LARGE-SCALE COMPUTATION OF PSEUDOSPECTRA USING ARPACK AND EIGS[*]

THOMAS G. WRIGHT[†] AND LLOYD N. TREFETHEN[†]

Abstract. ARPACK and its MATLAB counterpart, `eigs`, are software packages that calculate some eigenvalues of a large nonsymmetric matrix by Arnoldi iteration with implicit restarts. We show that at a small additional cost, which diminishes relatively as the matrix dimension increases, good estimates of pseudospectra in addition to eigenvalues can be obtained as a by-product. Thus in large-scale eigenvalue calculations it is feasible to obtain routinely not just eigenvalue approximations, but also information as to whether or not the eigenvalues are likely to be physically significant. Examples are presented for matrices with dimension up to 200,000.

Key words. Arnoldi, ARPACK, eigenvalues, implicit restarting, pseudospectra

AMS subject classifications. 65F15, 65F30, 65F50

PII. S106482750037322X

1. Introduction. The matrices in many eigenvalue problems are too large to allow direct computation of their full spectra, and two of the iterative tools available for computing a part of the spectrum are ARPACK [10, 11] and its MATLAB counterpart, `eigs`.[1] For nonsymmetric matrices, the mathematical basis of these packages is the Arnoldi iteration with implicit restarting [11, 23], which works by compressing the matrix to an "interesting" Hessenberg matrix, one which contains information about the eigenvalues and eigenvectors of interest. For general information on large-scale nonsymmetric matrix eigenvalue iterations, see [2, 21, 29, 31].

For some matrices, nonnormality (nonorthogonality of the eigenvectors) may be physically important [30]. In extreme cases, nonnormality combined with the practical limits of machine precision can lead to difficulties in accurately finding the eigenvalues. Perhaps the more common and more important situation is when the nonnormality is pronounced enough to limit the physical significance of eigenvalues for applications, without rendering them uncomputable. In applications, users need to know if they are in such a situation. The prevailing practice in large-scale eigenvalue calculations is that users get no information of this kind.

There is a familiar tool available for learning more about the cases in which nonnormality may be important: pseudospectra. Figure 1 shows some of the pseudospectra of the "Grcar matrix" of dimension 400 [6], the exact spectrum, and converged eigenvalue estimates (Ritz values) returned by a run of ARPACK (seeking the eigenvalues of largest modulus) for this matrix. In the original article [23] that described the algorithmic basis of ARPACK, Sorensen presented some similar plots of difficulties encountered with the Grcar matrix. This is an extreme example where the nonnormality is so pronounced that even with the convergence tolerance set to its lowest possible value, machine epsilon, the eigenvalue estimates are far from the true spectrum. From the Ritz values alone, one might not realize that anything was

---

[1]In MATLAB version 5, `eigs` was an M-file adapted from the Fortran ARPACK codes. Starting with MATLAB version 6, the `eigs` command calls the Fortran ARPACK routines themselves.
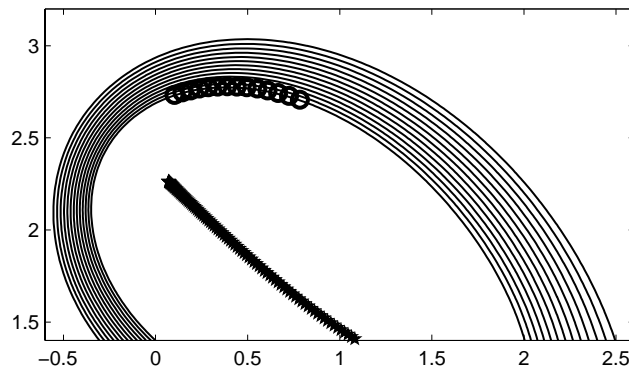
FIG. 1. *The ϵ-pseudospectra for $\epsilon = 10^{-5}, 10^{-6}, \ldots, 10^{-17}$ of the Grcar matrix (dimension 400), with the actual eigenvalues shown as solid stars and the converged eigenvalue estimates (for the eigenvalues of largest modulus) returned by ARPACK shown as open circles. The ARPACK estimates lie between the 10^{-16} and 10^{-17} pseudospectral contours.*

amiss. Once the pseudospectra are plotted too, it is obvious.

Computing the pseudospectra of a matrix of dimension N is traditionally an expensive task, requiring an $\mathcal{O}(N^3)$ singular value decomposition at each point in a grid. For a reasonably fine mesh, this leads to an $\mathcal{O}(N^3)$ algorithm with the constant of the order of thousands. Recent developments in algorithms for computing pseudospectra have improved the constant [28], and the asymptotic complexity for large sparse matrices [3, 12], but these are still fairly costly techniques. In this paper we show that for large matrices, we can cheaply compute an approximation to the pseudospectra in a region near the interesting eigenvalues. Our method uses the upper Hessenberg matrix constructed after successive iterations of the implicitly restarted Arnoldi algorithm, as implemented in ARPACK. Among other things, this means that after performing an eigenvalue computation with ARPACK or `eigs`, a user can quickly obtain a graphical check to indicate whether the Ritz values returned are likely to be physically meaningful. Our vision is that every ARPACK or `eigs` user ought to plot pseudospectra estimates routinely after their eigenvalue computations as a cheap "sanity check."

Some ideas related to ours have appeared in earlier papers by Nachtigal, Reichel, and Trefethen [17], Ruhe [19], Sorensen [23], Toh [24], and Toh and Trefethen [25]. For example, Sorensen plotted level curves of filter polynomials and observed that they sometimes approximated pseudospectra, and Ruhe showed that pseudospectra could be approximated by a rational Krylov method. What is new here is the explicit development of a method for approximating pseudospectra based on ARPACK. Of course, one could also consider the use of different low-dimensional compressions of a matrix problem such as those constructed by the Jacobi–Davidson algorithm [22]. Preliminary experiments, not reported here, show that this kind of Jacobi–Davidson approximation of pseudospectra can also be effective.

We start by giving an overview of pseudospectra calculations and the implicitly restarted Arnoldi iteration, followed by the practical details of our implementation along with a discussion of some of the problems we have had to deal with. After this we give some examples of the technique in practice. We also mention our MATLAB graphical user interface (GUI), which automates the computation of pseudospectra

after the eigenvalues of a matrix have been computed by `eigs` in MATLAB.

The computations presented in this paper were all performed using `eigs` in MAT-LAB version 6 (which essentially is ARPACK) and our GUI rather than the Fortran ARPACK, although our initial experiments were done with the Fortran code.

2. Pseudospectra. There are several equivalent ways of defining $\Lambda_\epsilon(A)$, the ϵ-pseudospectrum of a matrix A. The two most important (see, e.g., [27]) are perhaps

$$(2.1) \qquad \Lambda_\epsilon(A) = \{z \in \mathbb{C} : \|(zI - A)^{-1}\| \geq \epsilon^{-1}\}$$

and

$$(2.2) \qquad \Lambda_\epsilon(A) = \{z \in \mathbb{C} : z \in \Lambda(A + E) \text{ for some } E \text{ with } \|E\| \leq \epsilon\}.$$

When the norms are taken to be the 2-norm, the definitions are equivalent to

$$(2.3) \qquad \Lambda_\epsilon(A) = \{z \in \mathbb{C} : \sigma_{\min}(zI - A) \leq \epsilon\},$$

where $\sigma_{\min}(\cdot)$ denotes minimum singular value. This provides the basis of many algorithms for computing pseudospectra. The most familiar technique is to use a grid over the region of the complex plane of interest and calculate the minimum singular value of $zI - A$ at each grid point z. These values can then be passed to a contour plotter to draw the level curves. For the rest of this paper, we consider the 2-norm; other norms are discussed in [8, 28].

The reason for the cost of computation of pseudospectra is now clear: the amount of work needed to compute the minimum singular value of a general matrix of dimension N is $\mathcal{O}(N^3)$ (see, e.g., [5]). However, several techniques have been developed to reduce this cost [28]. Here are two important ones.

(I) Project the matrix onto a lower dimensional invariant subspace via, e.g., a partial Schur factorization (Reddy, Schmid, and Henningson [18]). This works well when the interesting portion of the complex plane excludes a large fraction of the eigenvalues of the matrix. In this case, the effect of the omitted eigenvalues on the interesting portion of the pseudospectra is typically small, especially if the undesired eigenvalues are well conditioned. Projection can significantly reduce the size of the matrix whose pseudospectra we need to compute, making the singular value computation dramatically faster. In general, the additional cost of projecting the matrix is much less than the cost of repeatedly computing the smallest singular value for the shifted original matrix.

(II) Perform a single matrix reduction to Hessenberg or triangular form before doing any singular value decompositions (Lui [12]), allowing the singular value calculations to be done using a more efficient algorithm.

One way of combining these ideas is to do a complete Schur decomposition of the matrix, $A = UTU^*$, and then to reorder the diagonal entries of the triangular matrix to leave the "wanted" eigenvalues at the top. The reordered factorization can then be truncated leaving the required partial Schur factorization. We can now find the singular values of the matrices shifted for each grid point z using either the original matrix A or the triangular matrix T:

$$(2.4) \qquad \sigma(zI - A) = \sigma(zUIU^* - UTU^*) = \sigma(U(zI - T)U^*) = \sigma(zI - T).$$

This allows us to work solely with the triangular matrix T once the $\mathcal{O}(N^3)$ factorization has been completed. The minimum singular value of $zI - T$ can be determined

in $\mathcal{O}(N^2)$ operations[2] using the fact that $\sigma_{\min}(zI - T) = \sqrt{\lambda_{\min}((zI - T)^*(zI - T))}$. This can be calculated using either inverse iteration or inverse Lanczos iteration, which require solutions to systems of equations with the matrix $(zI - T)^*(zI - T)$. These can be solved in two stages, each using triangular system solves.

By combining these techniques with more subtle refinements we have an algorithm which is much more efficient than the straightforward method. It is suggested in [28] that the speedup obtained is typically a factor of about $N/4$, assuming the cost of the Schur decomposition is negligible compared with that of the rest of the algorithm. This will be the case on a fine grid for a relatively small matrix (N of the order of a thousand or less), but for larger matrices the Schur decomposition is relatively expensive, and it destroys any sparsity structure.

3. Arnoldi iteration. The Arnoldi iteration for a matrix A of dimension N works by projecting A onto successive Krylov subspaces $\mathcal{K}_j(A, v_1) \subseteq \mathbb{C}^N$ of dimension j for some starting vector v_1 [1, 20]. It builds an orthonormal basis for the Krylov subspace by the Arnoldi factorization

$$(3.1) \qquad AV_j \;=\; V_j H_j + f_j e_j^* \;=\; V_{j+1} \widetilde{H}_j,$$

where H_j is an upper Hessenberg matrix of dimension j, the columns of V_j form an orthonormal basis for the Krylov subspace, and f_j is orthogonal to the columns of V_j. The residual term $f_j e_j^*$ can be incorporated into the first term $V_j H_j$ by augmenting the matrix H_j with an extra row, all zeros except for the last entry, which is $\|f_j\|$, and including the next basis vector, $v_{j+1} = f_j/\|f_j\|$, in the matrix V_j. The matrix $\widetilde{H}_j$ is now rectangular, of size $(j + 1) \times j$.

The matrix $\widetilde{H}_j$, being rectangular, does not have any eigenvalues, but we can define its pseudospectra in terms of singular values by (2.3). (With $\epsilon = 0$, we recover the definition occasionally used that λ is an eigenvalue of a rectangular matrix A if $A - \lambda I$ is rank-deficient, where I is the rectangular matrix of appropriate dimension with 1 on the diagonal and 0 elsewhere. In general, a rectangular matrix will have no eigenvalues, but it will have nonempty ϵ-pseudospectra for large enough ϵ.) It can then be shown [14, 25] that the pseudospectra of successive $\widetilde{H}_j$ are nested.

THEOREM 3.1. *Let A be an $N \times N$ matrix which is unitarily similar to a Hessenberg matrix H, and let $\widetilde{H}_j$ denote the upper left $(j + 1) \times j$ section (in particular $\widetilde{H}_j$ could be created using a restarted Arnoldi iteration). Then for any $\epsilon \geq 0$,*

$$(3.2) \qquad \Lambda_\epsilon(\widetilde{H}_1) \subseteq \Lambda_\epsilon(\widetilde{H}_2) \subseteq \cdots \subseteq \Lambda_\epsilon(A).$$

Thus as the iteration progresses, the pseudospectra of the rectangular Hessenberg matrices better approximate those of A, which gives some justification for the approximation $\Lambda_\epsilon(A) \approx \Lambda_\epsilon(\widetilde{H}_j)$. Unfortunately, this is only the case for the rectangular matrices $\widetilde{H}_j$. There do not appear to be any satisfactory theorems to justify a similar approximation for the square matrices H_j, and of course for ϵ sufficiently small the ϵ-pseudospectra of H_j must be disjoint from those of A, since they will be small sets surrounding the eigenvalues of H_j, which are in general distinct from those of A. This is not the case for the rectangular matrix as there will not be points in the complex plane with infinite resolvent norm unless a Ritz value exactly matches an

---

[2]An $\mathcal{O}(N^2)$ algorithm can also be used for Hessenberg matrices [12], but for those we do not have the advantage of projection.

eigenvalue of the original matrix. That is, $\Lambda_\epsilon(\widetilde{H}_j)$ is typically empty for sufficiently small ϵ.

Although the property (3.2) is encouraging, theorems guaranteeing rapid convergence in all cases cannot be expected. The quality of the approximate pseudospectra depends on the information in the Krylov subspace, which in turn depends on the starting vector v_1. Any guarantee of rapid convergence could at best be probabilistic.

3.1. Implicitly restarted Arnoldi. In its basic form, the Arnoldi process may require close to N iterations before the subspace contains good information about the eigenvalues of interest. However, the information contained within the Hessenberg matrix is very dependent on the starting vector v_1: if v_1 contains relatively small components of the eigenvectors corresponding to the eigenvalues which are *not* required, convergence may be quicker and the subspace size need not grow large. To avoid the size of the subspace growing too large, practical implementations of the Arnoldi iteration restart when the subspace size j reaches a certain threshold [20]. A new starting vector $\hat{v}_1$ is chosen which has smaller components in the directions of eigenvectors corresponding to unwanted eigenvalues, and the process is begun again.

Implicit restarting [11, 23] is based upon the same idea, except that subspace is only implicitly compressed to a single starting vector $\hat{v}_1$. What is explicitly formed is an Arnoldi factorization of size k based on this new starting vector, where k is the number of desired eigenvalues, and this Arnoldi factorization is obtained by carrying out $p - k$ implicitly shifted steps of the QR algorithm, with shifts possibly corresponding to unwanted eigenvalue estimates. The computation now proceeds in an accordion-like manner, expanding the subspace to its maximum size p, then compressing to a smaller subspace.[3] This is computationally more efficient than simple restarting because the subspace is already of size k when the iteration restarts, and in addition, the process is numerically stable due to the use of orthogonal transformations in performing the restarting. This technique has made the Arnoldi iteration competitive for finding exterior eigenvalues of a wide range of nonsymmetric matrices.

3.2. Arnoldi for pseudospectra. In a 1996 paper, Toh and Trefethen [25] demonstrated that the Hessenberg matrix created during the Arnoldi process can sometimes provide a good approximation to the pseudospectra of the original matrix. They provided results for both the square matrix H_j and the rectangular matrix $\widetilde{H}_j$. We choose to build our method around the rectangular Hessenberg matrices $\widetilde{H}_j$, even though this makes the pseudospectral computation harder than if we worked with the square matrix. The advantage of this is that we retain the properties of Theorem 3.1, and the following in particular:

For every $\epsilon \geq 0$, the approximate ϵ-pseudospectrum generated by our ARPACK algorithm is a subset of the ϵ-pseudospectrum of the original matrix,

$$(3.3) \qquad\qquad \Lambda_\epsilon(\widetilde{H}_p) \subseteq \Lambda_\epsilon(A).$$

This is completely different from the familiar situation with Ritz values, which are, after all, the points in the 0-pseudospectrum of a square Hessenberg matrix. Ritz values need not be contained in the true spectrum. Simply by adding one more row to consider a rectangular matrix, we have obtained a guaranteed inclusion for every ϵ.

The results presented by Toh and Trefethen focus on trying to approximate the full pseudospectra of the matrix (i.e., around the entire spectrum) and they do not use

---

[3] Our use of the variable p follows `eigs`. In ARPACK and in Sorensen's original paper [23], this would be $p + k$.

any kind of restarting in their implementation of the Arnoldi iteration. While this is a useful theoretical idea, we think it is of limited practical value for computing highly accurate pseudospectra since good approximations are often obtained generally only for large subspace dimensions.

Our work is more local; we want a good approximation to the pseudospectra in the region around the eigenvalues requested from ARPACK or `eigs`. By taking advantage of ARPACK's implicit restarting, we keep the size of the subspace (and hence $\widetilde{H}_p$) reasonably small, allowing us to compute (local) approximations to the pseudospectra more rapidly, extending the idea of [25] to a fully practical technique (for a more restricted problem).

4. Implementation. In deciding to use the rectangular Hessenberg matrix $\widetilde{H}_j$, we have made the post-ARPACK phase of our algorithm more difficult. While the simple algorithm of computing the minimum singular value of $zI - A$ at each point has approximately the same cost for a rectangular matrix as a square one, the speedup techniques described in section 2 are difficult to translate into the rectangular case.

The first idea, projection to a lower dimensional invariant subspace, does not make sense for rectangular matrices because there is no such thing as an invariant subspace. The second idea, preliminary triangularization using a Schur decomposition, also does not extend to rectangular matrices, for although it is possible to triangularize the rectangular matrix while keeping the same singular values (by performing a QR factorization, for example), doing so destroys the vital property of shift-invariance (see (2.4)).

However, our particular problem has a feature we have not yet considered: the matrix is Hessenberg. One way to exploit this property is to perform a QR factorization of the matrix obtained after shifting for each grid point. The upper triangular matrix R has the same singular values as the shifted matrix, and they are also unchanged on removing the last row of zeros, which makes the matrix square. We can now use the inverse Lanczos iteration as in section 2 to find its smallest singular value. The QR factorization can be done with an $\mathcal{O}(N^2)$ algorithm (see, e.g., [5, p. 228]), which makes the overall cost $\mathcal{O}(N^2)$. Unfortunately, the additional cost of the QR factorization at each stage makes this algorithm slightly slower for the small matrices (dimensions 50–150) output from ARPACK than for square matrices of the same size, but this appears to be the price to be paid for using matrices which have the property of (3.3).

4.1. Refinements. In some cases we have found that inverse iteration to find the minimum eigenvalue of $(zI - R)^*(zI - R)$ is more efficient than inverse Lanczos iteration but only when used with continuation (Lui [12]). Continuation works by using the vector corresponding to the smallest singular value from the previous grid point as the starting guess for the next grid point.

This sounds like a good idea; if the two shifted matrices differ by only a small shift, their singular values (and singular vectors) will be similar. When it works, it generally means that only a single iteration is needed to satisfy the convergence criterion. However, as Lui indicates, there is a problem with this approach if the smallest and second smallest singular values "change places" between two values of z: the iteration may converge to the second smallest singular value instead of the smallest, since the starting vector had such a strong component in the direction of the corresponding singular vector. This leads to the convergence criterion being satisfied for the wrong singular value (even after several iterations).

Choose max subspace size p—larger p for better pseudospectra.

Choose number of eigenvalues k—larger k for better pseudospectra.

Run ARPACK(A, p, k) to obtain $\widetilde{H}_p$.

Define a grid over a region of $\mathbb{C}$ enclosing converged Ritz values.

For each grid point z:

    Perform reduced QR factorization of shifted matrix: $z\tilde{I} - \widetilde{H}_p = \widetilde{Q}_p R$.

    Get $\lambda_{\max}(z)$ from Lanczos iteration on $(R^*R)^{-1}$, random starting vector.

    $\sigma_{\min}(z) := 1/\sqrt{\lambda_{\max}(z)}.$

end.

Start GUI and create contour plot of the $\sigma_{\min}$ values.

Allow adjustment of parameters (e.g., grid size, contour levels) in GUI.

FIG. 2. *Pseudocode for our algorithm.*

In the course of our research, we have found several test matrices which suffer from this problem, including the so-called Tolosa matrix [4]. Accordingly, because of our desire to create a robust algorithm, we do not use inverse iteration. In theory it is also possible to use continuation with inverse Lanczos iteration, but our experiments indicate that the benefit is small and it again brings a risk of misconvergence.

Our algorithm (the main loop of which is similar to that in [28]) is summarized in Figure 2.

5. Practical examples. While one aim of our method is to approximate the pseudospectra of the original matrix accurately, this is perhaps no more important than the more basic mission of exhibiting the degree of nonnormality the matrix has, so that the ARPACK or `eigs` user gets some idea of whether the Ritz values returned are likely to be physically meaningful. Even in cases where the approximations of the sets $\Lambda_\epsilon(A)$ are inaccurate, a great deal may still be learned from their qualitative properties.

In the following examples, ARPACK was asked to look for the eigenvalues of largest real part except where otherwise indicated. However, the choice of region of the complex plane to focus on is unimportant for our results and is determined by which eigenvalues are of interest for the particular problem at hand. The number of requested eigenvalues k was chosen rather arbitrarily to be large enough so that the approximate pseudospectra clearly indicate the true behavior in the region of the complex plane shown, and the maximum subspace size p was chosen to ensure convergence of ARPACK for the particular choice of k. Experiments show that the computed pseudospectra are not very sensitive to the choices of k and p, provided they are large enough, but we have not attempted to optimize these choices.

5.1. Two extremes. Our first example (Figure 3), from Matrix Market [15], shows a case where the approximation is extremely good. The matrix is the Jacobi matrix of dimension 800 for the reaction-diffusion Brusselator model from chemical engineering [7], and one seeks the rightmost eigenvalues. The matrix is not highly non-normal, and the pseudospectra given by the approximation almost exactly match the
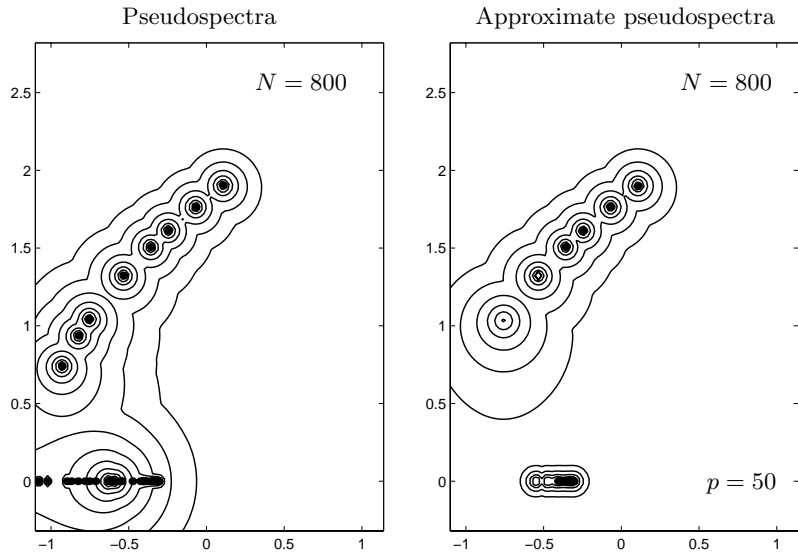
Pseudospectra　　　　　　　　　　Approximate pseudospectra

FIG. 3. *Pseudospectra for the matrix* `rdb800l` *(left) computed using the standard method, and pseudospectra of the upper Hessenberg matrix of dimension $p = 50$ computed using ARPACK (right) in about 9% of the computer time (on the fine grid used here). Levels are shown for $\epsilon = 10^{-1}, 10^{-1.2}, \ldots, 10^{-2.4}$. The number of matrix-vector products needed by ARPACK (n_v) is 1,493.*

true pseudospectra around the converged Ritz values. This is a case where the pseudospectra computed after running ARPACK indicate that the eigenvalues returned are both accurate and physically meaningful, and that no further investigation is necessary. In this computation we used a maximum subspace dimension of $p = 50$ and requested $k = 30$ eigenvalues.

The second case we consider is one where the matrix has a high degree of nonnormality—the Grcar matrix. As seen in Figure 1, ARPACK can converge to Ritz values which are eigenvalues of a perturbation of order machine precision of the original matrix, and the nonnormality of this particular matrix (here of dimension 400) means that the Ritz values found can lie a long way from the spectrum of the matrix. Figure 4 shows that the pseudospectra of the Hessenberg matrix (computed using $p = 50$, $k = 45$, and asking for eigenvalues of largest modulus) in this case are not good approximations to the pseudospectra of the original one.

This is typical for highly nonnormal matrices—the Hessenberg matrix cannot capture the full extent of the nonnormality, particularly when more than p eigenvalues of the original matrix lie within the region of the complex plane in which the pseudospectra are computed. In other words, the approximation is typically not so good in areas away from the Ritz values computed, and then only accurately approximates the pseudospectra of the original matrix when the Ritz values are good approximations to the eigenvalues. Despite this, a plot like that of Figure 4 will instantly indicate to the ARPACK user that the matrix at hand is strongly nonnormal and needs further investigation.

5.2. A moderately nonnormal example. While the above examples show two extreme cases, many important applications are more middle-of-the-range, where
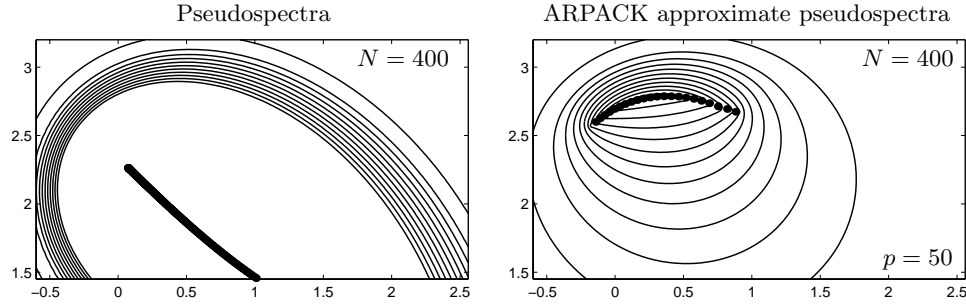
Pseudospectra   ARPACK approximate pseudospectra

FIG. 4. *The pseudospectra of the Grcar matrix of dimension* 400 *(left) computed using the standard method, and the pseudospectra of the upper Hessenberg matrix of dimension* 50 *computed using ARPACK (right) in about* 8% *of the computer time (on this fine grid). Contours are shown for $\epsilon = 10^{-1}, 10^{-2}, \ldots, 10^{-11}$, and $n_v = 8{,}284$.*

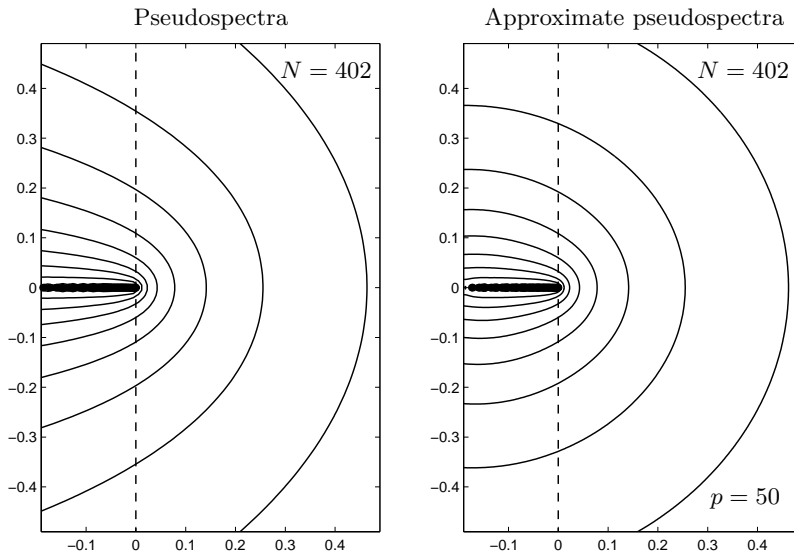Pseudospectra   Approximate pseudospectra

FIG. 5. *Pseudospectra for $\epsilon = 10^{-1}, 10^{-1.5}, \ldots, 10^{-5}$ for linearized fluid flow through a circular pipe at Reynolds number* 10,000 *(streamwise-independent disturbances with azimuthal wave number* 1*), $n_v = 708{,}678$.*

machine precision is sufficient to accurately converge the eigenvalues, but pronounced nonnormality may nevertheless diminish the physical significance of some of them. A good example of a case in which this is important is the matrix created by linearization about the laminar solution of the Navier–Stokes equations for fluid flow in an infinite circular pipe [26]. (Our matrix is obtained by a Petrov–Galerkin spectral discretization of the Navier–Stokes problem due to Meseguer and Trefethen [16]. The axial and azimuthal wave numbers are 0 and 1, respectively, and the matrix dimension is 402.) The pseudospectra are shown in Figure 5, and although the eigenvalues all have negative real part, implying stability of the flow, the pseudospectra protrude far into the right half-plane. This implies pronounced transient growth of some perturbations of the velocity field in the pipe, which in the presence of nonlinearities in practice may lead to transition to turbulence [30]. The approximate pseudospectra also highlight
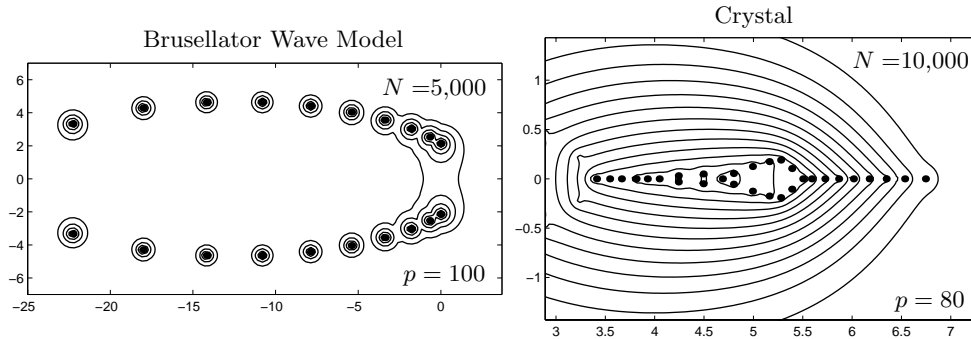
FIG. 6. *Left: The $\epsilon = 10^{-0.2}, 10^{-0.4}, \ldots, 10^{-1.6}$ pseudospectra for the Brusselator wave model, $n_v = 16,906$. Right: Pseudospectral contours for $\epsilon = 10^{-1}, 10^{-2}, \ldots, 10^{-13}$ for a matrix of dimension 10,000 from the Crystal set at Matrix Market, $n_v = 22,968$.*

this behavior. The parameters used here were $p = 50$ and $k = 25$.

5.3. Larger examples. We now consider four larger examples. The first is the Brusselator wave model from Matrix Market (not to be confused with the very first example), which models the concentration waves for reaction and transport interaction of chemical solutions in a tubular reactor [9]. Stable periodic solutions exist for a parameter when the rightmost eigenvalues of the Jacobian are purely imaginary. For a matrix of dimension 5,000, using a subspace of dimension 100 and asking ARPACK for 20 eigenvalues, we obtained the eigenvalue estimates and approximate pseudospectra shown in Figure 6 (left). The departure from normality is evidently mild, and the conclusion from this computation is that the Ritz values returned by ARPACK are likely to be accurate and the corresponding eigenvalues physically meaningful.

Figure 6 (right) shows approximate pseudospectra for a matrix of dimension 10,000, taken from the Crystal set at Matrix Market, which arises in a stability analysis of a crystal growth problem [32]. The eigenvalues of interest are the ones with largest real part. The fact that we can see the 10^{-13} pseudospectrum (when the axis scale is $\mathcal{O}(1)$) indicates that this matrix is significantly nonnormal, and although the matrix is too large for us to be able to compute its exact pseudospectra for comparison, this is certainly a case where the nonnormality could be important, making all but the rightmost few eigenvalues of dubious physical significance in an application. The ARPACK parameters we used in this case were $p = 80$ and $k = 30$, and the computation took about one hour on our Sun Ultra 5 workstation. Although we do not have the true pseudospectra in this case, we would expect that the rightmost portion should be fairly accurate where there is a good deal of Ritz data and relatively little nonnormality. We expect that the leftmost portion is less accurate where the effect of the remaining eigenvalues of the matrix unknown to the approximation begins to become important.

The third example, Figure 7 (left), shows the Airfoil matrix created by performing transient stability analysis of a Navier–Stokes solver [13], also from Matrix Market. In this case the matrix appears fairly close to normal, and the picture gives every reason to believe that the eigenvalues have physical meaning. Using $p = 80$ and $k = 20$, ARPACK took about 9 hours to converge to the eigenvalues, while we were able to plot the pseudospectra in about 3 minutes (even on the fine grid used here).

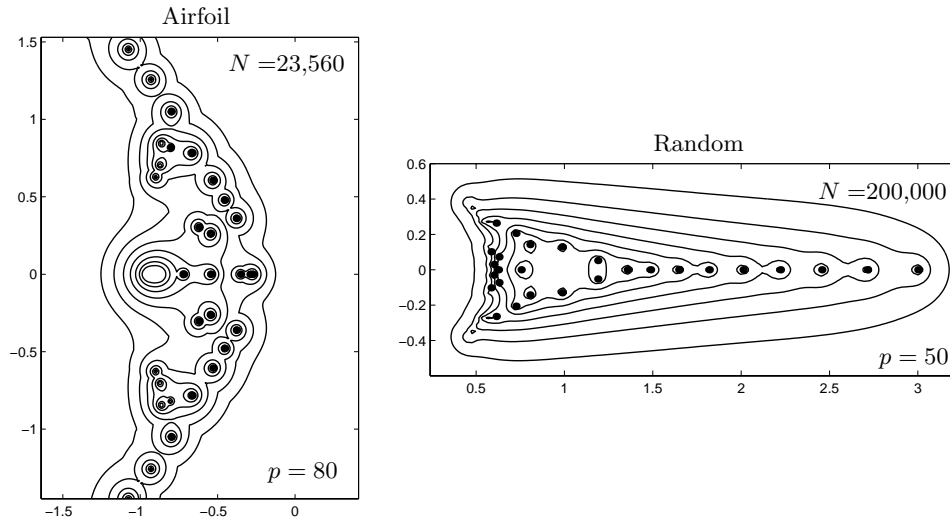Our final example is a matrix which is bidiagonal plus random sparse entries

Airfoil

Random

FIG. 7.   *Left: The $\epsilon = 10^{-1}, 10^{-1.25}, \ldots, 10^{-2.5}$ pseudospectra for the Airfoil matrix from Matrix Market of dimension 23,560, $n_v = 72,853$. Right: The $\epsilon = 10^{-1}, 10^{-1.5}, \ldots, 10^{-4.5}$ pseudospectra for our random matrix, with $n_v = 61,347$.*

elsewhere, created in MATLAB by

```
A = spdiags([3*exp(-(0:N-1)'/10) .5*ones(N,1)], 0:1, N, N)
       + .1*sprandn(N,N,10/N);
```

The approximation to the pseudospectra of the matrix of dimension 200,000 is shown in Figure 7 (right), from which we can conclude that the eigenvalue estimates returned are probably accurate, but that the eigenvalues toward the left of the plot would likely be of limited physical significance in a physical application, if there were one, governed by this matrix. We used a subspace size of 50 and requested 30 eigenvalues from this example, and the whole computation took about 26 hours.

6. MATLAB GUI. We have created a MATLAB GUI to automate the process of computing pseudospectra, and Figure 8 shows a snapshot after a run of ARPACK. Initially the pseudospectra are computed on a coarse grid to give a fast indication of the nonnormality of the matrix, but the GUI allows control over the number of grid points if a higher quality picture is desired. Other features include the abilities to change the contour levels shown without recomputing the underlying values, and to select parts of the complex plane to zoom in for greater detail. The GUI can also be used as a graphical front end to our other pseudospectra codes for computing pseudospectra of smaller general matrices. The codes are available on the World Wide Web from http://web.comlab.ox.ac.uk/oucl/work/nick.trefethen/.

7. Discussion. The examples of section 5 give an indication of the sort of pictures obtained from our technique. For matrices fairly close to normal, the approximation is typically a very close match to the exact pseudospectra, but for more highly nonnormal examples the agreement is not so close. This is mainly due to the effect of eigenvalues which the Arnoldi iteration has not found: their effect on the pseudospectra is typically more pronounced for nonnormal matrices.

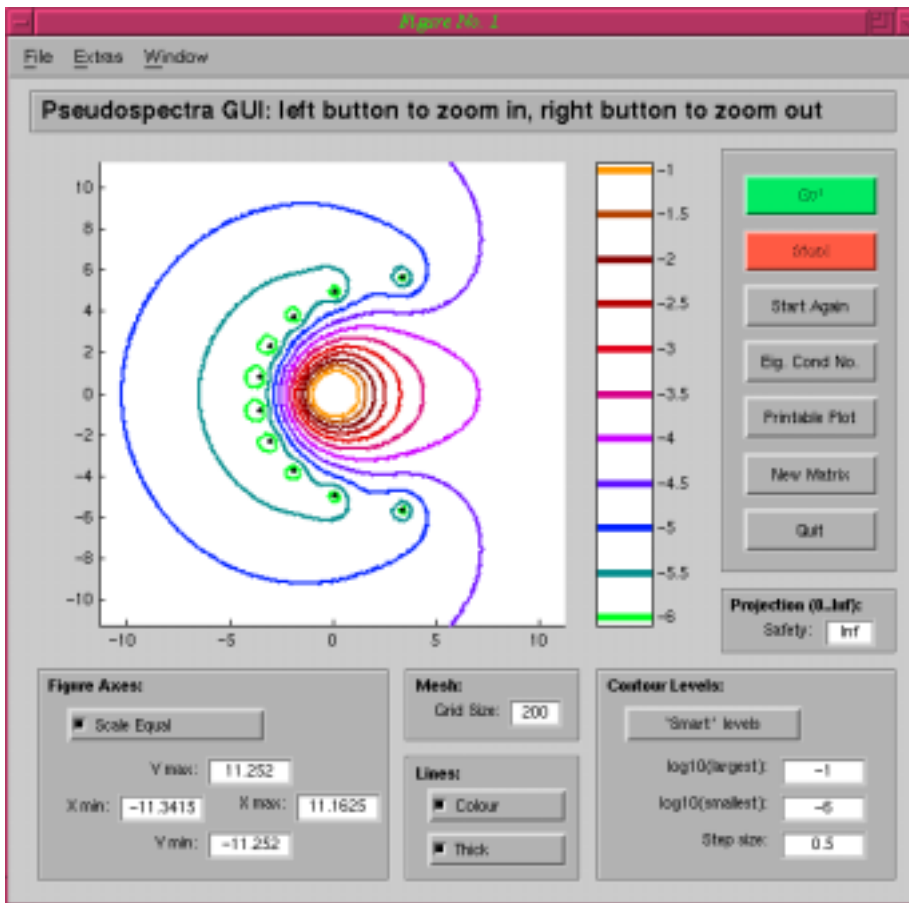The other point to note is that if we use the Arnoldi iteration to look (for ex-

FIG. 8. *A snapshot of the* MATLAB *GUI after computing the pseudospectra of a matrix.*

ample) for eigenvalues of largest real part, the rightmost part of the approximate pseudospectra will be a reasonably good approximation. This can clearly be seen in Figure 4 where we are looking for eigenvalues of largest modulus: the top parts of the pseudospectra are fairly good and only deteriorate lower down where the effect of the "unfound" eigenvalues becomes important.

However, as mentioned in the introduction, creating accurate approximations of pseudospectra was only part of the motivation for this work. Equally important has been the goal of providing information which can help the user of ARPACK or `eigs` decide whether the computed eigenvalues are physically meaningful. For this purpose, estimating the degree of nonnormality of the matrix is more important than getting an accurate plot of the exact pseudospectra.

One of the biggest advantages of our technique is that while the time spent on the ARPACK computation grows as the dimension of the matrix increases, the time spent on the pseudospectra computation remains roughly constant. This is because the pseudospectra computation is based just on the final Hessenberg matrix, of dimension typically in the low hundreds at most. Figure 9 shows the proportion of time spent on the pseudospectra part of the computation for the examples we have presented here. These timings are based on the time to compute the initial output from our
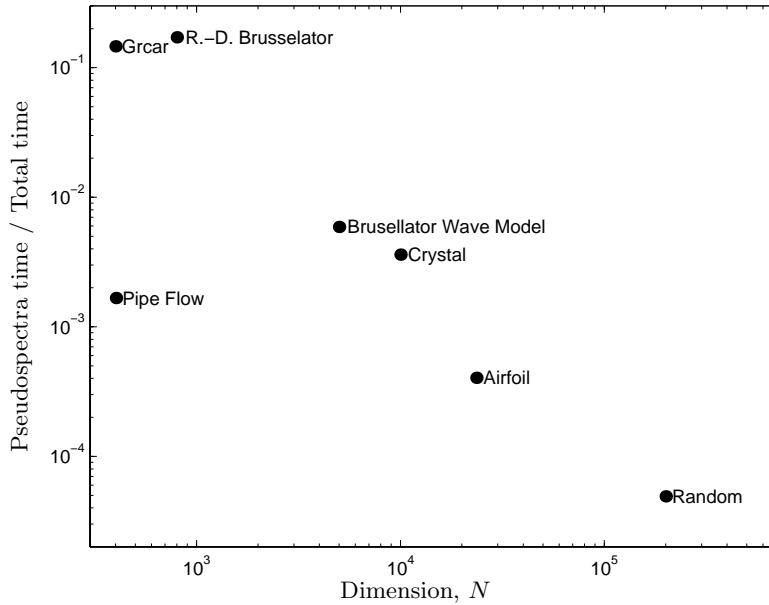
FIG. 9. *The proportion of the total computation time spent on computing the pseudospectra for the examples presented in this paper. For large N, the pseudospectra are obtained at very little additional cost.*
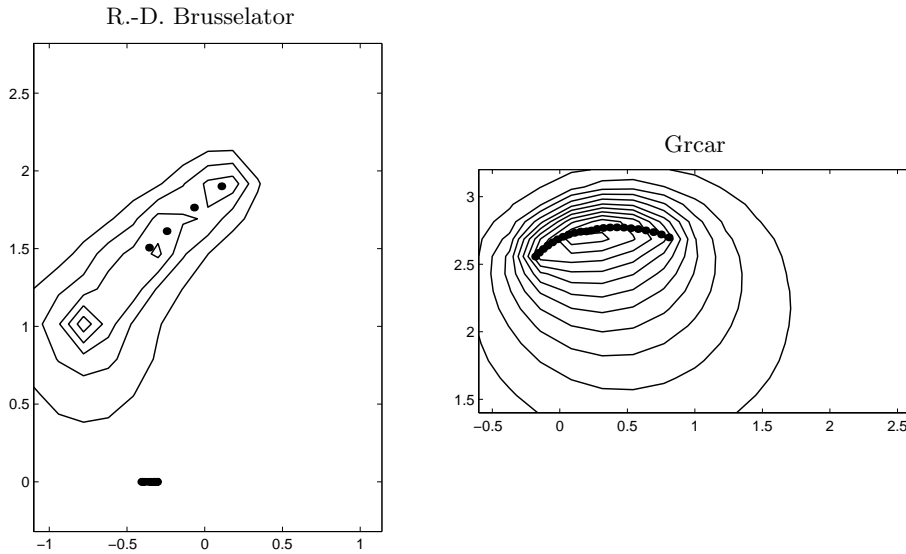

FIG. 10. *Lower-quality plots of pseudospectra produced by our GUI on its default coarse 15 × 15 grid. Such plots take just a few seconds. The matrices shown are* `rdb800l` *with $\epsilon = 10^{-1}, 10^{-1.2}, \ldots, 10^{-1.8}$ (cf. Figure 3) and the Grcar matrix with $\epsilon = 10^{-1}, 10^{-2}, \ldots, 10^{-10}$ (cf. Figure 4).*

GUI using a coarse grid such as those illustrated in Figure 10, which is our standard resolution for day-to-day work. For the "publication quality" pseudospectra of the resolution of the other plots in this paper, the cost is about thirty times higher, but

this is still much less than the cost of ARPACK for dimensions N in the thousands.

Acknowledgments. We would like to thank Rich Lehoucq for his advice on ARPACK beginning during his visit to Oxford in October–November 1999, Penny Anderson of The MathWorks, Inc., for her help with the beta version of the new `eigs` command, and Mark Embree for his comments on drafts of the paper.

REFERENCES

[1] W. E. ARNOLDI, *The principle of minimized iteration in the solution of the matrix eigenvalue problem*, Quart. Appl. Math., 9 (1951), pp. 17–29.

[2] Z. BAI, J. DEMMEL, J. DONGARRA, A. RUHE, AND H. VAN DER VORST, EDS., *Templates for the Solution of Algebraic Eigenvalue Problems: A Practical Guide*, SIAM, Philadelphia, 2000.

[3] T. BRACONNIER AND N. J. HIGHAM, *Computing the field of values and pseudospectra using the Lanczos method with continuation*, BIT, 36 (1996), pp. 422–440.

[4] F. CHATELIN AND S. GODET-THOBIE, *Stability analysis in aeronautical industries*, in Proceedings of the 2nd Symposium on High-Performance Computing, Montpellier, France, M. Durand and F. E. Dabaghi, eds., Elsevier–North-Holland, Amsterdam, 1991.

[5] G. H. GOLUB AND C. F. VAN LOAN, *Matrix Computations*, 3rd ed., The Johns Hopkins University Press, Baltimore, MD, London, UK, 1996.

[6] J. F. GRCAR, *Operator Coefficient Methods for Linear Equations*, Tech. report SAND89-8691, Sandia National Labs, Livermore, CA, 1989.

[7] B. HASSARD, N. KAZARINOFF, AND Y. WAN, *Theory and Applications of Hopf Bifurcation*, Cambridge University Press, Cambridge, UK, 1981.

[8] N. J. HIGHAM AND F. TISSEUR, *A block algorithm for matrix 1-norm estimation, with an application to 1-norm pseudospectra*, SIAM J. Matrix Anal. Appl., 21 (2000), pp. 1185–1201.

[9] P. J. HOLMES, ED., *Waves in Distributed Chemical Systems: Experiments and Computations*, Proceedings of the Asilomar Conference Ground, Pacific Grove, CA, 1979, SIAM, Philadelphia, 1980.

[10] R. B. LEHOUCQ, K. MASCHHOFF, D. SORENSEN, AND C. YANG, *ARPACK Software Package*, http://www.caam.rice.edu/software/ARPACK/, 1996.

[11] R. B. LEHOUCQ, D. C. SORENSEN, AND C. YANG, *ARPACK Users' Guide*, SIAM, Philadelphia, 1998.

[12] S. H. LUI, *Computation of pseudospectra by continuation*, SIAM J. Sci. Comput., 18 (1997), pp. 565–573.

[13] A. MAHAJAN, E. H. DOWELL, AND D. BLIS, *Eigenvalue calculation procedure for an Euler-Navier-Stokes solver with applications to flows over airfoils*, J. Comput. Phys., 97 (1991), pp. 398–413.

[14] T. A. MANTEUFFEL AND G. STARKE, *On hybrid iterative methods for nonsymmetric systems of linear equations*, Numer. Math., 73 (1996), pp. 489–506.

[15] *Matrix Market*, http://math.nist.gov/MatrixMarket/, 2000.

[16] A. MESEGUER AND L. N. TREFETHEN, *A Spectral Petrov-Galerkin Formulation for Pipe Flow I: Linear Stability and Transient Growth*, Tech. report NA-00/18, Oxford University Computing Laboratory, Oxford, UK, 2000.

[17] N. M. NACHTIGAL, L. REICHEL, AND L. N. TREFETHEN, *A hybrid GMRES algorithm for nonsymmetric linear systems*, SIAM J. Matrix Anal. Appl., 13 (1992), pp. 796–825.

[18] S. C. REDDY, P. J. SCHMID, AND D. S. HENNINGSON, *Pseudospectra of the Orr–Sommerfeld operator*, SIAM J. Appl. Math., 53 (1993), pp. 15–47.

[19] A. RUHE, *Rational Krylov algorithms for nonsymmetric eigenvalue problems. II. Matrix pairs*, Linear Algebra Appl., 197/198 (1994), pp. 283–295.

[20] Y. SAAD, *Variations on Arnoldi's method for computing eigenelements of large unsymmetric matrices*, Linear Algebra Appl., 34 (1980), pp. 269–295.

[21] Y. SAAD, *Numerical Methods for Large Eigenvalue Problems*, Manchester University Press, Manchester, UK, 1992.

[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] D. C. SORENSEN, *Implicit application of polynomial filters in a k-step Arnoldi method*, SIAM J. Matrix Anal. Appl., 13 (1992), pp. 357–385.

[24] K.-C. TOH, *Matrix Approximation Problems and Nonsymmetric Iterative Methods*, Ph.D. thesis, Cornell University, Ithaca, NY, 1996.

[25] K.-C. TOH AND L. N. TREFETHEN, *Calculation of pseudospectra by the Arnoldi iteration*, SIAM J. Sci. Comput., 17 (1996), pp. 1–15.

[26] A. E. TREFETHEN, L. N. TREFETHEN, AND P. J. SCHMID, *Spectra and pseudospectra for pipe Poiseuille flow*, Comput. Methods Appl. Mech. Engrg., 175 (1999), pp. 413–420.

[27] L. N. TREFETHEN, *Pseudospectra of matrices*, in Numerical Analysis 1991 (Dundee, 1991), Longman Sci. Tech., Harlow, UK, 1992, pp. 234–266.

[28] L. N. TREFETHEN, *Computation of Pseudospectra*, Acta Numer. 8, Cambridge University Press, Cambridge, UK, 1999.

[29] L. N. TREFETHEN AND D. BAU, III, *Numerical Linear Algebra*, SIAM, Philadelphia, 1997.

[30] L. N. TREFETHEN, A. E. TREFETHEN, S. C. REDDY, AND T. A. DRISCOLL, *Hydrodynamic stability without eigenvalues*, Science, 261 (1993), pp. 578–584.

[31] H. A. VAN DER VORST, *Computational Methods for Large Eigenvalue Problems*, North-Holland, Amsterdam, to appear.

[32] C. YANG, D. C. SORENSEN, D. I. MEIRON, AND B. WEDEMAN, *Numerical Computation of the Linear Stability of the Diffusion Model for Crystal Growth Simulation*, Tech. report TR96-04, Department of Computational and Applied Mathematics, Rice University, Houston, TX, 1996.