Numerical Analysis
Prof. Dr. S. Baskar
Department of Mathematics
Indian Institute of Technology-Bombay

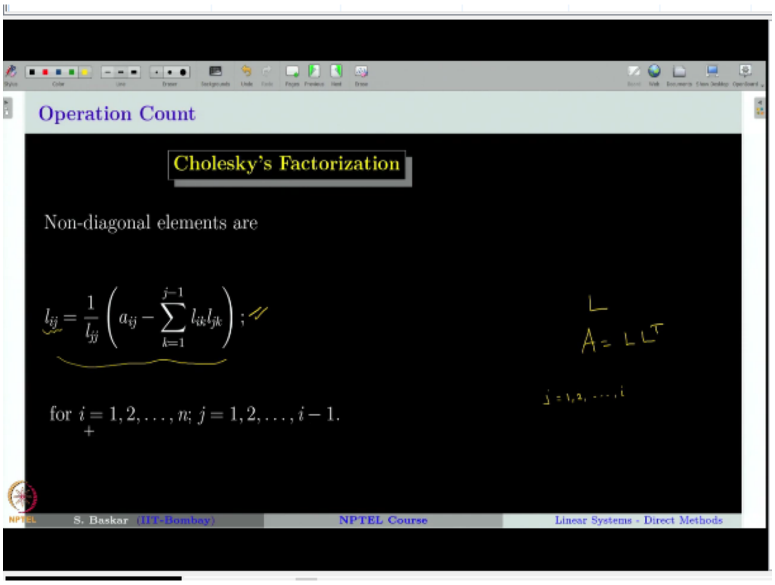
Lecture-12
Linear Systems: Operation Count for Direct Methods

Hi, we have introduced some direct methods for solving non-singular linear systems. In this we have introduced Gaussian elimination method, Doolittle factorization method, Crout factorization method and Cholesky's factorization method. Direct methods give exact solution when there is no arithmetic error involved in it. However, these methods are often not preferred in practical situations because these methods are quite costly to compute when the systems are large in size.

Now, when this is said our question is how to measure the cost involved in computing solution using any method? Well, that can be done by counting the number of arithmetic operations involved in a method. We will try to understand this by doing operation count for some direct methods, we just take Cholesky's factorization and Gaussian elimination method and we will count the number of arithmetic operations involved in these two methods.

And see how to compare their efficiency in terms of the number of arithmetic operations involved in them. Let us take the Cholesky's factorization first.

(Refer Slide Time: 01:53)



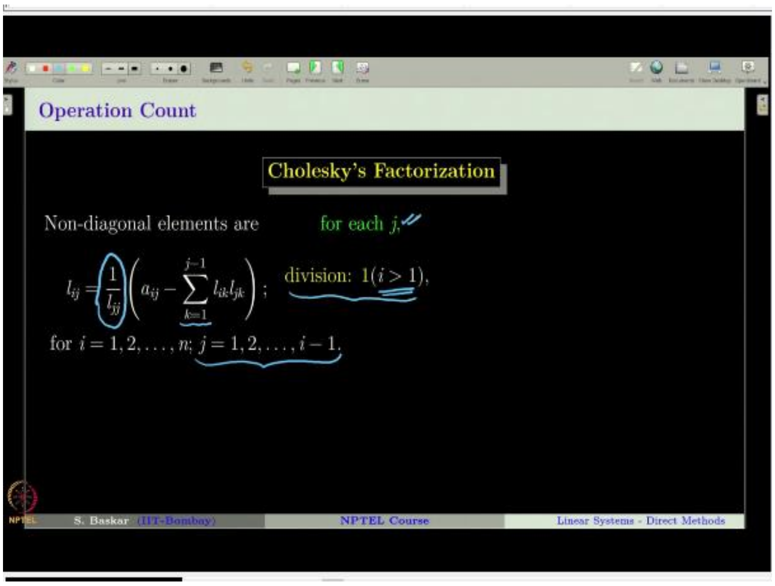
Recall that in the last class we have given two ways to construct Cholesky's factorization; one is by going step by step starting from the sub matrix of order 1 compute its Cholesky's factorization which is just direct and then using that you can find the Cholesky's factorization of the sub matrix of A of order 2. Then once you get that which we name as L_2 then you go to find L_3 and so on.

And in that way we construct the Cholesky's factorization for the given $n \times n$ matrix. That is one way of doing; another way is by direct comparison. In that way also we have given the expressions for diagonal and non diagonal elements of the Cholesky's factorization in the last class. We will use these expressions that are the expressions obtained using direct comparison to do the arithmetic operation count.

Because these are very explicit to see, therefore it is very easy for us to count the number of arithmetic operations involved in them. Recall the expression for non diagonal elements of the Cholesky's factorization is given by this expression. These are the non diagonal elements of the matrix L such that $A = LL^T$, just to recall. Now let us try to count the number of operations involved in this expression.

Remember that l_{ij} is non zero only for j = 1, 2 up to i, in that non diagonal elements are only up to i – 1. Therefore in this expression j runs from 1 to i - 1 and that gives you all the non diagonal elements of the ith row before the diagonal element and this has to be done for all the rows and there are n such rows. Therefore i should run from 1 to n.

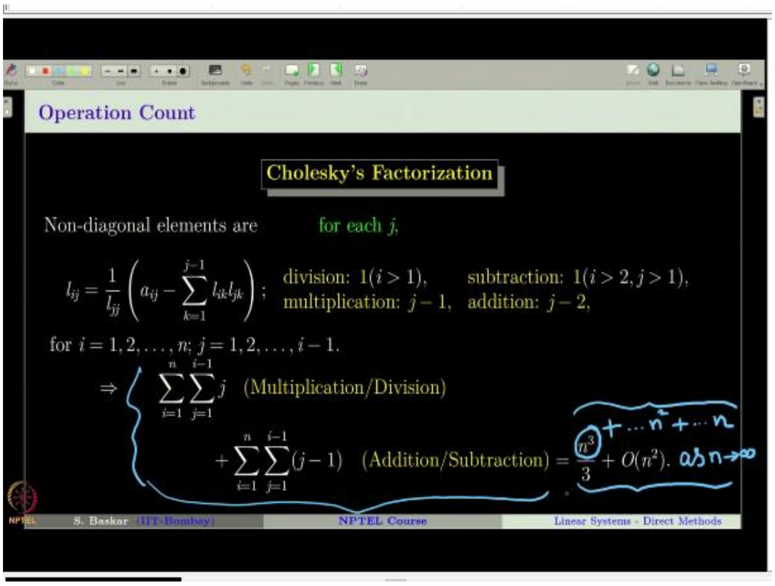
(Refer Slide Time: 04:26)



Now let us observe this expression and try to see how many arithmetic operations are involved in computing these quantities? For each j you can observe that there is one division involved in this expression. Of course it should be for $i > 1$ because when you take $i = 1$ there is no non diagonal element which can be non zero. That can be also seen in this expression.

j should run from 1 to $i - 1$, therefore when $i = 1$, j is not going to exist. Therefore you have one division for each j for i starting from 2 and goes up to n.

(Refer Slide Time: 05:26)



Similarly you can see that the expression has one subtraction and that should be for $i > 2$ and $j > 1$. You can observe that from these expressions and also the way the summation is written and now let us see how many multiplications are involved. You can see that there is one multiplication in each term of this summation and there are $j - 1$ such terms therefore there are $j - 1$ multiplication for each j.

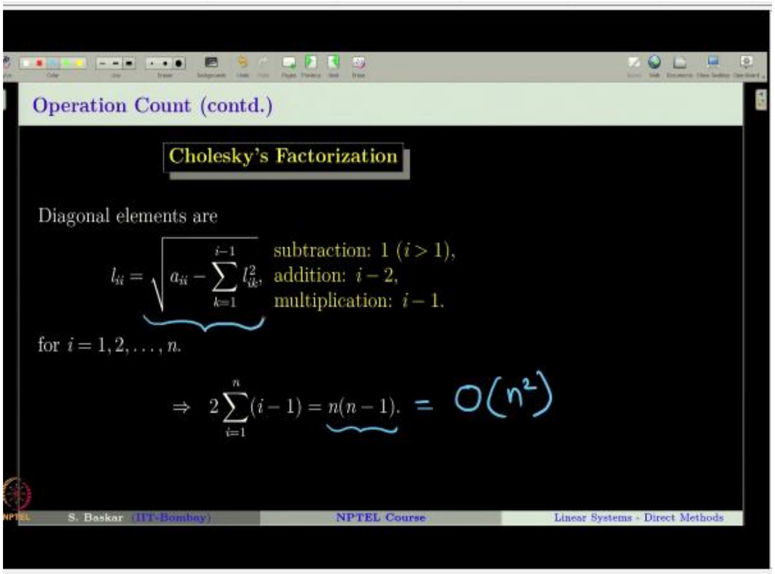
Similarly there are $j - 2$ additions for each j, with this you can see that each component l_{ij} for each j and each i you have this many arithmetic operations, you can see that there are $i - 1$ js like this and that will give us one typical row and there are n such rows. Therefore we have to sum them all suitably and we can see that you have these many multiplications and divisions plus this many additions and subtractions involved in obtaining the non diagonal elements of the Cholesky's factorization matrix L.

Now we just have to add these terms, there are some elementary formulas that can be used in order to simplify these expressions and you can see that this expression will finally simplifies to $\frac{n^3}{3}$ plus some constant times n^2 plus some constant times n. In all this operation counts generally we are not interested in the exact expression of this operation counts rather we are interested in what is the power of the leading term in this expression.

Because that tells us how fast the computational cost will increase as n increases. That is the main interest in studying the efficiency of any algorithm not only in Cholesky's factorization. That is why we are not interested in obtaining the exact expression for this term rather we are only interested in seeing what is the leading term. In fact we can say that the number of arithmetic operations involved in the non-diagonal elements of the Cholesky's factorization is of order n^3.

It is pretty costly in general. Let us keep this in mind and go ahead and count the number of operations involved in the diagonal elements.

(Refer Slide Time: 08:46)

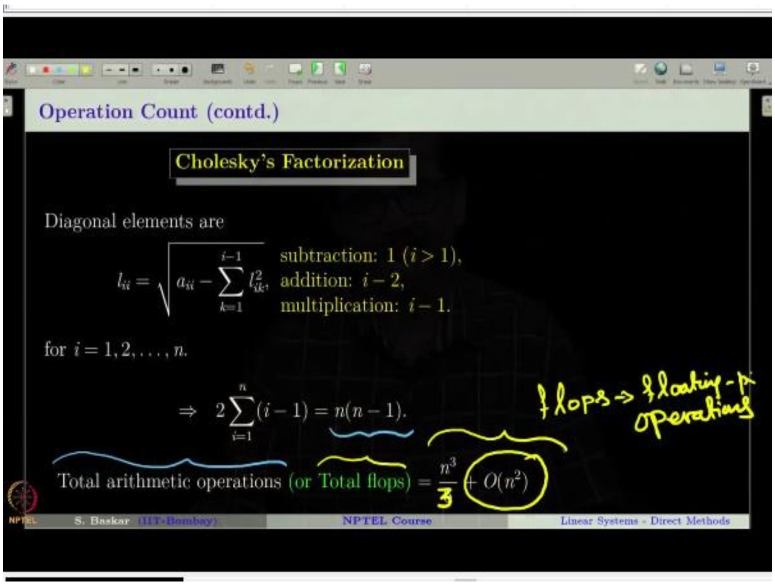


Recall that the diagonal elements are given by this expression, you can see that for each i we have one subtraction of course for $i > 1$, $i - 2$ additions and $i - 1$ multiplications. Remember when we say subtraction it may be also addition, because subtracting a positive number with a negative number leads to actually addition. So, when I say subtraction, I am just looking at this sign and then counting.

It may land up to be a addition of two positive numbers. Let us see what is the total number of arithmetic operations involved in computing diagonal elements and the total number of arithmetic operations is given by $n(n-1)$. Well, it is contributing to $O(n^2)$. Apart from that remember we also have to take a square root but generally square root will not contribute to order n^3.

Therefore, the leading order term is actually coming from the non diagonal elements and that is the costliest path which is contributing to order n^3 in the Cholesky's factorization.

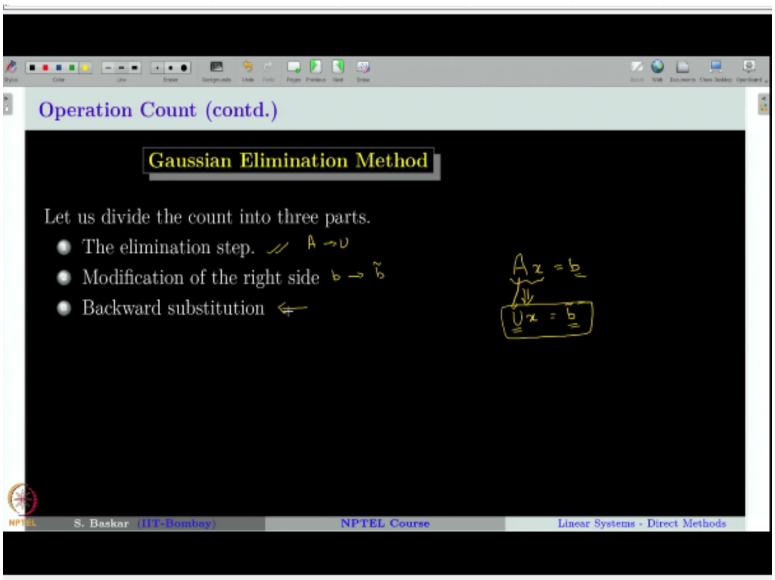
(Refer Slide Time: 10:22)



Therefore, the total number of arithmetic operations involved in Cholesky's factorization is nothing but the sum of the arithmetic operations involved in the non-diagonal elements computation plus the number of arithmetic operations involved in the computation of diagonal elements. You can see that that contributes to $\frac{n^3}{3}$, plus some expression whose leading term is n^2.

In literature this is also called the total flops. Flop means it is the floating point operations. Now we got an idea of how many arithmetic operations or flops involved in Cholesky's factorization? With this we can understand the time taken by the Cholesky's factorization in order to get a LU decomposition of a given symmetric positive definite matrix. With this let us now go to the Gaussian elimination method. And try to see how many arithmetic operations are involved in the Gaussian elimination method.

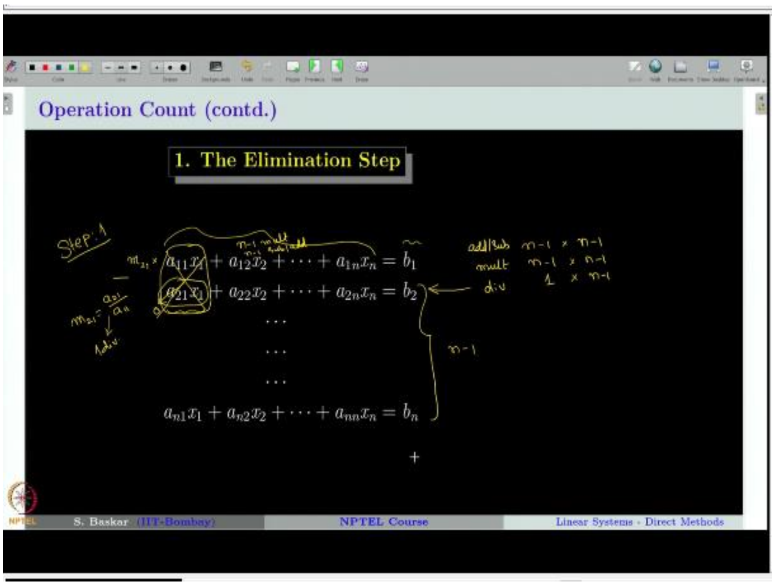
(Refer Slide Time: 11:59)



Let us recall that Gaussian elimination method has three parts. One is the left hand side elimination, remember we have to eliminate this to get an upper triangular matrix, Ux equal to some $\tilde{b}$, some other b will come because when you do all row operations correspondingly you will do the same operation on the right hand side vector also. Therefore, when you are modifying the left hand side coefficient matrix correspondingly the right hand side will also get modified.

Therefore, we can say that the Gaussian elimination method involves three steps. One is the elimination step that is eliminating A to the upper triangular matrix U that is A to U and then the modification of the right hand side vector that is b to $\tilde{b}$. Once you do that you get an upper triangular system then you go for the backward substitution to get the solution. So, that is the way we have done the Gaussian elimination method. Therefore the method involves three steps. Let us try to see how to do the operation count for each of these steps.

(Refer Slide Time: 13:26)



Let us take the elimination step. In this step if you see to make this coefficient 0 what we do is we multiply the first equation with m_{21} and then we will subtract that equation with the second equation, roughly speaking that is what we do. So, how many multiplications are involved in just making this term 0 and thereby getting a modification for the second equation at the step 1. That is our first task.

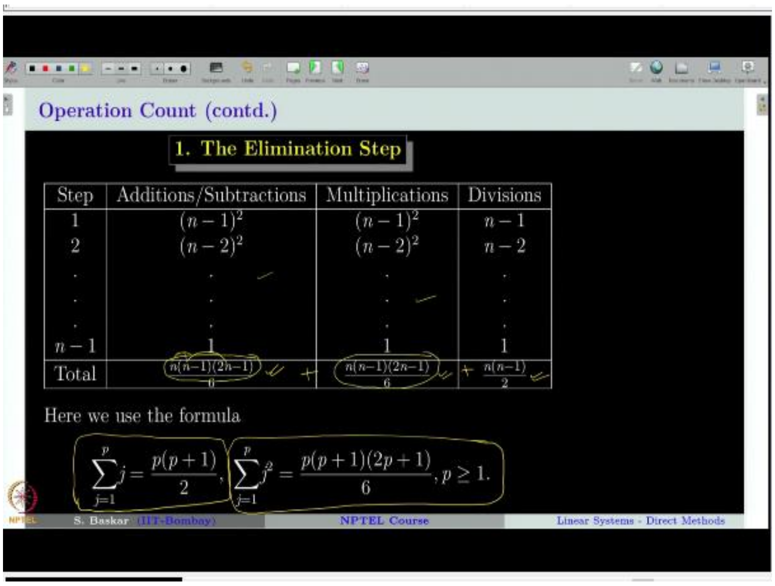
You can see that there are n multiplications, remember I am only doing the left hand side elimination, the right hand side we will do later. The left hand side elimination has in order to eliminate the first coefficient of equation 1 in step 1. We have to do n number of multiplications and then how many addition or subtraction is involved? Well again there are n terms, therefore n addition or subtractions are involved in this.

In all this you note that we already know what is the coefficient of the first term. Therefore we will not do this elimination process explicitly in our code, we will directly write it as 0. Therefore we will not ask our computer to make this multiplication and this subtraction. This is how we will write our code. Therefore this is not involved in our operation count. Apart from this you have $n - 1$ multiplication and $n - 1$ subtraction or addition.

Now how many divisions are involved? Well in m_{21} we are writing it as $\frac{a_{21}}{a_{11}}$. Therefore one division is involved in this. So, in order to eliminate the first term of the second equation. In step 1 we need $n - 1$ multiplication, $n - 1$ subtraction or addition and one division. Now this is for one typical equation. How many such equations are there? There are $n-1$ equations. So, therefore you need $n-1$ addition or subtraction for one single equation.

Like that you have $n-1$ such equation. Similarly how many multiplications? For one single equation you have $n-1$ multiplications and there are $n-1$ such equations. Therefore $(n-1)^2$ multiplications are involved. How many divisions are there? For one equation you have one division and there are $n-1$ such equations are there.

(Refer Slide Time: 16:53)



Therefore finally how many operations are involved at step 1? We have $(n-1)^2$ad dition or subtraction, $(n-1)^2$ multiplications and $n-1$ division are involved in step 1. I hope you have understood how we have counted the operations. Remember we have counted only the arithmetic operations and that too for Naive Gaussian elimination method. We are not counting the pivoting operations because that involves finding maximum.

Actually one has to also count that but we are just doing the arithmetic operation counts because that is rather simple to count. That is why we are counting and judging the efficiency of a method only through the arithmetic operations. Now let us go to the second step, well in the second step we have only $n-2$ equations, because already this much is done. Therefore we will start from third equation onwards and go up to n.

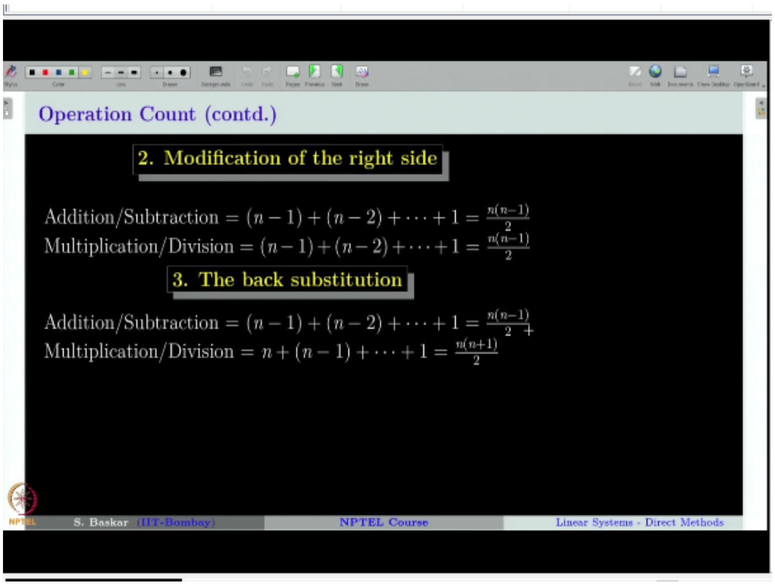
Therefore we have $n-2$ such equations and similarly all these elements are made 0. Therefore we will not be doing any operation with this; we will be doing operations in order to make this term to be 0. Therefore we will not even involve these terms in our operation because we will directly put their values as 0 Therefore we will be starting from this term onwards and do.

Therefore you have $n-2$ such addition subtraction and $n-2$ such multiplications are there and of course for each equation you have one division involved in that multiplication term m. Therefore in the second step we have $(n-2)^2$ addition, subtraction $(n-2)^2$ multiplication and $n-2$ divisions. Now you can go on doing this. In the third step you will have $(n-3)^2$ addition subtraction $(n-3)^2$ multiplication and $n-3$ divisions.

And similarly you go on like that. Now what is the total number of operations involved in the elimination step? Well you can add them all up and that can be written like this. Well, we are using an elementary formula which is known to us for this term and for these terms we use this formula. So, that is how we are getting these expressions and therefore the total number of operations is simply you add them up.

That is the total number of operations; you can see that you have again order of n^3, from here itself you can see order of n^3 is coming.

(Refer Slide Time: 20:14)

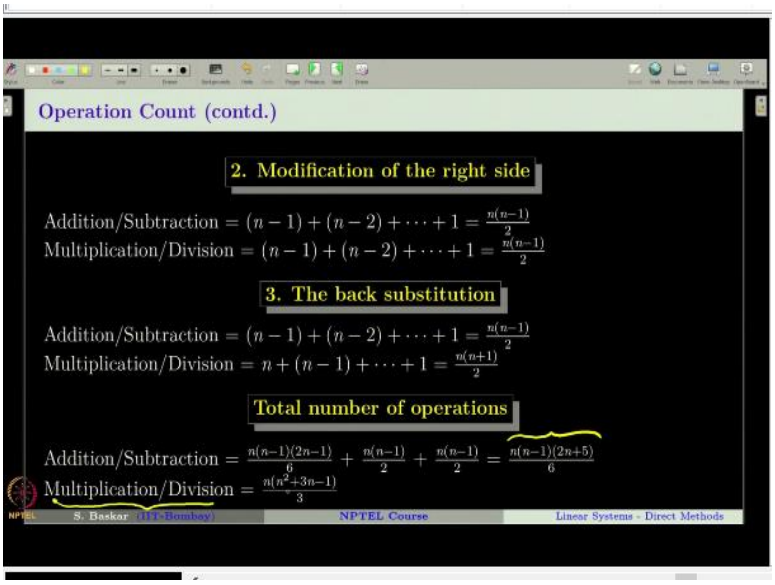


Now let us go to the right hand side elimination. Again you can do the counting in a similar way to see that the number of additions and subtractions involved in the right hand side vector that is to obtain the vector from b to $\tilde{b}$. That is what we have denoted in our first slide, for the system $Ux = \tilde{b}$. For that we need this many operations and how many multiplications.

Similarly you have at each step for each equation you have one addition of subtraction, one multiplication we need. So, the division is already done in the elimination process and when it comes to back substitution you have this many additions or subtractions, I leave it to you to go through the back substitution carefully and count the number of operations involved in getting each component of the vector x.

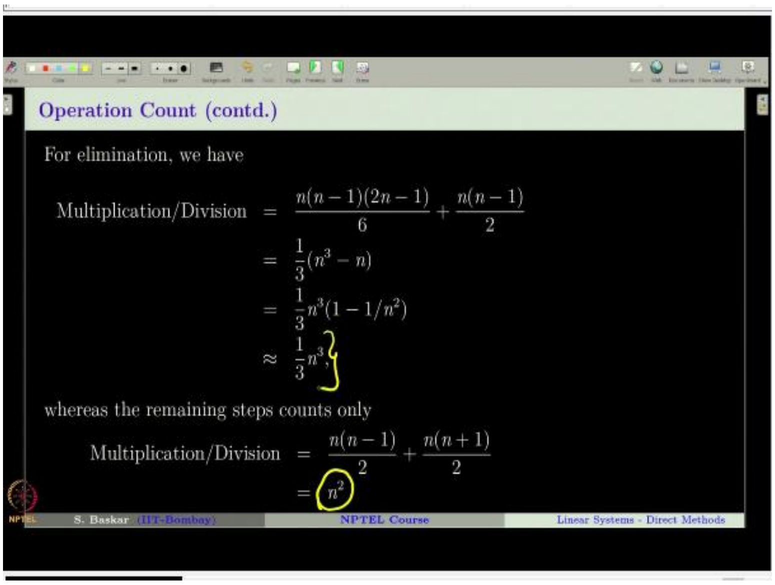
Then you can see that it counts to the number of addition or subtraction as $\frac{n(n-1)}{2}$ and similarly multiplication or division comes to $\frac{n(n-1)}{2}$.

(Refer Slide Time: 21:37)



Now coming to the total number of operations we have to add all the arithmetic operations involved in all the above steps, you can see that the number of additions and subtractions involved in all the above steps finally can be written as $\frac{n(n-1)(2n+5)}{6}$ and also the number of multiplications and divisions involved in all the above operations includes $\frac{n(n^2+3n-1)}{3}$.

(Refer Slide Time: 22:22)



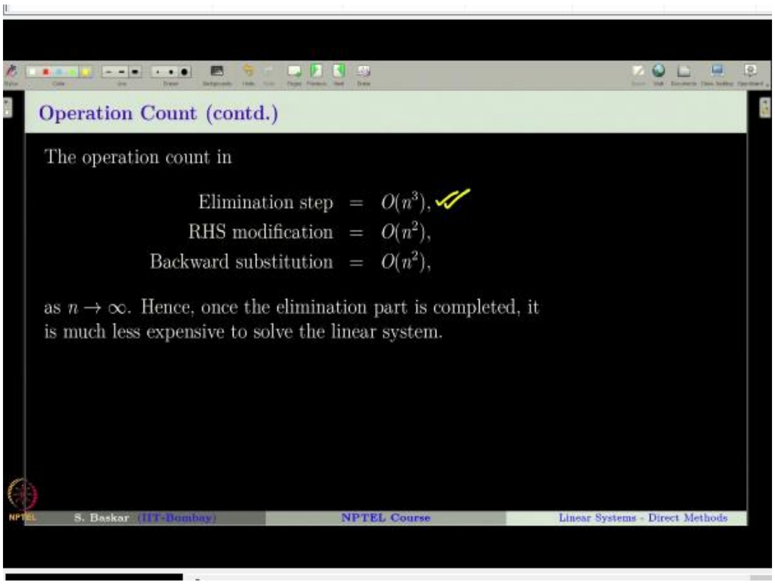
In this let us only consider the elimination part and particularly let us try to see how many multiplications and divisions are involved in the elimination part, you can see that this many multiplications and divisions are involved. We have just now seen this. Let us try to simplify this expression and see how it looks like. It looks like this and you can see from here that this is also of order n^3, just like what we saw in the case of Cholesky's factorization.

In fact you can see that this is approximately equal to $\frac{n^3}{3}$. That is the elimination process includes the number of operations which of order n^3, in particular it includes approximately $\frac{n^3}{3}$ number of multiplications and divisions. You can see that it may be little more or less, but what matters finally to understand how costly a particular method is we have to see what is the leading order in the number of operations.

Here we can see that the leading order is 3 and that is pretty big because generally we work with linear systems of large dimensions something like 1000 or more can be even less in some practical applications. In such cases these methods are pretty costly and of course apart from that you also have other part of the Gaussian elimination method like right hand side elimination and also back substitution.

They also include some operations when you only look at the multiplication or division. They happen to be this much, which is again n^2. But what matters is the leading order that tells us how costly the method is.

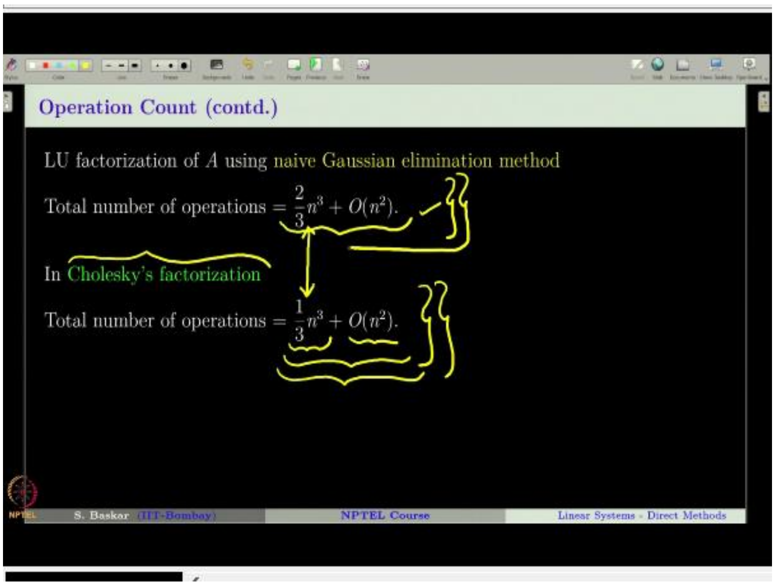
(Refer Slide Time: 24:41)



So, you can clearly see that the Gaussian elimination method has three parts. One is the elimination step which is the costliest part of the Gaussian elimination method which is of order n^3, the right hand side modification is of order n^2 and the back substitution is of order n^2 as $n \to \infty$. Hence the Gaussian elimination method is very costly in particular the elimination path is the most costliest part.

This is why often what people do is they make LU decomposition of the matrix A and in particular if your problem involves solving many linear systems where the matrix A is fixed but you have many right hand side vectors b to compute the solutions then what you can do is you do the LU decomposition once for A and then you use a forward substitution and backward substitution for every given b to get the solution of the linear system $Ax = b$. Let us try to now compare the Gaussian elimination method with the Cholesky's factorization.

(Refer Slide Time: 25:58)



For that we will only consider the elimination part that is the left hand side elimination of the linear system which is equivalent to the LU factorization of the matrix A using the naïve Gaussian elimination method. If you recall the number of multiplications involved in this process is $\frac{1}{3}n^3$ approximately. Similarly, you can also see that the number of additions or subtractions involved in the elimination process amounts to approximately again $\frac{1}{3}n^3$.

Therefore if you add these two we can say that the total number of operations that is multiplication or division and addition or subtraction involved in the LU factorization of the matrix A using the naïve Gaussian elimination method is $\frac{1}{3}n^3 + \frac{1}{3}n^3$. That amounts to $\frac{2}{3}n^3$ number of operations plus of course you have some lower order terms, but at the leading level you have this many arithmetic operations involved in the LU factorization of a using Gaussian elimination method.

Now let us recall from the Cholesky's factorization we had the total number of operations coming to be approximately $\frac{1}{3}n^3$ of course plus some lower order terms. Now the higher order level you can see that Cholesky's factorization needs only half of the effort put to get LU factorization for the matrix A using Gaussian elimination method. In fact you can see that the Doolittle factorization and the Crout's factorization are also equivalent to the Gaussian elimination method.

Therefore they also amount to something equivalent to this many arithmetic operations involved in them. This shows that when you are working with a symmetric and positive definite matrix it is always preferable to go for Cholesky's factorization. Of course it is not surprising for us to see that Cholesky's factorization is little efficient than the other factorizations.

Because Cholesky's factorization uses the symmetric nature of the matrix A and thereby it has to only compute L, but not U explicitly. Whereas the Gaussian elimination method, Doolittle factorization and also Crout's factorization they work for any suitable invertible matrix and not necessarily symmetric and they have to compute both L as well as U explicitly. That is why they involve more computational time than Cholesky's factorization.

Therefore the moral of this class is that whenever you are working with symmetric positive definite matrix you always go for Cholesky's factorization. Remember that this operation count is only for the way we have computed the Cholesky's factorization using the algorithm we introduced in the last class. There are other ways to obtain Cholesky's factorization. For instance one can also obtain Cholesky's factorization from the Doolittle or Crout's factorization.

That may not lead to this kind of efficiency because you will be spending time anyway equivalent to this much of time when you are computing the Doolittle or Crout's factorization. Therefore when you have a symmetric and positive definite matrix you go for the Cholesky's factorization in the methodology that we have introduced in the last class. With this node we complete our discussion on direct methods. Thank you for your attention.