MATHEMATICS OF COMPUTATION
VOLUME 42, NUMBER 166
APRIL 1984, PAGES 567–588

Chebyshev Acceleration Techniques for Solving Nonsymmetric Eigenvalue Problems*

By Youcef Saad

Abstract. The present paper deals with the problem of computing a few of the eigenvalues with *largest (or smallest) real parts*, of a large sparse nonsymmetric matrix. We present a general acceleration technique based on Chebyshev polynomials and discuss its practical application to Arnoldi's method and the subspace iteration method. The resulting algorithms are compared with the classical ones in a few experiments which exhibit a sharp superiority of the Arnoldi-Chebyshev approach.

1. Introduction. An important number of applications in applied sciences and engineering require the numerical solution of a large nonsymmetric matrix eigenvalue problem. Such is the case for example, in economical modeling [5], [16] where the stability of a model is interpreted in terms of the dominant eigenvalues of a large nonsymmetric matrix A. In Markov chain modeling of queueing networks [17], [18], [35], one is interested in an eigenvector associated with the eigenvalue unity of the transpose of a large nonsymmetric stochastic matrix. In structural engineering [6], [9], [10], [33], and in fluid mechanics [34] one often seeks to solve a bifurcation problem where a few of the eigenvalues of a family of nonsymmetric matrices $A(\alpha)$ are computed for several values of the parameter α in order to determine a critical value α_c such that some particular eigenvalue changes sign, or crosses the imaginary axis. When it is a pair of complex eigenvalues that crosses the imaginary axis, the bifurcation point α_c is referred to as a Hopf bifurcation point. This important problem was recently examined by Jepson [15] who proposes several techniques most of which deal with small dimension cases. Common bifurcation problems can be solved by computing *a few* of the eigenvalues *with largest real parts* of $A(\alpha)$ and then detecting when one of them changes sign. The study of stability of electrical networks is yet another interesting example requiring the numerical computation of the eigenvalue of largest real part. Finally, we can mention the occurrence of nonsymmetric generalized eigenvalue problems when solving the Riccati equations by the Schur techniques [20].

As suggested by the above important applications, we will primarily be concerned with the problem of computing a few of the eigenvalues with algebraically largest real parts, and their associated eigenvectors, of a large nonsymmetric matrix A. The literature in this area has been relatively limited as compared with that of the more

Received February 8, 1983; revised June 22, 1983.
1980 *Mathematics Subject Classification*. Primary 65F15.
* This work was supported in part by the U. S. Office of Naval Research under grant N000014-80-C-0276 and in part by NSF Grant MCS-8104874.

0025-5718/84 $1.00 + $.25 per page

common symmetric eigenvalue problem. The subspace iteration method [2], [4], [13], [38], [39], seems to have been the preferred algorithm for many years, and is still often recommended [12]. However, this algorithm computes the eigenvalues of largest modulus while the above-mentioned applications require those of algebraically largest (or smallest) real parts.

Furthermore, it is well known in the symmetric case that the Lanczos algorithm is far superior to the subspace iteration method [24]. Some numerical experiments described in [32] indicate that Krylov subspace based methods can be more effective in the nonsymmetric case as well. There are two known algorithms that generalize the symmetric Lanczos method to nonsymmetric matrices:

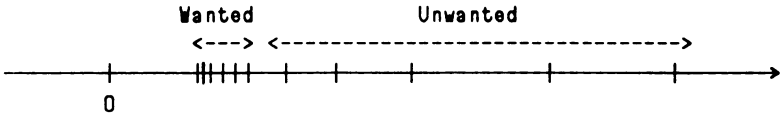
1. The Lanczos biorthogonalization algorithm [19];
2. Arnoldi's method [1].

The first method has been recently ressucitated by Parlett and Taylor [26], [40] who propose an interesting way of avoiding break-downs from which the method may otherwise suffer. The second was examined in [32] where several alternatives have been suggested and in [30] where some additional theory was established. With the appropriate initial vectors, both of these approaches reduce to the symmetric Lanczos algorithm when the matrix A is symmetric.

In the present paper we will describe a hybrid method based on the Chebyshev iteration algorithm and Arnoldi's method. These two methods taken alone face a number of limitations but, as will be seen, when combined they take full advantage of each other's attractive features.

The principle of Arnoldi's method is the following: start with an initial vector v_1 and at every step compute Av_i and orthogonalize it against all previous vectors to obtain v_{i+1}. At step m, this will build a basis $\{v_i\}_{i=1,m}$ of the Krylov subspace K_m spanned by $v_1, Av_1, \ldots, A^{m-1}v_1$. The restriction of A to K_m is then represented in the basis $\{v_i\}$ by a Hessenberg matrix whose elements are the coefficients used in the orthogonalization process. The eigenvalues of this Hessenberg matrix will provide approximations to the eigenvalues of A. Clearly, this simple procedure has the serious drawback of requiring the presence in memory of all previous vectors at a given step m. Also the amount of work increases drastically with the step number m. Several variations on this basic scheme have been suggested in [32] to overcome this difficulty, the most obvious of which is to use the method iteratively, i.e., to restart the process after every m steps. This alternative was shown to be quite effective when the number of wanted eigenvalues is very small, and outperformed the subspace iteration by a wide margin in an application related to the Markov chain modeling of queueing networks [32].

There are instances, however, where the iterative Arnoldi algorithm exhibits poor performances. In some cases the minimum number of steps m that must be performed in each *inner iteration* in order to ensure convergence of the process, is too large. Another typical case of poor performance is when the eigenvalues that are to be computed are clustered *while the unwanted ones have a very favorable separation* as is illustrated in the next figure, for example:



Then, it is observed that the iterative process has some difficulties to extract the wanted eigenvalues because the process tends to be highly dominated by the convergence in the eigenvectors associated with the right part of the spectrum. These difficulties may be overcome by taking a large enough m but this can become expensive and impractical.

In order to avoid these shortcomings of the iterative Arnoldi process, and, more generally, to improve its overall performance we propose to use it in conjunction with the Chebyshev iteration. The main part of this hybrid algorithm is a Chebyshev iteration which computes a vector of the form $z_i = p_i(A)z_0$, where p_i is a polynomial of degree i, and z_0 is an initial vector. The polynomial p_i is chosen so as to highly amplify the components of z_0 in the direction of the desired eigenvectors while damping those in the remaining eigenvectors. A suitable such polynomial can be expressed in terms of a Chebyshev polynomial of degree i of the first kind. Once $z_i = p_i(A)z_0$ is computed, a few steps of Arnoldi's method, starting with $v_1 = z_i/\|z_i\|$, are carried out in order to extract from z_i the desired eigenvalues.

We will also discuss the implementation of a Chebyshev accelerated subspace iteration algorithm following ideas developed in [30].

In the context of large nonsymmetric linear systems, extensive work has been devoted to the use of Chebyshev polynomials for accelerating linear iterative methods [11], [21], [22], [23], [43]. Manteuffel's work on the determination of the optimal ellipse containing the convex hull of the spectrum of A [23], has been decisive in making the method reliable and effective. For eigenvalue problems, Rutishauser has suggested the use of Chebyshev polynomials for accelerating subspace iteration in the symmetric case [29], [42]. However, Chebyshev acceleration has received little attention as a tool for accelerating the nonsymmetric eigenvalue algorithms. The algorithms that we propose in this paper can be regarded as a simple adaptation of Manteuffel's algorithm to the nonsymmetric eigenvalue problem.

The Chebyshev acceleration technique increases the complexity of the basic Arnoldi method and the resulting improvement may not be worth the extra coding effort for simple problems. However, for more difficult problems, acceleration is important not only because it speeds up the process but mostly because it provides a more reliable method.

We point out that a hybrid Arnoldi-Chebyshev method for solving nonsymmetric linear systems using ideas similar to the ones developed here is currently being developed [8].

In Section 2, we will describe the basic Chebyshev iteration for computing an eigenpair and analyze its convergence properties. In Sections 3, 4 and 5 we will show how to combine the Chebyshev iteration with Arnoldi's method and with the subspace iteration method. In Section 6 we will report a few numerical experiments, and in the last section we will draw a tentative conclusion.

2. Chebyshev Iteration for Computing Eigenvalues of Nonsymmetric Matrices.

2.1 *The Basic Iteration*. Let A be a nonsymmetric real matrix of dimension N and consider the eigenvalue problem:

$$Au = \lambda u. \tag{1}$$

Let $\lambda_1, \ldots, \lambda_N$ be the eigenvalues of A labelled in decreasing order of their real parts, and suppose that we are interested in λ_1 which, to start with, is assumed to be real.

Consider a polynomial iteration of the form: $z_n = p_n(A)z_0$, where z_0 is some initial vector and where p_n is a polynomial of degree n. We would like to choose p_n in such a way that the vector z_n converges rapidly towards an eigenvector of A associated with λ_1 as n tends to infinity. Assuming for simplicity that A is diagonalizable, let us expand z_0 and $z_n = p_n(A)z_0$ in the eigenbasis $\{u_i\}$:

If

$$z_0 = \sum_{i=1}^{N} \theta_i u_i, \tag{2}$$

then

$$z_n = \sum_{i=1}^{N} \theta_i p_n(\lambda_i) u_i = \theta_1 p_n(\lambda_1) u_1 + \sum_{i=2}^{N} \theta_i p_n(\lambda_i) u_i. \tag{3}$$

Expansion (3) shows that if z_n is to be a good approximation of the eigenvector u_1, then every $p_n(\lambda_j)$, with $j \neq 1$, must be small in comparison with $p_n(\lambda_1)$. This leads us to seek a polynomial which is small on the discrete set $R = \{\lambda_2, \lambda_3, \ldots, \lambda_N\}$ and which satisfies the normalization condition

$$p_n(\lambda_1) = 1. \tag{4}$$

An ideal such polynomial would be one which minimizes the (discrete) uniform norm on the discrete set R over all polynomials of degree n satisfying (4). However, this polynomial is clearly impossible to compute without the knowledge of all eigenvalues of A and this approach is therefore of little interest. A simple and more reasonable alternative, known for a long time [41], is to replace the discrete minimax polynomial by the continuous one on a domain containing R but excluding λ_1. Let E be such a domain in the complex plane, and let P_n denote the space of all polynomials of degree not exceeding n. We are thus seeking a polynomial p_n which achieves the minimum

$$\min_{p \in P_n, p(\lambda_1) = 1} \max_{\lambda \in E} |p(\lambda)|. \tag{5}$$

For an arbitrary domain E, it is difficult to solve explicitly the above minimax problem. Iterative methods can be used, however, and the exploitation of the resulting minimax polynomials for solving eigenvalue problems constitutes a promising research area. An alternative way around that difficulty is to restrict E to be an ellipse having its center on the real line, and containing the unwanted eigenvalues λ_i, $i = 2, \ldots, N$.

Let $E(d, c, a)$ be an ellipse with real center d, foci $d + c$, $d - c$, major semiaxis a, and containing the set $R = \{\lambda_2, \ldots, \lambda_N\}$. Since the spectrum of A is symmetric with respect to the real axis, we will restrict $E(d, c, a)$ to being symmetric as well. In other words, the main axis of the ellipse must be either the real axis or must be parallel to the imaginary axis. Therefore, a and c are either real or purely imaginary, see Figure 2-1.

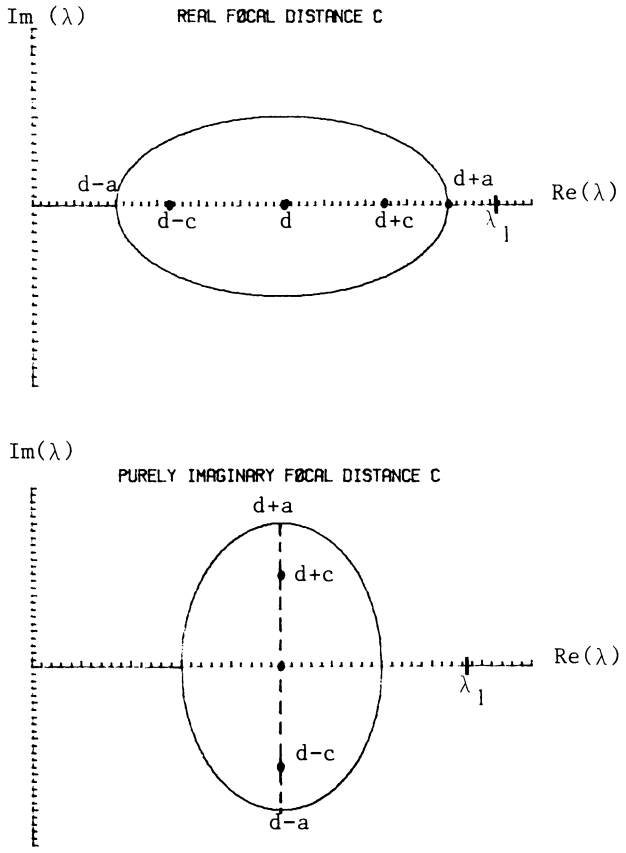


FIGURE 2-1: *Ellipses containing the set R of the remaining eigenvalues*

Then it is known that, when E is the ellipse $E(d, c, a)$ in (5), the best minimax polynomial is the polynomial

$$p_n(\lambda) = \frac{T_n[(\lambda - d)/c]}{T_n[(\lambda_1 - d)/c]}, \tag{6}$$

where T_n is the Chebyshev polynomial of degree n of the first kind, see [3], [21], [43].

The computation of z_n, $n = 1, 2, \ldots$, is simplified by the three-term recurrence for the Chebyshev polynomials:

$$T_1(\lambda) = \lambda, \qquad T_0(\lambda) = 1,$$
$$T_{n+1}(\lambda) = 2\lambda T_n(\lambda) - T_{n-1}(\lambda), \qquad n = 1, 2, \ldots.$$

Letting $\rho_n = T_n[(\lambda_1 - d)/c]$, $n = 0, 1, \ldots$, we obtain

$$\rho_{n+1} p_{n+1}(\lambda) = T_{n+1}[(\lambda - d)/c] = 2\frac{\lambda - d}{c}\rho_n p_n(\lambda) - \rho_{n-1} p_{n-1}(\lambda).$$

Let us transform this further by setting $\sigma_{n+1} = \rho_n/\rho_{n+1}$:

$$p_{n+1}(\lambda) = 2\sigma_{n+1}\frac{\lambda - d}{c} p_n(\lambda) - \sigma_n\sigma_{n+1} p_{n-1}(\lambda).$$

A straightforward manipulation using the definitions of σ_i, ρ_i and the three-term recurrence relation of the Chebyshev polynomials shows that σ_i, $i = 1,\ldots,$ can be obtained from the recursion:

$$\sigma_1 = c/(\lambda_1 - d); \qquad \sigma_{n+1} = \frac{1}{2/\sigma_1 - \sigma_n}, \qquad n = 1,2,\ldots.$$

The above two recursions can now be assembled together to yield a basic algorithm for computing $z_i = p_i(A)z_0$, $i = 1,2,\ldots$. Although λ_1 is not known, recall that it is used in the denominator of (6) for scaling purposes only, so we can replace it by some approximation ν in practice.

Algorithm: Chebyshev Iteration

1. *Start*: Choose an arbitrary initial vector z_0; compute

$$\sigma_1 = c/(\lambda_1 - d), \tag{7}$$

$$z_1 = \frac{\sigma_1}{c}(A - dI)z_0. \tag{8}$$

2. *Iterate*: For $n = 1,2,\ldots$ until convergence do:

$$\sigma_{n+1} = \frac{1}{2/\sigma_1 - \sigma_n}, \tag{9}$$

$$z_{n+1} = 2\frac{\sigma_{n+1}}{c}(A - dI)z_n - \sigma_n\sigma_{n+1}z_{n-1}. \tag{10}$$

An important detail, which we have not discussed for the sake of clarity, concerns the case when c is purely imaginary. It can be shown quite easily that even in this situation the above recursion can still be carried out in real arithmetic. The reason for this is that the scalars σ_i, $i = 1,\ldots,$ become all purely imaginary as can easily be shown by induction. Hence the scalars σ_{n+1}/c and $\sigma_{n+1}\sigma_n$ in the above algorithm are real numbers. The primary reason for scaling by $T_n[(\lambda_1 - d)/c]$ in (6) is to avoid overflow but, as was just explained, a secondary reason is to avoid complex arithmetic when c is purely imaginary.

2.2. *Convergence Properties.* In order to understand the convergence properties of the sequence of approximations z_n consider its expansion (3):

$$z_n = \sum_{i=1}^{N} \theta_i p_n(\lambda_i)u_i = \theta_1 u_1 + \sum_{i=2}^{N} \theta_i p_n(\lambda_i)u_i.$$

We would like to examine the behavior of each coefficient of u_i, for $i \neq 1$. We have:

$$p_n(\lambda_i) = \frac{T_n[(\lambda_i - d)/c]}{T_n[(\lambda_1 - d)/c]}.$$

From one of the various ways of defining the Chebyshev polynomials in the complex plane [27], the above expression can be rewritten as

$$p_n(\lambda_i) = \frac{w_i^n + w_i^{-n}}{w_1^n + w_1^{-n}}, \tag{11}$$

where w_i represents the root of largest modulus of the equation in w:

$$\tfrac{1}{2}(w + w^{-1}) = (\lambda_i - d)/c. \tag{12}$$

From (11), $p_n(\lambda_i)$ is asymptotic to $[w_i/w_1]^n$, hence:

Definition 1. We will refer to $\kappa_i = |w_i/w_1|$ as the damping coefficient of λ_i relative to the parameters d, c. The convergence ratio $\tau(\lambda_1)$ of λ_1 is the largest damping coefficient κ_i for $i \neq 1$.

This definition must be understood in the following sense: each coefficient in the eigenvector u_i of the expansion (3) behaves like κ_i^n, as n tends to infinity. The damping coefficient $\kappa(\lambda)$ can obviously be also defined for any value λ of the complex plane by replacing λ_i by λ.

One of the most important features in Chebyshev iteration lies in Eq. (12). Observe that there are infinitely many points λ in the complex plane whose damping coefficient $\kappa(\lambda)$ has the same value κ. These points λ are defined by $(\lambda - d)/c = (w + w^{-1})/2$ and $|w/w_1| = \kappa$ where ρ is some constant. Thus a great deal of simplification can be achieved by locating those points *that are real* as it is preferable to deal with real quantities than imaginary ones in the above expression defining κ_i. The well-known mapping $J(w) = \frac{1}{2}(w + w^{-1})$, often referred to as the Joukowski transform [27], maps a circle into an ellipse in the complex plane. More precisely, for $w = \rho e^{i\theta}$, $J(w)$ belongs to an ellipse of center the origin, focal distance 1, and major semiaxis $\alpha = \frac{1}{2}(\rho + \rho^{-1})$. Given the major semiaxis α, ρ is determined by $\rho = \frac{1}{2}[\alpha + (\alpha^2 - 1)^{1/2}]$. As a consequence the damping coefficient κ_i is simply ρ_i/ρ_1 where $\rho_j = \frac{1}{2}[\alpha_j + (\alpha_j^2 - 1)^{1/2}]$ and α_j is the major semiaxis of the ellipse centered at the origin, with focal distance one and passing through $(\lambda_j - d)/c$. Since $\alpha_1 > \alpha_i$, $i = 2, 3, \ldots, N$, it is easy to see that $\rho_1 > \rho_i$, $i > 1$, and hence that the process will converge. Note that there is a further mapping between λ_j and $(\lambda_j - d)/c$ which transforms the ellipse $E(d, c, a_j)$ into the ellipse $E(0, 1, \alpha_j)$ where a_j and α_j are related by $\alpha_j = a_j/c$. Therefore, the above expression for the damping coefficient can be rewritten as:

$$\kappa_i = \rho_i/\rho_1 = \frac{a_i + \left(a_i^2 - 1\right)^{1/2}}{a_1 + \left(a_1^2 - 1\right)^{1/2}}, \tag{13}$$

where a_i is the major semiaxis of the ellipse of center d, focal distance c, passing through λ_i. From the expansion (3), the vector z_n converges to $\theta_1 u_1$, and the error behaves like $\tau(\lambda_1)^n$. For nonnormal matrices A random vectors z_0 sometimes yield small values of θ_1 and then convergence is delayed.

The above algorithm ignores the following important points:

●It is unrealistic to assume that the parameters d and c are known beforehand, and some adaptive scheme must be implemented in order to estimate them dynamically.

●The algorithm does not handle the computation of more than one eigenvalue. In particular what to do in case λ_1 is complex, i.e. when λ_1 and $\lambda_2 = \bar{\lambda}_1$ form a complex pair?

Suppose that $E(d, c, a)$ contains all the eigenvalues of A except for a few. Looking closely at the expansion of z_n, we observe that it contains more than just an approximation to u_1 because we can write:

$$z_n = \theta_1 u_1 + \theta_{i_1} u_{i_2} + \cdots + \theta_{i_r} u_{i_r} + \varepsilon, \tag{14}$$

where $\lambda_{i_1}, \ldots, \lambda_{i_r}$ are the eigenvalues outside $E(d, c, a)$ and ϵ is a small term in comparison with the first r ones. All we need, therefore, is some powerful method to extract those eigenvalues from the single vector z_n. We will refer to such a method as a purification process. One such process among many others is the Arnoldi method considered in the next section.

3. Arnoldi's Method as a Purification Process. A brief description of Arnoldi's method is the following:

Arnoldi's Algorithm

1. *Start*: Choose an initial vector v_1 of norm unity, and a number of steps m.
2. *Iterate*: For $j = 1, 2, \ldots, m$ do:

$$\hat{v}_{j+1} = Av_j - \sum_{i=1}^{j} h_{ij} v_i, \tag{15}$$

$$\text{with } h_{ij} = (Av_j, v_i), \qquad i = 1, \ldots, j, \tag{16}$$

$$h_{j+1,j} = \|\hat{v}_{j+1}\|, \tag{17}$$

$$v_{j+1} = \hat{v}_{j+1}/h_{j+1,j}. \tag{18}$$

This algorithm produces an orthonormal basis $V_m = [v_1, v_2, \ldots, v_m]$ of the Krylov subspace $K_m = \text{span}\{v_1, Av_1, \ldots, A^{m-1}v_1\}$. In this basis the restriction of A to K_m is represented by the upper Hessenberg matrix H_m whose entries are the elements h_{ij} produced by the algorithm. The eigenvalues of A are approximated by those of H_m which is such that $H_m = V_m^T A V_m$. The associated approximate eigenvectors are given by:

$$\tilde{u}_i = V_m \tilde{y}_i, \tag{19}$$

where $\tilde{y}_i$ is an eigenvector of H_m associated with the eigenvalue $\tilde{\lambda}_i$. We will assume throughout that the pair of eigenvectors associated with a conjugate pair of eigenvalues are normalized so that they are conjugate to each other. Note that $\tilde{u}_i$ has the same Euclidean norm as $\tilde{y}_i$. The following relation is extremely useful for obtaining the residual norm of $\tilde{u}_i$ without even computing it explicitly:

$$\|(A - \tilde{\lambda}_i I)\tilde{u}_i\| = h_{m+1,m}|e_m^T \tilde{y}_i| \tag{20}$$

in which $e_m = (0, 0, \ldots, 0, 1)^T$. This result is well known in the case of the symmetric Lanczos algorithm [25], and its extension to the nonsymmetric case is straightforward [32].

The method of Arnoldi amounts to a Galerkin process applied to the Krylov subspace K_m [1], [32]. A few variations on the above basic algorithm have been proposed in [32] in order to overcome some of its impractical features, and a theoretical analysis was presented in [30].

One important property of the algorithm is that if the initial vector v_1 is exactly in an invariant subspace of dimension r and not in any invariant subspace of smaller dimension, i.e., if the degree of the minimal polynomial of v_1 is r, then the above algorithm cannot be continued after step r, because we will obtain $\|\hat{v}_{r+1}\| = 0$. However, the next proposition shows that in this case K_r will be invariant which implies, in particular, that the r computed eigenvalues are exact.

PROPOSITION 1. *Assume that the degree of the minimal polynomial of v_1 is equal to r. Then Arnoldi's method stops at step r and K_r is an invariant subspace. Furthermore, the eigenvalues of H_r are the eigenvalues of A associated with the invariant subspace K_r.*

Proof. $K_{r+1} = \text{span}\{v_1, Av_1, \ldots, A^r v_1\}$. By assumption there is a monic polynomial p of degree r such that $p(A)v_1 = 0$. Hence $A^r v_1 \in K_r$ and $K_{r+1} = K_r$. So AK_r which is a subset of K_{r+1} equals K_r and K_r is invariant under A. Consequently, H_r is not just a projection of A onto K_r but a restriction of A to K_r. As a consequence every eigenvalue of H_r is an eigenvalue of A. □

4. The Arnoldi-Chebyshev Method. Suppose that we can find an ellipse $E(d, c, a)$ that contains all the eigenvalues of A except the r wanted ones, i.e., the r eigenvalues of A with largest real parts. We will describe in a moment an adaptive way of getting such an ellipse. Then an appealing algorithm would be to run a certain number of steps of the Chebyshev iteration and take the resulting vector z_n as initial vector in the Arnoldi process. From the Arnoldi purification process one obtains a set of m eigenvalues, r of which are approximations to the r wanted ones, as suggested by Proposition 1, while the remaining ones will be useful for adaptively constructing the best ellipse. After a cycle consisting of n steps of the Chebyshev iteration followed by m steps of the purification process, the accuracy realized for the r rightmost eigenpairs may not be sufficient and restarting will then be necessary. The following is an outline of a simple algorithm based on the above ideas:

● *Start*: Choose an initial vector v_1, a number of Arnoldi steps m and a number of Chebyshev steps n.

● *Iterate*:

1. Perform m steps of the Arnoldi algorithm starting with v_1. Compute the m eigenvalues of the resulting Hessenberg matrix. Select the r eigenvalues of largest real parts $\tilde{\lambda}_1, \ldots, \tilde{\lambda}_r$ and take $\tilde{R} = \{\tilde{\lambda}_{r+1}, \ldots, \tilde{\lambda}_m\}$. If satisfied stop, otherwise continue.

2. Using $\tilde{R}$, obtain the new estimates of the parameters d and c of the best ellipse. Then compute the initial vector z_0 for the Chebyshev iteration as a linear combination of the approximate eigenvectors $\tilde{u}_i$, $i = 1, \ldots, r$.

3. Perform n steps of the Chebyshev iteration to obtain z_n. Take $v_1 = z_n/\|z_n\|$ and to back to 1.

Next, some important details left unclear in the above simplistic description will be examined.

4.1. *Getting the Optimal Ellipse.* As explained earlier we would like to find the 'best' ellipse enclosing the set R of nonwanted eigenvalues, i.e., the eigenvalues other than the ones with the r algebraically largest real parts. We must begin by clarifying what is meant by 'best' in the present context. Consider Figure 4-1 representing a spectrum of some matrix A and suppose that we are interested in the r rightmost eigenvalues, i.e. $r = 4$ in the figure.

We will extend Definition 1 of the convergence ratio $\tau(\lambda_i)$ of an eigenvalue λ_i, $i = 1, \ldots, r$, as the largest damping coefficient κ_j for $j = r + 1, \ldots, N$.

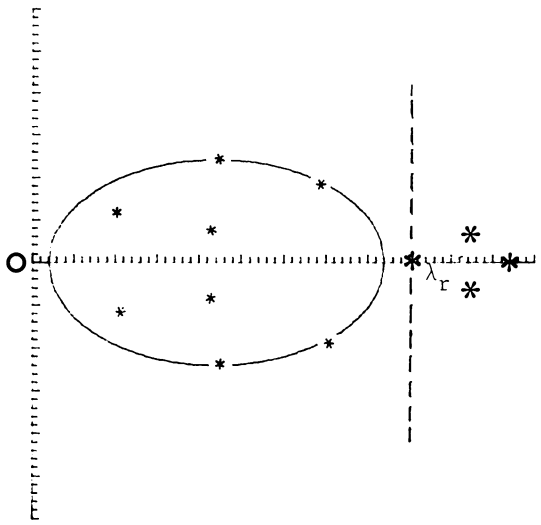


FIGURE 4-1: *The optimal ellipse*

In the context of linear systems, there is only one convergence ratio [21], [23], and the best ellipse is defined as being the one which maximizes that ratio. In our situation we have r different convergence ratios each corresponding to one of the desired eigenvalues λ_i, $i = 1, \ldots, r$.

Initially, assume that λ_r is real and consider any ellipse $E(d, c, a)$ including R and not $\{\lambda_1, \lambda_2, \ldots, \lambda_r\}$. It is easily seen from our comments of subsection 2.2 that if we draw a vertical line passing through the eigenvalue λ_r, all eigenvalues to the right of that line will converge faster than those to its left. Therefore, when λ_r is real, we may simply define the best ellipse as the one maximizing the convergence ratio of λ_r over the parameters d and c.

When λ_r is not real, the situation is more complicated. We could still attempt to maximize the convergence ratio for the eigenvalue λ_r, but the formulas giving the optimal ellipse do not extend to the case where λ_r is complex and the best ellipse becomes difficult to determine. But this is not the main reason why this choice is not suitable. A close look at Figure 4-2, in which we assume $r = 5$, reveals that the best ellipse for λ_r may not be a good ellipse for some of the desired eigenvalues. More precisely, in the figure, the pair λ_2, λ_3 is enclosed by the best ellipse for λ_5. As a consequence the components in u_2, u_3 will converge more slowly than those in some of the undesired eigenvectors, e.g. u_N in the figure.

The figure explains the difficulty more clearly: the problem comes from the relative position of λ_4 and λ_2 with respect to the rest of the spectrum, and it can be resolved by just maximizing the convergence ratio of λ_2 instead of λ_5 in this case.

In a more complex situation it is unfortunately more difficult to determine at which particular eigenvalue λ_k or more generally at which value μ it is best to maximize $\tau(\mu)$. Clearly, one could solve the problem by taking $\mu = \mathrm{Re}(\lambda_r)$, but this is not the best choice.

As an alternative, we propose to take advantage of the previous ellipse, i.e. the ellipse determined from the previous purification step, as follows. We determine a

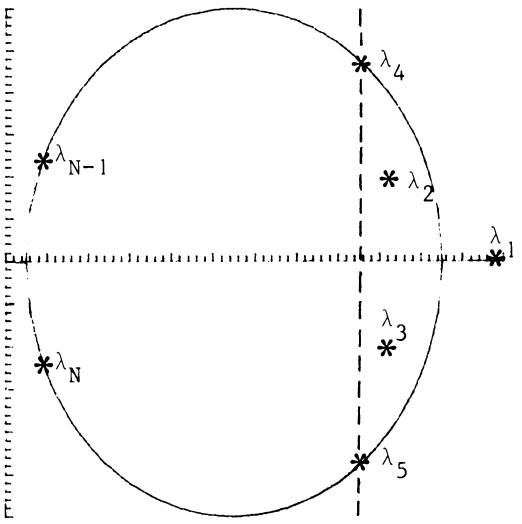


FIGURE 4-2: *A case where* λ_r *is complex*: *the eigenvalues* λ_2 *and* λ_3 *are inside the "best" ellipse*

point μ on the real line *having the same convergence ratio as* λ_r *with respect to the previous ellipse*. The next 'best' ellipse is then determined so as to maximize the convergence ratio for this point μ. This reduces to the previous choice $\mu = \mathrm{Re}(\lambda_r)$ when λ_r is real. At the very first iteration one can set μ to be $\mathrm{Re}(\lambda_r)$.

The question which we have not yet fully answered concerns the practical determination of the best ellipse. At a typical step of the Arnoldi process we are given m approximations $\tilde{\lambda}_i$, $i = 1,\ldots, m$, of the eigenvalues of A. This approximate spectrum is divided in two parts: the r wanted eigenvalues $\tilde{\lambda}_1,\ldots, \tilde{\lambda}_r$ and the set $\tilde{R}$ of the remaining eigenvalues $\tilde{R} = \{\tilde{\lambda}_{r+1}, \tilde{\lambda}_{r+2},\ldots, \tilde{\lambda}_m\}$. From the previous ellipse and the previous sets $\tilde{R}$, we would like to determine the next estimates for the optimal parameters d and c.

Fortunately, a similar problem was solved by Manteuffel [21], [23] and his work can easily be adapted to our situation. The change of variables $\xi = (\mu - \lambda)$ transforms μ into the origin in the ξ-plane and the problem of maximizing the ratio $\tau(\mu)$ is transformed into one of maximizing a similar ratio in the ξ-plane for the origin, with respect to the parameters d and c. An effective technique for solving this final problem has been developed in [21], [23] but we will not describe it here. Thus, all we have to do is pass the shifted eigenvalues $\mu - \tilde{\lambda}_j$, $j = r + 1,\ldots, m$ to the appropriate codes in [21], and the optimal values of $\mu - d$ and c will be returned.

4.2. *Starting the Chebyshev Iteration*. Once the optimal parameters d and c have been estimated we are ready to carry out a certain number n of steps of the Chebyshev iteration (10). In this subsection we would like to indicate how to select the starting vector z_0 for this iteration. Before doing so, we wish to deal with a minor difficulty encountered when λ_1 is complex. Indeed, it was mentioned after the algorithm described in subsection 2.1 that the eigenvalue λ_1 in (7) should, in practice, be replaced by some approximation ν of λ_1. If $\tilde{\lambda}_1$ defined in subsection 4.1

is real then we can take $\nu = \tilde{\lambda}_1$ and the iteration can be carried out in real arithmetic as was already shown, even when c is purely imaginary. However, the iteration will become complex if $\tilde{\lambda}_1$ is complex. To avoid this it suffices to take ν to be one of the two points where the ellipse $E(d, c, a_1)$ passing through $\tilde{\lambda}_1$, crosses the real axis. The effect of the corresponding scaling of the Chebyshev polynomial will be identical with that using $\tilde{\lambda}_1$ but will present the advantage of avoiding complex arithmetic.

Let us now indicate how one can select the initial vector z_0. In the hybrid algorithm outline in the previous section, the Chebyshev iteration comes after an Arnoldi step. It is then desirable to start the Chebyshev iteration by a vector which is a linear combination of the approximation eigenvectors (19) associated with the rightmost r eigenvalues.

Let ξ_i be the coefficients of the desired linear combinations. Then the initial vector for the Chebyshev process is

$$z_0 = \sum_{i=1}^{r} \xi_i \tilde{u}_i = \sum_{i=1}^{r} \xi_i V_m \tilde{y}_i = V_m \sum_{i=1}^{r} \xi_i \tilde{y}_i.$$

Hence

$$z_0 = V_m y, \quad \text{where } y = \sum_{i=1}^{r} \xi_i \tilde{y}_i. \tag{21}$$

Therefore, the eigenvectors $\tilde{u}_i$, $i = 1, r$ need not be computed explicitly. We only need to compute the eigenvectors of the Hessenberg matrix H_m and to select the appropriate coefficients ξ_i. An important remark is that if we choose the ξ's to be real and such that $\xi_i = \xi_{i+1}$ for all conjugate pairs λ_i, $\lambda_{i+1} = \bar{\lambda}_i$, then the above vector z_0 is real.

Assume that all eigenvectors, exact and approximate, are normalized so that their 2-norms are equal to one. One desireable objective when choosing the above linear combination is to attempt to make z_n, the vector which starts the *next* Arnoldi step, equal to a sum of eigenvectors of A of norm unity, i.e., the objective is to have $z_n = \theta_1 u_1 + \theta_2 u_2 + \cdots + \theta_r u_r$, with $|\theta_i| = 1$, $i = 1, 2, \ldots, r$. For this purpose, suppose that for each approximate eigenvector $\tilde{u}_i$ we have $\tilde{u}_i = \gamma_i u_i + \varepsilon_i$, where the vector ε_i has no components in $u_1, \ldots, u_r$. Then:

$$z_n = \xi_1\gamma_1 u_1 + \xi_2\gamma_2 u_2 + \cdots + \xi_r\gamma_r u_r + \varepsilon, \quad \text{where } \varepsilon = \sum_{i=1}^{r} \xi_i \varepsilon_i.$$

Near convergence $|\gamma_i|$ is close to one and $\|\varepsilon_i\|$ is small. The result of n steps of the Chebyshev iteration applied to z_0 will be a vector z_n such that:

$$z_n \approx \xi_1\gamma_1 u_1 + \kappa_2^n \xi_2\gamma_2 u_2 + \cdots + \kappa_r^n \xi_r\gamma_r u_r + p_n(A)\varepsilon.$$

Since ε has no components in u_i, $i = 1, \ldots, r$, $p_n(A)\varepsilon$ tends to zero faster than the first r terms, as n tends to infinity. Hence, taking $\xi_i = \kappa_i^{-n}$, $i = 1, 2, \ldots, r$, will give a vector which has components γ_i in the eigenvectors u_i, $i = 1, \ldots, r$. Since $|\gamma_i| \approx 1$ near convergence this is a satisfactory choice.

Another possibility suggested in [32] for the iterative Arnoldi process is to weigh the combination of $\tilde{u}_i$ according to the accuracy obtained after an Arnoldi step, for example:

$$\xi_i = \|(A - \tilde{\lambda}_i I)\tilde{u}_i\|.$$

Notice that the residuals of two complex conjugate approximate eigenelements are equal, so this choice will also lead to a real z_0. The purpose of weighing a vector $\tilde{u}_i$ by its residual norm is to attempt to balance the accuracy between the different eigenelements that would be obtained at the next Arnoldi step. Thus if too much accuracy is obtained for u_1 versus the other approximate eigenvectors, the above choice of the ξ_i's will put less weight on $\tilde{u}_1$ and more on the other vectors in order to attempt to reduce the advantage of u_1 in the next Arnoldi step.

In the experiments reported later, we have only considered the first possibility which was observed to be slightly more effective than the latter.

4.3. *Choosing the Parameters m and n.* The number of Arnoldi steps m and the number of Chebyshev steps n are important parameters that affect the effectiveness of the method. Since we want to obtain more eigenvalues than the r desired ones, in order to use the remainder in choosing the parameters of the ellipse, m should be at least $r + 2$ (to be able to compute a complex pair). In practice, however, it is preferable to take m several times larger than r. In typical runs m is at least $3r$ or $4r$ but can very well be even larger if storage is available. It is also possible to change m dynamically instead of keeping it fixed to a certain value but this variation will not be considered here.

When choosing n, we have to take into account the following facts:

●Taking n too small may result in a slowing down of the algorithm; ultimately when $n = 0$, the method becomes the simple iterative Arnoldi method.

●It may not be effective to pick n too large: otherwise the vector z_n may become nearly an eigenvector which could be troublesome for the Arnoldi process. Moreover, the parameters d, c of the ellipse may be far from optimal and it is better to reevaluate them frequently.

Recalling that the component in the direction of u_1 will remain constant while those in u_i, $i = 2, \ldots, r$, will be of the same order as κ_i^n, we should attempt to avoid having a vector z_n which is entirely in the direction of u_1. This can be done by requiring that all κ_i^n, $i = 2, \ldots, r$, be no less than a certain tolerance δ, i.e.:

$$n \approx \log(\delta)/\log[\kappa_j], \tag{22}$$

where κ_j is the largest convergence ratio among κ_i, $i = 2, \ldots, r$. In the code tested in Section 6, we have opted to choose δ to be nearly the square root of the unit round-off.

Other practical factors should also enter into consideration. For example, it is desirable that a maximum number of Chebyshev steps $n_{\max}$ be fixed by the user. Also in case we are close to convergence, we should avoid employing an unnecessarily large number of steps as might be dictated by a straightforward application of (22).

5. Application to the Subspace Iteration Algorithm.

5.1. *The Basic Subspace Iteration Algorithm.* The subspace iteration method, or simultaneous iteration method, can be regarded as a (Galerkin) projection method onto a subspace of the form $A^n X$, where $X \equiv [x_1, \ldots, x_m]$ is an initial system of m linearly independent vectors. There are many versions of the method [4], [13], [38],

[39], but a very simple one is the following:

1. *Start*: $Q \Leftarrow X$.
2. *Iteration*: Compute $Q \Leftarrow A^n Q$.
3. *Projection step*: Orthonormalize Q and get eigenvalues and eigenvectors of $C = Q^T A Q$. Compute $Q \Leftarrow QF$, where F is the matrix of eigenvectors of C.
4. *Convergence test*: If Q is not a satisfactory set of approximate eigenvectors go to 2.

The algorithm presented in [13] is equivalent to the above algorithm except that the approximate eigenelements are computed without having to orthonormalize Q. The SRRIT algorithm presented by Stewart [38], [39] aims at computing an orthonormal basis Q of the invariant subspaces rather than a basis formed of eigenvectors. It is also mathematically equivalent to the above in the restricted sense that the corresponding invariant subspaces are theoretically identical. We should point out that this latter approach is more robust because an eigenbasis of the invariant subspace may not exist or may be badly conditioned, thus causing serious difficulties for the other versions. We should stress however that the Chebyshev acceleration technique can be applied to *any version of the subspace iteration* although it will only be described for the simpler version presented above.

5.2. *Chebyshev Acceleration*. The use of Chebyshev polynomials for accelerating the subspace iteration was suggested by Rutishauser [29], [42] for the symmetric case. It was pointed out in [30] that this powerful technique can be extended to the nonsymmetric case but no explicit algorithm was formulated for computing the best ellipse.

We will use the same notation as in the previous sections. Suppose that we are interested in the rightmost r eigenvalues and that the ellipse $E(d, c, a)$ contains the set R of all the remaining eigenvalues. Then the principle of the Chebyshev acceleration method is simply to replace the powers A^n in the first part of the basic algorithm described above by $p_n(A)$ where p_n is the polynomial defined by (6). It can be shown [30] that the approximate eigenvector $\tilde{u}_i$, $i = 1, \ldots, r$ converges towards u_i, as $T_n(a/c)T_n[(\lambda_i - d)/c]$, which, using arguments similar to those of subsection 2.2, is equivalent to η_i^n where

$$\eta_i = \frac{a + [a^2 - 1]^{1/2}}{a_i + [a_i^2 - 1]^{1/2}}. \tag{23}$$

The above convergence ratio can be far better than the value $|\lambda_{r+1}/\lambda_i|$ which is achieved by the classical algorithm.**

On the practical side, the best ellipse is obtained dynamically in the same way as was proposed for the Chebyshev-Arnoldi process. The accelerated algorithm will then have the following structure:

1. *Start*: $Q \Leftarrow X$.
2. *Iteration*: Compute $Q \Leftarrow p_n(A)Q$.

** The subspace iteration method computes the eigenvalues of largest moduli. Therefore, the regular subspace iteration method and the accelerated method are comparable only when the $r + 1$ rightmost eigenvalues are also the $r + 1$ dominant ones.

3. *Projection step*: Orthonormalize Q and get eigenvalues and eigenvectors of $C = Q^T A Q$. Compute $Q \Leftarrow QF$, where F is the matrix of eigenvectors of C.

4. *Convergence test*: If Q is a satisfactory set of approximate eigenvectors then stop, else get new best ellipse and go to 2.

Most of the devices described for the Arnoldi process extend naturally to this algorithm, and we now discuss briefly a few of them.

1. *Getting the best ellipse.* The construction of the best ellipse is identical with that seen in subsection 4.1. The only difficulty we might encounter is that the extra eigenvalues used to build the best ellipse are now less accurate in general than those provided by the more powerful Arnoldi technique. More care must therefore be taken in order to avoid building an ellipse based on inaccurate eigenvalues as this may slow down considerably the algorithm.

2. *Parameters n and m.* Here, one can take advantage of the abundant work on subspace iteration available in the literature. All we have to do is replace the convergence ratios $|\lambda_{r+1}/\lambda_i|$ of the basic subspace iteration by the new ratios η_i of (23). For example, one way to determine the number of Chebyshev steps n, proposed in [29] and in [14] is:

$$n \approx \tfrac{1}{2}\left[1 + \log(\varepsilon^{-1})/\log(\eta_1)\right],$$

where ε is some parameter depending on the unit round-off. The goal of this choice is to prevent the rounding errors from growing beyond the level of the error in the most slowly converging eigenvector. The parameter n is also limited from above by a user supplied bound $n_{\max}$, and by the fact that if we are close to convergence a smaller n can be determined to ensure convergence at the next projection step.

The same comments as in the Arnoldi-Chebyshev method can be made concerning the choice of m, namely that m should be at least $r + 2$, but preferably even larger although in a lesser extent than for Arnoldi. Note that for the symmetric case it is often suggested to take $m = 2r$ or $m = 3r$.

3. *Deflation.* Another special feature of the subspace iteration is the deflation technique which consists in working only with the nonconverged eigenvectors, thus 'locking' those that have already converged, see [14], [29], [38]. Clearly, this can be used in the accelerated subspace iteration as well and will enhance its efficiency. For the more stable versions such as SRRIT, a similar device can be applied to the Schur vectors instead of the eigenvectors [39].

6. Numerical Experiments. The numerical experiments described in this section have been performed on a VAX11-780 computer using double precision (unit round-off $\varepsilon \approx 6.9 \times 10^{-18}$).

6.1. *An Example of Markov Chain Modeling.* An interesting class of test examples described by Stewart [39] deals with the computation of the steady state probabilities of a Markov chain. This example models a random walk on a $(k + 1)$ by $(k + 1)$ triangular grid. A particle moves randomly on the grid by jumping to one of the (at most) four adjacent grid points, see Figure 6-1. The probability of jumping from the node (i, j) to either of the nodes $(i - 1, j)$ or $(i, j - 1)$ is given by:

$$pd(i, j) = \frac{i + j}{2k}$$

this probability being doubled if either of i or j is zero. The probability of jumping from the node (i, j) to either of the nodes $(i + 1, j)$ or $(i, j - 1)$ is given by

$$pu(i, j) = \tfrac{1}{2} - pd(i, j).$$

(Note that this transition does not occur when $i + j = k$, which is expressed by the fact that $pu(i, j)$ is then equal to zero.) We are interested in the steady state probability distribution of the chain. Such probabilities are the components of the appropriately scaled eigenvector associated with the eigenvalue unity of the transpose of the transition probability matrix [18], [35].

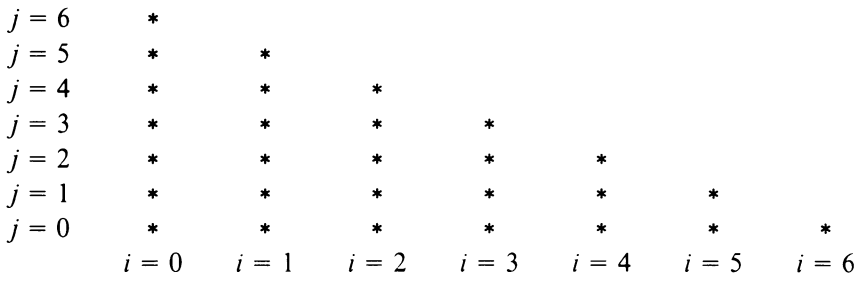


FIGURE 6-1: *Random walk on a triangular grid*

The nodes (i, j) are labelled in the order $(0,0)$, $(1,0) \cdots (k,0)$; $(0, 1)$, $(1, 1) \cdots (k - 1, 1)$; $\cdots (0, k)$. With this it is easy to form the matrix A. But this is not even necessary, nor is it necessary to store A in any way because the operations $y = Ax$ for any vector x can be performed by a simple subroutine.

In our first test we have taken $k = 30$, which means that the dimension of the problem is $N = \frac{1}{2}(k + 1)(k + 2) = 496$.

The subspace iteration method SRRIT was tested in [39] for this case. We have tried our simple version of the algorithm described in Section 5. As initial system X we have taken the system $[x, Ax \cdots A^{m-1}x]$ where x is a random vector. The following results were obtained for various values of the parameters m (the block size) and $n_{\max}$ (the maximum number of Chebyshev steps in each inner loop):

TABLE 1. *Subspace iteration*

m	n max	Iterations	Matrix-Vector multiplications	Execution times (Sec.)	Residual norms
6	20	252	1389	62.7	6.4 E − 06
6	50	262	1383	56.6	3.8 E − 06
8	20	180	1467	69.2	8.5 E − 06
8	50	182	1413	59.5	7.5 E − 06
10	20	145	1422	74.3	6.4 E − 06
10	50	150	1457	63.3	4.1 E − 06

The stopping criterion was that the residual norms of the eigenpair corresponding to the eigenvalue unity is less than the tolerance $\varepsilon = 10^{-5}$, the same as in [39].

The difference from the number of matrix by vector multiplications reported in [39], is due mostly to the fact that the two implementations are different. Part of the

difference is also due to the stopping criterion which, in [39], deals with the two dominant eigenvalues 1 and -1 (-1 is also known to be an eigenvalue). Observe from the above table that for the same block size m, the performance is better with the larger $n_{\max} = 50$ than with $n_{\max} = 20$. The reason is that although the two iterations are essentially equivalent, a smaller $n_{\max}$ leads to more frequent projections, and therefore to substantial overhead.

The next table shows the same example treated by the Chebyshev accelerated subspace iteration.

TABLE 2. *Chebyshev-subspace iteration*

m	n max	Iterations	Matrix-Vector multiplications	Execution times (Sec.)	*Residual norms*
6	50	25	1019	60.2	1.2 E − 07
6	20	30	645	41.1	4.6 E − 06
8	20	42	903	59.7	4.9 E − 06
8	50	28	1063	66.0	3.8 E − 09
10	20	45	909	64.3	1.0 E − 06
10	50	27	979	62.3	1.9 E − 07

The stopping criterion and the initial set X were the same as for the previous test. Notice that here the effect of the upper limit $n_{\max}$ of the number of Chebyshev iterations can be quite important, as for example when $m = 6$. In opposition with the observation made above for the nonaccelerated algorithm, the performance is now better for smaller values of the parameter $n_{\max}$. The explanation for this is provided by a close examination of the successive ellipses that are adaptively computed by the process. It is possible to observe that when the ellipse does not accurately represent the convex hull of the remaining eigenvalues, a larger $n_{\max}$ leads to wasting an important amount of computational work before having the chance of evaluating new parameters. Thus, for smaller values of $n_{\max}$, the process has a better ability to correct itself by computing a better ellipse more frequently. This is less critical with the Arnoldi process because the eigenvalues provided by Arnoldi's method are usually more accurate.

It is instructive to compare the above performances with those of the iterative Arnoldi, and the Chebyshev-Arnoldi methods. The next two tables summarize the results obtained with the iterative Arnoldi method (Table 3) and the Arnoldi-Chebyshev method (Table 4). The stopping criterion is the same as before, and the initial vector used in the first Arnoldi iteration is random.

TABLE 3. *Iterative Arnoldi*

m	Arnoldi iterations	Matrix-Vector multiplications	Execution times (Sec.)	Residual norms
5	36	180	21.8	7.5 E − 06
10	14	140	22.2	9.3 E − 06
15	8	120	25.7	7.3 E − 06
20	6	120	33.3	6.2 E − 06

TABLE 4. *Chebyshev-Arnoldi*

m	n max	Arnoldi calls	Matrix-Vector multiplications	Execution times (Sec.)	Residual norms
5	20	6	130	9.4	8.9 E − 06
5	50	4	142	8.9	3.9 E − 07
10	20	5	130	13.9	5.0 E − 06
10	50	3	113	9.6	7.1 E − 06
15	20	3	85	11.9	6.8 E − 06
15	50	3	122	14.6	3.2 E − 10
20	20	3	100	18.6	1.5 E − 07
20	50	3	88	14.2	4.3 E − 09

The results of Table 4 constitute a considerable improvement over those of the subspace iteration, both in execution time and in number of matrix by vector multiplications. Notice that in this example we are also able to reduce the execution time by a factor of nearly 2.5 from the iterative Arnoldi method.

6.2 *Computing Several Eigenvalues.* The above experiments deal with the computation of only one eigenpair, and we would like next to compare the performances of our methods on problems dealing with several eigenvalues. Consider the following partial differential linear operator on the unit square, with the Dirichlet boundary conditions, derived from [7]:

$$Lu \equiv \frac{\partial}{\partial x}\left(a\frac{\partial u}{\partial x}\right) + \frac{\partial}{\partial y}\left(b\frac{\partial u}{\partial y}\right) + \frac{\partial gu}{\partial x} + g\frac{\partial u}{\partial x}. \tag{24}$$

The functions a, b, and g are defined by:

$$a(x, y) = e^{-xy}; \qquad b(x, y) = e^{xy};$$
$$g(x, y) = \gamma(x + y); \qquad f(x, y) = \frac{1}{1 + x + y}.$$

Discretizing the operator (24) by centered differences with mesh size $h = 1/(p + 1)$ gives rise to a nonsymmetric matrix A of size $N = p^2$. The parameter γ is useful for varying the degree of symmetry of A.

Taking $p = 30$ and $\gamma = 20$ yields a matrix of dimension 900 which is not nearly symmetric. We computed the four rightmost eigenvalues of A by the Arnoldi-Chebyshev algorithm using $m = 15$ and $n_{\max} = 80$ and obtained the following results:

$\lambda_{1,2} = 9.4429 \pm 1.7290$; Residual norm: 5.4 E − 13,
$\lambda_{3,4} = 8.9561 \pm 1..3381$; Residual norm: 8.4 E − 08.
Total number of matrix by vector multiplications required: 110
CPU time: 26.0 sec.

The initial vector was a random vector, and the stopping criterion was that the residual norm be less than $\varepsilon = 10^{-6}$. Details of the execution are as follows: first an Arnoldi iteration (15 steps) was performed and provided the parameters $d = 3.803$, $c^2 = 14.36$. Then 80 steps of the Chebyshev iterations were carried out and finally another Arnoldi purification step was taken and the stopping criterion was satisfied. Note that we could have reduced the amount of work by using a smaller n in the Chebyshev iteration.

The same eigenvalues were computed by the subspace iteration method using the same stopping criterion and the parameters $m = 8$ and $n_{\max} = 50$. The results were delivered after 220 iterations which consumed 1708 matrix by vector multiplications and 220 CPU seconds.

A similar run with the accelerated subspace iteration, with $n_{\max} = 15$, took 104 iterations corresponding to a total of 928 matrix by vector multiplications and 188 seconds of CPU time. Observe that the gain in execution time does not reflect the gain in the number of matrix by vector multiplications because the overhead in the accelerated subspace iteration is substantial.

We failed to discuss in detail the use of our accelerated algorithms for the computation of the eigenvalues with algebraically smallest real parts, but the development is identical to that for the eigenvalues with largest real parts. It suffices to relabel the eigenvalues in increasing order of their real parts (instead of decreasing order). In the following test we have computed the four eigenvalues of smallest real parts of the matrix A defined above. Convergence has been more difficult to achieve than in the previous test. With $m = 20$, $n_{\max} = 250$, the Arnoldi-Chebyshev code satisfied the convergence criterion with $\varepsilon = 10^{-4}$, after three calls to Arnoldi and a total of 527 matrix by vector multiplications. The execution time was 106 sec. In order to obtain the smallest eigenvalues with the regular Chebyshev iteration, we had to shift A by a certain scalar so that the eigenvalues of smallest real parts become dominant. We used the shift 7.0, i.e.,the subspace iteration algorithm was applied to the shifted matrix $A - 7.I$, and the resulting eigenvalues are shifted back by 7. to obtain the eigenvalue of A. The process with $m = 10$ and $n_{\max} = 50$ was quite slow since it took a total of 3925 matrix by vector multiplications and 525 seconds to reach the same stopping criterion as above.

The accelerated subspace iteration did not perform better, however, since it required 4010 matrix by vector multiplications to converge with a total time of 736 seconds. Here we used $m = 10$ and $n_{\max} = 25$. The reason for this misbehavior was that the algorithm encountered serious difficulties to obtain a good ellipse as could be observed from the erratic variation of the parameters d and c. We believe that one important conclusion from this is that the Chebyshev subspace iteration can become unreliable for some shapes of spectra or when the eigenvalues are clustered in an unfavorable way. If the spectrum is entirely real (or almost real) this misbehavior is unlikely to happen in general. Perhaps, another important remark raised by this last experiment is that fitting a general spectrum with an ellipse may not be the best idea. If we were allowed to use domains E more general than ellipses, then the problem of fitting the spectrum would have been made easier. Clearly, the resulting best polynomials p_n would not be Chebyshev polynomials but this does not constitute a major disadvantage. Further investigation in this direction is worth pursuing.

7. Conclusion. The purpose of this paper was to show one way of using Chebyshev polynomials for accelerating nonsymmetric eigenvalue algorithms. The numerical experiments have confirmed the expectation that such an acceleration can be quite effective. To conclude, we would like to point out the following facts:

●It is not clear that representing general spectra by ellipses is the best that can be done. For the solution of linear systems, general domains have been considered by Smolarski and Saylor [36], [37] who use orthogonal polynomials in the complex plane. In [31] a similar technique is developed for solving nonsymmetric eigenvalue problems.

●Another way of combining polynomial iteration (e.g., Chebyshev iteration) or, more generally rational iteration, with Arnoldi's method has recently been proposed by Ruhe [28]. Briefly described the idea is to carry out Arnoldi's algorithm with the matrix $\phi(A)$, ϕ being a suitably chosen rational function. Then, an ingenious relation enables one to calculate the eigenvalues of A from the Hessenberg matrix built by the Arnoldi process.

●We have selected Arnoldi's method as a purification process, perhaps unfairly to other similar processes which may be just as powerful as Arnoldi's. One such alternative is the unsymmetric Lanczos algorithm [19], [26], [40]. Another possibility which we have failed to describe is a *projection process onto the latest m vectors of the Chebyshev iteration*. This can be realized at less cost than m steps of Arnoldi's method although it is not known whether the overall resulting algorithm is more effective.

●We have dealt with approximate eigenvectors of A, in particular for restarting, but everything can be rewritten in terms of Schur vectors and invariant subspaces as suggested by Stewart [39].

Acknowledgements. This work would not have been possible without the availability of the very useful code written by Thomas A. Manteuffel in [21]. The author is indebted to the referee for many useful suggestions.

Computer Science Department
Yale University
New Haven, Connecticut 06520

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

2. F. L. BAUER, "Das Verfahren der Treppeniteration und Verwandte Verfahren zur Losung Algebraischer Eigenwertprobleme," *Z. Angew. Math. Phys.*, v. 8, 1957, pp. 214–235.

3. A. CLAYTON, *Further Results on Polynomials Having Least Maximum Modulus Over an Ellipse in the Complex Plane*, Technical Report AEEW-7348, UKAEA, 1963.

4. M CLINT & A. JENNINGS, "The evaluation of eigenvalues and eigenvectors of real symmetric matrices by simultaneous iteration method," *J. Inst. Math. Appl.*, v. 8, 1971, pp. 111–121.

5. F. D'ALMEIDA, *Numerical Study of Dynamic Stability of Macroeconomical Models–Software for MODULECO*, Dissertation, Technical Report INPG–University of Grenoble, 1980. (French)

6. E. H. DOWELL, "Nonlinear oscillations of a fluttering plate. II," *AIAA J.*, v. 5, 1967, pp. 1856–1862.

7. H. C. ELMAN, *Iterative Methods for Large Sparse Nonsymmetric Systems of Linear Equations*, Ph.D. thesis, Technical Report 229, Yale University, 1982.

8. H. C. ELMAN, Y. SAAD & P.SAYLOR, *A New Hybrid Chebyshev Algorithm for Solving Nonsymmetric Systems of Linear Equations*, Technical Report, Yale University, 1984.

9. K. K. GUPTA, "Eigensolution of damped structual systems," *Internat. J. Numer. Methods Engrg.*, v. 8, 1974, pp. 877–911.

10. K. K. GUPTA, "On a numerical solution of the supersonic panel flutter eigenproblem," *Internat. J. Numer. Methods Engrg.*, v. 10, 1976, pp. 637–645.

11. A. L. HAGEMAN & D. M. YOUNG, *Applied Iterative Methods*, Academic Press, New York, 1981.

12. A. JENNINGS, "Eigenvalue methods and the analysis of structural vibration," In *Sparse Matrices and Their Uses* (I. S. Duff, ed.), Academic Press, New York, 1981, pp. 109–138.

13. A. JENNINGS & W. J. STEWART, "Simultaneous iteration for partial eigensolution of real matrices," *J. Math. Inst. Appl.*, v. 15, 1980, pp. 351–361.

14. A. JENNINGS & W. J. STEWART, "A simultaneous iteration algorithm for real matrices," *ACM Trans. Math. Software*, v. 7, 1981, pp. 184–198.

15. A. JEPSON, *Numerical Hopf Bifurcation*, Ph.D. thesis, California Institute of Technology, 1982.

16. S. KARLIN, *Mathematical Methods and Theory in Games, Programming, and Economics*, Vol. I, Addison-Wesley, Reading, Mass., 1959.

17. L. KAUFMAN, *Matrix Methods of Queueing Problems*, Technical Report TM-82-11274-1, Bell Laboratories, 1982.

18. L. KLEINROCK, *Queueing Systems*, Vol. 2: *Computer Applications*, Wiley, New York, London, 1976.

19. C. LANCZOS, "An iteration method for the solution of the eigenvalue problem of linear differential and integral operators," *J. Res. Nat. Bur. Standards*, v. 45, 1950, pp. 255–282.

20. A. J. LAUB, *Schur Techniques in Invariant Imbedding Methods for Solving Two Point Boundary Value Problems*, 1982. (To appear.)

21. T. A. MANTEUFFEL, *An Iterative Method for Solving Nonsymmetric Linear Systems with Dynamic Estimation of Parameters*, Ph.D. dissertation, Technical Report UIUCDCS-75-758, University of Illinois at Urbana-Champaign, 1975.

22. T. A. MANTEUFFEL, "The Tchebychev iteration for nonsymmetric linear systems," *Numer. Math.*, v. 28, 1977, pp. 307–327.

23. T. A. MANTEUFFEL, "Adaptive procedure for estimation of parameter for the nonsymmetric Tchebychev iteration," *Numer. Math.*, v. 28, 1978, pp. 187–208.

24. B. NOUR-OMID, B. N. PARLETT & R. TAYLOR, *Lanczos Versus Subspace Iteration for the Solution of Eigenvalue Problems*, Technical Report UCB/SESM-81/04, Dept. of Civil Engineering, Univ. of California at Berkeley, 1980.

25. B. N. PARLETT, *The Symmetric Eigenvalue Problem*, Prentice-Hall, Englewood Cliffs, N. J., 1980.

26. B. N. PARLETT & D. TAYLOR, *A Look Ahead Lanczos Algorithm for Unsymmetric Matrices*, Technical Report PAM-43, Center for Pure and Applied Mathematics, 1981.

27. T. J. RIVLIN, *The Chebyshev Polynomials*, Wiley, New York, 1976.

28. A. RUHE, *Rational Krylov Sequence Methods for Eigenvalue Computations*, Technical Report Uminf-97.82, University of Umea, 1982.

29. H. RUTISHAUSER, "Computational aspects of F. L. Bauer's simultaneous iteration method," *Numer. Math.*, v. 13, 1969, pp. 4–13.

30. Y. SAAD, "Projection methods for solving Large sparse eigenvalue problems," in *Matrix Pencils, Proceedings* (Pitea Havsbad, B. Kagstrom and A. Ruhe, eds.), Lecture Notes in Math., vol. 973, Springer-Verlag, Berlin, 1982, pp. 121–144.

31. Y. SAAD, *Least Squares Polynomials in the Complex Plane with Applications to Solving Sparse Nonsymmetric Matrix Problems*, Technical Report RR-276, Dept. of Computer Science, Yale University, 1983.

32. Y. SAAD, "Variations on Arnoldi's method for computing eigenelements of large unsymmetric matrices," *Linear Algebra Appl.*, v. 34, 1980, pp. 269–295.

33. G. SANDER, C. BON & M. GERADIN, "Finite element analysis of supersonic panel flutter," *Internat. J. Numer. Methods Engrg.*, v. 7, 1973, pp. 379–394.

34. D. H. SATTINGER, "Bifurcation of periodic solutions of the Navier Stokes equations," *Arch. Rational Mech. Anal.*, v. 41, 1971, pp. 68–80.

35. E. SENETA, "Computing the stationary distribution for infinite Markov chains," in *Large Scale Matrix Problems* (A. Björck, R. J. Plemmons & H. Schneider, eds.), Elsevier, North-Holland, New York, 1981, pp. 259–267.

36. D. C. SMOLARSKI, *Optimum Semi-Iterative Methods for the Solution of Any Linear Algebraic System with a Square Matrix*, Ph.D. Thesis, Technical Report UIUCDCS-R-81-1077, University of Illinois at Urbana-Champaign, 1981.

37. D. C. SMOLARSKI & P. E. SAYLOR, *Optimum Parameters for the Solution of Linear Equations by Richardson Iteration*, 1982. Unpublished Manuscript.

38. G. W. STEWART, "Simultaneous iteration for computing invariant subspaces of non-Hermitian matrices," *Numer. Math.*, v. 25, 1976, pp. 123–136.

39. G. W. STEWART, *SRRIT–A FORTRAN Subroutine to Calculate the Dominant Invariant Subspaces of a Real Matrix*, Technical Report TR-514, Univ. of Maryland, 1978.

40. D. TAYLOR, *Analysis of the Look-Ahead Lanczos Algorithm*, Ph.D. thesis, Technical Report, Univ. of California, Berkeley, 1983.

41. J. H. WILKINSON, *The Algebraic Eigenvalue Problem*, Clarendon Press, Oxford, 1965.

42. J. H. WILKINSON & C. REINSCH, *Handbook for Automatic Computation*, Vol. II, *Linear Algebra*, Springer-Verlag, New York, 1971.

43. H. E. WRIGLEY, "Accelerating the Jacobi method for solving simultaneous equations by Chebyshev extrapolation when the eigenvalues of the iteration matrix are complex," *Comput. J.*, v. 6, 1963, pp. 169–176.