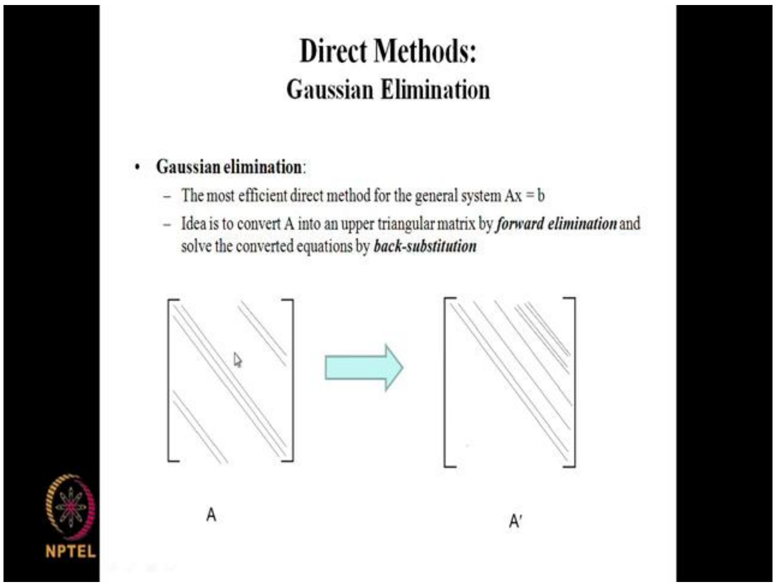
Computational Fluid Dynamics

Prof. Sreenivas Jayanti
Department of Computer Science and Engineering
Indian Institution of Technology, Madras

Lecture – 45
Direct Methods: solution of the system of algebraic equations

In the earlier lecture we have seen the need for speed in terms of solving the matrix equation Ax equal to b. We have seen that the Cramer's method, Cramer's rule that we are been thought in our school days is really not something that can be scaled to large number of equations, because of the number of the arithmetic operations it takes to get solution of the Ax equal to b type of system of linear algebraic equations. Perhaps the best general purpose method for the solution of Ax equal to b is the Gaussian elimination method.

(Refer Slide Time: 00:47)


Direct Methods:
Gaussian Elimination

- **Gaussian elimination**:
  - The most efficient direct method for the general system Ax = b
  - Idea is to convert A into an upper triangular matrix by ***forward elimination*** and solve the converted equations by ***back-substitution***

And one could call this as the most efficient direct method for the general system Ax equal to b. What we mean by general system here, is something that in which A is not specially structured. For example, in some kind of systems may be you do not have the partitive that we come to expect with the CFD type of things we have every element of this is filled. So when you have low sparsity and most of the elements of A are non zero, then for end there is no special diagonal structure like what we were saying here for A. For that kind of general type of system of equations one would say that Gaussian elimination method is probably the most efficient direct method.

There are other direct methods which are much faster which require far fewer numbers of arithmetic operations, but in the general case this would be the best thing. And the idea of this particular approach is to convert the coefficient matrix into an upper triangular matrix through a process known as forward elimination and solve the converted equation by back substitution.

So it is the pictorially shown here we have matrix A which has elements along the diagonal and above the diagonal and below the diagonal, all these things are non-zero elements. And through the process of forward elimination we convert this into an upper triangular matrix here in which we have non-zero coefficients are appearing only in the upper triangular part of it and not in the lower triangular part of it. Below the central diagonal all of them are 0. What it means is that each of them each row here corresponds to an equation.

And here we have this a 1 1 x 1 plus a 1 to x 2 plus a 1 x 3, all that equal to b 1 and then a 2 1 x 1 plus a 2 to x 2 plus a 2 to x 3 and so on up to equal to b 2 and so on like that we have the equations. In every row we have all the coefficients are coming here, but if you look at this particular thing all these elements are 0. For example, if you go to the last row here we will have some coefficient a n n times x n equal to b n. Where, n n and b n are different from what we originally started with, but still it is an equation which contains only one variable x n. So, you go to the last one and then you can solve it directly and say that x n equal to b n by a n n.

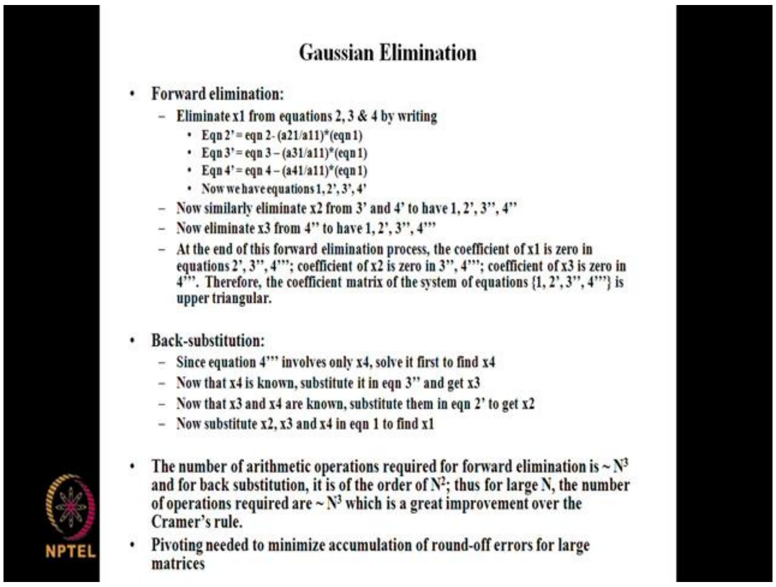
And if you come to the next row here then it is an equation which contains this non-zero diagonal and this non-zero diagonal, so that is like a n minus 1 a n minus 1 x i minus 1 plus a n minus 1 n x n equal to b n minus 1. So that has two variables x n and x n minus 1 out to which we have already completed x n from by solving the last equation and we substitute that value here and you have now an equation containing only one variable x n minus 1. So, you get that directly by solving this.

So, once you convert this coefficient matrix a containing 0's both above and below into an upper triangular matrix in which all those elements which are below the diagonal matrix are 0. Then it is possible to solve this system of equation which is nothing but a modified of writing this equation. This system of equation such that you can start with a last equation get x n and then substitute the value in the one above and then that becomes an equation for x n minus 1 and that will be solved.

And then you come to the next equation which has x n minus 2 x n minus 1 x n as a variables, but by this time we have already evaluated x n and x n minus 1. So, you substitute those values and then solve for x n minus 2. you can do back substitution from the bottom most things and then come up here and then solve for x 1. You can successively solve for x n x n minus 1, x n minus 2, x n minus 3, all the way up to x 2 x 1. So, that is known as a back substitution method.

So, we have a forward elimination method where you are converting these non-zero elements into 0 here and in the process convert the matrix into an upper triangular matrix. And then substitute this upper triangular matrix which is a different way of writing this Ax equal to b as a prime x equal to b prime, so that is derived from this and it is just modified through a certain multiplications and elimination process and the resulting equation structure is modified to give you solution by successive back substitution.

(Refer Slide Time: 06:33)

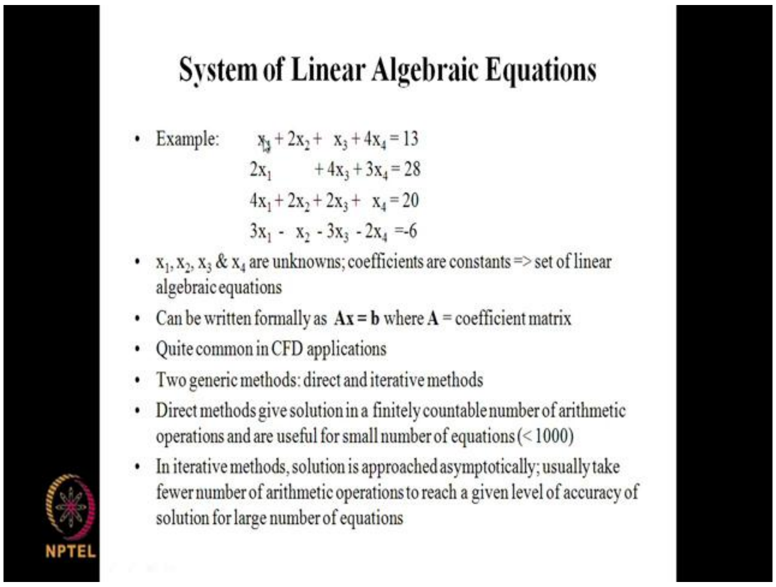


Gaussian Elimination

- Forward elimination:
  - Eliminate x1 from equations 2, 3 & 4 by writing
    - Eqn 2' = eqn 2 - (a21/a11)*(eqn 1)
    - Eqn 3' = eqn 3 – (a31/a11)*(eqn 1)
    - Eqn 4' = eqn 4 – (a41/a11)*(eqn 1)
    - Now we have equations 1, 2', 3', 4'
  - Now similarly eliminate x2 from 3' and 4' to have 1, 2', 3'', 4''
  - Now eliminate x3 from 4'' to have 1, 2', 3'', 4'''
  - At the end of this forward elimination process, the coefficient of x1 is zero in equations 2', 3'', 4'''; coefficient of x2 is zero in 3'', 4'''; coefficient of x3 is zero in 4'''. Therefore, the coefficient matrix of the system of equations {1, 2', 3'', 4'''} is upper triangular.
- Back-substitution:
  - Since equation 4''' involves only x4, solve it first to find x4
  - Now that x4 is known, substitute it in eqn 3'' and get x3
  - Now that x3 and x4 are known, substitute them in eqn 2' to get x2
  - Now substitute x2, x3 and x4 in eqn 1 to find x1
- The number of arithmetic operations required for forward elimination is ~ N^3 and for back substitution, it is of the order of N^2; thus for large N, the number of operations required are ~ N^3 which is a great improvement over the Cramer's rule.
- Pivoting needed to minimize accumulation of round-off errors for large matrices

So, let us just see how it actually works. You may be familiar with this.

(Refer Slide Time: 06:37)



System of Linear Algebraic Equations

- Example:

$$\begin{aligned} x_1 + 2x_2 + x_3 + 4x_4 &= 13 \\ 2x_1 \quad\quad + 4x_3 + 3x_4 &= 28 \\ 4x_1 + 2x_2 + 2x_3 + x_4 &= 20 \\ 3x_1 - x_2 - 3x_3 - 2x_4 &= -6 \end{aligned}$$

- x_1, x_2, x_3 & x_4 are unknowns; coefficients are constants => set of linear algebraic equations
- Can be written formally as **Ax = b** where **A** = coefficient matrix
- Quite common in CFD applications
- Two generic methods: direct and iterative methods
- Direct methods give solution in a finitely countable number of arithmetic operations and are useful for small number of equations (< 1000)
- In iterative methods, solution is approached asymptotically; usually take fewer number of arithmetic operations to reach a given level of accuracy of solution for large number of equations

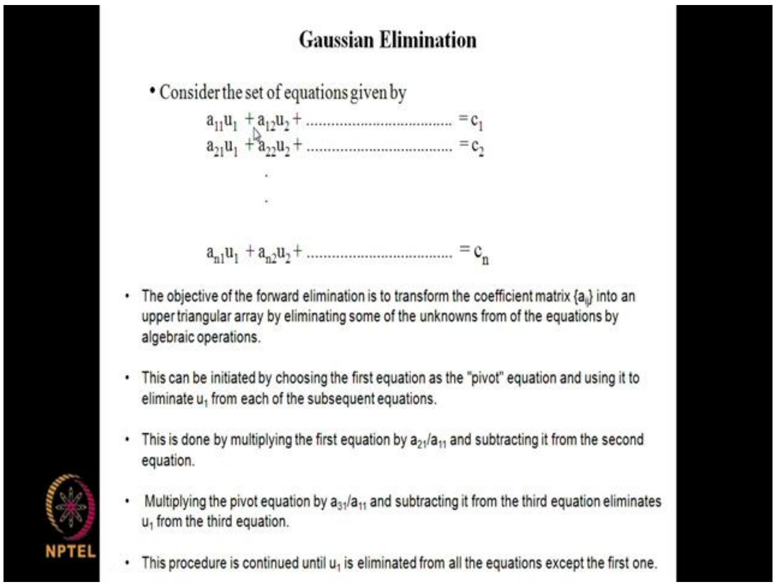
You have a set of 4 equations here, involving x 1 x 2 x 3 x there are 4 equations. If you want to eliminate x you first start by saying that you will eliminate x 1 from this equation and this equation this equation, and how can you do that. If you multiply this equation the first equation by 2 and subtract from this, then you would have gotten rid of this x 1 from the subtracted equation. In the process you will be eliminating from this equation with the help of the first equation you will be eliminating x 1. Similarly, you come to this equation, the third equation with the help of the first equation you can eliminate x 1 from this. For example, multiply the first equation by 4 and subtract from this that will get rid of this 4 x 1. And similarly you come back to this and then you multiply this by 3 and then subtract from this, the resulting equation will not have x 1.

So, in this process using the first equation you can eliminate x 1 from the second equation, third equation, fourth equation. In the process what are you doing you are just multiplying one equation and then subtracting from another equation, all valid operations. As long as it done equally to both left hand side and right hand side of the equation we are not changing anything except reordering a bit. So, in the process you can eliminate x 1 from the 3 preceding equations.

Now we have 1 equation involving all the four variables and three equations involving only 3 variables x 2 x 3 x 4, you have eliminated the participation of x 1 from this. Now you take the second equation which has x 2 x 3 x 4 and then use that to eliminate x 2 from the third equation and the fourth equation. And if you do that then you gets a first equation containing all the 4, second equation containing only 3; x 2 x 3 x 4, and third and fourth equation contain x 3 and x 4 in both equations. Now you take the third equation and eliminate x 3 from the fourth equation, so then you have a fourth equation which contains only x 4. Then you have fourth equation containing x 4 you solve for x 4

And then you come to the modified third equation which has x 3 and x 4 you use x 4 which substitute x 4 and then solve this for x 3. And then you come to the modified second equation and then substitute x 4 and x 3 that you got solve for x 2 and then you come to the first equation substitute for x 2 x 3 and x 4 and then you get x 1. So, that is the process by which we can do this.

(Refer Slide Time: 09:50)



Gaussian Elimination

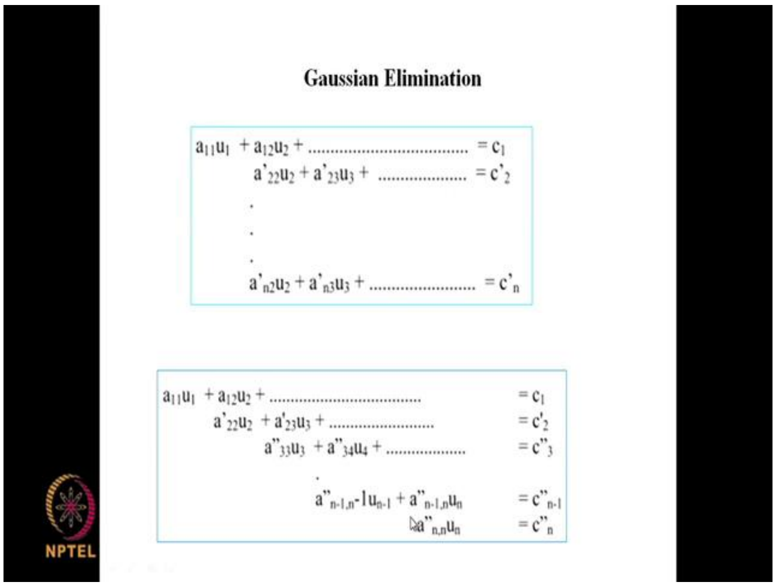
- Consider the set of equations given by

$$a_{11}u_1 + a_{12}u_2 + \ldots\ldots\ldots\ldots = c_1$$
$$a_{21}u_1 + a_{22}u_2 + \ldots\ldots\ldots\ldots = c_2$$
$$\vdots$$
$$a_{n1}u_1 + a_{n2}u_2 + \ldots\ldots\ldots\ldots = c_n$$

- The objective of the forward elimination is to transform the coefficient matrix $\{a_{ij}\}$ into an upper triangular array by eliminating some of the unknowns from of the equations by algebraic operations.
- This can be initiated by choosing the first equation as the "pivot" equation and using it to eliminate u_1 from each of the subsequent equations.
- This is done by multiplying the first equation by a_{21}/a_{11} and subtracting it from the second equation.
- Multiplying the pivot equation by a_{31}/a_{11} and subtracting it from the third equation eliminates u_1 from the third equation.
- This procedure is continued until u_1 is eliminated from all the equations except the first one.

And that is what is shown here, you take this set of a equations the objective of the forward elimination is to transform the coefficient matrix a i j into an upper triangular array by eliminating some of the unknowns from the equations by algebraic operations. This can be initiated by choosing the first equation as the pivot equation and using it to eliminate. In this case u 1 from each of the subsequent equations, and this is done by multiplying the first equation by a 21 by a 11 and subtracting it from the second equation. If you multiply this by a 21 divided by a 1 1 then obviously this will cancel out and you get a 21 u 1 subtract it from this you eliminate this. And then similarly a 31 by a 11 is what you multiply the first equation by and then subtract from the third equation and then you can do that.

(Refer Slide Time: 10:30)



Gaussian Elimination

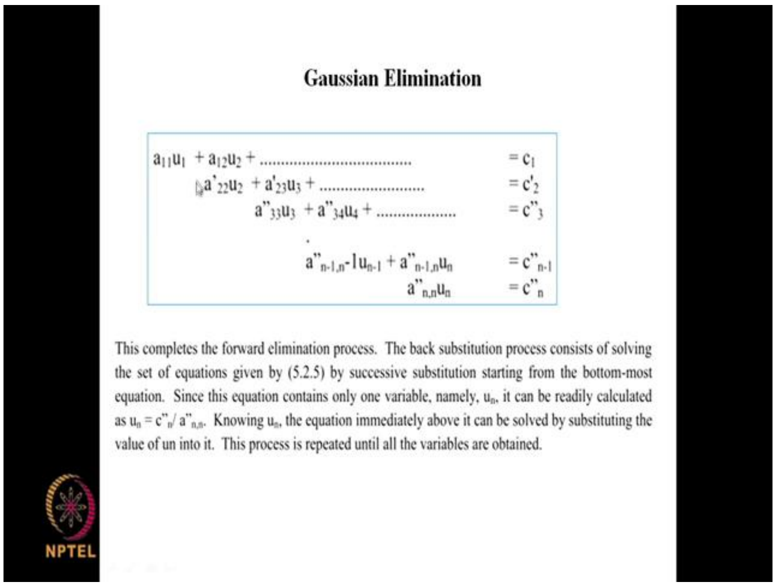
$$\begin{aligned} a_{11}u_1 + a_{12}u_2 + \dots\dots\dots &= c_1 \\ a'_{22}u_2 + a'_{23}u_3 + \dots\dots\dots &= c'_2 \\ &\vdots \\ a'_{n2}u_2 + a'_{n3}u_3 + \dots\dots\dots &= c'_n \end{aligned}$$

$$\begin{aligned} a_{11}u_1 + a_{12}u_2 + \dots\dots\dots &= c_1 \\ a'_{22}u_2 + a'_{23}u_3 + \dots\dots\dots &= c'_2 \\ a''_{33}u_3 + a''_{34}u_4 + \dots\dots\dots &= c''_3 \\ &\vdots \\ a''_{n-1,n-1}u_{n-1} + a''_{n-1,n}u_n &= c''_{n-1} \\ a''_{n,n}u_n &= c''_n \end{aligned}$$

So, if you do this at the end of the first set of elimination of u 1 from all these equations you have the first equation unchanged, the second equation is now with modified coefficients a primes and also the right hand side c prime here, and this set of equations will not contain u 1 as a variable. And now you take this set of equations and make use of this as the pivoting equation and eliminate u 2 from all these things and then if you keep on doing this until you have done for all of them you will end up with first equation containing all the n variables. Second equation, containing all variables except u 1 third equation containing all the variables except u 1 and u 2 and so on; N minus 1 th equation containing all the equations all the variables except up to n minus 2, it has n minus 1 and n minus n here u n here. Then the last equation contains only this equation.

So, now you can solve this for u n and then substitute in this solve for u n minus 1 and then you go back here and then solve this for u n minus 2 and so on. So, that is the back substitution process.

(Refer Slide Time: 11:50)



Gaussian Elimination

$$\begin{array}{llll} a_{11}u_1 + a_{12}u_2 + \ldots\ldots\ldots & & & = c_1 \\ & a'_{22}u_2 + a'_{23}u_3 + \ldots\ldots\ldots & & = c'_2 \\ & & a''_{33}u_3 + a''_{34}u_4 + \ldots\ldots & = c''_3 \\ & & . & \\ & & a''_{n-1,n-1}u_{n-1} + a''_{n-1,n}u_n & = c''_{n-1} \\ & & a''_{n,n}u_n & = c''_n \end{array}$$

This completes the forward elimination process. The back substitution process consists of solving the set of equations given by (5.2.5) by successive substitution starting from the bottom-most equation. Since this equation contains only one variable, namely, u_n, it can be readily calculated as $u_n = c''_n / a''_{n,n}$. Knowing u_n, the equation immediately above it can be solved by substituting the value of un into it. This process is repeated until all the variables are obtained.

So, this completes the forward elimination process by which time you have got this set of equations, in each of which one additional variable is missing. Back substitution process consists of solving the set of equations by successive substitution starting from the bottom most equation. Since, this equation contains only one variable u n it can be readily calculated as c double prime and divided by a double prime n n, and then u come here and then you can solve it like this. This process repeated until all the variables are obtained. So, this is the Gaussian elimination process.

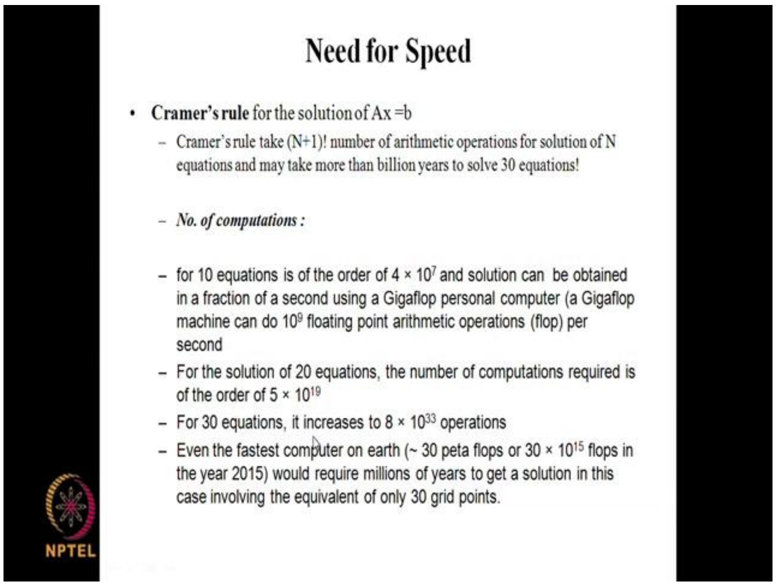
To summarize, you convert A into an upper triangular matrix by a process of forward elimination in which you make use of one equation to eliminate that particular variable one from all the subsequent equations. And then you go on with the process until you have eliminated n minus 1 variable from the last equation through a process of this successive forward elimination, and then you get the solution by backward substitution. And this is considered as very efficient method which will involve n cube by 3 number of arithmetic operations for large values of n.

When n is large that is when the number of unknowns is large then the number of multiplication and division required to do this whole process is about n cube by 3. And n cube by 3 is the number of arithmetic operations that are required to do the forward elimination process. The backward substitution process requires something like n square number of the arithmetic operations. So, the total number of operations says n cube by 3 plus a constant times n square.

And as n becomes very large, then n square contribution to the whole sum will be very small. For example, if n is equal to 1000 the first operation will required 10 cube 1000 to the power cubed so that is 10 to the power 9 by 3. And the second set of operations will involve only of the order of 1000 square that is million. So, we can forget the millions when we have the billions. And if is n is 10000 then the first one will have 10 to the power 12 and the second one will have only 10 to the power 8. So, that becomes even more negligible

So, as the number of equations becomes bigger the time of competition for forward elimination increases. And the backward substitution also increases, but not as much as rapidly as a forward elimination process. Therefore, for large values of n Gaussian elimination takes number of multiplications and divisions of the order of n cube by 3.

(Refer Slide Time: 15:21)



Need for Speed

- **Cramer's rule** for the solution of Ax =b
  - Cramer's rule take (N+1)! number of arithmetic operations for solution of N equations and may take more than billion years to solve 30 equations!
  - ***No. of computations :***
  - for 10 equations is of the order of 4×10^7 and solution can be obtained in a fraction of a second using a Gigaflop personal computer (a Gigaflop machine can do 10^9 floating point arithmetic operations (flop) per second
  - For the solution of 20 equations, the number of computations required is of the order of 5×10^{19}
  - For 30 equations, it increases to 8×10^{33} operations
  - Even the fastest computer on earth (~ 30 peta flops or 30×10^{15} flops in the year 2015) would require millions of years to get a solution in this case involving the equivalent of only 30 grid points.

And what is n cube by 3 if you go back to the morning example of, say now you say that n is equal to 10000 10 to the power 4. Then the number of arithmetic operations is 10 to the power 4 cubed, so that is 10 to the power of 12. So, even for unknowns 10000 equations here require only 10 to the power of 12 by 3 arithmetic operations. If we have a Gigaflop machine or a petaflop machine you will get the solution in fraction of a second; whereas, the Cramer's rule for 30 equations itself its taking 10 to the power 33 operations. And Gaussian elimination for 10000 equations will take of the order of 10 to the power 12 operations. And that is the advantage of the Gaussian elimination method.

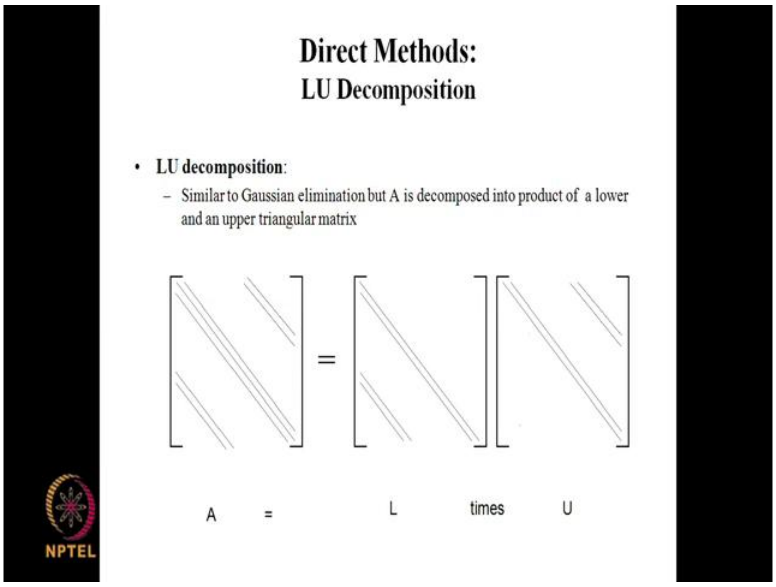
And this is considered as the most efficient method for the solution of Ax equal to b. When a does not have any specific advantage that we can exploit. Which is not the case in CFD problems? In many CFD problems we have sparsity as one very characteristic feature. It also plays to the advantage of Gaussian elimination, because in order to do the elimination process from your set of equations like this to reach up to this eliminated process in the process of that you would have multiplied these coefficients enormous number of times, we have 10000 equations you would have multiplied this by may be close to 10000 number of times.

So that means, any small errors that we make rounding of errors can build up in the process. It is important to choose the right kind of coefficients to eliminate and this is known as a pivoting strategies and I would prefer you to standard books of numerical methods to look at the pivoting strategy.

In case you are doing Gaussian elimination method for the solution of Ax equal to b in the general case. The buildup of round of errors is not such a big problem when you have sparse matrix as we have in CFD, but still it something that we have to watch out for. We also have to watch out for division by 0 and those type of things and if you do proper pivoting strategy then division by 0 will not happen.

Now, another important thing about Gaussian elimination process is, that it will work provided we have a set of equations which are irreducible and they promise unique solution. In the sense if determinant of a is not 0 then you can make use a Gaussian elimination method. If you want to get very good accuracy of competition then we have to use a proper pivoting strategy. So, that is Gaussian elimination method.

(Refer Slide Time: 18:44)



Direct Methods:
LU Decomposition

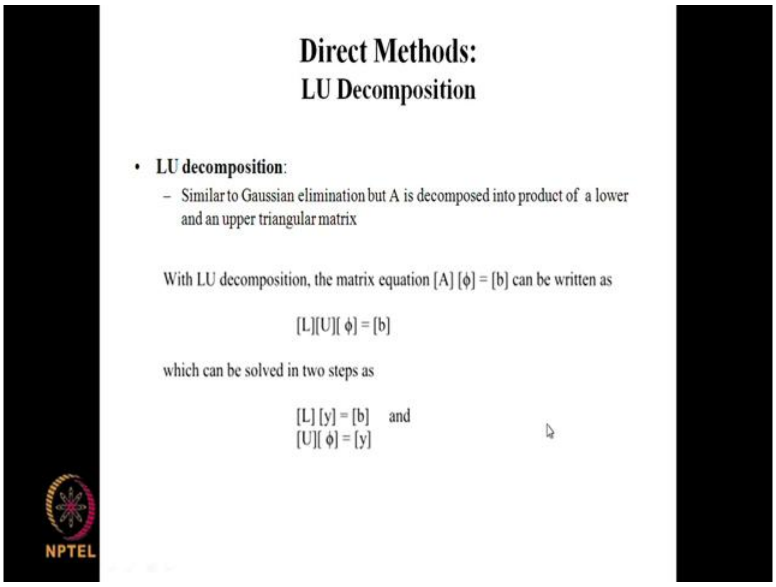
- **LU decomposition**:
  - Similar to Gaussian elimination but A is decomposed into product of a lower and an upper triangular matrix

=
A
=
L
times
U
NPTEL

We look at one more method which is sometimes used in CFD competitions. Gaussian elimination method is not so much used because it does not take advantage of the diagonal structure. Here, L u decomposition is also something very similar it also does not take advantage of the units, original form, in standard form it does not take advantage of the diagonal structure, but it has certain special features which will make it more attractive than Gaussian elimination method. Now what is L U decomposition method? Here the coefficient matrix A alone is decomposed into a product of a lower triangular matrix and an upper triangular matrix, so that is A is written as L times U. This is matrix multiplication something that we have to keep in mind.

So, you have a matrix which has both non-zero components, both above the main diagonal and below the main diagonal like this. And this is decomposed into a product of a lower diagonal matrix, so that is in which all the elements above the diagonal 0 and only those below the main diagonal may contain non 0 elements. So this L times U which is where U is an upper triangular matrix like this.

(Refer Slide Time: 20:22)



Direct Methods:
LU Decomposition

- **LU decomposition**:
  - Similar to Gaussian elimination but A is decomposed into product of a lower and an upper triangular matrix

With LU decomposition, the matrix equation $[A]\,[\phi] = [b]$ can be written as

$$[L][U][\phi] = [b]$$

which can be solved in two steps as

$$[L]\,[y] = [b] \quad \text{and}$$
$$[U][\phi] = [y]$$

NPTEL

So, if we can do this decomposition then the solution of the resulting L U equal to b is relatively easy. So with L U decomposition the matrix equation A phi equal to b is written as L times U times phi equal to b. And this equation is solved in two steps; first we solve for L y equal to b. And in this equation the unknown is y, because we know L we have decomposed A into L and U we will see how it can be done, and then b is all already known, so from this you get y here. And once you get y you can solve for U phi equal to y and from this you get phi. Suppose you put u phi equal to y in this equation then that becomes L y equal to b; if I solve for y and then substitute the y here and then solve for phi in this.

And the advantage of this and this solution is that L y equal to b is a forward substitution process. In this first equation has only one variable, second equation has two variables, and out of which we already got one and then you can get this by substitution. And then you can come to the third equation which has again some three values and you can continue like that. So, the solution of L U equal to b is by forward substitution, it is not by forward elimination.

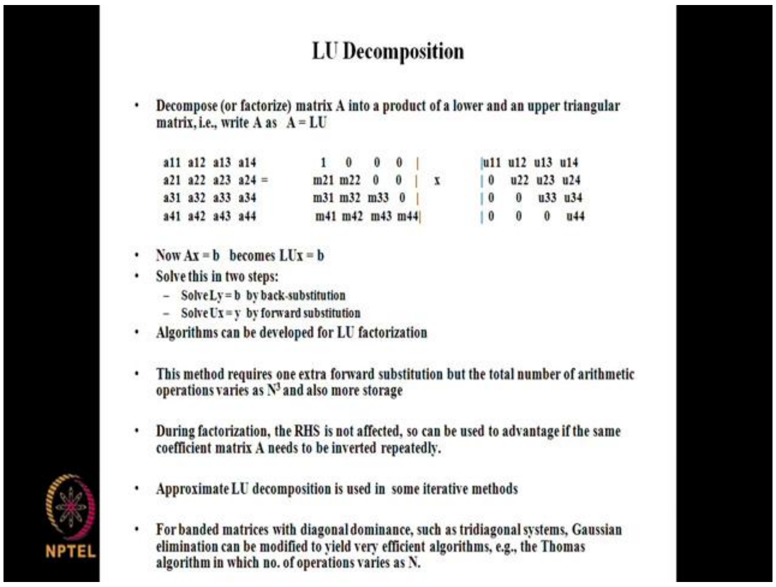
And similarly solution of U phi equal to y is by backward substitution and that again is not by elimination process it is by back substitution process. So, solution of this requires n square number of arithmetic operations and this requires n square number of arithmetic operations. It is the order of trice the number of back substitution process is what is required which is much less than the n cube by 3 numbers of mathematical operations that are required for the Gaussian elimination process.

But the catch is what it takes to decompose A into a product of L and U. And it turns out that is the really time consuming process and the process of writing A equal to L U is very similar to the process of Gaussian elimination, so that it will take n cube by 3 number of arithmetic operations just to do this decomposition. When you look at it L U decomposition is not as good as Gaussian elimination, because you have the same number of arithmetic operations which are required to do the decomposition. And after decomposition you have to solve twice this forward or backward substitution process. So you are solving at least one of these equations more than in the case of Gaussian elimination method. In fact, it is slightly more demanding in terms of number of arithmetic operations.

But if you have a problem in which A phi equal to b is followed by some A psi equal to c and A eta equal to d like that, in which you have to solve a number of equations for which A is same matrix. Then you need to do the decomposition once and then you can solve this as L U phi equal to b and then L U psi equal to c and then L U eta equal to delta in that way you can solve this every time. For the first equation you are taking more number of arithmetic operations than the Gaussian elimination method. But in the second and third equations you are taking only n square number of populations because you are solving only these and not for L and U.

If you have a special situation in which we have to solve Ax equal to b and Ay equal to c and Az equal to d and so on like that, then it would make sense to convert A into L U so that you can write this as L U x equal to b and L U y equal to c and L U z equal to d and solve these things by combination of forward and backward substitution. So when you do that then it becomes advantageous.

(Refer Slide Time: 25:20)



LU Decomposition

- Decompose (or factorize) matrix A into a product of a lower and an upper triangular matrix, i.e., write A as A = LU

$$\begin{matrix} a11 & a12 & a13 & a14 \\ a21 & a22 & a23 & a24 \\ a31 & a32 & a33 & a34 \\ a41 & a42 & a43 & a44 \end{matrix} = \left|\begin{matrix} 1 & 0 & 0 & 0 \\ m21 & m22 & 0 & 0 \\ m31 & m32 & m33 & 0 \\ m41 & m42 & m43 & m44 \end{matrix}\right| \times \left|\begin{matrix} u11 & u12 & u13 & u14 \\ 0 & u22 & u23 & u24 \\ 0 & 0 & u33 & u34 \\ 0 & 0 & 0 & u44 \end{matrix}\right.$$

- Now Ax = b becomes LUx = b
- Solve this in two steps:
  - Solve Ly = b by back-substitution
  - Solve Ux = y by forward substitution
- Algorithms can be developed for LU factorization
- This method requires one extra forward substitution but the total number of arithmetic operations varies as N^3 and also more storage
- During factorization, the RHS is not affected, so can be used to advantage if the same coefficient matrix A needs to be inverted repeatedly.
- Approximate LU decomposition is used in some iterative methods
- For banded matrices with diagonal dominance, such as tridiagonal systems, Gaussian elimination can be modified to yield very efficient algorithms, e.g., the Thomas algorithm in which no. of operations varies as N.

So, the factorization of A into L and U there are some specialized methods for this and we can use those methods for this and I would prefer you to standard books of numerical methods and they (Rfer Time: 25:44) the fact that this is equal to this times this, so this times this, plus this times this, plus this times this, and so on. Similarly, if you take this row a 22 it is equal to this row multiplied by this column. The corresponding elements m 21 u 12 plus m 22 u 22 plus n 23 u 32 plus m 24 u 42 is all equal to a 22.

So, you have that that condition based on that condition you can find out m and u elements of the lower and upper triangular matrix. In a certain sequence and that sequence is in the form of (Refer Time: 26:38) ateral algorithm. And those type of algorithms are there which will enable you to solve for elements of L and U in a sequential way and that decomposition takes n cube by 3 number of arithmetic operations.

In that sense L U decomposition is not more efficient than Gaussian elimination method when you are solving only 1 equation Ax equal to b. But if you are solving several equations with the same A, with the same coefficient matrix then you convert A into L U once and then use that L U decomposed form of the equation to solve subsequent equations. When you have this type of approach then it is possible to come up with in an efficient method. And there are other conditions which will see later on in which this L U decomposition used. For example, through a process known as approximate L U decomposition or incomplete L U decomposition, and these are used to construct more efficient methods than (Refer Time: 27:57) decomposition or Gaussian elimination or even Gaussian method.

So, what we have seen in this lecture are two special efficient methods for the solution of Ax equal to b. And these are two direct methods: the first one is the Gaussian elimination method; the other is the L U decomposition method. When you have banded matrices, when you have diagonal structures, and when you have sparsity then it is possible to take advantage of the sparsity and bandedness to come up with very efficient method especially for tridiagonal matrix.

In the next lecture we look at the tridiagonal matrix algorithm which is applicable for the special matrices which the coefficient matrix here has three adjacent diagonals including a main diagonal. So, that is the type of an A phi equal to b kind of solution that you have. You have this Ax equal to beta type of thing involving only three diagonals when you are discretizing. For example, d square t by d x square equal to constant. So, if you do that using central differencing then you get tridiagonal matrix. In case of 1 d type of equations involving a 1 d elliptic type of things we get a tridiagonal matrix, and then you have the tridiagonal matrix then we can we can have an efficient solution method. We have also seen this tridiagonal kind of structures when we are dealing with compressible flow calculations.

So, in the next lecture we will look at the tridiagonal matrix algorithms and then we will see under what conditions it can be applied. And we will look at some other basic methods like the Gauss seidel method which we have already used in the first module and the (Refer Time: 30:07) method.

Thank you.