 and x 2; it is a multimodal function. So, on the right hand side, we have mentioned that it is a multi-modal function; it has 4 minimum points. So, all the 4 points as you can see on the right hand side are given and at all the 4 point the function value is 0.

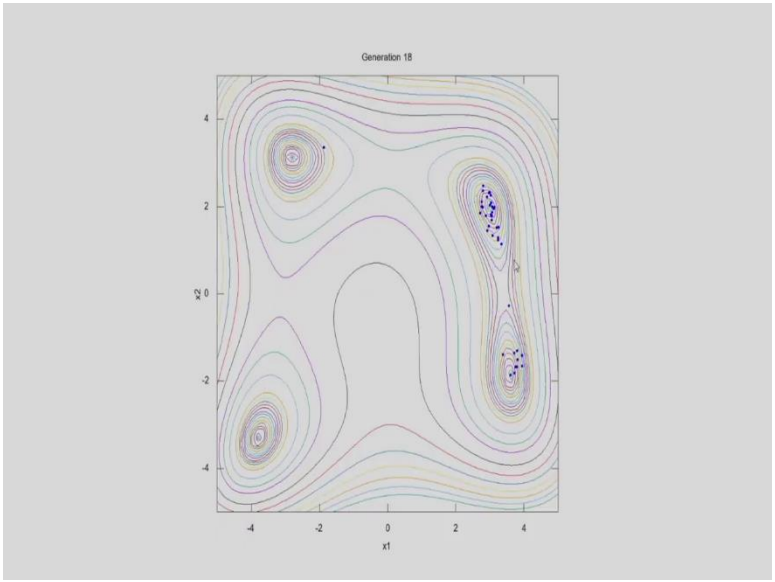
(Refer Slide Time: 16:27)



Now, coming to the DE parameters, the number of variable is 2 that is fixed for Himmelblau function. We have taken a small population size of say 40 number of a generation 200; the same variant of a DE which is DE rand one bin is used here.

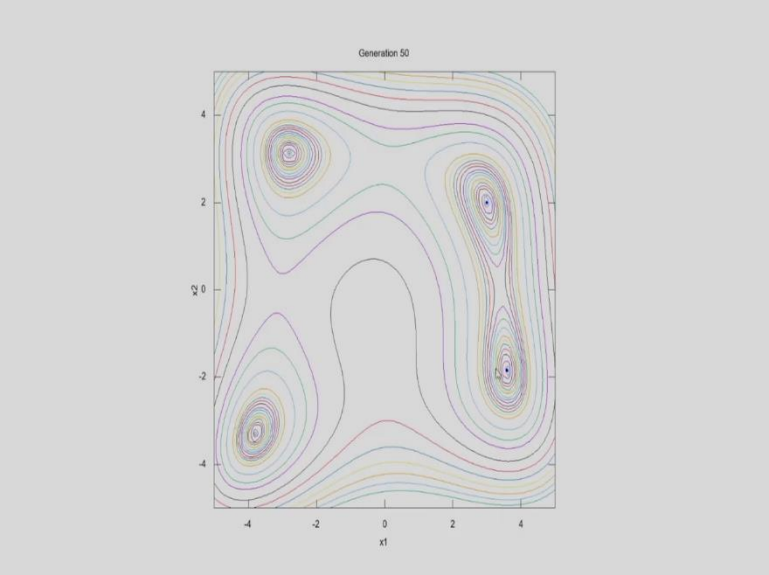
The scaling value is 0.5 and crossover rate is 0.5. On the right hand side, we can see that the initial population is generated throughout x 1 and x 2. Now since it is a multimodal problem, let us see how DE is going to solve this problem.

(Refer Slide Time: 17:07)



Now you can see these blue dots which are the target vectors here.

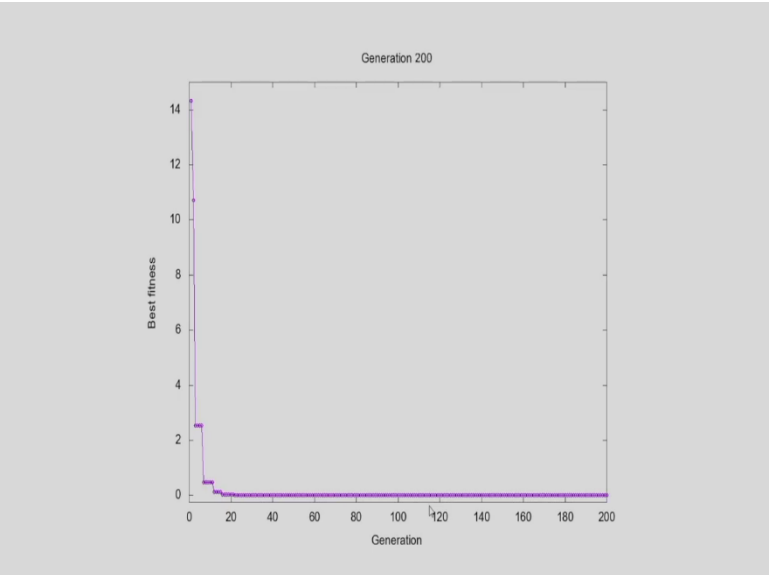
(Refer Slide Time: 17:12)



They have converged to the 2 optimum solution and now we will see that close to 105 generation all the solutions are now converge to the 1 optimum solution. So, let us see one more time how these solutions or the vectors are moving here and as of now there the are two peaks or the two optimum solutions are found, but close to 105 generation all the vectors in the population are converge to the 1 optimum solution.

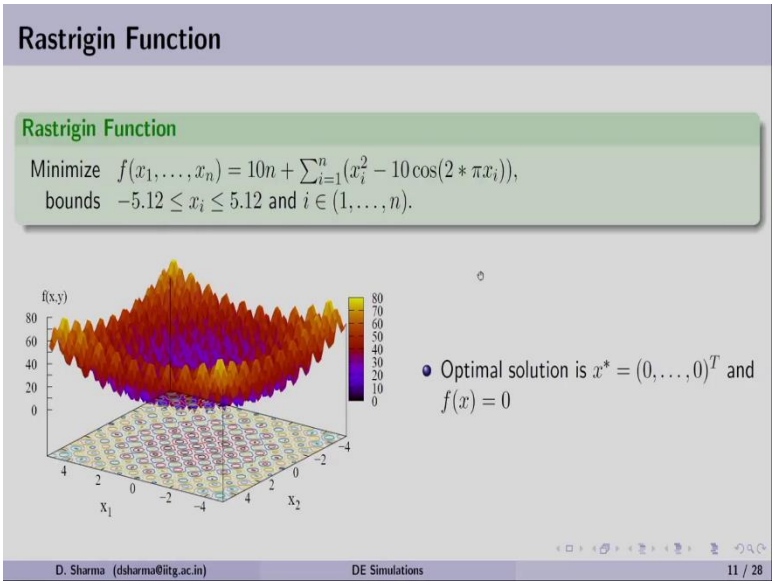
So, since this is a multimodal problem, we have to change our fitness and then only we can solve multimodal problem because the current version of DE is developed for identifying the one global optimum solution. So, in that case all vectors are converged to the one solution.

(Refer Slide Time: 18:15)



Now, let us see the progress here the progress of a DE here we can see that it is started somewhere more than 1400 and close to say 15 generation or even 22 generations, the DE has find one of the optimum solution. And in further generation, rest of the vectors were moving towards the optimum solution and finally, we got just one optimum solution for the given problem.

(Refer Slide Time: 18:43)



Now, after solving the Himmelblau function, let us solve the Rastrigin function. The Rastrigin function, the objective function value is given on the top. We want to minimize the function and you can see the equation of Rastrigin on the right hand side.

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

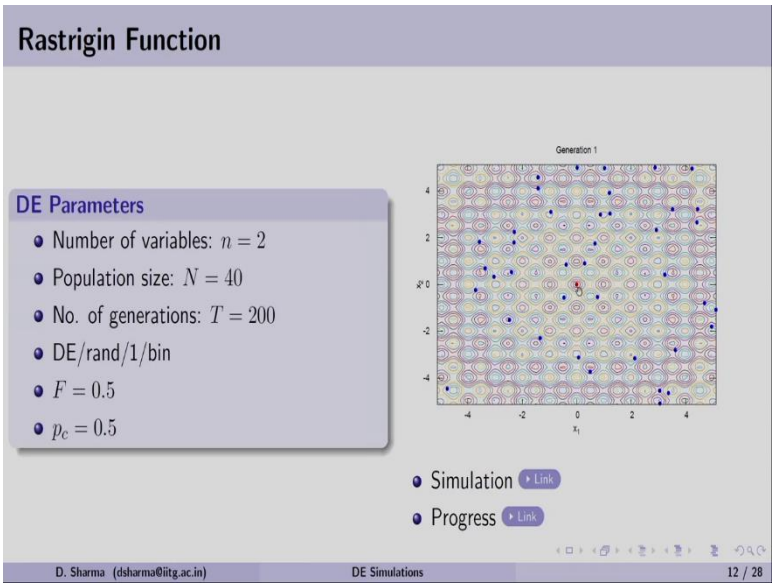
$$bounds\ \ -5.12 \le\ x_i \le\ 5.12\ \ and\ \ i \in\ (1, \ldots, n)$$

Since it is a scaling function, we can take any number of variables and every time, the variable should take the value from minus 5.1 to 2 plus 5.12. Now for a two variable case as you can see on the left hand side, x 1 and x 2 plane are given here and the function value on the third axis.

Now, looking at the peaks and valleys of this particular function, we can make it out that this function will be having lot of local optimum solution. The same thing can be seen in

the x 1 and x 2 plane when we see the contours of the function. The global optimum of this function is at the origin as you can see on the right hand side and the function value is 0.

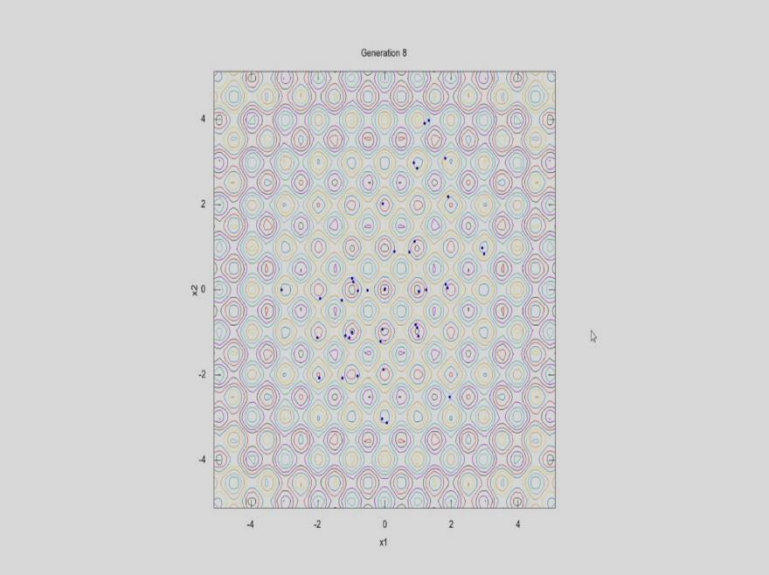
(Refer Slide Time: 19:51)



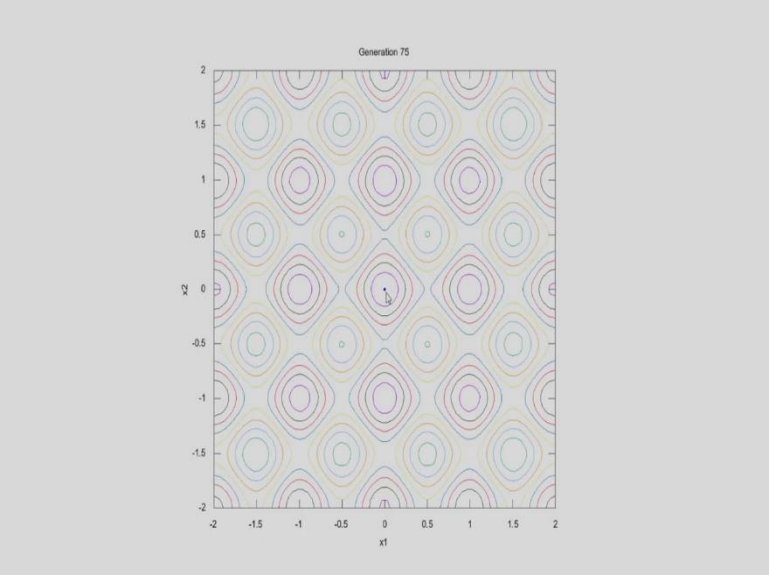
Now, looking at the DE parameter, we let us take the first simulation n equals to 2 for and then the population sizes 40 number of generation 200. We are taking DE rand 1 bin as the DE variant F and pc values are kept same as before.

Now, on the right hand side, we can see that in the initial population the solutions or the vectors are generated randomly in x 1 and x 2 plane. Now looking at the contours of the function, there are so many local optima and one global optimum solution which is at the origin where that is represented by the red color.

(Refer Slide Time: 20:41)

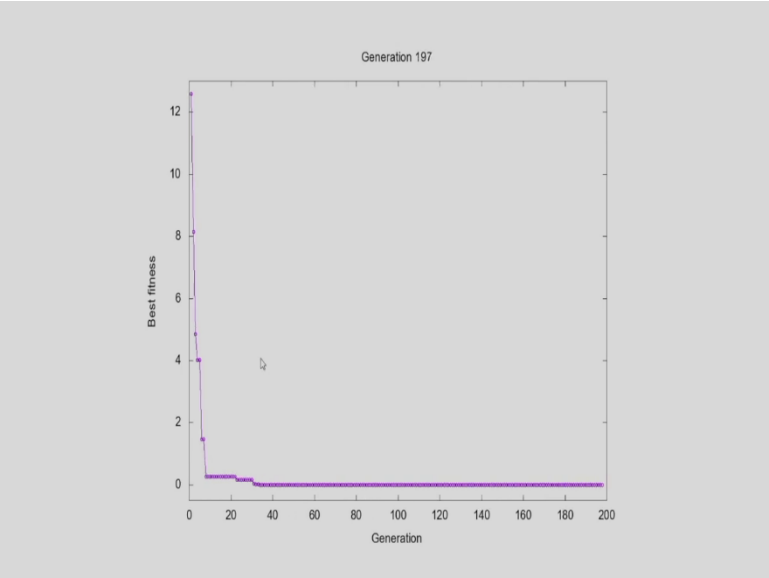


(Refer Slide Time: 20:43)



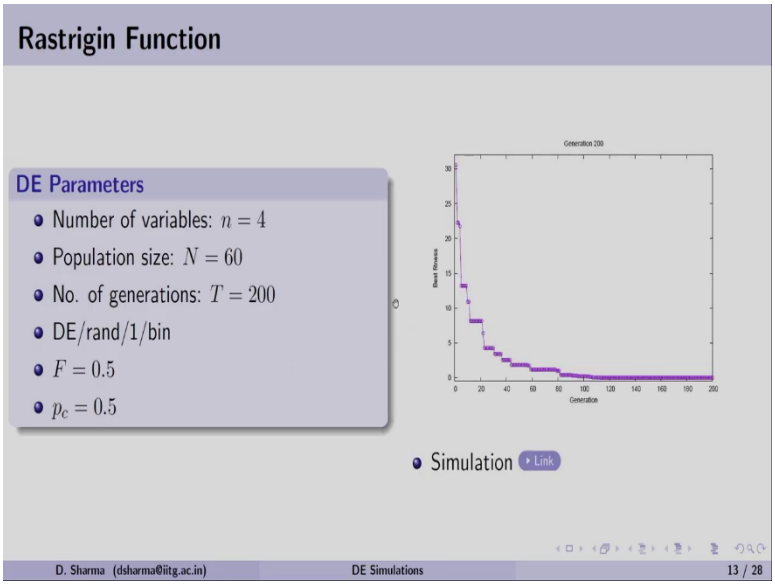
Let us see the simulation in this case. In this simulation, now you can see the blue dots which are the target vectors are keep on moving and even less than 30 generations, we are able to converge to this optimum solution and rest of the vectors are also converge to the optimum solution. Now, let us see one more time these solutions are moving even in less than 10 or 20 generations solutions are quite close to the optimum solution.

(Refer Slide Time: 21:12)



Let us see the progress of a DE, in this particular case. In this case, the best fitness has started somewhere close to 30 and as we can see somewhere close to 31 and a 32 generation DE was able to find the optimum solution for this particular function which is quite difficult to solve and then once we find it out, all the vectors are converge to the optimum solution.

(Refer Slide Time: 21:39)

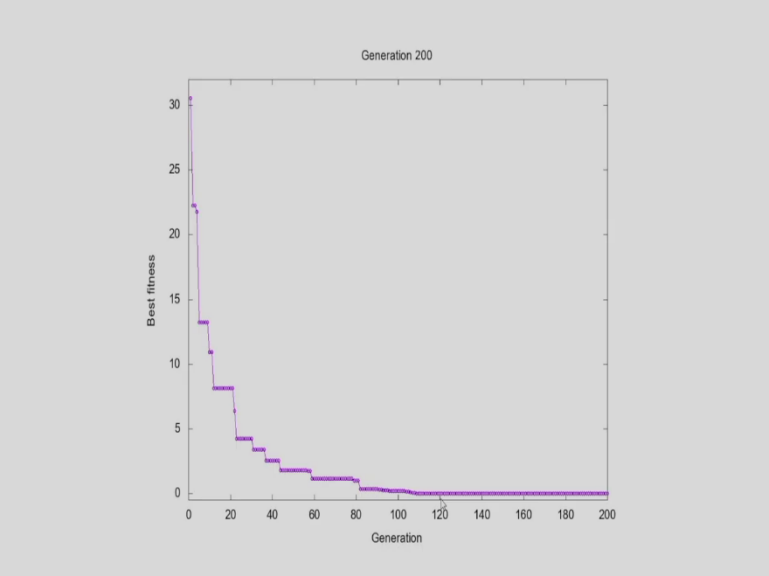


Now, let us take the case when differential evolution will be solving the Rastrigin function for 4 number of variable. The population size is now kept at 60 and the number of a

generation 200. The same variant of DE is used and F and pc value are kept same. Now, let us look on the right hand side.

Now since it is a 4 variable problem so, we can see the progress here that is the best fitness versus generation. Now here you can see the best fitness is in early generation, it is started close to little more than 30 and then with the number of generation, it is keep on improving.

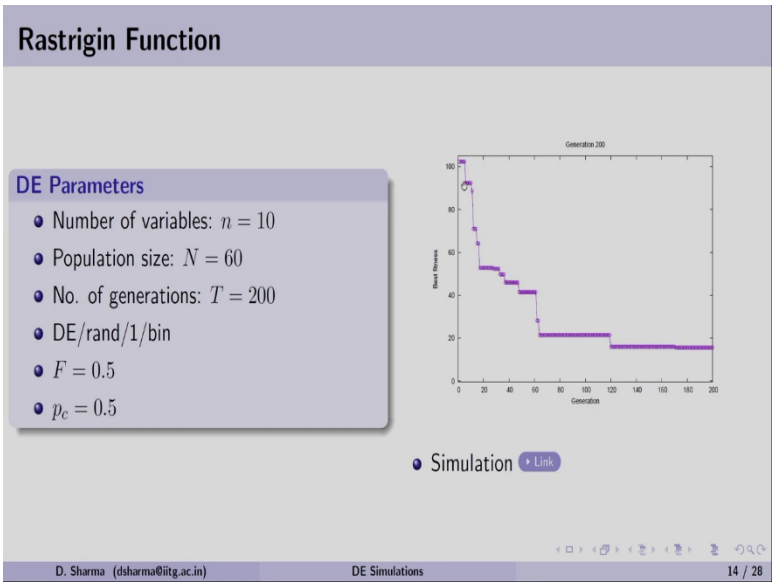
(Refer Slide Time: 22:34)



So, let us see what is the progress in this particular case when the number of variable is 4. So, as you can see with a number of a generation, the fitness is keep on improving and close to 81 generations we are already close to the optimum solution and close to 110 generations, we found the optimum solution using differential evolution. Let us look the progress one more time here.

So, you can see the continuously the fitness is keep on improving, the best fitness is improving and close to 110 generations. DE is already converged to the optimum solution.

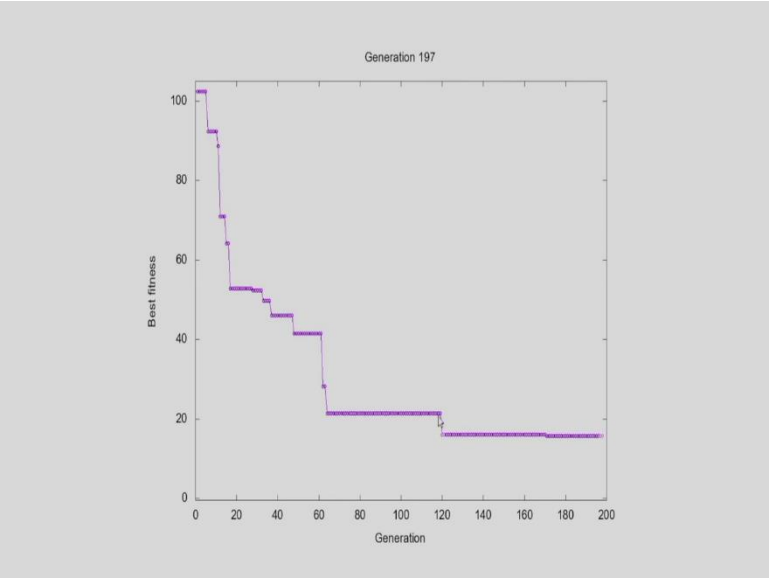
(Refer Slide Time: 23:16)



Let us take the another case. In this particular case, we are considering the number of variable 10 for Rastrigin function. So, let us see the performance. Now in this case, population size is 60, number of generation is 200 and we are keeping the same variant of a DE and F and pc are also the same.

Since it is a 10 variable problem so, we will be showing the progress that is between best fitness versus the number of generation. As you can see initially the fitness was more than 100 and with the number of generation the fitness was keep on improving. Now look at the 200 generation it we can see that differential evolution was unable to find the optimum solution for 10 number of variable. So, let us see what is this simulation now here.

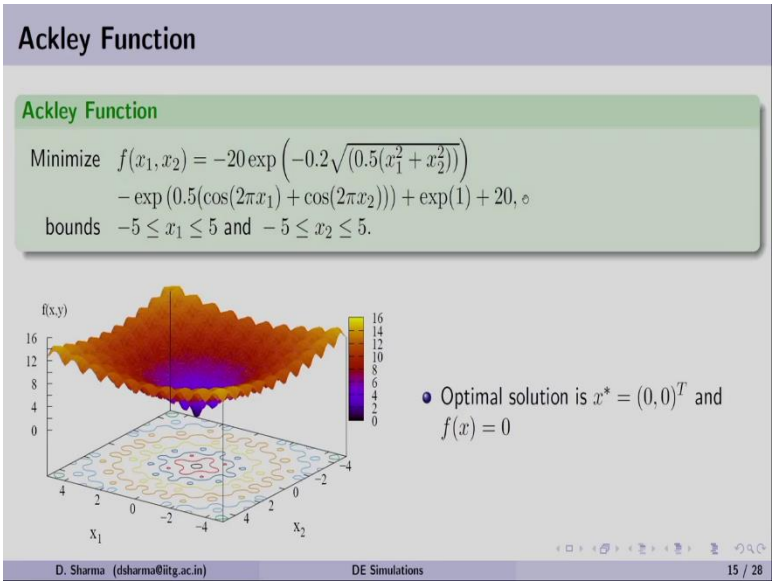
(Refer Slide Time: 24:13)



Now, as you can see the fitness is started improving from 100 and every generation, there is an improvement in the fitness. And since it is of it is a 10 variable problem, the DE at some point of a time say after 60th generation, we were unable to find the best solution.

So, as you can see that although there is the DE was unable to find the optimum solution in 200 generations, but looking at the progress of differential evolution if we allow this algorithm for more number of generation with larger population size, then DE can solve this problem as well.

(Refer Slide Time: 24:59)



Now, we have come to the function called Ackley's function. On the right hand side, you can see we want to minimize the function and this particular equation since it involves exponential as well as cost, this is going to be a difficult one of the difficult problem. It is a 2 variable problem and both the variable should lie between minus 5 to plus 5.

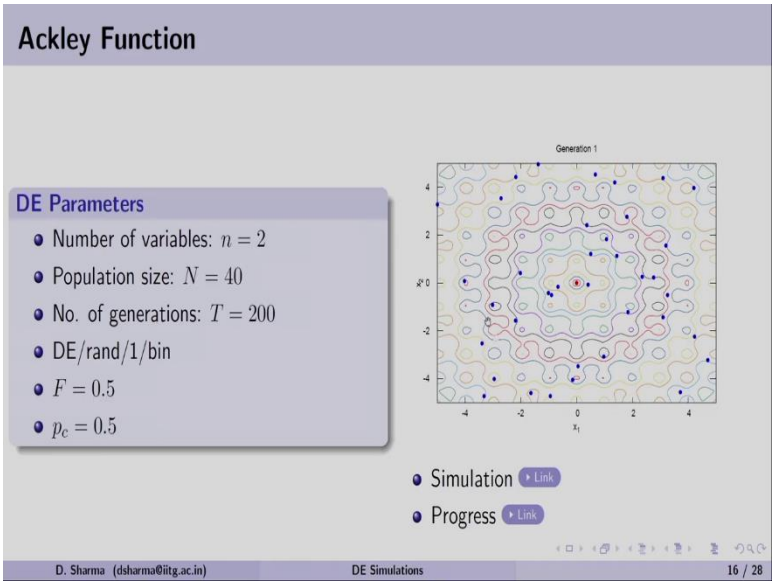
$$Minimize\ f(x_1, x\_2) = -20\, exp\left(-0.2\sqrt{\left(0.5(x_1^2 + x_2^2)\right)}\right)$$

$$\& - exp\left(0.5\left(cos(2\pi x_1) + cos(2\pi x_2)\right)\right) + exp(1) + 20$$

$$bounds\ -5 \leq x_1 \leq 5\ and\ -5 \leq x_2 \leq 5$$

Now, looking at the surface of this particular function x 1 and x 2 plane and the third x is the function value. Now looking at this particular surface as you can see there are small peaks and valleys that creates many local optimum solution. So, here in the x 1 and x 2 plane, we can see that there will be a lot of local optimum solution, but the global optimum solution for this problem will be at origin as you can see on the right hand side and the function value is 0.

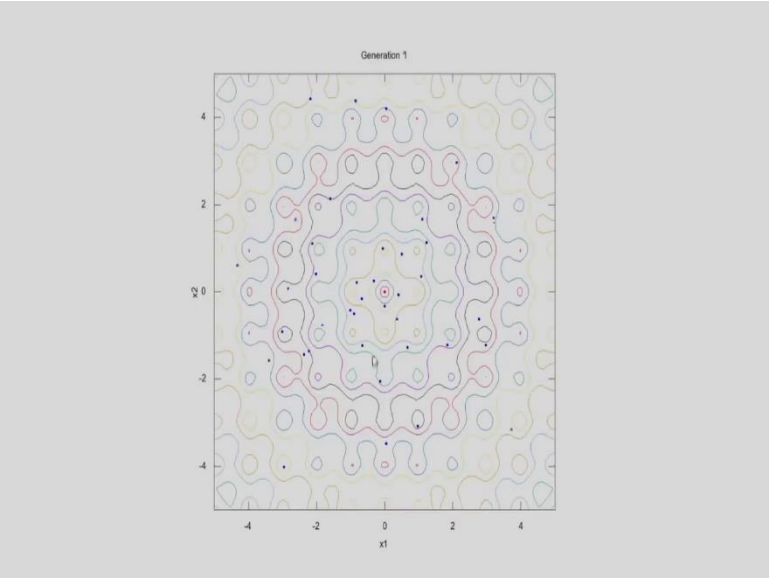
(Refer Slide Time: 26:00)



Since the Ackley's function is a 2 variable problem so, it is n equals to 2 population size for 40, number of generation 200, we are keeping the same variant of DE and F and pc are 0.5.

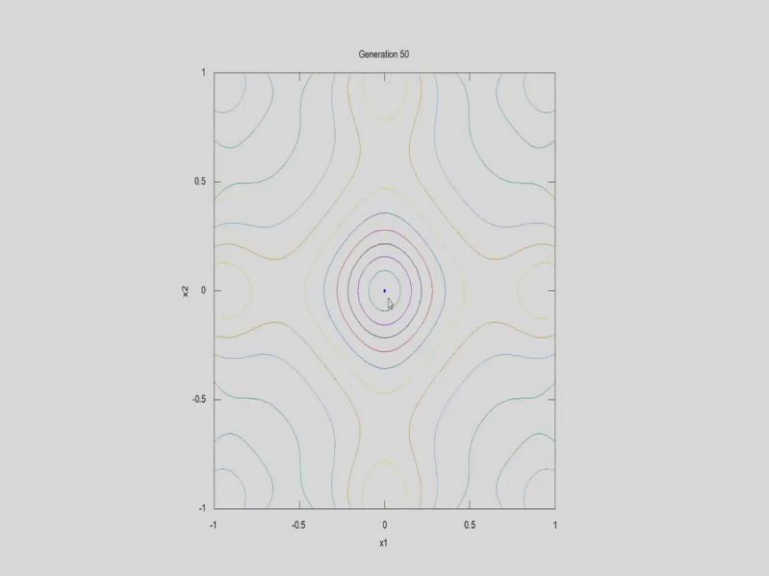
Now, here on the right hand side, we can see that the blue dots which are the target vectors which are generated randomly initially and these vectors or these points are distributed in x 1 and x 2 plane. Let us see how this DE is going to solve this problem.

(Refer Slide Time: 26:37)



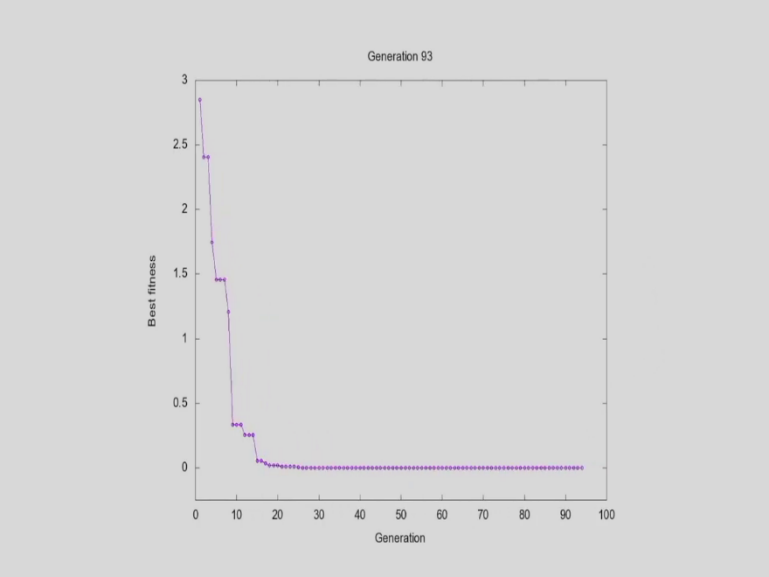
Now see the blue dots are keep on moving towards the optima.

(Refer Slide Time: 26:40)



And close to 30 generations, all the solutions were converged to the optimum solution. So, we can see even though there are many local optimum local optimum solutions are available, this DE was able to solve this problem quite efficiently.

(Refer Slide Time: 27:06)

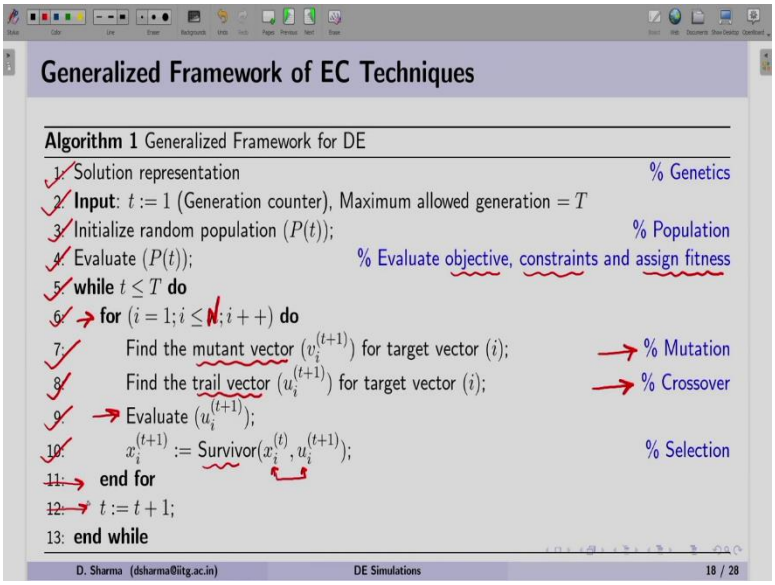


Now let us look at the progress of DE for Ackley function. It is started somewhere close to 2.7, the fitness is keep on improving and somewhere close to 25 number of a regeneration differential evolution was able to find the optimum solution. Let us see the simulation one more time. Now, in this case yes the fitness is keep on improving and somewhere close to 25, the DE was able to solve the given problem.

We have seen the simulation of DE on various function. What we have identified is that even the function has so many local optima's, DE was able to solve all those problems. Moreover when we have increased the number of variable say from 2 to 4, DE was found to be efficient for solving such kind of problems.

But when we take large number of variables say 10, then DE needs more generation as well as larger population size to solve the given problem. Now, with this see with these simulations, we are moving towards the algorithmic implementation of DE.

(Refer Slide Time: 28:30)



Now as you can see here, we are we have already gone through the generalized framework of DE here. As we can see that it starts in the step number 1, it starts with this solution representation and since we are considering all variables real so, we are showing this algorithm representation for the real numbers. To start the differential evolution, we need certain input.

So, here number of generation or generation counter are mentioned, but there are other parameters which we have to include here. Then we start the in a step number 3, we generate the initial population. In this case, we generate the random vectors in x 1 x 2 and x n plane; thereafter we evaluate the population.

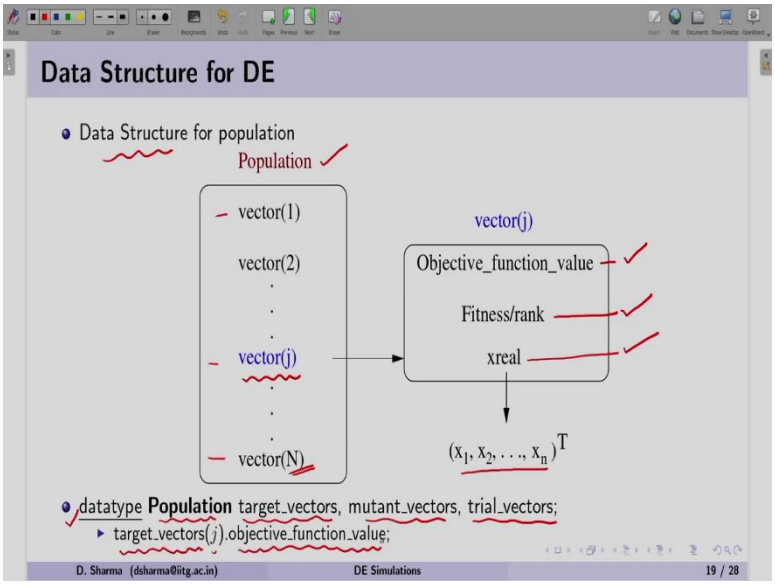
Now, in this evaluation, we include to calculate the objective function constraints as well as assigning the fitness. Once the population is once the population is evaluated, we are in the upper loop of the number of a generation in a step number 5.

Now, coming to the step number 6 as we know that we have to run a for loop for the number of vectors. So, in this case this n should be the capital N. So, in this case, we will be running this for loop for each target vector say so, we have a target vector i; for this target vector in a step 7. We are going to create the mutant vector. Now this mutant once this mutant vector is created as we know this mutant vector creation is called mutation here; thereafter in step number 8, we create trial vector.

Now, in this trial vector, we have we the way we make it, this is considered as a crossover for us. Now since we have generated a new vector, we have to evaluate it. So, this trial vector is evaluated in step number 9 and finally, in step number 10, we have the survivor.

So, here the survival is between the 2 vectors; that is the target vector x i and the trial vector u i. And this particular for loop will finish at step number 11. In 12, we increase the counter because increase the counter of number of a generation because that will terminate the DE algorithm. So, let us go one by one, how to we how we can make different functions to make DE.

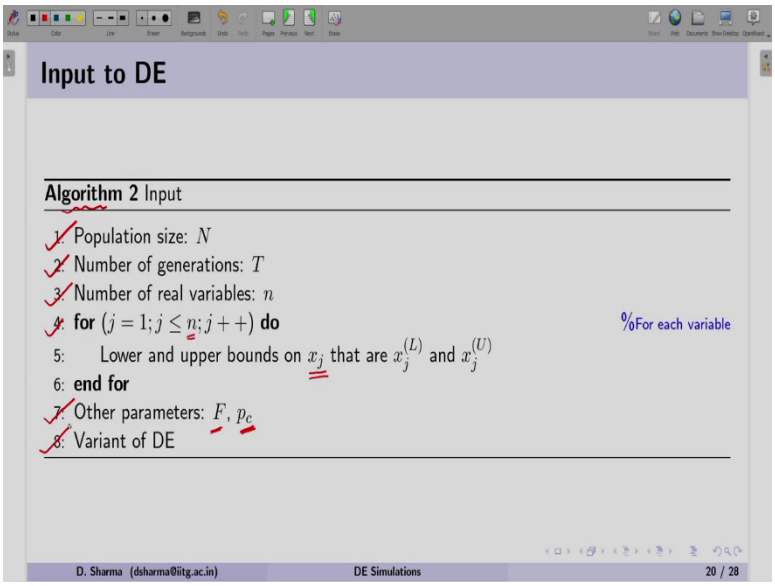
(Refer Slide Time: 31:28)



So, let us start with the data structure here. In this data structure as we have to as we have the population. So, this population is made of vectors. So, as you can see that different vectors are given here. So, the size of population is n as you can see at the bottom. Let us take a very generalized case of say vector j and in this particular vector j we should save the value of objective function, we should also save the value of fitness and we can should also save the value of the vector which is x 1 to x n as you can see here.

Now, here the data type what we can generate is the population and that population can have target vectors, mutant vectors and trial vectors. So, in this case if suppose we have to either write a value or update a value, we can call as a target vector j and dot objective function. So, this particular if we are going to call this data structure that will give me the objective function value of the j th target vector.

(Refer Slide Time: 32:43)

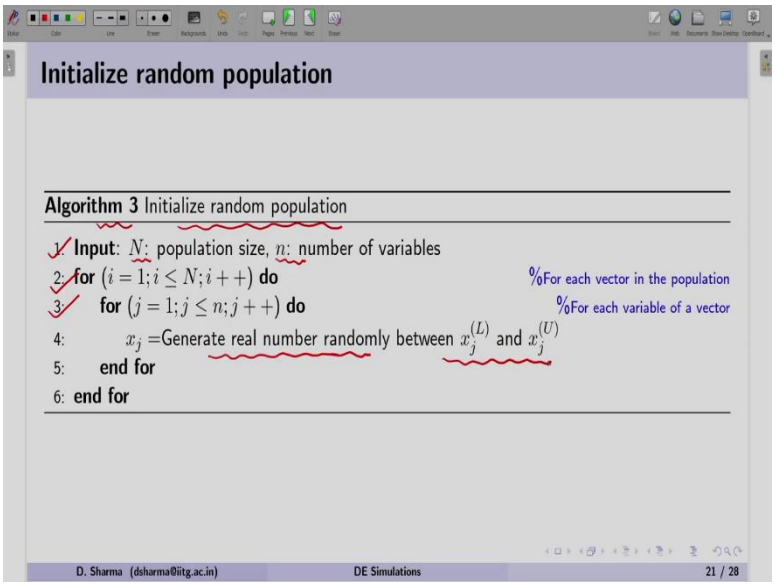


Now, coming to the input to DE looking at the algorithm number 2, first we have to set what is the population size. Once it is done, we will be setting how many generation we have to run our DE, then what is the number of variable as you remember that the Rosenbrok and Rastrigin function can take any number of a variable. So, we have to take this value of number of variable.

For each variable so, in a step number 4, the for loop is run for each variable. It is only because we have to tell what is the lower and upper bound for the variable say x j. Once it is done, then the other parameters can be given to DE; that is the scaling factor F and the cross and the crossover rates say p c.

Moreover just for our reference, I have included the variant of a DE because since the mutant vector that we generated we can have different ways to do it even the different types of crossover we can use it. So, that is why in the literature different variants are proposed. So, once we decide what kind of variant it is so, we can run our differential evolution. So, therefore, as a reference we are including the variant of a DE as an input.

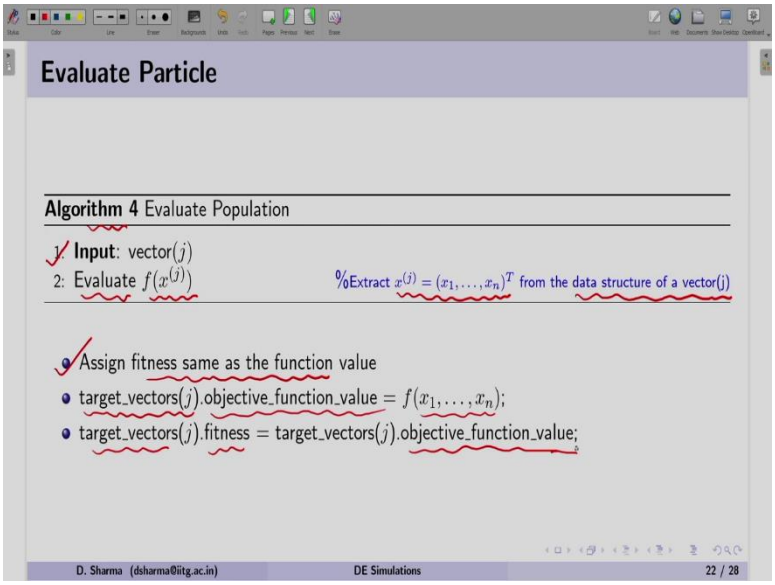
(Refer Slide Time: 34:11)



Now, coming to the algorithm number 3 here, we have to now generate the population and that to be random. So, in the first case, we have to tell what are the inputs and what is the population size and how many variable. So, we are going to have two loops. As you can see in step 2, the upper loop will be on the number of population or the vector size and the lower loop will be on the number of variables.

And in step 3, we will be generating a real number randomly between the range of this variable that is from the lower and the upper bound and that in that case our computer will help us to generate a random number between lower and upper bound of the given variable.

(Refer Slide Time: 35:01)

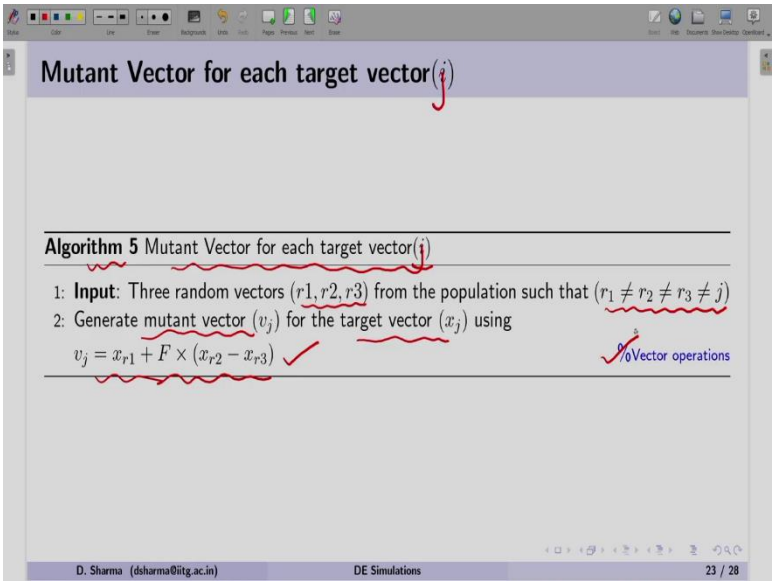


Now, let us evaluate the population as you can see in the algorithm number 4. The input to this algorithm is say vector j. So, we are passing one vector at a time. In this case the evaluation means we are going to evaluate the function value at vector x j. Now as we have mentioned here this x j is nothing, but we are going to find out what is that vector and that we can find out from the data structure of the vector j as we have understood earlier.

Now, here the previous simulations, we have considered that the fitness is the same as the function value. So, that we are keeping the same in this particular algorithmic implementation. Now once we have calculated the function value, we have to update our data structure. How we can update? Say for example, target vectors j objective function value is equals to the function value the mathematical value which we calculate.

Since the fitness is the same so, we have to update the fitness of this vector j with the function value of the same vector j. So, it is easy because we can just take the value from the objective function to the fitness as of now.

(Refer Slide Time: 36:21)

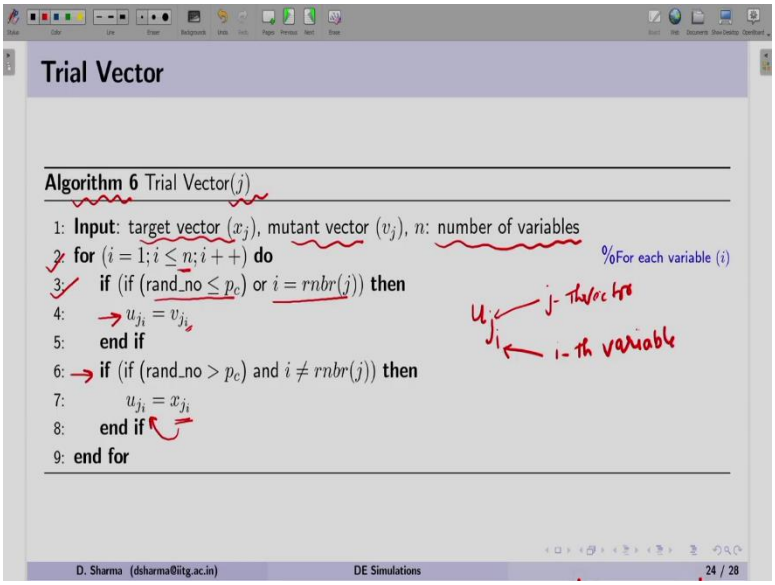


Now we are in the loop inside the second loop that is run for every target vector. So, let us see how we can create a mutant vector here. In algorithm 5, we are going to generate the mutant vector for each target vector say; i. So, instead of i for our similarity we let us take it as j; so, for the target vector j.

So, as we know we are going to have three vectors; r 1, r 2 and r t r 3. These three vectors we are randomly selecting from the population and we have to make sure that r 1 should not be close to r 2 and r 3 and j all of them should be different in order to generate the mutant vector v j, the target vector x j for the target vector, we can use the simple formula as we are using.

So, we did this hand calculation. So, this is nothing, but the vector operations is it is mentioned. So, it is quite simple to make the function for the mutant vector once the mutant vector is there, then we have to create the trial vector.

(Refer Slide Time: 37:32)

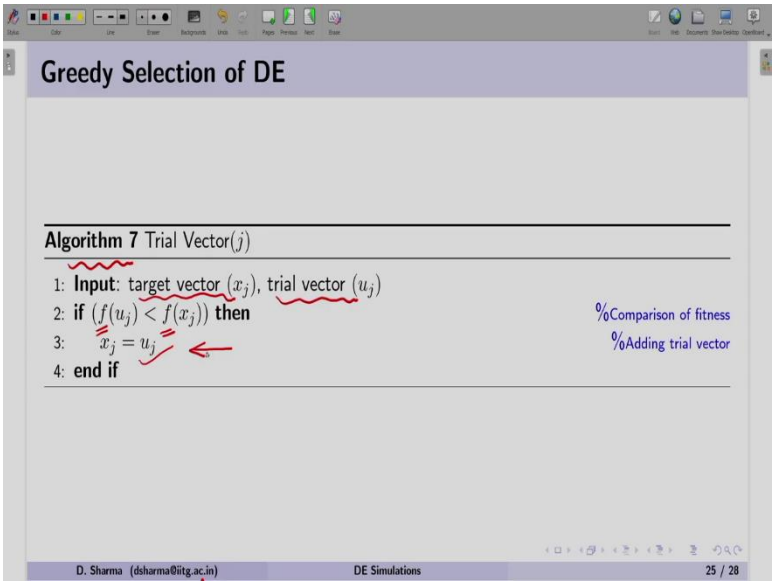


The algorithm 6 is for the trial vector; let us assume that we have the input as vector, so we have a j. So, in this case we are going to have our input as a target vector, we are going to have an input as a mutant vector and the number of variable. So, as we remember that we keep on exchanging the variables based on the condition.

So, looking at this step number 2, we have a for loop over the each variable and then we look at the condition given at step number 3 that if the first condition is satisfied or i is equals to random r and br j, then the j-th component of the j- th component of the mutant vector will be copied to the j-th vector of the trial vector.

If the second condition; in this case the i-th component of the target vector will be copied into the i-th component into the into the trial vector. So, here you can see that everywhere we are writing in this particular form. This says that if suppose I am taking u j and i so, as we know this is the j th vector. So, this is the j th vector and i represents the i th variable. So, this is the way, we are understanding this particular algorithmic representation.

(Refer Slide Time: 39:22)

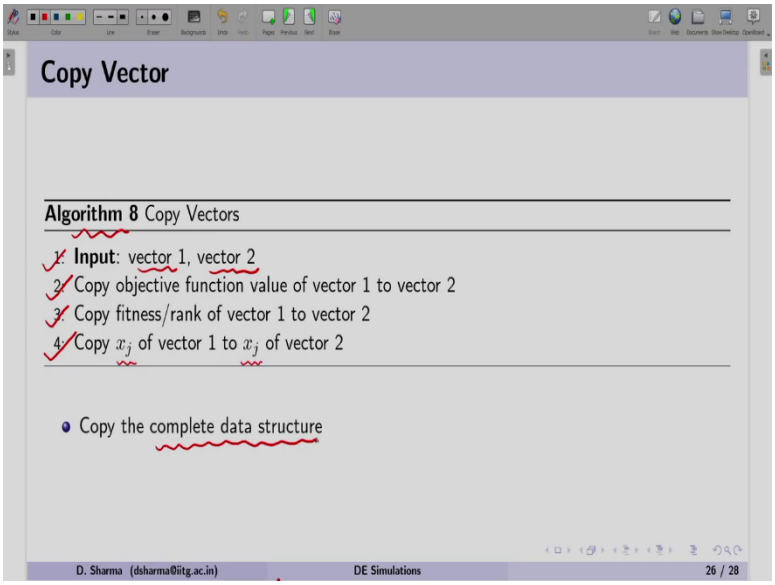


Once it is done we have to perform the greedy selection to select either we will be selecting the trial vector or we are keeping the same as the target vector. So, as we can see in algorithm number 7, the input to the input to the greedy selection is the target vector xj and the trial vector uj; here we will be comparing the fitness.

So, here i currently we have written a small f, but that should be the fitness here. So, when we are comparing the fitness of the trial vector with the target vector; if the fitness of the trial vector is smaller than the target vector, we should update the complete vector.

So, here we are assuming, it is a minimization problem and accordingly this condition is written. So, in this particular case, we are updating the full vector and as we know if this condition is not satisfied so, the same target vector is taken into the next generation.

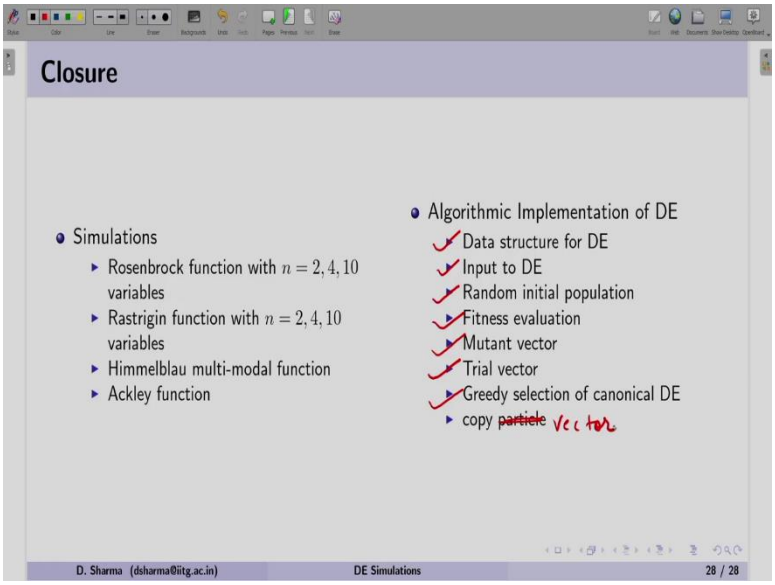
(Refer Slide Time: 40:30)



Now copy the vectors, it is an important function it is because when we are copying the function as we have seen in the selection. So, we have to copy the complete data structure. So, the data structure as of now, the basic data structure consists of three things. So, we have to complete each and everything here. So, let us look at the algorithm number 8 here. So, we have two inputs a vector 1 and vector 2.

In this case the data from vector 1 should be copied to the vector 2. So, in step 2, we can see first of all we are going to copy the objective function of vector 1 to vector 2. Thereafter we are going to copy the fitness of vector 1 to vector 2 and finally, the complete vector will be copied say x j of vector 1 2 x j of vector 2. So, as we have mentioned earlier, we have to copy the complete data structure. So, with this algorithmic representation, we have come to the closure of the session.

(Refer Slide Time: 41:41)



So, in this particular session what we have gone through is; first we started with the simulations. So, we have set the DE parameters and then we solve four types of problem. So, here initially we started with the Rosenbrock function and in that case since it is a scaling function, we have showed the simulation of the Rosenbrock function for 2, 4 and 10 number of variables. Similarly the Rastrigin function is also a scaling function. We showed the simulation of differential evolution on 2, 4 and 10 number of variables.

So, in this particular when we were solving these two problems, we found that even though the problems are difficult to solve, but for 2 and 4 number of variables with a limited number of population size and number of a generation DE was able to solve the problem nicely.

But when we took the number of variable 10 in both the cases, DE needs more number of generation with the larger population size. So, that we can try however, as and when this condition arises, we generally take large population size and more number of generation.

Then we have solved the Himmelblau function. As we understood that Himmelblau function is a multi modal function. DE was able to solve this particular problem very efficiently. But what we found that since the current variant of the DE which we used, it is basically developed for finding the 1 optimum solution and that is why in the simulation when we run it, all the vectors were converged to the one particular optimum solution in this.

If we want to convert if we want to make our DE to work for multi modal problem, then we have to bring other concepts while calculating the fitnesss. So, one of them is called fitness sharing, that concept definitely we can bring it. So, that DE can be solved can be used for solving multi modal problems.

Finally, we have solved the Ackley problem. Now Ackley problem was another 2 variable function which has lot of local optimas and one global optima. Although problem was difficult here, but DE was able to solve this problem in less than 30 generations; it is only because it works on the concept of evolutionary computing.

Once we have gone through the simulation, we have understood the algorithmic implementation. So, in this algorithmic implementation, we started with the data structure. So, we should the basic data structure for a DE thereafter what should be the input to the differential evolution. So, as you remember that it was claimed by the authors that DE require less number of parameters.

And we can also see that apart from capital F and the crossover rate the other parameters are common to all evolutionary computation technique. So, we have to just fix 2 number of parameters and DE can solve our problems. So, we can see that DE actually needs less number of parameters as compared to the other easy techniques. Then we have showed the the function or algorithm implementation of random initial population followed by the fitness.

In the current case, we ensure that we are solving a minimization problem and the function value is the the fitness value of the of the particular vector is the same as the function value. Then we have gone through the mutant vector, it was a simple function. In the algorithmic implementation, we found that when we get the required number of input, we have to just pick say four random vector to find the mutant vector for each target vectors say j.

So, it was easy to make it and then the trial vector. So, the trial vector is also quite simple. In that case we have to have certain input like target and trial target and mutant vectors and variable by variable will look for the condition and exchange the positions of the variable. Finally we discuss about the implementation of greedy selection of canonical DE.

So, it since we were just comparing two solutions so, we look at the fitness and whosoever is the best is chosen. And finally, copy the solutions so if they should be the copy vector here and in this case it was easy only one point we have to make it that as and when we are copying one particular vector to the another vector, we have to copy the complete data structure that we have made at the beginning.

So, with this particular understanding of differential evolution on solving different kind of a problem and the algorithmic implementation of a DE, I conclude this session.

Thank you very much.