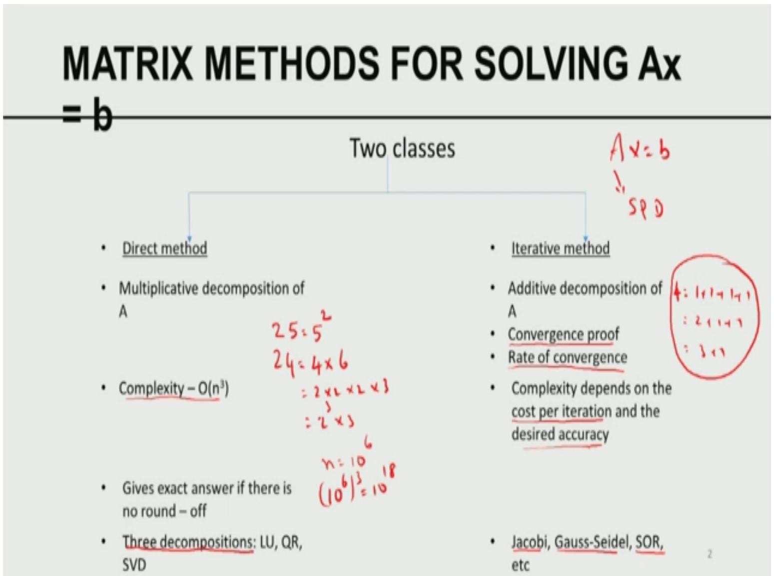
Dynamic Data Assimilation
Prof. S Lakshmivarahan
School of Computer Science
Indian Institute of Technology, Madras

Lecture - 16
Matrix Decomposition Algorithms

In the last module for 4.1, we provided an interlude where we summarized the computational problems arising from the formulation of the least square problems linear non-linear etcetera. And we also indicated some of the methods the two pathways one by matrix methods another by direct (Refer Time: 00:34). So, in this module two, we are going to take up the methods of matrix decomposition or matrix methods. Most of the matrix methods are decomposition based techniques; we are going to primarily talking about three decomposition techniques Choleskey decomposition, QR decomposition and SVD we are going to look at the details of how these are organized.

(Refer Slide Time: 01:00)



Matrix methods for solving A x is equal to b we are interested in solving a specific problem of A x is equal to b. So, in our case we are interested in solving A x is equal to b a is SPD that is what we are interested in. But before I get little deeper into solving systems with symmetric positive matrices I would like to provide a broader overview of the available classes of methods one can use to solve the matrix system.

One is called direct method; another is called the iterative methods this is within the matrix methodology. The direct methods they rely on what is called multiplicative decomposition of A. These methods have in general a complexity of n cube; these methods give exact result when there is no round-off errors in the computers.

In this class, we are going to be looking at three different classes of methods LU, QR and SVD. LU is the forerunner of all the others. So, I am going to start with basic LU, then QR then SVD. Multiplicative decomposition of matrices very similar to multiplicative decomposition of integers; if I have an integer n which is given any positive integer n, there is a fundamental theorem that says I can express it as a product of powers of prime that is called prime decomposition.

Prime decomposition is a multiplicative decomposition within the context of integers. For example, if you have the number 25, 25 is equal to phi square which is a square of the prime if I have the number 24 on the other hand I could express this as four times six two times two times two times three, so this could be 2 cube times 3. 2 is a prime, 3 is a prime. So, this is called the prime decomposition so much like any positive integer can be decomposed into product of powers of prime.

Given a matrix, given a non-singular matrix, I can also express it as product decomposition or multiplicative decomposition. So, what we have been used to doing in numbers I would like to be able to translate it to matrices and that is what these three decomposition methods all entail. These decomposition generally belong to the multiplicative class, but the details of the derivation of the factors differ because the property the factors differ.

On the other hand, the system like A x is equal to b can also be solved by iterative techniques. The iterative methods rely on what is called additive decomposition of A. This iterative method in order to be able to make it operative, we have to indulge in what I call convergence proves, we have to show the method iteratively converges to the solution I am seeking. In any iterative methods one has to be content with what is called the derivation of the rate of convergence. It is one thing to prove that the iterative method converges, another thing to find out what is the rate at which it converges to the optimum. The complexity of this method depends on cost per iteration and the desired accuracy as opposed to the fixed cost for the direct method, which are all of the type o of n cube.

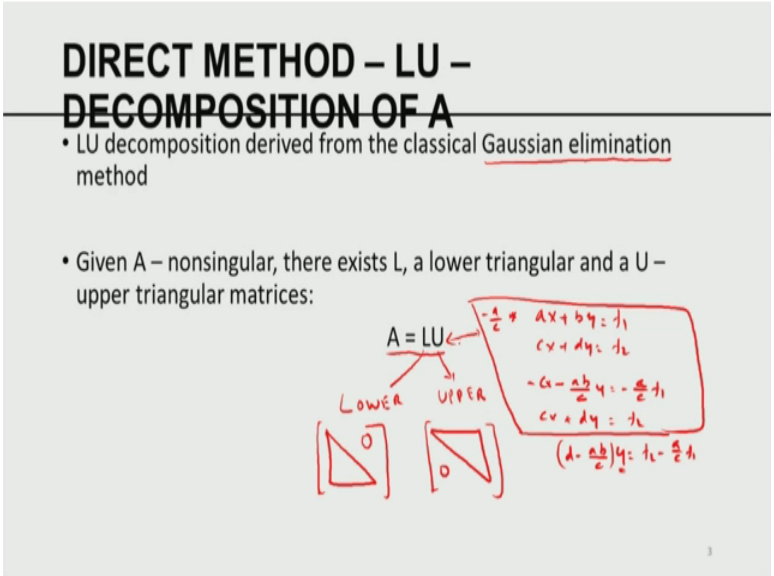
What is O of n cube? If A is a matrix of size n by n, the total amount of work to be done in solving the system A x equal to b is takes a total off of the order of n cube operation. For example, if n is equal to a million, a million size problem is very routine in geophysical domain. If I am interested in solving a static inverse problem, I convert the static inverse problem to one of solving a linear system A x is equal to b. So, A is a matrix of size 10 to the power of 6, the total amount of work to total amount of basic operation.

Basic operations are addition, multiplication, subtraction, division that the computer has to perform to be able to generate is of the order of 10 to the power of 6 cube which is equal to 10 to the power 18. 10 to the power of 18 operations is large amount of work and that is going to take quite a longer long time, we will try to provide an estimate of how long does it take to be able to solve a million by million system as we go by. But at this stage, I would like to be able to concentrate on two mutually exclusive classes of algorithm; one depends on multiplicative decomposition, another depends on additive decomposition.

There is also an analogy with respect to integers additive decomposition for example, if I have number 4, I can express it as 1 plus 1 plus 1 plus 1, I can express it as 2 plus 1 plus 1, I can express it as 3 plus 1. So, these are different ways of expressing four additively. So, these are called additive decomposition much like I can express numbers additively, I should also be able to express matrices additively.

So, multiplicative decomposition of numbers, additive decomposition numbers likewise for matrices these two methods depends on our ability to decompose matrices multiplicatively and additively. Some of the methods of the iterative type are called Jacobi method, Gauss-Seidel method, successive over relaxation methods, these iterative methods are easy to program. The total cost depends on how much it takes for you to be able to perform one iterative step. The number of iterative step depends on the total desired accuracy. So, these two methods are two competing methods one can utilize to be able to develop programs systems for doing data assimilation.

(Refer Slide Time: 07:49)



The LU decomposition method that is our first one, I am going to concentrate some of the major aspects of LU decomposition. It is derived from the classical Gaussian elimination methods that we are all introduced to in the first course in algebra to solve a 2 by 2 system. How do we solve a 2 by 2 system in when we learn algebra? I have a x plus b y is equal to f 1, c x plus d y is equal to f of 2.

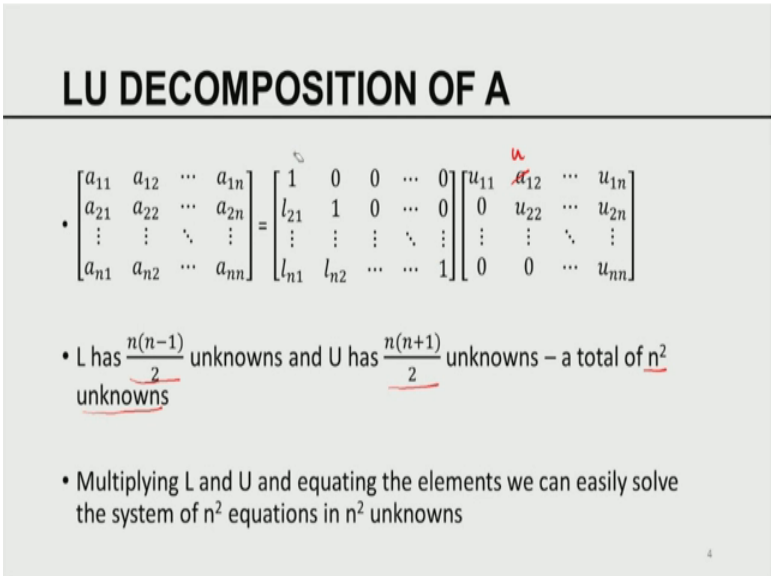
How do we do that, we eliminate x in one of the equations, so by multiplying the first equation by minus a by c we can make the first equation to be minus c minus a b times c y equal to minus c times f 1 I have c x plus d y is equal to f 2. Now, I add these to the first two term gets cancelled then I get d minus a b by c times y is equal to f 2 minus a by c f 1 by dividing I can get y once I get y I can substitute one of these equations and recover x. This is called the method of elimination.

And please remember gauss was the one who invented this method. Please realize now gausses fundamental inventions, he developed the Gaussian distribution to be able to describe observation error errors. He developed the least square methodology to be able to solve the problem in astronomy. He also invented this method of elimination to be able to solve linear system, we are cognizantly otherwise we use many of the results of gauss routinely in all our work. So, this is the method of Gaussian elimination. This method of Gaussian elimination when written in the matrix formulation can be shown to be equivalent to LU decomposition.

So, what is a LU decomposition? Given a matrix A, I can expresses this as a product. So, this is where the multiplicative decomposition comes in, as a product of two matrices where L is a lower triangular matrix and U is an upper triangular matrix. What is the lower triangular matrix? In a lower triangular matrix, this is nonzero everything above is 0; in upper triangular matrix, everything below is 0.

Therefore given any matrix I can express it as a product of two matrices with special structures, the structure lower triangular the structure upper triangular. So, there is a general theorem given any non singular matrix, it can be expressed the product of L and U, where L is lower triangular, U is upper triangular. And this decomposition is mathematically equivalent to the Gaussian elimination method we generally use, we are generally introduced when we first developed tools in algebra.

(Refer Slide Time: 11:19)



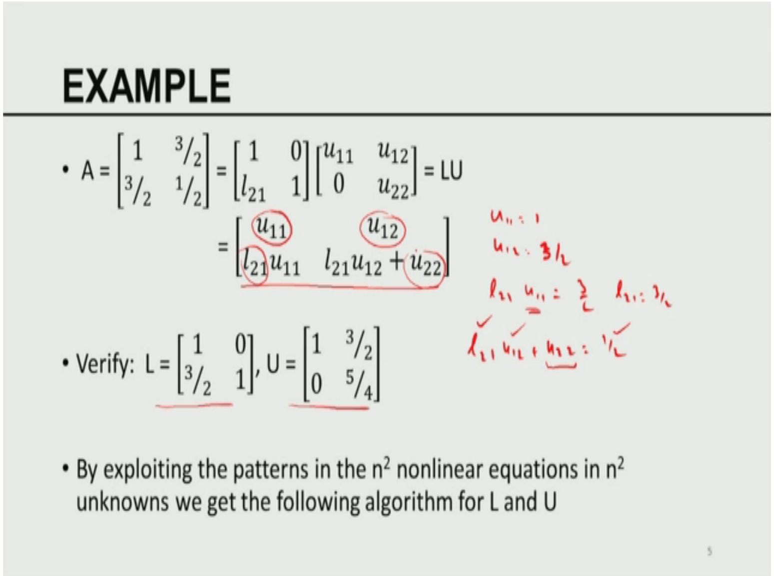
I am not going to show that it can be done; now I am going to approach it using a constructive procedure. One way would be to show such an L and U given a exists. Another way would be I am not going to worry about existence I am simply actually deliver it that will solve the equation A is equal to L times U. So, A is given, let this be the L matrix that let that be the U matrix I am sorry this must be U instead of A. So, I am assuming a particular structure for L, I am assuming a particular structure for U. If you compute the total number of unknowns in the l, matrix they are all below the diagonal if you compute the total number of unknowns in the U matrix anything on and above the diagonal. So, the L matrix has a total of n times n minus n by 2 unknowns. The U matrix has n times n plus 1 by two unknowns these two together consists of a total of n square unknowns.

So, to be able to compute the L and U is equivalent to given the a elements I have to compute the l elements and the u elements. So, if I multiply these two matrix on the right hand side, I am going to get expressions as the product of L and U, I am going to have to equate them to A s. There are n square elements in the left hand matrix; there are n square elements in the right hand product matrix. By equating the two, I am going to get n square equations in n square unknowns; by solving these n square unknowns, I am going to uncover the elements of l and u they uncover the elements of l and u. So, that is the general idea of this methodology.

But before I go further I would like to talk a little bit about the structure. In the case of U matrices I assume the diagonal to be arbitrary u 11 u 22, but I assumed the l to be fixed u 11. I am going to argue suppose I make this l 11, I suppose I make this also l 11, l 22, l n n, the total number of unknowns in the L matrix and the U matrix will be n times n plus 1 by 2. The total number of unknowns will be larger than n square, but there are only n square equations.

Now, please realize how we got here I started with the undetermined system or over determined system, I converted into a linear system A x is equal to b to solve the linear system I again cannot go back to an over determinant system underdetermined system, it becomes circular. So, I have to have a determinant system it turns out without loss of generality I can assume the diagonal factors of l they are all unity they are all unity. So, this is one way of being able to enforce solvability. So, there are n square equations, there are n square unknowns.

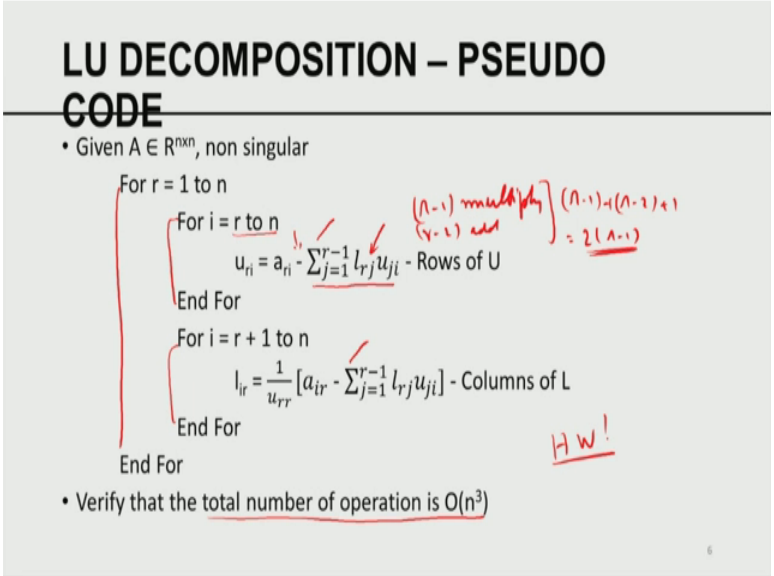
(Refer Slide Time: 14:38)



We will illustrate this idea by a very simple example. Suppose, I am given, so I am now going to take an example of a symmetric matrix. In general, the above decomposition holds good for any matrix, but for reasons that we are interested in symmetric matrix I am going to start with the simple matrix. Let A be the symmetric matrix; let L be this; U be this l and u follow the conditions that we have already stated if you multiply L and U, I get a matrix which is given by this. Now, I have to equate u 1 1. So, from here you already get u 1 1 is equal to 1, you also get u 1 2 is equal to 3 by 2. Now, you now have l 2 1 u 1 1 is equal to 3 by 2, but I already know u 1 1 is 1, therefore, l 2 1 also becomes l 2 1 becomes 3 by 2.

Now, if you consider the equation the last equation l 2 1 u 1 2 plus u 2 2 is equal to one half. I already know l 2 1, I already know u 1 2, I already know the right hand side is half. So, using that I can determine u 2 2. So, in this particular way, I have determined L is given by this, U is given by this. So, now, look at this now once u 1 1 is known, I can compute l 2 1, 1 u 1 2 is known I can compute u 2 2. So, there is a particular structure with which we can uncover the unknown elements. This structure is embodied in the LU decomposition algorithm.

(Refer Slide Time: 16:28)



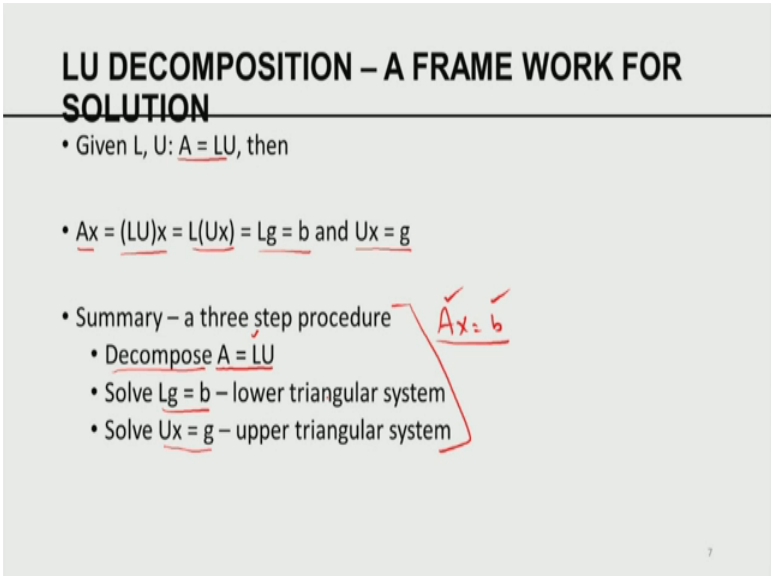
I am providing a pseudo code I am sure you can read this pseudo code, I do not think I need to I want to repeat the instructions in the pseudo code. You can readily see there is a for loop. There is a for loop, and there is one loop, another loop at the same level. There is a third loop which I did not write in, but there is an intrinsic third loop, there is a summation, there is summation, you cannot simply write in a computer programming sum you have to write it as a do loop.

So, you can see there is a triple nested do loop there is a triple nested do loop. When there is a triple nested do loop the overall cost is n cube that is where the total cost of n cube comes into play. It can be verified that the total number of operations to be performed is the order of n cube and I would leave it to you to verify this. For example, in this particular case, I am going to have to run r j is equal to 1 to r minus 1 it is summation, but before a sum I have to multiply.

So, there are r minus 1 multiply that will give me r minus 1 numbers then I have to add them all to add two numbers I have to make one addition, I have to make two additions; to add r minus 1 numbers I have to perform r minus 2 additions. And I have to perform one more addition subtraction, subtraction addition are essentially the same. Therefore, this particular step is going to require r minus 1 plus r minus 2 plus 1 that is going to be equal to 2 times r minus 1 operations, but r runs from 1 to n. Therefore, this particular do loop alone is going to now require 2 times r minus 1 r runs from 1 to n. I would like to revisit this issue once again, that is correct good.

And this is this one is embedded in this do loop. If it is embedded in this do loop this is repeated r running from 1 to n sorry that repeated r running from 1 to n, r to n is n minus r. So, this has to be done n minus r times where r changing. So, likewise you can compute the operations in here, then you have to compute the operation overall. So, if you add up all these expressions, you can verify the total amount of operation is the order of n cube. And I would like to leave this as a homework problem for you to compute. Please do and convince yourself that to solve any LU decomposition problem it requires n cube operations.

(Refer Slide Time: 19:38)

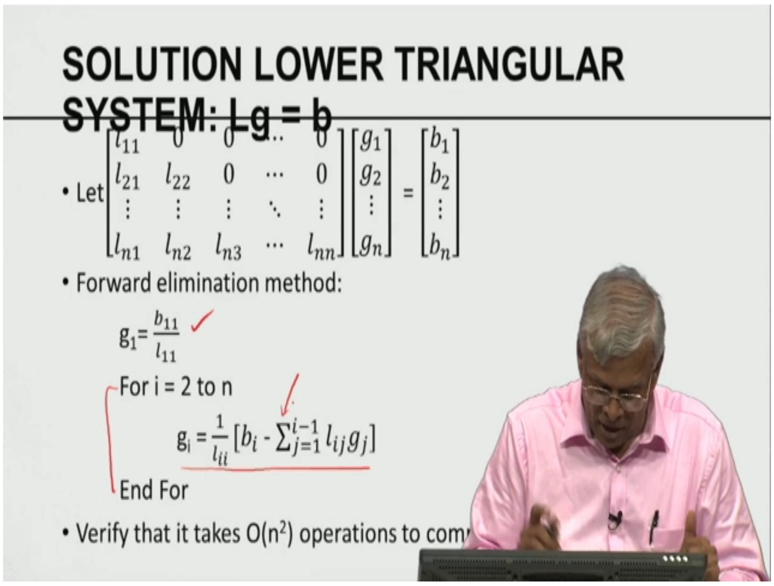


Now, the question is once I have computed l and u how what do I do with it. Let us go back. A is equal to LU, therefore, A x is equal to LU x LU x can be written as L times U of x. Now, I can write U of x is equal to g if U of x is equal to g then this becomes L g is equal to b. So, A x is equal to b reduces to L g is equal to b. And then trunk of reacts I have to solve U x is equal to g, therefore the LU decomposition framework essentially gives you first decomposed A is equal to LU. Then you have been given see you have been given A and b, use A find L and U. Then using the L, you found in the previous step and the b solve for g solve for lower triangular system.

You already know U you already know g, from the previous step solve U x is equal to x. So, that is how you solve in three steps LU decomposition step solution of lower triangular system, solution of upper triangular system, these three together constitute the method of LU decomposition and this method is applicable to any matrix so long as a is non singular. So, this is the mother of all direct methods LU decomposition the basis this is it is from here all the other methods start.

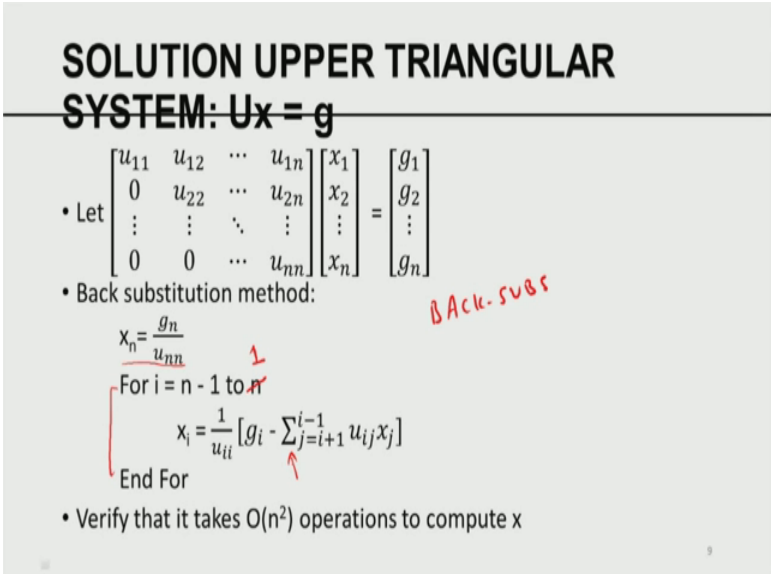
Now, I am going to talk about method solving the lower triangular system. So, please remember LU decomposition algorithm we have already seen how to decompose, please remember we have already seen how to decompose, I have given a pseudo code. Now, I want to be able to solve a lower triangular system, but before I go further I want to tell solving a lower triangular system and solving a low upper triangular system are essentially similar mathematically. Why is that? The transpose of a lower triangle has upper triangular and transpose of upper triangles lower triangular. Therefore, I could once I know how to solve a lower triangular system; I also know how to solve an upper triangular system. So, I do not have to deal with these separately.

(Refer Slide Time: 21:59)



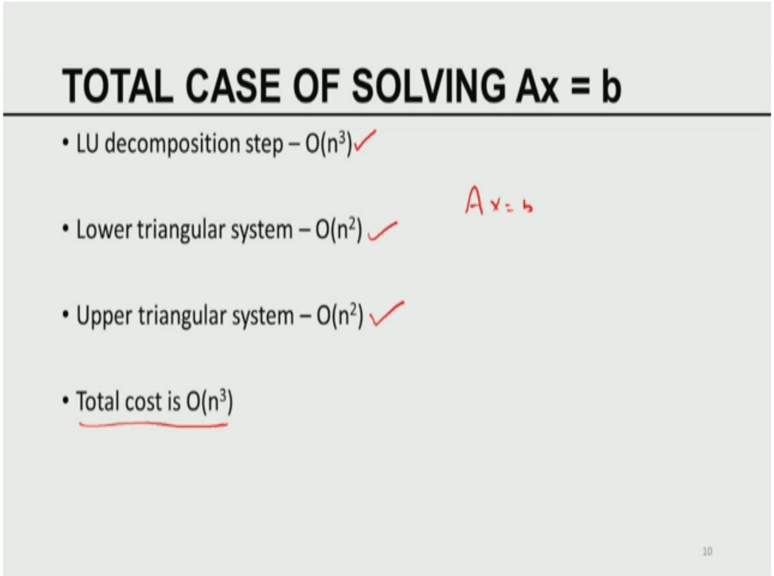
So, suffice to say that I would like to be able to solve a lower triangular system. So, l 1 1 0 0 l 2 1 l 2 2 0 0 g 1 g 2 g n b 1 b 2 b n I can recover please remember from the first equation I i can recover g one then for I running from two to n I can recover any of the other g is by this simple formula. So, using this loop and another embedded loop for the summation I can essentially solve the system it can again be verified the total operation that is required is only O n square please remember LU requires O n cube, but solving a lower triangular system is much cheaper is only O n square n square does not grow as fast as n cube therefore, solving lower triangular system is much cheaper

(Refer Slide Time: 22:55)



For sake of completeness, I am also giving you an algorithm for upper triangular system this is a typical upper triangular system, x the computer g from the previous cases. So, I can recover x n first, I can compute x n first. So, x n is given by this then x n minus 1 to 1 I am sorry this must be 1 that is a typo. So, I could be able to recover all the x i's by using what is called back substitution this method is called back substitution. So, by using a method of back substitution, I have a formula which is very similar to a do loop there is another embedded do loop because of the summation sign. So, these two together requires O n square operations.

(Refer Slide Time: 23:48)

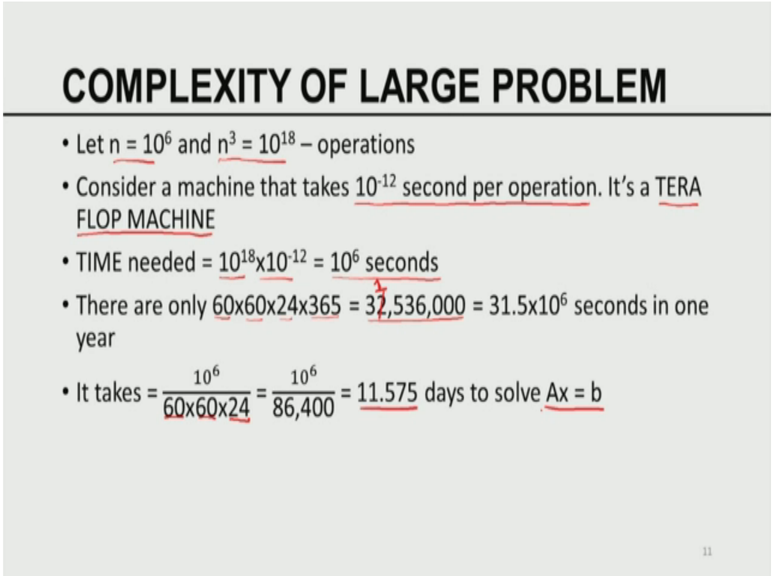


So, in summary if I have if I am going to solve A x is equal to b using LU decomposition the LU decomposition steps n cube lower triangular system takes n square upper triangular system takes n square of all the three O n cube is a dominant term. So, the total cost is the order of n cube that is where the whole thing comes into play. We can actually compute these are all actually polynomials in n, you can actually compute the actual polynomial that gives you the amount of work and that is a homework problem I am leaving it to the reader to fill up.

So, this essentially provides you I have provided the pseudo code for both the LU decomposition lower triangular upper triangular. So, you have a code you can code it in your favorite language C, C plus plus, fortran, matlab whatever it is; in fact, matlab has excellent programs written. While you may use readily matlab programs I think is better to understand the intelligence behind how matlab solves the problem it does to be able to have a total understanding of the programs that you may develop.

In fact, if you are trying to develop a data assimilation system far work in operational centers they generally do not depend on anybody they try to code everything ground up because they will have total control over how things are happening. So, in such cases if you are interested in developing large-scale systems, one need to know the nuts and bolts of how these algorithms are implemented to solve the least square problems.

(Refer Slide Time: 25:27)



Now, I am going to give you an indication of the time involved. I think this will be a very interactive exercise. Let us assume n is a million, n cube is equal to 10 to the power of 8 operations. I had to perform these operations in a machine. Let us pretend I have the best machine money can buy. So, I have a machine that takes 10 to the power of 12 seconds per operation, such a machine is called teraflop machines.

I want to quickly add there are not too many teraflop machines on the face of the earth right now. These are some they all talk about petaflop machine, petaflop machine is each operation takes 10 to the power of minus 15, but over the across the world there are only four five six petaflop machines at very special places, teraflop machines are much more popular. So, a teraflop machine let us pretend we have access to that.

So, if each operation takes 10 to the power of minus 12 seconds, I had to perform 10 to the power of 18 operations. So, I would like to. So, you do take 10 to the power of six seconds to solve the problem 10 to the power of 6 seconds is a million seconds. Now, let me estimate how much is a million second; in a day, there are 60 seconds in a minute; there are 60 minutes in an hour, there are 24 hours a day, there are 365 days in a year ordinary year excluding the leap years. So, there are only roughly this is 31 not 32, this is 31. There are roughly 31.5 million seconds in 1 year. So, I want 10 to the power of 6 seconds there are only 31.5 times 10 to the power of 6 seconds in a year.

So, how many years or how many days does it take? So, this is the total amount of seconds 60 seconds in an hour, I am sorry 60 seconds in a minute, 60 minutes in an hour and 24 hours, so 10 to the power of 6 divided by this many hours. So, I am sorry there are 6,let me start all over again there are 60 seconds in a minute, there are 60 minutes in an hour, there are 24 hours a day.

So, this fraction gives you the total number of days to complete 10 to the power of 6 operation all the operation 10 to the power of 18 operations on this machine and that is equivalent to 11.57 days to solve A x is equal to b if n is a million. This is conditioned on condition of the fact that I have a teraflop machines. If you do not have a teraflop machine the story is much different.

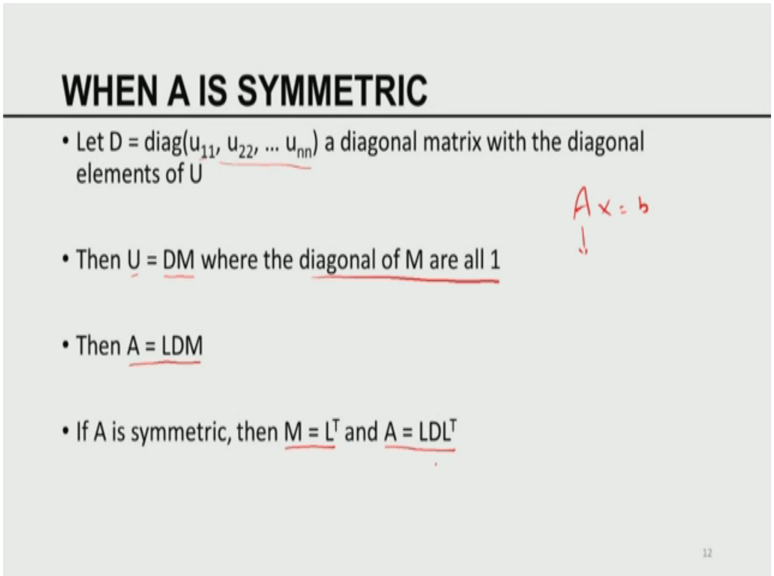
Now. I want to ask was ah a following hypothetical question, would you wait for eleven and a half day is to solve one problem, that is totally impractical. We have to create forecast especially in the meteorological contacts in atmospheric science every 24 hours in order to be able to make a forecast every 24 hours the data assimilation person may not get more than 5 hours, 6 hours, 4 hours. So, observations have to be collected, observations have to be made ready; the models have to be run the data assimilation part has to be established. Once the data assimilation part is done inverse problems are solve then one has to generate forecast. Here what we are talking about is solving one part of the data assimilation problem namely to solve A x is equal to b is going to take of the order of eleven and half days.

So, now you can see the monster. The monster is not because we do not know how to solve the problem, we know how to solve problem exactly, the monster nature of the problem because it comes from the size of the problems, sheer size 10 to the power of 6. If you talk to meteorologist, if you talk to oceanographers, 10 to the power of 6 is not all that bad they would like to be able to refine the grid much smaller grid resolutions. So, if you want to be able to improve the accuracy and predictions of the model.

On one hand you have to reduce the grid spacing reducing the grid space increase the size. If I increase the size, the problem becomes larger; if the problem becomes larger my computers are not enough to be able to solve the problem in a time that is allowed for me to be able to generate prediction. So, this is the dilemma that are faced world over by all the meteorological operation research centers.

So, what is the solution, what is the way out one way would be to reduce the size of the problem would not. To reduce the size that problem means what you make the problem course if you make the problem course there are more model errors or you buy the best machine money can buy, but there are no machines faster than peta flop these days. So, these kinds of problems provide impetus for the growth of ever faster computers tera to peta to exa flop machines. So, until faster machines, faster and faster machines come into being, we may have to content ourselves solving only your smaller sized problem because of the constraint of time within which they are allowed to operate, so that is the end result of these of this analysis the computational analysis.

(Refer Slide Time: 31:03)

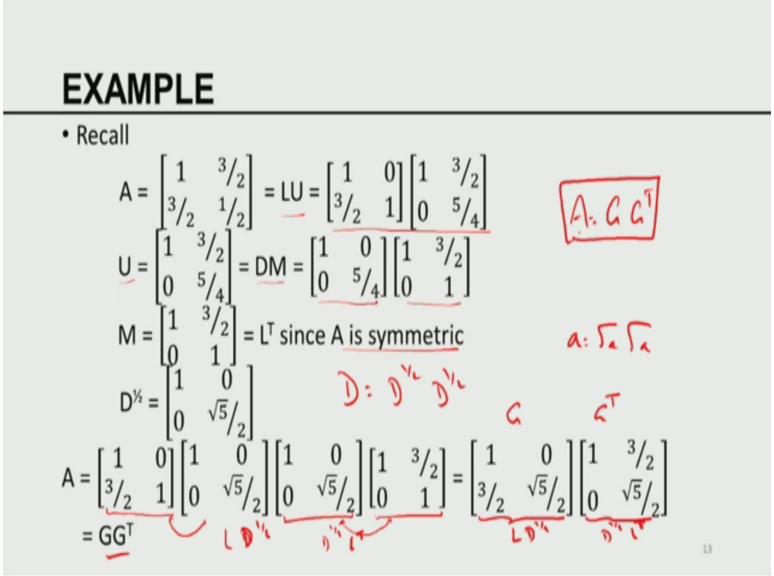


But then A is symmetric, now I am going to go over to having discussed the solution of A x is equal to b for a general matrix. Now, I would like to transcend slowly to the case of the matrix that we are interested in, I would like to be able to solve A x is equal to b when A is symmetric general to symmetric from symmetric to positive definite, so that is the stage we are going to utilize. So, if the matrix is symmetric, now I am going to concoct a diagonal matrix with the diagonal elements of u as a diagonal matrix b in that case I should be able to express u as a product of D and M. In this case, it can be easily verified is a simple matrix calculation M is a matrix whose diagonals are all one.

So, I should be able to use this in my LU decomposition to express A is equal to LDM, L is a lower triangular matrix with all ones along the diagonal, M is an upper triangular matrix with all one along the diagonal. D is a diagonal matrix whose diagonal elements were part of U I have separated that is a further decomposition here. Now, if A is symmetric it can be verified M is the L transpose; M and L are not distinct, therefore I should be able to express a as LD L transpose decomposition. So, this is a special form of the LU decomposition that A takes when A is symmetric.

So, in this case, if I compute L, I do not know in the case of LU, I have to compute L and U separately because L is not equal to U, but in this case u has been replaced by L transpose if I compute L, I already know L transpose. So, half the work is saved. So, all I need to do is compute L and compute D. So, when the matrix A is symmetric, I am saving a kind of money by not having to compute two matrices because of this simple form A is equal to LD L transpose.

(Refer Slide Time: 33:29)



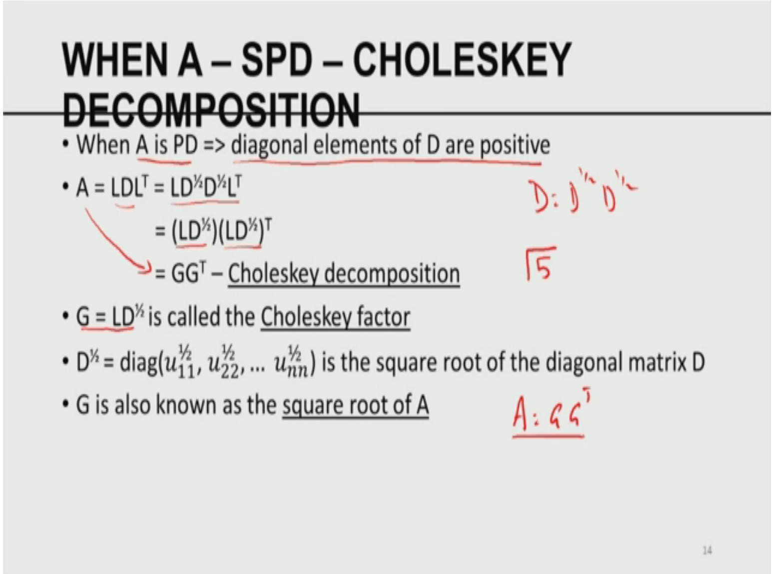
Again I am going to give you a simple example is a symmetric matrix it is instructive to go through these simple examples. I have already shown LU is given by this my U elements are 1 and 5 by 4. So, U can be expressed as L this U can be expressed as LM this is D, this is M. So, M is given by this and M and L are essentially transposes of each other since A is symmetric, and D the diagonal elements of D are positive if the diagonal elements are D are positive. I can take what is called the square root of D the square root of D, I can express this as the square root of a diagonal matrix is simply a matrix whose diagonal elements are square root of the corresponding diagonal elements. So, the diagonal elements are 1 square root of one is 1, square root of 5 by 4 is square root of 5 by 2. So, this is the square root matrix.

So, I can express D as D to the power half times D to the power half much like I can express any number a as square root of a times square root of a. So, what is that I have now done, I have identified L, I have identified U, I had identified D, I have identified M. I have shown M is L transpose further, I took the square root of D, D can be expressed as D to the power half D to the power half the power of the square root.

So, answer the product of the square roots therefore, A can be replaced by L D to the power of half D to the power of half L transpose. I can combine these two parts I can combine these two parts. So, this part is essentially LD to the power half I am sorry I would like to write it clearly. This part is essentially LD to the power half; this part is D to the power half L transpose D to the power of half L transpose. So, this element is L D to the power half, this element is D to the power half L transpose, I am sorry L transpose. I am going to call this as G, I am going to call this as G transpose therefore, this A is equal to G, G transpose.

Now, look at this now this is called the Choleskey factor, this is called the Choleskey decomposition, A is equal to G G transpose, where G is called the Choleskey factor. And I have shown you through using this example how to compute the Choleskey factor. So, in the case of Choleskey factor, there is no L, there is no U, there is only G. So, once you find G, I compute G transpose very readily. So, they are essentially the amount of work is reduced to half. Therefore, Choleskey decomposition the cost of it is about half of what it takes for LU decomposition. Of course, for large problems even this is going to be large, but we are trying to see how various decomposition methods are related to each other. So, we have talked about.

(Refer Slide Time: 37:09)

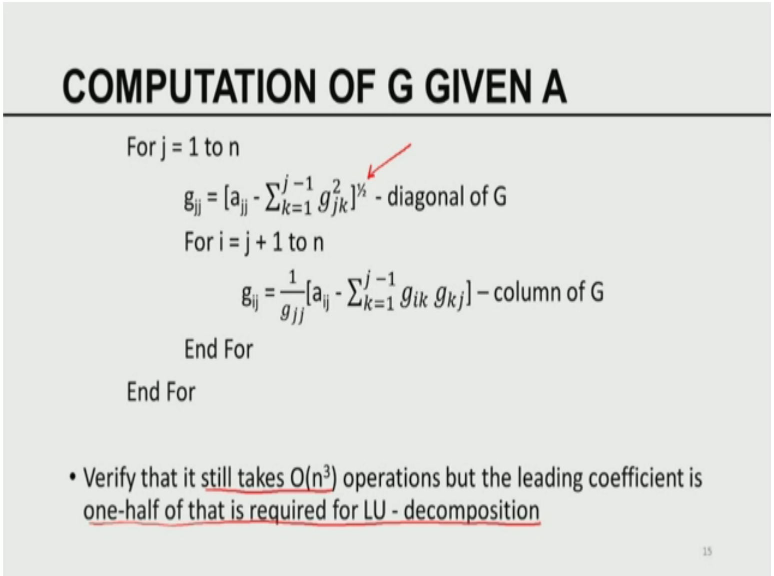


So, I am now going to summarize the whole thing. So, let A be a positive definite matrix, A be a positive definite matrix the diagonal elements of D are positive. If the diagonal elements are D are positive, I can afford to take the square root; in that case I can express D is equal to D to the power of half times D to the power of half. So, L D L becomes the L D to the power of half D to the power half L transpose which I can associate as the product of LD to the power of half LD to the power half transpose, you can readily see the transpose of the diagonal is the same. So, I can now define G is equal to L D to the power of half that is called the Choleskey factor in which case a becomes G G transpose, so that is called the Choleskey decomposition. The diagonal elements are given by u 1 1 to the power half u 2 2 power half u n n to the power half and that is the square root of the diagonal matrix.

In some circles, the matrix G that we compute they also call it as square root of A. So, if I can talk about square root of a diagonal matrix, if I can talk about a square root of a number, I should also be able to talk about the square root of a matrix, but here comes the difference. When you take numbers square root of a positive number, so if A is equal to 5, square root of 5.

If I consider a minus 5 square root of minus 5 is complex numbers. So, square root operation if you want to remain within the real world, square root is defined only for positive numbers. Likewise, if you want to be able to define square roots of matrices, the matrices have to be positive definite. So, square root of a positive number square root of a positive definite matrix Choleskey factor, Choleskey factor also being called the square root of A. So, if I say A is equal to G G transpose I call G Choleskey factor I call G the square root of A both the names simultaneously apply. Different people use different characterizations for G, but with the end result is this is a multiplicative decomposition.

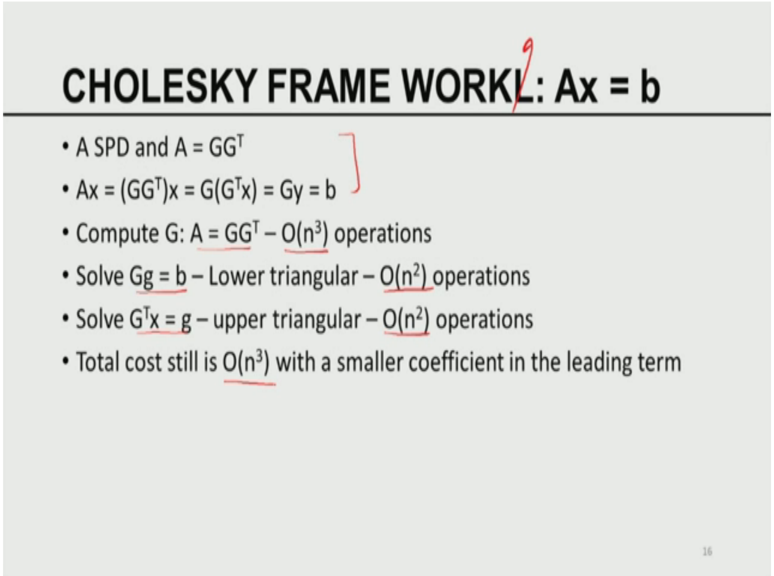
(Refer Slide Time: 39:29)



I am now expressing the computation of G in the form of a pseudo code, again you can readily follow, the only difference between LU decomposition and Choleskey decomposition is that in LU decomposition, there is no square root operation. In the case of Choleskey up there is a square root operation, square root operations are tricky. Square root is not a basic operation to be able to take a square root of a given number; it may take lot more a time. So, this is called Choleskey decomposition with square root operation.

So, I have to be able to perform addition, multiplication, subtraction, division and square root. So, if I consider that we still need to do O of n cube operations, but the leading coefficients of the polynomial that represent amount of work is about one half of that required for LU decomposition, therefore Choleskey decomposition of symmetric positive definite matrices are cheaper than LU decomposition for any general matrix.

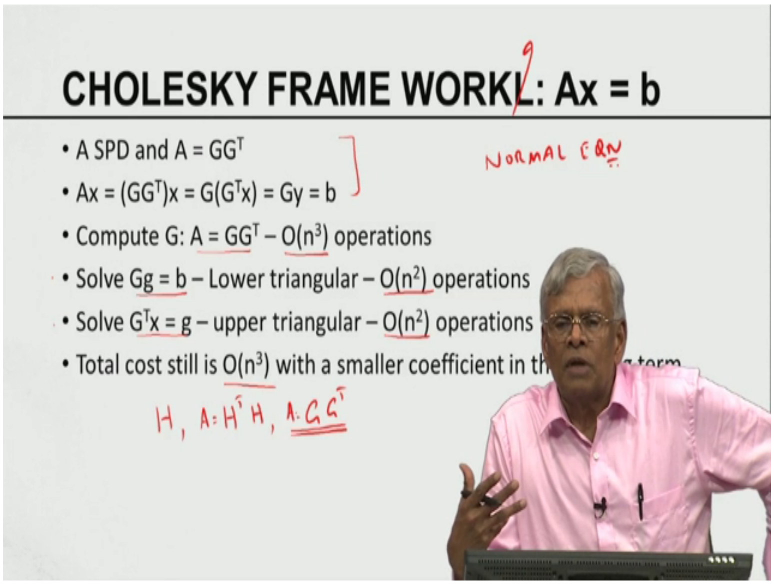
(Refer Slide Time: 40:39)



So, now I am going to talk about the framework. This is framework I think I do not think that is there as a L there. So, let has SPD, G is equal to G G transpose A x is equal to G G transpose x, G times G transpose x, I am going to call G transpose x as y. So, G y is equal to b. So, given this framework, I have a three step algorithm compute g then solve the lower triangular system G G is equal to b and then solve an upper triangular system G G transpose x is equal to g.

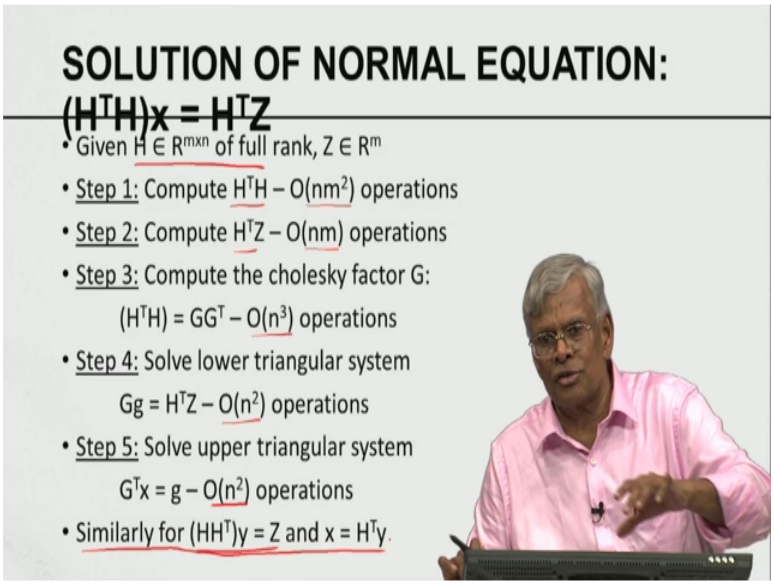
So, you can see the second depends on the first step the third step depends on the first step and the second step. So, we solve the system in again a three step procedure quite parallel to the LU decomposition, but at a lesser cost. The total cost of lower triangular system is n square upper triangular system n square, this is n cube. The overall cost is still n cube, but the smaller coefficient for the leading term. So, it is slightly cheaper. So, if you look into matlab for solving normal equations, if you apply Choleskey decomposition.

(Refer Slide Time: 42:10)



So, what is that that is one has to do one has to do the following you have H you compute H transpose H you call it A, then you split A is equal to G G transpose. Once G G transpose is computed, you can use the lower triangular system upper triangular system to solve the resulting linear least square problems. This method of solving the linear least square problem using Choleskey decomposition is a fundamental and a basic tool and that method has come to be called method of normal equations method of normal equations. So, we have this provides you the algorithmic setup by which we can solve the linear systems that linear least square problem give rise to in our analysis.

(Refer Slide Time: 43:06)

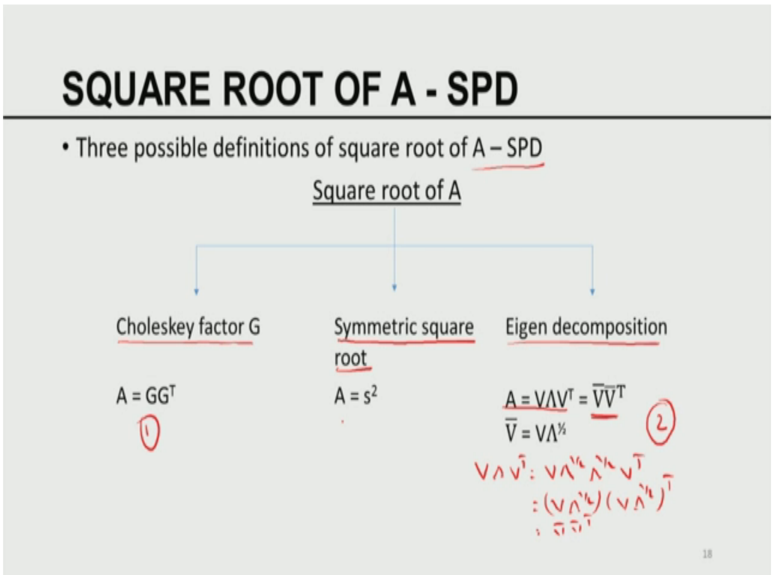


So, I am not going to summarize the method of normal equations. Let H be a full rank matrix. So, this is the overall summary. Let H be a full rank matrix. Compute H H transpose that is going to take an operation n m square compute H transpose Z that is going to take n m operations. Compute the Choleskey factors of G that is going to take n cube operations. Solve the resulting lower triangular system n square; solve the upper triangular system n square. So, you get to solve you get to solve the overall solution for the linear least square problem this is for the over determined system. For under determined system, we can again solve by the same procedure that is involved in here.

So, with this we are completed, we have provided a complete story starting with the formulation of the linear least square problem by converting into minimizing the square of the sum of the residual. Computing the gradient equating the gradient to 0 leading to solution of symmetric positive definite system and a given symmetric positive definite system can be solved by Choleskey.

So, we have provided a complete path from formulation to analysis to algorithms to computational complexity to pseudo program and be able to deliver the result. So, this is the pathway be a complete pathway that exactly what happens when they say I have developed a data assimilation system for use in practical applications.

(Refer Slide Time: 44:53)



I would like to now take a few minutes to be able to discuss the notion of square roots of a matrix. Square root of a number we know, square root of a positive number is real, square root of a negative number is complex. In the case of matrices, there are three possible ways to define the concept of a square root mathematically consistent way, one is by Choleskey factor that is one way. Please understand these definitions are manmade, you can define anything you want so long as you are consistent. So, it looks as though it is A is equal to G square even though G G transpose comes in, you can think of G transpose you can if you forget transfers for a moment it looks like G is equal to A is equal to G times G, so G square. So, G is a square root of A in that sense Choleskey factor differential square root.

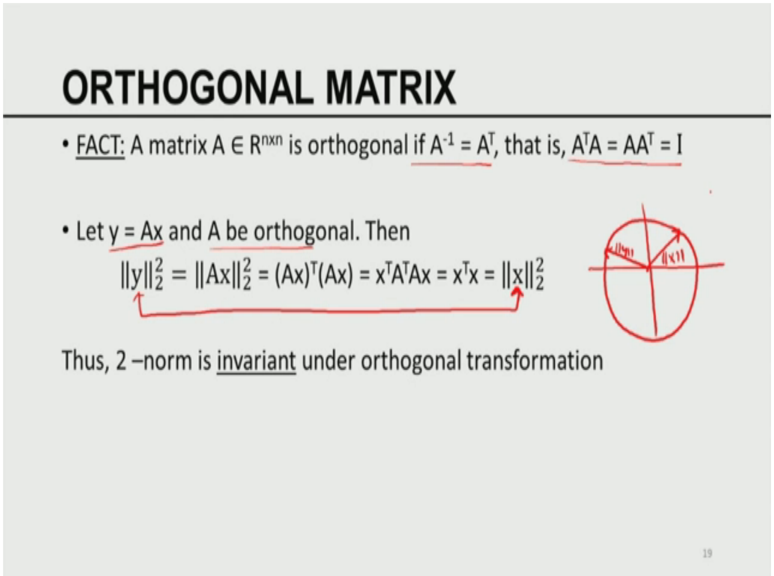
Secondly, we can express a as a product of a symmetric matrix x time symmetric matrix. So, A is equal to x s square you can parameterize the elements of s. You can equate the elements of s square of the A and solve for the elements of s and that is also possible mathematically even though I am not going to show the procedure. The procedure is not too different from the LU decomposition. There we assume the L and elements of L and elements of U are unknown. Here I am going to assume the elements of s are unknown when the elements are s are unknown, you multiply s square. The elements of s square are functions of the elements of s element you equate the corresponding elements you solve for the elements of s you get what is called symmetric square root.

So, what is the difference between the Choleskey square root and the symmetric square root, in the case of Choleskey the square root is a lower triangular matrix G is a square root lower triangular matrix. In the case of symmetric matrix s is a full matrix, it does not have any special structure, but it is a symmetric square root, s is a symmetric matrix, we require it to be a symmetric matrix it has an upper half, it is the lower half essentially they are the same because asymmetric symmetric square root. So, I can compute the symmetric square root that is another way to define.

A third kind of square root also comes from Eigen decomposition from the module on matrices we have already seen any matrix can be expressed the can be expressed as the product of V lambda V transpose this is called Eigen decomposition we are assuming a is SPD. Any SPD using Eigen value decomposition can be expressed this way that is called Eigen decomposition. So, V lambda V transpose is equal to V lambda to the power half lambda to the power half V transpose this is equal to V lambda to the power half times v lambda to the power half transpose that is equal to V bar. V bar transpose that is what we have. So, here again looks like the Choleskey factors, so V bar is considered to be a square root of A. So, this square root is given by the Eigen decomposition.

So, now, if you say a square root of matrix, there are three different ways of computing square roots you have to essentially specify the method by which you compute the square root. In data assimilation the Choleskey base square root as well as the Eigen decomposition base square roots are very popular, we sell them use the symmetric square root, but it is mathematically possible. So, from square root of a number to a square root of a matrix, Choleskey decomposition square root of matrices different ways of defining square roots in a consistent way and we can utilize these square roots to our benefit when we do analysis with respect to matrix algorithms.

(Refer Slide Time: 49:00)



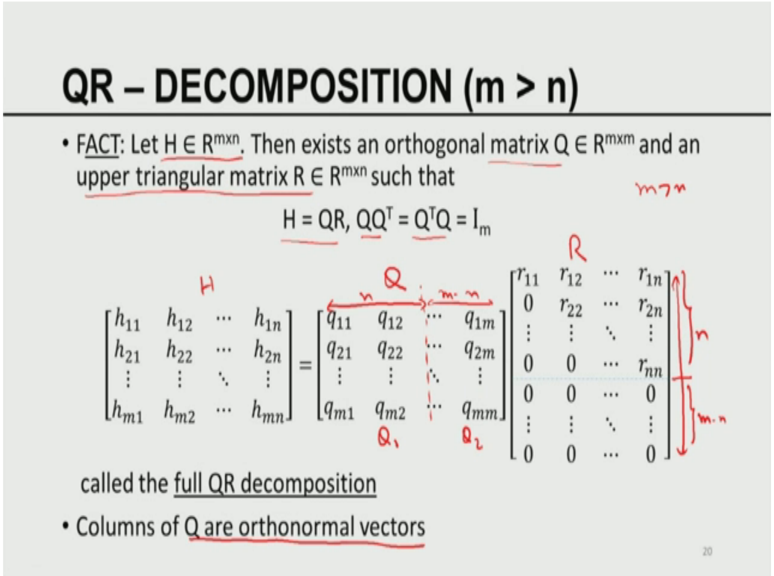
That is a summary of one class of matrix techniques that is based on Choleskey decomposition, which is essentially derived from LU decomposition. Now, I would like to slowly move into the next decomposition method that is called the QR decomposition; for that I need to have some preliminaries. So, I am going to recall some of the definitions from matrix theory. Let A be a matrix of size n by n. We say A as the matrix is an orthogonal matrix if the inverse is transpose; that means, A transpose A or A A transpose an identity. These orthogonal matrices are very powerful and they have very special property.

One's property of orthogonal matrix we are going to illustrate here. So, let A be orthogonal matrix, let x be any vector if you multiply a vector be matrix, I get another vector. So, y is a vector which is an orthogonal transformation of the vector x using the orthogonal matrix A. I want to be able to compute the square of the norm that the square of the 2-norm of y the square of the 2-norm of y is square of the 2-norm of A x. The 2-norm by definition is A x transpose times A x A x transpose is x transpose A transpose A times x A transpose A, if A is orthogonal is identity is equal to x transpose x which is equal to square of the norm of x.

So, look at this now. I start with a vector y I linearly transform y to A these two vectors happen to have the same length; that means, what does it mean this means that 2-norm is invariant under orthogonal transformation. So, what does this imply? If two vectors have the same norm means they lie on the same circle with radius norm of x. So, if y is another vector, y also lies on the same circle with center as origin radius as norm of x, this is equal to norm of y and we simply first saw norm of y is equal to norm of x.

So, what does this mean if you have a vector x if I multiply the vector x by orthogonal transformation, the orthogonal transformation simply rotates it without along aiding it without shrinking it. The length remains the same it simply rotates it. So, y is simply a rotation of x that is the fundamental property that I would like to be able to emphasize at this moment that 2-norm is invariant under orthogonal transformation. This is going to be very useful in the development of QR algorithms.

(Refer Slide Time: 52:14)



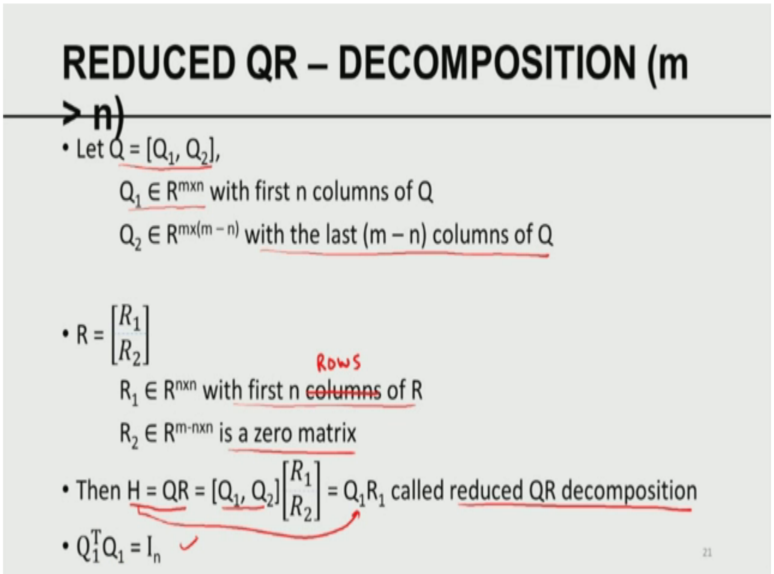
So, QR decomposition algorithm I am also going to consider the case when m is greater than n for simplicity. So, let H be a matrix, well let H be a matrix R m n. Then there is a fundamental theorem that says there exists an orthogonal matrix Q and an upper triangular matrix R such that H is equal to QR. Q Q transpose Q transpose Q is I m that because it is an orthogonal matrix, R is an upper triangular matrix. So, I am going to express that in notation. This is the matrix H, this is the matrix Q, this is the matrix R, R is upper triangular. H is m by n; Q is m by m; and R is m by n. So, R is a rectangular matrix, H is a rectangular matrix, but Q is the square matrix. So, this is called full QR decomposition.

Now look at that now. LU decomposition is for square matrices A is equal to LU. QR decomposition applies to even general matrices namely rectangular matrices. So, it is much more powerful is a generalization of sorts in one way. So, what is the idea, why to form, so you start with H, you compute H transpose H and then decomposed. Why do you create A and then decompose why not you decompose H itself directly that is idea H is the m by n matrix. So, this m by n matrix can be expressed as a product of two matrices with special structures, this is orthogonal that is upper triangular orthogonal matrix have special properties upper triangular matrices are very simple matrices structurally simpler matrices.

What do I mean by saying orthogonal matrices, in an orthogonal matrix, the columns of Q are orthonormal vectors. What do you mean by orthonormal, the length of each vector is one if I pick any two vectors and do an inner product that is zero that means, any two columns are mutually orthogonal every vector is of length one such a collection of vectors is called orthonormal vectors. So, this is a very special form of decomposition. It is also generalization of the LU decomposition. So, there are lots of beautiful properties from going from LU to Choleskey to QR.

Now, I would like to go back m is larger than n. So, this is an over determined system m is larger than m, m is larger than n. So, there are m rows, I am going to cut it at the n rows. So, this is n, this is m minus n. So, if I am going to partition the R like this, I should be able to partition the Q also like this n, I am going to have to this is m minus n. So, this part with the n columns, I am going to call it Q 1; this part with the n the m minus n columns I am going to call it Q 2. So, Q 1, Q 2 is the partition of Q, R and 0, R parties. So, this lower part is essentially a zero matrix.

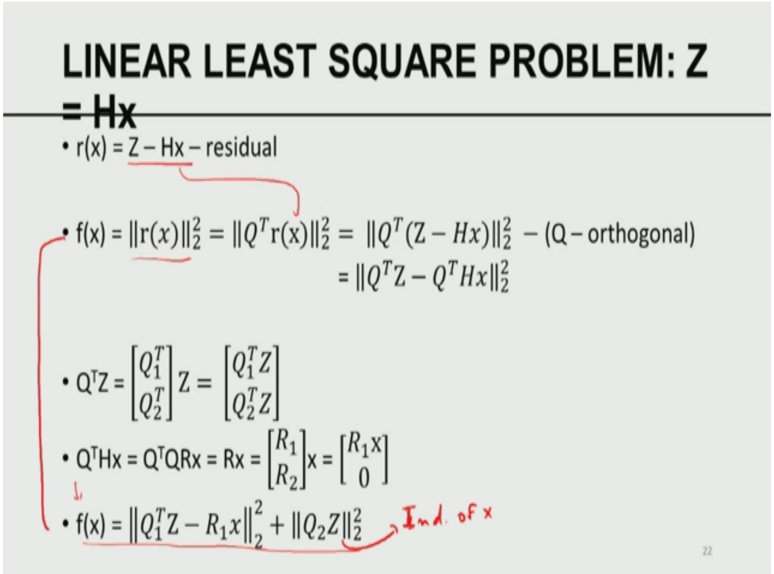
(Refer Slide Time: 56:05)



Therefore, I can partition Q as Q 1 Q 2, but Q is the first n columns of q here the rest of the n minus m columns of Q as be shown in the previous slide. R 2 is all symmetric is all zero matrix, R 1 is the first n by n is the first n rows of R I am sorry. First n rows of r therefore, I can express H is equal to QR, Q is equal to Q 1 Q 2, R 1 R 2. So, this is equal to Q 1 R 1 plus Q 2 R 2, but R 2 is 0, therefore, H is equal to Q 1 R 1. So, H is equal to Q

1 R 1 this is called reduce the QR decomposition I do not have to build those too many columns, too many zeros. And here the property of Q 1 is that Q 1 transpose Q 1 is I n, so that is the property of the sub matrix Q 1. So, Q 1 is a matrix is the rectangular matrix it has m rows and n columns and Q 1 transpose Q 1 is I n. So, let me go back here ah this is called the full QR decomposition in slide 20, this is called a reduced QR decomposition in slide 21.

(Refer Slide Time: 57:32)



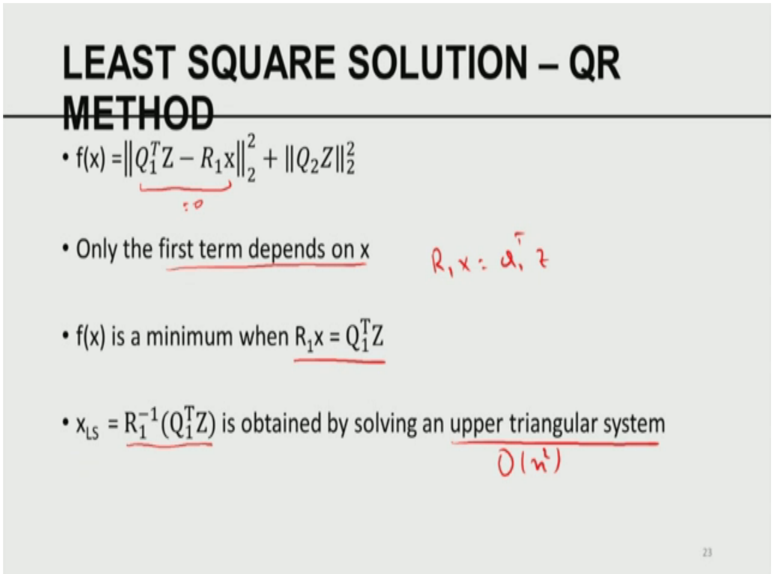
Now, let us see how I can utilize this in my least square problems. Please go back my r of x please go back my r of x is equal to z minus H of x, there is a residual. My f of x is equal to the square of the norm are the residual we already know that. We also know that the norm of a vector remains invariant under an orthogonal transformation we saw to start with. Columns of Q are orthogonal, so I can express this as Q transpose r of x.

So, r of x is equal to Q transpose r of x that is essentially an orthogonal transformation of the residual vector. If I substitute r of x is equal to Z of x in here I get this I can multiply Q transpose z is equal to Q transpose H of x. But we already know Q is Q 1 Q 2, therefore Q transpose Z is Q 1 transpose Q 2 Z z which is Q 1 transpose Z Q 2 transpose Z. Q 2 transpose H of x is equal to Q transpose QR of x and that is essentially r of x which is R 1 R 2 f x which is R 1 f x the bottom line is 0, because R 2 is 0.

So, when I combine these operations my f of x now becomes this f of x is equal to Q 1 transpose Z minus R 1 x square plus Q 2 Z. This is the sum of the squared residual. What is the import of this now x is the unknown. So, by changing x, I can so my job is to minimize f of x, f is a function of x, but the right hand side consists of two terms, the second term is independent of x. So, I cannot alter a term if it does not depend on x because the x is the free variable here. So, this is the second term is independent of x. So, it is a constant term I cannot do anything with that.

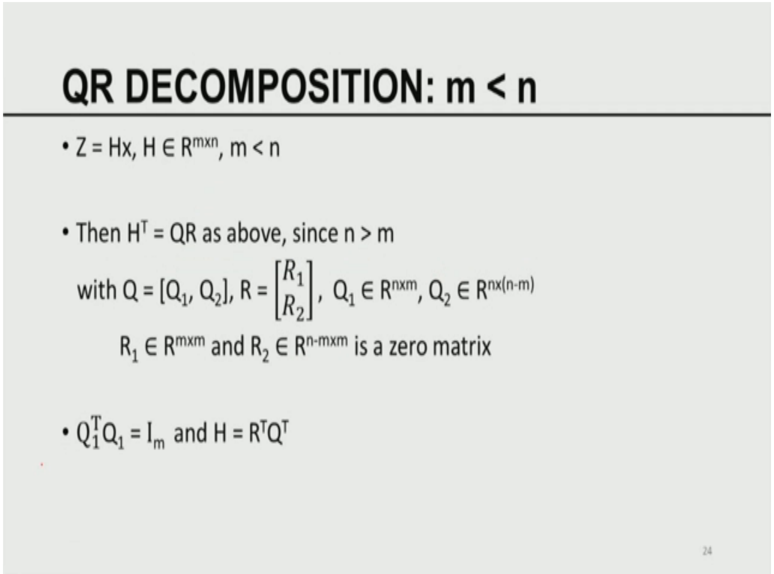
I can only manipulate the first term. So, I am going to minimize f of x only by minimizing the first term f of x by minimizing only the first term, I hope the argument is clear in here. This decomposition reduces f of x to a sum of two terms one depend on x another does not depend on f x, it depend on x. I can control only x, there is no other control I have; I have to minimize with respect to x. So, if I change x, the first term changes second term does not, therefore I need to concentrate only on manipulating the first term.

(Refer Slide Time: 60:21)



Only the first term depends on x this is again a summary of what I already talked about. The first term depends on x; the second term does not. And the first term is the minimum when does this get to be a minimum this is when the first term is zero when will the first term is 0 I will have R 1 of x is equal to Q 1 transpose z. Therefore, the least square solution is obtained by solving R 1 x is equal to Q 1 transpose z or least square solution is r one inverse Q 1 transpose z is obtained by solving an after upper triangular system. So, by spending money not on Choleskey decomposition you know H transpose H, but on decomposing H to be Q and R H to be Q and R, we have now reduced the problem of computing the solution to a linear least square problem to one of solving an upper triangular system. This is a very important development. Please understand solution of an upper triangular system and lower triangular system costs only O of n square.

(Refer Slide Time: 61:34)



The solution of an upper triangular system or lower triangular system is going to cost only O of n square, but the solution of a full system is going to cost you O of n cube therefore, this solution process is much cheaper than solving the normal equations. But what is the catch, catch is I still have to pay for the QR decomposition. Therefore, this methodology is a very elegant methodology, but their rests on being able to perform the QR decomposition on the matrix H, and that is a very fundamental operation.

So, the our next step in the development of matrix methods used to utilize the ability to factor H as Q times R, Q is orthogonal and R is upper triangular. If you can make it happen, I know R, because R is divided into R 1 and R 2 Q is dividing into Q 1 and Q 2, so R 1 is known, Q 1 is known, I can compute the solution of this very easily. So, the everything rests on our ability to do the QR decomposition to which we now turn.