APPENDIX 2
METHODS OF SOLVING LARGE SETS OF LINEAR EQUATIONS

Sets of equations occurring in the finite element method are characterised by large, rare and positive-definite matrices. Methods of solving sets of equations of such a type of matrices differ slightly from the ways of solving any other sets and all mentioned above methods have to consider ways of storing of matrices in the computer memory.

A.2.1. METHODS OF STORAGE OF STIFFNESS MATRICES

Not a very complex exercise on the use of the finite element method, for example a shell structure, generates a set of equations of the order of unknown parameters 1000÷10000. The quadratic matrix of this set of equations becomes a banded symmetric matrix with suitable numbering degrees of freedom (there are very complex procedures of numbering of degrees of freedom using the graph theory). Hence only half of this band is enough to be memorised in order to make the reconstruction of the whole information written in the stiffness matrix of a structure possible.

The simplest method of saving computer memory is recording the upper or lower matrix half bands in the rectangular table shown in Fig.A2.1.

It changes the location of matrix elements in the table so that elements from the main diagonal are located in the first column of the band and, for example, the element which originally was in the row i and the column j is still in the same row but in the column k. The new value of a column index should be calculated on the basis of a simple relation:

$k = j\text{-}i\text{+}1$

before getting a necessary component. Thus, we have $B_{ik}=A_{ij}$ for $j \geq i$. The half band width p for typical matrices is usually smaller by one order of value than the dimension n. Hence the lower triangle of the table **B** which is always „empty", does not have any particular significance for saving the core memory.

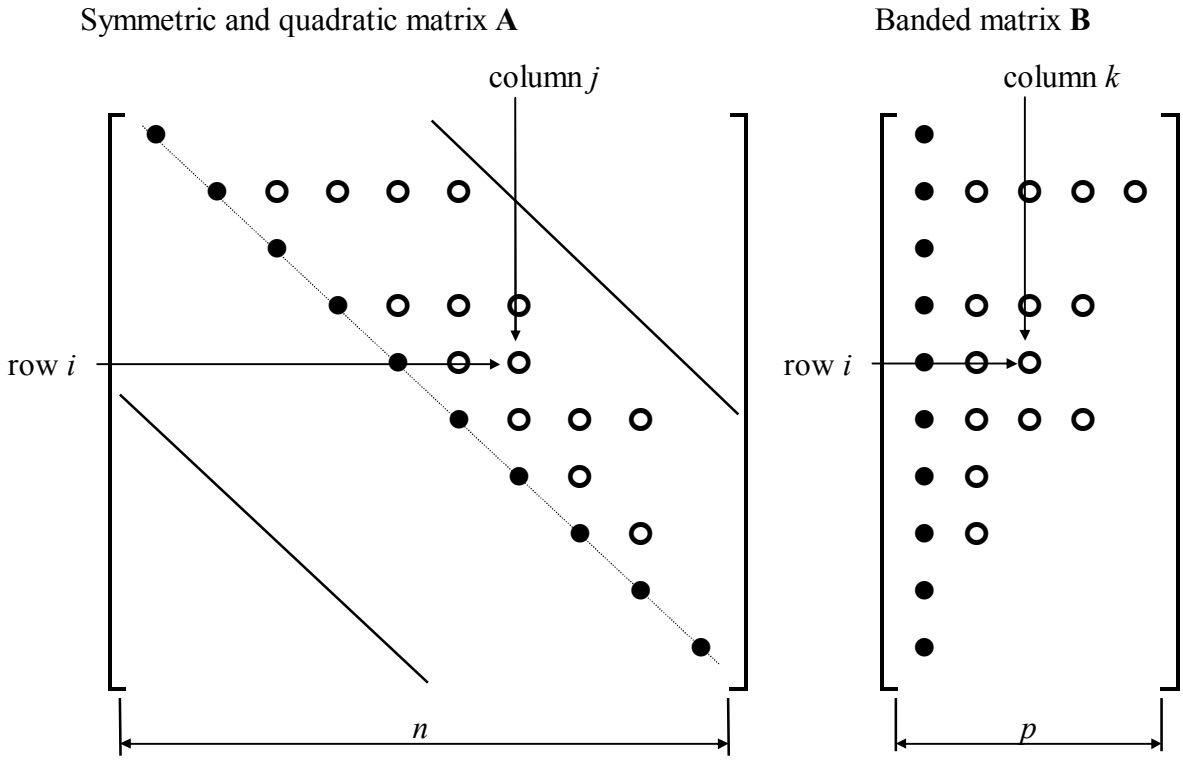


Fig.A2.1

Another economical method is *the sky-line method* which depends on memorising only these parts of rows (or columns) of the upper or lower half band which lie between the main diagonal and the last non-zero elements of the table (Fig.A2.2).

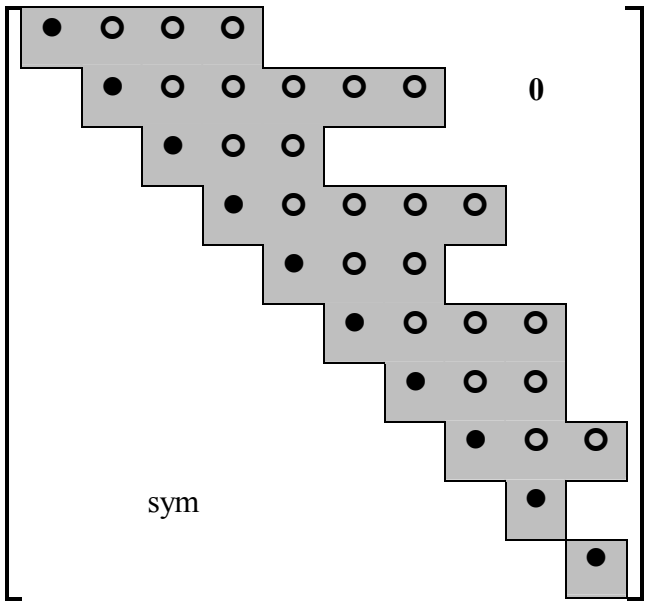
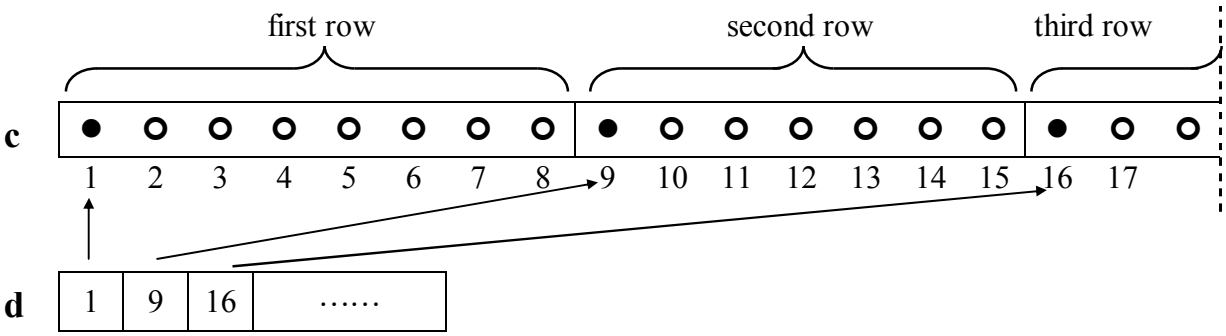


Fig.A2.2

The memorising area is shaded in Fig.A2.2.

Such a method of storing a matrix is possible thanks to the fact that non-zero elements of a triangular matrix never appear in areas lying behind the final non-zero components in rows when the decomposition of the matrix takes place. It is very important because procedures which memorise the matrix **L** in the same table in which the stiffness matrix has been memorised are usually applied to the FEM algorithm. The irregular shapes of the area shown in Fig.A2.2 prevent arranging data in the form of a two-dimensional table. Thus, two one-dimensional tables (vectors) are applied to the sky-line method. One of them stores real numbers which are components of a matrix and the other one stores indeces of the first terms of the successive rows of the matrix (Fig.A2.3).



$A_{ij} = C_k, \;\; k = \mathbf{d}[i]+j-i$

Fig.A2.3

This method is widely applied though it requires fairly complex operations while building a matrix and solving a set of equations (continual calculation of indeces) because it ensures very effective exploitation of the computer memory.

A.2.2. THE GAUSS ELIMINATION METHOD

The Gauss elimination method (in different variants) is one of the most often applied methods of solving sets of linear equations of the type $\mathbf{A}\,\mathbf{x} = \mathbf{y}$ where the matrix **A** is quadratic and singular.

We start solving it from the transformation of the first equation:

$$x_1 = \frac{1}{A_{11}}\left(y_1 - \sum_{k=2}^{n} A_{1k} x_k \right)$$

and the insertion of so determined unknown into other equations. It causes the elimination of the first column in the equations 2 to n (Fig.A2.4).

$$\left[\begin{array}{c|c} 1 & \mathbf{A}_{1k}^{(1)}\ \mathbf{A}_{2k}^{(1)}\ \ldots\ldots \\ \hline \mathbf{0} & \mathbf{A}^{(1)} \end{array}\right]\left[\begin{array}{c} x_1 \\ \hline \mathbf{x}^{(1)} \end{array}\right]\left[\begin{array}{c} y_1/A_{11} \\ \hline \mathbf{y}^{(1)} \end{array}\right]$$

Fig.A2.4. A set of linear equations after the first elimination.

We repeat this operation for the matrix $\mathbf{A}^{(1)}$ with dimensions (n-1)x(n-1) obtaining the matrix $\mathbf{A}^{(2)}$ with dimensions (n-2)x(n-2), etc. We carry on transformations as long as we obtain an equation with one unknown parameter:

$A_{nn}^{(n-1)} x_n = y_n^{(n-1)}$,

from which we determine x_n.

We can say that the Gauss elimination depends on such transformation of a matrix of a set of linear equations which leads to building a set of equations with an upper triangular matrix:

$$\mathbf{A}\,\mathbf{x} = \mathbf{y} \xrightarrow{\text{the Gauss elimination}} \mathbf{U}\,\mathbf{x} = \mathbf{y}^*,$$

which we solve by applying the back substitution method described in Appendix 1. The cost of the Gauss method is equal to $n^3/3$ and can really be proved (comp. [2]) that a cheaper algorithm cannot be found.

While eliminating unknown parameters the division operation by the diagonal component of the matrix **A** continually appears in those transformations. It can happen that $A_{ii}^{(k)}$ will be equal to zero or close to zero even for a nonsingular matrix. It can prevent obtaining the solution or leads to serious numerical errors. Such a situation can be avoided by conducting the elimination process in a different order. The change in the choice order of unknown parameters for the elimination enables to find such a diagonal component which is the biggest one in the matrix $A^{(k)}$ and to minimise the number of numerical errors.

The variant of the Gauss elimination with the choice of a middle element is called the Gauss-Jordan method. It enables to obtain a solution with an insignificant error even for slightly conditioned sets of equations, that is for sets with the determinant of the matrix **A** close to zero.

Part of the source code (in the PASCAL language) solving sets of linear equations (the Gauss procedure) presented in the following section is an example of the realisation of the Gauss algorithm.

A.2.3. THE GAUSS-SEIDEL ITERATIVE METHOD

The Gauss-Seidel iterative method is based on the assumption that the diagonal components of a matrix are considerably larger than components lying behind the diagonal. Thanks to it we can calculate

$$x_1 = \frac{1}{A_{11}}\left(y_1 - \sum_{k=2}^{n} A_{1k} x_k \right),$$

with the initial assumption that $x_k = 0$ for $k = 2 \ldots n$. We repeat this approximation for other unknown values:

$$x_i = \frac{1}{A_{ii}}\left(y_i - S_{iL} - S_{iR} \right),$$

where S_{iL} is the sum of all products of terms lying on the left side of x_i and suitable unknown values and S_{iR} is the sum of products of terms lying the right side of x_i and suitable unknown values:

$$S_{iL} = \sum_{k=1}^{i-1} A_{ik} x_k \ ,$$

$$S_{iR} = \sum_{k=i+1}^{n} A_{ik} x_k \ .$$

Successive approximation of unknown values done by this method is concurrent when a set of equations is well conditioned, which means that terms lying on the diagonal are larger than components lying behind it. The stiffness matrices of the finite element method are built in such a way. The Seidel modification of this method depends on the consideration of current unknown values while the iteration m which signifies the sum S_{iL} is calculated using unknown parameters during the iteration m, and the sum S_{iR} is calculated on the basis of unknown values determined in the previous iteration (m-1):

$$x_i^{(m)} = \frac{1}{A_{ii}}\left(y_i - S_{iL}^{(m)} - S_{iR}^{(m-1)} \right),$$

where $S_{iL}^{(m)} = \sum_{k=1}^{i-1} A_{ik} x_k^{(m)}$, $S_{iR}^{(m-1)} = \sum_{k=i+1}^{n} A_{ik} x_k^{(m-1)}$, $x_k^{(m)}$ is the value of the unknown x_k determined in the iteration m.

After every iterative step we calculate the difference $\Delta_i^{(m)} = x_i^{(m)} - x_i^{(m-1)}$ which allows to check the concurrence of the process. Iterations can be broken when $\mathrm{Max}\left(\left|\Delta_i\right|\right) < \varepsilon$, which means that the biggest difference is smaller than the permissible error of calculation. For large sets of equations we can often obtain the solution of a set of equations by the Gauss-Seidel method faster than by using the closed method (for example the Gauss-Jordan method).

A.2.4. THE AITKEN OVERRELAXATION METHOD

We note in the Gauss-Seidel iterative process that

$$x_i^{(m)} = x_i^{(m-1)} + \Delta_i^{(m)},$$

where the unknown value approaches the exact value with the step $\Delta_i^{(m)}$. Aitken noted that velocity of the process can be increased (that is, the number of necessary iterations can be decreased) if we calculate

$$x_i^{(m)} = x_i^{(m-1)} \omega\, \Delta_i^{(m)},$$

where ω is a overrelaxation coefficient. The value of this coefficient should be fitted on the basis of numerical experiments and it should be contained within the range $\langle 1.0 \div 2.0 \rangle$. Our calculations show that for the static problem of a 3D truss the optimal value of the overrelaxation coefficient is equal to 1.26.

A.2.5. OTHER METHODS OF SOLVING LARGE SETS OF EQUATIONS

Sets of equations of the finite element method are very often solved by methods depending on matrix decomposition, for example, the Banachewicz-Cholesky method presented in Appendix 1. The cost of this method is proportional to $n^3/6$ for the full symmetric matrix and it is equal to $np^2/6$, where p is the half band width of the matrix for banded matrices used in FEM problems.

Apart from the Banachewicz-Cholesky method some other methods of decomposition are also applied, for example, the Crout method consisting in splitting the matrix **A** into three matrices:

$$\mathbf{A} = \mathbf{L}\,\mathbf{D}\,\mathbf{L}^{\mathrm{T}},$$

where **D** is the diagonal matrix which means that it contains non-zero components only on the main diagonal. Such a type of distribution is not as unique as the Banachewich-Cholesky distribution, thus, the diagonal components of the matrix **L** are chosen so that they are equal to 1. The Crout decomposition is often applied to solving FEM problems, and particularly in

nonlinear problems where the stiffness matrix is not always positive-definite. In this case the Banachewicz-Cholesky method leads to the formation of the matrix **L** with complex numbers. It results from the fact that diagonal terms are calculated there by extracting roots. In the Crout method we always obtain a matrix with real components [1], [2] .

The Crout decomposition leads to the following relation:

$$D_{ii} = A_{ii} - \sum_{k=1}^{i-1} L_{ik}^2 D_{kk}$$

$$D_{ij} = 0 \text{ for } j \neq i,$$

$$L_{ij} = 0 \text{ for } j > i,$$

$$L_{ii} = 1.0,$$

$$L_{ij} = \frac{1}{D_{jj}} \left(A_{ij} - \sum_{k=1}^{j-1} L_{ik} L_{jk} D_{kik} \right) \text{ for } j < i,$$

The cost of matrix decomposition by the Crout method is proportional to $n^3/6$ for full matrices similarly to the cost of the process by the Banachewicz-Cholesky method.