Computational Mechanics (1992) 10, 247–262

Computational Mechanics

Multigrid solution procedures for structural dynamics eigenvalue problems

T. Hwang[1] and **I. D. Parsons**[2]

[1] TA Engineering Company, Moraga, California, USA
[2] Department of Civil Engıneering, Unıversıty of Illinois, Urbana, IL 61801, USA

Abstract. Three multigrid methods are described for solving the generalized symmetric eigenvalue problem encountered in structural dynamics. Two implicit algorithms are discussed that use a multigrid method to solve the linear matrix equations encountered ın each iteration of the standard subspace and block Lanczos methods. An explicit method is also outlined which explicitly applies the basic multigrid philosophy of fine mesh relaxation and coarse mesh correction to the eigenvalue problem. All of these algorithms are capable of extracting the lower modes of the system, provided each required eigenvector can be represented on each coarse mesh. The behavior of the methods is studied by examining the selection of convergence tolerances and the solution of some ill-conditioned problems. A well-conditioned plate problem is solved to demonstrate the computational resources required by the algorithms. The explicit method is observed to be the most efficient method (in terms of storage and CPU time), whereas the implicit Lanczos method requires the most computational effort. A comparison between the multigrid algorithms and a commercially available implementation of the subspace iteration method is also presented.

1 Introduction

The development of fast and efficient methods for solving eigenvalue problems has long been of great concern to scientists and engineers. In structural dynamics applications, the generalized symmetric eigenvalue problem

$$\mathbf{K}\boldsymbol{\phi} = \lambda \mathbf{M}\boldsymbol{\phi} \tag{1}$$

is of great importance. Here, **K** and **M** are the $n \times n$ stiffness and mass matrices of the structure under consideration (both symmetric and positive semi-definite), and $(\lambda, \boldsymbol{\phi})$ are the eigensolutions. Usually, only a few (q) of the lowest modes of the structure are of interest (i.e., $q \ll n$). Two of the most widely used methods for computing the first q eigensolutions are the subspace iteration (Bathe 1982) and Lanczos (Lanczos 1950) algorithms. These two techniques take initial guesses to the eigensolutions, and iteratively improve them until satisfactory approximations are obtained. Each method requires the solution of a matrix equation within each iteration. For large problems (e.g., $n > 10^5$), the solution of this equation can dominate the computational resources required to compute the eigensolutions. This paper examines the performance of the two methods when a multigrid method is used to solve these linear equations. In addition, comparisons are made with a multigrid method that is applied directly to Eq. (1).

Multigrid algorithms are fast, iterative methods that have been applied to a variety of problems in science and engineering (Brandt 1977; Hackbusch 1985; Stuben and Trottenberg 1982). Although much work has focussed on studying the mathematical convergence of these algorithms, most applications have centered on fluid mechanics problems, with structural mechanics implementations being relatively rare. The multigrid technique is motivated by the speed with which relaxation methods can reduce the high frequency errors in an approximate solution to a given problem. A basic two grid algorithm consists of two parts. First, a small number of relaxation cycles are performed on a fine mesh to produce a smooth error. A coarse mesh is then used to cheaply compute a coarse mesh correction, which is intended to be a good approximation to the smooth fine mesh error. This two mesh method can generally be extended to a true multigrid method by introducing a still coarser mesh, and recursively using the two mesh method to solve the coarse mesh equation. In some cases, the nature of this equation requires the development of a different multigrid algorithm to solve for the coarse mesh correction.

Although the multigrid method is a relatively new solution procedure, a complete review of the voluminous literature is possibly already beyond the scope of a single article. However, it is useful to briefly mention other attempts at developing multigrid algorithms for solving linear eigenvalue problems. In Hackbusch (1985), algorithms are presented that enable the direct application of fine mesh relaxation and coarse mesh correction to the eigenvalue problem (1); this method is the explicit method discussed later in Sect. 3 of this paper. Another approach was adopted in Bank (1982), in which a multigrid method was used to solve the linear equation encountered in the inverse iteration procedure. This algorithm is capable of extracting only the first eigensolution, and is similar to the implicit subspace method discussed below.

The outline of this paper is as follows. In Sect. 2, a multigrid method is described for solving linear matrix equations, and is used as the matrix equation solver in the basic subspace iteration and Lanczos algorithms. In Sect. 3, a multigrid method is outlined that applies the ideas of fine mesh relaxation and coarse mesh correction directly to Eq. (1). Initial approximations to the required eigensolutions and convergence criteria are discussed in Sect. 4. The various algorithms described in this paper are evaluated and compared in Sect. 5. Data is presented in Sect. 6 that shows the performance of the multigrid methods when they are applied to some large problems. Finally, conclusions are presented in Sect. 7.

2 Implicit multigrid algorithms

Perhaps the most straightforward way in which the multigrid idea can be used to solve eigenvalue problems is to incorporate the multigrid method as a linear matrix equation solver in standard eigensolution procedures. In this paper, these types of algorithms will be called implicit multigrid methods. This section outlines a multigrid method for solving linear matrix equations, and then summarizes the subspace iteration and block Lanczos algorithms. The multigrid method used here is identical to the one used in Parsons (1989), Parsons and Hall (1990a, b).

2.1 *The multigrid method for linear matrix equations*

The basic multigrid method used to solve linear matrix equations is given in Fig. 1. This shows one cycle of the method applied to the equation

$$\mathbf{K}_m x_m = f_m \tag{2}$$

```
Given   $\mathbf{x}_m^{(0)}, \mu_1, \mu_2, \gamma$

    (a)  $\bar{\mathbf{x}}_m^{(k)} = relax_{MG}^{\mu_1}(\mathbf{x}_m^{(k)}, \mathbf{K}_m, \mathbf{f}_m)$
    (b)  $\mathbf{r}_m^{(k)} = \mathbf{f}_m - \mathbf{K}_m \bar{\mathbf{x}}_m^{(k)}$
    (c)  $\mathbf{r}_{m-1}^{(k)} = I_m^{m-1} \mathbf{r}_m^{(k)}$
    (d)  $\Delta\mathbf{x}_{m-1}^{(k)} = MG_{m-1}^{\gamma}(\mathbf{K}_{m-1}, \mathbf{r}_{m-1}^{(k)}, \mu_1, \mu_2)$
    (e)  $\hat{\mathbf{x}}_m^{(k)} = \bar{\mathbf{x}}_m^{(k)} + I_{m-1}^{m} \Delta\mathbf{x}_{m-1}^{(k)}$
    (f)  $\mathbf{x}_m^{(k+1)} = relax_{MG}^{\mu_2}(\hat{\mathbf{x}}_m^{(k)}, \mathbf{K}_m, \mathbf{f}_m)$
```

Fig. 1. $MG_m^1(\mathbf{K}_m, f_m, \mu_1, \mu_2)$, a multigrid method for linear matrix equations on mesh m. First, μ_1 cycles of a relaxation scheme, denoted by $relax^{\mu_1}_{MG}$, is used to produce an approximate solution with a smooth error. Throughout this paper, the Jacobi preconditioned conjugate gradient method (Axelsson and Barker 1984) is used as the relaxation scheme. The smooth error is then represented on the coarse mesh (mesh $m-1$) by restricting the fine mesh residual, $\boldsymbol{r}_m$, to the coarse mesh using the fine-to-coarse mesh restriction operator, I_m^{m-1}. The coarse mesh correction, $\Delta \boldsymbol{x}_{m-1}$, is obtained by solving the coarse mesh correction equation

$$\mathbf{K}_{m-1}\Delta \boldsymbol{x}_{m-1} = \boldsymbol{r}_{m-1}. \tag{3}$$

This is done by recursively applying γ cycles of the multigrid algorithm to Eq. (3), denoted in Fig. 1 by MG^{γ}_{m-1}. The coarse mesh correction is then interpolated to the fine mesh using the coarse-to-fine mesh interpolation operator, I^m_{m-1}, to give the fine mesh correction, $\Delta \boldsymbol{x}_m$. This will be a good approximation to the smooth fine mesh error associated with $\bar{x}_m$. The cycle is completed by applying a further μ_2 relaxation cycles to the new fine mesh approximation. This is done to reduce any high frequency errors introduced by the interpolation operator. The multigrid algorithm is repeated until a converged solution is obtained on the fine mesh. The reader is referred to Parsons (1989), Parsons and Hall (1990a, b) for a complete discussion of the various components (e.g., interpolation and restriction operators) employed in this algorithm.

2.2 The subspace iteration method

Figure 2 lists the steps in the subspace iteration method. Detailed descriptions are given in Bathe (1982). A block of p vectors, $\mathbf{V}_j$, are updated until the required q eigenvectors have converged. Often, the block size p is chosen to be greater than q to improve the convergence of the method. For the purposes of this paper, it should be noted that each cycle of the algorithm requires the solution of the p matrix equations

$$\mathbf{K}\hat{\mathbf{V}}_j = \mathbf{M}\mathbf{V}_{j-1} \tag{4}$$

for $\hat{\mathbf{V}}_j$. These equations will be solved using the multigrid method described in Sect. 2.1.

2.3 The block Lanczos method

Lanczos algorithms have recently been developed to solve the eigenproblems encountered in structural dynamics. Figure 3 outlines a block Lanczos method applied to Eq. (1). This algorithm

```
Given   $\mathbf{V}_0$
For $j = 1,2,3 \ldots$
    (a)  solve $\mathbf{K}\hat{\mathbf{V}}_j = \mathbf{M}\mathbf{V}_{j-1}$ for $\hat{\mathbf{V}}_j$
    (b)  $\mathbf{K}_j^* = \hat{\mathbf{V}}_j^T\mathbf{K}\hat{\mathbf{V}}_j$
    (c)  $\mathbf{M}_j^* = \hat{\mathbf{V}}_j^T\mathbf{M}\hat{\mathbf{V}}_j$
    (d)  solve $\mathbf{K}_j^*\Phi_j = \Lambda_j\mathbf{M}_j^*\Phi_j$ for $\Phi_j$, $\Lambda_j$
    (e)  $\mathbf{V}_j = \hat{\mathbf{V}}_j\Phi_j$
    (f)  check for convergence using $\Lambda_j$, $\mathbf{V}_j$
```

Fig. 2. The subspace iteration method

```
Given   $\mathbf{R}_0$, $\tilde{\mathbf{Q}}_0 = 0$
For $j$ = 1,2,3 . . .
    (a)  compute $\mathbf{b}_j$ from $\mathbf{b}_j\mathbf{b}_j^T = \mathbf{R}_{j-1}^T\mathbf{M}\mathbf{R}_{j-1}$
    (b)  $\tilde{\mathbf{Q}}_j = \mathbf{R}_{j-1}\mathbf{b}_j^{-T}$
    (c)  solve $\mathbf{K}\bar{\mathbf{R}}_j = \mathbf{M}\tilde{\mathbf{Q}}_j$ for $\bar{\mathbf{R}}_j$
    (d)  $\hat{\mathbf{R}}_j = \bar{\mathbf{R}}_j - \tilde{\mathbf{Q}}_{j-1}\mathbf{b}_j$
    (e)  $\mathbf{a}_j = \tilde{\mathbf{Q}}_j^T\mathbf{M}\hat{\mathbf{R}}_j$
    (f)  check for convergence
    (g)  $\mathbf{R}_j = \hat{\mathbf{R}}_j - \tilde{\mathbf{Q}}_j\mathbf{a}_j$
```

Fig. 3. The block Lanczos algorithm

is basically an extension of the single vector method described in Nour-Omid (1987). Given a block of p starting vectors that form the columns of an $n \times p$ matrix $\mathbf{R}$, a sequence of subspaces $\{\mathbf{R}, \mathbf{K}^{-1}\mathbf{M}\mathbf{R}, (\mathbf{K}^{-1}\mathbf{M})^2\mathbf{R}, \ldots, (\mathbf{K}^{-1}\mathbf{M})^j\mathbf{R}\}$ is generated. This sequence of $j+1$ subspaces is then used to produce Ritz vectors to estimate the required eigensolutions using the Rayleigh–Ritz method. To do this, a block of Lanczos vectors, $\tilde{\mathbf{Q}}_j$, must be generated that is associated with each new subspace $(\mathbf{K}^{-1}\mathbf{M})^j\mathbf{R}$ so that the $n \times pj$ matrix

$$\mathbf{Q}_j = [\tilde{\mathbf{Q}}_1 | \tilde{\mathbf{Q}}_2 | \cdots | \tilde{\mathbf{Q}}_j] \tag{5}$$

is an $\mathbf{M}$-orthonormal basis for the generated sequence of the j subspaces. As a result of the orthogonality properties of the Lanczos vectors, the Rayleigh–Ritz method requires the solution of the reduced eigenvalue problems

$$\mathbf{T}_j s_i^{(j)} = \theta_i^{(j)} s_i^{(j)}, \quad i = 1, \ldots, pj. \tag{6}$$

The new approximations to the required eigenvalues are then $(1/\theta_i^{(j)}, \mathbf{Q}_j s_i^{(j)})$. The matrix $\mathbf{T}_j$ is a block tridiagonal matrix defined as

$$\mathbf{T}_j = \begin{bmatrix} \mathbf{a}_1 & \mathbf{b}_2 & & & \\ \mathbf{b}_2^T & \mathbf{a}_2 & \mathbf{b}_3 & & \mathbf{0} \\ & \ddots & \ddots & \ddots & \\ & & \mathbf{b}_{j-1}^T & \mathbf{a}_{j-1} & \mathbf{b}_j \\ \mathbf{0} & & & \mathbf{b}_j^T & \mathbf{a}_j \end{bmatrix}. \tag{7}$$

The algorithm is repeated until converged eigensolutions have been obtained using the columns of $\mathbf{Q}_j$ as trial vectors in the Rayleigh–Ritz procedure. In this paper, a full reorthogonalization procedure is used, whereby the columns of $\mathbf{R}_j$ are reorthogonalized with respect to the columns of $\mathbf{Q}_j$ after each Lanczos cycle. The multigrid method is used to solve the linear matrix Eq. (c) in Fig. 3, i.e., to solve

$$\mathbf{K}\bar{\mathbf{R}}_j = \mathbf{M}\tilde{\mathbf{Q}}_j \tag{8}$$

for $\bar{\mathbf{R}}_j$. In contrast to the subspace iteration method, both the number of Lanczos vectors and the dimension of the reduced system keep growing step by step. In other words, the storage required by the Lanczos vectors significantly increases as the number of Lanczos steps increases. Therefore, the s-step block Lanczos method (Golub and Van Loan 1983) is employed in this study, in which the block Lanczos algorithm is restarted after every s steps of iteration (i.e., after some eigenvectors have converged), using the vectors that have not yet converged as the initial vectors.

3 An explicit multigrid algorithm

An alternative to the implicit algorithms is to directly apply fine mesh relaxation and coarse mesh correction to Eq. (1). This explicit multigrid algorithm is described in detail in Hwang and Parsons (1992a, b), and is based on an algorithm presented in Hackbusch (1985). It is appropriate to summarize this algorithm here. First, v_1 cycles of a suitable relaxation method are applied to an approximate eigensolution, $(\lambda^{(k)}_m, v^{(k)}_m)$. This produces a smooth error on the fine mesh, which is also orthogonal to $\mathbf{M}_m\phi_m$. A coarse mesh correction, $\Delta v^{(k)}_m$, that approximates the smooth fine mesh error can be obtained by solving the coarse mesh correction equation

$$(\mathbf{K}_{m-1} - \lambda_{m-1}\mathbf{M}_{m-1})\Delta v^{(k)}_{m-1} = r^{(k)}_{m-1} - (r^{(k)^T}_{m-1}\phi_{m-1})\mathbf{M}_{m-1}\phi_{m-1}. \tag{9}$$

Note that the operator in this equation is singular. Equation (9) requires the coarse mesh residual (which is restricted from the fine mesh as before), and the coarse mesh eigensolution $(\lambda_{m-1}, \phi_{m-1})$. Once the coarse mesh correction has been computed, it is interpolated to the fine mesh, and a new approximation to the fine mesh eigenvector calculated. This vector can be used with the Rayleigh quotient to give a new fine mesh eigenvalue. This two mesh method is repeated until converged fine mesh eigensolutions have been computed. The algorithm can be extended to compute the first q eigensolutions of Eq. (1) (Hwang and Parsons 1992a, b).

Figure 4 lists the steps involved in one cycle of this explicit multigrid method to solve for the first eigensolution (λ_m, ϕ_m) on a mesh m. The fine mesh relaxation method is represented as $relax^{v_1}_{EMG}$; the method used in this study is the preconditioned conjugate gradient method described in Papadrakakis and Yakoumidakis (1987). The singular coarse mesh correction equation can be solved in several ways. One alternative is to use a suitable iterative method on the coarse mesh.

```
Given   $\nu_1, \mu_1, \mu_2, \gamma$

    (a)  $\bar{\mathbf{v}}^{(k)}_m = relax^{\nu_1}_{EMG}(\mathbf{v}^{(k)}_m, \lambda^{(k)}_m, \mathbf{K}_m, \mathbf{M}_m)$
    (b)  $\bar{\lambda}^{(k)}_m = \frac{\bar{\mathbf{v}}^{(k)^T}_m \mathbf{K}_m \bar{\mathbf{v}}^{(k)}_m}{\bar{\mathbf{v}}^{(k)^T}_m \mathbf{M}_m \bar{\mathbf{v}}^{(k)}_m}$
    (c)  $\mathbf{r}^{(k)}_m = -(\mathbf{K}_m - \bar{\lambda}^{(k)}_m \mathbf{M}_m)\bar{\mathbf{v}}^{(k)}_m$
    (d)  $\mathbf{r}^{(k)}_{m-1} = I^{m-1}_m \mathbf{r}^{(k)}_m$
    (e)  $\Delta\mathbf{v}^{(k)}_{m-1} = SMG^{\gamma}_{m-1}(\mathbf{r}^{(k)}_{m-1}, \mathbf{K}_{m-1}, \mathbf{M}_{m-1}, \lambda_{m-1}, \phi_{m-1}, \mu_1, \mu_2)$
    (f)  $\mathbf{v}^{(k+1)}_m = \bar{\mathbf{v}}^{(k)}_m + I^m_{m-1}\Delta\mathbf{v}^{(k)}_{m-1}$
    (g)  $\lambda^{(k+1)}_m = \frac{\mathbf{v}^{(k+1)^T}_m \mathbf{K}_m \mathbf{v}^{(k+1)}_m}{\mathbf{v}^{(k+1)^T}_m \mathbf{M}_m \mathbf{v}^{(k+1)}_m}$
    (h)  check for convergence using $(\lambda^{(k+1)}_m, \mathbf{v}^{(k+1)}_m)$
```

Fig. 4. Explicit multigrid method for the generalized symmetric eigenvalue problem

```
Given   $\mathbf{w}_m^{(0)}, \mu_1, \mu_2, \gamma$

    (a)   $\bar{\mathbf{w}}_m^{(j)} = relax_{SMG}^{\mu_1}(\mathbf{w}_m^{(j)}, \mathbf{r}_m^{(k)}, \mathbf{K}_m, \mathbf{M}_m, \lambda_m, \phi_m)$
    (b)   $\bar{\mathbf{r}}_m^{(j)} = \hat{\pi}_m(\mathbf{r}_m^{(k)}) - (\mathbf{K}_m - \lambda_m \mathbf{M}_m)\bar{\mathbf{w}}_m^{(j)}$
    (c)   $\mathbf{r}_{m-1}^{(j)} = I_m^{m-1}\bar{\mathbf{r}}_m^{(j)}$
    (d)   $\Delta\mathbf{w}_{m-1}^{(j)} = SMG_{m-1}^{\gamma}(\mathbf{r}_{m-1}^{(j)}, \mathbf{K}_{m-1}, \mathbf{M}_{m-1}, \lambda_{m-1}, \phi_{m-1}, \mu_1, \mu_2)$
    (e)   $\hat{\mathbf{w}}_m^{(j)} = \bar{\mathbf{w}}_m^{(j)} + I_{m-1}^{m}\Delta\mathbf{w}_{m-1}^{(j)}$
    (f)   $\tilde{\mathbf{w}}_m^{(j)} = relax_{SMG}^{\mu_2}(\hat{\mathbf{w}}_m^{(j)}, \mathbf{r}_m^{(k)}, \mathbf{K}_m, \mathbf{M}_m, \lambda_m, \phi_m)$
    (g)   $\mathbf{w}_m^{(j+1)} = \pi_m(\tilde{\mathbf{w}}_m^{(j)})$
```

Fig. 5. $SMG_m^1(r_m^{(k)}, \mathbf{K}_m, \mathbf{M}_m, \lambda_m, \phi_m, \mu_1, \mu_2)$, a multigrid method for singular matrix equations

Another approach is to introduce a still coarser mesh, and to use γ cycles of a multigrid method developed specifically to solve singular matrix equations. This method is denoted as SMG_{m-1}^{γ} in Fig. 4, and is outlined in Fig. 5. The reader is once again referred to Hwang and Parsons (1992a, b) for a full description of this approach.

Equation (9) can be solved to give a unique coarse mesh correction $\Delta v_{m-1}^{(k)}$ because the right-hand side is orthogonal to $\boldsymbol{\phi}_{m-1}$ and $\Delta v_{m-1}^{(k)}$ is orthogonal to $\mathbf{M}_{m-1}\boldsymbol{\phi}_{m-1}$. The algorithm shown in Fig. 5 is basically the same as the regular multigrid method for linear equations shown in Fig. 1. However, projection operators have been added in Eqs. (b) and (g) to enforce the necessary orthogonality conditions. The projection operators $\pi_m(\cdot)$ and $\hat{\pi}_m(\cdot)$ are defined as

$$\pi_m(\boldsymbol{x}_m) = \boldsymbol{x}_m - (\boldsymbol{x}_m^T \mathbf{M}_m \boldsymbol{\phi}_m)\boldsymbol{\phi}_m, \tag{10}$$

which produces a vector orthogonal to $\mathbf{M}_m\boldsymbol{\phi}_m$, and

$$\hat{\pi}_m(\boldsymbol{x}_m) = \boldsymbol{x}_m - (\boldsymbol{x}_m^T \boldsymbol{\phi}_m)\mathbf{M}_m \boldsymbol{\phi}_m, \tag{11}$$

which produces a vector orthogonal to $\boldsymbol{\phi}_m$. The singular multigrid method is recursively used to solve the coarse mesh correction equation on mesh $m - 1$. Note that, because of the construction of the coarse mesh correction Eq. (9), it is necessary to know the eigensolutions on all of the coarse meshes involved in the solution of the eigenvalue problem on the finest mesh.

4 Initial guesses and convergence criteria

A suitable choice for the initial approximate solution can substantially reduce the number of cycles required for convergence of an iterative method. However, it is not always possible to systematically generate good initial guesses. In this paper, the null vector was taken as the initial approximate solution when the multigrid method was used to solve linear matrix equations. This ensured that the initial error was no greater than 100%. However, careful consideration must be given to the choice of starting vectors for the eigensolvers. The standard approach for selecting initial eigenvectors for the subspace iteration method is described in Bathe (1982). The hierarchy of increasingly finer meshes required by the algorithms employed in both the implicit and explicit multigrid methods provides an alternative technique for choosing starting vectors. A nested iteration scheme can be used, whereby the starting vectors on a fine mesh are obtained by interpolating the corresponding eigenvectors from the previous coarse mesh. Eigensolutions on the finest mesh are computed by starting on the coarsest mesh and advancing through the hierarchy of meshes, calculating the solutions on the intermediate meshes using the chosen multigrid method. The solutions on the coarsest mesh are obtained using the standard subspace iteration method. The number of iterating vectors, p, is initially set equal to the number of required eigensolutions, q, for both the implicit and explicit methods. Note that p remains fixed for the implicit subspace and explicit methods; however, p will vary during the s-step Lanczos method, depending on how many eigensolutions have converged. Nested iteration also provides the coarse mesh eigensolutions that are required by the explicit multigrid algorithm.

There are basically two different convergence criteria involved in the multigrid algorithms: one for the linear matrix equations, and one for the eigensolutions themselves. When the algorithms given in Figs. 1 and 5 are applied to the linear matrix equation

$$\mathbf{A}\boldsymbol{x} = \boldsymbol{b}, \tag{12}$$

convergence is assumed when the approximate solution $\boldsymbol{x}^{(k)}$ satisfies

$$\frac{\| \boldsymbol{b} - \mathbf{A}\boldsymbol{x}^{(k)} \|}{\| \boldsymbol{b} \|} < \varepsilon_{lin}. \tag{13}$$

The convergence tolerance is ε_{lin}, and $\| \cdot \|$ denotes the Euclidean norm of a vector. An approximate eigensolution $(\lambda^{(k)}, \boldsymbol{v}^{(k)})$ is accepted when

$$\frac{\| \lambda^{(k)}\mathbf{M}\boldsymbol{v}^{(k)} - \mathbf{K}\boldsymbol{v}^{(k)} \|}{\| \mathbf{K}\boldsymbol{v}^{(k)} \|} < \varepsilon_{\phi}, \tag{14}$$

where ε_{ϕ} is the convergence tolerance for the eigenvalue problem.

5 Behavior of the multigrid algorithms

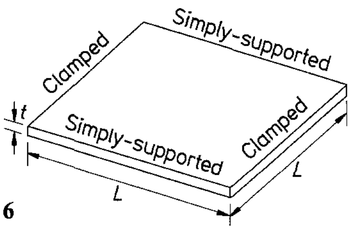
The family of implicit and explicit multigrid methods present a new approach for solving the symmetric generalized eigenvalue problem. However, there are a number of ad-hoc parameters associated with each algorithm that need to be carefully selected. The writers have adopted an approach that provides a physical understanding of the role of the different parts of the multigrid algorithms. This involves solving a series of relatively simple test problems, and studying the effects of changing the various multigrid parameters. Although this approach is not mathematically rigorous and cannot be expected to produce results that are universally true, it does produce a qualitative understanding of the behavior of the different multigrid algorithms that is valuable when attempting to solve practical structural engineering problems.

First, the behavior of the implicit and explicit algorithms was examined when they were used to solve a well-conditioned plate problem. Then, the effect different types of ill-conditioning have on the performance of the algorithms was investigated. Finally, some problems with closely spaced eigenvalues were solved to determine whether this practically important situation causes any convergence difficulties. All of the computations were performed in vector mode on a Convex C240 using 4-byte integer and 8-byte floating point arithmetic.

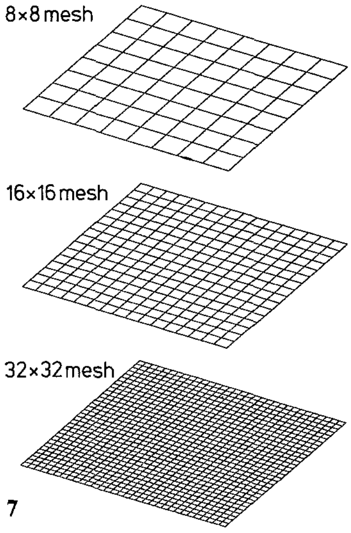
5.1 Selection of the convergence tolerances

The implicit subspace and Lanczos algorithms both require the solution of several linear matrix equations within each iteration. This section investigates the selection of the linear equation tolerance ε_{lin} by solving the square plate problem shown in Fig. 6 with $L/t = 40$. This problem was discretized with a hierarchy of increasingly fine meshes using a four node Mindlin plate element employing hourglass control that is described in Belytschko et al. (1981). Figure 7 shows the first three meshes in this sequence.

The first four eigensolutions of the plate were obtained using two meshes. The 16×16 element mesh served as the finest mesh; the 8×8 mesh was used as the coarsest mesh. The convergence



6



7

Figs. 6 and 7. 6 The square plate $E = 2.5 \times 10^{10}$, $\nu = 0.3$, $\rho = 2400$. **7** Hierarchy of meshes for the square plate

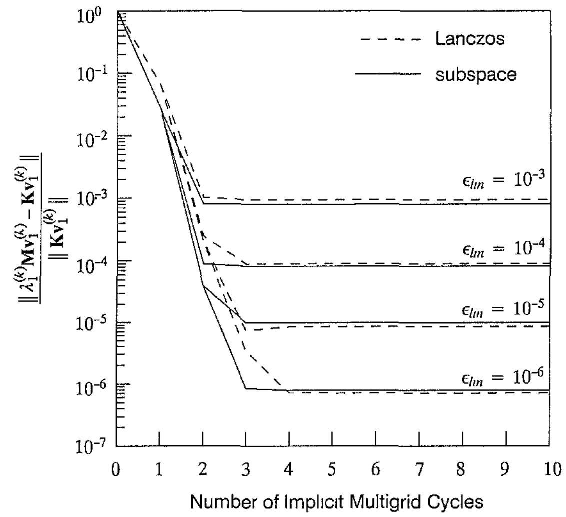
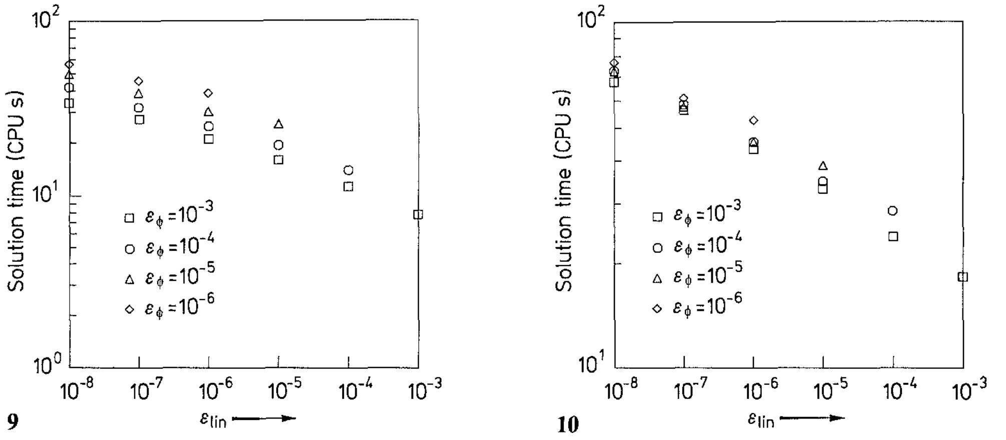


Fig. 8. Convergence of the implicit multigrid methods for the first eigensolution of the square plate on the 16 × 16 mesh

tolerances for both the eigensolutions (i.e., ε_ϕ) and the linear matrix equations (i.e., ε_{lin}) were varied. Figure 8 shows the convergence of the first eigensolution for the two implicit algorithms with various tolerances ε_{lin}. The accuracy of the final eigensolutions depends on the values of this tolerance. In fact, it appears for the problem considered that eigensolutions cannot be computed using the implicit algorithms beyond an accuracy roughly equal to ε_{lin}. That is, if the implicit algorithms are to converge, then ε_{lin} must be less than the specified value of ε_ϕ. Figures 9 and 10 show the solution times required by the implicit algorithms for different values of ε_{lin} and ε_ϕ. For both implicit algorithms, the total solution time is minimized by selecting ε_{lin} equal to ε_ϕ.

The writers have performed extensive numerical experiments to examine the behavior of the explicit multigrid method (Hwang and Parsons 1992a, b). It was found that, regardless of the desired precision of the eigensolutions, a value of ε_{lin} of 10^{-1} for the coarse mesh singular equation minimizes the total solution time for a given value of ε_ϕ for the plate problem. It is therefore reasonable to generalize these observations, and to employ the rather high value of 10^{-1} for ε_{lin} in the subsequent calculations in this study.



Figs. 9 and 10. Solution time **9** of the implicit subspace multigrid method for the first four eigensolutions of the square plate on the 16 × 16 mesh; **10** of the implicit Lanczos multigrid method for the first four eigensolutions of the square plate on the 16 × 16 mesh

5.2 *The effect of closely spaced eigenvalues*

Both the implicit and explicit algorithms use the nested iteration strategy to produce good initial iterating vectors on the finest mesh. This means that all of the meshes in the hierarchy employed must contain adequate representations of the desired eigenvectors. Usually, this is not a problem, since the algorithms presented in this paper are restricted to extracting a small number of the lowest modes of a given structure. However, as the discussion is this section will show, some difficulties can occur when closely spaced eigenvalues are encountered.

The curved shell shown in Fig. 11 was used as a test problem. The shell was discretized with a sequence of uniform meshes using four node shell elements with six degrees of freedom per node. These elements were formed by combining the plate element used previously with a plane stress element incorporating the hourglass controls described in Kosloff and Frazier (1978). The ratio R/t was set at 200 so that some closely spaced eigenvalues were present. Two mesh implicit and explicit algorithms were employed using 8 × 8 and 16 × 16 meshes. The tolerance ε_{lin} was set at 10^{-14} for the implicit methods so that the eigensolutions on the fine mesh could be computed to any desired degree of accuracy. For the explicit algorithm, ε_{lin} was set at 10^{-1}.

Table 1 lists the values of the sixth and seventh eigensolutions of the curved shell when discretized with 8 × 8, 16 × 16, and 32 × 32 meshes. The sixth and seventh eigenvalues are very close to each other. The multigrid algorithms were used to solve for the first six ($q = 6$) and seven ($q = 7$) eigensolutions. Figures 12 and 13 show the convergence behavior of the sixth eigensolution for the schemes with $q = 7$ and $q = 6$, respectively. It is apparent that the implicit algorithms are unaffected by the presence of closely spaced eigenvalues. Both the implicit subspace and the Lanczos algorithms will tend to converge to the eigenvector that is geometrically closest to the initial iterating vector. Since the nested iteration strategy produces initial vectors that are derived from coarse mesh eigenvectors, convergence should be expected to these eigensolutions.

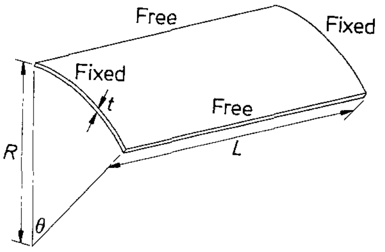
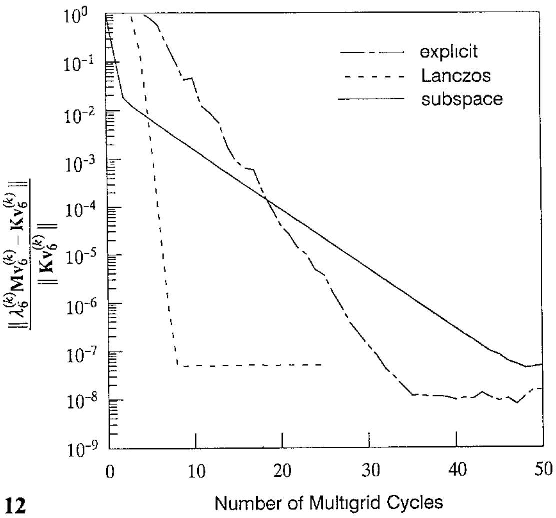


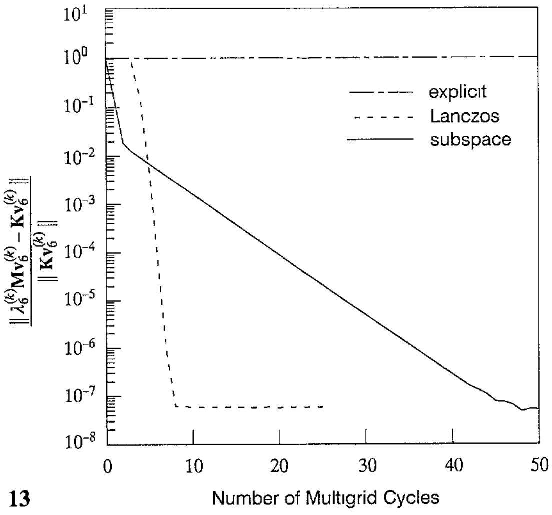
Fig. 11. The curved shell. $\theta = 40°$, $L = R$, $E = 2.5 \times 10^{10}$, $\nu = 0.3$, $\rho = 2400$

Table 1. Eigenvalues for the shell

Mesh	λ_6	λ_7
8×8	3.004×10^3	3.105×10^3
16×16	2.785×10^3	2.835×10^3
32×32	2.594×10^3	2.615×10^3



12



13

Figs. 12 and 13. Convergence of the sixth eigensolution for the curved shell on the 16×16 mesh **12** ($q = 7$); **13** ($q = 6$)

In contrast to the implicit algorithms, the convergence of the explicit algorithm is more sensitive to the presence of closely spaced eigenvalues. Solutions to the singular coarse mesh correction equation can have large errors in the direction of the null space of the coarse mesh operator (Parlett 1980). The coarse mesh correction is usually forced to be orthogonal to this null space. However, for the shell problem with $q = 6$, the closely spaced seventh eigenvalue has not been computed; therefore, errors can develop in the direction of the corresponding eigenvector. This results in inappropriate coarse mesh corrections, and thus failure to converge. The solution is to ensure that enough iterating vectors are used (i.e., increase q).

Different behavior has been observed when eigenvalues are exactly repeated. For example, consider the case when the first l eigenvalues are repeated (e.g., $l = 2$ for the cantilever beam problem considered later in Sect. 5.3) and $q < l$. The convergence of the implicit algorithms are not affected by the missing $l - q$ eigenvectors. The solution on the coarsest mesh produces q eigenvectors that lie in the space spanned by the l eigenvectors. These vectors are interpolated to the fine mesh as the q initial approximations, and the implicit algorithms converge to the fine mesh versions of the q coarse mesh eigenvectors. Similar observations can be made for the explicit algorithm. Even though the solution of the coarse mesh correction is not unique as a result of the missing $l - q$ coarse mesh eigenvectors, the coarse mesh correction algorithms have not been observed to produce solutions that grow in the space spanned by the missing $l - q$ coarse mesh eigenvectors. Therefore, convergence generally occurs to the q eigenvectors on the fine mesh that are geometrically similar to the q eigenvectors obtained on the coarse mesh. Note that although no difficulties to convergence have been observed in this case, it is still safer to compute all of the repeated eigenvalues.

5.3 *Effects of ill-conditioning*

The linear multigrid solver experiences some deterioration in convergence when problems are solved that contain large amounts of bending deformation (Parsons and Hall 1990a, b). This is because the coarse meshes are unable to accurately represent bending deformations, since the finite elements employed are generally too stiff in bending. Thus, the coarse mesh correction is usually underestimated, resulting in an increase in the total number of multigrid cycles required for convergence. Performance can be improved by increasing the value of γ, which basically increases the number of multigrid cycles used on each coarse mesh to compute the coarse mesh correction.

The effect of bending deformation on the performance of the multigrid eigensolvers was investigated by computing the first two eigensolutions of the cantilever beam problem shown in Fig. 14. These two modes consist of bending deformation in the horizontal and vertical planes. The beam was discretized with a sequence of uniform meshes using eight node brick elements with hourglass controls (Kosloff and Frazier 1978). The coarsest mesh was a $32 \times 2 \times 2$ element mesh in all cases. The implicit algorithms used a $128 \times 8 \times 2$ mesh as the finest mesh, whereas the explicit algorithm used a $256 \times 16 \times 16$ mesh. The tolerance ε_ϕ was set at 10^{-3}; for the implicit methods, ε_{lin} was set at 10^{-3}, whereas ε_{lin} was taken as 10^{-1} for the explicit algorithm. The relaxation parameters v_1, μ_1, and μ_2 were all taken as 5, and γ was varied from 1 to 4.

Table 2 shows data obtained by solving the beam problem using the implicit and explicit multigrid algorithms. The average number of cycles required to solve the linear matrix equations and the total solution times are shown for each multigrid algorithm for the various values of γ. For both implicit algorithms, increasing γ is accompanied by a reduction in the average number of cycles required to solve the linear matrix equations. This is a result of the excessive bending stiffness of the coarse meshes; more multigrid cycles applied to the coarse mesh equations produce a more accurate coarse mesh correction. However, the number of operations per cycle increases with γ. For the beam problem, this increase means that the total solution time is minimized by choosing γ equal to 2.

Table 2 presents similar data for the explicit multigrid eigensolver. Here, the finest mesh was the $256 \times 16 \times 16$ mesh, which meant that the coarse mesh correction equation had to be solved on the $128 \times 8 \times 8$ mesh. This data shows that the number of cycles required to solve the coarse mesh correction equation also decreases as γ is increased. In this case, $\gamma = 1$ is optimum for $\varepsilon_{lin} = 10^{-1}$.

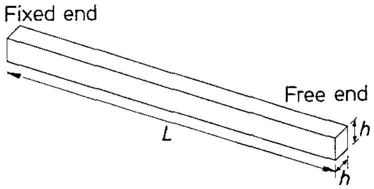


Fig. 14. The cantilever beam $L/h = 16$, $E = 2.5 \times 10^{10}$, $v = 0.3$, $\rho = 2400$

Table 2. Average number of cycles to convergence for the linear equations and total solution times for the first two eigensolutions of the cantilever beam problem

γ	Subspace		Lanczos		Explicit	
	Cycles	Time	Cycles	Time	Cycles	Time
1	18.00	5.564×10^2	22.00	2.621×10^3	2.75	1.773×10^3
2	13.00	5.194×10^2	12.25	1.961×10^3	2.50	1.835×10^3
3	11.00	5.309×10^2	11.00	2.102×10^3	2.50	1.937×10^3
4	11.00	6.089×10^2	10.25	2.267×10^3	2.50	2.048×10^3

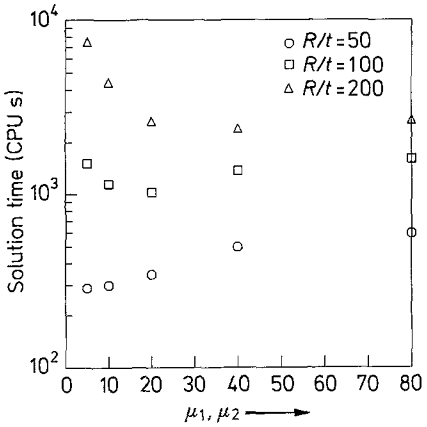
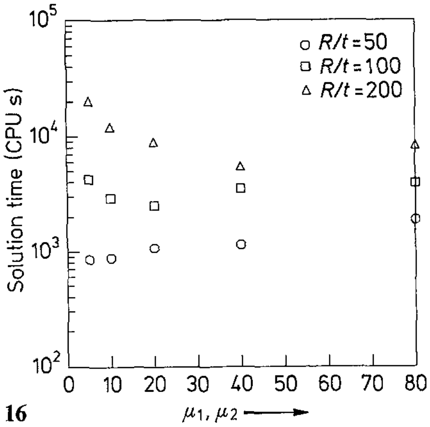
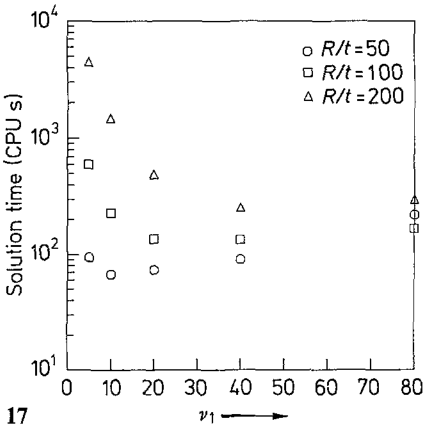


Fig. 15. Solution time of the implicit subspace multigrid method for the first four eigensolutions of the curved shell on the 32 × 32 mesh





Figs. 16 and 17. Solution tıme **16** of the implicit Lanczos multigrid method for the first four eigensolutions of the curved shell on the 32 × 32 mesh; **17** of the explicit multigrid method for the first four eigensolutions of the curved shell on the 32 × 32 mesh (Hwang and Parsons 1992a, b)

A common source of ill-conditioning is the coupling of stiffnesses having different orders of magnitude. One example is the case of nearly incompressible linear elasticity, in which the shear modes of an element are far less stiff then the volumetric modes. In Parsons and Hall (1990a, b), it was found that it was necessary to increase the number of relaxation cycles used to smooth the error on the fine mesh to preserve to convergence of the multigrid method. A similar source of ill-conditioning encountered in structural mechanics is the coupling of axial or shear deformation with bending deformation. This occurs in thin plates and shells. Therefore, it would be expected that increased values of the various relaxation parameters would be required in these cases.

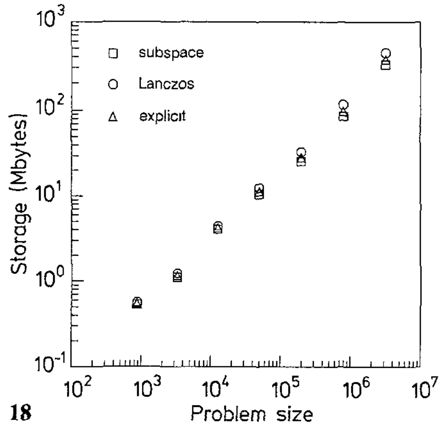
The effects of ill-conditioning on the behavior of the implicit and explicit algorithms were examined when the curved shell problem shown in Fig. 11 was solved. (Note that these results have previously been reported in Hwang and Parsons (1992a, b) for the explicit algorithm.) The value of R/t was varied from 50 to 200; higher values of R/t increase the ill-conditioning of the problems. In order to investigate the effect of R/t on optimum choices for the relaxation parameters μ_1 and μ_2 used in the implicit algorithms, and the fine mesh relaxation parameter ν_1 used in the explicit algorithm, a two mesh algorithm using 16 × 16 and 32 × 32 meshes was used to compute the first four eigensolutions of the curved shell. Figures 15 and 16 show the total solution times for the implicit algorithms for different values of μ_1, μ_2, and R/t. Figure 17 shows similar data obtained using the explicit algorithm. All three of these figures show that, as R/t is increased, more solution time is required to solve the problem for fixed values of μ_1, μ_2, and v_1. It is also clear that the optimum values for these relaxation parameters increases as R/t increases, i.e., as the problem becomes more ill-conditioned. This is a result of the decrease in the smoothing effect of the relaxation methods caused by the coupling of the different stiffnesses. However, this effect is not disastrous, and reasonable performance can still be obtained if the relaxation parameters are adjusted accordingly.

6 Performance of the multigrid methods

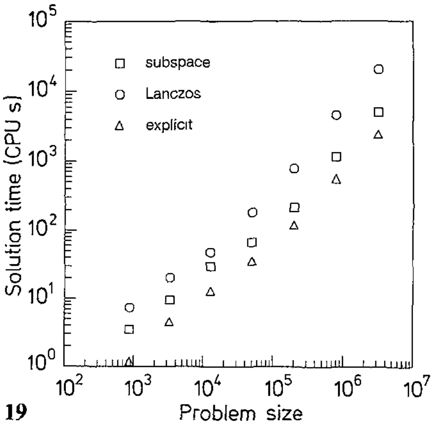
This section evaluates the performance of the implicit and explicit multigrid algorithms by solving some large-scale problems on the Convex C240 in vector mode. In addition, the computational requirements of the implicit and explicit algorithms are compared with a commercially available implementation of the subspace iteration method.

The computation of the first eigensolution of the square plate problem was used to examine the speed and storage requirements of the implicit and explicit algorithms. The 8 × 8 mesh served as the coarsest mesh; a hierarchy of meshes was once again generated by uniform refinement of the coarsest mesh. The relaxation parameters μ_1, μ_2, and v_1 were fixed at 5, γ was set at 1; the convergence tolerance ε_ϕ was chosen as 10^{-3} and ε_{lin} was set at 10^{-3} and 10^{-1} for the implicit and explicit methods, respectively.

Figure 18 shows the storage required by the various multigrid algorithms. The storage for large problems should be linearly proportional to the problem size, n, since most of the storage is assigned to vectors of length n. This was tested by applying linear regression to the storage measured on the Convex for problems with more than 100,000 degrees of freedom. On the coarser meshes, overhead arising partly from storage of data on the coarsest mesh can be expected to influence this proportionality. The storage was found to be proportional to $n^{0.92}$, $n^{0.94}$, and $n^{0.93}$, for the implicit subspace algorithm, the implicit Lanczos algorithm, and the explicit algorithm, respectively. Figure 19 shows the solution times required by the multigrid algorithms. Proportionality between solution time and problem size was examined by applying linear regression to the time measured on the Convex for problems with more than 100,000 degrees of freedom. The solution time was found to be proportional to $n^{1.15}$, $n^{1.18}$, and $n^{1.09}$, for the implicit subspace algorithm, the implicit Lanczos algorithm, and the explicit algorithm, respectively. It is interesting to note that the explicit method appears to be more efficient than either of the implicit algorithms. The largest problem (3,151,875 degrees of freedom, 1 eigensolution) was solved in about 42 minutes on the Convex using the explicit method, with about 370 Mbytes of memory.

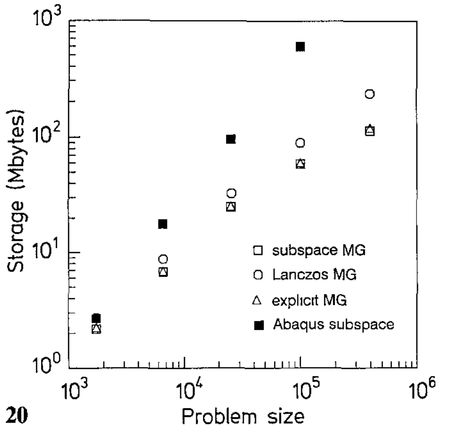


18

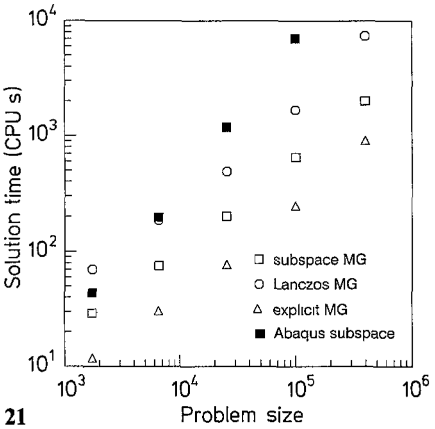


19

Figs. 18 and 19. **18** Storage required by the multigrid methods to compute the first eigensolution of the square plate problem. **19** Solution time required by the multigrid methods to compute the first eigensolution of the square plate problem



20



21

Figs. 20 and 21. 20 Storage of the multigrid methods and the subspace iteration method for the first ten eigensolutions of the square plate. **21** Solution time of the multigrid methods and the subspace iteration method for the first ten eigensolutions of the square plate

The performance of the implicit and explicit multigrid algorithms was compared to a commercially available implementation of the subspace iteration method. This was done by computing the first ten eigensolutions of the square plate problem on the Convex C240 with the multigrid code and the Abaqus finite element code (Abaqus 1990). Abaqus does not have the same four node plate element that was used in the multigrid code. However, Abaqus does have a four node shell element with a one point integration scheme (element S4R5 in the Abaqus library). Therefore the plate was discretized with shell elements for both codes. Note that the subspace iteration method requires the assembly and factorization of the global stiffness matrix.

Figures 20 and 21 compare the storage requirements and solution times of the two codes. The storage required by Abaqus was obtained by measuring the out-of-core disk usage. The Abaqus solution time is defined as the time spent in the eigensolver only, and thus excludes any preprocessing time, for example, time spent reading data or numbering nodes. It is observed that the implicit and explicit algorithms required less computational effort in terms of storage and time for various problem sizes, except when the implicit Lanczos algorithm was applied to the 1,734 degree of freedom problem. The multigrid algorithms require far less storage as the problem size increases: for example, the explicit algorithm needs only 10% of the storage required by the subspace iteration method for the problem with 100,000 degrees of freedom. Note that it is impractical to solve the problem with a mesh finer (i.e., 400,000 degrees of freedom) on the Convex available using Abaqus, since it can be estimated that Abaqus will require about 3.7 Gbytes of out-of-core disk space. In contrast, the same problem can be solved easily using the explicit algorithm with only 119 Mbytes of in-core storage (as shown in Fig. 20). The multigrid algorithms have a low storage requirement because no global matrices are assembled and stored. Figure 21 also demonstrates the speed of the multigrid algorithms; for example, the explicit algorithm requires only 3.5% of the time of the subspace iteration method for the 100,000 degrees of freedom problem.

7 Discussion

This paper has presented three different multigrid algorithms for solving the symmetric generalized eigenvalue problem. Two implicit methods were constructed that use a multigrid method for solving linear matrix equations in conjunction with the standard subspace and Lanczos algorithms. An explicit multigrid method was also examined, which applies the concepts of fine mesh relaxation and coarse mesh correction directly to the eigenvalue problem. The behavior of these algorithms was investigated by solving some relatively simple test problems. It was found that the convergence tolerance for the linear equation solvers had to be carefully selected in order to obtain optimum performance for a given eigensolution accuracy. Multiple eigenvalues were found to present no convergence difficulties for the implicit methods. However, a sufficient number of iterating vectors must be chosen if the explicit algorithm is to be used to compute closely spaced eigenvalues successfully. The solution of some ill-conditioned problems highlighted the role that correct selection of the multigrid parameters play in the performance of the algorithms.

A well-conditioned plate problem served to demonstrate the performance of the multigrid algorithms when used to solve large-scale problems. The storage required was observed to be almost linearly proportional to the problem size, with the implicit subspace method requiring the least storage. The relationship between solution time and problem size was measured to be approximately linear, with the explicit method being closest to linear. The explicit method was the fastest multigrid method, with the Lanczos method, perhaps surprisingly, being the slowest.

It is important to note the limitations of the algorithms presented here. All of the methods were designed to extract the lowest q modes from a given fine mesh. The explicit method requires the use of the nested iteration scheme so that the singular coarse mesh correction equations can be constructed and solved. Therefore, the required q modes must also be geometrically represented to some degree on all of the coarse meshes employed if the explicit multigrid method is to be used successfully. The same considerations apply to the use of the implicit methods with the nested iteration strategy. Shifting could be used to compute higher modes using the implicit methods (without the use of nested iteration, since the coarse meshes could not generally be expected to contain the required higher fine mesh modes). However, this would produce a series of indefinite linear matrix equations, which cannot be readily solved using the basic linear multigrid solver described in this paper.

It is perhaps worthwhile to briefly mention two important points that this paper has not discussed. First, multigrid algorithms in general are eminently suitable for use with adaptive mesh refinement schemes. The required hierarchy of increasingly finer meshes can be automatically generated from a given coarse mesh, and error estimators used to determine the discretization error on each mesh. (This approach has in fact been used by many other researchers in different applications, e.g., (Brandt 1977).) The marriage between the multigrid methods and effective adaptive mesh refinement algorithms could further improve the usefulness of the algorithms presented here. Second, both the implicit and explicit methods can be efficiently implemented on shared memory parallel computers (e.g., Cray Y-MP, Alliant FX/8, Convex C240) using an element level strategy. The reader is referred to Hwang (1991) for descriptions of this implementation and some results that demonstrate the parallel performance of the algorithms on these types of machines. This reference also demonstrates the successful application of the multigrid eigensolvers to some more practical problems.

Acknowledgements

The writers are grateful to Bahram Nour-Omid for helpful discussions regarding the Lanczos method. Financial support was provided by the National Science Foundation, grant number EET-8808190.

References

Abaqus Version 4-8 (1990). Hibbitt, Karlsson, and Sorensen, Inc., Providence, Rhode Island

Axelsson, O.; Barker, V. A. (1984): Finite element solution of boundary value problems. London: Academic Press

Bank, R. E. (1982): Analysis of a multilevel inverse iteration procedure for eigenvalue problems. SIAM J. Numer. Anal. 19, 886–898.

Bathe, K. J. (1982): Finite element procedures in engineering analysis. Englewood Cliff, N.J.: Prentice-Hall

Belytschko, T.; Tsay, C. S.; Liu, W. K. (1981): A stabilization matrix for the bilinear Mindlin plate element. Comput. Meth. Appl. Mech. Eng. 29, 313–327

Brandt, A. (1977): Multi-level adaptive solutions to boundary value problems. Math. Comput. 31, 333–390

Golub, G. H.; Van Loan, C. F. (1983): Matrix computations. John Hopkins University Press

Hackbusch, W. (1985): Multi-grid methods and applications. Berlin, Heidelberg, New York: Springer

Hwang, T. (1991): Multilevel solution procedures for structural dynamics eigenvalue problems. Thesis submitted in partial fulfillment of the degree of Doctor of Philosophy, Department of Civil Engineering, University of Illinois at Urbana-Champaign

Hwang, T.; Parsons, I. D. (1992a): A multigrid algorithm for the generalized symmetric eigenvalue problem: Part I—Algorithm and implementation. To appear in Int. J. Numer. Meth. Eng.

Hwang, T.; Parsons, I. D. (1992b): A multigrid algorithm for the generalized symmetric eigenvalue problem: Part II—Performance evaluation. To appear in Int. J. Numer. Meth. Eng.

Kosloff, D.; Frazier, G. A. (1978): Treatment of hourglass patterns in low order finite element codes. Int. J. Numer. Anal. Meth. Geomech. 2, 57–72

Lanczos, C. (1950): An iteration method for the solution of the eigenvalue problem of linear differential and integral operators. J. Res. Nat. Bur. Stand. 45, 255–281

Nour-Omid, B. (1987): The Lanczos algorithm for solution of large generalized eigenproblems. In chapter 6 of Hughes, T. J. R., The finite element method. Englewood Cliffs, N.J.: Prentice-Hall

Papadrakakis, M.; Yakoumidakis, M. (1987): A partial preconditioned conjugate gradient method for large eigenproblems. Comput. Meth. Appl. Mech. Eng. 62, 195–207

Parlett, B. N. (1980): The symmetric eigenvalue problem. Englewood Cliff, N.J.: Prentice-Hall

Parsons, I. D. (1989): The implementation of an element level multigrid algorithm on the Alliant FX/8. Comput. Phys. Comm. 53, 337–348

Parsons, I. D.; Hall, J. F. (1990a): The multigrid method in solid mechanics: Part I—Algorithm description and behavior. Int. J. Numer. Meth. Eng. 29, 719–737

Parsons, I. D.; Hall, J. F. (1990b): The multigrid method in solid mechanics: Part II—Practical applications. Int. J. Numer. Meth. Eng. 29, 739–753

Stuben, K.; Trottenberg, U. (1982): Multigrid methods: Fundamental algorithms, Model problem analysis and applications. In: Hackbusch, W.; Trottenberg, U. (eds.) Multigrid methods, 1–176. Berlin, Heidelberg, New York: Springer

Communicated by S. N. Atluri, April 20, 1992