INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN ENGINEERING, VOL. 39, 2209–2229 (1996)

CONJUGATE GRADIENT METHODS FOR SOLVING THE SMALLEST EIGENPAIR OF LARGE SYMMETRIC EIGENVALUE PROBLEMS

Y. T. FENG AND D. R. J. OWEN

Department of Civil Engineering, University of Wales Swansea, Singleton Park, Swansea, SA2 8PP, U.K.

SUMMARY

In this paper, a detailed description of CG for evaluating eigenvalue problems by minimizing the Rayleigh quotient is presented from both theoretical and computational viewpoints. Three variants of CG together with their asymptotic behaviours and restarted schemes are discussed. In addition, it is shown that with a generally selected preconditioning matrix the actual performance of the PCG scheme may not be superior to an accelerated inverse power method. Finally, some test problems in the finite element simulation of 2-D and 3-D large scale structural models with up to 20200 unknowns are performed to examine and demonstrate the performances.

KEY WORDS: eigenvalue problem; conjugate gradient method; Rayleigh quotient; asymptotic analysis; preconditioning

1. INTRODUCTION

There are a variety of scientific and technological fields, such as mathematics, physics, solid mechanics, and hydrodynamics, where the following (generalized) eigenvalue problem often arises

$$A\phi = \omega B\phi \tag{1}$$

in which A, B are $n \times n$ matrices, ω is an eigenvalue and ϕ is the corresponding eigenvector. If $B = I$ (identity matrix), equation (1) is a standard eigenvalue problem. We are interested in problems where A and B are very large and sparse symmetric positive definite (SPD) matrices. Furthermore, they are supposed to have no general sparsity pattern. In this case, the n eigenvalues of the system will be positive real and denoted by

$$0 \leqslant \omega_1 \leqslant \omega_2 \leqslant \cdots \leqslant \omega_n$$

and the corresponding eigenvectors are denoted by

$$\phi_1, \phi_2, \ldots, \phi_n$$

The evaluation of one or more smallest eigenpairs has much practical interest for describing the characteristics of physical phenomena.

In recent years, considerable effort has been devoted to the development of efficient and reliable methods for solving such problems, and a large number of algorithms are available which have been successfully applied to a wide range of practical problems. Among these methods, the subspace iteration method[1] and the Lanczos method[2] are considered to be the most efficient. Further details on these methods can be found in the literature, for example the work by Parlett.[3]

Received 15 June 1994
Revised 1 August 1995

The subspace iteration method or the Lanczos method for extracting the smallest eigenpairs of a system are inverse power based methods. In other words, the main operations in the iterative process are solutions of the linear algebraic equations which are generally performed by direct solvers such as LDL^{T}. Consequently, this gives rise to two severe problems: (1) For large scale problems, although sparse matrix techniques have been adopted in many direct solvers, the storage requirements are still extremely high, and may not be affordable even by current supercomputers. (2) The operations in LDL^{T}, or more generally, all methods which are Gauss elimination-based are essentially sequential, and as a result, it is very difficult to obtain a high speed-up in a parallel environment. Even though much work has been done to date, the efficient parallelization of direct algorithms still remains a challenging task.

The above difficulties may be partially overcome by using mixed schemes, where the linear equations are solved by iterative methods instead of direct approaches, see for instance Reference 4. However, the overall efficiency of the algorithms may become very low, due to the fact that iterative methods are not efficient in solving the same linear equations with different right-hand sides.

An alternative to the inverse power technique employs iterative schemes for minimizing the Rayleigh quotient

$$\lambda(x) = \frac{x^{\mathrm{T}}Ax}{x^{\mathrm{T}}Bx} \tag{2}$$

based on the fact that the minimum value $\lambda_{\min}$ of equation (2) is equal to the smallest eigenvalue ω_1, and the corresponding vector x^* coincides with the eigenvector ϕ_1 associated with ω_1.

The idea of transforming the eigenproblem (1) into a minimum optimization problem, first proposed by Hesteness and Karush[5] in 1951, opens a wide prospect for the evaluation of eigenvalues with the aid of the optimization procedures which have become well developed in recent decades. Certainly, some special conditions are needed in selecting the particular methods.

During the period 1950s to 1970s, several methods, such as the steepest descent method,[5] coordination relaxation scheme[6] and conjugate gradient method (CG)[7] were adopted to assess the smallest eigenpairs based on the minimization of the Rayleigh quotient. These methods can evaluate not only the smallest eigenpairs but also several smallest eigenpairs with the aid of deflation or subspace techniques.[8]

The most attractive characteristic of these methods is that the iterative process mainly consists of vector–vector operations or matrix–vector multiplications which are highly suitable for being parallelized or vectorized. Moreover, these methods can easily handle large scale problems due to their much reduced storage requirements.

Unfortunately, all of these methods suffer from poor convergence properties and cannot compete with inverse power technique based methods such as the Lanczos method or the subspace iteration method for solving intermediate sized problems. This shortcoming severely diminishes the reputation of the methods, and as a result, they have been largely abandoned and are not even considered as a possible solution strategy for eigenproblems in many texts such as Parlett[3] and Bathe and Wilson.[1] This situation had not changed until very recently when Perdon and Gambolati[9] treated large eigenvalue problems by the minimization of the Rayleigh quotient using a (preconditioned) conjugate gradient method.

Among the methods mentioned above for minimizing the Rayleigh quotient, the conjugate gradient method appears to be the most efficient and robust providing relatively faster convergence properties and is free of any required parameter estimation. However, as in the case of the solution of linear equations, its successful application to eigenvalue problems depends also upon the preconditioning techniques to a large extent. Preconditioning may dramatically improve the convergence of the iterative process if the preconditioning matrices are appropriately selected. Many preconditioning schemes are available and the development of new schemes is still an active research area.

Due to the non-quadratic form of the Rayleigh quotient, many properties of the conjugate gradient method valid for solving linear equations no longer hold. These, on one hand, decrease the efficiency of the CG, but on the other hand, offer the opportunity of employing several variants in the implementation of CG as mentioned in References 7, 9–11. However, insufficient theoretical and numerical results are provided to justify the performance of these approaches, although some work has been done by Bergamaschi *et al.*,[12] who show that one of the variants has the same asymptotic convergence rate as the CG method applied to the solution of linear systems. Moreover, in spite of the fact that some numerical experiments have been undertaken,[7, 9–13] the maximum size of the test problems is only up to 4560 variables,[10, 11] which is not large enough from the viewpoint of current application requirements.

This paper aims to give a detailed description of CG for evaluating eigenvalue problems in both theoretical and computational aspects. The paper is organized in the following way: In Section 2, the CG scheme is presented together with a brief introduction to the linear equation case; Some properties of the methods are also recalled, and several variants are then proposed and discussed in Section 3; Preconditioning strategies are discussed and the relation to inverse power methods is also outlined in Section 4; Finally, some test problems in the finite element simulation of 2-D and 3-D large scale structural models are performed in Section 5 to examine and demonstrate the results obtained in the previous sections.

2. CONJUGATE GRADIENT SCHEMES

2.1. The CG scheme for solving linear equations

The CG scheme is briefly presented for solving linear algebraic equations of the form

$$Ax = b \tag{3}$$

where A is a symmetric positive-definite matrix, b is a known vector and x is the solution to be found. We may construct a quadratic function $\lambda(x)$ of the form

$$\lambda(x) = \tfrac{1}{2}x^{\mathrm{T}}Ax - x^{\mathrm{T}}b \tag{4}$$

the minimum of which is the solution of equation (3).

For an arbitrary approximate solution x, the gradient of $\lambda(x)$ is

$$g(x) = Ax - b = r(x) \tag{5}$$

and the bigradient or Hessian matrix of $\lambda(x)$ is thus the coefficients matrix A, i.e.

$$H(x) = A \tag{6}$$

The CG algorithm is as follows:

1. Give an arbitrary initial guess x_0, compute $g_0 = Ax_0 - b$ and set $p_0 = g_0$.
2. For i = 0, 1, 2, . . . do

$$x_{i+1} = x_i + \alpha_i p_i$$

where α_i minimizes $\lambda(x_i + \alpha_i p_i)$

$$g_{i+1} = g_i + \alpha_i A p_i, \quad p_{i+1} = g_{i+1} + \beta_i p_i$$

where β_i makes $p_{i+1}^T A p_i = 0$.

Two well known properties of CG are: (1) The exact solution can be determined in at most n iteration steps in the sense of exact arithmetic; (2) The search directions p_i(i = 0, 1, 2, . . .) are A-conjugate, i.e.

$$p_i^T A p_j = 0 \quad (i \neq j) \tag{7}$$

In addition, β_i has five equivalent forms[14]

$$\beta_i = -\frac{g_{i+1}^T A p_i}{p_i^T A p_i} \tag{8a}$$

$$= -\frac{g_{i+1}^T (g_{i+1} - g_i)}{p_i^T (g_{i+1} - g_i)} \tag{8b}$$

$$= \frac{g_{i+1}^T (g_{i+1} - g_i)}{g_i^T g_i} \tag{8c}$$

$$= \frac{g_{i+1}^T g_{i+1}}{p_i^T g_i} \tag{8d}$$

$$= \frac{g_{i+1}^T g_{i+1}}{g_i^T g_i} \tag{8e}$$

in which the last one is the simplest and the most economical one in the sense of CPU requirements and is thus used in actual codes. The equivalence of the second and third with the first is valid, subject to the hypothesis that the gradient of the objective function should be linear so that the following relation holds

$$g_{i+1} - g_i = \alpha_i A p_i \tag{9}$$

whereas the equivalence of the last two with the first three further requires the objective functions to be quadratic.

2.2. *The CG scheme for minimizing the Rayleigh quotient*

For the sake of simplicity, we mainly deal with the standard eigenvalue problem while the scheme for generalized eigenvalue problems will be given at the end of this section. The smallest eigenpair is equivalent to the following non-constrained or constrained optimization problems

$$\operatorname*{Min}_{x \subset R^n} \lambda(x) = \operatorname*{Min}_{x \subset R^n} \frac{x^T A x}{x^T x} \tag{10}$$

or

$$\operatorname*{Min}_{x \subset R^n} \lambda(x) = \begin{cases} \operatorname*{Min}_{x \subset R^n} x^T A x \\ x^T x = 1 \end{cases} \tag{11}$$

We will continue our discussion based on the second form, i.e. the constrained optimization problem (11). The gradient and bigradient (or Hessian) of $\lambda(x)$ are, respectively,

$$g(x) = 2(Ax - \lambda(x)x) \tag{12}$$

$$H(x) = 2(A - \lambda(x)I - xg(x)^{\mathrm{T}} - g(x)x^{\mathrm{T}}) \tag{13}$$

We will remove the factor 2 in the above expressions in the later discussion, which does not affect the results but can save some floating point operations.

By exact analogy with the scheme for solving linear equations, the CG scheme to evaluate the smallest eigenpair can be outlined as

1. Given an initial guess x_0 with $\|x_0\| = 1$. Compute $\lambda_0 = x_0^{\mathrm{T}}Ax_0$, and $g_0 = Ax_0 - \lambda_0 x_0$. Set $p_0 = g_0$.
2. For i = 0, 1, 2, . . . Do

$$x^*_{i+1} = x_i + \alpha_i p_i, \quad x_{i+1} = x^*_{i+1}/\|x^*_{i+1}\|$$

where α_i minimizes $\lambda(x_i + \alpha_i p_i)$

$$\lambda_{i+1} = x^{\mathrm{T}}_{i+1}Ax_{i+1}, \quad g_{i+1} = Ax_{i+1} - \lambda_{i+1}x_{i+1}$$

$$p_{i+1} = g_{i+1} + \beta_i p_i$$

where β_i makes p_{i+1} and p_i A-conjugate.

Because of the non-quadratic form of the Rayleigh quotient, convergence will not generally be obtained in a finite number of iterations and the actual number of iterations required to attain a given accuracy depends upon the computation of β_i as well as the initial guess x_0. In addition, search directions p_i and p_j are no longer A-conjugate unless $|i - j| = 1$. This means that only adjacent search directions remain A-conjugate with each other.

There are two critical steps in the scheme. One is the computing of α_i by exact linear search and the other is the computing of β_i.

It is already known that an exact linear search of α_i along the direction p_i is equivalent to a Ritz analysis of (1) in the subspace spanned by x_i and p_i. More precisely, λ_{i+1} and $[1, \alpha_i]^{\mathrm{T}}$ are the minimum Ritz value and corresponding Ritz vector respectively, i.e.

$$A^* \begin{bmatrix} 1 \\ \alpha_i \end{bmatrix} = \lambda_{i+1} I^* \begin{bmatrix} 1 \\ \alpha_i \end{bmatrix} \tag{14}$$

where

$$A^* = [x_i, p_i]^{\mathrm{T}} A [x_i, p_i] = \begin{bmatrix} \lambda_i & a_i \\ a_i & b_i \end{bmatrix}$$

$$I^* = [x_i, p_i]^{\mathrm{T}} [x_i, p_i] = \begin{bmatrix} 1 & c_i \\ c_i & d_i \end{bmatrix}$$

in which

$$a_i = x_i^{\mathrm{T}} Ap_i, \quad b_i = p_i^{\mathrm{T}} Ap_i, \quad c_i = x_i^{\mathrm{T}} p_i, \quad d_i = p_i^{\mathrm{T}} p_i$$

By solving (14), we may get two groups of $(\alpha_i, \lambda_{i+1})$, which respectively correspond to the maximum and minimum value points of the Rayleigh quotient in the subspace $\{x_i, p_i\}$. Further study (e.g. Reference 9) shows that the following forms of α_i and λ_{i+1} ensure that α_i minimizes $\lambda(x_i + \alpha_i p_i)$

$$\alpha_i = \frac{\lambda_i d_i - b_i + \sqrt{\delta}}{2(b_i c_i - a_i d_i)} \tag{15}$$

where

$$\delta = (\lambda_i d_i - b_i)^2 - 4(b_i c_i - a_i d_i)(a_i - \lambda_i c_i)$$

and

$$\lambda_{i+1} = \frac{\lambda_i + a_i \alpha_i}{1 + c_i \alpha_i} \tag{16}$$

Meanwhile, the norm of x^*_{i+1}, γ_{i+1} may readily be attained as follows

$$\gamma_{i+1} = \| x^*_{i+1} \| = \sqrt{1 + 2c_i \alpha_i + d_i \alpha_i^2} \tag{17}$$

Therefore, by using (16) and (17) instead of the direct computation of $\lambda_{i+1} = x^T_{i+1} A x_{i+1}$ and $\gamma_{i+1} = \|x_{i+1}\|$, at least two inner products of vectors can be saved.

The computation of β_i, the other key step of the method, will be discussed in the next section.

The convergence rate of iterative optimization methods depends upon the condition number of the Hessian matrix at the minimum point $x^* = \phi_1$. Since

$$H(\phi_1) = A - \omega_1 I \tag{18}$$

regardless of the singularity of $H(\phi_1)$, we obtain

$$\kappa(H(\phi_1)) = \frac{\omega_n - \omega_1}{\omega_2 - \omega_1} \tag{19}$$

i.e. the condition number of $H(\phi_1)$ is related to the relative separation of ω_1 from the rest of the spectrum. Recalling that the convergence rate of the inverse power method depends only on the ratio of ω_2/ω_1, it suggests that conjugate gradient methods may be inferior to the inverse power method in terms of convergence rate, especially for ill-conditioned problems.

2.3. *The CG algorithm*

Summing up the discussion above, we list the CG algorithm as follows:

The conjugate gradient algorithm for solving the smallest eigenpair

```
1. Initialisation
     Given an initial guess $x_0$ with $\|x_0\| = 1$.
     Compute $w_0 = Ax_0$ and Rayleigh quotient $\lambda_0 = w_0^T x_0$.
     Construct the initial gradient direction $g_0 = w_0 - \lambda_0 x_0$.
     Set $p_0 = g_0$ and $z_0 = Ap_0$.
2. For i = 0, 1, 2, . . . , Do
     Compute $\alpha_i$, $\lambda_{i+1}$ and $\gamma_{i+1}$ by (15)–(17) in which
          $a_i = z_i^T x_i, \quad b_i = z_i^T p_i, \quad c_i = x_i^T p_i, \quad d_i = p_i^T p_i$
```

Update x_{i+1}, w_{i+1} and g_{i+1}

$$x_{i+1} = (x_i + \alpha_i p_i)/\gamma_{i+1}$$

$$w_{i+1} = Ax_{i+1} = (w_i + \alpha_i z_i)/\gamma_{i+1}$$

$$g_{i+1} = Ax_{i+1} - \lambda_{i+1}x_{i+1} = w_{i+1} - \lambda_{i+1}x_{i+1}$$

Check convergence
if

$$\frac{\|Ax_{i+1} - \lambda_{i+1}x_{i+1}\|}{\lambda_{i+1}} = \frac{\|g_{i+1}\|}{\lambda_{i+1}} < \varepsilon$$

then stop; otherwise continue.
Construct the new search direction p_{i+1} and z_{i+1}

$$p_{i+1} = g_{i+1} + \beta_i p_i, \quad z_{i+1} = Ap_{i+1}$$

where

$$\beta_i = -\frac{g_{i+1}^T A p_i}{p_i^T A p_i} = -\frac{g_{i+1}^T z_i}{b_i}$$

If we define a unit of vector operation to be an inner product of two vectors or a scalar multiplication of a vector, the above algorithm requires a total of 12 units of vector operations as well as one matrix–vector multiplication, and five vectors, x, p, g, w, z must be kept in core. Here we use two more vectors w and z than the standard CG algorithm so that the number of matrix–vector operations is reduced to a minimum.

It should be mentioned that the above algorithm is based on the constrained optimization (11), i.e. the approximate solutions x_i are maintained as unity at each iteration. If we do not normalize x_i at each iterative step until x_i converges, two more units of vector operations may be eliminated, provided that g_i takes the form of $g_i = Ax_i - \lambda_i x_i$ rather than $g_i = (Ax_i - \lambda_i x_i)/x_i^T x_i$, but numerical instability might occur if $\|x_i\|$ excessively increases and becomes too large before it begins to converge. For this reason, the CG algorithm currently used is thus recommended.

For generalized eigenvalue problems the above algorithm does not change except for the following

$$c_i = x_i^T B p_i, \quad d_i = p_i^T B p_i$$

and

$$g_{i+1} = Ax_{i+1} - \lambda_{i+1}Bx_{i+1} \tag{20}$$

3. SOME VARIANTS OF CG SCHEMES

The search directions generated in the CG algorithm cannot maintain A-conjugacy since the Rayleigh quotient is not a quadratic function. This makes it possible to adopt several different formulas in computing β_i, each giving rise to a particular CG scheme. In this section, we will present two other forms together with the original one, and the asymptotic behaviours of the corresponding schemes will also be analyzed.

3.1. *Three forms of β_i*

The three forms of β_i which have already been given in the literature are listed below.

Form 1 (Perdon and Gambolati[9]):

$$\beta_i^{(1)} = -\frac{g_{i+1}^T A p_i}{p_i^T A p_i} \tag{21}$$

Form 2 (Bradbury and Fletcher,[7] Ruhe[13]):

$$\beta_i^{(2)} = \frac{g_{i+1}^T g_{i+1}}{g_i^T g_i} \tag{22}$$

Form 3 (Polak[15]):

$$\beta_i^{(3)} = \frac{g_{i+1}^T (g_{i+1} - g_i)}{g_i^T g_i} \tag{23}$$

The first form is directly obtained from the conjugate condition of p_i and p_{i+1}, whilst both of the second and third forms are analogous with the forms for solving linear algebraic equations.

The number of main floating point operations for the first and last forms of β_i is almost the same, and only one inner product is actually required in these cases. The second one hardly needs any computation because $g_{i+1}^T g_{i+1}$ is already available after convergence check is made.

As x_i is sufficiently close to ϕ_1, the Hessian matrix $H(x)$ tends to $A - \omega_1 I$, rather than A. Therefore, it is reasonably concluded that the CG method with $\beta_i = \beta_i^{(1)}$ will exhibit a slow convergence. In fact, the other forms are more frequently applied in practice. The second form is used in many papers,[7,13,16,17] whereas the last one is adopted in References 11, 15 and 18. But it is somewhat difficult to compare the convergence rates of these two forms theoretically. Next, we will analyze the asymptotic behaviours of these two forms; particularly the last one. A similar analysis is conducted in Reference 12.

3.2. *Asymptotic behaviour analysis*

First of all, we present some basic properties of the CG algorithm. By the definitions of the Rayleigh quotient and its gradient, it is easy to prove that

$$x_i^T g_i = 0 \tag{24}$$

The condition that α_i minimizes $\lambda(x_i + \alpha_i p_i)$ implies that

$$x_i^T g_{i+1} = 0 \tag{25}$$

and

$$p_i^T g_{i+1} = 0 \tag{26}$$

Multiplying

$$p_{i+1} = g_{i+1} + \beta_i p_i$$

by g_{i+1}^{T} and making use of (26), we obtain

$$g_{i+1}^{\mathrm{T}} p_{i+1} = g_{i+1}^{\mathrm{T}} g_{i+1}$$

i.e.

$$g_i^{\mathrm{T}} p_i = g_i^{\mathrm{T}} g_i \tag{27}$$

Note that properties (24)–(27) will remain true no matter what form of β_i is chosen. When x_i is sufficiently close to ϕ_1, the following expressions hold approximately

$$\lambda_{i+1} = \lambda_i = \omega_1 \quad \text{and} \quad \gamma_{i+1} = \gamma_i = 1$$

Thus

$$\begin{aligned} g_{i+1} &= (A - \omega_1 I) x_{i+1} = (A - \omega_1 I)(x_i + \alpha_i p_i) \\ &= g_i + \alpha_i (A - \omega_1 I) p_i \end{aligned}$$

or

$$g_{i+1} - g_i = \alpha_i (A - \omega_1 I) p_i \tag{28}$$

Therefore,

$$g_i^{\mathrm{T}} g_i = -p_i^{\mathrm{T}} (g_{i+1} - g_i) = -\alpha_i p_i^{\mathrm{T}} (A - \omega_1 I) p_i \tag{29}$$

and

$$g_{i+1}^{\mathrm{T}} (g_{i+1} - g_i) = \alpha_i g_{i+1}^{\mathrm{T}} (A - \omega_1 I) p_i \tag{30}$$

We can now rewrite $\beta_i^{(3)}$ of equation (23) as

$$\beta_i^{(3)} = -\frac{g_{i+1}^{\mathrm{T}} (A - \omega_1 I) p_i}{p_i^{\mathrm{T}} (A - \omega_1 I) p_i} \tag{31}$$

which means that p_i, p_{i+1} become $(A - \omega_1 I)$-conjugate. The above expression is first presented in Reference 12 but based on additional assumptions.

The first order approximation to α_i can be found as[13]

$$\alpha_i = -\frac{p_i^{\mathrm{T}} (A - \omega_1 I) x_i}{p_i^{\mathrm{T}} (A - \omega_1 I) p_i} \tag{32}$$

Based on both (31) and (32), it cannot be concluded that the third CG scheme (with $\beta_i = \beta_i^{(3)}$) for solving the eigenvalue problem is asymptotically equivalent to the CG method for solving the following linear equation

$$(A - \omega_1 I) x = 0 \tag{33}$$

unless most of the important properties, such as

$$p_i^{\mathrm{T}} (A - \omega_1 I) p_j = 0, \quad g_i^{\mathrm{T}} g_j = 0 \quad \text{for } i \neq j$$

also asymptotically hold. Computational experiments do reveal, however, that $(A - \omega_1 I)$-conjugacy of p_{i+1} with other previously constructed directions $p_{i-1}, p_{i-2}, \ldots$, and orthogonality of g_{i+1} with $g_i, g_{i-1}, \ldots$ are gradually established. Therefore, it is reasonable to believe that the asymptotic equivalence between the third CG scheme and the CG in the linear system case exists.

Furthermore, since $H(x_{i+1})$ takes the form

$$H(x_{i+1}) = A - \lambda_{i+1}I - g_{i+1}x_{i+1}^T - x_{i+1}g_{i+1}^T \tag{34}$$

it follows from (24) and (26) that

$$p_i^T H(x_{i+1})p_i = p_i^T(A - \lambda_{i+1}I)p_i \tag{35}$$

and

$$\begin{aligned} g_{i+1}^T H(x_{i+1})p_i &= g_{i+1}^T(A - \lambda_{i+1}I)p_i - g_{i+1}^T g_{i+1} x_{i+1}^T p_i \\ &\approx g_{i+1}^T(A - \lambda_{i+1}I)p_i \end{aligned} \tag{36}$$

in which $g_{i+1}^T g_{i+1} x_{i+1}^T p_i$ is a higher order term which can be neglected. Therefore, $\beta_i^{(3)}$ can also be considered as an asymptotic equivalence to the following form of β_i

$$\beta_i = -\frac{g_{i+1}^T H(x_{i+1})p_i}{p_i^T H(x_{i+1})p_i} \tag{37}$$

which makes p_i and p_{i+1} $H(x_{i+1})$-conjugate.

$\beta_i^{(2)}$, on the other hand, has a relation with $\beta_i^{(3)}$ as

$$\beta_i^{(2)} = \beta_i^{(3)} + \frac{g_{i+1}^T g_i}{g_i^T g_i}$$

In the third CG scheme, g_{i+1} and g_i may become asymptotically orthogonal with each other, and thus $\beta_i^{(2)} = \beta_i^{(3)}$. However, in the second CG scheme (with $\beta_i = \beta_i^{(2)}$), the orthogonality between g_{i+1} and g_i cannot be guaranteed, and consequently, for the second CG scheme, the same asymptotic behaviour as the third scheme cannot be concluded here.

For further understanding of the behaviour of the second scheme, and comparison with the third one, more numerical experiments are thus necessary which will be undertaken in Section 5.

3.3. Restarted schemes

Generally speaking, a large number of iterations is required to obtain a solution which satisfies a given accuracy. As $\{p_i\}$ are not A-, or $H(x)$-conjugate, for large i, $(p_0, p_1, \ldots, p_i)$ may become nearly linearly dependent or linearly dependent, which may cause convergence of the iterative process to be very slow. A restarted version at this stage may accelerate the process. In this case, the process restarts after every m iterative steps if it has not yet converged.

In general, the restart number m should be prescribed in advance, and will have an effect on the overall efficiency of the scheme. For a large value, the algorithm tends to the full version. For a small value it may also be very slow because when the restart number equals 1, it recovers the steepest descent method, which is considered to be inefficient. Therefore, there should exist an optimal restart number which may minimize the total CPU requirement. Nevertheless, this number is problem-dependent, and more importantly, it cannot be estimated in advance. Therefore, an approximated optimal restart number could only be determined by empirical approaches, if necessary.

3.4. Other Considerations

One of the important aims of the restarted CG version is to avoid the linear dependence of search directions. A natural alternative is to force p_{i+1} to be A- or $H(x)$-conjugate with the previous directions at each iteration. However, this not only increases the computation time but also requires that all search directions are stored in memory, which places a heavy burden on computer resources in practice. An amendment is to keep the current direction to be conjugate with some of the previous directions but not all. Our numerical tests show, unexpectedly, that both approaches are much inferior in terms of convergence rate to the original ones, i.e. only ensuring adjacent directions to be conjugate. This illustrates that the generation of a suitable search direction is extremely important to any scheme.

4. PRECONDITIONED CONJUGATE GRADIENT SCHEME (PCG)

4.1 The algorithm of PCG

Preconditioning is a technique to accelerate the convergence of the CG scheme. The key step in the implementation of preconditioning is to develop the corresponding algorithm. Following the work of Gambolati *et al.*,[18] we perform the variable transformation

$$y = Lx \tag{38a}$$

or

$$x = L^{-1}y \tag{38b}$$

where L is an auxiliary non-singular symmetric matrix. By substituting (38b) into (2) (suppose that $B = I$), we obtain

$$\lambda(y) = \frac{y^{\mathrm{T}}L^{-1}AL^{-1}y}{y^{\mathrm{T}}L^{-1}L^{-1}y} = \frac{y^{\mathrm{T}}A^*y}{y^{\mathrm{T}}My} \tag{39}$$

where $A^* = L^{-1}AL^{-1}$, $M = L^{-1}L^{-1}$ both of which are SPD matrices. Here we obtain a new generalized eigenproblem which can be easily solved by the CG scheme. Restoring the original variables through (38a), we attain the algorithm of the preconditioned CG which is the same as the original one except that computations of the search directions are changed to

$$p_0 = Mg_0, \quad p_{i+1} = Mg_{i+1} + \beta_i p_i \tag{40}$$

and the corresponding three forms of β_i are also changed to

$$\beta_i^{(1)} = -\frac{g_{i+1}^{\mathrm{T}}MAp_i}{p_i^{\mathrm{T}}Ap_i}, \quad \beta_i^{(2)} = \frac{g_{i+1}^{\mathrm{T}}Mg_{i+1}}{g_i^{\mathrm{T}}Mg_i}, \quad \beta_i^{(3)} = \frac{g_{i+1}^{\mathrm{T}}M(g_{i+1} - g_i)}{g_i^{\mathrm{T}}Mg_i} \tag{41a, b, c}$$

respectively. We note that the first two formulas need the same amount of work in this case, but the last one needs one more inner product and probably one more vector work space.

The efficiency of PCG schemes depends upon the choice of the preconditioning matrix M. There are a number of choices of M, ranging from very simple to very complicated forms, among which the incomplete Cholesky decomposition that preserves exactly the non-zero pattern of A (called IC(0))[19] exhibits both efficiency and robustness. The CG method with IC(0) as preconditioner is denoted by IC-CG(0). In this paper, one particular preconditioning matrix $M = A^{-1}$, i.e. the complete Cholesky decomposition (CC) of A as preconditioner, is also used for a purely theoretical purpose. The corresponding PCG approach is denoted by CC-CG.

4.2. Estimation of the convergence rate

The convergence rate of PCG is, as stated in Section 2, dependent on the condition number of the Hessian matrix in $x = \omega_1$ as follows

$$H^*(\omega_1) = M(A - \omega_1 I) = A^* - \omega_1 M \tag{42}$$

For a general choice of M, it is not easy to compare PCG directly with the non-preconditioned CG. But in the case of CC-CG where $M = A^{-1}$, $A^* = I$, and

$$H^*(\omega_1) = I - \omega_1 A^{-1} \tag{43}$$

the condition number is thus equal to[18]

$$\kappa(H^*(\omega_1)) = \frac{1 - \omega_1/\omega_n}{1 - \omega_1/\omega_2} = \frac{\omega_n - \omega_1}{\omega_2 - \omega_1}\cdot\frac{\omega_2}{\omega_n} = \kappa(H(\omega_1))\cdot\frac{\omega_2}{\omega_n} \tag{44}$$

For many practical problems, especially for large scale cases, $\omega_2 \ll \omega_n$, which results in $\kappa(H^*(\omega_1)) \ll \kappa(H(\omega_1))$. Therefore we expect PCG to be much faster than CG.

The condition number of $H^*(\omega_1)$ may not entirely reflect the actual behaviour of PCG. Next we will compare CC-CG directly with the inverse power method. In this case,

$$p_{i+1} = A^{-1} g_{i+1} + \beta_i p_i \tag{45}$$

For the first scheme, because of (26), β_i becomes

$$\beta_i^{(1)} = -\frac{g_{i+1}^{\mathrm{T}} A^{-1} A p_i}{p_i^{\mathrm{T}} A p_i} = 0 \tag{46}$$

Therefore

$$\begin{aligned} p_{i+1} &= A^{-1} g_{i+1} = A^{-1}(A x_{i+1} - \lambda_{i+1} x_{i+1}) \\ &= x_{i+1} - \lambda_{i+1} A^{-1} x_{i+1} \end{aligned} \tag{47}$$

which means

$$span\{x_i, p_i\} \equiv span\{x_i, A^{-1}x_i\} \tag{48}$$

The above results make the first CC-CG scheme equivalent to the following iterative algorithm:

```
1. Given initial guess $x_0$.
2. For i = 0, 1, 2, . . . DO
    Inverse iteration
        $\bar{x}_{i+1} = A^{-1} x_i$
    Ritz analysis in subspace $\{x_i, \bar{x}_{i+1}\}$
        $\bar{A}\bar{\phi} = \bar{\lambda}\bar{I}\bar{\phi}$
    where
        $\bar{A} = [x_i, \bar{x}_{i+1}]^{\mathrm{T}} A [x_i, \bar{x}_{i+1}]$
        $\bar{I} = [x_i, \bar{x}_{i+1}]^{\mathrm{T}} [x_i, \bar{x}_{i+1}]$
    supposing that $(\bar{\lambda}, \bar{\phi})$ is the smallest Ritz pair.
    Approximate solution updating
        $x_{i+1} = [x_i, \bar{x}_{i+1}]\bar{\phi}$
```

It is well known that in the standard inverse power method, the Ritz value or approximated eigenvalue of the $(i + 1)$th iteration is

$$\bar{\lambda}_{i+1} = \frac{\bar{x}_{i+1}^{\mathrm{T}} A \bar{x}_{i+1}}{\bar{x}_{i+1}^{\mathrm{T}} \bar{x}_{i+1}} \tag{49}$$

In the above algorithm, $\bar{\lambda}$ is the minimum of the following Rayleigh quotient

$$\lambda(\phi) = \frac{\phi^{\mathrm{T}} \bar{A} \phi}{\phi^{\mathrm{T}} \bar{I} \phi} \quad (\forall \phi \in R^2, \phi \neq 0) \tag{50}$$

If we take $\phi = [0, 1]^{\mathrm{T}}$, the corresponding Ritz value will be

$$\lambda = \frac{\bar{x}_{i+1}^{\mathrm{T}} A \bar{x}_{i+1}}{\bar{x}_{i+1}^{\mathrm{T}} \bar{x}_{i+1}} = \bar{\lambda}_{i+1} \tag{51}$$

Therefore we have $\bar{\lambda} < \bar{\lambda}_{i+1}$. This means that the CC-CG scheme 1 (with $\beta_i = \beta_i^{(1')}$) is an acceleration to the standard inverse power method (SIP). Furthermore, if A is a second order SPD matrix, the algorithm needs only one iteration to obtain the exact eigenpairs of A. This implies that the convergence speed of the above algorithm may increase from ω_2/ω_1 of the standard inverse power method to ω_3/ω_1. We note that a similar accelerated inverse power method (AIP) is proposed by Roberti.[20]

Finally, regardless of the singularity of $(A - \omega_1 I)$, it is apparent from (38), that $M = (A - \omega_1 I)^{-1}$ will be the 'optimal' preconditioner in terms of convergence rate, to construct preconditioners which indicates that $M = A^{-1}$ will no longer be the 'best' preconditioner and other preconditioners such as IC(0) may occasionally be better choices than A^{-1}, as pointed out in Reference 12. However, since ω_1 is unknown, and we believe that $M = A^{-1}$ will perform better than other preconditioners in most cases, the actual performance of PCG with a generally selected preconditioning matrix M is thus expected to be between that of the standard CG, which may be much inferior to the inverse power method, and the accelerated inverse power method.

5. NUMERICAL EXPERIMENTS

In this section we choose some test and realistic problems upon which to carry out numerical experiments to firstly illustrate the results presented in the previous sections, and secondly to further investigate the performance of the various methods.

5.1. Problem descriptions

The tests are carried on a SUN SPARC 2 workstation in double precision, and the Fortran codes used are optimally compiled and linked.

The first problem is a clamped beam (1×10) modelled by 100×100 4-node plane linear elements. The second and third problems come from practical engineering situations, and the two meshes are shown in Figures 1 and 2 respectively. Table I summarizes the details of all three problems, including the number of nodes, elements, degrees of freedom, non-zero entries of global stiffness matrices as well as storage required (in *words*) by direct solvers. We note that the listed storage requirements of the last two problems are for the data after the nodes are renumbered by the reverse Cuthill–McKee algorithm (RCM). It is obvious that a much higher memory is required by direct solvers for large problems, even after node renumbering, compared with iterative solvers.

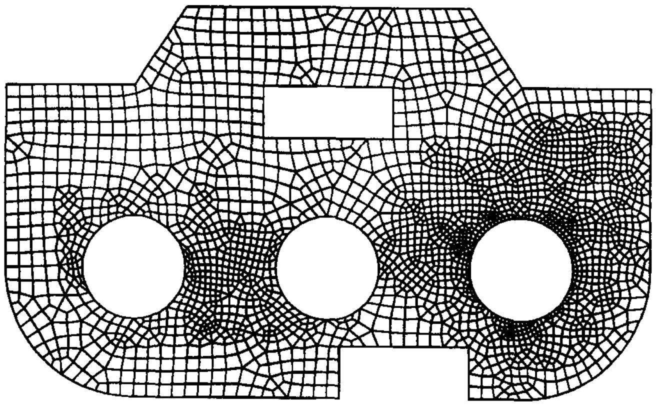

Figure 1. The finite element mesh of problem 2: elements = 2420, nodes = 2587, DOFs = 4876, element type = 4-node plane element

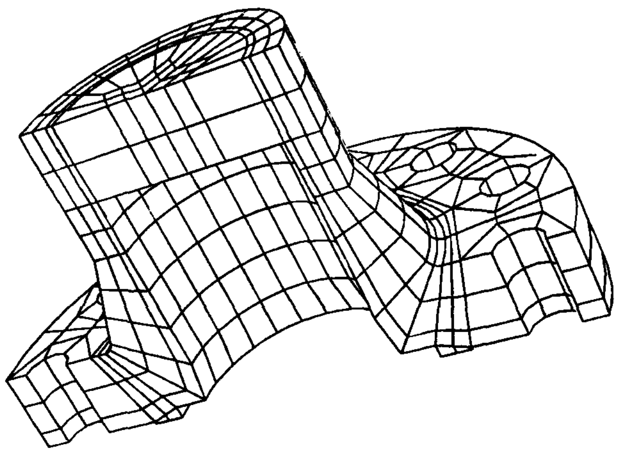

Figure 2. The finite element mesh of problem 3: elements = 684, nodes = 3885, DOFs = 11183, element type = 20-node brick element

In the following computations, the mass matrices are simply set to identity values, i.e. $B = I$, and the initial guess x_0 are chosen to be random vectors in all cases. The tolerance ε is set to be 10^{-5}.

5.2. Convergence comparison of CG with PCG

For examining the effects of preconditioning on the convergence of the CG methods, two preconditioners are used; one is Jacobi, i.e. diagonal and the other incomplete Cholesky decomposition without fill-in IC(0). Both PCG algorithms denoted by DP-CG and IC-CG with the second formula for calculating β_i are compared with non-preconditioned CG for solving the smallest eigenpair of all three problems. The convergence histories, as shown in Figure 3 for problem 2, for instance, illustrate that high quality preconditioning techniques are extremely important to achieve rapid convergence of the CG method. The non-preconditioned CG scheme

Table I. Details of the finite element models

Problem	Nodes	Elements	DOFs	Nonzero entries	Skyline storage
1	10021	10000	20200	189497	4070296
2	2587	2420	4876	44178	459086
3	3885	684	11183	761625	58942639

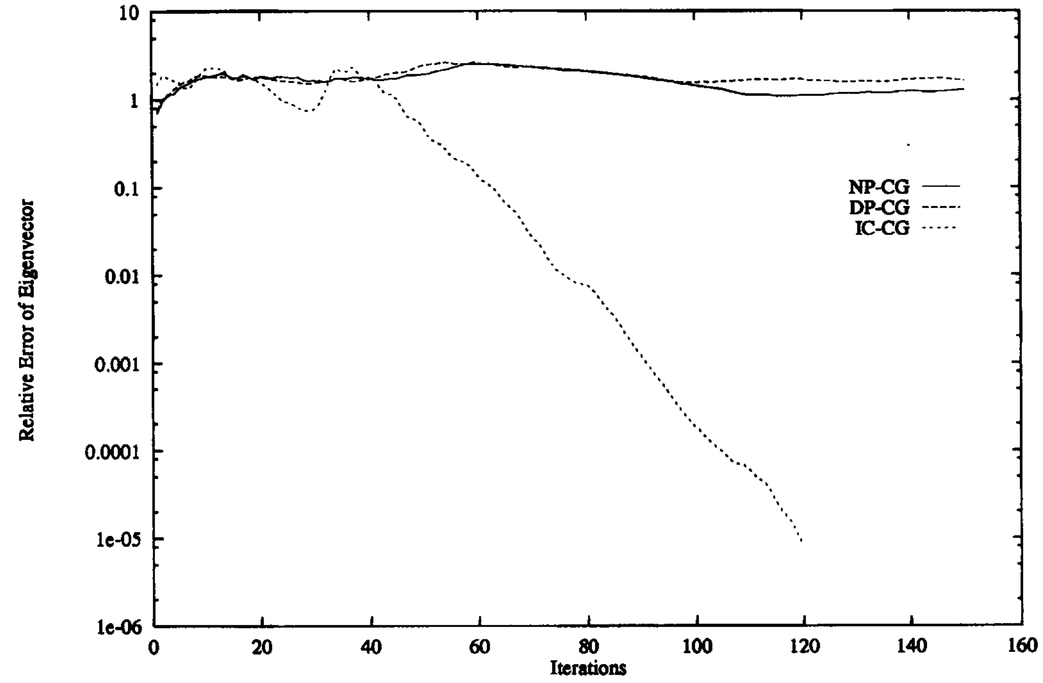

Figure 3. Convergence histories of NP-, DP- and IC-CG scheme 2 (with $\beta_i = \beta_i^{(2)}$) for problem 2

is very slow and cannot be applied in practice. In addition, diagonal preconditioning is almost as inefficient as non-preconditioning. In contrast, IC(0) exhibits quite good efficiency in these cases.

5.3. Numerical comparison of different CG schemes

5.3.1. Full version. We first compare the full versions of the three schemes for evaluating the smallest eigenpair of all three problems. The corresponding convergence histories are represented in Figures 4–6, and the iterations and CPU time requirements for the given tolerance ($\varepsilon = 10^{-5}$) can also be found in Table III.

From the convergence history curves we see that the whole iterative procedure may be divided into two stages. In the first stage, the relative error norms show some oscillation and the values are always high. This period is termed the *transient stage*. After the transient stage, the iterative procedures begin to converge with approximately linear rates, although some small oscillations may still occur. This period is thus termed the *convergent stage*.

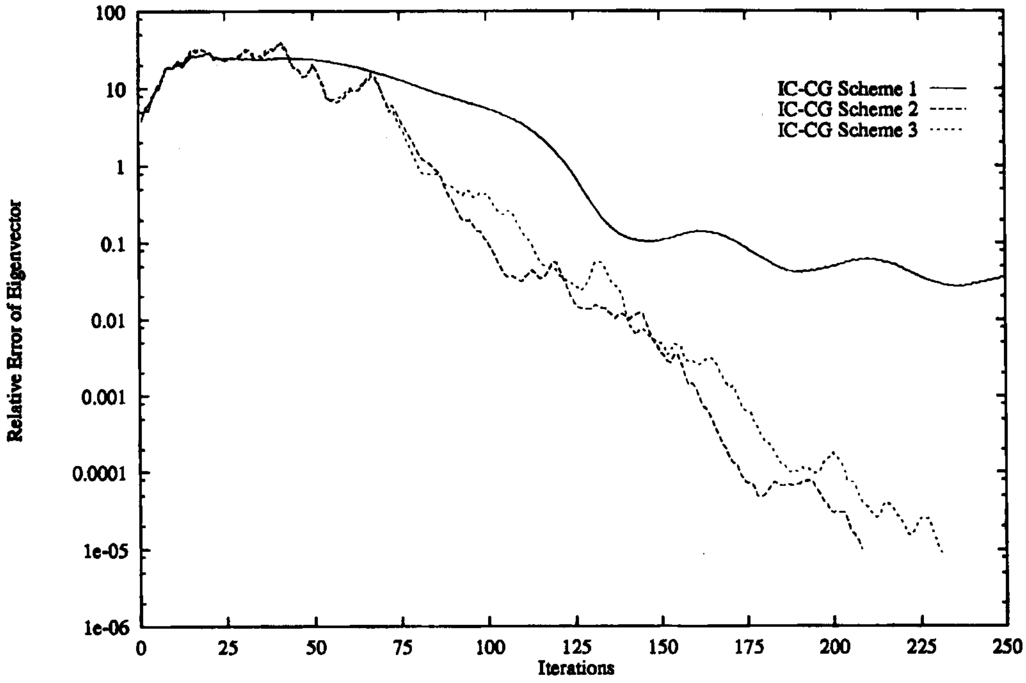

Figure 4. Convergence histories of three IC-CG schemes for problem 1

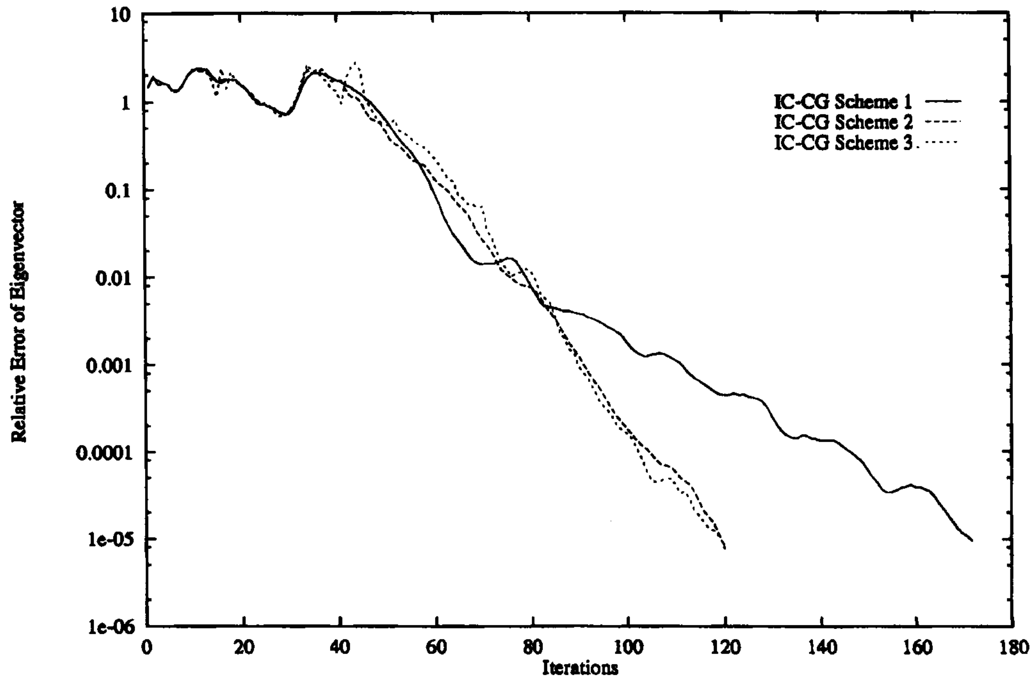

Figure 5. Convergence histories of three IC-CG schemes for problem 2

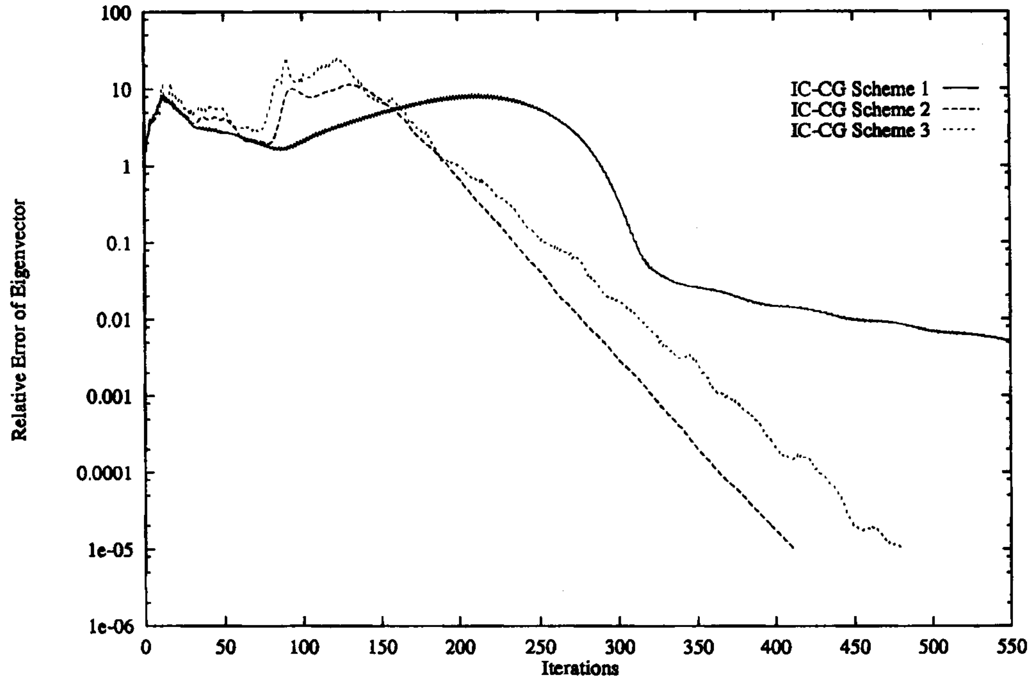

Figure 6. Convergence histories of three IC-CG schemes for problem 3

It is interesting to note that the length of the transient stage for the second and third schemes is almost the same, and that the error curves are nearly identical. At the convergent stage, both schemes also exhibit a similar linear convergence performance, but scheme 3 needs slightly more iterations and CPU time in problems 1 and 3, revealing that the second CG scheme has at least a similar asymptotic behaviour as the third scheme, which is proved to be asymptotically equivalent to the CG method for the solution of the linear equations $(A - \omega_1 I)x = 0$.

Scheme 1, on the other hand, shows slightly different behaviour; it sometimes has a non-linear convergence rate at the intermediate stage, such as the case from steps 250–350 in problem 3, and finally comes to the asymptotically convergent stage but with much lower convergence rate in comparison with the other two schemes; and also with an obvious periodic oscillation, resulting in a much lower overall efficiency. For instance, the first scheme needs three times as many iterations as the second one to converge in problem 3.

5.3.2. Restarted version. Different restart numbers *m* will inevitably have some effect on the overall efficiency of the schemes. We record the total number of iterations required to reach the given accuracy for different restart numbers of the three IC-CG schemes when solving the smallest eigenpairs of all three problems, and then obtain an approximated optimal restart number for each problem.

Table II lists the percentage improvement of the approximated optimal restarted CG schemes over the full versions for all problems. It appears that considerable improvements can be made for scheme 1, and about 10–20 per cent for schemes 2 and 3. This result reveals, on the one hand, that the linear independence between the search directions constructed in the first CG scheme becomes poor as the iteration procedure continues, while it confirms, on the other hand, that the independence in the second and third schemes is generally maintained.

From a practical point of view, the restart number must be specified *a priori*. However, the optimal number will be affected by several factors and may not be theoretically obtained. Our numerical experiment suggests, however, that if the process restarts when the eigenvalue begins to converge, approximated optimal results are always obtained. In these restarted versions we find that scheme 1 still compares unfavourably with the other two schemes.

5.4. Comparison of PCG schemes with inverse power methods

In order to illustrate our theoretical results obtained in Section 4.2 concerning the convergence rate of PCG schemes, we compare the actual performance of the PCG scheme with that of the standard inverse power method and the accelerated version. For PCG, two kinds of preconditioning matrices are used: (1) IC(0) and (2) CC. Due to the large memory requirements of the CC preconditioning and the two inverse power methods in the case of problem 3, which is far beyond the available memory resource, only the first two problems are examined. The corresponding iterations and total CPU times are listed in Table III.

It is obvious that although the convergence rate of IC-CG(0) is much slower than CC-CG, SIP and AIP, the overall efficiency is the best in terms of total computer time and storage requirements. It is observed that CC-CG schemes 2 and 3 are slightly less expensive in CPU cost for problem 2, but they also require much larger memory space.

CC-CG schemes have similar storage requirements to SIP and AIP. But normally CC-CG (scheme 1) has a faster convergence rate than SIP, and exactly the same rate as AIP, as deduced in Section 4.2. This feature is clearly demonstrated in Figure 7 which depicts the convergence histories of CC-CG, SIP and AIP for problem 2. Three iteration deficiencies between CC-CG

Table II. The percentage improvement of (approximate) optimal restarted versions

Scheme	Problem 1	Problem 2	Problem 3
1	61·8	39·4	67·7
2	25·0	12·4	18·1
3	22·7	11·8	15·4

Table III. Iterations and CPU time of PCG schemes, standard inverse power method (SIP) and accelerated inverse power method (AIP)

	Preconditioning type	Problem 1		Problem 2		Problem 3	
		Iterations	CPU (s)	Iterations	CPU (s)	Iterations	CPU (s)
PCG	IC(0)	> 250	—	172	39·14	> 550	—
$\beta_i = \beta_i^{(1)}$	CC	5	332·57	21	31·90	—	—
PCG	IC(0)	208	211·44	120	27·29	412	1262·72
$\beta_i = \beta_i^{(2)}$	CC	5	332·22	12	25·61	—	—
PCG	IC(0)	231	240·57	120	27·69	480	1469·58
$\beta_i = \beta_i^{(3)}$	CC	5	332·99	12	25·67	—	—
SIP	—	6	330·41	29	34·71	—	—
AIP	—	3	323·86	18	27·63	—	—

scheme 1 and AIP result from slightly different strategies employed in the initialisation of the methods. Since schemes 2 and 3 are superior to scheme 1, it is not surprising to see that they show improved convergence to AIP in problem 2.

Finally, we investigate the behaviours of the PCG schemes using the particular preconditioning matrix $M = (A - \mu I)^{-1}$, where μ is the shifting value. As μ varies from 0 to ω_1, M will change from A^{-1}, the CC preconditioner, to $(A - \omega_1 I)^{-1}$, the 'optimal' preconditioner. Corresponding to this new preconditioning matrix, we use the shifted inverse power method and its accelerated version to compare with the PCG methods. Table IV summarizes the total iterations of the three PCG schemes, shifted SIP and AIP with four different shifting values, μ, for problem 2. Clearly, the iterations for all methods are considerably reduced as μ approaches ω_1, which, in fact, could be somehow explained by the well-known fact that the Rayleigh quotient iteration (a shifted inverse power like method) at the vicinity of the eigenvalue considered has cubic convergence rate.[21]

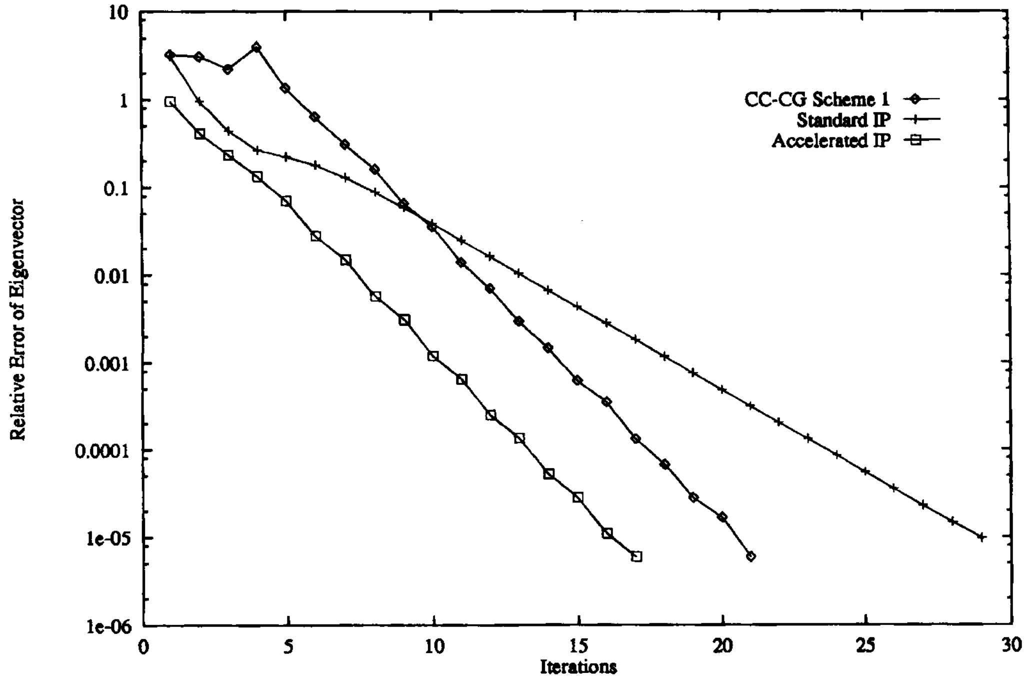

Figure 7. Convergence histories of CC-CG scheme 2, standard inverse power method (SIP) and accelerated inverse power method (AIP) for problem 2

Table IV. Iterations of three PCG schemes ($M = (A - \mu I)^{-1}$), shifted SIP and AIP with different shifted values μ for problem 2 ($\omega_1 = 0{\cdot}2968$)

Shifted value μ	PCG $\beta_i = \beta_i^{(1)}$	PCG $\beta_i = \beta_i^{(2)}$	PCG $\beta_i = \beta_i^{(3)}$	SIP	AIP
0·0000	21	12	12	29	18
0·2000	17	9	9	14	9
0·2900	9	4	4	6	4
0·2968	2	2	2	3	2

6. CONCLUSIONS AND REMARKS

From the above numerical experiments it appears that both second and third schemes exhibit similar performances, but the former may show slightly more overall efficiency than the latter. It also implies that the second scheme has at least a similar asymptotic behaviour as the third scheme, although there is no theoretical explanation provided at present. The first scheme, on the other hand, performs very poor compared with the other two. It should be mentioned that we also test another form of β_i which makes p_i and p_{i+1} $H(x)$-conjugate, but the results are worse than the third form. This fact together with the actual performance of the second scheme suggests that for solving problems with a non-quadratic objective function, keeping adjacent search directions A- or $H(x)$ conjugate in conjugate direction type methods may not be necessary.

'Good' preconditioning techniques are vital to achieve rapid convergence of the CG schemes. PCG schemes 2 and 3 with IC(0) as the preconditioning matrix overperform CC-CG, SIP and AIP, exhibiting the best overall efficiency in terms of total computer time and storage requirements. As far as the convergence rate is concerned, when preconditioning matrix M is close to the 'optimal' matrix $(A - \omega_1 I)^{-1}$, the number of iterations may be significantly reduced. However, for a generally selected preconditioner, the performance of the PCG scheme may not be superior to AIP.

Although the current work is restricted to the discussion of the theoretical and computational aspects of CG methods when applied to solving the smallest eigenpair, these methods can also be used to evaluate several smallest eigenpairs with the aid of deflation or subspace techniques.[8,10] In this case, most of the results obtained in this paper are still valid, or only need some minor amendments.

Finally, it should be pointed out that as the CG method is only a local optimization approach, the computed eigenvalue by CG cannot be guaranteed to be the minimum, or the desired one. Therefore, more work is needed to resolve this problem due to its important practical interest.

REFERENCES

1. K. J. Bathe and E. L. Wilson, *Numerical Methods in Finite Element Analysis*, Prentice-Hall, Englewood Cliffs, N.J., 1976.
2. C. Lanczos, 'An iteration method for the solution of eigenvalue problems of linear differential and integral operators', *J. Res. Nat. Bureau Standards*, **45**, 255–282 (1950).
3. B. N. Parlett, *The Symmetric Eigenvalue Problem*, Prentice-Hall, Englewood Cliffs, N.J., 1980.
4. G. Gambolati and M. Putti. 'A comparison of Lanczos and optimization methods in the partial solution of sparse symmetric eigenproblems', *Int. j. numer. methods eng.*, **37**, 605–621 (1994).
5. M. R. Hestenes and W. Karush, 'A method of gradients for the calculation of the characteristic roots and vectors of a real symmetric matrix', *J. Res. Nat. Bureau Standards*, **47**, 45–61 (1951).
6. D. K. Faddeev and V. N. Faddeeva, *Computational Methods of Linear Algebra*, Freeman, San Francisco, 1963.
7. W. W. Bradbury and R. Fletcher. 'New iterative methods for solution of the eigenproblem', *Numer. Math.*, **9**, 259–267 (1966).
8. J. H. Wilkinson, *The Algebraic Eigenvalue Problem*, Oxford University Press, London, 1965.
9. A. Perdon and G. Gambolati, 'Extreme eigenvalues of large sparse matrices by Rayleigh quotient and modified conjugate gradients', *Comp. Methods Appl. Mech. Eng.*, **56**, 251–264 (1986).
10. G. Gambolati, 'Solution of large scale eigenvalue problems', in M. Papadrakakis, (ed.), *Solving Large Scale Problems in Mechanics: The Development and Application of Computational Solution Methods*, Wiley, New York, 1993, pp. 125–156.
11. G. Gambolati, F. Sartoretto and P. Florian, 'An orthogonal accelerated deflation technique for large symmetric eigenproblems', *Comp. Methods Appl. Mech. Eng.*, **94**, 13–23 (1992).
12. L. Bergamaschi, G. Gambolati and G. Pini, 'Spectral analysis of large finite element problems by optimization methods', *J. Shock Vibration*, **1**(6), 529–540 (1994).
13. A. Ruhe, 'Computation of eigenvalues and eigenvectors', in V. A. Barker, (ed.), *Sparse Matrix Techniques, Lecture Notes in Mathematics*, Vol. 572, Springer, Berlin, 1977, pp. 130–184.
14. M. A. Wolfe, *Numerical Methods for Unconstrained Optimization, an Introduction*, Von Nostrand Reinhold Company, Berkshire, England, 1978.

15. E. Polak, *Computational Methods in Optimization: A Unified Approach*, Academic Press, New York, 1971.
16. D. E. Longsine and S. F. McCormick 'Simultaneous Rayleigh-quotient minimization methods for $ax = \lambda bx$', *Linear Algebra Applic.*, **34**, 195–234 (1980).
17. F. Sartoretto, G. Pini and G. Gambolati, 'Accelerated simultaneous iterations for large finite element eigenproblems', *J. Comp. Phys.*, **81**, 53–69 (1989).
18. G. Gambolati, G. Pini and F. Sartoretto, 'An improved iterative optimization technique for the leftmost eigenpairs of large symmetric matrices', *J. Comp. Phys.*, **74**, 41–60 (1988).
19. J. Meijerink and H. van der Vorst, 'An iterative solution method for linear systems of which the coefficient matrix is a symmetric M-matrix', *Math. Comp.*, **31**, 148–162 (1977).
20. P. Roberti, 'The accelerated power method', *Int. j. numer. methods eng.*, **20**, 1179–1191 (1984).
21. J. H. Wilkinson, 'Inverse iteration in theory and in practice', in *Symposia Mathematica X*, Academic Press, London, 1972, pp. 361–379.