CHAPTER 7

Ordering for Bandwidth, Profile and Frontwidth Optimisation

7.1 INTRODUCTION

The analysis of many problems in engineering mechanics involves the solution of a set of linear equations of the form,

$$\mathbf{Ax} = \mathbf{b}, \tag{7-1}$$

where **A** is a symmetric, positive definite and usually very sparse matrix. For large structures encountered in practice, 30-50% of the computer execution time may be devoted to solving these equations. This figure may rise to about 80% in non-linear, dynamic or structural optimisation problems.

Different methods can be used for the solution of the system of equations, of which Gaussian elimination is the most popular among structural analysts, since it is simple and has produced some very satisfactory error bounds.

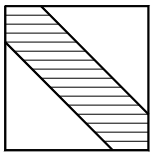
In the forward course of elimination, new non-zero entries may be created, but the back substitution does not lead to any new non-zero elements. It is beneficial to minimize the total number of such non-zero elements created during the forward course of the Gaussian elimination in order to reduce the round-off errors and the computer storage. Matrix **A** can be transformed by means of row and column operations to a form which leads to the creation of a minimum number of non-zero entries during the forward course of the elimination. This is equivalent to the "a priori" determination of permutation matrices **P** and **Q**, such that:

$$\mathbf{PAQ} = \mathbf{G}. \tag{7-2}$$

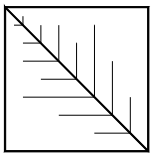
When **A** is symmetric and positive definite, it is advantageous to have **G** also symmetric so that only the non-zero elements on and above the diagonal of **G** need to be stored, and only about half as many arithmetic operations are needed in the elimination. The diagonal elements of **A** and **G** are the same, only in different positions. In order to preserve symmetry, **P** is taken as **Q** so that Eq. (7-2) becomes:

$$\mathbf{Q}^{t}\mathbf{A}\mathbf{Q} = \mathbf{G}. \tag{7-3}$$

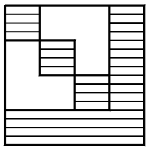
Some of the desirable forms of **G** are shown in Figure 7.1, consisting of the banded form, variable banded form, doubly bordered block diagonal form and doubly bordered banded form:

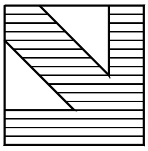


(a) Banded form.



(b) Variable banded form.





(c) Doubly bordered block diagonal form.(d) Doubly bordered banded form.

Fig. 7.1 Different forms for matrix **G**.

For transforming a symmetric matrix **A** into the above forms, various methods are available, some of which will be described in this chapter. However, due to the simplicity of the banded form, most of the material presented will be confined to optimising the bandwidth of the structural matrices, and other forms will be introduced briefly.

7.2 BANDWIDTH OPTIMISATION

In the Gaussian elimination method, the time required to solve the resulting equations by the banded matrix technique is directly proportional to the square of

the bandwidth of **A**. As mentioned before, the solution of these equations forms a large percentage of the total computational effort needed for the structural analysis. Therefore it is not surprising that a lot of attention is being paid to the optimisation of the bandwidth of these sparse matrices. A suitable ordering of the elements of a kinematical basis for a structure reduces the bandwidth of **A**, hence decreasing the solution time, storage and round-off errors. Similarly, ordering the elements of a statical basis results in the reduction of the bandwidth of the corresponding flexibility matrix of the structure.

Iterative methods using different criteria for the control of the process of interchanging rows and columns of **A** are described by Akyuz and Utka [2], Alway and Martin [3], Rosen [207] and Grooms [66]. For these methods, in general, the required storage and CPU time can be high, making them uneconomical.

The first direct method for bandwidth reduction was recognized by Harary [70] in 1967, who posed the following question:

> *For a graph S with N(S) nodes, how can labels 1,2, ... ,N(S) be assigned to nodes in order to minimize the maximum absolute value of the difference between the labels of all pairs of adjacent nodes?*

For a graph labelled in such an optimum manner, the corresponding adjacency matrix will have unit entries concentrated as closely as possible to its main diagonal.

In structural engineering, Cuthill and McKee [34] developed the first graph-theoretical approach for reducing the bandwidth of stiffness matrices. In their work, a level structure is used which is called a "spanning tree" of a structure. The author´s interest in bandwidth reduction was initially motivated by his interest in generating and ordering the elements of cycle bases and generalized cycle bases of a graph, as defined in Chapter 6, in order to reduce the bandwidth of the flexibility matrices, Ref. [89]. For this purpose, a *shortest route tree* (SRT) has been used. The application of this approach has been extended to the elements of a kinematical basis (cutset basis), in order to reduce the bandwidth of stiffness matrices. Subsequently, it has been noticed that there is a close relation between Cuthill-McKee´s level structure and the author´s SRT. However, there is a difference between these trees in that an SRT contains additional information about the connectivity properties of the corresponding structure.

Further improvements have been achieved by employing special types of SRTs such as the longest and narrowest ones, Ref. [100]. Generation of a suitable SRT depends on an appropriate choice of starting node. Kaveh [89] used an end node of an arbitrary SRT, which was chosen from its last contour (level) having the least valency. Gibbs et al. [57] employed a similar node and called it a pseudo-peripheral node. Cheng [25] used an algebraic approach to select a single node or

a set of nodes as the root of an SRT. Kaveh employed two simultaneous SRTs for selecting a pseudo-peripheral node. A comparison of six different algorithms was made in Ref. [103]. Algebraic graph theory has also been used for finding a starting node, Kaveh [101] Grimes et al. [65], Kaveh and Rahimi Bondarabady [121,122,127,128].

Extensions and applications of the nodal numbering algorithms to element ordering for bandwidth, profile and frontwidth optimisations are due to Kaveh [101,102], Akhras and Dhatl [1], Everstine [44], Razzaque [201], Pina [192], Fenves and Law [49], Sloan and Randolph [219], Sloan [220], Burgess and Lai [17], and excellent books on these topics are those of Duff et al. [39] and Pissanetsky [193].

7.3 PRELIMINARIES

A matrix **A** is called *banded*, when all its non-zero entries are confined within a band formed by diagonals parallel to the main diagonal. Thus $A_{ij} = 0$ when $|i - j| > b$ and $A_{k,k-b} \neq 0$ or $A_{k,k+b} \neq 0$ for at least one value of k. b is the half-bandwidth and 2b+1 is known as the *bandwidth* of **A**. As an example, for

$$\mathbf{A} = \begin{bmatrix} 1 & 6 & \cdot & \cdot & \cdot \\ 6 & 2 & 7 & 8 & \cdot \\ \cdot & 7 & 3 & 9 & \cdot \\ \cdot & 8 & 9 & 4 & \cdot \\ \cdot & \cdot & \cdot & \cdot & 5 \end{bmatrix}, \tag{7-4}$$

the bandwidth of **A** is 2b+1=2×2+1=5.

A banded matrix can be stored in different ways. The *diagonal storage* of a symmetric banded n×n matrix **A** is an n×(b+1) matrix **AN**. The main diagonals are stored in the last column, and lower co-diagonals are stored down-justified in the remaining columns. As an example, **AN** for the above matrix is:

$$\mathbf{AN} = \begin{bmatrix} \cdot & \cdot & 1 \\ \cdot & 6 & 2 \\ 0 & 7 & 3 \\ 8 & 9 & 4 \\ 0 & 0 & 5 \end{bmatrix}. \tag{7-5}$$

When **A** is a sparse matrix, this storage scheme is very convenient, since it provides direct access, in the sense that there is a simple one-to-one correspondence between the position of an entry in the matrix **A**(i,j) and its position in **AN**(i,j – i+b+1).

Obviously, the bandwidth depends on the order in which the rows and columns of **A** are arranged. This is why iterative techniques seek a permutation of the rows and a permutation of columns to make the resulting bandwidth small. For symmetric matrices, identical permutations are needed for both the rows and the columns. When a system of linear equations has a banded matrix of coefficients and the system is solved by the Gaussian elimination, with pivots being taken from the diagonals, all the operations are confined to the band and no new non-zero entries are generated outside the band. Therefore, the Gaussian elimination can be carried out in place, since a memory location is already reserved for any new non-zero that might be introduced within the band.

When a banded matrix of high order has a wide band and a large number of zeros inside it, the diagonal storage may become wasteful. Then an *envelope* (variable band) scheme of Jennings [85], the so-called *skyline* (Felippa [50]), may be used. For each row i of a symmetric matrix **A**, define

$$b_i = i - j_{min}(i), \tag{7-6}$$

where j_{min} (i) is the minimum column index in row i for which $A_{ij} \neq 0$. Therefore, the first non-zero of row i lies b_i positions to the left of the diagonal, and b is defined as:

$$b = \max (b_i). \tag{7-7}$$

The *envelope* of **A** is the set of elements A_{ij} such that $0 < i - j \leq b$. For a certain row i, all elements with column indices in the range $j_{min}(i)$ to i – 1 belong to the envelope, a total of b_i elements. Diagonal elements do not belong to the envelope. The profile of **A** is the number of elements in the envelope, i.e.

$$\text{Profile}\,(\mathbf{A}) = \sum_i b_i. \tag{7-8}$$

In Jennings´s storage scheme, all elements which belong to the envelope are stored row by row, including zeros, in a one-dimensional array, say **AN**. Diagonal elements are stored at the end of each row. The length of **AN** is equal to Profile (**A**) + n. An array of pointers **IN**, the entries of which are pointers to the locations of the diagonal elements in **AN**, is also necessary. Thus, the elements of row i, when i > 1, are in positions **IN**(i – 1)+1 to **IN**(i). The only element of row 1 is A_{11}, stored in **AN**(1). The elements have consecutive, easily calculable column indices.

For example, the matrix of the previous example, has a profile equal to 4, and its envelope storage is:

$$\begin{array}{lllllllllll} \text{Position} & = & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \\ \mathbf{AN} & = & [1 & 6 & 2 & 7 & 3 & 8 & 9 & 4 & 5] \\ \mathbf{IN} & = & [1 & 3 & 5 & 8 & 9] \end{array} \qquad (7\text{-}9)$$

A variant of Jennings´s scheme is obtained when the transpose of the lower envelope is stored. In this case elements are stored column-wise, and since the columns of the matrix retain their length, the scheme is often termed *skyline storage*. The profile of a matrix also changes if the rows and columns are permuted.

Another useful concept used in the design of storage schemes for symmetric matrices is wavefront or frontwidth. This method has application in the finite element method when a frontal solution is employed, Irons [84,121,149,158,215].

With reference to equation (7-1), column j is said to be *active* at stage i if $j \geq i$ and there is a non-zero entry in column j with a row index, k, such that $k \leq i$. Letting f_i denote the number of columns that are active at stage i, the *maximum frontwidth* of **A** is given by:

$$F_{max} = \max_{1 \leq i \leq n} \{f_i\}. \qquad (7\text{-}10)$$

The *root-mean-squared* (r.m.s.) *wavefront* is defined as:

$$\tilde{F} = \left(\frac{1}{n}\sum_{i=1}^{n} f_i^2\right)^{\frac{1}{2}} \qquad (7\text{-}11)$$

Assuming that n and the average value of f_i are reasonably large, it can be shown that a complete profile or front factorization requires approximately $O(n\tilde{F}^2)$ operations.

7.4 PATTERN EQUIVALENCE OF STIFFNESS AND CUTSET ADJACENCY MATRICES

In previous chapters, it has been shown that the stiffness matrix **K** of a structure is pattern equivalent to the cutset basis adjacency matrix $\mathbf{N} = \mathbf{L}\mathbf{L}^t$, where **L** is the cutset basis incidence matrix of the structural model S. Similarly, the flexibility

matrix **G** is pattern equivalent to the cycle basis adjacency matrix $\mathbf{D} = \mathbf{C}\mathbf{C}^t$, where **C** is the cycle basis incidence matrix of S.

Reducing the bandwidths of **N** and **D** directly influences those of **K** and **G**, respectively. Notice that the dimensions of **N** and **D**, for general space structures, are six-fold smaller than those of **K** and **G**, and therefore more simple to optimise.

For stiffness analysis there exists a special cutset basis whose elements correspond to stars of its nodes except for the ground node (cocycle basis). The adjacency matrix of such a basis naturally is the same as that of the node adjacency matrix of S with the row and column corresponding the datum node being omitted. In this chapter such a special cutset basis will be considered, and the nodes of S will be ordered such that the bandwidth of its node adjacency matrix is reduced to the smallest possible amount.

Let **A** be the adjacency matrix of a graph S. Let i and j be the nodal numbers of member k, and let $\alpha_k = |i - j|$. Then the bandwidth of **A** can be defined as,

$$b(\mathbf{A}) = 2\text{Max}\{\ \alpha_k\text{: k=1,2, ...,M(S)}\} + 1, \tag{7-12}$$

where M(S) is the number of members of S. In order to minimize the bandwidth of **A**, the value of b(**A**) should be minimized.

Papademetrious [185] has shown that the bandwidth minimization problem is an NP-complete problem. Therefore, any approach to it is of interest primarily because of its heuristic value.

7.5 A SHORTEST ROUTE TREE AND ITS PROPERTIES

A shortest route tree rooted at a node, called the *starting node* (root) of the tree, has the following properties:

The path from any node to the root through the tree is a shortest path. An algorithm for generating an SRT has already been described in Chapter 1 and therefore only its properties relevant to nodal number are discussed here.

An SRT decomposes (partitions) the node set of S into subsets according to their distance from the root. Each subset is called a *contour* (level) of the SRT, denoted by C_i. The contours of an SRT have the following properties:

$$\text{Adj}\,(C_i) \subseteq C_{i-1} \cup C_{i+1} \quad 1 < i < m$$

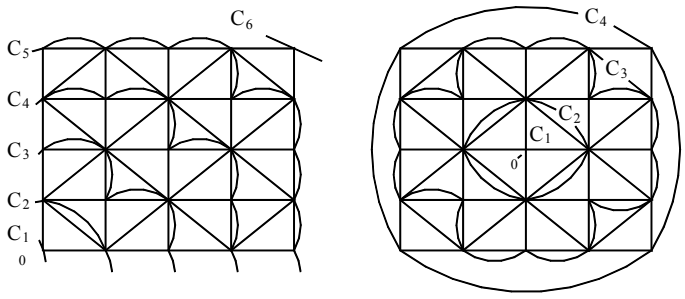
$$\text{Adj}\,(C_1) \subseteq C_2 \tag{7-13}$$

$$\text{Adj}\,(C_m) \subseteq C_{m-1}.$$

The number of nodes in each contour is called the *width* of that contour, and the largest width of the contours of an SRT is called the *width of the SRT* rooted at the starting node n_i. This number is known as the *width number* of n_i. The number of contours of an SRT (except the starting node contour) is the *height* of the tree. The *longest* SRT is the one with maximal height and the *narrowest* SRT is the one with minimal width.

As an example, an SRT of S as shown in Figure 7.2(a), rooted at O, has the following identities:

$W(C_1) = 1$, $W(C_2) = 2$, $W(C_3) = 5$, $W(C_4) = 7$, $W(C_5) = 9$, and $W(C_6) = 1$, leading to Width (SRT) = 9 and Length (SRT) = 5.



(a) An SRT rooted at O. (b) An SRT rooted at O'.

Fig. 7.2 A graph S and two of its SRTs.

The same graph model and an SRT rooted at O', as shown in Figure 7.2(b), leads to Width (SRT) = 16 and Length (SRT) = 3.

From this simple example one can realize the importance of selecting an appropriate starting node. This will be discussed in some detail in subsequent sections.

7.6 NODAL ORDERING FOR BANDWIDTH REDUCTION

A four-step algorithm is used for nodal ordering of structural models leading to banded stiffness matrices, in which the nodes of their models S have been ordered by:

Step 1: Finding a suitable starting node.

Step 2: Decomposing the node set of S into ordered subsets (contours).

Step 3: Selecting a connected path (transversal) containing one representative node from each contour.

Step 4: Ordering the nodes within each contour, to obtain the final nodal numbering of S.

All the above steps require the use of an SRT algorithm of Chapter 6. Therefore, a nodal ordering process may be considered as a multiply applied SRT algorithm.

The node set of S can be decomposed into ordered subsets by means of a breadth-first-search algorithm. The quality of the results depends upon the choice of an appropriate starting node, as the root of this tree. The results corresponding to the ordering within each contour, however, also depend upon the use of a suitable transversal containing one representative node from each contour.

Methods for finding suitable starting nodes have been developed by Cheng [25], Kaveh [89,103], Gibbs et al. [57], and Grimes et al. [65]. In the following, various topological and algebraic graph-theoretical methods are presented for finding good starting nodes and selecting suitable transversals.

7.6.1 A GOOD STARTING NODE

The algorithms described in this section are in two groups. The first group uses topological graph theory (TGT), and the second group employs algebraic graph theory (AGT).

Some Concepts of Algebraic Graph Theory: The standard definitions from TGT were described in Chapter 1, and only new terms and those of AGT are given here.

The *distance* $d(n_i,n_j)$ between two nodes n_i and n_j is defined to be the length of the shortest path between these nodes. The *eccentricity* of a node n_i is defined as:

$$e(n_i) = \text{Max } d(n_i,n_j) \quad \text{for} \quad j=1,...,N(S). \tag{7-14}$$

The *diameter* of S is defined as:

$$\delta(S) = \text{Max } e(n_i) \quad \text{for} \quad i=1,...,N(S). \tag{7-15}$$

As an example, the eccentricity of n_2 in Figure 7.3 is $e(n_2) = 3$, and the diameter of S is $\delta(S) = 4$.

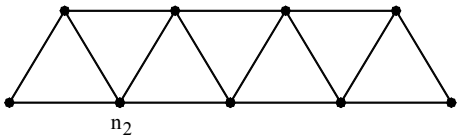


Fig. 7.3 A graph S.

A node n_i of S is called *peripheral*, if its eccentricity is the same as the diameter of S, i.e. $\delta(S) = e(n_i)$. If the eccentricity is close to the diameter, then n_i is called a *pseudo-peripheral node* or a *good starting node*.

Consider the node adjacency matrix **A** of S. Let

$$\mathbf{Q} = \mathbf{A} + \mathbf{I}, \tag{7-16}$$

where **I** is an N(S)×N(S) identity matrix. The eigenvalues of **Q** are one bigger than those of **A**, and the eigenvectors of **Q** are exactly the same as those of **A**. Matrix **Q** is real and symmetric, and it can easily be shown that all the entries of $\mathbf{Q}^k$ are positive; thus it is primitive and, according to the Perron-Frobenius theorem from matrix algebra:

i) λ_1 is real and positive, and a simple root of the characteristic equation;

ii) $\lambda_1 > |\lambda|$ for any eigenvalue $\lambda \neq \lambda_1$;

iii) λ_1 has a unique eigenvector $\mathbf{W}_1$ which may be taken to have all positive entries.

As $\mathbf{W}_i$ is the eigenvector corresponding to λ_i, $\mathbf{Q}\mathbf{W}_i = \lambda_i\mathbf{W}_i$ for i=1, ... ,N(S). Multiplying the two sides with **Q**, one obtains $\mathbf{Q}\mathbf{Q}\mathrm{W}_i = \lambda_i\mathbf{Q}\mathrm{W}_i = \lambda_i^2\mathrm{W}_i$. Repeating this process results in $\mathbf{Q}^k\mathrm{W}_i = \lambda_i^k\mathrm{W}_i$. Now consider any vector **x** not orthogonal to $\mathbf{W}_1$, as:

$$\mathbf{x} = \alpha_1\mathbf{W}_1 + \alpha_2\mathbf{W}_2 + \ldots + \alpha_{N(S)}\mathbf{W}_{N(S)} \qquad \alpha_1 \neq 0. \tag{7-17}$$

Multiplying the two sides with $\mathbf{Q}^k$, and using $\mathbf{Q}^k W_i = \lambda_i^k W_i$ for i=1, ... ,N(S) leads to:

$$\mathbf{Q}^k\mathbf{x} = \lambda_1^k \alpha_1\mathbf{W}_1 + \lambda_2^k \alpha_2\mathbf{W}_2 + \ldots + \lambda_{N(S)}^k \alpha_{N(S)}\mathbf{W}_{N(S)}, \tag{7-18}$$

and, as $k\to\infty$, we have,

$$\mathbf{Q}^k\mathbf{x}/\lambda_1^k = \alpha_1\mathbf{W}_1 + (\lambda_2/\lambda_1)^k\alpha_2\mathbf{W}_2 + \ldots + (\lambda_{N(S)}/\lambda_1)^k\alpha_{N(S)}\mathbf{W}_{N(S)} \to \alpha_1\mathbf{W}_1, \tag{7-19}$$

since λ_1 is the eigenvalue of strictly largest modulus, and (λ_i/λ_1) is less than unity and approaches to zero when $k \to \infty$. In other words, the ratios of the components of $\mathbf{Q}^k\mathbf{x}$ approach the ratios of the components of $\mathbf{W}_1$ as k increases.

Let $\mathbf{v} = \{1,1,...,1\}^t$, then the *i*th component of $\mathbf{Q}^k\mathbf{v}$ is the number of walks of length k beginning at an arbitrary node of S and ending at n_i. If n_i is a good starting node (peripheral node), this number will be smaller. Thus for $k\to\infty$, one should obtain some average number, defined as the *accessibility index* by Gould [64], which indicates how many walks go on average through a node. With a suitable normalization, $\mathbf{Q}^k\mathbf{v}$ converges to the largest eigenvector $\mathbf{W}_1$ of $\mathbf{Q}$, Straffing [226].

The following algorithms are designed to obtain a good starting node of a graph:

Algorithm A (a TGT approach)

Step 1: Construct an SRT on each node of S and select the narrowest SRT. In this algorithm the width numbers of all nodes are calculated. These numbers may further be used in the process of ordering.

Obviously, such an approach can be expensive for large structures, which defeats its application.

Algorithm B (a TGT approach)

Step 1: Start with a node of the least valency and form an SRT.

Step 2: Record all the nodes of the last contour of the selected SRT.

Step 3: Form SRTs rooted at each of the recorded nodes and choose the one which corresponds to the narrowest SRT. The process of constructing an SRT is terminated as soon as the width of one of its contours exceeds the width of the previously selected SRT.

Algorithm C (a TGT approach)

Step 1: Start from an arbitrary node of S. Construct an SRT on this node and take a node of least valency from its last contour.

Step 2: Form a new SRT from the selected node, and then repeat Steps 2 and 3 of Algorithm B and choose the starting node corresponding to the narrowest SRT.

Algorithm D (a TGT approach)

Step 1: Form an SRT on each node of S and select the narrowest SRT. Unlike Algorithm A, the construction of an SRT is terminated as soon as its width exceeds the width of the previously selected SRT.

This algorithm is more economical than Algorithm A; however, it does not provide the information that Algorithm A reveals, for subsequent use.

Algorithm E (a TGT approach)

Step 1: Start with an arbitrary node, form an SRT on this node and take a node n_i of least valency from its last contour.

Step 2: Generate an SRT on n_i and find all nodes contained in its even, first and last contours.

Step 3: Generate an SRT on each node of these contours, and find the narrowest one. The process of formation of an SRT is terminated as soon as the width of one of its contours exceeds the width of the previously selected SRT. Denote the selected node by n_j.

Step 4: Check adjacent nodes to n_j for possible reduction in width, to decide the final starting node.

Algorithm F (a TGT approach)

Step 1: From an arbitrary node generate an SRT, and from its last contour select a node X_1 of minimal valency. Observe the width of the selected SRT.

Step 2: Generate an SRT from X_1 and select X_2 of the least valency from its last contour, and observe the width.

Step 3: Generate two SRTs simultaneously rooted at X_1 and X_2 and find the node X_3 which is the last node of S included in one of the SR subtrees. Once X_3 is found, terminate the process of forming SRTs. Generate an SRT from X_3 and observe its width. X_1 and X_2 are called the *generators* of X_3.

Step 4: Repeat the process of Step 3, using the pairs (X_1,X_3) and (X_2,X_3) as the generators to find X_4 and X_5, respectively. Construct the corresponding SRTs and observe their widths.

Step 5: Repeat the process of Step 3 for X_i (i = 3, 4, ...) together with the corresponding generator, until no further improvement in width is observed. The narrowest SRT should be selected for nodal decomposition of S.

An example of the application of this algorithm is depicted in Figure 7.4, where a cross-shaped grid S is considered. Starting from an arbitrary node "O", an SRT is generated and X_1 is obtained from its last contour. Generating a new SRT from X_1, node X_2 is chosen from its last contour. X_3 is the result of generating two simultaneous SRTs from X_1 and X_2. Using (X_1,X_3) and (X_2,X_3), nodes X_4 and X_5 are obtained, respectively. The width of the selected SRTs rooted at X_1,X_2,X_3,X_4 and X_5 are 8,8,8,11, and 10, respectively. Therefore the process is terminated and X_3 is taken as a good starting node of S.

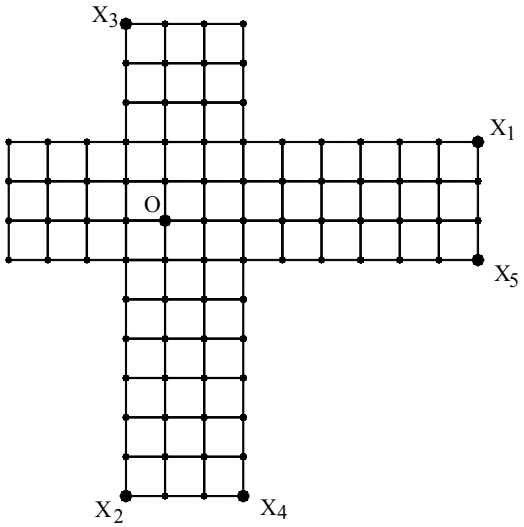


Fig. 7.4 A cross-shaped grid and the selected X_i (i=1,...,5) by Algorithm F.

Algorithm G (an AGT approach)

Step 1: Calculate the dominant eigenvector $\mathbf{W}_1 = \{W_1, W_2, ..., W_{N(S)}\}^t$ of matrix **Q**.

Step 2: Find Min W_i in $\mathbf{W}_1$. The node corresponding to this component is taken as a good starting node of S.

For calculating the dominant eigenvector $\mathbf{W}_1$ of **Q**, an iterative method is used, which starts with $\mathbf{v} = \{1,1, ...,1\}^t$ and calculates $\mathbf{Q}^v$. This vector is then normalized and multiplied by **Q**. This process is repeated until the difference between two consecutive eigenvalues, obtained from $\mathbf{Qv} = \lambda\mathbf{v}$, is reduced to a small value which, for example, can be taken as 10^{-3}.

The author has compared the efficiency of the above algorithms through some randomly generated graphs. A comparison of the time and bandwidth for the generated examples may be found in Kaveh [103].

7.6.2 PRIMARY NODAL DECOMPOSITION

Once a good starting node is selected, an SRT is constructed and its contours $\{C_1, C_2, ..., C_m\}$ are obtained. These subsets are now ordered according to their distances from the selected starting node. Obviously, many SRTs can be constructed on a node. Although all lead to the same nodal decompositions, different transversals will be obtained for different SRTs. Thus, in the generation process, the nodes of each contour C_i are considered in ascending order of their valencies (or entries in eigenvector $\mathbf{W}_1$) for selecting the nodes in C_{i+1}, in order to provide the conditions for the possibility of generating a minimal (or optimal) transversal as defined in the next section. Finding an optimal transversal before an SRT is fixed seems to be a time-consuming problem. However, for most of the models encountered in practice, optimal transversal is in between minimal ones. In the following, two algorithms are given for selecting a suboptimal transversal of an SRT.

7.6.3 TRANSVERSAL P OF AN SRT

A *transversal* of an SRT, is defined as a connected path P containing one distinct node N_i from each contour C_i of the SRT. A *minimal transversal* is the one for which $\sum_{i=1}^{m} \deg(N_i)$ is minimum. An *optimal transversal* is the one leading to the best nodal numbering, i.e. a numbering corresponding to smallest bandwidth for the selected decomposition. The weight of a node is defined as its degree or its

value W_i in $\mathbf{W}_1$, depending upon the use of a TGT approach or an AGT method, respectively.

Algorithm AA (a TGT approach)

Step 1: Take a node N_m of minimal weight from the last contour C_m of the selected SRT.

Step 2: Find N_{m-1} from C_{m-1} which is connected to N_m by a branch of the SRT.

Step 3: Repeat the process of Step 2, selecting nodes N_{m-2}, N_{m-3}, ... , N_1, as the representative nodes of the contours C_{m-2} , C_{m-3} , ... ,C_1, respectively.

The above algorithm is a backtracking process from a node of minimal weight in the last contour C_m, and selects a transversal $P = \{N_1, N_2, ... ,N_m\}$ which can now be used for ordering the nodes of the contours of the corresponding SRT.

ALGORITHM BB (an AGT approach)

Let $C_1,C_2,...,C_m$ be the selected contours of the SRT, and correspondingly put these subsets in $\mathbf{W}_1$ into a similar order, i.e.

$$\mathbf{W}_1 = \{W(C_1),W(C_2), ...,W(C_m)\}, \tag{7-20}$$

where $W(C_i)$ contains the entries of W_1 corresponding to the nodes of C_i. Now the algorithm can be described as follows:

Step 1: Label the root as N_1 and assign W_i of this node as its new weight denoted by $\overline{W_i}$.

Step 2: Calculate the new weight $\overline{W_i}$ of each node of C_2 by adding the W_i′s from $W(C_2)$ to $\overline{W_1}$.

Step 3: Repeat the process of Step 2, calculating $\overline{W_i}$ for each node of C_3,C_4, ... ,C_m.

Step 4: Take a node N_m of minimal weight from the last contour C_m of the selected SRT.

Step 5: Find N_{m-1} from C_{m-1}, which is connected to N_m by a branch of the SRT.

Step 6: Repeat the process of Step 5, selecting N_{m-2},N_{m-3}, ... ,N_1 as the representative nodes of the contours C_{m-2},C_{m-3}, ... ,C_1.

$P = \{N_1, N_2, ..., N_m\}$ forms a suboptimal transversal of the selected SRT.

In general, the results of the Algorithm BB have been better than those of Algorithm AA, at the expense of some additional computer time. As an example, for the grid model of Figure 7.5, a suboptimal transversal of the SRT rooted at X_3 is shown in bold lines.

7.6.4 NODAL ORDERING

Step 1: Number N_1 as "1".

Step 2: N_2 is given number "2" and an SR subtree is generated from N_2, numbering the nodes of C_2 in the order of their occurrence in this SR subtree.

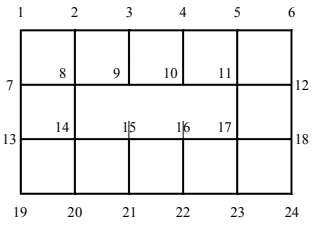
Step 3: The process of Step 2 is repeated for numbering the nodes of C_3, C_4, ... , C_m, sequentially using N_3, N_4, ... ,N_m as the starting nodes of SR subtrees, until all the nodes of S are numbered.

Now the numbering can be reversed, in a way similar to that of the Reverse Cuthill-McKee algorithm, for possible reduction of fill-ins in the process of Gaussian elimination, which will be discussed in Section 7.9.

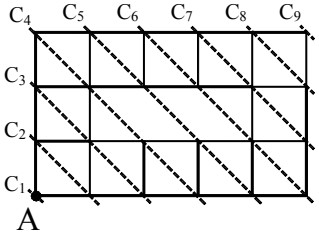
7.6.5 EXAMPLES

The following two simple examples are chosen to illustrate the steps of the presented approaches, but the applications are by no means limited to such simple cases.

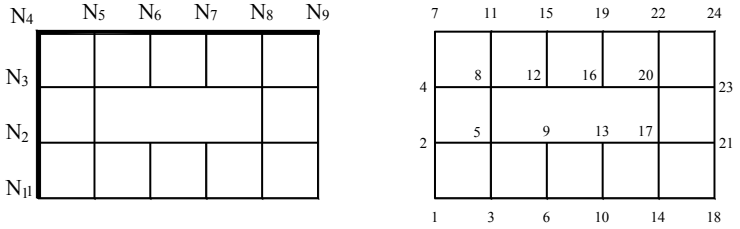
Example 1: Let S be the graph model of a truss structure, as shown in Figure 7.5(a). Using a TGT algorithm, a starting node A is found, and the corresponding SRTs are depicted in Figure 7.5(b). A transversal is selected as shown in bold lines, Figure 7.5(c). Then nodes are numbered contour by contour, employing the representative nodes as the starting nodes of SR subtrees, Figure 7.5(d).



(a) Initial numbering of S.



(b) The selected SRT.



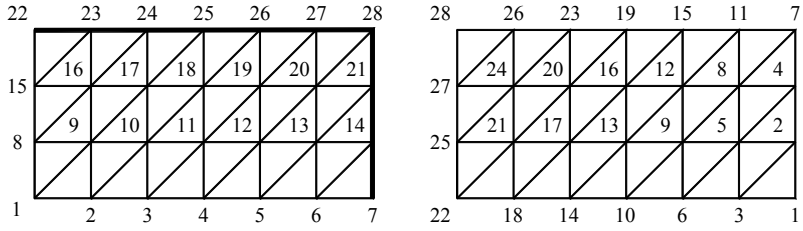
(c) The selected transversal P. (d) Final nodal numbering of S.

Fig. 7.5 Graph model S of Example 1.

Example 2: S is the model of a grid with uniform valency distribution, as shown in Figure 7.6(a). Using algorithm G, the following dominant eigenvector is obtained for matrix **Q** of S, in which for simplicity only four digits are provided:

$\mathbf{W}_1$ = {0.3344, 0.5298, 0.6161, 0.5951, 0.4791, 0.3011, 0.1180, 0.3972, 0.7432, 0.9540, 1.0000, 0.8786, 0.6183, 0.2875, 0.2875, 0.6183, 0.8786, 1.000, 0.9540, 0.7432, 0.3972, 0.1180, 0.3011, 0.4791, 0.5951, 0.6160, 0.5298, 0.3344}^{t}.

Thus node "7" is selected as a good starting node. An SRT is generated from this node and, using Algorithm BB, a transversal P = {7,14,21,28,27,26,25,24,23,22} is selected, which is shown in bold lines in Figure 7.6(a). Final nodal numbering is illustrated in Figure 7.6(b).



(a) Initial numbering and the selected transversal. (b) Final numbering.

Fig. 7.6 The graph model S of Example 2.

7.7 A CONNECTIVITY COORDINATE SYSTEM FOR NODAL ORDERING

In order to cast the concepts developed for nodal ordering in a mathematical form, a connectivity coordinate system is defined for nodal numbering of S, Kaveh [106]. Separate study of planar and space graphs results in clarification of further interesting points about nodal numbering of space structures, as described in the following.

7.7.1 A CONNECTIVITY COORDINATE SYSTEM FOR PLANAR GRAPHS

Let S be a connected planar graph. Select a good starting node "O" of S as the root of an SRT (analogous to the centre of a coordinate system). Generate an SRT of S rooted at O and find a suboptimal transversal P of the SRT (analogous to the x-axis). Now any node n_i of S can be associated with two integers, namely (r_i,d_i), where r_i is the shortest distance of n_i from O (distance of the contour C_i from O to which n_i belongs), and d_i is the shortest distance of n_i from the representative node $N_i \in C_i$ (analogous to (x_i,y_i) in a Cartesian coordinate system).

An order can now be defined for the nodes of S as,

$$n_i\ (r_i,d_i) < n_j(r_j,d_j), \tag{7-21}$$

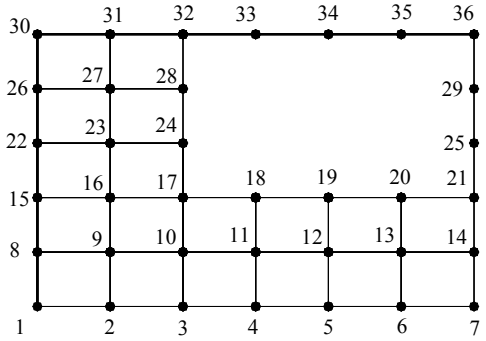
if (a) $r_i < r_j$;

(b) $r_i = r_j$ and $d_i < d_j$;

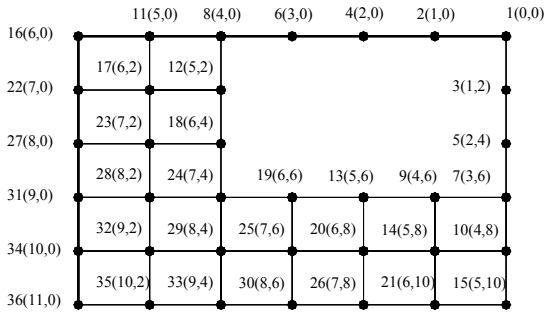
(c) $r_i = r_j$, $d_i = d_j$ and n_i is in a shorter distance from the nearest ordered node than n_j;

(d) otherwise n_i and n_j are arbitrarily ordered.

As an example, a grid structure shown in Figure 7.7(a) is considered.



(a) Primary nodal ordering and the selected transversal.



(b) The connectivity coordinate system and final nodal ordering.

Fig. 7.7 A grid structural model and its connectivity coordinate system.

Using the dominant eigenvector for matrix **Q** of S, node "36" is selected as a good starting node and P = {36,35,34,33,32,31,30,26,22,15,8,1} is chosen as a suboptimal transversal of the selected SRT. The connectivity coordinates of the nodes are shown in Figure 7.7(b), according to which the nodes of S are ordered.

7.7.2 A CONNECTIVITY COORDINATE SYSTEM FOR SPACE GRAPHS

The connectivity coordinate system defined in the previous section can be generalized to space graphs. This requires further refinement of the nodal ordering algorithm and leads to better results at the expense of additional comparisons.

Generate an SRT of S from the selected starting node "O" and find a suboptimal transversal P of the SRT rooted at O. Denote the contours of this tree by C_1,C_2, ... ,C_m, and their representative by nodes N_1,N_2, ... ,N_m, respectively. Generate one SR subtree rooted at N_i (i=1, ..., m), spanning the nodes of C_i. Select subtransversals P_m,P_{m-1}, ... ,P_1 as follows:

Construct a transversal of the SR subtree rooted at N_m spanning the nodes of C_m, and denote it by P_m. This can be achieved by a backtracking approach using the SR subtree generated from N_m. Obviously, a transversal P_m may be a disconnected path in C_m (connected in S), and will be referred to as a *subtransversal.* Then select P_{m-1} of C_{m-1}, which has its nodes adjacent to, or in a small distance from the nodes of P_m. Repeat this process until subtransversals for all the contours are obtained. Now, to every node of S three integers can be assigned, denoted by n_i (r_i,d_i,g_i), where r_i is the distance of n_i from O, d_i is the distance of n_i from N_i, and g_i is the distance of the node n_i from subtransversal P_i. An order is defined for the nodes of S as,

$$n_i\ (r_i,d_i,g_i) < n_j(r_j,d_j,g_j), \tag{7-22}$$

if (a) $r_i < r_j$;

(b) $r_i = r_j$ and $d_i < d_j$;

(c) $r_i = r_j$, $d_i = d_j$ and $g_i < g_j$;

(d) $r_i = r_j$, $d_i = d_j$, $g_i = g_j$ and n_i is at a nearer distance than n_j to an already numbered node;

(e) otherwise n_i and n_j are arbitrarily ordered.

As an example, consider a simple space frame as shown in Figure 7.8(a). The ground node is selected as a starting node, and the transversal P is obtained for the corresponding SRT, as shown in bold lines, Figure 7.8(b). The contours are simply the nodes of different storeys of the structure, for each of which subtransversals are selected as depicted by the dashed lines. The final ordering is given in Figure 7.8(c).

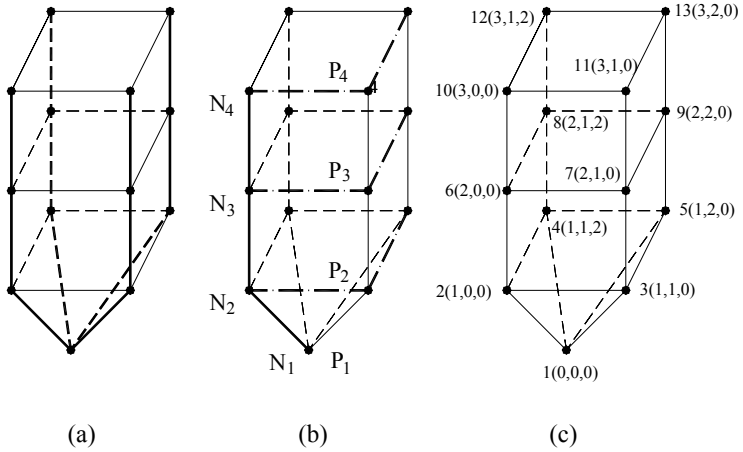


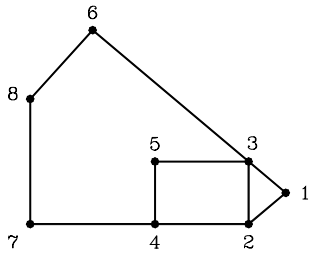
Fig. 7.8 A space structure and its connectivity coordinate system.

7.8 NODAL NUMBERING FOR PROFILE REDUCTION

Nodal numbering algorithms can also be applied to profile reduction. As mentioned before, after nodal numbering for bandwidth reduction, by reversing the ordering, a numbering corresponding to a much smaller profile can be found. This has been found by George [56] and proved by Liu and Sherman [163]. The

method is known as the *Reverse Cuthill-McKee algorithm*. For the Cuthill-McKee type of ordering, the bandwidth remains unchanged when the order is reversed; however, the profile can never increase.

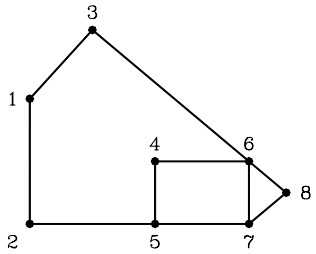
As an example, consider a nodal numbering for a graph as shown in Figure 7.9(a) with corresponding adjacency matrix **A**, in Figure 7.9(b). Reversing the nodal numbers as in Figure 7.9(c), leads to a matrix $\mathbf{A}'$ as depicted in Figure 7.9(d), with a reduction of the profile from 15 to 13.



(a) A nodal numbering.

	1	2	3	4	5	6	7	8
1	*	*	*	.	.	.	.	.
2	*	*	*	*	.	.	.	.
3	*	*	*	.	*	*	.	.
4	.	*	.	*	*	.	*	.
5	.	.	*	*	*	.	.	.
6	.	.	*	.	.	*	.	*
7	.	.	.	*	.	.	*	*
8	.	.	.	.	.	*	*	*

(b) Matrix **A**.



(c) Reverse of the nodal numbering of (a).

	1	2	3	4	5	6	7	8
1	*	*	*	.	.	.	.	.
2	*	*	.	.	*	.	.	.
3	*	.	*	.	.	*	.	.
4	.	.	.	*	*	*	.	.
5	.	*	.	*	*	.	*	.
6	.	.	*	*	.	*	*	*
7	.	.	.	.	*	*	*	*
8	.	.	.	.	.	*	*	*

(d) Matrix $\mathbf{A}'$.

Fig. 7.9 A Reverse Cuthill-McKee for nodal numbering.

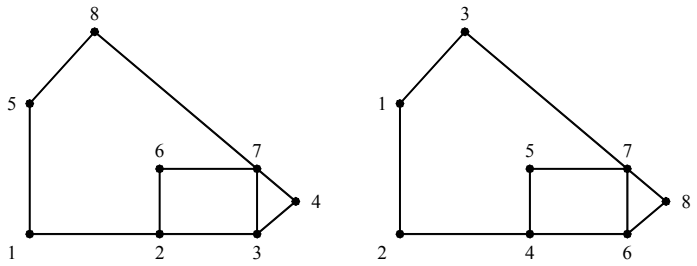
Another important algorithm for profile reduction is that of King [138], which operates as follows:

Take a node of minimum valency and number it "1". The set of nodes is now divided into three subsets, A, B and C. The subset A consists of nodes already numbered. The subset B is defined as B = Adj (A); i.e. it consists of all nodes adjacent to any node of A. C contains the remaining nodes. Then, at each step number the node of subset B which causes the smallest number of nodes of subset C to be transferred to subset B, and redefine A, B and C, accordingly.

Example: Consider a graph S with original nodal numbering as in Figure 7.10(a).

Take node "5" as a starting node and number it as "1". Then:

A = {5}, B = {1,8} and C = {the remaining nodes}.



(a) (b)

Fig. 7.10 An example of numbering by King´s algorithm.

At this stage, 1 and 8 are the next candidates. If 1 is taken to A, then 2 will come to B, and for 8, node 7 will join B. Therefore, arbitrarily, 1 is taken to A and numbered as "2". Now we have:

A = {5,1}, B = {8,2} and C = {the remaining nodes}.

From new candidates 8 and 2, naturally 8 will be selected because it brings only 7 to B, while 2 brings 3 and 6. Therefore 8 is numbered as "3". This process is continued until the nodal numbering of Figure 7.10(b) is obtained, which corresponds to a profile equal to 14.

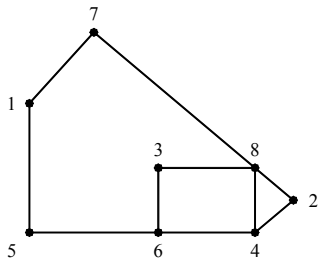
7.9 GRAPH-THEORETICAL INTERPRETATION OF GAUSSIAN ELIMINATION

Let **A** be a symmetric sparse matrix of order N and let S be the corresponding graph. Suppose that Gaussian elimination by columns is performed on **A** until the factorization $\mathbf{A} = \mathbf{U}^t\mathbf{D}\mathbf{U}$ is obtained. At the beginning of the *k*th step, all non-zeros in columns 1,2, ... ,k−1 below the diagonal, have been eliminated. Multiples of the *k*th row are then subtracted from all rows which have a non-zero in column k below the diagonal. On performing this operation, new non-zero entries may be introduced in row k+1, ... ,N to the right of column k. Cancellations may also occur, producing new zeros, but this is rare in practice and will be neglected.

Consider the active submatrix at the kth step (an active submatrix contains all elements $A_{ij}^{(k)}$ with i,j ≥ k). Let S^k be the graph associated with the active submatrix. S^k is called *an elimination graph*, Parter [186]. The nodes of this graph are N–k+1 last numbered nodes of S. S^k contains all members connecting those nodes which were present in S, and additional members corresponding to fill-ins produced during the k–1 initial elimination steps. The sequence of $S=S^1$, S^2, S^3, ... can be obtained using the following rule:

To obtain S^{k+1} from S^k, delete node k and add all possible members between nodes, which are adjacent to node k in S^k.

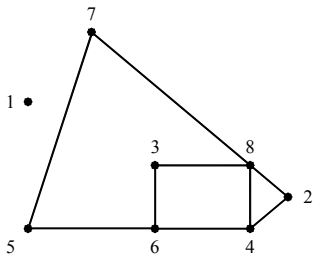
As an example, consider a graph S and the corresponding adjacency matrix, as shown in Figure 7.11. Two steps of the Gaussian elimination and the corresponding elimination graphs are also illustrated.



(a) $S=S^1$.

$$\mathbf{A}^1 = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \\ 8 \end{array} \begin{array}{c} \begin{array}{cccccccc} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \end{array} \\ \begin{bmatrix} * & \cdot & \cdot & \cdot & * & \cdot & * & \cdot \\ \cdot & * & \cdot & * & \cdot & \cdot & \cdot & * \\ \cdot & \cdot & * & \cdot & \cdot & * & \cdot & * \\ \cdot & * & \cdot & * & \cdot & * & \cdot & * \\ * & \cdot & \cdot & \cdot & * & * & \cdot & \cdot \\ \cdot & \cdot & * & * & * & * & \cdot & \cdot \\ * & \cdot & \cdot & \cdot & \cdot & \cdot & * & * \\ \cdot & * & * & * & \cdot & \cdot & * & * \end{bmatrix} \end{array}$$

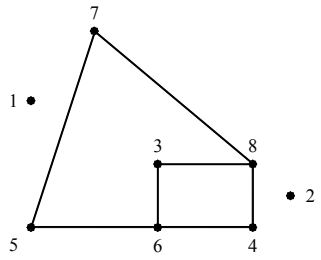
(b) Matrix $\mathbf{A}^1$.



(c) S^2.

$$\mathbf{A}^2 = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \\ 8 \end{array} \begin{array}{c} \begin{array}{cccccccc} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \end{array} \\ \begin{bmatrix} * & \cdot & \cdot & \cdot & * & \cdot & * & \cdot \\ \cdot & * & \cdot & * & \cdot & \cdot & \cdot & * \\ \cdot & \cdot & * & \cdot & \cdot & * & \cdot & * \\ \cdot & * & \cdot & * & \cdot & * & \cdot & * \\ \cdot & \cdot & \cdot & \cdot & * & * & \otimes & \cdot \\ \cdot & \cdot & * & * & * & * & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \otimes & \cdot & * & * \\ \cdot & * & * & * & \cdot & \cdot & * & * \end{bmatrix} \end{array}$$

(d) Matrix $\mathbf{A}^2$.



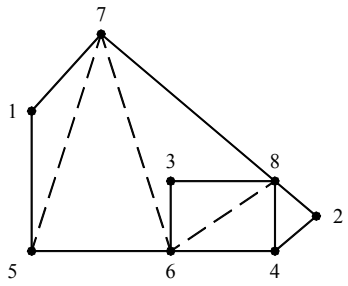
(e) S^3.

$$\mathbf{A}^3 = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \\ 8 \end{array} \begin{array}{c} \begin{array}{cccccccc} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \end{array} \\ \begin{bmatrix} * & \cdot & \cdot & \cdot & * & \cdot & * & \cdot \\ \cdot & * & \cdot & * & \cdot & \cdot & \cdot & * \\ \cdot & \cdot & * & \cdot & \cdot & * & \cdot & * \\ \cdot & \cdot & \cdot & * & \cdot & * & \cdot & * \\ \cdot & \cdot & \cdot & \cdot & * & * & \otimes & \cdot \\ \cdot & \cdot & * & * & * & * & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \otimes & \cdot & * & * \\ \cdot & \cdot & * & * & \cdot & \cdot & * & * \end{bmatrix} \end{array}$$

(f) Matrix $\mathbf{A}^3$.

Fig. 7.11 Illustration of two steps of Gaussian elimination.

Eliminating the rest of the nodes and considering a clique (a complete graph) between the nodes adjacent to each eliminated node (when such members are not present), matrix **U** is obtained. The structure of $\mathbf{U} + \mathbf{U}^t$ and the corresponding filled graph are shown in Figure 7.12.



(a) S^F.

$$\begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \\ 8 \end{array} \begin{array}{c} \begin{array}{cccccccc} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \end{array} \\ \begin{bmatrix} * & \cdot & \cdot & \cdot & * & \cdot & * & \cdot \\ \cdot & * & \cdot & * & \cdot & \cdot & \cdot & * \\ \cdot & \cdot & * & \cdot & \cdot & * & \cdot & * \\ \cdot & * & \cdot & * & \cdot & * & \cdot & * \\ * & \cdot & \cdot & \cdot & * & * & \otimes & \cdot \\ \cdot & \cdot & * & * & * & * & \otimes & \otimes \\ * & \cdot & \cdot & \cdot & \otimes & \otimes & * & * \\ \cdot & * & * & * & \cdot & \otimes & * & * \end{bmatrix} \end{array}$$

(b) Matrix $\mathbf{U}+\mathbf{U}^t$.

Fig. 7.12 The structure of $\mathbf{U}+\mathbf{U}^t$ and the corresponding graph.

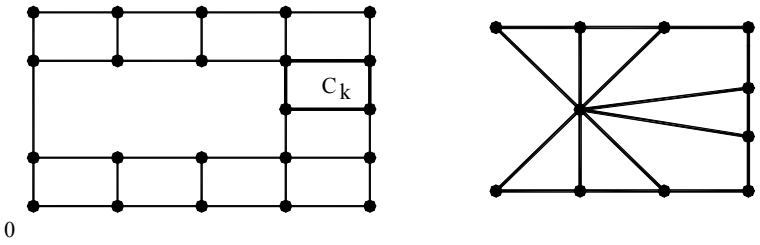
There are algorithms, which try to reduce the number of fill-ins caused by elimination. The minimum-degree algorithm of Tinney [237] is perhaps the best method for such a reduction. For brevity this will not be discussed here; the interested reader may refer to Tinney´s original paper.

7.10 ELEMENT ORDERING FOR BANDWIDTH OPTIMISATION OF FLEXIBILITY MATRICES

The elements of a generalized cycle basis (GCB), as defined in Chapter 6, must be ordered to obtain a banded flexibility matrix **G**. This is similar to ordering the elements of a cutset basis (nodal numbering) for reducing the bandwidth of the stiffness matrix **K**. This problem can be transferred to a nodal ordering algorithm by defining appropriate mathematical structures for the transformation of the connectivity properties. Two approaches for this problem are developed in the following:

7.10.1 AN ASSOCIATE GRAPH

An *associate graph* A(B(S)) of a generalized cycle basis B(S) of S is a graph whose nodes are in a one-to-one correspondence with the elements of B(S), and two nodes are connected if two elements of B(S) have at least one common member. As an example, the associate graph of the mesh basis in Figure 7.13(a) is depicted in Figure 7.13(b).



(a) A mesh basis B(S) of S. (b) The associate graph of B(S).

Fig. 7.13 A mesh basis and its associate graph.

A *weighted associate graph* can similarly be defined. For this graph, the nodes and members are assigned integer numbers. The *weight* of a node in A(B(S)) is taken as the number of members of the corresponding cycle in S, and the weight of a member $m_k = (n_i,n_j)$ in A(B(S)) is taken as the number of members of $C_i \cap C_j$, where C_i and C_j are the cycles of S corresponding to the nodes n_i and n_j of A(B(S)), respectively.

7.10.2 DISTANCE NUMBER OF AN ELEMENT

The *distance* d_i of a node n_i of S from a selected node O, is the length of the shortest path connecting n_i to O. The *distance number* of a cycle or a γ-cycle or an element C_k from O, is defined as one of the following:

(a) The distance of the nearest node of C_k from O, denoted by d_k^n.

(b) The distance of the furthest node of C_k from O, denoted by d_k^f.

(c) The mean value of d_k^n and d_k^f; i.e $| (1/2)(d_k^n + d_k^f) |$, where $|.|$ means the integer part of the number.

(d) The sum of $d_k^n + | (L(C_k)/2 |$, where $L(C_k)$ is the length of C_k.

(e) The mean value of the distance of the nodes of C_k; i.e. $\left| \sum_{i=1}^{L(C_k)} d_i / L(C_k) \right|$.

As an example, the values defined above for a cycle C_k are shown in bold lines in Figure 7.13(a), and with respect to a reference node O are 5, 6, 5, 7 and 5, respectively. For simplicity, only the integer parts of the divisions are considered.

Any of the definitions (a)-(e) can be used as the distance number of a cycle, a γ-cycle or an element of a finite element model (FEM).

7.10.3 ELEMENT ORDERING ALGORITHMS

In the following, two algorithms are presented for ordering the elements of a cycle basis, a GCB, an FEM or the substructures of a structure. However, for simplicity we will refer to a GCB only.

Algorithm A

Step 1: Order the nodes of S with a nodal numbering algorithm.

Step 2: Use the same starting node as in Step 1 to form an SRT and find the distance numbers of the elements of the GCB.

Step 3: Assign these distance numbers to the nearest (furthest or any other appropriate intermediate) nodes of the elements of the GCB. In this process a node

may become the representative node of p elements. Then p different (some may be equal) distance numbers will be assigned to the representative nodes.

Step 4: Order these nodes in ascending order of distance number. A node representing p elements will receive p different numbers. For equidistant nodes, the same sequence as the nodal numbering of Step 1 should be used.

Step 5: Order the elements of the GCB with the same numbers received by their representative nodes. This provides an efficient ordering for the elements of the GCB.

Algorithm B

Step 1: Construct the associate graph A(B(S)) of the GCB.

Step 2: Generate an SRT of S, starting from an appropriate node O, and find the distance numbers of the elements of the GCB.

Step 3: Assign these numbers to the nodes of A(B(S)), and order its nodes by a nodal numbering algorithm, with a starting node which corresponds to an element containing O .

Step 4: Reorder the nodes of A(B(S)) in ascending order of their distance numbers obtained in Step 2. For equidistant nodes, the same sequence as that obtained by the nodal numbering algorithm of Step 3 should be used.

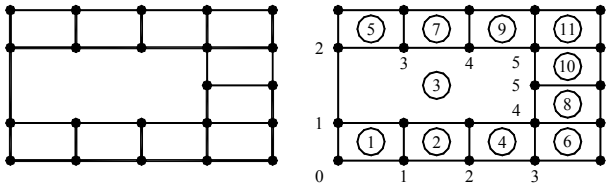
Step 5: Number the elements in the same order as that obtained for their representative nodes in A(B(S)). This leads to an efficient numbering of the elements of the considered GCB.

7.10.4 EXAMPLE

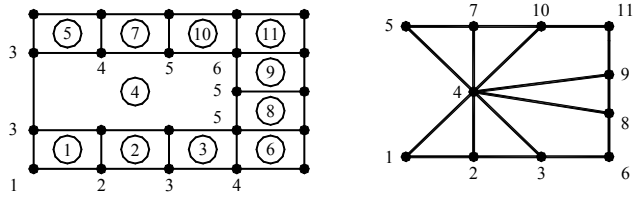
Let S be the model of a rigid-jointed planar frame. Suppose the selected cycle basis consists of the boundaries of the bounded regions of S (a mesh basis), Figure 7.13(a).

For Algorithm A, an SRT starting from O is generated, Figure 7.14(a), and the distance numbers of the cycles corresponding to definitions (a) and (e) of Section 7.10.2 are calculated and assigned to the representative nodes of the cycles. The nearest node of a cycle to O is taken as its representative node, Figures 7.14(b) and (c). These nodes are then ordered, leading to an ordered cycle basis. The bandwidths of the cycle adjacency matrices for these orderings are 15 and 13. The latter result can further be reduced to 11 by imposing additional restrictions in the

process of ordering. Since the frame is planar, the bandwidths of the corresponding flexibility matrices will be 45 and 39, respectively.



(a) An SRT of S. (b) Cycle ordering by definition (a).



(c) Cycle ordering by definition (e). (d) A(B(S)) and its nodal ordering.

Fig. 7.14 S, and ordering the elements of its cycle basis.

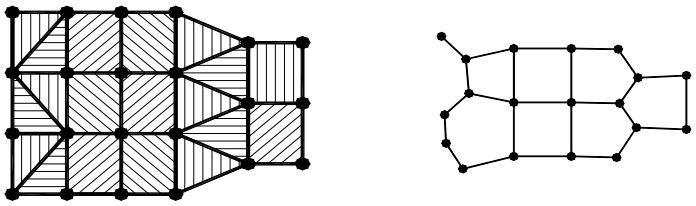
Algorithm B is also applied to this example. The associate graph A(B(S)) of the mesh basis is formed, Figure 7.14(d), and using definition (e) for distance number of the elements, the order of the nodes of A(B(S)) is obtained. The numbering of the cycles is shown in Figure 7.14(d), which corresponds to a bandwidth of 13 for its cycle adjacency matrix, and 39 for its flexibility matrix.

7.11 ORDERING FOR BANDWIDTH OPTIMISATION OF FINITE ELEMENT MESHES

For finite element nodal ordering, different methods are developed. The application of a natural associate graph, in a two-step approach, has been suggested by Kaveh [89] and Fenves and Law [49]. A corner node method is developed by Cassell et al. [21] and Kaveh and Ramachandran [117]. The application of an element clique graph is due to Sloan [218,220] and Livesley and Sabin [162]. A comparative study of the application of these graphs has been made by Kaveh and Behfar [120] Additional graphs for transforming the information concerning the connectivity of the FEM to those of different simple graphs are introduced and employed in efficient Finite element nodal numbering

by Kaveh and Roosta [116]. In the following, two important graphs associated with FEMs are defined. For other graphs and a detailed study, reference [111] can be consulted.

A *natural associate graph* can be defined in the same way as the associate graph of Section 10.4. In this graph, nodes correspond to the elements of the FEM and two nodes are connected if the corresponding elements have a common boundary. As an example, the natural associate graph of the FEM in Figure 7.15(a) is depicted in bold lines in Figure 7.15(b).

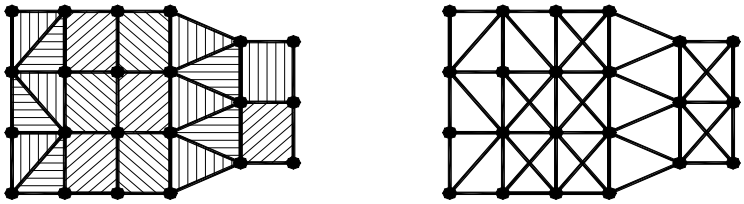


(a) A finite element model. (b) The natural associate graph of the model.

Fig. 7.15 An FEM and its natural associate graph.

Algorithms A and B of Section 7.10 can also be used for element ordering of an FEM, similarly to ordering the elements of a GCB. Once the elements are ordered, the nodal ordering within each element should be performed. Special considerations in specifying priority to various nodes of an element, such as mid-side nodes, mid-element nodes and corner nodes of different valencies, lead to a suitable nodal numbering of an FEM. This method has been developed by Kaveh [89,102] and applied successfully by Fenves and Law [49].

The *element clique graph* of an FE mesh, is a graph whose nodes are the same as those of the FE mesh and two nodes n_i and n_j of the element clique graph are connected with a member if n_i and n_j share an element of the FE mesh. A small FE mesh containing linear rectangular and triangular elements is depicted in Figure 7.16(a), and the element clique graph of this FE mesh is shown in Figure 7.16(b).



(a) A simple finite element mesh (b) The element clique graph of model (a)

Fig. 7.16 An FE mesh and its element clique graph.

Other mathematical models have been used for transforming the FE nodal ordering to graph nodal ordering by Kaveh [130] and Livesley and Sabin [162]. A comparative study of five such methods can be found in Kaveh and Behfar [120].

7.12 ORDERING USING ALGEBRAIC GRAPH THEORY

7.12.1 DEFINITIONS

Let S(N,M) be a graph with node set N, containing n nodes, and the member set M. The adjacency matrix $\mathbf{A} = [a_{ij}]_{n\times n}$ of the labelled graph S is defined as follows:

$$a_{ij} = \begin{cases} 1 & \text{if node } n_i \text{ is adjacent to } n_j \\ 0 & \text{otherwise} \end{cases}$$

The degree matrix $\mathbf{D} = [d_{ij}]_{n\times n}$ is a diagonal matrix of the node degrees; d_{ii} is equal to the degree of the *i*th node.

Consider the Laplacian matrix $\mathbf{L} = [l_{ij}]_{n\times n}$ of a graph as defined in chapter 1.

The *complementary Laplacian matrix* $\mathbf{L}_c$ of a graph is the same as the Laplacian of the complementary graph K_n/G, where K_n is the complete graph constructed on "n" nodes.

7.12.2 EIGENVALUES AND EIGENVECTORS OF MATRIX **A**

Consider an eigenproblem as,

$$\mathbf{A}\phi_i = \mu_i\phi_i\,, \tag{7-23}$$

where μ_i is the ith eigenvalue and ϕ_i is a corresponding eigenvector. If **A** is a symmetric real matrix, all its eigenvalues are real and can be expressed as:

$$\mu_1 \le \mu_2 \le \mu_3 \le \ldots \le \mu_{n-1} < \mu_n. \tag{7-24}$$

The largest eigenvalue μ_n is the single root of the characteristic equation of **A**. The corresponding eigenvector ϕ_n is the only eigenvector with all positive entries. This vector has attractive properties employed in geography and structural mechanics, Refs. [111, 25, 26].

Gould [64] appears to have introduced the first important application on using the properties of ϕ_n in calculating the accessibility index of the cities. The city with highest accessibility corresponds to the largest entry of ϕ_n. Further study and applications are due to Straffing [226] and Maas [166].

Grimes et al. [65] used the node with smallest accessibility as a pseudo-peripheral node, corresponding to the node with the least entry of ϕ_n. Kaveh [101] employed the properties of ϕ_n in the entire process of nodal ordering.

7.12.3 EIGENVALUES AND EIGENVECTORS OF MATRIX **L**

Consider the following eigenproblem,

$$\mathbf{L}\mathbf{v}_i = \lambda_i \mathbf{v}_i, \tag{7-25}$$

where λ_i is the *i*th eigenvalue and $\mathbf{v}_i$ is the corresponding eigenvector. As for **A**, all the eigenvalues of **L** are real. It can be shown that matrix **L** is a positive semi-definite matrix with:

$$0 = \lambda_1 \le \lambda_2 \le \ldots \le \lambda_{n-1} \le \lambda_n, \tag{7-26}$$

and

$$\mathbf{v}_1^t = \{1, 1, \ldots, 1\}. \tag{7-27}$$

The second eigenvalue λ_2 and the corresponding eigenvector $\mathbf{v}_2$ have attractive properties. Fiedler [51] has investigated various properties of λ_2. This eigenvalue which is positive for a connected graph is known as the *algebraic connectivity* of the graph, and $\mathbf{v}_2$ is known as the *Fiedler vector*. Mohar [175] has applied $(\lambda_2, \mathbf{v}_2)$ to different problems such as graph partitioning and ordering [7,218,122,128,135]. Paulino et al. [184] used $\mathbf{v}_2$ for element ordering and nodal numbering.

7.12.4 A HYBRID METHOD FOR ORDERING

In this method the advantages of both graph and algebraic graph methods are incorporated into an algorithm for ordering. In the algebraic graph method, general approaches are used to calculate the eigenvalues and eigenvectors, and the information available from the connectivity of their graph models is ignored. This is why the computational time and complexity of these algorithms are not low enough to compete with pure graph theory methods. In this section, graph parameters are used to increase the efficiency of the algebraic graph theory approaches. Typical graph parameters can be taken as the degrees of the nodes, the 1-weighted degrees of the nodes, distances of the nodes from two pseudo-peripheral nodes, and 2-weighted degrees of the nodes of the graph.

The algebraic graph theory method employed here is not the same as those employed in a general eigenproblem, but rather a specific method in which the valuable features of graph parameters are incorporated.

In the present method, the graph parameters are considered as Ritz vectors, and the first eigenvector of the complementary Laplacian matrix $\mathbf{L}_c$ (Fiedler vector) is considered as a linear combination of Ritz vectors [127]. The coefficients for these vectors are in fact the weights of the graph parameters, which are usually determined either by heuristic approaches or by experience.

Consider the following vector,

$$\bar{\phi} = \sum_{i=1}^{p} w_i \mathbf{v}_i, \tag{7-28}$$

where $\bar{\phi}$ is an approximation to the Fiedler vector, $\mathbf{v}_i$ (i=1,…,p) are the normalized Ritz vectors representing the graph parameters, and w_i (i=1,…,p) are the coefficients of the Ritz vectors (Ritz coordinates) which are unknowns, and p is the number of parameters being employed. Equation (7-28) can be written as,

$$\bar{\phi} = \mathbf{vw}, \tag{7-29}$$

where $\mathbf{w}$ is a p×1 vector and $\mathbf{v}$ is an N×p matrix containing the Ritz vectors.

Consider the eigenproblem of the complementary Laplacian as:

$$\mathbf{L}_c\phi = \rho\phi. \tag{7-30}$$

Approximating ϕ by $\bar{\phi}$ and multiplying by $\mathbf{v}^t$ results in,

$$\mathbf{v}^t\mathbf{L}_c\mathbf{vw} = \rho\mathbf{v}^t\mathbf{vw}, \tag{7-31}$$

or

$$\mathbf{Aw} = \rho\mathbf{Bw}, \tag{7-32}$$

where $\mathbf{A} = \mathbf{v}^t\mathbf{L}_c\mathbf{v}$ and $\mathbf{B} = \mathbf{v}^t\mathbf{v}$. Both $\mathbf{A}$ and $\mathbf{B}$ are p×p matrices and therefore Eq. (7-32) has a much smaller dimension compared to Eq. (7-30): ρ is the approximate eigenvalue of the original problem.

Solution of the reduced problem, with dimensions far less than the original one, results in the first eigenvector $\mathbf{w}_1$ and hence $\overline{\phi}$. Nodal ordering is then performed considering the relative entries of $\overline{\phi}$ in an ascending order.

Though the Complementary Laplacian is used in the above method, however, one can also employ the Laplacian matrix itself. In such a case, the eigenproblem $\mathbf{L}\boldsymbol{\phi} = \lambda\boldsymbol{\phi}$ is reduced to $\mathbf{Aw} = \rho\mathbf{Bw}$ using an identical approach, with p×p matrices being involved. For such a small problem, the eigenvalue λ_2 and the corresponding eigenvector leading to the vector $\mathbf{w}$ can easily be calculated. Substituting $\mathbf{w}$ in Eq. (7-29) leads to the approximate Fiedler vector to be used for ordering.

The present methods not only lead to a set of suitable coefficients for graph parameters, but also provide efficient means for measuring the relative significance of each considered graph parameter. These coefficients may also be incorporated in the design of other specific graph-theoretical algorithms for ordering.

7.12.5 NUMERICAL RESULTS

Many examples are studied and the results for three models are presented in this section. In Table 7.1, Column 2 contains the results of the Pure Algebraic Graph Method (PAGM) of Ref. [184].

For the first case, four vectors, representing Ritz vectors, are considered. For these vectors, $\mathbf{v}_1$ contains the degrees of the nodes, $\mathbf{v}_2$ comprises of the 1-weighted degrees of the nodes, and $\mathbf{v}_3$ and $\mathbf{v}_4$ are distances of the nodes from two pseudo-peripheral nodes. These nodes can be obtained using different algorithms, Kaveh [110]. The results are provided in column 3 of the Tables denoted by $\mathbf{v}^4$.

For the second case, five Ritz vectors are employed. The first four vectors are the same as those of the previous case, and the fifth vector $\mathbf{v}_5$ contains the 2-weighted degrees of the nodes of the graph. The results are provided in column 4 of the Tables labelled as $\mathbf{v}^5$.

It should be noted that other vectors containing graph properties which influence the ordering may be considered additional to the above five vectors. However, the formation of such additional vectors may require some extra computational time, reducing the efficiency of the algorithm.

Example 1: An FE mesh with one opening, comprising of 1248 nodes and 1152 rectangular elements is considered, as shown in Figure 7.17. The results for different methods and their computational time are illustrated in Table 7.1 to compare their efficiency.

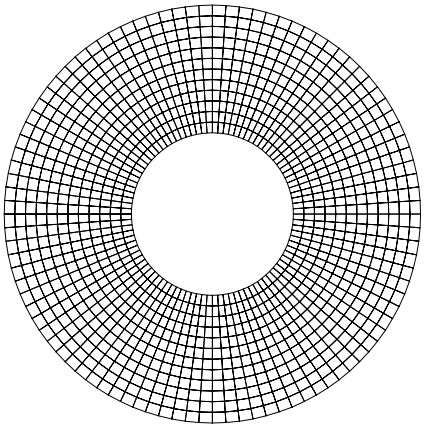


Fig. 7.17 An FE mesh with one opening.

Table 7.1 Results of Example 1.

	PAGM	v^4	v^5
B	46	43	45
P	34848	36243	36189
$\widetilde{F}$	28.07	29.44	29.25
F_{max}	35	39	39
Time (sec)	1400.3	2.8	2.9

Example 2: An H-shaped FE mesh, comprising of 2096 nodes and 3900 triangular elements is considered, as shown in Figure 7.18. The results for different methods and their computational time are illustrated in Table 7.2 to compare their efficiency.

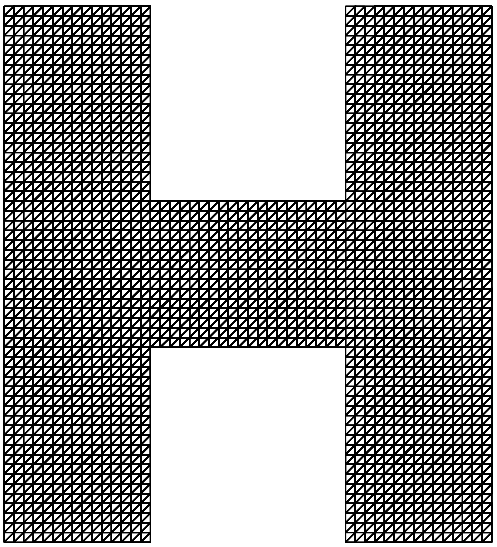


Fig. 7.18 An FE mesh with one opening.

Table 7.2 Results of Example 2.

	PAGM	v^4	v^5
B	74	77	77
P	47741	49400	48936
$\tilde{F}$	23.97	25.63	25.32
F_{max}	37	42	42
Time (sec)	large	2.63	2.89

Example 3: A two-dimensional FEM of a tunnel, comprising of 6888 nodes and 6720 rectangular elements is considered, as shown in Figure 7.19. The results of using different methods and their computational time are presented in Table 7.3.

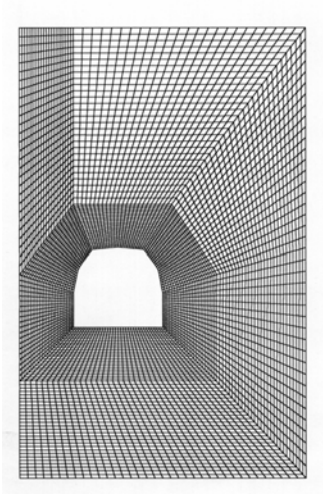


Fig. 7.19 A two-dimensional FEM of a tunnel.

Table 7.3 Results of Example 3.

	PAGM	v^4	v^5
B	455	331	332
P	731694	733738	733738
$\widetilde{F}$	112.99	112.93	112.93
F_{max}	164	175	175
Time (sec)	10.6	27.6	28.9

Example 4: A two-dimensional FE mesh with four openings comprising of 748 nodes and 1236 triangular elements is considered as shown in Figure 7.20. The results for different methods and their computational time are illustrated in Table 7.4 in order to compare their efficiency.

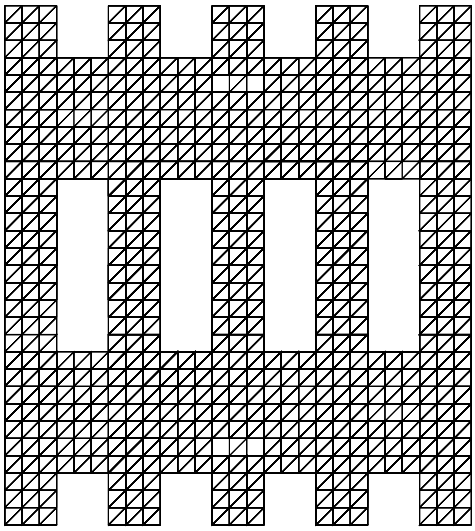
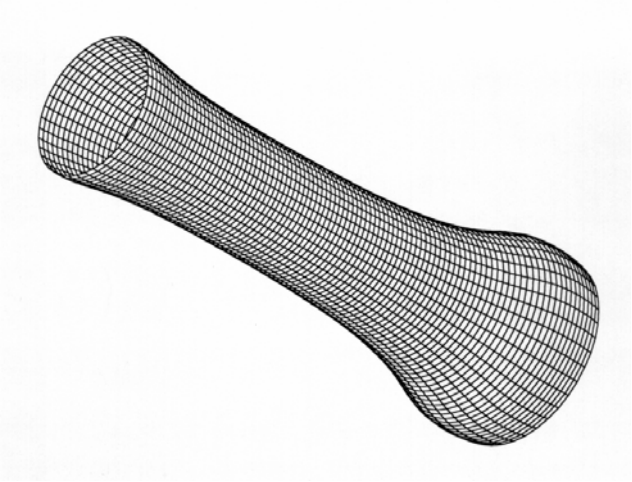


Fig. 7.20 An FEM with four openings.

Table 7.4 Results of Example 4.

	PAGM	v^4	v^5
B	39	49	47
P	13118	13162	13126
$\widetilde{F}$	18.42	18.61	18.56
F_{max}	29	29	29
Time (sec)	1677	1.2	1.3

Example 5: A three-dimensional finite element model of a nuzzle is considered as shown in Figure 7.21. This model, contains 4000 rectangular shell elements. The results for different methods and their computational time are illustrated in Table 7.5, in order to compare their efficiency.



7.21 A three-dimensional FE mesh of a nuzzle.

Table 7.5 Results of Example 5.

	PAGM	v^4	v^5
B	39	49	47
P	13118	13162	13126
$\widetilde{F}$	18.42	18.61	18.56
F_{max}	29	29	29
Time (sec)	1677	1.2	1.3

7.12.6 DISCUSSION

The performance of the present method, illustrated in the previous section, compares well with a pure algebraic graph method, with a substantial reduction in the computational time. Naturally, addition of extra graph parameters will increase the computational time required. Relative values of the coefficients of the Ritz vectors show the importance of the corresponding parameters in the ordering algorithm. For the examples presented in the previous section, the coefficient corresponding to $\mathbf{v}_3$ and $\mathbf{v}_4$ (the distances from the pseudo-peripheral nodes) seem to be more important, since most of the examples have a more or less uniform distribution of nodal degrees. Naturally, for models with non-uniform degree

distributions, the significance of the other graph parameters will also become apparent.

Though only nodal ordering is addressed in here, however, the application of the present method can easily be extended to the element ordering. For this purpose, the natural associate graph or the incidence graph of an FE mesh, should be used in place of the element clique graph.

7.13 BANDWIDTH REDUCTION FOR RECTANGULAR MATRICES

In previous sections, the bandwidth optimisation of square matrices has been discussed. In structural analysis, it may also be desirable to reduce the bandwidth of some sparse rectangular matrices. As an example, it may be beneficial to reduce the bandwidth of the equilibrium equations of a structure, Kaneko et al. [87]. This can be done by optimising the bandwidth of the corresponding cutset basis incidence matrix **L**. Similarly, for compatibility equations, one can optimise the bandwidth of **C**.

In this section, a K-total graph is defined and two algorithms are presented for bandwidth reduction of rectangular matrices.

7.13.1 DEFINITIONS

Let **B** be a rectangular matrix with m rows and n columns, whose entries are denoted by b_{ij}. For each row like i (except the first and the last row, where $i_d = 1$ and $i_d = n$, respectively), the integer part of the real number i(n/m) is defined as i_d. Therefore, the entry of **B** at position (i,i_d) is considered as the *i*th diagonal entry. For square matrices m = n and $i = i_d$. The bandwidth of **B** is then defined as,

$$b(\mathbf{B}) = m_r + m_l + 1, \tag{7-33}$$

where

$$m_r = \max\{k - i_d \,|\, b_{ik} \neq 0,\ k > i_d\},$$

and

$$m_l = \max\{i_d - k \,|\, b_{ik} \neq 0,\ k < i_d\}.$$

If **B** is a symmetric square matrix, then $m_r = m_l$ and b(**B**) reduces to the conventional definition of square matrices. A rectangular matrix is called *banded* if b(**B**) is small compared to m.

Matrix **B** in block submatrix form, has the same pattern as **L**, i.e. each non-zero entry of **L** corresponds to a $\eta \times \eta$ submatrix in **B**, where η is the degree of freedom of a node of the structure. Obviously, reduction of the bandwidth of **L** leads to a banded matrix **B**.

The terms "nodes" and "members" have been used for a graph S, and now we use "vertices" and "edges" for the elements of a *K-total graph*, which is defined as follows:

Associate one vertex with each member and each element of the selected cutset basis or a cycle (γ-cycle) basis of S. Connect two vertices with an edge if:

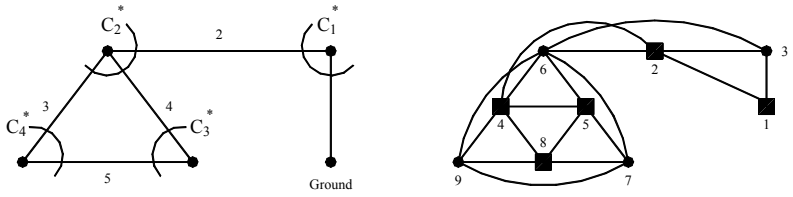
(a) the corresponding members are incident;

(b) the corresponding cutsets (cycles or γ-cycles) are adjacent;

(c) the corresponding member and cutset (cycle or γ-cycle) are incident.

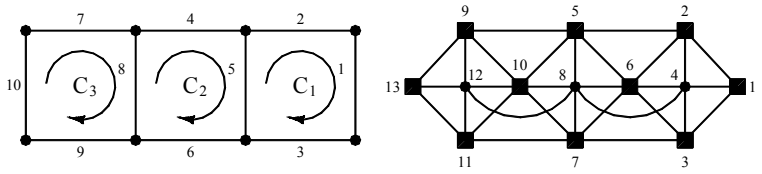
When a cutset or cycle is changed to a node of S, then the K-total graph becomes a total graph, as defined in graph theory (see Behzad [10]).

Examples of K-T(S) are shown in Figures 7.22 and 7.23, when the cocycle basis and cycle basis are considered, respectively. In these figures, small squares are used to represent members and circles are employed to show the elements of the considered basis.



(a) S and the considered cocycle basis. (b) K-T(S) and its nodal ordering.

Figure 7.22 Reduction of bandwidth for a cutest-member incidence matrix.



(a) S and the considered cycle basis. (b) K-T(S) and its nodal ordering.

Fig. 7.23 Reduction of bandwidth for a cycle-member incidence matrix.

7.13.2 ALGORITHMS

Algorithm A

Construct the K-total graph of S and order its vertices. The corresponding sequence, leads to a favourable order of cutsets (nodes) and members of S, to reduce the bandwidth of **L**, which is pattern equivalent to the coefficient matrix of the equilibrium equations. A similar approach, reduces the bandwidth of **C**, when cycles (γ-cycles) are considered in place of cutsets.

This algorithm is now applied to the examples of Figures 7.22 and 7.23, from which the corresponding orders for the elements of the bases and members of S are obtained.

Algorithm B

Order the nodes of S. Then order the unnumbered members of the stars of the nodes in the selected sequence, to obtain a reasonably banded **L** matrix.

In general, Algorithm A leads to a better result than Algorithm B, at the expense of additional computer time.

Examples: Consider a graph S as shown in Figure 7.24 with the corresponding member and cutset orders.

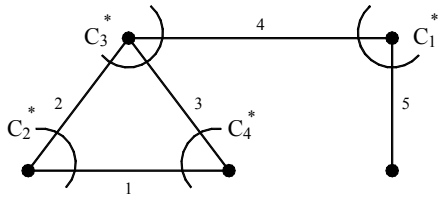


Fig. 7.24 S with an arbitrarily ordered members and cutsets.

The cutset basis incidence matrix of S can be written as,

$$\mathbf{C}^* = \begin{array}{c} \\ C_1^* \\ C_1^* \\ C_1^* \\ C_1^* \end{array} \begin{array}{c} \begin{array}{ccccc} m_1 & m_2 & m_3 & m_4 & m_5 \end{array} \\ \begin{bmatrix} \cdot & \cdot & \cdot & 1 & 1 \\ 1 & 1 & \cdot & \cdot & \cdot \\ \cdot & 1 & 1 & 1 & \cdot \\ 1 & \cdot & 1 & \cdot & \cdot \end{bmatrix} \end{array} \qquad b(\mathbf{L}) = 4+4+1 = 9,$$

where artificially defined diagonal entries are encircled. Using the ordering obtained by K-T(S), the cutset basis incidence matrix becomes,

$$\mathbf{C}^* = \begin{array}{c} \\ C_1^* \\ C_1^* \\ C_1^* \\ C_1^* \end{array} \begin{array}{c} \begin{array}{ccccc} m_1 & m_2 & m_3 & m_4 & m_5 \end{array} \\ \begin{bmatrix} 1 & 1 & \cdot & \cdot & \cdot \\ \cdot & 1 & 1 & 1 & \cdot \\ \cdot & \cdot & \cdot & 1 & 1 \\ \cdot & \cdot & 1 & \cdot & 1 \end{bmatrix} \end{array} \qquad b(\mathbf{L}) = 2+2+1 = 5,$$

in which the non-zero entries are clustered to the diagonal of the matrix.

As a second example, consider S as shown in Figure 7.25, in which the regional cycles and members are arbitrarily numbered.

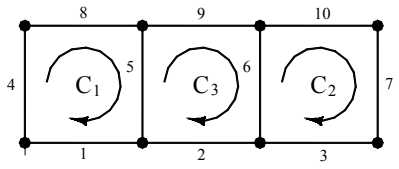


Fig. 7.25 S with arbitrarily numbered members and cycles.

The cycle basis incidence matrix for S is given as:

$$\mathbf{C} = \begin{array}{c} \\ C_1 \\ C_2 \\ C_3 \end{array} \begin{array}{c} \begin{array}{cccccccccc} m_1 & m_2 & m_3 & m_4 & m_5 & m_6 & m_7 & m_8 & m_9 & m_{10} \end{array} \\ \begin{bmatrix} 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 \end{bmatrix} \end{array}.$$

For this matrix, b(**C**) = 7+8+1 = 16. With ordering the cycles and members simultaneously, using Algorithm A, the following cycle basis incidence matrix is obtained,

$$\mathbf{C} = \begin{array}{c} \\ C_1 \\ C_2 \\ C_3 \end{array} \begin{array}{c} \begin{array}{cccccccccc} m_1 & m_2 & m_3 & m_4 & m_5 & m_6 & m_7 & m_8 & m_9 & m_{10} \end{array} \\ \begin{bmatrix} 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{bmatrix} \end{array}.$$

for which b(**C**) = 4+3+1 = 8.

For the force method of frames, the coefficient matrix of the equilibrium equations can be made banded through reducing the bandwidth of its member-cycle incidence matrix. After an algebraic force method is employed, a repeated application of the developed method makes the null basis matrix a banded one for subsequent applications. Similarly, if a combinatorial approach is used, the bandwidth reducing algorithm makes the cycle basis incidence matrix banded, leading to a banded statical basis (null basis) matrix.

7.14 SUBSTRUCTURING FOR DOUBLE BORDERED BLOCK DIAGONAL FORM

In many engineering applications, particularly in the analysis and design of large systems, it is convenient to allocate the design of certain components (substructures) to individual design groups [198]. The study of each substructure is carried out more or less independently, and the dependencies between the substructures resolved after the study of individual substructure is completed. The dependencies among the components may of course require redesign of some of the substructures so the above procedure may be iterated several times.

As an example, suppose for a structural model, we choose a set of nodes I and their incident members which, if removed, disconnect it into two substructures. If the variables associated with each substructure are numbered consecutively, followed by the variables associated with I, then the following partitioning of the stiffness matrix **A** of the entire structure will be induced:

$$\mathbf{A} = \begin{bmatrix} \mathbf{A}_{11} & \mathbf{0} & \mathbf{A}_{13} \\ \mathbf{0} & \mathbf{A}_{22} & \mathbf{A}_{23} \\ \mathbf{A}_{31} & \mathbf{A}_{32} & \mathbf{A}_{33} \end{bmatrix}. \tag{7-34}$$

The Cholesky factor **L** of **A**, correspondingly will be partitioned as,

$$\mathbf{L} = \begin{bmatrix} \mathbf{L}_{11} & & \\ \mathbf{0} & \mathbf{L}_{22} & \\ \mathbf{W}_{13}^{t} & \mathbf{W}_{23}^{t} & \mathbf{L}_{33} \end{bmatrix}, \tag{7-35}$$

where $\mathbf{A}_{11} = \mathbf{L}_{11}\mathbf{L}_{11}^{t}$, $\mathbf{A}_{22} = \mathbf{L}_{22}\mathbf{L}_{22}^{t}$, $\mathbf{W}_{13} = \mathbf{L}_{11}^{t}\mathbf{A}_{13}$, $\mathbf{W}_{23} = \mathbf{L}_{22}\mathbf{L}_{23}^{t}$, and $\mathbf{L}_{11}^{t}\mathbf{L}_{33} = \mathbf{A}_{33} - \mathbf{A}_{13}^{t}\mathbf{A}_{11}^{-1}\mathbf{A}_{23} - \mathbf{A}_{23}^{t}\mathbf{A}_{22}^{-1}\mathbf{A}_{23}$. Hence $\mathbf{A}_{11}$ and $\mathbf{A}_{22}$ correspond to each substructure and the matrices $\mathbf{A}_{13}$ and $\mathbf{A}_{23}$ represent the "glue" which relates the substructures through the nodes of I.

Since the factors of $\mathbf{A}_{11}$ and $\mathbf{A}_{22}$ are independent, they can be computed in either order, or in parallel if two processors are available, Finally, in some design applications, several substructures may be identical, for example, have the same configuration and properties, and each substructure may be regarded as a super-element, which is constructed once and used repeatedly in the design of several structures. In the above example, $\mathbf{A}_{11}$ and $\mathbf{A}_{22}$ could be identical.

In the following, an algorithm is presented for partitioning and ordering of the nodes of a structure, which can be incorporated in any program available for the analysis of structures.

7.14.1 MAIN ALGORITHM FOR SUBSTRUCTURING

Let S be the graph model of a structure. The following algorithm decomposes S into q subgraphs, with equal or near equal number of nodes (support nodes are not counted) and the least number of interface nodes.

Step 1: Delete all the support nodes with their incident members, and denote the remaining subgraphs by S_r.

Step 2: Determine the distance between each pair of nodes of S_r, and evaluate the eccentricities of its nodes.

Step 3: Sort the remaining nodes (RN) in ascending order of their eccentricities.

Step 4: Select the first node of RN as the representative node of the subgraph S_1 to be determined and find a second node as the representative node of subgraph S_2 with a maximum distance from S_1.

Step 5: Find the third representative node with the maximum least distance from S_1 and S_2 , and denote it with S_3.

Step 6: Subsequently, select a representative node of subgraph S_k for which the least distance from S_1, S_2, ... , S_{k-1} is maximum. Repeat this process until q representative nodes of the subgraphs to be selected are found.

Step 7: For each subgraph S_j (j=1,...,q), add an unselected node n_i of RN, if it is adjacent only to S_j and its least distance from all nodes of other subgraphs is maximum.

Step 8: Continue the process of Step 7, without the restriction of transforming one node to each subgraph S_j, until no further node can be transferred. The remaining nodes in RN are interface nodes.

Step 9: Transfer the support nodes to the nearest subgraph.

Once the nodes for each subgraph S_j are found, the incidence members can easily be specified.

The algorithm is recursively applied to the selected substructures, decomposing each substructure into smaller ones, resulting in a further refinement.

7.14.2 SIMPLIFIED ALGORITHM FOR SUBSTRUCTURING

In the following, a simplified algorithm is presented which requires less storage and computer time than the main algorithm at the expense of selecting subgraphs, with a slightly higher number of interface nodes for some structural models. In this approach, the number of distances to be considered and compared for finding the nodes of substructures is far less than when the main algorithm is used, where the distances between each pair of nodes of S are required. This simplified algorithm, consists of the following steps:

Step 1: Form an SRT rooted from an arbitrary node, in order to find a representative node of S_1 with maximum distance from the root. The selected node is also denoted by S_1.

Step 2: Form an SRT rooted from S_1, to calculate the distance between each node of S and S_1, and find the representative node S_2 in a maximum distance from S_1.

Step 3: Form an SRT rooted from S_2, to calculate the distance between each node of S and S_2 and find the representative node S_3 in a maximum least distance from the selected nodes. Repeat this process until q representative nodes S_1, S_2, ... , S_q forming a transversal , are selected.

Step 4: For each subgraph S_i, find a node adjacent to the previously formed S_i only, with maximum least distance from other representative nodes, in turn.

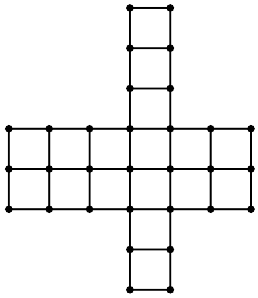
Step 5: Continue the process of Step 4, without the restriction of transforming one node to each subgraph S_i, until no further node can be transferred.

In this algorithm, support nodes are treated in a similar way to Steps 1 and 9 of the main algorithm.

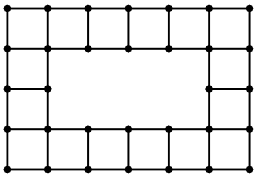
The above algorithms, can easily be applied to subdomaining of finite element models, Kaveh and Roosta [115].

EXERCISES

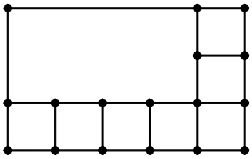
7.1 Find a good starting node for nodal numbering of the following structural models, using graph-theoretical approaches:



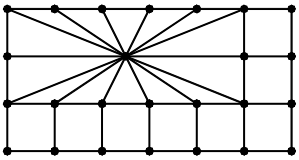
(a)



(b)



(c)



(d)

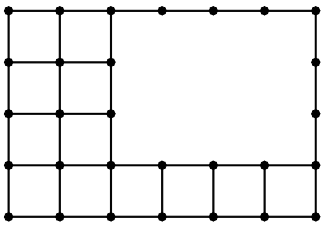
7.2 Find a good starting node for the following models using an algebraic graph-theoretical method, i.e. calculate the dominant eigenvector of the corresponding adjacency matrices.

(a) (b)

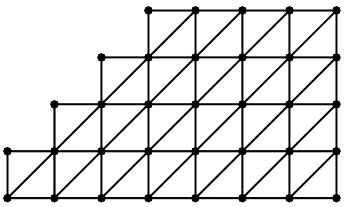
7.3 For the models of Exercise 7.1, find a suboptimal transversal and perform the ordering. Calculate the bandwidth of the corresponding stiffness matrices when the models are viewed as planar trusses.

7.4 Find a connectivity coordinate system for models (a) and (b) in Exercise 7.1.

7.5 For the following graph S, consider a mesh basis and order the cycles, using different distance numbers, to optimise the bandwidth of the corresponding cycle adjacency matrix:



7.6 Order the nodes of the following FEM using the natural associate graph of the model:



7.7 Order the members and elements of a fundamental cycle basis of the following graph, in order to reduce the bandwidth of its cycle basis incidence matrix. Repeat the process, to optimise the bandwidth of its cocycle basis incidence matrix.