A JACOBI–DAVIDSON METHOD FOR SOLVING COMPLEX-SYMMETRIC EIGENVALUE PROBLEMS

PETER ARBENZ* AND MICHIEL E. HOCHSTENBACH†

Abstract. We discuss variants of the Jacobi–Davidson method for solving the generalized complex-symmetric eigenvalue problem. The Jacobi–Davidson algorithm can be considered as an accelerated inexact Rayleigh quotient iteration. We show that it is appropriate to replace the Euclidean inner product $\mathbf{x}^*\mathbf{y}$ in $\mathbb{C}^n$ by the bilinear form $\mathbf{x}^T\mathbf{y}$. The Rayleigh quotient based on this bilinear form leads to an asymptotically cubically convergent Rayleigh quotient iteration. Advantages of the method are illustrated by numerical examples. We deal with problems from electromagnetics that require the computation of interior eigenvalues.

Key words. Generalized complex-symmetric eigenvalue problem, interior eigenvalues, Jacobi–Davidson algorithm.

AMS subject classifications. 65F15, 65F50.

1. Introduction. In this paper we consider variants of the Jacobi–Davidson (JD) algorithm [22] for computing a few eigenpairs of

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

where the large and sparse matrix A is *complex-symmetric*: $A = A^T \in \mathbb{C}^{n\times n}$. Eigenvalue problems of this type, and of the related generalized complex-symmetric eigenvalue problem

$$A\mathbf{x} = \lambda B\mathbf{x}, \qquad B \text{ invertible}, \tag{1.2}$$

where both A and B are complex-symmetric are becoming of increasing importance in applications, most notably in the field of electro-magnetic simulations. High quality particle accelerators can be modeled by the time-independent Maxwell equations, assuming perfectly conducting cavity walls. This approach leads to a generalized real-symmetric eigenvalue problem [2]. However, in cases where the damping of higher modes is more important than the high efficiency of a cavity, and for cavities with ferrite inserts for tuning purposes, the currents produced in the walls or in the ferrite lead to a damping of the eigenmodes. In this situation these surfaces are treated as lossy material which introduces a complex permittivity which in turn leads to complex-symmetric matrices in (1.1) or (1.2).

Open cavities are often modeled on bounded domains. Lossy perfectly matched layers (PMLs) along the boundary are introduced to prevent reflection of waves. PMLs, also called absorbing boundary conditions, are again modeled by complex permittivities [27]. The PML scheme has the potential to extend the range of applications for these eigenvalue solvers to the wide field of the design of antennas.

Notice that complex-symmetric matrices are not Hermitian. So, they do not possess the favorable properties of Hermitian matrices. In particular, complex-symmetric

*Swiss Federal Institute of Technology (ETH), Institute of Scientific Computing, CH-8092 Zurich, Switzerland (arbenz@inf.ethz.ch). Some of the work of this author was done while visiting the Department of Mathematics at Utrecht University.

†Utrecht University, Department of Mathematics, NL-3508 TA Utrecht, The Netherlands (hochstenbach@math.uu.nl)

matrices may have complex eigenvalues, and can be arbitrarily nonnormal. In fact, *every* matrix is similar to a complex-symmetric matrix [10, 14], whence it may be arbitrarily difficult to solve (1.1) or (1.2), respectively. Nevertheless, complex-symmetric matrices do have special properties. If $\mathbf{x}$ is a right eigenvector of A, $A\mathbf{x} = \lambda\mathbf{x}$, then it is also a left eigenvector, in the sense that $\mathbf{x}^T A = \lambda\mathbf{x}^T$. Eigenvectors $\mathbf{x}, \mathbf{y}$ corresponding to different eigenvalues $\lambda \neq \mu$ are *complex-orthogonal*, i.e., they satisfy

$$(\mathbf{x}, \mathbf{y})_T := \mathbf{y}^T\mathbf{x} = 0. \tag{1.3}$$

If A is diagonalizable then the diagonalization can be realized by a complex-orthogonal matrix Q, $Q^TQ = I$ [14].

We call the (indefinite) bilinear form $(\mathbf{x}, \mathbf{y})_T$ in (1.3)—somewhat abusively—an *"inner product"*. For brevity, we write $\mathbf{x} \perp_T \mathbf{y}$ if two vectors $\mathbf{x}$ and $\mathbf{y}$ are complex-orthogonal. A vector $\mathbf{x}$ is called *quasi-null* if $(\mathbf{x}, \mathbf{x})_T = 0$.

When treating the generalized eigenvalue problem (1.2) it is natural to use the indefinite bilinear form

$$[\mathbf{x}, \mathbf{y}]_T := (\mathbf{x}, B\mathbf{y})_T = \mathbf{y}^T B\mathbf{x}. \tag{1.4}$$

The matrix $B^{-1}A$ is then complex-symmetric with respect to $[\mathbf{x}, \mathbf{y}]_T$ as A is complex-symmetric with respect to $(\mathbf{x}, \mathbf{y})_T$. We therefore restrict ourselves to the special eigenvalue problem (1.1) whenever there is no loss in generality. The numerical examples that we will discuss later are all generalized eigenvalue problems of the form (1.2).

A number of algorithms have been designed for solving complex-symmetric linear systems of equations. Van der Vorst and Melissen [28] modified the bi-conjugate gradient algorithm to obtain the complex conjugate gradient algorithm COCG. The crucial idea is to set the initial shadow vector equal to the initial residual. (If one works with the Euclidean inner product, the shadow vector has to be the complex conjugate of the initial residual, see [28].) With regard to the relation among right and left eigenvectors mentioned before this choice of the shadow vector is very natural. Freund used the same idea to adapt the quasi-minimal residual (QMR) algorithm to the complex-symmetric case [9]. In COCG and QMR, the same Krylov subspaces are generated. However, the approximate solutions are extracted differently from these subspaces. Recently, Bunse-Gerstner and Stöver [5] introduced an algorithm, CSYM, that is closely related to the special form that the singular value decomposition (or Takagi factorization) takes on for complex-symmetric matrices [14]. Every complex-symmetric matrix is unitarily similar to a complex-symmetric tridiagonal matrix. CSYM constructs the three-term recurrence that holds among the columns of the unitary matrix that realizes the similarity transformation. Notice that CSYM is not a Krylov subspace method.

Algorithms for solving complex-symmetric eigenvalue problems are investigated even less often. Eberlein adapted the classical Jacobi algorithm (for full matrices). Cullum and Willoughby [6, Chapter 6] proposed a Lanczos type eigensolver employing the bilinear form (1.3). The same authors suggested a complex-symmetric tridiagonal QR algorithm [7]. Recently, Luk and Qiao [17] introduced a fast $\mathcal{O}(n^2 \log n)$ eigensolver for complex Hankel matrices, that is based on the works of Cullum and Willoughby and the fast Fourier transform.

In this paper we present a Jacobi–Davidson type algorithm for computing a few eigenpairs of a complex-symmetric matrix that exploit the structure of the matrices. For the original Jacobi–Davidson algorithm see [22], [21], [8]. In contrast to the

complex-symmetric methods mentioned before, our Jacobi–Davidson algorithm can be transcribed quite easily into a solver for the generalized eigenvalue problem (1.2).

The paper is organized as follows. In Section 2 we investigate how the Rayleigh quotient is best defined for dealing with complex-symmetric eigenvalue problems. In Section 3 we adapt a variant of the two-sided Jacobi–Davidson algorithm to this problem and discuss its application to the generalized complex-symmetric eigenvalue problem. The convergence behavior of exact and inexact variants of the Jacobi–Davidson algorithm is investigated in Section 4. Numerical experiments are presented in Section 5. A discussion and some conclusions can be found in Section 6.

2. A Rayleigh quotient for complex-symmetric matrices. Let us first introduce some notations. Throughout the paper, λ denotes a simple eigenvalue of the complex-symmetric $n \times n$ matrix A, with $\mathbf{x}$ its corresponding eigenvector. Since λ is simple, it has a finite condition $\kappa(\lambda)$. Because

$$\infty > \kappa(\lambda) = |\mathbf{x}^T\mathbf{x}|^{-1} = |(\mathbf{x},\mathbf{x})_T|^{-1}, \tag{2.1}$$

an eigenvector corresponding to a simple eigenvalue is not quasi-null whence it can be "normalized" such that $(\mathbf{x},\mathbf{x})_T = 1$ [11, p.323]. Let $\mathbf{u} \approx \mathbf{x}$ be an approximate eigenvector. If $\mathbf{u}$ is close enough to $\mathbf{x}$, then $\mathbf{u}$ is not quasi-null either, and we "normalize" $\mathbf{u}$ such that $(\mathbf{u},\mathbf{u})_T = 1$.

Given $\mathbf{u}$, the corresponding eigenvalue is usually approximated by the Rayleigh quotient

$$\rho = \rho(\mathbf{u}) := \frac{\mathbf{u}^* A\mathbf{u}}{\mathbf{u}^*\mathbf{u}}. \tag{2.2}$$

Alternatively, with regard to the the "inner product" (1.3), we can also define the Rayleigh quotient by

$$\theta = \theta(\mathbf{u}) := \frac{\mathbf{u}^T A\mathbf{u}}{\mathbf{u}^T\mathbf{u}}. \tag{2.3}$$

One may check that for complex-symmetric A, the latter definition has the desirable property (cf. [20, p. 688], [13])

$$\theta(\mathbf{u}) \text{ is stationary} \iff \mathbf{u} \text{ is an eigenvector of } A. \tag{2.4}$$

(Recall that stationary means that all directional derivatives are zero.) By writing

$$\mathbf{u} = (\mathbf{x}\mathbf{x}^T)\mathbf{u} + \left(I - \mathbf{x}\mathbf{x}^T\right)\mathbf{u},$$

we see that $\mathbf{u}$ can be written in the form

$$\mathbf{u} = \alpha\mathbf{x} + \delta\mathbf{d}, \tag{2.5}$$

where $\alpha^2 + \delta^2 = 1$, $(\mathbf{d},\mathbf{d})_T = 1$ and $\mathbf{x} \perp_T \mathbf{d} = 0$. Direct computation shows that

$$\lambda - \theta = \delta^2\, \mathbf{d}^T(\lambda I - A)\mathbf{d}.$$

So, we conclude that

$$|\lambda - \theta| = \mathcal{O}(\delta^2), \tag{2.6}$$

while $|\lambda - \rho|$ is in general "only" $\mathcal{O}(\delta)$. (The reason for the last statement is that in general the eigenvectors are not stationary points of $\rho(\mathbf{u})$.) Therefore, the Rayleigh quotient θ is asymptotically (i.e., when $\mathbf{u}$ converges to $\mathbf{x}$) more accurate than the usual Rayleigh quotient ρ.

3. Jacobi–Davidson for complex-symmetric matrices. In this section we introduce a Jacobi–Davidson method for complex-symmetric matrices that we denominate JDCS. A subspace method typically consists of two ingredients: *extraction* and *expansion*. Suppose we have a k-dimensional search space $\mathcal{U}$ where typically $k \ll n$. The crucial observation is that if $\mathcal{U}$ is the search space for the (right) eigenvector, then with regard to the "inner product" (1.3), $\mathcal{U}$ forms a search space for the left eigenvector of equal quality. So, the fundamental difference with the two-sided Jacobi–Davidson algorithm [13] is that as we build up a right search space (i.e., a search space for the right eigenvector), we get a reasonable left search space for free. We do not have to (approximately) solve a left correction equation as in the two-sided Jacobi–Davidson algorithm.

3.1. Extraction. We first study the subspace extraction for complex-symmetric matrices. Given a search space $\mathcal{U}$, we would like to get an approximate eigenpair $(\theta, \mathbf{u})$ where $\mathbf{u} \in \mathcal{U}$. Let the columns of U form a basis for $\mathcal{U}$, and define the *residual* $\mathbf{r}$ by

$$\mathbf{r} := A\mathbf{u} - \theta\mathbf{u}.$$

In view of (2.4) and (2.6), we take, instead of the usual Ritz–Galerkin condition on the residual $\mathbf{r} = A\mathbf{u} - \theta\mathbf{u} \perp \mathcal{U}$, the same condition but with respect to the "inner product" (1.3)

$$\mathbf{r} = A\mathbf{u} - \theta\mathbf{u} \perp_T \mathcal{U}, \tag{3.1}$$

Writing $\mathbf{u} = U\mathbf{c}$, $\mathbf{c} \in \mathbb{C}^k$, we find that $(\theta, \mathbf{c})$ must be a solution of the projected eigenproblem

$$U^T AU\mathbf{c} = \theta\, U^T U\mathbf{c}. \tag{3.2}$$

Thus, a *Ritz pair* $(\theta, \mathbf{u}) = (\theta, U\mathbf{c})$ is obtained by backtransforming an eigenpair of the *projected pencil* $(U^T AU, U^T U)$. In particular, if $(\theta, \mathbf{u})$ is a Ritz pair, we have

$$\theta = \theta(\mathbf{u}) := \frac{\mathbf{u}^T A\mathbf{u}}{\mathbf{u}^T\mathbf{u}} \qquad \text{and} \qquad \mathbf{r} \perp_T \mathbf{u}.$$

3.2. Expansion. Let us now examine the subspace expansion for JDCS: having an approximate eigenpair $(\theta, \mathbf{u})$ to $(\lambda, \mathbf{x})$, how do we expand the search space $\mathcal{U}$ in order to get an even better approximation? Jacobi–Davidson type methods look for a correction $\mathbf{s}$ such that

$$A(\mathbf{u} + \mathbf{s}) = \lambda(\mathbf{u} + \mathbf{s}), \tag{3.3}$$

i.e., such that $\mathbf{u} + \mathbf{s}$ is a multiple of the eigenvector $\mathbf{x}$. This equation can be rewritten in two different ways, depending on whether we wish that $\mathbf{s} \perp_T \mathbf{u}$ or $\mathbf{s} \perp \mathbf{u}$. Let us start with $\mathbf{s} \perp_T \mathbf{u}$. Write (3.3) as

$$(A - \theta I)\mathbf{s} = -(A - \theta I)\mathbf{u} + (\lambda - \theta)\mathbf{u} + (\lambda - \theta)\mathbf{s}. \tag{3.4}$$

In view of (2.6), the term $(\lambda - \theta)\mathbf{s}$ is asymptotically of third order. When we neglect this term, we still have cubic convergence, see Theorem 4.1. During the process, λ and hence also $(\lambda - \theta)\mathbf{u}$ are unknown. Therefore it is interesting to consider the projection of this equation that maps $\mathbf{u}$ to $\mathbf{0}$ and keeps $\mathbf{r} = (A - \theta I)\mathbf{u}$ fixed. Because $\mathbf{r} \perp_T \mathbf{u}$, this

projector is $I - \mathbf{u}\mathbf{u}^T$, the oblique projection onto $\mathbf{u}^{\perp_T}$. The result of projecting (3.4) is

$$\left(I - \mathbf{u}\mathbf{u}^T\right)(A - \theta I)\mathbf{s} = -\mathbf{r}.$$

Using the constraint

$$\left(I - \mathbf{u}\mathbf{u}^T\right)\mathbf{s} = \mathbf{s},$$

we derive the first possibility for the JDCS correction equation:

$$\left(I - \mathbf{u}\mathbf{u}^T\right)(A - \theta I)\left(I - \mathbf{u}\mathbf{u}^T\right)\mathbf{s} = -\mathbf{r}, \qquad \text{where } \mathbf{s} \perp_T \mathbf{u}. \tag{3.5}$$

The operator in this equation is complex-symmetric. So, we can try to solve (3.5) by a linear solver that is especially designed for complex-symmetric systems, such as CSYM [5], complex-symmetric QMR [9], or COCG [28].

Second, we investigate the situation where we wish to have $\mathbf{s} \perp \mathbf{u}$. We rewrite (3.3) as

$$(A - \theta I)\mathbf{s} = -(A - \rho I)\mathbf{u} + (\lambda - \rho)\mathbf{u} + (\lambda - \theta)\mathbf{s}.$$

Again neglecting the last term and noting that

$$\widetilde{\mathbf{r}} := (A - \rho I)\mathbf{u} \perp \mathbf{u},$$

this leads to an alternative JDCS correction equation:

$$(I - \mathbf{u}\mathbf{u}^*)(A - \theta I)(I - \mathbf{u}\mathbf{u}^*)\mathbf{s} = -\widetilde{\mathbf{r}}, \qquad \text{where } \mathbf{s} \perp \mathbf{u}. \tag{3.6}$$

Unless A is Hermitian, this operator does not have any particular properties. Therefore, GMRES is a reasonable solver. In practice, a correction equation is often solved only *approximately* (or *inexactly*). The approximate solution is used to expand the search space $\mathcal{U}$, this is called *subspace acceleration*.

Next, we mention that JDCS can be viewed as an *accelerated inexact Newton method* for the eigenvalue problem. For the correction equation (3.6) such a result has been given in [23]. For the correction equation (3.5), we define

$$F(\mathbf{u}) = A\mathbf{u} - \frac{\mathbf{a}^T A\mathbf{u}}{\mathbf{a}^T\mathbf{u}}\mathbf{u},$$

then a Newton step $DF(\mathbf{u})\mathbf{s} = -F(\mathbf{u})$ becomes, with $\mathbf{a} = \mathbf{u}$

$$\left(I - \mathbf{u}\mathbf{u}^T\right)(A - \theta I)\mathbf{s} = -\mathbf{r}.$$

Algorithm 3.1 summarizes our algorithm JDCS as developed so far. In Step 2, MGS stands for any numerically stable form of Gram–Schmidt to form a complex-orthogonal basis for the search space. Because of the complex-orthogonal basis, the matrix on the right hand side of (3.2) is the identity whence we only have to solve a standard eigenvalue problem in Step 4. In Step 5 and elsewhere in the paper, $\|\cdot\|$ denotes the Euclidean norm. In Step 8 of the algorithm the correction equation (3.5) could be replaced by (3.6). Some practical issues have been omitted in Algorithm 3.1 for ease of presentation:

ALGORITHM 3.1: **The JDCS algorithm for the computation of an eigenpair of a complex-symmetric matrix closest to a target τ**

Input: *a device to compute* $A\mathbf{x}$ *for arbitrary* $\mathbf{x}$*, a starting vector* $\mathbf{u}_1$*, and a tolerance* ε*.*

Output: *an approximation* $(\theta, \mathbf{u})$ *to an eigenpair of* A *satisfying* $\|A\mathbf{u} - \theta\mathbf{u}\| \leq \varepsilon$*.*

1. $s = \mathbf{u}_1$
for $k = 1, 2, \ldots$
2. $U_k = MGS\,(U_{k-1}, \mathbf{s})$
3. *Compute* k*th column of* $W_k = AU_k$
Compute k*th row and column of* $H_k = U_k^T W_k$
4. *Compute the eigenpair* $(\theta, \mathbf{c})$ *of* $U_k^T AU_k$ *that is closest to the target* τ
5. $\mathbf{u} = U_k\mathbf{c} \,/\, \|U_k\mathbf{c}\|$
6. $\mathbf{r} = (A - \theta I)\mathbf{u} = W_k\mathbf{c} \,/\, \|U_k\mathbf{c}\| - \theta\mathbf{u}$
7. *Stop if* $\|\mathbf{r}\| \leq \varepsilon$
8. *Solve (approximately) for* $\mathbf{s} \perp_T \mathbf{u}$
$$\left(I - \mathbf{u}\mathbf{u}^T\right)(A - \theta I)\left(I - \mathbf{u}\mathbf{u}^T\right)\mathbf{s} = -\mathbf{r}$$

(1) In our actual computations we replace the shift θ in the correction equation of Step 8 in the first couple of iterations by a fixed target τ which we know or hope to be close to the desired eigenvalue. This is reasonable as the correction equation, if solved exactly, amounts to one step of shift-and-invert. As initially the Ritz value θ is far from the desired eigenvalue using θ as shift does not give any benefit. We switch the shift from τ to θ as soon as the residual $\|\mathbf{r}\|$ is below some threshold like 1/10.

(2) To restrict the memory consumption of our algorithm, we limit the dimension of the search space $\mathcal{U}$. If this limit is reached, we *restart*, i.e., we replace $\mathcal{U}$ by a given number of the "best" Ritz vectors contained in $\mathcal{U}$.

(3) If we need to compute several eigenpairs, we apply the algorithm repeatedly. Hereby, we use the search space of the previous iteration as our initial search space. Furthermore, the correction equation (3.5) is replaced by

$$(I - \widetilde{U}\widetilde{U}^T)(A - \theta I)(I - \widetilde{U}\widetilde{U}^T)\mathbf{s} = -\mathbf{r}, \qquad \widetilde{U}^T\mathbf{s} = \mathbf{0}. \tag{3.7}$$

Here, $\widetilde{U} = [\mathbf{u}, \mathbf{x}_1, \ldots]$ contains besides the Ritz vector $\mathbf{u}$ the eigenvectors $\mathbf{x}_i$ that have been computed previously. Notice that $\widetilde{U}$ may contain some further orthogonality constraints, see Section 5.1.

Now we consider the correction equation for the generalized eigenproblem. In this case (3.4) becomes

$$(A - \theta B)\mathbf{s} = -(A - \theta B)\mathbf{u} + (\lambda - \theta)B\mathbf{u} + (\lambda - \theta)B\mathbf{s}. \tag{3.8}$$

One may check that with the Galerkin condition $\mathbf{r} = A\mathbf{u} - \theta B\mathbf{u} \perp_T \mathcal{U}$, leading to

$$\theta = \frac{\mathbf{u}^T A\mathbf{u}}{\mathbf{u}^T B\mathbf{u}},$$

the last term on the right hand side of (3.8) is of third order. The projector $I - B\mathbf{u}(\mathbf{u}^T B\mathbf{u})^{-1}\mathbf{u}^T$ annihilates the term $(\lambda - \theta)B\mathbf{u}$. So, the correction equation for the

generalized eigenvalue problem corresponding to (3.5) is

$$(I - B\mathbf{u}\mathbf{u}^T)(A - \theta B)(I - \mathbf{u}\mathbf{u}^T B)\mathbf{s} = -(A - \theta B)\mathbf{u}, \qquad \mathbf{s} \perp_T B\mathbf{u}. \tag{3.9}$$

Notice that the operator is complex-symmetric. By analogous manipulations, (3.7) becomes

$$(I - B\widetilde{U}\widetilde{U}^T)(A - \theta B)(I - \widetilde{U}\widetilde{U}^T B)\mathbf{s} = -\mathbf{r}, \qquad \mathbf{s} \perp_T B\widetilde{U}. \tag{3.10}$$

3.3. Harmonic Ritz vectors. It is well known that Ritz–Galerkin extraction (see Section 3.1) works out nicely for exterior eigenvalues, but may give poor approximations to interior eigenvalues. For these eigenvalues, we can apply a harmonic Ritz approach, just as in the standard Jacobi–Davidson method [19, 3]. Suppose that we are interested in one or more interior eigenpairs near the target τ. One idea is to consider a ("complex-symmetric") Galerkin condition on $(A - \tau I)^{-1}$:

$$(A - \tau I)^{-1}\widetilde{\mathbf{u}} - (\widetilde{\theta} - \tau)^{-1}\widetilde{\mathbf{u}} \perp_T \widetilde{\mathcal{U}}, \quad \widetilde{\mathbf{u}} \in \widetilde{\mathcal{U}}. \tag{3.11}$$

With $\widetilde{\mathcal{U}} := (A - \tau I)\mathcal{U}$ and $\widetilde{\mathbf{u}} = \widetilde{U}\widetilde{\mathbf{c}}$ this condition becomes

$$U^T(A - \tau I)^2 U\widetilde{\mathbf{c}} = (\widetilde{\theta} - \tau)U^T(A - \tau I)U\widetilde{\mathbf{c}}. \tag{3.12}$$

The solutions $(\tilde{\theta}, U\widetilde{\mathbf{c}})$ to this small complex-symmetric eigenvalue problem are called harmonic Ritz pairs. If we are to compute interior eigenvalues of A then the common procedure is to replace the eigenvalue problem in Step 4 of Algorithm 3.1 by (3.12) and extract the harmonic Ritz pair closest to the target value τ. We can multiply (3.12) from the left by $\widetilde{\mathbf{c}}^T$ to obtain

$$((A - \tau I)U\widetilde{\mathbf{c}}, (A - \tau I)U\widetilde{\mathbf{c}})_T = (\tilde{\theta} - \tau)((A - \tau I)U\widetilde{\mathbf{c}}, U\widetilde{\mathbf{c}})_T.$$

In contrast to the case where A is Hermitian the expression on the left is not a residual norm, whence a small eigenvalue of (3.12) does not necessarily imply that $U\widetilde{\mathbf{c}}$ is a good eigenvector approximation; the harmonic Ritz vector does not necessarily have a small residual norm.

Therefore, it is more promising to use the harmonic approach that is based on the usual Euclidean inner product. This approach leads to the generalized eigenproblem (see, for instance, [26, p. 296]):

$$U^*(A - \tau B)^*(A - \tau B)U\widetilde{\mathbf{c}} = (\widetilde{\theta} - \tau)U^*(A - \tau B)^* BU\widetilde{\mathbf{c}}. \tag{3.13}$$

This extraction has the mathematical justification that

$$\|(A - \tau B)U\widetilde{\mathbf{c}}\| \leq |\widetilde{\theta} - \tau|\|BU\|$$

but the reduced system (3.13) is not complex-symmetric.

3.4. Refined Ritz vectors. A second approach to compute interior eigenvalues is through refined Ritz vectors. Let $(\theta, \mathbf{u})$ be a Ritz pair, i.e., a solution of (3.1). The Ritz value θ may "by coincidence" be close to an interior eigenvalue of A although the corresponding Ritz vector is a linear combination of eigenvectors that are not close to the particular eigenvector. In this situation that is common when computing interior eigenvalues, the computed Ritz vectors are of no use as approximate eigenvectors in

the correction equation. A remedy suggested in [15] (see also [26, p. 289]) is to *refine* the Ritz vectors. A refined Ritz vector is defined to be a solution of the problem

$$\text{minimize } \|A\widehat{\mathbf{x}} - \theta\widehat{\mathbf{x}}\| \quad \text{subject to} \quad \widehat{\mathbf{x}} \in \mathcal{U},\ \|\widehat{\mathbf{x}}\| = 1. \tag{3.14}$$

Let $\widehat{\mathbf{x}} = U\widehat{\mathbf{c}}$. Then $\widehat{\mathbf{c}}$ is the 'smallest' right singular vector of $(A - \theta I)U$. It is much cheaper [16] to use the characterization of $\widehat{\mathbf{c}}$ being the 'smallest' eigenvector of

$$U^*(\bar{A} - \bar{\theta}I)(A - \theta I)U\mathbf{c} = (U^*\bar{A}AU - \bar{\theta}U^*\bar{A}U - \theta U^*AU + |\theta|^2U^*U)\mathbf{c} = \mu\mathbf{c}. \tag{3.15}$$

The matrices $U^*\bar{A}AU$, $U^*\bar{A}U$, and U^*AU can be computed incrementally along with H_k in Algorithm 3.1. It is straightforward how to modify (3.14) and (3.15) for the generalized eigenvalue problem.

In our numerical experiments we use a modification of Algorithm 3.1 that uses this refined approach. In our implementation Steps 4 and 5 become

4.	Compute the eigenpair $(\theta, \mathbf{c})$ of $U_k^TAU_k$ that is closest to the target τ Determine an eigenvector $\tilde{\mathbf{c}}$ corresponding to the smallest eigenvalue of (3.15)
5.	$\mathbf{u} = U_k\tilde{\mathbf{c}}/\|U_k\tilde{\mathbf{c}}\|, \quad \theta = \mathbf{u}^TA\mathbf{u}/\mathbf{u}^T\mathbf{u}$

4. Convergence of (inexact) JD for complex-symmetric matrices. When we solve any of the two correction equations (3.5) or (3.6) exactly, then we find (see e.g. [22])

$$\mathbf{s} = -\mathbf{u} + \alpha\,(A - \theta I)^{-1}\,\mathbf{u},$$

where α is such that $\mathbf{s} \perp_T \mathbf{u}$ or $\mathbf{s} \perp \mathbf{u}$. JDCS uses $\mathbf{s}$ to expand the search space $\mathcal{U}$. Since already $\mathbf{u} \in \mathcal{U}$, we get the same subspace expansion using $\widetilde{\mathbf{s}} = (A - \theta I)^{-1}\,\mathbf{u}$. Here we recognize a step of RQI, and we conclude that *exact* JDCS (i.e. JDCS where we solve the correction equation exactly) can also be interpreted as (subspace-)accelerated RQI.

Therefore, we first define a Rayleigh quotient iteration for complex-symmetric matrices, and show that this RQI has asymptotically cubic convergence for eigenpairs of which the vector is not quasi-null; see Algorithm 4.2.

ALGORITHM 4.2: **Rayleigh quotient iteration for complex-symmetric matrices**

Input: *An initial vector* $\mathbf{u}_1$*, not quasi-null*
Output: *an eigenpair of A (or failure)*
for $k = 1, 2, \ldots$
1. *Compute* $\theta_k := \theta_k(\mathbf{u}_k) = \dfrac{\mathbf{u}_k^TA\mathbf{u}_k}{\mathbf{u}_k^T\mathbf{u}_k}$
2. **If** $A - \theta_k I$ *is singular* **then** *solve* $(A - \theta_k I)\mathbf{x} = 0$
3. *Solve* $(A - \theta_k I)\mathbf{u}_{k+1} = \mathbf{u}_k$
4. **If** $(\mathbf{u}_{k+1}, \mathbf{u}_{k+1})_T = 0$ **then** *method fails*
   else *"normalize"* $\mathbf{u}_{k+1}$ *such that* $(\mathbf{u}_{k+1}, \mathbf{u}_{k+1})_T = 1$

THEOREM 4.1 (Locally cubic convergence of RQI for complex-symmetric matrices, cf. [20, p. 689], [13, Theorem 5.2]). *Suppose that* $\mathbf{u}_k = \alpha_k\mathbf{x} + \delta_k\mathbf{d}_k$ *(cf.* (2.5)*) converges*

to $\mathbf{x}$*, where* $\mathbf{x}$ *is not quasi-null, as* $k \to \infty$*. Then* $\theta_k \to \lambda$*, and we have*

$$\delta_{k+1} = \mathcal{O}(\delta_k^3).$$

Proof. We have

$$\mathbf{u}_{k+1} = \alpha_{k+1}(\mathbf{x} + \delta_k(\lambda - \theta_k)\,(A - \theta_k I)^{-1}\,\mathbf{d}_k).$$

Now, a combination of (2.6), and the fact that $A - \lambda I$ exists on $\mathbf{x}^{\perp_T}$ (since $A - \lambda I : \mathbf{x}^{\perp_T} \to \mathbf{x}^{\perp_T}$ is a bijection) proves the theorem. ∎

Now as a corollary, the asymptotically cubic convergence of JDCS follows. (See [25, p. 652] for a discussion about the term "asymptotic convergence" for subspace methods.)

JD and RQI type methods are in practice often very expensive when we solve the linear systems, occurring in the methods, accurately. We therefore consider inexact variants. where the linear systems are solved to a certain precision (minimal residual approach).

First consider the situation where we solve the linear system of complex-symmetric RQI method inexactly, by which we mean that we are satisfied with a $\mathbf{u}_{k+1}$ if

$$\|(A - \theta_k I)\mathbf{u}_{k+1} - \mathbf{u}_k\| \leq \xi < 1. \tag{4.1}$$

Notice that it may become increasingly difficult to satisfy (4.1) as θ_k tends to λ because $A - \theta_k I$ is almost singular.

The following two theorems can be proved by similar methods as in [13, Lemma 5.1, Theorems 5.2 and 5.3], exploiting the fact that the right eigenvector $\mathbf{x}$ is a left eigenvector as well.

THEOREM 4.2 (Locally quadratic convergence of inexact RQI for complex-symmetric matrices, cf. [13, Theorem 5.2]). *Let the iterates of the inexact RQI* $\mathbf{u}_k = \alpha_k\mathbf{x} + \delta_k\mathbf{d}_k$ *satisfy* (4.1) *with an accuracy* ξ *such that* $\xi \cdot |(\mathbf{x}, \mathbf{x})_T| < 1$*. Then*

$$\delta_{k+1} = \mathcal{O}(\delta_k^2).$$

Consider the situation where we solve the correction equation (3.5) or (3.6) of the complex-symmetric JD method inexactly, by which we mean that we are satisfied with $\widetilde{\mathbf{s}} \perp_T \mathbf{u}_k$ where

$$\left\|\left(I - \mathbf{u}_k{\mathbf{u}_k}^T\right)(A - \theta I)\widetilde{\mathbf{s}} + \mathbf{r}_k\right\| \leq \eta\,\|\mathbf{r}_k\| \tag{4.2}$$

for some $0 < \eta < 1$.

THEOREM 4.3 (Locally linear convergence of inexact Jacobi–Davidson for complex-symmetric matrices, cf. [13, Theorem 5.3]). *Let* $\mathbf{u}_k = \alpha_k\mathbf{x} + \delta_k\mathbf{d}_k$ *be the iterates of inexact JDCS satisfying* (4.2)*. Then*

$$\delta_{k+1} \leq \gamma\delta_k + \mathcal{O}(\delta_k^2),$$

where $\gamma = \kappa((A - \lambda I)|_{\mathbf{x}^{\perp_T} \to \mathbf{x}^{\perp_T}})$.

We remark that for the variant where $\widetilde{\mathbf{s}} \perp \mathbf{u}_k$ we have a similar statement, now with $\gamma = \kappa((A - \lambda I)|_{\mathbf{x}^{\perp} \to \mathbf{x}^{\perp_T}})$. So, for inexact JDCS we expect linear convergence

which can be observed in practice, see the figures in the next section. As mentioned before, the condition number γ in Theorem 4.3 is bounded. Nevertheless it can be very large if the distance of λ to the eigenvalue is small. This is a situation that we encounter in our numerical examples.

As also discussed in [13], we cannot conclude from the theorems that inexact RQI is faster than inexact Jacobi–Davidson. The theorems only say something on the local, not the global, rate of convergence (note that Jacobi–Davidson has subspace acceleration); moreover, (4.2) is potentially easier to satisfy than (4.1) as $\left(I - \mathbf{u}_k \mathbf{u}_k{}^T\right)(A - \theta I)$ considered as a mapping from $\mathbf{x}^{\perp_T}$ into itself has a bounded inverse.

5. Numerical experiments. In this section we discuss the applicability of our algorithm to three test problems from electro-magnetics. All tests have been executed with MATLAB 6.1 (Release 12.1) on a PC with a 1.8GHz Pentium 4 processor and 1GB of main memory running the Linux 2.4.9 operating system.

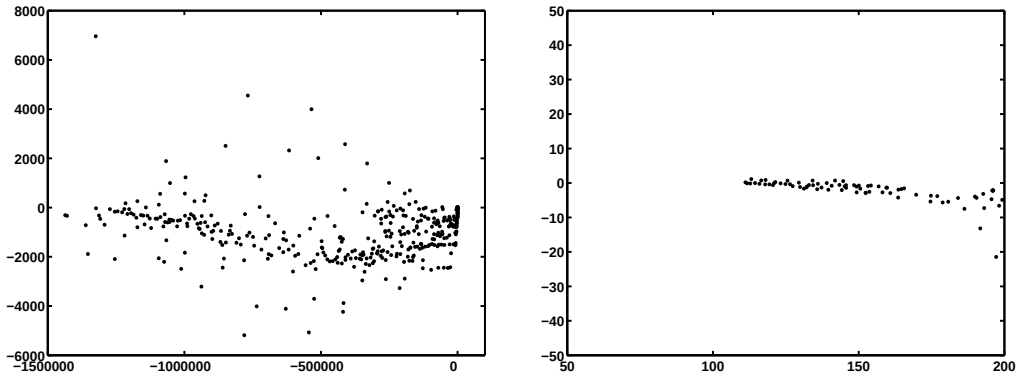
5.1. A dispersive waveguide structure. We consider first the generalized eigenvalue problem `dwg961`, that is available from the Matrix Market [18]. It originates from a finite element discretization of a waveguide problem with conductors of finite cross-section in a lossy dielectric medium. The eigenvalue problem has the form

$$A\mathbf{x} = \begin{bmatrix} A_{11} & O \\ O & O \end{bmatrix} \begin{pmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \end{pmatrix} = \lambda \begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{bmatrix} \begin{pmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \end{pmatrix} = \lambda B\mathbf{x}. \tag{5.1}$$

These matrix structures are obtained if the Maxwell equations are discretized by finite edge elements. The order of the overall problem is 961. The order of A_{11} is 705. The matrix B as well as the submatrix A_{11} are nonsingular. Thus, (5.1) has 256 zero eigenvalues. The corresponding eigenspace is

$$\mathcal{N}(A) = \mathcal{R}(Y), \qquad Y = \begin{bmatrix} O \\ I_{256} \end{bmatrix}.$$

Here, $\mathcal{N}(\cdot)$ and $\mathcal{R}(\cdot)$ denote nullspace and range of the respective matrices. To prevent the iterates from converging to eigenvectors corresponding to the zero eigenvalue we force them to be complex-orthogonal to $\mathcal{R}(BY)$. Technically, we can do that by incorporating Y into the set of found eigenvectors $\widetilde{U}$ in the correction equation (3.10).



(a) (b)

FIG. 5.1. *Complete spectrum of* `dwg961` *(a) and portion of the spectrum close to* $\tau = 100$ *(b). The plot shows imaginary vs. real parts of the eigenvalues.*

We want to find a few of the smallest eigenvalues with positive real part. In Figure 5.1 the spectrum of the matrix pencil $(A; B)$ of (5.1) is plotted. Besides the 256 zero eigenvalues there are 362 eigenvalues with negative real part and 343 with positive real part. Although there is no eigenvalue with real part between -2500 and 100 the two sets are in a relative sense not well separated. From Figure 5.1(a) we see that the eigenvalues with positive real part are much more clustered than those with negative real part. The smallest of the former are about 0.5 to 1 apart. The largest eigenvalue (in modulus) is about $1.4 \cdot 10^6$. Thus, the relative gap is quite small and the condition number of the correction equation is about 10^6.

We determine the six smallest eigenvalues with positive real part by the Jacobi–Davidson Algorithm 3.1 for computing interior eigenvalues close to the shift $\tau = 100$ employing refined Ritz vectors. The practical considerations (1)–(3) mentioned after Algorithm 3.1 apply.

The efficient solution of the correction equation (3.10) requires preconditioning. We employ complex symmetric (left) preconditioners of the form

$$(I - B\widetilde{U}\widetilde{U}^T)M(I - \widetilde{U}\widetilde{U}^T B)$$

where M approximates $A - \tau B$. The computation of the preconditioned residual $\mathbf{s}$ from the residual $\mathbf{r}$ amounts to solving

$$(I - B\widetilde{U}\widetilde{U}^T)M(I - \widetilde{U}\widetilde{U}^T B)\mathbf{s} = \mathbf{r}, \qquad \widetilde{U}^T B\mathbf{s} = \mathbf{0}. \tag{5.2}$$

The solution of (5.2) is given by

$$\mathbf{s} = (I - M^{-1}B\widetilde{U}(\widetilde{U}^T BM^{-1}B\widetilde{U})^{-1}\widetilde{U}^T B)M^{-1}\mathbf{r}. \tag{5.3}$$

Usually, the order of the matrix $\widetilde{U}^T BM^{-1}B\widetilde{U}$ in the projector in (5.3) is small, i.e. less than or equal to the number of desired eigenpairs. In the present example $\widetilde{U}$ also comprises the basis of the null space of A whence it is quite big. Note, that we keep the preconditioner fixed throughout the computation. In the course of the computation, the quality of the preconditioner may deteriorate if many eigenpairs are desired. Then the preconditioner may have to be updated by techniques like those suggested in [24]. As we compute only a few close eigenvalues a constant preconditioner turns out be good enough.

The special solvers discussed in the previous sections can only be employed for solving (3.10)–(5.2) if the preconditioner and thus M is complex symmetric. We assume M to be of the form $M = LDL^T$ where D is a diagonal and L is a unit lower triangular matrix. Actually, it is not important to have the preconditioner available in factored form, but it makes the symmetry more evident and also exploits the fact that we only store a triangle of A and B. We experiment with

- *diagonal preconditioning.* $M = D = \operatorname{diag}(A - \tau B)$,
- *symmetric Gauss-Seidel (or SSOR(1)) preconditioning.* $M = (D + L)D^{-1}(D + L^T)$ where $D = \operatorname{diag}(A - \tau B)$ and L is the strict lower triangular portion of $A - \tau B = L + D + L^T$.
- *incomplete (complex symmetric) LDL^T factorization preconditioning.* Here, L is a unit lower triangular matrix with the same zero pattern as the lower triangle of $A - \tau B$. The factors can formally be computed as in the real symmetric case [11, §10.3.2]. Notice, that this factorization does not always exist. The procedure could be stabilized by a 2×2 pivoting strategy [4].

- *LDL^T factorization preconditioning.* Here, $M = A-\tau B$. L and D are obtained by ordinary Gaussian elimination of $A-\tau B$ with pivoting suppressed, i.e. with diagonal pivoting*.

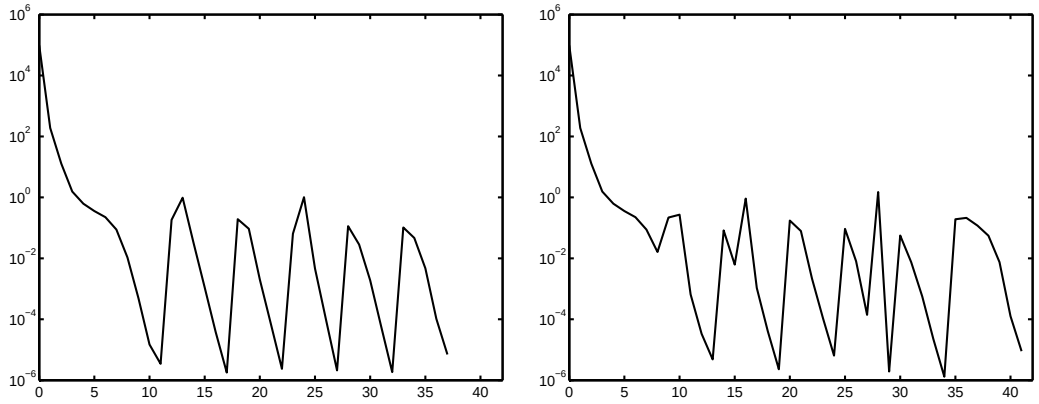


FIG. 5.2. *Convergence history of the JDCS algorithm combined with QMR (a) and COCG (b) solvers for* `dwg961`*: residual norm* $\|(A-\theta(\mathbf{x})B)\mathbf{x}\|_2$ *vs. outer iteration count.*

We compute the six smallest eigenvalues with positive real part. The convergence criterion for the outer iteration (JDCS) is $\|(A-\theta(\mathbf{x})B)\mathbf{x}\|_2 < 10^{-6}\|\mathbf{x}\|_2$. The inner iteration (QMR or COCG†) is considered converged if the relative residual is smaller than $\max\{2^{-1-j}, 10^{-6}\}$ where j is the iteration count of the Jacobi–Davidson algorithm. As suggested in [8], j is set to zero at the beginning of the computation and when an eigenpair has been found. The shift σ in the correction equation is set equal to the target τ as long as the residual norm $\|(A-\theta(\mathbf{x}_j)B)\mathbf{x}_j\|_2 \geq 1/10$. If the residual norm drops below $1/10$ then $\sigma = \theta(\mathbf{x}_j)$ is set. The value $1/10$ was found to be satisfactory by experimentation. The iteration is started with a random vector. The maximally allowed steps for the inner iteration is chosen small, here 10. This limit is actually hit quite often if $\sigma \neq \tau$.

Convergence histories are found in Figure 5.2. These results are obtained with the QMR (left) and COCG (right) solvers. Although the convergence behavior of these two solvers looks quite different, the number of outer iteration steps is within 10%. The 2-norm of the residual is smoother with QMR. JDCS/COCG in three cases gets quite close to an eigenpair but then switches to another one.

Here, the LDL^T preconditioner was chosen. $A-\tau B$ is very sparse and after application of the MATLAB minimum degree reordering there is not too much fill-in produced in the factorization. The number of nonzeros of the lower triangle of $A-\tau B$ is 5776, the number of nonzeros of the L-factor is 20740. All other preconditioners mentioned above turn out to be insufficient in that the complex-symmetric linear system solvers do not converge at all independently of the number of iteration steps that are permitted for the inner iteration. In the successful computations corresponding to the plots in Figure 5.2 the inner iteration is stopped after convergence or after 10

*The MATLAB command `[L,U]=lu(A - tau*B,0)` provides the desired factor L while D is the diagonal of U. The second input parameter suppresses pivoting.

†As CSYM was not competitive with the other two complex-symmetric solvers in preliminary tests, we decided not to consider CSYM in these numerical experiments.

iteration steps.

The Jacobi–Davidson algorithm is restarted as soon as the search space had dimension 20. The iteration is continued (restarted) with the 10 best refined Ritz vectors available.

In Table 5.1 some statistics are listed for all three test problems that we solved. Besides the number nit_{out} of outer iteration steps, the total number nit_{in} of steps of the inner iteration is given. Thus, the correction equation is solved in the average in $nit_{\text{in}}/nit_{\text{out}}$ steps. Notice that problem `dwg961` is relatively expensive to solve compared with the much larger test problems that we are going to discuss as $\widetilde{U}$ in (3.10)–(5.2) also contains the basis of the null space of A. The execution time of JDCS combined with COCG is shorter than JDCS combined with QMR although the number of inner iteration steps is larger. We surmise that this is due to the better implementation of COCG which is obtained by a few trivial modifications of MATLAB's efficient implementation of the preconditioned conjugate gradient (PCG) algorithm.

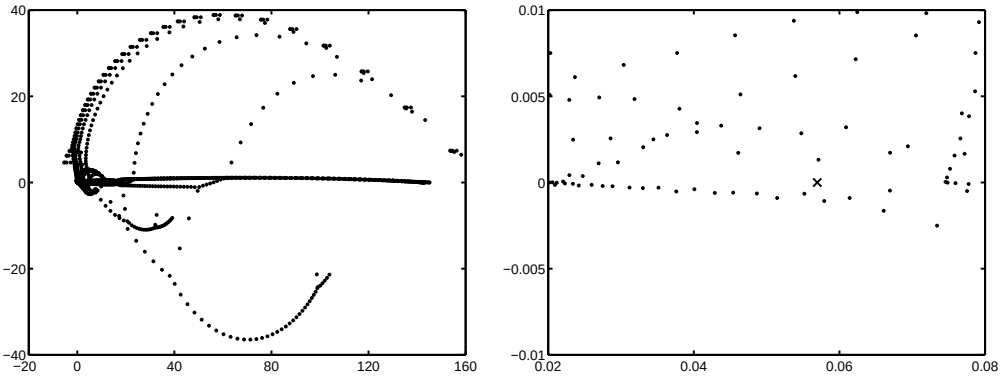
	`dwg961` ($n=961$)			`toy2` ($n=6243$)			`wg` ($n=32098$)		
	nit_{out}	nit_{in}	time	nit_{out}	nit_{in}	time	nit_{out}	nit_{in}	time
JDCS/QMR	37	221	91.1	20	79	21.1	19	88	247.8
JDCS/COCG	41	234	77.6	19	75	15.3	21	104	183.6

TABLE 5.1

Iteration counts and execution times (in seconds) for all the three numerical examples.

We succeeded in solving this smallest problem with the incomplete LDL^T preconditioner if we replaced the complex-symmetric system solver by GMRES (without restarts). However, the correction equation always had to be solved to the high relative accuracy 10^{-6} which required about $n/2$ inner iteration steps. Thus, the solution of one correction equation needed about twice as many inner iteration steps as JDCS/QMR or JDCS/COCG with the LDL^T preconditioner, cf. Table 5.1. By consequence, the computation took 3895 seconds! This approach is not feasible in the larger eigenvalue problems.

5.2. A radiating dielectric cavity problem. Our second test example that we call `toy2` is a one-dimensional layered dielectric cavity with six distributed Bragg reflector (DBR) pairs at the top and the bottom, and a quantum well active region sandwiched between the two DBR pairs. A PML lining terminates the one-dimensional structure at the two ends. This structure has no practical significance for vertically-cavity surface-emitting lasers (VCSEL) design. Nevertheless, the treatment of an open cavity using PML can be illustrated. Here, A and B are sparse nonsingular matrices of order 6243. We are looking for a few of the interior eigenvalues close to the real axis in the neighborhood of a real target value that is determined by the laser designer analytically. On the left of Figure 5.3 the whole spectrum is depicted. On the right the vicinity of the spectrum near the target value ($\times$) is shown. The plots indicate that the condition number of the correction equation is at least 10^6. We proceed just as in the first problem except that we allow now up to 20 (interior) iteration steps per call of the linear system solver, a limit that is reached only a few times. Again we observe convergence only with the LDL^T preconditioner. Diagonal and symmetric Gauss-Seidel preconditioning does not reduce the condition of the original problem sufficiently. We factor $A-\tau B$ after applying the symmetric minimum degree



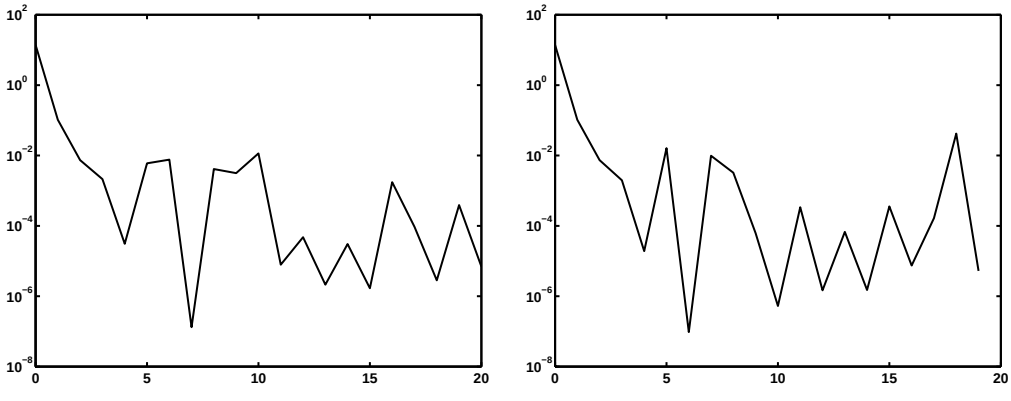
(a) (b)

FIG. 5.3. *Complete spectrum of* `toy2` *(a) and portion of the spectrum close to target* $\tau = 0.057$ *(b). The target is indicated by* $\times$*. The plot shows imaginary vs. real parts of the eigenvalues.*

algorithm available in MATLAB. The number of nonzeros of a triangle of $A - \tau B$ is 43038. The number of nonzeros of the factor L is only 94404.

The convergence histories of the complex-symmetric Jacobi–Davidson algorithm combined with QMR and COCG system solvers are given in Figure 5.4. Here, the convergence behavior of the two solvers is quite similar. Both algorithms exhibit a misconvergence before they find the first eigenpair. Misconvergence means that a sequence of refined Ritz pairs seems to converge to some eigenpair, but close to convergence a Ritz pair emerges that is closer to some other eigenpair.

According to Table 5.1 JDCS/COCG now requires fewer inner and outer iterations steps. However, the difference in the number of inner iteration steps is just about 5%.



(a) (b)

FIG. 5.4. *Convergence history of the JDCS algorithm combined with QMR (a) and COCG (b) solvers for* `toy2`*: residual norm* $\|(A - \theta(\mathbf{x})B)\mathbf{x}\|_2$ *vs. outer iteration count.*

The eigenvalues (and associated eigenvectors) that are actually computed are those closest to τ. We have verified this by computing all eigenvalues of the `toy2` problem with the QR algorithm[‡]. When the problem to be solved is large this is of course

[‡]This takes a few hours on a PC with a decent main memory.

infeasible. In such a case it is difficult if not impossible to check whether all eigenvalues in a given neighborhood of the target value have been found. We have kept the target value τ fixed throughout the computation. One could set τ to be equal to the most recently computed eigenvalue. This has usually the effect that the target values move away from the original one. In this way it is not easy to control which eigenvalues are found. In general they are not the ones closest to τ.

5.3. A waveguide problem with PML. In the third example we deal with a two-dimensional optical waveguide problem. The cross-section of the waveguide is considered. The waveguide is designed such that the fundamental optical mode experiences considerably lower losses by leakage into the substrate compared to the higher order optical modes. In this way more reliable single/fundamental-mode operation can be achieved in practice. A PML lining terminates the two-dimensional structure on the boundary. The PML is used to render the effect of leakage into substrate [12]. The order of A and B is $n = 32098$. As in the first example, A has a 2×2 block structure where only the block A_{11} is nonzero, cf. (5.1). The dimension of the null

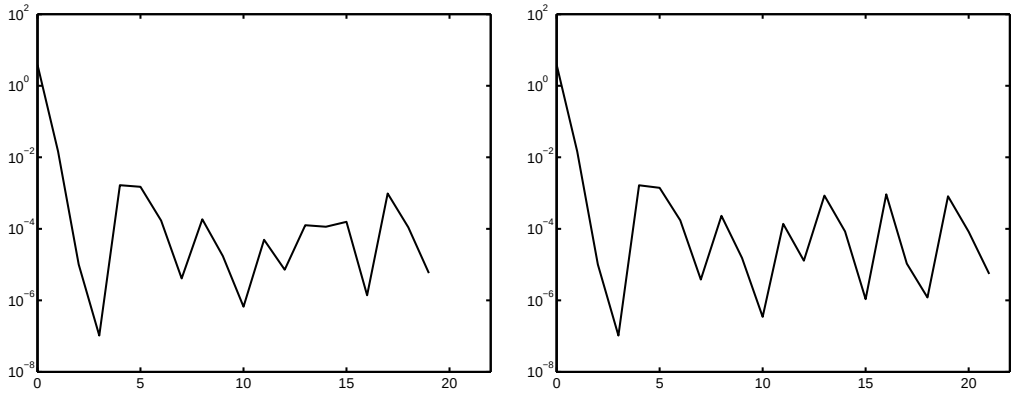


FIG. 5.5. *Convergence history of the JDCS algorithm combined with QMR (a) and COCG (b) solvers for* `wg`*: residual norm* $\|(A - \theta(\mathbf{x})B)\mathbf{x}\|_2$ *vs. outer iteration count.*

space is now $m = 10680$. We did not exploit the knowledge of the basis of the null space because of its size. By applying the projector $I - Y[B_{22}^{-1}B_{21}, I]$ we could force the iterates to be B-orthogonal to the null space $\mathcal{N}(A)$ [1].

As in the second problem we are looking for eigenvalues with small imaginary parts closest to the real target value τ. All parameters were set as in Section 5.2. Again, only the LDL^T preconditioner worked for solving the correction equation. The LDL^T factorization of $M = A - \tau B$ (after minimum degree permutation) yields a L factor with 2595171 nonzeros that consumes 62MB of computer memory in MATLAB's sparse matrix format. The lower triangle of $A - \tau B$ has 264194 nonzeros. The factorization itself took 15 seconds.

The convergence histories of the JDCS algorithm combined with QMR and COCG for solving the correction equations are given in Figure 5.5. Iteration counts and execution times are found in Table 5.1. The convergence histories and thus numbers of outer iterations are again very similar. But JDCS/COCG is now *much* faster than JDCS/QMR. The execution time of the former is about 75% of the latter although the overall number of inner iteration steps is almost 20% higher. This shows that

QMR better solves the correction equation but also that it has a higher overhead than COCG as system matrix and preconditioner are applied twice per inner iteration step.

6. Discussion and Conclusions. We have suggested an algorithm, JDCS, for finding a few eigenvalues and corresponding eigenvectors of special and generalized eigenvalue problems with complex-symmetric matrices. JDCS is a natural generalization of standard JD for complex-symmetric matrices. Most of the techniques known in JD (such as preconditioning the correction equation, using a target, restarting, and techniques for computing interior eigenvalues) easily carry over to JDCS.

Exact JDCS has asymptotically cubic convergence for simple eigenvalues of complex-symmetric matrices. To get this high convergence rate it is crucial to replace the Euclidean inner product $\mathbf{x}^*\mathbf{y}$ by the bilinear form $\mathbf{x}^T\mathbf{y}$. We have shown that the Rayleigh quotient θ based on this "inner product" is closer to the exact eigenvalue λ ($|\lambda - \theta| = \mathcal{O}(\delta^2)$) than the Rayleigh quotient ρ derived from the Euclidean inner product ($|\lambda - \rho| = \mathcal{O}(\delta)$), where δ is the angle between exact and approximating eigenvector.

Compared with the Lanczos algorithm for complex-symmetric matrices [6], JDCS is more flexible, in that we can restart with any vectors we like, and add some extra vectors to the search space. JDCS is also more stable than Lanczos, in the sense that it can easily cope with breakdown, no look-ahead versions are necessary (see [13]).

Of course, JDCS can have disadvantages. We may expect problems when we try to approximate an eigenvector $\mathbf{x}$ that is (approximately) quasi-null: the oblique projections and the Rayleigh Quotient (1.3) may affect the accuracy and stability. A standard JD algorithm that computes a partial Schur form could be better suited in such a situation.

We solved several numerical problems. In all of them we observed convergence in very few iteration steps. However, we were unable to solve our three test problems with preconditioners other than $(I - B\widetilde{U}\widetilde{U}^T)(A - \tau B)(I - \widetilde{U}\widetilde{U}^T B)$. That means that we had to compute the LDL^T factorization of $A - \tau B$ which foils the big advantage of the Jacobi–Davidson algorithm to be factorization-free. As the shift θ in the correction equation is set equal to the target τ in the initial phase of the eigenvalue search, using this preconditioner implies that we can solve the correction equation accurately in a single iteration step. The initial phase actually amounts to performing (accelerated) inverse iteration with shift τ. So, initially we construct a very good search space. Only when this search space contains a good approximation to a desired Ritz pair, "true" JDCS sets in. Our approach is thus very powerful as long as the LDL^T factorization of $A - \tau B$ can be afforded. Unfortunately, this preconditioner will not be feasible for very large systems due to fill-in. It is therefore of paramount importance to find effective complex-symmetric preconditioners that on the one hand approximate well $A - \tau B$ and on the other hand do not consume too much memory space.

Acknowledgments. We thank Henk van der Vorst and Gerard Sleijpen (University of Utrecht) for valuable discussions. We thank Matthias Streiff (Integrated Systems Laboratory, ETH Zurich) for providing the test examples `toy2` and `wg`. We thank Roland Freund (Bell Laboratories, Murray Hill) for providing us with a MATLAB code for the complex-symmetric QMR algorithm. We also would like to thank Angelika Bunse-Gerstner (University of Bremen) for a code of the CSYM algorithm.

REFERENCES

[1] P. Arbenz and Z. Drmač, *On positive semidefinite matrices with known null space*, SIAM J. Matrix Anal. Appl., 24 (2002), pp. 132–149.

[2] P. Arbenz and R. Geus, *A comparison of solvers for large eigenvalue problems originating from Maxwell's equations*, Numer. Lin. Alg. Appl., 6 (1999), pp. 3–16.

[3] Z. Bai, J. Demmel, J. Dongarra, A. Ruhe, and H. van der Vorst, *Templates for the Solution of Algebraic Eigenvalue Problems: A Practical Guide*, Society for Industrial and Applied Mathematics, Philadelphia, PA, 2000.

[4] J. R. Bunch and L. Kaufman, *Some stable methods for calculating inertia and solving symmetric linear systems*, Math. Comp., 31 (1977), pp. 163–179.

[5] A. Bunse-Gerstner and R. Stöver, *On a conjugate gradient-type method for solving complex symmetric linear systems*, Linear Algebra Appl., 287 (1999), pp. 105–123.

[6] J. K. Cullum and R. A. Willoughby, *Lanczos Algorithms for Large Symmetric Eigenvalue Computations*, vol. 1: Theory, Birkhäuser, Boston, 1985.

[7] ———, *A QL procedure for computing the eigenvalues of complex symmetric tridiagonal matrices*, SIAM J. Matrix Anal. Appl., 17 (1996), pp. 83–109.

[8] D. R. Fokkema, G. L. G. Sleijpen, and H. A. van der Vorst, *Jacobi-Davidson style QR and QZ algorithms for the partial reduction of matrix pencils*, SIAM J. Sci. Comput., 20 (1998), pp. 94–125.

[9] R. W. Freund, *Conjugate gradient-type methods for linear systems with complex symmetric coefficient matrices*, SIAM J. Sci. Comput., 13 (1992), pp. 425–448.

[10] F. R. Gantmacher, *The Theory of Matrices*, vol. 2, Chelsea, New York, 1959.

[11] G. H. Golub and C. F. van Loan, *Matrix Computations*, The Johns Hopkins University Press, Baltimore, MD, 3rd ed., 1996.

[12] J. Heaton, M. Bourke, S. Jones, B. Smith, K. Hilton, G. Smith, J. Birbeck, G. Berry, S. Dewar, and D. Wight, *Optimization of deep-etched, single-mode GaAs-AlGaAs optical waveguides using controlled leakage into the substrate*, IEEE J. Lightwave Technology, 17 (1999), pp. 267 –281.

[13] M. E. Hochstenbach and G. L. G. Sleijpen, *Two-sided and alternating Jacobi–Davidson*, Preprint 1196, Dept. Math., University Utrecht, Utrecht, the Netherlands, June 2001. To appear in LAA.

[14] R. A. Horn and C. R. Johnson, *Matrix Analysis*, Cambridge University Press, Cambridge, 1985.

[15] Z. Jia, *Refined iterative algorithms based on Arnoldi's process for large unsymmetric eigenproblems*, Linear Algebra Appl., 259 (1997), pp. 1–23.

[16] ———, *A refined subspace iteration algorithm for large sparse eigenproblems*, Appl. Numer. Math., 32 (2000), pp. 35–52.

[17] F. T. Luk and S. Qiao, *A fast eigenvalue algorithm for Hankel matrices*, Linear Algebra Appl., 316 (2000), pp. 171–182.

[18] *The Matrix Market.* A repository of test data for use in comparative studies of algorithms for numerical linear algebra. Available at URL `http://math.nist.gov/MatrixMarket/`.

[19] C. C. Paige, B. N. Parlett, and H. A. van der Vorst, *Approximate solutions and eigenvalue bounds from Krylov subspaces*, Numer. Linear Algebra Appl., 2 (1995), pp. 115–134.

[20] B. N. Parlett, *The Rayleigh quotient iteration and some generalizations for nonnormal matrices*, Math. Comp., 28 (1974), pp. 679–693.

[21] 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.

[22] 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.

[23] G. L. G. Sleijpen and H. A. van der Vorst, *The Jacobi-Davidson method for eigenvalue problems and its relation with accelerated inexact Newton schemes*, in Iterative Methods in Linear Algebra, II., S. D. Margenov and P. S. Vassilevski, eds., vol. 3 of IMACS Series in Computational and Applied Mathematics, New Brunswick, NJ, U.S.A., 1996, IMACS, pp. 377–389. Proceedings of the Second IMACS International Symposium on Iterative Methods in Linear Algebra, June 17-20, 1995, Blagoevgrad.

[24] G. L. G. Sleijpen and F. W. Wubs, *Effective preconditioning techniques for eigenvalue problems*, Preprint 1117, Utrecht University, Department of Mathematics, August 1999. (Avail-

able at URL `http://www.math.uu.nl/people/sleijpen/`).

[25] A. van der Sluis and H. A. van der Vorst, *The convergence behaviour of Ritz values in the presence of close eigenvalues*, Linear Algebra Appl., 88/89 (1987), pp. 651–694.

[26] G. W. Stewart, *Matrix Algorithms II: Eigensystems*, Society for Industrial and Applied Mathematics, Philadelphia, PA, 2001.

[27] M. Streiff, A. Witzig, and W. Fichtner, *Computing optical modes for VCSEL device simulation*, Tech. Report 2002/1, ETH Zürich, Integrated Systems Laboratory, Jan. 2002.

[28] H. A. van der Vorst and J. B. M. Melissen, *A Petrov-Galerkin type method for solving $Ax = b$, where A is symmetric complex*, IEEE Trans. Magnetics, 26 (1990), pp. 706–708.