Orthogonal Eigenvectors and Relative Gaps

Inderjit S. Dhillon * and Beresford N. Parlett †

January 29, 2002

Abstract

Let LDL^t be the triangular factorization of a real symmetric $n \times n$ tridiagonal matrix so that L is a unit lower bidiagonal matrix, D is diagonal. Let $(\lambda, \boldsymbol{v})$ be an eigenpair, $\lambda \neq 0$, with the property that both λ and $\boldsymbol{v}$ are determined to high relative accuracy by the parameters in L and D. Suppose also that the relative gap between λ and its nearest neighbor μ in the spectrum exceeds $1/n$; $n|\lambda - \mu| > |\lambda|$.

This paper presents a new $O(n)$ algorithm and a proof that, in the presence of round-off error, the algorithm computes an approximate eigenvector $\hat{\boldsymbol{v}}$ that is accurate to working precision: $|\sin \angle(\boldsymbol{v}, \hat{\boldsymbol{v}})| = O(n\varepsilon)$, where ε is the round-off unit. It follows that $\hat{\boldsymbol{v}}$ is numerically orthogonal to all the other eigenvectors. This result forms part of a program to compute numerically orthogonal eigenvectors without resorting to the Gram-Schmidt process.

The contents of this paper provide a high-level description and theoretical justification for LAPACK (version 3.0) subroutine DLAR1V.

*Department of Computer Sciences, University of Texas, Austin, TX 78712-1188, USA. Part of the author's research was supported while the author was at the University of California, Berkeley, by DARPA Contract No. DAAL03-91-C-0047, NSF Grant Nos. ASC-9313958 and CDA-9401156. At the University of Texas, this research is supported by a University startup grant and NSF Grant No. ACI-0093404.

†Mathematics Department and Computer Science Division, EECS Department, University of California, Berkeley, CA 94720, USA.

Contents

1 Setting the Scene

A real symmetric $n \times n$ 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 could say that the problem of computing orthogonal eigenvectors is solved. Unfortunately users are always in a hurry and the standard programs require $O(n^3)$ arithmetic operations in difficult cases. The time consuming calculation in the standard QR algorithm 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-based algorithm.

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 $O(n^2)$ algorithm to do the job. An expert will point out that it requires $(8/3)n^3$ operations to reduce a dense matrix to tridiagonal form so that an $O(n^2)$ algorithm is not possible. Nevertheless operation counts, though useful, are not a sure guide to execution time on current computers. Even with n exceeding 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 eigenpairs. So it seems desirable to seek a guaranteed $O(n^2)$ algorithm for T's eigenproblem.

It is the presence of parallel distributed memory computer systems that has vitalized the search for algorithms that can compute each eigenvector of a tridiagonal matrix independently of the others. Ideally the n eigenvalues would be distributed to n processors, along with a copy of the tridiagonal, and all n 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 useful step towards the goal. The main Theorem 9 in Section 7 shows that in special, but important, situations 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 $1/n$. 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 average $(\lambda_{max} - \lambda_{min})/(n-1)$. So our contribution is to change absolute to relative in the separation condition. Our examples show that the resulting speedups can be dramatic (from 822 seconds to 6 seconds). See Section 8 for details. 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. Properties of L and D allow us to compute eigenpairs of LDL^T very accurately.

The proof of the main Theorem 9 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 $\bar{L}\bar{D}\bar{L}^t - \lambda I = \tilde{N}\tilde{D}\tilde{N}^t$, where $\tilde{N}$ is a twisted factor defined in Section 4. The translation by λ 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 $\hat{N}$ and $\hat{D}$ (rather than 300 or 30000 units) suffice to give the exact relation.

Let us sketch our new sequential algorithm that 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 find the eigenvectors for all the shifted eigenvalues $\lambda - \tau$ that have large relative gaps. If some eigenvalues remain without eigenvectors then pick a new base τ_{new} at, or close to, one end of the remaining spectrum. Perform a careful factorization $L_{new}D_{new}L^t_{new} = LDL^t - \tau_{new}I$ and monitor element growth. If growth is too great then perturb τ (away from the cluster) until growth is acceptable. Then refine, to high relative accuracy, all new small eigenvalues with large relative gaps and compute their eigenvectors. Repeat the process with suitable bases τ until all eigenvectors have been computed. A more detailed outline of this algorithm is given in [9] and [10].

The organization of the paper is revealed in the list of contents. Householder notation (capital letters for matrices, Greek lower case for scalars, and lower case bold Roman for vectors) is generally followed. Eigenvalues are ordered by $\lambda_1 \le \lambda_2 \le \lambda_3 \le \cdots \le \lambda_n$. Section 4 is derived from Chapter 4 of [9].

2 Difficulties

The quality of an approximate eigenvector $\boldsymbol{y}$ is measured by its residual. The basic result that goes back to Temple in the 1930's, if not earlier, will be needed later. See [33, Chaps. 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 $\boldsymbol{y}$ and a scalar μ, closer to λ than to any other eigenvalue,*

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

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

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

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 $\boldsymbol{y}$ and $\boldsymbol{z}$ to $\boldsymbol{u}$ and $\boldsymbol{v}$ respectively (where $\boldsymbol{u}$ and $\boldsymbol{v}$ are eigenvectors), 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 roundoff unit, do ensure numerically 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, or even the best way, to compute numerically orthogonal eigenvectors. The QR algorithm does produce a numerically orthonormal basis for all the invariant subspaces that are well defined by the tridiagonal.

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

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

The average separation between eigenvalues is

$$\frac{\lambda_{max} - \lambda_{min}}{n-1} \tag{4}$$

and so, by (1) and (3), if $\mathrm{gap}(\mu)$ is above this average then

$$|\sin \angle(\boldsymbol{v}, \boldsymbol{y})| \leq (n-1)\varepsilon$$

and accuracy is assured. On the other hand in the many cases when $\mathrm{gap}(\mu) \ll (4)$ then the residual norm must be much smaller than the right hand side of (3) in order to deliver such accuracy.

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

$$\|\boldsymbol{r}\| = \|T\boldsymbol{y} - \boldsymbol{y}\hat{\lambda}\| \leq K\varepsilon|\hat{\lambda}|, \tag{5}$$

for some modest constant K independent of $\boldsymbol{y}$ and $\hat{\lambda}$, so that

$$|\sin \angle(\boldsymbol{v}, \boldsymbol{y})| \leq \frac{K\varepsilon|\hat{\lambda}|}{\mathrm{gap}(\hat{\lambda})} = \frac{K\varepsilon}{\mathrm{relgap}(\hat{\lambda})}. \tag{6}$$

Note that if $\hat{\lambda} = O(\varepsilon(\lambda_{max} - \lambda_{min}))$ then (5) requires $\|\boldsymbol{r}\| = O(\varepsilon^2)$. How is that possible since even the rounded version of the 'true' eigenvector may not achieve (5)?

We can achieve (5) in the presence of three separate properties.

(I) λ must be determined to high relative accuracy by the matrix parameters.

(II) The computed $\hat{\lambda}$ must approximate λ to high relative accuracy.

(III) The vector $\boldsymbol{y}$ must be computed so that $\|\boldsymbol{r}(\boldsymbol{y})\| \approx |\lambda - \hat{\lambda}| \approx \varepsilon|\hat{\lambda}|$.

A tridiagonal matrix T is traditionally represented by its diagonal and off-diagonal entries. We achieve Property I by discarding this representation in favor of $LDL^t = T - \tau I$ for a suitable shift τ. Section 3 shows the necessity for this change of representation. Property II is then easily achieved by using bisection or, in the positive definite case, by the dqds algorithm, see [13]. Given a factorization LDL^t, and a highly accurate $\hat{\lambda}$, 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 $T - \hat{\lambda} I$.

A subtle point in our analysis is that (5) 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 a proof 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. However for most 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 contrived 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]

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

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

while

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

[1]we carefully constructed this matrix to have the desired behavior which may be verified by using a symbol manipulator such as Maple [4] or Mathematica [40].

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 eigenvector components 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 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 eigenvalues to high relative accuracy. See [8, Theorem 5.13] for more details. Thus these factored forms are preferable to the standard form for eigenvalue calculations.

When D is not positive definite the situation is more complicated. Often LDL^t determines its eigenvalues to high relative accuracy, particularly the small ones. Of course we may use the representation $U_- D_- U_-^t$ derived from Gaussian elimination in reverse order or even a twisted factorization. The important point is that the positive definite case is not the only one in which some eigenvalues are determined to high relative accuracy by a factored form.

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

$$\mathrm{relcond}(\lambda) := \boldsymbol{v}^t L|D|L^t\boldsymbol{v}/|\lambda|.$$

Note that when D is positive definite then relcond$(\lambda) = 1$ but we do not need such stability for our results. A value of relcond(λ) such as 10 or 20 is adequate to ensure numerically orthogonal eigenvectors.

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 5 and [30] for more details.

4 Computation with Bidiagonals

In the remaining pages, we show that we can compute a very accurate eigenvector when (i) relcond(λ) is modest and (ii) λ has a large relative gap. Our algorithm achieves this by obtaining residual norms that are small in a relative sense. In this section, we first review twisted factorizations, and then present a novel "mixed" relative error analysis for the methods that compute them. This error analysis, given in Section 4.3, is essential for our results; indeed a "standard" backward error analysis turns out to be totally inadequate.

4.1 Twisted Factorizations

If $\hat{\lambda}$ is an extremely 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 $(T - \hat{\lambda}I)z \approx 0$, we seek a factorization that reveals this singularity. In the tridiagonal case we can always construct such a factorization from the forward and backward triangular factors. This procedure is described in [29] along with the necessary theory. For reference in later sections we quote here the results we need, without proof, and add a few comments and refinements.

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. 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 rank. A twisted factorization, written as

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

is constructed as follows. N_k and D_k are formed by factoring the matrix from top down and from bottom up meeting at row k. N_k takes rows $1:k$ of L_+ and rows $k:n$ of U_-. Thus row k has three nonzero entries

$$(l_{k-1}^+ \quad 1 \quad u_k^-)$$

and

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

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

The only new entry is γ_k and it is of great importance. There are several formulae for γ_k and we will give some of them in Fact 2.

Fact 1.

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

Our twisted factorization will reveal the rank if $\gamma_k \approx \lambda - \hat{\lambda}$. Fact 1 implies that, in cases of interest, there exists such a γ_k (see Theorem 2 below). The goal is to find an

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

Figure 1: Twisted Triangular Factor N_k with $n = 6$, $k = 3$.

appropriate index k and we do so by computing γ_k for every choice of k, $1 \leq k \leq n$, and then choosing an index which gives a minimal or nearly minimal value to $|\gamma_k|$. The surprise is that this can be done for little extra work as shown in Fact 2 below. 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, Sec. 3.3].

Fact 2. In exact arithmetic,

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

The expression in parentheses in the first formula above is the (k,k) entry of $LDL^t - \hat{\lambda}I$ (here $d_k = D(k,k)$ and $l_{k-1} = L(k, k-1)$). More robust expressions are given in (16).

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 rely on the tridiagonal form.

Let $LDL^t = V\Lambda V^t$. Replace LDL^t by $V\Lambda V^t$ in Fact 1 to find, for each k,

$$\frac{1}{\gamma_k} = \frac{|v_j(k)|^2}{\lambda_j - \hat{\lambda}} + \sum_{i \neq j} \frac{|v_i(k)|^2}{\lambda_i - \hat{\lambda}}, \tag{7}$$

where $\lambda = \lambda_j$ is the eigenvalue closest to $\hat{\lambda}$ and its normalized eigenvector is $\boldsymbol{v}_j$. Theorem 2 shows that the twist index k for which $|v_j(k)|$ is large leads to a small value of γ_k.

Theorem 2 *Let γ_k be as in (7), where $\hat{\lambda}$ approximates λ_j, and let λ_j be isolated enough, i.e.,*

$$\frac{|\lambda_j - \hat{\lambda}|}{gap(\hat{\lambda})} \leq \frac{1}{M} \cdot \frac{1}{n-1},$$

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

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

Proof. A proof is given in [9, Section 3.2]. □

Next we show how to exploit the twisted factorizations to compute an accurate approximate eigenvector. Let $\boldsymbol{z}^{(k)}$ be defined by $(LDL^t - \hat{\lambda}I)\boldsymbol{z}^{(k)} = \boldsymbol{e}_k\gamma_k$ where $I = [\boldsymbol{e}_1, \boldsymbol{e}_2, \ldots, \boldsymbol{e}_n]$

and $z^{(k)}(k) = 1$. Theorem 3 shows that $\boldsymbol{z}^{(k)}$ enjoys a small relative residual norm under suitable conditions and serves as an excellent approximation to the eigenvector $\boldsymbol{v}_j$ [15, 29]. Note that our approximate eigenvector $\boldsymbol{z}^{(k)}$ is a carefully chosen column of $(LDL^t - \hat{\lambda} I)^{-1}$.

Theorem 3 *Let γ_k be as in (7), where $\hat{\lambda}$ approximates λ_j, $\hat{\lambda} \neq \lambda_j$. Then, if $v_j(k) \neq 0$, the residual norm*

$$\frac{|\gamma_k|}{\|\boldsymbol{z}^{(k)}\|} \le \frac{|\lambda_j - \hat{\lambda}|}{|v_j(k)|},$$

and thus for at least one k,

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

Proof. A proof is given in [29, Section 5] and [9, Section 3.2], but we repeat it here for the sake of completeness. Recall that $LDL^t = V \Delta V^t$. Then

$$\begin{aligned} \boldsymbol{z}^{(k)} &= (LDL^t - \hat{\lambda} I)^{-1} \boldsymbol{e}_k \gamma_k, \\ \Rightarrow \|\boldsymbol{z}^{(k)}\|^2 &= |\gamma_k|^2 \boldsymbol{e}_k^T V (\Lambda - \hat{\lambda} I)^{-2} V^T \boldsymbol{e}_k, \\ &= |\gamma_k|^2 \sum_{i=1}^{n} \frac{|v_i(k)|^2}{|\hat{\lambda} - \lambda_i|^2}, \\ \Rightarrow \frac{|\gamma_k|}{\|\boldsymbol{z}^{(k)}\|} &\le \frac{|\lambda_j - \hat{\lambda}|}{|v_j(k)|}, \qquad \forall k. \end{aligned}$$

Noting that $|v_j(k)| \ge 1/\sqrt{n}$ 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 1 *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*

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

Proof. Write $\boldsymbol{z}$ for $\boldsymbol{z}^{(k)}$, 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

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

□

4.2 qd-like Recurrences

To find an individual eigenvector we need to know the $L_+D_+L_+^t$ and $U_-D_-U_-^t$ decompositions. Algorithm 4.1 given below implements the transformation

$$LDL^t - \mu I \;=\; L_+D_+L_+^t. \tag{8}$$

We call this the "**st**ationary **q**uotient-**d**ifference with **s**hift"(stqds) transformation 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], [36] and [37]. Although (8) 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 (8) 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 $UDU^t - \mu I$.

In the rest of the paper, we will 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.

Algorithm 4.1 (stqds)

$$\begin{aligned}
&D_+(1) \;:=\; d_1 - \mu \\
&\text{for } i = 1, n-1 \\
&\quad L_+(i) \;:=\; (d_i l_i)/D_+(i) && (9)\\
&\quad D_+(i+1) \;:=\; d_i l_i^2 + d_{i+1} - L_+(i) d_i l_i - \mu && (10)\\
&\text{end for}
\end{aligned}$$

We now see how to eliminate some of the additions and subtractions from the above algorithm. We introduce the intermediate variable

$$\begin{aligned}
s_{i+1} &= D_+(i+1) - d_{i+1}, \\
&= d_i l_i^2 - L_+(i) d_i l_i - \mu, \quad \text{by } (10) \\
&= L_+(i) l_i (D_+(i) - d_i) - \mu, \quad \text{by } (9) \\
&= L_+(i) l_i s_i - \mu. && (11)
\end{aligned}$$

Using this intermediate variable, we get the so-called *differential form* of the stationary qd transformation (dstqds). This term was again coined by Rutishauser in the context of similar transformations in [34], [36]. We will see later that the differential transformations play a crucial role in proving the main result of the paper, Theorem 9.

Algorithm 4.2 (dstqds)-*differential form* of the stationary qd transformation

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

In the next section we will show that the above differential algorithm has some nice properties in the face of roundoff errors.

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) transformation. The following algorithm gives an obvious way to implement this transformation.

Algorithm 4.3 (qds)

$$\begin{aligned}
&U_-(n) := 0\\
&\text{for } i = n-1, 1, -1\\
&\quad D_-(i+1) := d_i l_i^2 + d_{i+1} - U_-(i+1) d_{i+1} l_{i+1} - \mu && (12)\\
&\quad U_-(i) := (d_i l_i)/D_-(i+1) && (13)\\
&\text{end for}\\
&D_-(1) := d_1 - U_-(1) d_1 l_1 - \mu
\end{aligned}$$

As in the stationary transformation, we introduce the intermediate variable

$$\begin{aligned}
p_i &= D_-(i) - d_{i-1} l_{i-1}^2, && (14)\\
&= d_i - U_-(i) d_i l_i - \mu, \quad \text{by } (12)\\
&= \frac{d_i}{D_-(i+1)}(D_-(i+1) - d_i l_i^2) - \mu, \quad \text{by } (13)\\
&= \frac{d_i}{D_-(i+1)} \cdot p_{i+1} - \mu. && (15)
\end{aligned}$$

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

Algorithm 4.4 (dqds)-*differential form* of the progressive qd transformation

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

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. Since $(LDL^t)_{k,k+1} = d_k l_k$, Fact 2 in Section 4.1 leads to

$$\begin{aligned}
\gamma_k &= D_+(k) - \frac{(d_k l_k)^2}{D_-(k+1)}, \\
&= s_k + d_k - \frac{(d_k l_k)^2}{D_-(k+1)}, \quad \text{by (Algorithm 4.2)} \\
&= s_k + \frac{d_k}{D_-(k+1)}\left(D_-(k+1) - d_k l_k^2\right).
\end{aligned}$$

Substituting from (14), (15) and (11) in the above equation, we can express γ_k by any of the following formulae:

$$\gamma_k = \begin{cases} s_k + \frac{d_k}{D_-(k+1)} \cdot p_{k+1}, \\ s_k + p_k + \mu, \\ p_k + L_+(k-1) l_{k-1} s_{k-1}. \end{cases} \tag{16}$$

In the next section, we will see that the top and bottom formulae in (16) are 'better' for computational purposes. To reveal the near-singularity of $LDL^T - \mu I$, we choose r as the index where $|\gamma_k|$ is minimum. The twisted factorization at position r is given by

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

where $D_r = \operatorname{diag}(D_+(1), \ldots, D_+(r-1), \gamma_r, D_-(r+1), \ldots, D_-(n))$ and N_r is the corresponding twisted factor, see the beginning of Section 4.1. It may be formed by the following "**d**ifferential **tw**isted **q**uotient-**d**ifference with **s**hift"(dtwqds) transformation which is just the appropriate blend of Algorithms 4.2 and 4.4.

Algorithm 4.5 (dtwqds)

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

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.

In the next section, we exhibit desirable properties of the differential forms of our qd-like transformations in the face of roundoff errors. Before we do so, we emphasize that *the particular qd-like transformations presented in this section are new.* Similar qd recurrences have been studied by Rutishauser [34], [36] and [37], Henrici [20], [21, Chapter 7], Fernando and Parlett [13], and Yao Yang [41].

4.3 Roundoff Error Analysis

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 transformations introduced in Section 4.2 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 [41] 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*.

Notational Guide. In all results of this section, numbers in the computer are represented 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 $\overset{\frown}{Z}_+$, are introduced for our analysis but are typically unrepresentable in a computer's limited precision. Note that we have chosen the symbols $\rightarrow$ and $\frown$ in Figure 2 to indicate a process that takes rows and columns in increasing order, i.e., from "left to right" and "top to bottom". Later, in Figure 3 we use $\leftarrow$ and $\smile$ to indicate a "right to left" and "bottom to top" process.

$$\begin{tikzcd}[column sep=huge, row sep=huge]
Z \arrow[r, "\textbf{dstqds}", "\text{computed}"'] \arrow[d, "\begin{array}{l}\text{change each}\\ d_i \text{ by } 1\ ulp,\\ l_i \text{ by } 3\ ulps.\end{array}"'] & \hat{Z}_+ \\
\vec{Z} \arrow[r, "\textbf{dstqds}", "\text{exact}"'] & \overset{\frown}{Z}_+ \arrow[u, "\begin{array}{l}\text{change each}\\ \overset{\frown}{D}_+(i) \text{ by } 2\ ulps,\\ \overset{\frown}{L}_+(i) \text{ by } 3\ ulps.\end{array}"']
\end{tikzcd}$$

Figure 2: Effects of roundoff — **dstqds** transformation

Figure 2 states that the computed outputs of the dstqds transformation (see Algorithm 4.2), $\hat{D}_+(i)$ and $\hat{L}_+(i)$ are small relative perturbations of the quantities $\overset{\frown}{D}_+(i)$ and $\overset{\frown}{L}_+(i)$ which in turn are the results of an EXACT dstqds transformation 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 transformations (see 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 [3] and [6] but note that our perturbations are relative ones. A trustful reader may wish to skip the proofs but the very special 'interpretation' of the roundoff errors is the rock on which our results are founded.

Theorem 4 *Let the dstqds transformation 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 1 (3) ulps, while $\hat{D}_+(i)$ ($\hat{L}_+(i)$) differs from $\overset{\frown}{D}_+(i)$ ($\overset{\frown}{L}_+(i)$) by 2 (3) ulps.*

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

$$\begin{aligned}
\hat{D}_+(i) &= (\hat{s}_i + d_i)/(1+\varepsilon_+), \\
\hat{L}_+(i) &= d_i\, l_i (1+\varepsilon_*)(1+\varepsilon_/)/\hat{D}_+(i) \;=\; \frac{d_i\, l_i (1+\varepsilon_*)(1+\varepsilon_/)(1+\varepsilon_+)}{\hat{s}_i + d_i}, \\
\text{and}\quad \hat{s}_{i+1} &= \frac{\hat{L}_+(i)\, l_i \hat{s}_i (1+\varepsilon_\circ)(1+\varepsilon_{**}) - \mu}{1+\varepsilon_{i+1}}.
\end{aligned}$$

In the above, all ε's depend on i but we have chosen to single out the one that accounts 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 $\vec{d}_i$ and $\vec{l}_i$ so that the exact dstqds relation

$$\vec{s}_{i+1} \;=\; \frac{\vec{d}_i\, \vec{l}_i^{\,2}\, \vec{s}_i}{\vec{s}_i + \vec{d}_i} - \mu \tag{17}$$

$$\begin{tikzcd}[column sep=huge, row sep=huge]
Z \arrow[r, "\textbf{dqds}", "\text{computed}"'] \arrow[d, "\begin{array}{l}\text{change each}\\ d_i \text{ by } 3\ ulps,\\ l_i \text{ by } 3\ ulps.\end{array}"'] & \hat{Z}_- \\
\overleftarrow{Z} \arrow[r, "\textbf{dqds}", "\text{exact}"'] & \breve{Z}_- \arrow[u, "\begin{array}{l}\text{change each}\\ \breve{D}_-(i) \text{ by } 2\ ulps,\\ \breve{U}_-(i) \text{ by } 4\ ulps.\end{array}"']
\end{tikzcd}$$

Figure 3: Effects of roundoff — **dqds** transformation

is satisfied. This may be achieved by setting

$$\begin{aligned}
\overrightarrow{d}_i &= d_i(1+\varepsilon_i),\\
\overrightarrow{s}_i &= \hat{s}_i(1+\varepsilon_i),\\
\overrightarrow{l}_i &= l_i\sqrt{\frac{(1+\varepsilon_*)(1+\varepsilon_/)(1+\varepsilon_+)(1+\varepsilon_\circ)(1+\varepsilon_{**})}{1+\varepsilon_i}}.
\end{aligned} \tag{18}$$

In order to satisfy the exact mathematical relations of dstqds,

$$\widehat{D}_+(i) = \overrightarrow{s}_i + \overrightarrow{d}_i, \tag{19}$$

$$\widehat{L}_+(i) = \frac{\overrightarrow{d}_i\,\overrightarrow{l}_i}{\overrightarrow{s}_i + \overrightarrow{d}_i}, \tag{20}$$

we set

$$\begin{aligned}
\widehat{D}_+(i) &= \hat{D}_+(i)(1+\varepsilon_+)(1+\varepsilon_i),\\
\widehat{L}_+(i) &= \hat{L}_+(i)\sqrt{\frac{(1+\varepsilon_\circ)(1+\varepsilon_{**})}{(1+\varepsilon_*)(1+\varepsilon_/)(1+\varepsilon_+)(1+\varepsilon_i)}}
\end{aligned} \tag{21}$$

and the result holds. □

A similar result holds for the dqds transformation.

Theorem 5 *Let the dqds transformation be computed as in Algorithm 4.4. In the absence of overflow and underflow, the diagram in Figure 3 commutes and* $\overleftarrow{d}_i$ ($\overleftarrow{l}_i$) *differs from* d_i (l_i) *by* 3 (3) *ulps, while* $\hat{D}_-(i)$ ($\hat{U}_-(i)$) *differs from* $\breve{D}_-(i)$ ($\breve{U}_-(i)$) *by* 2 (4) *ulps.*

Proof. The proof is similar to that of Theorem 4. The computed quantities satisfy

$$
\begin{aligned}
\hat{D}_-(i+1) &= (d_i\, l_i^2(1+\varepsilon_*)(1+\varepsilon_{**}) + \hat{p}_{i+1})/(1+\varepsilon_+), && (22)\\
\hat{t} &= d_i(1+\varepsilon_/)/\hat{D}_-(i+1),\\
\hat{U}_-(i) &= l_i\hat{t}(1+\varepsilon_\circ) \;=\; \frac{d_i\, l_i(1+\varepsilon_/)(1+\varepsilon_\circ)(1+\varepsilon_+)}{d_i\, l_i^2(1+\varepsilon_*)(1+\varepsilon_{**}) + \hat{p}_{i+1}},\\
\hat{p}_i &= \frac{(d_i/\hat{D}_-(i+1))\hat{p}_{i+1}(1+\varepsilon_/)(1+\varepsilon_{\circ\circ}) - \mu}{1+\varepsilon_i},\\
\Rightarrow \quad (1+\varepsilon_i)\hat{p}_i &= \frac{d_i\, \hat{p}_{i+1}}{d_i\, l_i^2(1+\varepsilon_*)(1+\varepsilon_{**}) + \hat{p}_{i+1}}(1+\varepsilon_/)(1+\varepsilon_{\circ\circ})(1+\varepsilon_+) - \mu.
\end{aligned}
$$

Note that the above ε's are different from the ones in the proof of the earlier Theorem 4. As in Theorem 4, the trick is to satisfy the exact relation,

$$
\overleftarrow{p}_i = \frac{\overleftarrow{d}_i\, \overleftarrow{p}_{i+1}}{\overleftarrow{d}_i\, \overleftarrow{l}_i^{\,2} + \overleftarrow{p}_{i+1}} - \mu, \tag{23}
$$

which is achieved by setting

$$
\begin{aligned}
\overleftarrow{d}_i &= d_i(1+\varepsilon_/)(1+\varepsilon_{\circ\circ})(1+\varepsilon_+),\\
\overleftarrow{p}_i &= \hat{p}_i(1+\varepsilon_i), && (24)\\
\text{and} \quad \overleftarrow{l}_i &= l_i\sqrt{\frac{(1+\varepsilon_*)(1+\varepsilon_{**})(1+\varepsilon_{i+1})}{(1+\varepsilon_/)(1+\varepsilon_{\circ\circ})(1+\varepsilon_+)}}, && (25)\\
\text{so that} \quad \overleftarrow{d}_i\, \overleftarrow{l}_i^{\,2} &= d_i\, l_i^2(1+\varepsilon_*)(1+\varepsilon_{**})(1+\varepsilon_{i+1}).
\end{aligned}
$$

The other dqds relations,

$$
\begin{aligned}
\breve{D}_-(i+1) &= \overleftarrow{d}_i\, \overleftarrow{l}_i^{\,2} + \overleftarrow{p}_{i+1}, && (26)\\
\breve{U}_-(i) &= \frac{\overleftarrow{d}_i\, \overleftarrow{l}_i}{\overleftarrow{d}_i\, \overleftarrow{l}_i^{\,2} + \overleftarrow{p}_{i+1}}, && (27)
\end{aligned}
$$

may be satisfied by setting

$$
\begin{aligned}
\breve{D}_-(i+1) &= \hat{D}_-(i+1)(1+\varepsilon_+)(1+\varepsilon_{i+1}),\\
\breve{U}_-(i) &= \frac{\hat{U}_-(i)}{1+\varepsilon_\circ}\sqrt{\frac{(1+\varepsilon_*)(1+\varepsilon_{**})(1+\varepsilon_{\circ\circ})}{(1+\varepsilon_/)(1+\varepsilon_+)(1+\varepsilon_{i+1})}}. && (28)
\end{aligned}
$$

□

By combining parts of the analyses for the dstqds and dqds transformations, 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

$$
\begin{tikzcd}[column sep=huge, row sep=huge]
Z \arrow[r, "\textbf{dtwqds}", "\text{computed}"'] \arrow[d, "\begin{array}{l}\text{change each}\\ d_i \text{ by } 1 \; ulp,\ 1 \le i < k,\\ l_i \text{ by } 3 \; ulps,\ 1 \le i < k,\\ d_k \text{ by } 4 \; ulps,\ l_k \text{ by } 3\frac{1}{2} \; ulps,\\ d_i \text{ by } 3 \; ulps,\ k < i \le n,\\ l_i \text{ by } 3 \; ulps,\ k < i < n.\end{array}"'] & \hat{Z}_k \\
\bar{Z} \arrow[r, "\textbf{dtwqds}", "\text{exact}"'] & \tilde{Z}_k \arrow[u, "\begin{array}{l}\text{change each}\\ \overset{\frown}{D}_+(i) \text{ by } 2 \; ulps,\ 1 \le i < k,\\ \overset{\frown}{L}_+(i) \text{ by } 3 \; ulps,\ 1 \le i < k.\\ \tilde{\gamma}_k \text{ by } 2 \; ulps,\ \tilde{U}_-(k) \text{ by } 4\frac{1}{2} \; ulps,\\ \overset{\smile}{D}_-(i) \text{ by } 2 \; ulps,\ k < i \le n,\\ \overset{\smile}{U}_-(i) \text{ by } 4 \; ulps,\ k < i < n.\end{array}"']
\end{tikzcd}
$$

Figure 4: Effects of roundoff — **dtwqds** transformation

by "concatenating" the stationary and progressive factors. In particular, for any twist position k,

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

while

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

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

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

respectively (note that $\sim$ is a concatenation of the symbols $\frown$ and $\smile$, while $-$ may also be derived by concatenating $\leftarrow$ and $\rightarrow$).

Theorem 6 *Let the dtwqds transformation be computed as in Algorithm 4.5. In the absence of overflow and underflow, the diagram in Figure 4 commutes and $\overrightarrow{d}_i$ ($\overrightarrow{l}_i$) differs from d_i (l_i) by 1 (3) ulps for $1 \le i < k$, $\bar{d}_k$ ($\bar{l}_k$) differs from d_k (l_k) by 4 ($3\frac{1}{2}$) ulps, while $\overleftarrow{d}_i$ ($\overleftarrow{l}_i$) differs from d_i (l_i) by 3 (3) ulps for $k < i \le n$. On the output side, $\hat{D}_+(i)$ ($\hat{L}_+(i)$) differs from $\overset{\frown}{D}_+(i)$ ($\overset{\frown}{L}_+(i)$) by 2 (3) ulps for $1 \le i < k$, $\hat{\gamma}_k$ ($\tilde{U}_-(k)$) differs from $\tilde{\gamma}_k$ ($\tilde{U}_-(k)$) by 2 ($4\frac{1}{2}$) ulps, while $\hat{D}_-(i)$ ($\hat{U}_-(i)$) differs from $\overset{\smile}{D}_-(i)$ ($\overset{\smile}{U}_-(i)$) by 2 (4) ulps for $k < i \le n$.*

Proof. The crucial observation is that for the exact stationary transformation (i. e., (17), (19) and (20)) to be satisfied for $1 \le i \le k-1$, 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 transformation (i. e., (23),

(26) and (27)) to hold for $k+1 \le i < n$, roundoff errors need to be put only on the bottom part of the matrix, i.e., on $d_{k+1}, \dots, d_n$ and $l_{k+1}, \dots, l_{n-1}$.

Next we turn to the entries associated with the twist position k. By the top formula in (16),

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

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 5. By (18) and (22),

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

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

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

and recalling that $\overleftarrow{p}_{k+1} = \hat{p}_{k+1}(1+\varepsilon_{k+1}^-)$ (see (24)), the following exact relation holds,

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

In addition, the exact relation

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

holds if we set

$$\tilde{U}_-(k) \;=\; \frac{\hat{U}_-(k)}{1+\varepsilon_\circ^-}\sqrt{\frac{(1+\varepsilon_*^-)(1+\varepsilon_{**}^-)(1+\varepsilon_{\circ\circ}^-)(1+\varepsilon_k^+)}{(1+\varepsilon_/^-)(1+\varepsilon_{k+1}^-)(1+\varepsilon_+^-)}}, \tag{29}$$

where $\varepsilon_\circ^-$ is identical to the $\varepsilon_\circ$ of (28). Note that since $\bar{d}_k\, \bar{l}_k^2 = \overleftarrow{d}_k\, \overleftarrow{l}_k^{\,2}$ the $(k+1)$-st diagonal element in $\tilde{Z}_k$ remains $\breve{D}_-(k+1)$ as:

$$\bar{d}_k\, \bar{l}_k^2 + \overleftarrow{p}_{k+1} = \overleftarrow{d}_k\, \overleftarrow{l}_k^{\,2} + \overleftarrow{p}_{k+1} = \breve{D}_-(k+1), \qquad \text{from (26).} \qquad \square$$

Note: A similar result may be obtained if γ_k is computed by the last formula in (16).

5 Perturbations of Products of Bidiagonals

This section studies the effect of small relative changes in the nontrivial entries of L and D on the eigenvalues and eigenvectors of LDL^t. However 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.

5.1 Multiplicative Form

For the sake of completeness, we present the following well-known lemma and its proof.

Lemma 2 *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 $E = diag(e_1, \dots, e_n)$ is a diagonal scaling matrix unique to within a constant multiple.

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

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

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 non-trivial entries of L and D lead to the perturbed matrix

$$\bar{T} \;=\; E^{-1}LEFDFEL^tE^{-1} \tag{30}$$

5.2 Perturbation Bounds

Let $(\lambda, \boldsymbol{u})$ be an eigenpair of LDL^t, $\lambda \neq 0$, $\|\boldsymbol{u}\| = 1$. We may write $\bar{T}$ in (30) in standard multiplicative form as

$$\bar{T} \;=\; G^tLDL^tG, \tag{31}$$

$$\text{where} \quad G \;:=\; L^{-t}FEL^tE^{-1} \tag{32}$$

is an upper triangular matrix sometimes close to I. There is an eigenpair $(\bar{\lambda}, \bar{\boldsymbol{u}})$ of $\bar{T}$ associated with $(\lambda, \boldsymbol{u})$ and we want to investigate the closeness of λ to $\bar{\lambda}$ and $\boldsymbol{u}$ to $\bar{\boldsymbol{u}}$. We first look at the published bounds, in terms of G, on $|\lambda - \bar{\lambda}|$ and $|\sin\angle(\boldsymbol{u}, \bar{\boldsymbol{u}})|$. For our case of a single eigenvector, not a subspace, the results of Ipsen and Eisenstat [11] and Ren-Cang Li [24] are extremely close to each other. Since Li chose to keep $\boldsymbol{u}$ explicit in his bounds we use a slight variant of the bound (3.5) from [24]:

Theorem 7 (Variant of Theorem 3.1 in [24]) *There is an eigenpair* $(\bar{\lambda}, \bar{\boldsymbol{u}})$ *of* $\bar{T}$*, with* $\bar{\lambda} \neq 0$*, such that*

$$|\sin \angle(\boldsymbol{u}, \bar{\boldsymbol{u}})| \leq \|(I - G^{-1})\boldsymbol{u}\| + \frac{\|(G^t - G^{-1})\boldsymbol{u}\|}{\mathrm{relgap}(\lambda)} \tag{33}$$

where

$$\mathrm{relgap}(\lambda) := \frac{\min\{|\lambda - \bar{\mu}| : \ \bar{\mu} \neq \bar{\lambda}, \ det[\bar{T} - \bar{\mu} I] = 0\}}{|\lambda|}.$$

A bound on $|\lambda - \bar{\lambda}|/|\lambda|$ comes from a residual norm by standard techniques. Try $(G^{-1}\boldsymbol{u}, \lambda)$ as an approximate eigenpair of $\bar{T}$;

$$\begin{aligned} \bar{\boldsymbol{r}} &:= \frac{\bar{T} G^{-1}\boldsymbol{u} - G^{-1}\boldsymbol{u}\lambda}{\|G^{-1}\boldsymbol{u}\|} \\ &= \frac{(G^t\boldsymbol{u} - G^{-1}\boldsymbol{u})\lambda}{\|G^{-1}\boldsymbol{u}\|}, \quad \text{by (31).} \end{aligned}$$

By Theorem 1,

$$|\lambda - \bar{\lambda}| \ \leq \ \|\bar{\boldsymbol{r}}\| \ = \ \frac{\|G^t\boldsymbol{u} - G^{-1}\boldsymbol{u}\| \, |\lambda|}{\|G^{-1}\boldsymbol{u}\|}. \tag{34}$$

Note that (34) and (33) yield uniform relative condition numbers for all the eigenvalues and eigenvectors respectively since

$$\frac{|\lambda - \bar{\lambda}|}{|\lambda|} \ \leq \ \frac{\|(G^t G - I) G^{-1}\boldsymbol{u}\|}{\|G^{-1}\boldsymbol{u}\|} \ \leq \ \|G^t G - I\|, \tag{35}$$

$$\text{and} \quad |\sin \angle(\boldsymbol{u}, \bar{\boldsymbol{u}})| \ \leq \ \|I - G^{-1}\| + \frac{\|G^t - G^{-1}\|}{\mathrm{relgap}(\lambda)}. \tag{36}$$

Writing $E = I + \Delta_1$, $E^{-1} = I + \Delta_2$, $EF = I + \Delta_3$ and $(EF)^{-1} = I + \Delta_4$, it can be shown that

$$\begin{aligned} \|I - G^{-1}\| &\leq \|\Delta_1\| + \mathrm{cond}(L)\|\Delta_4\|(1 + \|\Delta_1\|), \\ \|G^t - G^{-1}\| &\leq \|\Delta_1\| + \|\Delta_2\| + \mathrm{cond}(L)\left\{\|\Delta_3\|(1 + \|\Delta_2\|) + \|\Delta_4\|(1 + \|\Delta_1\|)\right\}, \\ \text{and} \quad \|G^t G - I\| &\leq \|\Delta_2\|(2 + \|\Delta_2\|) + \mathrm{cond}(L)\|\Delta_3\|(1 + \|\Delta_2\|)^2(2 + \mathrm{cond}(L)\|\Delta_3\|). \end{aligned}$$

Thus, after substituting the above values in (35) and (36), we can define

$$\begin{aligned} \mathrm{relcond}(\lambda) &:= 1 + \mathrm{cond}(L), \\ \text{and} \quad \mathrm{relcond}(\boldsymbol{u}) &:= (1 + \mathrm{cond}(L))\left(1 + \frac{1}{\mathrm{relgap}(\lambda)}\right) \end{aligned}$$

for all eigenpairs $(\lambda, \boldsymbol{u})$ of LDL^t. Hence when L is well-conditioned, all eigenpairs of LDL^t are "relatively robust".

However, we have encountered many cases where L is ill-conditioned, and some of the eigenpairs of LDL^T, often its small eigenvalues and corresponding eigenvectors, are determined to high relative accuracy. To get bounds that make a distinction between different eigenpairs we need to retain the vector $\boldsymbol{u}$ in the bounds (33) and (34).

Thus we manipulate (33) into a revealing form using (32). Write $L = I + \overset{o}{L}$ and exploit the bidiagonal form of L to pass the diagonal matrix EF to the other side of L. From Lemma 2, $E = \mathrm{diag}(e_1, \dots, e_n)$ and $F = \mathrm{diag}(f_1, \dots, f_n)$ satisfy

$$\begin{aligned} e_n &= 1, \; e_j = (1+\eta_j)e_{j+1}, \quad 1 \le j < n, \\ f_j &= \sqrt{1+\varepsilon_j}, \quad \forall\, j. \end{aligned}$$

It may be verified that

$$LEF = EF(L + H_1 \overset{o}{L}) \tag{37}$$

where

$$H_1 = \mathrm{diag}\left(0, (1+\eta_1)\frac{f_1}{f_2} - 1, \dots, (1+\eta_{n-1})\frac{f_{n-1}}{f_n} - 1\right).$$

Hence, to first order,

$$\|H_1\| \le h := \max_i |\eta_i| + \|F^2 - I\|. \tag{38}$$

Note that in contrast to the bound on E there is no factor of n in h. In Section 7 we shall give specific values to $\max_i |\eta_i|$ and $\max_j |\varepsilon_j|$, the relative changes in the l_i and d_j respectively. Use (37) to find that

$$\begin{aligned} G^t &= E^{-1} LEFL^{-1} \\ &= E^{-1} EF(L + H_1 \overset{o}{L})L^{-1} \\ &= F(I + H_1 \overset{o}{L} L^{-1}). \end{aligned}$$

In order to keep our bound (49) as simple as possible we derive an expression, in (40), for G^{-1} that avoids the inverse of $I + H_1 \overset{o}{L} L^{-1}$. As in (37), we can write

$$(EF)^{-1} L^t = (L^t + \overset{o}{L}{}^t H_2)(EF)^{-1} \tag{39}$$

where

$$H_2 = \mathrm{diag}\left(0, \frac{1}{1+\eta_1}\frac{f_2}{f_1} - 1, \dots, \frac{1}{1+\eta_{n-1}}\frac{f_n}{f_{n-1}} - 1\right),$$

with $\|H_2\| \le h$, to first order. Hence, by (39),

$$G^{-1} = EL^{-t}(EF)^{-1}L^t = (I + E(\overset{o}{L} L^{-1})^t E^{-1} H_2)F^{-1}. \tag{40}$$

Letting $P_1 = H_1 \overset{o}{L} L^{-1}$ and $P_2 = E(\overset{o}{L} L^{-1})^t E^{-1} H_2$, we can write

$$G^t = F(I + P_1), \quad \text{and} \quad G^{-1} = (I + P_2)F^{-1}. \tag{41}$$

Given G^t and G^{-1} in the above form,

$$\begin{aligned} \|(G^t - G^{-1})\boldsymbol{u}\| &= \|(F - F^{-1})\boldsymbol{u} + (FP_1 - P_2F^{-1})\boldsymbol{u}\| \\ &\leq \|F - F^{-1}\| + h\|F\|\| \overset{o}{L} L^{-1}\boldsymbol{u}\| + h\|E\|\|E^{-1}\|\|F^{-1}\|\|| \overset{o}{L} L^{-1}|^t|\boldsymbol{u}||, \\ &\leq h\|E\|\|E^{-1}\|\|F^{-1}\| \left(1 + \| \overset{o}{L} L^{-1}\boldsymbol{u}\| + \|| \overset{o}{L} L^{-1}|^t|\boldsymbol{u}||\right), \end{aligned} \tag{42}$$

since $\|F - F^{-1}\| \leq \|F^2 - I\| \, \|F^{-1}\| \leq h \, \|F^{-1}\|$, by (38). Note that $|M|$ denotes the matrix with entries $|m_{ij}|$. In order to derive relcond(λ) from (34), we need to bound $\|G^{-1}\boldsymbol{u}\|$ from below. From (41),

$$\begin{aligned} FG^{-1}\boldsymbol{u} &= (I + P_1^t)^{-1}\boldsymbol{u}, \\ \Rightarrow \quad \|G^{-1}\boldsymbol{u}\| &\geq \frac{\|(I + P_1^t)^{-1}\boldsymbol{u}\|}{\|F\|}. \end{aligned} \tag{43}$$

Writing $\boldsymbol{u} = (I + P_1^t)(I + P_1^t)^{-1}\boldsymbol{u} = (I + P_1^t)^{-1}\boldsymbol{u} + P_1^t(I + P_1^t)^{-1}\boldsymbol{u}$, we get

$$\|(I + P_1^t)^{-1}\boldsymbol{u}\| \geq \frac{1}{1 + \|P_1\|}. \tag{44}$$

By (43) and (44), and using $P_1 = H_1 \overset{o}{L} L^{-1}$,

$$\|G^{-1}\boldsymbol{u}\| \; \geq \; \frac{1}{\|F\|(1 + h\| \overset{o}{L} L^{-1}\|)}. \tag{45}$$

Hence, by (34), (42) and (45),

$$\frac{|\lambda - \bar{\lambda}|}{|\lambda|} \; \leq \; \frac{h\|E\|\|E^{-1}\|\|F^{-1}\|}{\|F\|(1 + h\| \overset{o}{L} L^{-1}\|)} \left(1 + \| \overset{o}{L} L^{-1}\boldsymbol{u}\| + \|| \overset{o}{L} L^{-1}|^t|\boldsymbol{u}||\right), \tag{46}$$

whence we define (assuming that $h\| \overset{o}{L} L^{-1}\| \ll 1$)

$$\text{relcond}(\lambda) \; := \; 1 + \| \overset{o}{L} L^{-1}\boldsymbol{u}\| + \|| \overset{o}{L} L^{-1}|^t|\boldsymbol{u}||. \tag{47}$$

Furthermore,

$$\begin{aligned} \|(I - G^{-1})\boldsymbol{u}\| &= \|(I - F^{-1})\boldsymbol{u} - P_2F^{-1}\boldsymbol{u}\|, \\ &\leq \|I - F^{-1}\| + h\|E\|\|E^{-1}\| \, \|F^{-1}\| \, \|| \overset{o}{L} L^{-1}|^t|\boldsymbol{u}||, \\ &\leq h\|E\|\|E^{-1}\| \, \|F^{-1}\| \, \left(1 + \|| \overset{o}{L} L^{-1}|^t|\boldsymbol{u}||\right), \end{aligned} \tag{48}$$

where the last inequality above holds since F is diagonal, thus implying

$$\|I - F^{-1}\| \leq \|F^{-1}\| \, \|F - I\| \leq \|F^{-1}\| \, \|F^2 - I\| \leq h \, \|F^{-1}\|, \qquad \text{by (38)}.$$

By (33), (42) and (48),

$$\begin{aligned} |\sin\angle(\boldsymbol{u},\bar{\boldsymbol{u}})| &\le h\|E\|\|E^{-1}\|\,\|F^{-1}\|\,\Big(1+\||\overset{o}{L}L^{-1}|^t|\boldsymbol{u}|\| \\ &+ \frac{1+\|\overset{o}{L}L^{-1}\boldsymbol{u}\|+\||\overset{o}{L}L^{-1}|^t|\boldsymbol{u}|\|}{\operatorname{relgap}(\lambda)}\Big). \end{aligned} \tag{49}$$

The above bound persuades us to define

$$\operatorname{relcond}(\boldsymbol{u}) := \left(1+\|\overset{o}{L}L^{-1}\boldsymbol{u}\|+\||\overset{o}{L}L^{-1}|^t|\boldsymbol{u}|\,\|\right)\left(1+\frac{1}{\operatorname{relgap}(\lambda)}\right). \tag{50}$$

Thus, from (46), (47) and (49), (50) we have

$$\begin{aligned} \frac{|\lambda-\bar{\lambda}|}{|\lambda|} &\le \frac{h\|E\|\|E^{-1}\|\|F^{-1}\|}{\|F\|}\operatorname{relcond}(\lambda), \\ |\sin\angle(\boldsymbol{u},\bar{\boldsymbol{u}})| &\le h\|E\|\|E^{-1}\|\|F^{-1}\|\operatorname{relcond}(\boldsymbol{u}), \end{aligned} \tag{51}$$

where h is defined in (38).

In cases where there is no element growth when factoring T into LDL^t, say $\|\overset{o}{L}\|\le 0.96$, then

$$\begin{aligned} \|\overset{o}{L}L^{-1}\| &\le \|\overset{o}{L}\|\|L^{-1}\|, \\ &\le \frac{\|\overset{o}{L}\|}{1-\|\overset{o}{L}\|}\le 24, \end{aligned} \tag{52}$$

and, from (50),

$$\operatorname{relcond}(\boldsymbol{u})\le 49\left(1+\frac{1}{\operatorname{relgap}(\lambda)}\right)$$

for *all* eigenvectors of LDL^t.

This result shows the importance of not automatically using LDL^t but choosing the twisted factorization NDN^t with minimal $\|\overset{o}{N}\|$. An extreme example is the following matrix, with $\beta\gg 1$:

$\operatorname{diag}(T)=\operatorname{diag}(1,1+\beta^2,\dots,1+\beta^2),\quad T_{i,i+1}=\beta$ for all i.

The factorization LD_+L^t, with twist at n, has $D_+=I$, $\overset{o}{L}=\beta\operatorname{diag}([1,\dots,1];-1)$ whereas the bottom-up factorization, UD_-U^t, with twist at 1, has $\overset{o}{U}\approx\beta^{-1}\operatorname{diag}([1,\dots,1];+1)$ and $D_-\approx\operatorname{diag}(\beta^{2(1-n)},\beta^2,\dots,\beta^2+1)$. The omitted entries in D_- increase slowly from β^2 to β^2+1.

So, as in (52), $\|\overset{o}{U}U^{-1}\|\le\beta^{-1}/(1-\beta^{-1})=(\beta-1)^{-1}$. For this factorization,

$$\begin{aligned} \operatorname{relcond}(\boldsymbol{u}) &= \left(\|1+\overset{o}{U}U^{-1}\boldsymbol{u}\|+\||\overset{o}{U}U^{-1}|^t|\boldsymbol{u}|\,\|\right)\left(1+\frac{1}{\operatorname{relgap}(\lambda)}\right) \\ &\le \left(1+\frac{2}{\beta-1}\right)\left(1+\frac{1}{\operatorname{relgap}(\lambda)}\right) \end{aligned}$$

for all $\boldsymbol{u}$, whereas relcond($\boldsymbol{u}$) for LD_+L^t is much larger.

The relative condition numbers given in (50) and (47) are specific to each eigenvalue, and we use them in the proof of Theorem 9 in Section 7. However, as we discuss in the next section, these relconds are not entirely satisfactory.

5.3 Element Growth

The above analysis suggests that element growth ($\| \overset{o}{L} L^{-1} \| \gg 1$) is dangerous. However, we have found that the presence of element growth does not prevent *some* of the eigenvectors of LDL^t, usually those with small eigenvalues, from being relatively robust. The relative robustness of $(\lambda, \boldsymbol{u})$ seems to be governed by Dhillon's relative condition number

$$\kappa_{\text{rel}}(\lambda) := \frac{\boldsymbol{u}^t L|D|L^t \boldsymbol{u}}{|\lambda|} = \frac{\boldsymbol{u}^t L|D|L^t \boldsymbol{u}}{|\boldsymbol{u}^t LDL^t \boldsymbol{u}|}, \tag{53}$$

introduced in [9], and in many cases the relcond($\boldsymbol{u}$) given in (50) is too pessimistic. Unfortunately, as yet we have not been able to *prove* a guaranteed bound in terms of (53). We have estimates that are correct to first order but no bounds. The small relative perturbations relevant to our algorithm, i.e., E and F, are not independent and it may be necessary to use this property. In [31] one of us connected the study of LDL^t to an indefinite (or hyperbolic) singular value decomposition. We report on these results but will not give proofs. Write $D = \Delta\Omega\Delta$, $\Omega = \text{sign}(D)$, and $\lambda = \sigma^2 \text{sign}(\lambda)$. Then if $LDL^t\boldsymbol{u} = \boldsymbol{u}\lambda$, $\|\boldsymbol{u}\| = 1$, we write

$$\begin{aligned} \Delta L^t \boldsymbol{u} &= \boldsymbol{p}\sigma, \\ L\Delta\Omega\boldsymbol{p} &= \boldsymbol{u}\sigma\text{sign}(\lambda), \\ \boldsymbol{p}^t\Omega\boldsymbol{p} &= \text{sign}(\lambda). \end{aligned}$$

The new quantity $\boldsymbol{p}$ is called the left Ω-singular vector of ΔL^t. It is not hard to see that $\kappa_{\text{rel}}(\lambda)$ defined above in (53) equals $\|\boldsymbol{p}\|^2$. There is an expression for the 'relative' derivative of σ with respect to each of the entries of ΔL^t, namely the diagonal elements $\delta_i = \sqrt{|d_i|}$, and off-diagonals $b_i := l_i\delta_i$. Theorem 2 of [31] shows that for $\sigma > 0$ and $\omega_i = \Omega_{ii}$,

$$\begin{aligned} \Theta(k) &:= \frac{\delta_k}{\sigma} \cdot \frac{\partial\sigma}{\partial\delta_k} = \sum_{i=1}^{k} u(i)^2 - \text{sign}(\lambda) \sum_{j=1}^{k-1} \omega_j p(j)^2 \\ \Gamma(k) &:= \frac{b_k}{\sigma} \cdot \frac{\partial\sigma}{\partial b_k} = \text{sign}(\lambda) \sum_{i=1}^{k} \omega_i p(i)^2 - \sum_{j=1}^{k} u(j)^2. \end{aligned}$$

It was shown that $|\Theta(k)| \leq \|\boldsymbol{p}\|^2$ and $|\Gamma(k)| \leq \|\boldsymbol{p}\|^2$. The total 'relative derivative' of σ is bounded by $(2n-1)\|\boldsymbol{p}\|^2$. When $\Omega = I$ then $\|\boldsymbol{p}\| = 1$ and we recover the known (almost) attainable bound in [7].

Our first order perturbation analysis (derivation omitted) reveals the dominant role of $\boldsymbol{p}$ in determining relative robustness. Let $\bar{\varepsilon} := \max_{i,j}\{|\eta_i|, |\varepsilon_j|\}$ where $l_i \longrightarrow l_i(1+\eta_i)$, $d_i \longrightarrow d_i(1+\varepsilon_i)$ and $T = LDL^t$. Then for $(\lambda, \boldsymbol{u})$ we can show that

$$|\delta\lambda| \leq |\lambda| \left(2 \sum_{k=1}^{n-1} |\Gamma(k)| + \|\boldsymbol{p}\|^2 \right) (\bar{\varepsilon} + \bar{\varepsilon}^2) + \boldsymbol{u}^t \overset{o}{L} |D| \overset{o}{L}{}^t \boldsymbol{u}\, \bar{\varepsilon}^2 + O(\|\delta T\|^2). \tag{54}$$

Recall that $\delta\lambda = \lambda - \bar{\lambda} = \boldsymbol{u}^t \delta T \boldsymbol{u} + O(\|\delta T\|^2)$.

In order to get a bound on $\tan\angle(\boldsymbol{u}, \bar{\boldsymbol{u}})$ we must refer to all eigenpairs and put subscripts on $\lambda, \boldsymbol{u}, \boldsymbol{p}$ and on the Γ's, denoting by $\Gamma_j(k)$ the $\Gamma(k)$ for the triplet $(\sigma_j, \boldsymbol{u}_j, \boldsymbol{p}_j)$. In addition we must define the quantity

$$\Psi_{ij} := \left(\sum_{k=1}^{n-1} |\Gamma_i(k)| \left| \frac{p_j(k)}{p_i(k)} \right| + |\Gamma_j(k)| \left| \frac{p_i(k)}{p_j(k)} \right| \right) + |\boldsymbol{p}_i|^t \, |\boldsymbol{p}_j|.$$

Then, for the jth eigenvector $\boldsymbol{u}_j$,

$$\begin{aligned} |\tan\angle(\boldsymbol{u}_j, \bar{\boldsymbol{u}}_j)| \quad \leq \quad & \left[\sum_{i \neq j} \left(\frac{\Psi_{ij} \sqrt{|\lambda_i \lambda_j|}}{|\lambda_i - \lambda_j|} \right)^2 \right]^{1/2} (\bar{\varepsilon} + \bar{\varepsilon}^2) \\ & + \bar{\varepsilon}^2 \left[\sum_{i \neq j} \left(\frac{\sqrt{|\lambda_i \lambda_j|}}{|\lambda_i - \lambda_j|} \sum_{k=1}^{n-1} \left| \frac{\Gamma_i(k)}{p_i(k)} \frac{\Gamma_j(k)}{p_j(k)} \right| \right)^2 \right]^{1/2} + O(\|\delta T\|^2). \quad (55) \end{aligned}$$

The leading term in (55) is complicated. It is well approximated, for very small λ_j, by

$$\Psi_{jj} \left[\sum_{i \neq j} \frac{|\lambda_i \lambda_j|}{(\lambda_i - \lambda_j)^2} \right]^{1/2} \bar{\varepsilon}$$

and $\Psi_{jj} < (2n-1)\|\boldsymbol{p}_j\|^2$.

In the cases we have examined, the quantities in (54) and (55) have been realistic and much smaller than our relcond$(\boldsymbol{u})$ in Section 5.2. The second term in (54) is not, in general, proportional to λ and we hope to show that it is cancelled by the $O(\|\delta T\|)^2$ term in the expansion of $\delta\lambda$ as a power series. We hope that future work will show that the first order terms do, in fact, dominate the higher order ones and then we may incorporate a more realistic definition of relcond$(\boldsymbol{u})$, namely the leading term in (55), into the bounds of Theorem 9.

We emphasize that relative perturbation theory is not the main concern of this paper. More analysis of relative condition numbers is given in [30, 32]. For the rest of this paper we assume that all relconds are bounded by a modest constant like 10.

6 Algorithm for an Eigenvector

The method presented below is close in spirit to the one presented by Godunov and his co-workers in the USSR in 1985, see [16] 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 into an accurate eigenvector. Independently Fernando discovered a similar idea in terms of running the well known two-term recurrence for D_+, both forwards from $D_+(1)$ and backwards from $D_+(n) = 0$, and then joining the two sequences where they are closest. In [29], Parlett and Dhillon formulated and proved Theorems 2 and 3 in Section 4 which show that at least one twisted factorization must reveal the size of the smallest eigenvalue, thus yielding an accurate eigenvector.

However neither Godunov nor Fernando reap 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, modern programming languages do not make it available to the user. 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 [15], even with highly accurate eigenvalue approximations, yields 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 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 9 in the next section. That result requires a second innovation, beyond the use of LDL^t, namely use of the *differential* qd algorithms introduced in Section 4.2 to compute the entries of the twisted factors. The commutative diagrams in Section 4.3 are not valid for Rutishauser's implementation. Hence the LDL^T representation and differential qd transformations are crucial to our goal of computing orthogonal eigenvectors when relative gaps are large. We now give details of our algorithm.

Algorithm Getvec

Assume that $\hat{\lambda}$ is much closer to one eigenvalue of LDL^t than to any other.

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

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

III. Compute γ_k, $k = 1, \ldots, n$ by the top formula of (16). 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 Section 4.1.

IV. Form the approximate eigenvector $\boldsymbol{z}$ by solving $N_r^t \boldsymbol{z} = \boldsymbol{e}_r$ which is equivalent to solving $N_r D_r N_r^t \boldsymbol{z} = \boldsymbol{e}_r \gamma_r$ via

$$
\begin{aligned}
z(r) &= 1, \\
\text{For } i = r-1, \ldots, 1, \quad z(i) &= \begin{cases} -L_+(i) z(i+1), & z(i+1) \neq 0, \\ -(d_{i+1} l_{i+1} / d_i l_i) z(i+2), & \text{otherwise.} \end{cases} \\
\text{For } j = r, \ldots, n-1, \quad z(j+1) &= \begin{cases} -U_-(j) z(j), & z(j) \neq 0, \\ -(d_{j-1} l_{j-1} / d_j l_j) z(j-1), & \text{otherwise.} \end{cases}
\end{aligned}
$$

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

Remark 1 In Step IV above, a zero entry in an eigenvector requires special handling. For example, when $z(i+1)=0$, $i<r$, 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 $z(j)=0$, $j>r$, is handled similarly.

Remark 2 It is possible to avoid some computation in Steps I and II by using Gerschgorin disks. In particular, it is easy to show that if the eigenvalue is not contained in the i-th Gerschgorin disk, then $r\neq i$. See [9, Sec. 3.4.1] for details.

Remark 3 The above algorithm can also be used to improve the accuracy of $\hat{\lambda}$. By Lemma 1, $\gamma_r/\|\boldsymbol{z}\|^2$ 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.

Remark 4 The vector $\boldsymbol{z}$ sometimes has small numerical support. During the computation of $\boldsymbol{z}$ this situation can be detected as follows. We continue the recurrence for $\boldsymbol{z}$ until 2 consecutive entries fall below ε in magnitude. In many cases all further entries of $\boldsymbol{z}$ can be set to 0. Suppose $|z(i-1)|<\varepsilon$ and $|z(i)|<\varepsilon$, $i<r$. If the elements $z(j)$, $j<i-1$, 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 that equals $\beta_{i-2}(z(i-1)\boldsymbol{e}_{i-2}-z(i-2)\boldsymbol{e}_{i-1})$, where $\beta_{i-2}=D_+(i-2)L_+(i-2)$. For the computed vector $\boldsymbol{z}$ to be accurate (see Theorem 1), we must ensure 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 $i>r$ and both $z(i-1)$ and $z(i)$ dip below ε we set the elements $z(j)$, $j>i$, to 0 if

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

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

$$|\mathrm{supp}(\boldsymbol{z})|=last-first+1. \tag{56}$$

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 so 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 [29], as $\hat{\lambda}\to\lambda_j$,

$$\frac{\gamma_r}{\gamma_i}\;\to\;\frac{v_j(i)^2}{v_j(r)^2}, \tag{57}$$

where $\boldsymbol{v}_j$ is λ_j'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 sometimes not sufficient to put $\hat{\lambda}$ close enough to λ_j for (57) to hold for indices where $|v_j(i)|\ll\sqrt{\varepsilon}$. However, when $|\hat{\lambda}-\lambda_j|=O(\varepsilon|\hat{\lambda}|)$ the above strategy almost always gives us the correct list of indices with $|v_j(i)|\geq\sqrt{\varepsilon}$ (see (7)).

Remark 5 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 6 Suppose $\hat{\lambda}$ approximates λ_j. In the next section we will show that in the presence of roundoff errors, the computed vector $\boldsymbol{z}$ satisfies

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

and thus $\boldsymbol{z}$ is an accurate eigenvector when $\mathrm{relgap}(\hat{\lambda}) = O(1)$. 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: $(LDL^T - \hat{\lambda}I)\boldsymbol{y} = \boldsymbol{z} \;\Rightarrow (LDL^T - \hat{\lambda}I)^2\boldsymbol{y} = \gamma_r\boldsymbol{e}_r$. The tempting argument is that by doing so,

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

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

Unfortunately this simple solution does not work. In our experience the extra step of inverse iteration increases the accuracy by a factor of .1 or .01 and not by a factor of $\sqrt{\varepsilon}$ as the above reasoning indicates. As the analysis of the next section will show, this failure is due to the presence of roundoff errors and the relative perturbation theory of Section 5.

The case of $\mathrm{relgap}(\hat{\lambda}) \ll 1/n$ requires radically different strategies. One strategy is to take a new shift to improve the relative gaps and to stay with the $\boldsymbol{z}$ vector. This is not the subject 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 [27] and [28].

7 Bounds on Accuracy (Proof of Correctness)

The formal analysis begins here. We start by showing that the vector $\hat{\boldsymbol{z}}$ computed by Algorithm Getvec is very close to a vector $\tilde{\boldsymbol{z}}$ that obeys the exact relationship (58), where $\tilde{N}_r$ and $\tilde{D}_r$ are perturbed factors determined by step IV of the algorithm.

Theorem 8 *Let $\hat{N}_r$ and $\hat{D}_r$, $\tilde{N}_r$ and $\tilde{D}_r$ be the twisted factors represented by $\hat{Z}_r$ and $\tilde{Z}_r$ respectively in Figure 4 (see also Theorem 6 and Figure 5). Let $\hat{\boldsymbol{z}}$ be the vector computed in Step IV of Algorithm Getvec, and let $\tilde{\boldsymbol{z}}$ be the exact solution of*

$$\tilde{N}_r\tilde{D}_r\tilde{N}_r^t\tilde{\boldsymbol{z}} = \tilde{\gamma}_r\boldsymbol{e}_r. \tag{58}$$

Then, barring underflow, $\hat{\boldsymbol{z}}$ is a small relative perturbation of $\tilde{\boldsymbol{z}}$. Specifically,

$$\begin{aligned} \hat{z}(r) &= \tilde{z}(r) \;=\; 1, \\ \hat{z}(i) &= \tilde{z}(i)\cdot(1+\eta_i), \;\; i \neq r, \quad |\eta_i| \;\leq\; 5|i-r|\varepsilon, \end{aligned} \tag{59}$$

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 $\hat{\boldsymbol{z}}$ is zero (so that only the top formulae for $\hat{z}(i)$ and $\hat{z}(j+1)$ in Step IV are used). The matrix $\tilde{N}_r$, built out of $\widehat{L}_+$ and $\breve{U}_-$, was defined in Theorem 6 so that the equality $\bar{L}\bar{D}\bar{L}^t - \hat{\lambda}I = \tilde{N}_r\tilde{D}_r\tilde{N}_r^t$ holds. Thus $\tilde{N}_r$ 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

$$\hat{z}(i) = -\hat{U}_-(i-1)\hat{z}(i-1)(1+\varepsilon_i), \quad i > r, \tag{60}$$

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

$$\begin{aligned} \tilde{z}(i) &= -\widehat{L}_+(i)\tilde{z}(i+1), & i < r, \\ \text{and} \quad \tilde{z}(i) &= -\breve{U}_-(i-1)\tilde{z}(i-1), & i > r. \end{aligned} \tag{61}$$

Since $\hat{z}(r) = \tilde{z}(r) = 1$, we may define $\eta_r = 0$ and trivially write $\hat{z}(r) = \tilde{z}(r)(1+\eta_r)$ with $|\eta_r| \le 4(r-r)\varepsilon$. Now proceed by induction as i decreases in order to prove (59). Examine (21) to find that

$$\begin{aligned} \hat{L}_+(i) &= \widehat{L}_+(i)(1+\delta_i), \\ \text{where} \quad |\delta_i| &< \sqrt{(1+\varepsilon)^6 - 1} = 3\varepsilon + O(\varepsilon^2) \quad \text{for all } i < r. \end{aligned}$$

Thus

$$\begin{aligned} \hat{z}(i-1) &= -\widehat{L}_+(i-1)(1+\delta_{i-1})\hat{z}(i)(1+\varepsilon_{i-1}), \\ &= -\widehat{L}_+(i-1)(1+\delta_{i-1})\tilde{z}(i)(1+\eta_i)(1+\varepsilon_{i-1}), \quad |\eta_i| \le 4(r-i)\varepsilon \text{ by induction}, \\ &= \tilde{z}(i-1)(1+\delta_{i-1})(1+\eta_i)(1+\varepsilon_{i-1}), \quad \text{by (61)} \\ &= \tilde{z}(i-1)(1+\eta_{i-1}), \quad \text{thus defining } \eta_{i-1} \equiv (1+\eta_i)(1+\delta_{i-1})(1+\varepsilon_{i-1}) - 1, \\ |\eta_{i-1}| &\le (1+|\eta_i|)(1+\varepsilon)^3(1+\varepsilon) - 1 = [4(r-i)+4]\varepsilon + O(\varepsilon^2), \quad \text{as claimed.} \end{aligned}$$

For the lower half of $\hat{\boldsymbol{z}}$, $i \ge r$, the argument is similar with $\hat{U}_-$ and $\breve{U}_-$ involved instead of $\hat{L}_+$ and $\widehat{L}_+$. Note that $\hat{U}_-$ is related to $\breve{U}_-$ by (29) and (28), which, respectively, involve $1\frac{1}{2}$ and 1 more ulps than (21).

To begin, define $\eta_r = 0$ so that $|\eta_r| \le 5(r-r)\varepsilon$. For $i = r+1$, (59) holds since (29) gives 4.5 ulps for $\breve{U}_-(r)$ in (60), while $\varepsilon_{r+1} = 0$ (because $\hat{z}(r) = 1$). For $i > r+1$, (28) gives 4 ulps and ε_i gives one more ulp for an increase of at most 5 ulps each time i increases. Thus (59) holds for all values of i.

We now consider the case when an eigenvector entry vanishes, i.e., $\hat{z}(i+1) = 0$. 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

$$z(i) = -(d_{i+1}l_{i+1}/d_il_i)z(i+2), \tag{62}$$

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 4, we can see that the product

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

Thus the term $(d_{i+1}l_{i+1}/d_i l_i)$ in (62) contributes 6 ulps, and combining these with the 4 arithmetic operations in (62), 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 $|\delta_i| \le 10\varepsilon$ (a closer analysis reveals that $|\delta_i| \le 8\varepsilon$). Thus (59) holds in this case also. The case when $\hat{z}(i) = 0$, $i > r$ can be handled similarly. □

Corollary 1 (to Theorem 8) *Under the hypotheses of Theorem 8,*

$$|\sin\angle(\tilde{\boldsymbol{z}}, \hat{\boldsymbol{z}})| \le 5\varepsilon|supp(\hat{\boldsymbol{z}})| + O(\varepsilon^2)$$

where $|supp(\hat{\boldsymbol{z}})|$ is the numerical support of $\hat{\boldsymbol{z}}$ as defined in (56).

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

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

$$\begin{aligned} \mathrm{avg}(\eta_i;\boldsymbol{v}) &= \sum \eta_i v(i)^2, \\ \mathrm{var}(\eta_i;\boldsymbol{v}) &= \sum \eta_i^2 v(i)^2 - \mathrm{avg}(\eta_i;\boldsymbol{v})^2, \\ \text{std. dev.}(\eta_i;\boldsymbol{v}) &= \sqrt{\mathrm{var}(\eta_i;\boldsymbol{v})}. \end{aligned}$$

Now,

$$\begin{aligned} |\cos^2\angle(\boldsymbol{v},\bar{\boldsymbol{v}})| &= \frac{(\bar{\boldsymbol{v}}^t\boldsymbol{v})^2}{\bar{\boldsymbol{v}}^t\bar{\boldsymbol{v}}} \\ &= \frac{1+2\sum\eta_i v(i)^2 + (\sum\eta_i v(i)^2)^2}{1+2\sum\eta_i v(i)^2 + \sum\eta_i^2 v(i)^2} \\ |\sin^2\angle(\boldsymbol{v},\bar{\boldsymbol{v}})| &= \frac{\sum\eta_i^2 v(i)^2 - (\sum\eta_i v(i)^2)^2}{1+2\sum\eta_i v(i)^2 + \sum\eta_i^2 v(i)^2} \\ &\le \frac{\mathrm{var}(\eta_i;\boldsymbol{v})}{1+2\,\mathrm{avg}(\eta_i;\boldsymbol{v}) + \mathrm{avg}(\eta_i;\boldsymbol{v})^2}, \quad \text{since} \quad \mathrm{avg}^2 \le \sum\eta_i^2 v(i)^2, \\ \Rightarrow \quad |\sin\angle(\boldsymbol{v},\bar{\boldsymbol{v}})| &\le \frac{\text{std. dev.}(\eta_i;\boldsymbol{v})}{1+\mathrm{avg}(\eta_i,\boldsymbol{v})}. \end{aligned}$$

A crude but simple bound on the numerator is $\max_i |\eta_i|$ and, if each $\eta_i = O(\varepsilon)$, then $1+\mathrm{avg}(\eta_i,\boldsymbol{v}) = 1+O(\varepsilon)$. Finally substitute $\tilde{\boldsymbol{z}}$ for $\boldsymbol{v}$ and $\hat{\boldsymbol{z}}$ for $\bar{\boldsymbol{v}}$ and use (56) and (59) to verify that

$$\max_i |\eta_i| \le 5\varepsilon(last - r) + 5\varepsilon(r - first) \le 5\varepsilon|\mathrm{supp}(\hat{\boldsymbol{z}})|. \qquad \square$$

$$
\begin{tikzcd}[column sep=large, row sep=huge]
LDL^t\boldsymbol{u} = \boldsymbol{u}\lambda \quad \bullet \arrow[r, "\textbf{dtwqds}", "\text{computed}"'] \arrow[d, "\begin{array}{l}3 \text{ to } 3\frac{1}{2} \text{ ulps in } L \\ 1 \text{ to } 4 \text{ ulps in } D\end{array}"'] & \bullet \quad \hat{N}_r\hat{D}_r\hat{N}_r^t,\ \hat{\boldsymbol{z}} \\
\bar{L}\bar{D}\bar{L}^t\bar{\boldsymbol{u}} = \bar{\boldsymbol{u}}\bar{\lambda} \quad \bullet \arrow[r, "\textbf{dtwqds}", "\text{exact}"'] & \bullet \quad \tilde{N}_r\tilde{D}_r\tilde{N}_r^t\tilde{\boldsymbol{z}} = \boldsymbol{e}_r\tilde{\gamma}_r \arrow[u, "\begin{array}{l}3 \text{ to } 4\frac{1}{2} \text{ ulps in } \tilde{N}_r \\ 2 \text{ ulps in } \tilde{D}_r\end{array}"']
\end{tikzcd}
$$

$$
\begin{aligned}
& \bar{L}\bar{D}\bar{L}^t - \hat{\lambda}I = \tilde{N}_r\tilde{D}_r\tilde{N}_r^t, \\
\Rightarrow \quad & (\bar{L}\bar{D}\bar{L}^t - \hat{\lambda}I)\tilde{\boldsymbol{z}} = \boldsymbol{e}_r\tilde{\gamma}_r.
\end{aligned}
$$

Figure 5: Relationships connecting $\boldsymbol{u}$ to $\hat{\boldsymbol{z}}$.

The following theorem is the heart of the paper. Figure 5 lays out the essentials given in Figure 4 and should be consulted.

Theorem 9 *Let $(\lambda, \boldsymbol{u})$ be an eigenpair of the real symmetric unreduced tridiagonal matrix LDL^t with $\|\boldsymbol{u}\| = 1$. 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 6 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 4.3 and let $(\bar{\lambda}, \bar{\boldsymbol{u}})$ be the eigenpair of $\bar{L}\bar{D}\bar{L}^t$ with $\bar{\lambda}$ the closest to $\hat{\lambda}$. Let ε denote the roundoff unit. Then*

$$
|\sin\angle(\hat{\boldsymbol{z}}, \boldsymbol{u})| \leq \quad 5|supp(\hat{\boldsymbol{z}})|\varepsilon + \frac{|\bar{\lambda} - \hat{\lambda}|}{|\bar{u}(r)|gap(\hat{\lambda})} + 7.5\varepsilon\,\mathrm{relcond}(\boldsymbol{u}) + O(n^2\varepsilon^2). \tag{63}
$$

Here $|supp(\hat{\boldsymbol{z}})|$ is the numerical support of $\hat{\boldsymbol{z}}$ defined in (56) and

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

For the definition of relcond$(\boldsymbol{u})$ see Section 5.

Proof. There are three terms in the upper bound on $\sin\angle(\hat{\boldsymbol{z}}, \boldsymbol{u})$ because we connect $\hat{\boldsymbol{z}}$ to $\boldsymbol{u}$ via two 'ideal' vectors $\tilde{\boldsymbol{z}}$, $\bar{\boldsymbol{u}}$ and each transition contributes a term: $\hat{\boldsymbol{z}} \longrightarrow \tilde{\boldsymbol{z}}$, $\tilde{\boldsymbol{z}} \longrightarrow \bar{\boldsymbol{u}}$, $\bar{\boldsymbol{u}} \longrightarrow \boldsymbol{u}$, see Figure 5. Recall from Theorem 6 that the matrices $\bar{L}$, $\bar{D}$, $\tilde{N}_r$, $\tilde{D}_r$ depend on $\hat{\lambda}$ and were defined so that the equality

$$
\bar{L}\bar{D}\bar{L}^t - \hat{\lambda}I = \tilde{N}_r\tilde{D}_r\tilde{N}_r^t \tag{64}
$$

holds. That was the culmination of the error analysis in Section 4.3. Recall that $\tilde{D}_r(r) = \tilde{\gamma}_r$. Then $\tilde{\boldsymbol{z}}$ is defined as the exact solution of

$$\tilde{N}_r \tilde{D}_r \tilde{N}_r^t \tilde{\boldsymbol{z}} = \boldsymbol{e}_r \tilde{\gamma}_r. \tag{65}$$

First consider $\hat{\boldsymbol{z}}$ and $\tilde{\boldsymbol{z}}$. Theorem 8 shows that each $\tilde{z}(i)$ is of the form $\hat{z}(i)(1+\eta_i)$ and Corollary 1 proves that

$$|\sin \angle(\hat{\boldsymbol{z}}, \tilde{\boldsymbol{z}})| < 5\varepsilon|\mathrm{supp}(\hat{\boldsymbol{z}})| + O(\varepsilon^2). \tag{66}$$

Next consider $\tilde{\boldsymbol{z}}$ and $\bar{\boldsymbol{u}}$. Combine (64) and (65) and then invoke Theorem 3, in Section 4, to find that

$$\frac{|\tilde{\gamma}_r|}{\|\tilde{\boldsymbol{z}}\|} \leq \frac{|\bar{\lambda} - \hat{\lambda}|}{|\bar{u}(r)|}.$$

By Theorem 1,

$$|\sin \angle(\bar{\boldsymbol{u}}, \tilde{\boldsymbol{z}})| < \frac{|\bar{\lambda} - \hat{\lambda}|}{|\bar{u}(r)|\mathrm{gap}(\hat{\lambda})}. \tag{67}$$

Finally consider $\bar{\boldsymbol{u}}$ and $\boldsymbol{u}$. The left side of Figure 5 indicates that $\bar{\boldsymbol{u}}$ and $\boldsymbol{u}$ are related through the matrix perturbations given in Section 5 (see Lemma 2):

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

From Theorem 6, no entry in L changes by more than 3 ulps except for the entry at the twist which changes by at most 3.5ε. By Lemma 2, the largest entry in $I - E$ is bounded by $\left\{3(n-1) + \frac{1}{2}\right\}\varepsilon$ so that

$$\max\left(\|E\|, \|E^{-1}\|\right) \;\leq\; 1 + 3n\varepsilon. \tag{68}$$

The perturbation F comprises half the ulps needed for changes to entries of D, namely $\frac{1}{2}$ for $i < r$, 2 for $i = r$ and $\frac{3}{2}$ for $i > r$ (see Figure 4 and Theorem 6). Thus

$$\max\left(\|F\|, \|F^{-1}\|\right) \;\leq\; 1 + 2\varepsilon. \tag{69}$$

By (38),

$$h \;\leq\; 3.5\varepsilon + 4\varepsilon \;=\; 7.5\varepsilon. \tag{70}$$

Substituting (68), (69) and (70) into the perturbation bound (51) we obtain,

$$|\sin \angle(\boldsymbol{u}, \bar{\boldsymbol{u}})| \;\leq\; 7.5\varepsilon\left\{1 + (6n+2)\varepsilon\right\} \mathrm{relcond}(\boldsymbol{u}) + O(n^2\varepsilon^2). \tag{71}$$

Note that $\mathrm{relcond}(\boldsymbol{u})$ has the term $\mathrm{relgap}(\lambda)$ in the denominator, see (50). Adding the contributions in (66), (67), and (71) yields the theorem's bound on $|\sin \angle(\hat{\boldsymbol{z}}, \boldsymbol{u})|$. □

The above theorem is the main result of this paper. We now examine its implications in obtaining numerically orthogonal eigenvectors from Algorithm Getvec. The best we can hope for is that

$$|\sin\angle(\hat{\boldsymbol{z}},\boldsymbol{u})| \;=\; O\left(\frac{n\,\varepsilon}{\mathrm{relgap}(\hat{\lambda})}\right), \tag{72}$$

where $\mathrm{relgap}(\hat{\lambda}) = \mathrm{gap}(\hat{\lambda})/|\lambda|$. Let us examine (63) to understand the conditions under which (72) can be achieved. The first term in (63) is always $O(n\varepsilon)$. The second term (with $\bar{u}(r)$) requires that the twist index should not be perversely chosen. We aim for $|\bar{u}(r)| = \|\bar{\boldsymbol{u}}\|_\infty$ but as long as $|\bar{u}(r)|$ is above average, $1/|\bar{u}(r)| \le \sqrt{n}$. When λ is relatively well-conditioned, i.e., $\mathrm{relcond}(\lambda) = O(1)$, then it is possible to compute $\hat{\lambda}$ such that $|\hat{\lambda}-\bar{\lambda}| \le K\varepsilon|\hat{\lambda}|$, and so the middle term is $O(n\varepsilon/\mathrm{relgap}(\hat{\lambda}))$. Note that with our definition we can have $\mathrm{relgap}(\hat{\lambda}) \gg 1$ and to obtain an accurate eigenvector in this case, it is not necessary to compute $\hat{\lambda}$ to full relative accuracy. However whenever $\mathrm{relgap}(\hat{\lambda}) < 1$ then it is essential to compute $\hat{\lambda}$ so that $|\hat{\lambda} - \bar{\lambda}| \le K\varepsilon|\hat{\lambda}|$. The final term in the bound depends entirely on $\mathrm{relcond}(\boldsymbol{u})$, which is a property of the factorization LDL^t. For most LDL^t, $\mathrm{relcond}(\boldsymbol{u})$ is bounded by $M/\mathrm{relgap}(\lambda)$ where M is a modest constant; see Section 5 for more details. Thus (72) holds when (i) $\mathrm{relcond}(\lambda) = O(1)$, (ii) $\mathrm{relcond}(\boldsymbol{u}) = O(1/\mathrm{relgap}(\hat{\lambda}))$, and (iii) $\hat{\lambda}$ is computed accurately enough (often to high relative accuracy).

The reader may have noticed that the bound (63) 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 $\bar{L}\bar{D}\bar{L}^t$. Recall that $\bar{L}\bar{D}\bar{L}^t$ is an intermediate factorization created solely for our roundoff error analysis. We could try and obtain a bound just in terms of the input factorization LDL^t, as in our stated goal at the beginning of the paper, see (6). However we choose not to do so since we invoke Algorithm Getvec only when $\mathrm{relgap}(\hat{\lambda})$ is not too small ($> 1000\varepsilon$) and $\mathrm{relcond}(\lambda)$ and $\mathrm{relcond}(\boldsymbol{u})$ are modest, implying that $\bar{\lambda}$ and $\bar{\boldsymbol{u}}$ are close to λ and $\boldsymbol{u}$ respectively. Thus we can preserve the spirit of (63) by replacing the eigenvalues and eigenvectors of $\bar{L}\bar{D}\bar{L}^t$ by those of LDL^t; a formal argument is possible but is messy and does not add to our exposition, so we omit it.

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

Corollary 2 *In addition to the assumptions of Theorem 9 suppose that (i) r is such that $\bar{u}(r) \ge 1/\sqrt{n}$, (ii) $\hat{\lambda}$ is computed to satisfy $|\hat{\lambda} - \bar{\lambda}|/|\hat{\lambda}| < K\varepsilon$, (iii) relgap$(\hat{\lambda})$ exceeds 2^{-8}, and (iv) relcond$(\boldsymbol{u}) \le M$. Then*

$$|\sin\angle(\hat{\boldsymbol{z}},\boldsymbol{u})| < 5n\varepsilon + 2^8K\sqrt{n}\varepsilon + 7.5M\varepsilon. \qquad \Box$$

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 1 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

$$\boldsymbol{v}_1 = \begin{bmatrix} -\sqrt{\varepsilon/2} + O(\varepsilon^{3/2}) \\ \frac{1}{\sqrt{2}}(1+\frac{\varepsilon}{4}) + O(\varepsilon^2) \\ -\frac{1}{\sqrt{2}}(1-\frac{3\varepsilon}{4}) + O(\varepsilon^2) \end{bmatrix}, \ \boldsymbol{v}_2 = \begin{bmatrix} -\sqrt{\varepsilon/2} + O(\varepsilon^{3/2}) \\ \frac{1}{\sqrt{2}}(1-\frac{5\varepsilon}{4}) + O(\varepsilon^2) \\ \frac{1}{\sqrt{2}}(1+\frac{3\varepsilon}{4}) + O(\varepsilon^2) \end{bmatrix}, \ \boldsymbol{v}_3 = \begin{bmatrix} 1 - \frac{\varepsilon}{2} + O(\varepsilon^3) \\ \sqrt{\varepsilon} + O(\varepsilon^{3/2}) \\ \frac{\varepsilon^{3/2}}{4} + O(\varepsilon^{5/2}) \end{bmatrix}.$$

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

$$|\hat{\lambda}_i - \lambda_i| \ \leq 2\varepsilon|\hat{\lambda}_i|, \quad 1 \leq i \leq 3.$$

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 in Step IV of Algorithm Getvec are such that

$$\begin{aligned} \max|\hat{\boldsymbol{v}}_i^T \hat{\boldsymbol{v}}_j| &= 1.66 \cdot 10^{-16} < \varepsilon, \qquad 1 \leq i \leq 3, \ 1 \leq j < i, \\ \max \frac{|\hat{\boldsymbol{v}}_i(k) - \boldsymbol{v}_i(k)|}{|\boldsymbol{v}_i(k)|} &= 8.88 \cdot 10^{-16} < 4\varepsilon, \qquad 1 \leq i \leq 3, \ 1 \leq k \leq 3. \end{aligned}$$

   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 $O(1)$ 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 4 in Section 6. □

Example 2 The above matrix T_0 is a "benign" example. Our second example, 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

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

while the corresponding normalized eigenvectors are

$$\boldsymbol{v}_1 = \begin{bmatrix} \frac{\varepsilon^{1/4}}{\sqrt{2}}(1+\frac{\sqrt{\varepsilon}}{2}) + O(\varepsilon^{5/4}) \\ -\frac{1}{\sqrt{2}}(1-\frac{\sqrt{\varepsilon}}{2}) + O(\varepsilon) \\ \frac{1}{\sqrt{2}}(1-\frac{3\varepsilon}{4}) + O(\varepsilon^{3/2}) \end{bmatrix}, \boldsymbol{v}_2 = \begin{bmatrix} \frac{\varepsilon^{1/4}}{\sqrt{2}}(1+\frac{\sqrt{\varepsilon}}{2}) + O(\varepsilon^{5/4}) \\ -\frac{1}{\sqrt{2}}(1-\frac{\sqrt{\varepsilon}}{2}) + O(\varepsilon) \\ -\frac{1}{\sqrt{2}}(1+\frac{3\varepsilon}{4}) + O(\varepsilon^{3/2}) \end{bmatrix}, \boldsymbol{v}_3 = \begin{bmatrix} 1-\frac{\sqrt{\varepsilon}}{2} + O(\varepsilon) \\ \varepsilon^{1/4}(1+\frac{\sqrt{\varepsilon}}{2}) + O(\varepsilon^{5/4}) \\ \frac{\varepsilon^{5/4}}{4}(1+\frac{\sqrt{\varepsilon}}{2}) + O(\varepsilon^{9/4}) \end{bmatrix}.$$

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

$$L_1^{exact} = \begin{bmatrix} 1 & 0 & 0 \\ \frac{\varepsilon^{1/4}\sqrt{1-7\varepsilon/4}}{1-\sqrt{\varepsilon}} & 1 & 0 \\ 0 & \frac{1-\sqrt{\varepsilon}}{3} & 1 \end{bmatrix}, \quad \text{and} \quad D_1^{exact} = \begin{bmatrix} 1-\sqrt{\varepsilon} & 0 & 0 \\ 0 & \frac{3\varepsilon}{4(1-\sqrt{\varepsilon})} & 0 \\ 0 & 0 & \frac{\varepsilon(8+\sqrt{\varepsilon})}{12} \end{bmatrix}.$$

On this example, Algorithm Getvec behaves quite differently than on T_0 from Example 1:

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,

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

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

$$\gamma^{(1)} = \begin{bmatrix} -4.13 \cdot 10^{-24} \\ -7.40 \cdot 10^{-32} \\ -9.86 \cdot 10^{-32} \end{bmatrix}, \quad \gamma^{(2)} = \begin{bmatrix} -6.62 \cdot 10^{-24} \\ -9.86 \cdot 10^{-32} \\ -9.86 \cdot 10^{-32} \end{bmatrix}, \quad \gamma^{(3)} = \begin{bmatrix} 2.22 \cdot 10^{-16} \\ 1.49 \cdot 10^{-8} \\ -1.00 \end{bmatrix}.$$

   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 $\hat{v}_i$ 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 following inverse iteration step:

$$\begin{aligned} L_1 D_1 L_1^t - \hat{\lambda}_i I &= L_+ D_+ L_+^T, \\ L_+ D_+ L_+^T \boldsymbol{x}_i &= \text{random vector}, \end{aligned} \tag{73}$$

   also leads to computed eigenvectors that are numerically orthogonal when the dstqds transformation is used to compute (73). 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 3 Our third 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

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

As discussed in Remark 6 in Section 6, 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 6. For an approach that can achieve orthogonality in this situation, 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 are $O(1)$ but the absolute gaps are less than 10^{-3}. We illustrate this speedup on four examples in Table 1. Matrices of the first type have eigenvalues in an arithmetic progression,

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

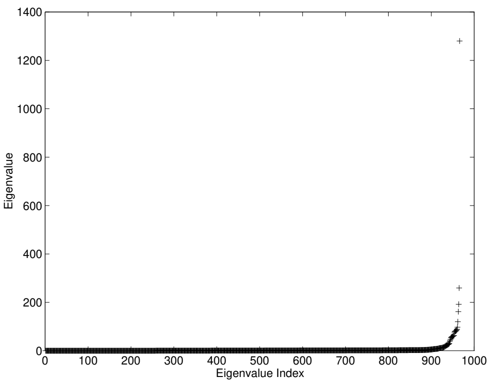
The second type has eigenvalues that come from a uniform random distribution in the interval $[\varepsilon, 1]$. The third type are the Toeplitz tridiagonal matrices with 2's on the diagonals and 1's as the off-diagonal elements, with eigenvalues $\lambda_i = 4\sin^2[i(n+1)\frac{\pi}{2}]$. The final 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 6 for a plot of its 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. 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 $|\lambda_{i+1} - \lambda_i| \le 10^{-3}\|T\|$. DSTEQR uses the QR iteration to compute orthogonal eigenvectors[22] while DSTEDC is the Divide and Conquer code in LAPACK[19]. Table 1 shows that on most examples, Algorithm Getvec is about two orders of magnitude faster than DSTEIN, TINVIT and DSTEQR. Also see

[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

Matrix Type	Matrix Size	Time Taken (in seconds) LAPACK DSTEIN	EISPACK TINVIT	LAPACK DSTEDC	LAPACK DSTEQR	Algorithm Getvec
Arithmetic Progression (ε apart)	125	0.21	0.14	0.01	0.13	**0.04**
	250	1.30	0.73	0.04	0.99	**0.12**
	500	8.36	4.42	0.20	7.76	**0.40**
	1000	91.98	40.10	1.26	91.18	**1.51**
	2000	824.00	335.41	6.66	3212.80	**6.77**
Uniform Distribution (ε to 1)	125	0.11	0.10	0.05	0.13	**0.04**
	250	0.44	0.38	0.26	1.04	**0.11**
	500	1.81	1.55	1.63	7.78	**0.38**
	1000	91.74	6.25	12.87	91.65	**1.54**
	2000	823.63	336.04	161.60	1308.26	**6.34**
(1,2,1) Matrix	125	0.12	0.10	0.05	0.13	**0.02**
	250	0.44	0.38	0.17	0.94	**0.09**
	500	1.95	1.60	1.09	7.25	**0.33**
	1000	13.23	7.58	8.84	100.79	**1.41**
	2000	821.85	130.64	109.91	1737.15	**5.94**
Biphenyl	966	85.11	33.78	9.71	238.42	**2.41**

Table 1: Timing Results



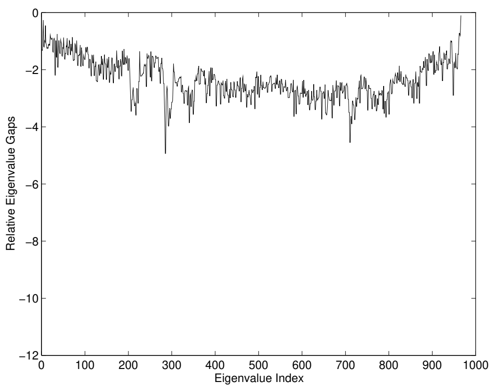


Figure 6: (a)Eigenvalue distribution, and (b)Relative Gaps for Biphenyl

that Algorithm Getvec is several times faster than DSTEDC on three of the four 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[3]. 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

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

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

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

The defect of the high level procedure mentioned above is that the singular values will be computed twice; a significant waste. The remedy is to apply the reversal mechanism locally. When an eigenvalue (σ^2) has been computed our algorithm 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 6, 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 previous paragraph is *not* adequate. It produces matrices U and V that are orthogonal to working precision but the extra coupling

[3]all timings were measured using Fortan BLAS on a 333-MHz UltraSPARC processor

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 an interesting recent dissertation [18], Benedict Grosser has presented coupling relations that connect factorizations of $LL^t - \mu^2 I$ and $L^t L - \mu^2 I$. By forcing these relations to hold for the computed factorizations he found a way to use our Algorithm Getvec and satisfy all the desired properties to working accuracy:

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

This algorithm is to become part of the LAPACK library.

Acknowledgements. 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 some delay, greatly improved the presentation of this paper.

References

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

[2] J. Barlow and J. Demmel, 'Computing Accurate eigensystems of scaled diagonally dominant matrices'. SIAM J. Num. Anal., vol. 27, (1990), pp. 762–791.

[3] J. R. Bunch, 'The weak and strong stability of algorithms in numerical linear algebra'. Lin. Alg. Appl., vol. 88/89 (1987), pp. 49–66.

[4] B. Char, K. Geddes, G. Gonnet, B. Leong, M. Monagan, and S. Watt, '*Maple V Library Reference Manual*'. Springer-Verlag, Berlin, 1991.

[5] T. S. Chihara, 'An Introduction to Orthogonal Polynomials'. Gordon and Breach, 1978.

[6] L. S. De Jong, 'Towards a formal definition of numerical stability'. Numer. Math., vol. 28 (1977), pp. 211–219.

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

[8] J. Demmel, *Applied Numerical Algebra.* SIAM Publications, 1997.

[9] I. S. Dhillon, 'A New $O(n^2)$ Algorithm for the Symmetric Tridiagonal Eigenvalue/Eigenvector Problem'. PhD thesis, Computer Science Division, University of California, Berkeley, May 1997. Also available as Computer Science Division Technical Report No. UCB//CSD-97-971.

[10] I.S. Dhillon and B.N. Parlett. Multiple representations for orthogonality. *Lin. Alg. Appl.*, 2001. Submitted for publication.

[11] S. Eisenstat and I. C. F. Ipsen, 'Relative perturbation bounds for eigenspaces and singular vector subspaces'. In J.G. Lewis, editor, Proceedings of the Fifth SIAM Conf. on Applied Linear Algebra, pp. 62–65. SIAM, 1994.

[12] S. Eisenstat and I. C. F. Ipsen, 'Relative perturbation techniques for singular value problems'. SIAM J. Numer. Anal., vol. 32 (1995).

[13] K. V. Fernando and B. N. Parlett, 'Accurate singular values and differential qd algorithms'. Numer. Math., vol. 67, (March 1994), no. 2, pp. 191–229.

[14] K. V. Fernando and B. N. Parlett, 'Implicit Cholesky algorithms for singular values and vectors of triangular matrices'.
Numer. Linear Algebra with Applications, vol. 2, no. 6 (1995), pp. 507–531.

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

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

[17] S. K. Godunov, A. G. Antonov, O. P. Kiriljuk, and V. I. Kostin, 'Guaranteed Accuracy in Numerical Linear Algebra'. Kluwer Academic, 1993. (A revised translation of a Russian text first published in 1988 in Novosibirsk.)

[18] B. Grosser and B. Lang. An $O(n^2)$ algorithm for the bidiagonal svd. *Lin. Alg. Appl.*, 2002. To appear.

[19] M. Gu and S. C. Eisenstat. A divide-and-conquer algorithm for the symmetric tridiagonal eigenproblem. *SIAM J. Mat. Anal. Appl.*, 16(1):172–191, January 1995.

[20] P. Henrici, 'The quotient-difference algorithm'. Nat. Bur. Standards Appl. Math. Series, vol. 49 (1958), pp. 23–46.

[21] P. Henrici, 'Applied and Computational Complex Analysis' vol. I. John Wiley & Sons, New York, 1974.

[22] 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).

[23] Ren-Cang Li, 'Relative perturbation theory: (I) Eigenvalue and singular value variations'. SIAM J. Matrix Anal. Appl., vol. 19 (1998), pp. 956–982.

[24] Ren-Cang Li, 'Relative perturbation theory: (II) Eigenspace and singular subspace variations'. SIAM J. Matrix Anal. Appl., vol. 20 (1998), pp. 471–492.

[25] Ren-Cang Li, 'Relative perturbation theory: (III) More bounds on eigenvalue variation'. Lin. Alg. Appl., vol. 266 (1997), pp. 337–345.

[26] B. N. Parlett, 'Reduction to tridiagonal form and minimal realizations'. SIAM J. Matrix Anal. Appl., vol. 13 (1992), no. 2, pp. 567–593.

[27] B. N. Parlett, 'The construction of orthogonal eigenvectors for tight clusters by use of submatrices'. Report CPAM-664 (January 1996).

[28] B. N. Parlett, 'Invariant subspaces for tightly clustered eigenvalues of tridiagonals'. BIT, vol. 36, no. 3 (1996), pp. 542–562.

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

[30] B. N. Parlett and I. S. Dhillon. Relatively robust representations of symmetric tridiagonals. *Lin. Alg. Appl.*, 309:121–151, 2000.

[31] B. N. Parlett, 'Spectral sensitivity of products of bidiagonals'. Lin. Alg. Appl., vol. 275-276 (May 1998), pp. 417–431.

[32] B. N. Parlett. Relative condition numbers for eigenpairs of symmetric tridiagonals. *Foundations of Computational Mathematics (issue dedicated to M. J. D. Powell)*, 2002. Submitted for publication.

[33] B. N. Parlett, 'The Symmetric Eigenvalue Problem'. (2nd Edition) SIAM, Philadelphia, 1998. 398 pp.

[34] H. Rutishauser, 'Der Quotienten-Differenzen-Algorithmus'. Z. Angew. Math. Phys., vol. 5 (1954), pp. 233–251.

[35] H. Rutishauser, 'Solution of eigenvalue problems with the LR-transformation'. Nat. Bur. Standards Appl. Math. Series, vol. 49 (1958), pp. 47–81.

[36] H. Rutishauser, *'Vorlesungen über Numerische Mathematik'*. Birkhäuser, Basel, 1976.

[37] H. Rutishauser, *'Lectures on Numerical Mathematics'*. Birkhäuser, Boston, 1990.

[38] Veselić, K. and Slapničar, I., 'Floating point perturbations of Hermitian matrices'. Lin. Alg. Appl., vol. 195 (1993), pp. 81–116.

[39] J. H. Wilkinson, *The Algebraic Eigenvalue Problem*, Clarendon Press, Oxford, 1965.

[40] S. Wolfram, '*Mathematica: A System for doing Mathematics by Computer*'. Addison-Wesley, Reading, MA, USA, 2nd ed., 1991.

[41] Yao Yang, 'Error Analysis of the dqds Algorithm'. Ph. D. Thesis, UCB, Berkeley, 1994.