

Indian Institute of Science

Variational Methods in Mechanics and Design

Prof. G. K. Ananthasuresh

Department of Mechanical Engineering

Indian Institute of Science, Bangalore

NPTEL Online Certification Course

Hello we continue with the bar optimization problems that we were discussing in the last lecture where we had considered this problem let us just write the problem so that we understand and recall what we had discussed.

(Refer Slide Time: 00:34)

Min $A(x)$ $SE = \int_0^L \frac{1}{2} EAU^2 dx$

Subject to $\lambda(x): (EAU)' + P = 0$ ✓ $\leftarrow A(x)$

$\lambda: \int_0^L A dx - V^* = 0$ ✓

Data: $L, E, p(x), V^*$

Find: $A(x), u(x), \lambda, \lambda(x)$

$p(x) = P$

$A(x)$

$L =$

$\delta L = 0 \Rightarrow$ design eqn. ✓ $\leftarrow A(x)$

$\delta \lambda = 0 \Rightarrow$ Adjoint (equi) eqn. ✓ $\leftarrow \lambda(x)$

$(\lambda) \left\{ \begin{array}{l} \int_0^L A dx - V^* = 0, \\ \lambda \geq 0 \end{array} \right.$ Complementarity condition.

So what we were doing was minimize the strain energy of the bar which is of them of length L so strain energy is given by $\frac{1}{2} EAU^2$ square dx and our variable was A(x) subject to we have governing equation because U is a state variable $EAU' + P$ the loading is equal to 0 and then we

had volume constraint that $\int dx - V^* \leq 0$ as we had discussed if you have a functional constraint which is a second one we use upper case Greek letter and for the Lagrange multiplier if you have a function type or local type constraint Lagrange multiplier will be a function okay $\lambda(x)$ with that we wrote Lagrangian.

So the way if you recall we wrote Lagrangian and we took variation of Lagrangian with respect to area of cross section equal to 0 that gives us what we called design equation and then when we take variation with respect to the state variable and equate it to 0 that gives us what we called adjoint equation or adjoint equilibrium equation and then we have the governing equation and the volume constraint.

And we also have the complementarity condition that is λ times we have any quantity when you have any quantity we have complementarity condition say zero to $\int dx - V^*$ is equal to zero and we require λ to be greater than or equal to 0 this is what we call complementarity condition the reason I am repeating this is that if we understand the bar up relation problem we can actually do any structural optimization problem again the steps you note from the problem where we always have to write the data.

So we are given $L(x)$ modulus and the loading $p(x)$ and v^* given these what we need to find are area of cross section of course we need to find these area of cross section and the state variable $U(x)$ and λ and $\lambda(x)$ so we have three functions one two three let account whether we have those equations we have design equation would say a differential equation a joint equation in differential equation and equilibrium equation differential equation we have three differential equations and three functions.

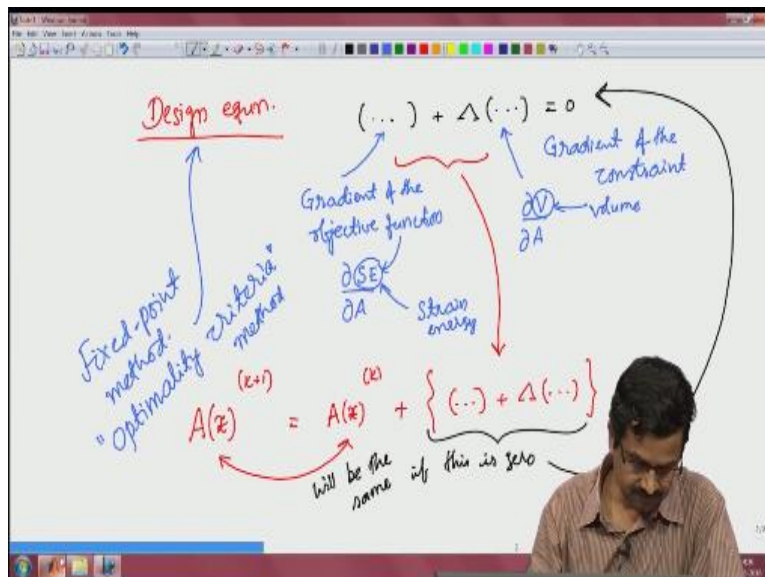
That are not known we use the differential equation to solve for $U(x)$ and then we use a joint equation to solve for $\lambda(x)$ and then we use design equation to solve $A(x)$ these are the steps and then we also have the λ upper case λ Lagrange multiplier correspond to the volume constraint for that we have this complementarity condition either λ is 0 or $\int dx - V^* = 0$ okay either these are zero that are zero when this is 0 we say the constraint is not active when this is 0 that is

the portion that we have here when I say this is 0 then the constraint is active because volume is constraint is satisfied then λ can be 0 or when it is not 0 it should be greater than 0 okay.

These are the steps so we had taken for this problem after solving we have taken a special case of $P(x)$ equal to P_0 that is uniform load everywhere that is we are discussing let us recall a bar problem if this is the axis of the bar we are trying to determine the profile of the bar whatever that area of cross section we are trying to define this so this is our area of cross section okay this is what we are trying to do right and what we had assumed to solve it analytically was that there is a uniform loading $p(x)$.

If there is a uniform loading $p(x)$ for that which is we said is p_0 then we solve the problem analytically what if we have a function that cannot be integrated right or is it tedious to integrate then we want to go for a numerical method which is what we will discuss today but also see how this problem can be solved in mat lab okay.

(Refer Slide Time: 06:27)



So what we do is we have the design equation that is our starting point for the algorithm design equation which will be of this form there will be something you know what that something is

from the previous lecture $+ \lambda$ times something equal to zero and we had also mention that this something will be the gradient of the objective function in this problem the objective function is strain energy objective function.

Where is a gradient it with respect to the design variable in this case if the objective function what we have here is $\partial SE / \partial A$ that is what this will be similarly this is the constraint that is a ∂ volume by ∂A in general this will be objective function gradient this will be constrained radiant this is a gradient of the constraint functional type of constraint that we have which is volume that is what we have in general this is not strain energy this will be objective function here it is strain energy and normally constraint but here the constraint is the volume.

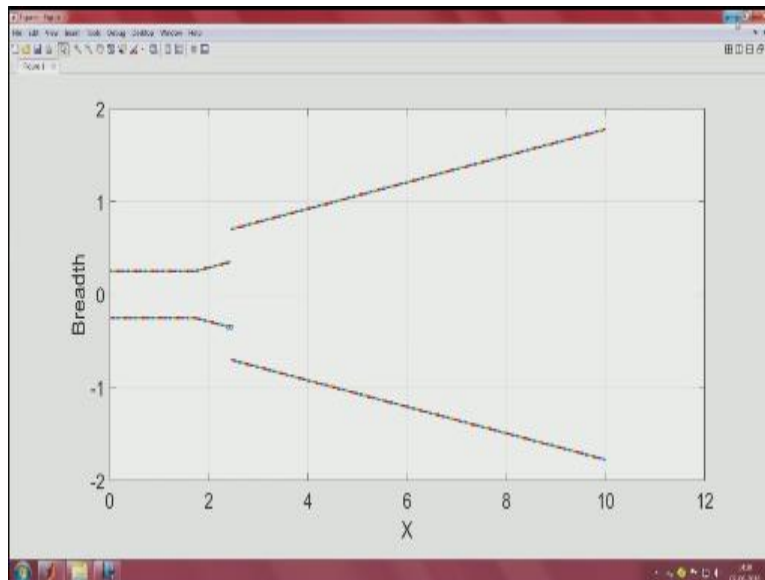
So here it is strain energy in this problem and this is volume that is what we get okay and from here by knowing that in this particular problem λ is equal to U that is a joint variable on Lagrange multiplier $\lambda(x)$ is same as $U(x)$ we got an equation that were design equation which became simple enough to interpret saying that strain should be constant everywhere U' equal to constant everywhere okay.

So basically that how we solved the EA (x) in this problem okay again substituting that back into the equilibrium equation in general that may not be possible that is interpreting that U' equal to constant and using that in the equilibrium coercion and solving for A is a function that we are given $P(x)$ is not integral or difficult to do it in which case what we do is to use iterative scheme where area of cross section let us say in iteration $k+1$ we would obtain it from area of cross section from k + we would add this thing when I say that whole thing which is the design equation which should be 0.

If we have indeed found a an area cross section that makes this 0 we are done even if you go to the next iteration $K+1$ your area cross and k at iteration be same as area of cross section at $k+v$ when it is not 0 it will get there meaning that if we iteratively do it as K increases these two will be the same so these two will be the same will be the same if this by this I mean that thing if this is 0 and that is what we want here right and that is why it makes sense to say that if this method of iterating we call it actually fixed point method which we had discussed last time.

Let us look at this problem let me first just run it and show what we get okay and I run it is working on it gave us a result like this okay here x let me make it a little bigger.

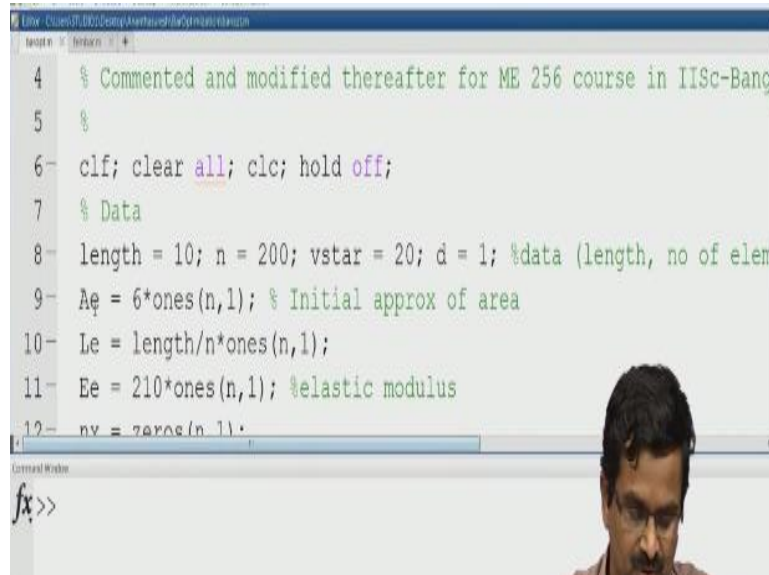
(Refer Slide Time: 12:47)



x here goes from 0 to 10 that means that we have taken a bar of 10 units long you can call it meters if you will and on this side we are showing the area of cross section the y-axis are sorry we are showing the breadth of the rectangular square whatever cross section that bread is being shown on they-axis here for this problem the optimal cross-section profile looks like this break that you see is basically there should be a lines here okay.

So joining here and here there is a jump okay there will be a reason as to why the jump is occurring in the data that we have given.

(Refer Slide Time: 13:29)



```
4 % Commented and modified thereafter for ME 256 course in IISc-Bang
5 %
6 clf; clear all; clc; hold off;
7 % Data
8 length = 10; n = 200; vstar = 20; d = 1; %data (length, no of elem
9 Aq = 6*ones(n,1); % Initial approx of area
10 Le = length/n*ones(n,1);
11 Ee = 210*ones(n,1); %elastic modulus
12 nv = zeros(n,1);
```

So now go back to the code and then see what we are doing but you will have the code available to you on the website so you can take it this was written by one of my students Gaurav Nair in 2011 all students who take this class write their own codes I am sure you also will be able to do it just is given to you so that you can see how it works there are some basic housekeeping commands as we call them clearing the figures clearing all variables and cleaning the screen and hold off on the figure so that if you draw something new previous one will not be there when I do it the data for this problem we are taking length of the bar to be 10 units okay.

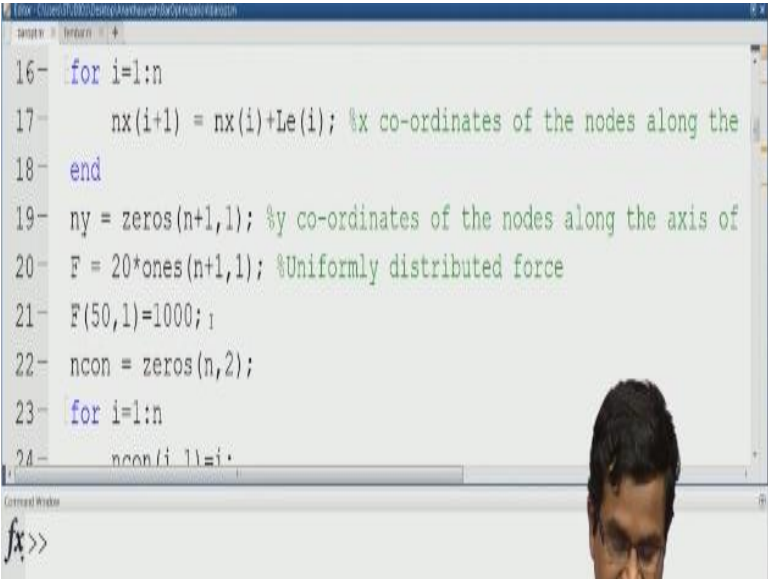
That is 10 units and we are discretized in the bar to 200 let us remember that whenever we do structure optimization a computer we do not do analytical we do it discretized way that is we have broken the bar into 200 parts and trying to make the area of cross section of each of these Bar let or bar elements to be different from one another that is what we are doing even though a continuous function $A(x)$ we got when you do it in the computer you discretize undo it that

refers to number of discretization and v^* is our volume that we are taking 20units here and the d which is the diameter.

If you take circular cross-section that that is a rectangular section here the depth is taken that is fixed not diameter here depth of the cross section is fixed as you saw the bread was the variable so area of cross section we are defining initial approximate area all once okay 1, 6 right that is being taken n is the number of elements we are just taking six and then the length of each element being computed elastic modulus is taken you can change these values as you wish.

And then defining several things we also have upper and lower bounds for the area of cross section of the bar which we did not discuss but we will be doing it later so we have upper and lower bounds and some tolerance for making the algorithm stop when a certain condition is satisfied and you have to go through the code yourself to see how this reflects what we have derived okay I will point to the critical place where our fixed point method is in action.

(Refer Slide Time: 16:05)



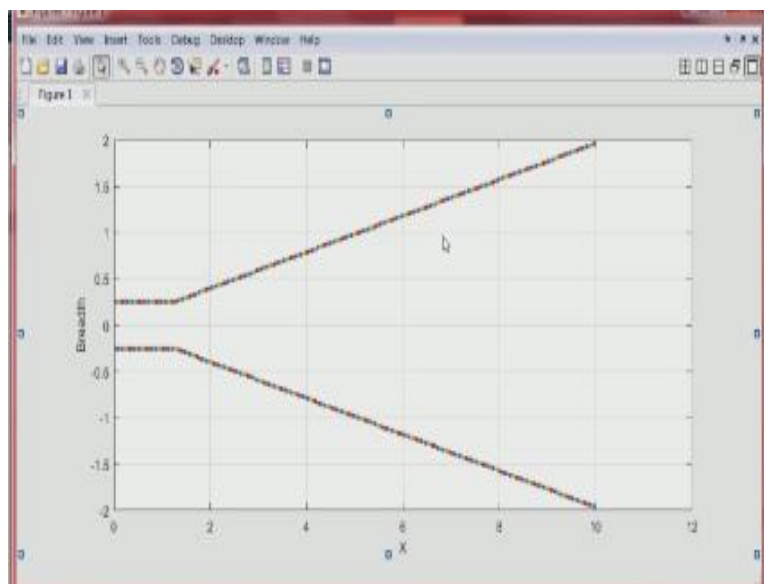
```
16- for i=1:n
17-     nx(i+1) = nx(i)+Le(i); %x co-ordinates of the nodes along the
18- end
19- ny = zeros(n+1,1); %y co-ordinates of the nodes along the axis of
20- F = 20*ones(n+1,1); %Uniformly distributed force
21- F(50,1)=1000; ;
22- ncon = zeros(n,2);
23- for i=1:n
24-     ncon(i,1)=i;
```

So we are specifying the forces in this case force at 51 that is we have the 50 is the node number there we are specifying thousand but I can change we just ran it so what I will do is I will change

I will comment it out okay I will comment out this line over here okay that is commented now in MATLAB when you put a percentage sign that line now what we have is we have distributed force is that okay everywhere 20 units.

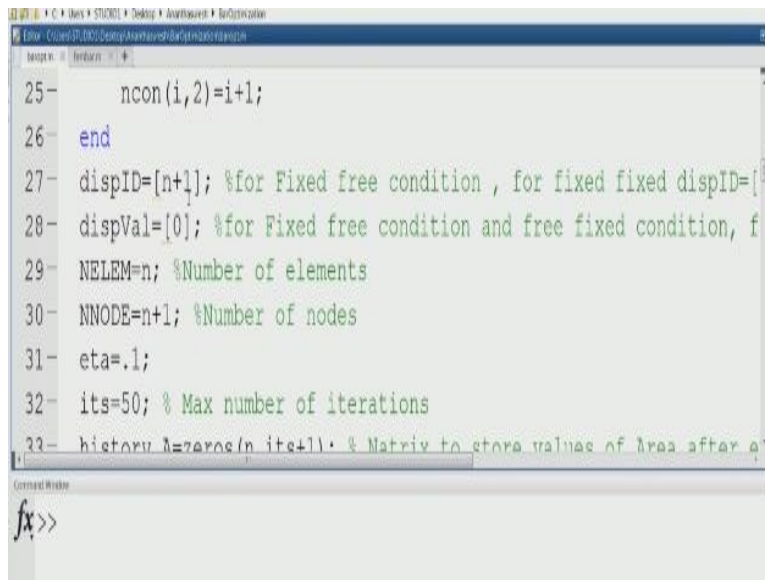
Now let us run this all right so this is bar opt one time let us say one run code and measure excellent I am okay this is asking for something more okay let us see I will just run this code does it run like this or we can just type the bar up here but they command prompt so it should be running now we have changed the loading now if we go to the finger file right so if you if you remember how it was earlier to now because we change the conditions here that is the force is uniform.

(Refer Slide Time: 17:47)



Now what was there now is not optimal solution we had different one this is the a minimum that we have and then if you just tapers up like this what are the boundary conditions where is it fixed let us check that okay so let us go and check what we are fixing here is despite $n + 1$ we have n right side is what is being fixed to a value 0 here okay that is why when we go here this one it is fixed on this side and it is free on this side.

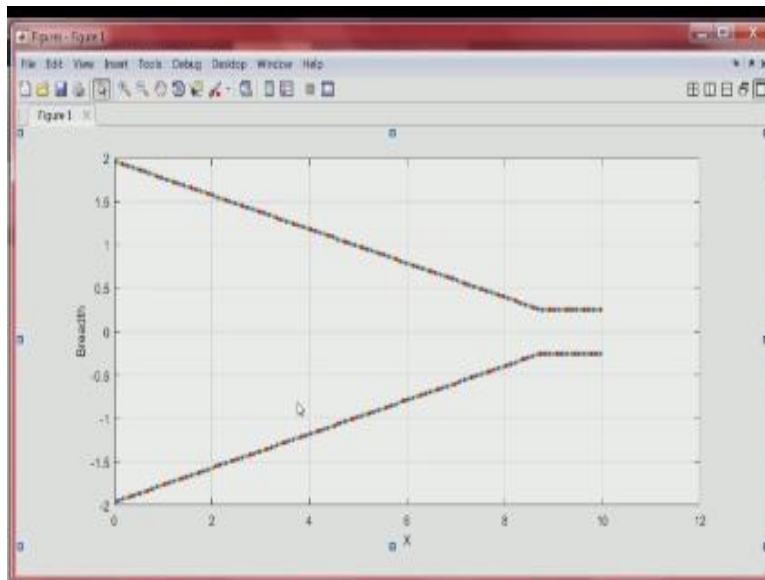
(Refer Slide Time: 18:18)



```
25-     ncon(i,2)=i+1;
26- end
27- dispID=[n+1]; %for Fixed free condition , for fixed fixed dispID=[
28- dispVal=[0]; %for Fixed free condition and free fixed condition, f
29- NELEM=n; %Number of elements
30- NNODE=n+1; %Number of nodes
31- eta=.1;
32- its=50; % Max number of iterations
33- history A=zeros(n its+1); % Matrix to store values of Area after a'
```

So free fixed case of a bar and that is why we got a solution that looks like this now I will do we do is I will switch that is I will fix the left end and free the right hand for that I need to go here and make this one that is the first node is what you will fix to zero value then the right one will be free right so here let us run this when this command prompt appears in the command window then that yeah it is done now right.

(Refer Slide Time: 19:20)



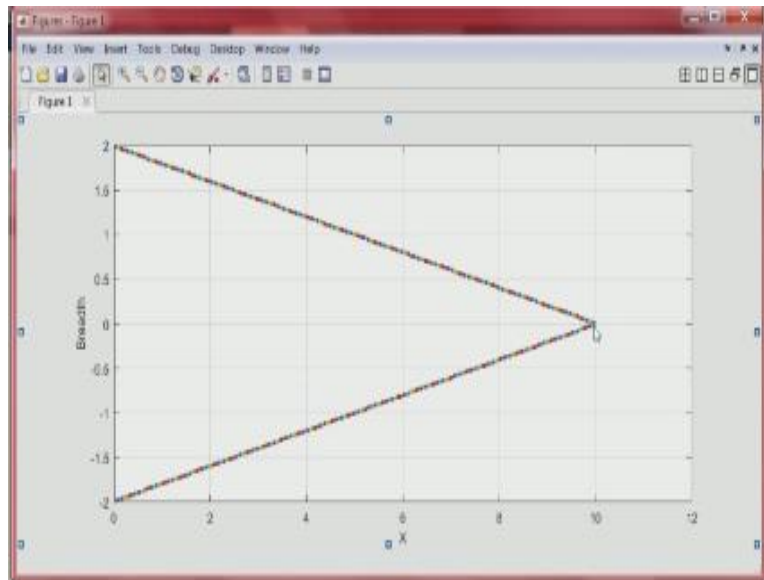
So if we go back and look at the figure what do you see when I fix it here and free it here it comes like this if you remember analytically that is what we had observed when you can I give uniform distributed load we get a linearly tapering down profile for the area of cross section large area here goes down it could have gone to zero that was our condition but now we put a minimum here that is why it getting there to show you that it works let me make that a minimum actually zero.

(Refer Slide Time: 19:55)

```
10- Le = length/n*ones(n,1);
11- Ee = 210*ones(n,1); %elastic modulus
12- nx = zeros(n,1);
13- Amax = 10; %upper bound on area
14- Amin = 0; %lower bound on area
15- tol = 1e-3; %tolerance to stop iteration
16- for i=1:n
17-     nx(i+1) = nx(i)+Le(i); %x co-ordinates of the nodes along the
18- end
```

Let us go up here there is a min which was which is now 0.5 let us make it actually 0 okay wherein on you now it is running command prompt has to come back here to indicate that the program has finished running which it has now so let us look at this so 10 it has indeed gone to zero cross section this is what we had solved analytically but now we have a numerical one when you have a numerical program we can put any boundary condition and any loading we can put the loading whichever way we want.

(Refer Slide Time: 20:06)



So what I will do is here the force which I had commented let me remove that thing and low load there okay here I will put the load at the last point the rightmost point we have fixed that at left one I made it $n + 1$ meaning that is the $n + 4$ node the last one extreme right node that is we are applying thousand and everywhere else I have zero okay force let us see what solution comes up here again it is running it finished we got a solution like this.

So here it is uniform cross section that actually makes sense if you want to make a stiffer structure for a fixed free bar with a constant load of the point load at the right hand the best solution is simply uniform cross section that is what we got with the volume that we have now the volume is two times the length times the depth that we have specified for the cross section okay so that way once we have this we can play with instead of applying their total we had 200 elements okay let us say I apply somewhere as we had here 50 50th node okay.

If I run it so it is giving some errors because something went in MATLAB in case n is not a number okay what we have given 50th node here for that what we are giving let us see what the figure also it is not giving because it actually says that somewhere matrix is singular so you have to check why that matrix is becoming singular meaning a determinant is zero so the loading that I am giving at the 50th node somehow is causing the matrix to become singular In the program somewhere okay.

So let me change it back to what we had which is $n + 1$ that means whenever this program does not runs an error if you can look at it but know that somewhere something is wrong with the input given whatever input is proper we got back a solution okay, so let me comment that that is I do not want to even n load and we can make it again the constant load we have we have played with the if I look at the displacement boundary condition okay.

Which is over here we fixed node 1 let me also fix the last one $n + 1$ that means our displacement value I have to give another 0 now it becomes fixed-fixed bar let us see what we get it is running as you can see command window it is done now so we got something like this okay when in fact fixed- fixed bar should be slightly different from it let me see if I have done it right 1 and $n + 1$ and then 00 we have the loading have to see f value 20 all of this okay.

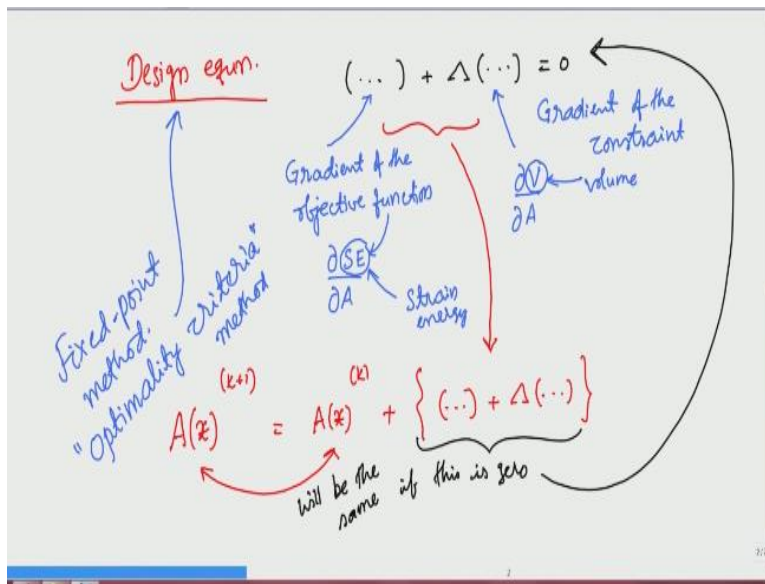
Fixed- fixed condition displacement 1 and $n + 1$ node okay let me that that we can actually see these values despite ID that is a variable okay one and two hundred two hundred and one so that is correct this value is also 00 so that part is fine iterations are self 250 the force we have is 20 so it is giving it is running something but the result will analyze it later let me point out the code now so if you see the loop starts here 12 ITS is number of iterations initially the λ that we have that is the volume construct a multiplier initialized to 0.

Here is where we are calling the finite element bar finite element routine okay and then you dash that returns the displacement state basically this \emptyset m bar routine solves the equilibrium equation and returns u and few others reactions and internal forces and meet in the stiffness matrix and energy and all that we are getting u and then we are computing u' here using finite

difference method and then here is where we are updating λ that we have seen the λ has to be computed using the volume constraint.

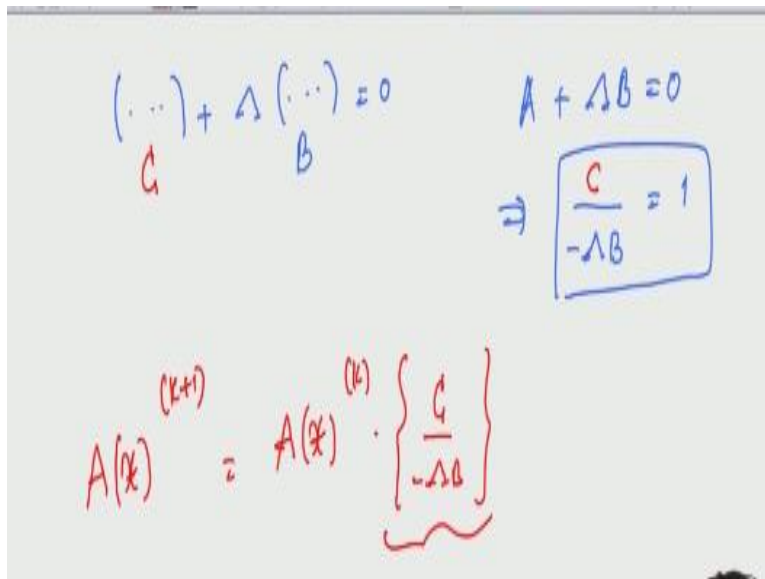
And that is being done over here okay, so the λ is being computed there that the Lagrange multiplier after we have that the λ which is computed then you zoom here we are updating the area of cross section okay ΔA @ j theta rays for all the elements j equal to 1 to n we are using the current j and multiplying something raised to some later.

(Refer Slide Time: 26:02)



So if I go back and show our algorithm if you see here we are adding what should be 0 instead we could also do the following.

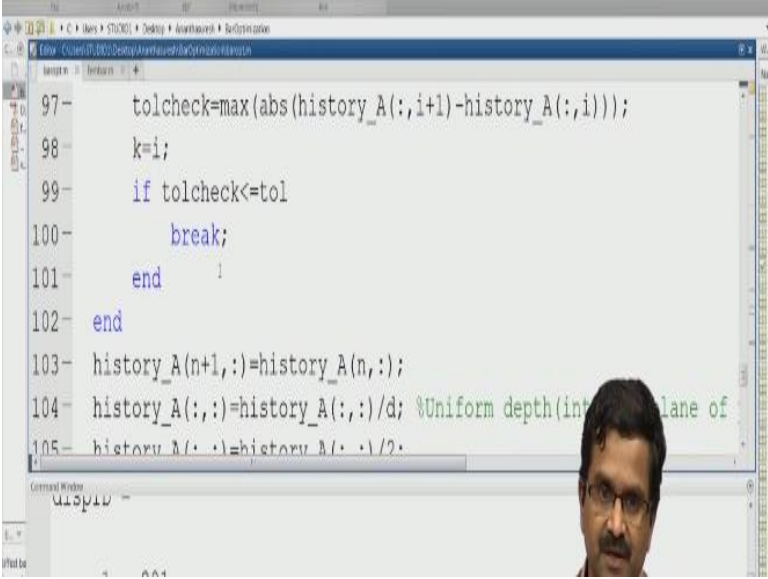
(Refer Slide Time: 26:13)


$$\begin{aligned} & \begin{pmatrix} \dots \\ C \end{pmatrix} + \lambda \begin{pmatrix} \dots \\ B \end{pmatrix} = 0 & A + \lambda B = 0 \\ & \Rightarrow \boxed{\frac{C}{-\lambda B} = 1} \\ & A(x)^{(k+1)} = A(x)^{(k)} \cdot \underbrace{\left\{ \frac{C}{-\lambda B} \right\}} \end{aligned}$$

So we have the design equation something $+ \lambda x$ something equal to zero let us call this you know as let us call this some A and B so what I have is $a + \lambda b$ is equal to zero I can write this as $A / - \lambda B = 1$ basically I have taken λ to the other side and got it downstairs so I have this if I have this I can do my area of cross section anyway this a is not area of cross I just used a now if I write my area of cross section at K^{th} iteration if I want instead of adding something that should be made 0 I can multiply by something that should eventually become 0.

That is this $a x - \lambda p$ again this a you may get confused so let me actually put some A^{\wedge} or something so that that a is not same as this a hat that we have here okay or better it let me just you know so that there is no confusion at all let us just call it subsea then just call let us just calling that as C and C okay we are multiplying with what should become one this is another way of doing it and that is what we are doing in this case here.

(Refer Slide Time: 27:56)



```
97-   tolcheck=max(abs(history_A(:,i+1)-history_A(:,i)));
98-   k=i;
99-   if tolcheck<=tol
100-       break;
101-   end
102- end
103- history_A(n+1,:)=history_A(n,:);
104- history_A(:,:)=history_A(:,:)/d; %Uniform depth(int... lane of
105- history_A(:,:)=history_A(:,:)-1/2;
```

So we are multiplying that raised to some data to make it faster right and then we are updating the areas of cross section here and then if anything has exceeded a max we are setting at a max anything goes below A min we are setting at A min and running the algorithm and in that process we have to do this as long as this flag that we have that is he is not going above the upper bounded below the lower bound and trying to adjust it and get here okay.

This is just a starting program that you should run understand each line compared with the algorithm we have discussed and learned how to right struck stretch optimization routines on your own once you have finite element routine for that here we have FM bar which is the finite element routine for analyzing bar that is to solve the governing equation of the bars then you can write this code then you have structure optimization for A bar okay so study this code and we

will be coming back to this type of code in the case of beams and other structures as well we will stop here and continue the next half.