NUMERICAL LINEAR ALGEBRA WITH APPLICATIONS
Numer. Linear Algebra Appl. 2002; **9**:21–44 (DOI: 10.1002/nla.246)

Combination of Jacobi–Davidson and conjugate gradients for the partial symmetric eigenproblem

Y. Notay[*,†]

Service de Métrologie Nucléaire, Université Libre de Bruxelles (C.P. 165/84), 50, Av. F.D. Roosevelt, B-1050 Brussels, Belgium

SUMMARY

To compute the smallest eigenvalues and associated eigenvectors of a real symmetric matrix, we consider the Jacobi–Davidson method with inner preconditioned conjugate gradient iterations for the arising linear systems. We show that the coefficient matrix of these systems is indeed positive definite with the smallest eigenvalue bounded away from zero. We also establish a relation between the residual norm reduction in these inner linear systems and the convergence of the outer process towards the desired eigenpair. From a theoretical point of view, this allows to prove the optimality of the method, in the sense that solving the eigenproblem implies only a moderate overhead compared with solving a linear system. From a practical point of view, this allows to set up a stopping strategy for the inner iterations that minimizes this overhead by exiting precisely at the moment where further progress would be useless with respect to the convergence of the outer process. These results are numerically illustrated on some model example. Direct comparison with some other eigensolvers is also provided. Copyright © 2001 John Wiley & Sons, Ltd.

KEY WORDS: eigenvalue; Jacobi–Davidson; conjugate gradients; preconditioning

1. INTRODUCTION

This paper is concerned with the computation of a few of the smallest eigenvalues and associated eigenvectors of a (large sparse) real symmetric matrix A.

As preconditioning techniques became increasingly popular and efficient in the context of linear systems solution, many recent works focus on their use within the context of eigenvalue computation (see Reference [1] for a recent survey). The Davidson method and its generalizations [2–6] are quite popular in this respect, but have to face the difficulty that, beyond some point, increasing the quality of the preconditioner destroys the convergence

[*]Correspondence to: Y. Notay, Université Libre de Bruxelles, Service de Métrologie Nucléaire, C.P. 165/84, 50 Av. F.D. Roosevelt, B-1050 Bruxelles, Belgium
[†]E-mail: ynotay@ulb.ac.be

Contract/grant sponsor: Fonds National de la Recherche Scientifique, Maître de recherches.

Received 10 July 2000
Revised 21 January 2001

[2, 7]. Besides, it is also possible to simply solve iteratively the systems arising in classical shift-and-invert methods, but this is somewhat unattractive because these systems tend to be ill-conditioned and need to be solved with increasing accuracy as the convergence towards the desired eigenvector proceeds [8–11].

The recently proposed Jacobi–Davidson (JD) method [7, 12–15] appears here to be an efficient way to overcome these difficulties. It originates form the combination of the above-mentioned Davidson method with an older method by Jacobi, and may also be seen as an inexact Newton process [16–18]. Given an approximate eigenvector **u** of norm unity and associated residual

$$\mathbf{r} = A\mathbf{u} - \theta\mathbf{u} \tag{1}$$

where

$$\theta = (\mathbf{u}, A\mathbf{u}) \tag{2}$$

is the Rayleigh quotient, this method basically computes a correction **t** to the current approximation **u** by solving (approximately) the so-called *correction equation*. In the context considered here, the latter is written

$$(I - \mathbf{u}\mathbf{u}^{\mathrm{T}})(A - \eta I)(I - \mathbf{u}\mathbf{u}^{\mathrm{T}})\mathbf{t} = -\mathbf{r}, \quad \mathbf{t} \perp \mathbf{u} \tag{3}$$

where η either equals some fixed 'target' τ (e.g. $\tau = 0$ if A is positive definite) or the Rayleigh quotient θ, see Section 2 below.

As shown in Reference [7] (see also Section 2), if Equation (3) is solved exactly, the method converges as fast as inverse or Rayleigh quotient iterations, according to the choice of η. However, a moderate accuracy is generally sufficient to ensure convergence, making the method attractive when iterative solvers have to be preferred because the matrix A is large and sparse. Moreover, thanks to the projection onto the space orthogonal to **u**, systems (3) are expected to remain reasonably well conditioned even though $A - \theta I$ is closer and closer to a singular matrix as θ converges to an eigenvalue.

However, these views are till now only supported by heuristic arguments and numerical results. In particular, no analysis is available that connects the accuracy of the inner systems solution with the convergence of **u** towards the desired eigenvector. Consequently, no proof exists that if there exists an optimal preconditioner (in some sense), then the overall scheme is also optimal (in the same sense).

On the other hand, concerning more specifically symmetric matrices, the method looses part of its attractivity because the indefiniteness of $A - \theta I$ seems prevent the use of the conjugate gradient method and may further slow down the convergence speed of MINRES.

In this paper, we aim at filling these gaps in the context of the computation of the smallest eigenvalue of a symmetric matrix, with straightforward extension to the case where one computes a sequence of eigenpairs according to the deflation strategy generally used with the JD method.

We first prove that the projected matrix

$$(I - \mathbf{u}\mathbf{u}^{\mathrm{T}})(A - \theta I)(I - \mathbf{u}\mathbf{u}^{\mathrm{T}})$$

is actually positive definite on the subspace orthogonal to **u** as soon as the eigenvalue closest to θ is effectively the smallest one. Moreover, its conditioning does not deteriorate as θ converges to the smallest eigenvalue.

On the other hand, we obtain an expression for the next residual (1) of the eigenvalue equation that is easily computable during the course of inner conjugate gradient iterations. Consequently, we propose a criterion that stops these iterations nearly exactly when one has reached with respect to the outer process essentially the same progress as with an exact solution of the correction equation.

Combining both these results, we reach then the conclusion that the JD method is effectively globally optimal since finding an eigenpair implies only a moderate overhead compared with solving a positive definite linear system by the preconditioned conjugate gradient method.

Before closing this introduction, let us mention some related works. The close connection between Newton methods for eigenproblems and Rayleigh quotient iterations was first investigated in Reference [19]. The convergence rate of inexact Newton methods is analysed in Reference [20], whereas an interesting relation with inexact Rayleigh quotient iterations is established in Reference [21]. Further insight on the convergence of preconditioned inverse iterations can be gained at the light of References [22, 23], where methods with only one inner iteration are analysed. Finally, many recent works focus on methods that minimize the Rayleigh quotient by means of gradient methods, among which the most effective seem to be the *non-linear* conjugate gradient methods, see References [24, 25] and the references therein. That approach also uses an algorithm based on conjugate directions, but has otherwise little in common with the one considered in this paper. Below, we numerically compare our method with one representative of this family, namely the LOBPCG method from Reference [25].

The remainder of this paper is organized as follows. The main features of the JD method are reviewed in Section 2. Our conditioning analysis of the correction equation is developed in Section 3. In Section 4, we establish the connection between the convergence of the outer process and the accuracy of the inner conjugate gradient solutions. Algorithmic details are discussed in Section 5, and Section 6 is devoted to numerical results.

1.1. Notation

Throughout this paper, A is a symmetric $n \times n$ matrix with eigenvalues $\lambda_1 \leqslant \lambda_2 \leqslant \cdots \leqslant \lambda_n$ and associated eigenvectors $\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_n$.

2. THE JACOBI–DAVIDSON METHOD

In this section, we review the features of the method that are the most relevant to our purpose, referring the reader to the literature [7, 12–15] for additional details and further developments.

We first recall the following property.

Property 2.1 Let $\hat{\mathbf{t}}$ be the exact solution to (3). One has

$$\mathbf{u} + \hat{\mathbf{t}} = \gamma^{-1}(A - \eta I)^{-1}\mathbf{u}$$

where $\gamma = (\mathbf{u}, (A - \eta I)^{-1}\mathbf{u})$.

Thus, as stated above, the JD method may be seen as a variant of the inverse iteration method in which the use of iterative solvers is made easier.

Concerning η, when searching for the smallest eigenvalue of a symmetric matrix, one starts iterations with $\eta = \tau$, where τ is some 'target' less than λ_1, for instance $\tau = 0$ if A is positive definite. This favours convergence towards $\mathbf{u}_1$. When a sufficiently close approximation has been found so that θ is closer to λ_1 than to any other eigenvalue, it is then advantageous to use $\eta = \theta$. This indeed allows to recover the cubic convergence of Rayleigh quotient iterations [26], at least when the correction equation is solved sufficiently accurately. Of course, in practice, the choice has to be based on some heuristics, but we defer their discussion to Section 5.

Now, we have to recall here that the JD method combines generally the principles exposed so far with a Ritz–Galerkin approach, that is the (approximate) solution $\mathbf{t}$ to the correction equation is used to expand a subspace from which one extracts the next approximate eigenvector by solving the eigenproblem projected onto that subspace [7, 12–15]. However, we shall here focus the analysis on the 'simplified' JD procedure given in Algorithm 2.1, in which the next approximate eigenvector $\mathbf{u}^{(m+1)}$ is directly obtained by adding the computed correction to the current approximation. Since the complete JD algorithm is expected to be faster in general (see Section 6 for some numerical results), the convergence analysis developed below for the simplified method may also be seen as a worst-case analysis for the complete procedure.

Algorithm 2.1 (Simplified Jacobi–Davidson).

```
• Initialization
        $\mathbf{u}^{(0)}$ arbitrary with norm unity
• Iteration ($m = 0, 1, \ldots$)
    $\theta^{(m)} = \left(\mathbf{u}^{(m)}, A\mathbf{u}^{(m)}\right)$
    $\mathbf{r}^{(m)} = A\mathbf{u}^{(m)} - \theta^{(m)}\mathbf{u}^{(m)}$
    If $\|\mathbf{r}^{(m)}\| \leqslant \varepsilon$ exit
    Select $\eta = \tau$ or $\eta = \theta^{(m)}$
    Solve (approximately)
        $(I - \mathbf{u}^{(m)}\mathbf{u}^{(m)\mathrm{T}})(A - \eta I)(I - \mathbf{u}^{(m)}\mathbf{u}^{(m)\mathrm{T}})\,\mathbf{t} = -\,\mathbf{r}(m)$
        for $\mathbf{t} \perp \mathbf{u}^{(m)}$
    $\mathbf{u}^{(m+1)} = \dfrac{\mathbf{u}^{(m)} + \mathbf{t}}{\|\mathbf{u}^{(m)} + \mathbf{t}\|}$
$\varepsilon$: required accuracy
$\tau$: real number less than $\lambda_1$
```

2.1. Preconditioning and projection

To solve the correction equation iteratively, it is essential to have a good preconditioner. Since preconditioning of the projected matrix is far from obvious, one prefers generally to set up a preconditioner K for $A - \eta I$, and to define the preconditioner for the projected matrix

$$A_{\eta,\mathbf{u}} = (I - \mathbf{u}\mathbf{u}^{\mathrm{T}})(A - \eta I)(I - \mathbf{u}\mathbf{u}^{\mathrm{T}}) \tag{4}$$

by applying the same projectors to K, yielding

$$M_{\mathbf{u}} = \left(I - \mathbf{u}\mathbf{u}^{\mathrm{T}}\right) K \left(I - \mathbf{u}\mathbf{u}^{\mathrm{T}}\right) \tag{5}$$

Note in this respect that the solution of

$$M_{\mathbf{u}}\mathbf{w} = \mathbf{g}$$

with the constraint $\mathbf{w} \perp \mathbf{u}$ is actually given by

$$\mathbf{w} = K^{-1}\mathbf{g} - \frac{(\mathbf{g}, K^{-1}\mathbf{u})}{(\mathbf{u}, K^{-1}\mathbf{u})} K^{-1}\mathbf{u}$$

(see Reference [15]). Thus, since $K^{-1}\mathbf{u}$ is computed once for all at the beginning of the iterations, only one application of K^{-1} is needed per iteration. Moreover, further saving is also possible in the projection steps, see Section 5 for details.

Concerning K, it has been observed in References [12, 14, 15] that the preconditioner initially defined for $A - \tau I$ remains efficient in the phase where one selects $\eta = \theta$. Thus, K does not need to be updated as the convergence of the outer process proceeds. We will follow this approach here, and we will justify it in the next section. At this stage, we note that it has the further advantage that $A - \tau I$ is positive definite, which in general facilitates the definition of efficient preconditioners. In particular, if A is positive definite and $\tau = 0$, one just needs a preconditioner for A. On the other hand, any reasonable preconditioner K will also be positive definite, favouring the use of the conjugate gradient method.

2.2. *Computing several eigenpairs*

Once some approximations $(\tilde{\lambda}_1, \tilde{\mathbf{u}}_1), \ldots, (\tilde{\lambda}_{j-1}, \tilde{\mathbf{u}}_{j-1})$ to the first $j-1$ eigenpairs have been computed with sufficient accuracy, the JD method may compute the next eigenpair using a simple deflation strategy [12]. Namely, the next search subspace is enforced to be orthogonal to $\tilde{\mathbf{u}}_1, \ldots, \tilde{\mathbf{u}}_{j-1}$, whereas one works with the matrix restricted to that subspace, i.e.

$$A_{Q_{j-1}} = (I - Q_{j-1}Q_{j-1}^{\mathrm{T}}) A (I - Q_{j-1}Q_{j-1}^{\mathrm{T}})$$

where

$$Q_{j-1} = (\tilde{\mathbf{u}}_1 \;\ldots\; \tilde{\mathbf{u}}_{j-1})$$

is the $n \times (j-1)$ matrix whose columns are the so far computed eigenvectors, supposed to be mutually orthogonal (to working precision).

In fact, this approach is closely related to the one used with the QR method, see Reference [12]. In the context of our paper, it has the pleasant feature that, since one just works within a subspace, all our results on Algorithm 2.1 in the following sections actually apply when computing a sequence of eigenpairs in this way, $(\lambda_1, \mathbf{u}_1), (\lambda_2, \mathbf{u}_2), \ldots$ playing the role of the non-trivial eigenpairs of $A_{Q_{j-1}}$ ('non-trivial' referring to those for which the eigenvector is orthogonal to $\tilde{\mathbf{u}}_1, \ldots, \tilde{\mathbf{u}}_{j-1}$).

Concerning the preconditioning, the proper generalization of the above approach is obtained by using

$$M_{\mathbf{u}, Q_{j-1}} = \left(I - \mathbf{u}\mathbf{u}^{\mathrm{T}}\right) \left(I - Q_{j-1}Q_{j-1}^{\mathrm{T}}\right) K \left(I - Q_{j-1}Q_{j-1}^{\mathrm{T}}\right) \left(I - \mathbf{u}\mathbf{u}^{\mathrm{T}}\right)$$

Since $\mathbf{u}$ has to be orthogonal to $\tilde{\mathbf{u}}_1, \ldots, \tilde{\mathbf{u}}_{j-1}$, this may be equivalently rewritten

$$M_{\mathbf{u},Q_{j-1}} = \left(I - \tilde{Q}_j\tilde{Q}_j^{\mathrm{T}}\right) K \left(I - \tilde{Q}_j\tilde{Q}_j^{\mathrm{T}}\right)$$

where

$$\tilde{Q}_j = (\tilde{\mathbf{u}}_1 \ldots \tilde{\mathbf{u}}_{j-1}\mathbf{u})$$

is obtained by expanding Q_{j-1} with $\mathbf{u}$.

This raises not much additional difficulty since a system

$$M_{\mathbf{u},Q_{j-1}}\mathbf{w} = \mathbf{g}$$

with the constraint $\tilde{Q}_j^{\mathrm{T}}\mathbf{w} = 0$ is solved with

$$\mathbf{w} = K^{-1}\mathbf{g} - K^{-1}\tilde{Q}_j \left(\tilde{Q}_j^{\mathrm{T}}K^{-1}\tilde{Q}_j\right)^{-1} \left(K^{-1}\tilde{Q}_j\right)^{\mathrm{T}} \mathbf{g}$$

which again requires the computation of $K^{-1}\tilde{Q}_j$ as preprocessing and next only one application of K^{-1} per iteration [12, 14, 15].

3. CONDITIONING OF THE CORRECTION EQUATION

We first prove the following lemma.

Lemma 3.1. *Let A be a real symmetric $n \times n$ matrix with eigenvalues $\lambda_1 \leqslant \lambda_2 \leqslant \cdots \leqslant \lambda_n$. For any vector $\mathbf{u}$ with norm unity,*

$$\min_{\substack{\mathbf{z}\perp\mathbf{u} \\ \|\mathbf{z}\|=1}} (\mathbf{z}, A\mathbf{z}) \geqslant \lambda_1 + \lambda_2 - \theta \tag{6}$$

where $\theta = (\mathbf{u}, A\mathbf{u})$.

Proof

Let $\mu = \min_{\substack{\mathbf{z}\perp\mathbf{u} \\ \|\mathbf{z}\|=1}} (\mathbf{z}, A\mathbf{z})$. Obviously, μ is the smallest eigenvalue of the matrix

$$A_{0,\mathbf{u}} = (I - \mathbf{u}\mathbf{u}^{\mathrm{T}})A(I - \mathbf{u}\mathbf{u}^{\mathrm{T}})$$

for which the corresponding eigenvector $\mathbf{y}$ is orthogonal to $\mathbf{u}$. Let, for $i = 1, \ldots, n$, $\xi_i = (\mathbf{u}_i, \mathbf{u})$ and $\zeta_i = (\mathbf{u}_i, \mathbf{y})$, so that $\mathbf{u} = \sum_{i=1}^n \xi_i\mathbf{u}_i$ and $\mathbf{y} = \sum_{i=1}^n \zeta_i\mathbf{u}_i$. The eigenvalue equation

$$A_{0,\mathbf{u}}\mathbf{y} = \mu\mathbf{y}$$

gives then

$$\sum_{i=1}^n \zeta_i\lambda_i(I - \mathbf{u}\mathbf{u}^{\mathrm{T}})\mathbf{u}_i = \mu\sum_{i=1}^n \zeta_i\mathbf{u}_i$$

whence

$$\sum_{i=1}^n \zeta_i\lambda_i\mathbf{u}_i - \left(\sum_{i=1}^n \zeta_i\,\lambda_i\xi_i\right)\mathbf{u} = \mu\sum_{i=1}^n \zeta_i\mathbf{u}_i$$

Multiplying both sides to the left by $\mathbf{u}_j^{\mathrm{T}}$ yields

$$\lambda_j \zeta_j - \left(\sum_{i=1}^{n} \zeta_i \lambda_i \xi_i \right) \xi_j = \mu \zeta_j$$

i.e.

$$\zeta_j = \left(\sum_{i=1}^{n} \zeta_i \lambda_i \xi_i \right) \frac{\xi_j}{\lambda_j - \mu}$$

On the other hand, the condition $\mathbf{y} \perp \mathbf{u}$ may be rewritten $\sum_{i=1}^{n} \xi_i \zeta_i = 0$ and plugging the above result in the latter equation gives

$$\sum_{i=1}^{n} \frac{\xi_i^2}{\lambda_i - \mu} = 0$$

Now, if $\mu \geqslant \lambda_2$ the lemma holds trivially since $\theta \geqslant \lambda_1$. Otherwise, if $\mu < \lambda_2$, one obtains, using the above equality,

$$\begin{aligned} 1 &= \sum_{i=1}^{n} \frac{\xi_i^2 \lambda_i}{\lambda_i - \mu} - \mu \sum_{i=1}^{n} \frac{\xi_i^2}{\lambda_i - \mu} \\ &= \sum_{i=1}^{n} \frac{\xi_i^2 \lambda_i}{\lambda_i - \mu} - \lambda_1 \sum_{i=1}^{n} \frac{\xi_i^2}{\lambda_i - \mu} \\ &\leqslant \frac{1}{\lambda_2 - \mu} \sum_{i=2}^{n} \xi_i^2 (\lambda_i - \lambda_1) \\ &= \frac{1}{\lambda_2 - \mu} \left(\sum_{i=1}^{n} \xi_i^2 \lambda_i - \lambda_1 \right), \end{aligned}$$

which gives the required result since $\theta = \sum_{i=1}^{n} \xi_i^2 \lambda_i$. ■

Now, consider solving the correction equation (3) by the conjugate gradient method with preconditioner (5), K being, as discussed in Section 2, a preconditioner for $A - \tau I$.

At first sight, both the system matrix and the preconditioner are singular, but this does not matter because the conjugate gradient algorithm is ran within the subspace of vectors orthogonal to $\mathbf{u}$ (one checks that $\mathbf{r} \perp \mathbf{u}$). Thus, notions of positive definiteness, eigenvalue distribution, condition number, etc., apply as usual but with respect to the matrices restricted to this subspace.

Therefore, we first need to check that both the system matrix and the preconditioner are positive definite on this subspace, that is that both $(\mathbf{z}, K\mathbf{z})$ and $(\mathbf{z}, (A - \eta I)\mathbf{z})$ are positive for all non-zero vectors $\mathbf{z}$ orthogonal to $\mathbf{u}$.

As argued in Section 2, K is positive definite, whereas $A - \eta I$ is obviously positive definite when $\eta = \tau$. On the other hand, with Lemma 3.1,

$$\min_{\substack{\mathbf{z} \perp \mathbf{u} \\ \|\mathbf{z}\| = 1}} (\mathbf{z}, (A - \theta I)\mathbf{z}) \geqslant \lambda_1 + \lambda_2 - 2\theta$$

which will be positive as soon as θ is closer to λ_1 than λ_2. Since there is no reason to use $\eta=\theta$ when this condition is not met, there is therefore no difficulty from a theoretical point of view except when $\lambda_1=\lambda_2$ that is when λ_1 is a multiple eigenvalue. In practice, it is however not possible to always avoid miss-selection of η since λ_1 and λ_2 are unknown. How to face properly the potential difficulties in such cases and avoid a breakdown of the method is discussed in the next section.

Here, we pursue the analysis of most frequent cases, assuming that $\eta=\theta$ only when

$$\delta=\frac{\theta-\lambda_1}{\lambda_2-\lambda_1}<\frac{1}{2} \tag{7}$$

(δ is in fact very small in the final convergence phase of the outer process). Since the positive definiteness is then guaranteed we turn now our attention to the condition number. Letting

$$q(\mathbf{z})=\frac{(\mathbf{z},(A-\eta I)\mathbf{z})}{(\mathbf{z},K\mathbf{z})}$$

the latter is given by

$$\kappa=\frac{\max_{\substack{\mathbf{z}\perp\mathbf{u}\\ \mathbf{z}\neq 0}} q(\mathbf{z})}{\min_{\substack{\mathbf{z}\perp\mathbf{u}\\ \mathbf{z}\neq 0}} q(\mathbf{z})}$$

If $\eta=\tau$, one finds

$$\kappa\leqslant\frac{v_{\max}(K^{-1}(A-\tau I))}{v_{\min}(K^{-1}(A-\tau I))}=\kappa(K^{-1}(A-\tau I))$$

i.e. the condition number is not larger than that of $K^{-1}(A-\tau I)$, and fast convergence is ensured if K is a good preconditioner for $A-\tau I$.

If $\eta=\theta$, using

$$q(\mathbf{z})=\frac{(\mathbf{z},\,(A-\tau I)\,\mathbf{z})}{(\mathbf{z},\,K\,\mathbf{z})}\cdot\frac{(\mathbf{z},\,(A-\theta I)\,\mathbf{z})}{(\mathbf{z},\,(A-\tau I)\,\mathbf{z})}$$

one obtains, since $\theta\geqslant\lambda_1>\tau$,

$$\max_{\substack{\mathbf{z}\perp\mathbf{u}\\ \mathbf{z}\neq 0}} q(\mathbf{z})<v_{\max}(K^{-1}(A-\tau I))$$

whereas

$$\min_{\substack{\mathbf{z}\perp\mathbf{u}\\ \mathbf{z}\neq 0}} q(\mathbf{z})\geqslant v_{\min}(K^{-1}(A-\tau I))\cdot\left(1+\frac{\theta-\tau}{\min_{\substack{\mathbf{z}\perp\mathbf{u}\\ \|\mathbf{z}\|=1}}(\mathbf{z},\,A\,\mathbf{z})-\theta}\right)^{-1}$$

Therefore, using Lemma 3.1 and assumption (7),

$$\kappa<\kappa(K^{-1}(A-\tau I))\left(1+\frac{\theta-\tau}{(\lambda_2-\lambda_1)(1-2\,\delta)}\right)$$

showing that the condition number does not deteriorate as $\theta\rightarrow\lambda_1$ and is at worst only slightly larger than that of $K^{-1}(A-\tau I)$.

4. CONVERGENCE WITH INEXACT LINEAR SYSTEM SOLUTIONS

As noted in Section 2, the JD method converges fast (cubically in the asymptotic phase) when the correction equation is solved exactly. Here we analyse the influence of inexact solutions obtained with k preconditioned conjugate gradient iterations. To make notation clear, we first present in synthetic form the considered algorithm, which is based on the standard implementation of the conjugate gradient method [27, 28]. Note that specific details, allowing some saving in the projection steps, are considered in the next section.

Algorithm 4.1 (Conjugate gradients for the correction equation: generic algorithm).

```
• Initialization
      $\mathbf{t}_0^{(m)} = \mathbf{d}_{-1} = 0$
      $\mathbf{g}_0^{(m)} = -\mathbf{r}^{(m)}$
      $\rho_{-1} = 1$
• Iteration $(k = 0, 1, \ldots)$
      Solve  $M_{\mathbf{u}^{(m)}}\,\mathbf{w}_k = \mathbf{g}_k^{(m)};\ \mathbf{w}_k \perp \mathbf{u}^{(m)}$
      $\rho_k = (\mathbf{g}_k^{(m)}, \mathbf{w}_k)$
      $\mathbf{d}_k = \mathbf{w}_k + \frac{\rho_k}{\rho_{k-1}}\,\mathbf{d}_{k-1}$
      $\mathbf{v}_k = A_{\eta,\mathbf{u}^{(m)}}\,\mathbf{d}_k$
      $\alpha_k = (\mathbf{d}_k, \mathbf{v}_k)$
      $\mathbf{t}_{k+1}^{(m)} = \mathbf{t}_k^{(m)} + \frac{\rho_k}{\alpha_k}\,\mathbf{d}_k$
      $\mathbf{g}_{k+1}^{(m)} = \mathbf{g}_k^{(m)} - \frac{\rho_k}{\alpha_k}\,\mathbf{v}_k$
```

A: symmetric $n \times n$ matrix;

$\mathbf{u}^{(m)}$: vector of norm unity;

$\theta^{(m)} = (\mathbf{u}^{(m)}, A\,\mathbf{u}^{(m)})$;

$\mathbf{r}^{(m)} = A\,\mathbf{u}^{(m)} - \theta^{(m)}\,\mathbf{u}^{(m)}$;

$A_{\eta,\mathbf{u}^{(m)}} = (I - \mathbf{u}^{(m)}\,\mathbf{u}^{(m)^{\mathrm{T}}})(A - \eta I)(I - \mathbf{u}^{(m)}\mathbf{u}^{(m)^{\mathrm{T}}})$;

$M_{\mathbf{u}^{(m)}} = (I - \mathbf{u}^{(m)}\,\mathbf{u}^{(m)^{\mathrm{T}}})\,K\,(I - \mathbf{u}^{(m)}\,\mathbf{u}^{(m)^{\mathrm{T}}})$

with K symmetric and positive definite.

As noted in the previous section, $A_{\eta,\mathbf{u}^{(m)}}$ is mostly positive definite on the subspace orthogonal to $\mathbf{u}^{(m)}$, but may occasionally be indefinite. We therefore briefly discuss this case. In fact, nothing prevents us to use the above algorithm, but convergence is no more guaranteed and breakdown is possible with $\alpha_k = 0$ (or below some reasonable threshold). Therefore, since $\alpha_k > 0$ if the system matrix is positive definite, most conjugate gradient codes issue an error when $\alpha_k \leqslant 0$. When solving the correction equation, we also suggest to stop iterations as soon as this occurs. However, this does not mean that the JD method breaks down. Indeed, the preconditioned correction equation will be at worst only slightly indefinite. We therefore expect that $\alpha_k \leqslant 0$ may occur only after some significant convergence has been achieved, so that progress is still obtained in the outer process. Moreover, would this be not the case, one may just discard the computed $\mathbf{t}_k^{(m)}$ and restart the iterations with $\eta = \tau$ to ensure positive definiteness. In this respect, using results below, it is also possible to set up some safeguard that restarts Algorithm 4.1 if no progress has been made after a few iterations.

However, we stress that these aspects should not be understood as very essential for our results. Indeed, in our numerical experiments, we never needed to exit inner iterations because of an $\alpha_k \leqslant 0$, even though we considered situations with multiple eigenvalues. The above considerations are mainly to show that, with a proper implementation, the potential indefiniteness of $A_{\eta,\mathbf{u}^{(m)}}$ cannot spoil the method.

We now develop our analysis, stressing that our main result (12) does actually apply independently of the definiteness of $A_{\eta,\mathbf{u}^{(m)}}$ (we only assume that no breakdown has occurred). We first define

$$\mathbf{u}_k^{(m+1)} = \frac{\mathbf{u}^{(m)} + \mathbf{t}_k^{(m)}}{\|\mathbf{u}^{(m)} + \mathbf{t}_k^{(m)}\|} \tag{8}$$

$$\mathbf{r}_k^{(m+1)} = A\,\mathbf{u}_k^{(m+1)} - (\mathbf{u}_k^{(m+1)}, A\,\mathbf{u}_k^{(m+1)})\mathbf{u}_k^{(m+1)} \tag{9}$$

Thus, $\mathbf{r}_k^{(m+1)}$ is the residual at the next step in Algorithm 2.1 if inner iterations are stopped at step k. We also define the corresponding quantities when one computes the exact solution $\hat{\mathbf{t}}^{(m)}$ to the correction equation

$$\hat{\mathbf{u}}^{(m+1)} = \frac{\mathbf{u}^{(m)} + \hat{\mathbf{t}}^{(m)}}{\|\mathbf{u}^{(m)} + \hat{\mathbf{t}}^{(m)}\|}, \tag{10}$$

$$\hat{\mathbf{r}}^{(m+1)} = A\,\hat{\mathbf{u}}^{(m+1)} - (\hat{\mathbf{u}}^{(m+1)}, A\,\hat{\mathbf{u}}^{(m+1)})\hat{\mathbf{u}}^{(m+1)} \tag{11}$$

The following theorem contains the main results of this section. Note that we use the fact that the initial approximation is set equal to zero in Algorithm 4.1. Other situations are of no practical interest, since any meaningful non-zero initial guess can be immediately added to $\mathbf{u}^{(m)}$.

Theorem 4.1. *Consider the application of Algorithm* 4.1, *and assume that* $\alpha_i \neq 0$ *for* $i = 0,\ldots,k-1$, *where k is some positive integer. Let* $\mathbf{r}_k^{(m+1)}$ *be defined by* (8) *and* (9). *There holds*

$$\|\mathbf{r}_k^{(m+1)}\|^2 = \frac{\|\mathbf{g}_k^{(m)}\|^2}{1 + \|\mathbf{t}_k^{(m)}\|^2} + \left(\frac{\|\mathbf{t}_k^{(m)}\|}{1 + \|\mathbf{t}_k^{(m)}\|^2} (\theta^{(m)} - \eta + \beta_k) \right)^2 \tag{12}$$

where

$$\beta_k = (\mathbf{u}^{(m)}, (A - \eta I)\,\mathbf{t}_k^{(m)}) = -\sum_{i=0}^{k-1} \frac{\rho_i^2}{\alpha_i} \tag{13}$$

Assume also that $A_{\eta,\mathbf{u}^{(m)}}$ *is positive definite on the subspace orthogonal to* $\mathbf{u}^{(m)}$, *let* $\hat{\mathbf{t}}^{(m)}$ *be the exact solution to* $A_{\eta,\mathbf{u}^{(m)}}\mathbf{t}^{(m)} = -\mathbf{r}^{(m)}$ *with* $\mathbf{t}^{(m)} \perp \mathbf{u}^{(m)}$, *and let* $\hat{\mathbf{r}}^{(m+1)}$ *be defined by* (10) *and* (11). *Then*

$$\left| \frac{\|\mathbf{t}_k^{(m)}\|}{1+\|\mathbf{t}_k^{(m)}\|^2}(\theta^{(m)} - \eta - \beta_k) \right| \leqslant \|\hat{\mathbf{r}}^{(m+1)}\| \cdot \max\left(\frac{\|\mathbf{t}_k^{(m)}\|}{\|\hat{\mathbf{t}}^{(m)}\|}, \frac{\|\hat{\mathbf{t}}^{(m)}\|}{\|\mathbf{t}_k^{(m)}\|} \right) \cdot (1 + f_k) \tag{14}$$

with

$$f_k \leqslant \begin{cases} 0 & \text{if } \eta = \theta^{(m)} \\ \dfrac{\|\hat{\mathbf{t}}^{(m)} - \mathbf{t}_k^{(m)}\|^2_{A_{\eta,\mathbf{u}^{(m)}}}}{\|\hat{\mathbf{t}}^{(m)}\|^2_{A_{\eta,\mathbf{u}^{(m)}}}} \cdot \dfrac{\theta^{(m)} - \lambda_1}{\lambda_1 - \eta} & \text{if } \eta < \lambda_1 \end{cases} \tag{15}$$

where $\|\mathbf{z}\|^2_{A_{\eta,\mathbf{u}^{(m)}}} = (\mathbf{z}, A_{\eta,\mathbf{u}^{(m)}}\mathbf{z})$. *Moreover,*

$$\frac{\|\mathbf{r}_k^{(m+1)}\|^2}{\|\mathbf{r}^{(m)}\|^2} \leqslant \frac{\|\mathbf{g}_k^{(m)}\|^2}{\|\mathbf{g}_0^{(m)}\|^2} + \frac{\|\hat{\mathbf{r}}^{(m+1)}\|^2}{\|\mathbf{r}^{(m)}\|^2} \cdot \max\left(\frac{\|\mathbf{t}_k^{(m)}\|}{\|\hat{\mathbf{t}}^{(m)}\|}, \frac{\|\hat{\mathbf{t}}^{(m)}\|}{\|\mathbf{t}_k^{(m)}\|} \right) \cdot (1 + f_k) \tag{16}$$

Proof
We first show that

$$(\mathbf{d}_k, A_{\eta,\mathbf{u}^{(m)}}\mathbf{d}_i) = 0 \quad \text{for } i = 0, \ldots, k-1 \tag{17}$$

$$(\mathbf{g}_k^{(m)}, \mathbf{d}_i) = 0 \quad \text{for } i = 0, \ldots, k-1 \tag{18}$$

$$(\mathbf{g}_k^{(m)}, \mathbf{t}_k^{(m)}) = 0 \tag{19}$$

$$(\mathbf{g}_0^{(m)}, \mathbf{t}_k^{(m)}) = \sum_{i=0}^{k-1} \frac{\rho_i^2}{\alpha_i} \tag{20}$$

$$(\mathbf{g}_k^{(m)}, \hat{\mathbf{t}}^{(m)} - \mathbf{t}_k^{(m)}) = (\mathbf{g}_0^{(m)}, \hat{\mathbf{t}}^{(m)} - \mathbf{t}_k^{(m)}) \tag{21}$$

Indeed, (17) and (18) are the usual conjugacy properties that only depends on the symmetry of the system matrix. Equation (19) then straightforwardly follows from $\mathbf{t}_k^{(m)} = \sum_{i=0}^{k-1}(\rho_i/\alpha_i)\mathbf{d}_i$. The latter relation also entails (20) since, for all $i \leqslant k$, by virtue of (17),

$$\rho_i = (\mathbf{g}_i^{(m)}, \mathbf{w}_i) = (\mathbf{g}_i^{(m)}, \mathbf{d}_i) = (\mathbf{g}_0^{(m)}, \mathbf{d}_i) - \sum_{j=0}^{i-1} \frac{\rho_j}{\alpha_j}(\mathbf{d}_j, A_{\eta,\mathbf{u}^{(m)}}\mathbf{d}_i) = (\mathbf{g}_0^{(m)}, \mathbf{d}_i)$$

Finally, (21) follows from $\mathbf{g}_0^{(m)} - \mathbf{g}_k^{(m)} = \sum_{i=0}^{k-1} \frac{\rho_i}{\alpha_i} A_{\eta,\mathbf{u}^{(m)}}\mathbf{d}_i$ together with

$$(A_{\eta,\mathbf{u}^{(m)}}\mathbf{d}_i, \hat{\mathbf{t}}^{(m)} - \mathbf{t}_k^{(m)}) = (\mathbf{d}_i, A_{\eta,\mathbf{u}^{(m)}}(\hat{\mathbf{t}}^{(m)} - \mathbf{t}_k^{(m)})) = (\mathbf{d}_i, \mathbf{g}_k^{(m)}) = 0$$

We now show (13). Since

$$(I - \mathbf{u}^{(m)}\mathbf{u}^{(m)\mathrm{T}})(A - \eta I)\mathbf{u}^{(m)} = \mathbf{r}^{(m)} = -\mathbf{g}_0^{(m)} \tag{22}$$

and since $\mathbf{t}_k^{(m)} \perp \mathbf{u}^{(m)}$, one verifies with (20) that

$$\begin{aligned}(\mathbf{u}^{(m)}, (A-\eta I)\,\mathbf{t}_k^{(m)}) &= ((A-\eta I)\,\mathbf{u}^{(m)}, \mathbf{t}_k^{(m)}) \\ &= ((I-\mathbf{u}^{(m)}\mathbf{u}^{(m)^{\mathrm{T}}})(A-\eta I)\,\mathbf{u}^{(m)}, \mathbf{t}_k^{(m)}) \\ &= -(\mathbf{g}_0^{(m)}, \mathbf{t}_k^{(m)}) \\ &= -\sum_{i=0}^{k-1}\frac{\rho_i^2}{\alpha_i}\end{aligned}$$

Let then $\theta_k^{(m+1)} = (\mathbf{u}_k^{(m+1)}, A\,\mathbf{u}_k^{(m+1)})$. Since $\|\mathbf{u}^{(m)}\| = 1$ and $\mathbf{u}^{(m)}$ is orthogonal to $\mathbf{t}_k^{(m)}$, $\|\mathbf{u}^{(m)} + \mathbf{t}_k^{(m)}\|^2 = 1 + \|\mathbf{t}_k^{(m)}\|^2$, whence

$$\begin{aligned}\|\mathbf{r}_k^{(m+1)}\|^2 &= \|(A-\theta_k^{(m+1)}I)\,\mathbf{u}_k^{(m+1)}\|^2 \\ &= \|(A-\eta I)\,\mathbf{u}_k^{(m+1)}\|^2 - (\theta_k^{(m+1)}-\eta)^2 \\ &= ((1+\|\mathbf{t}_k^{(m)}\|^2)^{-1}\,\|(A-\eta\, I)(\mathbf{u}^{(m)}+\mathbf{t}_k^{(m)})\|^2 - (\theta_k^{(m+1)}-\eta)^2 \qquad (23)\end{aligned}$$

On the other hand, note that (22) implies

$$(I-\mathbf{u}^{(m)}\mathbf{u}^{(m)^{\mathrm{T}}})(A-\eta\, I)(\mathbf{u}^{(m)}+\mathbf{t}_k^{(m)}) = \mathbf{r}^{(m)} + A_{\eta,\mathbf{u}^{(m)}}\,\mathbf{t}_k^{(m)} = -\mathbf{g}_k^{(m)} \qquad (24)$$

Therefore,

$$\begin{aligned}\|(A-\eta\, I)(\mathbf{u}^{(m)}+\mathbf{t}_k^{(m)})\|^2 &= \|(I-\mathbf{u}^{(m)}\mathbf{u}^{(m)^{\mathrm{T}}})(A-\eta\, I)(\mathbf{u}^{(m)}+\mathbf{t}_k^{(m)})\|^2 \\ &\quad + \|\mathbf{u}^{(m)}\mathbf{u}^{(m)^{\mathrm{T}}}(A-\eta\, I)(\mathbf{u}^{(m)}+\mathbf{t}_k^{(m)})\|^2 \\ &= \|\mathbf{g}_k^{(m)}\|^2 + (\theta^{(m)}-\eta+\beta_k)^2 \qquad (25)\end{aligned}$$

Moreover, using (24) and (19),

$$\begin{aligned}((1+\|\mathbf{t}_k^{(m)}\|^2)(\theta_k^{(m+1)}-\eta) &= (\mathbf{u}^{(m)}+\mathbf{t}_k^{(m)}, (A-\eta\, I)(\mathbf{u}^{(m)}+\mathbf{t}_k^{(m)})) \\ &= (\theta^{(m)}-\eta+\beta_k) + (\mathbf{t}_k^{(m)}, (A-\eta\, I)(\mathbf{u}^{(m)}+\mathbf{t}_k^{(m)})) \\ &= (\theta^{(m)}-\eta+\beta_k) + (\mathbf{t}_k^{(m)}, \mathbf{g}_k^{(m)}) \\ &= (\theta^{(m)}-\eta+\beta_k)\end{aligned}$$

Equation (12) then follows from the latter result and (23), (25).

To prove (14), note first that by letting $\mathbf{g}_k^{(m)} \to 0$ and $\mathbf{t}_k^{(m)} \to \hat{\mathbf{t}}^{(m)}$ in (12), one has

$$\|\hat{\mathbf{r}}^{(m+1)}\| = \left| \frac{\|\hat{\mathbf{t}}^{(m)}\|}{1 + \|\hat{\mathbf{t}}^{(m)}\|^2} (\mathbf{u}^{(m)}, (A - \eta I)(\mathbf{u}^{(m)} + \hat{\mathbf{t}}^{(m)})) \right|$$

Since $x(1+y^2)/y(1+x^2) \leqslant \max(x/y, y/x)$ for any positive number x and y, (14) therefore holds with

$$f_k = -\frac{(\mathbf{u}^{(m)}, (A - \eta I)(\hat{\mathbf{t}}^{(m)} - \mathbf{t}_k^{(m)}))}{(\mathbf{u}^{(m)}, (A - \eta I)(\mathbf{u}^{(m)} + \hat{\mathbf{t}}^{(m)}))}$$

Furthermore, with (22) and (21),

$$\begin{aligned} -(\mathbf{u}^{(m)}, (A - \eta I)(\hat{\mathbf{t}}^{(m)} - \mathbf{t}_k^{(m)})) &= -((I - \mathbf{u}^{(m)} \mathbf{u}^{(m)^{\mathrm{T}}})(A - \eta I)\mathbf{u}^{(m)}, \hat{\mathbf{t}}^{(m)} - \mathbf{t}_k^{(m)}) \\ &= (\mathbf{g}_0^{(m)}, \hat{\mathbf{t}}^{(m)} - \mathbf{t}_k^{(m)}) \\ &= (\mathbf{g}_k^{(m)}, \hat{\mathbf{t}}^{(m)} - \mathbf{t}_k^{(m)}) \\ &= \|\hat{\mathbf{t}}^{(m)} - \mathbf{t}_k^{(m)}\|^2_{A_{\eta, \mathbf{u}^{(m)}}} \end{aligned}$$

whereas

$$\begin{aligned} (\mathbf{u}^{(m)}, (A - \eta I)(\mathbf{u}^{(m)} + \hat{\mathbf{t}}^{(m)})) &= \theta^{(m)} - \eta + ((I - \mathbf{u}^{(m)} \mathbf{u}^{(m)^{\mathrm{T}}})(A - \eta I)\mathbf{u}^{(m)}, \hat{\mathbf{t}}) \\ &= \theta^{(m)} - \eta + (\mathbf{r}^{(m)}, \hat{\mathbf{t}}) \\ &= \theta^{(m)} - \eta - \|\hat{\mathbf{t}}^{(m)}\|^2_{A_{\eta, \mathbf{u}^{(m)}}} \end{aligned}$$

Therefore,

$$f_k = \frac{\|\hat{\mathbf{t}}^{(m)} - \mathbf{t}_k^{(m)}\|^2_{A_{\eta, \mathbf{u}^{(m)}}}}{\|\hat{\mathbf{t}}^{(m)}\|^2_{A_{\eta, \mathbf{u}^{(m)}}}} \cdot \left(\frac{\theta^{(m)} - \eta}{(\mathbf{u}^{(m)}, (A - \eta I)(\mathbf{u}^{(m)} + \hat{\mathbf{t}}^{(m)}))} - 1 \right)$$

which proves (15) when $\eta = \theta^{(m)}$. On the other hand, to complete the proof in the case $\eta < \lambda_1$, one just needs Lemma 2.1 that yields

$$(\mathbf{u}^{(m)}, (A - \eta I)(\mathbf{u}^{(m)} + \hat{\mathbf{t}}^{(m)}))^{-1} = (\mathbf{u}^{(m)}, (A - \eta I)^{-1} \mathbf{u}^{(m)}) \leqslant (\lambda_1 - \eta)^{-1}.$$

Finally, (16) follows from (12), (14), (15) together with the fact that $\|\mathbf{g}_0^{(m)}\| = \|\mathbf{r}^{(m)}\|$. ■

Several important conclusions can be drawn from the theorem.

First of all, $\|\mathbf{r}_k^{(m+1)}\|$ can be computed as soon as one has the norms of $\mathbf{g}_k^{(m)}$ and $\mathbf{t}_k^{(m)}$. Hence the convergence of the outer process may be monitored while inner iterations are running. Moreover, $\|\mathbf{r}_k^{(m+1)}\|^2$ is basically the sum of two terms, one that vanishes along with the residual of the linear system, and one that converges quickly to $\|\hat{\mathbf{r}}^{(m+1)}\|^2$, that is the value that one would obtain with exact inverse or Rayleigh quotient iterations. Therefore, it is advisable to stop inner iterations as soon as the first term does no longer dominate the second one. To detect this moment, we propose to compare the ratios

$$\frac{\|\mathbf{r}_k^{(m+1)}\|}{\|\mathbf{r}_{k-1}^{(m+1)}\|} \quad \text{and} \quad \frac{\|\mathbf{g}_k^{(m)}\|}{\|\mathbf{g}_{k-1}^{(m)}\|}$$

Indeed, it is clear that they have to behave similarly until the point where $\|\mathbf{r}_k^{(m+1)}\|$ stagnates because it is close to its minimal value. This suggests to stop inner iterations when

$$\frac{\|\mathbf{r}_k^{(m+1)}\|}{\|\mathbf{r}_{k-1}^{(m+1)}\|} > \left(\frac{\|\mathbf{g}_k^{(m)}\|}{\|\mathbf{g}_{k-1}^{(m)}\|}\right)^a \tag{26}$$

for some $a<1$ (say $a=0.9$), provided that

$$\frac{\|\mathbf{g}_k^{(m)}\|}{\|\mathbf{g}_0^{(m)}\|} \leqslant 0.5 \tag{27}$$

the latter condition being a safeguard against a too early exit.

Besides, we note that condition (26) makes no sense if $\|\mathbf{g}_k^{(m)}\| \geqslant \|\mathbf{g}_{k-1}^{(m)}\|$. Although it is possible to just bypass the stopping test until $\|\mathbf{g}_k^{(m)}\|$ starts decreasing again, our experiments indicated that, on the whole, it is preferable to exit as soon as both the safeguard condition (27) and

$$\|\mathbf{r}_k^{(m+1)}\| \geqslant \|\mathbf{r}_{k-1}^{(m+1)}\| \tag{28}$$

hold, keeping then $\mathbf{t}_{k-1}^{(m)}$ as approximate solution to the correction equation. Besides the fact that quite large plateaus are possible while $\|\mathbf{r}_k^{(m+1)}\|$ is already relatively close to its minimal value, this explains also by the convergence of the conjugate gradient method when the eigenvalue distribution favors an irregular behaviour, convergence which may be actually faster in the initial phase than in the subsequent one, see Reference [29, Chapter 13] for an analysis.

On the other hand, from a theoretical point of view, (16) ensures that it is not necessary to increase the accuracy requirements for the linear systems solution as $\mathbf{u}^{(m)}$ converges towards the desired eigenvector. This is potentially useful since, on account of the asymptotically cubic convergence, $\|\hat{\mathbf{r}}^{(m+1)}\|/\|\mathbf{r}^{(m)}\|$ will rapidly decrease in the final phase, making more effort in reducing $\|\mathbf{g}_k^{(m)}\|/\|\mathbf{g}_0^{(m)}\|$ worthwhile.

Finally, a straightforward consequence of our stopping criterion (26) is that

$$\frac{\|\mathbf{r}_k^{(m+1)}\|}{\|\mathbf{r}^{(m)}\|} = \frac{\|\mathbf{r}^{(m+1)}\|}{\|\mathbf{r}^{(m)}\|} \approx \frac{\|\mathbf{g}_k^{(m)}\|}{\|\mathbf{g}_0^{(m)}\|}$$

whence

$$\frac{\|\mathbf{r}_k^{(m)}\|}{\|\mathbf{r}^{(0)}\|} \approx \prod_{i=0}^{m-1} \frac{\|\mathbf{g}_k^{(i)}\|}{\|\mathbf{g}_0^{(i)}\|}$$

that is, the relative accuracy reached for the eigenvalue problem is approximately equal to the product of the relative accuracies in the successive inner linear systems. Comparing then the effort needed to solve the eigenvalue problem with that needed to solve a linear system

$(A - \tau I)\mathbf{x} = \mathbf{b}$, and taking into account results of the preceding section, we conclude that the main overhead is the one associated with the periodic restart of the conjugate gradient process. This will be illustrated in Section 6.

5. DETAILED ALGORITHM

For GMRES like methods with left preconditioning, it is shown in Reference [12] that it is possible to skip the projection steps needed in the multiplication by $A_{\eta,\mathbf{u}^{(m)}}$. Here we first generalize this idea to the context of Algorithm 4.1.

In principle, since $\mathbf{d}_k \perp \mathbf{u}^{(m)}$, $\mathbf{v}_k = A_{\eta,\mathbf{u}^{(m)}}\,\mathbf{d}_k$ is implemented with

$$\tilde{\mathbf{v}}_k = (A - \eta I)\,\mathbf{d}_k$$
$$\mathbf{v}_k = \tilde{\mathbf{v}}_k - (\mathbf{u}^{(m)}, \tilde{\mathbf{v}}_k)\mathbf{u}^{(m)}$$

However, one wants to skip the explicit computation of $\mathbf{v}_k$ and work with $\tilde{\mathbf{v}}_k$ only.

No problem arises in the computation of α_k since, because $\mathbf{d}_k \perp \mathbf{u}^{(m)}$,

$$\alpha_k = (\mathbf{d}_k, \mathbf{v}_k) = (\mathbf{d}_k, \tilde{\mathbf{v}}_k)$$

Next, consider that $\mathbf{g}_k^{(m)}$ is not formed explicitly, and that one works with $\tilde{\mathbf{g}}_k^{(m)}$ computed according to

$$\tilde{\mathbf{g}}_{k+1}^{(m)} = \tilde{\mathbf{g}}_{k+1}^{(m)} - \frac{\rho_k}{\alpha_k}\,\tilde{\mathbf{v}}_k$$

Note that $\tilde{\mathbf{g}}_k^{(m)} = \mathbf{g}_0^{(m)} - (A - \eta I)\,\mathbf{t}_k^{(m)}$ holds whereas $\mathbf{g}_k^{(m)} = \tilde{\mathbf{g}}_k^{(m)} + \xi\,\mathbf{u}^{(m)}$ for some ξ. Therefore, since

$$\begin{aligned}\mathbf{w}_k &= K^{-1}\,(\tilde{\mathbf{g}}_k^{(m)} + \xi\,\mathbf{u}^{(m)}) - \tfrac{(\tilde{\mathbf{g}}_k^{(m)} + \xi\,\mathbf{u}^{(m)}, K^{-1}\,\mathbf{u}^{(m)})}{(\mathbf{u}^{(m)}, K^{-1}\,\mathbf{u}^{(m)})} K^{-1}\,\mathbf{u}^{(m)} \\ &= K^{-1}\,\tilde{\mathbf{g}}_k^{(m)} - \frac{(\tilde{\mathbf{g}}_k^{(m)}, K^{-1}\,\mathbf{u}^{(m)})}{(\mathbf{u}^{(m)}, K^{-1}\,\mathbf{u}^{(m)})} K^{-1}\,\mathbf{u}^{(m)},\end{aligned}$$

$\mathbf{w}_k$ may also be computed directly from $\tilde{\mathbf{g}}_k^{(m)}$. Furthermore, this also holds for ρ_k because $\mathbf{w}_k \perp \mathbf{u}^{(m)}$, whence

$$\rho_k = (\mathbf{g}_k^{(m)}, \mathbf{w}_k) = (\tilde{\mathbf{g}}_k^{(m)}, \mathbf{w}_k)$$

Finally, the last difficulty comes from the need to compute $\|\mathbf{g}_k^{(m)}\|$. Here we use (remembering that $\mathbf{g}_0^{(m)} = -\mathbf{r}^{(m)} \perp \mathbf{u}^{(m)}$)

$$\begin{aligned}\|\mathbf{g}_k^{(m)}\|^2 &= \|\tilde{\mathbf{g}}_k^{(m)}\|^2 - (\mathbf{u}^{(m)}, \tilde{\mathbf{g}}_k^{(m)})^2 \\ &= \|\tilde{\mathbf{g}}_k^{(m)}\|^2 - ((\mathbf{u}^{(m)}, \mathbf{g}_0^{(m)}) - (\mathbf{u}^{(m)}, (A - \eta I)\,\mathbf{t}_k{}^{(m)}))^2 \\ &= \|\tilde{\mathbf{g}}_k^{(m)}\|^2 - \beta_k^2\end{aligned}$$

with β_k given by (13). In practice we cannot exclude negative values when computing $\|\mathbf{g}_k^{(m)}\|^2$ in this way, due to roundoff. However, this is possible only when $\mathbf{t}_k^{(m)}$ is very close to the exact solution, hence the problem is cured by setting $\|\mathbf{g}_k^{(m)}\|$ to zero, entailing the exit of the inner iterations by criterion (26).

These considerations result in Algorithm 5.1, that implements the conjugate gradient method for the inner iterations with the stopping strategy discussed in Section 4.

Algorithm 5.1 (Conjugate gradients for the correction equation: detailed algorithm).

```
• Initialization
        $\mathbf{t}_0^{(m)} = \mathbf{d}_{-1} = 0$
        $\tilde{\mathbf{g}}_0^{(m)} = -\mathbf{r}^{(m)}$
        $\mathbf{y} = K^{-1}\mathbf{u}^{(m)};\ \zeta = (\mathbf{u}^{(m)}, \mathbf{y})$
        $\rho_{-1} = 1;\ \beta_0 = 0$
• Iteration $(k = 0, 1, \ldots)$
        $g_k = \sqrt{\max(\|\tilde{\mathbf{g}}_k^{(m)}\|^2 - \beta_k^2, 0)}$
        $r_k = \sqrt{\frac{g_k^2}{1+\|\mathbf{t}_k^{(m)}\|^2} + \left(\frac{\|\mathbf{t}_k^{(m)}\|}{1+\|\mathbf{t}_k^{(m)}\|^2}(\theta^{(m)} - \eta + \beta_k)\right)^2}$
        If $r_k \leqslant \varepsilon$ : exit with $\mathbf{t}_k^{(m)}$
        If $g_k/g_0 \leqslant 0.5$ then
            If $r_k \geqslant r_{k-1}$ : exit with $\mathbf{t}_{k-1}^{(m)}$
            If $(r_k/r_{k-1}) > (g_k/g_{k-1})^a$ : exit with $\mathbf{t}_k^{(m)}$
        $\mathbf{w}_k = K^{-1}\tilde{\mathbf{g}}_k^{(m)} - \frac{(\tilde{\mathbf{g}}_k^{(m)}, \mathbf{y})}{\zeta}\mathbf{y}$
        $\rho_k = (\tilde{\mathbf{g}}_k^{(m)}, \mathbf{w}_k)$
        $\mathbf{d}_k = \mathbf{w}_k + \frac{\rho_k}{\rho_{k-1}}\mathbf{d}_{k-1}$
        $\tilde{\mathbf{v}}_k = (A - \eta I)\mathbf{d}_k$
        $\alpha_k = (\mathbf{d}_k, \tilde{\mathbf{v}}_k)$
        If $\alpha_k \leqslant 0$ : exit with $\mathbf{t}_k^{(m)}$
        $\mathbf{t}_{k+1}^{(m)} = \mathbf{t}_k^{(m)} + \frac{\rho_k}{\alpha_k}\mathbf{d}_k$
        $\tilde{\mathbf{g}}_{k+1}^{(m)} = \tilde{\mathbf{g}}_k^{(m)} - \frac{\rho_k}{\alpha_k}\tilde{\mathbf{v}}_k$
        $\beta_{k+1} = \beta_k - \frac{\rho_k^2}{\alpha_k}$
```

A: symmetric $n \times n$ matrix; $\mathbf{u}^{(m)}$: vector of norm unity;
$\theta^{(m)} = (\mathbf{u}^{(m)}, A\,\mathbf{u}^{(m)})$; $\mathbf{r}^{(m)} = A\,\mathbf{u}^{(m)} - \theta^{(m)}\mathbf{u}^{(m)}$;
K: symmetric and positive definite $n \times n$ matrix;
ε: accuracy requirement for the residual of the eigenvalue equation;
a: parameter < 1 (typically $a = 0.9$).

5.1. Computing several eigenpairs

In the case where one applies the deflation strategy mentioned in Section 2, one just needs to replace the computation of $\mathbf{w}_k$ with

$$\mathbf{w}_k = K^{-1}\,\tilde{\mathbf{g}}_k^{(m)} - Y_j\,H_j^{-1}\,Y_j^{\mathrm{T}}\,\tilde{\mathbf{g}}_k^{(m)},$$

where

$$Y_j = K^{-1}\tilde{Q}_j \quad \text{and} \quad H_j = \tilde{Q}_j^{\mathrm{T}}\,Y_j$$

have been computed during the initialization phase.

The same reasonings as above show then that no problems are expected with the computation of α_k, $\mathbf{w}_k$ and ρ_k even though $\tilde{\mathbf{v}}_k$ is not projected explicitly onto the subspace orthogonal to $\tilde{\mathbf{u}}_1, \ldots, \tilde{\mathbf{u}}_{j-1}$. However, it is not possible here to subtract the components $(\tilde{\mathbf{u}}_i, \tilde{\mathbf{g}}_k^{(m)})^2$ from $\|\tilde{\mathbf{g}}_k^{(m)}\|^2$ in order to get the true value of $\|\mathbf{g}_k^{(m)}\|$. Nevertheless, since the $\tilde{\mathbf{u}}_i$ are approximate eigenvectors with residual error less than ε and since $\mathbf{d}_k$ is orthogonal to them, these components will be at most of order $\varepsilon^2\,\|\mathbf{g}_0^{(m)}\|^2$. They are therefore negligible except when $\|\mathbf{g}_k^{(m)}\|/\|\mathbf{g}_0^{(m)}\| \approx \varepsilon$, which is very unlikely since with our stopping strategy $\|\mathbf{g}_k^{(m)}\|$ will approach ε only at the final JD iteration, and at this stage $\|\mathbf{g}_0^{(m)}\| = \|\mathbf{r}^{(m)}\|$ is usually already much smaller than 1.

5.2. Selection of η

Ideally, one should select $\eta = \theta^{(m)}$ only if $\theta^{(m)} - \lambda_1 < \lambda_2 - \theta^{(m)}$. In practice, neither λ_1 nor λ_2 is known. However, see e.g. Reference [30, p. 81],

$$\theta^{(m)} - \lambda_1 \leqslant \frac{\|\mathbf{r}^{(m)}\|^2}{\lambda_2 - \theta^{(m)}}$$

hence the switch condition for η is satisfied when $\|\mathbf{r}^{(m)}\| \leqslant \lambda_2 - \theta^{(m)}$.

Furthermore, in the complete JD procedure (that projects the eigenproblem onto a subspace of increasing dimension), $\lambda_2 - \theta^{(m)}$ may be estimated with $\theta_2^{(m)} - \theta^{(m)}$, where $\theta_2^{(m)}$ is the second eigenvalue of the projected problem. This leads to the selection procedure:

$$\eta = \theta^{(m)} \;\Leftrightarrow\; \|\mathbf{r}^{(m)}\| \leqslant \theta_2^{(m)} - \theta^{(m)} \quad \text{and} \quad \left| \frac{\theta_2^{(m)} - \theta^{(m)}}{\theta_2^{(m-1)} - \theta^{(m-1)}} - 1 \right| \leqslant b \tag{29}$$

where b is a given threshold, e.g. $b = 0.1$.

When using the simplified JD procedure of Algorithm 2.1, one has to use a more heuristic approach. In our numerical experiments with the discrete Laplacian, we considered

$$\eta = \theta^{(m)} \;\Leftrightarrow\; \|\mathbf{r}^{(m)}\| \leqslant \theta^{(m)} \quad \text{and} \quad \left| \frac{\theta^{(m)}}{\theta^{(m-1)}} - 1 \right| \leqslant b \tag{30}$$

Indeed, in that specific context, one has often $\lambda_2 = c\,\lambda_1$ with c not too far from 2, hence $\lambda_2 - \theta^{(m)}$ may be roughly estimated by $(c-1)\theta^{(m)} \approx \theta^{(m)}$.

6. NUMERICAL RESULTS

We first illustrate our results with the computation of the smallest eigenvalues and associated eigenvectors of the matrix A resulting form the 5-point finite difference approximation of the Laplacian on the unit square with Dirichlet boundary conditions on the entire boundary. Using a uniform mesh with mesh size h in both directions, the eigenvalues of A are

$$\frac{2}{h^2}\,(2-\cos \pi h k_1-\cos \pi h k_2)\,,\quad k_1=1,\ldots,h^{-1},\ k_2=1,\ldots,h^{-1}$$

(assuming h^{-1} a positive integer). By way of illustration, for $h=1/180$, the smallest eigenvalues are

$$\lambda_1=19.7$$
$$\lambda_2=\lambda_3=49.34$$
$$\lambda_3=78.95$$
$$\lambda_5=\lambda_6=98.68$$
$$\lambda_7=\lambda_8=128.3$$

and they do not differ much for smaller mesh sizes.

For our experiments, we use random initial approximation and considered both the simplified JD procedure of Algorithm 2.1 and the complete algorithm based on subspace expansion, see References [12, 14] for implementation details.† However, we restricted Algorithm 2.1 to the computation of a single eigenpair. Indeed, it is otherwise always worthwhile to use the complete procedure since the already built subspace allows then to find a good initial approximation when starting the search for a new eigenpair.

For K, we considered three different preconditioners computed from A: the standard incomplete Cholesky factorization without fill-in (ILU) [27, 29, 31–33], the modified version for which K and A have same row-sum (MILU) [27, 29, 31, 34–36], and the algebraic multilevel preconditioner from Reference [37] (AML), which leads to a condition number independent of h although it is only twice as expensive per iteration than the incomplete factorization preconditioners.

The results are reported in Figures 1 and 2, where we have plotted the norm of the current residual of the eigenvalue equation $\|\mathbf{r}_k^{(m+1)}\|$ against the number of matrix vector multiplication (MVM). Note that 1 MVM is required for each outer JD iteration, that is for each restart of the inner conjugate gradient process. The corresponding residual norm, marked by a $\diamond$, is then the true residual norm in the outer JD algorithm, whereas at other points in the curve the residual norm is the one computed according to (4.7) as indicated in Algorithm 5.1. The figures on the left correspond to a single eigenpair computation whereas the figures on the right correspond to the computation of the 8 smallest eigenpairs, each peak being of course caused by the restart of the JD procedure with the matrix increasingly deflated. On

† In the latter case, we decided to let the size of the basis vary between 7 and 14, and we always enforced the compression of the basis to no more than 7 vectors when starting the search for a new eigenpair.

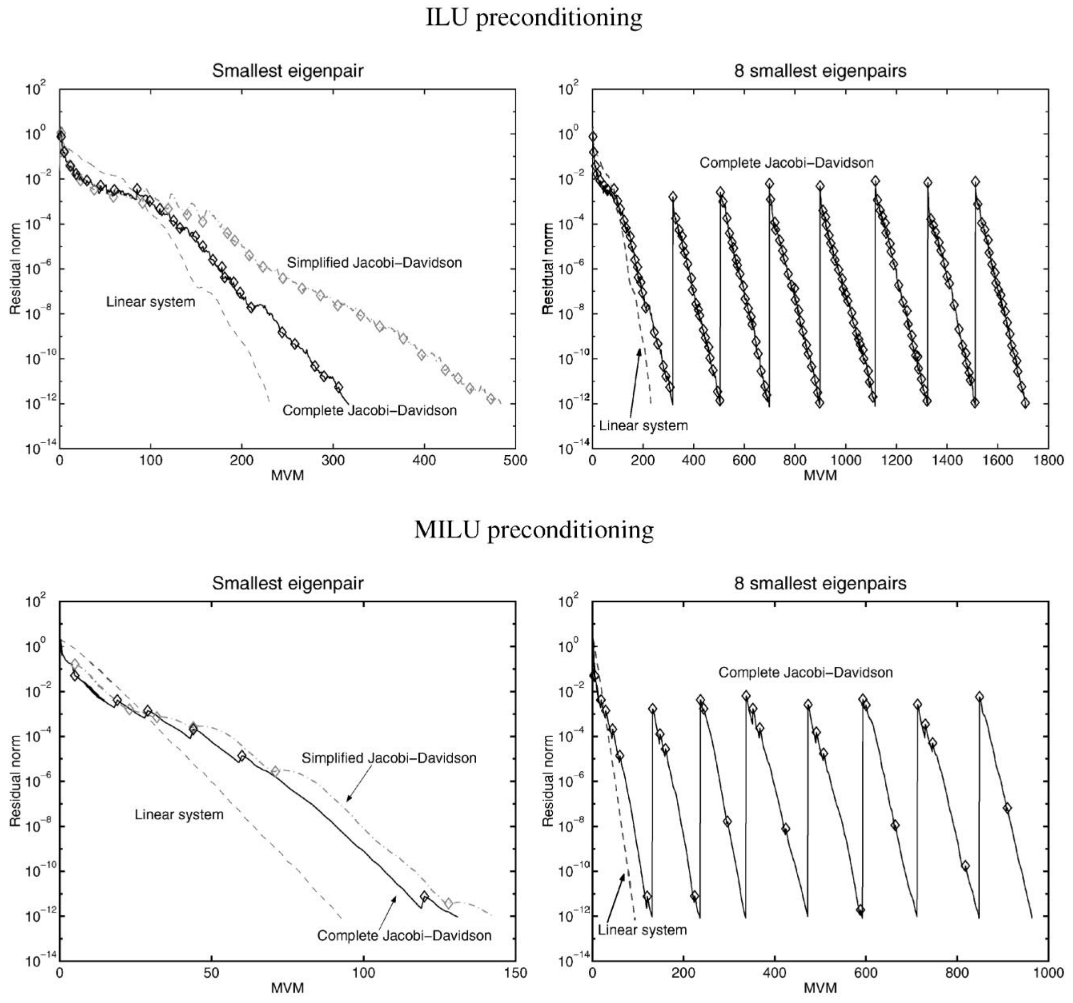

Figure 1. Residual norm against MVM for ILU and MILU preconditioning ($h^{-1} = 180$).

the figures, we also report for comparison purpose the evolution of the residual norm when solving a simple linear system $A\mathbf{x} = \mathbf{r}^{(0)}$ by the standard conjugate gradient method with the same preconditioner and the zero vector as initial approximation.

We make the following observations.

- With MILU, the convergence of the conjugate gradients is relatively regular. There is then not much difference between the simplified and the complete JD procedure. The computation of the first eigenpair requires only about 35 per cent more MVMs than the simple linear system solution, whereas the convergence towards next eigenvectors is slightly faster, probably thanks to the use of the previously built search subspace.

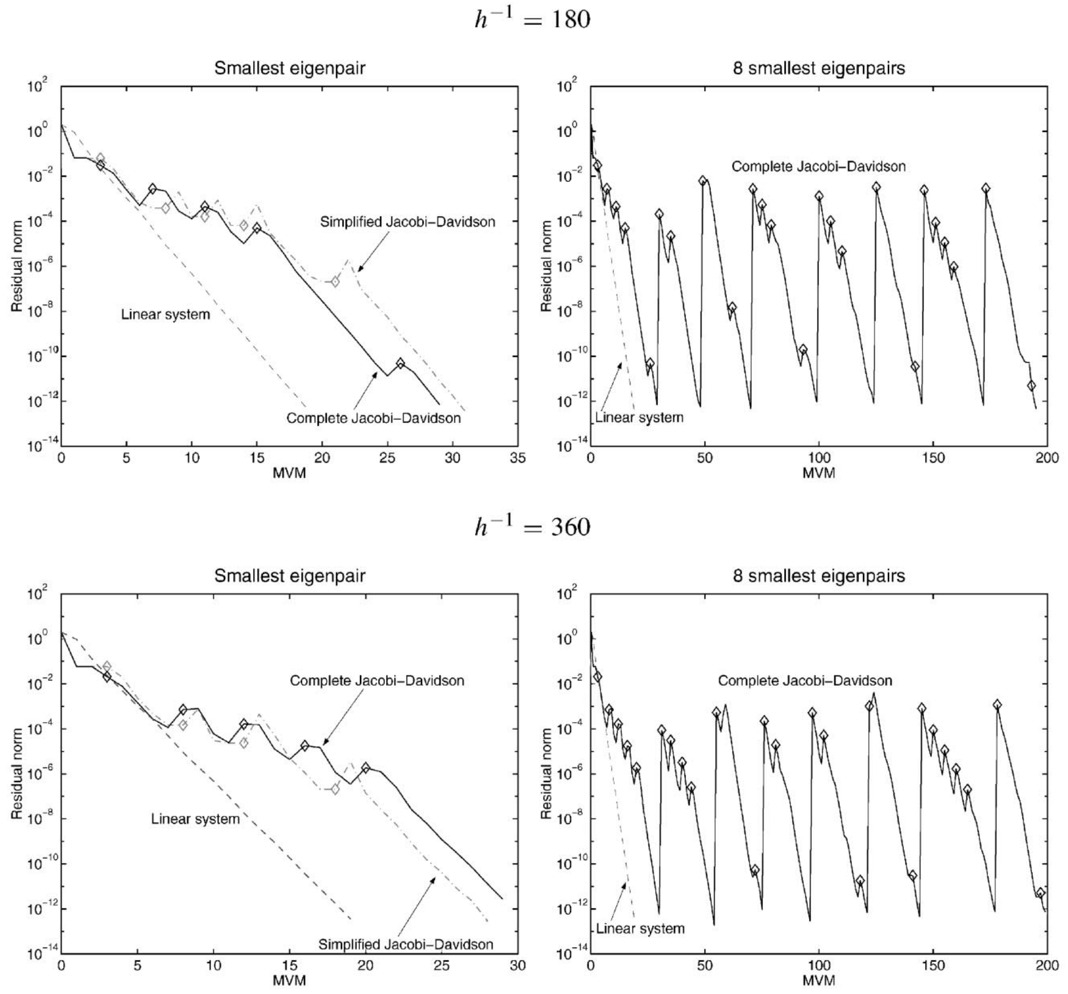

Figure 2. Residual norm against MVM for AML preconditioning.

- The same remarks apply also to AML, except that the *relative* overhead is somewhat larger. In fact, it is clear from Figure 2 that the overhead is in this case directly proportional to the number of outer JD iterations, and becomes more important in a relative sense essentially because the global number of inner iterations is here not much larger than the number of outer iterations (as a side effect of the quality of the preconditioner).
- The convergence is less regular with ILU. Therefore, because of criterion (2.8), one exits earlier from the inner iterations, entailing more outer iterations and a larger overhead compared with the solution of a simple linear system. However, the complete JD procedure compensates here partly this loss of optimality, limiting the overhead to about 35 per cent as with MILU.

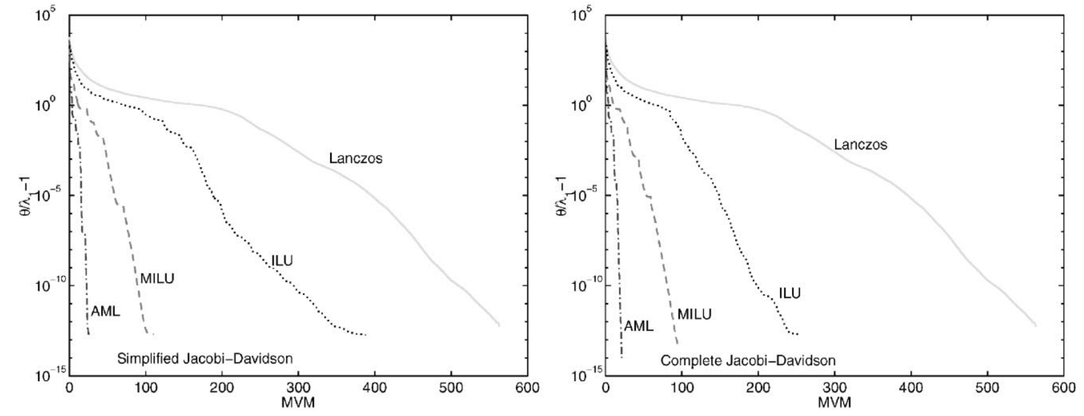

Figure 3. Relative error for the first eigenvalue against MVM ($h^{-1}=180$).

- No specific difficulty arises from the multiplicity of λ_2, λ_5 and λ_7. In particular, we never needed to exit because α_k was negative. Furthermore, we did not need to adapt the method for the subsequent computation of $\mathbf{u}_3$, $\mathbf{u}_6$ and $\mathbf{u}_8$ even though $A_{\eta,\mathbf{u}^{(m)}}$ is then at the first stage very close to a singular matrix since we always restart the JD algorithm with τ equal to the previously computed eigenvalue.
- Considering Figure 2, it suggests that an optimal order preconditioner for A makes the solution of the eigenproblem with the proposed algorithms also of optimal order.
- We also plotted in Figure 3 the evolution of the relative error for the first eigenvalue, comparing the three different preconditioned JD schemes with simple Lanczos iterations (e.g. Ref. [28]). Clearly, the gains are impressive, especially if one has a very good preconditioner.

6.1. Comparison with other eigensolvers

Few results compare Jacobi–Davidson with preconditioned eigensolvers that use gradient methods to minimize the Rayleigh quotient. In Reference [25], the LOBPCG method seems also compare favorably with linear systems solution.[§] Since a MATLAB package is available for LOBPCG and since we develop our own JDCG package, we then decided to perform a direct comparison of both methods. We further included in the test the standard Jacobi–Davidson package JDQR by Gerard Sleijpen, our code being actually a modified version of JDQR based on the results in this paper. The test where made within MATLAB 5.3 running on a SGI Origin 2000 server, using for JDCG the software referenced below, and for LOBPCG & JDQR the codes downloaded from, respectively, *http://www-math.cudenver.edu/˜aknyazev/software/CG/* and *http://www.math.uu.nl/people/sleijpen/JD_software/JDQR.html* (versions currently available by beginning of January 2001).

[§]The tests in Reference [25] are however not directly comparable to ours, since there the reference linear system is $(A-\lambda_1 I)\mathbf{u}=\mathbf{b}$ (which is solved in order to find $\mathbf{u}_1$).

Table I. Comparison of the MATLAB packages JDCG, LOBPCG and JDQR; ε is the required accuracy and '% flops for lin. alg.' refers to the percentage of flops spent outside matrix vector multiplications and preconditioning operations.

	1 eigenpair			10 eigenpairs		
	JDCG	LOBPCG	JDQR	JDCG	LOBPCG	JDQR
	ILU(10^{-3}) preconditioning, $\varepsilon = 10^{-5}$					
# MVM	21	15	27	165	350	144
# Prec. solve	21	13	28	164	200	174
# flops $\times 10^{-8}$	0.49	0.51	0.97	6.08	70.8	13.8
% flops for lin. alg.	29	55	52	53	95	79
	MILU(0) preconditioning, $\varepsilon = 10^{-5}$					
# MVM	48	34	54	500	970	592
# Prec. solve	500	600	699	1055	1470	1388
# flops $\times 10^{-8}$	0.57	0.86	1.275	9.56	201.0	31.5
% flops for lin. alg.	52	78	75	70	98	88
	ILU(10^{-3}) preconditioning, $\varepsilon = 10^{-10}$					
# MVM	45	46	50	375	903	288
# Prec. solve	45	33	55	375	380	357
# flops $\times 10^{-8}$	1.07	1.47	1.83	12.5	188.0	23.6
% flops for lin. alg.	28	59	49	48	96	74
	MILU(0) preconditioning, $\varepsilon = 10^{-10}$					
# MVM	98	125	120	1055	2700	1181
# Prec. solve	98	80	136	1055	1470	1388
# flops $\times 10^{-8}$	1.15	2.98	2.77	18.9	565.0	55.9
% flops for lin. alg.	52	81	73	68	98	87

As test example, we selected the matrix resulting form the 5-point finite difference approximation of the Laplacian on an L-shaped domain embedded in the unit square. We specified Dirichlet boundary conditions on the entire boundary and used a uniform mesh with mesh size $h = 1/180$ in both directions. Two preconditioners were considered: MILU without fill-in (MILU(0)), and ILU with fill entries discarded when smaller than 10^{-3} in a relative sense (ILU (10^{-3}), see MATLAB function *cholinc* for details). In each case, we ran two tests: one for which a single eigenpair was required, and one for which the ten smallest where sought. For each test, we checked that all desired eigenpairs were found within the prescribed accuracy, that is $\|AX - X\Lambda\|_2 < \varepsilon$, where X is the matrix whose columns are the computed eigenvector(s) (with norm unity), and Λ is the diagonal matrix with the computed eigenvalue(s) on its diagonal.

The detailed results for each solver are given in Table I. Clearly, JDCG outperforms JDQR, which demonstrates the usefulness of our results. On the other hand, LOBPCG is roughly equivalent to JDQR when one eigenpair is required with high accuracy, slightly better when one eigenpair is required with moderate accuracy, and much worse when many eigenpairs are sought.¶ JDCG is thus the winner in all cases.

6.2. Software

The JDCG package, written in MATLAB, is available from author's home page: *http://homepages.ulb.ac.be/˜ynotay*.

ACKNOWLEDGEMENTS

I thank Prof. Henk van der Vorst for useful comments on an earlier version of this work. English correction and further suggestions by a careful referee are also deeply acknowledged. I thank Gerard Sleijpen for the kind permission to develop the JDCG package as a modified version of JDQR.

REFERENCES

1. Knyazev AV. Preconditioned eigensolvers—an oxymoron? *ETNA* 1998; **7**:104–123.
2. Crouzeix M, Philippe B, Sadkane M. The Davidson method. *SIAM Journal on Scientific Computing* 1994; **15**:62–76.
3. Davidson ER. The iterative calculation of a few of the lowest eigenvalues and corresponding eigenvectors of large real-symmetric matrices. *Journal of Computational Physics* 1975; **17**:817–825.
4. Morgan RB, Scott DS. Generalizations of Davidson's method for computing eigenvalues of sparse symmetric matrices. *SIAM Journal on Scientific and Statistical Computing* 1986; **7**:817–825.
5. Morgan RB. Scott DS. Preconditioning the Lanczos algorithm for sparse symmetric eigenvalue problems. *SIAM Journal on Scientific Computing* 1993; **14**:585–593.
6. Stathopoulos A, Saad Y, Fischer CF. Robust preconditioning of large, sparse, symmetric eigenvalue problems. *Journal of Computational and Applied Mathematics* 1995; **64**:197–215.
7. Sleijpen G., van der Vorst H. A Jacobi–Davidson iteration method for linear eigenvalue problems. *SIAM Journal on Matrix Analysis and Applications* 1996; **17**:401–425.
8. Golub GH, Q Ye. Inexact inverse iterations for the generalized eigenvalue problems. *BIT* 2000; **40**:672–684.
9. Yu-Ling Lai, Kun-Yi Lin, Wen-Wei Lin. An inexact inverse iteration for large sparse eigenvalue problems. *Numerical Linear Algebra with Applications* 1997; **4**:425–437.
10. Lehoucq RB, Meerbergen K. Using generalized Cayley transformations within an inexact rational Krylov sequence method. *SIAM Journal on Matrix Analysis and Applications* 1998; **20**:131–148.
11. Smit P, Paardekooper MHC. The effects of inexact solvers in algorithms for symmetric eigenvalue problems. *Linear Algebra and its Applications* 1999; **287**:337–357.
12. Fokkema D, Sleijpen D, van der Vorst H. Jacobi–Davidson style QR and QZ algorithms for the reduction of matrix pencils. *SIAM Journal on Scientific Computing* 1999; **20**:94–125.
13. Sleijpen G, Booten A, Fokkema D, van der Vorst H. Jacobi–Davidson type methods for generalized eigenproblems and polynomial eigenproblems. *BIT* 1996; **36**:595–633.
14. Sleijpen G, van der Vorst H. In *Templates for the Solution of Algebraic Eigenvalue Problems: A Practical Guide*, chapter 4.7, Bai Z, Demmel J, Dongarra J, Ruhe A, van der Vorst H. (eds.) SIAM: Philadelphia, 2000.
15. Sleijpen G, van der Vorst H, Meijerink E. Efficient expansion of subspaces in the Jacobi–Davidson method for standard and generalized eigenproblems. *ETNA* 1998; **7**:75–89.
16. Edelman A, Arias TA, Smith ST. The geometry of algorithms with orthogonality constraints. *SIAM Journal on Matrix Analysis and Applications* 1999; **20**:303–353.
17. Sleijpen G, van der Vorst H. The Jacobi–Davidson method for eigenvalue problems and its relation with accelerated inexact newton schemes. In *Iterative Methods in Linear Algebra II*, Margenov S, Vassilevski P. (eds), *IMACS: Series in Computational and Applied Mathematics*, vol. 3, IMACS: Rutgers University, New Jersey, USA, 1996; 377–389.
18. Wu K, Saad Y, Stathopoulos A. Inexact newton preconditioning techniques for large symmetric eigenvalue problems. *ETNA* 1998; **7**:202–214.

¶According some reviewer comments, one of the problems faced by LOBPCG in such cases is that to reach our global stopping criterion the algorithm has to compute some eigenpairs with excessive accuracy.

19. Peters G, Wilkinson JH. Inverse iteration, ill-conditioned equations and Newton's method. *SIAM Review* 1979; **21**:339–360.
20. Zhang T, Golub GH, Law KH. Subspace iterative methods for eigenvalue problems. *Linear Algebra and its Applications* 1999; **294**:239–258.
21. Eldén L, Simoncini V. Inexact Rayleigh quotient-type methods for subspace tracking. *Technical Report* 1172, IAN–CNR, Pavia, Italy, 1999.
22. Neymeyr K. A geometric theory for preconditioned inverse iteration, I: extrema of the Rayleigh quotient. *Linear Algebra and its Applications* 2001; **322**:61–85.
23. Neymeyr K. A geometric theory for preconditioned inverse iteration, II: convergence estimates. *Linear Algebra and its Applications* 2001; **322**:87–104.
24. Bergamaschi L, Gambolati G, Pini G. Asymptotic convergence of conjugate gradient methods for the partial symmetric eigenproblem. *Numerical Linear Algebra with Applications* 1997; **4**:69–84.
25. Knyazev AV. Toward the optimal preconditioned eigensolver: Locally optimal block preconditioned conjugate gradient method. *SIAM Journal on Scientific Computing* 2000, to appear.
26. Parlett BN. *The Symmetric Eigenvalue Problem*. Prentice-Hall: Englewoood Cliffs, NJ, 1980.
27. Barrett R, Berry M, Chan TF, Demmel J, Donato J, Dongarra J, Eijkhout V, Pozo R, Romine C, van der Vorst H. *Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods.* SIAM: Philadelphia, 1993.
28. Golub GH, van Loan CF. *Matrix Computations* (3rd edn). The John Hopkins University Press: Baltimore, MA, 1996.
29. Axelsson O. *Iterative Solution Methods*. Cambridge University Press: Cambridge, 1994.
30. Saad Y. *Numerical Methods for Large Eigenvalue Problems*. Halstead Press: New York, 1992.
31. Chan TF, van der Vorst H. Approximate and incomplete factorizations. In *Parallel Numerical Algorithms*, Keyes DE, Samed A, Venkatakrishnan K (eds). *ICASE/LaRC Interdisciplinary Series in Science and Engineering*, vol. 4. Kluwer Academic: Dordecht, 1997; 167–202.
32. Meijerink JA, van der Vorst H. An iterative solution method for linear systems of which the coefficient matrix is a symmetric M-matrix. *Mathematics of Computation* 1977; **31**:148–162.
33. Saad Y. *Iterative Methods for Sparse Linear Systems*. PWS publishing: New York, 1996.
34. Beauwens R. Upper eigenvalue bounds for pencils of matrices. *Linear Algebra and its Applications* 1984; **62**:87–104.
35. Dupont T, Kendall RP, Rachford HH. An approximate factorization procedure for solving self-adjoint elliptic difference equations. *SIAM Journal on Numerical Analysis* 1968; **5**:559–573.
36. Gustafsson I. A class of first order factorization methods. *BIT* 1978; **18**:142–156.
37. Notay Y. Optimal order preconditioning of finite difference matrices. *SIAM Journal on Scientific Computing* 2000; **21**:1991–2007.