

Computational Fluid Dynamics
Prof. Dr. Suman Chakraborty
Department of Mechanical Engineering
Indian Institute of Technology, Kharagpur

Lecture No. # 22

Solution of System of Linear Algebraic Equations: Elimination Methods

(Refer Slide Time: 00:32)

$x_1 + x_2 + 10x_3 = 12$ (E1)
 $10x_1 + x_2 + x_3 = 12$ (E2)
 $10x_1 + x_2 + x_3 = 12$ (E3)

FORWARD ELIMINATION

Step 1: Row 1 as the pivotal row

$(E2) \rightarrow (E2) - \frac{1}{10}(E1) \Rightarrow (10-0.1)x_2 + (1-0.1)x_3 = 10.8$
 $\Rightarrow 9.9x_2 + 0.9x_3 = 10.8$

$(E3) \rightarrow (E3) - \frac{1}{10}(E1) \Rightarrow (1-0.1)x_2 + (1-0.1)x_3 = 10$
 $\Rightarrow 0.9x_2 + 9.9x_3 = 10$

Step 2:

$(E3) \rightarrow (E3) - \frac{0.9}{9.9}(E2) \Rightarrow (9.9 - \frac{0.9 \cdot 9.9}{9.9})x_3 = 9.818$

Up to this \rightarrow forward elimination

BACKWARD SUBSTITUTION

In the previous lecture, we started discussions on the elimination methods. Let us continue with that. So, let us take up an example. So, we have 3 equations with 3 unknowns and you want to solve for x_1 x_2 x_3 by an elimination technique which we will formally describe as the Gaussian elimination method.

To see how the method works, first we will go through this example. Then, try to formulize this through a more general algorithm. So, the first and foremost objective is to operate on the co-efficients in the co-efficient matrix. So, if you write the coefficient matrix, you have 10 1 1 1 10 1 1 1 10 and if you write the augmented matrix, you have additional twos.

Now, as we discussed in the previous lecture, our objective will be to convert this co-efficient matrix into an upper triangular matrix. So, we want to have non 0 terms only

within this portion, but not below that. So, all the terms below that should be eliminated and converted to 0.

So, first we will convert the terms in the first column below the diagonal to 0. To do that, we will take the first row as the key row using which, we will eliminate these 1s to make them 0s and that key row, we call as the pivotal row. So, in step 1, you have row 1 as the pivotal row. So, you want to eliminate this x_1 from equation 2 using equation 1. Equation 1 confirms to row 1. So, what we will do? We will have equation 2. New equation 2 is nothing, but old equation 2 minus 1 by 10 into equation 1.

So, what will that imply? Your x_1 will be eliminated because this is 1 minus 10 into 1 by 10 that is 1 minus 1. That will be 0. Next will be 10 minus 1 by 10. That is $0.1x_2 + 3$ is 1 minus point $1x_3$ is equal to 12 minus 1.2. So, you have $9.9x_2 + 0.9x_3$ is equal to 10.8. Similarly, equation 3 is, equation 3 minus again 1 by 10 equation 1. So, the first column is eliminated. So, then we have 1 minus $0.1x_2 + 10$ minus $0.1x_3$ is equal to 10.8. That means, $0.9x_2 + 9.9x_3$ is equal to 10.8. So, what is the corresponding matrix form? This is the original matrix form. Then, after step 1, first row remains unchanged. Second row, now this has become 0 co-efficient of x_2 is 9.9. Co-efficient of x_3 is 0.9 right hand side, this 10.8.

Equation 3 co-efficient of x_1 is 0, coefficient of x_2 is 0.9, co-efficient of x_3 is 9.9 and right hand side is 10.8. This is after step 1. Then step 2. Now, what we will do? We will consider the second row as the pivotal row and try to eliminate the term which is in the column confirming to the diagonal element of the pivotal row. So, as we eliminated the first column, now we will eliminate the second column below the diagonal.

So, to do that, now remember equation 2 and equation 3 have changed. So, this is new equation 2. This is new equation 3. So, these we will call as, refer to as equation 2 and equation 3. No more the original equation 2 and equation 3. So, step 2, what will be equation 3? Equation 3 will be equation 3 minus. So, this has to be made 0.9. So, how do you make it 0.9? Just use these co-efficients and tell. So, 0.9 by 9.9 into 9.9 . So, that makes, that converts this 9.9 to 0.9 . So, this you want to apply. So, remember. So, what you are doing when we generalize? We keep this in mind the co-efficient divided by the pivotal co-efficient that is the diagonal co-efficient of the pivotal row multiplied by the corresponding term in the pivotal row. That is what we are doing.

So, this into in general, it is equation 2. So, 9.9 was for the first term, 0.9 will be for the second term. So, for the first term, we wanted to make it 0. So, that is why 9.9 for the second term, it will not be 9.9, but 0.9. So, second term, therefore will be non 0. So, this will be what the first term will be 0 because this is as good as 0.9 by 9.9 into 0.9 into 9.9. That we have seen. That is the first term. Second term will be what? 9.9 minus 0.9 by 9.9 into 0.9 that x_3 is equal to 10.8 minus 0.9 by 9.9 into 10.8.

So, if you work it out, it may be 9.818 x_3 is equal to 9.818. So, x_3 is equal to 1, but before that, let us formalize what happens after step 2. So, we have the first row 10 1 1 12, second row 0.9, 9.9, 10.8, third Row has now changed. So, it has become 0 9.818. This one also 9.818.

So, the entire system has been reduced to an upper triangular system. It require 2 steps for 3 equations. So, in general, if there are n equations, it will require n minus 1 number of steps to reduce it to an upper triangular form. So, this upto this, we call as forward elimination. Next will be backward substitution. So, these steps, 1 and 2 we can say that these belongs to forward elimination and next will be backward substitution.

(Refer Slide Time: 00:32)

The whiteboard shows the following steps:

$$E_2 \rightarrow E_2 - \frac{1}{10} E_1 \Rightarrow (10 - 0.1)x_2 + (1 - 0.1)x_3 = 12 - 1.2$$

$$\Rightarrow 9.9x_2 + 0.9x_3 = 10.8 \quad \text{New E2}$$

$$E_3 \rightarrow E_3 - \frac{1}{10} E_1 \Rightarrow (1 - 0.1)x_2 + (0 - 0.1)x_3 = 10.8$$

$$\Rightarrow 0.9x_2 + 9.9x_3 = 10.8 \quad \text{New E3}$$

Step 2: $E_3 \rightarrow E_3 - \frac{0.9}{9.9} E_2 \Rightarrow (9.9 - \frac{0.9}{9.9} \times 0.9)x_3 = 10.8 - \frac{0.9}{9.9} \times 10.8$

$$9.818x_3 = 9.818 \quad \text{New E3}$$

upto this \rightarrow forward elimination

BACKWARD SUBSTITUTION

$$\text{New E3} \rightarrow x_3 = \frac{9.818}{9.818} = 1$$

$$\text{New E2} \rightarrow 9.9x_2 = 10.8 - 0.9 \times 1 = 9.9 \Rightarrow x_2 = 1$$

$$\text{New E1} \rightarrow 10x_1 = 12 - x_2 - x_3 \Rightarrow x_1 = 1$$

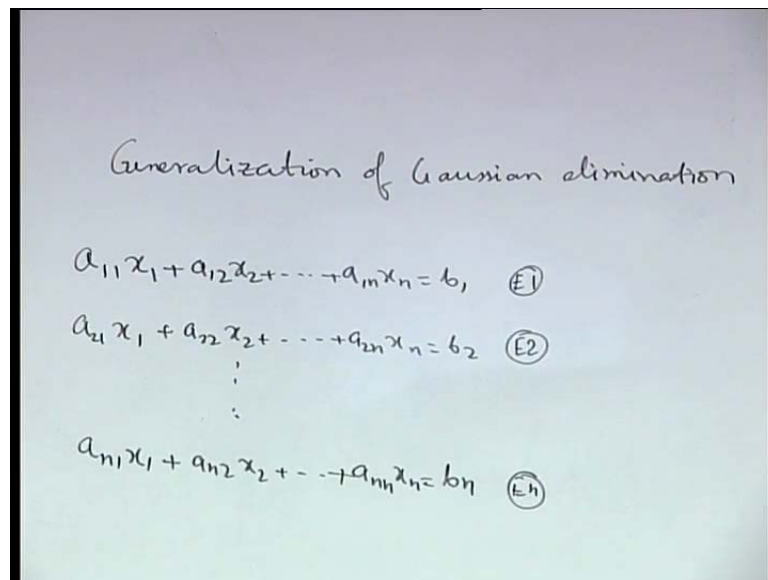
So, for backward substitution, we have x_3 is equal to 9.818 by 9.818 that is 1. This is from new E3. Then, from new E2, you have 9.9 x_2 is equal to 10.8 minus 0.9 x_3 . So, that is equal to 9.9 because x_3 is 1. That means, x_2 equal to 1 and from new E1 which is

same as the old, new one of course, you have $10x_1$ is equal to 12 minus x_2 minus x_3 . That means, x_1 is 1 because both x_2 and x_3 are 1.

So, this is an example just to illustrate how the Gaussian elimination method works. We will later see whether, it will work for all cases or not. See, whenever we are learning a method, what we are essentially following is like this. First, we are trying to get a feel of the method, then we are going to see whether the method will work or not and of course, if it works, how efficiently it works. So, these are the three questions that we would always like to answer. What is the method? Will it work for all cases? If it works, how efficient it is? These are the three standard questions that we will like to implicitly answer as we progress through the different methods that we learn.

Now, before answering the later 2 questions. The first question remains to be answered that is what is the method? We have seen what is the method through an example, but this is not general. We have to generalize this example for, say n number of equations with n number of unknowns. So, let us try to do that.

(Refer Slide Time: 16:31)



Generalization of Gaussian elimination

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + \dots + a_{1n}x_n &= b_1 & \text{(E1)} \\ a_{21}x_1 + a_{22}x_2 + \dots + a_{2n}x_n &= b_2 & \text{(E2)} \\ & \vdots & \\ a_{n1}x_1 + a_{n2}x_2 + \dots + a_{nn}x_n &= b_n & \text{(En)} \end{aligned}$$

So, generalization of Gaussian elimination. To generalize the Gaussian elimination, let us consider a system of equation like this, $a_{11}x_1$ plus $a_{12}x_2$, in this way plus $a_{1n}x_n$ is equal to b_1 , $a_{21}x_1$ plus $a_{22}x_2$ plus $a_{2n}x_n$ is equal to b_2 . In this way, $a_{n1}x_1$ plus $a_{n2}x_2$ plus $a_{nn}x_n$ is equal to b_n . Give these equations some number. This is E1, this is E2,

in this way this is E_n . So, what we first make? We first make the first row as the pivotal row.

(Refer Slide Time: 17:59)

$a_{n1}x_1 + a_{n2}x_2 + \dots + a_{nn}x_n$

For step! Row 1 \rightarrow Pivotal

$$E_2 \rightarrow E_2 - \frac{a_{21}}{a_{11}} E_1$$

$$E_3 \rightarrow E_3 - \frac{a_{31}}{a_{11}} E_1$$

$$\vdots$$

$$E_i \rightarrow E_i - \frac{a_{i1}}{a_{11}} E_1$$

$$a_{ij} \rightarrow a_{ij} - \frac{a_{i1}}{a_{11}} a_{1j}$$

So, row 1 pivotal row. Then, using row 1 as the pivotal row, we will eliminate all the terms which are there in the column of the diagonal element corresponding to row 1 below the diagonal. The effort is to slowly go through steps by virtue of which, it will have all 0 below the diagonal. So, first 0 in the first column below the diagonal. Next step will give you 0 below the second column of the diagonal. Third step will give you 0 below the third column of the diagonal in this way.

So, 0s before the first column of the diagonal, how you can do that? You have equation 2, will be equation 2 minus, see the first row. So, how it can be eliminated? By a_{21} by a_{11} into a_{11} will make it a_{21} . So, that minus a_{21} will be 0. So, E_2 minus this into E_1 . So, you can see that this manipulation is d_1 with an objective that the corresponding terms become 0. So, you can figure it out by putting a_{21} in place of this 1. So, that it is a 21 by a_{11} into a_{11} will make this also a_{21} . The difference will be 0.

So, what will be E_3 ? E_3 minus in place of 2, it will be 3. So, a_{31} by a_{11} into E_1 . So, in this way, what is, say E_j or E_i whatever name you give. E_i will be E_i minus a_{i1} by a_{11} into E_1 . So, how do you generalize it in place of E_i ? What you are basically doing when you are considering equation number i ? There you are playing with the i -th row and j -th column where j varies from 1 to n plus 1.

You also have an extra column to accommodate the right hand side. Remember, that you are writing the corresponding equation form of the augmented matrix or augmented matrix including the right hand side of the equation form. So, in place of E_i , if you right it as a_i , then subscript j . So, a_i is the first subscript for the row i -th equation and the j -th column. That will be a_{ij} minus this is a_{i1} . So, we keep it a_{i1} divided by a_{11} into what is E_1 , is a_{1j} .

If E_i is a_{ij} , then E_1 is a_{1j} . The first subscript is for the row for the i -th row. If the first subscript is i for the first row, the first equation, the first subscript is 1. So, this you are doing for step 1.

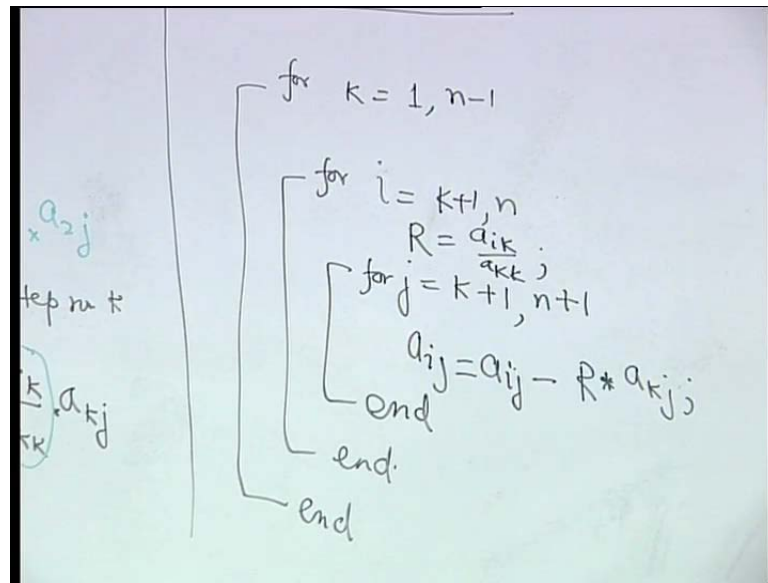
(Refer Slide Time: 22:46)

The image shows handwritten notes on a whiteboard. At the top, it says $a_{n2}x_2 + \dots + a_{nn}x_n = b_n$ with a circled (E_n) next to it. Below this, it says "For step 2". To the left, there are three rows of operations: $\frac{a_{i1}}{a_{11}} \times (E_1)$, $\frac{a_{j1}}{a_{11}} \times (E_1)$, and $\frac{a_{i1}}{a_{11}} \times (E_1)$. In the center, the formula $a_{ij} \rightarrow a_{ij} - \frac{a_{i2}}{a_{22}} \times a_{2j}$ is written. Below that, it says "Generalize for step number k". At the bottom, the generalized formula $a_{ij} \rightarrow a_{ij} - \frac{a_{ik}}{a_{kk}} \times a_{kj}$ is written.

Then, for step 2, what will be this formula for a_{ij} ? Now, let us see, it will be a_{ij} minus, in place of a_{i1} , it will be a_{i2} because now, second row will be the pivotal row. So, this one indicates the corresponding number of the pivotal Row. So, this will become a_{i2} divided by a_{22} into a_{2j} . So, in general, for step number k , so generalize for step number k , what does this k indicate? It indicates what is the pivot.

So, k equal to 1 means that the first row is the pivotal row, k equal to 2 means second row is the pivotal row and k equal to 3 is the third row pivotal row and so on So, a_{ij} will become a_{ij} minus a_{ik} by a_{kk} into a_{kj} . So, this is essentially the formula that we are applying for the forward elimination method and because there are 3 indices ij and k , we can put this entire algorithm in 3 for loops.

(Refer Slide Time: 24:46)



So, let us try to put that. So, first what we are trying to do is, we are trying to now formalize the forward elimination. So, let us write for loops for k equal to what? What will be the range of k ? The first value of k is 1. What is the last value of k ? n minus 1 is the last pivot because using that the term, immediately below that is eliminated, if you consider n as a hypothetical pivot, there is nothing below that.

So, n is not a pivot. Then, for what is j ? What columns you are eliminating? Why you are eliminating the columns? So, let us look into the first case. When this is the diagonal, you are eliminating the columns below. Of course, the rows below, but for the columns from the right of the diagonal. So, if it is a_{11} , the next that you are looking for, so if j is equal to 1, you are looking for j equal to 2 onwards for elimination.

So, you are not doing it for 1. Why you are actually doing, but not putting in the algorithm because you know, it will come out to be 0. So, why do you waste a calculation in coming up with this equal to 0 because already you have designed the calculation. So, that this should come to be 0. Once you have designed this calculation, that this should come to 0, you do not repeat it once more implementation. So, for implementation, you only calculate the non 0 elements. The 0s are already known that they will be 0s by virtue of this algorithm that you have implemented.

So, if this is kj becomes k plus 1 to n plus 1 and what is ik plus 1 to n plus 1 because what this is the k -th row. Example, k is equal to 1. So, you are manipulating with the

rows below the starting, from just below the k -th row. So, $k + 1$ to, not $n + 1$, there is no $n + 1$ row. Now, let us see we have not yet formalized. See, I am trying to go through the mental exercise instead of writing the algorithm as it is. Let us try to go through some mental exercises to see that what should be the best way to put it.

So, I have just put the structure j is from $k + 1$ to $n + 1$ because that you have the right hand side also included in the column, but i is upto n . Now, next, is it legitimate to put for loop on j first or for loop on i first? Which one we should do? Let us try to see. See what calculation we will be doing, it depends on that. So, these are certain things try to carefully learn. Our objective is not to just write an algorithm for Gaussian elimination in any book. You can see the algorithm and you can implement.

The objective is to have a mental exercise to learn yourself how to design a new algorithm if you know the method. That what needs to be implemented. So, how to translate a method into a systematic algorithm? That is what we are trying to learn through this example. So, now let us see, you have this a_{ik} by a_{kk} . So, what you have to do is for each, so you have i, j . So, immediately next step you are prompted to write a_{ij} minus $a_{ik} a_{kj}$ into a_{ij} .

So, if you have a_{ik} by a_{kk} , you can put it inside this loop, but you can also put it outside this loop by making this as i and making this as j . So, let us make a conversion of that. So, let us make this j . Make the first 1 as i because of course, $n + 1$ and n , those things will change because once you enter the loop, you require a_{ik} by a_{kk} . So, if that is already ready before entering the loop, you can just use that value.

So, let us say that you have R equal to a_{ik} / a_{kk} because k is already known and i is already specified. So, this you can specify in terms of 2 indices. So, once you know this R , you do not have to calculate it again and again and again once you enter the j loop for each j . Now, you can calculate a_{ij} is equal to a_{ij} minus R into a_{kj} . If you had put this R inside this j , what would have happened. That it would have been useless. I mean, it would have not been a wrong thing, but what would have been there is every time for a new j , this will be calculated unnecessarily. This requires only the index i and k for its calculation.

So, when j goes from $k + 1$ to $n + 1$, if this R was inside, it would have $n - k + 1$ upto that many number of times extra calculation of R unnecessarily. So, this is an important thing

that you have to be careful about that what calculation you put inside a loop, what calculation you put outside the loop, you should not put a calculation inside a loop where it is not necessary to go through that many number of times inside the loop. So, you put it outside the loop.

Then, you end each for loop with an n. Remember, this is not any program language that we are writing. So, it is just like a pseudo code. You can convert it into a corresponding programming language as per your choice. So, this is the forward elimination part. So, at the end of the forward elimination, we expect that what will be the form of the system of equations and then, of the forward elimination, you have an nxn equal to Bn because all other a's in this row have been made to 0. So, next will be backward substitution. So, again what we will do? We will see how to formalize the backward substitution and then, we will write its corresponding algorithm.

(Refer Slide Time: 33:22)

The image shows handwritten mathematical equations for backward substitution. The equations are:

$$x_n = \frac{b_n}{a_{nn}}$$

$$x_{n-1} = \frac{b_{n-1} - a_{n-1,n} x_n}{a_{n-1,n-1}}$$

$$x_{n-2} = \frac{b_{n-2} - [a_{n-2,n} x_n + a_{n-2,n-1} x_{n-1}]}{a_{n-2,n-2}}$$

$$\vdots$$

$$x_{n-i} = \frac{b_{n-i} - \sum_{j=0}^{i-1} a_{n-i,n-j} x_{n-j}}{a_{n-i,n-i}}$$

So, backward substitution, the first one is the most straight forward. So, x_n is equal to B_n by a_{nn} , then what is the equation? Immediately before that $a_{n-1,n} x_n$ is equal to B_{n-1} plus $a_{n-1,n} x_n$ is equal to B_{n-1} . Remember, anything to the left of the diagonal has been eliminated to 0. So, only starting from the diagonal, it is there. So, you have x_{n-1} is equal to what B_{n-1} minus $a_{n-1,n} x_n$ by $a_{n-1,n-1}$. Let us write the third x_{n-2} . So, when you are interested to generalize an algorithm, it is always advisable that you write 2, 3 such steps which will give you the

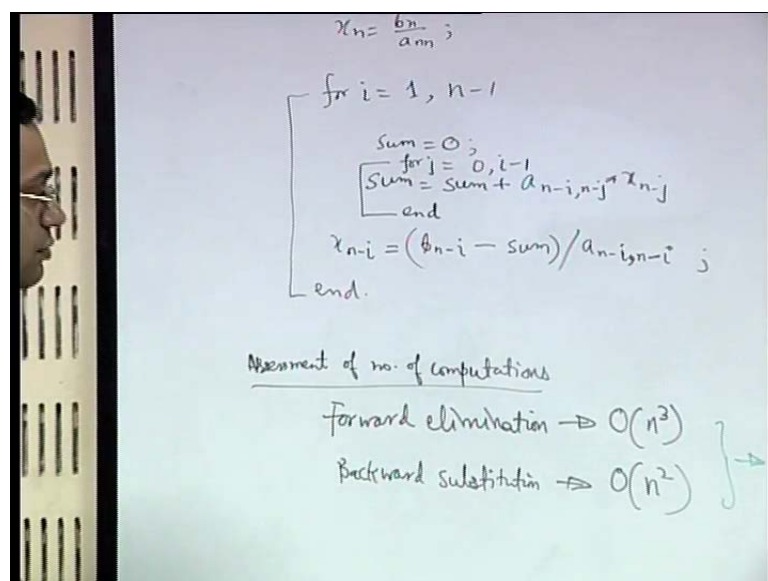
idea of the generality. The general parity between each steps that will help you to write it in terms of general indices ij etcetera, instead of 1, 2, 3 like that.

So, the equation which is before this $1a_n$ minus $2x_n$ minus $2x_n$ minus 2 plus a_n minus $2x_n$ minus $1x_n$ minus 1 plus a_n minus $2x_n$ x_n is equal to B_n minus 2. So, what is x_n minus 2? That is B_n minus 2 minus a_n minus $2x_n$ minus $1a_n$ minus $2x_n$ that added with a_n minus $2x_n$ minus $1x_n$ minus 1 divided by a_n minus $2x_n$ minus 2.

In this way, if you want to write x_n minus i minus $1x_n$ minus 2, then what is that B_n minus i minus. There is a summation of a_n minus ijx_j . What is the range of j ? So, it is better to write x_n minus j . So, a_n minus $1j$, you can write this also as n minus j because you have put the index with x as n minus j . You can clearly see that whatever is the index with x , the corresponding index is with the second index of a . So, here, it is n minus 1. This is also n minus. So, here this is n minus j , this is also n minus j .

So, j will run from first j is 0 because then, this is n . Then, j is 1. In this way, where will it stop? So, i minus 1 because this is 2. It stops with 1. This is i . It will stop with i minus 1. This divided by a_n minus i minus i . So, let us try to write a formal algorithm for backward substitution as we have $d1$ for the forward elimination. See, once you get a formula in terms of indices and summation, it is very easy to put that formula in the form of a loop.

(Refer Slide Time: 39:53)



So, backward substitution algorithm. So, first is x_n is equal to B_n by a_{nn} . This is straight forward. This is not within any loop. Then, we start with x_{n-1} calculation. So, I will be starting from 1. So, for i is equal to 1. To what is the last value of x ? That you calculate x_1 . So, that will become 1 when i is $n-1$ because $n - n - 1$ is 1.

So, for i is equal to 1 to $n-1$, then to implement this formula, you require this summation calculation. So, you can sum equal to 0 and then, you add on with the variable sum to calculate this 1. So, sum equal to, we will put the j index after writing the formula. Some is equal to sum plus a_{ij} minus x_j into x_{i-1} where what is j for j is equal to 0 to $i-1$. Then, that calculation ends. So, you get a new value of the sum which is this sigma. So, at the end of the calculation of the loop, you have the variables sum which is giving this summation. Then, you implement the formula. So, x_{i-1} is equal to B_{i-1} minus $\sum_{j=0}^{i-1} a_{ij} x_j$ divided by $a_{i,i-1}$ and that ends for loop over i .

Now, once we have generalized this algorithm, the next objective will be to assess the algorithm. That is, what will be the number of steps necessary? What is the limiting step that is out of the forward elimination and backward substitution? Which one is computationally more expensive? Possibly, when something is computationally expensive, how to reduce that number of computations? So, these are certain things that we will now try to assess.

So, the assessment of number of computations. So, it is very easy to assess the number of computations in terms of the order of computations. So, when we say in terms of order of computations. If we say something of the order of n , for that what we mean is, that it is not exactly equal to n , but it scales with n . So, it can be cn .

In that way, we will try to assess the order of the number of calculations. So, it is very easy to do that. If you look into the algorithm, the algorithm has 3 nested for loops. Each of the loops runs upto the order of n . This is 1 to $n-1$. Do not try to count exactly how many number of times, but it runs upto the order of n because the parameter here itself is n . Similarly, for the loop of i that also runs upto the order of n . Also, runs up to the order of n .

So, within this, if there is some calculation, that calculation is repeated of the order of n times. That is again repeated. For each of these calculations, the loop on i is repeated

upto order of n times. So, taking this together, these equations are $d1$ upto a order of n square times and when you have upto a order of n square times calculation, there is an outer loop that itself runs up to order of n . So, n number of times to say in order.

Therefore, the total number of calculations will run upto the order of n cube. So, just like as if n for this, for loop n , for loop there nested. So, n into n into n . Remember always, that this is not the exact number of calculation. This is just the order of calculations. Number of calculations will require more detailing. So, when we say that this runs upto order of n for each, then you have 1. This R multiplication with this 1, this is 1 calculation, then addition of that negative of that with a_{ij} .

So, you can break each into some additional parts 2 2 calculations or 3 calculations, but that will not change the order because that does not parameterize with n . So, you have to be careful that out of the calculation, there are certain calculations which are fixed in number and there are certain calculations which parameterize with n . So, whatever is parameterizing with n , that we are including in this order related to n .

So, the forward elimination will have of the order of n cube and backward substitution here, you have 2 such loops. If you see here, you have 1 loop within which this summation is calculated. This I will be scaling with n and then, you have the outer i also scaling with n . So, this is of the order of n square.

So, what is the rate determining step forward elimination because n order of n cube is more than order of n square. So, if you say, what is the computational complexity of the algorithm, that is of the order of n cube because of the order of n square will be much much less than of the order of n cube. You have to remember that, when you are assessing an algorithm, you are considering the value of n beyond the threshold numbers. So, that this comparison become you are not doing it for n equal to 1.

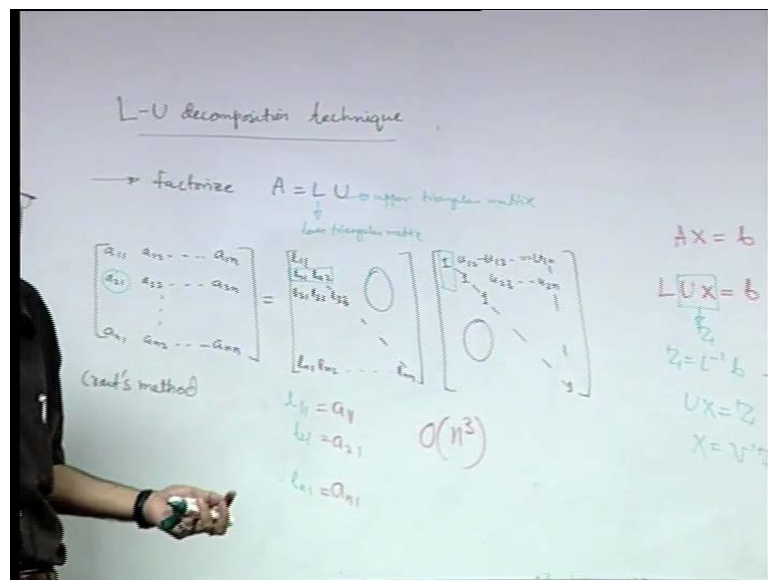
So, you are doing it beyond the threshold where n cube is definitely over weighing the effect of n square significantly. So, that the net effect is of the order of n cube. So, we say that, for Gaussian elimination, the number of calculations is of the order of n cube where n is the number of variables and the number of equations that you have.

Now, when we do this assessment, we learn a moral out of it. What is the moral? The moral is that if you could have the step only as the backward substitution like step, then

you could have reduced the order of calculations from n^3 to n^2 . How that was possible? In the backward substitution, you dealt with only triangular matrices or in this example, only upper triangular matrix, but if you are dealing with a lower triangular matrix also, it becomes the same thing.

Only your indices $n - i$ will be replaced with i , but it is the triangular nature of the matrix that is being handled. That reduces your number of calculations here to n^2 of the order of n^2 . So, this motivates us to look into possible algorithms where the number of calculations themselves can be of the order of n^2 and with that motivation, a variant of the Gaussian elimination is designed which is known as LU decomposition method.

(Refer Slide Time: 50:00)



So, what is d_i ? In this method is a new step that is implemented that is you factorize A as a product of 2 matrices. One is the lower triangular matrix, L for lower triangular matrix and U for upper triangular matrix. So, a_{11} a_{12} . In this way, a_{1n} a_{21} a_{22} . In this way, a_{2n} a_{n1} a_{n2} . In this way, a_{nn} is equal to lower triangular matrix L_{11} . Then, L_{21} L_{22} L_{31} L_{32} L_{33} . In this way, L_{n1} L_{n2} . In this way L_{nn} . These are all 0. This times upper triangular matrix to match the number of unknowns with the number of equations. That you get by equating these 2. You can normalize the diagonal elements of this to 1, instead of writing U_{11} U_{22} U_{33} . It is possible to normalize those to 1, so that you can determine the others deterministically.

So, this is U_{11} , then U_{12} U_{13} . In this way, U_{1n} , then this is U_{22} , then U_{23} , then in this way, U_{2n} . In this way, the last one is 1 and this is 0 here. So, this factorization, what it does is, you have to find out what are these L_{ij} 's and U_{ij} 's. Given a_{ij} 's, the corresponding algorithm is known as Crout's method. I will not go into the details of this method because it is quite straight forward.

Let us just look into one step of the method. I will advise you to go through the remaining steps of this method by looking into any text book or numerical analysis. Now, the first step is like this. You have a_{11} that is equated to what the first row of this with the first column. So, a_{11} is L_{11} . So, L_{11} into 1, the remaining is 0.

What is a_{21} ? Second row multiplied with the second column, second row with the first column. So, L_{21} into 1. So, this row multiplied by this column will give you a_{21} . So, in this way, a_{n1} is equal to L_{n1} . Actually, you want L_{11} L_{21} L_{n1} like this, not the other way. So, it is better to write L_{11} equal to a_{11} because unknown is L_{11} L_{21} is a_{21} , L_{n1} is a_{n1} . So, in this way, you can determine the unknown co-efficients 1 by 1.

So, once you have found it out, then you can write your system of equation ax equal to b in the following form LUX equal to b . So, once you have factorized a into a as L into U , next Step you can write U into xU is a matrix into x as some z . So, you can write LZ equal to b . So, now, you can see this is a triangular matrix. So, Z is L inverse b where you can use the same algorithm as that of the backward substitution. It is just not backward substitution. It is like forward substitution because it is a lower triangular matrix instead of the upper triangular matrix, but it is a triangular matrix.

So, its calculation is of the order of n square. Then, once you know this Z , then LZ equal to b , sorry, so UX equal to z . So, x equal to U inverse Z . This is like backward substitution because this is U is the triangular matrix. So, this is of the order of n square. So, it gives you a feeling that this algorithm is of the order of n square and it is better than the Gaussian elimination, but that is illusive. Why that is illusive because by assessing the algorithm, in this way you have not taken care of the fact that you require some calculations also for this factorization. Calculation for this factorization comes out to be of the order of n cube.

So, it actually does not over weigh this one. When it does not over weigh the number of calculations in terms of order than that of the Gaussian elimination, then what is the

utility of this method. So, if you have a method where you have the solution part of the order of n square, but factorization part of the order of n cube, then the resultant algorithm is of the order of n cube. That becomes same as the Gaussian elimination.

So, why at all this method is used when you already have Gaussian elimination? What advantage does it give to us? So, let us stop here today. We will try to answer these questions in the next class. Thank you.