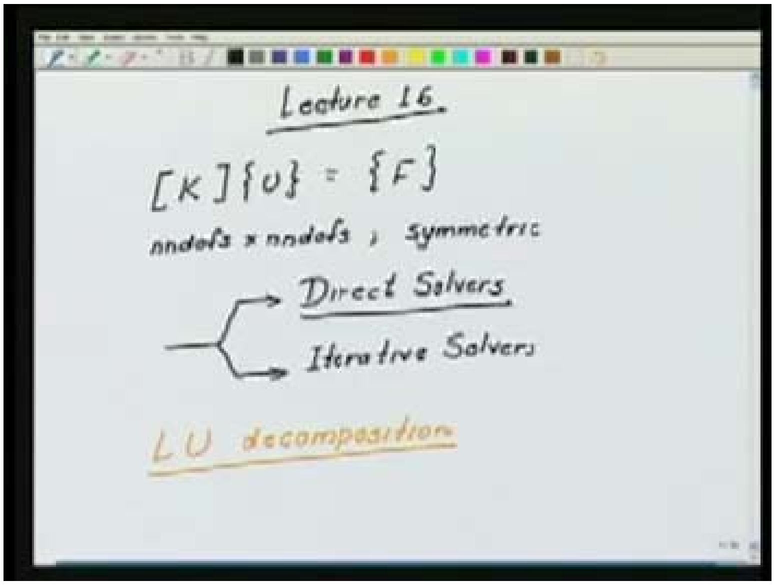
# Finite Element Method

**Prof. C. S. Upadhyay**
**Department of Mechanical Engineering**
**Indian Institute of Technology, Kanpur**

## Module – 5 Lecture - 3

In this lecture, we are going to continue our 1D programming assignments. We had stopped at the assembly of the element equations their application of the boundary conditions and the point load for the one dimensional problem. We had raised a question of how to solve the resulting system that we had, which is of the form, K U is equal to F and K was of dimension nndofs by nndofs and it was symmetric.

(Refer Slide Time: 00:18)



Lecture 16

$[K]\{u\} = \{F\}$

nndofs x nndofs ; symmetric

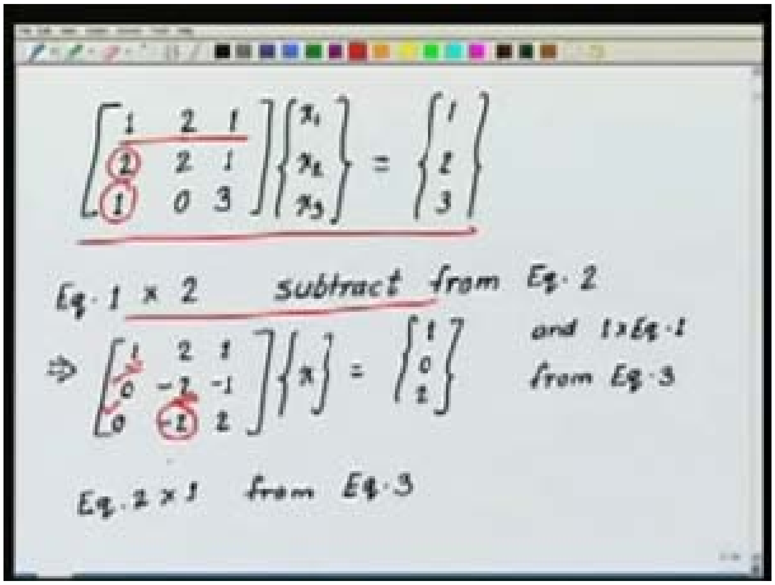
Direct Solvers

Iterative Solvers

LU decomposition

As I had mentioned in passing at the end of the last lecture, there are 2 types of solvers which are basically popular in engineering analysis, especially when we use FEM. One class of solvers is called the Direct Solvers. The other class is called the Iterative Solvers. The direct solvers work by direct elimination of the unknowns to get the solution. The iterative solver, it works by first taking an initial guess of this U and then trying to correct the solution then add correction to this initial guess to get to the solution of the problem. Each one has its own advantage but for our case, (Refer Slide Time: 02:09) for the system that we are interested in, let us look at only the direct solvers. Within this class of direct solvers, we will look at an important category of LU decomposition based solvers. What do we mean by LU decomposition? Let us see this through an example.

(Refer Slide Time: 02:36)



$$\begin{bmatrix} 1 & 2 & 1 \\ 2 & 2 & 1 \\ 1 & 0 & 3 \end{bmatrix} \begin{Bmatrix} x_1 \\ x_2 \\ x_3 \end{Bmatrix} = \begin{Bmatrix} 1 \\ 2 \\ 3 \end{Bmatrix}$$

Eq. 1 × 2 subtract from Eq. 2 and 1×Eq. 1 from Eq. 3

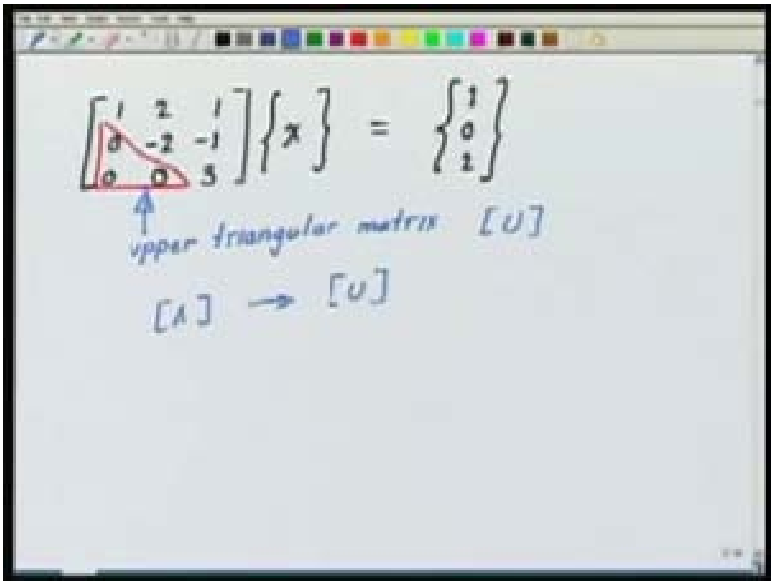
$$\Rightarrow \begin{bmatrix} 1 & 2 & 1 \\ 0 & -2 & -1 \\ 0 & -2 & 2 \end{bmatrix} \{x\} = \begin{Bmatrix} 1 \\ 0 \\ 2 \end{Bmatrix}$$

Eq. 2 × 1 from Eq. 3

Let us take this matrix problem which is a 3 by 3 problem given by this kind of a system. Deliberately, I have made the matrix A or the K here a non symmetric, this is a general problem (Refer Slide Time: 02:58). When we have to do Gauss elimination then we simply go and subtract; I want to eliminate first the $x_1$ variable from the remaining two equations not from the first one. I simply subtract twice the first equation from the second equation. To do this job, I do this and I get this one (Refer Slide Time: 03:25). I subtract once the first equation from the third equation. By doing that, I have reached this form, where I have eliminated $x_1$ from this equation and this equation that is, from the second and third equation. First variable is eliminated from all the remaining equations. This is how, if we remember, the Gauss elimination is done.

Once we have eliminated $x_1$ from the remaining two equations then we want to eliminate $x_2$ from the third equation so we are going downwards. From the second equation, we take this diagonal (Refer Slide Time: 04:05) and we would like to eliminate everything under this diagonal. Like in the first equation, we have eliminated everything under the first diagonal. These diagonal elements are called the pivots and we are going to simply eliminate this one from here, by simply taking the second equation and subtracting it from the third one. If I subtract the second equation from the third one, I will get to this form.

(Refer Slide Time: 04:34)



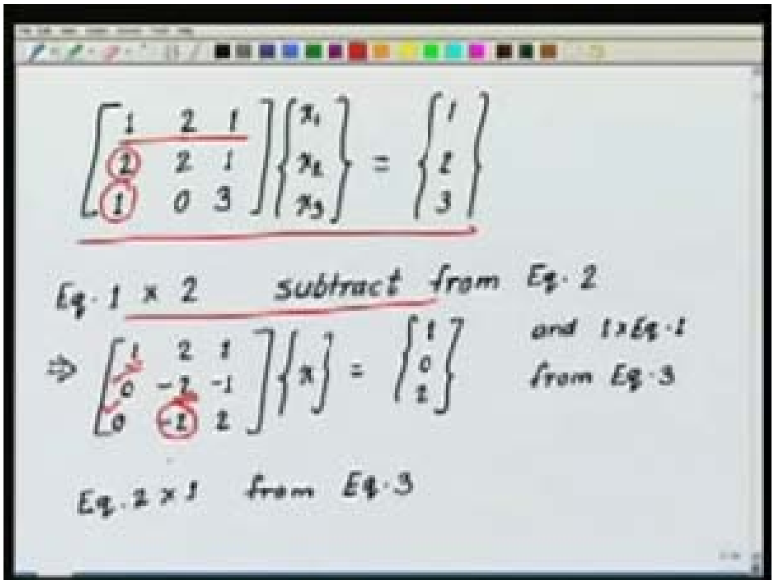
$$\begin{bmatrix} 1 & 2 & 1 \\ 0 & -2 & -1 \\ 0 & 0 & 3 \end{bmatrix} \{x\} = \begin{Bmatrix} 1 \\ 0 \\ 2 \end{Bmatrix}$$

upper triangular matrix $[U]$

$[A] \rightarrow [U]$

1, 2, 1, 0, minus 2, minus 1, 0, 0, 3, x is 1, 0, 2. If we see this one, this has 0 entries in the part of the matrix below the diagonal. This type of a matrix where everything below the diagonal is 0 is called an upper triangular matrix. We will call this by a name U. We want to know how through a series of operations can we take A and get U. We have done those operations. How can we write these operations in terms of matrix manipulations?

(Refer Slide Time: 05:45)



$$\begin{bmatrix} 1 & 2 & 1 \\ 2 & 2 & 1 \\ 1 & 0 & 3 \end{bmatrix} \begin{Bmatrix} x_1 \\ x_2 \\ x_3 \end{Bmatrix} = \begin{Bmatrix} 1 \\ 2 \\ 3 \end{Bmatrix}$$

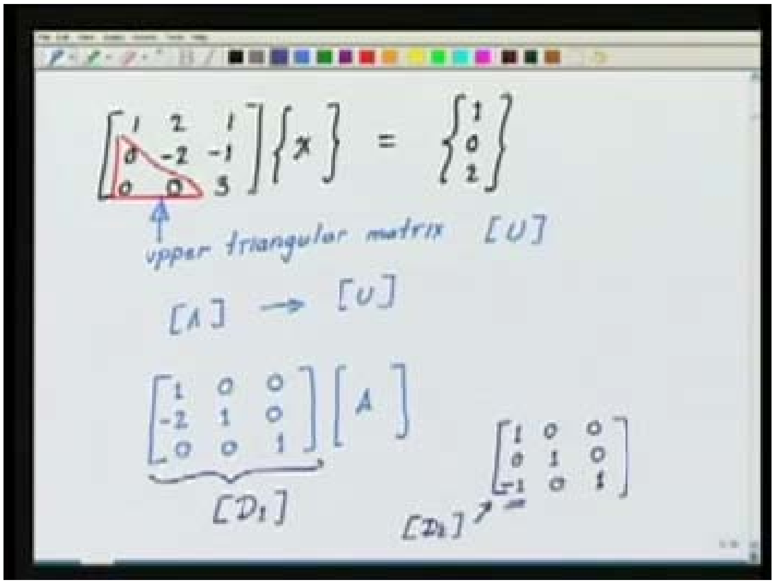
Eq. 1 × 2 subtract from Eq. 2 and 1 × Eq. 1 from Eq. 3

$$\Rightarrow \begin{bmatrix} 1 & 2 & 1 \\ 0 & -2 & -1 \\ 0 & -2 & 2 \end{bmatrix} \{x\} = \begin{Bmatrix} 1 \\ 0 \\ 2 \end{Bmatrix}$$

Eq. 2 × 1 from Eq. 3

If we go back, here the first operation that we did was we took twice the first equation and subtracted it from the second one.

(Refer Slide Time: 06:04)



$$\begin{bmatrix} 1 & 2 & 1 \\ 0 & -2 & -1 \\ 0 & 0 & 3 \end{bmatrix} \{x\} = \begin{Bmatrix} 1 \\ 0 \\ 2 \end{Bmatrix}$$

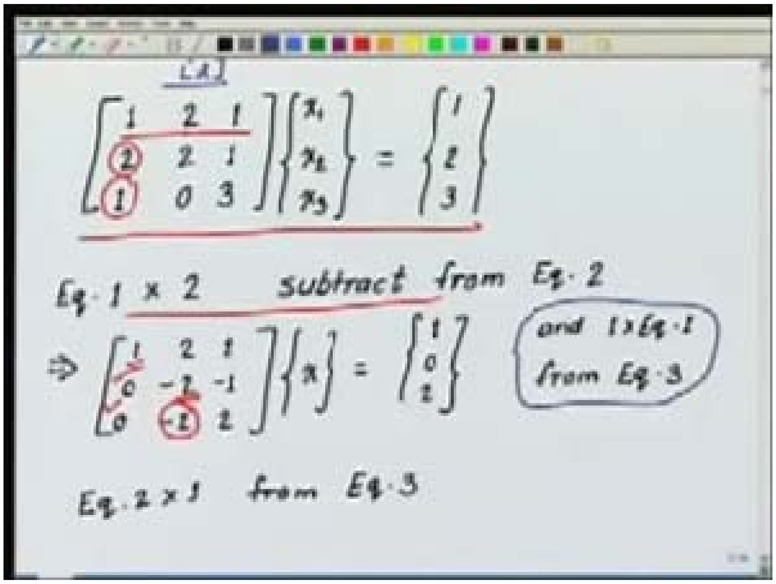
upper triangular matrix [U]

$$[A] \rightarrow [U]$$

$$\underbrace{\begin{bmatrix} 1 & 0 & 0 \\ -2 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}}_{[D_1]} [A] \qquad \underbrace{\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -1 & 0 & 1 \end{bmatrix}}_{[D_2]}$$

This operation I would like to write through a matrix that I leave the first equation unchanged. I subtract twice the first equation from the second one. I do not do anything third one here and I am leaving the third one fixed. This (Refer Slide Time: 06:22) if I multiply it with A does the same job as subtracting twice the first equation from the second one of the matrix A. This one I am going to call with a name matrix $D_1$. Similarly, if I go and look at the second part, second part from is here. To eliminate $x_1$ from the third equation, I take once the first equation that is the first equation itself and subtract it from the third equation. This part will be given by a matrix; I have the first equation which remains unchanged. I am doing nothing to the current second equation. It remains unchanged and I subtract once the first equation from the third one. This is the subtraction process this (Refer Slide Time: 07:35) I am going to call as the matrix $D_2$.

(Refer Slide Time: 07:46)



$[A]$

$$\begin{bmatrix} 1 & 2 & 1 \\ 2 & 2 & 1 \\ 1 & 0 & 3 \end{bmatrix} \begin{Bmatrix} x_1 \\ x_2 \\ x_3 \end{Bmatrix} = \begin{Bmatrix} 1 \\ 2 \\ 3 \end{Bmatrix}$$

Eq. 1 × 2 subtract from Eq. 2

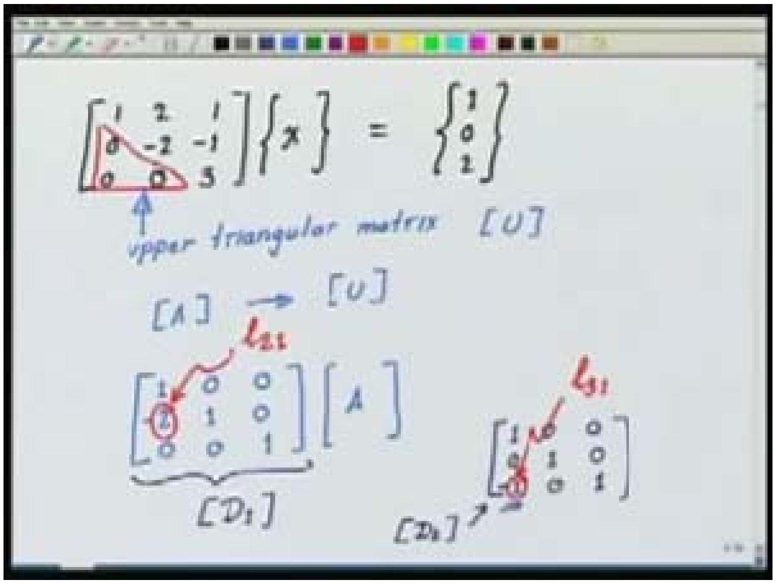
and 1 × Eq. 1 from Eq. 3

$$\Rightarrow \begin{bmatrix} 1 & 2 & 1 \\ 0 & -2 & -1 \\ 0 & -2 & 2 \end{bmatrix} \{x\} = \begin{Bmatrix} 1 \\ 0 \\ 2 \end{Bmatrix}$$

Eq. 2 × 1 from Eq. 3

Similarly, the third part was, after we have done this, we have a resultant modified matrix A. This is matrix A (Refer Slide Time: 07:57). I go and take the second equation and subtract it from the third equation.

(Refer Slide Time: 08:12)

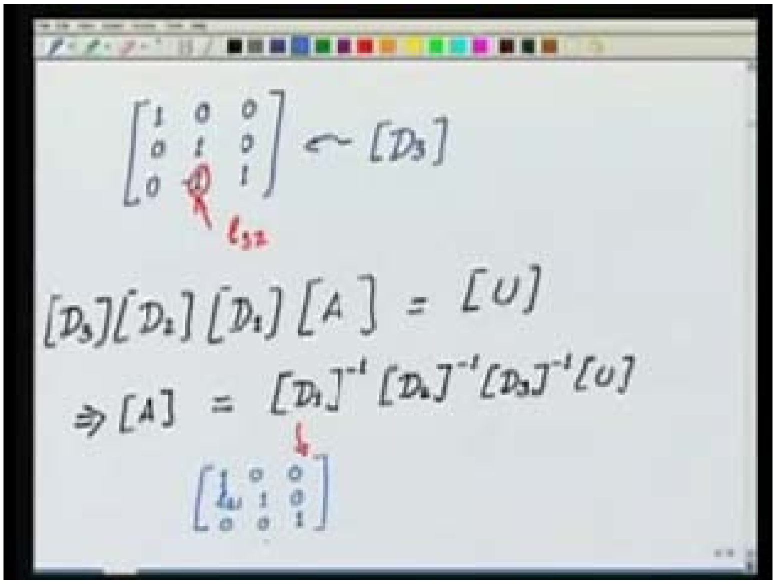


$\begin{bmatrix} 1 & 2 & 1 \\ 0 & -2 & -1 \\ 0 & 0 & 3 \end{bmatrix} \{x\} = \begin{Bmatrix} 1 \\ 0 \\ 2 \end{Bmatrix}$
upper triangular matrix [U]
$[A] \rightarrow [U]$
$l_{21}$
$\begin{bmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} [A]$
$[D_1]$
$l_{31}$
$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 1 & 0 & 1 \end{bmatrix}$
$[D_2]$

How do I write that? This subtraction process can be written in the form. I leave the first equation unchanged. I leave the second equation unchanged and in third equation, I am going to subtract the second one from it. So I take the third equation from it to subtract the second one. This is the process of doing this job. This is given by the matrix $D_3$. In terms of these three matrices, now can I write U? Yes, I can. If I take the matrix A, first I operate on it with the matrix $D_1$ that is the first subtraction of twice the first row from the second row. Then I do the second operation $D_2$ that is, subtraction of the first row from the third row. Then I do the third operation $D_3$, subtraction of the second row from the third row. This job should give me the upper triangular form U. If I want to now write A in terms of U is equal to, I will take the inverse of this set. Inverse of this will be $D_1$ inverse into $D_2$ inverse into $D_3$ inverse into U, finding the inverses of these matrices is very simple. Let us see how simple.

This quantity I am going to call by a name. I am subtracting this quantity from the essentially, tells me by a how much amount, I multiplied the first equation to subtract from the second one. This one I am going to call $l_{21}$. Similarly, this quantity I am going to call as $l_{31}$ (Refer Slide Time: 10:30).

(Refer Slide Time: 10:40)



$$\begin{bmatrix}1&0&0\\0&1&0\\0&0&1\end{bmatrix} \leftarrow [D_3]$$

$l_{32}$

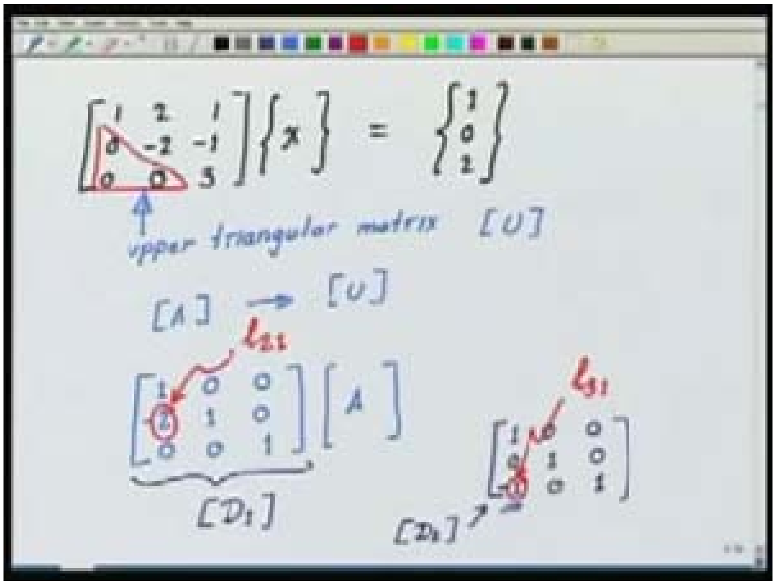
$$[D_3][D_2][D_1][A] = [U]$$

$$\Rightarrow [A] = [D_1]^{-1}[D_2]^{-1}[D_3]^{-1}[U]$$

$$\begin{bmatrix}1&0&0\\l_{21}&1&0\\0&0&1\end{bmatrix}$$

This quantity I am going to call as $l_{32}$, it tells me from the third equation. This is the amount by which I multiply the second equation and subtract. It turns out that if I look at $D_1$ inverse, it is quite simple. $D_1$ inverse will remain 1, 0, 0. We had subtracted twice the first equation from the second. I simply have $l_{21}$, 1, 0, 0, 0, 1.

(Refer Slide Time: 11:27)



$$\begin{bmatrix}1&2&1\\0&-2&-1\\0&0&5\end{bmatrix}\{x\} = \begin{Bmatrix}1\\0\\2\end{Bmatrix}$$

upper triangular matrix $[U]$

$$[A] \rightarrow [U]$$

$l_{21}$

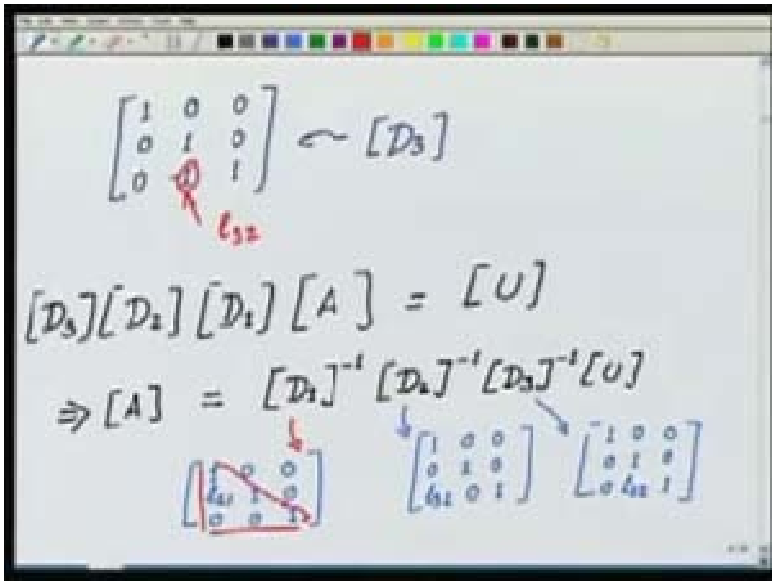
$$\underbrace{\begin{bmatrix}1&0&0\\2&1&0\\0&0&1\end{bmatrix}}_{[D_1]}[A]$$

$l_{31}$

$$[D_2] \rightarrow \begin{bmatrix}1&0&0\\0&1&0\\0&0&1\end{bmatrix}$$

Here I have taken $D_1$ and simply taken this part which corresponds to subtraction and taken the negative of that, put it there, back into the matrix that becomes the inverse which as simple that.

(Refer Slide Time: 11:48)



$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \leftarrow [D_3]$$

$l_{32}$

$$[D_3][D_2][D_1][A] = [U]$$

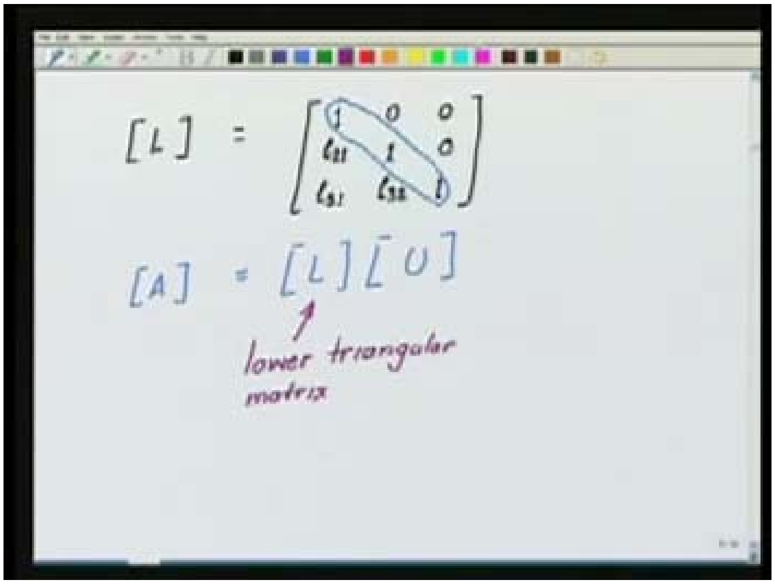
$$\Rightarrow [A] = [D_1]^{-1}[D_2]^{-1}[D_3]^{-1}[U]$$

$$\begin{bmatrix} 1 & 0 & 0 \\ l_{21} & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ l_{31} & 0 & 1 \end{bmatrix} \quad \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & l_{32} & 1 \end{bmatrix}$$

Similarly $D_2$ inverse will be 1, 0, 0. Here I will have $l_{31,}$ 0, 1. The third one will be 1, 0, 0, 0, 1, 0, 0, $l_{32}$, 1. So finding the inverse of these matrices $D_1$, $D_2$, $D_3$ is very easy. Once I know what the operation through which I can to this form.

These matrices have a nice property that these are only concentrated in the lower triangular part of the matrix that is, everything above the diagonal is 0, everything below the diagonal is non zero. The product of these three things will be matrix L which will have once on the diagonal $l_{21}$ in the half diagonal here.

(Refer Slide Time: 13:04)

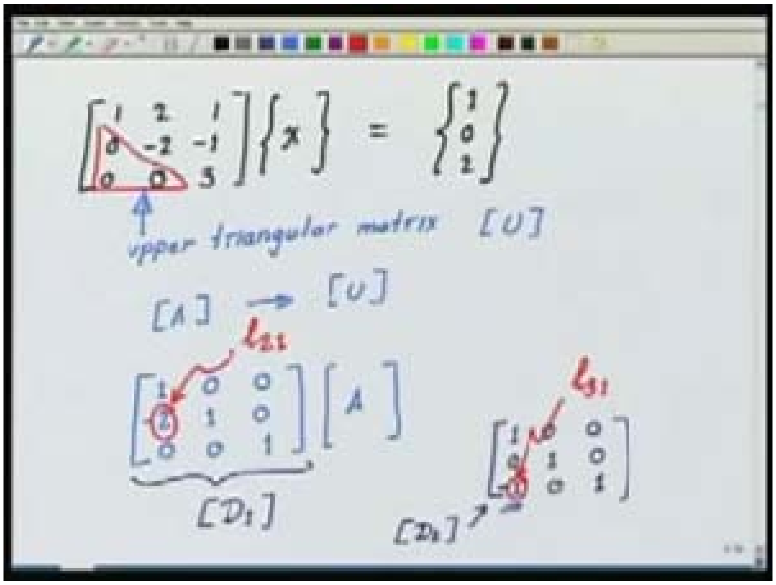


$$[L] = \begin{bmatrix} 1 & 0 & 0 \\ l_{21} & 1 & 0 \\ l_{31} & l_{32} & 1 \end{bmatrix}$$

$$[A] = [L][U]$$

$l_{31}$ in the half diagonal here. $l_{32}$ in the half diagonal here, 1 here. Now I have obtained a lower triangular matrix and the diagonals are all 1. This is the beautiful property of this matrix L (Refer Slide Time: 13:35). I can write the matrix A as a lower triangular matrix L into an upper triangular matrix U. Let me get back and look at what was U.

(Refer Slide Time: 14:10)



$$\begin{bmatrix} 1 & 2 & 1 \\ 0 & -2 & -1 \\ 0 & 0 & 5 \end{bmatrix}\{x\} = \begin{Bmatrix} 1 \\ 0 \\ 2 \end{Bmatrix}$$

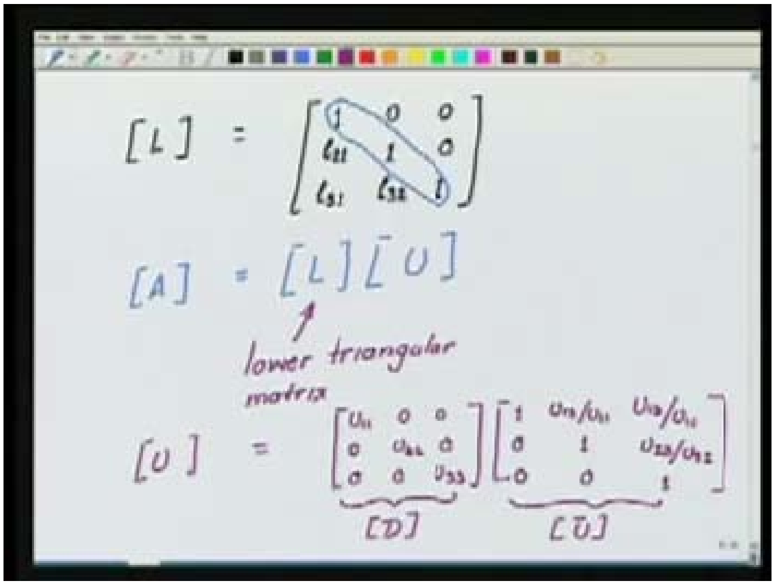
$$[A] \rightarrow [U]$$

$$\underbrace{\begin{bmatrix} 1 & 0 & 0 \\ -2 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}}_{[D_1]}[A]$$

$$[D_2] \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -0 & 0 & 1 \end{bmatrix}$$

If we see U, the diagonal entries of U are not 0. So we can make by a suitable modification, the diagonal entries of U also 0. How can we do that?

(Refer Slide Time: 14:38)



$$[L] = \begin{bmatrix} 1 & 0 & 0 \\ l_{21} & 1 & 0 \\ l_{31} & l_{32} & 1 \end{bmatrix}$$

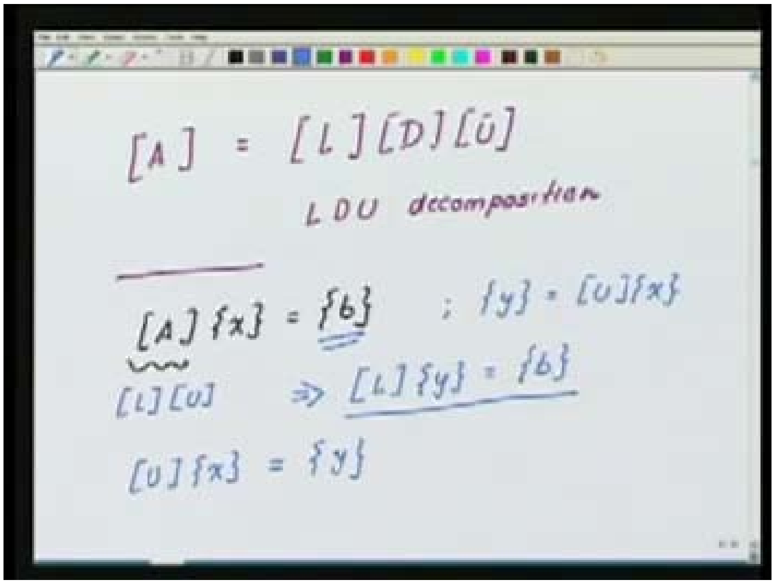
$$[A] = [L][\bar{U}]$$

lower triangular matrix

$$[U] = \underbrace{\begin{bmatrix} U_{11} & 0 & 0 \\ 0 & U_{22} & 0 \\ 0 & 0 & U_{33} \end{bmatrix}}_{[D]} \underbrace{\begin{bmatrix} 1 & U_{12}/U_{11} & U_{13}/U_{11} \\ 0 & 1 & U_{23}/U_{22} \\ 0 & 0 & 1 \end{bmatrix}}_{[\bar{U}]}$$

We can now say matrix U is equal to $U_{11}$, 0, 0, 0, $U_{22}$, 0, 0, 0, $U_{33}$ into 1, $U_{12}$ by $U_{11}$, $U_{13}$ by $U_{11}$ here I will have 0, here 1, $U_{23}$ by $U_{22}$, 0, 0, 1. This I am going to call a diagonal matrix D. I have simply taken the diagonal entries and written it in a diagonal form here and if I now multiply I should get back the U. This matrix, I am going to call it as U bar.

(Refer Slide Time: 15:52)



$[A] = [L][D][\bar{U}]$

L D U decomposition

$[A]\{x\} = \underline{\{b\}}$ ; $\{y\} = [U]\{x\}$

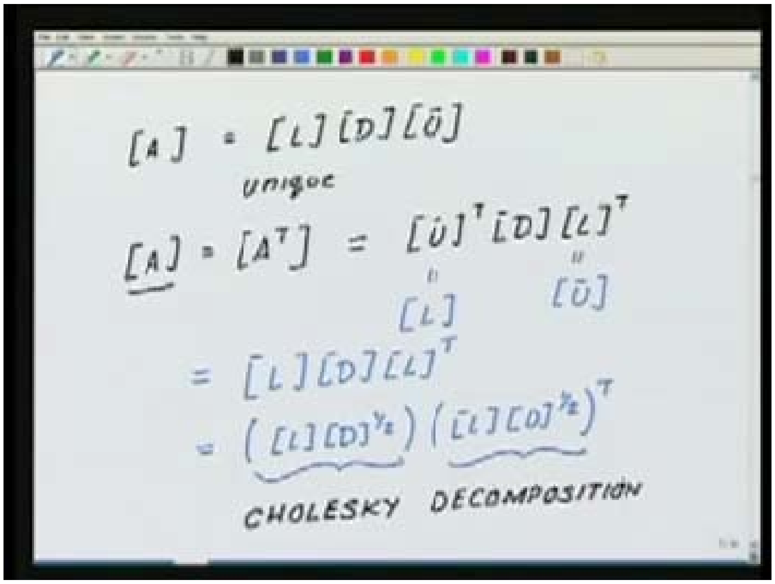
$[L][U] \Rightarrow \underline{[L]\{y\} = \{b\}}$

$[U]\{x\} = \{y\}$

I can write A as equal to L D U bar. So this is called the L D U decomposition of A. This is something that we can also do. Let us get back. What is the advantage of this? Even U bar has one on the diagonals. Before go ahead now we see how this decomposition is going to be advantageous. If I have A x is equal to b; instead of A, I write L U. I will call y is equal to U into x, implies we will have L y is equal to b. Here (Refer Slide Time: 17:10) we have the lower triangular matrix into y is equal to b, solving for y is very easy, because here it is explicit. We do not have to do any elimination explicitly. We will first start with a first term, find $y_1$ put that into the second equation, find $y_2$ and so on; solving this is very easy.

Once I have obtained y then I will obtain x as U x is equal to y. Instead of operating with A, now we can explicitly operate with the L and U. This makes life very simple because if I have a stiffness matrix, I want to change the loads applied on the structure keeping the boundary conditions of the material fixed. Then the stiffness matrix is going to remain the same. As the load is changing, the load vector which will be this side is going to change (Refer Slide Time: 18:15). When the load vector changes then I do not have to eliminate A, do a Gauss elimination on A all the time. I can have the L and U in store and simply take the new load vector operate on it with the L and U and I get a solution. So that is something we would like to have.

(Refer Slide Time: 18:39)



$$[A] = [L][D][\bar{U}]$$

unique

$$\underline{[A]} = [A^T] = \underset{\overset{\shortparallel}{[L]}}{[\bar{U}]^T}[D]\underset{\overset{\shortparallel}{[\bar{U}]}}{[L]^T}$$

$$= [L][D][L]^T$$

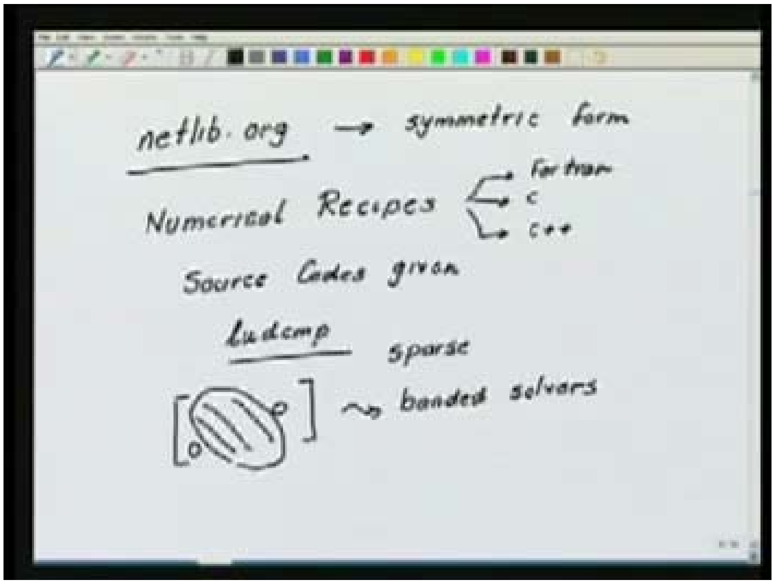
$$= \underbrace{\left([L][D]^{1/2}\right)}\underbrace{\left([L][D]^{1/2}\right)^T}$$

CHOLESKY DECOMPOSITION

Let me now get back to the L D U decomposition. We had said A is equal to L D U bar. This turns out to be unique. There is only such decomposition which is present that is, if I have another decomposition and that decomposition equals A then each of these terms have to be equal.

We see something very interesting; when A is symmetric then A is equal to A transpose due to symmetry. This is equal to U bar transpose D L transpose, but we know that there is a unique decomposition of A (Refer Slide Time: 19:20). We know that for a symmetric matrix U bar transpose is equal to L and L bar transpose is equal to U bar is just consistent. So then for these cases, for symmetric matrices, the decomposition becomes very simple. It becomes L D L transpose. This we can write as is equal to L D to the power of half into L D to the power half transpose. I take this part (Refer Slide Time: 20:32). Take the transpose of that multiply by 2. I get this decomposition for A is symmetry. This decomposition is called Cholesky decomposition. To take this information about what is L U D decomposition base solver is going to do and take a commercial version of this solver which is available in various places.

(Refer Slide Time: 21:18)



netlib. org → symmetric form

Numerical Recipes ⟨ Fortran, C, C++

Source Codes given

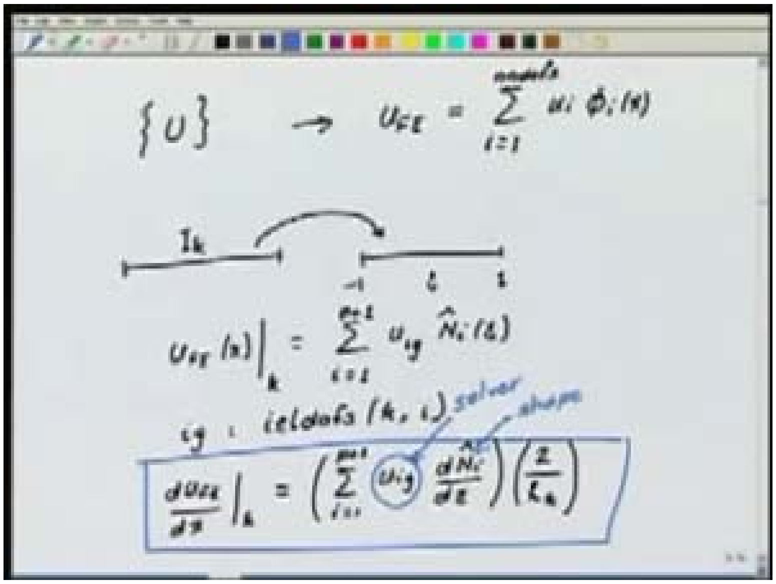
ludcmp

sparse

banded solvers

Again we can go to netlib.org, get the symmetric form because the matrix is symmetric that is L D L decomposition of the source code for this decomposition and in whichever form we want, Fortran, C or C++ or we can go to the book. This is a fantastic book in numerical analysis, numerical recipes. This is available in various versions. It is for Fortran, C and C++. On all these versions, this book is available. In these numerical recipes, source codes are given; the programs are given. We can take the program from there for the L U D decomposition; I think it comes under the name ludcmp. I would recommend use of this book to find the source codes. Once we have now obtained the solver which is the L U decomposition based certainly for us, we know our stiffness matrix is going to be banded in nature. There is going to be only a certain number of elements on either side of the main diagonal which are going to be non zero and everything else is 0. In that case, I can further reduce my cost of doing the elimination or finding the solution. We can find in these books, the banded solvers. Essentially what happens, when we have this kind of a property that the matrices sparse. If there are a lot of 0s in the matrix then I can simply store only the part which is non zero. There are these solvers which take advantage of that and the elimination also I will do using that information. So the cost of computation then comes down. One can also use the banded solvers.

(Refer Slide Time: 23:53)



$$\{U\} \rightarrow u_{FE} = \sum_{i=1}^{nnodes} u_i \phi_i(x)$$

$$u_{FE}(x)\Big|_k = \sum_{i=1}^{p+1} u_{ig} \hat{N}_i(\xi)$$

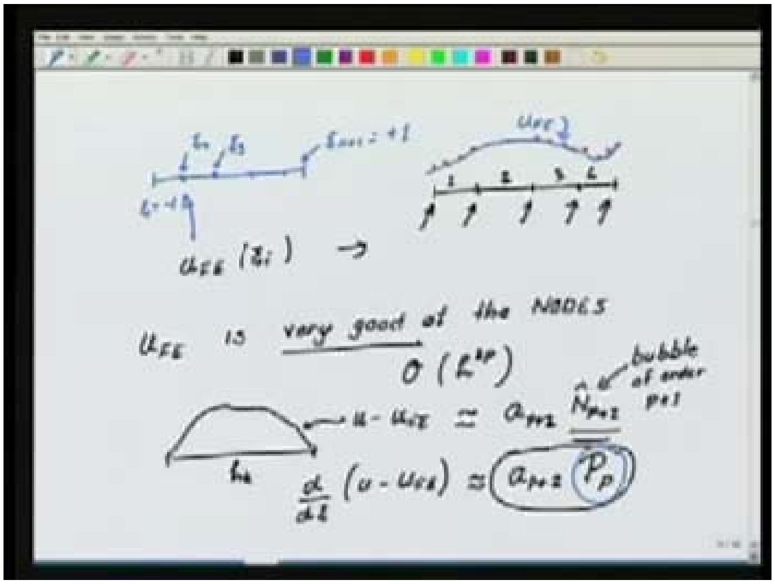
$$ig : ieldofs(k, i)$$

$$\boxed{\frac{du_{FE}}{dx}\Big|_k = \left(\sum_{i=1}^{p+1} u_{ig} \frac{d\hat{N}_i}{d\xi}\right)\left(\frac{2}{h_k}\right)}$$

We obtained the solution that is we have obtained this degrees of freedom U which implies that I should be able to now get the finite element solution, anywhere in the domain (Refer Slide Time: 24:12). Specifically, in an element k, I can get the solutions in terms of the master element coordinates. $u_{FE}$ at any point x in the element k is equal to sigma i is equal to 1 to P plus 1 which is ndof for the element $u_{ig}$ $N_i$ hat where psi corresponds to the x that we taken. Now what is ig?

ig if we remember is nothing but ieldofs (k, i), for the kth element the ith row. It is very easy to construct the solution in each element. We can do piece by piece and also $du_{FE}$ by dx in the element k is equal to nothing but sigma i is equal to 1 to P plus 1 $u_{ig}$ $dN_i$ hat d psi, whole thing into d psi dx which is 2 by $h_k$. This will give me the derivative of the finite element solution at any point in the element. These are the degrees of freedom which we have already solved for (Refer Slide Time: 26:00). This we will obtain from the shape function routine. This I will get from the solver. This will come from the shape $N_i$ hat, give me any point.

(Refer Slide Time: 26:26)


$\xi_1$
$\xi_2$
$\xi_{N+1} = +1$
$\xi = -1$
$u_{FE}$
1
2
3
4
$u_{FE}(\xi_i) \rightarrow$
$u_{FE}$ is very good at the NODES
$O(h^{2p})$
bubble of order p+1
$u - u_{FE} \simeq a_{p+1} \hat{N}_{p+1}$
$h_k$
$\frac{d}{d\xi}(u - u_{FE}) \simeq a_{p+1} P_p$

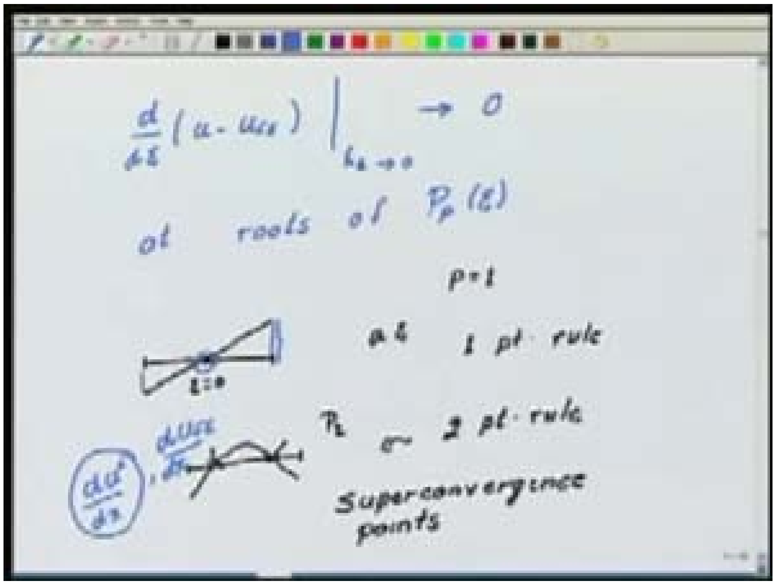
As far as plotting is concerned, take the master element divide it into some number of sub divisions. I say psi$_1$ is equal to minus 1. This is the psi$_1$, psi$_2$, psi$_3$ so on (Refer Slide Time: 26:40). This becomes psi$_{\text{N plus 1}}$ is equal to plus 1. I can find the size. At this size, I will find the solution, by calling the shape function routine, telling it to give me the shape functions in the derivatives and using that to construct u$_{\text{FE}}$ at the point psi$_\text{i}$. These I can output, I can loop element by element that is, I go over the various elements in the domain - elements 1, 2, 3 and 4, for each element put this uniformly distributed set of points, output the value of the displacement and the derivatives at this point. Let us say I get something like this as the output (Refer Slide Time: 27:25). I can connect these by a line to get a plot of what the finite element solution will look over the whole domain. As far as the plotting is concerned, I can output the values of the finite element solution at these points connect them by a curve and I am done.

Similarly, I can do the same thing for the derivative. We see that the u$_{\text{FE}}$ is very good at the nodes. What are the nodes? Nodes here we mean as the extremities of the elements. These are our nodes. It is very good. What do you mean by very good? It is very close to the exact value, if I had an access to that at these points. There is a rate of conversion for these essentially for such problems that we have taken, it goes as order h to the power of 2P that is the error between the finite element solution and the exact solution at this point.

Further, if I have this then I can see that essentially, if I plot the difference between the exact solutions, if I had access to it and the finite element solution in an element, it will essentially look like this (Refer Slide Time: 29:08); this is u minus $u_{FE}$. Imagine that the size of the element tends to 0 that is I am shrinking the size of the element. Then I can show that this is essentially equal to some constant $a_{P \text{ plus } 2}$ $N_{P \text{ plus } 2}$ hat that is it is essentially given by the next higher bubble functions. So I have taken all the bubble functions of order P. Now this one corresponds to the bubble function of order P plus 1.

If we see that somehow from here I am not doing it explicitly, essentially it can be written in terms of what we had given as the Legendre polynomials of order P. If I take the derivative, this is all I am doing in the master element. If I take the derivative of this finite element solution in the master element, the error in the finite element solution, it will essentially be dominated by this term which is the term corresponding to the P th order of Legendre polynomial. If this term has to be 0, what do we mean by this term being 0? This is 0, d d psi of u minus $u_{FE}$ as $h_k$ tends to 0.

(Refer Slide Time: 31:06)



$\frac{d}{d\xi}(u - u_{FE})\Big|_{h_k \to 0} \to 0$

at roots of $P_p(\xi)$

$p = 1$

at 1 pt rule

$P_2$ ~ 2 pt rule

Superconvergence points

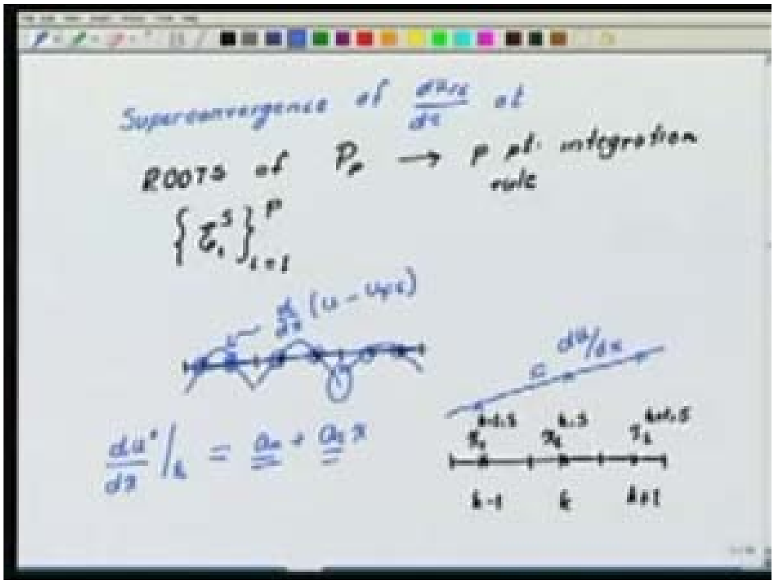
It tends to 0 at the points which are the roots of the Legendre polynomial of degree P. It is as simple as this. Let us take an example. If I have taken linear approximation, this derivative, if I have to find d dx of u minus $u_{FE}$, I simply multiply it by 2 by $h_k$. That is not a big problem.

Wherever, this is 0, the actual derivatives are also going to be 0. If I take the linear approximation that is P is equal to 1, then the derivative in the finite element solution should tend to be exact at the root of the first Legendre polynomial. First Legendre polynomial is some constant into psi. So it should vanish at psi is equal to 0. In fact that is true that the error in the derivative is 0, essentially at the centre of the element.

Similarly, if I go to P equal to 2, I will see that the derivative vanishes at the 2 roots of $P_2$. What are these roots of $P_1$ and $P_2$? These are nothing but the Gauss points corresponding to the 1 point rule, the integration points. This is at the points corresponding to the 2 point rule. I can essentially if I have obtained the solution using a P order approximation then I go to the P order or P point rule and find those points corresponding to the P point rule. At those points, I know that the finite element solution, the derivative of it is going to be very good. These points are called Super Convergence points.

If I have obtained super convergence points, I should, at the nodes that are at the extremities of the element, the derivative obtained from the finite element solution is the worst that I can get. In fact, I should try not to use that information (Refer Slide Time: 34:05). How can I use this information about these good values of the derivatives of the finite element solution, to obtain a better approximation of the derivative by some post processing? What we are actually doing? With the previous transparency also we have done that thing. Here also we had said the plotting. We are actually in the regime of the post processor. Once the solution is obtained, we are talking of how best to interpret our data, to represent our data and how best to extract something better from it. I would like here to get a better value of du star dx from the $du_{FE}$ dx. I know that I can use the values of the finite element solution, the derivative of it, at these points which correspond to the roots of the Legendre polynomial of order P which are nothing but the P point integration rule.

(Refer Slide Time: 35:24)



Superconvergence of $\frac{du_{FE}}{dx}$ at

ROOTS of $P_p \rightarrow p$ pt. integration rule

$\{\xi_i^s\}_{i=1}^{p}$

$\frac{du^*}{dx}\Big|_k = \underline{\underline{a_0}} + \underline{\underline{a_1}}\, x$

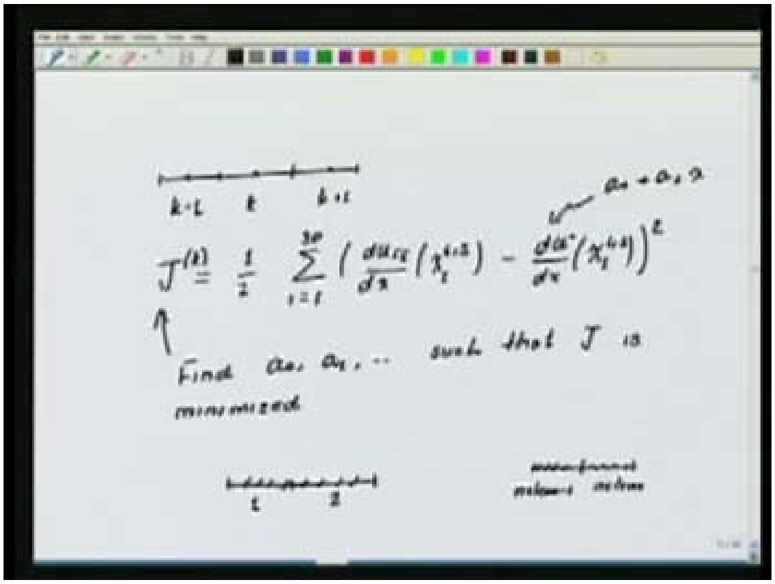
If I have given these points, let me call these points as super convergence points (Refer Slide Time: 35:25), super convergence of derivative of du$_{FE}$ dx at roots of the Legendre polynomial of order P where P is the order of approximation which corresponds to P point integration rule. I already have this information stored with me. Where are these points? Simply take these points and I will them as points psi$_i$ super convergence, i going from 0 to P. Given these points, if I go to the elements by mapping back from the master to the physical element, I will have some points in each of the elements where the derivative of the finite element solution is going to be good. If I plot these derivatives in the elements, we will see they will do like this (Refer Slide Time: 36:52) and there will be a jump in the value here at the interface; derivative will do this.

Now this is something bad, I do not want this, so this is d dx of u minus u$_{FE}$ where u is the exact solution. Here I am talking of the derivative of the error and we want this to be essentially a straight line 0. I do not want the derivative of the error to be big anywhere. Further at the interfaces, I get a jump in the derivative of the finite element solution is the something which I do not want physically, because it leads to a jump in the axial force that we obtained. How best can we now use this information about this point to create better derivative information? The simplest possible way is the following.

Let us say this is the element k (Refer Slide Time: 38:01). This is its neighbor k minus 1 and k plus 1. Let us say, here I am doing the one point rule but we can have the various things. This is corresponding to $x_1$ k minus 1 super convergence here. This is $x_1$ k plus 1 super convergence here (Refer Slide Time: 39:12). These are the mapped physical points corresponding to the super convergence points in these elements. Let us say, I have taken linear approximation for the time being. At these points, I know the derivative is going to be good. Let us say the derivative here is this, derivative here is this, derivative here is this (Refer Slide Time: 39:30). To obtain d u star dx in element k, because I have used linear approximation, I am going to now represent du star dx by a linear in this element. So I am going to represent as $a_0$ plus $a_{1x}$ in this element and I also extended by the same thing in the neighbors that is the k minus 1 element and k plus 1.

I would like to get these coefficients such that I hope that it is going to be a better derivative information in this element then we have out of d $u_{FE}$ dx. I would like to obtain essentially using this information, the straight line fit and I hope that this straight line fit is essentially equal to du dx. How do I construct this straight line fit? I take for the element k, remember that I am doing it for the element k. I have to do it for each element separately. Find these coefficients $a_0$ $a_1$ from there construct the value of the variation of du star dx in the element k.

(Refer Slide Time: 41:08)



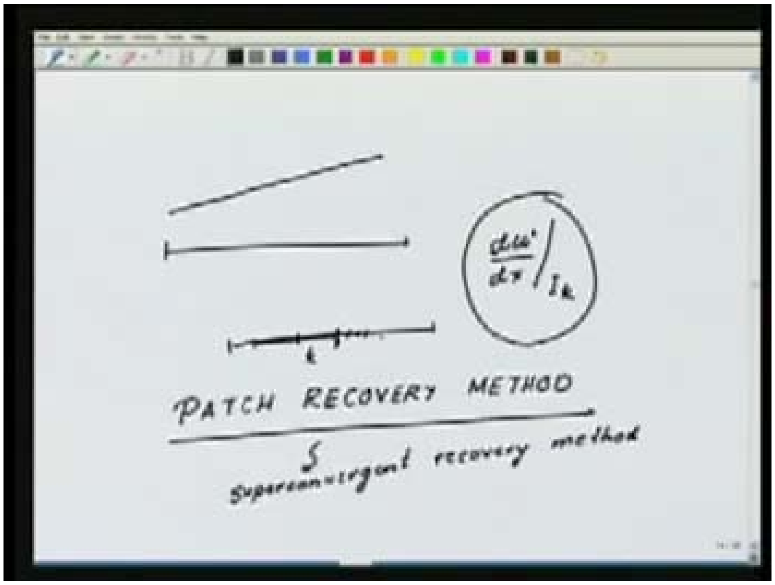
$$J^{(k)} = \frac{1}{2} \sum_{i=1}^{3n} \left( \frac{du_{FE}}{dx}(x_i^{k+1}) - \frac{du^*}{dx}(x_i^{k+1}) \right)^2$$

$a_0 + a_1 x$

Find $a_0, a_1, \ldots$ such that J is minimized

To do that again I take this neighborhood; it is k, k minus 1, k plus 1. I am going to define this functional as 1/2 sum over all the i(s), i is equal to 1 to 3 into (3 actually should be 3P of) $du_{FE}$ dx evaluated at the point x for P is equal to 1, i s minus du star dx evaluated at the point $x_1$ i s whole squared. I am taking this term is given by $a_0$ plus $a_{1x}$. I am taking this value of x from super convergence point for the neighbors, putting it here I am doing discrete sum. This is the functional J and I am looking for the coefficients $a_0$ and $a_1$ which minimize. So find $a_{0,}$ $a_1$ such that J is minimized (Refer Slide Time: 42:55). J, I would say is defined for the element k. This definition, I have to do for any generic element k. I have to go from the first element to the second one, to third one. I will find this coefficients $a_{0,}$ $a_1$ corresponding to the element. From there I can construct du star divided by dx at the element level. What happens at the various places? If I have an element at the boundary, that is the first element. In this case, I do not have the 0th element; I only take this neighborhood. For the last element, again I will take the n minus 1th element and the nth element. So this is nelem and this is nelem minus one (Refer Slide Time: 43:50). This is how I am going to construct these so called recovered coefficients.

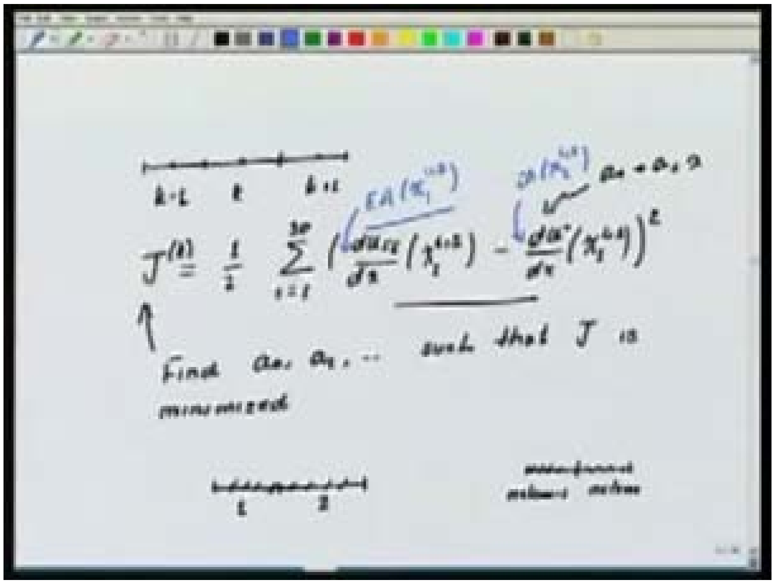
(Refer Slide Time: 44:01)



When I piece them together, I will see that essentially the du star dx it will look like a piecewise linear. If I am at a material interface, if this is the element (Refer Slide Time: 44:18) which has this material interface, I am sitting at the material interface then I do not take the neighbor. If this is the element k, I do not take the k plus 1 neighbor because it is lying in a different material. I will only take the neighbors from the same material neighborhood. This is how we can construct the recovered du star dx in each element k. This is called the Patch Recovery method and this indeed gives us the values of the derivatives which are very good.

If we do this kind of a post-processing, we can construct very good recovered derivatives without doing any extra computation. The computation here is minimal and it is done at the element level. I will do for each element separately. Out of what I did here, I will get the system of equations in terms of $a_0$, $a_1$ up to $a_p$. I will solve that system of equation for the element. I have to solve it for all the elements separately one by one and I have constructed the du star dx information for all the elements. This one (Refer Slide Time: 45:50) said to be a Super Convergence Recovery method.

(Refer Slide Time: 46:09)

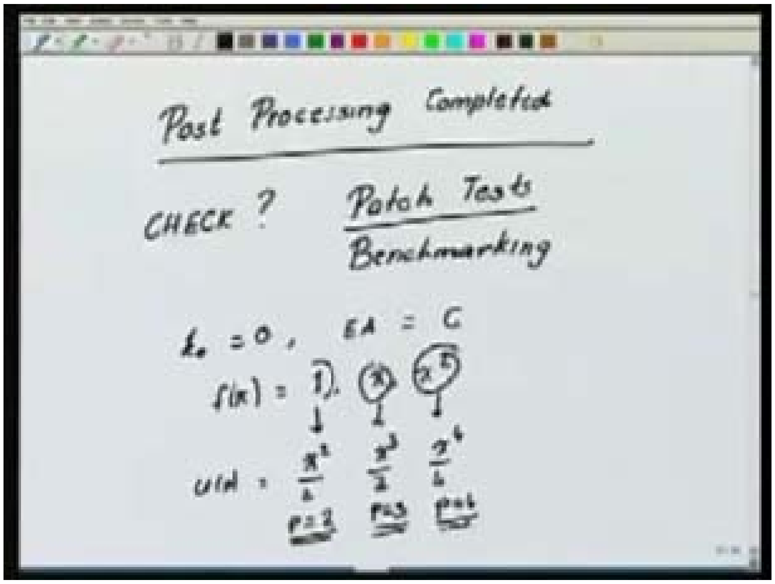


$$J^{(k)} = \frac{1}{2}\sum_{i=1}^{3n}\left(EA(x_i^{k,s})\frac{du_k}{dx}(x_i^{k,s}) - EA(x_i^{k,s})\frac{du^*}{dx}(x_i^{k,s})\right)^2$$

Find $a_0, a_1, \ldots$ such that J is minimized

There is another version of it where I can actually modify J to have EA at the point $x_1$, i comma s. Here I have taken 1 but in our sum we should have all these integration points in the three elements which form the neighborhood of the super convergence points. Here also (Refer Slide Time: 46:40) I will have EA at $x_1$, i comma s. If I add this, this is another way, an alternative definition for getting these super convergent recoveries. Especially when the EA is non uniform, but not transitional one it is simply if it is a tapered beam for example, then this will do the job.

(Refer Slide Time: 47:13)



Post Processing Completed

CHECK ? Patch Tests

Benchmarking

$k_0 = 0, \quad EA = C$

$f(x) = \boxed{1}, \ \boxed{x}, \ \boxed{x^2}$

$u(x) = \frac{x^2}{2}, \ \frac{x^3}{3}, \ \frac{x^4}{4}$

$\underline{P=2} \quad \underline{P=3} \quad \underline{P=4}$

We have done these recoveries – certain things; once we have done this, our post-processing has also been completed. After we have done all this, we have written a program, how do we check our program? We check it using so-called Patch Tests. Patch tests or benchmark tests, the basic idea in all these are that we take problems for which we can get the exact solution. Then we go and try to solve this problem and we see whether our finite element solution is close to the exact solution. It should not be too far away. If it is too far away, there is some problem.

In fact, in the patch tests, we can construct at least for the case $k_0$ equal to 0, EA is equal to constant. If we take our f(x) is equal to 1 or x or x square, we know our exact solution in this case will be of the type x squared by 2 is a polynomial (Refer Slide Time: 48:35). Here it will be x cube by 3 and here it will be x to the power of 4 by 4 plus some constants. In these cases, when $k_0$ equal to 0, is EA is equal to C, taking either this as the load or this as the load or this as the load (Refer Slide Time: 48:52), correspondingly, I will get the quadratic, the cubic and the fourth order solution. We can simply pay for this solution.

Let us say I have taken f(x) is equal to 1, I will take P is equal to 2, the quadratic approximation and with this quadratic approximation I should get the exact solution to the problem back. Similarly, if I take P is equal to 3 and take f(x) is equal to x, I should get $u_{FE}$ is equal to u exact and for P is equal to 4, I do this (Refer Slide Time: 49:26). If somehow the code does not give the finite element solution equal to the exact solution then there has to be a bug in whatever we have done. We have to go back and debug our program and set it right so that till at least the machine precision that is specified. Generally, we do double precision calculations. The numbers will match with the exact solution.

So with this, we have come to the end of the one dimensional programming assignment. With all the tools given, we can develop the one dimension code. I tried to give a feel of what will be coming or needed in the 2 and 3 dimensions. Essentially, the data structure that we have constructed here, the kind of procedure we followed here, will carry over almost unaltered into the 2 and 3 dimensional cases. It is for you to write the program, test it for this patch test or for benchmark test for which you construct the exact solutions and then use it for other problems for which you would like to get a solution for which getting a exact solution is difficult.

Thank you.