Universiteit Utrecht

Department
of Mathematics

Effective preconditioning techniques for eigenvalue problems

by

Gerard L.G. Sleijpen and Fred W. Wubs

Preprint
nr. 1117
August, 1999

EFFECTIVE PRECONDITIONING TECHNIQUES FOR EIGENVALUE PROBLEMS

GERARD L.G. SLEIJPEN* AND FRED W. WUBS‡

Abstract. In the Davidson method, any preconditioner can be exploited for the iterative computation of eigenpairs. However, the convergence of the eigenproblem solver may be poor if the quality of the preconditioner for linear systems solvers is good. Theoretically, this counter-intuitive phenomenon with the Davidson method is remedied by the Jacobi-Davidson approach, where the preconditioned system is restricted to appropriate subspaces of co-dimension one. However, it is not clear how the restricted system can be solved accurately and efficiently in case of a good preconditioner. The obvious approach introduces instabilities that hampers convergence.

In this paper, we show how an incomplete decomposition based on the MRILU approach can be used in a stable way. We also show how this preconditioner can be efficiently improved when better approximations for the eigenvalue of interest become available. Our approach leads to a good initial guess for the wanted eigenpair and to high quality preconditioners for nearby eigenvalues. The additional costs for updating the preconditioner are negligible.

Keywords: Eigenvalues and eigenvectors, Davidson method, Jacobi-Davidson, multilevel ILU-preconditioners.

AMS(MOS) subject classification: 65F15, 65N25.

1. Introduction. The Jacobi-Davidson method [20, 18] is an iterative method for computing one selected eigenvalue with associated eigenvector of standard as well as of generalized eigenvalue problems. In [8], an extension of the Jacobi-Davidson method is given that computes a set of eigenpairs. The method is flexible and efficient.

For the generalized eigenvalue problem,

$$\mathbf{A}\mathbf{x} = \lambda \mathbf{B}\mathbf{x}, \tag{1.1}$$

the method selects its approximate eigenvector $\mathbf{u}$ from a *search subspace* by testing $\mathbf{u}$ with associated approximate eigenvalue ϑ against a *test subspace* (a Petrov-Galerkin approach). The subspaces have the same dimension that is relatively low as compared to the dimension n of problem (1.1). The approximate eigensolution $(\vartheta, \mathbf{u})$ is used to compute effective expansion vectors for search and test subspace. The larger subspaces lead to better approximate eigenvectors, and so on.

The computation of the expansion vectors requires the (approximate) solution of a *correction equation*: with $\mathbf{u}$ normalized, the search subspace is expanded with the (approximate) solution $\mathbf{z}$ of

$$\mathbf{z} \perp \mathbf{u}, \quad (\mathbf{I} - \mathbf{q}\,\mathbf{q}^*)(\mathbf{A} - \vartheta\mathbf{B})(\mathbf{I} - \mathbf{u}\,\mathbf{u}^*)\mathbf{z} = -\mathbf{r}, \tag{1.2}$$

where $\mathbf{r} \equiv \mathbf{A}\mathbf{u} - \vartheta\mathbf{B}\mathbf{u}$ is the residual and $\mathbf{q} \equiv \mathbf{B}\mathbf{u}/\|\mathbf{B}\mathbf{u}\|$. The residual is orthogonal to the test subspace. The test subspace is expanded with $\mathbf{B}\mathbf{z}$ (or another convex combination of $\mathbf{A}\mathbf{z}$ and $\mathbf{B}\mathbf{z}$ [18, 8]; see also §2.3): the test subspace is the image of the search subspace under $\mathbf{B}$. Note that $\mathbf{r} \perp \mathbf{q}$.

If the correction equation (1.2) is solved exactly then the method can be viewed as an accelerated Newton method [19, 18, 24], and with proper selections of the approximate eigensolutions $(\vartheta, \mathbf{u})$, the method converges quadratically [20]. For realistic, high dimensional problems (n large), it is usually not feasible to compute the solution of (1.2) exactly. However, solutions that are accurate enough also lead to fast convergence [20, 8]. The preconditioning

*Mathematical Institute, Utrecht University, P.O. Box 80010, 3508 TA Utrecht, the Netherlands, e-mail: sleijpen@math.uu.nl

‡Dept. of Mathematics, University of Groningen, P.O. Box 800, 9700 AV Groningen, The Netherlands e-mail: F.W.Wubs@math.rug.nl

techniques that we will discuss in this paper aim to accelerate iterative solvers for the linear system (1.2). We will explain how preconditioners of multilevel ILU-type [15, 13, 4, 2] can be used efficiently: often one preconditioned step solves (1.2) sufficiently accurately.

The projections in (1.2) have a double effect: not only do they lead to effective expansion vectors, but they also improve the conditioning of the linear system (think of the realistic situation where $\mathbf{u} \approx \mathbf{x}$, $\vartheta \approx \lambda$ and 0 is a simple eigenvalue of $\mathbf{A}-\lambda\mathbf{B}$). The conditioning is of importance for numerical stability of the solution and for the speed of convergence of iterative linear solvers. Unfortunately, the projections also complicate some of the computations.

Although (1.2) is a linear system that can be treated without reference to the eigenproblem, ignoring the special nature of the system will affect the efficiency of the eigenproblem solver. The approximate eigenvalue ϑ as well as the approximate eigenvector $\mathbf{u}$ is updated and changes with every expansion step. The Davidson method [7, 10, 11] ignores $\mathbf{u}$ and replace ϑ by some fixed target value τ: for the expansion vector it takes the solution $\mathbf{z}$ of the system

$$\mathbf{M}\mathbf{z} = -\mathbf{r}, \tag{1.3}$$

where $\mathbf{M}$ is some convenient but fixed preconditioner for $\mathbf{A} - \tau\mathbf{B}$. A better approximation $\mathbf{M}$ of $\mathbf{A} - \lambda\mathbf{B}$ may be expected to lead to faster convergence. Unfortunately, as was observed in [6, 14], the opposite can happen; preconditioners that are excellent in the sense that they approximate $\mathbf{A} - \lambda\mathbf{B}$ very well, may even lead to stagnation of the Davidson method. Olsen, et al, [12] noted the importance of the vector $\mathbf{u}$, and — for symmetric standard eigenproblems ($\mathbf{A}^* = \mathbf{A}$ and $\mathbf{B} = \mathbf{I}$) — they suggested to compute $\mathbf{z}$ as

$$\mathbf{z} = -\mathbf{M}^{-1}\mathbf{r} + \alpha\mathbf{M}^{-1}\mathbf{q}, \quad \alpha \equiv \mathbf{u}^*\mathbf{M}^{-1}\mathbf{r}/\mathbf{u}^*\mathbf{M}^{-1}\mathbf{q}, \tag{1.4}$$

thus solving (see [20])

$$\mathbf{z} \perp \mathbf{u}, \quad \mathbf{M}_{\mathbf{u}}\mathbf{z} = -\mathbf{r} \quad \text{with} \quad \mathbf{M}_{\mathbf{u}} \equiv (\mathbf{I} - \mathbf{q}\,\mathbf{q}^*)\mathbf{M}(\mathbf{I} - \mathbf{u}\,\mathbf{u}^*), \tag{1.5}$$

still with $\mathbf{M}$ a fixed preconditioner for $\mathbf{A} - \tau\mathbf{B}$. For experimental results showing the improvements of Olsen's modification of Davidson method, see [12]. As pointed out in [20, 8, 21], approach (1.4) can be used conveniently as preconditioner for iterative linear solvers of the system (1.2). Approach (1.5) still does not take advantage of an improved approximate eigenvalue ϑ, but, more importantly, its solution $\mathbf{z}$ can be affected seriously by rounding errors if it is computed according to (1.4). To see this, consider the case where $\mathbf{M} \approx \mathbf{A} - \vartheta\mathbf{B}$. Then $\mathbf{M}^{-1}\mathbf{r} \approx \mathbf{u}$. Since $\mathbf{z} \perp \mathbf{u}$, approach (1.4) computes $\mathbf{z}$ as the difference of two nearby vectors. Therefore, serious pollution by rounding errors and loss of effectivity as expansion vector is to be expected. Note that this may happen in cases where problem (1.5) is well conditioned (if, for instance, $\mathbf{M} \approx \mathbf{A} - \lambda\mathbf{B}$, $\vartheta \approx \lambda$, $\mathbf{u} \approx \mathbf{x}$, and 0 is a simple eigenvalue of $\mathbf{A} - \lambda\mathbf{B}$). So, also with Olsen's approach, the method runs into problems if $\mathbf{M}$ is a good preconditioner for $\mathbf{A} - \lambda\mathbf{B}$. An example of the effect of this instability in the Olson approach can be found in §4.3.

The approaches and observations discussed above can also be formulated as follows. Since λ is not known in advance, a good preconditioner for $\mathbf{A} - \tau\mathbf{B}$ is used, hoping this is a good preconditioner for $\mathbf{A}-\lambda\mathbf{B}$ as well. It is expected that this leads to a good preconditioner for (1.2). Since $\mathbf{A} - \lambda\mathbf{B}$ is singular, the preconditioners that are expected to be effective, will be ill-conditioned. The ill-conditioning will be in the direction of the wanted eigenvector, which is precisely the reason why the preconditioner is expected to be effective for eigenvector computation: components in the direction of the wanted eigenvector will be amplified. Unfortunately, it also hampers stable computation, which obstructs the exploitation of the potentials of the preconditioner for eigenvalue computation.

The preconditioners for (1.2) that we introduce in this paper are of the form $\mathbf{M_u}$ with $\mathbf{M_u}$ as in (1.5), but now with $\mathbf{M} = \mathbf{M}_\vartheta$ a good preconditioner for $\mathbf{A} - \lambda\mathbf{B}$ that improves if ϑ approaches λ. For a certain representation of (1.5), we will show that, with our preconditioner, stability problems can be avoided: the solution of (1.5) can efficiently and accurately be computed. Although our $\mathbf{M}_\vartheta$ depends on ϑ and aims for some "target" eigenvalue λ, the expensive constructional work has to be done only once: the results can efficiently be used for updated ϑ's converging to λ, but also for ϑ's converging to other nearby eigenvalues.

As noted before, good preconditioners for $\mathbf{A} - \lambda\mathbf{B}$ will be ill-conditioned. If they are of block ILU-type then some diagonal block of its factors will have a relatively small singular value. For a proper and stable treatment of, among others, the projections in (1.5) we would like to control the position of this 'ill-conditioned' block. Preconditioners of multilevel ILU-type, as NGILU [22], ILUM [15], MRILU [4] and MLILU [2], offer this possibility: they push the 'ill-conditioned' block to the right-lower position while keeping the factors sparse. In this paper, we follow the MRILU approach of [4]. First, we construct a MRILU preconditioner $\mathbf{K}$ for $\mathbf{A} - \tau\mathbf{B}$. Then we modify $\mathbf{K} - \delta\mathbf{B}$ to allow efficient computations. This leads to the preconditioner $\mathbf{M}_\vartheta$ mentioned above: $\mathbf{M}_\vartheta$ is a preconditioner for the shifted problem $\mathbf{A} - \vartheta\mathbf{B} = (\mathbf{A} - \tau\mathbf{B}) - \delta\mathbf{B}$. The modification uses first order Neumann series, which also requires control over the position of the ill-conditioned block. For a stable treatment of the projections in relation to the MRILU preconditioner, it is convenient to formulate (1.5) as an augmented problem.

In §2, we discuss the ingredients for our preconditioner: Neumann series (§2.1), MRILU (§2.2), augmented systems (§2.3). Then, in §3, we explain how to put the ingredients together. There, we will also see how the preconditioner can be used for an efficient computation of a good initial search subspace (see §3.3). Numerical results are presented in §4.

2. Ingredients.

We discuss the ingredients for our preconditioner.

2.1. Neumann series.

We will be interested in eigenvalues λ close to some target value τ. Suppose we have some appropriate preconditioner $\mathbf{K}$ for $\mathbf{A} - \tau\mathbf{B}$. Since τ approximates λ, $\mathbf{K}$ can be viewed as a preconditioner for $\mathbf{A} - \lambda\mathbf{B}$. Nevertheless, we would like to improve $\mathbf{K}$ efficiently to an even better preconditioner for $\mathbf{A} - \lambda\mathbf{B}$ as better approximations for λ become available. For this, we use first order Neumann series. Our approach can be described as follows.

Suppose ϑ is a better approximation for λ than τ. Then, with $\delta_\vartheta \equiv \vartheta - \tau$, it is tempting to take $\mathbf{K} - \delta_\vartheta\mathbf{B}$ as preconditioner for $\mathbf{A} - \vartheta\mathbf{B}$. Note that inclusion of the term $\delta_\vartheta\mathbf{B}$ will not lead to a better preconditioner if $\mathbf{K}$ itself is not a good preconditioner for $\mathbf{A} - \tau\mathbf{B}$, since then the 'error' in $\mathbf{K}$ will dominate improvements from the $\delta_\vartheta\mathbf{B}$ term. Naturally, the shifted operator $\mathbf{K} - \delta_\vartheta\mathbf{B}$ can only be viewed as preconditioner if its associated systems can be solved efficiently. In our applications, rank-one projections will be involved (see (1.5)), but, for simplicity, let us first follow Olson's approach (1.4) and concentrate on solving the system $(\mathbf{K} - \delta_\vartheta\mathbf{B})\mathbf{s} = \mathbf{r}$. This can be done approximately using first order Neumann series,

$$(\mathbf{K} - \delta_\vartheta\mathbf{B})^{-1} = (\mathbf{I} - \delta_\vartheta\mathbf{K}^{-1}\mathbf{B})^{-1}\mathbf{K}^{-1} \approx \mathbf{K}^{-1} + \delta_\vartheta\mathbf{K}^{-1}\mathbf{B}\mathbf{K}^{-1},$$

provided that $\mathbf{K}$ is well conditioned (that is, $\|\mathbf{K}^{-1}\mathbf{B}\|$ is not too large). As preconditioner for $\mathbf{A} - \lambda\mathbf{B}$, we could use $\mathbf{M} = \mathbf{M}_\vartheta$ with $\mathbf{M}_\vartheta$ such that $\mathbf{M}_\vartheta^{-1} = \mathbf{K}^{-1} + \delta_\vartheta\mathbf{K}^{-1}\mathbf{B}\mathbf{K}^{-1}$. The system $\mathbf{M}_\vartheta\mathbf{s} = \mathbf{r}$ can be solved relatively efficiently. Therefore, if $\mathbf{K}$ is well conditioned and a good preconditioner for $\mathbf{A} - \tau\mathbf{B}$, then $\mathbf{M}_\vartheta$ may be expected to improve $\mathbf{K}$ as preconditioner for $\mathbf{A} - \lambda\mathbf{B}$.

Unfortunately, if τ is close to λ, then $\mathbf{K}$ may not be expected to be both well conditioned and a good preconditioner for $\mathbf{A} - \tau\mathbf{B}$. Therefore, the approach sketched above needs some modification. We will partition $\mathbf{K}$ and use first order Neumann series on well-conditioned parts and exact inversion on others (see §2.2). Since, specifically for good preconditioners $\mathbf{M}$ for $\mathbf{A}-\lambda\mathbf{B}$, Olsen's approach (1.4) is instable, we will use another representation of (1.5) (see §2.3).

2.2. Matrix Renumbering Incomplete LU-decomposition (MRILU). If τ is a good target, i.e., there is some eigenvalue λ close to τ, then $\mathbf{A}-\tau\mathbf{B}$ will be ill-conditioned. Block LU-decompositions of $\mathbf{A}-\tau\mathbf{B}$ will have an ill-conditioned[1] diagonal block. This will also be the case for good incomplete decompositions. Rows and columns can be simultaneously reordered such that the ill-conditioned block will appear at the right-lower position. A reordering and partitioning strategy can be used to identify a well-conditioned diagonal block $\mathbf{A}_1$ of $\mathbf{A}-\tau\mathbf{B}$ of almost full dimension:

$$\mathbf{A}-\tau\mathbf{B}=\begin{bmatrix}\mathbf{A}_1 & \mathbf{E}_u\\ \mathbf{E}_\ell & \mathbf{A}_2\end{bmatrix}, \tag{2.1}$$

some details will be given below. The Schur complement of the well-conditioned part is an ill-conditioned matrix block but it is one of low dimension. A preconditioner $\mathbf{K}$ can now be constructed by approximating the well-conditioned diagonal block by an incomplete LU-decomposition $\mathbf{K}_1$. The other blocks, in particular the diagonal block associated with the ill-conditioned Schur complement, are included in $\mathbf{K}$ as they are:

$$\mathbf{K}=\begin{bmatrix}\mathbf{K}_1 & \mathbf{E}_u\\ \mathbf{E}_\ell & \mathbf{A}_2\end{bmatrix}=\begin{bmatrix}\mathbf{K}_1 & \mathbf{0}\\ \mathbf{E}_\ell & \mathbf{I}\end{bmatrix}\begin{bmatrix}\mathbf{I} & \mathbf{K}_1^{-1}\mathbf{E}_u\\ \mathbf{0} & \widetilde{\mathbf{A}}_2\end{bmatrix}. \tag{2.2}$$

In our applications here, we are interested in a modification of the preconditioner $\mathbf{K}$ in which the projections of (1.5) are used for stabilization. In §2.3 we will explain how to accommodate these projections such that the ill-conditioning of the Schur complement $\widetilde{\mathbf{A}}_2\equiv\mathbf{A}_2-\mathbf{E}_\ell\mathbf{K}_1^{-1}\mathbf{E}_u$ of $\mathbf{K}_1$ in $\mathbf{K}$ is harmless. The preconditioner $\mathbf{K}$, or rather the factors of $\mathbf{K}_1$, can be constructed simultaneously with the reordering and the partitioning following a recursive construction.

We give some details of the strategy followed in the construction of MRILU [4] that we employ in our experiments.[2] For simplicity of presentation, we assume that $\tau=0$: in the general situation, $\mathbf{A}$ in the construction below can be replaced by $\mathbf{A}-\tau\mathbf{B}$.

In the first step, the columns and the rows of $\mathbf{A}$ are reordered simultaneously such that, with respect to the new ordering, $\mathbf{A}$ can be partitioned as

$$\mathbf{A}=\begin{bmatrix}\mathbf{A}_{11} & \mathbf{A}_{12}\\ \mathbf{A}_{21} & \mathbf{A}_{22}\end{bmatrix}$$

with square diagonal blocks with the following properties: $\mathbf{A}_{11}$ is well conditioned, $\mathbf{A}_{11}$ is of size of order n (i.e., $\mathbf{A}_{11}$ is $\kappa n\times\kappa n$ with $0\ll\kappa\leq 1$), and a matrix $\mathbf{C}_1$ that approximates $\mathbf{A}_{11}^{-1}$ well, is explicitly available. In fact, the ordering and partitioning strategy in MRILU yields a block $\mathbf{A}_{11}$ that is strongly diagonal dominant, and for $\mathbf{C}_1$, the inverse of the diagonal of $\mathbf{A}_{11}$ is taken. With the Schur complement

$$\widetilde{\mathbf{A}}_{22}\equiv\mathbf{A}_{22}-\mathbf{A}_{21}\mathbf{C}_1\mathbf{A}_{12}, \tag{2.3}$$

[1] One of the diagonal blocks will have a singular value that is relatively small with respect to the singular values of the other blocks. For ease of conversation, we say that this block is ill-conditioned.

[2] In practical computations, the reordering is through renumbering of indices. This explains why 'renumbering' is used in the name MRILU.

the preconditioner

$$\begin{bmatrix} \mathbf{C}_1^{-1} & \mathbf{A}_{12} \\ \mathbf{A}_{21} & \mathbf{A}_{22} \end{bmatrix} = \begin{bmatrix} \mathbf{C}_1^{-1} & \mathbf{0} \\ \mathbf{A}_{21} & \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{I} & \mathbf{C}_1\mathbf{A}_{12} \\ \mathbf{0} & \widetilde{\mathbf{A}}_{22} \end{bmatrix} \tag{2.4}$$

can be associated with this first step.
The computation of an exact solution of the preconditioned system is feasible only if the size of $\widetilde{\mathbf{A}}_{22}$ is much less than n. If this is not the case, a reordering, partitioning and approximation strategy as for $\mathbf{A}$ can be applied to $\widetilde{\mathbf{A}}_{22}$ in the second step. This can be repeated until, say, in step k, the Schur complement $\widetilde{\mathbf{A}}_{k+1\,k+1}$ is of low dimension, or the strategy fails to identify a well-conditioned part of relatively large size.

In the end, we have reordered and partitioned the matrix $\mathbf{A}$ as a $k+1$ by $k+1$ block matrix with, from top to bottom, blocks of exponentially shrinking order. The k by k left upper part is the matrix $\mathbf{A}_1$ of (2.1), $\widetilde{\mathbf{A}}_{k+1\,k+1}$ is $\widetilde{\mathbf{A}}_2$ of (2.2). The preconditioner $\mathbf{K}$ in (2.2) is exactly the one that is obtained if the standard construction of the factors L and U of the block LU-decomposition of the reordered $\mathbf{A}$ is followed, but with the inverses of the "pivot blocks" replaced by $\mathbf{C}_1, \mathbf{C}_2, \ldots$. In our actual computations, we store the ingredients (permutation, partitioning, the approximate inverses $\mathbf{C}_j, \ldots$) and solve systems associated with $\mathbf{K}$ by forward elimination and back-substitution. Note that, in the successive steps, the Schur complements (as $\widetilde{\mathbf{A}}_{22}$) will be more sparse if small elements of the off-diagonal blocks in ($\mathbf{A}_{21}$ and $\mathbf{A}_{12}$) are removed as well (dropped or lumped to the diagonal). Sparse Schur complements allow diagonal blocks of higher order (i.e., they allow larger κ). This is desirable since it may increase the efficiency of the preconditioning steps. On the other hand, inaccurate Schur complements may result in a block $\widetilde{\mathbf{A}}_2$ that is less well conditioned or of higher order, which may result in a less effective preconditioner. The dropping and lumping strategies form the major differences within the class of the preconditioners of multilevel ILU-type. In MRILU, the removed elements are lumped to the diagonal, and the dropping tolerance, determined by a parameter ε, is a relative one. Elements are dropped when they are small with respect to relevant diagonal elements of the Schur complement yet to be formed. For the purpose of the dropping criterion, these diagonal elements are temporarily computed in advance. With this "look-ahead" step, growth of the ill-conditioning of the Schur complements due to the lumping can be avoided. Note that the parameter ε determines the fill. For a detailed discussion, we refer to [4]. We also refer to this citation for details on the relation to similar approaches as, e.g., in [22, 15, 13].

Of course, $\mathbf{B}$ and all n-vectors should be ordered and partitioned as $\mathbf{A}-\tau\mathbf{B}$ (cf. (2.1)). In the sequel of this paper, we will implicitly assume that this is the case.

2.3. Augmented matrices. As explained in §2.2, the preconditioner $\mathbf{M}$ may be expected to be ill-conditioned. However, this will not be the case for its projection $(\mathbf{I}-\mathbf{q}\,\mathbf{q}^*)\mathbf{M}(\mathbf{I}-\mathbf{u}\,\mathbf{u}^*)$. The see this, we first note that (see [18]) (1.5) is mathematically equivalent to the augmented system

$$\begin{bmatrix} \mathbf{M} & \mathbf{q} \\ \mathbf{u}^* & 0 \end{bmatrix} \begin{bmatrix} \mathbf{z} \\ \alpha \end{bmatrix} = -\begin{bmatrix} \mathbf{r} \\ 0 \end{bmatrix}, \tag{2.5}$$

that is, $\mathbf{z}$ is an exact solution of (1.5) if and only if it exactly solves (2.5) for some scalar α. In §3.1, we will explain why representation (2.5) is also convenient in connection with the multilevel ILU preconditioner of the previous section.

We will argue that (2.5) is well conditioned if λ is a well-conditioned eigenvalue of the generalized eigenproblem (1.1), $\mathbf{M}$ is a reasonable preconditioner for $\mathbf{A}-\lambda\mathbf{B}$, and the angle between the vector $\mathbf{u}$ and the eigenvector $\mathbf{x}$ is not too wide (see also [5, 1]).

Consider the angle ϕ between the left kernel vector $\mathbf{y}$ of $\mathbf{A} - \lambda\mathbf{B}$ and the vector $\mathbf{Bx}$. Note that $\mathbf{y}^*\mathbf{B}(\mathbf{B}^{-1}\mathbf{A} - \lambda\mathbf{I}) = \mathbf{0}^*$ and $\mathbf{B}^{-1}\mathbf{Ax} = \lambda\mathbf{x}$. Moreover, $(\mathbf{y}^*\mathbf{B})\mathbf{x} = \mathbf{y}^*(\mathbf{Bx})$. Therefore, $1/\cos\phi$ (cf. [14, Def. 3.1]) can be viewed as the conditioning of the eigenvalue λ of the generalized eigenproblem.

Suppose that 0 is a simple eigenvalue of $\mathbf{M}$. Consider a right eigenvector $\widetilde{\mathbf{x}}$ and a left eigenvector $\widetilde{\mathbf{y}}$ of $\mathbf{M}$, both vectors are normalized: $\mathbf{M}\widetilde{\mathbf{x}} = \mathbf{0}$, $\widetilde{\mathbf{y}}^*\mathbf{M} = \mathbf{0}^*$. Then $\mathbf{M}$ maps the space orthogonal to $\widetilde{\mathbf{y}}$ onto itself. If $\mathcal{C}$ is the condition number of this map, then for some moderate constant κ, the condition number of the system in (2.5) is bounded by

$$\frac{\kappa\,\mathcal{C}}{\cos^2\angle(\widetilde{\mathbf{x}},\widetilde{\mathbf{y}})}\cdot\frac{1}{\cos\angle(\mathbf{u},\widetilde{\mathbf{x}})}\cdot\frac{1}{\cos\angle(\mathbf{q},\widetilde{\mathbf{y}})}.$$

The angle between the left and the right eigenvector determines the conditioning of the eigenvalue 0 [14, Def. 3.1]: $\mathrm{Cond}(0) = 1/\cos\angle(\widetilde{\mathbf{x}},\widetilde{\mathbf{y}})$. For similar results, see [17, Lemma 5]. Eigenproblem solvers will compute vectors $\mathbf{u}$ that are directionally close to $\mathbf{x}$ and $\cos(\mathbf{u},\widetilde{\mathbf{x}})$ will be bounded away from zero if $\widetilde{\mathbf{x}}$ has a reasonable component in the direction $\mathbf{x}$. Similarly, $\mathbf{q} = \mathbf{Bu}$ will be directionally close to $\mathbf{Bx}$, and $\cos(\mathbf{q},\widetilde{\mathbf{y}}) \approx \cos\phi$ if $\widetilde{\mathbf{y}}$ is directionally close to $\mathbf{y}$. Therefore, if the eigenvalue λ is well conditioned, then the $\cos(\mathbf{q},\widetilde{\mathbf{y}})$ will be bounded away from 0 if $\widetilde{\mathbf{y}}$ is sufficiently directionally close to $\mathbf{y}$. Note that what is 'sufficient' depends on the conditioning of λ.

The result on the conditioning of the system in (2.5) assumes $\mathbf{u}$ to be close (in angle) to $\mathbf{x}$. In [24], an example is given in which the system in (2.5), for relevant approximations $\mathbf{u}$ of $\mathbf{x}$, is ill-conditioned if $\mathbf{u}$ is far from $\mathbf{x}$, while for the same vectors $\mathbf{u}$ the conditioning of (1.5) is much better. For $\mathbf{u}$ close to $\mathbf{x}$ the conditioning of both systems is comparable and acceptable. In §3.3 we will explain how the multilevel ILU preconditioners of §2.2 can provide accurate approximations of the desired eigenvectors $\mathbf{x}$ before starting the Jacobi-Davidson process. Then, by taking such an approximation as initial guess in the Jacobi-Davidson method, the stage in which (2.5) is ill-conditioned can be avoided.

In the above discussion we assumed that $\mathbf{q} = \mathbf{Bu}$. This choice corresponds to a test subspace that is the image of the search subspace under $\mathbf{B}$. We considered this choice only for ease of explanation: alternatives, where $\mathbf{B}$ is replaced by another linear or convex combination of $\mathbf{B}$ and $\mathbf{A}$, can improve the performance of the eigenproblem solver [8]. In [8], it is argued that, for test subspaces of this type, the correction equation (1.2) with smallest condition number is to be expected for $\bar{\tau}\mathbf{A} + \mathbf{B}$. The same observation applies to (1.5) and (2.5). Note that working with images under $\bar{\tau}\mathbf{A} + \mathbf{B}$ is equivalent to the above approach for the shifted eigenproblem $(\mathbf{A} - \tau\mathbf{B})\mathbf{x} = \varepsilon(\bar{\tau}\mathbf{A} + \mathbf{B})\mathbf{x}$. However, for the computation of interior eigenvalues, images under $\mathbf{A} - \tau\mathbf{B}$ (the harmonic Petrov-Galerkin approach, cf. [8]) facilitate safer selection of the approximate eigenvector from the search subspace. Experiments in [8] show that, in particular, in case restarts are required, the harmonic Petrov approach is to be preferred. Safer selection appears to compensate for a weaker conditioning of the augmented systems (2.5) that are associated with harmonic Petrov values.

3. The recipe. Now, we only have to put the ingredients properly together to find effective preconditioners for (1.2).
First, construct a preconditioner $\mathbf{K}$ for $\mathbf{A} - \tau\mathbf{B}$ as explained in §2.2. Next, follow the ideas of §2.1 and modify $\mathbf{K}$ to find a preconditioner for $\mathbf{A} - \vartheta\mathbf{B} = (\mathbf{A} - \tau\mathbf{B}) - \delta_\vartheta\,\mathbf{B}$ with the same block structure and similar conditioning as $\mathbf{K}$. We give details in §3.2. Finally, augment the resulting preconditioning system (see §2.3) to accommodate the rank-one projections.

In §3.1, we will explain why and how augmentation of $\mathbf{K}$ leads efficiently to an accurate solution of the projected preconditioning equation (1.5) with $\mathbf{M} = \mathbf{K}$.

Once the factors of the preconditioner are available, a good initial approximation for the eigenpair $(\lambda, \mathbf{x})$, with λ a generalized eigenvalue close to τ, can also be efficiently computed, as we will see in §3.3.

In this section, we suppose that $\mathbf{K}$ is a good preconditioner for $\mathbf{A} - \tau\mathbf{B}$, τ is a good target, i.e., τ is close to some eigenvalue λ, and $\mathbf{K}$ is ordered (same ordering for the rows and the columns) and partitioned as in (2.2): $\mathbf{K}_1$ is well conditioned and of dimension almost n, and the systems associated with $\mathbf{K}_1$ can be solved efficiently. For $\mathbf{B}$ we use the same ordering and partitioning:

$$\mathbf{K} = \begin{bmatrix} \mathbf{K}_1 & \mathbf{E}_u \\ \mathbf{E}_\ell & \mathbf{A}_2 \end{bmatrix} \quad \& \quad \mathbf{B} = \begin{bmatrix} \mathbf{B}_1 & \mathbf{F}_u \\ \mathbf{F}_\ell & \mathbf{B}_2 \end{bmatrix}. \tag{3.1}$$

3.1. Accurate solution of systems with the augmented preconditioner. To see why the present preconditioner $\mathbf{K}$ allows a stable and efficient solution method of (2.5) with $\mathbf{M} = \mathbf{K}$, note that, with $(\mathbf{u}_1^*, \mathbf{u}_2^*) = \mathbf{u}^*$, $(\mathbf{q}_1^*, \mathbf{q}_2^*) = \mathbf{q}^*$,

$$\widetilde{\mathbf{u}}_2^* \equiv \mathbf{u}_2^* - \mathbf{u}_1^*\mathbf{K}_1^{-1}\mathbf{E}_u, \quad \widetilde{\mathbf{q}}_2 \equiv \mathbf{q}_2 - \mathbf{E}_\ell\mathbf{K}_1^{-1}\mathbf{q}_1, \quad \text{and} \quad \beta \equiv -\mathbf{u}_1^*\mathbf{K}_1^{-1}\mathbf{q}_1,$$

we have

$$\begin{bmatrix} \mathbf{K}_1 & \mathbf{E}_u & \mathbf{q}_1 \\ \mathbf{E}_\ell & \mathbf{A}_2 & \mathbf{q}_2 \\ \mathbf{u}_1^* & \mathbf{u}_2^* & 0 \end{bmatrix} = \begin{bmatrix} \mathbf{K}_1 & \mathbf{0} & \mathbf{0} \\ \mathbf{E}_\ell & \mathbf{I} & \mathbf{0} \\ \mathbf{u}_1^* & \mathbf{0}^* & 1 \end{bmatrix} \begin{bmatrix} \mathbf{I} & \mathbf{K}_1^{-1}\mathbf{E}_u & \mathbf{K}_1^{-1}\mathbf{q}_1 \\ \mathbf{0} & \widetilde{\mathbf{A}}_2 & \widetilde{\mathbf{q}}_2 \\ \mathbf{0}^* & \widetilde{\mathbf{u}}_2^* & \beta \end{bmatrix}. \tag{3.2}$$

Since τ is close to λ, $\mathbf{K}$ will also be a good preconditioner for $\mathbf{A} - \lambda\mathbf{B}$ and the matrix at the left-hand side of (3.2) may expected to be well conditioned (cf. §2.3). Moreover, $\mathbf{K}_1$ is well conditioned. Therefore, although $\widetilde{\mathbf{A}}_2$ is an ill-conditioned block, the augmented matrix

$$\begin{bmatrix} \widetilde{\mathbf{A}}_2 & \widetilde{\mathbf{q}}_2 \\ \widetilde{\mathbf{u}}_2^* & \beta \end{bmatrix} \tag{3.3}$$

may be expected to be well conditioned, and its associated systems can be accurately solved with direct methods (LU-decomposition, possibly using entries from the last row as pivot). Since (3.3) is of low dimension, direct methods for (3.3) are efficient enough.

Note that (3.2) can also be obtained, if the reordering, partitioning, and approximation strategy leading to $\mathbf{K}$ is applied to the augmented system associated with $\mathbf{A} - \tau\mathbf{B}$ and the reordering did not touch the position of the last row.

3.2. Updating the incomplete factorization. The preconditioner for $\mathbf{A} - \vartheta\mathbf{B} = (\mathbf{A} - \tau\mathbf{B}) - \delta_\vartheta\mathbf{B}$ is constructed from $\mathbf{K} - \delta_\vartheta\mathbf{B}$. A straight-forward factorization according to the block partitioning of (3.1) would require inversion of $\mathbf{K}_1 - \delta_\vartheta\mathbf{B}_1$, or solving systems involving this operator, which are both unattractive options. Therefore, we choose to approximate the inverse by $\widetilde{\mathbf{C}}_\vartheta \equiv (\mathbf{I} + \delta_\vartheta[\mathbf{K}_1^{-1}\mathbf{B}_1])\mathbf{K}_1^{-1}$. Since $\mathbf{K}_1$ is well conditioned, this is accurate provided that δ_ϑ is not too large (cf. §2.1). The resulting approximate Schur complement of $\mathbf{K}_1 - \delta_\vartheta\mathbf{B}_1$ in $\mathbf{K} - \vartheta\mathbf{B}$ is given by

$$\widetilde{\widetilde{\mathbf{A}}}_2 \equiv \mathbf{A}_2 - \delta_\vartheta\mathbf{B}_2 - (\mathbf{E}_\ell - \delta_\vartheta\mathbf{F}_\ell)\widetilde{\mathbf{C}}_\vartheta(\mathbf{E}_u - \delta_\vartheta\mathbf{F}_u)$$

which can be simplified further by neglecting other terms of order δ_ϑ^2 as well:

$$\widetilde{\widetilde{\mathbf{A}}}_2 \approx \widetilde{\mathbf{A}}_2 - \delta_\vartheta\widetilde{\mathbf{B}}_2$$

with

$$\left\{\begin{array}{l}\widetilde{\mathbf{A}}_2 = \mathbf{A}_2 - \mathbf{E}_\ell \mathbf{K}_1^{-1}\mathbf{E}_u \\ \widetilde{\mathbf{B}}_2 \equiv [\mathbf{B}_2 - \mathbf{E}_\ell \mathbf{K}_1^{-1}\mathbf{F}_u] - [\mathbf{F}_\ell - \mathbf{E}_\ell \mathbf{K}_1^{-1}\mathbf{B}_1]\,[\mathbf{K}_1^{-1}\mathbf{E}_u].\end{array}\right. \tag{3.4}$$

Note that $\widetilde{\mathbf{A}}_2$ is the Schur complement of $\mathbf{K}_1$ in $\mathbf{K}$ (cf. (2.2)). The simplifications lead to the following approximate block factorization of $\mathbf{K} - \delta_\vartheta \mathbf{B}$, that we use as preconditioner for $\mathbf{A} - \vartheta \mathbf{B}$:

$$\mathbf{M} \equiv \mathbf{M}_{\delta_\vartheta} \equiv \begin{bmatrix} \widetilde{\mathbf{C}}_\vartheta^{-1} & \mathbf{0} \\ \mathbf{E}_\ell - \delta_\vartheta \mathbf{F}_\ell & \mathbf{I} \end{bmatrix} \begin{bmatrix} \mathbf{I} & \widetilde{\mathbf{C}}_\vartheta(\mathbf{E}_u - \delta_\vartheta \mathbf{F}_u) \\ \mathbf{0} & \widetilde{\mathbf{A}}_2 - \delta_\vartheta \widetilde{\mathbf{B}}_2 \end{bmatrix}. \tag{3.5}$$

The system $\mathbf{M}(\mathbf{x}_1^{\mathrm{T}}, \mathbf{x}_2^{\mathrm{T}})^{\mathrm{T}} = (\mathbf{b}_1^{\mathrm{T}}, \mathbf{b}_2^{\mathrm{T}})^{\mathrm{T}}$ can be solved as follows:

(1)	*Solve* $\mathbf{x}_1$ *from*	$\mathbf{K}_1\mathbf{x}_1 = \mathbf{b}_1$
(2)	*Compute*	$\mathbf{x}_1 \leftarrow \mathbf{x}_1 + \delta_\vartheta[\mathbf{K}_1^{-1}\mathbf{B}_1]\mathbf{x}_1$
(3)	*Compute*	$\widetilde{\mathbf{x}}_2 = \mathbf{b}_2 - \mathbf{E}_\ell\mathbf{x}_1 - \delta_\vartheta \mathbf{F}_\ell \mathbf{x}_1$
(4)	*Solve* $\mathbf{x}_2$ *from*	$(\widetilde{\mathbf{A}}_2 - \delta_\vartheta \widetilde{\mathbf{B}}_2)\mathbf{x}_2 = \widetilde{\mathbf{x}}_2$
(5)	*Compute*	$\widetilde{\mathbf{x}}_1 = [\mathbf{K}_1^{-1}\mathbf{E}_u]\mathbf{x}_2 - \delta_\vartheta[\mathbf{K}_1^{-1}\mathbf{F}_u]\mathbf{x}_2$
(5)	*Compute*	$\mathbf{x}_1 \leftarrow \mathbf{x}_1 - \widetilde{\mathbf{x}}_1 - \delta_\vartheta[\mathbf{K}_1^{-1}\mathbf{B}_1]\widetilde{\mathbf{x}}_1$

The components $\mathbf{K}_1$, $\widetilde{\mathbf{A}}_2$, $\widetilde{\mathbf{B}}_2$ and the quantities in square brackets are computed in the construction phase of the preconditioner (as will be explained below). None of these quantities depends on ϑ. The only non-linear dependency on ϑ is in $(\widetilde{\mathbf{A}}_2 - \delta_\vartheta \widetilde{\mathbf{B}}_2)^{-1}$ (in step (4)). But the matrix here is of low dimension. Therefore, the scalar ϑ can be changed in the preconditioner at virtually no extra costs: the preconditioner can be updated efficiently whenever a better approximate eigenvalue ϑ becomes available.

The matrix

$$\widehat{\mathbf{B}} \equiv \begin{bmatrix} \mathbf{K}_1^{-1}\mathbf{B}_1 & \mathbf{K}_1^{-1}\mathbf{F}_u \\ \mathbf{F}_\ell - \mathbf{E}_\ell\mathbf{K}_1^{-1}\mathbf{B}_1 & \mathbf{B}_2 - \mathbf{E}_\ell\mathbf{K}_1^{-1}\mathbf{F}_u \end{bmatrix} \equiv \begin{bmatrix} \mathbf{K}_1 & \mathbf{0} \\ \mathbf{E}_\ell & \mathbf{I} \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{B}_1 & \mathbf{F}_u \\ \mathbf{F}_\ell & \mathbf{B}_2 \end{bmatrix} \tag{3.6}$$

is computed recursively from $\mathbf{B}$ simultaneously with the preconditioner $\mathbf{K}$: whenever a reordering, partioning and block elimination step is applied to the appropriate Schur complement in $\mathbf{A}$, the same steps are applied to the corresponding blocks in the 'updated $\mathbf{B}$' (the 'current' $\widehat{\mathbf{B}}$ is multiplied by the inverse of the left factor in (2.4)). Further, to maintain sparsity, our lumping strategy for $\mathbf{A}$ is also followed for $\mathbf{B}$, in the intermediate steps as well as in the construction of $\widetilde{\mathbf{B}}_2$ from the contributing blocks (cf. (3.6) and (3.4)).

The major costs for constructing the preconditioners $\mathbf{K}$ and the components for its update $\mathbf{M}$ are in the construction of $\mathbf{K}$ (finding the reordering, partitioning and diagonal approximations). Moreover, in our applications, the matrix $\mathbf{B}$ is more sparse than $\mathbf{A}$. Therefore, the additional costs for constructing the factors in (3.6) and $\widetilde{\mathbf{B}}_2$ are small. Similarly, the additional costs for working with $\mathbf{M}$ rather than with $\mathbf{K}$ are small.

The modification as suggested in §3.1 for $\mathbf{K}$ (see (3.2)) can also be applied to $\mathbf{M}$ and leads to accurate solutions of system (1.5).

3.3. Computing approximate eigenpairs of the preconditioner. Since $\mathbf{K}$ is a good preconditioner for $\mathbf{A} - \tau\mathbf{B}$, the small eigenvalue δ of the generalized eigenvalue problem

$$(\mathbf{K} - \delta\mathbf{B})\mathbf{y} = \mathbf{0} \tag{3.7}$$

may be expected to be a good approximation for $\lambda - \tau$, for λ close to τ. The eigenvector $\mathbf{y}$ associated with δ will have a relatively large component in the direction of the eigenvector $\mathbf{x}$ of (1.1) associated with λ. We will indicate how, for the present preconditioner $\mathbf{K}$, an approximate solution of problem (3.7) can be obtained efficiently. This approximation for $\mathbf{y}$, and thus for $\mathbf{x}$, can be included in the initial search subspace of the Jacobi-Davidson process.

Consider the operator $\mathbf{M}_\delta = \mathbf{M}$ of (3.5), now with $\delta_\vartheta = \delta$. As motivated in §3.2, $\mathbf{M}_\delta$ approximates $\mathbf{K} - \delta\mathbf{B}$ well for small δ. Therefore, if $\mathbf{M}_{\widetilde{\delta}}$ is singular for some small value $\widetilde{\delta}$ and $\mathbf{M}_{\widetilde{\delta}}\widetilde{\mathbf{y}} = \mathbf{0}$ then $(\mathbf{K} - \widetilde{\delta}\mathbf{B})\widetilde{\mathbf{y}} \approx \mathbf{0}$ and $(\widetilde{\delta}, \widetilde{\mathbf{y}})$ will solve (3.7) approximately (which can be seen from Bauer-Fike's theorem). For ease of discussion, we call these approximate eigenpairs $(\widetilde{\delta}, \widetilde{\mathbf{y}})$ of the generalized eigenproblem (3.7) *pre-eigenpairs* (of the preconditioner). With $\widetilde{\mathbf{y}} = (\widetilde{\mathbf{y}}_1^{\mathrm{T}}, \widetilde{\mathbf{y}}_2^{\mathrm{T}})^{\mathrm{T}}$, the problem $\mathbf{M}_{\widetilde{\delta}}\widetilde{\mathbf{y}} = \mathbf{0}$ is equivalent to

$$(3.8)\qquad \begin{cases} \widetilde{\mathbf{A}}_2\widetilde{\mathbf{y}}_2 - \widetilde{\delta}\,\widetilde{\mathbf{B}}_2\widetilde{\mathbf{y}}_2 = \mathbf{0}, \\ \widetilde{\mathbf{y}}_1 = -(\mathbf{I} + \widetilde{\delta}\,[\mathbf{K}_1^{-1}\mathbf{B}_1])([\mathbf{K}_1^{-1}\mathbf{E}_u] - \widetilde{\delta}\,[\mathbf{K}_1^{-1}\mathbf{F}_u])\widetilde{\mathbf{y}}_2. \end{cases}$$

Note that $\widetilde{\mathbf{y}}_1$ can easily be computed if $\widetilde{\delta}$ and $\widetilde{\mathbf{y}}_2$ are available (second eq. of (3.8)). These quantities $\widetilde{\delta}$ and $\widetilde{\mathbf{y}}_2$ are the solution of a generalized eigenvalue problem of low dimension (first eq. of (3.8)) and they can be computed exactly using dense matrix techniques (QZ-algorithm [9]).

Since the matrices $\widetilde{\mathbf{A}}_2$ and $\widetilde{\mathbf{B}}_2$ do not depend on δ or $\mathbf{y}$, this approach can be used to obtain approximations for all eigenvectors for problem (3.7) that are associated with sufficiently small eigenvalues δ. If the ingredients for the preconditioner $\mathbf{M}$ have been computed, then the pre-eigenpairs can be efficiently computed at hardly any additional computational costs.

In case of a standard symmetric eigenvalue problem (i.e., $\mathbf{A}^* = \mathbf{A}$ and $\mathbf{B} = \mathbf{I}$), our approach here for computing approximate eigenvalues coincides with one in [3], where wavelet-based type of preconditioners are discussed for certain symmetric eigenvalue problems. The derivation in [3] runs along other lines and it seems that it cannot be easily extended to the case of generalized eigenvalue problems. For standard symmetric eigenvalue problems, [3] also provides error bounds for the approximate eigenvalues $\widetilde{\delta}$.

3.4. Discussion. The preconditioner $\mathbf{M}_\delta$ in (3.5) may be expected to be effective for eigenvectors that are close to singular vectors of $\mathbf{M}_\delta$ for some δ close to $\tau - \vartheta$. For eigenvalues λ that are further away from τ, $\mathbf{M}_\delta$ could be employed as well, but success can not be guaranteed. For larger δ, the Neumann series approximation $\mathbf{K}_1^{-1} + \delta\mathbf{K}_1^{-1}\mathbf{B}_1\mathbf{K}_1^{-1}$ of $(\mathbf{K}_1 - \delta\mathbf{B}_1)^{-1}$ may not be accurate enough (on the space spanned by the components $\mathbf{x}_1$ of wanted eigenvectors $\mathbf{x}$), $\mathbf{A}_1 - \delta\mathbf{B}_1$ can be ill-conditioned, etc..

Small singular values in the Schur complement $\widetilde{\mathbf{A}}_2$ of $\mathbf{K}_1$ in (2.2) may reflect the fact that $\mathbf{K}$ is a preconditioner for the near-singular matrix $\mathbf{A} - \tau\mathbf{B}$. They also may have been introduced by the approximation of $\mathbf{A}_1^{-1}$ by $\mathbf{K}_1^{-1}$. This approximation can be relatively accurate ($\|\mathbf{A}_1^{-1} - \mathbf{K}_1^{-1}\| \ll \|\mathbf{A}_1^{-1}\|$), while the error in the resulting approximate Schur complement can be relatively large:

$$\|\mathbf{E}_\ell^*(\mathbf{A}_1^{-1} - \mathbf{K}_1^{-1})\mathbf{E}_u\| \not\ll \|\mathbf{A}_2 - \mathbf{E}_\ell^*\mathbf{A}_1^{-1}\mathbf{E}_u\|.$$

If the desired eigenvalue is in a a cluster of, say, ℓ eigenvalues, the rank one projections in (1.2) and (1.5) and the 1-dimensional expansion (2.5) will not substantially improve the conditioning of the systems. For this type of problems, a block version of the Jacobi-Davidson method would be more appropriate. In such a version, the vector $\mathbf{u}$ is replaced by an $n\,\times$

ℓ orthogonal matrix $\mathbf{U}$, where the columns of $\mathbf{U}$ form a basis for the invariant subspace associated with the cluster of eigenvalues. The eigensystem of the preconditioner (cf. §3.3) can provide an estimate of the size of the cluster.

If one approximation of an eigenpair has been accepted, a search can be started for another eigenpair. To enhance the performance of the method, the detected eigenvector should be included in the process. Including the detected eigenvector in the search subspace (explicit deflation) prevents the method from recomputing the same old vector. This can also be achieved by restricting the eigenproblem to some appropriate complement of the detected eigenspace [8]. We will give some details on this last approach since it also improves the conditioning of the correction equation. The effects of this improvement (more stability, faster converging linear solver) compensate for the additional computational costs for handling the restrictions [8]. For stability reasons and to facilitate computations, orthogonal vectors are preferred: rather than computing ℓ eigenvectors, a *partial generalized Schur form of order* ℓ is computed:

$$\mathbf{AQ} = \mathbf{Z}S \quad \text{and} \quad \mathbf{BQ} = \mathbf{Z}T$$

with $\mathbf{Q}$ and $\mathbf{Z}$ are $n \times \ell$ orthonormal and S and T are $\ell \times \ell$ upper triangular. Eigenpairs for the pencil $\mathbf{A} - \lambda\mathbf{B}$ can easily be extracted from this partial Schur form, since $(\mathbf{A} - \lambda\mathbf{B})\mathbf{Q}x = \mathbf{0}$ if $(S - \lambda T)x = 0$; λ is a diagonal element of $T^{-1}S$.

The next *Schur vector*, the new $(\ell+1)$st column for $\mathbf{Q}$, is an eigenvector $\mathbf{x}$ of the deflated generalized eigenproblem

$$\mathbf{Q}^*\mathbf{x} = 0, \quad (\mathbf{I} - \mathbf{Z}\,\mathbf{Z}^*)\mathbf{A}(\mathbf{I} - \mathbf{Q}\,\mathbf{Q}^*)\mathbf{x} - \lambda(\mathbf{I} - \mathbf{Z}\,\mathbf{Z}^*)\mathbf{B}(\mathbf{I} - \mathbf{Q}\,\mathbf{Q}^*)\mathbf{x} = \mathbf{0}.$$

In line with the Jacobi-Davidson approach, the restriction to a complement of the detected eigenspace is formulated as an orthogonal projection. The eigenvector $\mathbf{x}$ of the deflated system can be computed approximately with Jacobi-Davidson: the $\mathbf{u}$ converges to the new, the $(\ell+1)$st, column for $\mathbf{Q}$ and the $\mathbf{q} \equiv \mathbf{Bu}$ converges to the new column for $\mathbf{Z}$. The "deflated" correction equation that is involved can be written as

$$[\mathbf{Q}, \mathbf{u}]^*\mathbf{z} = 0, \quad (\mathbf{I} - [\mathbf{Z}, \mathbf{q}]\,[\mathbf{Z}, \mathbf{q}]^*)(\mathbf{A} - \vartheta\mathbf{B})(\mathbf{I} - [\mathbf{Q}, \mathbf{u}]\,[\mathbf{Q}, \mathbf{u}]^*)\mathbf{z} = -\mathbf{r}. \tag{3.9}$$

It is easy to include the projections of rank $\ell + 1$ in the preconditioner: simply replace in (1.5), (2.5), and (3.2) the $\mathbf{u}$ and $\mathbf{q}$ by $[\mathbf{Q}, \mathbf{u}]$ and $[\mathbf{Z}, \mathbf{q}]$, respectively. Note that $\mathbf{K}_1^{-1}\mathbf{Q}$ (see (3.2)) will be available from the computation of the first ℓ Schur vectors. After accepting an approximate Schur vector, the current search subspace can be deflated and used as initial search subspace for the next Schur vector.

In practical computations, the dimension of the search subspace and the test subspace can become too large and the Jacobi-Davidson process can be restarted with some appropriate lower dimensional subspace of the current search subspace. The reduced subspace will consist of the most promising eigenvector approximations. The resulting algorithm is called the *JDQZ algorithm*. For more details and an efficient implementation, see [8].

4. Numerical Experiments. In the experiments below, we apply the preconditioning techniques described in the preceding sections in the JDQZ algorithm. We apply this algorithm to compute eigenpairs of discretized convection diffusion problems and linearized Navier-Stokes equations (see §4.2). The preconditioners can be used in iterative solvers such as GMRES [16] for solving the correction equations (1.2) and (3.9) approximately. MRILU is a high quality preconditioner for vectors with large components in the direction of the eigenvectors associated with absolute small eigenvalues (in general, vectors associated with 'smooth' functions) and we expect that the solution of (1.5) already provides a good expansion vector for the search subspace. Therefore, we do not apply an iterative method to solve the correction equation. We simply take the preconditioned residual as expansion vector, using the augmented preconditioner as preconditioner.

The example in §4.3 illustrates how Olsen's approach can suffer from instabilities in case of an excellent preconditioner. Furthermore we present comparisons of the '*K-variant*' from §2.2 and its updated version presented in §3.2, the '*M-variant*' (see §§4.4.2 and 4.6). The results are obtained by a non-optimized MATLAB code, hence timings are unreliable and will not be presented. In part, we can however make use of a FORTRAN code of MRILU for the solution of linear systems in order to estimate the performance of the methods presented here (see §4.4.1). We also discuss effects of the grid size on the convergence and the computational costs (see §§4.5 and 4.6).

In all our examples here we use the JDQZ algorithm to compute the six eigenpairs with eigenvalues with smallest modulus.

The figures that show the convergence history give the $\log_{10}$ of the Euclidean norm of the residual (along the vertical axis) as a function of the iteration number (along the horizontal axis). The huge jumps in the curves mark the detection of eigenpairs: an eigenpair approximation is accepted if the norm of the residual is less than 10^{-12}. Then, in the same iteration step, the search is started for another eigenpair, which entails for the non-small residual at that moment.

4.1. Technicalities. The parameters for the JDQZ algorithm in our experiments were selected as follows.

Initiation. In the initiation phase of JDQZ, we specify an initial search subspace and (six) values $\widetilde{\tau}$: among the remaining eigenvalues, the jth eigenvalue to be computed should be closest to the jth $\widetilde{\tau}$. The selection of the $\widetilde{\tau}$ and of the initial search subspace depends on the preconditioner to be used.

For the K-variant, we take all $\widetilde{\tau}$ equal to 0. The initial search subspace is one-dimensional and spanned by a random vector.

If we use the M-variant, then we have the ingredients that allow efficient computation of good approximations to the smallest eigenvalues and associated eigenvectors (cf. §3.3). For this variant, we compute the (six) pre-eigenvalues (cf. §3.3) that are smallest in modulus. The associated pre-eigenvectors are computed as well and form the initial search subspace. For $\widetilde{\tau}$, we successively take the pre-eigenvalues in increasing magnitude.

Targets. We search for the eigenvalue nearest to a target τ. The value for τ is reset after each detection of an eigenvalue and is determined as follows. The generalized eigenvalue problem is projected on the current search subspace and the eigenvalue of the projected problem that is closest to the next $\widetilde{\tau}$ is taken as the new value for τ, where $\widetilde{\tau}$ is as selected in the initiation phase (see the previous paragraph). The new τ can be viewed as the best approximation of the next wanted eigenvalue that can be computed from the available data. For the first value of τ we take the first $\widetilde{\tau}$.

Preconditioners. $\mathbf{K}$ is constructed for $\mathbf{A}$ (cf. §2.2). The preconditioner $\mathbf{M}$ is constructed for $\mathbf{A} - \tau\mathbf{B}$ (cf. §3.2) and is updated whenever a new value for τ is selected. Note that $\mathbf{M}$ is updated only after detection of an eigenvalue and not in each step of JDQZ (when a new approximation ϑ for the wanted eigenvalue is available).

Restarts. The dimension of the search and test subspace increases with one in each iteration step of the JDQZ algorithm. If dimension 11 is reached a restart is performed reducing the dimension to 6.

Test subspace. We follow the harmonic Petrov-Galerkin approach to construct the test subspace (cf. §2.3) with the value τ computed as explained above, that is, if $\mathbf{v}$ is the expansion vector for the search space then the test space is expanded by $(\mathbf{A} - \tau\mathbf{B})\mathbf{v}$.

The approximate eigenpairs resulting from testing against this space (cf. §1) are called *harmonic* Petrov pairs. In case of a standard symmetric eigenvalue problem, the harmonic Petrov vector associated with the harmonic Petrov value closest to the target τ is close (in angle) to the wanted eigenvector, provided that the residual is relatively small. With other test subspaces, as in the Ritz-Galerkin approach (where the test subspace is equal to the search subspace), this need not be the case. Therefore, harmonic Petrov values can be used safely for selecting the most promising approximate eigenpairs. Experiments in [8] suggest that the same conclusion holds also for more general eigenvalue problems. The concerning residual should be small relative to the distance of τ to the nearby wanted eigenvalue λ. Our way of determining the value for τ yields τ for which $|\lambda - \tau|$ is small and misselection in the first few steps of JDQZ may result from this otherwise desirable situation. In [21] a simple but efficient strategy is given to circumvent this type of misselection and we follow this strategy here; for details see [21].

Stopping criterion. We accept an approximate eigenpair if the Euclidean norm of the associated residual is less than 10^{-12}. Then also $|\lambda - \vartheta| \leq 10^{-12}$.

4.2. Test problems. We will concentrate on computing eigenvalues of a simple convection-diffusion operator, and, as an example of a generalized eigenvalue problem, we will compute eigenvalues that are relevant in the stability analysis of a solution of the Navier-Stokes equation.

The convection-diffusion eigenvalue problem is defined on the unit square and given by

$$\Delta u + c(u_x + u_y) = \lambda u \tag{4.1}$$

with $u(0, y) = u(x, 0) = 0$ and $u_x(1, y) = u_y(x, 1) = 0$ on the boundaries. c is a constant to be specified below. It is discretized on a uniform mesh with central differences leading to the eigenvalue problem

$$\mathbf{A}\mathbf{x} = \lambda\mathbf{x}. \tag{4.2}$$

The boundary conditions are incorporated in the discretized operator (matrix) $\mathbf{A}$ of the differential operator in the left hand-side of (4.1).

4.3. Olsen versus Augmented. In §1 we noted the potential danger of solving the preconditioned correction equation (1.5) with the strategy of (1.4) (Olson's approach) in case of a high quality preconditioner $\mathbf{M}$. We argued that instabilities could then be avoided with representation (2.5) and the strategy of §3.1 (augmented approach). This is illustrated in FIG. 4.1, where Olson's approach is depicted in the left figure and the augmented one in the right figure. In this example, problem (4.1) is solved for $c = 1$ on a uniform grid of 32 by 32 unknowns, but the phenomenon is not typical for this situation only.

The initial slow convergence in Olson's approach is caused by the fact that the first pre-eigenvalue is used as first target which renders $\mathbf{M}$ singular to machine precision. Initially the method has severe difficulties to get relevant information in the search space. Nevertheless, the search space expands (though only through noise) and relevant information increases slowly during the first 80 iterations. This, in turn, leads to a right-hand side nearly orthogonal to the wanted eigenvector, which is close (in angle) to the right singular vector of $\mathbf{M}$. Since $\mathbf{M}$ is symmetric, left and right singular vector coincide and the right-hand side will have a large angle with the left singular vector: the right-hand side is now better 'compatible' with the near-singularity of $\mathbf{M}$. The MRILU approach pushes the ill-conditioning to a low-dimensional diagonal block to which a robust (direct) solution method for dense linear

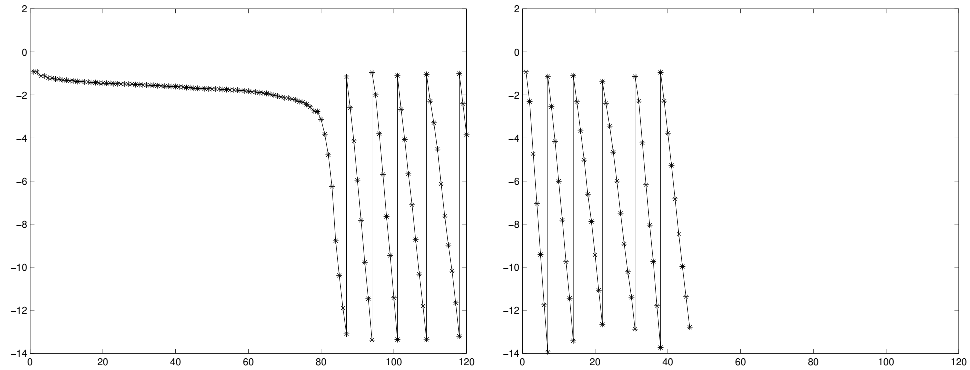

FIG. 4.1. *For both figures, the expansion vector for the search subspace is obtained from the preconditioned correction equation* (1.5) *with an updated MRILU preconditioner (the M-variant). In the left figure, the preconditioned correction equation* (1.5) *is solved as indicated in* (1.4) *(Olsen's approach), whereas in the right figure the equation solved as explained in* §§3.1 *and* 3.2 *Failure of Olsen approach*

systems is applied. Such dense method can handle the situation where the left hand-side vector is (nearly) orthogonal to the left singular vector (producing least norm solutions). Hence, the effects of ill-conditioning diminish, and the speed of convergence increases.

In the augmented approach (right figure), the ill-conditioning of the right bottom diagonal block is annihilated by the bordering of a vector that is close to the singular vector. Now, we have fast convergence from the beginning.

As mentioned before, in this example, the convection coefficient is $c = 1$. The situation aggravates with increasing c (results not shown here). The situation is less dramatic if the shift τ for $\mathbf{M}$ is selected less close to the pre-eigenvalue, but, of course, one wishes to exploit the best approximations that are available.

In Olsen's approach, two systems have to be solved (see (1.4)). For the augmented variant, the update of the borders in (3.2) costs about the same as solving one system. The costs for solving a system involving (2.2) are comparable to the costs for solving its augmented version (involving (3.2)). Therefore, the computational costs for the augmented approach and Olsen's are the same. Hence the augmented approach is to be preferred and this is the one that is followed in our other experiments below.

4.4. The effect of updating the MRILU preconditioner. We are interested in the effect of updating the MRILU preconditioner in JDQZ: the K-variant versus the M-variant. We use JDQZ to compute eigenpairs of discretized versions of the convection-diffusion problem (4.1) with $c = 0.1$ on a uniform square grids (see §4.4.2). First we make some observations on the computational costs of the separate steps.

4.4.1. Cost considerations. When writing the paper, we only have a MATLAB code for our eigenvalue experiments and timings are unreliable. However, the computational costly ingredients for preconditioned iterative solution methods as Bi-CGSTAB for linear systems [23] and JDQZ for eigenvalue problems are comparable if the matrices involved are the same. This fact can be exploited to get an indication of the performance of JDQZ by using timings from a FORTRAN code in which MRILU is used as preconditioner for Bi-CGSTAB.

In JDQZ as well as in Bi-CGSTAB, a preprocessing phase where the preconditioner is constructed (i.e., the reordering and factors for $\mathbf{K}$ as well as, in case of JDQZ, the additional ingredients for $\mathbf{M}$) can be distinguished from the iteration phase in which the problem is actually solved. In both methods, each iteration step requires high dimensional operations

TABLE 4.1
Timing for Bi-CGSTAB with MRILU preconditioner for a problem with 256^2 unknowns.

fill	factorization time	solution time	time per iteration	flops per iteration
13	8.4	13.5	0.82	74
18	16.0	10.9	0.95	93
29	47.2	9.2	1.2	135
35	78.5	8.9	1.4	160

such as matrix-vector multiplications (MVs), solves with the preconditioner (solves), vector updates (AXPYs), and inner products (DOTs), plus some low dimensional operations. It is realistic to assume that the computational work for the low dimensional actions is negligible with respect to the work for the high dimensional ones. The number of iterations steps is code-independent and the number of AXPYs and DOTs per step can be counted. With this information, the performance of a FORTRAN code of JDQZ can be predicted from timings for a FORTRAN code of Bi-CGSTAB. Therefore, we first discuss the performance of Bi-CGSTAB in a relevant setting. Then we interpret the results for JDQZ.

MRILU timings. We discretize (4.1) with $c = 0.1$ on a uniform grid of 256^2 unknowns and apply preconditioned Bi-CGSTAB to solve $\mathbf{A}\mathbf{x} = \mathbf{b}$ with $\mathbf{A}$ as in (4.2) and $\mathbf{b}$ some non-trivial vector. The timings, to be shown, are virtually independent of $\mathbf{b}$. The Bi-CGSTAB iteration with initial guess $\mathbf{0}$ is stopped at a reduction of the norm of the preconditioned residual by 10^{-15}.

Table 4.1 shows the time needed to construct the preconditioner (factorization time, second column) and the time spent in the iteration phase (solution time, third column). The first column shows the fill, that is, the average number of non-zeros in the rows of the factors of the preconditioner. Recall that for MRILU the fill is not to be specified but a drop (or lump) tolerance ε that controls the fill (cf. §2.2). Bi-CGSTAB requires two MVs and two solves per iteration step; see the fourth column for the time per iteration step. The fifth column shows the number of flops per iterations step divided by the number of unknowns.

Note that the factorization time rapidly increases with increasing fill. The time per iteration step increases more slowly. With more fill, a 'better' approximation of $\mathbf{A}$ can be anticipated, leading to a reduction in the number of iteration steps of Bi-CGSTAB. The solution time may decrease, and from the table we see that it does. The decrease here depends less sensitively on the fill than on the factorization time. Note that more fill pays itself back if more linear systems with the same matrix are to be solved. Recall that we want to compute 6 eigenpairs with JDQZ.

Preprocessing phase. The costs for the factorization should not dominate the entire computation. This was not case in our experiments, where JDQZ needs 50 to 100 iterations. Each iteration step requires two solves and two MVs as for Bi-CGSTAB (although one of the MVs in a JDQZ step is by the matrix $\mathbf{B}$. In our applications, $\mathbf{B}$ is much sparser than $\mathbf{A}$). The other costs per steps (DOTs and AXPYs) are much higher for JDQZ (see the paragraph below on 'JDQZ iteration').

For the computation of the update $\widehat{\mathbf{B}}$, that is, the matrix in (3.6), we can only give an indication. In our examples, the fill of $\widehat{\mathbf{B}}$ is only a fraction of that of $\mathbf{K}$. Since the time for the factorization is related to the amount of fill, the time to build $\widehat{\mathbf{B}}$ will only be a fraction of that of $\mathbf{K}$ (in MATLAB this was indeed the case). Moreover, $\widehat{\mathbf{B}}$ uses the reordering and partioning of $\mathbf{K}$.

TABLE 4.2
Operations and approximate flop count for one iteration step of JDQZ (fill to be counted only for the M-variant).*

Operation	**flops per unknown**
Solve	$4(\text{fill}(\mathbf{K}) + \text{fill}^*(\widehat{\mathbf{B}}))$
Expand search and test subspace:	
orthogonalize search subspace	$4(\dim(\mathbf{Q}) + \dim(\mathbf{V}))$
multiply by $\mathbf{A}$ and $\mathbf{B}$	$2(\text{fill}(\mathbf{A}) + \text{fill}(\mathbf{B}))$
orthogonalize test subspace	$4(\dim(\mathbf{Q}) + \dim(\mathbf{V}))$
Expand projected eigenvalue problem	4
Solve projected eigenvalue problem	0
Compute approximate eigenpair	$6\dim(\mathbf{V}) + \dim(\mathbf{Q})$

Iteration phase. The number of flops per unknown for one solve with the factorization $\mathbf{K}$ is twice the amount of fill (multiplication and addition counted separatedly). For the M-variant, twice the fill of $\widehat{\mathbf{B}}$ has to be added.

JDQZ iteration. In Table 4.2 the flop count per unknown per iteration step of JDQZ is shown. In our calculation $\dim(\mathbf{Q}) \approx 3$ and $\dim(\mathbf{V}) \approx 8$. Moreover, for the convection-diffusion problem, $\text{fill}(\mathbf{A}) = 5$ and $\text{fill}(\mathbf{B})$ is 1 (of course, in this example, $\mathbf{B} = \mathbf{I}$. So, there is no need to multiply by $\mathbf{B}$). Hence, apart from the 'solve' part, each step requires already about 160 flops per unknown. This is substantial. The double application of the factorization with the highest fill that we will apply (see Table 4.1) costs a similar amount of flops (one should keep in mind that for the latter much indirect addressing is used). Hence, also from this point of view, it is worthwhile to keep the number of iterations low.

4.4.2. The K-variant versus the M-variant. We illustrate the effect of updating the preconditioner for problem (4.1) with $c = 0.1$ on a uniform grid of 32 by 32 unknowns: the matrices have size $n = 1024$.

We apply the MRILU preconditioner $\mathbf{K}$ with fill 30 (high fill) and with fill 19 (moderate fill). The corresponding fill in the update $\widehat{\mathbf{B}}$ is 8 and 5, respectively. FIG. 4.2 displays the convergence history of JDQZ with MRILU for the K-variant (top figures) and for the M-variant (bottom figures) for high fill (left figures) and for moderate fill (right figures).

We see that the M-variant leads to a significant reduction of the number of iteration steps. In case of moderate fill (the left figures), the K-variant requires 72 iterations, whereas the M-variant detects the wanted eigenpairs in 45 iterations. Therefore, the 'solve' part in the iteration phase of JDQZ needs $72 \cdot 4 \cdot 30 = 8640$ flops per unknown for the K-variant (with a factorization with fill 30) and $45 \cdot 4 \cdot (30 + 8) = 6840$ flops per unknown for the M-variant (with fill $30 + 8$). This already shows a gain for the M-variant of about 21%. The lower number of iteration steps leads to an even higher gain, since the substantial costs for the other operations in JDQZ (see Table 4.2) should also be taken into account.
For low fill, the gain is less (17% in the 'solve' part): 81 iterations with fill 19 ($81 \cdot 4 \cdot 19 = 6156$ flops per unknown) and 61 iterations with fill $19 + 5$ ($61 \cdot 4 \cdot 24 = 5124$ flops per unknown).

The gain in the M-variant cannot be explained only from the fact that this variant uses a better initial search subspace than the K-variant. The figures, but also the target values (see the discussion below), show that the effect of a better initial search subspace diminishes after a few steps. Inspection of the slopes in the convergence histories reveals that the speed of convergence of JDQZ for the M-variant is higher than for the K-variant, so, updating

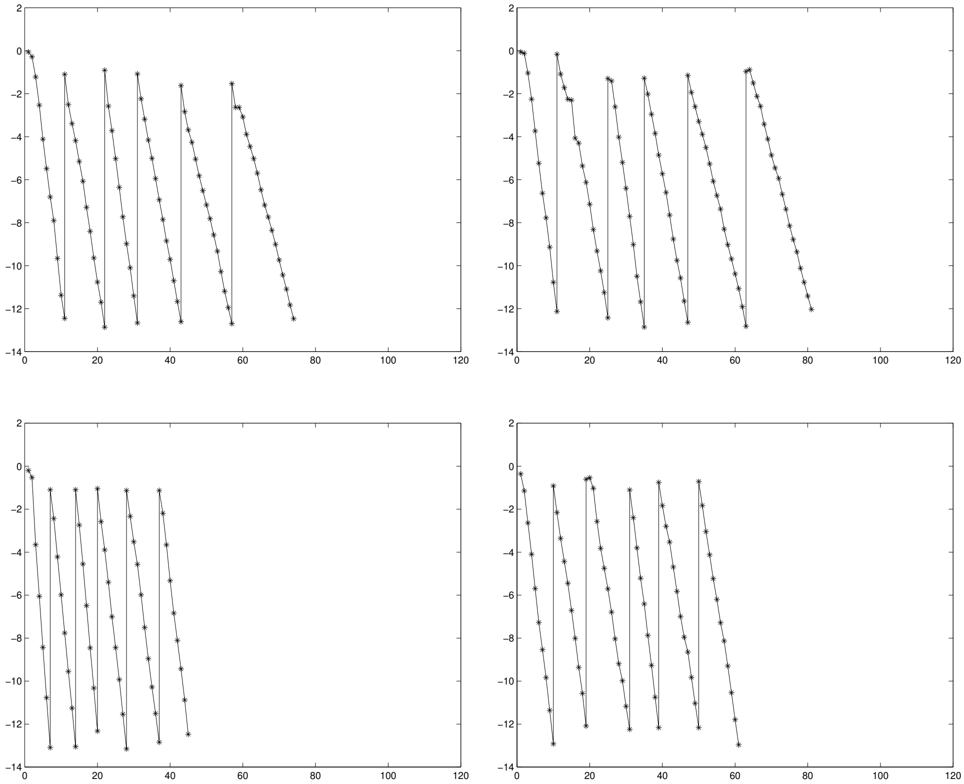

FIG. 4.2. *Convergence history of JDQZ with MRILU preconditioner for the K-variant (top figures) and M-variant (bottom figures) with high fill (left figures) and moderate fill (right figures). The convection-diffusion equation* (4.1) *is solved for* $c = 0.1$ *on a grid of* $32^2 = 1024$ *unknowns.*

the MRILU preconditioner $\mathbf{K}$ improves the quality of the preconditioner. The increasing computational costs per solve are compensated by this improvement.

For both variants, we obtain a fairly high speed of convergence and there seems to be no need to obtain a more accurate solution of the correction equations with an iterative linear solver (such as GMRES; cf. the introduction of this §4).

From the figures in FIG. 4.2, we see that the speed of convergence for the K-variant for the first eigenvalue is quite high already after a few steps. For the M-variant, pre-eigenpairs are available and form a better start (see Table 4.3)), but the M-variant does not seem to profit from a better start. This is not surprising since, as can be seen from Table 4.3, zero is also a reasonable guess for the first eigenvalue in this example. Hence, also for the K-variant, relevant information is added to the search subspace straight from the beginning. We will see another example (in §4.6), were the K-variant needs more steps to reach the phase were JDQZ converges rapidly (even quadratically if the correction equations would have been solved exactly from this step on; see §1). In such cases, the M-variant profits more from the better start with pre-eigenpairs.

In Table 4.3 the pre-eigenvalues and targets are listed for the case of moderate fill. The values for the case of low fill are similar.
We see that the pre-eigenvalues are quite accurate. As expected (see §3.3), the difference with the exact eigenvalue increases with the magnitude.
The target value for the last eigenvalue in the K-variant is close to the seventh eigenvalue

TABLE 4.3
Pre-eigenvalues, targets and computed eigenvalues for the convection-diffusion problem.

	K-variant					
target	0	25.0	25.1	44.9	64.1	83.8
computed eigenvalue	5.14	24.8	24.8	44.5	64.1	83.8
	M-variant					
pre-eigenvalue	5.13	25.3	25.6	46.6	67.9	69.0
target	5.13	25.2	25.0	44.9	65.0	64.5
computed eigenvalue	5.14	24.8	24.8	44.5	64.1	64.1

and the method converges to this 'unwanted' eigenvalue. Convergence to an unwanted eigenvalue (or detection of the eigenvalues in non-increasing order) may happen for any iterative eigenproblem solver. Here, misselection of the target causes the problem. The danger of misselection can be reduced by increasing the minimal dimension of the search subspace (cf. 'Restarts' in §4.1).
JDQZ tends to produce initial search subspaces for the second and following eigenvalues containing good approximations to the corresponding eigenvectors (see [8]). The approximations tend to be better if more steps are needed for detecting the preceding eigenpair(s). This explains why a better start for the M-variant with pre-eigenpairs is not always reflected in better targets for the second and following eigenvalues (cf. Table 4.3).

From these results we conclude that updating the MRILU preconditioner improves the performance and that an accurate factorization (higher fill) is helpful.

4.5. Grid-independence. The computational costs per unknown for solving discretized convection diffusion problems with preconditioned Bi-CGSTAB using MRILU is almost independent of the mesh size [4]. It is of interest to know how JDQZ with MRILU behaves for larger problems.

FIG. 4.3 shows the results for the M-variant on a grid of 64^2 unknowns (the left figure) and on a grid of 128^2 unknowns (the right figure) with fills of $34 + 9$ and $37 + 10$, respectively. The result on 32^2 unknowns with a fill of $30 + 8$ displayed before, at the bottom left figure in FIG. 4.2, also fits in this sequence. Recall that in MRILU a drop/lump parameter ε controls the fill. We used here the same value for ε for all grid sizes. The fills appear to be comparable, and so are the computational costs per unknown per iterations step, as well as the costs per unknown for the factorization. Therefore, the computational costs per unknown for solving the eigenvalue problem are nearly independent of the grid-size if the number of JDQZ iterations steps is nearly independent of the grid-size.

We see that the first refinement, with $n = 64^2$, shows only a very modest increase in iteration steps and fill compared to the case $n = 32^2$. In the right plot we see one aberration, although most eigenvalues converge as expected. This is again due to a misselection of the target: the targets for both the third and fourth eigenvalue are close to the fourth. Hence, the third eigenvector is not yet deflated from the problem in the search for the fourth and hampers its convergence. The pre-eigenvalues are accurate in this case. Hence, apart from the aberration, the convergence here is nearly independent of the mesh size.

4.6. A generalized eigenvalue problem. We will now show a result for a generalized eigenvalue problem. The eigenvalues to be calculated are of interest in a stability analysis of some stationary solution of the incompressible Navier-Stokes equations.

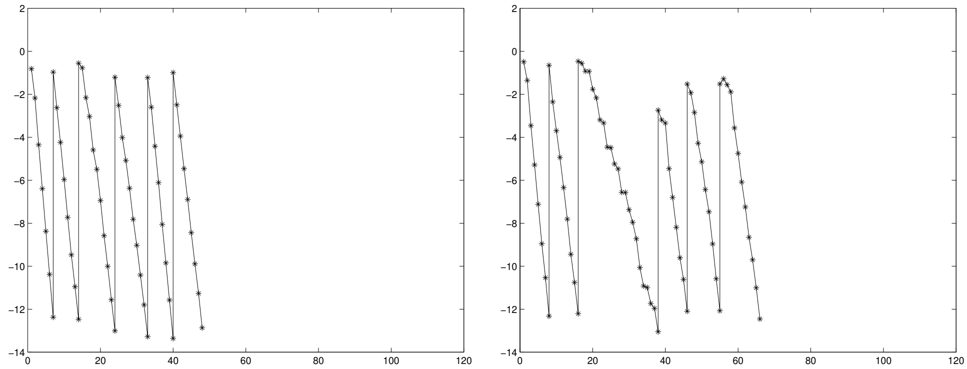

FIG. 4.3. *These figures illustrate the effect of grid refinement on convergence. The convection-diffusion equation* (4.1) *is solved for* $c = 0.1$ *on a grid of* $64^2 = 4096$ *unknows (the left figure) and on a grid of* $128^2 = 16384$ *unknows (the right figure). In both cases, the M-variant is used with approximately the same fill (moderate fill). The bottom left figure in* FIG. 4.2 *fits in this sequence.*

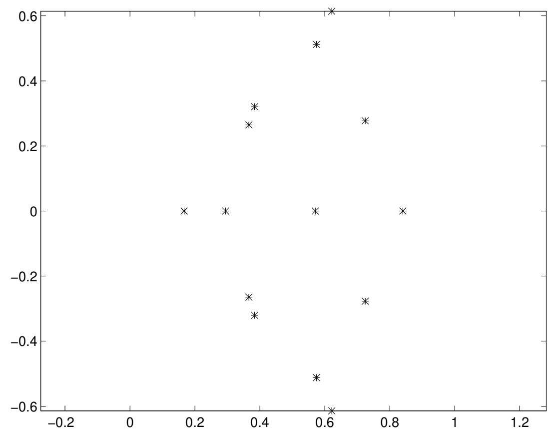

FIG. 4.4. *A relevant part of the spectrum of a Jacobian system for Navier-Stokes.*

The Navier-Stokes equations were discretized with finite differences on a staggered grid. The stability analysis leads to a generalized eigenvalue problem involving a Jacobian system. For an impression of the relevant part of the spectrum, see FIG. 4.4, where the smallest eigenvalues in modulus are displayed in a case of 675 unknowns. In the computation, we tried to find the eigenvalues closest to the origin.

The Navier-Stokes equations consist of three coupled PDEs. Therefore, it is convenient to work in the factorization with block diagonals with blocks of size 3 by 3 rather than with diagonal matrices (cf. §2.2). The Jacobian system leads to the matrix $\mathbf{A}$. Since we consider incompressible Navier-Stokes equations, one of the PDEs does not contain time-derivatives. This leads to a non-standard eigenvalue problem: $\mathbf{B}$ is diagonal with diagonal-entries $\mathbf{B}_{ii} = 0$ if $i = 0 \bmod 3$ and 1 else. The matrices are real and the eigenvalues appear in conjugate pairs. This is exploited in the algorithm: a jump in the convergence history may mark detection of one real eigenvalue, but also one conjugate pair. In FIG. 4.5, results for the K-variant (left figure) and for the M-variant (right figure) are shown in the case of 675 unknowns. The fill is here 44 for preconditioner $\mathbf{K}$ and 16 for the update $\widehat{\mathbf{B}}$.

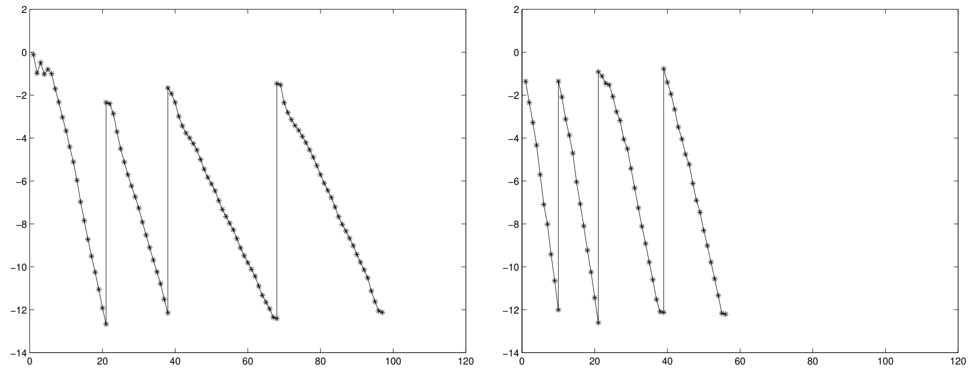

FIG. 4.5. *Convergence history of JDQZ for the computation of the six smallest eigenvalues of the Jacobian system of Navier-Stokes. The matrices are* 675 *by* 675*. For the left figure, the K-variant is used, for the right figure, the M-variant.*

TABLE 4.4
Pre-eigenvalues, targets and computed eigenvalues for the Jacobian of the Navier-Stokes equations. All values should be multiplied by 0.1.

	K-variant					
target	0	2.94	3.68+2.62i	3.66-2.65i	3.80-3.12i	3.84+3.21i
computed eigenvalue	1.67	2.94	3.66+2.65i	3.66-2.65i	3.84-3.21i	3.84+3.21i
	M-variant					
pre-eigenvalue	1.84	3.40	4.11-4.08i	4.11+4.08i	4.78+3.63i	4.78+3.63i
target	1.84	3.09	3.49-2.60i	3.66-2.65i	4.10+3.49i	3.84+3.21i
computed eigenvalue	1.67	2.94	3.66−2.65i	3.66+2.65i	3.84+3.21i	3.84-3.21i

As for the convection-diffusion problem, we see that the use of the update improves the performance. The speed of convergence for the M-variant is higher than for the K-variant and forms the main reason for the better performance. However, here, the better start for the M-variant with a space of pre-eigenvectors also pays off. Although the initial guess for the M-variant is only slightly better (by one digit) than for the K-variant, the M-variant brings JDQZ in the fast converging phase right from the beginning, whereas the K-variant needs seven steps to achieve this. Note that this effect on the performance would have been relatively much more significant if the required tolerance would have been larger (say 10^{-5} instead of the 10^{-12} as it is now).

In Table 4.4 the targets and computed eigenvalues are shown. The pre-eigenvalues give a good approximation of the exact eigenvalues, although they are less accurate than in the convection-diffusion case.

We also did experiments on finer grids. The results do not affect the conclusion for the M-variant versus the K-variant, but, as it is now, we may not state that the results are independent of the size of the grid. This asks for an improvement of MRILU, which is a subject of current research.

5. Conclusions. Multilevel ILU-type of preconditioners as MRILU can be exploited for the efficient computation of the absolute smallest eigenvalues and associated eigenvectors. Residuals that are properly preconditioned by MRILU form effective expansion vectors for the search subspaces in the Jacobi-Davidson algorithm: there is no need to employ an iterative linear solver such as GMRES to obtain more accurate solutions of the correction equation. In general, only one incomplete factorization will suffice to compute a range of eigenvalues with associated eigenvectors. The dimension of the search subspace can be kept low, and solving an eigenvalue problem for the absolute smallest eigenvalues is not more costly than solving a linear system of equations with the same matrix. The factorization can be efficiently updated to accommodate better eigenvalue approximations whenever they become available during the computational process. Updated factorizations enhance the performance of the Jacobi-Davidson process. The preconditioner can be implemented in a stable fashion; but, specifically, the implementation of the updated preconditioner needs some care. The updated factorization can be used efficiently (i.e., at the cost of the solve of one preconditioner equation) to find accurate initial eigenvalue and eigenvector approximations. These accurate initial approximations put the process in the 'quadratic converging phase' straight from the beginning.

For convection diffusion type of problems, the MRILU approach seems to lead to a speed of convergence that is independent of the grid-size. For more complicated eigenvalue problems, such as problems associated with the stability analysis of Navier-Stokes equations, the approach is effective, but a speed of convergence independent of the grid-size has not been achieved yet.

REFERENCES

[1] E. L. ALLGOWER AND H. SCHWETLICK, *A general view of minimally extended systems for simple bifurcation points*, Z. Angew. Math. Mech., 77 (1997), pp. 83–97.

[2] R. E. BANK AND C. WAGNER, *Multilevel ILU decomposition*, Numerische Mathemathik, (2000).

[3] G. BEYLKIN AND N. COULT, *A multiresolution strategy for reduction of elliptic PDEs and eigenvalue problems*, Appl. Comput. Harmon. Anal., 5 (1998), pp. 129–155.

[4] E. F. F. BOTTA AND F. W. WUBS, *Matrix renumbering ILU: An effective algebraic multilevel ILU preconditioner for sparse matrices*, SIAM J. Matrix Anal. Appl., 20 (1999), pp. 1007–1026.

[5] T. F. CHAN, *Deflation techniques and block-elimination algorithms for solving bordered singular systems*, SIAM J. Sci. Statist. Comput., 5 (1984), pp. 121–134.

[6] M. CROUZEIX, B. PHILIPPE, AND M. SADKANE, *The Davidson method*, SIAM J. Sci. Comput., 15 (1994), pp. 62–76.

[7] E. R. DAVIDSON, *The iterative calculation of a few of the lowest eigenvalues and corresponding eigenvectors of large real-symmetric matrices*, J. Computational Phys., 17 (1975), pp. 87–94.

[8] D. R. FOKKEMA, G. L. G. SLEIJPEN, AND H. A. VAN DER VORST, *Jacobi-Davidson style QR and QZ algorithms for the reduction of matrix pencils*, SIAM J. Sci. Comput., 20 (1999), pp. 94–125 (electronic).

[9] G. H. GOLUB AND C. F. VAN LOAN, *Matrix Computations*, The John Hopkins University Press, Baltimore and London, 2nd ed., 1989.

[10] R. B. MORGAN, *Generalizations of Davidson's method for computing eigenvalues of large nonsymmetric matrices*, J. Comput. Phys., 101 (1992), pp. 287–291.

[11] R. B. MORGAN AND D. S. SCOTT, *Preconditioning the Lanczos algorithm for sparse symmetric eigenvalue problems*, SIAM J. Sci. Comput., 14 (1993), pp. 585–593.

[12] J. OLSEN, P. JØRGENSEN, AND J. SIMONS, *Passing the one-billion limit in full configuration-interaction (FCI) calculations*, Chemical Physics Letters, 169(6) (1990), pp. 463–472.

[13] A. REUSKEN, *A multigrid method based on incomplete Gaussian elimination*, Numer. Linear Algebra Appl., 3 (1996), pp. 369–390.

[14] Y. SAAD, *Numerical methods for large eigenvalue problems*, Manchester University Press, Manchester, 1992.

[15] Y. SAAD, *ILUM: a multi-elimination ILU preconditioner for general sparse matrices*, SIAM J. Sci. Comput., 17 (1996), pp. 830–847.

[16] Y. SAAD AND M. H. SCHULTZ, *GMRES: a generalized minimal residual algorithm for solving nonsymmetric linear systems*, SIAM J. Sci. Statist. Comput., 7 (1986), pp. 856–869.

[17] H. SCHWETLICK AND R. LÖSCHE, *A generalized inverse iteration for computing simple eigenvalues of nonsymmetric matrices*, Tech. Report IOKOMO-07-97, Technische Universität Dresden, July, 1997.

[18] G. L. G. SLEIJPEN, A. G. L. BOOTEN, D. R. FOKKEMA, AND H. A. VAN DER VORST, *Jacobi-Davidson type methods for generalized eigenproblems and polynomial eigenproblems*, BIT, 36 (1996), pp. 595–633. International Linear Algebra Year (Toulouse, 1995).

[19] G. L. G. SLEIJPEN AND H. A. VAN DER VORST, *The Jacobi-Davidson method for eigenvalue problems and*

its relation to accelerated inexact Newton schemes, in Iterative methods in Linear Algebra, II. Proceedings of the Second IMACS International Symposium on Iterative Methods in Linear Algebra. June 17-20, 1995, Blagoevgrad, S. Margenov and P. Vassilevski, eds., vol. 3 of IMACS Series in Computational and Applied Mathematics, World Scientific Publishing Co. Inc., 1995.

[20] G. L. G. SLEIJPEN AND H. A. VAN DER VORST, *A Jacobi-Davidson iteration method for linear eigenvalue problems*, SIAM J. Matrix Anal. Appl., 17 (1996), pp. 401–425.

[21] G. L. G. SLEIJPEN, H. A. VAN DER VORST, AND E. MEIJERINK, *Efficient expansion of subspaces in the Jacobi-Davidson method for standard and generalized eigenproblems*, Electron. Trans. Numer. Anal., 7 (1998), pp. 75–89 (electronic). Large scale eigenvalue problems (Argonne, IL, 1997).

[22] A. VAN DER PLOEG, E. F. F. BOTTA, AND F. W. WUBS, *Nested grids ILU-decomposition (NGILU)*, J. Comput. Appl. Math., 66 (1996), pp. 515–526.

[23] H. A. VAN DER VORST, *Bi-CGSTAB: a fast and smoothly converging variant of Bi-CG for the solution of nonsymmetric linear systems*, SIAM J. Sci. Statist. Comput., 13 (1992), pp. 631–644.

[24] K. WU, Y. SAAD, AND A. STATHOPOULOS, *Inexact Newton preconditioning techniques for large symmetric eigenvalue problems*, Electron. Trans. Numer. Anal., 7 (1998), pp. 202–214 (electronic). Large scale eigenvalue problems (Argonne, IL, 1997).