SIAM J. MATRIX ANAL. APPL.
Vol. 32, No. 4, pp. 1524–1536

RELIABLE EIGENVALUES OF SYMMETRIC TRIDIAGONALS[*]

RUI RALHA[†]

Abstract. For the eigenvalues of a symmetric tridiagonal matrix T, the most accurate algorithms deliver approximations which are the exact eigenvalues of a matrix $\widetilde{T}$ whose entries differ from the corresponding entries of T by small relative perturbations. However, for matrices with eigenvalues of different magnitudes, the number of correct digits in the computed approximations for eigenvalues of size smaller than $\|T\|_2$ depends on how well such eigenvalues are defined by the data. Some classes of matrices are known to define their eigenvalues to high relative accuracy but, in general, there is no simple way to estimate well the number of correct digits in the approximations. To remedy this, we propose a method that provides sharp bounds for the eigenvalues of T. We present some numerical examples to illustrate the usefulness of our method.

Key words. symmetric tridiagonals, bisection method, bounds for eigenvalues

AMS subject classifications. 65F15, 65G30

DOI. 10.1137/100817413

1. Introduction.

There are fast and reliable methods for computing the eigenvalues (and eigenvectors) of a symmetric tridiagonal matrix T which are implemented in LAPACK [1]. The routine DSTERF uses the Pal–Walker–Kahan variant (square-root free) of the QR algorithm for computing eigenvalues only [16, p. 164], DSTEQR and DSTEDC use the implicitly shifted QR algorithm [13, p. 421], and the divide and conquer algorithm [4], [14], respectively, to compute eigenvalues and also eigenvectors. The routine DSTEMR uses bisection and the dqds algorithm [12], [17] to compute selected eigenvalues; numerically orthogonal eigenvectors (optional) are computed with the use of various suitable LDL^T factorizations near clusters of close eigenvalues (referred to as MRRR, multiple relatively robust representations [10], [11]). New ideas which may lead to improved MRRR codes have been presented in [20]. Finally, there is an implementation of simple bisection (routine DSTEBZ) and we have observed in our experiments that this is the only routine in LAPACK that consistently computes eigenvalues as accurate as the data warrant (provided that the appropriate stopping criterion is enforced).

However, even when one uses DSTEBZ tailored to full precision, it is no simple matter, in many cases, to estimate the number of correct digits in the computed approximations for the eigenvalues of smaller size. It is a well-known fact that, for symmetric matrices, all eigenvalues are perfectly conditioned with respect to the norm of the matrix and any backwards stable algorithm will produce approximations $\widetilde{\lambda}_j$ for the exact eigenvalues λ_j such that

$$\left|\lambda_j - \widetilde{\lambda}_j\right| \leq O(\epsilon) \cdot \|T\|_2 \qquad (1.1)$$

holds for every $j = 1, \ldots, n$ (ϵ denotes the rounding error unit). When $\|T\|_2 / |\lambda_j|$ is

[*]Received by the editors December 8, 2010; accepted for publication (in revised form) by J. L. Barlow September 15, 2011; published electronically December 20, 2011. This research was financed by FEDER Funds through "Programa Operacional Factores de Competitividade - COMPETE" and by Portuguese Funds through FCT - "Fundação para a Ciência e a Tecnologia," within the Project PEst-C/MAT/UI0013/2011.

http://www.siam.org/journals/simax/32-4/81741.html

[†]Centro de Matemática, Universidade do Minho, 4710-057 Braga, Portugal (r_ralha@math.uminho.pt).

large, the previous bound may be too pessimistic, as it happens when T defines even tiny eigenvalues to high relative accuracy. This is the case of tridiagonal matrices with diagonal entries equal to zero (see Theorem 2 and Corollary 1 in [5]) whose importance derives from its connection with bidiagonal matrices. In [18], for errors induced in the eigenvalues of a general symmetric tridiagonal matrix by relative perturbations not larger than η, we have proved the bound $|\lambda_j - \widetilde{\lambda}_j| \leq 2.02n\eta(M + |\widetilde{\lambda}_j|)$, where M denotes the second largest absolute value of the diagonal entries. If $M = 0$, then we have high relative precision even for tiny eigenvalues. Scaled diagonally dominant matrices [2] is another important class of matrices for which some eigenvalues may be computed with errors smaller than $O(\epsilon) \cdot \|T\|_2$. For a more complete description of works on matrices that define well their eigenvalues and/or singular values, see [18] and the references there.

However, the question of knowing, for each particular eigenvalue, how many correct digits can be computed has no answer in general. We will show that by rounding towards $-\infty$ and $+\infty$ in the computation of the usual recurrence produces approximations $q_k^-(x)$ and $q_k^+(x)$, respectively, for each pivot $q_k(x)$, $k = 1, \ldots, n$, with very useful properties. A major result of this paper is Proposition 4.4 which states that the number of negative $q_k^+(x)$ and $q_k^-(x)$ are bounds, left and right, respectively, for the number of negative pivots. This result on its own allows us to produce guaranteed bounds for each eigenvalue if bisection is carried out based upon $q_k^-(x)$ and $q_k^+(x)$. Furthermore, the tightness of these bounds depends solely on the conditioning of the eigenvalue. Moreover, we have also derived bounds for an eigenvalue from a given approximation x when bounds for $q_n(x)$ are known.

The outline of the remainder of the paper is as follows. In the next section we recall the most basic facts about bisection; in section 3 we present the first result which gives sufficient conditions for $q_k^-(x)$ and $q_k^+(x)$ to be bounds for $q_k(x)$, for each $k = 1, \ldots, n$. In section 4 we analyze the cases where $q_k^-(x) \leq q_k(x) \leq q_k^+(x)$ does not hold for every k; in particular, we prove Proposition 4.4 mentioned before; in section 5 we discuss some implementation details on the computation of $q_k^-(x)$ and $q_k^+(x)$, in particular when only the rounding mode to zero is available; in section 6 we derive bounds for eigenvalues and illustrate their use with numerical examples given in section 7. We end up with some conclusions.

2. The bisection method.

A detailed description of the bisection algorithm can be found in [9], [13], or [16]. Let

$$T = \begin{bmatrix} d_1 & e_1 & & \\ e_1 & d_2 & \ddots & \\ & \ddots & \ddots & e_{n-1} \\ & & e_{n-1} & d_n \end{bmatrix} \tag{2.1}$$

be an $n \times n$ symmetric tridiagonal matrix. For any given real number x, if

$$T - xI = LDL^T, \tag{2.2}$$

where L is unit lower triangular and $D = \mathrm{diag}(q_1(x), \ldots, q_n(x))$ is diagonal, then

$$q_1(x) = d_1 - x, \tag{2.3}$$

$$q_k(x) = d_k - x - e_{k-1}^2/q_{k-1}(x), \qquad k = 2, \ldots, n. \tag{2.4}$$

According to Sylvester's law of inertia, the inertia of D equals the inertia of $T - xI$ so that the number of negative $q_k(x)$ gives the number of eigenvalues of T which are smaller than x. Following [8], we will use *count*(x) to denote this number. Kahan [15] carried out the first error analysis of the computation (2.3)–(2.4) which he has shown to be very stable. The following result is well known (see Lemma 5.3 in [9, p. 230]).

PROPOSITION 2.1. *The values $q_k(x)$ computed in floating point arithmetic using (2.3)–(2.4) have the same signs (and so compute the same inertia) as the $\widehat{q}_k(x)$ that would be obtained if exact arithmetic was carried out with the matrix $\widehat{T}$ such that*

$$\widehat{d}_k = d_k, \tag{2.5}$$

$$\widehat{e}_k = e_k\left(1+\delta_k\right), \quad \text{where } |\delta_k| \le 2.5\epsilon + O\left(\epsilon^2\right). \tag{2.6}$$

Therefore, the bisection method, correctly implemented, is able to deliver eigenvalues to high relative accuracy, even if they are much smaller than $\|T\|_2$, provided that T defines them well. Using the exception handling facilities of IEEE arithmetic, the computation produces a correct *count*(x) even when some $q_{k-1}(x)$ in (2.4) is exactly zero. In this case, $q_k(x) = -\infty$, $q_{k+1}(x) = d_{k+1} - x$, and the computation continues unexceptionably [7], [8]. A numerically robust, vectorized implementation of the algorithm is available in LAPACK's routine DSTEBZ (SSTEBZ for single precision), as mentioned before. For parallel processing, care must be taken to ensure the correctness of the results. The logic of the bisection algorithm depends on *count*(x) being a monotonic increasing function of x. However, depending upon the features of the arithmetic, monotonicity can fail and incorrect eigenvalues may be computed, because of rounding or as a result of using networks of heterogeneous parallel processors. In [8], several parallel algorithms are proposed and detailed analysis are carried out to ensure the correctness of the codes even when the arithmetic is nonmonotonic. One of such algorithms has been implemented in ScaLAPACK [3]. For an implementation of the bisection algorithm on GPUs (graphics processing units), see [19]. In the present work, we will assume monotonic arithmetic.

3. Guaranteed bounds for the pivots.

Interval arithmetic is a general computing technique that automatically provides guaranteed enclosures for the results. In general, results become less meaningful as the intervals become larger, but in our problem the correctness of *count*(x) depends only upon the signs of the $q_k(x)$ computed with (2.3)–(2.4), and not upon their numerical values; therefore, the size of each interval is irrelevant, as long as it is entirely contained in the positive part or in the negative part of the real axis. Now, we introduce the sequences $\{q_k^-(x)\}_{k=1,\dots,n}$ and $\{q_k^+(x)\}_{k=1,\dots,n}$ and show that their terms are usually bounds for the exact values $q_k(x)$. We have the following proposition.

PROPOSITION 3.1. *For any real number y, let $fl^-(y)$ and $fl^+(y)$ denote the floating point numbers obtained from the exact y with rounding to $-\infty$ and $+\infty$, respectively. Then, for exact values of d_k, e_k^2, and x, if we compute*

$$q_1^-(x) = fl^-\left(d_1 - x\right), \tag{3.1}$$

$$q_k^-(x) = fl^-\left(fl^-\left(d_k - x\right) + fl^-\left(-\frac{e_{k-1}^2}{q_{k-1}^-}\right)\right), \qquad k = 2,\dots,n, \tag{3.2}$$

$$q_1^+(x) = fl^+\left(d_1 - x\right), \tag{3.3}$$

$$q_k^+(x) = fl^+\left(fl^+\left(d_k - x\right) + fl^+\left(-\frac{e_{k-1}^2}{q_{k-1}^+}\right)\right), \qquad k = 2,\dots,n, \tag{3.4}$$

we have for the exact value of $q_k(x)$

$$q_k^-(x) \le q_k(x) \tag{3.5}$$

as long as

$$q_{k-1}^-(x) \le q_{k-1}(x), \tag{3.6}$$

and $q_{k-1}^-(x)$*, and* $q_{k-1}(x)$ *have the same sign. Similarly, if*

$$q_{k-1}(x) \le q_{k-1}^+(x)\,, \tag{3.7}$$

and $q_{k-1}^+(x)$*, and* $q_{k-1}(x)$ *have the same sign, then*

$$q_k(x) \le q_k^+(x)\,. \tag{3.8}$$

Proof (by induction). From (3.1) we have

$$q_1^-(x) \le q_1(x). \tag{3.9}$$

Assume that

$$q_{k-1}^-(x) \le q_{k-1}(x) \tag{3.10}$$

holds. If $q_{k-1}^-(x)$ and $q_{k-1}(x)$ are both positive or both negative, we may write, omitting x for simplicity,

$$\frac{e_{k-1}^2}{q_{k-1}} \le \frac{e_{k-1}^2}{q_{k-1}^-} \tag{3.11}$$

and

$$-\frac{e_{k-1}^2}{q_{k-1}^-} \le -\frac{e_{k-1}^2}{q_{k-1}}. \tag{3.12}$$

Therefore, we get

$$fl^-\left(fl^-(d_k - x) + fl^-\left(-\frac{e_{k-1}^2}{q_{k-1}^-}\right)\right) \le q_k(x). \tag{3.13}$$

The proof of 3.8 is similar. ▯

In practice, the sign of $q_k(x)$ will be guaranteed as long as it is bounded (from both sides) by quantities of the same sign. We have the following corollary.

COROLLARY 3.2. *If* $q_{k-1}^-(x)$ *and* $q_{k-1}^+(x)$ *agree in sign and*

$$q_{k-1}^-(x) \le q_{k-1}(x) \le q_{k-1}^+(x)$$

holds, then

$$q_k^-(x) \le q_k(x) \le q_k^+(x)\,.$$

Proof. This follows immediately from the previous proposition. ▯

4. When bounds are not all guaranteed.

When x is very close to some eigenvalue of the leading principal submatrix of T of order $k-1$, for some $k \leq n$, we may get

$$q_{k-1}^{-}(x) < 0 < q_{k-1}^{+}(x), \tag{4.1}$$

and the previous corollary does not guarantee bounds for $q_k(x)$. Nevertheless, in this situation we are likely to get

$$q_k^{+}(x) < 0 < q_k^{-}(x) \tag{4.2}$$

because from (4.1) it follows that

$$-\frac{e_{k-1}^2}{q_{k-1}^{+}(x)} < 0 < -\frac{e_{k-1}^2}{q_{k-1}^{-}(x)},$$

and in most cases, these ratios will have a bigger size than $d_k - x$, so that (4.2) follows. Now, it is straightforward to show that, for $k < n$, we have

$$\left[q_k^{+}(x) < 0 < q_k^{-}(x)\right] \Rightarrow q_{k+1}^{-}(x) < q_{k+1}^{+}(x), \tag{4.3}$$

and it is natural to ask whether the term $q_{k+1}(x)$ is again between $q_{k+1}^{-}(x)$ and $q_{k+1}^{+}(x)$. It turns out that this can be guaranteed if $q_k(x) \leq q_k^{+}(x)$ or $q_k^{-}(x) \leq q_k(x)$. We have the following proposition.

PROPOSITION 4.1. *Let x be such that* (4.2) *holds, for $k < n$, and one of the bounds $q_k^{-}(x)$ and $q_k^{+}(x)$ is correct, i.e., we have either* (a) $0 < q_k^{-}(x) < q_k(x) < +\infty$ *or* (b) $-\infty < q_k(x) < q_k^{+}(x) < 0$. *Then, we get*

$$q_{k+1}^{-}(x) < q_{k+1}(x) < q_{k+1}^{+}(x)\,. \tag{4.4}$$

Furthermore, if none of the bounds $q_k^{-}(x)$ and $q_k^{+}(x)$ is correct, we may still guarantee that one of the bounds for $q_{k+1}(x)$ is correct.

Proof. We are observing each case separately.

(a) From Proposition 3.1 it follows that

$$q_{k+1}^{-}(x) < q_{k+1}(x).$$

Further we have

$$\begin{aligned} q_{k+1}(x) = d_{k+1} - x - \frac{e_k^2}{q_k(x)} &< d_{k+1} - x \leq fl^{+}(d_{k+1} - x) \\ &< fl^{+}(d_{k+1} - x) + fl^{+}\left(-\frac{e_k^2}{q_k^{+}(x)}\right) \leq q_{k+1}^{+}(x). \end{aligned} \tag{4.5}$$

(b) From Proposition 3.1 it follows that

$$q_{k+1}(x) < q_{k+1}^{+}(x).$$

Further we have

$$\begin{aligned} q_{k+1}(x) = d_{k+1} - x - \frac{e_k^2}{q_k(x)} &> d_{k+1} - x \geq fl^{-}(d_{k+1} - x) \\ &> fl^{-}(d_{k+1} - x) + fl^{-}\left(-\frac{e_k^2}{q_k^{-}(x)}\right) \geq q_{k+1}^{-}(x). \end{aligned} \tag{4.6}$$

Finally, for the bound (4.5) to hold it just needs to be $q_k(x) > 0$ and $q_k^{+}(x) < 0$. For the bound (4.6) to hold it just needs to be $q_k(x) < 0$ and $q_k^{-}(x) > 0$. ∎

COROLLARY 4.2. *If, with $q_{k-1}^{-}(x) < 0$ and $q_{k-1}^{+}(x) > 0$, we have $q_{k-1}^{-}(x) \leq q_{k-1}(x) \leq q_{k-1}^{+}(x)$, and also $q_k^{+}(x) < 0 < q_k^{-}(x)$, then*

$$q_{k+1}^{-}(x) < q_{k+1}(x) < q_{k+1}^{+}(x)\,.$$

Proof. From Proposition 3.1, it is either $q_k^{-}(x) < q_k(x)$ or $q_k(x) < q_k^{+}(x)$, depending upon $q_{k-1}(x)$ being negative or positive, respectively. Therefore, in this case, if $q_k^{+}(x) < 0 < q_k^{-}(x)$ holds, Proposition 4.1 guarantees the bounds for $q_{k+1}(x)$. ▯

There are situations in which (4.1) holds but not (4.2), so that Corollary 4.2 does not apply. This happens in the following.

Example 4.3. Let

$$T = \begin{bmatrix} 10^{-7} & 1 & & \\ 1 & 10^{7}\left(1+2^{-5}\right) & 10^{-9} & \\ & 10^{-9} & 1 & 1 \\ & & 1 & 1+6\times 10^{-9} \end{bmatrix}.$$

The eigenvalues of T, as given by function eig in MATLAB, are

$$\begin{aligned} \widetilde{\lambda}_1 &= 3.001332515850663e-9, \\ \widetilde{\lambda}_2 &= 3.030303030303128e-9, \\ \widetilde{\lambda}_3 &= 2.000000003724001, \\ \widetilde{\lambda}_4 &= 1.031250000000010e+7. \end{aligned}$$

The smallest eigenvalue of the leading principal submatrix of order 2, as given by eig, is $x = 3.030303030302996e-9$, quite close to $\widetilde{\lambda}_2$. For this value x, we get

$$\begin{aligned} &q_1^{-}(x) = 9.6969,\ldots,e-8, \quad && q_1^{+}(x) = 9.6969,\ldots,e-8; \\ &q_2^{-}(x) = -1.8626,\ldots,e-9, \quad && q_2^{+}(x) = 1.8626,\ldots,e-9; \\ &q_3^{-}(x) = 9.999999975065678e-1, \quad && q_3^{+}(x) = 9.999999964328261e-1; \\ &q_4^{-}(x) = 4.7626,\ldots,e-10, \quad && q_4^{+}(x) = -5.9747,\ldots,e-10. \end{aligned}$$

Corollary 4.2 does not apply, for $k = 3$, because $q_3^{+}(x) > 0$. Nevertheless, we may conclude that $count(x) = 1$ by considering all possible cases for the signs of $q_2(x)$, $q_3(x)$ and $q_4(x)$. We do not know whether $q_2(x) < 0$ or $q_2(x) > 0$. If $q_2(x) < 0$, then Proposition 3.1 gives $0 < q_3^{-}(x) < q_3(x)$ and $0 < q_4^{-}(x) < q_4(x)$. Now the case $q_2(x) > 0$, for which, again using Proposition 3.1, we get $q_3(x) < q_3^{+}(x)$. If $q_3(x) > 0$, we get $q_4(x) < q_4^{+}(x) < 0$, and if $q_3(x) < 0$ then, according to (4.6), we have $q_4(x) > q_4^{-}(x) > 0$. Therefore we conclude that $count(x) = 1$.

From now on, we will use $count^{+}(x)$ and $count^{-}(x)$ to denote the number of negative occurrences in the recurrences to compute $q_k^{+}(x)$ and $q_k^{-}(x)$, for a given matrix and a given point x. The previous example leads us to raise the following question: if $count^{+}(x) = count^{-}(x)$, can we conclude that this number is the right value of $count(x)$? The answer is yes. We have the following proposition.

PROPOSITION 4.4. *For every x, we have*

$$count^{+}(x) \leq count(x) \leq count^{-}(x). \tag{4.7}$$

Proof. Let us prove that $count^{+}(x) \leq count(x)$. We use $count_j^{+}(x)$ and $count_j(x)$ to denote the number of negative $q_k^{+}(x)$ and $q_k(x)$ for $k \leq j$. From Proposition 3.1

we know that

$$q_{k-1}(x) \leq q^+_{k-1}(x) \Rightarrow q_k(x) \leq q^+_k(x)$$

as long as $q_{k-1}(x)$ and q^+_{k-1} have the same sign. The first disagreement in sign, if any, occurs with

$$q_j(x) < 0 < q^+_j(x)$$

for some $j \leq n$. At this point we have $count^+_j(x) = count_j(x) - 1$. Now, let $p > j$ be the next first index, if any, such that

$$q^+_p(x) < 0 < q_p(x).$$

At this point, it is $count^+_p(x) \leq count_p(x)$ and, as in Proposition 4.1, we have $q_{p+1}(x) < q^+_{p+1}(x)$. Applying the same reasoning to the rest of the sequence, we conclude that $count^+(x) \leq count(x)$. The proof of $count(x) \leq count^-(x)$ is similar. ◻

5. Rounding towards $-\infty$, $+\infty$, and zero. In a practical implementation, to compute $q^-_k(x)$ and $q^+_k(x)$, as expressed in (3.1)–(3.4), we do not need to switch from one rounding mode to the other, since, for every real number y,

$$fl^-(y) = -fl^+(-y) \tag{5.1}$$

holds. Therefore, we may set the rounding mode set to $+\infty$ and compute

$$q^-_1(x) = -(x - d_1), \tag{5.2}$$
$$q^-_k(x) = -\left(x - d_k + e^2_{k-1}/q^-_{k-1}\right), \quad k = 2, \ldots, n, \tag{5.3}$$
$$q^+_1(x) = d_1 - x, \tag{5.4}$$
$$q^+_k(x) = d_k - x + \left(-e^2_{k-1}/q^+_{k-1}\right), \quad k = 2, \ldots, n. \tag{5.5}$$

We have implemented these computations in a MATLAB code which we have dubbed *BoundsQInf.* Each one of the relations (5.2)–(5.5) results from the corresponding relation in (3.1)–(3.4) by simply removing fl^+ and applying the rule in (5.1) to fl^-.

Although the IEEE754 arithmetic advocates the existence of four rounding modes, to nearest, to −Inf, to Inf or to zero (chopping), there are processors that do not offer such options. This is the case of the IBM Cell processor which always rounds to zero. For this reason, it is of interest to produce the bounds $q^-_k(x)$ and $q^+_k(x)$ without using rounding to Inf, and we now show how to achieve this.

For a given number y, we keep using $fl^-(y)$ and $fl^+(y)$ to denote the consecutive floating point numbers of the representation system such that $fl^-(y) \leq y \leq fl^+(y)$. It is trivial to observe that rounding to zero produces $fl^-(y)$ and $fl^+(y)$ for y positive and negative, respectively. By adding one unit in the last position (ulp) of the mantissa of $fl^-(y)$ or $fl^+(y)$ we get the other bound. This can be achieved at the cost of an extra multiplication as we show in the following proposition.

PROPOSITION 5.1. *Let y be a number which has no exact representation in the system being used. Denoting by $fl(y)$, the IEEE-754 normalized representation of y*

with the rounding to zero (chopping), we have

$$fl^+(y) = \begin{cases} fl(y) \text{ if } fl(y) < 0, \\ fl(fl(y) * (1+2^{-t})) \text{ if } fl(y) > 0, \end{cases} \tag{5.6}$$

$$fl^-(y) = \begin{cases} fl(y) \text{ if } fl(y) > 0, \\ fl(fl(y) * (1+2^{-t})) \text{ if } fl(y) < 0, \end{cases} \tag{5.7}$$

where t denotes the number of bits in the mantissa (in IEEE-754, this is 23 for single precision and 52 for double precision).

Proof. Let us consider the bound $fl^+(y)$ when $fl(y) > 0$. Writing

$$fl(y) = \left(1 + b_1 \times 2^{-1} + \cdots + b_t \times 2^{-t}\right) \times 2^E,$$

where $b_i \in \{0,1\}$ and E is the exponent in the normalized representation of $fl(y)$, we have

$$fl(y) \cdot \left(1 + 2^{-t}\right) = \left(1 + b_1 \times 2^{-1} + \cdots + b_t \times 2^{-t} + 2^{-t} + R\right) \times 2^E,$$

where

$$R = b_1 \times 2^{-t-1} + \cdots + b_t \times 2^{-2t}.$$

Since $R < 2^{-t}$, it will be chopped, and we conclude that $fl\left(fl(y) \cdot (1+2^{-t})\right)$ differs from $fl(y)$ by one ulp in the mantissa. Finally, if $fl(y) < 0$, the relation (5.1) immediately gives the expression for $fl^-(y)$ in (5.7). ◻

Therefore, if we use the relations (5.6) and (5.7) to compute each fl^- and fl^+ in (3.1)–(3.4), we produce the bounds $q_k^-(x)$ and $q_k^+(x)$, as desired. We have implemented these computations in a MATLAB code which we have dubbed *BoundsQchop*. At this point, we note that the bounds for the pivots produced with our code *BoundsQchop* are somewhat more slack than those produced with *BoundsQinf*. This follows from the fact that when y has an exact representation, we have $fl^-(y) = y = fl^+(y)$ for the bounds computed with *BoundsQinf* whereas *BoundsQchop* always produces an interval $[fl^-(y), fl^+(y)]$ which is one ulp wide. In our numerical examples we found this difference to have little impact in the accuracy of the eigenvalues computed with *BoundsQchop*.

6. Computing bounds for an eigenvalue.

Each one of the two sequences $\left\{q_k^+(x)\right\}$ and $\left\{q_k^-(x)\right\}$ may be used in an independent manner to compute the eigenvalues of T

$$\lambda_1 \le \lambda_2 \le \cdots \le \lambda_n$$

with the bisection method (the fact that we are using rounding to ∞ only makes ϵ twice as large in the bound (2.6), as compared to rounding to the "nearest"). When looking for λ_k, for some k, suppose that we have produced the intervals $[a^-, b^-]$ and $[a^+, b^+]$, as thin as possible, such that $count^-(a^-) < k$, $count^-(b^-) \ge k$ and $count^+(a^+) < k$, $count^+(b^+) \ge k$. Then, we certainly have

$$a^- \le \lambda_k \le b^+$$

since from (4.7) it follows that

$$count(a^-) \le count^-(a^-) < k$$

and

$$count(b^+) \geq count^+(b^+) \geq k.$$

The cost of this method is, of course, twice the cost of usual bisection but it may be of interest for parallel processing since the computation of each sequence $\left\{q_k^+(x)\right\}$ and $\left\{q_k^-(x)\right\}$ is independent of the other. Most important, it provides a trustful interval $[a^-, b^+]$ whose relative gap will be very small if λ_k is defined to high relative accuracy. For sequential processing, a more efficient alternative consists of producing only one of the bounds, let it be a^-, and then searching for b^+, as small as possible, to the right of a^-, such that $count^+(b^+) \geq k$.

Here, we also envisage an alternative use of (4.7). Suppose that we are given an approximation $\widetilde{\lambda}_k$ (which may have been produced by any method, not necessarily bisection). From this, we may derive an interval that contains λ_k and that, if necessary, may be refined with bisection (computing $count^-(x)$ and $count^+(x)$ for some points x) or with a faster method like Newton's. We have the following proposition.

PROPOSITION 6.1. *Let $x = \widetilde{\lambda}_j$ be an approximation for an eigenvalue λ_j such that there is no pole of q_n (root of q_{n-1}) between x and λ_j. Then, we have*
(i)

$$|\lambda_j - x| \leq |q_n(x)|\,; \tag{6.1}$$

(ii) *if $q_n(x)$ and $q_{n-1}(x)$ have the same sign, then*

$$|\lambda_j - x| \leq \left|\frac{q_n(x)}{1 + e_{n-1}^2/q_{n-1}^2(x)}\right|. \tag{6.2}$$

Proof. Since there is no pole of q_n between x and λ_j, the mean value theorem tells us that there is θ between x and λ_j such that

$$q_n(\lambda_j) - q_n(x) = q_n'(\theta)\,(\lambda_j - x)$$

or, since λ_j is a root of q_n,

$$\lambda_j - x = -\frac{q_n(x)}{q_n'(\theta)}. \tag{6.3}$$

From (2.3) and (2.4) we get

$$q_1'(\theta) = -1, \tag{6.4}$$

$$q_i'(\theta) = -1 + \frac{e_{i-1}^2}{q_{i-1}^2(\theta)} \cdot q_{i-1}'(\theta), \quad i = 2, \ldots, n. \tag{6.5}$$

Simple induction shows that $q_i'(\theta) \leq -1$ for every $i = 1, \ldots, n$ and (6.1) follows immediately from (6.3). Similarly, for any B such that

$$q_n'(\theta) \leq B \leq -1, \tag{6.6}$$

from (6.3) we get

$$|\lambda_j - x| \leq \frac{|q_n(x)|}{|B|}. \tag{6.7}$$

We now prove (ii). Both q_{n-1} and q_n are decreasing, inside their intervals of continuity. If $q_{n-1}(x)$ and $q_n(x)$ are both positive, we have $x < \theta < \lambda_j$ and

$$q_{n-1}^2(\theta) < q_{n-1}^2(x). \tag{6.8}$$

Therefore

$$B = -1 - \frac{e_{n-1}^2}{q_{n-1}^2(x)} \tag{6.9}$$

satisfies (6.6). If $q_{n-1}(x)$ and $q_n(x)$ are both negative, we have $\lambda_j < \theta < x$, so that (6.8) is true. □

In practice, if $q_n^-(x)$ and $q_n^+(x)$ are bounds with the same sign, then it is straightforward to verify whether Proposition 6.1 can be applied. We have the following proposition.

PROPOSITION 6.2. *There is no pole of q_n between x and λ_j if and only if one of the following conditions is true:*

$$count(x) = j - 1 \textit{ and } q_n(x) > 0, \tag{6.10}$$

$$count(x) = j \textit{ and } q_n(x) < 0. \tag{6.11}$$

Proof. Assume that (6.10) holds and let μ_{j-1} be the pole between λ_{j-1} and λ_j. Since $count(x) = j-1$, x is also a point between λ_{j-1} and λ_j. Because q_n is negative in $]\lambda_{j-1}, \mu_{j-1}[$, we conclude that $q_n(x) > 0$ implies that there is no pole between x and λ_j. The rest of the proof is similar. □

In practice, we use the following corollaries.

COROLLARY 6.3. *Let x be such that $count^+(x) = count^-(x) = j-1$. Then*

$$0 < q_n^-(x) \le q_n(x) \le q_n^+(x) \Rightarrow x \le \lambda_j \le x + q_n^+(x). \tag{6.12}$$

Furthermore, if we also have

$$0 < q_{n-1}^-(x) \le q_{n-1}(x) \le q_{n-1}^+(x),$$

then we get

$$x \le \lambda_j \le x + \frac{q_n^+(x)}{1 + e_{n-1}^2/(q_{n-1}^+(x))^2}. \tag{6.13}$$

COROLLARY 6.4. *Let x be such that $count^+(x) = count^-(x) = j$. Then*

$$q_n^-(x) \le q_n(x) \le q_n^+(x) < 0 \Rightarrow x + q_n^-(x) \le \lambda_j \le x. \tag{6.14}$$

Furthermore, if we also have

$$q_{n-1}^-(x) \le q_{n-1}(x) \le q_{n-1}^+(x) < 0,$$

then we get

$$x + \frac{q_n^-(x)}{1 + e_{n-1}^2/(q_{n-1}^-(x))^2} \le \lambda_j \le x. \tag{6.15}$$

7. Numerical examples. We now present some examples to illustrate the use of the bounds given in the previous section.

Example 7.1. The matrix

$$T = \begin{bmatrix} 1 & b & 0 \\ b & a & b \\ 0 & b & 1 \end{bmatrix} \tag{7.1}$$

is positive definite for $a > 2b^2$ and has eigenvalues 1 and $\frac{1}{2}a \pm \frac{1}{2}\sqrt{(a-1)^2 + 8b^2} + \frac{1}{2}$. For very small $|a|$ and $|b|$, one of the eigenvalues gets close to 0. For $a = 10^{-32}$ and $b = 0.15 \times 10^{-16}$, we get T as given in [6]. Using the function eig in MATLAB we get

$$\begin{aligned} \widetilde{\lambda}_1 &= 9.550000000000001e-033, \\ \widetilde{\lambda}_2 &= 1.000000000000000e+000, \\ \widetilde{\lambda}_3 &= 1.000000000000000e+000. \end{aligned}$$

How accurate, in fact, is $\widetilde{\lambda}_1$? For $x_0 = \widetilde{\lambda}_1$, our code *BoundsQinf* produces the following guaranteed intervals for the pivots:

$$\begin{aligned} [q_1^-(x_0), q_1^+(x_0)] &= [9.9999, \ldots, e-001,\ 1.0000, \ldots, e+000], \\ [q_2^-(x_0), q_2^+(x_0)] &= [2.2499, \ldots, e-034,\ 2.2499, \ldots, e-034], \\ [q_3^-(x_0), q_3^+(x_0)] &= [-1.4432, \ldots, e-015,\ \ -1.3322, \ldots, e-015]. \end{aligned}$$

We conclude that $count(x_0) = 1$ and, since $q_3(x_0)$ and $q_2(x_0)$ disagree in sign, Corollary 6.4 allows us to guarantee that

$$\lambda_1 \in \left[x + q_3^-(x_0), x_0\right]$$

only, which does not guarantee any relative accuracy in the computed $\widetilde{\lambda}_1$. To produce better bounds, if required, one may carry out a few bisection steps or, since a good approximation is already available, use a method with better asymptotic convergence rate. In this case, if we carry out one iteration of Newton's method, the computed values are $q_3'(x_0) = -4.44, \ldots, e+033$ and $x_1 = x - q_3(x)/q_3'(x)$, which turns out to be equal to x_0. The latter result makes us believe that $x_0 = \widetilde{\lambda}_1$ is indeed very accurate. To confirm this, we compute the bounds $q_i^-(z)$ and $q_i^+(z)$, $i = 1, 2, 3$, for $z = \widetilde{\lambda}_1 \left(1 - 2^{-53}\right)$, where z is the largest floating point number smaller than $\widetilde{\lambda}_1$, and observe that those bounds are all positive so that

$$\lambda_1 \in \left[\widetilde{\lambda}_1 \left(1 - 2^{-53}\right), \widetilde{\lambda}_1\right].$$

In the previous example, the initial approximation $\widetilde{\lambda}_1$ is quite accurate so that our algorithm just confirms such accuracy. In other cases, the initial approximation is not as accurate as the data warrants and there is scope for improvement. This is illustrated in the following example.

Example 7.2. For

$$T = \begin{bmatrix} 1 & 10^{10} & 0 \\ 10^{10} & 10^5 & 10^3 \\ 0 & 10^3 & 3 \end{bmatrix} \tag{7.2}$$

eig delivers

$$\begin{aligned}\widetilde{\lambda}_1 &= -9.999949999625046e+009,\\ \widetilde{\lambda}_2 &= 2.999997255728966e+000,\\ \widetilde{\lambda}_3 &= 1.000005000062505e+010.\end{aligned}$$

We are interested in the eigenvalue of smaller size. With $x = \widetilde{\lambda}_2$ we get

$$\begin{aligned}[q_1^-(x), q_1^+(x)] &= [-1.9999,\ldots,e-000,\ \ -1.9999,\ldots,e+000],\\ [q_2^-(x), q_2^+(x)] &= [5.0000,\ldots,e+019,\ 5.0000,\ldots,e+019],\\ [q_3^-(x), q_3^+(x)] &= [2.7442,\ldots,e-006,\ 2.7442,\ldots,e-006].\end{aligned}$$

In this case, the bounds in (6.13) hold but they are not better than those in (6.12) since $e_2^2/\left(q_2^+(x)\right)^2$ is very small. We have

$$\lambda_2 \in \left[\widetilde{\lambda}_2, \widetilde{\lambda}_2 + q_3^+(\widetilde{\lambda}_2)\right],$$

and with $x = \widetilde{\lambda}_2 + q_3^+(\widetilde{\lambda}_2) = 2.999999999999980$ we get

$$\begin{aligned}[q_1^-(x), q_1^+(x)] &= [-1.9999,\ldots,e-000,\ \ -1.9999,\ldots,e+000],\\ [q_2^-(x), q_2^+(x)] &= [5.0000,\ldots,e+019,\ 5.0000,\ldots,e+019],\\ [q_3^-(x), q_3^+(x)] &= [-1.5986,\ldots,e-017,\ \ -1.5986,\ldots,e-017],\end{aligned}$$

and from (6.14) conclude that $x = 2.999999999999980$ is a full accurate approximation of λ_2.

8. Conclusions.

The bisection method, as implemented in the LAPACK routine DSTEBZ, is able to compute approximations for the eigenvalues of a symmetric tridiagonal matrix T that are the exact ones corresponding to a matrix which differs from T by small relative perturbations. Eigenvalues of magnitude much smaller than $\|T\|_2$ may be computed with absolute errors much smaller than $\epsilon\,\|T\|_2$, depending upon the way they are defined by the entries of T. The question of knowing, for each eigenvalue, how many correct digits can be computed has no general answer. We have shown that rounding towards $+\infty$ and $-\infty$ in the computation of the usual recurrence allows us to produce guaranteed bounds for the eigenvalues. These bounds are tight when the eigenvalues are defined well.

Acknowledgments. The author is indebted to B. Parlett and K. Veselić for various comments and suggestions.

REFERENCES

[1] E. Anderson et al., *LAPACK Users' Guide*, SIAM, Philadelphia, 1999.

[2] J. Barlow and J. Demmel, *Computing accurate eigensystems of scaled diagonally dominant matrices*, SIAM J. Numer. Anal., 27 (1990), pp. 762–791.

[3] L. Blackford et al., *ScaLAPACK Users' Guide*, SIAM, Philadelphia, 1997.

[4] J. J. M. Cuppen, *A divide and conquer method for the symmetric tridiagonal eigenvalue problem*, Numer. Math., 36 (1981), pp. 177–195.

[5] J. Demmel and W. Kahan, *Accurate singular values of bidiagonal matrices*, SIAM J. Sci. Statist. Comput., 11 (1990), pp. 873–912.

[6] J. W. Demmel, *The inherent inaccuracy of implicit tridiagonal QR*, LAPACK Working Note #45, 1992.

[7] J. W. Demmel and X. Li, *Faster numerical algorithms via exception handling*, IEEE Trans. Comput., 43 (1994), pp. 983–992. (Also LAPACK Working Note #59.)

[8] J. W. Demmel, I. Dhillon, and H. Ren, *On the correctness of some bisection-like parallel eigenvalue algorithms in floating point arithmetic*, Electron. Trans. Numer. Anal., 3 (1995), pp. 116–149. (Also LAPACK Working Note #70.)

[9] J. W. Demmel, *Applied Numerical Linear Algebra*, SIAM, Philadelphia, 1997.

[10] I. S. Dhillon, *A New $O(n^2)$ Algorithm for the Symmetric Tridiagonal Eigenvalue/Eigenvector Problem*, Ph.D. Thesis, University of California, Berkeley, CA, 1997.

[11] I. S. Dhillon and B. N. Parlett, *Multiple representations to compute orthogonal eigenvectors of symmetric tridiagonal matrices*, Linear Algebra Appl., 387 (2004), pp. 1–28.

[12] K. V. Fernando and B. N. Parlett, *Accurate singular values and differential qd algorithms*, Numer. Math., 67 (1994), pp. 191–229.

[13] G. Golub and C. V. Loan, *Matrix Computations*, 2nd ed., The Johns Hopkins University Press, Baltimore, 1989.

[14] M. Gu and S. C. Eisenstat, *A divide-and-conquer algorithm for the symmetric tridiagonal eigenvalue problem*, SIAM J. Matrix Anal. Appl., 16 (1995), pp. 172–191.

[15] W. Kahan, *Accurate Eigenvalues of a Symmetric Tri-diagonal Matrix*, Technical Report CS41, Computer Science Department, Stanford University, Palo Alto, CA, 1966.

[16] B. Parlett, *The Symmetric Eigenvalue Problem*, Prentice-Hall, Englewood Cliffs, NJ, 1980.

[17] B. N. Parlett, *The new qd algorithms*, Acta Numer., 4 (1995), pp. 459–491.

[18] R. Ralha, *Perturbation splitting for more accurate eigenvalues*, SIAM J. Matrix Anal. Appl., 31 (2009), pp. 75–91.

[19] V. Volkov and J. Demmel, *Using GPUs to accelerate the bisection algorithm for finding eigenvalues of symmetric tridiagonal matrices*, LAPACK Working Note #197, 2008.

[20] P. Willems, *On MR^3-type Algorithms for the Tridiagonal Symmetric Eigenproblem and the Bidiagonal SVD*, Ph.D. Thesis, Bergische Universitat Wuppertäl, Fachbereich Mathematik und Naturwissenschaften, Wuppertäl, Germany, 2010.