MITOCW | MITRES2_002S10linear_lec09_300k-mp4

PROFESSOR: Ladies and gentlemen, welcome to lecture number 9. In this lecture, I would like to discuss with you the solution of equilibrium equations in static analysis. In the earlier lectures, we derived equations KU equals R. And we now have to solve these equations for the displacements stored in the vector U.

This is an important phase of the finite element analysis, because much of the computational effort goes into the solution of these equations. In the earliest finite element analyses, iterative methods were used, among them the Gauss-Seidel method. Now, the basic disadvantage of using an iterative method is that we do not know how many iterations have to be performed for solution, and thus it is very difficult to estimate the total solution effort to solve the equations KU equals R. Now we use almost exclusively direct methods, or computer programs that are in industry, in abundance use, are basically using direct methods. And these direct methods are basically variations of Gauss elimination, a method that Gauss some 100 years ago proposed for the solution of equations.

Now we are talking about static condensation, substructuring, frontal solution, LDL transpose factorization, Cholesky decomposition. We are also talking about the crude method, the column reduction skyline solver. But all of these methods are indeed variations of the basic Gauss elimination procedure. These are direct methods because the number of operations can be counted for the solution of the equations.

In fact, the number of operations for a system with n equations, a half bandwidth m, is 1/2 nm squared, where we of course use a half bandwidth m that is constant for the system. If the bandwidth varied, some adjustments have to be made to that variable m. A mean value has to be used. However, the important point is that we actually can count the number of operations to obtain the solution, which was not the case in the iterative method, because we did not know how many iterations have to be performed.

What I would like to do now in this lecture is to very briefly review with you how the basic Gauss elimination is performed, and then I would like to show to you how this method is indeed the basis of static condensation substructuring, et cetera. We will finally also consider the column reduction skyline solver, which is very effectively used, and that is in fact the equation solver that is used in Sab and the ADINA program.

Well, I have prepared, as before, some view graphs here. And the first view graph here shows a system of equations. This is the K matrix here. This is a displacement vector. This a load vector. And I will show you later on a physical system which in fact gives this stiffness matrix K. But let's look now at the system simply as an example for solving equations.

In the basic Gauss elimination procedure, we proceed as follows. We subtract a multiple of this equation here from the second equation and the third equation in order to produce here a 0, and to produce here a 0. Well, to obtain a 0 here, we have to take 1/5 of this equation, 1/5 of that equation, and subtract it from this equation. 1/5 because then we obtain a 1 here, and that 1 knocks that 1 out to obtain a 0. We have to subtract minus 4/5 of this equation from the second equation, and we obtain a 0 here, too.

While this process is shown on the next view graph, the result of that process is this set of equations. We now have produced our 0 here, our 0 here. Of course we had already a 0 here. We will see later on that the matrix in here, the 3-by-3 matrix, bounded by the dash lines on the side, in fact represents a stiffness matrix. That's important, because a stiffness matrix has always positive elements on the diagonal. Otherwise it is not a stiffness matrix. It does not govern a stable structure. See, here we have diagonal elements that are all positive, and if we can show that in the Gauss elimination procedure, we are still always producing new stiffness matrices, here a 3-by-3 matrix, then we directly can conclude that the diagonal elements must remain positive.

Well, in the Gauss elimination procedure, the same process that we use to obtain

from the 4-by-4 or 3-by-3 system is now carried on to obtain, from the 3-by-3 system, a new 2-by-2 system. This means that we have to subtract a multiple of the third equation from the fourth equation and the fifth equation in order to obtain a 0 here and a 0 here. For example, 5/14 of the second equation is subtracted from the fourth equation to obtain a 0 here. The result of that operation is this matrix here.

Now we have a 2-by-2 matrix here. Again, positive elements on the diagonal. In fact, this is a 2-by-2 stiffness matrix, as I will be showing to you just now.

Finally we proceed again, once more, with the same step, to produce a 0 here, which means that we have to subtract from this fourth equation a minus 20/7 divided by 15/7 of this equation. And the result, then, is this set of equations here. Notice that we now have a single element here, and in fact, this is a stiffness matrix, in this case, a 1-by-1 stiffness matrix, just one element, that governs the behavior of a single spring.

The solution off the equations is now completed by solving for u4 from this relation down here. This is a 1 degree of freedom system. Knowing u4, we can go back to this equation here, substitute u4 in, and the only unknown variable is u3. We solve, therefore, u3 from this equation. Now we know u4 and u3. We now go to this equation. Knowing u3 u4, the only unknown in this equation here is now u2. We can calculate u2, and repeating the process, we can calculate u1. A very simple, straightforward procedure, but extremely powerful. The result of that solution is shown here. Notice u1 is simply obtained by this operation here.

Of course, here we're having u2 and u3 in that we calculated already. u2 is calculated here. u4 is calculated here. It might be easier to actually look at it the way we discussed it just now. We call that the back substitution, because we are coming from below upwards, and in the back substitution, we have u4 first. We then calculate u3. u4 is known now, remember. We can substitute directly u4 into here, and we get u3. Knowing now u3 and u4, we can get u2, as shown here. And finally, we get u2.

This is the basic Gauss elimination procedure, an extremely powerful process. And I

would like to show you know how this procedure relates to the static condensation and relates to the other techniques, and how we then actually use this basic Gauss elimination procedure in an actual finite element program. Well, static condensation is the first topic that I'd like to talk about. The basic process is here that we are partitioning the total matrix into the following parts. A Kaa, Kac, Kca, and Kcc part. The cc part goes with Uc, the displacements Uc, which we want to condense statically out. Notice I'm putting the degrees of freedom that I want to condense out at the bottom of the U vector.

Well, by static condensation we mean that we express Uc in terms of Ua, which this equation here is obtained from the bottom equation here. We then substitute Uc in the top equation here, and the result is this equation here. Notice that the results can be remembered quite easily because we have here an aa, Kaa, then we have an a here, a c, a cc, ca. Notice that one might say, just for remembering it, that this c knocks out that c, this c knocks out that c, and what we are left with are aa's. And that means an aa here, an aa here. So the dimensions are compatible of the matrices, and the result is, I call it here, K bar aa. Of course the right-hand side has also changed, because the components must have an effect on the equation that give Ua.

Well, let's look at this process, called the static condensation process, for our simple example. And in this example now, I deliberately put the Kcc up here. In other words, the Uc degree of freedom is now the U1. This is all our Uc vector only containing now one component. The static condensation was originally proposed by considering or statically condensing out of the last equation, and that is how it's usually used also in finite element analysis. As an example, if we have incompatible modes, we might condense out the incompatible degrees of freedom, just as shown here. Or if we have internal degrees of freedom in finite elements that only that only couple with the bound the degrees of freedom of a finite element, we could condense them out this way, and then we would have this as the last degree of freedom. This would be Uc degrees of freedom, and we would condense them out, just as shown here.

However, in order to show you how the static condensation relates to Gauss elimination, I want to look at the example that we looked at earlier already, and this the partitioning I have used. U1 now being the UC, U2 U4 being the Ua degrees of freedom. So I want to now statically condense out Uc. The result of the K bar aa is shown here. Notice we are taking this part, this part here, this part here goes right there, this vector here is this vector here, 1/5 is the inverse of Kcc, and this vector here is that vector there.

Well, performing these operations, the result is this one here. That is the result. And if we go back to our earlier Gauss elimination procedure, we in fact notice that this matrix, K bar aa, is nothing else than the matrix that we derived from the 4-by-4 system when we eliminated the non-zero elements here. In other words, we subtracted in the Gauss elimination from the second equation and the third equation a multiple of the first equation, we got these zeros here. The result was this 3-by-3 matrix. And that 3-by-3 matrix is exactly this matrix. So what have we been doing then in the Gauss elimination? We have really statically condensed out the first degree of freedom, and we have obtained a stiffness matrix, K bar aa, in that Gauss elimination procedure.

We can now also look at the physics that we have been going through, and this is, in my opinion, a very important part also. We start off with this matrix here. Notice the following00 this now is the stiffness matrix of this beam for the U1 to the U4 degrees of freedom. This stiffness matrix actually has been derived by finite differences. It is a nice example to show demonstratively what happens in the Gauss elimination. Of course, we could've also used a finite element system, but I chose here a finite difference matrix, which is still a stiffness matrix, and which I can use very nicely to show to you the basic procedures of the Gauss elimination.

So this stiffness matrix governed this system here. Notice the following. The first column, this column here, represents the forces. When I displace this node, if you think of this as a note, by unity, and keep all the other degrees of freedom to 0-- in other words, if I put this deflected shape onto the beam-- 0 displacements right here, but a unit displacement here. That is a unit displacement.

Now notice, these are the forces that have to act at these degrees of freedom in order to give this deflected shape. The force here, if I put them here in blue, is 5, as an example. The force here is minus 4. The force here is 1, and the force here turns out to be 0. That element, or that element, of course, we have a symmetric matrix.

And of course, in the same way, we can interpret other elements in this matrix. 1 column represents always forces that have to act onto the system in order to obtain unit displacements at 1 degree of freedom. If you look at the I's column, then it's the I's degree of freedom. And all the other degrees of freedom are set to 0.

Well, if we perform our Gauss elimination or static condensation-- we have shown that it is the same-- on this system, in the first step, we obtain a 3-by-3 matrix. Physically, what we have obtained is the stiffness matrix of this system now. And the following important point is now to be noted. If I look at this column just in the same way as I looked at the first column of the 4-by-4 system, then this column represents now the forces that are required at these degrees of freedom to impose a unit displacement here at this degree of freedom, this now being unity, with 0 displacements here. And the force required here is 14/5. That is the force that I have to apply here to get a unit displacement here. At the same time, I have to apply here a force of minus 16/5, and I have to apply here a force of 1.

So I'm talking again about forces acting onto the system in this matrix here. And these forces are stored in the columns or the rows off the matrix, and in the I's column or row, I have the forces required to impose a unit displacement at the I's degree of freedom with all the degrees of freedom being 0. And the important point is that having performed now the Gauss elimination, here I do not have anymore a degree of freedom. The system is free to go down here. And that's what we mean by static condensation, really. We have taken a degree of freedom out of the system.

Proceeding in the same way, we obtain this 2 by 2 system. Now we only have 2 degrees of freedom. I have condensed out this degree of freedom and that degree

of freedom, which were there. The beam is now free to move down here, so the deflected shape corresponding to U3, this one here, would be like that, with 0 displacement here, and unit displacement there. The forces required at these degrees of freedom are stored in this column here.

Now, we immediately can observe that since we are talking about a physical system into which we have to put energy to deform it, we can conclude that these elements here must be positive elements. In other words, the diagonal elements in the stiffness matrices that we derive must be positive. They must be positive because if I push down the U3 degree of freedom to obtain a unit displacement, surely the force that I have to apply here must be positive. In other words, it must be a force into the direction of the degree of freedom U3. And that means this must be a positive element.

So if, in a Gauss elimination procedure, we obtain, all of a sudden, a negative diagonal element, this one or that one, as an example, then we would know that the structure we are analyzing is unstable. In the final step of the Gauss elimination, we obtain a single degree of freedom system. And that single degree of freedom system behaves like that. Notice no more degree of freedoms here.

And the equation-- I have to apologize that this matrix has been written a little bit down. It should be a little bit higher. They should, of course, be aligned these three. But basically, we have 5/6 times U4 equals 7/6. And this here is really nothing else than the stiffness of a spring. A spring K having us a number 5/6. In other words, a spring stiffness of 5/6 here. This is U4, the U4 degree of freedom. And the force applied to that spring is 7/6. We have a single degree of freedom system.

But notice that this single degree of freedom system, this stiffness now, K equals 5/6, contains really all the physical properties of that beam. The effect of the other degrees of freedom, U1, U2, U3, have been carried over into this spring stiffness. Similarly, 7/6 is the force which is a result off the other forces that were applied to the other degrees of freedom.

And now from the single degree of freedom system, of course, we can solve for U4,

and then we go back to the 2 degree of freedom system to solve for U3, and so on. By that back substitution process, we obtain thus the displacement that the beam is actually undergoing.

So the important point is that when we do perform Gauss elimination, we really operate always on physical systems. We operate always from one stiffness matrix on to another stiffness matrix, and the next stiffness matrix is basically obtained by static condensation. We are statically condensing out 1 degree of freedom after the next.

Well, if that is very well understood, and I urge you to read up in the book on this procedure, then I think you can very well appreciate how the substructuring process, frontal solution method, directly relates, again, to Gauss elimination. And here I summarize once the basic procedure that we use in substructuring analysis. We really go through static condensation on the internal degrees of freedom of a substructure. Here I'm showing a substructure, which is an assemblage, in this case, of the plane stress 4-node elements, just for illustration. Of course, there could be beam elements internally here or externally here. There could be shell elements. Any kind of elements. But the important point is that we have an assemblage of elements, which by itself, of course, gives us a structure already.

We use static condensation on the internal degrees of freedom of the substructure. Well, the result, then, is a new stiffness matrix of the substructure involving boundary degrees of freedom only. Let us look, then, at this example. Here we have 2 degrees of freedom at each node. At the boundary nodes, which are shown solid, here, and the hollow points here, the circles here are the internal nodes. 2 degrees of freedom at each. The result is that we have a 50-by-50 stiffness matrix, because we have 1, 2, 3, 4, 5 nodes along this length, and 5 layers this way. So 25 nodes times 2 degrees of freedom per node gives us a 50-by-50 stiffness matrix.

Now what we do in the substructuring analysis, we statically condense out the internal degrees of freedom, just in the way I have shown how we condense out in that beam element, in that beam analysis, U1 to U3, the same way we are

condensing out the internal degrees of freedom. Notice that we do not need to perform Gauss elimination from U1 to U3 or U4 onwards. We could also first deal with U2, then with U4, then with U1, and then solve for U3.

So we have really a lot of flexibility in the Gauss elimination procedure. The important point is that now we're using it to condense these internal degrees of freedom out. The result, then, is a stiffness matrix that only involves the boundary degrees of freedom. Now in this case, we have 1, 2, 3, 4, 5 nodes here. Another 5 is 10. 3 here and 3 here means 16, times 2 means it's going to be is 32 by 32 stiffness matrix.

Notice that the internal degrees of freedom have been statically condensed out, which means that if I impose, for example, a unit displacement here, keeping all the other displacements 0 at the other nodes, and I keep this one also 0, then in this case, I would have internal displacements at these nodes. Whereas in this case, if I put a unit displacement only corresponding to this degree of freedom and keep all other displacements equal to 0, I would have 0 displacements at these nodes, at these points. Here these nodes are free to go wherever they want to go in the finite element solution. And if I put a unit displacement on here, they will be taking on a particular value. These nodal displacements will take on a particular value.

What we really have derived, and this is important, is a new stiffness matrix. A new, we might call it, element stiffness matrix. A superelement, or substructure stiffness matrix. These are all equivalent words that, however, only involve boundary degrees of freedom. This is now a new finite element that can be used just as any ordinary finite element in the assemblage process to obtain the stiffness matrix of the complete structure.

Let me show to you a simple example. Here is a very simple structure, a truss element that has only 3 degrees of freedom. U1, U2, U3. The stiffness matrix of that truss element is shown here as 3-by-3. Matrix here, we see U1, U2, U3, and the forces corresponding to these degrees of freedom, of course, are listed here.

The objective in this example is to statically condense out this degree of freedom,

and thus obtain a new element stiffness matrix that only involve the boundary degrees of freedom. Well, the first step that I've pursued here was to rearrange the equation so as to have U1 and U3 as the top equations, and U2 as the bottom equation. So we want to do static condensation now from the back, or from the bottom, the way it's usually done. And it's usually done that way because it is numerically effective to proceed that way.

The result, then, is shown here. We write down just to identify, to use the various parts. This matrix here is this 2-by-2 matrix. This minus 20, minus 28 is that one here. Here we get the Kcc inverse, which is this part here, and then of course here we have this vector, shown here. If we perform these multiplications here, we obtain this stiffness matrix here. A 1 minus 1 minus 11 with a constant in front.

Of course, the load vector also has changed. There is a carry over from the 2 degree of freedom in to the 1 and 3 degrees of freedom. The U2 degree of freedom then can be recovered via this equation here, going back to the original equation. The important point is that we have now obtained the stiffness matrix of the truss element, this stiffness matrix of the trust element, involving U1 and U3 only. U2 is not anymore present. It has been statically condensed out. The U2 will adjust itself to a particular value, and that value, of course, is given right here. But the stiffness matrix that we have now here only corresponds to U1 and U3.

Well, that process can then be repeated, and we can go through a process that we call multilevel substructuring. In other words, we don't just go once through a substructuring process, but we go repeatedly through substructuring processes. And basically, what we are doing is we are solving the system of equations KU equals R of the complete system in a very effective way. Let me show you here an example.

Here we have a simple truss assemblage. We have 1, 2, 3, 4 elements. The important point is here that the elements are all similar. The area is varying 2A1, 4A1, 8A1, 16A1. And what I want to do is, I want to deal now, via substructuring with the equations to obtain a very effective solution. The first step is to look at this

element, which involves U1 and U3 and U2. And here you see, once more, a sketch of that element. That is the element here.

Now I condense out U2, and that's what we just did in the previous example. The only degrees of freedom that we are left with, then, are U1 U3. So now we have an element stiffness matrix, this element stiffness matrix, but in terms of U1 and U3 only.

The next step, then, is to say, let us repeat this process. Let us now say, since we have this element stiffness matrix here in terms of U1 and U3 only, and since this adamant is similar to this element here, there is only a scalar involved because the area is varying as shown, from A1 to 2A1 to 4A1 and so on, we can use this stiffness matrix here, this element stiffness matrix, which, however, is already a substructure the way I've defined it, to assemble the stiffness matrix corresponding to both of these elements. And that process is schematically shown here. This stiffness matrix now will involve U1, U3, and U5. Notice U2 is gone, and the important point U4 is gone also, because I've dealt with U2, I have statically condensed out U2, and by that, of course, I also have statically condensed out U4 simultaneously, because I'm using this element stiffness matrix here again. So U4 is also gone already.

And now all I need to do is condense out U3, and I obtain a second level substructure, which I can use again now to assemble this element here with that element. And that is shown on the next view graph here. I repeat that process using this element here now, which is this part here, involving only U1 and U5, and this element here, which is similar to that element. Again, an area variation there, of course. And I obtain an element stiffness matrix, or a structural stiffness matrix, whichever way you want to look at it, that involves U1, U5, and U9 only. I can now statically condense out U5. The only unknowns I'm left with are U1 and U9, and of course then I can simply solve for U1 and U9. Of course, we would have to have also one of the displacement being prescribed to actually perform a solution. Otherwise we have a rigid body mode in the system. Assuming that, say, U1 is 0, we now would impose U1 being 0, solve for U9, go back, solve for U5, and proceed

backwards to solve for all the displacements.

The important point is that via this substructuring process, a multilevel substructuring process, we have saved operations. We have saved operations because in this particular case, you see, I have condensed out these degrees of freedom already, and I use this fact in this assemblage process and solution in such a way as that I do not need to condense out these degrees of freedom, that I do not need to deal with these degrees of freedom again, because I've dealt with them in effect already by condensing out these degrees of freedom here.

There are also other advantages using substructuring, and one important advantage that I should mention is that in the input definition phase, we only need to define the elements, nodal points, and so on of a typical substructure, and we can use that definition in a computer program, then, again and again, to assemble the complete structure. In other words, if I have defined, typically for a high-rise building, one typical floor, and I have assembled the floor stiffness matrix with columns, beams, and so on, that pertain to that floor, if I then statically condense out the degrees of freedom that we want to condense out for that part of the substructure, I can use that floor substructure, so to say, again and again, to assemble the complete building structure. And this means I only have to define really the particular substructure input, and I deal, then, with that substructure numerically, as I have shown here in the example.

The next important procedure that I like to discuss very briefly with you is the frontal solution. And the frontal solution really, again, only represents a variation of the basic Gauss elimination procedure. It represents, really, a static condensation procedure. And that static condensation, however, is performed in a particular way. And this is really where the advantages of the frontal solution can lie. It's performed via looking at the elements. Let me describe to you the basic process.

Here we have, as an example, a two-dimensional plane stress element mesh. We have 4-node elements, as shown here, element 1, element 2, element 3, element 4, q, q plus 1, and so on. In other words, there's a whole lay of elements this way, and

that way, and so on. This mesh could represent, for example, a cantilever plate that acts in plane stress.

In the frontal solution, the solution of the KU equals R equation is performed by elements. So we're putting in element 1, which couples into node 1, node 2, node m, and node n plus 1. Since the degrees of freedom at node 1 only obtain stiffness from element 1, and no other element, the assemblage of this element into the global structure stiffness matrix gives the only stiffness contribution at node 1. Therefore, we have the final equations at node 1, and we can deal with them already.

And that's how the frontal solution proceeds now. It statically condenses out these degrees of freedom. Then we move on to the next node of element 1, this node here. However, this node gets stiffness from element 1 and element 2. So in order to obtain the final equations here, we have to add the element 2 stiffness matrix to the element 1 stiffness matrix. And that means in this way, then, of course, we obtain the final equation, corresponding to node 2. We now can statically condense out the degrees of freedom at node 2.

We then move on to the next node, involving element 1 and element 2, which is node 3, say. And now we have to, however, assemble the stiffness matrix of elements 3, also into the equations that we have assembled already. Having done so, we can now use static condensation to condense out these degrees of freedom.

We proceed, therefore, via elements, and we are talking about a wave front-- which is this one for node 1, this is the wave front for node 2, this is the wave front-- in other words, what we are saying is, the wave front really embodies or contains all the elements that we need to have assembled before we can condense out degrees of freedom. To condense out the degrees of freedom at node 2, we have to have a wave front such as shown here. We have to have assembled element 1 and 2, because element 1 and element 2 give stiffness to node 2.

The important point is that we are proceeding via static condensation from node 1 to node 2 and so on, and this means really that we are performing Gauss

elimination again. The process is summarized here once more. The frontal solution consists of successive static condensation of nodal degrees for freedom. The solution is performed in the order of the element numbering. That's important. Same number of operations are performed in the frontal solution as in the skyline solution, if the element numbering in the wave front solution corresponds to the nodal point numbering in the skyline solution. So if the element numbering in the frontal solution corresponds to the nodal point numbering in the skyline solution, we perform in the skyline solution, which I have not really talked about yet, and that's what I want to get on now, the same number of operations as in the frontal solution.

By corresponding, I mean in this particular case here, we would condense out in the frontal solution the degrees of freedom at node 1, then 2, then 3, then 4, and so on. And of course, in the skyline solution, or in the Gauss elimination procedure, we also proceed via node 1, 2, 3, 4, and so on. So in this particular example, the frontal solution and the skyline solution would give the same number of operations.

Well, let me go on to the skyline solution, then. And the basis of that solution is LDL transpose factorization. Basically again, Gauss elimination. The basic step here is the following. We are having our stiffness matrix, K here, and now I'm going back to that example which I discussed earlier, the beam example. We're talking about the 4-by-4 stiffness matrix again. We're using this stiffness matrix, and we premultiply it by an L1 inverse matrix, this matrix here. In fact we never perform formally this premultiplication via matrix, because what this does is, it subtracts a multiple of this first row from the rows below it to update a 0 here and a 0 there. That's all this premultiplication does. And the result is this matrix here, which I've shown to you earlier already.

The next step is then to operate on this matrix in the same way. However, before doing so, let me show you here a view graph in which I have written out specifically the L1 inverse. And the important point, L1. Notice that L1, the inverse of this matrix, is obtained by trivial operation. Namely, simply changing the plus here to a minus sign, and the minus sign to a plus here. Changing the signs of the off-diagonal elements gives us directly the inverse of that matrix.

Well, having obtained, therefore, via this operation, or having carried out via this operation, I should say, the first step of the basic Gauss elimination procedure, namely subtracting the top row from the rows below it to obtain zeroes in the first column below the diagonal element, we proceed in the same way. and via this procedure here-- now, of course, we have an L2 inverse, and so on-- we obtain what we call an upper triangular matrix. S, an upper triangular matrix. Zeroes all below the diagonal, and only non-zero elements right here. Of course, if the matrix has a bandwidth originally, that bandwidth would be preserved, and we would have no non-zero elements outside that band. In other words, all elements outside the band would, in fact, be also zero elements.

This upper triangular matrix, then, obtained-- this is the S matrix, here-- will be used in the back substitution. This is here the element that gives us the single degree of freedom stiffness, which I talked about earlier, from which we can solve Un. Here I'm talking about a general case with n degrees of freedom.

However, looking at this equation, let us first identify that we can put all of these inverse matrices onto the right-hand side, and we obtain this equation here. Now notice that this L1 is obtained by trivial operation from L1 inverse. The same holds for L2 and so on. So we never really had to invert a matrix to obtain this part here. Furthermore, this L1 was an identity matrix with only non-zero, off-diagonal elements in the first column. This one is a matrix which looks like the identity matrix, but has also non-zero, off-diagonal elements in the second column. And so on.

And the product of these matrices here, then, is very simply obtained by writing the 1's into the diagonal positions, and the off-diagonal elements here are those of L1 in the first column, those of L2 in the second column, and so on. So even this multiplication is obtained in a very simple way. We only need to write the first column of L1 into here, the second column of L2 into here, the third column of L3 into here. And the result, all of that, I call the L matrix.

Now notice that once again, we do not perform any matrix multiplication. We have n minus 1 matrices here, but we do not perform a matrix multiplication, because all we

do is, we take the first column of L1 and put it into the first column of L. The second column of L2 is the second column in L. The n minus second column in L is nothing else than the n minus second column in L n minus 2. And so on. The result, therefore, is that we have written k equal to Lf, L being equal to this product here.

Now because K is symmetric, we can, in fact, write this F matrix here as D times L transposed, where d is a diagonal matrix, dii being equal to sii. So now we have factorized the K matrix into LDL transposed. The d elements, or the dii elements, must all be positive if our structure is stable, as I pointed out earlier.

The Cholesky factorization, just by remark, uses this form, K being L curl times L curl transpose, where we should note that L curl is really nothing else than L times d to the 1/2, where d to the 1/2 is obtained by using the D matrix and taking square roots of the diagonal elements. In other words, the ith elemental of d to the 1/2 is nothing else than square root out of dii.

Well, having performed the LDL transpose factorization, we now are ready to solve the equations, and they are solved in the following steps. We have LV equals R, where V is defined as shown here. Notice that if I were to put this part into there, I really obtain nothing else than LDL transpose times U equals R, where this is, of course, nothing else than K times U equals R, if I were to substitute from here into there. However, we separate these two steps out, and we call this the forward reduction of R, which is really obtained by subtracting specific elements of R from other elements. We never really perform a matrix product here. We write it that way, but really, it means nothing else than taking, say, the ith element off R and subtracting that element, or multiple of that element, from the elements below it. That is a typical operation performed when we form the V vector.

Having obtained V, we go back to this equation here, and we perform this step. Notice that this step is nothing else than the division off the V elements by the D elements. In other words, here we take Vi and divide it by dii.

The final step, then, is the back substitution, which gives us U. The back substitution is performed as I discussed earlier. Already we're going from UN, from the last

degree of freedom, upwards to calculate the values of the degrees of freedom.

Let us now look at the actual implementation. The implementation has to be performed very effectively, of course. All I have been giving to you now so far were the mathematical equations that we are basically operating on. The LDL transpose factorization is, of course, a way of writing a set of steps of solution that we are performing. However, we want to perform them as effectively as possible, and the scheme that we're using is related, really, to the crude scheme-- we call it the column reduction scheme-- that we are going through for the solution of the equation.

And the procedure that we're using is as follows. Here we have the stiffness matrix. Notice that we only store the upper part. Since we have a asymmetric matrix, we can directly obtain the lower part elements from the upper part elements. We do not need to store them in the computer program.

We also perform the Gauss elimination in a very, you might say, strange way, when you look at it first. And the way we're performing it is the following. That we are reducing this column first. In other words, we are going column-wise. We are reducing this column first. The result is that one.

In the next step, then, we are going through the reduction of the next column. See here, we are now reducing this column, and the result is that one. The dashed line always is on the right-hand side of that part of the matrix, which has already been reduced. And now we are looking here at the third column, and this is the reduction off the third column The final step is then that we are reducing the fourth column, and this is the result.

Now notice that having performed these operations column-wise-- and I have not given you the details. We do not have time in this lecture to look at the details. The details are in the book, if you want to read them up. All I'd like to convey to you now are the procedures that we're using to show you the effectiveness. Notice that the final result here in this operation is the D matrix on the diagonal and the L transpose in the off-diagonal elements. And we have done it column-wise.

Well, what does this mean, then, in practice? In practice it means the following. Notice that I have now a system which has different column heights for different equations. The column heights are being defined by the first non-zero element in a column. So this is the column height for this last column here, because this is the first non-zero element in the column. The zero elements below this element here, in general, become non-zero elements. In general, not always. But because they in general become non-zero elements, we have defined our column height this way. And what we are now doing in the actual solution is that we are reducing, column by column, just the way I have shown it, to obtain finally the elements of the D matrix on the diagonal, and the elements of the L transpose matrix in the off-diagonal elements here.

And we can do so-- this is important-- by simply replacing elements. If you go back to an earlier step that we looked at, what we have done here is looked at this matrix here, and we, in the reduction process, simply replaced this element by 1/5, replaced this element by the minus 4 element, by minus 8/7, over and so on. So we don't need additional storage. We can simply replace elements.

In the same way, we proceed now here, as shown, on the next view graph. Here we have filled in the zero element, and the reduction has been performed column after column. The effectiveness of this whole procedure is really seen on this view graph here, when we talk about large systems. I want to denote here this as a large system, because the high speed core storage is not sufficient to store the whole matrix in. And so what we have been doing is, we are storing the matrix in blocks. This is here the first block, shown by the dashed line. This is the second block. This is the third block, and that's the fourth block.

What I'm saying is that we have divided the total stiffness matrix into blocks, and in fact, we divided the blocks in such a way as to be able to take always at least two blocks into core. In other words, in this solution process, we have to be able to take this block into core, that block into core. Two blocks. We also would have to be able to take this block into core, that block into core, and so on.

Of course, the way this is actually automatized in a computer program, is that we say we have so much storage available. Say that we have 40,000 storage locations available. Then for the solution of the equations, or for the storing of the stiffness matrix, then 20,000 locations would be used per block. And what the solution routine simply does-- it counts these elements and puts this dashed line in such a way that 20,000 elements are in here, 20,000 elements are in here, 20,000 elements are in here, and so on.

The solution procedure, or the solution of the equation, is now performed in the following way, using the block structure. We proceed with the reduction off this column, of that column, of that column and that column. Now, I should point out the following important point. When we do we do reduce this column, this column coupled into the first equation. So we have to have these elements in core. Of course, since this is 1 block, and since we can store that block in core anyway, there is no problem. We can reduce this column directly. These elements that we already have obtained are indeed in core.

Now let's go on to the second block. The second block coupled into the first block, because these elements here couple into these equations. So in order to reduce the second block column by column, we have to have the first block in core. And that's what we do. We take this block into core and reduce the second block.

Having finished the second block reduction, we go on to the fourth block. And the fourth block now is reduced by first taking the first block into core, because the first block also couples into the third block. Then taking the second block into core together with this one, and reducing this one, and then of course, finally, we reduce this one.

So the general procedure, therefore, is that we take a particular block into core that we want to reduce, we have reduced already all the previous blocks, we have storage available also for one of the previous blocks, and we sequentially take the previous blocks into core to take into account all the coupling that these previous blocks have onto the current block that we want to reduce. In this example, we're

taking this block into core. We then take first also this block into core. That means we have two blocks in core. We take the effect of this block into account onto that block. Then we throw this book out. We take this block into core. We take the effect of this block into account onto this block, and finally, we reduce the block by itself.

In the same way, we would proceed with this block, just as another example. We take this block into a core. This one couples into all previous blocks, so what we have to do is take this into core, and first take this into core, take the effect of this coupling into account onto this block, throw this out, take this one into core, put this block coupling onto that block, throw this out, take this one into core, and finally take the effect of this block coupling onto that one into account, and finally then, of course, reduce this block by itself.

This, then, completes the complete solution of a large system. I only was able to give you the basic principles, the basic ideas. Some of the details are described in the textbook, and you can read up there further on it.

Thank you very much for your attention. This was all what I wanted to say in this lecture.