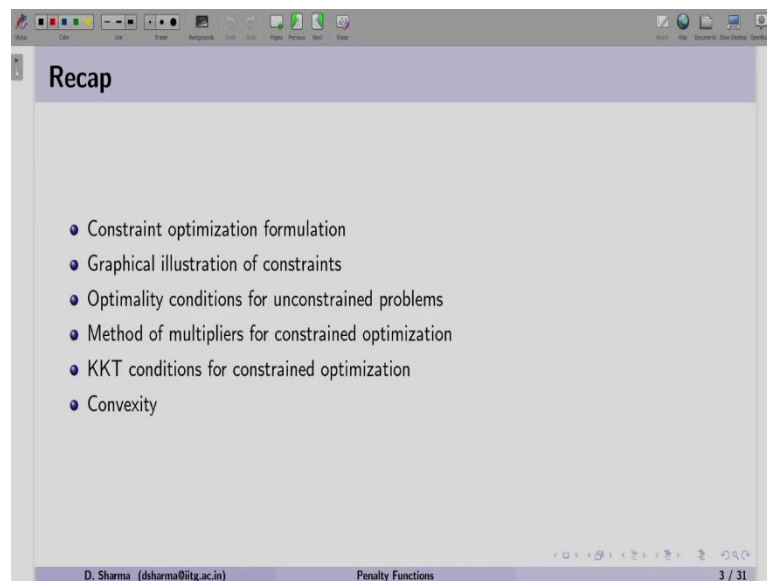


Evolutionary Computation for Single and Multi-Objective Optimization
Dr. Deepak Sharma
Department of Mechanical Engineering
Indian Institute of Technology, Guwahati

Lecture - 13
Penalty Function Methods for Evolutionary Computing Techniques

Welcome to the session on Penalty Function Methods for Evolutionary Computation Technique. In this particular session we will be going through the introduction of the penalty function methods there after we will discuss some of the methods, such as static penalty, death penalty, dynamic penalty method and so on. There after we will perform some hand calculations. So, that we will understand how this penalty function method works and finally, we will conclude this session.

(Refer Slide Time: 01:20)



So, let us start with the recap in the previous session we started with the constraint optimization formulation in which our objective function was subjected to inequality constraint, equality constraints and the variable bound. So, for this particular problem or the constraint optimization problems we have gone through the visualization of different types of spaces that will be made by the constraints.

So, we had continuous search space followed by we can have discontinuous search space. We have also gone through the different types of constraint such as linear, non-linear as well as equality and inequality constraints. After that we started with the optimality

condition for unconstrained optimization because that laid down the basis for finding the KKT condition or the optimality condition for constrained optimization.

We used Taylor series function, Taylor series to find the sufficient optimality condition for an unconstrained problem, which suggest that at a stationary point say \bar{x} gradient of the function should be 0. There after we have gone through the sufficient condition in this sufficient condition. We come up with the Hessian matrix and as and when this Hessian matrix is positive definite we say that the stationary point is a local minima for the function.

Similarly, we also conclude about the maximization when the Hessian of the matrix is positive definite, if the matrix is not positive not negative then we have to say the point is a saddle point. Thereafter we discussed about the method of multipliers, in that method of multipliers we divided that section into two part.

First part was that we took an objective function with equality constraint, since it was easy we came up with the optimality condition via adding the Lagrangian multiplier or the multiplier with the constraint. So, each constraint with Lagrangian multiplier was added into the objective function and one unconstrained function was made.

After finding the optimality condition with respect to the decision variable of the problem and with respect to the Lagrangian multiplier we got the conditions. Similarly when we took the objective function with respect to the inequality constraint we first converted the inequality constraint into equality constraint with the help of slack variable and then we followed the similar procedure.

Finally, using the basis of method of multiplier we came up with the KKT condition which is referred as Karush Kuhn Tucker conditions these conditions we have written for the one constraint optimization problem, that involves both inequality as well as equality constraint. So, apart from all those conditions we also had complimentary slackness conditions and the Lagrangian multiplier with respect to the inequality constraint should be greater than 0.

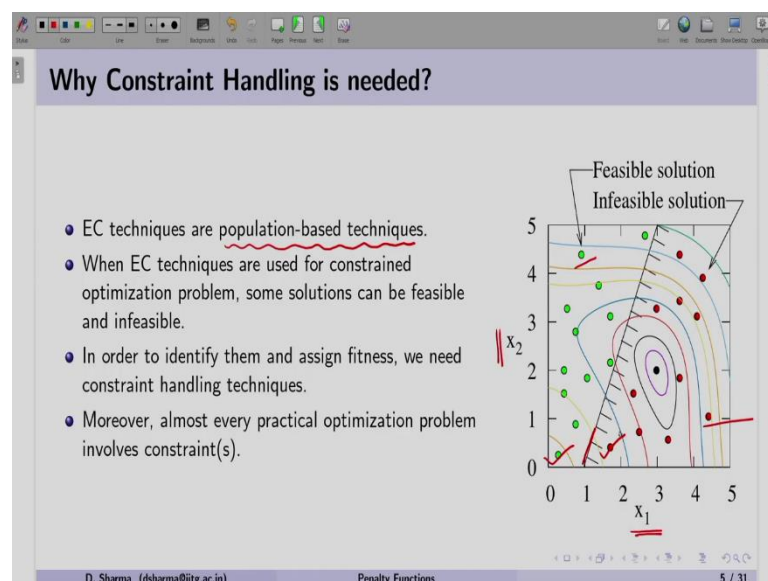
In that particular KKT condition we found that the point which is satisfying all the KKT conditions it is a likely candidate for the optimum. However, if the point is not satisfying

any of the KKT condition we can always conclude that the point cannot be an optimal solution.

Thereafter we have gone through the necessary condition and the sufficient condition and in that sufficient condition again the function should be convex, the inequality, active inequality constraints should be concave and we should have linear equality constraint. For that particular point if any point satisfying the KKT condition in that scenario we can always say that a point is a local minima.

And the convexity of the function we defined with the help of a relationship and also if we find the Hessian of the matrix at a given point looking at the positive definite of the matrix, we can conclude whether the function is convex or a not. So, that a mathematical understanding on the optimality of unconstrained, as well as constrained optimization. Now, we will discuss the penalty function method. So, let us begin with the introduction here.

(Refer Slide Time: 06:25)



So, here we can see first of all we have to find why the constraint handling is needed with evolutionary computation technique. So, as we can see that EC techniques are population-based technique meaning that, we will be working with many of the points. In this case you can see on the figure on the right hand side that we are using multiple solution or the points as we used in EC techniques.

Now looking at this figure we have box constraint based on X_1 and X_2 . Similarly we also have the constraint as you can see here. So, because of this constraint few solutions are feasible, which are represented by the green dots and few solutions are infeasible, which are shown in the red dots.

Now, the question comes that if we are working on multiple points how we can differentiate the feasible solution versus infeasible solutions. Therefore, we have to need some way to find out that who are feasible solutions and who are infeasible solution. And accordingly we should assign a fitness to those solution. So, that is why we need constraint handling

Second important point is that we know that most of the practical optimization problem involve constraint. It can be one constraint or it can be multiple constraint, since we want to solve those practical optimization problem we want to know what is the optimum solution for that. So, we need constraint handling techniques with EC techniques.

(Refer Slide Time: 08:13)

Types of Constraint Handling Techniques

- Carlos A Coello Coello, "Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: a survey of the state of the art," Computer Methods in Applied Mechanics and Engineering, Volume 191, Issues 1112, 2002, Pages 1245-1287.

Types

- ✓ Penalty Function Methods
- ✓ Special representations and operators
- ✓ Separation of constraints and objectives
- ✓ Hybrid Methods

D. Sharma (dsharma@iitg.ac.in) Penalty Functions 6 / 31

So, there are different types of constraint handling technique here this is one of the good paper given by professor Coello, although it is quite old which was published in 2002, but this particular paper has almost has covered most of the constraint handling techniques that can be used with EC techniques.

So, on the basis of that paper we can have different kinds of constraint handling methods. For example, we can have penalty based methods. We can have special representation and operators so, that we can find feasible solutions. Similarly we can have separation of constraint and objectives. And finally, we can also come up with the hybrid methods.

So, since there are various methods available some methods we can use it directly into our formulation. So, the objective here again is how we can find a fitness and then how we can differentiate the feasible solution verses the infeasible solution. In this particular session we will be focusing on Penalty Function Methods.

(Refer Slide Time: 09:34)

Penalty Function Methods

- These methods are found be simple and most popular methods for handling constraints.
- If a solution violates any constraint, the solution is penalized by adding penalty with the objective function.
- The constraint problem is transformed to an unconstrained problems by adding penalty term of each constraint violation to the objective function value.

Penalty Function Methods

- Interior penalty methods: These methods work for feasible points and penalize points that are close to the constraint boundary.
- Exterior penalty methods: These methods penalize infeasible points but not the feasible solutions.

D. Sharma (dsharma@itg.ac.in) Penalty Functions 8 / 31

Now, penalty function methods these are found to be simple and most popular methods for handling the constraints. In these methods, if a solution violates any of the constraints, the solution is penalized by adding penalty with the objective function. So, as the statement says that, when a solution is infeasible, so, we will try to find out that how much it is infeasible.

So, the term which we use is called constraint violation. So, based on that constraint violation we add penalty to the objective function. See suppose we are solving a minimization problem, in that case we will be adding a penalty into the objective function value, so, that the infeasible solution will be penalized. On the other hand if we are solving a maximization problem we will be subtracting this penalty into the objective function

method. So, accordingly we will use those penalty function method for minimization and for maximization types of problem.

Now, here when we will be using penalty function method all these methods what they do is- they convert, the constraint optimization problem into the unconstrained optimization problem by adding this penalty term of each constraint violation to the objective function, and that is the whole purpose.

That instead of solving the method as a constrained optimization problem let us solve it as a unconstrained problem, because those unconstrained problem can be handled relatively easier when we are solving with the techniques which we have gone through now.

So, in this case when we talk about the penalty function method, we have two types of penalty function methods here. First is called interior penalty methods, as we can see here these methods work for feasible points and penalize points that are close to the constraint boundary.

What this suggest that if the solution is feasible then only we can use interior penalty method and when all solutions are feasible, then we will be penalizing the solutions which are close to the boundary. So, here the optimum solution will be somewhere inside the feasible search space and that accordingly these method work.

The another type of penalty method is exterior penalty method here. These method penalize infeasible points, but not the feasible solution. And that is an important statement, why because the interior penalty method cannot handle any infeasible point. However, the exterior penalty methods can handle both feasible as well as infeasible.

So, whenever there is a feasible point there will be no penalty at all, but when if the solution is infeasible then a penalty will be added to the objective function method and that is the whole objective.

(Refer Slide Time: 13:04)

Penalty Function Methods

- A constrained optimization problem can be written as

$$\begin{aligned} &\text{Minimize} && f(x), \\ &\text{subject to,} && g_j(x) \geq 0, && j = 1, 2, \dots, J, \\ &&& h_k(x) = 0, && k = 1, 2, \dots, K, \\ &&& x_i^{(L)} \leq x_i \leq x_i^{(U)}, && i = 1, 2, \dots, N. \end{aligned} \quad (1)$$

- EC techniques normally adopt exterior penalty methods.
- The penalty function method can be written as

$$P(x, R) = f(x) + \Omega(R, g(x), h(x)), \quad (2)$$

where R is a set of penalty parameters, Ω is the penalty term chosen to favor the selection of feasible point over infeasible point.

D. Sharma (dsharma@itg.ac.in) Penalty Functions 9 / 31

$$\text{Minimize } f(x),$$

$$\text{subject to, } g_j(x) \geq 0, \quad j = 1, 2, \dots, J,$$

$$h_k(x) = 0, \quad k = 1, 2, \dots, K,$$

$$x_i^{(L)} \leq x_i \leq x_i^{(U)}, \quad i = 1, 2, \dots, N.$$

$$P(x, R) = f(x) + \Omega(R, g(x), h(x))$$

Now, if we look at the constrained optimization problem. So, this is we are familiar with this representation where we want to minimize our objective function. We have inequality constraint, which is $g_j(x)$ we have equality constraints and we have variable bounds.

Now, from the definition of interior penalty methods and exterior penalty methods. We can see that EC techniques generally employ the external penalty methods why because, as we refer to our first figure when we are using multiple solutions, some solutions can be feasible and some solution can be infeasible.

So, in order to identify them give a fitness to them. So, we need a penalty we need a penalty function method. So, for EC computation techniques we need exterior penalty methods. So, as you can see in equation number 2: The penalty function method can be written as,

this is a penalty function method based on the problem variable x and we have a set of parameters or penalty parameters R .

On the right hand side, we have function value and as per the definition we are adding a penalty term. So, the omega is going to be the penalty term, which we are adding. Looking at this representation we can see this omega is a function of R is also a function of inequality constraint and equality constraint.

So, all these constraints will be considered together. Now, as we know that there could be multiple inequality and equality constraint. So, R is representing a set of penalty parameters, as we know omega is the penalty term that is chosen to favor the selection of feasible points over infeasible point and that is the whole objective here. That we should penalize the infeasible solution more and we should not penalize the feasible solution. So, that we can differentiate.

(Refer Slide Time: 15:28)

Penalty Function Methods

Parabolic Penalty

$$\Omega = R\{h(x)\}^2 \quad (3)$$

- It is used for handling equality constraints only.
- Since all infeasible points are penalized, it is an exterior penalty term.
- It starts with small value of R which increases gradually.

Bracket Operator Penalty

$$\Omega = R \langle g(x) \rangle^2 \quad (4)$$

where $\langle \alpha \rangle = \alpha$, when α is negative; zero otherwise.

- Since it assigns positive value to infeasible point, it is an exterior penalty term.
- It starts with small value of R which increases gradually.
- It is one of the commonly used methods.

Handwritten notes: $g(x) \geq 0$, $\alpha = -1.23$, $\langle \alpha \rangle = -1.23$, $\alpha = 0.27$, $\langle \alpha \rangle = 0$.

$$\Omega = R \{h(x)\}^2 \quad (3)$$

$$\Omega = R \langle g(x) \rangle^2 \quad (4)$$

So, let us start with some of the exterior penalty methods here. So, the first method that comes that we use for equality constraint is called parabolic penalty. Looking at equation number 3: The omega can be written as in terms of R and inside the bracket we have the equality constraint and we are making a square.

This square is important because when we have this equality constraint means the left hand side equals to the right hand side, then whenever a point is infeasible. So, the value of $h(x)$ can be negative as well as positive. Here we are considering the minimization problem.

So, whatever be the penalty or a constraint violation we get it we have to square them. So, we are always adding a positive value with the objective function for minimization problem. So, as we can see that this particular penalty method can handle only the equality constraint. Second thing is we can use it for the infeasible point. So, we can say that it is an exterior penalty term and we can very well use with EC techniques.

In this particular case generally we take a small value of R and we keep on increasing this value gradually. So, initially when we start with a small value of R . So, the solutions will be moving towards the close to the optimum solution without these constraints and as and when we are increasing the value of this R . These solutions will be moving towards the optimum solution of constraint optimization problem this is for the equality constraint.

Now, we have another method for inequality constraint, here we can see we can have bracket operator here, this bracket operator in equation 4 can be seen as $\max(0, R \cdot g(x))$. So, we are using these two brackets and inside it we have $g(x)$ and we are making a square. So, I will show you the significance of a square. Afterwards let us see what we mean by this bracket operator.

So, if we say this bracket of α this is going to be the exactly α , when α is negative meaning that if α is minus say 1.23 then bracket operator of α is also minus 1.23, otherwise it is zero meaning that whether if we are going to take say α is equals to 0.27 and if we find the bracket of α then it is going to be 0 because it is a positive value.

So, here this particular bracket operator is complying with our representation. So, as we know these constraints in our formulation is written as $g_j(x) \leq 0$. So, as and when the value of a g_j is equals to 0 or greater than 0, the solution is feasible. And as and when it is feasible we do not have to apply any kind of a penalty and when the solution is infeasible this means that $g_j(x)$ will take value smaller than 0.

So, we can very well fit this representation of g_j of x greater than equals to 0 into this bracket operator. Now, as you can see that in this equation number 4 the square is also multiplied. It is only because that since if the alpha is negative it will give me the same value and we are solving a minimization problem in which we want to add penalty into the objective function method.

So, the square term will make this negative value positive and we will be adding this penalty into the objective function value. So, here as we know as we can understood from the discussion that we can handle the infeasible point with this penalty function method. So, it is an exterior penalty term. In this case also we start with the small value of R and which we keep on increasing gradually with the iterations.

And the important point about this method is, it is one of the commonly used methods for handling inequality constraint. So, with the help of these two types of exterior penalty methods, we will see the different kind of penalty function methods that are being used with EC techniques. So, let us move forward with the representation of penalty function.

(Refer Slide Time: 20:54)

Penalty Function Methods

- The penalty function method can be written as

$$P(x, R) = f(x) + \sum_{k=1}^K R_k \{h_k(x)\}^\gamma + \sum_{j=1}^J R_j \langle g_j(x) \rangle^\beta, \quad (5)$$

where R_k and R_j are the penalty parameters. Normally, β and γ are kept 1 and 2, respectively.

Handling Equality Constraints

- The penalty function methods can handle both equality and inequality constraints.
- One of the suggested ways to handle the equality constraints is $|h(x)| - \epsilon \leq 0$, where ϵ is the tolerance allowed (a small value).

D. Sharma (dsharma@iitg.ac.in) Penalty Functions 11 / 31

$$P(x, R) = f(x) + \sum_{k=1}^K R_k (h_k(x))^\gamma + \sum_{j=1}^J R_j \langle g_j(x) \rangle^\beta,$$

As we can see in equation number 5: The penalty function method can be written as P of x R the objective function. Now, we are adding all the penalties with respect to the

constraint violation of equality constraints and then we are adding the penalty with respect to the constraint violation of inequality constraints.

Now, here as we have mentioned R_k and R_j are the penalty parameters for every constraint in our problem. So, generally we keep β is equals to 1 and γ as 2. But from our previous discussion, we know that this γ is going to be small. So, either we take 1.

So, then in that case we have to take a mod or we have to or we have to square the term of bracket operator. Now, from the previous discussion we can very well use the parabolic penalty for the equality constraint, but still handling equality constraint is always difficult, why? Because it is it says that the left hand side should be equals to the right hand side.

So, having such point or satisfying such kind of equality constraint it is always a challenging task for all optimization methods. So, one of the way to handle the equality constraint is that we can convert this inequality equality constraint into inequality constraint.

So, the point is we can very well use the penalty function method for both equality and inequality. However, in order to suggest more efficiently the equality constraint, we can have a representation; such as $\text{mod of } h x \text{ minus epsilon should be smaller than } 0$. Where, this epsilon is the tolerance allowed generally we keep a small value set n to the power minus 3 minus 5, etcetera,

So, this particular new kind of a constraint for equality it will may it will make 1 equality constraint into 2 inequality constraints. And this inequality we can use it with our formulation and we can very well use the bracket operator with such kind of inequality constraint.

(Refer Slide Time: 23:34)

Types of Penalty Functions

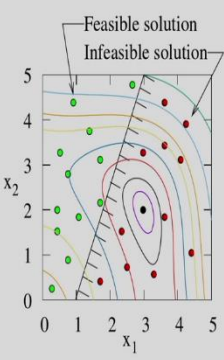
- Death Penalty ✓
- Static Penalty ✓
- Dynamic Penalty ✓
- Adaptive Penalty ✓
- Other Approaches
 - ▶ Self-Adaptive Fitness Formulation ✓
 - ▶ Stochastic Ranking, etc ✓

D. Sharma (dsharma@itg.ac.in) Penalty Functions 12 / 31

Now, let us move with the types of penalty function methods. So, there are various kinds such as we can have death penalty, we can have a static penalty, similarly the dynamic penalty functions. Adaptive penalty and there are some other approaches which are available in the literature; such as self-adaptive fitness formulation, stochastic ranking etcetera. So, the literature has so many penalty function based methods, here we will be focusing on few of them.

(Refer Slide Time: 24:11)

Death Penalty



- **Death Penalty:** Infeasible solution is rejected and re-generated again.
- This is the easiest way to handle the constraints.
- It is considered as computationally efficient.

It does not require to estimate the degree of violation for assigning the fitness to a solution.

D. Sharma (dsharma@itg.ac.in) Penalty Functions 13 / 31

So, let us start with death penalty function. Now, the death penalty function as you can see here on the right hand side it says that infeasible solution is rejected and re-generated again. So, point is that this death penalty death penalty method does not require or does not want any infeasible solution. So, as and when an infeasible solution is generated it is re-generated. So, that it will become feasible

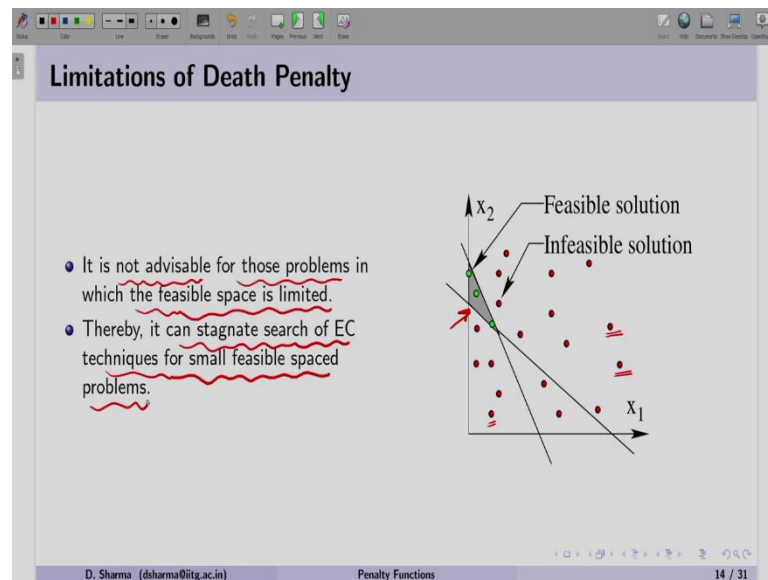
Now, let us look at the figure. So, we can see that with EC techniques we work on multiple solutions as you can see in the figure. Now, when E-C techniques is generating the green dots as you can see, these green dots are feasible and death penalty can take care of these solutions.

However, as and when there is a infeasible solution, which is represented by the red color dots then this death penalty function will reject such kind of solution. So, since we are not performing any kind of violation here we can say that it is the easiest way to handle the constraint.

Moreover we are not looking at the penalized function. So, we can say this is the computationally efficient, in a way to handle with the feasible solution only. Moreover it does not require to estimate the degree of violation for assigning the fitness to a solution. Now, it means that since the dynamic penalty method does not need any kind of infeasible solution or does not work on the infeasible solution.

So, all solutions for us are feasible. So, in that case we do not need any constraint violation a particular method that will help us to differentiate the feasible and infeasible solution. And therefore, it will not calculate any kind of a constraint violation, it will accept if the point is feasible, it will reject the point if it is infeasible. Although this method looks simple, but there are certain limitations, so let us see that.

(Refer Slide Time: 26:40)

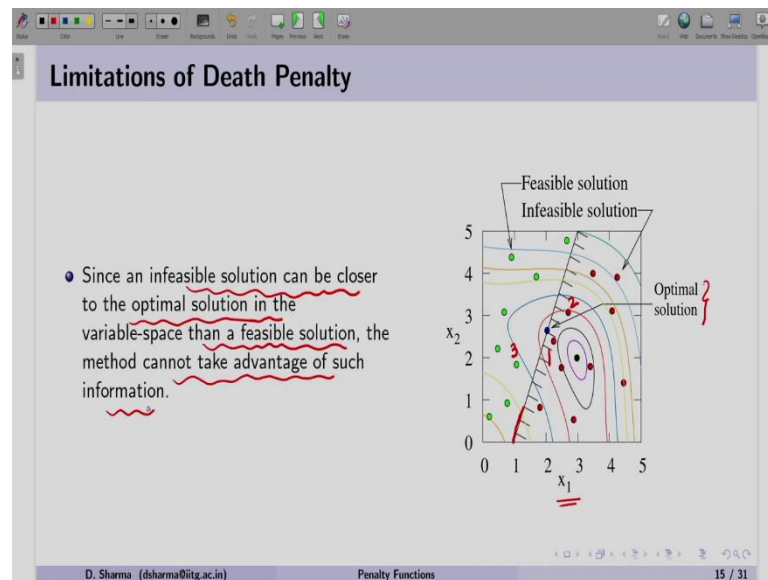


So, looking here that, if we look at this particular figure on the right hand side. Now, suppose we have a very small feasible region as it is shown in the shaded region. And again the green dots are the feasible points and the red dots are the infeasible point. So, when we want to use this death penalty? It is advisable for those problems in which the feasible space is limited.

So, in that case if we are going to solve a problem where the overall feasible search space is small then it is not advisable to use such function looking at the figure on the right hand side. We can see that with the while using this death penalty many of the solutions are infeasible. This means this method is going to reject all these infeasible solution and will only take the feasible solution.

So, as of now there are only three feasible solution and rest of them are infeasible solution. So, which means that it can stagnate search of EC techniques for small feasible spaced problem, why? Because, we have less number of feasible solution as compared to the infeasible solution.

(Refer Slide Time: 28:09)



There is another limitation as well here. Since, we are rejecting the infeasible solution straight forward here the point is there could be some scenarios where these infeasible solutions can be closer to the optimum as compared to the feasible solution. So, let us look a case on the right hand side.

So, this for this particular problem so, it is a the contours are drawn for a Himmelblau function and we have just one constraint as you can see here. Now, since E C techniques generated the random points in the plane of x_1 and x_2 . We have the green dots as a feasible points and the red dots as a infeasible point. See for the given problem the optimum solution is shown here, which is in the blue color.

Now, as compared to this optimum solution we can see that the solution number 1, solution number 2 they are more closer to the optimum solution than this say solution number 3 and that is the disadvantage, why? Because, we are not able to exploit the infeasibility of the solution.

So, basically looking at their constraint violation, if the constraint violation is small our solution can be close to the optima or the boundary. Since we are rejecting those solutions and dealing with feasible solution only, then EC techniques may require more number of a generation or function evaluation to reach to an optimum.

However, if we consider this infeasibility with the death penalty we can very well exploit the area or the space near the optimum solution using feasible and infeasible solution. And EC technique can find this solution quickly. So, therefore, this is one of the limitations, that the infeasible solution can be closer to the optimum solution in the variable-space than a feasible solution. The method cannot take advantage of such information as we discussed earlier.

(Refer Slide Time: 30:31)

Static Penalty

- The penalty function method can be written as

$$P(x, R) = f(x) + \sum_{k=1}^K R_k \{h_k(x)\}^\gamma + \sum_{j=1}^J R_j \langle g_j(x) \rangle^\beta, \quad (6)$$

where R_k and R_j are the penalty parameters. Normally, β and γ are kept 1 and 2, respectively.

- The penalty parameters remain constant throughout evolutionary process.

D. Sharma (dsharma@iitg.ac.in) Penalty Functions 16 / 31

$$P(x, R) = f(x) + \sum_{k=1}^K R_k \{h_k(x)\}^\gamma + \sum_{j=1}^J R_j \langle g_j(x) \rangle^\beta$$

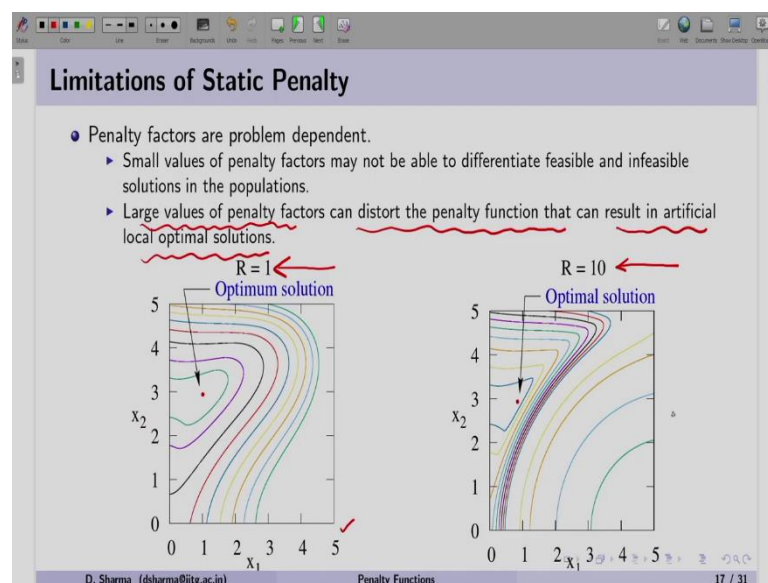
Now, the death penalty was not using any kind of penalty term. Now, we are coming to the static penalty which will be using some penalty term to penalize the infeasible solution. So, if we look at the equation number 6: This is the same penalty function method we have written earlier, where we have the penalty function method written with the help of x and R .

And on the right hand side we have objective function, then we have constrained violation with respect to equality constraint and then we have constrained violation with respect to inequality constraint. So, base using those constraint violations we are making a penalty term here. So, why we call this is static it is because the penalty parameters remain constant

throughout the evolutionary process. So, if we discuss if we when we have gone through the two types of penalty methods that are parabolic as well as the bracket operator.

In both the in both the penalty function methods we discussed that we start with the small value of R and then we keep on increasing this particular value in with the number of generation or iteration. But with the static penalty, we will always keep the value of R same. So, for example, R_k value for k th constraint will remain the same throughout the generation. So, what could be the possible limitations of static penalty?

(Refer Slide Time: 32:15)



Now, let us see that penalty function method that is an important point that the penalty function methods are problem dependent. Meaning that the depending on the problem the penalty value the value of R , may be sufficient for a one particular problem, but may not be sufficient for other kind of a problem.

So, as it is stated that if we take a small value of a penalty factors then it may not be able to differentiate feasible and infeasible solution in the population. So, as we know that with the objective function we are adding the penalty. Now, the value of R is itself small then the this penalty function method we cannot differentiate, which is feasible and infeasible and accordingly the there will be not difference in the fitness assignment.

However, if we take a large value. So, the if large value of a penalty factor can distort the penalty function that can result in an artificial local optimum solution. So, that is another

disadvantage, that if we keep the value of R say large then in that case the contours of the penalty function method will distort. And because of that there could be some local or artificial optima's.

So, let us look this particular case with an example given here. So, the figure on the left hand side. So, again it is a Himmelblau function with just one constraint and we have taken the value of R say 1. So, we are considering R 1 is a small value. So, you can see that when the value of R equals to 1, we have such kind of a contours.

Now, looking at the R value 10, the same penalty function method with the R equals to 10, you can see there is a change in the contours. And this change is with respect to just one constraint. Now, we can imagine a situation when we have many number of objective functions. We can imagine a situation, when we have many number of constraints and those constraints are added with the penalty parameters and when we take a large value, the penalty function method will distort.

So, the contours represents the distortion in the penalty function method and therefore, if we take a small value we cannot differentiate the feasible and infeasible. If we take a large value then that will generate artificial optima's, then in that case our algorithm should be efficient enough to move towards the global optima rather than stuck in the local optima.

(Refer Slide Time: 35:18)

Dynamic Penalty

- The penalty factors include the current generation counter (t) in its computation.

$$P(x, R) = f(x) + (C \times t)^\alpha \left[\sum_{k=1}^K \{h(x)\}^\gamma + \sum_{j=1}^J \langle g_j(x) \rangle^\beta \right], \quad (7)$$

where C , α , β and γ are the user defined constants.

- It suggests that the penalty term $((C \times t)^\alpha)$ is increasing with the generation counter (t).
- Joines and Houck [1994] used $C = 0.5$, $\alpha = 1$ or 2 , β and γ are kept 1 and 2 , respectively.
- Here, $\langle g_j(x) \rangle = g_j(x)$, when $g_j(x) < 0$, otherwise, it is zero.
- We convert an equality constraint into two inequality constraints as $|h(x)| - \epsilon \leq 0$.

Limitation

- Although it is considered good for many EC techniques, it is difficult to produce good dynamic penalty factors for static functions.

D. Sharma (dsharma@iitg.ac.in) Penalty Functions 18 / 31

$$P(x, R) = f(x) + (C \times t)^\alpha \left[\sum_{k=1}^K \{h(x)\}^\gamma + \sum_{j=1}^J \langle g_j(x) \rangle^\beta \right]$$

Let us move to the another kind of penalty method, which is dynamic penalty, this dynamic penalty as you can see here that includes the current generation counter t. So, equation 7: We can see we have a penalty function method equals to the objective function and we have included a term called C multiplied by t and that everything power alpha.

Now, here t will be representing the generation counter and then rest of the term as we have we are writing for the equality and the inequality constraint. So, in this formulation C, alpha, beta and gamma all are the user defined constraints. Now, this particular equation 7 that suggest that if suppose the value of a t is 1.

So, basically the generation number 1 then this particular factor will be small, but as and when we will be moving towards the say generation number 10, 20, 30 this particular factor will increase. So, meaning that at the end of the iteration if the solution is infeasible, we will be giving more penalty to the solution as compared to the solution at the beginning of the generation.

Here, these two author they suggested to use a value of a C as 0.5, and as per our previous discussion we can keep the value of a beta and a gamma, here the value suggested for alpha is 1. Now, in this particular discussion we already did it that the bracket operator is the g j g j of x value remains the same, when it is smaller than 0, otherwise it is zero and equality constraint can be converted into an inequality constraint with this representation.

What is the limitation of death penalty here, that although it is considered good for many EC techniques. It is difficult to produce good dynamic penalty factors for static functions, why? Because the functions which we are using especially the objective functions, objective function and constraints we are considering them static. So, in that case although it is performing good, but it is difficult to produce good dynamic penalty factors for those kind of functions.

(Refer Slide Time: 37:59)

Adaptive Penalty

- Similar to dynamic penalty, adaptive penalty also changes with generation counter (t)
- Bean and Hadj-Alouane [1992,1997] developed such an adaptive penalty

$$P(x, R) = f(x) + \lambda(t) \left[\sum_{k=1}^K |h(x)| + \sum_{j=1}^J \langle g_j(x) \rangle^2 \right], \quad (8)$$

where $\lambda(t)$ is getting updated in every generation.

- It is calculated as

$$\lambda(t+1) = \begin{cases} (1/\beta_1) \cdot \lambda(t), & \text{if case \# 1} \\ \beta_2 \cdot \lambda(t), & \text{if case \# 2} \\ \lambda(t), & \text{otherwise,} \end{cases}$$

where case # 1 denotes situation where the best solution in the last k generations was always feasible, case # 2 denotes situation where the best individual in the last k generations was never feasible, $\beta_1, \beta_2 > 1$, $\beta_1 > \beta_2$.

D. Sharma (dsharma@itg.ac.in) Penalty Functions 19 / 31

$$P(x, R) = f(x) + \lambda(t) \left[\sum_{k=1}^K |h(x)| + \sum_{j=1}^J \langle g_j(x) \rangle^2 \right]$$

With this explanation let us move to the adaptive penalty. Now, this adaptive penalty also takes also uses the generation counter. So, as we can see similar to the dynamic penalty, adaptive penalty also changes with the generation counter t . By the authors by these authors, they come up with the representation of adaptive penalty.

Now, here we have the objective function plus lambda. So, this is the new term which they have added and then we have the constraint violations that is coming from the equality and inequality constraint. Now, here lambda is getting updated in every generation. So, this is the, this is how the lambda is taking care of the t , which is the number of or the counter on the generation. So, that adaptively it will either increase or reduce the value.

$$\lambda(t+1) = \begin{cases} \left(\frac{1}{\beta_1}\right) \cdot \lambda(t), & \text{if case \# 1} \\ \beta_2 \cdot \lambda(t), & \text{if case \# 2} \\ \lambda(t), & \text{otherwise.} \end{cases}$$

Let us look the alpha this lambda value as we can see here the lambda can be calculated as. So, there is a case 1 for a case 1 we will be using 1 divided by beta and multiplied by lambda t . In case 2- we have beta 2 multiplied by lambda and if there is no case from 1 and a 2, we will be keeping the same value of lambda.

So, let us understand what is case 1? Now, case 1 suggests that it denotes a situation where the best solution in the last k generation was always feasible. So, in this case suppose we are running an EC technique and we have taken say k equals to 6. So, from the last six generation, if we have the best solution always the feasible solution, then in that case we will be using the case 1.

Case 2 here, denotes the situation where the best individual or a solution in the last k generation was never feasible, meaning that the for some kind of a problem when we are using EC techniques the problems are so difficult that we may not get the feasible solution at all.

So, if we consider say k equals to six generation. So, from last six generation if we find that the best solution was always infeasible, it was never be a feasible solution in that case we use case number 2. Here beta 1 and a beta 2 both of them should be greater than 1 and beta 1 should be greater than beta 2. So, let us understand more about this lambda term here.

(Refer Slide Time: 41:04)

Adaptive Penalty

$$\lambda(t+1) = \begin{cases} (1/\beta_1) \cdot \lambda(t), & \text{if case \# 1} \\ \beta_2 \cdot \lambda(t), & \text{if case \# 2} \\ \lambda(t), & \text{otherwise,} \end{cases}$$

- The penalty term $\lambda(t+1)$ decreases if all the best solutions in the last k generations were feasible
- It increases if they were all infeasible.
- If there are some feasible and infeasible solutions tied as best in the population, then the penalty does not change.

Limitation

- The major issue with this penalty is the setting of the parameters that can be difficult sometimes.

D. Sharma (dsharma@itg.ac.in) Penalty Functions 20 / 31

$$\lambda(t+1) = \begin{cases} \left(\frac{1}{\beta_1}\right) \cdot \lambda(t), & \text{if case \# 1} \\ \beta_2 \cdot \lambda(t), & \text{if case \# 2} \\ \lambda(t), & \text{otherwise.} \end{cases}$$

Now, here what we can see in case 1, which says that our best solution was a feasible for k number of a generation, meaning that we should reduce as you can see 1 by β since β was greater than 1 . So, we are actually reducing the value of λ_{t+1} . In the other case, when the best solution is infeasible from the last k generation.

So, we have to increase this penalty and therefore, as per the case 2, we are multiplying β_2 with λ_t for the next iteration. Otherwise if there is a mix of population, meaning having feasible and infeasible then we do not have to change the value of λ .

So, as per our discussion we can see here the term λ_{t+1} decreases if all the best solution in the last k generation are feasible. So, this means we should reduce it. We have to increase the λ_{t+1} if all solutions are infeasible, if there is a mix of feasible and infeasible solution then we should not change this particular penalty.

So, what could be the limitation here the major issue with this penalty is the setting of the parameters that can be difficult sometime. So, the point is, so the point is we have to take β_1 and a β_2 . Now, since β_1 and a β_2 should be greater than 1 the question is how much? Can we take $2, 2.5, 10$. So, setting those values for the problem is an is a difficult task and that is why this is the limitation.

(Refer Slide Time: 42:58)

Major Issues with Penalty Function Methods

- Choosing the best penalty term is not known a priori for any arbitrary problem.
- Large value of penalty terms can distort the function with artificial optimal solutions.
 - If the optimal solution is on the constraint boundary, EC techniques will push all solutions inside the feasible search space and may find difficulty to move toward the boundary.
- Small penalty terms will not penalize the infeasible solution as compared to the objective function value.
 - EC techniques will waste their effort in searching the solution in the infeasible space.

D. Sharma (dsharma@iitg.ac.in) Penalty Functions 21 / 31

So, what are the major issues? If we look all these penalty function methods which we have gone through, what are the major issues? First major issue is choosing the best penalty term is not known a priori for any arbitrary problem and that is difficult. Since, in any every penalty function method, we have to choose those parameters.

So, in that case a parameter which may be sufficient for a problem 1 may be very large for very large for problem 2 or very small for problem 3. So, we have if we are working on an arbitrary problem, in that case setting such kind of a value sometimes may not help to find an optimum solution for the constraint optimization.

Second issue is the value if we take a large value of the penalty term that can distort the function with artificial optimum solution. So, this case we discussed earlier when we are taking a large value of a R , it distorts the contour of the penalty function method

So, those functions will be distorted and if we have more number of constraint, the distortion will be up to the level that it can generate some artificial optimum solution. So, if, so in this case if optimum solution is on the constraint boundary then EC technique will push all the solution inside the feasible search space and may find difficult to move towards the boundary.

So, this is the case suppose we are starting with a very large value. So, as compared to the feasible, infeasible solution will be penalized heavily. So, since we are penalizing those solutions heavily what will happen that, this EC techniques will move or will take the solution in the feasible region.

And suppose the optimum solution is on the boundary which we have shown in one of the cases, then in that case the EC techniques has to further work. So, that these feasible solution will move towards the optimum solution. In case, we take a small penalty term then as we know that we will not be penalizing sufficiently the infeasible solution as compared to the feasible solution.

So, this in this way we are not able to differentiate, which one is feasible, which one is infeasible and if and our EC technique will be just wasting its effort towards searching in the infeasible space. So, therefore, whether we have to set a small value or a large value and setting any value can be a large for a one problem, can be small for the another problem. So, these are always the issues with the penalty function methods. Now, with this

introduction and understanding on the different types of penalty function method let us perform some hand calculation.

(Refer Slide Time: 46:18)

Hand Calculations

Himmelblau Function

Minimize $f(x_1, x_2) = (x_1^2 + x_2 - 11)^2 + (x_1 + x_2^2 - 7)^2$,
 subject to $(x_1 - 5)^2 + x_2^2 \leq 26$,
 $4x_1 + x_2 \leq 20$,
 $x_1, x_2 \geq 0$.

• Let us convert the inequality constraints in the form of $g_j(x) \geq 0$.

$g_1(x) = 26 - (x_1 - 5)^2 - x_2^2 \geq 0$,
 $g_2(x) = 20 - 4x_1 - x_2 \geq 0$.

D. Sharma (dsharma@iitg.ac.in) Penalty Functions 23 / 31

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

$$\text{subject to } (x_1 - 5)^2 + x_2^2 \leq 26,$$

$$4x_1 + x_2 \leq 20,$$

$$x_1, x_2 \geq 0$$

$$g_1(x) = 26 - (x_1 - 5)^2 - x_2^2 \geq 0,$$

$$g_2(x) = 20 - 4x_1 - x_2 \geq 0.$$

Here, for performing the hand calculations we are using Himmelblau function. So, this Himmelblau function is a two variable function and this is subjected to one constraint and constraint number 2, here we are assuming that x_1 and x_2 both are greater than 0. Since, the constraints are given as smaller than type. Let us convert them in a greater than type why because, it will fit into our generalized format of constraint optimization.

So, what we can do here is, we can multiply the constraint minus 1 or we can take the left hand term on the right hand side and then every constraint will be of greater than type. So,

by converting them we have g_1 and g_2 and you can see that both the constraints are converted into the greater than an equality type.

(Refer Slide Time: 47:16)

Hand Calculations

- Let us generate random solutions in the range of $0 \leq x_1, x_2 \leq 6$.

Index(i)	$(x_1, x_2)^T$
1	$(3.660, 4.595)^T$
2	$(2.380, 5.561)^T$
3	$(4.698, 3.219)^T$
4	$(3.755, 5.151)^T$
5	$(1.976, 1.754)^T$
6	$(3.654, 5.160)^T$
7	$(0.100, 3.858)^T$
8	$(2.446, 0.880)^T$

- Consider **solution 1**
 $x^{(1)} = (3.660, 4.595)^T$.
- Calculate $f(x_1, x_2) = (x_1^2 + x_2 - 11)^2 + (x_1 + x_2^2 - 7)^2 = 364.823$
- Calculate
 $g_1(x_1, x_2) = 26 - (x_1 - 5)^2 - x_2^2 = 3.089$
- Calculate
 $g_2(x_1, x_2) = 20 - 4x_1 - x_2 = 0.765$
- Since both the constraints are satisfied, solution 1 is a feasible solution.
- Death Penalty:** it will keep this solution because it is feasible.

D. Sharma (dsharma@iitg.ac.in) Penalty Functions 24 / 31

Let us generate the solution in the range of 0 to 6 for both variable x_1 and x_2 . So, let us take this table here. So, we have index means we are considering 8 solutions and their x_1 , x_2 values are given in the column number 2 consider a solution number 1. Now, this solution 1 x_1 and x_2 are given to us, as we know if we fit this particular x_1 and x_2 value in our objective function we get a value say 364.823. Now, let us calculate the constraints, why? Because we want to know which one feasible or infeasible. So, in g_1 we are again putting the value we get plus 3.089.

Similarly, if we put the value of x_1 and x_2 of solution 1, we get a value as 0.765. Now, as we can see both are greater than 0. So, we can say that a solution 1 is a feasible solution because both the constraints are satisfied. If we take death penalty right now. So, we will keep this solution because it is infeasible.

(Refer Slide Time: 48:42)

Hand Calculations

- For solution 1, $f(x^{(1)}) = 364.823$, $g_1(x^{(1)}) = 3.089$ and $g_2(x^{(1)}) = 0.765$.
- Consider **Static Penalty**: $P(x, R) = f(x) + \sum_{j=1}^2 R_j (g_j(x))^2$
- Let us take $R_1 = 1$ and $R_2 = 5$.
- The penalty function value is

$$P(x^{(1)}, R) = 364.823 + R_1 (3.089)^2 + R_2 (0.765)^2 = 364.823 + 0 + 0 = 364.823$$

$\alpha \geq 0$ $\langle \alpha \rangle = 0$

D. Sharma (dsharma@iitg.ac.in) Penalty Functions 25 / 31

$$f(x^1) = 364.823, g_1(x^1) = 3.089 \text{ and } g_2(x^1) = 0.765$$

Coming to the another kind of penalty functions method. So, let us see well let us take the same solution here. So, the function value which we have found in the previous slide. So, we are writing similarly the g_1 value and g_2 value at x^1 . Now, let us consider the static penalty. Now, since we know the formula here we can directly fit the $f(x)$ and we have two constraints. So, we are writing this formula.

So, here j is equals to 1 to 2 R_j and we have a bracket operator and a square here. Now, let us take R_1 is 1 and R_2 is 5. So, as you can see that R_1 and R_2 are user defined parameters. So, we have to start or we have to choose those two parameters. Now, here the penalty function for x^1 solution you can find that it is given as the $f(x)$ and then R_1 inside the bracket the g_1 value and then R_2 inside the bracket we have g_2 value.

Now, since we know that if α is greater than or equals to 0 then α is going to be 0, when we use the bracket operator here, right. So, in that case we can see that both the constraints the values since both of them are feasible. So, their values 0s we are having here. So, in this case the penalty function value is 364.823, which is the same as the objective function value.

(Refer Slide Time: 50:25)

Hand Calculations

- For solution 1, $f(x^{(1)}) = 364.823$, $g_1(x^{(1)}) = 3.089$ and $g_2(x^{(1)}) = 0.765$.
- Consider **Static Penalty**: $P(x, R) = f(x) + \sum_{j=1}^2 R_j \langle g_j(x) \rangle^2$
- Let us take $R_1 = 1$ and $R_2 = 5$.
- The penalty function value is

$$P(x^{(1)}, R) = 364.823 + R_1 \langle 3.089 \rangle^2 + R_2 \langle 0.765 \rangle^2 = 364.823 + 0 + 0 = 364.823$$
- Consider **Dynamic Penalty**: $P(x) = f(x) + (C \times t)^\alpha \left[\sum_{j=1}^2 \langle g_j(x) \rangle^2 \right]$
- Considering $C = 0.5$, and $\alpha = 1$. Assuming it is the first generation, that is, $t = 1$.

$$f(x^1) = 364.823, g_1(x^1) = 3.089 \text{ and } g_2(x^1) = 0.765$$

$$P(x, R) = f(x) + \sum_{j=1}^2 R_j \langle g_{j(x)} \rangle^2$$

$$P(x^1, R) = 364.823 + R_1 \langle 3.089 \rangle^2 + R_2 \langle 0.765 \rangle^2 = 364.823 + 0 + 0 = 364.823$$

$$P(x) = f(x) + (C \times t)^{\alpha} \left[\sum_{j=1}^J \langle g_{j(x)} \rangle^2 \right]$$

Let us take dynamic penalty now, in the dynamic penalty we have objective function plus we have the factor, which is changing with respect to the number of generation and then we have a summation of a inequality constraints since we have 2 so, this should run for 2 only. Now, considering C equals to 0.5 as we have found in the previous slide alpha equals to 1 and let us assume that it is the generation number 1.

(Refer Slide Time: 50:57)

Hand Calculations

- For solution 1, $f(x^{(1)}) = 364.823$, $g_1(x^{(1)}) = 3.089$ and $g_2(x^{(1)}) = 0.765$.
- Consider **Static Penalty**: $P(x, R) = f(x) + \sum_{j=1}^2 R_j \langle g_j(x) \rangle^2$
- Let us take $R_1 = 1$ and $R_2 = 5$.
- The penalty function value is
 $P(x^{(1)}, R) = 364.823 + R_1 \langle 3.089 \rangle^2 + R_2 \langle 0.765 \rangle^2 = 364.823 + 0 + 0 = 364.823$
- Consider **Dynamic Penalty**: $P(x) = f(x) + (C \times t)^\alpha \left[\sum_{j=1}^J \langle g_j(x) \rangle^2 \right]$
- Considering $C = 0.5$, and $\alpha = 1$. Assuming it is the first generation, that is, $t = 1$.
- The penalty function value is
 $P(x^{(1)}) = f(x) + (0.5 \times 1) \left[\langle 3.089 \rangle^2 + \langle 0.765 \rangle^2 \right] = 364.823 + 0 + 0 = 364.823$
- Since it is a feasible solution, the penalty function value remains the same for static and dynamic penalty functions.

D. Sharma (dsharma@iitg.ac.in) Penalty Functions 25 / 31

Now in this case, if we fit these value here. So, the penalty function now there is no R here as you can see and when we are putting the value here. So, you can see that since g_1 and g_2 values are greater than 0. So, we have taken 0, here. So, the overall penalty term is 0 and we get the penalty function value same as the objective function value.

$$P(x^1) = f(x) + (0.5 \times 1)[\langle 3.089 \rangle^2 + \langle 0.765 \rangle^2] = 364.823 + 0 + 0 = 364.823$$

So, our observation here is since the solution 1 is a feasible solution the penalty function value remains the same for static and for dynamic penalty function method. So, you can see here this is these are the penalty function values, which we get it from static and dynamic and on the top we can see we have the function value. So, that is good point that if the solution is feasible then we are not going to add any kind of penalty with the solution.

(Refer Slide Time: 52:03)

Hand Calculations

- For solution 2, we calculate $f(x^{(2)}) = (2.380, 5.561)^T = 692.216$, $g_1(x^{(2)}) = -11.791$ and $g_2(x^{(2)}) = 4.917$.
- The constraint $g_1(x^{(2)})$ is not satisfied, hence the solution is infeasible.
- Consider **Static Penalty**: $P(x, R) = f(x) + \sum_{j=1}^2 R_j \langle g_j(x) \rangle^2$
- Let us take $R_1 = 1$ and $R_2 = 5$.
- The penalty function value is

$$P(x^{(2)}, R) = 692.216 + \langle -11.791 \rangle^2 + 5.0 \langle 4.917 \rangle^2 = 692.216 + 139.021 + 0 = 831.236$$
- Consider **Dynamic Penalty**: $P(x) = f(x) + (C \times t)^\alpha \left[\sum_{j=1}^J \langle g_j(x) \rangle^2 \right]$
- Considering $C = 0.5$, and $\alpha = 1$. Assuming it is the first generation, that is, $t = 1$.
- The penalty function value is $P(x^{(2)}) = 692.216 + (0.5 \times 1) \left[\langle -11.791 \rangle^2 + \langle 4.917 \rangle^2 \right] = 692.216 + 0.5[139.021 + 0] = 761.726$
- If we consider the tenth generation, that is, $t = 10$, the penalty function value becomes

$$P(x^{(2)}) = 692.216 + (0.5 \times 10) [139.021 + 0] = 1387.319$$

D. Sharma (dsharma@iitg.ac.in) Penalty Functions 26 / 31

Let us take a solution 2: So, I will be discussing different kinds of situations or scenarios. So, the scenario one was that the solution 1 was feasible. Now, let us take a solution 2 in the solution 2 the objective function we calculate it is coming out to be 692.216. We get a value of a g_1 . Now, you can see that g_1 value is minus 11.791 meaning the constraint is not satisfied, g_2 is a plus value means it is satisfied.

Since, g_1 is not satisfied we have to say the solution is infeasible. Since solution is infeasible let us see, how this penalty methods will change. Now, let us consider the static penalty right now, we have penalty function given here and we have the same formula as we have used earlier taking R_1 equals to 1 R_2 is equals to 5, let us put the value. Here for the penalty function for a 0.2 we can see the objective function and inside the bracket operators since R_1 is 1. So, we are writing directly this square then we have 5 and then g_2 value of a square.

Now, we know that this is feasible. So, we this is going to be 0 by the bracket operator and since it is infeasible. So, we are taking a square term and we are adding into the penalty into the objective function. So, overall we can see that the penalty function value is more than the objective function value, which is given on the top for an infeasible solution. Similar thing let us see with the dynamic penalty now.

So, the given the formula here considering $C = 0.5$ $\alpha = 1$ and let us assume we have first generation right now. So, the penalty function value for a solution 2 we have function

value now 0.5 into 1 and then we have a bracket operator for both the constraint, since this is satisfied it is going to be 0 and it is not satisfied so, we are taking a square term here.

And finally, we get a value as 761.726. Now, here since we consider the iteration or the generation number 1, suppose if we get the same situation in the 10th generation. Then this, the same penalty dynamic penalty function value will become the objective function. Now, you can see the change here that 0.5 into 10.

So, this means that 5 and then we have the same values of the bracket operator square for g_1 and g_2 and we get a penalty function value. Now, if I compare this penalty function value with the penalty function value with t equals to 1. Now, what we can see here is that in the 10th generation the penalty function value will be large, also if we compare with the static penalty the term is quite large.

So, this means that with this static penalty whether it is generation 1 or a 10 the penalty function value will remain the same. However, with the dynamic penalty as we have seen here if it is generation number 1 then the penalty term will be less. However, when you consider penalty say 10th generation the penalty term will be large and accordingly, the penalty function value will be so large. Let us take a third situation we by considering the solution number 3, here.

(Refer Slide Time: 56:16)

Hand Calculations

- For solution 3, we calculate $f(x^{(3)}) = (4.698, 3.219)^T = 269.112$, $g_1(x^{(3)}) = 15.548$ and $g_2(x^{(3)}) = -2.010$.
- The constraint $g_2(x^{(3)})$ is not satisfied, hence the solution is infeasible.
- Consider **Static Penalty**: $P(x, R) = f(x) + \sum_{j=1}^2 R_j \langle g_j(x) \rangle^2$
- Let us take $R_1 = 1$ and $R_2 = 5$.
- The penalty function value is

$$P(x^{(3)}, R) = 269.112 + \underbrace{(15.548)^2}_{\text{circled}} + 5.0 \underbrace{(-2.010)^2}_{\text{circled}} = 269.112 + 0 + 20.208 = \boxed{289.320}$$

D. Sharma (dsharma@iitg.ac.in) Penalty Functions 27 / 31

$$P(x, R) = f(x) + \sum_{j=1}^2 R_j \langle g_j(x) \rangle^2$$

So, we are going to have solution 3, we calculate the objective function as given here the g_1 value is positive, but the g_2 value is negative since g_2 is not satisfied. So, the solution is infeasible as per our definition, let us find out the static penalty now. Now, the formulation is the same as we know we are considering R_1 is equals to 1 and R_2 is equals to 5 by putting the value of x_3 here.

We have objective function value and then we have bracket operators. Now, we know that since it is satisfied it is going to be 0 and here this constraint violation will be square and multiplied by 5, we can we are adding a penalty here. So, the overall penalty function value we can see it is 289.320, which is larger as compared to the function value 269.

(Refer Slide Time: 57:20)

Hand Calculations

- For solution 3, we calculate $f(x^{(3)}) = (4.698, 3.219)^T = 269.112$, $g_1(x^{(3)}) = 15.548$ and $g_2(x^{(3)}) = -2.010$.
- The constraint $g_2(x^{(3)})$ is not satisfied, hence the solution is infeasible.
- Consider **Static Penalty**: $P(x, R) = f(x) + \sum_{j=1}^2 R_j \langle g_j(x) \rangle^2$
- Let us take $R_1 = 1$ and $R_2 = 5$.
- The penalty function value is

$$P(x^{(3)}, R) = 269.112 + (15.548)^2 + 5.0 (-2.010)^2 = 269.112 + 0 + 20.208 = 289.320$$
- Consider **Dynamic Penalty**: $P(x) = f(x) + (C \times t)^\alpha \left[\sum_{j=1}^J \langle g_j(x) \rangle^2 \right]$
- Considering $C = 0.5$, and $\alpha = 1$. Assuming it is the first generation, that is, $t = 1$.
- The penalty function value is $P(x^{(3)}) = 269.112 + (0.5 \times 1) \left[(15.548)^2 + (-2.010)^2 \right] = 269.112 + 0.5[0 + 4.042] = 271.133$

D. Sharma (dsharma@itg.ac.in) Penalty Functions 27 / 31

For the given problem let us have this dynamic penalty, in this case using the formula given here, a C 0.5 alpha 1 and for first generation. If we put all the values here we can see that we have the objective function we have the; we have this C into t and then the bracket operator we know this is going to be 0. So, the overall penalty function value is 271.133.

$$P(x) = f(x) + (C \times t)^\alpha \left[\sum_{j=1}^J \langle g_{j(x)} \rangle^2 \right]$$

(Refer Slide Time: 57:55)

Hand Calculations

- For solution 4, we calculate $f(x^{(4)}) = (3.755, 5.151)^T = 610.196$, $g_1(x^{(4)}) = -2.081$ and $g_2(x^{(4)}) = -0.169$.
- Both constraints are not satisfied, hence the solution is infeasible.
- Consider **Static Penalty**: $P(x, R) = f(x) + \sum_{j=1}^2 R_j \langle g_j(x) \rangle^2$
- Let us take $R_1 = 1$ and $R_2 = 5$.
- The penalty function value is $P(x^{(4)}, R) = 610.196 + \langle -2.081 \rangle^2 + 5.0 \langle -0.169 \rangle^2 = 614.671$
- Consider **Dynamic Penalty**: $P(x) = f(x) + (C \times t)^\alpha \left[\sum_{j=1}^J \langle g_j(x) \rangle^2 \right]$
- Considering $C = 0.5$, and $\alpha = 1$. Assuming it is the first generation, that is, $t = 1$.
- The penalty function value is $P(x^{(4)}) = 610.196 + (0.5 \times 1) \left[\langle -2.081 \rangle^2 + \langle -0.169 \rangle^2 \right] = 612.376$

D. Sharma (dsharma@iitg.ac.in) Penalty Functions 28 / 31

$$P(x, R) = f(x) + \sum_{j=1}^2 R_j \langle g_{j(x)} \rangle^2$$

Now, let us take the 4th case, now in this particular case what you will see that the solution number 4, it has the objective function value and look at the value of a g_1 it is infeasible. Look at the value of a g_2 this is also negative. Since, both the constraints are negative, we will say that this particular solution is also infeasible.

$$P(x) = f(x) + (C \times t)^\alpha \left[\sum_{j=1}^J \langle g_{j(x)} \rangle^2 \right]$$

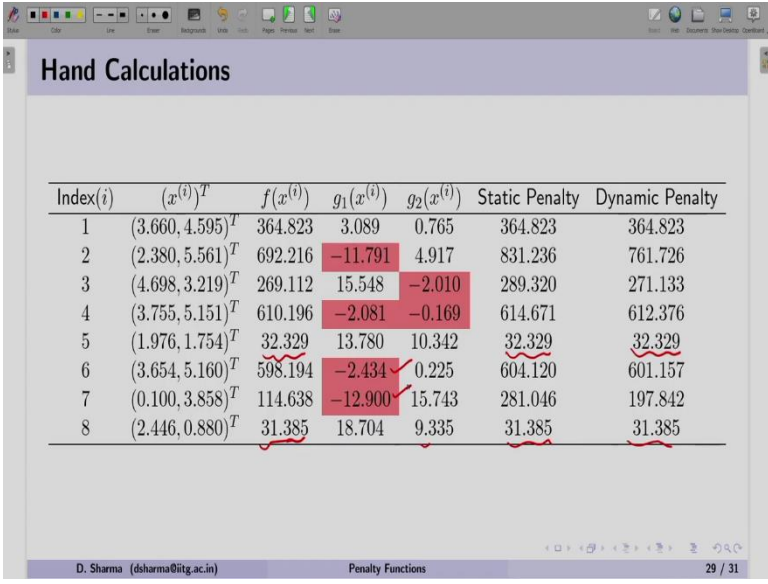
So, as of now we have discussed 4 cases, first case is solution was feasible. Second case g_1 was not satisfied so, it is infeasible. Third case g_2 was not satisfied that is why it was infeasible. And in the last case both g_1 and g_2 are not satisfied. So, that is why it is also an infeasible solution. So, let us find the penalty function value for solution number 4.

Here, the static penalty using the same formula taking the value of R 1 equal to 1 and R 2 equals to 0.5, by putting the value for a solution 4 we will get 614.671 why because you can see that this particular terms though constraint violation will be added this is also added. So, the both the constraint violation coming from g 1 and a g 2 are added and make a penalty for the given for the given point.

Looking at the dynamic penalty using the formula given here, taking C 0.5 alpha 0.1 and t equals to 1. If we put these values together we will get as you can see here we have objective function 0.5 into 1, we have first constraint, we have second constraint, since both of them are not satisfied. So, their contribution will be added as a penalty and we get the penalized function as 612.376.

So, from these scenarios you can see that we are adding only those we are adding the penalty corresponding to the constraint violation. So, whether it is first constraint or second constraint or both of the constraint, when they are not satisfied their constraint violation is added into the objective function and we find the penalty function value for the given solution. So, let us look at the table now.

(Refer Slide Time: 60:28)



The image shows a presentation slide titled "Hand Calculations" containing a table with 8 rows of data. The table columns are: Index(i), $(x^{(i)})^T$, $f(x^{(i)})$, $g_1(x^{(i)})$, $g_2(x^{(i)})$, Static Penalty, and Dynamic Penalty. Rows 2, 3, and 4 have red background shading for the g_1 and g_2 columns. Red checkmarks are present in the g_1 column for rows 5, 6, and 7. Red underlines are present under the $f(x^{(i)})$ and $g_2(x^{(i)})$ columns for rows 5, 6, and 7. The bottom of the slide shows the presenter's name "D. Sharma (dsharma@itg.ac.in)", the title "Penalty Functions", and the page number "29 / 31".

Index(i)	$(x^{(i)})^T$	$f(x^{(i)})$	$g_1(x^{(i)})$	$g_2(x^{(i)})$	Static Penalty	Dynamic Penalty
1	(3.660, 4.595) ^T	364.823	3.089	0.765	364.823	364.823
2	(2.380, 5.561) ^T	692.216	-11.791	4.917	831.236	761.726
3	(4.698, 3.219) ^T	269.112	15.548	-2.010	289.320	271.133
4	(3.755, 5.151) ^T	610.196	-2.081	-0.169	614.671	612.376
5	(1.976, 1.754) ^T	32.329	13.780	10.342	32.329	32.329
6	(3.654, 5.160) ^T	598.194	-2.434	0.225	604.120	601.157
7	(0.100, 3.858) ^T	114.638	-12.900	15.743	281.046	197.842
8	(2.446, 0.880) ^T	31.385	18.704	9.335	31.385	31.385

So, here in this particular table. So, the solution 1 was feasible. So, that is why the objective function value, static penalty function value and dynamic penalty function value are the same. In solution 2: g 1 was infeasible. So, accordingly the penalty was added into the

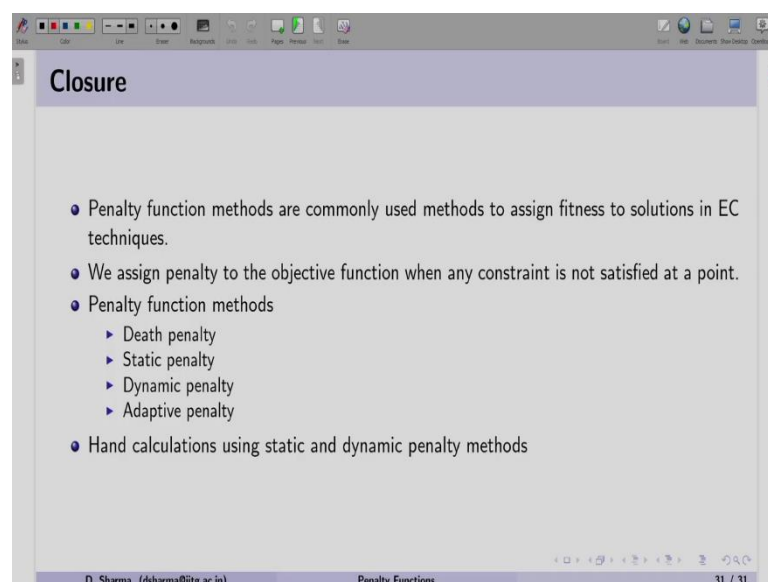
objective function and we got the two values different values for static penalty and dynamic penalty.

In the 3rd case: g_2 was infeasible so, that constraint violation was added and we get a static penalty function and in the dynamic penalty function values. In the case 4: both g_1 and g_2 were infeasible and accordingly we get our solution here. So, basically the fitness to a solution using static penalty and dynamic penalty function method.

Similarly, if we follow the same procedure for solution 5 to 8, what you can see? The solution 5 since it is feasible. So, the function value is the same as the static penalty function and dynamic. Similarly for solution number 8, but solution number 6 and 7, the solution is infeasible. Therefore, the penalty term corresponding to their constraint violation are added into their objective function value.

So, with this hand calculation we can see that as and when the infeasible solution is there, we are penalizing that solution adding the penalty corresponding to the constraint violation and the value of R_1 and R_2 accordingly. So, with this we have come to the closure of this particular session. So, what we have gone through in this that what we found that these penalty function methods.

(Refer Slide Time: 62:25)



They are commonly used for handling the constraint with EC techniques. In this way when we are using this penalty function method we can assign a fitness to a solution. So,

generally the penalty term is added, so that we will get smaller fitness for the feasible solution as compared to the infeasible solution.

So, in that case we can differentiate and you remember that this fitness can be used when we want to select any solution. And with, in this particular method we assign penalty to the objective function when any constraint is not satisfied. In this particular class of penalty function method, we have gone through various different kinds of penalty function method, starting with the dynamic penalty and then we have adaptive penalty function methods.

So, as you can see that all these penalty function methods they work differently and therefore, the fitness value which we will get specially for the infeasible solution is different with different from each other. And all these understanding the analysis of the penalty function method mainly the static and the adaptive penalty we consider dynamic penalty.

So, we have taken the static and dynamic penalty function methods. And we have taken a set of a solution we perform the hand calculation. So, that we can understand how the solution is getting the fitness with the explanation of these penalty function methods I conclude this session.

Thank you.