SIAM J. MATRIX ANAL. APPL.
Vol. 25, No. 3, pp. 858–899

ORTHOGONAL EIGENVECTORS AND RELATIVE GAPS*

INDERJIT S. DHILLON† AND BERESFORD N. PARLETT‡

Abstract. This paper presents and analyzes a new algorithm for computing eigenvectors of symmetric tridiagonal matrices factored as LDL^t, with D diagonal and L unit bidiagonal. If an eigenpair is well behaved in a certain sense with respect to the factorization, the algorithm is shown to compute an approximate eigenvector which is accurate to working precision. As a consequence, all the eigenvectors computed by the algorithm come out numerically orthogonal to each other without making use of any reorthogonalization process. The key is first running a qd-type algorithm on the factored matrix LDL^t and then applying a fine-tuned version of inverse iteration especially adapted to this situation. Since the computational cost is $O(n)$ per eigenvector for an $n \times n$ matrix, the proposed algorithm is the central step of a more ambitious algorithm which, at best (i.e., when all eigenvectors are well-conditioned), would compute all eigenvectors of an $n \times n$ symmetric tridiagonal at $O(n^2)$ cost, a great improvement over existing algorithms.

Key words. eigenvectors, orthogonality, symmetric tridiagonal matrix, bidiagonal factorization, high relative accuracy

AMS subject classifications. 15A18, 15A23

DOI. 10.1137/S0895479800370111

1. Setting the scene.

This section is addressed to a broader audience than is the rest of the paper.

A real symmetric matrix has a full set of orthogonal eigenvectors, and users of software expect computed eigenvectors to be orthogonal to working accuracy. Excellent programs are available to diagonalize real symmetric matrices, so we might be tempted to say that the problem of computing orthogonal eigenvectors is solved. The best approach has three phases: (1) reducing the given dense symmetric matrix A to tridiagonal form T, (2) computing the eigenvalues and eigenvectors of T, and (3) mapping T's eigenvectors into those of A. For an $n \times n$ matrix the first and third phases require $O(n^3)$ arithmetic operations each. There are several choices for the second phase. The QR algorithm is simple but rather slow. The time-consuming part is the accumulation of $O(n^2)$ plane rotations, each of which requires $O(n)$ operations. Yet we must remember that it is this accumulation that guarantees numerically orthogonal eigenvectors, however close some of the eigenvalues may be, and that is a beautiful feature of the QR algorithm [15, 26]. An attractive feature of QR is that it requires only $O(n^2)$ operations to compute the eigenvalues alone. In principle, once the eigenvalues are known, one can invoke inverse iteration to independently compute each eigenvector at a cost of $O(n)$ per eigenvector. For distributed memory computers this feature would permit computation of the eigenvectors in parallel. The blemish in this approach is that the computed eigenvectors may not be numerically orthogonal when some eigenvalues are close, say agreeing to more than three decimals. So inverse

*Received by the editors March 31, 2000; accepted for publication (in revised form) July 8, 2003; published electronically March 30, 2004.

http://www.siam.org/journals/simax/25-3/37011.html

†Department of Computer Sciences, University of Texas, Austin, TX 78712-1188 (inderjit @cs.utexas.edu). Part of this author's research was supported while the author was at the University of California, Berkeley, by DARPA contract DAAL03-91-C-0047 and NSF grants ASC-9313958 and CDA-9401156. At the University of Texas, this research was supported by a University startup grant and NSF CAREER Award ACI-0093404.

‡Mathematics Department and Computer Science Division, EECS Department, University of California, Berkeley, CA 94720 (parlett@math.berkeley.edu).

iteration is augmented with a Gram–Schmidt process to enforce orthogonality, thus making it an $O(n^3)$ procedure in the worst case. A very careful implementation of this approach is available in the LAPACK library [25]. Since the mid 1990s, stable versions of the divide and conquer method for Phase 2 have been available [21]. Divide and conquer is a fast method much of the time but can reduce to an $O(n^3)$ process for rather uniform eigenvalue distributions.

With n near 1000 there are cases where the $O(n^3)$ reduction of a dense matrix to tridiagonal form T takes *much less* time (10–20%) than computing T's eigenvectors by inverse iteration. For example, the tridiagonal reduction of a certain 1000×1000 dense matrix takes about 10 seconds, while LAPACK's bisection and inverse iteration software takes 93 seconds to compute all the eigenvalues and eigenvectors of the tridiagonal. The timings for a 2000×2000 matrix clearly show the $O(n^3)$ behavior: 101 seconds for tridiagonalization and 839 seconds for solving the tridiagonal eigenproblem; detailed timing results are given in section 8.1. This behavior was an unpleasant surprise for the guardians of LAPACK and was brought to our attention in 1995 by a group of computational quantum chemists who were interested in a parallel solution. For one of their examples of order 966 there was a cluster of 931 eigenvalues deemed to be close to each other, and the Gram–Schmidt process in inverse iteration was consuming all the time (see section 8.1). It was time to re-evaluate Phase 2.

As values of n near 10^3 become common and values exceeding 10^4 do occur, it is hard to stop people dreaming of an algorithm that is guaranteed to compute a numerically orthogonal set of eigenvectors of T in $O(n^2)$ operations in the worst case. The presence of parallel distributed memory computer systems has vitalized the search for such an algorithm. Ideally the n eigenvalues would be equally distributed among all processors, along with a copy of the tridiagonal, and the eigenvectors would be computed independently at the same time and would turn out to be orthogonal to working accuracy.

There are formidable obstacles that impede the realization of this dream, and these will be reviewed in the next section.

This paper presents a central step towards the goal. The method proposed, in section 4, is Algorithm Getvec for computing a single eigenvector in $O(n)$ time. The main theorem, Theorem 15 in section 7, shows that in special, but important, situations (see below) our new algorithm produces an eigenvector that is guaranteed to be within $O(n\varepsilon)$ of the true eigenvector whenever the eigenvalue has a *relative* separation from its neighbors that exceeds a threshold *tol*, say 10^{-3}. It has been known for years that inverse iteration can produce fully accurate eigenvectors whenever the eigenvalue has an *absolute* separation that is above the threshold. So our contribution is to change absolute to relative in the separation condition. Our examples show that the resulting speedups can be dramatic (from 839 seconds to 4.6 seconds). Section 8 contains detailed experimental results. To establish our result, roundoff errors included, we were obliged to jettison the traditional representation of a tridiagonal matrix by its diagonal and next-to-diagonal entries. Instead, we use a bidiagonal factorization LDL^t of a carefully chosen translate of the original tridiagonal T. The crucial properties that must be satisfied in order for Algorithm Getvec to compute an accurate approximation to eigenvector $\boldsymbol{v}$ (corresponding to eigenvalue λ of LDL^t) are that (i) both λ and $\boldsymbol{v}$ must be determined to high relative accuracy by the parameters in L and D and (ii) the *relative gap* between λ and its nearest neighbor μ in the spectrum should exceed *tol*; $|\lambda - \mu| > tol \cdot |\lambda|$. The phrase "determined to high relative accuracy" is explained in section 2. We say that an LDL^t factorization that satisfies

property (i) is a *relatively robust representation* (RRR) for the eigenpair $(\lambda, \boldsymbol{v})$. A positive (or negative) definite LDL^t factorization is known to be an RRR for all its eigenpairs [7]; in section 6.1 we give conditions for an indefinite LDL^t to be an RRR.

The proof of the main theorem, Theorem 15, rests on the existence of relative perturbation results for the bidiagonal factors and on a special interpretation of the roundoff errors in differential qd algorithms that yields what is called mixed stability: carefully selected small relative perturbations of both the input and the output of our subroutines reveal the existence of an exact relationship of the form $\bar{L}\bar{D}\bar{L}^t - \hat{\lambda}I = \bar{N}\bar{D}\bar{N}^t$, where $\bar{N}$ is a "twisted" factor defined in section 4 and $\hat{\lambda}$ is an approximation to λ. The translation by $\hat{\lambda}$ preserves eigenvectors while shifting the eigenvalue of interest very close to 0. The middle part of this paper presents the relevant error analysis. Although essential for our results, this analysis will be indigestible for most readers, but it tells us that changes of only 3 or 4 units in the last digit of each entry of the input L, D and the output twisted factors suffice to give the exact relation.

The algorithm presented in this paper, Algorithm Getvec, allows us to compute a very accurate eigenvector for each eigenvalue that has a large relative separation from its neighbors ($> tol$). How do we compute numerically orthogonal approximations to eigenvectors when relative gaps are smaller? The full method, Algorithm MRRR or MR^3 (Algorithm of Multiple Relatively Robust Representations), crucially depends on Algorithm Getvec but is beyond the scope of the paper and is described in detail in [10]. Here we briefly sketch the outline of Algorithm MR^3 to show how it is based on the results of this paper: Compute the extreme eigenvalues of T and start with a base τ at one end of the spectrum. Compute the positive (or negative) definite factorization $LDL^t = \pm(T - \tau I)$ and find all its eigenvalues to high relative accuracy. Next invoke Algorithm Getvec on LDL^t to compute eigenvectors for all the eigenvalues λ of LDL^t that have large relative gaps. For each cluster of relatively close eigenvalues, pick a new base τ_c at, or close to, one end of the cluster. Perform a careful factorization $L_cD_cL_c^t = LDL^t - \tau_c I$ to get a new RRR. Shifting by τ_c increases the relative separations of eigenvalues in the cluster. Refine, to high relative accuracy, the shifted eigenvalues (of $L_cD_cL_c^t$) that now have relative gaps exceeding tol, and invoke Algorithm Getvec on the new factorization $L_cD_cL_c^t$ to compute their eigenvectors. Repeat the process with suitable bases τ until all eigenvectors have been computed. For more details on Algorithm MR^3 the reader is referred to [10], which also addresses the tricky aspect of showing that the eigenvectors computed using the various LDL^t factorizations are numerically orthogonal.

We now give a brief outline of the paper. Section 2 elaborates on the difficulties in achieving our goals, and section 3 demonstrates the need to use an LDL^t factorization to replace T. We present the proposed algorithm in section 4, while the "mixed" roundoff error analyses and associated commutative diagrams are given in section 5. To prove the correctness of the proposed algorithm, the various parts of a commutative diagram are analyzed in detail in section 6, and the main theorem is proved in section 7. Numerical results and timing comparisons are given in section 8. Finally, section 9 discusses an extension to the computation of singular vectors.

Householder notation (capital letters for matrices, Greek lowercase for scalars, and lowercase bold roman for vectors) is generally followed. The norm $\|\cdot\|$ will refer to the 2-norm. Sections 3, 4, 5, and 6.2 are derived from [9].

2. Difficulties. The quality of an approximate eigenvector $\boldsymbol{z}$ is measured by its residual. The following basic result, which goes back to Temple in the 1930s, if not earlier, will be needed later. See [28, Chapters 10 and 11] for details and a proof.

THEOREM 1. *Let $A = A^t$ be a real matrix that has a simple eigenvalue λ with normalized eigenvector $\boldsymbol{v}$. For any unit vector $\boldsymbol{z}$ and a scalar μ, closer to λ than to any other eigenvalue,*

$$|\sin\angle(\boldsymbol{v},\boldsymbol{z})| \le \|A\boldsymbol{z} - \boldsymbol{z}\mu\|/\mathrm{gap}(\mu), \tag{1}$$

where $\mathrm{gap}(\mu) = \min\{|\nu - \mu| : \nu \ne \lambda,\ \nu \in \mathit{spectrum}(A)\}$. *In addition, the error in the eigenvalue is bounded by the residual norm, i.e.,*

$$|\mu - \lambda| \ \le\ \|A\boldsymbol{z} - \boldsymbol{z}\mu\|.$$

The sad fact is that a small residual norm does not guarantee an accurate eigenvector when $\mathrm{gap}(\mu)$ is also small. On the other hand, accurate approximations $\boldsymbol{y}$ to $\boldsymbol{u}$ and $\boldsymbol{z}$ to $\boldsymbol{v}$ (where $\boldsymbol{u}$ is an eigenvector orthogonal to $\boldsymbol{v}$), in the strong sense that

$$|\sin\angle(\boldsymbol{u},\boldsymbol{y})| < n\varepsilon \ \text{ and } \ |\sin\angle(\boldsymbol{v},\boldsymbol{z})| < n\varepsilon, \tag{2}$$

where ε is the machine precision, do ensure numerical orthogonality of the *computed* eigenvectors since

$$|\cos\angle(\boldsymbol{y},\boldsymbol{z})| \le |\sin\angle(\boldsymbol{u},\boldsymbol{y})| + |\sin\angle(\boldsymbol{v},\boldsymbol{z})| < 2n\varepsilon.$$

Thus accuracy yields orthogonality. This observation is not as vacuous as it appears. In the QR algorithm the computed eigenvectors are acceptable because they are orthogonal (numerically) and their residuals are small *but* they are not always accurate in the sense of (2). Part of the explanation for this anomaly is that A may not determine some of its eigenpairs to high accuracy. Thus the eigenvector $\boldsymbol{v}$ used above may be highly sensitive as soon as there is uncertainty in the entries of A and so the concept of accuracy goes out of focus. That is why, in the sense of (2), accuracy is not the only way to compute numerically orthogonal eigenvectors; the QR algorithm is a good example.

Let us return to the residual norm. *In general*, the best we can hope for is to produce residuals $\boldsymbol{r} = \boldsymbol{r}(\boldsymbol{z}) = A\boldsymbol{z} - \boldsymbol{z}\mu$ satisfying

$$\|\boldsymbol{r}\| \le n\varepsilon(\lambda_{max} - \lambda_{min}). \tag{3}$$

By (1) and (3), if $\mathrm{gap}(\mu) \ge \mathit{tol}\cdot(\lambda_{max} - \lambda_{min})$, where tol is the gap threshold (say 10^{-3}), then

$$|\sin\angle(\boldsymbol{v},\boldsymbol{z})| \le n\varepsilon/\mathit{tol}$$

and accuracy is assured (throughout the paper we assume that $n^3\varepsilon \le 1$). On the other hand, in the many cases when $\mathrm{gap}(\mu) \ll \mathit{tol}$, the residual norm must be much smaller than the right-hand side of (3) in order to deliver the accuracy of (2).

In general we see no possibility for reducing the residuals without using higher precision arithmetic in parts of the computation. Instead we turn to special matrices and special situations, in particular, to a symmetric tridiagonal matrix T. Our goal is to compute residuals satisfying

$$\|\boldsymbol{r}\| = \|T\boldsymbol{z} - \boldsymbol{z}\hat{\lambda}\| \le Kn\varepsilon|\hat{\lambda}| \tag{4}$$

for some modest constant K independent of $\boldsymbol{z}$ and $\hat{\lambda}$, so that, by Theorem 1,

$$|\sin\angle(\boldsymbol{v},\boldsymbol{z})| \le \frac{Kn\varepsilon|\hat{\lambda}|}{\mathrm{gap}(\hat{\lambda})} = \frac{Kn\varepsilon}{\mathrm{relgap}(\hat{\lambda})},$$

where relgap$(\hat{\lambda}) := \text{gap}(\hat{\lambda})/|\hat{\lambda}|$. Note that if $\hat{\lambda} = O(\varepsilon(\lambda_{max} - \lambda_{min}))$, then (4) requires $\|\boldsymbol{r}\| = O(\varepsilon^2(\lambda_{max} - \lambda_{min}))$. How is that possible since even the rounded version of the "true" eigenvector may not achieve (4)?

To achieve (4) we need three separate properties:

I. The eigenpair $(\lambda, \boldsymbol{v})$ must be determined to high relative accuracy by the matrix parameters. See the next paragraph for definitions.

II. The computed $\hat{\lambda}$ must approximate λ to high relative accuracy, i.e., $|\lambda - \hat{\lambda}| = O(n\varepsilon|\hat{\lambda}|)$.

III. The vector $\boldsymbol{z}$ must then be computed to satisfy (4).

To achieve property I we discard the traditional representation of T in favor of a suitable LDL^t factorization of T or some translate $T - \tau I$. Write l_i for $L(i+1, i)$ and d_i for $D(i, i)$. We say that $(\lambda, \boldsymbol{v})$ is *determined to high relative accuracy* by L and D if small relative changes, $l_i \to l_i(1 + \eta_i)$, $d_i \to d_i(1 + \delta_i)$, $|\eta_i| < \xi$, $|\delta_i| < \xi$, cause changes $\delta\lambda$ and $\delta\boldsymbol{v}$ that satisfy

$$\frac{|\delta\lambda|}{|\lambda|} \leq K_1 n\xi, \quad \lambda \neq 0, \tag{5}$$

$$|\sin\angle(\boldsymbol{v}, \boldsymbol{v} + \delta\boldsymbol{v})| \leq \frac{K_2 n\xi}{\text{relgap}(\lambda)} \tag{6}$$

for modest constants K_1 and K_2, say, smaller than 100. We call such an LDL^t factorization an RRR for $(\lambda, \boldsymbol{v})$. The backward stable QR algorithm on T cannot guarantee such accuracy.

Section 3 shows the necessity for the change of representation to LDL^t. Property II is then easily achieved by a bisection algorithm that uses differential qd transforms (see section 4.1) or, in the positive definite case, by the dqds algorithm; see [13]. Given properties I and II, we can think of satisfying property III by using inverse iteration. While traditional inverse iteration often works well in practice, we employ an elegant alternative that uses a rank-revealing twisted factorization of $T - \hat{\lambda}I$ to obtain a starting vector that is guaranteed to be good.

A subtle point in our analysis is that (4) is achieved, not for T or LDL^t but for a small relative perturbation of LDL^t.

Much of this paper, from section 4 onwards, is devoted to the algorithm and a proof to show that property III can be achieved in the presence of roundoff error.

3. Standard tridiagonal form is inadequate. In this section we show that the standard representation of tridiagonals is inadequate for our purpose of computing highly accurate eigenvectors. Recent work has shown that some tridiagonal classes do determine all their eigenvalues to high relative accuracy [8]. However, for many tridiagonals small relative changes in the diagonal and off-diagonal entries can cause huge relative changes in the small eigenvalues.

We now give a carefully constructed example which exhibits this relative instability even when $n = 3$.

Example 1. Consider the tridiagonal

$$T_1 = \begin{bmatrix} 1 - \sqrt{\varepsilon} & \varepsilon^{1/4}\sqrt{1 - 7\varepsilon/4} & 0 \\ \varepsilon^{1/4}\sqrt{1 - 7\varepsilon/4} & \sqrt{\varepsilon} + 7\varepsilon/4 & \varepsilon/4 \\ 0 & \varepsilon/4 & 3\varepsilon/4 \end{bmatrix}$$

and a small relative perturbation to the off-diagonals of T_1,

$$T_1 + \delta T_1 = \begin{bmatrix} 1-\sqrt{\varepsilon} & \varepsilon^{1/4}(1+\varepsilon)\sqrt{1-7\varepsilon/4} & 0 \\ \varepsilon^{1/4}(1+\varepsilon)\sqrt{1-7\varepsilon/4} & \sqrt{\varepsilon}+7\varepsilon/4 & \varepsilon(1+\varepsilon)/4 \\ 0 & \varepsilon(1+\varepsilon)/4 & 3\varepsilon/4 \end{bmatrix},$$

where ε is a small quantity of the order of the machine precision. The two smallest eigenvalues of T_1 and $T_1 + \delta T_1$ are[1]

$$\lambda_1 = \varepsilon/2 + \varepsilon^{3/2}/8 + O(\varepsilon^2), \quad \lambda_1 + \delta\lambda_1 = \varepsilon/2 - 7\varepsilon^{3/2}/8 + O(\varepsilon^2),$$

$$\lambda_2 = \varepsilon - \varepsilon^{3/2}/8 + O(\varepsilon^2), \quad \lambda_2 + \delta\lambda_2 = \varepsilon - 9\varepsilon^{3/2}/8 + O(\varepsilon^2),$$

while

$$\lambda_3 = 1 + \varepsilon + O(\varepsilon^2), \qquad \lambda_3 + \delta\lambda_3 = 1 + \varepsilon + 2\varepsilon^{3/2} + O(\varepsilon^2).$$

Thus

$$\left|\frac{\delta\lambda_i}{\lambda_i}\right| = (3-i)\sqrt{\varepsilon} + O(\varepsilon), \quad i = 1, 2,$$

and the relative change in these eigenvalues is much larger than the initial relative perturbations in the entries of T_1. Similarly the corresponding eigenvectors of T_1 and $T_1 + \delta T_1$ are

$$\boldsymbol{v}_1 = \begin{bmatrix} \frac{\varepsilon^{1/4}}{\sqrt{2}}(1+\frac{\sqrt{\varepsilon}}{2}) + O(\varepsilon^{5/4}) \\ -\frac{1}{\sqrt{2}}(1-\frac{\sqrt{\varepsilon}}{2}) + O(\varepsilon) \\ \frac{1}{\sqrt{2}}(1-\frac{3\varepsilon}{4}) + O(\varepsilon^{3/2}) \end{bmatrix}, \quad \boldsymbol{v}_1 + \delta\boldsymbol{v}_1 = \begin{bmatrix} \frac{\varepsilon^{1/4}}{\sqrt{2}}(1+\frac{5\sqrt{\varepsilon}}{2}) + O(\varepsilon^{5/4}) \\ -\frac{1}{\sqrt{2}}(1+\frac{3\sqrt{\varepsilon}}{2}) + O(\varepsilon) \\ \frac{1}{\sqrt{2}}(1-2\sqrt{\varepsilon}) + O(\varepsilon) \end{bmatrix}$$

and

$$\boldsymbol{v}_2 = \begin{bmatrix} -\frac{\varepsilon^{1/4}}{\sqrt{2}}(1+\frac{\sqrt{\varepsilon}}{2}) + O(\varepsilon^{5/4}) \\ \frac{1}{\sqrt{2}}(1-\frac{\sqrt{\varepsilon}}{2}) + O(\varepsilon) \\ \frac{1}{\sqrt{2}}(1+\frac{3\varepsilon}{4}) + O(\varepsilon^{3/2}) \end{bmatrix}, \quad \boldsymbol{v}_2 + \delta\boldsymbol{v}_2 = \begin{bmatrix} -\frac{\varepsilon^{1/4}}{\sqrt{2}}(1-\frac{3\sqrt{\varepsilon}}{2}) + O(\varepsilon^{5/4}) \\ \frac{1}{\sqrt{2}}(1-\frac{5\sqrt{\varepsilon}}{2}) + O(\varepsilon) \\ \frac{1}{\sqrt{2}}(1+2\sqrt{\varepsilon}) + O(\varepsilon) \end{bmatrix},$$

whereby

$$\left|\frac{\delta v_i(j)}{v_i(j)}\right| = O(\sqrt{\varepsilon}) \text{ for } i = 1, 2 \text{ and } j = 1, 2, 3.$$

Since a small relative change of ε in the off-diagonal entries of T_1 results in a much larger relative change in its eigenvalues and eigenvectors, we say that T_1 does not determine its eigenvalues and eigenvectors to high relative accuracy. Consequently, in the face of roundoff errors, it is unlikely that we can compute numerically orthogonal eigenvectors without explicit orthogonalization. To corroborate this, we gave the best possible approximations to λ_1 and λ_2 as input to the EISPACK and LAPACK implementations of inverse iteration but turned off all orthogonalization within these

[1] We carefully constructed this matrix to have the desired behavior, which may be verified by using a symbol manipulator such as Maple [5] or Mathematica [37].

procedures. As expected, we found the computed vectors to have dot products as large as $O(\sqrt{\varepsilon})$. □

In contrast, when T is positive definite, the representations LDL^t and $\tilde{L}\tilde{L}^t$, where $\tilde{L} = LD^{1/2}$, each determine *all* the eigenpairs to high relative accuracy. See [8, Theorem 5.13] for more details. Thus these factored forms are preferable to the standard form for eigenvalue/eigenvector calculations.

When D is not positive definite the situation is more complicated. Extensive testing shows that even in the face of element growth, LDL^t determines its small eigenpairs to high relative accuracy; see Example 2 in section 6.1. Of course we may also use the representation UDU^t derived from Gaussian elimination in reverse order or even a twisted factorization (see section 4). The important point is that the positive definite case is not the only one in which some eigenpairs are determined to high relative accuracy by a factored form.

Let $LDL^t\boldsymbol{v} = \boldsymbol{v}\lambda$, $\lambda \neq 0$. A relative condition number defined in [9] is

$$\boldsymbol{\kappa}_{rel}(\lambda) := \boldsymbol{v}^t L|D|L^t\boldsymbol{v}/|\lambda|.$$

In section 6.1 the true relative condition number covering all the relative perturbations in L and D is shown to be $1 + \boldsymbol{\kappa}_{rel}(\lambda)$. Note that when D is positive definite, then $\boldsymbol{\kappa}_{rel}(\lambda) = 1$, but we do not need such strong stability for our results. A value of $\boldsymbol{\kappa}_{rel}(\lambda)$ such as 10 or 100 is adequate.

The focus of this paper is on how to exploit high relative accuracy when it occurs, not to give conditions for its occurrence. See section 6.1 for some discussion on the latter.

4. Algorithm Getvec. In this section we present our procedure, Algorithm Getvec, for computing an eigenvector.

If $\hat{\lambda}$ is an accurate approximation to an eigenvalue λ of T, then $T - \hat{\lambda}I$ is almost singular. In order to compute the eigenvector, i.e., to solve $(T - \hat{\lambda}I)\boldsymbol{z} \approx 0$, we seek a factorization that reveals this singularity. As we show below, in the tridiagonal case we can easily construct such a "twisted" factorization from the forward and backward triangular factors [31].

Suppose that

$$LDL^t - \hat{\lambda}I = L_+D_+L_+^t = U_-D_-U_-^t,$$

where L_+ is unit lower bidiagonal and U_- is unit upper bidiagonal. The $L_+D_+L_+^t$ and $U_-D_-U_-^t$ factorizations may be obtained by Gaussian elimination in the forward and backward directions, respectively. Note that by the discussion in section 3, we have replaced T by LDL^t. It may happen that neither D_+ nor D_- reveals the singularity of $LDL^t - \hat{\lambda}I$. A twisted factorization, written as

$$LDL^t - \hat{\lambda}I = N_kD_kN_k^t,$$

may be constructed by factoring the matrix from top down and from bottom up meeting at row k. The twisted factor N_k takes rows $1:k$ of L_+ and rows $k:n$ of U_-. Thus all rows of N_k have two nonzeros, except row k, which has three nonzero entries

$$(L_+(k-1) \quad 1 \quad U_-(k)),$$

while D_k is diagonal,

$$D_k = \mathrm{diag}(D_+(1), \ldots, D_+(k-1), \gamma_k, D_-(k+1), \ldots, D_-(n)),$$

where we denote $L_+(i+1,i)$ by $L_+(i)$, $U_-(i,i+1)$ by $U_-(i)$, and the ith diagonal entries of D_+ and D_- by $D_+(i)$ and $D_-(i)$, respectively. We will continue to use this notation for the rest of the paper.

Clearly, there are n twisted factorizations, one for each $k = 1, \ldots, n$. One such twisted factor, with $n = 6$ and $k = 3$, is shown in Figure 1.

$$\begin{bmatrix} \mathrm{x} & & & & & \\ \mathrm{x} & \mathrm{x} & & & & \\ & \mathrm{x} & \mathrm{x} & \mathrm{x} & & \\ & & & \mathrm{x} & \mathrm{x} & \\ & & & & \mathrm{x} & \mathrm{x} \\ & & & & & \mathrm{x} \end{bmatrix}$$

FIG. 1. *Twisted triangular factor N_k with $n = 6$, $k = 3$.*

The only new entry is γ_k, the kth diagonal element of D_k, and it is of great importance. As we show in section 6.2, the nearness to singularity of $LDL^t - \hat{\lambda}I$ is revealed by a small value of $|\gamma_k|$ for an appropriate choice of k. There are several formulae for γ_k, for example,

$$\gamma_k = \begin{cases} D_+(k) + D_-(k) - (d_{k-1}l_{k-1}^2 + d_k - \hat{\lambda}), \\ D_+(k) - (d_k l_k)^2 / D_-(k+1), \end{cases} \tag{7}$$

where $d_k = D(k,k)$, $l_{k-1} = L(k,k-1)$, and so $(LDL^t)_{kk} = d_{k-1}l_{k-1}^2 + d_k$ and $(LDL^t)_{k,k+1} = d_k l_k$.

Naive ways of computing twisted factorizations will not satisfy our demands of high relative accuracy. The so-called "differential qd transforms" allow accurate computation of twisted factorizations, including more robust expressions for γ_k. We will, however, wait until section 4.1 to give details of the qd transforms. Without further ado, we present Algorithm Getvec, which computes an approximate eigenvector by first forming the appropriate twisted factorization. In the following we assume that LDL^t is an irreducible tridiagonal, i.e., $d_i l_i \neq 0$ for $1 \leq i \leq n-1$.

ALGORITHM Getvec(LDL^t, $\hat{\lambda}$).

I. Factor $LDL^t - \hat{\lambda}I = L_+D_+L_+^t$ by the *dstqds* transform (Algorithm 4.2 in section 4.1).

II. Factor $LDL^t - \hat{\lambda}I = U_-D_-U_-^t$ by the *dqds* transform (Algorithm 4.4 in section 4.1).

III. Compute γ_k for $k = 1, \ldots, n$ (by the top formula of (18)). Pick an r such that $|\gamma_r| = \min_k |\gamma_k|$. Then

$$N_r D_r N_r^t = LDL^t - \hat{\lambda}I$$

is the desired twisted factorization (see Algorithm 4.5 in section 4.1).

IV. Form the approximate eigenvector $\boldsymbol{z}$ by solving $N_r^t \boldsymbol{z} = \boldsymbol{e}_r$, where $\boldsymbol{e}_r$ is the rth column of the identity matrix, which is equivalent to solving

$$(LDL^t - \hat{\lambda}I)\boldsymbol{z} = N_r D_r N_r^t \boldsymbol{z} = \boldsymbol{e}_r \gamma_r \quad (\text{since } D_r \boldsymbol{e}_r = \boldsymbol{e}_r \gamma_r \text{ and } N_r \boldsymbol{e}_r = \boldsymbol{e}_r)$$

via

$$z(r) = 1,$$

$$\text{for } i = r-1, \ldots, 1, \quad z(i) = \begin{cases} -L_+(i)z(i+1), & D_+(i) \neq 0, \\ -(d_{i+1}l_{i+1}/d_i l_i)z(i+2), & \text{otherwise,} \end{cases}$$

$$\text{for } j = r, \ldots, n-1, \quad z(j+1) = \begin{cases} -U_-(j)z(j), & D_-(j+1) \neq 0, \\ -(d_{j-1}l_{j-1}/d_j l_j)z(j-1), & \text{otherwise.} \end{cases}$$

V. If wanted, compute $znrm = \|\boldsymbol{z}\|$ and set $\tilde{\boldsymbol{z}} = \boldsymbol{z}/znrm$.

Remark 1. Steps I–III above employ differential qd transforms that are essential in order to exploit the RRR properties of the bidiagonal representation LDL^t. The choice of r in step III ensures that $|\gamma_r| \leq 2n|\hat{\lambda} - \lambda|$ and the residual norm $\|(LDL^t - \hat{\lambda}I)\boldsymbol{z}\|/\|\boldsymbol{z}\| \leq \sqrt{n}|\hat{\lambda} - \lambda|$ under suitable conditions; see Theorems 10 and 11 in section 6.2.

Remark 2. No pivoting is done in steps I and II since the computation assumes IEEE arithmetic [1]. If some $D_+(i)$ (or $D_-(i)$) equals zero, then infinity is produced at the next step, and the computation of $\boldsymbol{z}$ in step IV handles this special case. See Remark 3 below.

Remark 3. Let us explain the special handling in step IV above of the case of a zero entry in D_+ or D_-. In exact arithmetic, when $\hat{\lambda}$ is an eigenvalue, zero entries in D_+ and D_- can occur if and only if the corresponding eigenvector has a zero entry. In particular, when $\hat{\lambda}$ is an eigenvalue, then $D_+(i) = 0$ and $D_-(i+2) = 0$ if and only if $z(i+1) = 0$. See [31] for more details. Thus, when $D_+(i) = 0$, $i < r$, we use the $(i+1)$st equation of the tridiagonal system $(LDL^t - \hat{\lambda}I)\boldsymbol{z} = \boldsymbol{e}_r\gamma_r$ to connect $z(i)$ with $z(i+2)$. The case when $D_-(j+1) = 0$, $j > r$, is handled similarly.

Remark 4. The index r is desired to be such that the rth component of the eigenvector $\boldsymbol{v}$ is largest in magnitude [31]. It is possible to avoid up to half of the $2n$ divisions in steps I and II by observing that $v(i)$ cannot be the largest in magnitude if the eigenvalue is not contained in the ith Gerschgorin disk. This observation enables us to identify the smallest and largest indices that are candidates for the twist index r. The savings are often real when n is large since eigenvectors of large matrices often have negligible entries at either end. See [9, section 3.4.1] for details.

Remark 5. In addition to computing an eigenvector approximation, the above algorithm can also be used to improve the accuracy of $\hat{\lambda}$. By Lemma 12 in section 6.2, $\gamma_r/\|\boldsymbol{z}\|^2$ is the Rayleigh quotient correction to $\hat{\lambda}$ and so it can double the number of correct digits when $\hat{\lambda}$ is not quite acceptable, for example, when $|\hat{\lambda} - \lambda| = O(\sqrt{\varepsilon}|\lambda|)$ where λ is the eigenvalue closest to $\hat{\lambda}$. Indeed, the refinement of eigenvalues in Algorithm MR^3 of [10] is done by switching from bisection to this Rayleigh quotient correction for increased efficiency.

Remark 6. The vector $\boldsymbol{z}$ often has small numerical support (defined below) when n is large. This situation can be detected when consecutive entries in $\boldsymbol{z}$ are small enough in magnitude. Then the remaining entries in $\boldsymbol{z}$ may be set to zero. Suppose all elements $z(j)$, $j < i-1 < r-1$, are set to zero; then equations $i-2$ and $i-1$ of $(LDL^t - \hat{\lambda}I)\boldsymbol{z} = \boldsymbol{e}_r\gamma_r$ are no longer satisfied and result in a residual $\beta_{i-2}(z(i-1)\boldsymbol{e}_{i-2} - z(i-2)\boldsymbol{e}_{i-1})$, where $\beta_{i-2} = D_+(i-2)L_+(i-2)$. For the vector $\boldsymbol{z}$ to be an accurate eigenvector (see Theorem 1), it suffices to ensure that $|z(i-1)|$ and $|z(i-2)|$ are small enough that

$$|D_+(i-2)L_+(i-2)|\,(|z(i-1)| + |z(i-2)|) \;<\; \varepsilon \cdot \mathrm{gap}(\hat{\lambda}),$$

where $z(i-2) = -L_+(i-2)z(i-1)$. Similarly when $i > r$ we set the elements $z(j)$, $j > i$, to 0 if $|z(i)|$ and $|z(i+1)|$ are small enough that

$$|D_-(i)U_-(i-1)|\,(|z(i)| + |z(i+1)|) \;<\; \varepsilon \cdot \mathrm{gap}(\hat{\lambda}),$$

where $z(i+1) = -U_-(i)z(i)$. Thus all our computed vectors have a first and last nonzero component and we call the index set {first:last} the *numerical support* of $\boldsymbol{z}$ and so

$$|\mathrm{supp}(\boldsymbol{z})| = \mathrm{last} - \mathrm{first} + 1.$$

Note that in exact arithmetic the first and last entries of an eigenvector of an unreduced tridiagonal matrix are nonzero but in practice they are often extremely small, and so the above situation is not uncommon.

There is more to be said about the support. Before $\boldsymbol{z}$ is computed all the $\{\gamma_i\}$ are computed in order to find the smallest among them. By Lemma 11 in [31], as $\hat{\lambda} \to \lambda$,

$$\frac{\gamma_r}{\gamma_i} \;\to\; \frac{v(i)^2}{v(r)^2}, \tag{8}$$

where $\boldsymbol{v}$ is λ's eigenvector. This suggests that if $|\gamma_i| > |\gamma_r|/\varepsilon^2$, then $z(i)$ may be neglected and it might be argued that this gives us a better way to approximate $\mathrm{supp}(\boldsymbol{z})$ at the time r is chosen. Unfortunately, machine precision is often not sufficient to put $\hat{\lambda}$ close enough to λ for (8) to hold for indices where $|v(i)| \ll \sqrt{\varepsilon}$, and so this strategy does not work in practice.

Remark 7. It is not essential that $|\gamma_r|$ be minimal. In principle one keeps a list of indices i such that $|\gamma_{min}| < |\gamma_i| < 2|\gamma_{min}|$, and can choose r to be any of these indices.

Remark 8. Suppose $\hat{\lambda}$ approximates λ. As will be shown in section 7, in the presence of roundoff errors, the best we can hope for is that the computed vector $\hat{\boldsymbol{z}}$ satisfies

$$|\sin\angle(\hat{\boldsymbol{z}}, \boldsymbol{v})| \;=\; O\left(\frac{|\lambda - \hat{\lambda}|}{\mathrm{gap}(\hat{\lambda})}\right) \;=\; O\left(\frac{n\varepsilon|\hat{\lambda}|}{\mathrm{gap}(\hat{\lambda})}\right) \;=\; O\left(\frac{n\varepsilon}{\mathrm{relgap}(\hat{\lambda})}\right).$$

Such a $\hat{\boldsymbol{z}}$ will be an accurate eigenvector when $\mathrm{relgap}(\hat{\lambda}) \geq tol$. A natural question to ask is: can such an accurate approximation be computed when the relative gap is smaller, say, $\mathrm{relgap}(\hat{\lambda}) = \sqrt{\varepsilon}$? A tempting solution is to extend Algorithm Getvec to do a step of inverse iteration: $(LDL^t - \hat{\lambda}I)\boldsymbol{y} = \hat{\boldsymbol{z}} \;\Rightarrow (LDL^t - \hat{\lambda}I)^2\boldsymbol{y} \approx \gamma_r \boldsymbol{e}_r$. The tempting argument is that by doing so,

$$|\sin\angle(\boldsymbol{y}, \boldsymbol{v})| \;=\; O\left(\frac{|\lambda - \hat{\lambda}|^2}{\mathrm{gap}(\hat{\lambda})^2}\right) \;=\; O\left(\frac{n^2\varepsilon^2}{\mathrm{relgap}(\hat{\lambda})^2}\right),$$

since the eigenvalues of $(LDL^t - \hat{\lambda}I)^2$ are just $(\lambda_i - \hat{\lambda})^2$. When $\mathrm{relgap}(\hat{\lambda}) = \sqrt{\varepsilon}$ and n is modest, this strategy appears to yield an accurate eigenvector $\boldsymbol{y}$.

Unfortunately this simple solution does not work. In our experience, even for small n the extra step of inverse iteration increases the accuracy by a factor of .1 or .01 but not by a factor of $\sqrt{\varepsilon}$ as the above reasoning indicates. The failure is due to the presence of roundoff errors and limitations due to relative perturbation theory.

The case of relgap$(\hat{\lambda}) \ll tol$ requires radically different strategies. One strategy is to take a new shift to improve the relative gaps and then invoke Algorithm Getvec. Small relative gaps are not the concern of this paper, but the interested reader may see [9, 10] for details. Very tight clusters of eigenvalues that are well-separated from the rest of the spectrum may also be handled by the overlapping submatrix ideas of [29].

4.1. Differential qd transforms. This section completes the description of Algorithm Getvec by presenting the differential qd transforms that are needed to compute the $L_+D_+L_+^t$, $U_-D_-U_-^t$ and $N_rD_rN_r^t$ decompositions in steps I–III of the algorithm. Algorithm 4.1 given below is a straightforward implementation of the transformation

$$LDL^t - \mu I \;=\; L_+D_+L_+^t. \tag{9}$$

We call this the "**st**ationary **q**uotient-**d**ifference with **s**hift"(stqds) transform for historical reasons. The term was first coined by Rutishauser for similar transformations that formed the basis of his qd algorithm first developed in 1954 [34, 35, 36]. Although (9) is not identical to the stationary transformation given by Rutishauser, the differences are not significant enough to warrant inventing new terminology. The term "stationary" is used for (9) since it represents an identity transformation when $\mu = 0$. Rutishauser used the term "progressive" instead for the formation of $U_-D_-U_-^t$ from $LDL^t - \mu I$ or of $L_+D_+L_+^t$ from $UDU^t - \mu I$.

ALGORITHM 4.1. **(stqds)**-stationary qd transform.

$D_+(1) \;:=\; d_1 - \mu$
for $i = 1, n-1$
$\quad L_+(i) \;:=\; (d_il_i)/D_+(i)$ (10)
$\quad D_+(i+1) \;:=\; d_il_i^2 + d_{i+1} - L_+(i)d_il_i - \mu$ (11)
end for

This algorithm loses accuracy when there is element growth. Next we show how to eliminate some of the additions and subtractions from Algorithm 4.1. We introduce the intermediate variable $s_i := D_+(i) - d_i$, $1 \le i \le n$. A two-term recurrence between s_i and s_{i+1}, $1 \le i \le n-1$, may be obtained as follows:

$$\begin{aligned} s_{i+1} &= D_+(i+1) - d_{i+1} \\ &= d_il_i^2 - L_+(i)d_il_i - \mu \quad \text{by (11)} \\ &= L_+(i)l_i(D_+(i) - d_i) - \mu \quad \text{by (10)} \\ &= L_+(i)l_is_i - \mu. \end{aligned} \tag{12}$$

Using this intermediate variable, we get the so-called *differential form* of the stationary qd transform (dstqds). This term was again coined by Rutishauser in the appendix of [36].

ALGORITHM 4.2. **(dstqds)**-*differential form* of the stationary qd transform.

$$
\begin{aligned}
& s_1 := -\mu \\
& \text{for } i = 1, n-1 \\
(13) \qquad & \quad D_+(i) := s_i + d_i \\
& \quad L_+(i) := (d_i l_i)/D_+(i) \\
& \quad s_{i+1} := L_+(i) l_i s_i - \mu \\
& \text{end for} \\
& D_+(n) := s_n + d_n
\end{aligned}
$$

In section 5 we will show that the differential transforms, in the face of roundoff errors, have attractive properties which play a crucial role in proving the main result of the paper, Theorem 15.

We also need to compute the transformation

$$LDL^t - \mu I = U_- D_- U_-^t,$$

which we call the "progressive **q**uotient-**d**ifference with **s**hift"(qds) transform. The following algorithm gives an obvious way to implement this transformation.

ALGORITHM 4.3. **(qds)**-progressive qd transform.

$$
\begin{aligned}
& D_-(n) := d_{n-1} l_{n-1}^2 + d_n - \mu \\
& \text{for } i = n-1, 1, -1 \\
(14) \qquad & \quad U_-(i) := (d_i l_i)/D_-(i+1) \\
(15) \qquad & \quad D_-(i) := d_{i-1} l_{i-1}^2 + d_i - (d_i l_i) U_-(i) - \mu \\
& \text{end for}
\end{aligned}
$$

Here we have adopted the convention that $d_0 = l_0 = 0$, which justifies (15) for $i = 1$. As in the stationary transformation, we introduce the intermediate variable $p_i := D_-(i) - d_{i-1} l_{i-1}^2$, $1 \le i \le n$. A two-term recurrence between p_i and p_{i+1}, $1 \le i \le n-1$, may be obtained as follows:

$$
\begin{aligned}
(16) \qquad p_i &= D_-(i) - d_{i-1} l_{i-1}^2 \\
&= d_i - U_-(i) d_i l_i - \mu \quad \text{by (15)} \\
&= \frac{d_i}{D_-(i+1)} (D_-(i+1) - d_i l_i^2) - \mu \quad \text{by (14)} \\
(17) \qquad &= \frac{d_i}{D_-(i+1)} \cdot p_{i+1} - \mu.
\end{aligned}
$$

Using this intermediate variable, we get the *differential form* of the progressive qd transform.

ALGORITHM 4.4. **(dqds)**-*differential form* of the progressive qd transform.

$$
\begin{aligned}
&p_n \;:=\; d_n - \mu \\
&\text{for } i = n-1, 1, -1 \\
&\quad D_-(i+1) \;:=\; d_i l_i^2 + p_{i+1} \\
&\quad t \;:=\; d_i / D_-(i+1) \\
&\quad U_-(i) \;:=\; l_i t \\
&\quad p_i \;:=\; p_{i+1} t - \mu \\
&\text{end for} \\
&D_-(1) \;:=\; p_1
\end{aligned}
$$

Note that we have denoted the intermediate variables by the symbols s_i and p_i to stand for *stationary* and *progressive*, respectively.

We also need to find all the γ_k's in order to choose the appropriate twisted factorization for computing the eigenvector. By (7),

$$
\begin{aligned}
\gamma_k &= D_+(k) - \frac{(d_k l_k)^2}{D_-(k+1)} \\
&= s_k + d_k - \frac{(d_k l_k)^2}{D_-(k+1)} \quad \text{by (13)} \\
&= s_k + \frac{d_k}{D_-(k+1)} \left(D_-(k+1) - d_k l_k^2 \right).
\end{aligned}
$$

Substituting from (16), (17), and (12) in the above equation, we can express γ_k by any of the following formulae:

$$
\gamma_k \;=\; \begin{cases} s_k + \frac{d_k}{D_-(k+1)} \cdot p_{k+1}, \\ s_k + p_k + \mu, \\ p_k + L_+(k-1) l_{k-1} s_{k-1}. \end{cases} \tag{18}
$$

In section 5, we will see that the top and bottom formulae in (18) are "better" in the presence of roundoff. When μ is close to an eigenvalue of LDL^t, the near-singularity of $LDL^t - \mu I$ can be revealed by choosing $r = \text{argmin}_k |\gamma_k|$.The twisted factorization at position r is given by

$$
LDL^t - \mu I \;=\; N_r D_r N_r^t,
$$

where $D_r = \text{diag}(D_+(1), \ldots, D_+(r-1), \gamma_r, D_-(r+1), \ldots, D_-(n))$, and N_r is the corresponding twisted factor that takes rows $1:r$ of L_+ and rows $r:n$ of U_- (see the beginning of section 4). It may be formed by the following "**d**ifferential **tw**isted **q**uotient-**d**ifference with **s**hift"(dtwqds) transform which is just the appropriate blend of Algorithms 4.2 and 4.4.

ALGORITHM 4.5. **(dtwqds)**-differential twisted qd transform.

$s_1 := -\mu$
for $i = 1, r-1$
  $D_+(i) := s_i + d_i$
  $L_+(i) := (d_i l_i)/D_+(i)$
  $s_{i+1} := L_+(i) l_i s_i - \mu$
end for
$p_n := d_n - \mu$
for $i = n-1, r, -1$
  $D_-(i+1) := d_i l_i^2 + p_{i+1}$
  $t := d_i/D_-(i+1)$
  $U_-(i) := l_i t$
  $p_i := p_{i+1} t - \mu$
end for
if $r < n$
  $$\gamma_r := s_r + \frac{d_r}{D_-(r+1)} \cdot p_{r+1}$$
else
  $\gamma_r := s_n + d_n$
end if

Note: In cases where we have already computed the stationary and progressive transformations, i.e., we have computed L_+, D_+, U_-, and D_-, the only additional work needed for dtwqds is one multiplication and one addition to compute γ_r.

We emphasize that *the particular qd transforms presented in this section are new.* Similar qd recurrences have been studied by Rutishauser [34, 35, 36]; Henrici [22], [23, Chapter 7]; Fernando and Parlett [13]; and Yao Yang [38].

4.2. Relation to previous work. Algorithm Getvec presented earlier is close in spirit to the one presented by Godunov and his co-workers in the USSR in 1985; see [18] and [17]. They formulated the idea of taking the top entries in the vector from one sequence and the bottom entries from another one and then choosing the right index at which to join the two pieces. Independently, Fernando discovered a similar idea in terms of running the well-known two-term recurrence for D_+, both forward from $D_+(1)$ and backward from $D_+(n) = 0$, and then joining the two sequences where they are closest. In [31], Parlett and Dhillon formulated and proved the double factorization theorem that gave a formula for computing γ_k, and showed the relationship of γ_k to the diagonal of the inverse. Further, [31] showed that at least one twisted factorization must reveal the size of the smallest eigenvalue thus yielding an accurate eigenvector (see Theorem 11 in section 6.2).

However, neither Godunov nor Fernando reaps the full reward for choosing the best place to join two pieces.

The reasons are quite different in the two cases. Godunov et al. carefully select approximate eigenvalues on opposite sides of the true eigenvalue for the two sequences that provide the eigenvector entries. However, they need *directed* rounding in order to establish their bounds in finite precision arithmetic. Directed rounding is available

in most modern computer hardware since it is part of the IEEE floating point standard [1]; however, the only programming language that makes it available in 2002 is C99. It is not yet implemented in Fortran 2000. Fernando does not consider the effects of roundoff error but, as with Godunov et. al., computes the two factorizations from a translate of the original matrix T that may not define its eigenvalues to high relative accuracy. The 3×3 example in section 3 illustrates the problem: the algorithm given by Fernando in section 5 of [14], even with highly accurate eigenvalue approximations, can yield eigenvectors with error exceeding $\sqrt{\varepsilon}$.

Thus we use the LDL^t representation instead of the diagonal and off-diagonal elements of T. Even use of a good representation is not enough to ensure that the residual norm $\|(LDL^t - \hat{\lambda}I)\boldsymbol{z}\| = O(\varepsilon|\lambda - \hat{\lambda}|)$ for the computed $\boldsymbol{z}$. For example, if Rutishauser's stationary qd algorithm (stqds) were used to compute L_+ and D_+ satisfying $LDL^t - \hat{\lambda}I = L_+D_+L_+^t$ we could not prove our main result, Theorem 15 in section 7. That result requires a second innovation, beyond the use of LDL^t, namely use of the *differential* qd algorithms introduced in section 4.1 to compute the entries of the twisted factors. The crucial relative mixed error analyses as will be shown by the commutative diagrams in section 5 are not valid for Rutishauser's implementation. Hence the LDL^t representation and differential qd transforms are both crucial to our goal of computing orthogonal eigenvectors when relative gaps are large.

5. Roundoff error analysis. In this section, we exhibit desirable properties of the differential qd transforms of section 4.1 in the face of roundoff errors. The error analysis that follows is somewhat daunting and a trustful reader may wish to skip the proofs. However, the very special "interpretation" of the roundoff errors is the rock on which our main result, Theorem 15, is built.

First, we introduce our model of arithmetic. We assume that the floating point result of a basic arithmetic operation $\circ$ satisfies

$$fl(x \circ y) = (x \circ y)(1 + \eta) = (x \circ y)/(1 + \delta),$$

where η and δ depend on x, y, $\circ$, and the arithmetic unit but satisfy

$$|\eta| < \varepsilon, \quad |\delta| < \varepsilon$$

for a given ε that depends only on the arithmetic unit. We shall choose freely the form (η or δ) that suits the analysis. As usual, we will ignore $O(\varepsilon^2)$ terms in our analyses. We also adopt the convention of denoting the computed value of x by $\hat{x}$.

Ideally, we would like to show that the differential qd transforms introduced in section 4.1 produce an output that is exact for data that is very close to the input matrix. Since we desire relative accuracy, we would like this backward error to be relative. However, our algorithms do not admit such a pure backward analysis (see [38] for a backward analysis where the backward errors are absolute but not relative). Nevertheless, we will give a hybrid interpretation involving both backward and forward relative errors.

The best way to understand our first result is by studying Figure 2. Following Rutishauser, we merge elements of L and D into a single array,

$$Z := \{d_1, l_1, d_2, l_2, \ldots, d_{n-1}, l_{n-1}, d_n\}.$$

Likewise, the array $\vec{Z}$ is made up of elements $\vec{d_i}$ and $\vec{l_i}$; $\hat{Z}_+$ contains elements $\hat{D}_+(i)$, $\hat{L}_+(i)$, and so on. The acronym *ulp* in Figure 2 stands for *u*nits in the *l*ast *p*lace held. It is the natural way to refer to *relative* differences between numbers. When a result is correctly rounded the error is not more than half an *ulp*.

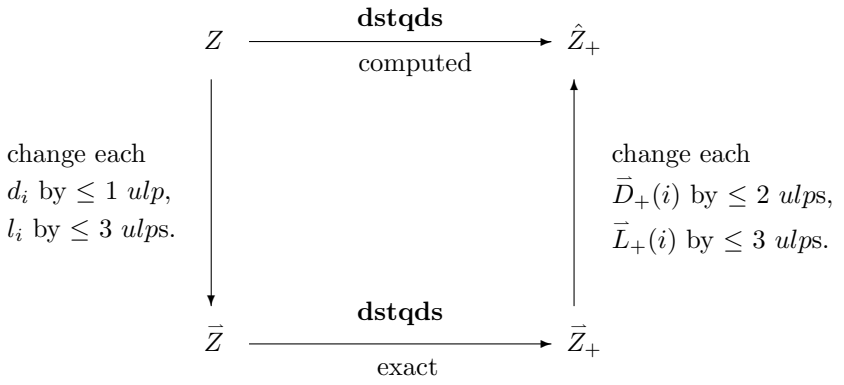

FIG. 2. *Effects of roundoff*—**dstqds** *transform.*

Notational Guide. In all results of this section, numbers in the computer are represented either by letters without any overbar, such as Z, or by "hatted" symbols, such as $\hat{Z}_+$. For example, in Figure 2, Z represents the input data, while $\hat{Z}_+$ represents the output data obtained by executing the dstqds algorithm in finite precision. Intermediate arrays, such as $\vec{Z}$ and $\vec{Z}_+$, are introduced for our analysis but are typically unrepresentable in a computer's limited precision. Note that we have chosen the symbol $\rightharpoonup$ in Figure 2 to indicate a process that takes rows and columns of a tridiagonal in increasing order, i.e., from "left to right." Later, in Figure 3 we use $\leftharpoonup$ to indicate a "right to left" process.

Figure 2 states that the computed outputs of the dstqds transform (Algorithm 4.2), $\hat{D}_+(i)$ and $\hat{L}_+(i)$, are small relative perturbations of the quantities $\vec{D}_+(i)$ and $\vec{L}_+(i)$ which in turn are the results of an EXACT dstqds transform applied to the perturbed matrix represented by $\vec{Z}$. The elements of $\vec{Z}$ are obtained by small relative changes in the inputs L and D. Analogous results hold for the dqds and dtwqds transforms (Algorithms 4.4 and 4.5). As we mentioned above, this is not a pure backward error analysis. We have put small perturbations not only on the input but also on the output in order to obtain an exact dstqds transform. This property is called mixed stability in [4, 6] and numerical stability in [24] but note that our perturbations are relative, not absolute.

THEOREM 2. *Let the dstqds transform be computed as in Algorithm* 4.2. *In the absence of overflow and underflow, the diagram in Figure* 2 *commutes and* $\vec{d}_i$ $(\vec{l}_i)$ *differs from* d_i (l_i) *by at most* 1 (3) *ulps, while* $\hat{D}_+(i)$ $(\hat{L}_+(i))$ *differs from* $\vec{D}_+(i)$ $(\vec{L}_+(i))$ *by at most* 2 (3) *ulps.*

Proof. We write the exact equations satisfied by the computed quantities:

$$\hat{D}_+(i) = (\hat{s}_i + d_i)/(1+\varepsilon_+),$$

$$\hat{L}_+(i) = d_i\, l_i(1+\varepsilon_*)(1+\varepsilon_/)/\hat{D}_+(i) \;=\; \frac{d_i\, l_i(1+\varepsilon_*)(1+\varepsilon_/)(1+\varepsilon_+)}{\hat{s}_i + d_i},$$

$$\text{and} \quad \hat{s}_{i+1} = \frac{\hat{L}_+(i)\, l_i \hat{s}_i(1+\varepsilon_\circ)(1+\varepsilon_{**}) - \mu}{1+\varepsilon_{i+1}}.$$

In the above, all ε's depend on i but we have chosen to single out the one that accounts

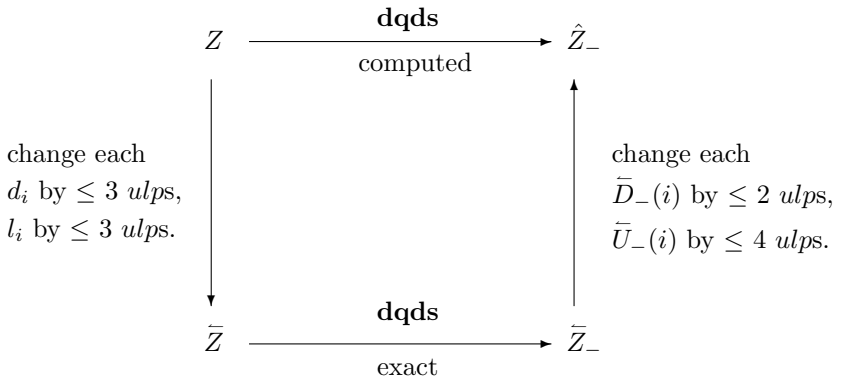

FIG. 3. *Effects of roundoff*—**dqds** *transform.*

for the subtraction as it is the only one where the dependence on i must be made explicit. In more detail the last relation is

$$(1+\varepsilon_{i+1})\hat{s}_{i+1} = \frac{d_i\, l_i^2\, \hat{s}_i}{\hat{s}_i + d_i}\,(1+\varepsilon_*)(1+\varepsilon_/)(1+\varepsilon_+)(1+\varepsilon_\circ)(1+\varepsilon_{**}) - \mu.$$

The trick is to define $\bar{d}_i$ and $\bar{l}_i$ so that the exact dstqds relation

$$\bar{s}_{i+1} = \frac{\bar{d}_i\, \bar{l}_i^2\, \bar{s}_i}{\bar{s}_i + \bar{d}_i} - \mu \tag{19}$$

is satisfied. This may be achieved by setting

$$\begin{aligned} \bar{d}_i &= d_i(1+\varepsilon_i), \\ \bar{s}_i &= \hat{s}_i(1+\varepsilon_i), \\ \bar{l}_i &= l_i \sqrt{\frac{(1+\varepsilon_*)(1+\varepsilon_/)(1+\varepsilon_+)(1+\varepsilon_\circ)(1+\varepsilon_{**})}{1+\varepsilon_i}}. \end{aligned} \tag{20}$$

In order to satisfy the exact mathematical relations of dstqds,

$$\bar{D}_+(i) = \bar{s}_i + \bar{d}_i, \tag{21}$$

$$\bar{L}_+(i) = \frac{\bar{d}_i\, \bar{l}_i}{\bar{s}_i + \bar{d}_i}, \tag{22}$$

we set

$$\begin{aligned} \bar{D}_+(i) &= \hat{D}_+(i)(1+\varepsilon_+)(1+\varepsilon_i), \\ \bar{L}_+(i) &= \hat{L}_+(i)\sqrt{\frac{(1+\varepsilon_\circ)(1+\varepsilon_{**})}{(1+\varepsilon_*)(1+\varepsilon_/)(1+\varepsilon_+)(1+\varepsilon_i)}}, \end{aligned} \tag{23}$$

and the result holds. □

A similar result holds for the dqds transform.

THEOREM 3. *Let the dqds transform be computed as in Algorithm* 4.4. *In the absence of overflow and underflow, the diagram in Figure* 3 *commutes and* $\bar{d}_i$ $(\bar{l}_i)$

differs from d_i (l_i) *by at most* 3 (3) *ulps, while* $\hat{D}_-(i)$ ($\hat{U}_-(i)$) *differs from* $\bar{\bar{D}}_-(i)$ ($\bar{\bar{U}}_-(i)$) *by at most* 2 (4) *ulps.*

Proof. The proof is similar to that of Theorem 2. The computed quantities satisfy

$$\begin{aligned}
\hat{D}_-(i+1) &= (d_i\, l_i^2(1+\varepsilon_*)(1+\varepsilon_{**}) + \hat{p}_{i+1})/(1+\varepsilon_+), \\
\hat{t} &= d_i(1+\varepsilon_/)/\hat{D}_-(i+1), \\
\hat{U}_-(i) &= l_i\hat{t}(1+\varepsilon_\circ) \;=\; \frac{d_i\, l_i(1+\varepsilon_/)(1+\varepsilon_\circ)(1+\varepsilon_+)}{d_i\, l_i^2(1+\varepsilon_*)(1+\varepsilon_{**}) + \hat{p}_{i+1}}, \\
\hat{p}_i &= \frac{(d_i/\hat{D}_-(i+1))\hat{p}_{i+1}(1+\varepsilon_/)(1+\varepsilon_{\circ\circ}) - \mu}{1+\varepsilon_i}, \\
\Rightarrow \quad (1+\varepsilon_i)\hat{p}_i &= \frac{d_i\,\hat{p}_{i+1}}{d_i\, l_i^2(1+\varepsilon_*)(1+\varepsilon_{**}) + \hat{p}_{i+1}}(1+\varepsilon_/)(1+\varepsilon_{\circ\circ})(1+\varepsilon_+) - \mu.
\end{aligned} \tag{24}$$

Note that the above ε's are different from the ones in the proof of the earlier Theorem 2. As in Theorem 2, the trick is to satisfy the exact relation,

$$\bar{\bar{p}}_i = \frac{\bar{\bar{d}}_i\,\bar{\bar{p}}_{i+1}}{\bar{\bar{d}}_i\,\bar{\bar{l}}_i^2 + \bar{\bar{p}}_{i+1}} \;-\; \mu, \tag{25}$$

which is achieved by setting

$$\bar{\bar{d}}_i = d_i(1+\varepsilon_/)(1+\varepsilon_{\circ\circ})(1+\varepsilon_+), \tag{26}$$

$$\bar{\bar{p}}_i = \hat{p}_i(1+\varepsilon_i),$$

$$\text{and} \quad \bar{\bar{l}}_i = l_i\sqrt{\frac{(1+\varepsilon_*)(1+\varepsilon_{**})(1+\varepsilon_{i+1})}{(1+\varepsilon_/)(1+\varepsilon_{\circ\circ})(1+\varepsilon_+)}}, \tag{27}$$

$$\text{so that} \quad \bar{\bar{d}}_i\,\bar{\bar{l}}_i^2 = d_i\, l_i^2(1+\varepsilon_*)(1+\varepsilon_{**})(1+\varepsilon_{i+1}).$$

The other dqds relations,

$$\bar{\bar{D}}_-(i+1) = \bar{\bar{d}}_i\,\bar{\bar{l}}_i^2 \;+\; \bar{\bar{p}}_{i+1}, \tag{28}$$

$$\bar{\bar{U}}_-(i) = \frac{\bar{\bar{d}}_i\,\bar{\bar{l}}_i}{\bar{\bar{d}}_i\,\bar{\bar{l}}_i^2 + \bar{\bar{p}}_{i+1}}, \tag{29}$$

may be satisfied by setting

$$\begin{aligned}
\bar{\bar{D}}_-(i+1) &= \hat{D}_-(i+1)(1+\varepsilon_+)(1+\varepsilon_{i+1}), \\
\bar{\bar{U}}_-(i) &= \frac{\hat{U}_-(i)}{1+\varepsilon_\circ}\sqrt{\frac{(1+\varepsilon_*)(1+\varepsilon_{**})(1+\varepsilon_{\circ\circ})}{(1+\varepsilon_/)(1+\varepsilon_+)(1+\varepsilon_{i+1})}}.
\end{aligned} \tag{30}$$

∎

By combining parts of the analyses for the dstqds and dqds transforms, we can also exhibit a similar result for the twisted factorization computed by Algorithm 4.5. In Figure 4, the various Z arrays represent corresponding twisted factors that may be obtained by "concatenating" the stationary and progressive factors. In particular, for any twist position k,

$$\begin{aligned}
\hat{Z}_k &:= \{\hat{D}_+(1), \hat{L}_+(1), \ldots, \hat{L}_+(k-1), \hat{\gamma}_k, \hat{U}_-(k), \hat{D}_-(k+1), \ldots, \hat{U}_-(n-1), \hat{D}_-(n)\}, \\
\bar{Z}_k &:= \{\vec{D}_+(1), \vec{L}_+(1), \ldots, \vec{L}_+(k-1), \bar{\gamma}_k, \bar{U}_-(k), \bar{\bar{D}}_-(k+1), \ldots, \bar{\bar{U}}_-(n-1), \bar{\bar{D}}_-(n)\},
\end{aligned}$$

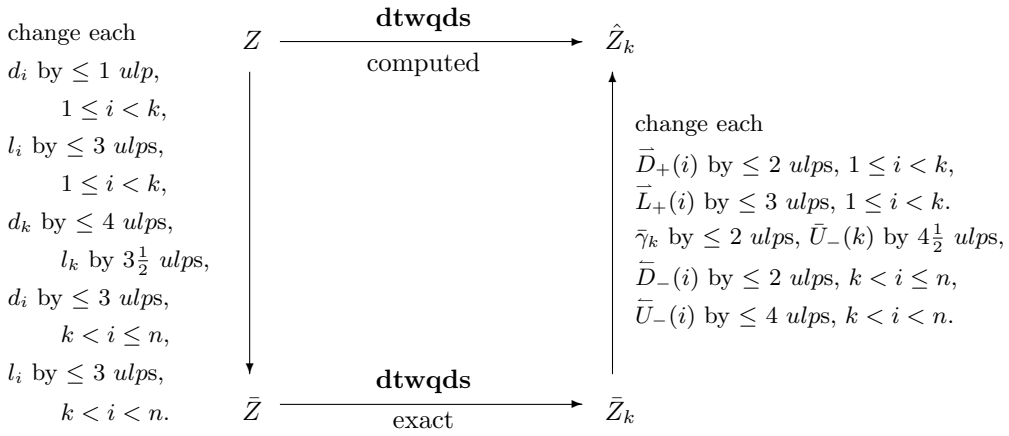

FIG. 4. *Effects of roundoff*—**dtwqds** *transform.*

while

$$\bar{Z} := \{\vec{d}_1, \vec{l}_1, \ldots, \vec{l}_{k-1}, \bar{d}_k, \bar{l}_k, \overleftarrow{d}_{k+1}, \ldots, \overleftarrow{l}_{n-1}, \overleftarrow{d}_n\}.$$

$\hat{Z}_k$ and $\bar{Z}_k$ represent the twisted factorizations

$$\hat{N}_k \hat{D}_k \hat{N}_k^t \qquad \text{and} \qquad \bar{N}_k \bar{D}_k \bar{N}_k^t,$$

respectively.

THEOREM 4. *Let the dtwqds transform be computed as in Algorithm* 4.5. *In the absence of overflow and underflow, the diagram in Figure* 4 *commutes.*

Proof. The crucial observation is that for the exact stationary transform (i.e., (19), (21), and (22)) to be satisfied for $1 \leq i \leq k-1$, roundoff errors need to be put only on $d_1, d_2, \ldots, d_{k-1}$ and $l_1, l_2, \ldots, l_{k-1}$. Similarly for the progressive transform (i.e., (25), (28) and (29)) to hold for $k+1 \leq i < n$, roundoff errors need to be put only on the bottom part of the matrix, i.e., on $d_{k+1}, \ldots, d_n$ and $l_{k+1}, \ldots, l_{n-1}$.

Next we turn to the entries associated with the twist position k. By the top formula in (18),

$$\hat{\gamma}_k = \left(\hat{s}_k + \frac{d_k}{\hat{D}_-(k+1)} \hat{p}_{k+1}(1+\varepsilon_{/}^-)(1+\varepsilon_{\circ\circ}^-) \right) \Big/ (1+\varepsilon_k).$$

Note that in the above, we have put the superscript "$-$" on some ε's to indicate that they are identical to the corresponding ε's in the proof of Theorem 3. By (20) and (24),

$$(1+\varepsilon_k)\hat{\gamma}_k = \frac{\vec{s}_k}{1+\varepsilon_k^+} + \frac{\hat{p}_{k+1} \cdot d_k (1+\varepsilon_{/}^-)(1+\varepsilon_{\circ\circ}^-)(1+\varepsilon_{+}^-)}{d_k\, l_k^2 (1+\varepsilon_{*}^-)(1+\varepsilon_{**}^-) + \hat{p}_{k+1}}$$

$$\Rightarrow \quad (1+\varepsilon_k)(1+\varepsilon_k^+)\hat{\gamma}_k = \vec{s}_k + \frac{\hat{p}_{k+1}(1+\varepsilon_{k+1}^-) \cdot d_k (1+\varepsilon_{/}^-)(1+\varepsilon_{\circ\circ}^-)(1+\varepsilon_{+}^-)(1+\varepsilon_k^+)}{d_k\, l_k^2 (1+\varepsilon_{*}^-)(1+\varepsilon_{**}^-)(1+\varepsilon_{k+1}^-) + \hat{p}_{k+1}(1+\varepsilon_{k+1}^-)},$$

where the superscript "+" indicates that the corresponding ε's are identical to those in the proof of Theorem 2. Note that we are free to attribute roundoff errors to d_k and l_k in order to preserve exact mathematical relations at the twist position k. In particular, by setting

$$\begin{aligned}
\bar{\gamma}_k &= \hat{\gamma}_k(1+\varepsilon_k)(1+\varepsilon_k^+),\\
\bar{d}_k &= d_k(1+\varepsilon_{/}^-)(1+\varepsilon_{\circ\circ}^-)(1+\varepsilon_{+}^-)(1+\varepsilon_k^+),\\
\bar{l}_k &= l_k\sqrt{\frac{(1+\varepsilon_{*}^-)(1+\varepsilon_{**}^-)(1+\varepsilon_{k+1}^-)}{(1+\varepsilon_{/}^-)(1+\varepsilon_{\circ\circ}^-)(1+\varepsilon_{+}^-)(1+\varepsilon_k^+)}}
\end{aligned}$$

and recalling that $\bar{\bar{p}}_{k+1} = \hat{p}_{k+1}(1+\varepsilon_{k+1}^-)$ (see (26)), the following exact relation holds:

$$\bar{\gamma}_k = \bar{\bar{s}}_k + \frac{\bar{d}_k\,\bar{\bar{p}}_{k+1}}{\bar{\bar{d}}_k\,\bar{l}_k^2 + \bar{\bar{p}}_{k+1}}.$$

In addition, the exact relation

$$\bar{U}_-(k) = \frac{\bar{d}_k\,\bar{l}_k}{\bar{\bar{d}}_k\,\bar{l}_k^2 + \bar{\bar{p}}_{k+1}}$$

holds if we set

$$\bar{U}_-(k) = \frac{\hat{U}_-(k)}{1+\varepsilon_{\circ}^-}\sqrt{\frac{(1+\varepsilon_{*}^-)(1+\varepsilon_{**}^-)(1+\varepsilon_{\circ\circ}^-)(1+\varepsilon_k^+)}{(1+\varepsilon_{/}^-)(1+\varepsilon_{k+1}^-)(1+\varepsilon_{+}^-)}}, \tag{31}$$

where $\varepsilon_{\circ}^-$ is identical to the $\varepsilon_{\circ}$ of (30). Note that since $\bar{d}_k\,\bar{l}_k^2 = \bar{\bar{d}}_k\,\bar{\bar{l}}_k^2$, the $(k+1)$st diagonal element in $\bar{Z}_k$ remains $\bar{\bar{D}}_-(k+1)$ as

$$\bar{d}_k\,\bar{l}_k^2 + \bar{\bar{p}}_{k+1} \;=\; \bar{\bar{d}}_k\,\bar{\bar{l}}_k^2 + \bar{\bar{p}}_{k+1} \;=\; \bar{\bar{D}}_-(k+1) \qquad \text{from (28).} \qquad \square$$

Note: A similar result may be obtained if γ_k is computed by the last formula in (18).

6. Analysis of the commutative diagram. The roundoff error analysis of the previous section shows that the commutative diagram of Figure 4 holds, with $k = r$, for Algorithm Getvec's computation, which forms the twisted factorization $N_r D_r N_r^t$ and then computes an approximate eigenvector. Figure 5 lays out the essentials given in Figure 4 and shows that the computed vector $\hat{\boldsymbol{z}}$ can be connected to the eigenvector $\boldsymbol{v}$ in three steps: (i) the right side relates $\hat{\boldsymbol{z}}$ to a vector $\bar{\boldsymbol{z}}$, (ii) the bottom arrow connects $\bar{\boldsymbol{z}}$ to an eigenvector $\bar{\boldsymbol{v}}$, and (iii) the left side relates $\bar{\boldsymbol{v}}$ to the desired eigenvector $\boldsymbol{v}$. In the rest of this section, we analyze each of these relationships in detail, before bringing it all together in section 7.

6.1. The left side—relative perturbation theory. The left side of Figure 5 examines the closeness of the eigenvector $\boldsymbol{v}$ to $\bar{\boldsymbol{v}}$ when small relative changes are made to the nontrivial entries of L and D. When LDL^t is positive (or negative) definite, it is well known that it determines its eigenvalues and eigenvectors to high relative accuracy [7], i.e., LDL^t is an RRR; see (5) and (6). However, in many cases, an indefinite LDL^t factorization also determines its eigenpairs to high relative accuracy. This section discusses conditions under which this can happen.

In the following analysis LDL^t should be thought of as the most familiar of the n twisted factorizations and the results below extend, with small modifications, to any twisted factorization.

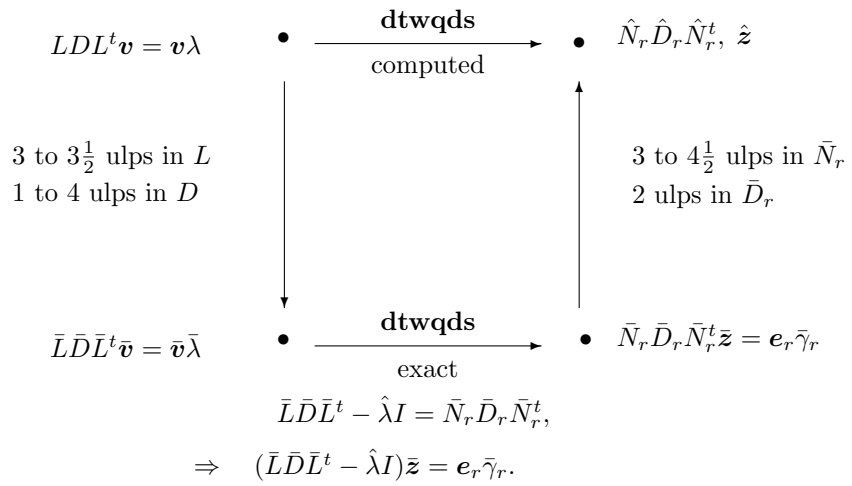

FIG. 5. *Relationships connecting* $\boldsymbol{v}$ *to* $\hat{\boldsymbol{z}}$.

6.1.1. Multiplicative form. For the sake of completeness, we present the following well-known lemma; see [8, Lemma 5.7] and its proof.

LEMMA 5. *Let L be a unit bidiagonal matrix with no zero off-diagonal entries. Independent relative perturbations in the off-diagonals may be represented by the two-sided scaling*

$$E^{-1}LE,$$

where $E = \mathrm{diag}(e_1, \ldots, e_n)$ is a diagonal scaling matrix unique to within a constant multiple, and independent of L.

Proof. Let $L_{ij}\alpha_{ij}$ represent the perturbation of L_{ij}. The equations to be solved are

$$\frac{L_{i+1,i}e_i}{e_{i+1}} = L_{i+1,i}\alpha_{i+1,i}, \qquad 1 \le i < n.$$

Letting $e_n = 1$ we get $e_{n-1} = \alpha_{n,n-1}$. Decreasing the index i further, we get

$$e_i = e_{i+1} \cdot \alpha_{i+1,i} = \prod_{j=i}^{n-1} \alpha_{j+1,j}, \qquad i = n-1, n-2, \ldots, 1. \qquad \square$$

Independent relative perturbations to nonzero entries of D are directly represented by a diagonal scaling matrix that we choose to write as F^2. Thus independent relative perturbations to the nontrivial entries of L and D lead to the perturbed matrix

$$\bar{T} = \bar{L}\bar{D}\bar{L}^t = (E^{-1}LE)(FDF)(EL^tE^{-1}). \tag{32}$$

LEMMA 6. *Let Algorithm dtwqds be executed in finite precision arithmetic. The matrices E and F that account for changes in the input L and D in order for the commutative diagram of Figure 4 to hold (see Theorem 10) satisfy*

$$(1-\varepsilon)^{6n-1} < \|(EF)^2\| < (1+\varepsilon)^{6n-1},$$

where ε is the machine precision.

Proof. Let $\bar{l}_i = l_i(1+\eta_i)$ and $\bar{d}_i = d_i(1+\delta_i)$. With twist index k, the bounds on η_i and δ_i satisfy

$$\begin{aligned}
i<k &: (1-\varepsilon)^3 < 1+\eta_i < (1+\varepsilon)^3, && 1-\varepsilon < 1+\delta_i < 1+\varepsilon,\\
i=k &: (1-\varepsilon)^{7/2} < 1+\eta_i < (1+\varepsilon)^{7/2}, && (1-\varepsilon)^4 < 1+\delta_i < (1+\varepsilon)^4,\\
i>k &: (1-\varepsilon)^3 < 1+\eta_i < (1+\varepsilon)^3, && (1-\varepsilon)^3 < 1+\delta_i < (1+\varepsilon)^3.
\end{aligned}$$

From Lemma 5 the bound on $\|E\| = \max_i |e_i|$ is maximized for e_1, the product of all the independent perturbations, which is upper bounded by $(1+\varepsilon)^{3(n-1)+1/2}$. In contrast $d_i \longrightarrow d_i f_i^2$, $i = 1, \ldots, n$, and $f_1^2 < 1+\varepsilon$ unless the twist is at $k=1$ when $f_1^2 < (1+\varepsilon)^4$. Thus

$$\begin{aligned}
\|(EF)^2\| &= \max_i (e_i f_i)^2 < (1+\varepsilon)^{6(n-1)+1}(1+\varepsilon)^4 = (1+\varepsilon)^{6n-1},\\
\|(EF)^2\| &= \max_i (e_i f_i)^2 > (1-\varepsilon)^{6(n-1)+1}(1-\varepsilon)^4 = (1-\varepsilon)^{6n-1}. \qquad \square
\end{aligned}$$

Let $(\lambda, \boldsymbol{v})$ be an eigenpair of LDL^t, $\lambda \neq 0$, $\|\boldsymbol{v}\| = 1$. We may write $\bar{T}$ in (32) in standard multiplicative form, i.e., with "outer" perturbations only, as

$$\begin{aligned}
\bar{T} &= G^t T G = G^t L D L^t G,\\
\text{where} \quad G &:= L^{-t} F E L^t E^{-1}
\end{aligned} \tag{33}$$

is an upper triangular matrix *sometimes* close to I. There is an eigenpair $(\bar{\lambda}, \bar{\boldsymbol{v}})$ of $\bar{T}$ associated with $(\lambda, \boldsymbol{v})$ and we want to investigate the closeness of $\bar{\lambda}$ to λ and $\bar{\boldsymbol{v}}$ to $\boldsymbol{v}$.

All the published results in relative perturbation theory known to us (see [11, 27, 2]) consider the form (33) above and do not require bidiagonal form for L. The perturbation bounds depend on $\|G^t - G^{-1}\|$ and/or $\|G^t G - I\|$; when these quantities are small, LDL^t can be shown to be an RRR for all the eigenvalues [12, 11, 27]. Yet G depends on L and will be far from orthogonal when L is ill-conditioned for inversion. However, in our experiments, we have often encountered situations where LDL^t is indefinite, L is ill-conditioned and the small eigenvalues in the interior of the spectrum are relatively robust while some of the larger ones are very sensitive.

So the desired bounds must not be uniform over the eigenvalues. In the work we have examined (see [2]), the bounds either are uniform or do not treat eigenvectors or are not computable. The value of the representation (32), along with Lemma 6 above, is that E and F are independent of L in the bidiagonal case. The price we pay for (32) is the presence of the "inner" scalings EF that bring us to new territory. There is a way to turn this inner scaling into a standard congruence and it was introduced in the earliest papers on computing singular values of a matrix C. Thus $C \longrightarrow XCY^t$ corresponds to

$$\begin{pmatrix} O & C \\ C^t & O \end{pmatrix} \longrightarrow \begin{pmatrix} X & O \\ O & Y \end{pmatrix} \begin{pmatrix} O & C \\ C^t & O \end{pmatrix} \begin{pmatrix} X^t & O \\ O & Y^t \end{pmatrix},$$

and the eigenvalues of the double matrix are the singular values of C and their negations, while the eigenvectors contain the right and left singular vectors. All the extensive perturbation theory for symmetric matrices has been brought to bear on the double matrix [11].

Our case $T = LDL^t$ is more difficult. In order to follow the approach indicated above, let $\Omega := \text{sign}(D) = \text{diag}(\pm 1)$ and observe that (32) corresponds to

(34)
$$\begin{pmatrix} O & L|D|^{1/2} \\ \Omega|D|^{1/2}L^t & O \end{pmatrix} \longrightarrow \begin{pmatrix} E^{-1} & O \\ O & EF \end{pmatrix} \begin{pmatrix} O & L|D|^{1/2} \\ \Omega|D|^{1/2}L^t & O \end{pmatrix} \begin{pmatrix} E^{-1} & O \\ O & EF \end{pmatrix},$$

where we have used the commutativity $EF\Omega = \Omega EF$. When $\Omega \neq I$, our double matrix is not normal and its eigenvalues are the square roots of those of LDL^t together with their negations. So the spectrum of the double matrix lies on both the real and the imaginary axes.

The first order perturbation analyses in [32, 30] foreshadow the upcoming results in section 6.1.2 and give realistic (relative) condition numbers that discriminate among the eigenpairs. Nevertheless those first order expressions do not yield bounds; the higher order terms are not controlled. One of us has developed bounds [33] on the change in both λ and $\boldsymbol{v}$, under mild conditions, and these bounds are close in form to the first order perturbation results in [32, 30], and are close to Demmel and Kahan's results in [7] when T is positive definite. In the next section we adapt these bounds to our situation. Of particular interest is the bound on the change in an eigenvector $\boldsymbol{v}$.

6.1.2. Perturbation bounds. We present here the quantities that govern the sensitivity of λ and $\boldsymbol{v}$ to the special perturbations $L \longrightarrow E^{-1}LE, D \longrightarrow FDF$ as given in section 6.1.1. In [9] Dhillon used first order perturbation theory to introduce a relative condition number corresponding to (only) the inner perturbations EF in (32),

$$\boldsymbol{\kappa}_{rel}(\lambda) := \frac{\boldsymbol{v}^t L|D|L^t\boldsymbol{v}}{|\lambda|} = \frac{\boldsymbol{v}^t L|D|L^t\boldsymbol{v}}{|\boldsymbol{v}^t LDL^t\boldsymbol{v}|},$$

and it plays the dominant role in λ's sensitivity to inner and outer perturbations.

This section gives bounds that account for both inner and outer perturbations and are derived in [33] using the double matrix form of (34); however, we quote results from [33] without giving the derivations, as they are much too long to be included here. In (34), when $\Omega = I$ the spectral decomposition of the symmetric double matrix is intimately related to the SVD of $L|D|^{1/2}$. For general Ω, we need to introduce the hyperbolic SVD (HSVD) of a matrix.

In the definitions that follow, K denotes a general real square matrix, and unlike $L|D|^{1/2}$, K need not be bidiagonal. Given a signature matrix $\Omega = \text{diag}(\omega_1, \ldots, \omega_n)$, $\omega_i = \pm 1$, and the spectral decomposition $K\Omega K^t = V\Lambda V^t, V^t = V^{-1}$, the hyperbolic SVD (HSVD) of K, introduced in [3], is defined as

$$K = V\Sigma P^t \quad \text{with} \quad P^t\Omega P = \bar{\Omega},$$

where $\bar{\Omega}$ is another signature matrix congruent to Ω. Note that $\Lambda = \Sigma^2\bar{\Omega}$. Without loss of generality, we can order the eigenvalues in Λ so that $\bar{\Omega} = \Omega$; hence the HSVD can be written as

$$K = V\Sigma P^t \quad \text{with} \quad P^t\Omega P = \Omega. \tag{35}$$

When $\Omega = I$, the standard SVD is recovered. ΩP holds the right Ω-singular vectors of K since $K\Omega P = V\Sigma\Omega$ while V holds the left Ω-singular vectors since $K^tV = P\Sigma$. Thus $(\sigma, \boldsymbol{v}, \Omega\boldsymbol{p})$ can be called an Ω-singular triple since

$$K\Omega\boldsymbol{p} = \boldsymbol{v}\sigma\omega, \quad K^t\boldsymbol{v} = \boldsymbol{p}\sigma. \tag{36}$$

Note that there are alternate interpretations: (i) P is the eigenvector matrix of the definite pair (K^tK, Ω) as $K^tK - \mu\Omega = P(\Sigma^2 - \mu\Omega)P^t, \Sigma^2 = |\Lambda|$, and (ii) ΩP is the eigenvector matrix of $\Omega K^tK = (\Omega P)\Lambda(\Omega P)^{-1}$ since, by (35), $(\Omega P)^{-1} = \Omega P^t$.

It is not hard to see that with $K = L|D|^{1/2}$, then, since $K^t\boldsymbol{v} = \boldsymbol{p}\sigma$,

$$\boldsymbol{\kappa}_{rel}(\lambda) = \frac{\boldsymbol{v}^tKK^t\boldsymbol{v}}{\sigma^2} = \boldsymbol{p}^t\boldsymbol{p} \geq 1.$$

We now present the main theorem of [33] which quantifies the change in the HSVD when K is perturbed to D_lKD_r. Before we do so, we must introduce some terminology.

Write $V = [\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n], P = [\boldsymbol{p}_1, \ldots, \boldsymbol{p}_n]$. One ingredient in the bounds on the change in $\boldsymbol{v}_j$ is the set $\{\|\boldsymbol{p}_i\|^2\}$. The other ingredient is the relative separation between eigenvalues. Actually it is the separations between the $\sigma_i = \sqrt{|\lambda_i|}$ that emerge naturally in the theory in [33],

$$\delta_{ji} := \frac{|\lambda_j - \lambda_i|}{\sigma_j + \sigma_i} = \begin{cases} |\sigma_j - \sigma_i| & \text{if } \omega_i = \omega_j, \\ \frac{\sigma_j^2 + \sigma_i^2}{\sigma_j + \sigma_i} & \text{if } \omega_i \neq \omega_j, \end{cases} \tag{37}$$

but the factors that govern the sensitivity of $\boldsymbol{v}_j$ involve both relative separations $(\delta_{ji}/\sigma_j)^2$ and $(\delta_{ji}/\sigma_i)^2$ for $i \neq j$. Note that

$$\left(\frac{\delta_{jk}}{\sigma_j}\right)^2 = \left(\frac{|\lambda_j - \lambda_k|}{\sigma_j(\sigma_j + \sigma_k)}\right)^2 = \frac{|\lambda_j - \lambda_k|}{|\lambda_j|} \cdot \frac{|\lambda_j - \lambda_k|}{(\sigma_j + \sigma_k)^2} \geq \frac{|\lambda_j - \lambda_k|}{|\lambda_j|} \cdot \frac{|\lambda_j - \lambda_k|}{2(|\lambda_j| + |\lambda_k|)}.$$

The upcoming bounds concern a particular eigenpair $(\lambda_j, \boldsymbol{v}_j)$, $\|\boldsymbol{v}_j\| = 1$, and as noted above, $\|\boldsymbol{p}_j\|$ plays a leading role. The other quantities of interest are

$$\begin{aligned} \mathrm{rgap}_j &:= \min_{i \neq j} \frac{\delta_{ji}}{\sigma_j}, \\ \|\boldsymbol{m}_j\|^2 &:= \sum_{i \neq j} \left(\frac{\|\boldsymbol{p}_i\|}{\delta_{ji}/\sigma_j}\right)^2, \\ \|\boldsymbol{m}_{\langle j \rangle}\|^2 &:= \sum_{i \neq j} \left(\frac{\|\boldsymbol{p}_i\|}{\delta_{ji}/\sigma_i}\right)^2, \\ \|P_{\langle j \rangle}\|_F^2 &:= \sum_{i \neq j} \|\boldsymbol{p}_i\|^2, \end{aligned}$$

where $\langle j \rangle$ denote the index set complementary to j in $\{1, \ldots, n\}$. The actual vectors $\boldsymbol{m}_j$ and $\boldsymbol{m}_{\langle j \rangle}$ play no role in the bounds; only their norms are needed. Note that $1/\mathrm{rgap}_j \leq \|\boldsymbol{m}_j\| \leq \|P_{\langle j \rangle}\|_F/\mathrm{rgap}_j$. Of the four expressions given above, the second appears in the bounds and the other three in the restriction on the perturbation level of Theorem 7.

We cite the needed part of the relevant theorem from [33] and then adapt it to our situation.

THEOREM 7. *Let Ω be a signature matrix, $\Omega = \mathrm{diag}(\omega_1, \ldots, \omega_n)$, $\omega_i = \pm 1$. Consider an invertible matrix K with HSVD as in* (35). *Define $\Lambda = \mathrm{diag}(\lambda_1, \ldots, \lambda_n) := \Sigma^2\Omega$ and assume that $\lambda_i \neq \lambda_j$, $i \neq j$. Let K be perturbed to D_lKD_r, with D_l and D_r diagonal, and let $\bar{\varepsilon}_d := \max\{\|D_l^{\pm 2} - I\|, \|D_r^{\pm 2} - I\|\}$. If $\bar{\varepsilon}_d$ is small enough that*

$$8\bar{\varepsilon}_d\|\boldsymbol{p}_j\|^2 \leq \mathrm{rgap}_j, \qquad 4\bar{\varepsilon}_d\|\boldsymbol{m}_{\langle j \rangle}\|\,\|P_{\langle j \rangle}\|_F \leq 1, \tag{38}$$

then there is an Ω-singular triple $(\bar{\sigma}_j, \bar{\boldsymbol{v}}_j, \Omega\bar{\boldsymbol{v}}_j)$ of D_lKD_r such that

$$|\sin\angle(\bar{\boldsymbol{v}}_j, \boldsymbol{v}_j)| \le \bar{\varepsilon}_d \left(2\|\boldsymbol{p}_j\|\|\boldsymbol{m}_j\| + \frac{1}{2}\right)/(1-\bar{\varepsilon}_d),$$

$$\text{and} \quad \frac{|\bar{\sigma}_j^2 - \sigma_j^2|}{\sigma_j^2} \le \frac{\bar{\varepsilon}_d(\|\boldsymbol{p}_j\|^2+1)+\beta_2}{1-\bar{\varepsilon}_d(\|\boldsymbol{p}_j\|^2+1)-\beta_2},$$

where $0 < \beta_2 \le 2(\bar{\varepsilon}_d\|\boldsymbol{p}_j\|)^2\{(\|P_{\langle j\rangle}\|_F + 2\|\boldsymbol{m}_j\|)^2 + 4\|\boldsymbol{m}_j\|\,\|\boldsymbol{m}_{\langle j\rangle}\|\}$.

Theorem 7 reveals how we should define the condition numbers:

$$\text{relcond}(\boldsymbol{v}_j) := 2\|\boldsymbol{p}_j\|\,\|\boldsymbol{m}_j\| + 1/2, \tag{39}$$

$$\text{relcond}(\lambda_j) := \|\boldsymbol{p}_j\|^2 + 1. \tag{40}$$

Note that relcond(λ_j) exceeds Dhillon's $\boldsymbol{\kappa}_{rel}(\lambda_j)$ by 1 and it accounts for both the inner and outer perturbations.

The following lemma shows that in the indefinite well-conditioned case and in the definite case, these relative condition numbers are small. For any invertible M define $\text{cond}_F(M) := \|M\|_F\|M^{-1}\|_F$ and $\text{cond}_2(M) := \|M\|_2\|M^{-1}\|_2$.

LEMMA 8. *With the notation developed above (setting $K = L|D|^{1/2}$, $\Omega = \text{sign}(D)$),*

$$\text{relcond}(\boldsymbol{v}_j) \;\le\; \frac{\text{cond}_F(L|D|^{1/2})}{\text{rgap}_j} + 1/2, \;\text{ and }\; \text{relcond}(\lambda_j) \;\le\; \text{cond}_2(L|D|^{1/2}).$$

When LDL^t is definite, then

$$\text{relcond}(\boldsymbol{v}_j) \;\le\; \frac{2\sqrt{n-1}}{\text{rgap}_j} + 1/2 \;\text{ and }\; \text{relcond}(\lambda_j) \;=\; 2.$$

Proof. In [33, Lemma 2.1] it is shown that

$$\|P\|_F^2 = \sum \|\boldsymbol{p}_i\|^2 \le \text{cond}_F(L|D|^{1/2}).$$

Observe, from the definition of $\|\boldsymbol{m}_j\|$, that $H_j^2 := \|P_{\langle j\rangle}\|_F^2/\|\boldsymbol{m}_j\|^2$ is a weighted harmonic mean of the relative gaps $(\delta_{ji}/\sigma_j)^2$ and, as for any mean, $\text{rgap}_j^2 \le H_j^2$. From the definition of relcond$(\boldsymbol{v}_j)$ in (39),

$$\begin{aligned}\text{relcond}(\boldsymbol{v}_j) &= 2\|\boldsymbol{p}_j\|\frac{\|P_{\langle j\rangle}\|_F}{H_j} + 1/2, \\ &\le \frac{\|P\|_F^2}{\text{rgap}_j} + 1/2, \\ &\le \frac{\text{cond}_F(L|D|^{1/2})}{\text{rgap}_j} + 1/2.\end{aligned} \tag{41}$$

The bound on relcond(λ_j) is an easy consequence of (36). When D is definite, then $\|\boldsymbol{p}_i\| = 1$ for all i, and $\|P_{\langle j\rangle}\|_F^2 = n-1$. Thus, by (41), the result holds. ∎

Next we apply Theorem 7 to our situation.

COROLLARY 9. *With the notation presented above let invertible tridiagonal $T = LDL^t$ have eigenpairs $(\lambda_i, \boldsymbol{v}_i)$, $\|\boldsymbol{v}_i\| = 1$, with L a proper bidiagonal matrix (all*

$l_i \neq 0$) with 1's on the diagonal, and D a diagonal matrix. Consider perturbations $L \longrightarrow E^{-1}LE$, $D \longrightarrow FDF$ with $\|(EF)^2 - I\| = \bar{\varepsilon}_d$. If $\bar{\varepsilon}_d$ is small enough that (38) *holds, then there is an eigenpair $(\bar{\lambda}_j, \bar{\boldsymbol{v}}_j)$ of $\bar{T} = E^{-1}LEFDFEL^tE^{-1}$ such that*

$$|\sin \angle(\bar{\boldsymbol{v}}_j, \boldsymbol{v}_j)| \leq \mathrm{relcond}(\boldsymbol{v}_j)\bar{\varepsilon}_d/(1-\bar{\varepsilon}_d),$$
$$\textit{and} \quad \frac{|\bar{\lambda}_j - \lambda_j|}{|\lambda_j|} \leq \frac{\bar{\varepsilon}_d \mathrm{relcond}(\lambda_j) + \beta_2}{1 - \bar{\varepsilon}_d \mathrm{relcond}(\lambda_j) - \beta_2},$$

where $\mathrm{relcond}(\boldsymbol{v}_j)$ *and* $\mathrm{relcond}(\lambda_j)$ *are defined in* (39) *and* (40)*, respectively, and β_2 is as in Theorem* 7.

Proof. L is proper so that no subdiagonal entry vanishes, T is invertible, and thus its eigenvalues are simple and do not vanish. Therefore we may apply Theorem 7 with $\bar{\varepsilon}_d := \max\{\|E^{-2} - I\|, \|(EF)^2 - I\|\} = \|(EF)^2 - I\|$. The left Ω-singular vectors V of $K = L|D|^{1/2}$ are the eigenvectors of $K\Omega K^t = T$. ∎

The bounds in Theorem 7 make no explicit reference to n, the order of the matrix. This factor appears when we relate $\bar{\varepsilon}_d$ to the machine precision ε. In our application, Lemma 6 above gives $1 + \bar{\varepsilon}_d < (1+\varepsilon)^{6n-1}$. Corollary 9 and this bound on $\bar{\varepsilon}_d$ will be applied in Theorem 15.

Lemma 8 covers the "easy" cases. However, in the case when LDL^t is indefinite and ill-conditioned the small eigenpairs are often relatively robust, while the large ones are not. Corollary 9 covers such cases. The following example illustrates one such situation.

Example 2. To bring life to the quantities of this section we exhibit a 4×4 symmetric matrix $T = T(\eta)$ that depends on a parameter η (think of η as 10^{-8}):

$$T = T(\eta) := \begin{bmatrix} \eta & \frac{1}{\sqrt{2}} & 0 & 0 \\ \frac{1}{\sqrt{2}} & -2\eta & \frac{1}{\sqrt{2}} & 0 \\ 0 & \frac{1}{\sqrt{2}} & 3\eta & \eta \\ 0 & 0 & \eta & 2\eta \end{bmatrix},$$
$$T = LDL^t = V(\Sigma^2\Omega)V^t,$$
$$\Omega = \mathrm{diag}(1,-1,1,1), \quad L|D|^{1/2} = V\Sigma P^t.$$

T is indefinite but permits triangular factorization $T = LDL^t$ with large element growth, like $1/\eta$, in the multipliers. This ill-conditioned $L|D|^{1/2}$ has HSVD: $L|D|^{1/2} = V\Sigma P^t$; recall from (40) that the $\|\boldsymbol{p}_i\|$ govern the relative conditioning of the eigenvalues. We present only the leading terms in the quantities shown below. The eigenvalues $\omega\sigma^2$ of T are not presented in monotonic order because of the constraint that $P^t\Omega P = \Omega$. For full details see [33].

$$\Lambda = \Sigma^2\Omega = \mathrm{diag}\left(\frac{4-\sqrt{2}}{2}\eta, -1, \frac{4+\sqrt{2}}{2}\eta, +1\right),$$
$$\Sigma = \mathrm{diag}(\sqrt{\eta}\,\mu_-, 1, \sqrt{\eta}\,\mu_+, 1) \quad \mu_-^2 := \frac{4-\sqrt{2}}{2}, \quad \mu_+^2 := \frac{4+\sqrt{2}}{2},$$
$$\left\{\|\boldsymbol{p}_i\|^2\right\} = \left\{\ 2 - \frac{\sqrt{2}}{4}, \quad \frac{1}{2\eta}, \quad 2 + \frac{\sqrt{2}}{4}, \quad \frac{1}{2\eta}\ \right\}.$$

The two small σ's are close, $\sigma_1 \approx 1.137\eta$ and $\sigma_3 \approx 1.645\eta$, while the other σ's are almost 1 but have differing ω values. The large singular values are extremely

sensitive, "condition number" $1/\eta$, but the two small σ's are relatively robust and the associated $\boldsymbol{v}$'s also turn out to be relatively robust. We demonstrate the latter for $\boldsymbol{v}_1$.

For $\sigma_1 = \sqrt{\eta}\,\mu_- \approx 1.137\sqrt{\eta}$,

$$\text{rgap}_1 = \frac{\mu_+ - \mu_-}{\mu_-} \approx .447,$$

$$\delta_{12} = \frac{1+\eta\mu_-^2}{1+\sqrt{\eta}\mu_-}, \quad \delta_{13} = \sqrt{\eta}(\mu_+ - \mu_-), \quad \delta_{14} = 1 - \sqrt{\eta}\mu_-.$$

Recall from (37) that δ_{12} is a quotient, not just a difference, because $\omega_1 \neq \omega_2$.

The point of this example is the neutralizing of large $\|\boldsymbol{p}\|$ values in the first and third terms by equally large relative separations in $\|\boldsymbol{m}_1\|$ below:

$$\begin{aligned}
\|\boldsymbol{m}_1\| &= \left[\sum_{i\neq 1} \|\boldsymbol{p}_i\|^2 \left(\sigma_1/\delta_{1i}\right)^2\right]^{1/2} \\
&= \left[\frac{1}{2\eta}\left(\frac{\sqrt{\eta}\mu_-}{1-\sqrt{\eta}\mu_-}\right)^2 + \left(2+\frac{\sqrt{2}}{4}\right)\left(\frac{\sqrt{\eta}\mu_-}{\sqrt{\eta}(\mu_+-\mu_-)}\right)^2 + \frac{1}{2\eta}\left(\frac{\sqrt{\eta}\mu_-}{1-\sqrt{\eta}\mu_-}\right)^2\right]^{1/2} \\
&\approx \left[\frac{1}{2}\mu_-^2 + \left(2+\frac{\sqrt{2}}{4}\right)\left(\frac{\mu_-}{\mu_+-\mu_-}\right)^2 + \frac{1}{2}\mu_-^2\right]^{1/2} \\
&\approx (\mu_-)\left[1+(8+\sqrt{2})\right]^{1/2} \approx 3.44.
\end{aligned}$$

Combined with the modest value of $\|\boldsymbol{p}_1\|$, relcond$(\boldsymbol{v}_1)$ defined in (39) is less than 10. Similarly $\boldsymbol{v}_3$ is seen to be relatively robust, but $\boldsymbol{v}_2$ and $\boldsymbol{v}_4$ are not. ∎

6.2. The bottom arrow—rank-revealing twisted factorization. The bottom arrow in Figure 5 represents an exact relation $\bar{L}\bar{D}\bar{L}^t - \hat{\lambda}I = \bar{N}_r\bar{D}_r\bar{N}_r^t$. Consider the vector $\bar{\boldsymbol{z}}^{(r)}$ (denoted as $\bar{\boldsymbol{z}}$ in Figure 5) such that $\bar{\boldsymbol{z}}^{(r)}(r) = 1$ and $(\bar{L}\bar{D}\bar{L}^t - \hat{\lambda}I)\bar{\boldsymbol{z}}^{(r)} = \boldsymbol{e}_r\bar{\gamma}_r$. This section presents the desired bound on the residual norm $\|(\bar{L}\bar{D}\bar{L}^t - \hat{\lambda}I)\bar{\boldsymbol{z}}^{(r)}\|/\|\bar{\boldsymbol{z}}^{(r)}\|$ when r is chosen appropriately. For ease of notation, we drop the overbars for the rest of section 6.2; thus the quantities L, D, γ_k, and $\boldsymbol{z}$ below can be thought of as $\bar{L}$, $\bar{D}$, $\bar{\gamma_k}$, and $\bar{\boldsymbol{z}}$, respectively, in Figure 5. We first establish that in cases of interest, when $\hat{\lambda}$ approximates λ, then one of the γ_k, $1 \leq k \leq n$, reveals that $LDL^t - \hat{\lambda}I$ is nearly singular.

Let $\hat{\lambda} \neq \lambda$. Since $\boldsymbol{e}_k^t\boldsymbol{z}^{(k)} = 1 = \gamma_k\boldsymbol{e}_k^t(LDL^t - \hat{\lambda}I)^{-1}\boldsymbol{e}_k$,

$$\gamma_k^{-1} = \boldsymbol{e}_k^t(LDL^t - \hat{\lambda}I)^{-1}\boldsymbol{e}_k. \tag{42}$$

We present next the relation of γ_k to the spectral factorization of $LDL^t - \hat{\lambda}I$ using an eigenvector expansion. These results do not require the tridiagonal form.

Let $LDL^t = V\Lambda V^t$. Replace LDL^t with $V\Lambda V^t$ in (42) to find, for each k,

$$\frac{1}{\gamma_k} = \frac{|v_j(k)|^2}{\lambda_j - \hat{\lambda}} + \sum_{i\neq j}\frac{|v_i(k)|^2}{\lambda_i - \hat{\lambda}}, \tag{43}$$

where $\lambda = \lambda_j$ is an eigenvalue closest to $\hat{\lambda}$ and its normalized eigenvector is $\boldsymbol{v}_j$. The following theorem shows that when λ_j is isolated the twist index k for which the eigenvector component $|v_j(k)|$ is large leads to a small $|\gamma_k|$.

THEOREM 10. *Let γ_k be as in (43), where $\hat{\lambda}$ approximates λ_j, and let λ_j be isolated enough, i.e.,*

$$\frac{|\lambda_j - \hat{\lambda}|}{\mathrm{gap}(\hat{\lambda})} \;\le\; \frac{1}{M} \cdot \frac{1}{n-1}, \tag{44}$$

where $M > 1$ and $\mathrm{gap}(\hat{\lambda}) := \min_{i \neq j} |\lambda_i - \hat{\lambda}|$. Then, for k such that $v_j(k) \ge 1/\sqrt{n}$,

$$|\gamma_k| \;\le\; \frac{|\lambda_j - \hat{\lambda}|}{|v_j(k)|^2} \cdot \frac{M}{M-1} \;\le\; n|\lambda_j - \hat{\lambda}| \cdot \frac{M}{M-1}. \tag{45}$$

Proof. A proof is given in [9, Section 3.2], which we repeat here for the sake of completeness. By (43),

$$\frac{1}{\gamma_k} = \frac{|v_j(k)|^2}{\lambda_j - \hat{\lambda}} \left[1 + \sum_{i \neq j} \left| \frac{v_i(k)}{v_j(k)} \right|^2 \left(\frac{\lambda_j - \hat{\lambda}}{\lambda_i - \hat{\lambda}} \right) \right]. \tag{46}$$

Since

$$\sum_{i \neq j} \left| \frac{v_i(k)}{v_j(k)} \right|^2 = \frac{1 - |v_j(k)|^2}{|v_j(k)|^2},$$

we can rewrite (46) as

$$\frac{1}{\gamma_k} = \frac{|v_j(k)|^2}{\lambda_j - \hat{\lambda}} \left[1 + \left(|v_j(k)|^{-2} - 1 \right) \mathcal{A}_1 \right], \tag{47}$$

where

$$\mathcal{A}_1 = \sum_{i \neq j} w_i \left(\frac{\lambda_j - \hat{\lambda}}{\lambda_i - \hat{\lambda}} \right), \quad 1 = \sum_{i \neq j} w_i, \;\; w_i \ge 0,$$

and so

$$|\mathcal{A}_1| \le |\lambda_j - \hat{\lambda}| / \mathrm{gap}(\hat{\lambda}), \quad \text{with } \mathrm{gap}(\hat{\lambda}) = \min_{i \neq j} |\lambda_i - \hat{\lambda}|. \tag{48}$$

If (44) holds, then by (47) and (48),

$$|\gamma_k| \le \frac{|\lambda_j - \hat{\lambda}|}{|v_j(k)|^2} \left| 1 - \left(|v_j(k)|^{-2} - 1 \right) \left(\frac{1}{M \cdot (n-1)} \right) \right|^{-1}.$$

For k such that $|v_j(k)| \ge 1/\sqrt{n}$,

$$|\gamma_k| \le \frac{|\lambda_j - \hat{\lambda}|}{|v_j(k)|^2} \left[1 - \frac{1}{M} \right]^{-1},$$

and so the result holds. ∎

In general, the case $\gamma_k = \infty$ for all k can occur, but we are free to choose $\hat{\lambda}$ to avoid such situations; see also [9, section 3.3]. In cases of interest, $|\lambda_j - \hat{\lambda}|/\mathrm{gap}(\hat{\lambda}) = O(\varepsilon)$, implying that $M \gg 1$ and $M/(M-1) \approx 1$, whence (45) shows that when

$|v_j(k)|$ is above average, $|\gamma_k|$ reveals the near singularity. This justifies step III in Algorithm Getvec.

We now show that under suitable conditions the vector $\boldsymbol{z}^{(k)}$ enjoys a small residual norm and serves as an excellent approximation to the eigenvector $\boldsymbol{v}_j$ [14, 31].

THEOREM 11. *Let* $\boldsymbol{z}^{(k)}$ *satisfy*

$$(LDL^t - \hat{\lambda}I)\boldsymbol{z}^{(k)} = \boldsymbol{e}_k\gamma_k$$

with $z^{(k)}(k) = 1$, *and let* γ_k *be as in* (43) *where* $\hat{\lambda}$ *approximates* λ_j, $\hat{\lambda} \neq \lambda_j$. *Then, if* $v_j(k) \neq 0$, *the residual norm*

$$\frac{\|(LDL^t - \hat{\lambda}I)\boldsymbol{z}^{(k)}\|}{\|\boldsymbol{z}^{(k)}\|} \;=\; \frac{|\gamma_k|}{\|\boldsymbol{z}^{(k)}\|} \;\leq\; \frac{|\lambda_j - \hat{\lambda}|}{|v_j(k)|},$$

and thus for at least one k,

$$\frac{|\gamma_k|}{\|\boldsymbol{z}^{(k)}\|} \;\leq\; \sqrt{n}|\lambda_j - \hat{\lambda}|.$$

Proof. A proof is given in [31, section 5] and [9, section 3.2], but we repeat it here for the sake of completeness. Recall that $LDL^t = V\Lambda V^t$. Then

$$\begin{aligned}
\boldsymbol{z}^{(k)} &= (LDL^t - \hat{\lambda}I)^{-1}\boldsymbol{e}_k\gamma_k, \\
\Rightarrow \|\boldsymbol{z}^{(k)}\|^2 &= |\gamma_k|^2\boldsymbol{e}_k^t V(\Lambda - \hat{\lambda}I)^{-2}V^t\boldsymbol{e}_k, \\
&= |\gamma_k|^2 \sum_{i=1}^{n} \frac{|v_i(k)|^2}{|\hat{\lambda} - \lambda_i|^2}, \\
\Rightarrow \frac{|\gamma_k|}{\|\boldsymbol{z}^{(k)}\|} &\leq \frac{|\lambda_j - \hat{\lambda}|}{|v_j(k)|} \quad \text{for all } k \text{ with } v_j(k) \neq 0.
\end{aligned}$$

Noting that $|v_j(k)| \geq 1/\sqrt{n}$ for at least one k completes the proof. ∎

However, $(\hat{\lambda}, \boldsymbol{z}^{(k)})$ is not the best approximate eigenpair because $\hat{\lambda}$ is not the Rayleigh quotient of $\boldsymbol{z}^{(k)}$. By using the Rayleigh quotient we obtain a useful decrease in residual norm.

LEMMA 12. *Let* $LDL^t = T$ *and* $(T - \hat{\lambda}I)\boldsymbol{z}^{(k)} = \boldsymbol{e}_k\gamma_k$, $z^{(k)}(k) = 1$. *Then the Rayleigh quotient* ρ *with respect to* $T - \hat{\lambda}I$ *is*

$$\rho(\boldsymbol{z}^{(k)}) = \gamma_k/\|\boldsymbol{z}^{(k)}\|^2,$$

$$\text{and} \quad \|(T - (\hat{\lambda} + \rho)I)\boldsymbol{z}^{(k)}\|/\|\boldsymbol{z}^{(k)}\| = \frac{\gamma_k}{\|\boldsymbol{z}^{(k)}\|^2}\left(\|\boldsymbol{z}^{(k)}\|^2 - 1\right)^{1/2}.$$

Proof. Write $\boldsymbol{z}$ for $\boldsymbol{z}^{(k)}$ and γ for γ_k, and note that

$$\boldsymbol{z}^t(T - \hat{\lambda}I)\boldsymbol{z} = \boldsymbol{z}^t\boldsymbol{e}_k\gamma = \gamma, \quad \text{since } z(k) = 1,$$

and

$$\begin{aligned}
(T - (\hat{\lambda} + \rho)I)\boldsymbol{z} &= \boldsymbol{e}_k\gamma - \boldsymbol{z}\rho, \\
\|(T - (\hat{\lambda} + \rho)I)\boldsymbol{z}\|^2 &= \gamma^2 + \|\boldsymbol{z}\|^2\rho^2 - 2\gamma\rho, \\
&= \frac{\gamma^2}{\|\boldsymbol{z}\|^2}\left(\|\boldsymbol{z}\|^2 - 1\right).
\end{aligned}$$

∎

The above lemma justifies the use of Algorithm Getvec in increasing $\hat{\lambda}$'s accuracy; see Remark 5 in section 4.

6.3. The right side—computing the eigenvector by multiplications. This section looks at the right side of Figure 5 and shows that the vector $\hat{\boldsymbol{z}}$ computed by Algorithm Getvec is very close to a vector $\bar{\boldsymbol{z}}$ that obeys the exact relationship (49), where $\bar{N}_r$ and $\bar{D}_r$ are perturbed factors determined by step IV of Algorithm Getvec.

THEOREM 13. *Let $\hat{N}_r$ and $\hat{D}_r$, $\bar{N}_r$ and $\bar{D}_r$ be the twisted factors represented by $\hat{Z}_r$ and $\bar{Z}_r$, respectively, in Figure* 4 *(see also Theorem* 4 *and Figure* 5*). Let $\hat{\boldsymbol{z}}$ be the vector computed in step* IV *of Algorithm* Getvec*, and let $\bar{\boldsymbol{z}}$ be the exact solution of*

$$\bar{N}_r \bar{D}_r \bar{N}_r^t \bar{\boldsymbol{z}} = \bar{\gamma}_r \boldsymbol{e}_r, \tag{49}$$

where $\bar{z}(r) = 1$. Then, barring underflow, $\hat{\boldsymbol{z}}$ is a small relative perturbation of $\bar{\boldsymbol{z}}$. Specifically,

$$\begin{aligned} &\hat{z}(r) = \bar{z}(r) \;=\; 1, \\ &\hat{z}(i) = \bar{z}(i)\cdot(1+\eta_i), \;\; i \neq r, \quad (1-\varepsilon)^{5|i-r|}\varepsilon \leq 1+\eta_i \leq (1+\varepsilon)^{5|i-r|\varepsilon}, \end{aligned} \tag{50}$$

where ε is the machine precision.

Proof. The above bound accounts for the roundoff errors in the recurrence in step IV of Algorithm Getvec. For now, assume that no component of D_+ or D_- is zero (so that only the top formulae for $\hat{z}(i)$ and $\hat{z}(j+1)$ in step IV are used). The matrix $\bar{N}_r$, built out of $\vec{L}_+$ and $\overleftarrow{U}_-$, was defined in Theorem 4 so that the equality $\bar{L}\bar{D}\bar{L}^t - \hat{\lambda}I = \bar{N}_r \bar{D}_r \bar{N}_r^t$ holds. Thus $\bar{N}_r$ is a given matrix, not to be modified, in the context of this theorem. Because of the roundoff error in multiplication the top entries of $\hat{\boldsymbol{z}}$ computed in step IV of Algorithm Getvec satisfy

$$\hat{z}(i) = -\hat{L}_+(i)\hat{z}(i+1)(1+\varepsilon_i), \quad i < r,$$

and the bottom entries satisfy

$$\hat{z}(i) = -\hat{U}_-(i-1)\hat{z}(i-1)(1+\varepsilon_i), \quad i > r, \tag{51}$$

where $|\varepsilon_i| < \varepsilon$. In contrast, the ideal vector $\bar{\boldsymbol{z}}$ satisfies

$$\begin{aligned} \bar{z}(i) &= -\vec{L}_+(i)\bar{z}(i+1), && i < r, \\ \text{and} \quad \bar{z}(i) &= -\overleftarrow{U}_-(i-1)\bar{z}(i-1), && i > r. \end{aligned} \tag{52}$$

Since $\hat{z}(r) = \bar{z}(r) = 1$, we may define $\eta_r = 0$ and trivially write $\hat{z}(r) = \bar{z}(r)(1+\eta_r)$ with $|\eta_r| \leq 4(r-r)\varepsilon$. Now proceed by induction as i decreases in order to prove (50). Examine (23) to find that

$$\hat{L}_+(i) = \vec{L}_+(i)(1+\delta_i), \quad (1-\varepsilon)^3 < 1+\delta_i < (1+\varepsilon)^3 \quad \text{for all } i < r.$$

Thus

$$\begin{aligned} \hat{z}(i-1) &= -\vec{L}_+(i-1)(1+\delta_{i-1})\hat{z}(i)(1+\varepsilon_{i-1}), \\ &= -\vec{L}_+(i-1)(1+\delta_{i-1})\bar{z}(i)(1+\eta_i)(1+\varepsilon_{i-1}), \\ &\quad \text{where } (1-\varepsilon)^{4(r-i)} \leq 1+\eta_i \leq (1+\varepsilon)^{4(r-i)} \text{ by induction}, \\ &= \bar{z}(i-1)(1+\delta_{i-1})(1+\eta_i)(1+\varepsilon_{i-1}), \;\; \text{by (52)} \\ &= \bar{z}(i-1)(1+\eta_{i-1}), \quad \text{thus defining } 1+\eta_{i-1} := (1+\eta_i)(1+\delta_{i-1})(1+\varepsilon_{i-1}), \end{aligned}$$

and $(1-\varepsilon)^{4(r-i)+4} \leq 1+\eta_{i-1} \leq (1+\varepsilon)^{4(r-i)+4}$, as claimed.

For the lower half of $\hat{\boldsymbol{z}}$, $i \geq r$, the argument is similar with $\hat{U}_-$ and $\overleftarrow{U}_-$ involved instead of $\hat{L}_+$ and $\vec{L}_+$. Note that $\hat{U}_-$ is related to $\overleftarrow{U}_-$ by (31) and (30), which, respectively, involve $1\frac{1}{2}$ and 1 more ulps than (23).

To begin, define $\eta_r = 0$ so that $|\eta_r| \leq 5(r-r)\varepsilon$. For $i = r+1$, (50) holds since (31) gives 4.5 ulps for $\overleftarrow{U}_-(r)$ in (51), while $\varepsilon_{r+1} = 0$ (because $\hat{z}(r) = 1$). For $i > r+1$, (30) gives 4 ulps and ε_i in (51) gives one more ulp for an increase of at most 5 ulps each time i increases. Thus (50) holds for all values of i.

We now consider the case when an eigenvector entry vanishes, i.e., $D_+(i) = 0$ (see Remark 3 in section 4). In this case the alternate formulae in step IV of Algorithm Getvec are used to compute the next eigenvector entry, i.e., if $i < r$, then

$$z(i) = -(d_{i+1}l_{i+1}/d_i l_i)z(i+2), \tag{53}$$

where d_i and l_i are elements of the input matrices L and D. Examining the relations between d_i and $\vec{d}_i$, and between l_i and $\vec{l}_i$ in the proof of Theorem 2, we can see that the product

$$d_i l_i \;=\; \vec{d}_i\vec{l}_i(1+\xi_i) \;=\; \vec{D}_+(i)\vec{L}_+(i)(1+\xi_i), \quad (1-\varepsilon)^3 < 1+\xi_i < (1+\varepsilon)^3, \quad i < r.$$

Thus the term $(d_{i+1}l_{i+1}/d_i l_i)$ in (53) contributes 6 ulps, and combining these with the 4 arithmetic operations in (53), we can write

$$\hat{z}(i) = -(\vec{d}_{i+1}\vec{l}_{i+1}/\vec{d}_i\vec{l}_i)\hat{z}(i+2)\cdot(1+\delta_i),$$

where $(1-\varepsilon)^{10} < 1+\delta_i < (1+\varepsilon)^{10}$. (A closer analysis reveals that $(1-\varepsilon)^8 < 1+\delta_i < (1+\varepsilon)^8$.) Thus (50) holds in this case also. The case when $D_-(i+1) = 0$, $i > r$, is similar. ∎

Corollary 14 (to Theorem 13). *Under the hypotheses of Theorem* 13,

$$|\sin\angle(\bar{\boldsymbol{z}}, \hat{\boldsymbol{z}})| \leq \frac{(1+\varepsilon)^{5(n-1)} - 1}{(1-\varepsilon)^{5(n-1)}}.$$

Proof. First we establish a general result on elementwise perturbation of vectors which shows that the term $(n-1)$ above could be replaced by a weighted standard deviation of the relative changes to $\hat{\boldsymbol{z}}$'s entries.

Let $\mathbf{0} \neq \boldsymbol{u} \in \mathbb{R}^n$ and let $\bar{\boldsymbol{u}}$ be given by $\bar{u}(i) = (1+\eta_i)u(i)$. For expressions concerning the angle $\angle(\boldsymbol{u}, \bar{\boldsymbol{u}})$ there is no loss in assuming that $\|\boldsymbol{u}\|^2 = \boldsymbol{u}^t\boldsymbol{u} = 1$.

Now,

$$\begin{aligned}
\cos^2\angle(\boldsymbol{u}, \bar{\boldsymbol{u}}) &= \frac{(\bar{\boldsymbol{u}}^t\boldsymbol{u})^2}{\bar{\boldsymbol{u}}^t\bar{\boldsymbol{u}}} \\
&= \frac{1 + 2\sum \eta_i u(i)^2 + (\sum \eta_i u(i)^2)^2}{1 + 2\sum \eta_i u(i)^2 + \sum \eta_i^2 u(i)^2}, \\
\sin^2\angle(\boldsymbol{u}, \bar{\boldsymbol{u}}) &= \frac{\sum \eta_i^2 u(i)^2 - (\sum \eta_i u(i)^2)^2}{1 + 2\sum \eta_i u(i)^2 + \sum \eta_i^2 u(i)^2}.
\end{aligned}$$

The numerator is a weighted variance of the η_i which we denote by (std. dev.$(\eta_i; \boldsymbol{u})$)^2. The denominator exceeds $(1+\text{avg})^2$, where avg $=$ avg$(\eta_i; \boldsymbol{u}) = \sum \eta_i u(i)^2$ because, by Cauchy–Schwarz, avg$^2 = (\sum \eta_i u(i)\cdot u(i))^2 \leq \sum \eta_i^2 u(i)^2$. On taking square roots,

$$|\sin\angle(\boldsymbol{u}, \bar{\boldsymbol{u}})| \leq \frac{\text{std. dev.}(\eta_i; \boldsymbol{u})}{1 + \text{avg}}. \tag{54}$$

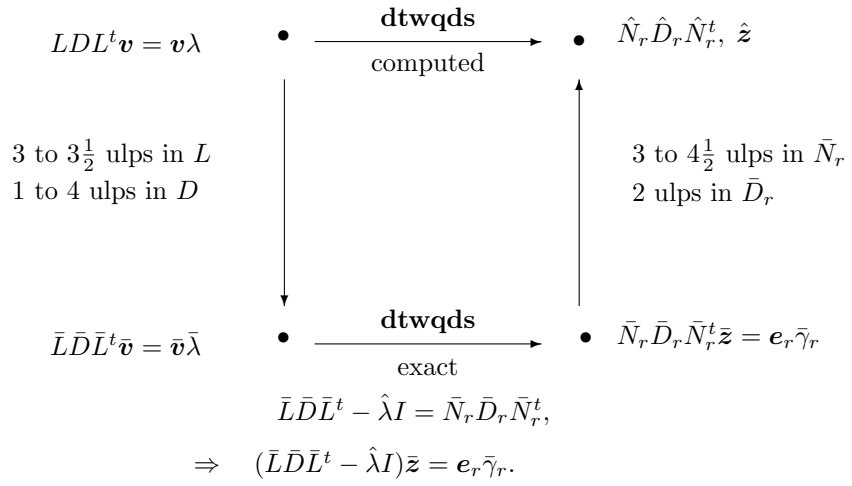

FIG. 6. *Relationships connecting* $\boldsymbol{v}$ *to* $\hat{\boldsymbol{z}}$.

A crude but simple bound on a standard deviation of the η_i is $\max_i |\eta_i|$. Finally substitute $\bar{\boldsymbol{z}}$ for $\boldsymbol{u}$ and $\hat{\boldsymbol{z}}$ for $\bar{\boldsymbol{u}}$ and use (50) to verify that

$$1 + \max_i |\eta_i| \ \le\ \max\left((1+\varepsilon)^{5(n-r)}, (1+\varepsilon)^{5(r-1)}\right),$$

and, by Theorem 13,

$$(1-\varepsilon)^{5(n-1)} \ \le\ 1 + \text{avg.}$$

Since r might be 1 or n the corollary is established. ∎

Note: The standard deviation in (54) *is weighted by the squares of the eigenvector entries. So, in practice, for localized eigenvectors we can replace* $n-1$ *in the above bound with the size of the numerical support of* $\hat{\boldsymbol{z}}$.

7. Bounds on accuracy (proof of correctness). The following is the main theorem of the paper. Figure 6 is identical to Figure 5 and we repeat it here so that it can be readily consulted.

THEOREM 15. *Let* $(\lambda, \boldsymbol{v}) = (\lambda_j, \boldsymbol{v}_j)$ *be an eigenpair of the real symmetric unreduced* $n \times n$ *tridiagonal matrix* LDL^t *with* $\|\boldsymbol{v}\| = 1$. *Let* $\hat{\lambda}$ *be an accurate approximation closer to* λ *than to any other eigenvalue of* LDL^t *and let* $\hat{\boldsymbol{z}}$ *be the vector computed in step* IV *of Algorithm* Getvec *in section* 4 *using* $\hat{\lambda}$, $\hat{N}_r$, $\hat{D}_r$, *and twist index* r. *Let* $\bar{L}$ *and* $\bar{D}$ *be the perturbations of* L *and* D *determined by the error analysis of section* 5, *and let* $(\bar{\lambda}, \bar{\boldsymbol{v}})$ *be the eigenpair of* $\bar{L}\bar{D}\bar{L}^t$ *with* $\bar{\lambda}$ *the closest eigenvalue to* $\hat{\lambda}$, *and* $\|\bar{\boldsymbol{v}}\| = 1$. *Let* ε *denote the machine precision, and for convenience, let* $\varepsilon_* := (1+\varepsilon)^{6n-1} - 1$. *If* ε *is small enough that* (38) *holds with* ε_* *instead of* $\bar{\varepsilon}_d$, *then*

$$|\sin\angle(\hat{\boldsymbol{z}}, \boldsymbol{v})| \le \frac{(1+\varepsilon)^{5(n-1)} - 1}{(1-\varepsilon)^{5(n-1)}} + \frac{|\bar{\lambda} - \hat{\lambda}|}{|\bar{v}(r)|\text{gap}(\hat{\lambda})} + \frac{\varepsilon_* \text{relcond}(\boldsymbol{v})}{1-\varepsilon_*}, \tag{55}$$

where relcond$(\boldsymbol{v})$ *is as in* (39), *and*

$$\text{gap}(\hat{\lambda}) := \min\{|\hat{\lambda} - \bar{\mu}|,\ \bar{\lambda} \ne \bar{\mu} \in \textit{spectrum of } \bar{L}\bar{D}\bar{L}^t\}.$$

Proof. There are three terms in the upper bound on $\sin\angle(\hat{\boldsymbol{z}}, \boldsymbol{v})$ because we connect $\hat{\boldsymbol{z}}$ to $\boldsymbol{v}$ via two "ideal" vectors $\bar{\boldsymbol{z}}$, $\bar{\boldsymbol{v}}$ and each transition contributes a term: $\hat{\boldsymbol{z}} \to \bar{\boldsymbol{z}}$, $\bar{\boldsymbol{z}} \to \bar{\boldsymbol{v}}$, $\bar{\boldsymbol{v}} \to \boldsymbol{v}$; see Figure 6. Recall from Theorem 4 that the matrices $\bar{L}$, $\bar{D}$, $\bar{N}_r$, $\bar{D}_r$ depend on $\hat{\lambda}$ and were defined so that the equality

$$\bar{L}\bar{D}\bar{L}^t - \hat{\lambda}I = \bar{N}_r\bar{D}_r\bar{N}_r^t \tag{56}$$

holds. That was the culmination of the error analysis in section 5. Recall that $\bar{D}_r(r) = \bar{\gamma}_r$. Then $\bar{\boldsymbol{z}}$ is defined as the exact solution of

$$\bar{N}_r\bar{D}_r\bar{N}_r^t\bar{\boldsymbol{z}} = \boldsymbol{e}_r\bar{\gamma}_r, \tag{57}$$

with $\bar{z}(r) = 1$. First consider $\hat{\boldsymbol{z}}$ and $\bar{\boldsymbol{z}}$. Theorem 13 shows that each $\bar{z}(i)$ is of the form $\hat{z}(i)(1+\eta_i)$ and Corollary 14 proves that

$$|\sin\angle(\hat{\boldsymbol{z}}, \bar{\boldsymbol{z}})| < \frac{(1+\varepsilon)^{5(n-1)} - 1}{(1-\varepsilon)^{5(n-1)}}. \tag{58}$$

Next consider $\bar{\boldsymbol{z}}$ and $\bar{\boldsymbol{v}}$. Combine (56) and (57) and then invoke Theorem 11 in section 6.2 to find that

$$\frac{\|(\bar{L}\bar{D}\bar{L}^t - \hat{\lambda}I)\bar{\boldsymbol{z}}\|}{\|\bar{\boldsymbol{z}}\|} = \frac{|\bar{\gamma}_r|}{\|\bar{\boldsymbol{z}}\|} \le \frac{|\bar{\lambda} - \hat{\lambda}|}{|\bar{v}(r)|}.$$

So by Theorem 1 (the gap theorem),

$$|\sin\angle(\bar{\boldsymbol{v}}, \bar{\boldsymbol{z}})| \le \frac{|\bar{\lambda} - \hat{\lambda}|}{|\bar{v}(r)|\mathrm{gap}(\hat{\lambda})}. \tag{59}$$

Finally consider $\bar{\boldsymbol{v}}$ and $\boldsymbol{v}$. The left side of Figure 6 indicates that $\bar{\boldsymbol{v}}$ and $\boldsymbol{v}$ are related through the matrix perturbations given in section 6.1 (see Lemma 5):

$$LDL^t \longrightarrow \bar{L}\bar{D}\bar{L}^t = E^{-1}LEFDFEL^tE^{-1}.$$

Theorem 4 bounded the entries in the specific matrices E and F and, by Lemma 6 in section 6.1.1,

$$\bar{\varepsilon}_d := \|(EF)^2 - I\| < \varepsilon_* := (1+\varepsilon)^{6n-1} - 1.$$

Thus Corollary 9 yields

$$|\sin\angle(\boldsymbol{v}, \bar{\boldsymbol{v}})| \le \frac{\mathrm{relcond}(\boldsymbol{v})\varepsilon_*}{1-\varepsilon_*}. \tag{60}$$

Add (58), (59), and (60) to obtain the theorem's bound on $|\sin\angle(\hat{\boldsymbol{z}}, \boldsymbol{v})|$. ∎

Next we discuss the implications of Theorem 15 for computing numerically orthogonal eigenvectors from Algorithm Getvec. The first term is essentially $5n\varepsilon$ and the last is $\mathrm{relcond}(\boldsymbol{v})6n\varepsilon$, and we are concerned only with cases when $\mathrm{relcond}(\boldsymbol{v})$ is $O(1)$. The middle term is the delicate one. If we bound each term separately, we would have $|\hat{\lambda} - \bar{\lambda}| \le Kn\varepsilon|\lambda|$, $1/|v(r)| \le \sqrt{n}$, and $\mathrm{relgap}(\hat{\lambda}) \ge tol$, giving a bound that exceeds $O(n\varepsilon)$. However for symmetric tridiagonal matrices the three terms are not independent. Moreover Getvec is often invoked for small isolated eigenvalues that have very large $\mathrm{relgap}(\lambda)$. For example, let us consider the extreme example introduced by

Demmel and Kahan in [7] that shows the bound of $Kn\varepsilon$ on the change in eigenvalue is attainable. The matrix is LL^t, L bidiagonal, with $L_{ii} = 1$ and $L_{i+1,i} = \beta \gg 1$. Think of $\beta = 10$. Small relative changes of $1+\varepsilon$ to the off-diagonals and $1-\varepsilon$ to the diagonal entries change $\lambda_{min} = \lambda_1 \approx \beta^{2(1-n)}$ by $2(2n-1)\varepsilon|\lambda| < 4n\varepsilon|\lambda|$. Only the smallest eigenvalue suffers this degree of sensitivity but $\mathrm{relgap}(\lambda_1) \approx \beta^2 n - 1$ while $|v(r)| \approx 1$. In this case the middle term in (55) is negligible compared to the other two terms. In fact, the corresponding eigenvector $\boldsymbol{v}$ decays very rapidly and very low accuracy in the eigenvalue (correct exponent) is sufficient to produce a very good eigenvector. All the other eigenvalues of this example are clustered in $[\beta^2 - 2/\beta, \beta^2 + 1]$ and so their eigenvectors should be calculated from a factorization of $LL^t - (\beta^2+1)I$, not from LL^t (see Algorithm MR^3 in [10]).

In general this middle term warrants further study, but we must recall from Theorem 11 that part of the middle term is a bound on the quantity $|\gamma_r|/\|\boldsymbol{z}\|$, and in practice, we have good approximations (up to order of magnitude) on it as well as on $\mathrm{gap}(\lambda)$. So any situation in which the middle term is too large is detectable. The algorithm monitors this term before accepting an eigenvector.

The reader may have noticed that the bound (55) contains quantities from both the factorizations LDL^t and $\bar{L}\bar{D}\bar{L}^t$; for example $\mathrm{gap}(\hat{\lambda})$ in the middle term is with respect to the eigenvalues of $\bar{L}\bar{D}\bar{L}^t$. However, the entries in L and $\bar{L}$ differ by at most 3 ulps and those of D and $\bar{D}$ by at most 4 ulps. Our application is only to well-conditioned eigenpairs, and so such λ and $\bar{\lambda}$ will differ only by a few ulps and the computed $\hat{\lambda}$ must be a good approximation to each of them. We feel that it is satisfactory to present our results in this form.

The following corollary summarizes a typical situation in which Algorithm Getvec is invoked.

COROLLARY 16. *In addition to the assumptions of Theorem* 15 *suppose that* (i) r *is such that* $\bar{v}(r) \geq 1/\sqrt{n}$, (ii) $\hat{\lambda}$ *is computed to satisfy* $|\hat{\lambda} - \bar{\lambda}|/|\hat{\lambda}| \leq K\varepsilon$, (iii) $\mathrm{relgap}(\hat{\lambda})$ *exceeds* 2^{-8}, *and* (iv) $\mathrm{relcond}(\boldsymbol{v}) \leq M/\mathrm{relgap}(\hat{\lambda})$. *Then*

$$|\sin\angle(\hat{\boldsymbol{z}}, \boldsymbol{v})| \leq 5n\varepsilon + 2^8 K\sqrt{n}\varepsilon + 2^8 M\varepsilon + O(\varepsilon^2). \qquad \square$$

8. Numerical examples. We first compare and contrast the behavior of Algorithm Getvec on two 3×3 tridiagonals. These aptly illustrate various aspects of the theory.

Example 3. First consider the matrix

$$T_0 = \begin{bmatrix} 1 & \sqrt{\varepsilon} & 0 \\ \sqrt{\varepsilon} & 7\varepsilon/4 & \varepsilon/4 \\ 0 & \varepsilon/4 & 3\varepsilon/4 \end{bmatrix},$$

where ε is the machine precision ($\varepsilon \approx 2.2 \times 10^{-16}$ in IEEE double precision). The eigenvalues of T_0 are

$$\lambda_1 = \varepsilon/2 + O(\varepsilon^2), \quad \lambda_2 = \varepsilon + O(\varepsilon^2), \quad \lambda_3 = 1 + \varepsilon + O(\varepsilon^2),$$

while the corresponding normalized eigenvectors are

$$\boldsymbol{v}_1 = \begin{bmatrix} -\sqrt{\varepsilon/2} + O(\varepsilon^{3/2}) \\ \frac{1}{\sqrt{2}}(1 + \frac{\varepsilon}{4}) + O(\varepsilon^2) \\ -\frac{1}{\sqrt{2}}(1 - \frac{3\varepsilon}{4}) + O(\varepsilon^2) \end{bmatrix}, \qquad \boldsymbol{v}_2 = \begin{bmatrix} -\sqrt{\varepsilon/2} + O(\varepsilon^{3/2}) \\ \frac{1}{\sqrt{2}}(1 - \frac{5\varepsilon}{4}) + O(\varepsilon^2) \\ \frac{1}{\sqrt{2}}(1 + \frac{3\varepsilon}{4}) + O(\varepsilon^2) \end{bmatrix},$$

$$\boldsymbol{v}_3 = \begin{bmatrix} 1 - \frac{\varepsilon}{2} + O(\varepsilon^2) \\ \sqrt{\varepsilon} + O(\varepsilon^{3/2}) \\ \frac{\varepsilon^{3/2}}{4} + O(\varepsilon^{5/2}) \end{bmatrix}.$$

The exact triangular factorization is given by $T_0 = L_0^{exact} D_0^{exact} (L_0^{exact})^t$, where

$$L_0^{exact} = \begin{bmatrix} 1 & 0 & 0 \\ \sqrt{\varepsilon} & 1 & 0 \\ 0 & 1/3 & 1 \end{bmatrix} \quad \text{and} \quad D_0^{exact} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 3\varepsilon/4 & 0 \\ 0 & 0 & 2\varepsilon/3 \end{bmatrix}.$$

When applying Algorithm Getvec to the above matrix, we observe the following.

1. The factorization computed in IEEE double precision arithmetic, $L_0 D_0 L_0^t$, turns out to be exact, i.e., $L_0 = L_0^{exact}$ and $D_0 = D_0^{exact}$.
2. The computed eigenvalues $\hat{\lambda}_i$ satisfy
$$|\hat{\lambda}_i - \lambda_i| \le 2\varepsilon |\hat{\lambda}_i|, \qquad 1 \le i \le 3.$$
3. For each $\hat{\lambda}_i$, $\gamma_k^{(i)}$ can be computed by applying steps I–III of Algorithm Getvec. The computed values are
$$\gamma^{(1)} = \begin{bmatrix} 1.11 \cdot 10^{-16} \\ 2.46 \cdot 10^{-32} \\ 2.46 \cdot 10^{-32} \end{bmatrix}, \quad \gamma^{(2)} = \begin{bmatrix} 2.22 \cdot 10^{-16} \\ 4.93 \cdot 10^{-32} \\ 4.93 \cdot 10^{-32} \end{bmatrix}, \quad \gamma^{(3)} = \begin{bmatrix} 4.44 \cdot 10^{-16} \\ -2.00 \\ -1.00 \end{bmatrix}.$$
Algorithm Getvec chooses $r = 2$ for $\hat{\lambda}_1$, $r = 2$ for $\hat{\lambda}_2$, and $r = 1$ for $\hat{\lambda}_3$. Note that for the first two eigenvalues $|\gamma_r| = O(\varepsilon^2) = O(\varepsilon|\lambda_i|) \ll \varepsilon\|T_0\|$.
4. The eigenvectors $\hat{\boldsymbol{v}}_i$ computed by Algorithm Getvec are such that
$$\max |\hat{\boldsymbol{v}}_i^t \hat{\boldsymbol{v}}_j| = 1.66 \cdot 10^{-16} < \varepsilon, \qquad 1 \le i \le 3, \;\; 1 \le j < i,$$
$$\max \frac{|\hat{\boldsymbol{v}}_i(k) - \boldsymbol{v}_i(k)|}{|\boldsymbol{v}_i(k)|} = 8.88 \cdot 10^{-16} < 4\varepsilon, \qquad 1 \le i \le 3, \;\; 1 \le k \le 3.$$
Amazingly each eigenvector entry is computed to high relative accuracy, even the tiny $v_3(3)$ entry.
5. Instead of Algorithm Getvec, we can use one step of inverse iteration,
$$(L_0 D_0 L_0^t - \hat{\lambda}_i I)\, \boldsymbol{x}_i \;=\; \text{random vector},$$
to compute the eigenvectors. These computed vectors also turn out to be accurate and numerically orthogonal (however, the tiny $v_3(3)$ entry is not computed to high relative accuracy). Note that the analysis of section 7 does not extend to random right-hand sides.
6. Both $|\gamma_2^{(3)}|$ and $|\gamma_3^{(3)}|$ are large while the corresponding eigenvector entries are $O(\sqrt{\varepsilon})$ and $O(\varepsilon^{3/2})$, respectively. Thus the numerical support of an eigenvector cannot solely be determined by the magnitudes of γ_i, and illustrates our comments at the end of Remark 6 in section 4. ∎

Example 4. The above matrix T_0 is a "benign" example. The following, also discussed in section 3, is a harder case:

$$T_1 = \begin{bmatrix} 1-\sqrt{\varepsilon} & \varepsilon^{1/4}\sqrt{1-7\varepsilon/4} & 0 \\ \varepsilon^{1/4}\sqrt{1-7\varepsilon/4} & \sqrt{\varepsilon}+7\varepsilon/4 & \varepsilon/4 \\ 0 & \varepsilon/4 & 3\varepsilon/4 \end{bmatrix}.$$

The eigenvalues of T_1 are

$$\lambda_1 = \frac{\varepsilon}{2} + \frac{\varepsilon^{3/2}}{8} + O(\varepsilon^2), \quad \lambda_2 = \varepsilon - \frac{\varepsilon^{3/2}}{8} + O(\varepsilon^2), \quad \lambda_3 = 1 + \varepsilon + O(\varepsilon^2),$$

while the corresponding normalized eigenvectors are

$$\boldsymbol{v}_1 = \begin{bmatrix} \frac{\varepsilon^{1/4}}{\sqrt{2}}(1+\frac{\sqrt{\varepsilon}}{2}) + O(\varepsilon^{5/4}) \\ -\frac{1}{\sqrt{2}}(1-\frac{\sqrt{\varepsilon}}{2}) + O(\varepsilon) \\ \frac{1}{\sqrt{2}}(1-\frac{3\varepsilon}{4}) + O(\varepsilon^{3/2}) \end{bmatrix}, \qquad \boldsymbol{v}_2 = \begin{bmatrix} \frac{\varepsilon^{1/4}}{\sqrt{2}}(1+\frac{\sqrt{\varepsilon}}{2}) + O(\varepsilon^{5/4}) \\ -\frac{1}{\sqrt{2}}(1-\frac{\sqrt{\varepsilon}}{2}) + O(\varepsilon) \\ -\frac{1}{\sqrt{2}}(1+\frac{3\varepsilon}{4}) + O(\varepsilon^{3/2}) \end{bmatrix},$$

$$\boldsymbol{v}_3 = \begin{bmatrix} 1-\frac{\sqrt{\varepsilon}}{2} + O(\varepsilon) \\ \varepsilon^{1/4}(1+\frac{\sqrt{\varepsilon}}{2}) + O(\varepsilon^{5/4}) \\ \frac{\varepsilon^{5/4}}{4}(1+\frac{\sqrt{\varepsilon}}{2}) + O(\varepsilon^{9/4}) \end{bmatrix}.$$

In exact arithmetic, $T_1 = L_1^{exact} D_1^{exact} (L_1^{exact})^t$, where

$$L_1^{exact} = \begin{bmatrix} 1 & 0 & 0 \\ \frac{\varepsilon^{1/4}\sqrt{1-7\varepsilon/4}}{1-\sqrt{\varepsilon}} & 1 & 0 \\ 0 & \frac{1-\sqrt{\varepsilon}}{3} & 1 \end{bmatrix} \quad \text{and}$$

$$D_1^{exact} = \begin{bmatrix} 1-\sqrt{\varepsilon} & 0 & 0 \\ 0 & \frac{3\varepsilon}{4(1-\sqrt{\varepsilon})} & 0 \\ 0 & 0 & \frac{\varepsilon(8+\sqrt{\varepsilon})}{12} \end{bmatrix}.$$

On this example, Algorithm Getvec behaves quite differently than on T_0 from Example 3:

1. The computed factorization $L_1 D_1 L_1^t$ does not have high relative accuracy. The relative errors in $L_1(2), D_1(2)$, and $D_1(3)$ are as large as $4.97 \cdot 10^{-9}$.
2. Consequently, some of the computed eigenvalues $\hat{\lambda}_i$ do not have high relative accuracy with respect to the eigenvalues of T_1. In particular,

   $$|\hat{\lambda}_i - \lambda_i| \approx 10^{-9} |\hat{\lambda}_i| \quad \text{for } i = 1, 2.$$

   Unlike λ_1 and λ_2, the third eigenvalue λ_3 is computed to high relative accuracy, i.e., $|\hat{\lambda}_3 - \lambda_3| = O(\varepsilon)$. However, the important point is that all the $\hat{\lambda}_i$ have high relative accuracy with respect to the eigenvalues of $L_1 D_1 L_1^t$.
3. The values of $\gamma_k^{(i)}$ computed by steps I–III of Algorithm Getvec are

   $$\gamma^{(1)} = \begin{bmatrix} -4.13 \cdot 10^{-24} \\ -7.40 \cdot 10^{-32} \\ -9.86 \cdot 10^{-32} \end{bmatrix}, \qquad \gamma^{(2)} = \begin{bmatrix} -6.62 \cdot 10^{-24} \\ -9.86 \cdot 10^{-32} \\ -9.86 \cdot 10^{-32} \end{bmatrix},$$

   $$\gamma^{(3)} = \begin{bmatrix} 2.22 \cdot 10^{-16} \\ 1.49 \cdot 10^{-8} \\ -1.00 \end{bmatrix}.$$

Algorithm Getvec chooses $r = 2$ for $\hat{\lambda}_1$, $r = 2$ for $\hat{\lambda}_2$, and $r = 1$ for $\hat{\lambda}_3$. Note that for the first two eigenvalues $|\gamma_r| = O(\varepsilon^2) \ll \varepsilon\|T\|$.

4. The eigenvectors $\hat{\boldsymbol{v}}_i$ computed in step IV of Algorithm Getvec are numerically orthogonal, i.e.,

$$\max |\hat{\boldsymbol{v}}_i^t \hat{\boldsymbol{v}}_j| \;=\; 5.55 \cdot 10^{-17} < \varepsilon, \qquad 1 \le i \le 3,\;\; 1 \le j < i.$$

But as in the case of the computed eigenvalues, the relative errors in the computed eigenvectors (with respect to the eigenvectors of T_1) are much larger than $O(\varepsilon)$, i.e.,

$$\max \frac{|\hat{\boldsymbol{v}}_i(k) - \boldsymbol{v}_i(k)|}{|\boldsymbol{v}_i(k)|} \;=\; 3.72 \cdot 10^{-9}, \qquad 1 \le i \le 2,\;\; 1 \le k \le 3.$$

All components of the third eigenvector $\boldsymbol{v}_3$ are computed to high relative accuracy.

5. The inverse iteration step

$$\begin{aligned} L_1 D_1 L_1^t - \hat{\lambda}_i I &= L_+ D_+ L_+^t, \\ L_+ D_+ L_+^t \, \boldsymbol{x}_i &= \text{random vector} \end{aligned} \tag{61}$$

also leads to computed eigenvectors that are numerically orthogonal when the dstqds transform is used to compute (61). From our experience, the use of a twisted factorization in Algorithm Getvec does not appear to be essential in practice; inverse iteration using dstqds also works well. However, twisted factorizations are more elegant to use, have better numerical behavior, and allow us to prove the accuracy of our algorithm.

6. When the diagonal and off-diagonal elements of T_1 are directly used to compute eigenvalues and eigenvectors (either by using inverse iteration or twisted factorizations as in Algorithm Getvec), the dot products between the computed eigenvectors are as large as 10^{-8}. See Example 1 in section 3 for an explanation of this failure. Thus the use of $L_1 D_1 L_1^t$ is essential for achieving numerical orthogonality in this case. ∎

The above example beautifully illustrates our techniques. We do not promise high relative accuracy for eigenvalues and eigenvectors of the given tridiagonal matrix. In fact, it is unrealistic to hope for such accuracy as explained in section 3. However, we get a "good" factorization of the tridiagonal and then proceed to compute its eigenvalues and eigenvectors to high accuracy, which automatically leads to orthogonality.

Example 5. Our next example is

$$T_2 = \begin{bmatrix} .520000005885958 & .519230209355285 & & \\ .519230209355285 & .589792290767499 & .36719192898916 & \\ & .36719192898916 & 1.89020772569828 & 2.7632618547882 \cdot 10^{-8} \\ & & 2.7632618547882 \cdot 10^{-8} & 1.00000002235174 \end{bmatrix}$$

with eigenvalues

$$\lambda_1 \approx \varepsilon, \quad \lambda_2 \approx 1 + \sqrt{\varepsilon}, \quad \lambda_3 \approx 1 + 2\sqrt{\varepsilon}, \quad \lambda_4 \approx 2.0.$$

Note that the interior eigenvalues have $\text{relgap}(\lambda_i) = O(\sqrt{\varepsilon})$. When we apply Algorithm Getvec to the LDL^t factorization of T_2, the corresponding computed eigenvectors have

$$|\hat{\boldsymbol{v}}_2^t \hat{\boldsymbol{v}}_3| \;=\; 1.12 \cdot 10^{-8} \;=\; O(\sqrt{\varepsilon}).$$

TABLE 1
Timing comparisons for computing all eigenvalues and eigenvectors.

Matrix type	Matrix size	Time taken (in seconds)				
		Lapack DSTEBZ +DSTEIN	DSTEBZ + Eispack TINVIT	Lapack DSTEDC	Lapack DSTEQR	Lapack DLASQ1 +Algorithm Getvec
Arithmetic progression (ε apart)	125	0.20 (.09+.11)	0.14 (.09+.05)	0.01	0.13	**0.05** (.01+.04)
	250	1.12 (.32+.80)	0.67 (.32+.35)	0.03	0.98	**0.08** (.02+.06)
	500	7.81 (1.25+6.06)	4.17 (1.25+2.92)	0.20	7.46	**0.39** (.11+.28)
	1000	93.97 (4.87+89.10)	39.44 (4.87+34.57)	1.22	74.33	**1.19** (.36+.83)
	2000	839.5 (20.9+818.6)	343.9 (20.9+323.0)	6.03	913.7	**4.50** (1.5+3.0)
Uniform distribution (ε to 1)	125	0.11 (.08+.03)	0.10 (.08+.02)	0.05	0.13	**0.05** (.01+.05)
	250	0.45 (.33+.12)	0.38 (.33+.05)	0.24	0.99	**0.11** (.04+.07)
	500	1.74 (1.24+.50)	1.47 (1.24+.23)	1.51	7.51	**0.31** (.11+.20)
	1000	93.37 (4.87+88.50)	5.68 (4.87+.81)	10.47	74.07	**1.16** (.40+.76)
	2000	839.5 (20.9+818.6)	344.8 (20.9+323.8)	159.3	648.2	**4.57** (1.6+3.0)
Uniform distribution (ε to 1 with random signs)	125	0.11 (.08+.03)	0.09 (.08+.01)	0.05	0.13	**0.05** (.01+.04)
	250	0.45 (.33+.12)	0.39 (.33+.06)	0.25	0.96	**0.13** (.04+.09)
	500	1.78 (1.27+.51)	1.50 (1.27+.23)	1.51	7.40	**0.37** (.09+.28)
	1000	7.21 (4.96+2.25)	5.89 (4.96+.93)	10.25	68.85	**1.51** (.37+1.14)
	2000	31.40 (21.3+10.1)	25.20 (21.3+4.10)	160.8	955.6	**5.10** (1.5+3.6)
(1,2,1) Matrix	125	0.12 (.09+.03)	0.10 (.09+.01)	0.05	0.13	**0.04** (.01+.03)
	250	0.44 (.32+.12)	0.37 (.32+.05)	0.16	0.92	**0.08** (.03+.05)
	500	1.85 (1.24+.61)	1.49 (1.24+.25)	1.02	7.01	**0.39** (.09+.30)
	1000	12.38 (4.88+7.50)	7.06 (4.88+2.18)	7.26	71.75	**1.22** (.32+.90)
	2000	840.0 (21.0+819.0)	128.8 (21.0+107.8)	105.8	678.7	**4.71** (1.6+3.1)
Biphenyl	966	77.61 (4.6+73.0)	33.02 (4.6+28.4)	7.66	73.96	**1.33** (.3+1.03)

As discussed in Remark 8 in section 4, inverse iteration appears to be a natural remedy to cure the problem. However, even after ten inverse iteration steps

$$|\hat{\boldsymbol{v}}_2^t \hat{\boldsymbol{v}}_3| \;=\; 3.45 \cdot 10^{-9} \;=\; O(\sqrt{\varepsilon}).$$

Thus the simple approach of using multiple inverse iteration steps does not lead to numerical orthogonality, as explained in Remark 8. For an approach that can achieve orthogonality in this situation, the reader is referred to [10]; also see Chapter 5 in [9]. □

8.1. Timing comparisons. Algorithm Getvec can lead to substantial speedups over earlier LAPACK software[2] to compute eigenvectors when the relative gaps between eigenvalues exceed *tol* ($= 10^{-3}$) but the absolute gaps are smaller. We illustrate this speedup on various examples in Table 1. Matrices of the first type have eigenvalues in an arithmetic progression,

$$\lambda_i = i \cdot \varepsilon, \quad i = 1, 2, \ldots, n-1, \;\text{ and }\; \lambda_n = 1.$$

The second type has eigenvalues that come from a uniform random distribution in the interval $[\varepsilon, 1]$, while the third type has a similar eigenvalue distribution as the second

[2]Since we first wrote this paper, our software has been incorporated in the latest release of LAPACK, where Algorithm Getvec appears as subroutine DLAR1V.

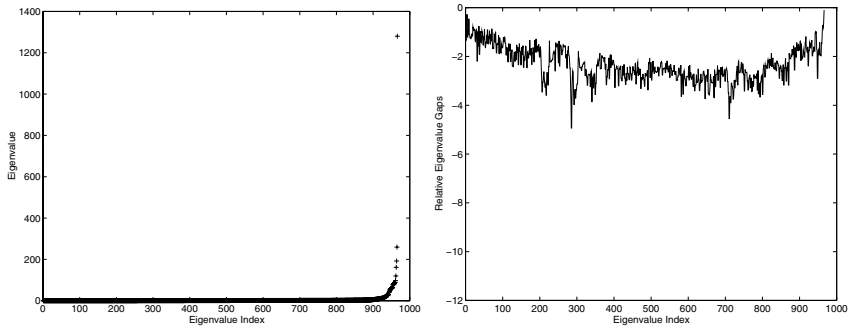

FIG. 7. *Left, eigenvalue distribution; right, relative gaps for the* Biphenyl *matrix.*

type except random $\pm$ signs are placed on the eigenvalues. The fourth type are the Toeplitz tridiagonal matrices with 2's on the diagonals and 1's as the off-diagonal elements, with eigenvalues $\lambda_i = 4\sin^2[\pi i/(2(n+1))]$. The final example comes from a real application in computational quantum chemistry—more specifically it arises in the modeling of the biphenyl molecule using Møller–Plesset theory [9]. Most of the eigenvalues of this positive definite 966×966 Biphenyl matrix are small compared to its norm. See Figure 7 for a plot of the eigenvalues and their relative gaps.

In Table 1 we compare the speed of Algorithm Getvec to various existing algorithms. In our implementation, we factor $T = LDL^t$ and then use the dqds software in LAPACK (subroutine DLASQ1) to compute all eigenvalues of LDL^t to high relative accuracy before invoking Algorithm Getvec to compute all eigenvectors. DSTEBZ is the bisection routine in LAPACK, while DSTEIN and TINVIT are inverse iteration routines from LAPACK and EISPACK, respectively, that perform Gram–Schmidt orthogonalization when eigenvalues have small absolute gaps, in particular, when $|\lambda_{i+1} - \lambda_i| \leq 10^{-3}\|T\|$ (actually TINVIT uses $\max_i |T_{i,i}| + |T_{i,i+1}|$ instead of $\|T\|$ while DSTEIN uses the 1-norm of T). DSTEQR uses the QR iteration to compute orthogonal eigenvectors [19], while DSTEDC is the divide and conquer code in LAPACK [21]. The QR algorithm and divide and conquer method compute eigenvalues and eigenvectors simultaneously, while our strategy and the inverse iteration routines first find the eigenvalues and then the eigenvectors—for these cases Table 1 gives the breakup of the time needed to compute eigenvalues as compared to eigenvectors.

The QR code DSTEQR always takes $O(n^3)$ time irrespective of the eigenvalue distribution. Due to the orthogonalization criterion, DSTEIN and TINVIT take $O(n^3)$ time on matrices of type 1 and the Biphenyl matrix and on large matrices of type 2 and 4. Table 1 shows that on these examples, Algorithm Getvec can be about two orders of magnitude faster than DSTEIN, TINVIT, and DSTEQR. Even on matrices where DSTEIN and TINVIT show $O(n^2)$ behavior, such as matrices of type 3 and matrices of type 2 and 4 with $n \leq 500$, Algorithm Getvec is generally faster. Also see that Algorithm Getvec is several times faster than DSTEDC on four of the five matrix types, and is comparable in speed on the first example, where DSTEDC is very fast due to deflation of clustered eigenvalues. The reader should observe the $O(n^2)$ behavior of Algorithm Getvec, whereas the other subroutines, in general, show an $O(n^3)$ behavior (all timings were measured using Fortran BLAS on a 333-MHz UltraSPARC processor with 1 GByte main memory). All algorithms delivered adequate numerical orthogonality on the test cases.

9. Singular vectors. A natural application of the procedures analyzed in this paper is to compute the SVD of a bidiagonal matrix L^t: $L^t = U\Sigma V^t$, $U^t = U^{-1}$, $V^t = V^{-1}$. Since $LL^t = V\Sigma^2 V^t$, the Cholesky factor of the symmetric positive definite matrix LL^t is the initial input and so the output of our method is V whose columns are the right singular vectors of L^t.

What must be done to compute U? The tempting formula

$$\begin{aligned} \boldsymbol{u} &= L^t\boldsymbol{v}/\sigma, \quad \sigma \neq 0, \\ \text{solve } \ L\boldsymbol{u} &= \mathbf{0}, \qquad\quad \sigma = 0, \end{aligned}$$

is well known to be treacherous. Orthogonal $\boldsymbol{v}$'s do not give rise to orthogonal $\boldsymbol{u}$'s because of the cancellation in forming $L^t\boldsymbol{v}$.

A better way is to invoke Algorithm Getvec again, as shown below. Note that a natural operation on bidiagonal and diagonal arrays is to "flip" them: $L \longrightarrow\sim L$. In practice the order of the entries is reversed. Formally

$$\sim L = \tilde{I}L^t\tilde{I},$$

where $\tilde{I}$ is the reversal matrix, $\tilde{I} = (\boldsymbol{e}_n, \ldots, \boldsymbol{e}_1)$ when $I = (\boldsymbol{e}_1, \ldots, \boldsymbol{e}_n)$. For diagonal matrices, flipping is just reversal. If cost were of no consequence, then U could be computed by flipping the given L^t, calling our algorithm, and reversing the output. The justification is that

$$\begin{aligned} (\sim L)(\sim L^t) &= (\tilde{I}L^t\tilde{I})(\tilde{I}L^t\tilde{I})^t \\ &= \tilde{I}L^tL\tilde{I} = \tilde{I}U\Sigma^2U^t\tilde{I}. \end{aligned}$$

The reversal mechanism needs to be applied locally. When an eigenvalue (σ^2) has been computed Algorithm Getvec invokes Algorithms 4.2 and 4.4 to obtain a double factorization and, after selecting an index, the desired singularity-revealing twisted factorization. From this comes the singular vector $\boldsymbol{v}$. In order to compute $\boldsymbol{u}$ it is only necessary to reverse L, apply Algorithms 4.2 and 4.4 again, select a possibly different index, and form the corresponding twisted factorization. Then Algorithm Getvec, in section 4, will yield $\{\tilde{I}\boldsymbol{u}\}$. In other words very little extra code is needed in order to compute $\boldsymbol{u}$ as well as $\boldsymbol{v}$.

However, even the use of Getvec outlined in the above paragraph is *not* adequate. It produces matrices U and V that are orthogonal to working precision, but the extra coupling relations $\|L^t\boldsymbol{v} - \boldsymbol{u}\sigma\| = O(\varepsilon\|L\|)$ and $\|L\boldsymbol{u} - \boldsymbol{v}\sigma\| = O(\varepsilon\|L\|)$ may fail when singular values are clustered.

In recent work [20], Großer and Lang have presented coupling relations that connect factorizations of $LL^t - \mu^2 I$ and $L^tL - \mu^2 I$. By forcing these relations to hold for the computed factorizations they found a way to use Algorithm Getvec and satisfy all the desired properties to working accuracy:

$$L^t\boldsymbol{v} - \boldsymbol{u}\sigma \approx \mathbf{0}, \quad L\boldsymbol{u} - \boldsymbol{v}\sigma \approx \mathbf{0}, \quad U^tU - I \approx 0, \quad V^tV - I \approx 0.$$

This algorithm is to become part of the LAPACK library.

Acknowledgment. We would like to thank an anonymous referee for an extraordinarily detailed reading of our original manuscript and for several constructive suggestions that, at the cost of considerable delay, greatly improved the presentation of this paper.

REFERENCES

[1] ANSI/IEEE, *IEEE Standard for Binary Floating Point Arithmetic*, Std 754-1985 ed., New York, 1985.

[2] J. Barlow, B. Parlett, and K. Veselić, eds., Linear Algebra Appl., 309, no. 1–3 (2000), pp. 1–361.

[3] A. W. Bojanczyk, R. Onn, and A. O. Steinhardt, *Existence of the hyperbolic singular value decomposition*, Linear Algebra Appl., 185 (1993), pp. 21–30.

[4] J. R. Bunch, *The weak and strong stability of algorithms in numerical linear algebra*, Linear Algebra Appl., 88/89 (1987), pp. 49–66.

[5] B. W. Char, K. O. Geddes, G. H. Gonnet, B. L. Leong, M. B. Monagan, and S. M. Watt, *Maple* V *Library Reference Manual*, Springer-Verlag, Berlin, 1991.

[6] L. S. DeJong, *Towards a formal definition of numerical stability*, Numer. Math., 28 (1977), pp. 211–220.

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

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

[9] I. S. Dhillon, *A New* $O(n^2)$ *Algorithm for the Symmetric Tridiagonal Eigenvalue/Eigenvector Problem*, Ph.D. thesis, Computer Science Division, University of California, Berkeley, California, 1997. Available as UC Berkeley Technical Report UCB//CSD-97-971.

[10] I. S. Dhillon and B. N. Parlett, *Multiple representations to compute orthogonal eigenvectors of symmetric tridiagonal matrices*, Linear Algebra Appl., to appear.

[11] S. Eisenstat and I. Ipsen, *Relative perturbation techniques for singular value problems*, SIAM J. Numer. Anal., 32 (1995), pp. 1972–1988.

[12] S. C. Eisenstat and I. C. F. Ipsen, *Relative perturbation bounds for eigenspaces and singular vector subspaces*, in Proceedings of the Fifth SIAM Conference on Applied Linear Algebra, J. G. Lewis, ed., SIAM, 1994, pp. 62–66.

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

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

[15] J. G. F. Francis, *The QR transformation, part* I, Comput. J., 4 (1961–62), pp. 265–271.

[16] J. G. F. Francis, *The QR transformation, part* II, Comput. J., 4 (1961–62), pp. 332–345.

[17] S. K. Godunov, A. G. Antonov, O. P. Kiriljuk, and V. I. Kostin, *Guaranteed Accuracy in Numerical Linear Algebra*, Kluwer Academic Publishers, Dordrecht, the Netherlands, 1993.

[18] S. K. Godunov, V. I. Kostin, and A. D. Mitchenko, *Computation of an eigenvector of symmetric tridiagonal matrices*, Siberian Math. J., 26 (1985), pp. 71–85.

[19] A. Greenbaum and J. Dongarra, *Experiments with QL/QR Methods for the Symmetric Tridiagonal Eigenproblem*, Computer Science Dept. Technical Report CS-89-92, University of Tennessee, Knoxville, 1989. (LAPACK Working Note 17, available electronically on netlib.)

[20] B. Grosser and B. Lang, *An* $O(n^2)$ *algorithm for the bidiagonal SVD*, Linear Algebra Appl., 358 (2002), pp. 45–70.

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

[22] P. Henrici, *The quotient-difference algorithm*, Nat. Bur. Standards Appl. Math. Series, 19 (1958), pp. 23–46.

[23] P. Henrici, *Applied and Computational Complex Analysis*, Vol. I, John Wiley & Sons, New York, 1974.

[24] N. J. Higham, *Accuracy and Stability of Numerical Algorithms*, 2nd ed., SIAM, Philadelphia, 2002.

[25] E. R. Jessup and I. Ipsen, *Improving the accuracy of inverse iteration*, SIAM J. Sci. Statist. Comput., 13 (1992), pp. 550–572.

[26] V. N. Kublanovskaya, *On some algorithms for the solution of the complete eigenvalue problem*, Zh. Vychisl. Mat., 1 (1961), pp. 555–570.

[27] R.-C. Li, *Relative perturbation theory:* II. *Eigenspace and singular subspace variations*, SIAM J. Matrix Anal. Appl., 20 (1998), pp. 471–492.

[28] B. Parlett, *The Symmetric Eigenvalue Problem*, Classics Appl. Math. 20, SIAM, Philadelphia, 1997.

[29] B. N. Parlett, *Invariant subspaces for tightly clustered eigenvalues of tridiagonals*, BIT, 36 (1996), pp. 542–562.

[30] B. N. Parlett, *Spectral sensitivity of products of bidiagonals*, Linear Algebra Appl., 275/276 (1998), pp. 417–431.
[31] 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.
[32] B. N. Parlett and I. S. Dhillon, *Relatively robust representations of symmetric tridiagonals*, Linear Algebra Appl., 309 (2000), pp. 121–151.
[33] B. N. Parlett, *A bidiagonal matrix determines its hyperbolic SVD to (varied) relative accuracy*, SIAM J. Matrix Anal. Appl., submitted.
[34] H. Rutishauser, *Der Quotienten-Differenzen-Algorithmus*. Z. Angew. Math. Phys., 5 (1954), pp. 223–251.
[35] H. Rutishauser, *Vorlesungen über numerische Mathematik*, Birkhäuser, Basel, 1976.
[36] H. Rutishauser, *Lectures on Numerical Mathematics*, Birkhäuser, Boston, 1990.
[37] S. Wolfram, *Mathematica: A System for Doing Mathematics by Computer*, 2nd ed., Addison-Wesley, Reading, MA, 1991.
[38] Y. Yang, *Error Analysis of the dqds Algorithm*, Ph.D. thesis, University of California, Berkeley, California, 1994.