Computer Aided Applied Single Objective Optimization
Dr. Prakash Kotecha
Department of Chemical Engineering
Indian Institute of Technology, Guwahati

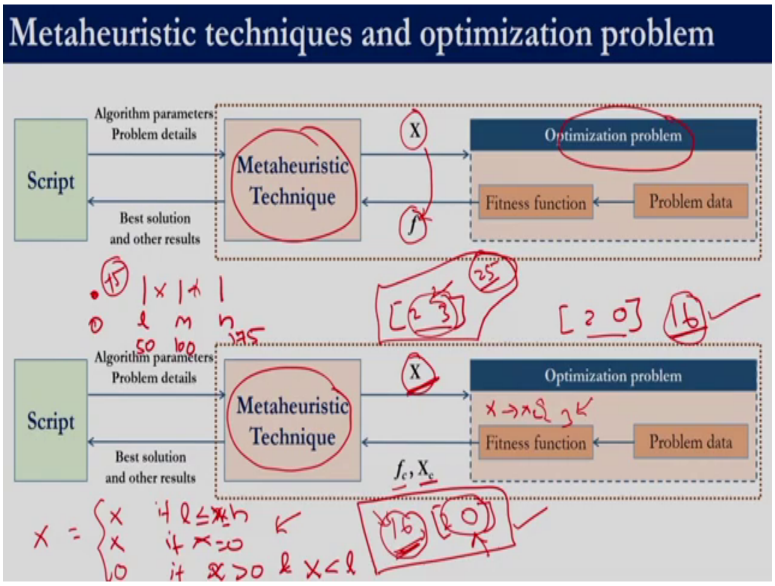
Lecture – 26
Constraint-Handling using Correction Approach Case Study: Production Planning

Welcome, in the previous session when we were talking about Constraint Handling. So, we only discussed on penalty approach right. So, as and when a metaheuristic techniques provides the solution, we check for the constraints right if that constraints are not satisfied, we assign an appropriate penalty right. So, the penalty can be either hard penalty or it can be determined based on the amount of violation right. So, previously we had solve production planning problem using this approach wherein we had used a hard penalty approach for handling the domain constraint, right.

So, any solution which gave a production which is non zero, but it is less than 1 of the penalized that solution with a constant value of 10 power 5 right. Whereas, for budget and investment constraint, we determine what is the extent of violation and we assigned a penalty appropriately. Another way to handle constraint is to correct the solution right. So, based on the nature of the problem; whatever solution we receive from the algorithm can be corrected right. Based on the domain specific knowledge right and the corrected solution can be returned back to the metaheuristic technique right.

So, it is a very simple approach that approach can provide significant improvement in the results right. So, that is the approach we will be looking in the session right and we will demonstrate it on the production planning problem. So, that you can realize the benefit of employing your correction approach rather than a penalty approach.

(Refer Slide Time: 01:49)



If you recollect whatever we have been discussing so far. This was the communication between the metaheuristic technique and the optimization problem which we had right. So, the metaheuristic technique will pass the solution X to the optimization problem and the optimization problem when in turn will provide the fitness function of this X right.

If this X is an infeasible solution and if we are solving a minimization problem this f would be very high value right. Whereas, in the correction approach, what we do is; the metaheuristic techniques still passes the solution X right. The optimization problem does not merely determine the fitness function of X right, but checks for the violation.

And if there are any violation in any of those constraint right, if there is an inbuilt mechanism right. So, if we employ some mechanism right which will help to improve the solution x right then that improved solution is passed back to the algorithm along with the improved solution we also pass the fitness function of the improved solution right.

So, it is not correct to convert X to X c let us consider we have a two variable problem and the decision variable passed are 2 and 3. And let us say there is a mechanism in the optimization problem which looks into the solution and corrects the solution to 2 0 right. And the fitness function of this 2 0 is let us say 16 right. For some evaluation of the objective function and the other constraints, the fitness function value turns out to be 16 and for this 2 and 3, let us say it is 25 right. So, what we obtain from the algorithm is 2 3, it has a fitness of 25 and it violates some constraint right.

There is some mechanism in this fitness function which corrects the solution right. The solution is corrected to 2 0. And the fitness function of this 2 0 is let us say it is 16. The solution is better than this solution and this betterment is not directly because of the metaheuristic technique, but because of the mechanism which we had employed over here to correct the solution right.

So, in this case to what the optimization problem is supposed to communicate to the metaheuristic technique is not the value 16, but also the solution 2 and 0 right. So, what should happen in this metaheuristic technique is; this 2 and 3 should be replaced with this 2 and 0 and the corresponding fitness function value right

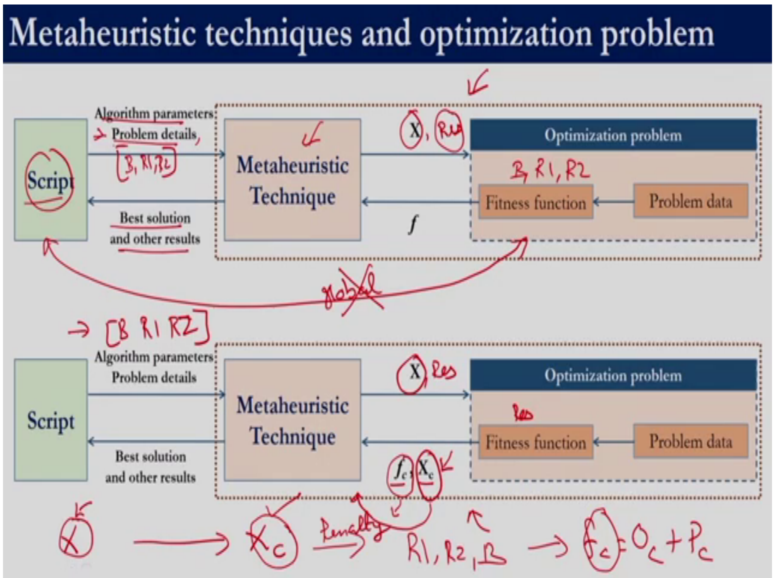
So, this is the correction approach wherein we get a solution from the metaheuristic technique, we employ some mechanism which is problem specific right to correct the solution and we will send back the corrected solution in this case 2 0 along with the fitness function value of the corrected solution right.

So, for example, if we consider teaching learning based optimization; we generate a new solution. That new solution let us say it is 2 3. So, when it is communicated to the optimization problem; the optimization problem corrects it to 2 0 and this 2 0 is sent back to the metaheuristic technique along with the fitness function value of 16.

So, the metaheuristic technique or TLBO in this case, should consider 2 0 as the newly generated solution along with its fitness function. So, it is this solution which has to be now used for greedy selection right. So, this correction approach as you can see it is a very straightforward approach only thing is that we should be able to develop a mechanism for the problem at hand right.

So, it is very problem specific, the correction approach is not part of the metaheuristic technique, but it is a part of the optimization problem. But it is implemented in the fitness function evaluation right. There are couple of things which you need to remember while employing a correction approach right.

(Refer Slide Time: 05:28)



The first thing is that the fitness function should be calculated for the corrected solution right. So, what we get from the algorithm is the set of decision variable let us call it as X right. So, that is corrected to a new solution let us say X c right. So, penalty is not supposed to be calculated for this X right. So, penalty is supposed to be calculated after correcting the solution. This is because when we are conveying the fitness function value right. We should convey the fitness function value of the solution which we are passing to the algorithm.

So, the solution that which we are passing to the algorithm is X c right. So, the fitness function should correspond to this corrected solution right. So, for corrected solution; if you think about it at least for this production planning problem if you see that, it will not have any violation of domain constraint right. So, this will have a better fitness than X right.

So, there is no need to calculate the fitness function of X itself right because, we are anyway correcting the solution. Once we correct the solution; we can find out the penalty for that right. So, in this case there would not be any violation with respect to domain constraint, but there maybe violation with respect to raw material 1, raw material 2 and the budget constraint.

So, the penalty is to be calculated for the corrected solution and then fitness function is to be calculated for the corrected solution. Because, this is the solution which we are conveying back to the metaheuristic technique plus penalty of the corrected solution. So, this will be the fitness which we need to pass through the algorithm. So, here you need to remember that whenever we are employing a correction approach. We are doing something on top of what the algorithm is doing.

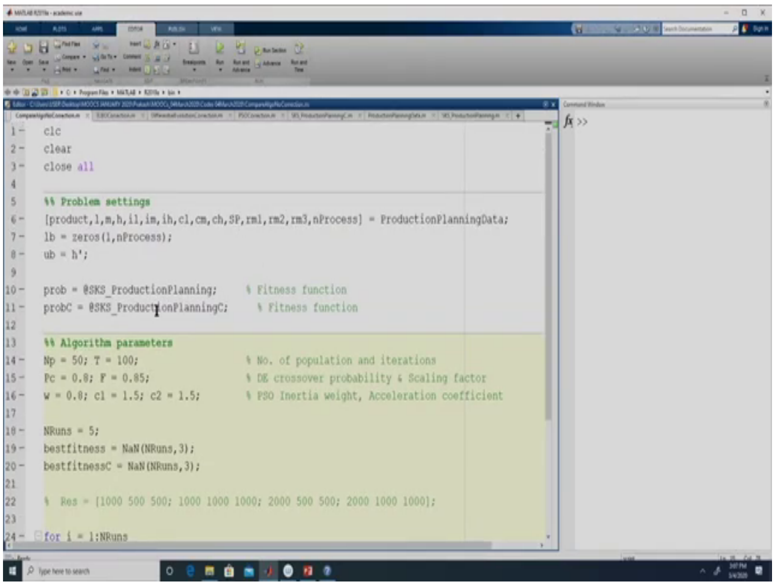
So, the algorithm let us say it suggested a solution 1, 2, 3 right. So, for some reason we are correcting that solution in the optimization algorithm let us say that solution 1, 2, 3 becomes 3, 8, 7 right. So, we have done something on top of the algorithms. So, in some sense you can say that we are disturbing the algorithm. For many problems depending upon the correction approach that you employ you may get a better result, but there may be cases where in a correction approach does not enable you to get a better solution.

So, in the case of production planning problem. So, this was l this was h right and let us say m. Let us say l is 50, m is 100 and h is let us 175 and this 0 is a feasible solution right. So, if we get any solution over here right. It does not violate domain constraint. So, you are not adding penalty.

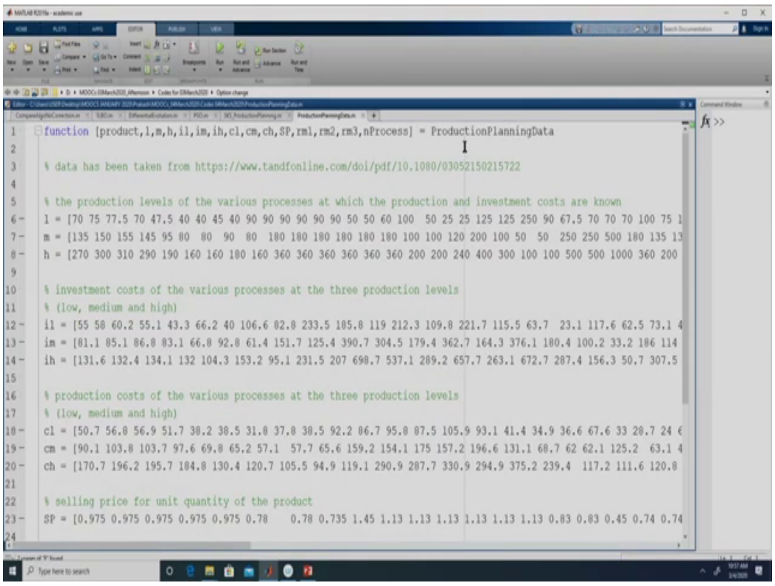
If the solution is here, we again do not add any penalty right or if the solution is 0, we do not add penalty. So, what we were doing previously is that if any solution o was over here which is greater than 0, but less than 50. Let us say it was 45 right. So, what we did for this solution 45 is that; we assigned a penalty for it right. So, instead of assigning penalty what we can do as part of the correction approach for production planning problem is that any solution which is greater than 0, but less than l can be converted to 0 right. So, this is some mechanism that we are employing right.

So, solution X remains X right. If this X is greater than or equal to l and less than or equal to h, remains X if X is 0 right. And what we are saying is; we will convert it to 0. The algorithm gave us some value, but we will convert it to 0 if this X is greater than 0 and it is less than l right. So, this is the correction approach that we are employing right. So, we will correct the X and we will no longer assign the penalty. Because, we have converted the solution in such a way that it no longer violates the domain constraint. So, we will no longer penalize it right.

(Refer Slide Time: 09:15)

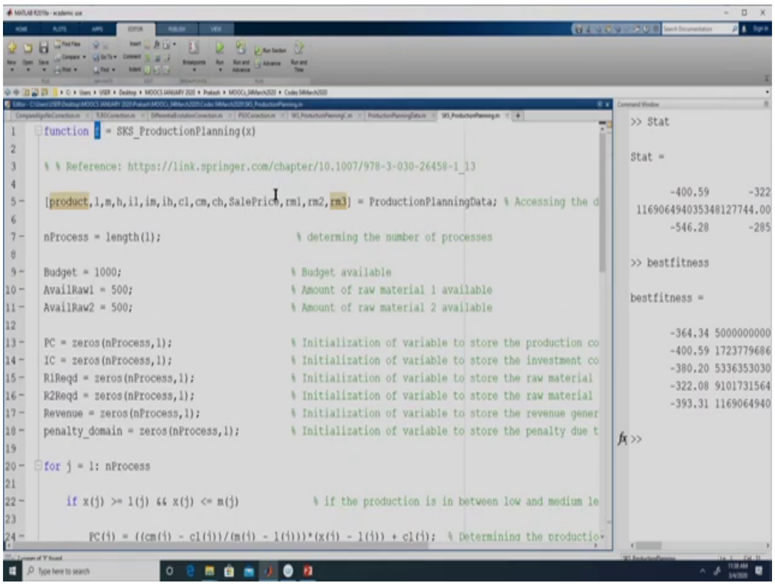


(Refer Slide Time: 09:16)



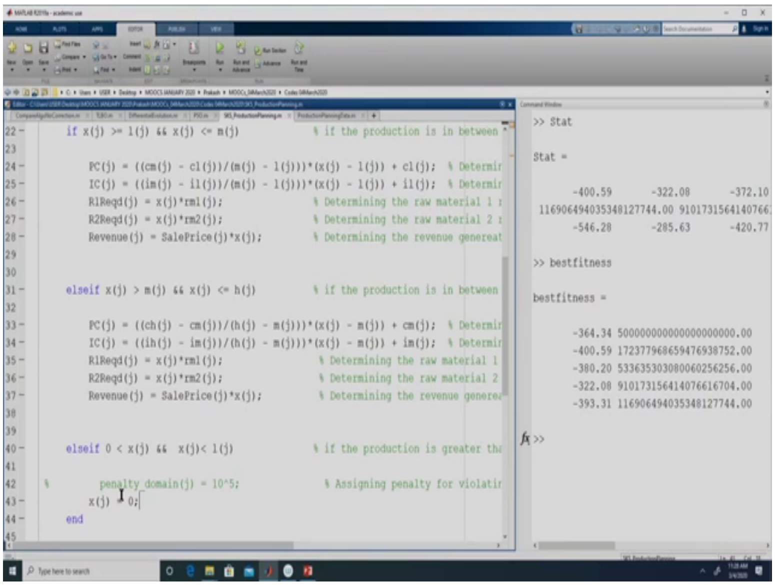
So, to better understand let us consider an example let us look into the production planning data.

(Refer Slide Time: 09:21)



Now, that we have designed a correction approach. We can implement that. So, this fitness function will receive the solution X, but it will return not only f, but it will also return the corrected solution ok. So, this is done. We are no longer going to have any variable which is going to violate the domain constraint right, because we are going to correct it right. So, if X lies between l and m; we do not need to do anything, because no penalty was calculated.

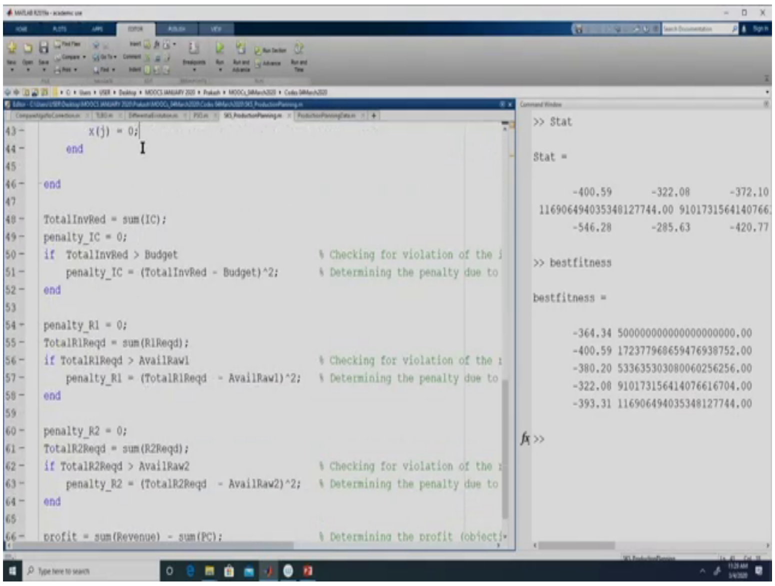
(Refer Slide Time: 09:42)



If X lies between m and h again, we do not do anything because it was already in a domain right. So, previously what we had done is; if a solution is greater than 0 right and less than l; we were assigning a penalty. So, right now what we will do is; we will not assign a penalty, but we will merely correct the solution right. So, we will say X of j is equal to 0 right. So, no matter where the solution lies right. If it is greater than 0, but less than l, it does not satisfy the constraint. Since it does not satisfy the constraint previously we were assigning a penalty.

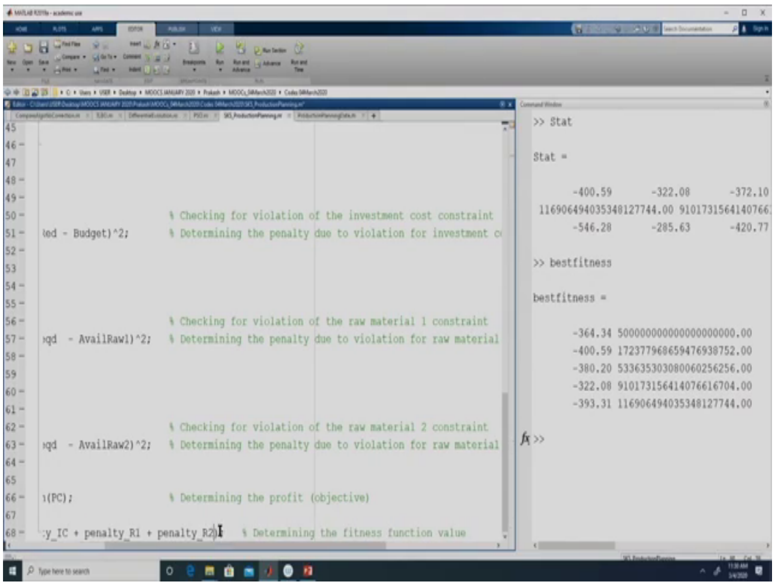
Now, we will not assign a penalty, but we will merely correct the variable. So, the corrected value of the variable is 0 because, 0 we know for sure does not violate the domain constraint right. So, we assign it to 0 as discussed earlier right you can also assign it the value of l. As discussed earlier you could even choose it to assign it a value of l right or you can generate a random number and see if the random number is greater than 0.5; you can assign it to 0, if it is less than 0.5; you can assign it to l right. So, here we are the demonstrating one of the scheme, you can try out the other two schemes right.

(Refer Slide Time: 10:58)



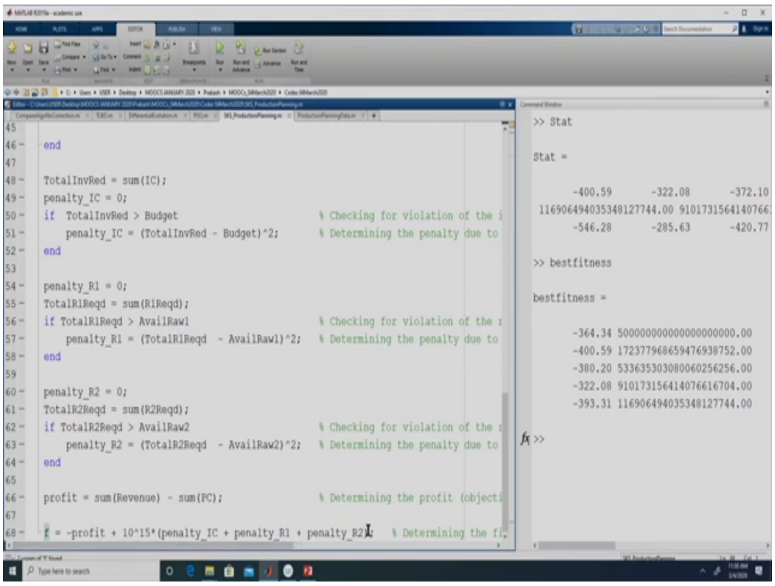
So, this x of j is equal to 0, we do not need to change anything with respect to calculating the investment cost or the raw material requirement of 1 and 2 right.

(Refer Slide Time: 11:11)



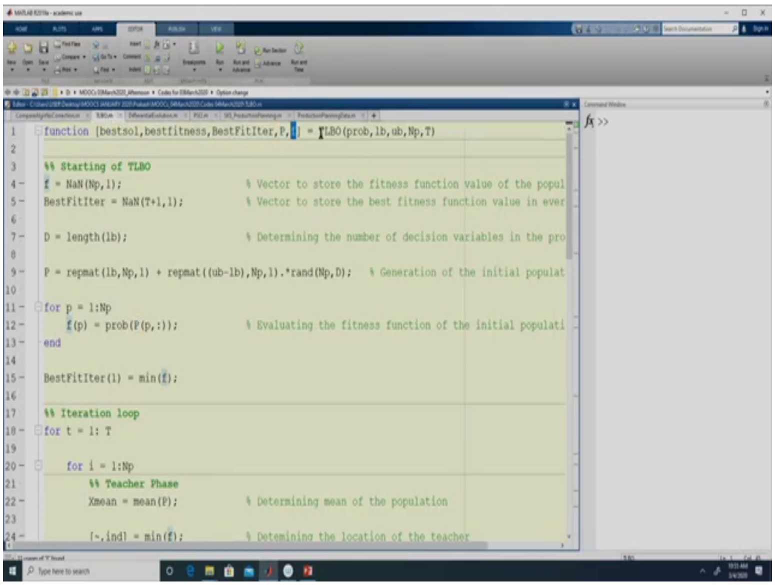
So, everything will remain same except that this term is no longer needed. Because, there is no penalty with respect to violation of the domain constraint, because if any solution violated the domain constraint, we corrected it right.

(Refer Slide Time: 11:23)



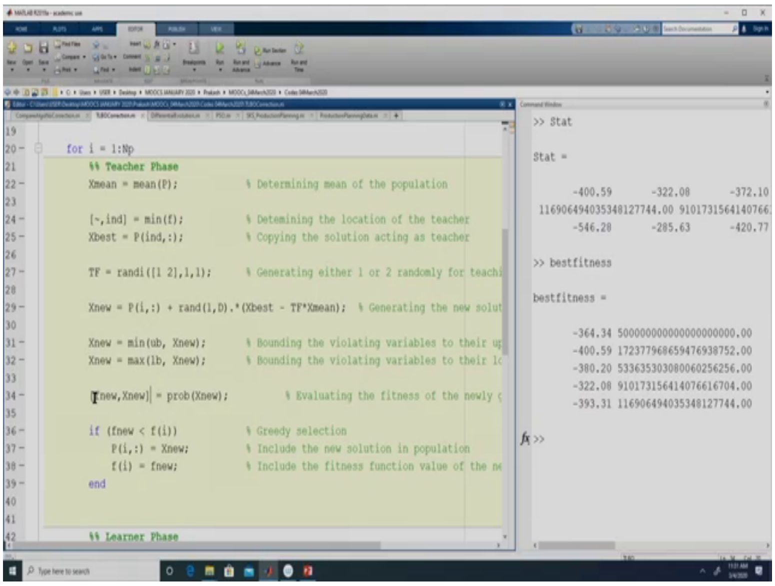
So, over here we have removed that thing right. So, now, our objective function is compatible.

(Refer Slide Time: 11:29)



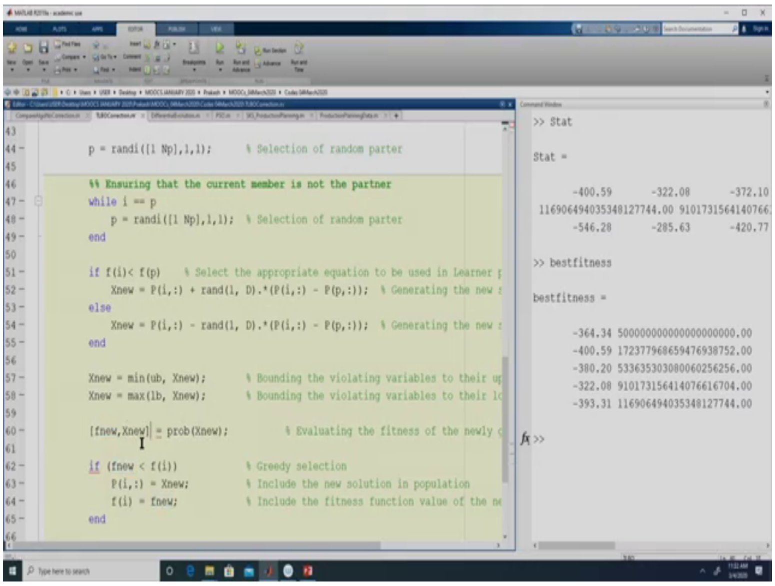
But if you look at our algorithms our algorithms were designed in such a way that it can receive only the fitness function right. So, what we will do is; we will make a copy of this. So, that we can compare both the approaches right. So, let me say this is TLBO correction right and over here, we will receive not only the fitness function, but we will also replace the member which we sent right. So, the member which we are sending is the p th row of the population right. So, that we are replacing with the solution provided by the problem right.

(Refer Slide Time: 12:09)



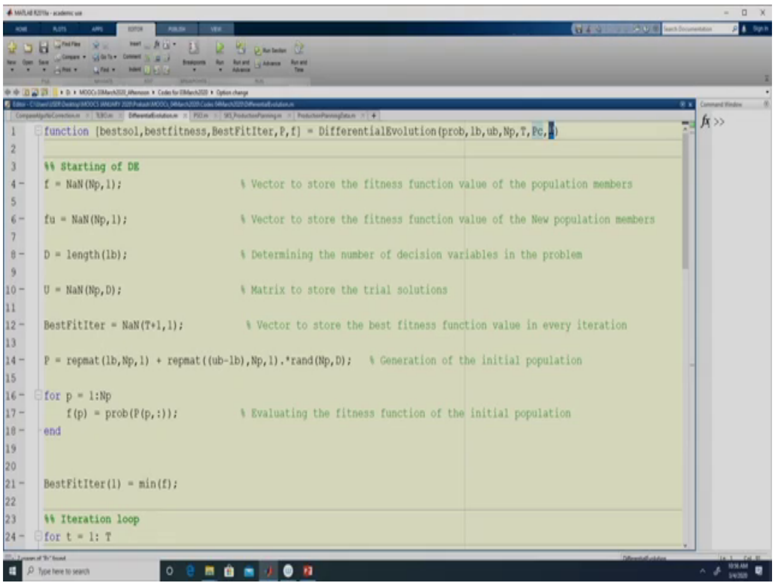
So, this line has become compatible now right. In two other places we would be calling the objective function; one is after generating the new solution in teacher phase right, so over here. So, here we will say f new comma Xnew right. So, we now receive not only the fitness function, but also the corrected solution right. So, if the solution is already within the domain nothing is going to change right, we will not be modifying the correction approach does not help, but if the solution is violating the domain constraint it is corrected in such a way that the solution no longer violates the domain constraint right.

(Refer Slide Time: 12:41)



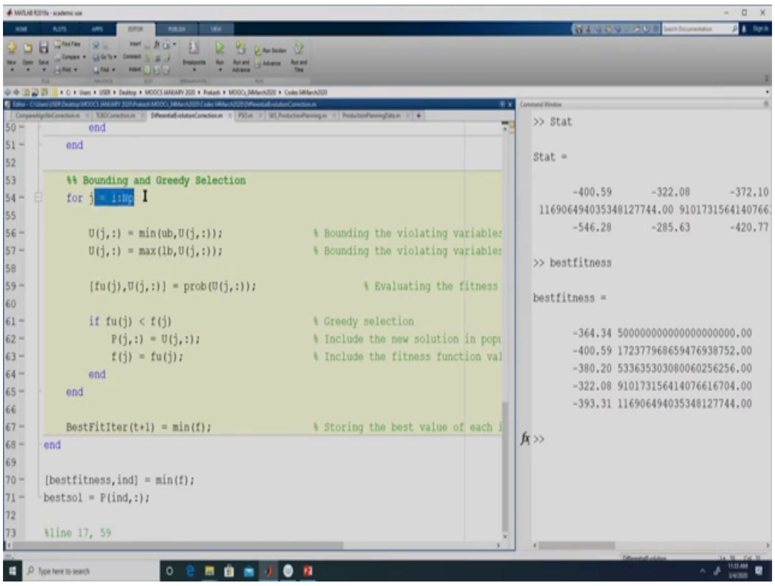
So, similarly over here f new comma Xnew right. So, this we have converted for TLBO right.

(Refer Slide Time: 12:57)



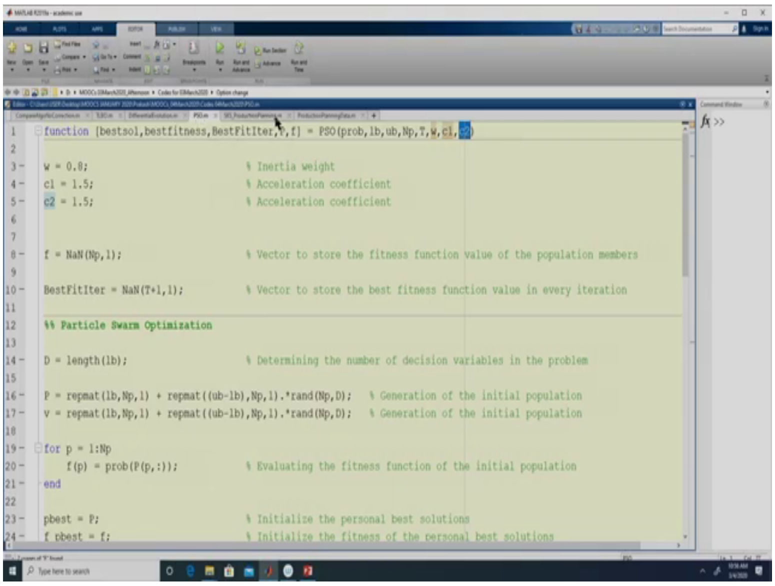
Similarly, let us convert for differential evolution right. So, this is differential evolution, it employs correction approach right. So, remember we are able to do this because we have complete control over the algorithm as well as the problem right. So, since the algorithms we had coded it ourself since we know this algorithms; we can employ this correction approach, we can easily employ this correction approach. So, over here we are again receiving the p th member of the population right and then we will be evaluating the objective function over here right.

(Refer Slide Time: 13:29)



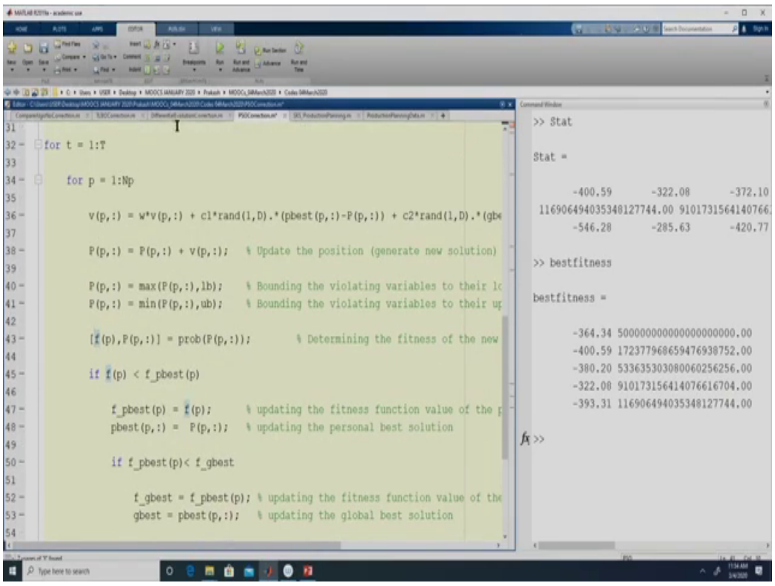
So, the solution which we are sending is the j th row of the variable u right. So, what we will get back is another solution. So, the newly obtain solution is plugged in the same position right. So, this we have done for differential evolution. So, differential evolution; we call the objective function only twice once is for the initial population and then once when we have calculated all the solutions right. So, this is inside a loop, so that way we are calling it np times.

(Refer Slide Time: 14:01)



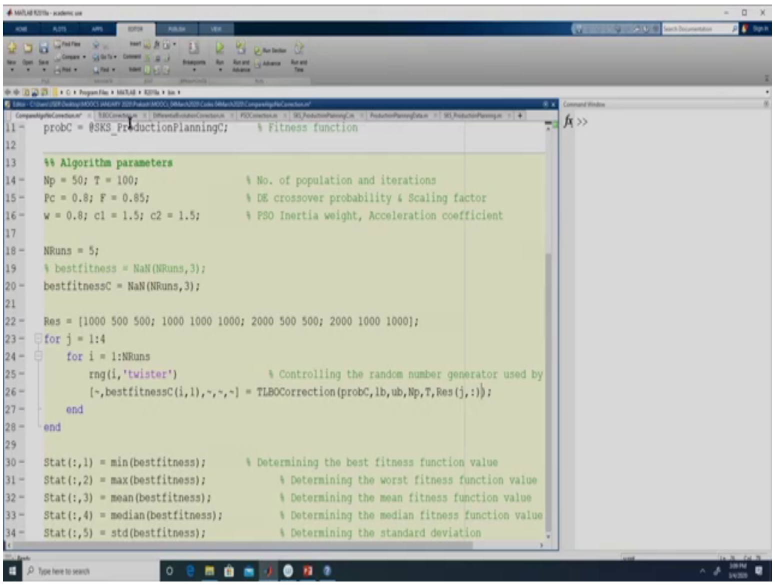
For particle swarm optimization also, we will be calling it only twice. One is during the initial phase. So, we need to make a copy of this right. So, now here we will have the new solution similarly, we will be using the objective function over here right.

(Refer Slide Time: 14:24)

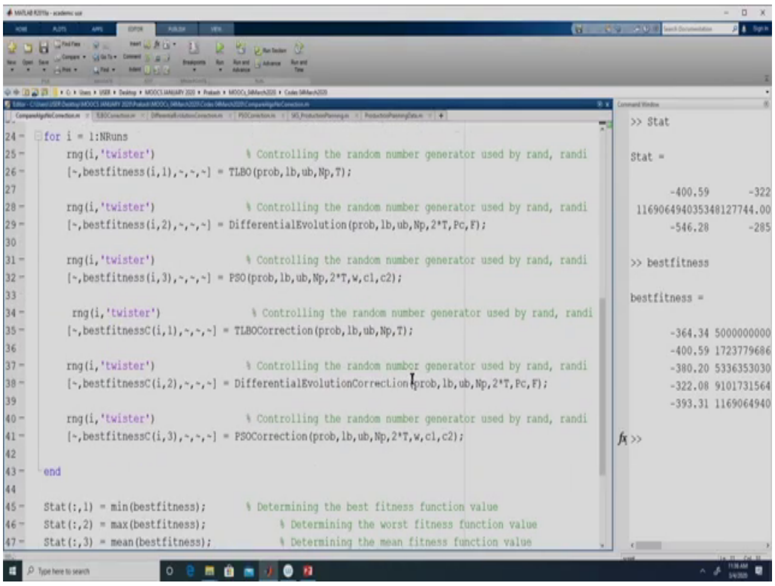


So, over here also it is the p th member of all columns. So, with these three files right, T L B O correction, differential evolution correction, PSO correction. We have converted the algorithm such that it is compatible to be used with the correction approach right. So, let us go back to this right we will repeat this.

(Refer Slide Time: 14:50)



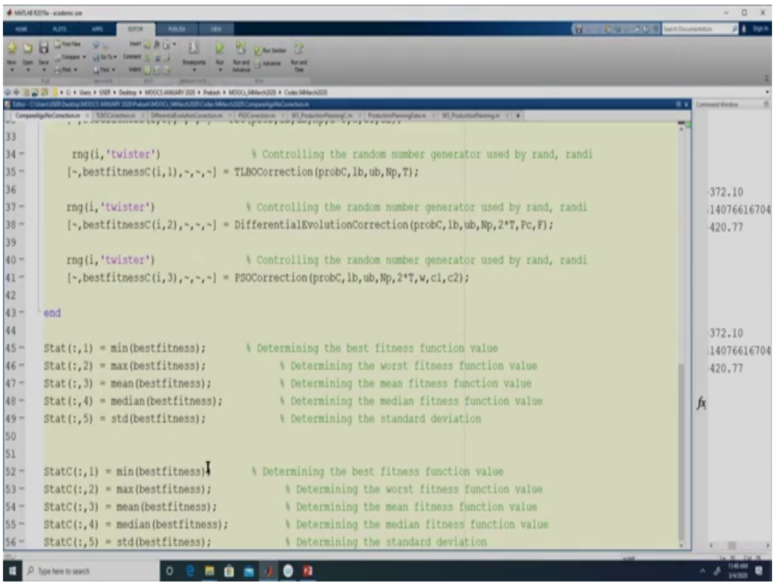
(Refer Slide Time: 14:55)



So, that we can compare the correction approach and the original approach right. So, here we need to call this correction. So, this correction has to be given over here right and again over here right.

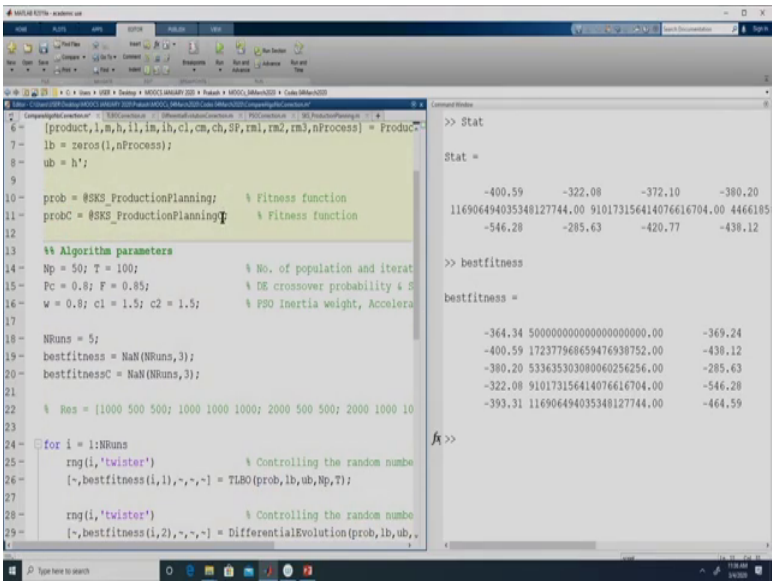
So, let me use another variable for this thing, best fitness c right and similarly, let me define this over here right. So, this best fitness will give us the performance of the algorithms without correction whereas, best fitness c would give us the performance of the algorithm with correction. We can also repeat these lines right.

(Refer Slide Time: 15:30)



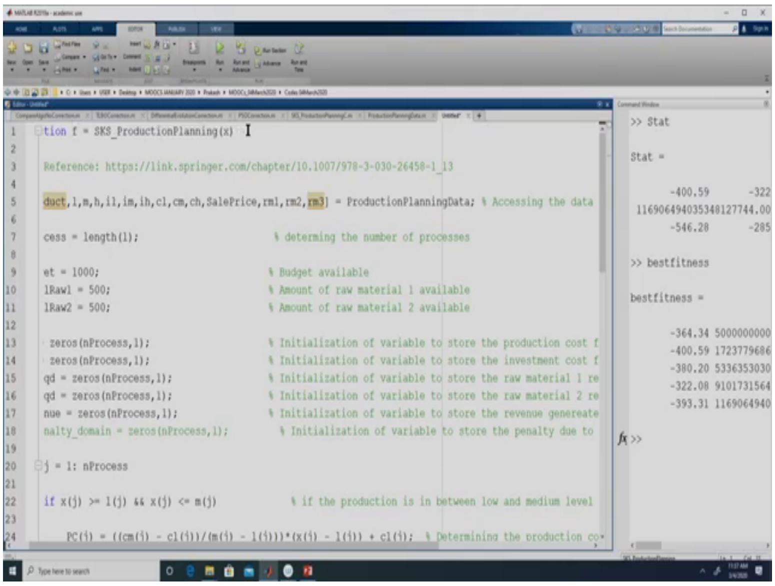
So, let me use this variable. So, again this is a bit crude way of doing it. If you have sufficient coding skills, you can do this same thing in a much better way right. So, now, what we are doing is we are solving the same production planning problem with three different algorithms. TLBO differential evolution and particle swarm optimization with correction and without correction right. So, these are without correction. So, the first three are without correction the last three are with correction right.

(Refer Slide Time: 16:06)



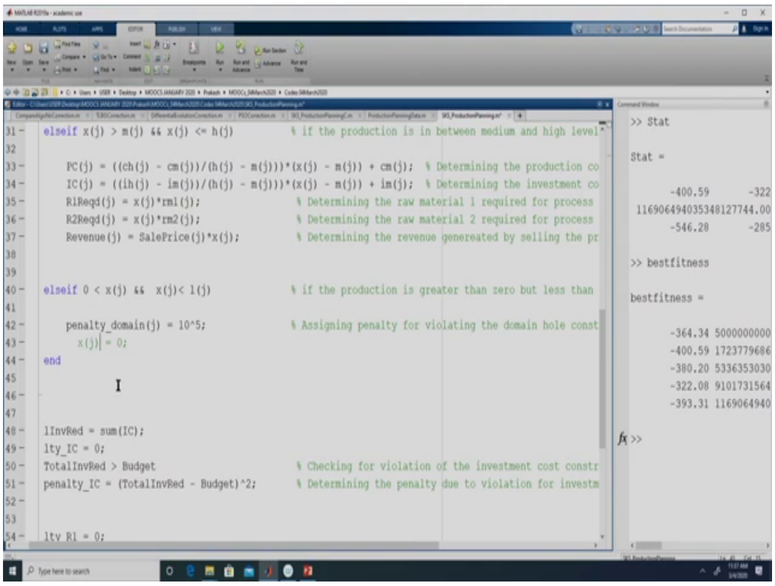
So, the fitness function also we need to have it in two forms right. So, one is prob another one is prob C. So, one employs correction another one does not employ correction right. So, for those functions which do not employ corrections, we do not need to make any changes wherever we are employing correction we need to pass the appropriate file right. And then this production planning we need to have us production planning c. So, this is production planning correction approach right.

(Refer Slide Time: 16:47)

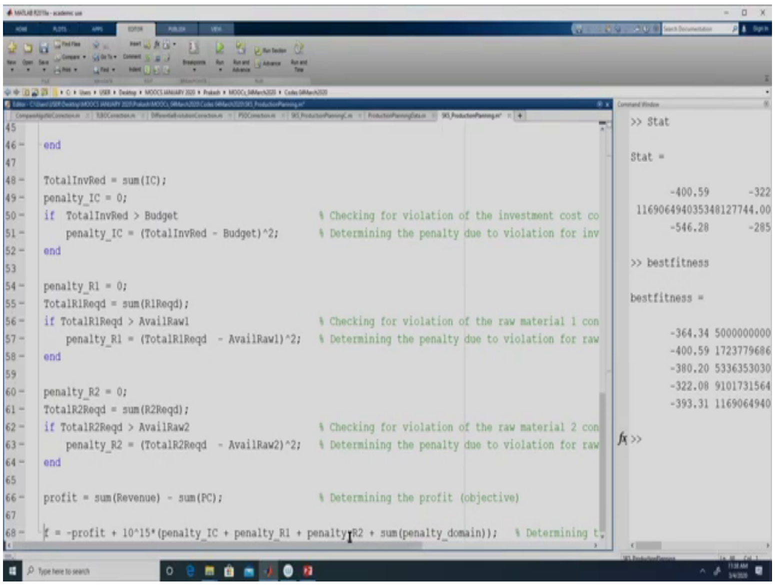


So, for without correction we will have this only f is written right without correction right and so this line would be active right.

(Refer Slide Time: 17:01)



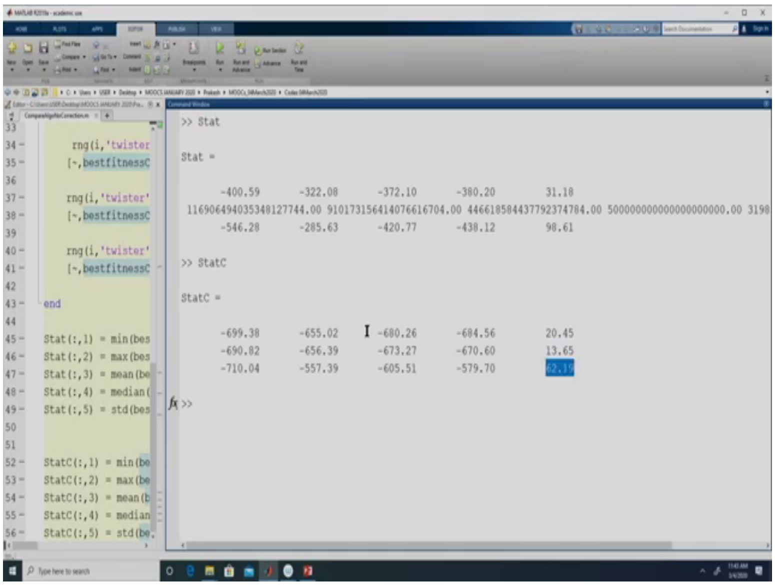
(Refer Slide Time: 17:04)



So, this is to be converted plus sum of penalty underscore domain and we can also uncomment this line right. So, what we have now is two objective function file; one employs the correction approach right f comma X right. The other one returns only the fitness function value f right. Over here in this SKS underscore production planning c, we employ the correction approach right. No penalty is being assigned and the variable value is converted to 0 and that set of modified decision variable is passed back to the algorithm and we do not have any penalty over here right.

Whereas this production planning is the same file which we have been using in the previous session right. So, we assign a penalty over here and we add the penalty right. So, this is the violation with respect to every decision variable. We sum it up and add it to total fitness function right.

(Refer Slide Time: 18:15)



So, now if we execute this, let us see if we are able to execute it without any error. So, now, let us look at both the variables stat and stat C right. So, remember stat and statC are identical in nature right. Only difference is for stat we did not employ a correction approach for statC we employed a correction approach.

So, this has to be bestfitness C. So, now, if we execute this, so we can have a look at the variable stat right statC remember the nature of stat and statC is the same right. So, each row indicates an algorithm, the first column indicates the best value determined by the algorithm, the second column indicates the worst value, third one mean the fourth one median and the fifth one is standard deviation.

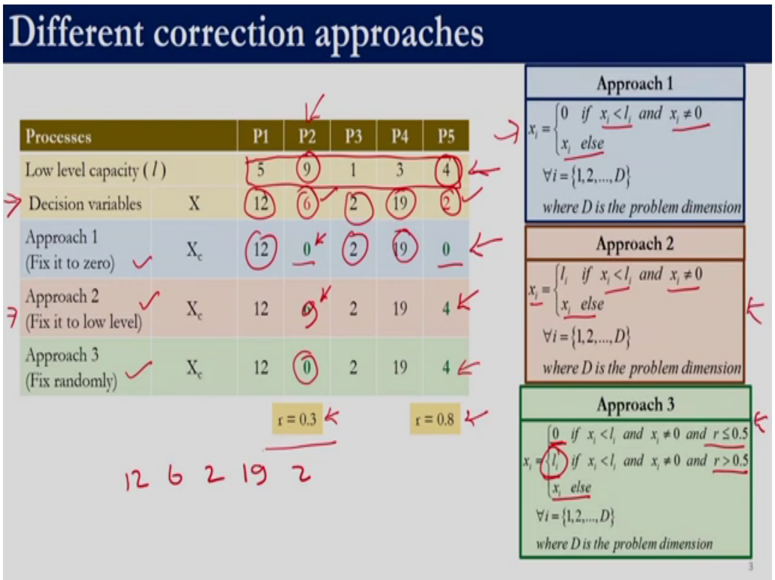
The first row is for TLBO, second row is for differential evolution and the third row is for particle swarm optimization. So, stat is the same set of results which we would have obtained previous right. Whereas, statC, the columns are the same the rows are the same only in this case we employed a correction approach right.

Now, if you see the best value reported by TLBO right was minus 400.59 without correction approach whereas, with correction approach it gives a value minus 699.38 which is significantly better right. Even the best solution was not feasible right. Here now it is able to find a solution which is minus 690.82, particle swarm optimization was previously able to determine a solution of minus 546.28. Now, is able to determine a solution of minus 710 right. So, if you think about it, we employed a very simple procedure right, but that simple procedure has helped us to get significantly better result right.

So, even if the standard deviation also has considerably dropped right. So, from 31.18 and 98.61, it has dropped to 20.45 and 62.19 right. That is the benefit of correction approach right. So, again we need to remember that the correction approach is very problem specific right. So, for this problem, pushing the variable to a value of 0 helped us to satisfy the domain constraint right. So, it is not necessary that the same correction procedure will work in all problems. It is very problem specific that is why we chose to include that correction mechanism in the fitness function evaluation file and not in the algorithm as such right.

We could have even implemented it over there, but then that may or may not work for all problems right, but the correction approach, usually is expected to give better results right than an approach without correction. So, in this case we did not assign any penalty, but we helped algorithm by correcting the solution. So, we need to remember that when we correct a solution; we not only should pass the objective function value, but should also pass the corrected solution, that is one of the correction approach right. So, you can also think of other approaches right.

(Refer Slide Time: 21:17)



So, for example, let us assume this is the low level capacity, this is coming from the data right. So, it is more like we cannot produce anything less than 5, we cannot produce anything less than 9 for second process. We cannot produce anything less than 1 right. We cannot produce anything less than 3 for process 4 and we cannot produce less than 4 for process 5, again 0 is allowed right.

So, if there is a 0 that is not an issue right. So, let us assume that this is the decision variable that we are getting from a metaheuristic technique right. Let us say we get 12, 6, 2, 19, 2 right. These two values are non-zero, but they are also lower than the respective l value right.

So, this is the approach which we discuss right. So, for all those variables which do not satisfy the domain constraint right. We will assign it a value of 0. So, this is the approach which we discussed earlier. So, here what we are doing is if it is not equal to 0 and if it is less than l, we are assigning a value of 0 else we retain the value of X i as given by the metaheuristic technique. So, for example, here the value given by metaheuristic technique is 12, 2 and 19. So, we are not modifying those values right, those values satisfy the domain constraint. So, we do not change them only those things which violate the domain constraint we change it to 0 right.

So, the other approach could be fixing it to low level right. So, in this case what we are doing is; if the value is within the domain, then it is fine right. Otherwise, if it is less than l and if it is non zero, we make it as l right. So, for example, this variable was violating the 6 and 2 are violating. So, right now what we do is; this 6 will be converted to 9. So, this is nine right because the low level value is 9. Previously, in this first approach we converted it to 0, here we will convert it into nine right whereas, for the fifth variable the value that we get from the metaheuristic technique is 2 right, but the l value is 4.

So, here we fix it to 4 right. So, this is another approach. So, we can either employ this approach or we can either employ this approach. So, there is also another approach wherein we fix it randomly right. So, here we generate a random number for every decision variable that is violating the domain constraint. If the random number generated is less than or equal to 0.5, we fix it to 0. If the random number is greater than 0.5, we fix it to corresponding l i values right. Otherwise if it is in the proper domain, we do not need to change it. So, for example, consider the second process right.

So, the low level value is 9, the value that we get is 6 right. So, if we generate a random number and let us say the random number happens to be 0.3 right. So, if r is less than or equal to 0.5, so 0.3 is less than or equal to 0.5 so, we fix it to 0 right. Similarly if we see the fifth variable r is 0.8 right. So, 0.8 is greater than 0.5. So, we fix it to its low level value. So, this two gets converted into. So, what we got from the algorithm is 12, 6, 2, 19 and 2, what we will be returning back is one of these threes right. Either 12, 0, 2, 19, 0 or 12, 9, 2, 19, 4 or 12, 0, 2, 19, 4. Depending upon which approach we take.

So, over here we will demonstrate the approach 1 right. We will implement it on the course that we have you can evaluate both of these approaches right or design your own approach to see if you are able to get better results than what we are discussing over here.

Now the question can be why did we correct, only for the domain constraint, why not for raw material constraint or the investment cost constraint? If you can design a correction mechanism for investment cost as well as raw material constraint you can implement it. So, if you are able to design a correction mechanism to handle investment cost and raw material constraint, you can even choose to implement it and check if it is actually benefiting the algorithm.

In the previous session we had seen solution of the production planning problem using teaching learning based optimization. We did not compare it with other algorithms right. So, first what we will do is; we will solve the same problem right with particle swarm optimization and differential evolution along with TLBO right. So, out of the 5 techniques which we have discussed in this course. Out of the 5 techniques which we discussed in this course we are selecting only 3 right. So, particle swarm optimization, teaching learning based optimization and differential evolution.

The reason for doing this is that for these three algorithms the maximum number of functional evaluation is a deterministic expression right. So, for example, for teaching learning based optimization, it was Np plus 2 Np T right where Np was the population size and T was number of iterations. For particle swarm optimization and differential evolution the expression if you remember it is Np plus Np T this is because in teaching learning based optimization in every iteration for every member we evaluate the fitness function twice whereas, for the other two algorithms particle swarm optimization and differential evolution it was only once.

We want to compare these algorithms with respect to maximum number of fitness function evaluation. Whereas, the other two algorithms; genetic algorithm and artificial bee colony optimization did not have a deterministic expression for the number of maximum functional evaluation right. So, for example, in artificial bee colony optimization depending upon when we encounter the scout phase. The number of fitness function evaluation would change right. So, it can vary from Np plus 2 Np T right. So, that is the minimum number of functional evaluation and the maximum number of functional evaluation is np plus 2 Np T plus t.

Assuming that the scout phase is encountered in every iteration. Similarly, for genetic algorithm also the number of maximum functional evaluation is not a deterministic expression. It depends upon whether an offspring undergoes mutation or not. So, for this particular comparison, we are only taking three algorithms because for all the three algorithms we have coded with respect to maximum number of iterations right. So, if you remember that loop for t is equal to one to t that was there in all three of them right. We did not write it with respect to number of fitness function evaluation right.

So, we can do that right. So, for example, all the 5 codes we can convert it for maximum number of fitness function evaluation that only requires a little bit of coding skill you should be able to do it on your own. We will also upload those codes with respect to the number of fitness function evaluation on the course base. So, for this session we will only restrict with particle swarm optimization, differential evolution and teaching learning based optimization right. So, we are not going to change anything in this function file. So, this is the SKS underscore production planning. So, this function file is what gives us the fitness function value right.

So, we are not going to do anything to this function file. Production planning data is just passing the data to this file which evaluates the fitness function right and these are the three algorithms; teaching learning based optimization, differential evolution and particle swarm optimization. So, all these three are now function files right. So, the TLBO returns the best solution at the end of specified number of iteration. The value of the fitness function corresponding to this best solution. So, bestsol is actually the set of decision variable best fitness is the fitness function value of bestsol BestFitIter gives the convergence curve.

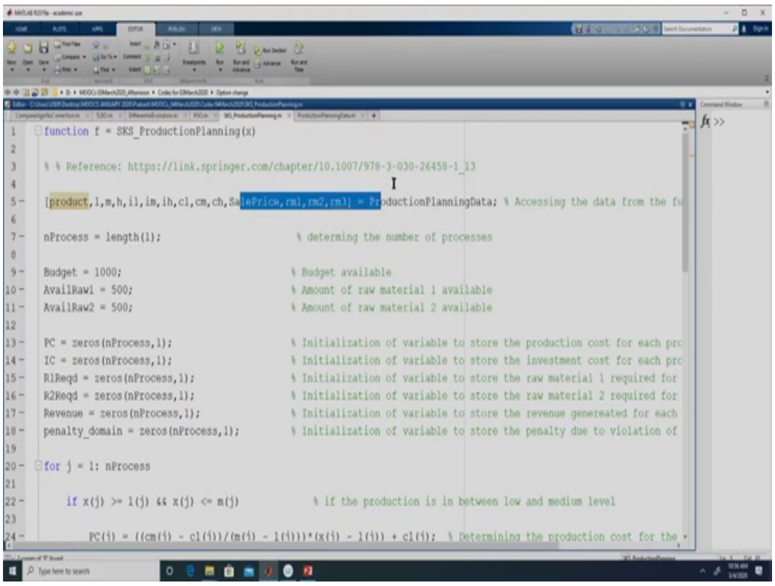
So, it basically tells what is the best fitness function value obtained in every iteration and that includes the first initial population. This p is the final population. So, the dimension of p would be N p cross D where Np is the population size and D is the number of decision variable.

Whereas, f would be Np cross 1. So, for each solution in p, we will have a fitness function value. So, that is what is given in f right. So, the output is same for the three algorithm; bestsol, best fitness, BestFitIter p and f. For TLBO, the input is the fitness function file, the lower bound, the upper bound, population size and the number of iteration right.

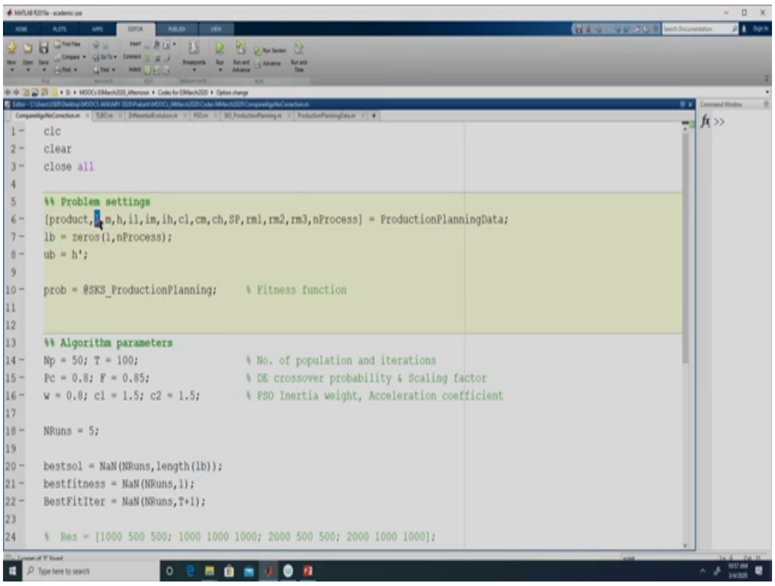
For differential evolution it is the same thing right. In addition we need to provide crossover probability as well as the scaling factor. So, this crossover probability is required in crossover operation and the scaling factor is required in the mutation right.

So, similarly for particle swarm optimization, the output from the algorithm is same right, input is the problem the lower and upper bounds, the population size number of iteration. In addition to that we need to give the inertia weight and two acceleration coefficients c 1 and c 2 right. So, now, we have these three metaheuristic techniques right and we have our problem over here the production planning problem.

(Refer Slide Time: 29:54)



(Refer Slide Time: 30:01)



And this is the data for the production planning problem right. So, this is a script wherein we are going to compare the three algorithms right. So, the first three lines uses to clear the command window the workspace and close any figure if it is open right.

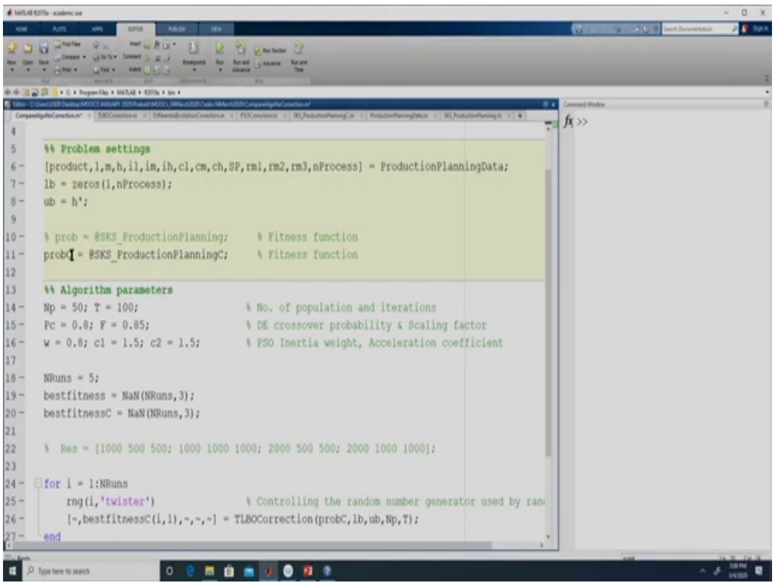
So, here in line 6 we are accessing this function production planning data which contains the data right. So, because we need to provide the upper bound right. Upper bound if you remember it is the h value right the maximum production that is possible right and lower bound is 0. Remember it is not the low level value because if we put it as low level value, then we are enforcing an artificial constraint that each process has to be use right.

So, that constraint is not part of the problem. The problem specifies that either we can choose not to produce or if you decide to produce it has to be l or greater than l and it has to be less than h. So, there is a domain hold. So, that is why we are taking the lower bound as zeros. So, this n process will also tell us the number of processes or in this case the number of decision variable right. So, this is lower bound this is the upper bound. So, prob is a function handle right. So, we are assigning the name of this file SKS underscore production planning right.

To prob right, and then since we are using three algorithms. We need to specify the parameters that we are going to use. So, we are fixing the population size to be 50 and the number of iterations to be 100 right. So, these two values remain constant for all the three algorithms right. In addition to this for TLBO we do not require any other parameter. Whereas, for D E, we require the crossover probability and scaling factor. So, we have fix the crossover probability to 0.8 and the scaling factor to 0.85 right and for particle swarm optimization, we need to provide the inertia weight. We have taken it as 0.8.

The acceleration coefficients c 1 and c 2 as 1.5, 1.5. So, that helps in defining the parameters which are required for the algorithm. So, NRuns is the number of runs. So, so for the purpose of demonstration we are just taking 5 runs right.

(Refer Slide Time: 32:04)



Let me just remove this. We are not going to use these two, right. We are going to use this variable best fitness to store the fitness function value reported by every algorithm right and each run right. So, we are going to have NRuns. So, in this case we are going to have 5 runs right and we have 3 algorithms. So, that is why we have taken 3 columns right.

So, what we are basically trying to do is; the first column we will use it for TLBO, the second column we will use it for differential evolution and third column we will use it for particle swarm optimization of this variable fitness and for each algorithm, we are going to run 5 times right. For each run that we complete, we will populate best fitness with the solution obtain from that algorithm right. So, this is the for loop we are going to compare these three algorithms right based on 5 runs right. So, that is why we have this loop for i is equal to 1 to NRuns right and again we are fixing the seed right.

So, rng of i comma twister. So, that we can reproduce the result right and in line 24, we are solving the problem with TLBO right. So, we are giving the necessary input prob lb, ub, Np and T. So, the solution that we will be returned by TLBO is all of this thing the set of decision variable, the fitness function the convergence curve for that run, the final population and the final fitness function right. So, right now we are interested only in the fitness function value. So, we will do statistical analysis based on that fitness function value. So, that is why we are not receiving any of this four values right

We are not receiving what is the set of decision variable, we are not receiving the values for plotting the convergence curve, we are not receiving the final population and we are also not receiving the fitness function values of the final population right. So, if you are interested you can just specify a variable name and then you can appropriately analyze whatever you wish to. So, this will help us to solve with TLBO right. So, this line again specifies that. So, for differential evolution also we want to control the random numbers right. So, that we can reproduce it.

So, that is why we are including this line right. So, over here as well as in over here for particle swarm optimization. Even if we remove line 26 and line 29, it is correct right. Only thing is that we will no longer be able to reproduce the results of a particular run right. So, for example, let us say we do not have this, let us say we comment these lines right.

And if you want to reproduce the third run of PSO. Only the third run of PSO it is not possible right, but if we have these lines over here right, then we do not need to execute TLBO differential evolution. We can merely say rng of whichever run was the best run comma twister and we can get those results.

So, that is why we are fixing the random number for each of the algorithm right. So, similar to TLBO for differential evolution and particle swarm optimization. We are only interested in the fitness function value of the best solution right. So, here we stored in the first column of best fitness, here we are storing it in the second column, here we are storing it in the third column right.

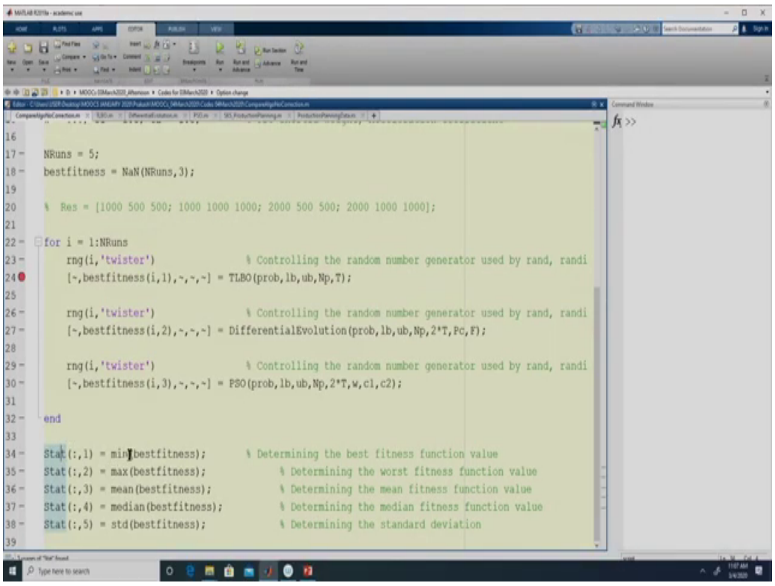
So, the input for differential evolution is problem its lower bound and its upper bound the population size right, the crossover probability and the fitness function. Similarly, for particle swarm optimization these three are with respect to the problem population size, inertia weight and the two acceleration coefficient.

For TLBO, we had given only T iterations right. So, here we have specify T, but for differential evolution and particle swarm optimization, we are providing the number of iterations as 2 into T. So, in this case T we have set as 100 iterations. So, we will run TLBO for 100 iteration, but we will execute TLBO for 100 iteration, but for differential evolution and particle swarm optimization. We will execute it for 200 iterations right.

So, only then the number of fitness function used by TLBO, differential evolution and particle swarm optimization would be identical right. So, remember we want to compare the results of these three algorithms, but the termination criteria cannot be the number of iterations, then TLBO will be using more number of fitness function evaluation compared to particle swarm optimization and differential evolution right.

In order to avoid that we are executing differential evolution as well as particle swarm optimization for twice the number of iterations as compared to teaching learning based optimization right. So, this will complete execution of all the runs right.

(Refer Slide Time: 36:44)



So, here we have the statistical analysis right. We are finding the best fitness of all the three algorithms right. So, when we do min of best fitness it will give us a column vector right. So, the first column of the variable stat right. So, this is just a variable name, we are saying that the first column of the variable stat is the best value obtained by each of the algorithm right.

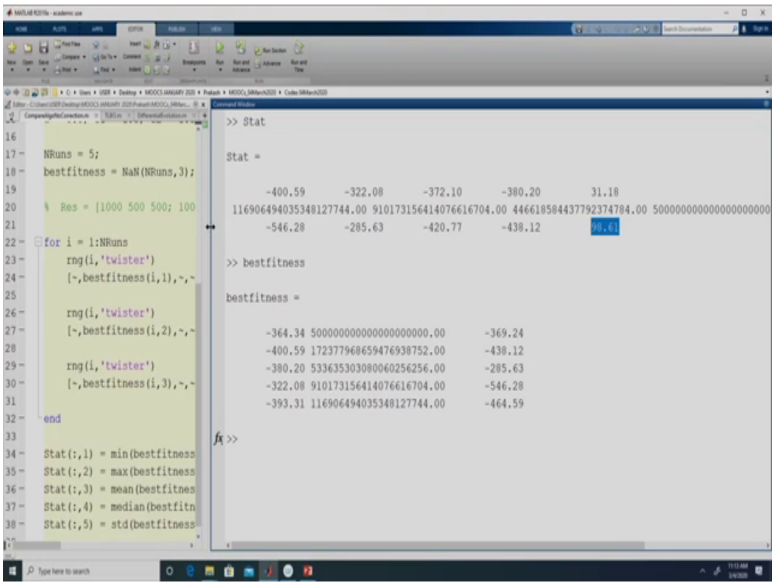
So, that will have three rows right and the first column will contain the best objective function value. The second column similarly for all the three algorithms we will contain the worst fitness function value. Remember all these three algorithms have been written for minimization right. So, this is max of best fitness and then similar to what we did previously mean for each algorithm, median for each algorithm, standard deviation for each algorithm considering the 5 runs which we are executing right. So, let us see what happens if we execute this right. So, here I had kept a break point right let me remove that breakpoint. So, it takes a little while to solve.

So, now, we have the result correct. So, the first toe is for teaching learning based optimization right. So, the best value obtained by TLBO is minus 400, because it is in the first column right. So, the first column gives the best solution with respect to each of the three algorithm, the second column gives us the worst solution with respect to each of the three algorithm, the third column gives us mean fourth median and the fifth one gives us standard deviation right. So, remember our previous discussion that for the current problem the fitness function will have to have a negative value for the solution to be even feasible.

So, in this case we can see that differential evolution is not able to obtain any feasible solution right. Because, the best solution obtained by differential evolution in all of the 5 runs right is positive value; that means, there is some penalty which indicates that the solution is not feasible with respect to the best solution obtained by the three algorithm. The best algorithm is particle swarm optimization because it discovers a solution of minus 546.28. Whereas, TLBO is able to determine solution which has a fitness of minus 400.59 right.

So, if one way to implement a production plan right, they would prefer to choose the solution given by particle swarm optimization right. So, similarly if we compare the worst value, the worst which is determined by TLBO is better than what is determined by PSO right.

(Refer Slide Time: 39:21)



So, we can actually look at this best fitness. So, these are the results of the 3 run. The first column indicates TLBO, the second column indicates D, the third column indicates particle swarm optimization right. So, in this case the best value was minus 400.59 and the worst is minus 322.08. Whereas the worst in this is minus 285.63 right though particle swarm optimization discovers a better solution than teaching learning based optimization for this set of settings.

But the standard deviation across the 5 runs of TLBO is less than particle swarm optimization, but the standard deviation of TLBO is less than that of particle swarm optimization right. Again, remember this set of result is only for the set of parameters right. So, if we change this parameters of differential evolution, the solution might even improve right.

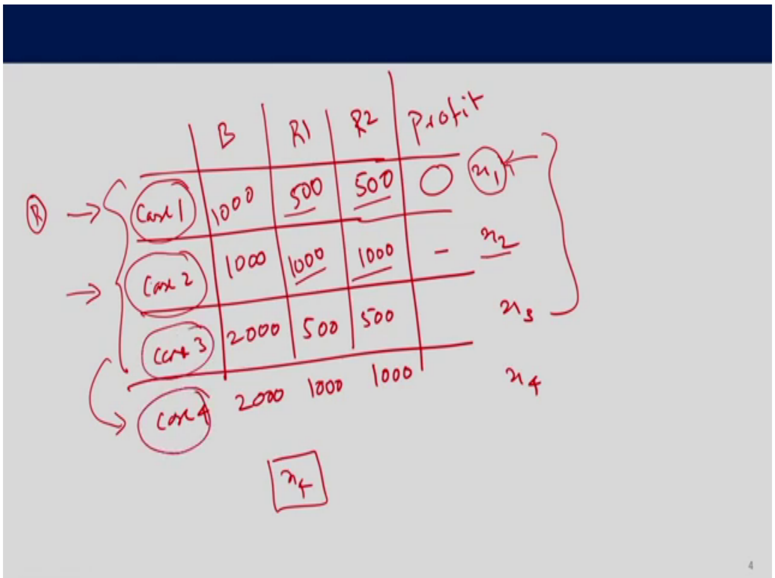
So, for an arbitrary problem it is not possible to say what is the best value of Pc, F, w, c1 and c 2 right. So, when we execute an algorithm we need to try with multiple values of this tuning parameter right. Though we have shown your statistical analysis right it is very preliminary right because we are also supposed to change this parameters and then we are also supposed to analyze the impact of these user defined parameters.

In this case we showed you preliminary comparison of these three algorithms right. So, if we have code of ga and abc in which we can specify the number of functional evaluation. The algorithm would stop after utilizing that many number of functional evaluation. We can also include them over here.

Right now, we know how many functional evaluation we took for TLBO all right. So, that is Np plus 2 Np T right. So, that value can be calculated and it can be given to that specific code of genetic algorithm and artificial bee colony optimization. So, in that case all the five algorithms utilize the same number of fitness function evaluation and similar statistical analysis can be performed right. So, that concludes the comparison of algorithms for this production planning problem.

Many times what happens is we want to solve the same problem with different set of data. For example, in the case study which we were discussing so far, the budget value was 1000 and the raw material 1 that is available was 500, the raw material 2 that was available is 500 right.

(Refer Slide Time: 41:48)



So, this is what we were having, let me say this as case 1 right. So, for case 1; the budget that is available is 1000 right and the raw material 1 that is available is 500 and the raw material 2 that is available is 500 right. So, for this let us say we get some profit right. So, depending upon which algorithm we are working with, we will get some value of profit over here right.

So, same problem we want to solve, let us say with budget as 1000 right. And the amount of raw material 1 that is available is not 500, but it is 1000, the amount of raw material 2 that is available is also 1000. So, we will get some profit over here let us term this profit that we get over here as x 1 and the profit that we get over here in case 2 as x 2 right. So, if you think about it, here we have only increase the resources right. Some 500 to 1000, we have only increase the resources. So, any solution which is feasible over here is also feasible for this one right. So, because we are only relaxing the problem right.

So, we should at least get a solution which has a profit of x 1. We can get a solution which may have a better profit than x 1, but it should not happen that x 2 is inferior to x 1. Because x 2 is the solution for a relaxed problem right. So, similarly for this study we have two more cases right. Where in the budget is increased to 2000, the raw material is at 500 right and the 4th case is the budget available is 2000 and the raw material that is available is 1000. So, let us call this as profit as x 3 and x 4 right. So, over here if we see x 4 cannot be inferior to any other value right. Because x 4 is a relaxed version of any of the other three problem.

So, any solution which satisfies these three cases would also satisfy this. We should not have a solution whose profit is inferior to any of this three. Now, the question is how do we implement this? Right. So, remember for each of this case, we need to run multiple times right because case 1 is a individual optimization problem, case 2 is an individual optimization problem similarly, case 3 and case 4.

So, now, we will see how to execute such a problem. Over here if we see the first figure shows the approach without correction. The second figure shows the approach with correction right. So, whatever we are discussing holds true for both of them right. If you look at our implementation we had those metaheuristic technique that was sending the value of X and it was receiving the value of f right.

And we had the script file right. So, through the script file we were passing the algorithm parameter and problem details right. So, every time what we were getting from the solution of metaheuristic technique was the best solution its corresponding objective function, the final population, the fitness function corresponding to the final population and the best fitness function value obtained in every iteration, that was what we use for plotting convergence curve. So, this is what we were receiving from the metaheuristic technique right. So, now, what we can do is; when we pass this problem details, we can also pass the values of budget, the amount of raw material 1, the amount of raw material 2, that is required.

So, previously we are only passing the upper bound, lower bound and name of the fitness function file right. So, this B, R 1, R 2 can be pass to the metaheuristic technique right, but this B, R 1, R 2 is actually required in this fitness function right. So, we will have to now change our metaheuristic technique rightto receive this input from the script file right and as well as transfer that data to the optimization problem. So, now, we will say that what we get from the metaheuristic technique is x as well as we will say what are the resources right. So, this three put together let me call it as variable Res. So, this Res can be pass to the optimization problem.

So, the fitness function which could previously receive only X should now also be capable of receiving Res right. So, what we are discussing here is more with respect to the implementation right and not necessarily with respect to optimization But, since implementation becomes a part of the optimization study that is why we are discussing it over here.

So, this is also true for even if we have correction right. So, we will have to pass B, R 1, R 2 every time right. So, we have 4 sets; sofirst time we will pass the first set, second time we will pass the second set. So, that will pass 4 times and this metaheuristic technique along with the decision variable will pass the parameters and those parameters will be utilize in fitness function and the corrected fitness value and corrected solution will be returned back to the metaheuristic technique right.

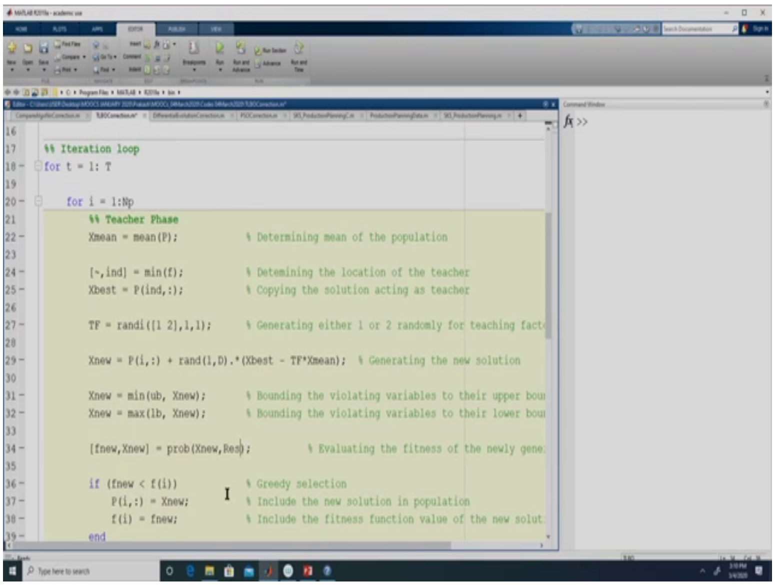
So, this is one way of doing it. Otherwise in MATLAB a direct communication can be establish between these two files right using a global variable. So, we will not use that approach. So, this is the file that we were working with right. So, here let us get rid of things that we do not want. So, we do not want this. Now, we are working only with the correction approach right. Let me uncomment this right. So, these are the 4 cases that for the first time we need to pass 1000, 500, 500, for the second time we need to pass 1000, 1000, 1000, for the third time we need to pass this value and the for the fourth time we need to pass this value. So, every case is a row now right.

And we have four such problems. So, we will put another loop over here and now in addition to this we need to pass Res, that is the name of the variable in which we have defined the 4 cases right. The j th row we need to pass and all the columns right. So, these 3 are details with respect to the problem, these 2 are details with respect to the algorithm right and again this is a detail with respect to the problem.

So, so far we were saying the input to the algorithm is just fitness function lower bound and upper bound, but we can also pass other parameters. But this will not be used in the algorithm. Since algorithm happens to be in between our script file and the fitness function file, we are passing it to the algorithm and subsequently the algorithm will pass it on to the fitness function right.

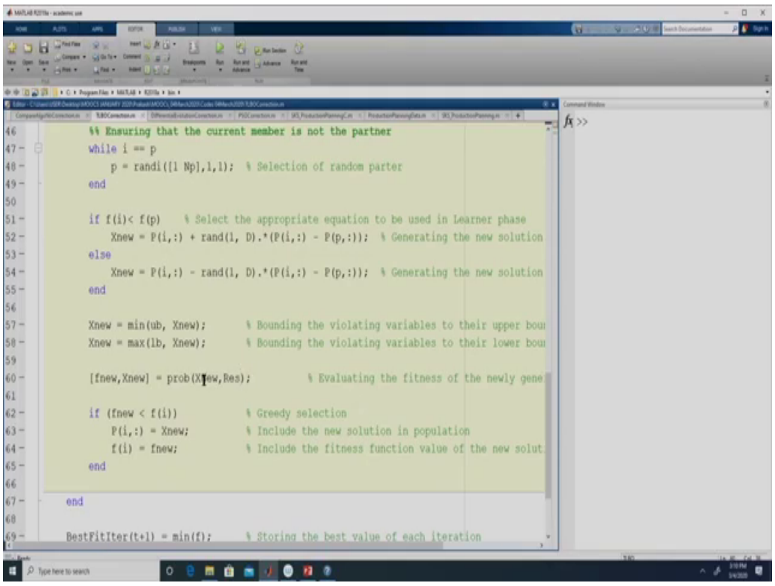
So, over here we have change the algorithm right. So, our algorithm is over here. So, this should be capable of receiving that variable right and wherever we call the objective function, we need to pass this value.

(Refer Slide Time: 48:01)



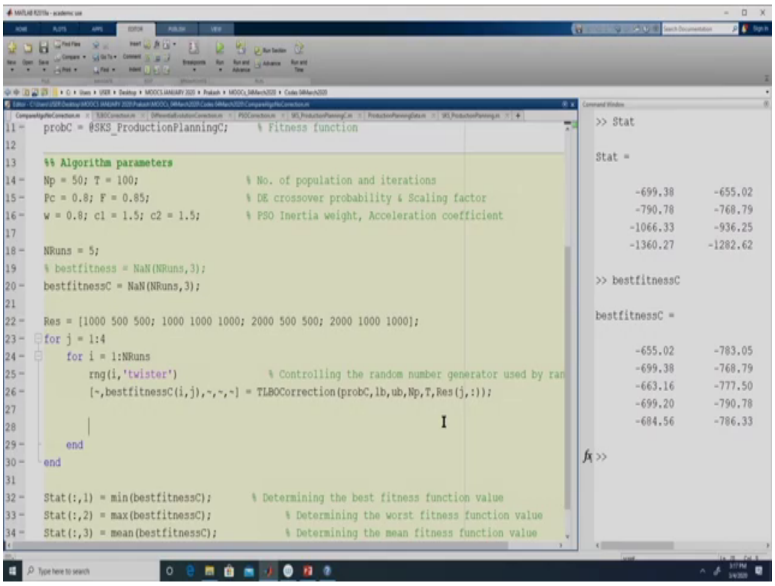
So, we call the objective function here and then over here in the teacher phase and in the student phase right.

(Refer Slide Time: 48:13)



So, Res, we are not actually using it right its Res; if you see we are not actually using it in the algorithm. So, going back to this problem right. So, over here we will get Res right.

(Refer Slide Time: 48:26)

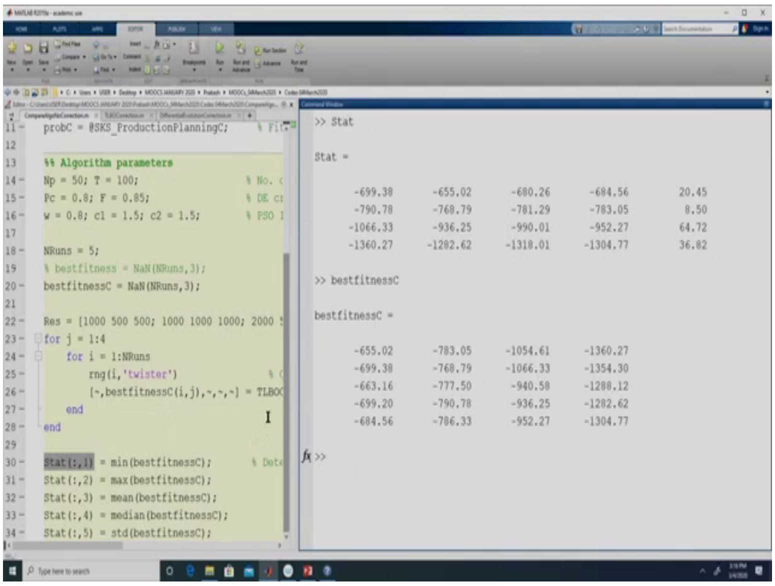


So, these values are to come from the variables. So, we have written in such a way that the first value contains the budget right. The second value contains the amount of raw material 1, that is available and the third value contains the amount of raw material 2 that is available to us.

So, now from the script file we can run all this 4 cases. Now, we have defined this outer loop right. So, this outer loop we will take care of the four problems that we have and for each of the problem we need to implement 10 runs right and this is the statistical analysis part which we are doing over here right. So, right now we are not saving the best solution or the values for the convergence curve or the final population. We are only storing the fitness function value right; so best fitness C. So, the first time it will be the first column right. So, instead of this 1, we need to give j. So, that we will get 4 columns right and we will have 5 rows.

So, each row corresponds to a particular run and each column will correspond to these cases right. So, for the first case, second case, third case and the fourth case right. So, if we execute this it will take a little while right. So, now, we have the command prompt. So, we can look at this variable stat right.

(Refer Slide Time: 49:43)

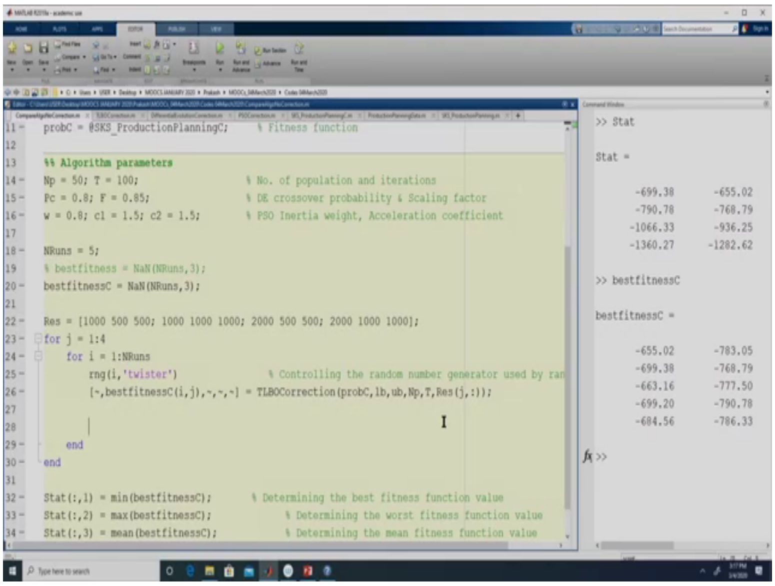


So, each row indicates a particular run. So, if we look at best fitness right. So, these are the 4 cases which we have right, case 1, case 2, case 3, case 4. So, the first column of best fitness c is case 1. So, for the case 1; we see that the values in 5 run are the 5 rows of the first column. Similarly, for the second problem right. Second problem in the sense with the budget of 1000, raw material 1 available being 1000 and the amount of raw material 2 available being 1000.

So, these are the results of 5 run. Similarly, the third case and the fourth case right. Now, if we implement a statistical analysis of this. Let us say the best value that we obtain is for case 1 is minus 699.38 right. So, that is what is over here right. So, the first column indicates the best values, because over here if you see stats the first column indicates the best value right.

So, the third value this minus 1066.33 corresponds to the best value of the 5 runs for case 3 and similarly for case 4 right. The second column of stat indicates the worst value, the third column indicates the mean, the fourth one indicates the median and the fifth one indicates the standard deviation right. So, here we have shown you only for TLBO right. So, even in that case we get a table right.

(Refer Slide Time: 51:10)



So, what you can do is in addition to TLBO you can also run the problem with particle swarm optimization, differential evolution, genetic algorithm and artificial bee colony optimization. This is how we can run the same problem with different set of data for any number of specified runs with a particular algorithm. With that we will conclude the session.

Thank you.