THE EHRLICH-ABERTH METHOD FOR THE NONSYMMETRIC TRIDIAGONAL EIGENVALUE PROBLEM [*]

DARIO A. BINI [†], LUCA GEMIGNANI [‡], AND FRANÇOISE TISSEUR [§]

Abstract. An algorithm based on the Ehrlich-Aberth iteration is presented for the computation of the zeros of $p(\lambda) = \det(T - \lambda I)$, where T is an irreducible tridiagonal matrix. The algorithm requires the evaluation of $p(\lambda)/p'(\lambda) = -1/\mathrm{trace}(T - \lambda I)^{-1}$, which is done here by exploiting the QR factorization of $T - \lambda I$ and the semiseparable structure of $(T - \lambda I)^{-1}$. Two choices of the initial approximations are considered; the most effective relies on a divide-and-conquer strategy, and some results motivating this strategy are given. A Fortran 95 module implementing the algorithm is provided and numerical experiments that confirm the effectiveness and the robustness of the approach are presented. In particular, comparisons with the LAPACK subroutine `dhseqr` show that our algorithm is faster for large dimensions.

Key words. nonsymmetric eigenvalue problem, symmetric indefinite generalized eigenvalue problem, tridiagonal matrix, root finder, QR decomposition, divide and conquer.

AMS subject classifications. 65F15

1. Introduction.

Nonsymmetric tridiagonal eigenvalue problems arise as intermediate steps in a variety of eigenvalue problems. For example, the nonsymmetric eigenvalue problem can be reduced in a finite number of steps to nonsymmetric tridiagonal form [11], [14]. In the sparse case, the nonsymmetric Lanczos algorithm produces a nonsymmetric tridiagonal matrix. Other motivation for this work comes from the symmetric quadratic eigenvalue problem

$$(\lambda^2 M + \lambda D + K)x = 0, \quad M^T = M,\ D^T = D,\ K^T = K,$$

which is frequently encountered in structural mechanics [28]. The standard way of dealing with this problem in practice is to reformulate it as a generalized eigenvalue problem (GEP) $Ax = \lambda Bx$ of twice the dimension, a process called linearization. Symmetry in the problem can be maintained with an appropriate choice of linearization [28], such as, for example,

$$A = \begin{bmatrix} 0 & K \\ K & C \end{bmatrix}, \quad B = \begin{bmatrix} K & 0 \\ 0 & -M \end{bmatrix}, \quad x = \begin{bmatrix} u \\ \lambda u \end{bmatrix}.$$

The resulting A and B are symmetric but not definite, and in general the pair (A, B) is indefinite. When the pair (A, B) is of small to medium size, it can be reduced to a symmetric tridiagonal-diagonal pair (S, D) using one of the procedures described by Tisseur [27]. This is the most compact form that can be obtained in a finite number of steps. For large and sparse matrices the pseudo-Lanczos algorithm of Parlett and Chen [24] applied to $A - \lambda B$, yields a projected problem $S - \lambda D$ with S symmetric

[*]Numerical Analysis Report 428, Manchester Centre for Computational Mathematics, May 2003.

[†]Dipartimento di Matematica, Università di Pisa, via Buonarroti 2, 56127 Pisa (`bini@dm.unipi.it`). This work was supported by MIUR, grant number 2002014121

[‡]Dipartimento di Matematica, Università di Pisa, via Buonarroti 2, 56127 Pisa (`gemignan@dm.unipi.it`). This work was supported by GNCS of Istituto Nazionale Di Alta Matematica.

[§]Department of Mathematics, University of Manchester, Manchester, M13 9PL, England (`ftisseur@ma.man.ac.uk`, `http://www.ma.man.ac.uk/~ftisseur/`). This work was supported by Engineering and Physical Sciences Research Council grant GR/L76532 and Nuffield Foundation grant NAL/00216/G.

tridiagonal and D diagonal. In both cases, the eigenvalues of the symmetric pair (S, D) are the same as the eigenvalues of the nonsymmetric tridiagonal matrix $T = D^{-1}S$.

Our aim is to derive a robust algorithm that computes all the eigenvalues of T in $O(n^2)$ operations. The QR algorithm [15] does not preserve tridiagonal structure: the matrix T is considered as a Hessenberg matrix and the upper part of T is filled in along the iterations. Therefore the QR algorithm requires some extra storage and the eigenvalues are computed in $O(n^3)$ operations. Two alternatives are the LR algorithm [26] for nonsymmetric tridiagonal matrices and the HR algorithm [7], [8]. Both algorithms preserve the tridiagonal form of T but may be unstable as they use non-orthogonal transformations. Attempts to solve the nonsymmetric tridiagonal eigenvalue by generalizing Cuppen's divide and conquer algorithm have been unsuccessful because of a lack of good root finders and because deflation is not as advantageous as it is in the symmetric case [2], [19].

In this paper we propose a root finder for the characteristic polynomial of T based on the Ehrlich-Aberth method [1], [12]. This method approximates simultaneously all the zeros of a polynomial $p(z)$: given a vector $z^{(0)} \in \mathbb{C}^n$ of initial approximations to the zeros of $p(z)$, the Ehrlich-Aberth iteration generates a sequence $z^{(j)} \in \mathbb{C}^n$ which locally converges to the n-tuple of the roots of $p(z)$, according to the equation

$$z_j^{(k+1)} = z_j^{(k)} - \frac{\frac{p(z_j^{(k)})}{p'(z_j^{(k)})}}{1 - \frac{p(z_j^{(k)})}{p'(z_j^{(k)})} \sum_{k=1, k\neq j}^{n} \frac{1}{z_j^{(k)} - z_k^{(k)}}}, \quad j = 1{:}\, n. \tag{1.1}$$

The convergence is superlinear (cubic or even higher if the implementation is in the Gauss-Seidel style) for simple roots and linear for multiple roots. In practice, the Ehrlich-Aberth iteration has good global convergence properties, though no theoretical results seems to be known about global convergence. The main requirements when using the Ehrlich-Aberth method for computing the roots of $p(z)$ are

1. A fast, robust and stable computation of the Newton correction $p(z)/p'(z)$.
2. A criterion for choosing the initial approximations to the zeros, $z^{(0)}$, so that the number of iterations needed for convergence is not too large.

For the first issue, Bini [4] shows that Horner's rule is an effective tool when $p(z)$ is expressed in terms of its coefficients. In this case the cost of each simultaneous iteration is $O(n^2)$ operations. Moreover Horner's rule is backward stable and its computation provides a cheap criterion to test if the given approximation is in the root-neighborhood (pseudospectrum) of the polynomial [4]. This makes the Ehrlich-Aberth method an effective tool for approximating polynomial roots [6] and it is now part of the MPSolve package (Multiprecision Polynomial Solver) [5].

In our context, where $p(\lambda) = \det(T - \lambda I)$ is not available explicitly, we need a tool having the same features as Horner's rule, that is, a tool that allows us to compute in a fast, stable and robust way the Newton correction $p(\lambda)/p'(\lambda)$. This issue is discussed in section 2 where we use the QR factorization of $T - \lambda I$ and the semiseparable structure of $(T - \lambda I)^{-1}$ to compute the Newton correction by means of the equation

$$\frac{p(\lambda)}{p'(\lambda)} = -\frac{1}{\operatorname{trace}(T - \lambda I)^{-1}}.$$

The algorithm that we obtain in this way fulfills the desired requirements of robustness and stability. It does not have any difficulty caused by underflow and overflow problems.

Two approaches are considered in section 3 for the second issue concerning the choice of initial approximations. Following Bini [4] and Bini and Fiorentino [6], we first apply a criterion based on Rouché's theorem and on the Newton polygon, which is particularly suited for matrices having eigenvalues of both large and small moduli. However, the specific features of our eigenvalue problem motivate a divide and conquer strategy: the initial approximations are obtained by computing the eigenvalues of two suitable tridiagonal matrices of sizes $m = \lceil n/2 \rceil$ and $n-m$. Even though there are no theoretical results guaranteeing convergence under this choice, we provide in section 3 some theoretical results that motivate this strategy.

The complete algorithm is described in section 4, where we also deal with the issues of computing eigenvectors and running error bounds. Numerical experiments in section 5 illustrate the robustness of our algorithm. In particular, our results show that in most cases our algorithm performs faster than the LAPACK subroutine `dhseqr` already for $n \geq 800$ and the speed-up for $n = 1600$ ranges from 3 to 70. The implementation in Fortran 95 is available as a module at `www.dm.unipi.it/~bini/software`.

2. Computing the Newton correction.

Our aim in this section is to derive a fast, robust and stable method for computing the Newton correction $p(\lambda)/p'(\lambda)$, where $p(\lambda) = \det(T - \lambda I)$.

The tridiagonal matrix

$$T = \begin{bmatrix} \alpha_1 & \gamma_1 & & & 0 \\ \beta_1 & \alpha_2 & \gamma_2 & & \\ & \beta_2 & \ddots & \ddots & \\ & & \ddots & \alpha_{n-1} & \gamma_{n-1} \\ 0 & & & \beta_{n-1} & \alpha_n \end{bmatrix} \in \mathbb{R}^{n\times n} \tag{2.1}$$

is said to be *unreduced* or *irreducible* if $\beta_i\gamma_i \neq 0$ for $i = 1{:}\,n-1$. We denote by T_k the leading principal submatrix of T in rows and columns 1 through k and let $p_k = \det(T_k - \lambda I)$.

A natural approach is to compute $p(\lambda) = p_n(\lambda)$ and its derivative by using the recurrence

$$\begin{aligned} &p_0(\lambda) = 1, \\ &p_k(\lambda) = (\alpha_k - \lambda\sigma_k)p_{k-1}(\lambda) - \beta_{k-1}\gamma_{k-1}p_{k-2}(\lambda), \quad k = 2{:}\,n, \end{aligned} \tag{2.2}$$

obtained by expanding $\det(T_k - \lambda I_k)$ by its last row. Since this recurrence is known to suffer from overflow and underflow problems [23], we adopt a different strategy.

Assume that λ is not a zero of p, that is, $p(\lambda) \neq 0$. Then

$$\frac{p'(\lambda)}{p(\lambda)} = -\sum_{i=1}^{n} \frac{1}{\lambda_i - \lambda} = -\mathrm{trace}\big((T-\lambda I)^{-1}\big) = -\sum_{i=1}^{n}\theta_i, \tag{2.3}$$

where θ_i is the ith diagonal element of $(T - \lambda I)^{-1}$.

In what follows, S denotes the shifted tridiagonal matrix

$$S := T - \lambda I.$$

If S is unreduced, S^{-1} can be characterized in terms of two vectors $u = [u_1, \ldots, u_n]^T$ and $v = [v_1, \ldots, v_n]^T$ such that

$$(S^{-1})_{ij} = \begin{cases} u_i v_j \text{ if } i \leq j, \\ u_j v_i \text{ otherwise.} \end{cases} \tag{2.4}$$

We refer to Meurant's survey on the inverse of tridiagonal matrices [21]. If we set $u_1 = 1$, the vectors u and v can be computed in $O(n)$ operations by solving

$$Sv = e_1, \qquad v_n S u = e_n,$$

where $v_n \neq 0$ since S is irreducible. It is tempting to use the vectors u and v representing S^{-1} for the computation of the Newton's correction via

$$p(\lambda)/p(\lambda)' = \sum_{i=1}^{n} u_i v_i.$$

However, as illustrated in [18], u and v can be extremely badly scaled and their computation can break down because of overflow and underflow. In the next two subsections, we describe two robust and efficient approaches for computing the trace of the inverse of a tridiagonal matrix and discuss our choice.

2.1. Dhillon's approach. Dhillon [10] proposes an algorithm to compute the 1-norm of the inverse of a tridiagonal matrix S in $O(n)$ operations that is more reliable than Higham's algorithm [17] based on the compact representation (2.4). As a by-product, Dhillon's approach provides trace(S^{-1}). His algorithm relies on the computation of the two triangular factorizations

$$S = L_+ D_+ U_+, \qquad S = U_- D_- L_-, \tag{2.5}$$

where L_+ and L_- are unit lower bidiagonal, U_+ and U_- are unit upper bidiagonal, while $D_+ = \mathrm{diag}(d_1^+, \ldots, d_n^+)$ and $D_- = \mathrm{diag}(d_1^-, \ldots, d_n^-)$. If these factorizations exist, the diagonal entries of S^{-1} denoted by θ_i, $i = 1{:}\,n$, can be expressed in terms of the diagonal factors D_+ and D_- through the recurrence

$$\theta_1 = 1/d_1^-, \qquad \theta_{i+1} = \theta_i \frac{d_i^+}{d_{i+1}^-}, \quad i = 1{:}\,n-1. \tag{2.6}$$

Note that the triangular factorizations (2.5) may suffer element growth. They can also break down prematurely if a zero pivot is encountered, that is, if $d_i^+ = 0$ or $d_{i+1}^- = 0$ for some i. To overcome this latter drawback, Dhillon [10] makes use of IEEE floating point arithmetic, which permits computations with $\pm\infty$. With this approach, Dhillon's algorithm always returns an approximation of trace(S^{-1}).

In our implementation of the Ehrlich-Aberth method the computation of $\det(S)$ is needed at the last stage of the algorithm to provide an error bound for the computed eigenvalues (see section 4). As D_+ and D_- may have 0 and $\pm\infty$ entries Dhillon's algorithm cannot be used to evaluate

$$\det(S) = \prod_{i=1}^{n} d_i^+ = \prod_{i=1}^{n} d_i^-$$

since $0 \times \pm\infty$ does not make sense mathematically and produces a NaN (Not a Number) in extended IEEE floating point arithmetic.

2.2. A QR factorization approach. In this section we present an alternative algorithm for the computation of trace(S^{-1}) in $O(n)$ operations that is based on the properties of QR factorizations of tridiagonal matrices. Our algorithm keeps the element growth under control and does not have any difficulty caused by overflow and underflow, so there is no need to augment the algorithm with tests for dealing with degenerate cases as in [10]. In addition, it provides $\det(S)$.

Recall that in our application S is the shifted tridiagonal matrix $T - \lambda I$, where T is given by (2.1). Since λ can be complex, S has real subdiagonal and superdiagonal elements and complex diagonal elements. We denote by G_i the $n \times n$ unitary Givens rotation equal to the identity matrix except in rows and columns i and $i+1$, where

$$G_i([i,i+1],[i,i+1]) = \begin{bmatrix} \phi_i & \psi_i \\ -\bar{\psi}_i & \bar{\phi}_i \end{bmatrix}, \quad |\phi_i|^2 + |\psi_i|^2 = 1.$$

Let $S = QR$ be the QR factorization of S obtained by means of Givens rotations, so that,

$$G_{n-1} \cdots G_2 G_1 S = R \quad \text{and} \quad Q^* = G_{n-1} \cdots G_2 G_1. \tag{2.7}$$

Since S is tridiagonal, R is an upper triangular matrix of the form

$$R = \begin{bmatrix} r_1 & s_1 & t_1 & & 0 \\ & \ddots & \ddots & \ddots & \\ & & r_{n-2} & s_{n-2} & t_{n-2} \\ & & & r_{n-1} & s_{n-1} \\ 0 & & & & r_n \end{bmatrix}.$$

If $\phi_1 = \bar{\alpha}_1 \tau_1$ and $\psi_1 = \beta_1 \tau_1$ with $\tau_1 = 1/\sqrt{|\alpha_1|^2 + \beta_1^2}$, then

$$S_1 := G_1 S = \left[\begin{array}{c|cccc} r_1 & s_1 & t_1 & 0 & \dots \\ \hline 0 & \widetilde{\alpha}_2 & \widetilde{\gamma}_2 & & 0 \\ \vdots & \beta_2 & \alpha_3 & \gamma_3 & \\ \vdots & 0 & \ddots & \ddots & \ddots \end{array}\right]$$

with

$$\begin{aligned} &r_1 = \phi_1 \alpha_1 + \psi_1 \beta_1, \quad && s_1 = \phi_1 \gamma_1 + \psi_1 \alpha_2, \quad t_1 = \psi_1 \gamma_2, \\ &\widetilde{\alpha}_2 = -\psi_1 \gamma_1 + \bar{\phi}_1 \alpha_2, \quad && \widetilde{\gamma}_2 = \bar{\phi}_1 \gamma_2. \end{aligned}$$

Recursively applying the same transformation to the $(n-1)\times(n-1)$ trailing principal submatrix of S_1 yields the factorization (2.7), where

$$\begin{aligned} &\tau_i = 1/\sqrt{|\widetilde{\alpha}_i|^2 + \beta_i^2}, && \phi_i = \overline{\widetilde{\alpha}}_i \tau_i, && \psi_i = \beta_i \tau_i, \\ &r_i = \phi_i \widetilde{\alpha}_i + \psi_i \beta_i, && s_i = \phi_i \widetilde{\gamma}_i + \psi_i \alpha_{i+1}, && t_i = \psi_i \gamma_{i+1}, \\ &\widetilde{\alpha}_{i+1} = -\psi_i \widetilde{\gamma}_i + \bar{\phi}_i \alpha_{i+1}, && \widetilde{\gamma}_{i+1} = \bar{\phi}_i \gamma_{i+1}, && \end{aligned} \tag{2.8}$$

for $i = 1{:}\,n-1$, with $\widetilde{\alpha}_1 = \alpha_1$ and $\widetilde{\gamma}_1 = \gamma_1$. Note that all the ψ_i are real, and if S is unreduced, the ψ_i are nonzero.

The following result concerns the semiseparable structure of Q^* and is crucial to compute the diagonal entries of S^{-1} in $O(n)$ arithmetic operations.

THEOREM 2.1. *Let $S \in \mathbb{C}^{n\times n}$ be tridiagonal and unreduced and let $S = QR$ be its QR factorization computed according to* (2.8)*. Define*

$$\begin{aligned} &D = \operatorname{diag}(1, -\psi_1, \psi_1\psi_2, \dots, (-1)^{n-1}\psi_1\psi_2\cdots\psi_{n-1}), \\ &u = D^{-1}[1, \bar{\phi}_1, \bar{\phi}_2, \dots, \bar{\phi}_{n-1}]^T, \\ &v = D[\phi_1, \phi_2, \phi_3, \dots, \phi_{n-1}, 1]^T. \end{aligned} \tag{2.9}$$

Then

$$Q^* = \begin{bmatrix} v_1u_1 & \psi_1 & & & 0 \\ v_2u_1 & v_2u_2 & \psi_2 & & \\ \vdots & \vdots & \ddots & \ddots & \\ \vdots & & & v_{n-1}u_{n-1} & \psi_{n-1} \\ v_nu_1 & v_nu_2 & \cdots & v_nu_{n-1} & v_nu_n \end{bmatrix}.$$

Proof. We proceed by induction on n. For $n = 2$ the theorem trivially holds. Assume that the result holds for $n-1$, that is,

$$\widetilde{Q}^*_{n-1} = \widetilde{G}_{n-2}\cdots\widetilde{G}_2\widetilde{G}_1 = \begin{bmatrix} \widetilde{v}_1\widetilde{u}_1 & \psi_1 & & 0 \\ \widetilde{v}_2\widetilde{u}_1 & \ddots & \ddots & \\ \vdots & & \ddots & \psi_{n-2} \\ \widetilde{v}_{n-1}\widetilde{u}_1 & \cdots & \widetilde{v}_{n-1}\widetilde{u}_{n-2} & \widetilde{v}_{n-1}\widetilde{u}_{n-1} \end{bmatrix} \in \mathbb{C}^{(n-1)\times(n-1)},$$

where $\widetilde{G}_i$, is the matrix G_i with its last row and column removed, and

$$\widetilde{u} = \operatorname{diag}(1, -\psi_1, \psi_1\psi_2, \ldots, (-1)^{n-2}\psi_1\psi_2\cdots\psi_{n-2})^{-1}[1, \bar{\phi}_1, \bar{\phi}_2, \ldots, \bar{\phi}_{n-2}]^T \in \mathbb{C}^{n-1},$$
$$\widetilde{v} = \operatorname{diag}(1, -\psi_1, \psi_1\psi_2, \ldots, (-1)^{n-2}\psi_1\psi_2\cdots\psi_{n-2})[\phi_1, \phi_2, \ldots, \phi_{n-2}, 1]^T \in \mathbb{C}^{n-1}.$$

Since

$$Q^*_n = G_{n-1}\begin{bmatrix} \widetilde{Q}^*_{n-1} & 0 \\ 0 & 1 \end{bmatrix} := G_{n-1}Q_{n-1},$$

the first $n-2$ rows of Q^*_{n-1} and Q^*_n coincide. Note that

$$u(1\colon n-1) = \widetilde{u}, \quad v_{n-1} = \phi_{n-1}\widetilde{v}_{n-1}, \quad v_n = -\psi_{n-1}\widetilde{v}_{n-1}, \quad \bar{\phi}_{n-1} = u_nv_n.$$

If e_k denotes the kth column of the identity matrix, we have

$$\begin{aligned} e^*_{n-1}Q^*_n &= \phi_{n-1}\widetilde{v}_{n-1}[\widetilde{u}^T, 0] + \psi_{n-1}e^*_n \\ &= [v_{n-1}u_1, v_{n-1}u_2, \cdots, v_{n-1}u_{n-1}, \psi_{n-1}]^T \end{aligned}$$

and

$$\begin{aligned} e^*_nQ^*_n &= -\psi_{n-1}\widetilde{v}_{n-1}[\widetilde{u}^T, 0] + \bar{\phi}_{n-1}e^*_n \\ &= [v_nu_1, v_nu_2, \cdots, v_nu_{n-1}, v_nu_n]^T \end{aligned}$$

which completes the proof. □

For simplicity, we assume that S is nonsingular so that R^{-1} exists. Let w be the solution of the system $Rw = v$, where v is defined in (2.9). Then, using Theorem 2.1 and the fact that R is triangular, the jth diagonal elements of S^{-1}, θ_j, is given by

$$\theta_j = e^*_jS^{-1}e_j = e^*_jR^{-1}Q^*e_j = u_je^*_jR^{-1}v = u_je^*_jw = u_jw_j,$$

and hence

$$\operatorname{trace}(S^{-1}) = \sum_{j=1}^{n} u_jw_j.$$

Observe that the computation of u and v by mean of (2.9) generates underflow and overflow problems: since the diagonal entries of D are products of the ψ_i with $|\psi_i| \le 1$, then for large n, D may have diagonal entries that underflow to zero and inverting D would generate overflow. A way of avoiding this drawback is by scaling the system $Rw = v$ with the diagonal matrix D of Theorem 2.1. This yields $\widehat{R}\widehat{w} = \widehat{v}$, where

$$\widehat{R} = D^{-1}RD, \quad \widehat{w} = D^{-1}w, \quad \widehat{v} = D^{-1}v = [\phi_1, \dots, \phi_{n-1}, 1]^T. \tag{2.10}$$

With this scaling, no accumulation of products of ψ_i is needed. The entries of the matrix $\widehat{R}$ are given by

$$\widehat{r}_i = r_i, \quad \widehat{s}_i = -\psi_i s_i, \quad \widehat{t}_i = \psi_i \psi_{i+1} t_i \tag{2.11}$$

and their computation does not generate overflow since $|\psi_i| \le 1$. Underflow in the computation of $\widehat{s}_i$ and $\widehat{t}_i$ is not a problem since their inverses are not needed in the solution of $\widehat{R}\widehat{w} = \widehat{v}$. The only terms that must be inverted in the computation of $\widehat{w}$ are the diagonal elements of $\widehat{R}$. Since $\|\widehat{v}\|_\infty = 1$, we have $\|\widehat{w}\|_\infty \le \|R^{-1}\|_\infty$. Overflow in the computation of the $\widehat{w}_i$ implies that $\widehat{R}$ and therefore $S = T - \lambda I$ is numerically singular. In that case we have detected an eigenvalue. Let

$$\widehat{u} = [1, \bar{\phi}_1, \bar{\phi}_2, \dots, \bar{\phi}_{n-1}]^T. \tag{2.12}$$

Note that because $|\phi_i|^2 + |\psi_i|^2 = 1$, the components of $\widehat{u}$ are all bounded by 1 in modulus. Since $w = D\widehat{w}$ and $u = D^{-1}\widehat{u}$, we find that $u_i w_i = \widehat{u}_i \widehat{w}_i$, $i = 1, \dots, n$, so that

$$\operatorname{trace}(S^{-1}) = \sum_{j=1}^{n} \widehat{u}_j \widehat{w}_j \tag{2.13}$$

and

$$\det(S) = \prod_{i=1}^{n} r_i. \tag{2.14}$$

Equations (2.8) and (2.10)–(2.13) constitute our algorithm for the computation of $p'(\lambda)/p(\lambda) = \operatorname{trace}\big((T - \lambda I)^{-1}\big) = \operatorname{trace}(S^{-1})$, which we summarize below in pseudocode. The function *Givens* constructs ϕ_i and ψ_i and guards against the risk of overflow. We refer to Bindel et al. [3] for a detailed explanation on how this function should be implemented.

```
function $\tau = trace\_Tinv(\beta, \alpha, \gamma)$
% Compute $\tau = trace(S^{-1})$, where $S = \text{tridiag}(\beta, \alpha, \gamma)$ is $n \times n$ tridiagonal
% with real off-diagonals and complex diagonal.
$a = \alpha_1$, $g = \gamma_1$, $u_1 = 1$
%Computes vectors $\widehat{r}$, $\widehat{s}$, $\widehat{t}$ in (2.11), $\widehat{u}$ in (2.12) and $\widehat{v}$ in (2.10).
for $i = 1 : n-1$
    $(\phi, \psi) = Givens(a, \beta_i)$
    $r_i = \phi a + \psi \beta_i$, $s_i = -\psi(\phi g + \psi \alpha_{i+1})$
    $a = -\psi g + \bar{\phi} \alpha_{i+1}$, $u_{i+1} = \bar{\phi}$, $v_i = \phi$
    if $i < n-1$, $t_i = \psi^2 \gamma_{i+1}$, $g = \bar{\phi} \gamma_{i+1}$, end
    if $i > 1$, $t_{i-1} = \psi t_{i-1}$, end
```

```
end
$r_n = a$, $v_n = 1$
% Solve the linear system $\widehat{R}\widehat{w} = \widehat{v}$.
$w_n = v_n/r_n$
$w_{n-1} = (v_{n-1} - w_n s_{n-1})/r_{n-1}$
for $i = n-2 : -1 : 1$
        $w_i = (v_i - w_{i+1}s_i - w_{i+2}t_i)/r_i$
        if $w_i$ = Inf, Tr = Inf, return, end
end
$\tau = \sum_{i=1}^{n} u_i w_i$
```

The function *trace_Tinv* requires $O(n)$ operations.

3. Choosing initial approximations.

The choice of the initial approximations $z_j^{(0)}$, $j = 1{:}\,n$, used to start the Ehrlich-Aberth iteration (1.1) crucially affects the number of steps needed by the method. We consider two approaches, one based on Rouché's theorem and one based on a divide and conquer strategy. The former is well suited for matrices having eigenvalues with both large and small moduli, while the latter better exploits the tridiagonal nature of the problem and seems to perform better in practice.

3.1. Criterion based on Rouché's theorem.

Here we recall a criterion for selecting initial approximations that has been introduced by Bini [4] and is based on a combination of Rouché's theorem and the use of the Newton polygon.

Let $p(x) = \sum_{j=0}^{n} a_j x^j$ be a polynomial of degree n with $a_0 \neq 0$ and let

$$q_\ell(x) = \sum_{j=0}^{n} |a_j| x^j - 2|a_\ell| x^\ell, \quad 0 \le \ell \le n.$$

Observe that, if θ is a positive zero of $q_\ell(x)$ then $|a_\ell|\theta^\ell = \sum_{i=0, i\neq\ell}^{n} |a_i|\theta^i > |a_j|\theta^j$ for any $j \neq \ell$. Whence

$$\begin{aligned} & u_\ell < \theta < v_\ell, \\ & u_\ell = \max_{i<\ell} \left|\frac{a_i}{a_\ell}\right|^{\frac{1}{\ell - i}}, \quad v_\ell = \min_{i>\ell} \left|\frac{a_\ell}{a_i}\right|^{\frac{1}{i-\ell}}, \end{aligned} \tag{3.1}$$

where, u_ℓ and v_ℓ are finite and nonzero since $a_0, a_n \neq 0$. Ostrowski [22] pointed out that if $0 < \ell < n$, the polynomial $q_\ell(x)$ can have either two positive real roots, say $s_\ell \le t_\ell$ or no positive roots and, if $\ell = 0$ or $\ell = n$, there exists only one positive root, denoted by t_0 and s_n, respectively. Let

$$0 = \ell_0 < \ell_1 < \ell_2 < \cdots < \ell_k < \ell_{k+1} = n$$

be the values of ℓ for which $q_\ell(x) = 0$ has positive real roots. Then it holds that

$$t_0 = t_{\ell_0} \le s_{\ell_1} \le t_{\ell_1} \le \cdots \le s_{\ell_k} \le t_{\ell_k} \le s_{\ell_{k+1}} = s_n.$$

THEOREM 3.1. *The closed annulus* $\mathcal{A}_j = \{z \in \mathbb{C} : t_{\ell_j} \le |z| \le s_{\ell_{j+1}}\}$, $0 \le j \le k$ *contains* $\ell_{j+1} - \ell_j$ *roots of* $p(x)$ *whereas the open annulus of radii* s_{ℓ_j}, t_{ℓ_j}, $0 \le j \le k+1$ *contains no roots of* $p(x)$, *where* $s_0 = 0$ *and* $t_n = +\infty$.

Proof. See Ostrowski [22]. □

The inclusion results in Theorem 3.1 can be used to determine a set of initial points for the Ehrlich-Aberth iterations that, unlike the criterion in [1], selects complex numbers along different circles. For example, we may choose $\ell_{j+1} - \ell_j$ equispaced points on the circle of radius $s_{\ell_{j+1}}$. Unfortunately, the indices ℓ_j and roots s_{ℓ_j}, t_{ℓ_j}, $j = 0{:}\,k$ are expensive to compute. However, from (3.1) we deduce that the roots s_{ℓ_j}, t_{ℓ_j}, must belong to the interval (u_{ℓ_j}, v_{ℓ_j}) such that $u_{\ell_j} < v_{\ell_j}$. Therefore, computing all the values r_j, $j = 1{:}\,h$ for which $u_{r_j} \le v_{r_j}$, $j = 1{:}\,h$, provides a superset of $\{\ell_0, \dots, \ell_{k+1}\}$ together with the values of u_{r_i} and v_{r_i} which yield bounds to the roots s_{ℓ_j}, t_{ℓ_j}. With the help of the Newton polygon, Bini [4] computes the set $\{r_0 = 0, r_1, \dots, r_h, r_{h+1} = n\}$ and provides a cheap way to select initial approximations to the roots of $p(x)$ as summarized below.

The upper convex hull of the set

$$\mathcal{S} = \{(\ell, \log|a_\ell|) \in \mathbb{R}^2 : \quad a_\ell \neq 0, \quad \ell = 0{:}\,n\}$$

is the set of points $(r_j, \log|a_{r_j}|)$, $0 = r_0 < r_1 < \cdots < r_h < r_{h+1} = n$ such that the piecewise linear function obtained by joining the points $(r_j, \log|a_{r_j}|)$, $(r_{j+1}, \log|a_{r_{j+1}}|)$, $j = 0{:}\,h$ is convex and lies above the points $(\ell, \log|a_\ell|)$, $\ell = 0{:}\,n$ (Newton's polygon). We recall that the computation of the upper convex hull is inexpensive: it can be carried out in $O(n \log n)$ arithmetic operations and comparisons.

Theorem 3.2. *Let u_ℓ and v_ℓ, $\ell = 1{:}\,n-1$ be defined in* (3.1)*, moreover let*

$$u_0 = (1 + \max_{\ell > 0} |a_\ell / a_0|)^{-1}, \quad v_n = 1 + \max_{\ell < n} |a_\ell / a_n|\,.$$

Denote $r_0, r_1, \dots, r_{h+1}$ the abscissas of the upper convex hull of $\mathcal{S}$. Then the following implication holds: $u_\ell \le v_\ell$ if and only if $\ell \in \{r_1, \dots, r_k\}$, moreover

$$\{\ell_0, \ell_1, \dots, \ell_k, \ell_{k+1}\} \subset \{r_0, r_1, \dots, r_h, r_{h+1}\}\,,$$
$$u_{\ell_j} < s_{\ell_j} \le t_{\ell_j} < v_{\ell_j}, \quad 0 \le j \le k+1,$$
$$v_{r_j} = u_{r_{j+1}} = \left| \frac{a_{r_j}}{a_{r_{j+1}}} \right|^{1/(r_{j+1} - r_j)}, \quad 0 \le j \le h.$$

Proof. See Bini [4, Sec.2]. □

According to the above theorem, the set $\{r_0, r_1, \dots, r_{h+1}\}$ and the values v_{r_j}, $j = 0 : h$, which can be computed at a low cost, provide us with information about the radii t_{ℓ_j} and $s_{\ell_{j+1}}$ of the annuli $\mathcal{A}_j$ which contain $\ell_{j+1} - \ell_j$ roots of $p(x)$ for $j = 0 : k$. This result avoids the expensive task of computing the roots s_ℓ and t_ℓ of the polynomial $q_\ell(x)$ for $\ell = \ell_j$, $j = 0 : k$. More precisely, if

$$u_{\ell_1} < s_{\ell_1} \le t_{\ell_1} < u_{\ell_1} = u_{r_1} \le u_{r_2} \le \cdots \le u_{r_q} = u_{\ell_2} < s_{\ell_2} \le t_{\ell_2}$$

then, choosing initial approximations along the circles of radii $u_{r_1} \le u_{r_2} \le \cdots \le u_{r_q}$ provides approximations inside the annulus formed by the radii t_{ℓ_1} and s_{ℓ_2} in accordance with Theorem 3.1.

Hence we use the following semi-randomized criterion for choosing initial approximations to all the roots of the polynomial $p(x)$:

For $j = 0{:}\,n$, we select $n_j = r_{j+1} - r_j$ complex numbers of moduli v_{r_j} according to the formula

$$z^{(0)}_{r_j + k} = v_{r_j} e^{i\theta_k}, \quad \theta_k = \frac{2\pi k}{n_j} + \frac{2\pi k}{n} + \sigma, \quad k = 0{:}\,n_j - 1,$$

TABLE 3.1
Rouché based criterion: the initial approximations are chosen along circles of radius v_{r_i}.

ℓ	$[s_\ell, t_\ell]$	$\lvert\lambda\rvert$	v_r		r
0	$(0, 0.00098)$	0.001	0.0014	(2)	0
		0.0029			
		0.0029	0.005	(2)	2
		0.0069			
4	$(0.0072, 94.43)$	99.99	97.08	(1)	4
5	$(100.057, 2562.6)$	10000	3399	(1)	5
		10000	10066	(1)	6
		10000	30100	(1)	7
8	$(38634.5, \infty)$				8

for a random $\sigma \in [-\pi, \pi]$.

With this criterion, the number of initial approximations chosen in the annulus $\mathcal{A}(t_{\ell_j}, s_{\ell_{j+1}})$ coincides with the number of roots of $p(x)$ in $\mathcal{A}(t_{\ell_j}, s_{\ell_{j+1}})$, $j = 0{:}\, k$. The above criterion is particularly effective for polynomials having roots with very different moduli. For more details and for the theoretical tools on which the criterion is based we refer the reader to Bini [4].

In order to apply this criterion to our eigenvalue problem we have to devise an efficient technique for computing $\log |a_i|$, where the a_i's are the coefficients of $p(\lambda) = \det(T - \lambda I)$. Observe that $a_0 = p(0)$ and, more generally, $a_\ell = p^{(\ell)}(0)/\ell!$, $\ell = 0{:}\, n$. The quantities $p^{(\ell)}(0)$ can be computed by differentiating the recurrence (2.2) ℓ times at a cost of $O(n^2)$ operations. However, to avoid overflow and underflow problems, it is more convenient to evaluate $p(\lambda)$ at the 2^k roots of unity for $2^k > n$ and then interpolate these values by means of the FFT in order to recover the coefficients a_ℓ. Note that a robust way for evaluating $p(\lambda)$, based on the QR factorization of $T - \lambda I$ and on (2.14), is presented in section 2.2.

As an example consider the 8×8 matrix $T = \mathrm{tridiag}(\beta, \alpha, \gamma)$ with

$$\begin{aligned}
\beta &= [-1, 40, -1, 40, -1, 40, -1, 40],\\
\alpha &= [10^{-3}, 10^4, 10^{-3}, 10^4, 10^{-3}, 10^4, 10^{-3}, 10^2],\\
\gamma &= [1, 1, 1, 1, 1, 1, 1].
\end{aligned}$$

The set of indices for which equation $q_\ell(x) = 0$ has solution is $\{\ell_0, \ell_1, \ell_2, \ell_3\} = \{0, 4, 5, 8\}$ and the indices of the vertices of the Newton polygon are $\{r_0, \dots, r_6\} = \{0, 2, 4, 5, 6, 7, 8\}$, moreover $\{v_0, \dots, v_7\} = \{0.0014, 0.005, 97.08, 3399, 10066, 30100\}$, while the moduli of the eigenvalues of T are $\{0.001, 0.0029, 0.0029, 0.0069, 99.99, 10000, 10000, 10000\}$. We observe that the values v_{r_i}, $i = 0{:}\, 5$ computed by means of the Newton polygon are good approximations to the magnitude of the eigenvalues of T. Table 3.1 summarizes the situation concerning the matrix $T = \mathrm{tridiag}(\beta, \alpha, \gamma)$: in column 1 we report the indices ℓ for which equation $q_\ell(x) = 0$ has solution, column 2 reports the open intervals (s_{ℓ_i}, t_{ℓ_i}), $i = 0{:}\, 3$ which, according to Theorem 3.1 do not contain the moduli of the eigenvalues λ_i, $i = 1{:}\, 8$. The remaining columns contain the moduli of the eigenvalues $|\lambda_i|$, $i = 1{:}\, 8$, the values v_{r_i}, $i = 0{:}\, 6$, of the radii of the circles where initial approximations are chosen, together with the number of these approximations in parentheses, and the values of the indices r_i, $i = 0{:}\, 6$ of the vertices of the Newton polygon.

3.2. Divide and conquer strategy. Another way of getting initial approximations for the Ehrlich-Aberth iteration is from the eigenvalues of two suitable tridiagonal matrices of order roughly $n/2$. Rewrite T as

$$T = \widetilde{T} + uv^T, \tag{3.2}$$

where

$$\widetilde{T} = \begin{bmatrix} T_1 & 0 \\ 0 & T_2 \end{bmatrix} = T_1 \oplus T_2, \quad u = e_k + e_{k+1}, \quad v = \beta_k e_k + \gamma_k e_{k+1} \tag{3.3}$$

and

$$T_1 = \begin{bmatrix} \alpha_1 & \gamma_1 & & 0 \\ \beta_1 & \alpha_2 & \ddots & \\ & \ddots & \ddots & \gamma_{k-1} \\ 0 & & \beta_{k-1} & \alpha_k - \beta_k \end{bmatrix}, \qquad T_2 = \begin{bmatrix} \alpha_{k+1} - \gamma_k & \gamma_{k+1} & & 0 \\ \beta_{k+1} & \alpha_{k+2} & \ddots & \\ & \ddots & \ddots & \gamma_{n-1} \\ 0 & & \beta_{n-1} & \alpha_n \end{bmatrix}.$$

There are no obvious connections between the eigenvalues of T and those of $\widetilde{T}$, unlike in the symmetric case, in which the eigenvalues of T interlace those of $\widetilde{T}$. In this section, we show that the eigenvalues of $\widetilde{T}$ generally provide good starting points for the Ehrlich-Aberth iteration for $p(\lambda) = \det(T - \lambda I)$.

We also remark that if the eigenvalue problem comes from the discretisation of a continuous problem (e.g., some partial differential equation), the submatrices T_1 and T_2 can be viewed as the matrices obtained from a discretization with a coarser grid of the same (or of a similar) continuous problem and their eigenvalues should be good approximations of the eigenvalues of T.

Stronger properties can be proved if T_1 and T_2 have a common eigenvalue.

Theorem 3.3. *If λ is an eigenvalue of T_1 and T_2 then λ is an eigenvalue of $T = (T_1 \oplus T_2) + uv^T$ for any vectors u and v.*

Proof. Let $T_1 x_1 = \lambda x_1$ and $T_2 x_2 = \lambda x_2$ with nonzero x_1 and x_2, so that $x = (\nu_1 x_1^T, \nu_2 x_2^T)^T$ is eigenvector of $\widetilde{T} = T_1 \oplus T_2$ for any ν_1 and ν_2. Then, $Tx = (\widetilde{T} + uv^T)x = \lambda x + (\nu_1 v_1^T x_1 + \nu_2 v_2^T x_2)u$, where $v = [v_1^T, v_2^T]^T$ has been partitioned conformably with x. The scalars ν_1 and ν_2 can be chosen so that $\nu_1 v_1^T x_1 + \nu_2 v_2^T x_2 = 0$, making x an eigenvector of T corresponding to the eigenvalue λ. □

By following a continuity argument, we may deduce that if λ_1 and λ_2 are eigenvalues of T_1 and T_2, respectively, such that $|\lambda_1 - \lambda_2|$ is small, then $T = (T_1 \oplus T_2) + uv^T$ has an eigenvalue close to λ_1 and λ_2.

Our next theorem relies on the following lemma from Henrici [16].

Lemma 3.4. *Let $p(\lambda)$ be a polynomial of degree n in λ, and let z be any complex number. Then the disk of center z and radius $n|p(z)/p'(z)|$ contains at least one zero of $p(\lambda)$.*

Theorem 3.5. *Assume that $T_1 \in \mathbb{R}^{k\times k}$ and $T_2 \in \mathbb{R}^{(n-k)\times(n-k)}$ are both diagonalizable, that is, there exist X_1, X_2 nonsingular such that $T_1 = X_1 D_1 X_1^{-1}$ and $T_2 = X_2 D_2 X_2^{-1}$, with $D_1 = \mathrm{diag}(d_1, d_2, \ldots, d_k)$ and $D_2 = \mathrm{diag}(d_{k+1}, d_{k+2}, \ldots, d_n)$. Let $\beta_k, \gamma_k \in \mathbb{R}$ and*

$$\eta_i = \begin{cases} \beta_k (e_i^T X_1^{-1} e_k)(e_k^T X_1 e_i) & \text{if } i \le k, \\ \gamma_k (e_{i-k}^T X_2^{-1} e_1)(e_1^T X_2 e_{i-k}) & \text{if } i > k. \end{cases} \tag{3.4}$$

Then in any disk of center d_i and radius

$$\rho_i = \frac{n|\eta_i|}{\left|1 + \sum_{\substack{j=1 \\ j\neq i}}^{n} \frac{\eta_i + \eta_j}{d_j - d_i}\right|}$$

there exists an eigenvalue of $T = T_1 \oplus T_2 + uv^T$*, where* $u = e_k + e_{k+1}$ *and* $v = \beta_k e_k + \gamma_k e_{k+1}$.

Proof. Let $\widetilde{T} = T_1 \oplus T_2$. We have

$$T - \lambda I = (\widetilde{T} - \lambda I)(I + (\widetilde{T} - \lambda I)^{-1} uv^T),$$

and taking determinants on both sides of the equation gives

$$p(\lambda) = \prod_{i=1}^{n} (d_i - \lambda)(1 + v^T (\widetilde{T} - \lambda I)^{-1} u).$$

Using the eigendecomposition of T_1 and T_2 and the definition of u and v, the expression for $p(\lambda)$ simplifies to

$$p(\lambda) = \prod_{i=1}^{n} (d_i - \lambda) \left(1 + \sum_{i=1}^{n} \frac{\eta_i}{d_i - \lambda}\right). \tag{3.5}$$

Thus

$$p(d_i) = \eta_k \prod_{\substack{j=1 \\ j \neq i}} (d_j - d_i),$$

and

$$p'(\lambda) = -\sum_{j=1}^{n} \prod_{\substack{k=1 \\ k \neq j}} (d_k - \lambda) - \sum_{\ell=1}^{n} \eta_\ell \Big(\sum_{j=1}^{n-1} \prod_{\substack{k=1 \\ k \neq \ell, j+1}} (d_k - \lambda) \Big)$$

so that

$$p'(d_i) = -\prod_{\substack{j=1 \\ j \neq i}} (d_j - d_i) \Big(1 + \sum_{\substack{j=1 \\ j \neq i}}^{n} \frac{\eta_i + \eta_j}{d_j - d_i} \Big).$$

Applying Lemma 3.4 completes the proof. □

According to Theorem 3.5, a small value for ρ_i indicates that there is an eigenvalue of diag(T_1, T_2) close to an eigenvalue of T. An important question related to the effectiveness of using the eigenvalues of T_1 and T_2 as initial approximations for starting the Ehrlich-Aberth iteration is whether the number of "large" values for ρ_i is small or not. Note that if $i \leq k$, η_i is a multiple of the product of the last components of the ith right and left eigenvector of T_1 and, if $i > k$, η_i is a multiple of the product of the first components of the $(i - k)$-th left and right eigenvectors of T_2. If T is unreduced, then η_i is nonzero since the eigenvectors of unreduced tridiagonal matrices cannot have a 0 in the first and last component. We report in Table 3.2 ranges of values for ρ_i and $|\eta_i|$ obtained from 1000 randomly generated tridiagonal matrices T of size $n = 100$. The table shows that in more than 80% of the cases, $|\eta_i|$ and ρ_i are smaller than 10^{-4}. Note that the denominator $\left|1 + \sum_{j=1, j \neq i}^{n} \frac{\eta_i + \eta_j}{d_j - d_i}\right|$ in the definition of ρ_i does not seem to play an important role. The probability that for almost all the values of i this denominator is close to zero seems to be small. These experiments suggest that most eigenvalues of $T_1 \oplus T_2$ should be good initial values for the Ehrlich-Aberth iteration for $p(\lambda) = \det(T - \lambda I)$.

TABLE 3.2

Ranges of values for ρ_i and $|\eta_i|$ obtained from 1000 randomly generated tridiagonal matrices T of size $n = 100$.

%	$\leq 10^{-16}$	$\leq 10^{-12}$	$\leq 10^{-8}$	$\leq 10^{-4}$
ρ_i	48	60	71	82
$\|\eta_i\|$	54	65	77	88

From the results of this section we propose the following divide and conquer strategy to compute initial approximations for the Ehrlich-Aberth iterations. The matrix T is recursively split according to the rank-one tearing in (3.2)–(3.3) until 2×2 or 1×1 subproblems are reached. The Ehrlich-Aberth iteration is then used to glue back the subproblems using the previously computed eigenvalues as starting guesses for the iterations.

REMARK 3.6. A similar divide and conquer strategy can be obtained by choosing, as initial approximations, the eigenvalues of the leading principal $m \times m$ submatrix T_1 of T and of the trailing principal $(n-m) \times (n-m)$ submatrix T_2 of T for $m = \lceil n/2 \rceil$. These matrices are obtained by zeroing the entries in position $(m, m+1)$ and $(m+1, m)$ of T which correspond to applying a rank-2 correction to the matrix T. An analysis similar to the one performed for the rank-1 tearing can be carried out.

4. The algorithm.

In this section we describe an implementation of the Ehrlich-Aberth iteration where the choice of the initial approximations is performed by means of a divide-and-conquer strategy. Then we provide running error bounds needed for the validation of the computed approximations and discuss the computation of the eigenvectors.

The main algorithm for eigenvalue approximation is described below in pseudocode. This implementation follows section 3.2. A different implementation can be based on the rank-2 tearing of Remark 3.6 where the initial approximations are the eigenvalues of the principal submatrices obtained by zeroing the entries in position $(m, m+1)$ and $(m+1, m)$, where $m = \lfloor n/2 \rfloor$.

First we report the recursive part of the algorithm and then the main refinement engine, i.e., Ehrlich-Aberth's iteration.

```
function $z$ =eigen$(\beta, \alpha, \gamma)$
% Computes the eigenvalues of the $n \times n$ tridiagonal matrix $T = \mathrm{tridiag}(\beta, \alpha, \gamma)$
% $perturb$ is a small scalar set to the maximum relative perturbation
% of intermediate eigenvalues.
if $n = 1$, $z = \alpha_1$, return, end
if $n = 2$, set $z$ to the eigenvalues of $\begin{bmatrix} \alpha_1 & \gamma_1 \\ \beta_1 & \alpha_2 \end{bmatrix}$, return, end
% Recursive stage
if $n > 2$
    $m = \lfloor n/2 \rfloor$
    $\alpha' = \alpha(1:m)$, $\alpha'_m = \alpha'_m - \beta_m$
    $z(1{:}m)$ =eigen$\big(\beta(1:m-1), \alpha', \gamma(1:m-1)\big)$
    $\alpha'' = \alpha(m+1:n)$, $\alpha''_1 = \alpha''_1 - \gamma_m$
    $z(m+1:n)$ =eigen$\big(\beta(m+1:n-1), \alpha'', \gamma(m+1:n-1)\big)$
    Choose random $\rho$ such that $perturb/2 < \rho < perturb$
    $z(1:m) = (1 + i\rho) * z(1:m)$
    $z(m+1:n) = (1 - i\rho) * z(m+1:n)$
```

```
        % Refine the current approximations
        $z$=Aberth($\beta, \alpha, \gamma, z$)
    end
```

The role of *perturb* is to avoid different approximations collapsing to a single value. In fact, if $z_i = z_j$ with $i \neq j$ then the Ehrlich-Aberth iteration cannot be applied. Moreover, since in practice the Ehrlich-Aberth iteration performs better if the intermediate approximations are in the complex field, we have chosen a pure imaginary value as perturbation. The pseudocode for the Ehrlich-Aberth iteration is reported below.

```
    function $z$ = Aberth($\beta, \alpha, \gamma, z$)
    % Refine the $n$ approximations $z = (z_i)$, $i = 1 : n$ to the eigenvalues of
    % $T$ = tridiag($\beta, \alpha, \gamma$) by means of the Ehrlich-Aberth iteration.
    % $maxit$ is the maximum number of iterations,
    % $tol$ is a small quantity used for the convergence criteria.
    $it = 0$
    $\zeta = zeros(n, 1)$ % $\zeta_i = 1$ if $z_i$ has converged to an eigenvalue and 0 otherwise.
    while $(\sum_{i=1}^{n} \zeta_i < n$ & $it < maxit)$
        $it = it + 1$
        for $i = 1 : n$
            if $\zeta_i = 0$
                $\tau = trace\_Tinv(\beta, \alpha - z_i, \gamma)$
                if $\tau$ =Inf
                    $nwtc = 0$
                else
                    $nwtc = -1/\tau$
                end
                if $|nwtc| < tol\|T - z_i I\|_\infty$, $\zeta_i = 1$, end
                $z_i = z_i - nwtc/(1 - nwtc \sum_{j=1, j\neq i}^{n} \frac{1}{z_i - z_j})$
            end
        end
    end
```

Observe that at the general *it*-th sweep, the Ehrlich-Aberth correction is applied only at the approximations z_i which have not yet converged. This makes the cost of each sweep depend on the number of approximated eigenvalues. For this reason, in order to evaluate the complexity of the algorithm, it is convenient to introduce the number μ of *average iterations per eigenvalue* given by the overall number of Ehrlich-Aberth iterations applied to each eigenvalue divided by n. For example, if $n = 4$ and the number of iterations needed for computing $\lambda_1, \lambda_2, \lambda_3, \lambda_4$ is 5,10,14,21, respectively, then $\mu = 50/4 = 12.5$.

4.1. Running error bound. In this section we derive a running error bound for the error in the computed eigenvalues λ_ℓ, $\ell = 1\colon n$. Our bounds are based on the following result of Carstensen [9].

LEMMA 4.1. *Let $p(\lambda)$ be a monic polynomial of degree n in λ, and let $\lambda_1, \lambda_2, \ldots, \lambda_n$ be pairwise distinct complex numbers. Denote by $\mathcal{D}(\lambda_\ell, \rho_\ell)$ the disk of center λ_ℓ and* *radius*

$$\rho_\ell = \frac{n|p(\lambda_\ell)|}{\left|\prod_{\substack{j=1\\ j\neq \ell}}^n (\lambda_\ell - \lambda_j)\right|}.$$

Then $\mathcal{U} = \bigcup_\ell \mathcal{D}(\lambda_\ell, \rho_\ell)$ contains all the zeros of $p(\lambda)$. Moreover, any connected component of $\mathcal{U}$ made up by k disks contains k zeros. In particular, any isolated disk contains a single zero.

The set of disks $\{\mathcal{D}(\lambda_\ell, \rho_\ell),\ \ell = 1{:}\,n\}$ defined in the above lemma is called a *set of inclusion disks.* Note that, because of rounding errors in the computation of $p(\lambda_\ell)$ and $\prod_{j=1,\ j\neq\ell}^n (\lambda_\ell - \lambda_j)$, the computed ρ_ℓ denoted by $\widehat{\rho}_\ell$, may be inaccurate. Thus, the disks $\mathcal{D}(\lambda_\ell, \widehat{\rho}_\ell)$ may not provide a set of inclusion disks.

Consider the standard model of floating point arithmetic [18, Section 2.2]

$$fl(x \operatorname{op} y) = (x \operatorname{op} y)(1+\delta), \qquad |\delta| \le u, \quad \operatorname{op} = +, -, *, /, \tag{4.1}$$

where u is the unit roundoff. Since λ can be complex, part of the computation is carried out in complex arithmetic. Under the standard model (4.1) (see [18, Lemma 3.5]),

$$\begin{aligned} fl(x \pm y) &= (x \pm y)(1+\delta), \quad |\delta| \le u,\\ fl(xy) &= xy(1+\delta), \quad |\delta| \le 2\sqrt{2}u,\\ fl(x/y) &= (x/y)(1+\delta), \quad |\delta| \le 4\sqrt{2}u, \end{aligned} \tag{4.2}$$

where we ignore second order terms in u.

Suppose we can compute an upper bound Δp_ℓ for the error $|\hat{p}_\ell - p(\lambda_\ell)|$, where $\hat{p}_\ell = fl(p(\lambda_\ell))$. Then from Lemma 4.1 we immediately deduce that the disk $\mathcal{D}(\lambda_\ell, \widetilde{\rho}_\ell)$, where

$$\widetilde{\rho}_\ell = (1+2nu)\frac{n(|\hat{p}_\ell| + \Delta p_\ell)}{\hat{\pi}_\ell}, \qquad \hat{\pi}_\ell = fl\Big(\Big|\prod_{\substack{j=1\\ j\neq\ell}}^n (\lambda_\ell - \lambda_j)\Big|\Big),$$

is such that $\mathcal{D}(\lambda_\ell, \rho_\ell) \subset \mathcal{D}_\ell(\lambda_\ell, \widetilde{\rho}_\ell)$, $\ell = 1:n$, therefore the set $\{\mathcal{D}(\lambda_\ell, \widetilde{\rho}_\ell), \quad \ell = 1:n\}$ is a set of inclusion disks.

The main difficulty is in determining Δp_ℓ. Recall that $p(\lambda_\ell) = \prod_{i=1}^n r_i$, where r_i is the ith diagonal element of R in the QR factorization of $T - \lambda_\ell I$. Let $fl(r_i) = r_i + \delta r_i$. Then

$$\widehat{p}_\ell = fl\left(\prod_{i=1}^n r_i\right) = \prod_{i=1}^n (r_i + \delta r_i) \cdot \prod_{i=1}^{n-1}(1+\epsilon_i), \quad |\epsilon_i| \le 2\sqrt{2}u$$

and, if we ignore the second order terms in u, we have $\widehat{p}_\ell = p(\lambda_\ell) + \delta p_\ell$, with

$$|\delta p_\ell| \le \widehat{p}_\ell \left(\sum_{i=1}^n \Delta r_i / \widehat{r}_i + (n-1)2\sqrt{2}u\right) =: \Delta p_\ell. \tag{4.3}$$

The Δr_i are computed along with the $\widehat{r}_i = fl(r_i)$ thanks to a systematic running error analysis of all the quantities involved in the calculation of r_i. For that we make use of the following lemma.

LEMMA 4.2. *Let $x = yz + \alpha v \in \mathbb{C}$, where α is given data, $y = \widehat{y} + \delta y$ with $|\delta y| \le \Delta y$, $z = \widehat{z} + \delta z$ with $|\delta z| \le \Delta z$, $v = \widehat{v} + \delta v$ with $|\delta v| \le \Delta v$, and $\widehat{y}, \widehat{z}, \widehat{v}, \Delta y, \Delta z, \Delta v$ are known computed quantities. Then $\widehat{x} = fl(x) = x + \delta x$ with*

$$|\delta x| \le \Delta x := |\widehat{y}|\Delta z + |\widehat{z}|\Delta y + 2\sqrt{2}u|\widehat{y}|\,|\widehat{z}| + |\alpha|\Delta v + 2\sqrt{2}u|\alpha|\,|\widehat{v}|.$$

Proof. The proof is a straightforward application of (4.1) and (4.2). □

We notice that in the function *trace_Tinv* (see section 2.2) all the quantities used to compute r_i can be rewritten in the form $x = yz + \alpha v$. Hence, assuming that the function *Givens* returns $\Delta\phi$ and $\Delta\psi$, we can add the following lines

$$\begin{aligned}
\Delta r_i &= |\phi|\Delta a + |a|\Delta\phi + 2\sqrt{2}u|a|\,|\phi| + |\beta_i|\Delta\psi + 2\sqrt{2}u|\beta_i|\,|\psi|,\\
\Delta a &= |\psi|\Delta g + |g|\Delta\psi + 2\sqrt{2}u|\psi|\,|g| + |\alpha_{i+1}|\Delta\phi + 2\sqrt{2}u|\alpha_{i+1}|\,|\phi|,\\
\Delta g &= |\gamma_{i+1}|\Delta\phi + 2\sqrt{2}u|\gamma_{i+1}|\,|\phi|
\end{aligned}$$

to the function *trace_Tinv* after the computation of r_i, a and g, respectively. The two quantities Δa and Δg are initially set to zero and $\Delta r_n = \Delta a$. The error bound Δp_ℓ is then obtained using (4.3).

4.2. Computing the eigenvectors. One of the most convenient methods for approximating an eigenvector of T once we are given an approximation λ of the corresponding eigenvalue is the inverse power iteration applied to the matrix $T - \lambda I$. A crucial computational issue is to determine a suitable initial guess $v^{(0)}$ for the eigenvector in order to start the iteration:

$$\begin{array}{ll}
(T-\lambda I)w^{(i+1)} = v^{(i)} & \\
v^{(i+1)} = w^{(i+1)}/\|w^{(i+1)}\| & \qquad i = 0, 1, 2, \dots
\end{array}$$

This problem has been studied in several recent papers [10], [13], [25]. In particular, in [13] a strategy is described for the choice of $v^{(0)}$ which relies on the evaluation of the index k of the entry of maximum modulus in the main diagonal of $(T - \lambda I)^{-1}$. Our algorithm for the approximation of the eigenvalues provides, as a byproduct, the diagonal entries of $(T - \lambda I)^{-1}$. Therefore the value of k is determined at no cost. Moreover, the QR factorization of the matrix $T - \lambda I$ that is computed by our algorithm can be used for performing each inverse power iteration without any significant additional cost.

5. Numerical experiments. We have implemented the algorithm for the approximation of the eigenvalues of the tridiagonal matrix T in Fortran 95. The code, organized as a Fortran 95 module, can be downloaded from `http://www.dm.unipi.it/~bini/software`. The tests have been performed on an Athlon 1800 with IEEE double precision arithmetic.

In what follows `eigen` refers to our subroutine implementing the Ehrlich-Aberth iteration. In order to evaluate the performance of `eigen`, beside the cpu time, we report the number μ of average iterations per eigenvalues required in the last recursive step.

We considered two sets of tests: tests with matrices of relatively small size and tests with matrices of large size. The latter test suite is used for testing the asymptotic cost of `eigen` and its reliability; the former test suite is useful for checking the numerical quality of the computed approximations with `eigen` compared with the LAPACK subroutine `dhseqr` that implements the QR algorithm for computing the eigenvalues of an upper Hessenberg matrix.

TABLE 5.1
CPU time in seconds for `eigen` *(in boldface) versus LAPACK's* `dhseqr`.

n	100	200	400	800	1600	3200	6400
Test 1	**0.04**/0.01	**0.1**/0.06	**0.4**/0.6	**1.4**/4.8	**5.8**/76	**25**/661	**112**/6209
Test 2	**0.04**/0.02	**0.1**/0.10	**0.4**/0.8	**1.5**/7.5	**5.7**/56	**17**/315	**68**/2378
Test 3	**0.04**/0.02	**0.1**/0.07	**0.4**/0.7	**1.5**/6.7	**5.7**/152	**22**/1510	**78**/15188
Test 4	**0.25**/0.02	**1.1**/0.07	**4.2**/0.6	**16.6**/4.3	**67.6**/75.6	**294**/896	**1293**/6092
Test 5	**0.06**/0.02	**0.1**/0.10	**0.5**/0.9	**1.5**/6.8	**5.8**/135	**20**/1239	**83**/11088
Test 6	**0.26**/0.02	**1.02**/0.08	**4.0**/0.6	**15.8**/5.4	**63**/93	**283**/744	**757**/7351
Test 7	**0.07**/0.02	**0.3**/0.10	**0.9**/0.8	**3.2**/6.7	**12.0**/124	**66**/1460	**204**/9155
Test 8	**0.02**/0.02	**0.7**/0.08	**0.2**/0.6	**0.8**/4.0	**3.2**/49.9	**15**/382	**63**/2838
Test 9	**0.13**/0.02	**0.5**/0.11	**1.9**/0.9	**7.9**/8.2	**32**/175	**139**/1450	**562**/13250
Test 10	**0.06**/0.02	**0.2**/0.08	**0.6**/0.7	**2.1**/6.0	**7.7**/130	**32**/975	**137**/8990

5.1. Large matrices. We have considered the following test problems which we describe in their normalized form $\det(T-\lambda D)=0$ where matrices are scaled so that $t_{i,i+1}=t_{i+1,i}=1$, $i=1,:n-1$. Here α represents the vector of the diagonal entries of T and δ the vector of the diagonal entries of D.
Test 1: $\alpha_i = i*(-1)^{\lfloor i/8\rfloor}$, $\delta_i=(-1)^i/i$, $i=1:n$
Test 2: $\alpha_i = 10*(-1)^{\lfloor i/8\rfloor}$, $\delta_i = i*(-1)^{\lfloor i/9\rfloor}$, $i=1:n$
Test 3: $\alpha_i=i$, $\delta_i=n-i+1$, $i=1:n$
Test 4: $\alpha_i=(-1)^i$, $\delta_i=20*(-1)^{\lfloor i/5\rfloor}$, $i=1:n$
Test 5: $\alpha_i=10^{5(-1)^i}*(-1)^{\lfloor i/4\rfloor}$, $\delta_i=(-1)^{\lfloor i/3\rfloor}$, $i=1:n$
Test 6: $\alpha_i=2$, $\delta_i=1$, $i=1:n$
Test 7: $\alpha_i=\frac{1}{i}+\frac{1}{n-i+1}$, $\delta_i=\frac{1}{i}(-1)^{\lfloor i/9\rfloor}$, $i=1:n$
Test 8: $\alpha_i=i*(-1)^{\lfloor i/13\rfloor+\lfloor i/5\rfloor}$, $\delta_i=(n-i+1)^2*(-1)^{\lfloor i/11\rfloor}$, $i=1:n$
Test 9: $\alpha_i=1$, $i=1:n$; $\delta_i=1$, for $i<n/2$, $\delta_i=-1$ if $i\ge n/2$.
Test 10: α_i and δ_i take random values uniformly distributed in $[-0.5,0.5]$.
The spectrum of the test matrices is plotted in Figures 5.1 and 5.2 for $n=600$.

The tests have been performed with $n\in\{100,200,400,800,1600,3200,6400\}$ on a PC with an Athlon 1800 cpu. Table 5.1 reports the cpu time in seconds required by the `eigen` versus the time required by the LAPACK subroutine `dhseqr`. These timings show that in all the cases the growth of the cpu time required by our algorithm is a quadratic function of n, whereas the cost of `dhseqr` grows cubically with n. For certain tests (1,2,3,5,7,8,10) the algorithm is very fast: μ is between 2 and 10. For tests 4,6,9 the algorithm requires more iterations: μ is in the range 17–42. The threshold value for which our algorithm is faster than the LAPACK subroutine is about $n=400$ for tests 1,2,3,5,8,10, $n=800$ for tests 7,9, and $n=1600$ for tests 4 and 6. The speed-up reached for $n=6400$ is in the range 4.7–194.7.

The average number μ of Ehrlich-Aberth iterations per eigenvalue, reported in Table 5.2, seems to be almost independent of n and it varies according to the specific problem.

5.2. Small matrices. We tested Liu's 14×14 matrix [20], which in normalized form is defined by $(\alpha_1,\dots,\alpha_{14})=(0,0,0,0,0,0,-1,-1,0,0,0,0,0,0)$ and $(\delta_1,\dots,\delta_{14})=(-1,1,1,1,-1,-1,1,-1,1,1,-1,-1,-1,1)$. This problem has only one zero eigenvalue of multiplicity 14. As shown in Figure 5.3, the accuracy of the approximations delivered by `eigen` are not worse than those provided by `dhseqr`.

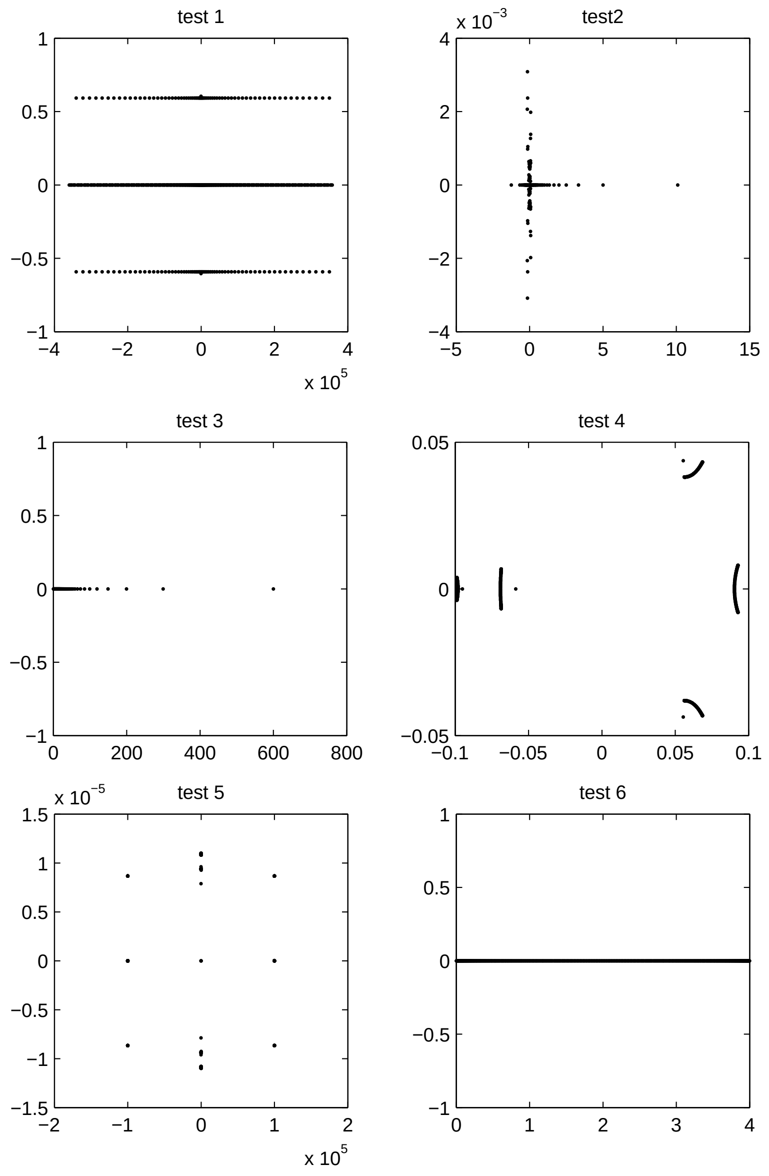

FIG. 5.1. *Eigenvalues of the test matrices 1–6 for $n = 600$.*

We have also considered the matrix in test 5 for $n = 20$, which has clustered eigenvalues in groups with large moduli and in groups with small moduli. More precisely, there are 4 eigenvalues close to 10^5, 6 eigenvalues close to 10^{-5} and 10 eigenvalues of modulus less than 10^{-4}. In Figure 5.4 we show the relative errors of the approximations computed by the algorithm `dhseqr` and by our algorithm. The approximations have been compared with the values obtained by performing the computation with 50 decimal digits. Eigenvalues have been ordered with nondecreasing real parts. The largest relative errors of the LAPACK routine are obtained with the eigenvalues of smallest modulus. The Ehrlich-Aberth iteration provides better approximations.

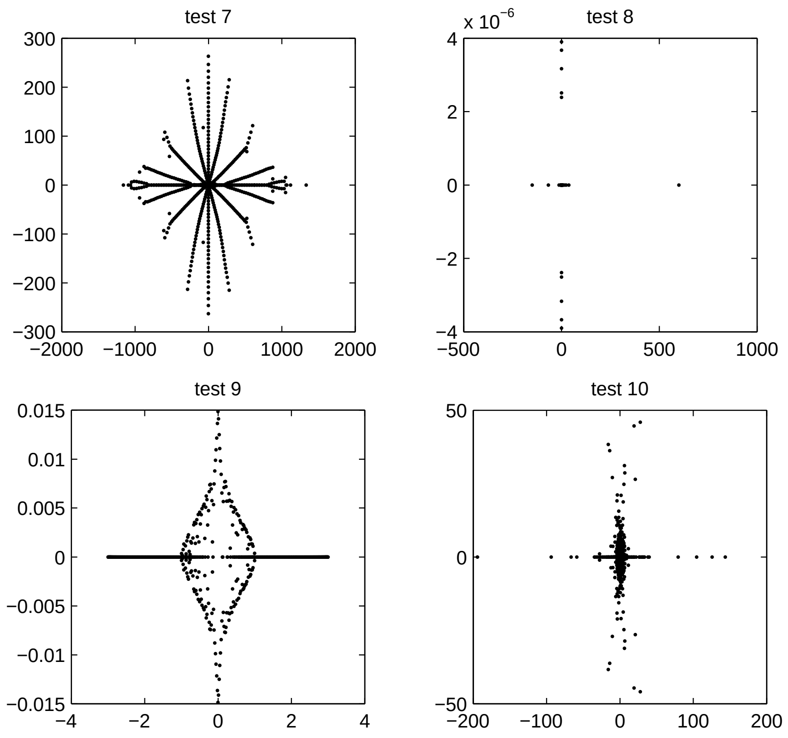

FIG. 5.2. *Eigenvalues of the test matrices 7–10 for $n = 600$.*

TABLE 5.2
Average number of iterations per eigenvalue and maximum number of iterations (in parentheses) of the Ehrlich-Aberth method.

n	100	200	400	800	1600	3200	6400
Test 1	1.8 (3)	1.8 (3)	1.8 (19)	1.8 (18)	1.8 (18)	1.8 (18)	1.8 (18)
Test 2	2.0 (4)	1.9 (4)	2.0 (14)	1.9 (4)	1.7 (3)	1.0 (6)	1.0 (6)
Test 3	2.1 (4)	2.0 (4)	1.9 (4)	1.8 (4)	1.7 (4)	1.5 (4)	1.0 (4)
Test 4	22.2 (26)	21.7 (31)	21.2 (29)	21.1 (39)	20.4 (62)	19.9 (86)	19.4 (156)
Test 5	3.1 (11)	1.5 (10)	2.4 (12)	1.2 (10)	1.6 (24)	1.1 (6)	1.1 (12)
Test 6	22.1 (26)	21.6 (26)	21.2 (27)	20.5 (28)	20.1 (29)	19.5 (29)	18.9 (31)
Test 7	4.5 (9)	4.5 (13)	3.8 (13)	3.4 (11)	3.3 (14)	3.0 (14)	2.4 (16)
Test 8	1.1 (3)	1.0 (3)	1.0 (3)	1.0 (3)	1.0 (3)	1.0 (3)	1.0 (1)
Test 9	6.6 (31)	5.9 (13)	5.8 (30)	5.6 (15)	5.9 (43)	5.7 (16)	5.7 (27)
Test 10	3.4 (10)	2.7 (7)	2.4 (7)	2.3 (8)	2.1 (12)	2.1 (9)	2.0 (9)

5.3. Initial approximations. We have compared the divide-and-conquer strategy (D&C) with the sequence generated by the Ehrlich-Aberth iteration starting from the perturbed n-th roots of 1 and starting by selecting as initial approximations the values provided by the criterion based on Rouché's theorem. The average number of iterations per eigenvalue needed by the three techniques are reported in Table 5.3 for $n = 50$ and show that the divide-and-conquer strategy provides the best performance with a very low number of average iterations per eigenvalue.

The choice of the initial approximations provided by Rouché's theorem improves the performance by a factor of up to 5.6 with respect to the customary choice of the perturbed n-th roots of 1.

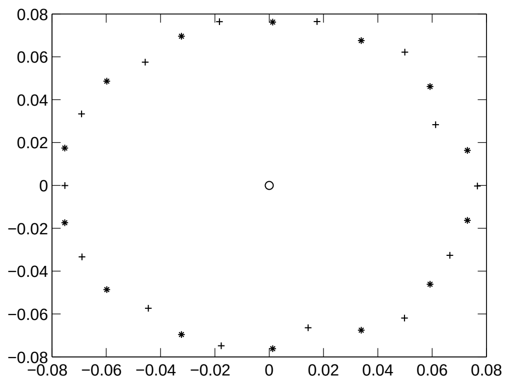

FIG. 5.3. *Liu's matrix: approximations provided by* `eigen` *(+) and by the Lapack subroutine* `dhseqr` *(∗).*

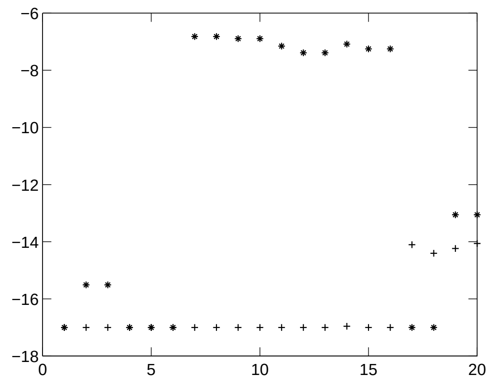

FIG. 5.4. *Matrix of test 5 with $n = 20$: log scale relative errors provided by* `eigen` *(+) and by the Lapack subroutine* `dhseqr` *(∗).*

From the numerical experiments we have also verified that the evaluation of the coefficients of the characteristic polynomial by means of the three-term recurrence is prone to overflow problems but it is numerically more stable than the approach based on FFT. In fact, with the latter computation we can provide a uniform upper bound to the *absolute* error whereas the *relative* errors in the coefficients with the lowest modulus may reach values even greater than one. This fact may limit the efficiency of the Rouché-based criterion if the coefficients of the polynomial are very unbalanced.

5.4. Error bounds. We report the results of the tests concerning the computation of the *a posteriori* error bounds $\widetilde{\rho}_\ell$, $\ell = 1 : n$, obtained by applying Lemma 4.1 and equation (4.1) with the running error analysis of section 4.1. We have considered the following two instances of the problem $\det(T - \lambda D) = 0$ where $t_{i+1,i} = t_{i,i+1} = 1$, $i = 1 : n-1$, and δ_i, $i = 1 : n$ denote the diagonal elements of D.

case 1: $n = 10$ $t_{i,i} = i$, $\delta_i = (-1)^{i+1}$, $i = 1 : n$;

case 2: $n = 10$, $t_{i,i} = 10^{6\cdot(-1)^{i+1}}$, $\delta_i = 1$, $i = 1 : n/2$, $\delta_i = -1$, $i = n/2 + 1, n$.

The first problem has well conditioned eigenvalues that are computed with full precision: all the digits of their computed approximations are correct. The second problem has eigenvalues in clusters, some of which are ill conditioned. Therefore some

TABLE 5.3
Average number of iterations of the Ehrlich-Aberth method with initial approximations selected in three different ways for matrices of size $n = 50$.

Test	Rouché	roots of 1	D&C
1	50.4	110.7	4.5
2	13.7	23.1	2.1
3	23.7	31.8	2.1
4	18.5	65.5	23.2
5	62.6	178.5	3.2
6	31.0	32.6	22.3
7	56.9	57.3	5.8
8	8.3	45.7	1.7
9	11.9	17.3	5.3
10	32.3	32.2	4.5

TABLE 5.4
Bounds on the relative errors: $n = 10$, $\alpha_i = i$, $\delta_i = (-1)^{i+1}$, $i = 1 : n$.

eigenvalues	rel. errors	upper bounds
1.312128484159043	1.01E-16	2.73E-16
-0.5176028578197819	7.39E-17	1.04E-15
-2.648431399492776	1.99E-17	3.61E-16
3.741351244815260	3.46E-17	1.77E-16
-4.79575908601544	2.14E-17	2.14E-16
5.830924274494304	7.11E-17	7.12E-16
-6.855643258908795	1.95E-17	1.66E-16
7.874047291369576	2.39E-17	2.05E-16
9.948577928314871	2.39E-17	3.89E-16
-8.889592620916256	2.29E-17	2.79E-16

approximation is less accurate than in the previous case.

We have computed the eigenvalues with double and with quartuple precision. In Tables 5.4, 5.5 we report the approximation to the eigenvalues $\widetilde{\lambda}$ provided by our algorithm in double precision; and the relative error bound $|\widetilde{\lambda} - \lambda|/|\lambda|$, where λ is the approximation to the eigenvalue computed with quartuple precision; the relative bound $\widetilde{\rho}/|\widetilde{\lambda}|$ computed by means of (4.1), Lemma 4.1 and the running error analysis. The inaccurate digits of the approximated eigenvalues are typed in boldface.

Tables 5.4 and 5.5 confirm that the computed upper bounds are greater than the actual corresponding relative errors, moreover, the bound is generally sharp. Only for the clustered eigenvalues having small moduli the upper bounds may differ much from the actual relative errors.

6. Conclusion. We have introduced an algorithm for the computation of the Newton quotient $p(\lambda)/p'(\lambda)$ for $p(\lambda) = \det(T - \lambda I)$, and T a tridiagonal matrix, based on the QR factorization of $T - \lambda I$ and on the semiseparable structure of $(T - \lambda I)^{-1}$. The algorithm, whose arithmetic cost is linear in the size n of the matrix T, is robust.

This algorithm has been used for implementing the Ehrlich-Aberth iteration, which approximates all the eigenvalues of T. Besides the straightforward way of choosing the initial approximations in the unit circle, two more elaborated strategies for the choice of the initial approximations have been proposed and compared. The most interesting one is based on a divide-and conquer technique, which, even though heuristic, is motivated by some inclusion results that we have proved in Section 3.2. Running error bounds for the errors in the computed eigenvalues are also provided.

The numerical experiments, performed with a large set of test matrices, have

TABLE 5.5
Bounds on the relative errors: $n = 10$, $\alpha_i = 10^{6\cdot(-1)^{i+1}}$, $\delta_i = 1$ *for* $i \le n/2$, $\delta_i = -1$ *for* $i > n/2$, $i = 1 : n$.

eigenvalues	rel. errors	upper bounds
2.26423363920225**2**E-07+i5.29165323433**2500**E-07	1.91E-16	3.57E-9
2.26423363920**3277**E-07-i5.291653234332**789**E-07	1.84E-13	3.57E-9
2.1377858**87708795**E-06	2.06E-8	2.07E-7
-1.810912991**456234**E-06	6.86E-11	7.18E-10
1000000.000000653	3.70E-17	3.70E-16
1000000.000002879	3.13E-17	3.13E-16
999999.9999994679	4.55E-17	4.55E-16
-1000000.000003000	2.28E-17	2.28E-16
-1000000.000001000	7.61E-18	7.61E-17
-7.7971**809885387**	1.89E-6	1.90E-5

confirmed the effectiveness of the algorithm. Comparisons with the Lapack subroutine `dhseqr` have shown that for moderately large values of n our algorithm is faster. In particular, the Ehrlich-Aberth iteration has a cost which is $O(n^2)$, whereas the Lapack subroutine has a cost $O(n^3)$.

REFERENCES

[1] O. ABERTH, *Iteration methods for finding all zeros of a polynomial simultaneously*, Math. Comp., 27 (1973), pp. 339–334.

[2] L. ADAMS AND P. ARBENZ, *Towards a divide and conquer algorithm for the real nonsymmetric eigenvalue problem*, SIAM J. Matrix Anal. Appl., 15 (1994), pp. 1333–1353.

[3] D. BINDEL, J. DEMMEL, W. KAHAN, AND O. MARQUES, On computing Givens rotations reliably and efficiently. *ACM Trans. Math. Software*, 28(2):206–238, 2002.

[4] D. A. BINI, *Numerical computation of polynomial zeros by means of Aberth's method*, Numerical Algorithms, 13 (1996), pp. 179–200.

[5] D. A. BINI AND G. FIORENTINO, *MPSsolve: Numerical computation of polynomial roots v. 2.0*, FRISCO report, 1999. Software and papers available from ftp://ftp.dm.unipi.it/pub/mpsolve.

[6] ———, *Design, analysis, and implementation of a multiprecision polynomial rootfinder*, Numer. Algorithms, 23 (2000), pp. 127–173.

[7] M. A. BREBNER AND J. GRAD, *Eigenvalues of $Ax = \lambda Bx$ for real symmetric matrices A and B computed by reduction to a pseudosymmetric form and the HR process*, Linear Algebra Appl., 43 (1982), pp. 99–118.

[8] A. BUNSE-GERSTNER, *An analysis of the HR algorithm for computing the eigenvalues of a matrix*, Linear Algebra Appl., 35 (1981), pp. 155–173.

[9] C. CARSTENSEN, *Inclusion of the roots of a polynomial based on Gerschgorin's theorem*, Numer. Math., 59 (1991), pp. 349–360.

[10] I. S. DHILLON, *Reliable computation of the condition number of a tridiagonal matrix in $O(n)$ time*, SIAM J. Matrix Anal. Appl., 19 (1998), pp. 776–796.

[11] J. J. DONGARRA, G. A. GEIST, AND C. H. ROMINE, *Algorithm 710: FORTRAN subroutines for computing the eigenvalues and eigenvectors of a general matrix by reduction to general tridiagonal form*, ACM Trans. Math. Software, 18 (1992), pp. 392–400.

[12] L. W. EHRLICH, *A modified Newton method for polynomials*, Comm. ACM, 10, 2 (1967), pp. 107–108.

[13] K. V. FERNANDO, *On computing an eigenvector of a tridiagonal matrix. I. Basic results*, SIAM J. Matrix Anal. Appl., 18 (1997), pp. 1013–1034.

[14] G. A. GEIST, *Reduction of a general matrix to tridiagonal form*, SIAM J. Matrix Anal. Appl., 12 (1991), pp. 362–373.

[15] GENE H. GOLUB AND C. F. VAN LOAN, *Matrix Computations*, Johns Hopkins University Press, Baltimore, MD, USA, third ed., 1996.

[16] P. HENRICI, *Applied and Computational Complex Analysis*, vol. 1, Wiley, New York, 1974.

[17] N. J. HIGHAM, *Efficient algorithms for computing the condition number of a tridiagonal matrix*, SIAM J. Sci. Statist. Comput., 7 (1986), pp. 150–165.

[18] ———, *Accuracy and Stability of Numerical Algorithms*, Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, second ed., 2002.

[19] E. R. Jessup, *A case against a divide and conquer approach to the nonsymmetric eigenvalue problem*, Appl. Numer. Math., 12 (1993), pp. 403–420.

[20] Z.-S. Liu, *On the extended HR algorithm*, Technical report PAM-564, Center for Pure and Applied Mathematics, University of California, Berkley, USA, 1992.

[21] G. Meurant, *A review on the inverse of symmetric tridiagonal and block tridiagonal matrices*, SIAM J. Matrix Anal. Appl., 13 (1992), pp. 707–728.

[22] A. Ostrowski, *On a theorem by J. L. Walsh concerning the moduli of roots of algebraic equations*, Bull. Am. Math. Soc., 47 (1941), pp. 742–746.

[23] B. N. Parlett, *The Symmetric Eigenvalue Problem*, Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 1998. Unabridged, amended version of book first published by Prentice-Hall in 1980.

[24] B. N. Parlett and H. C. Chen, *Use of indefinite pencils for computing damped natural modes*, Linear Algebra Appl., 140 (1990), pp. 53–88.

[25] B. N. Parlett and I. S. Dhillon, *Fernando's solution to Wilkinson's problem: an application of double factorization*, Linear Algebra Appl., 267 (1997), pp. 247–279.

[26] H. Rutishauser, *Solution of Eigenvalue Problems with the LR-Transformation*, vol. 49, Nat. Bur. Standards Appl. Math. Ser., 1958, pp. 47–81.

[27] F. Tisseur, *Tridiagonal-diagonal reduction of symmetric indefinite pairs*, Numerical Analysis Report No. 409, Manchester Centre for Computational Mathematics, Manchester, England, 2002.

[28] F. Tisseur and K. Meerbergen, *The quadratic eigenvalue problem*, SIAM Rev., 43 (2001), pp. 235–286.