Additional Note #1

Solvers of a System of Linear Equations

97W

Introduction

There are two kinds of finite element equation solvers : the direct and iteration methods. The direct method is based on the Gaussian elimination method, and its original form was modified by various ways to reduce the number of operations in computing, to improve the matrix condition number for accuracy, and to reduce the storage space of the coefficient matrix. Roughly speaking the total number of operations to finish up the Gaussian elimination procedure to solve a system of linear equations with n number of equations, is the order of n^3 , that is, $O(n^3)$, while the required space to store the coefficient matrix is $O(n^2)$ if all of the coefficients are stored. If a coefficient is stored in 32 bit form, then we need $64n^2$ bits space. In other words, if we have 20MB space for the coefficient matrix, then the largest n is about 790. This means that we can solve only 790 equations, and then if a cubic hexagonal structure is modeled by 8 node slid brick elements, we can only decompose it into 5x5x5 meshes. This is not acceptable setting in practice. If we have 20MB space for the coefficients, we would like to solve at least 10,000 equations. We can find considerably many researches in the 1970s and 80s to reduce the storage space required. Typical outcome from such heavy research in the area of the finite element method and computational science, are, for example, the band method, skyline (or profile) method, wave front (frontal) method, and others. Especially, the wave front method that was developed by B. Iron was the highest achievement of the finite element method in the early 1970s. Because of this program to solve a system of linear equations that is specially designed by using the nature of linear elasticity, the finite element method could attract wide range people for its application, and we started solving “large scale” problems with O(1000) equations using 128K main memory computers with magnetic tape drives which provide leap frog type advancement in engineering analysis. He basically developed an algorithm which only requires storage of non-zero coefficients. It is noted that the coefficient matrix, that is the global stiffness matrix in the finite element analysis, contains mostly zeros. In other words, there are considerably less number of non-zero terms than zeros. Thus, as Iron developed, if we have an algorithm of the Gaussian elimination method which requires only non-zero terms, we can reduce large amount of storage space as well as computing time by skipping for zeros which do not make any influence. After Iron’s work, there were many modifications and variations, and then the skyline (profile) method became popular among researchers and students because of its simplicity. A short coming of the wave front method is its complexity of algorithm and programming, although it was considerably simplified later, and it was applied by very specialists of the finite element method. On the other hand, the skyline method published in the book by Bathe was so simple that any engineering students could understand if they have background of the Gaussian elimination method, and they could make their own computer programs without much effort. The skyline method requires more space than the wave front method, but it only requires storage of non-zero and zero terms bounded by the so-called skyline that is the most outer limit of non-zeros in columns of the coefficient matrix. It can be regarded as a variation of the band method that stores the non-zero terms in a rectangular matrix.

Direct Methods

The original Gaussian elimination method without pivoting can be explained as follows. Let a discrete finite element equation (which is a typical matrix equation, and represents a system of linear equation with n number of unknowns and n number of equations) :

$$\boldsymbol{Ax} = \boldsymbol{b}$$

or

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & .. & .. & a_{1n} \\ a_{21} & a_{22} & a_{23} & .. & .. & a_{2n} \\ a_{31} & a_{32} & a_{33} & .. & .. & a_{3n} \\ : & : & : & & & : \\ : & : & : & & & : \\ a_{n1} & a_{n2} & a_{n3} & .. & ,, & a_{nn} \end{bmatrix} \begin{Bmatrix} x_1 \\ x_2 \\ x_3 \\ : \\ : \\ x_n \end{Bmatrix} = \begin{Bmatrix} b_1 \\ b_2 \\ b_3 \\ : \\ : \\ b_n \end{Bmatrix}$$

The first step is elimination of the terms $a_{21}, a_{31}, ..., a_{n1}$ of the first column using the first row of the coefficient matrix $\boldsymbol{A}$ under the assumption that a_{11} is not equal to zero by using the algorithm

$$a_{km} - \frac{a_{k1}}{a_{11}} a_{1m} \to a_{km} \quad , \quad k = 2,3,...,n \quad \text{and} \quad m = 1,2,...,n$$

$$b_k - \frac{a_{k1}}{a_{11}} b_1 \to b_k \quad , \quad k = 2,3,...,n$$

This operation leads the new matrix equation

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & .. & .. & a_{1n} \\ 0 & a_{22} & a_{23} & .. & .. & a_{2n} \\ 0 & a_{32} & a_{33} & .. & .. & a_{3n} \\ : & : & : & & & : \\ : & : & : & & & : \\ 0 & a_{n2} & a_{n3} & .. & ,, & a_{nn} \end{bmatrix} \begin{Bmatrix} x_1 \\ x_2 \\ x_3 \\ : \\ : \\ x_n \end{Bmatrix} = \begin{Bmatrix} b_1 \\ b_2 \\ b_3 \\ : \\ : \\ b_n \end{Bmatrix}$$

Here note that although we are using same characters a_{ij} with the original matrix equation, their values are different since we replaced the original ones by the algorithm described. The next is to eliminate the terms $a_{32}, a_{42}, ..., a_{n2}$ of the second column by the similar algorithm

$$a_{km} - \frac{a_{k2}}{a_{22}} a_{2m} \to a_{km} \quad , \quad k = 3,4,...,n \quad \text{and} \quad m = 2,3,...,n$$

$$b_k - \frac{a_{k2}}{a_{22}} b_2 \to b_k \quad , \quad k = 3,4,...,n$$

and obtain the new matrix equation

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & .. & .. & a_{1n} \\ 0 & a_{22} & a_{23} & .. & .. & a_{2n} \\ 0 & 0 & a_{33} & .. & .. & a_{3n} \\ : & : & : & & & : \\ : & : & : & & & : \\ 0 & 0 & a_{n3} & .. & ,, & a_{nn} \end{bmatrix} \begin{Bmatrix} x_1 \\ x_2 \\ x_3 \\ : \\ : \\ x_n \end{Bmatrix} = \begin{Bmatrix} b_1 \\ b_2 \\ b_3 \\ : \\ : \\ b_n \end{Bmatrix}.$$

Repeat this process until the upper triangular coefficient matrix is obtained :

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & .. & .. & a_{1n} \\ 0 & a_{22} & a_{23} & .. & .. & a_{2n} \\ 0 & 0 & a_{33} & .. & .. & a_{3n} \\ : & : & : & & & : \\ : & : & : & & & : \\ 0 & 0 & 0 & .. & ,, & a_{nn} \end{bmatrix} \begin{Bmatrix} x_1 \\ x_2 \\ x_3 \\ : \\ : \\ x_n \end{Bmatrix} = \begin{Bmatrix} b_1 \\ b_2 \\ b_3 \\ : \\ : \\ b_n \end{Bmatrix}$$

so that it can be solved from the last equation step by step :

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

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

and

$$x_k = \frac{b_k - \sum_{j=k+1}^{n} a_{kj} x_j}{a_{kk}} \quad , \quad k = n-1, n-2, ..., 1.$$

The process making the upper triangular coefficient matrix is called the forward elimination, the process modifying the right hand side is called the reduction, and the last step to solve the system linear equations with the upper triangular coefficient matrix is called the back substitution. If we examine this process, and if the coefficient matrix has the special structure such that all of non-zero terms are located within the band from the diagonals :

$$\begin{bmatrix} * & * & * & * & * & 0 & 0 & 0 & 0 & 0 \\ * & * & * & * & * & * & 0 & 0 & 0 & 0 \\ * & * & * & * & * & * & * & 0 & 0 & 0 \\ * & * & * & * & * & * & * & * & 0 & 0 \\ * & * & * & * & * & * & * & * & * & 0 \\ 0 & * & * & * & * & * & * & * & * & * \\ 0 & 0 & * & * & * & * & * & * & * & * \\ 0 & 0 & 0 & * & * & * & * & * & * & * \\ 0 & 0 & 0 & 0 & * & * & * & * & * & * \\ 0 & 0 & 0 & 0 & 0 & * & * & * & * & * \end{bmatrix}$$

If we examine the algorithm of the Gaussian elimination, we can easily find that the zeros outside the band will neither make any affect to the coefficient matrix nor be changed by the algorithm. That is, we need not store them and we need not process them. This way we can save the storage space for the coefficient matrix and also we can save computing time by neglecting them. This is the idea of the band method. Most of systems of linear equations obtained in the finite element method for linear elastic structure are this type.

As a variation of the Gaussian elimination algorithm, we can find the Crout method that makes a LU decomposition of the coefficient matrix ***A*** by a lower triangular matrix ***L*** and the upper triangular matrix ***U***, or the Cholesski method that makes a ***LDU*** decomposition of A into a lower triangular matrix ***L***, diagonal matrix ***D***, and upper triangular matrix ***U***. Suppose that a given matrix ***A*** is decomposed into a LU form :

$$\boldsymbol{LUx} = \boldsymbol{b}$$

where

$$\boldsymbol{L} = \begin{bmatrix} l_{11} & 0 & 0 & \ldots & 0 \\ l_{21} & l_{22} & 0 & \ldots & 0 \\ l_{31} & l_{32} & l_{33} & \ldots & 0 \\ : & : & : & & : \\ l_{n1} & l_{n2} & l_{n3} & \ldots & l_{nn} \end{bmatrix} \text{ ,and } \boldsymbol{U} = \begin{bmatrix} u_{11} & u_{12} & u_{13} & \ldots & u_{1n} \\ 0 & u_{22} & u_{23} & \ldots & u_{2n} \\ 0 & 0 & u_{33} & \ldots & u_{3n} \\ : & : & : & & : \\ 0 & 0 & 0 & \ldots & u_{nn} \end{bmatrix}.$$

Then the original problem can be decomposed to two matrix equations :

$$\boldsymbol{Ly} = \boldsymbol{b} \quad \text{and} \quad \boldsymbol{Ux} = \boldsymbol{y}$$

which can be solved easily by the following algorithm :

$$l_{11}y_1 = b_1 \quad \Rightarrow \quad y_1 = \frac{b_1}{l_{11}}$$

$$l_{21}y_1 + l_{22}y_2 = b_2 \quad \Rightarrow \quad y_2 = \frac{b_2 - l_{21}y_1}{l_{22}}$$

$$\cdots\cdots\cdots\cdots$$

that is

$$y_k = \frac{b_k - \sum_{j=1}^{k-1} l_{kj} y_j}{l_{kk}} \quad , \quad k = 2,3,....,n$$

and similarly the second equation can solved as

$$x_k = \frac{y_k - \sum_{j=k+1}^{n} u_{kj} x_j}{u_{kk}} \quad , \quad k = n-1, n-2, ...,1 \quad \text{with} \quad x_n = \frac{y_n}{u_{nn}}.$$

Thus the main issue is how to make a LU decomposition. Noting that

$$a_{ij} = \sum_{k=1}^{n} l_{ik} u_{kj} \quad , \quad i,j = 1,2,....,n,$$

For the first column vector, we have

$$\begin{Bmatrix} a_{11} \\ a_{21} \\ : \\ a_{n1} \end{Bmatrix} = \begin{Bmatrix} l_{11}u_{11} \\ l_{21}u_{11} \\ : \\ l_{n1}u_{11} \end{Bmatrix}$$

and then if we set up $u_{11} = 1$, we can determine the first column of ***L***. Similarly, the first row becomes

$$\begin{Bmatrix} a_{11} & a_{12} & ... & a_{1n} \end{Bmatrix} = \begin{Bmatrix} l_{11}u_{11} & l_{11}u_{12} & ... & l_{11}u_{1n} \end{Bmatrix}$$

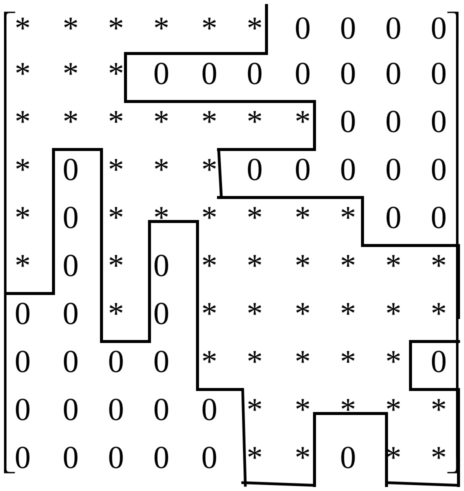
and then we can determine the first row of ***U***. For the second column and row, we have

$$\begin{Bmatrix} a_{12} \\ a_{22} \\ : \\ a_{n2} \end{Bmatrix} = \begin{Bmatrix} l_{11}u_{12} \\ l_{21}u_{12} + l_{22}u_{22} \\ : \\ l_{n1}u_{12} + l_{n2}u_{22} \end{Bmatrix}$$

and

$$\begin{Bmatrix} a_{21} & a_{22} & ... & a_{2n} \end{Bmatrix} = \begin{Bmatrix} l_{21}u_{11} & l_{21}u_{12} + l_{22}u_{22} & ... & l_{21}u_{1n} + l_{22}u_{2n} \end{Bmatrix}.$$

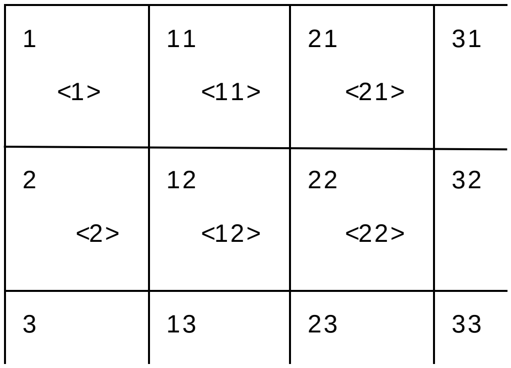
By setting $u_{22} = 1$, we can determine the second column of ***L*** and second row of ***U***. Continuing this process step by step, yields the LU decomposition. If we examine this procedure, it can be found that zeros outside the profile limit (that is called the skyline) shown in the figure, would not make any contribution to the Crout algorithm, that is, they need not be stored and can be skipped for their operation.



$$\begin{bmatrix} * & * & * & * & * & * & 0 & 0 & 0 & 0 \\ * & * & * & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ * & * & * & * & * & * & * & 0 & 0 & 0 \\ * & 0 & * & * & * & 0 & 0 & 0 & 0 & 0 \\ * & 0 & * & * & * & * & * & * & 0 & 0 \\ * & 0 & * & 0 & * & * & * & * & * & * \\ 0 & 0 & * & 0 & * & * & * & * & * & * \\ 0 & 0 & 0 & 0 & * & * & * & * & * & 0 \\ 0 & 0 & 0 & 0 & 0 & * & * & * & * & * \\ 0 & 0 & 0 & 0 & 0 & * & * & 0 & * & * \end{bmatrix}$$

This method has more advantage than the band method, since we need not keep some of zeros within the band of the coefficient matrix. Based on this algorithm, many skyline (or profile) methods have been developed for finite element analysis.

Both the band and skyline methods require to form the global stiffness matrix of the whole structure by assembling all of element stiffness matrices, and then after forming the coefficient matrix completely, the Gaussian elimination algorithm is applied. Therefore, we cannot reduce storage space too much because of the size of the global stiffness matrix which is roughly speaking, *mn*, where *m* is the band width and *n* is the size of the stiffness matrix, respectively. The wave front method is the one which utilizes the special structure of the finite element approximation, and is the one which may not require complete formation of the global stiffness matrix (that is, the coefficient matrix). It eliminates the terms (components of the coefficient matrix) whenever they can be eliminated before assembling the rest of finite element stiffness matrices, but at the moment they are assembled. In other words, it eliminates the terms “detached” from the rest of the structure while assembling. For example, node 1 is only used by element <1>, and then the terms related to node 1 can be eliminated at the time of “assembling” of element <1>, since the components of the global stiffness matrix related to node 1 are the same with the completely assembled global stiffness matrix. Similarly, node 2 is sheared with element 1 and element 2, and then at the stage of assembling the element stiffness matrices up to element 2, the terms related to node can be eliminated. Repeating this elimination procedure while assembling, the front line of the elimination is moving like waves. It seems that the naming of the wave front, or frontal method reflects this nature. Since we do not form the global stiffness matrix in complete form before the Gaussian elimination procedure, it does not require the space for the global stiffness matrix. It only requires the sufficiently large space that can store the assembled stiffness matrix related to the elements on the wave front. Using this nature of the method, Iron could solve fairly large scale problems with relatively small core memory in computers. At the time Iron introduced his beautiful engineering idea to solve a system of linear equations, the success of the finite element method became quite sound. It is strongly recommended that Iron’s monumental work should be thoroughly studied to understand the role of the finite element method in computer technology.



Iteration Methods

Another way to eliminate the completely assembled global stiffness matrix to reduce the core memory and storage disk space, is application of iteration methods which requires only multiplication of the element stiffness matrices and the associated degrees of freedom. Noting that the global stiffness matrix is formed by assembling of the element stiffness matrices, we have the following relation

$$\boldsymbol{K}\boldsymbol{u} = \sum_{e=1}^{N_e} \boldsymbol{K}_e \boldsymbol{u}_e$$

where $\boldsymbol{K}$ is the global stiffness matrix, $\boldsymbol{u}$ is the global generalized displacement vector, $\boldsymbol{K}_e$ is the element stiffness matrix, $\boldsymbol{u}_e$ is the element generalized displacement vector, and N_e is the total number of elements. If a system of linear equations

$$\boldsymbol{K}\boldsymbol{u} = \boldsymbol{f}$$

is considered, it is equivalent to the following :

$$\Leftrightarrow \quad \boldsymbol{P}^{-1}(\boldsymbol{K}\boldsymbol{u} - \boldsymbol{f}) = \boldsymbol{0}$$

$$\Leftrightarrow \boldsymbol{u} - \boldsymbol{u} + \boldsymbol{P}^{-1}(\boldsymbol{K}\boldsymbol{u} - \boldsymbol{f}) = \boldsymbol{0}$$

$$\Leftrightarrow \boldsymbol{u} = \boldsymbol{u} - \boldsymbol{P}^{-1}(\boldsymbol{K}\boldsymbol{u} - \boldsymbol{f})$$

where P is an arbitrary invertible matrix, and is called the pre-conditioning matrix. Using this relation, we can expect the iteration scheme

$$\boldsymbol{u}^{(k+1)} = \boldsymbol{u}^{(k)} - \boldsymbol{P}^{-1}\left(\boldsymbol{K}\boldsymbol{u}^{(k)} - \boldsymbol{f}\right) = \boldsymbol{u}^{(k)} - \boldsymbol{P}^{-1}\left(\sum_{e=1}^{N_e} \boldsymbol{K}_e \boldsymbol{u}_e{}^{(k)} - \boldsymbol{f}\right)$$

for an initial guess $\boldsymbol{u}^{(0)}$. If this algorithm can lead convergence by iteration, we can find the solution of the system of linear equations. It is clear that this does not require the formation of the global stiffness matrix. Because of this nature, required core memory can be very small, and then it was very popular in the early 1960s. However, since it is iterated, its convergence was not necessarily guaranteed. Furthermore, estimation of the required iteration number was difficult. Because of such uncertainties in iteration methods, they are gradually replaced by the direct methods, especially by the wave front method in the finite element community in the 1970s.

They were, however, revived at the time of introduction of supercomputers in the 1980s which are specially design with vector and parallel processing. In order to take advantage of these specially designed computer architecture, we once again found that the iteration method is the best fit to these new type of computers, and computer scientists started showing capability to solve more than a million equations. At this moment, the most promising iteration method is regarded as the preconditioned conjugate gradient method (PCG method), convergence of which can theoretically be proven, and required number of iterations can also be reduced significantly by introduction of an appropriate pre-conditioning matrix. It is believed that the iteration method would be dominant for solving large scale problems involving more that a million equations. At this moment (1996), researchers are challenging to solve 10 to 20 million equations.

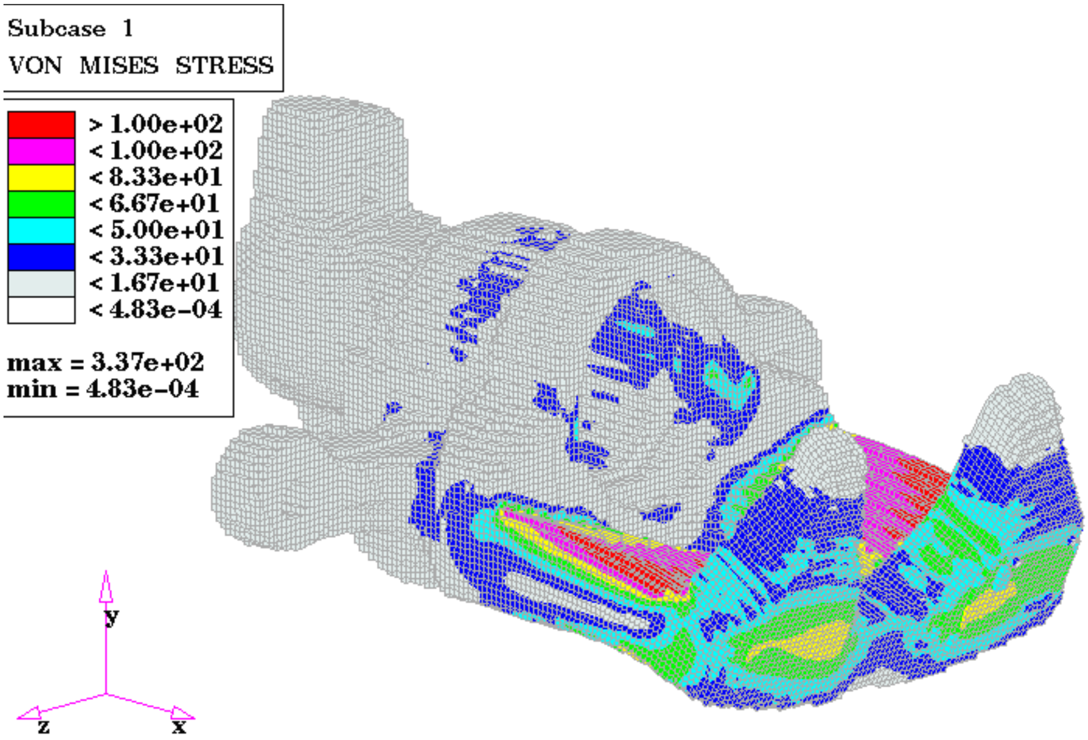


Figure X Application of a PCG method to FEA by S. Holister (Voxelcon Inc.) using about 150,000 Solid Elements (Approximately 450,000 degrees of freedom)

References

1. Irons, M.M., A frontal solution program, Int. J. Num. Meth. Eng., 2,5-32, 1970

2. Hinton, E., and Owen, D.R.J., Finite Element Programing, Academic Press, 1977

3. Bathe, K.J., and Wilson, E.L., Numerical Methods in Finite Element Analysis, Prentice-Hall, 1976

4. Felippa, C.A., Solution of linear equations with skyline-stored symmetric matrix, Comp. Struct., 5, 13-30, 1975

5. Hughes, T.J.R., Levit, I., and Winget, J., Element-by-element implicit algorithms for problems of structural and solid mechanics, Comp. Meth. Appl. Mech. Eng., 36, 241-54, 1983