**Electronic Transactions on Numerical Analysis.**
**Volume 38, pp. 363-400, 2011.**

**ISSN 1068-9613.**

**ETNA**
**Kent State University**
**http://etna.math.kent.edu**

# BLOCK FACTORIZATIONS AND QD-TYPE TRANSFORMATIONS FOR THE MR$^3$ ALGORITHM *

PAUL R. WILLEMS† AND BRUNO LANG‡

**Abstract.** Factorizing symmetric tridiagonal matrices and propagating the factorizations to shifted matrices are central tasks in the MR$^3$ algorithm for computing partial eigensystems. In this paper we propose block bidiagonal factorizations $\mathsf{LDL}^*$ with $1 \times 1$ and $2 \times 2$ blocks in $\mathsf{D}$ as an alternative to the bidiagonal and twisted factorizations used hitherto. With block factorizations, the element growth can be reduced (or avoided altogether), which is essential for the success of the MR$^3$ algorithm, in particular, if the latter is used to determine the singular value decomposition of bidiagonal matrices. We show that the qd algorithm used for shifting bidiagonal factorizations, e.g., $\mathsf{LDL}^* - \tau\mathsf{I} =: \mathsf{L}^+\mathsf{D}^+(\mathsf{L}^+)^*$ can be extended to work with blocks in a mixed stable way, including criteria for determining a suitable block structure dynamically.

**Key words.** symmetric tridiagonal matrix, eigensystem, MRRR algorithm, block bidiagonal factorizations, qd algorithm, theory and implementation

**AMS subject classifications.** 65F15, 65G50, 15A18

**1. Introduction.** The MR$^3$ (multiple relatively robust representations, MRRR) algorithm [3, 4, 5] allows one to solve the (partial) symmetric tridiagonal eigenvalue problem, TSEP, i.e., to compute selected eigenpairs $(\lambda_i, \mathsf{q}_i)$, $i \in I$, of a symmetric tridiagonal matrix $\mathsf{T} \in \mathbb{R}^{n\times n}$. The algorithm requires roughly $\mathcal{O}(|I| \cdot n)$ operations, which is asymptotically optimal.

From a distant point of view, MR$^3$ works as follows. Determine the eigenvalues of $\mathsf{T}$ to such precision that they can be classified as *singletons* (with sufficient *relative* distance to the other eigenvalues, e.g., agreement to at most three leading decimal digits) and *clusters*. For singletons $\lambda_i$, a variant of Rayleigh quotient iteration (RQI) and inverse iteration yields extremely accurate eigenpairs (tiny residuals). Then these eigenvectors are automatically orthogonal to working precision, thus removing any need for Gram–Schmidt orthogonalization and the burden of communication in a parallel implementation. Clusters $\lambda_i \approx \cdots \approx \lambda_{i+s}$ cannot be handled directly. Instead, for each cluster one chooses a *shift* $\tau \approx \lambda_i$ very close to (or even inside) the cluster and considers the matrix $\mathsf{T} - \tau\mathsf{I}$. The eigenvalues $\lambda_i - \tau, \ldots,$ $\lambda_{i+s} - \tau$ of that matrix will then enjoy much larger relative distances than $\lambda_i, \ldots, \lambda_{i+s}$ did, and therefore they may be singletons for $\mathsf{T} - \tau\mathsf{I}$, meaning that now eigenvectors can be computed very accurately. If some of these eigenvalues are still clustered, then the shifting is repeated. Proceeding this way amounts to traversing a so-called *representation tree* with the original matrix $\mathsf{T}$ at the root, and children of a node standing for shifted matrices due to clusters. The eigenvectors are computed at the leaves of the tree. An analysis of the MR$^3$ algorithm exposing five criteria that must be fulfilled in order to guarantee small residuals and good orthogonality may be found in [16, 17].

As the name conveys, the MR$^3$ algorithm is intimately coupled with the concept of *representations* of matrices. A representation is just a minimal set of numbers defining a matrix. For MR$^3$ to work, the transition from a node to its child, $\mathsf{T} - \tau\mathsf{I} =: \mathsf{T}^+$, must not change the invariant subspace of a cluster—and at least some of its eigenvalues—by too much. In other

†WestLB AG (willems@math.uni-wuppertal.de).
‡University of Wuppertal, Faculty of Mathematics and Natural Sciences, Gaußstr. 20, D-42097 Wuppertal (lang@math.uni-wuppertal.de).
*Received May 13, 2011. Accepted August 14, 2011. Published online December 20, 2011. Recommended by M. Hochstenbach. This work was carried out while P. Willems was with the Faculty of Mathematics and Natural Sciences at the University of Wuppertal. The research was partially funded by the Bundesministerium für Bildung und Forschung, contract number 01 IH 08 007 B, within the project *ELPA—Eigenwert-Löser für Petaflop-Anwendungen*.

ETNA
Kent State University
http://etna.math.kent.edu



words, the employed representations at the nodes must be *relatively robust*. In general, this robustness cannot be achieved if one works directly with the "default representation" of a symmetric tridiagonal matrix by its $2n-1$ entries because they do not necessarily determine small eigenvalues to high relative precision. Alternative representations have been found to be superior when it comes to relative robustness. Prominent examples are the entries of lower bidiagonal factorizations

$$\mathsf{T} = \mathsf{LDL}^* = \begin{bmatrix} 1 & & & \\ \ell_1 & 1 & & \\ & \ddots & \ddots & \\ & & \ell_{n-1} & 1 \end{bmatrix} \begin{bmatrix} d_1 & & & \\ & d_2 & & \\ & & \ddots & \\ & & & d_n \end{bmatrix} \begin{bmatrix} 1 & \ell_1 & & \\ & 1 & \ddots & \\ & & \ddots & \ell_{n-1} \\ & & & 1 \end{bmatrix}$$

and the analogous upper bidiagonal factorizations $\mathsf{T} = \mathsf{URU}^*$ (obtained by starting the factorization at the bottom of the matrix). A generalization of both leads to the so-called *twisted* factorizations $\mathsf{T} = \mathsf{N}_k \mathsf{G}_k \mathsf{N}_k^*$, which one gets by "fusing" together the upper part of an $\mathsf{LDL}^*$ and the lower part of a $\mathsf{URU}^*$ decomposition. Note that we write $^*$ for the transpose of a matrix.

The sole possible cause of trouble with the MR$^3$ algorithm is that relatively robust representations cannot always be guaranteed a priori in a practicable way. Robustness is intimately linked to *element growth* when forming the bidiagonal (or twisted) factorizations. Element growth means that some of the data representing these factorizations are substantially larger than the entries in the tridiagonal matrix.

In [16, Chapter 3] it was shown how the MR$^3$ algorithm can also be used to compute the bidiagonal singular value decomposition, BSVD, for an upper bidiagonal matrix $\mathsf{B}$, i.e., to determine singular values $\sigma_i \geq 0$ and the corresponding left and right singular vectors $\mathsf{u}_i$, $\mathsf{v}_i$ such that $\mathsf{B}\mathsf{v}_i = \mathsf{u}_i\sigma_i$. It is well-known that the BSVD is tightly coupled with the TSEP for the so-called Golub–Kahan matrix $\mathsf{T}_{\text{GK}}$, which is obtained by interleaving the diagonal and subdiagonal elements of $\mathsf{B}$ on the subdiagonal of a double-sized matrix:

$$\mathsf{B} = \begin{bmatrix} a_1 & b_1 & & \\ & a_2 & \ddots & \\ & & \ddots & b_{n-1} \\ & & & a_n \end{bmatrix} \quad \rightsquigarrow \quad \mathsf{T}_{\text{GK}} = \begin{bmatrix} 0 & a_1 & & & & & \\ a_1 & 0 & b_1 & & & & \\ & b_1 & 0 & a_2 & & & \\ & & a_2 & 0 & \ddots & & \\ & & & \ddots & \ddots & b_{n-1} & \\ & & & & b_{n-1} & 0 & a_n \\ & & & & & a_n & 0 \end{bmatrix}.$$

Extracting the odd- and even-numbered components of $\mathsf{T}_{\text{GK}}$'s eigenvectors into $\mathsf{u}$ and $\mathsf{v}$ indeed gives numerically orthogonal singular vectors *if* the essential property of the matrix $\mathsf{T}_{\text{GK}}$ having a *nearly constant diagonal (ncd)* can be maintained during the initial factorization and the ensuing shifting [16, Chapter 3]. Therefore, in the context of the BSVD the problem of element growth is magnified since local element growth in successive translates of a Golub–Kahan matrix can cause loss of the ncd property.

This paper is devoted to a technique that reduces the problems involving element growth. The approach is to allow $2 \times 2$ pivots, or *blocks*, in $\mathsf{D}$, leading to a *block factorization* (BF). Before going into details we illustrate the benefits of BFs with two examples.

EXAMPLE 1.1. A Golub–Kahan matrix does not admit a bidiagonal (or twisted) factorization, because the first pivot is zero regardless of where you start. The natural way to

ETNA
Kent State University
http://etna.math.kent.edu



factorize an (unreduced) Golub–Kahan matrix is using $2\times 2$ pivots all the way:

$$\begin{bmatrix} 0 & a_1 & & \\ a_1 & 0 & b_1 & \\ & b_1 & 0 & a_2 \\ & & a_2 & 0 \end{bmatrix} = \mathsf{LDL}^* \quad \text{with} \quad \mathsf{L} = \begin{bmatrix} 1 & & & \\ & 1 & & \\ \frac{b_1}{a_1} & & 1 & \\ & & & 1 \end{bmatrix}, \quad \mathsf{D} = \begin{bmatrix} 0 & a_1 & & \\ a_1 & 0 & & \\ & & 0 & a_2 \\ & & a_2 & 0 \end{bmatrix}.$$

At no time was there a choice to take a single pivot; the block structure is indeed unique.

EXAMPLE 1.2. Let $\alpha \ll 1$ and consider the bidiagonal matrix $\mathsf{B} = \left[\begin{smallmatrix} 1 & 1 \\ & \alpha \end{smallmatrix}\right]$ with singular values $\sigma_1 \approx \alpha$, $\sigma_2 \approx 2$. Shifting the corresponding Golub–Kahan matrix by $-\alpha$ gives a bidiagonal factorization

$$\begin{bmatrix} -\alpha & 1 & & \\ 1 & -\alpha & 1 & \\ & 1 & -\alpha & \alpha \\ & & \alpha & -\alpha \end{bmatrix} = \mathsf{LDL}^*$$

with $\mathsf{D} = \operatorname{diag}\big(-\alpha,\ \frac{1-\alpha^2}{\alpha},\ -\alpha\frac{2-\alpha^2}{1-\alpha^2},\ -\alpha\frac{1}{2-\alpha^2}\big)$. Clearly there is a huge local element growth in $\mathsf{D}(2)$. The representation $\mathsf{LDL}^*$ still is ncd, but if we had to shift it again, the property would probably be lost completely due to rounding errors.

Using $2\times 2$ pivots, the same matrix can be factorized as follows:

$$\begin{bmatrix} -\alpha & 1 & & \\ 1 & -\alpha & 1 & \\ & 1 & -\alpha & \alpha \\ & & \alpha & -\alpha \end{bmatrix} = \mathsf{LDL}^* \quad \text{with} \quad \mathsf{L} = \begin{bmatrix} 1 & & & \\ & 1 & & \\ k_1 & \ell_2 & 1 & \\ & & \ell_3 & 1 \end{bmatrix}, \quad \mathsf{D} = \begin{bmatrix} d_1 & 1 & & \\ 1 & c_2 & & \\ & & d_3 & \\ & & & d_4 \end{bmatrix},$$

where $k_1 = \frac{1}{1-\alpha^2}$, $\ell_2 = \frac{\alpha}{1-\alpha^2}$, $\ell_3 = -\frac{1-\alpha^2}{2-\alpha^2}$, $d_1 = -\alpha$, $c_2 = -\alpha$, $d_3 = -\alpha\frac{2-\alpha^2}{1-\alpha^2}$, and $d_4 = -\alpha\frac{1}{2-\alpha^2}$. For this case the block structure is not unique, but the chosen one makes the most sense, showing no discernible element growth at all.

For factorizing a symmetric tridiagonal matrix $\mathsf{T}$ given by its entries as $\mathsf{T} = \mathsf{LDL}^*$, block factorizations have been employed with great success [7, 10]. Higham shows in [11] how $2\times 2$ pivots can be used to solve a symmetric tridiagonal linear equation system in a normwise backward stable way.

The idea to employ block factorizations as representations within MR$^3$ and its derivatives is not new. However, up to now it was not known how to do so. One of the five criteria that are necessary for MR$^3$ to work, coined SHIFTREL in [17], states that we must be able to shift representations in a componentwise mixed relatively stable way, that is, we need to compute

$$\mathsf{LDL}^* - \tau\mathsf{I} \ =: \ \mathsf{L}^+\mathsf{D}^+(\mathsf{L}^+)^* \tag{1.1}$$

such that small relative changes to the inputs (the representation of $\mathsf{LDL}^*$) and outputs (the representation of $\mathsf{L}^+\mathsf{D}^+(\mathsf{L}^+)^*$) give an exact relation. To achieve this requirement for the unblocked case, with diagonal matrices $\mathsf{D}$ and $\mathsf{D}^+$, the inventors of MR$^3$ developed the so-called *differential stationary quotient-differences with shift* algorithm `dstqds`, and variants thereof [3, 5]. As representations they employed the entries $d_i$, $\ell_i$, $d_i^+$, $\ell_i^+$ of the factors involved. In another paper [18], we have studied variations on this theme. We found that, even while working with the same matrices $\mathsf{LDL}^*$ and $\mathsf{L}^+\mathsf{D}^+(\mathsf{L}^+)^*$, it may be beneficial to define the representations differently. One example having a direct impact on the following pages is what we call an $e$–representation: use the offdiagonal elements of the tridiagonal matrix instead of the entries $\ell_i$, $\ell_i^+$ of the bidiagonal factors. The motivation for doing so is

```
Input:   (d_1, ..., d_n), (e_1, ..., e_{n-1}),
         where e_i = ℓ_i d_i, shift τ
Output:  (d_1^+, ..., d_n^+)

s_1 := 0
for i = 1 to n − 1 do
    d_i^+ := d_i + (s_i − τ)
    s_{i+1} := e_i^2 (s_i − τ)/(d_i d_i^+)
endfor
d_n^+ := d_n + (s_n − τ)
```

$$\begin{tikzcd}[column sep=huge, row sep=huge]
\mathsf{LDL}^* \arrow[r, "\texttt{dstqds}", "\text{computed}"'] \arrow[d, squiggly, "\begin{array}{r}\text{Perturb}\\ d_i \text{ by } 1\epsilon_\diamond,\\ e_i \text{ by } 3\epsilon_\diamond\end{array}"'] & \mathsf{L}^+\mathsf{D}^+(\mathsf{L}^+)^* \arrow[d, squiggly, "\begin{array}{l}\text{Perturb}\\ d_i^+ \text{ by } 2\epsilon_\diamond\end{array}"] \\
\widetilde{\mathsf{L}}\widetilde{\mathsf{D}}\widetilde{\mathsf{L}}^* \arrow[r, "-\tau", "\text{exact}"'] & \widetilde{\mathsf{L}}^+\widetilde{\mathsf{D}}^+(\widetilde{\mathsf{L}}^+)^*
\end{tikzcd}$$

FIG. 1.1. Left-hand side: *Algorithm* `dstqds` *to compute* $\mathsf{L}^+\mathsf{D}^+(\mathsf{L}^+)^* = \mathsf{LDL}^* - \tau$ *for e–representations. The offdiagonal elements* $e_i$ *are unchanged by shifting, thus they can be reused in the representation of* $\mathsf{L}^+\mathsf{D}^+(\mathsf{L}^+)^*$. Right-hand side: *Mixed relative error analysis of* `dstqds`*, cf. [18, Theorem 5.3].* $\epsilon_\diamond$ *stands for machine epsilon, cf. Section 3.1. Only first-order bounds are shown, that is,* $e_i$ *may be perturbed by* $3\epsilon_\diamond + \mathcal{O}(\epsilon_\diamond^2)$.

that shifting does not change the offdiagonal entries, i.e., the same data $e_i = \ell_i d_i = \ell_i^+ d_i^+$ can be used for representing $\mathsf{LDL}^*$ and $\mathsf{L}^+\mathsf{D}^+(\mathsf{L}^+)^*$. An adapted version of `dstqds` is shown in Figure 1.1, together with the results of a mixed relative error analysis. For full details and proofs we refer the reader to [18]. Indeed, the paper [18] can be seen as a foundation for the present one and we will have to refer to it occasionally. Nevertheless, to ensure that this paper is stand-alone the necessary techniques and results will be recalled where needed, if only quite tersely.

The goal to achieve mixed relative stability for (1.1) becomes rather more intricate if $\mathsf{D}$ and $\mathsf{D}^+$ are block-diagonal of bandwidth one, maybe even with non-conforming structure. We have devised a new algorithm to compute (1.1), with the feature to change the block structure from $\mathsf{D}$ to $\mathsf{D}^+$ on-the-fly. One could call it blocked `dstqds`. After studying some general structural properties of block factorizations with $2 \times 2$ pivots in Section 2, we will present the algorithm in Section 3 and provide a complete relative error analysis. Finally, Section 4 contains numerical experiments to show how the MR$^3$-based methods for TSEP and BSVD can benefit from using block factorizations.

To the best of our knowledge, most of the contents of this paper are new. Many details have been inspired by private communications with Beresford Parlett. The only previous work pertaining to qd-like algorithms for block factorizations that we know of is unpublished work by Carla Ferreira and Lisa Miranian [8]. We took the motivation for the conditions when to choose a $2 \times 2$ pivot (Block Criterion I in Section 3.2) from there, but except for that, the approach we take is different.

## 2. Properties and representation of block factorizations.

In this section we will derive the structural properties of block factorizations with $1 \times 1$ and $2 \times 2$ pivots and introduce the quantities we have selected to represent such factorizations in a non-redundant way.

### 2.1. The structure of block factorizations with $2 \times 2$ pivots.

For a given symmetric tridiagonal matrix $\mathsf{T}$, we denote the diagonal and offdiagonal entries as $c_i$ and $e_i$, respectively:

$$\mathsf{T} = \begin{bmatrix} c_1 & e_1 & & & \\ e_1 & c_2 & \ddots & & \\ & \ddots & \ddots & \ddots & \\ & & \ddots & c_{n-1} & e_{n-1} \\ & & & e_{n-1} & c_n \end{bmatrix} \in \mathbb{R}^{n \times n}.$$

ETNA
Kent State University
http://etna.math.kent.edu



We only consider unreduced matrices, that is, $e_i \neq 0$ for $i = 1, \dots, n-1$.

Suppose we have a decomposition $\mathsf{T} = \mathsf{LDL}^*$ with a unit lower *triangular* $\mathsf{L}$ and a *block*-diagonal $\mathsf{D}$ with blocks of size one or two, i.e.,

$$\mathsf{D} = \operatorname{diag}(\mathsf{D}_1, \dots, \mathsf{D}_N), \quad \text{where } \mathsf{D}_j \in \mathbb{R}^{\mathrm{size}(j)\times\mathrm{size}(j)}, \quad \mathrm{size}(j) \in \{1,2\}.$$

Partition the matrices $\mathsf{L}$ and $\mathsf{T}$ conformably, with $\mathsf{T}_{k,j}$ and $\mathsf{L}_{k,j}$ referring to individual blocks of the lower triangles ($k \geq j$). Let $\mathsf{L}^*_{k,j} := (\mathsf{L}_{k,j})^* = (\mathsf{L}^*)_{j,k}$, which is different from $(\mathsf{L}^*)_{k,j}$.

As $\mathsf{L}$ is unit triangular, the blocks $\mathsf{L}_{j,j}$ must be nonsingular. Multiply the $j$th block column of $\mathsf{L}$ by $\mathsf{L}^{-1}_{j,j}$ and take $\mathsf{L}_{j,j}\mathsf{D}_j\mathsf{L}^*_{j,j}$ instead of $\mathsf{D}_j$ to see that, without loss of generality, we can assume them to be identities, i.e.,

$$\mathsf{L}_{j,j} = \mathsf{I}_{\mathrm{size}(j)}, \quad j = 1, \dots, N.$$

Using the block triangular structure of $\mathsf{L}$, the relation $\mathsf{T} = \mathsf{LDL}^*$ becomes

$$\mathsf{T}_{k,j} = \sum_{i=1}^{\min\{j,k\}} \mathsf{L}_{k,i}\mathsf{D}_i\mathsf{L}^*_{j,i}, \qquad 1 \leq j,k \leq N. \tag{2.1}$$

The following lemma summarizes some properties of block factorizations, in particular that they can only exist if the diagonal blocks $\mathsf{D}_j$ are nonsingular, except possibly for $\mathsf{D}_N$.

LEMMA 2.1. *Let the unreduced tridiagonal* $\mathsf{T}$ *have a block factorization* $\mathsf{T} = \mathsf{LDL}^*$ *as outlined above. Then* $\mathsf{T}_{1,1} = \mathsf{D}_1$*, and for* $j = 1, \dots, N-1$ *the following holds:*

*(i)* $\mathsf{T}_{j+1,j} = \mathsf{L}_{j+1,j}\mathsf{D}_j$.
*(ii)* $\mathsf{D}_j$ *is nonsingular. (* $\mathsf{D}_N$ *may be singular.)*
*(iii)* $\mathsf{L}_{k,j} = 0$ *for* $k > j+1$.
*(iv)* $\mathsf{T}_{j+1,j+1} = \mathsf{D}_{j+1} + s\,\mathsf{e}_1\mathsf{e}_1^*$ *with* $s \in \mathbb{R}$, $\mathsf{e}_1 = \mathsf{I}_{\mathrm{size}(j+1)}(:,1)$.

*Proof.* We will proceed by induction on $j$. For $j = 0$ the claims hold because (i)–(iii) are void, and (2.1) gives $\mathsf{T}_{1,1} = \mathsf{D}_1$, which also is (iv) with $s = 0$.

Now let $j \geq 1$ and assume that the claims hold for all $j' < j$. Invoking (2.1) and making use of (iii) for all $1 \leq j' < j$ gives us (i). From that and $\mathsf{T}$ being unreduced we have

$$\mathsf{L}_{j+1,j}\mathsf{D}_j = \mathsf{T}_{j+1,j} \neq 0, \tag{2.2}$$

and therefore $\mathsf{D}_j \neq 0$. Hence, $\mathsf{D}_j$ can be singular only if $\mathrm{size}(j) = 2$ and $\mathsf{D}_j$ has rank one. Using the induction hypothesis on (iv) for $j-1$ yields $\mathsf{T}_{j,j} = \mathsf{D}_j + s\,\mathsf{e}_1\mathsf{e}_1^*$, which implies $\mathsf{D}_j(2,1) \neq 0$, again due to the irreducibility of $\mathsf{T}$. Thus

$$\mathsf{D}_j = (\mathsf{v}, \beta\mathsf{v}) \quad \text{for some } 0 \neq \mathsf{v} \in \mathbb{R}^2, \beta \in \mathbb{R}. \tag{2.3}$$

Since $\mathsf{T}_{j+1,j}$ has only zeros in its first column, we have $\mathsf{L}_{j+1,j}\mathsf{v} = 0$ by (i). But then (2.3) implies $\mathsf{L}_{j+1,j}\mathsf{D}_j = 0$, a contradiction to (2.2). So $\mathsf{D}_j$ must be nonsingular after all.

Since $\mathsf{T}$ is tridiagonal, we have $0 = \mathsf{T}_{k,j} = \mathsf{L}_{k,j}\mathsf{D}_j$ for all $k > j+1$, by virtue of (2.1) and (iii) holding for $j' < j$. Together with (ii) this gives us (iii) for $j$.

For (iv) there is only something left to prove if $\mathrm{size}(j) = 2$. Let $e_i = \mathsf{T}_{j+1,j}(1,2)$ denote the one nonzero entry in $\mathsf{T}_{j+1,j}$ (top right position). Then the nonsingularity of $\mathsf{D}_j$ and (i) yields

$$\mathsf{L}_{j+1,j}\mathsf{D}_j\mathsf{L}^*_{j+1,j} = \mathsf{L}_{j+1,j}\mathsf{D}_j\mathsf{D}_j^{-1}\mathsf{D}_j\mathsf{L}^*_{j+1,j} = \mathsf{T}_{j+1,j}\mathsf{D}_j^{-1}\mathsf{T}^*_{j+1,j} = \underbrace{e_i^2\mathsf{D}_j^{-1}(2,2)}_{=:s}\,\mathsf{e}_1\mathsf{e}_1^*,$$

which gives us (iv). □

Thus, $\mathsf{L}$ has in fact only $N-1$ nontrivial blocks, making double indices superfluous. With $\mathsf{L}_j := \mathsf{L}_{j+1,j} \in \mathbb{R}^{\mathrm{size}(j+1)\times\mathrm{size}(j)}$ and $\mathsf{I}_j := \mathsf{I}_{\mathrm{size}(j)}$ we can summarize the situation as

$$\mathsf{T} = \begin{bmatrix} \mathsf{I}_1 & & & \\ \mathsf{L}_1 & \mathsf{I}_2 & & \\ & \ddots & \ddots & \\ & & \mathsf{L}_{N-1} & \mathsf{I}_N \end{bmatrix} \begin{bmatrix} \mathsf{D}_1 & & & \\ & \mathsf{D}_2 & & \\ & & \ddots & \\ & & & \mathsf{D}_N \end{bmatrix} \begin{bmatrix} \mathsf{I}_1 & \mathsf{L}_1^* & & \\ & \mathsf{I}_2 & \ddots & \\ & & \ddots & \mathsf{L}_{N-1}^* \\ & & & \mathsf{I}_N \end{bmatrix}.$$

Glancing at this formula one might think that $\mathsf{L}$ has bandwidth three. In fact, by (i) and (ii) of Lemma 2.1 we have $\mathsf{L}_{j+1,j} = \mathsf{T}_{j+1,j}\mathsf{D}_j^{-1}$, and since only the top right entry of $\mathsf{T}_{j+1,j}$ is nonzero, only the first row of $\mathsf{L}_{j+1,j}$ can be nonzero. This reveals the rather special structure of $\mathsf{L}$: a bandwidth bounded by two but nevertheless only $n - \mathrm{size}(N) \le n-1$ nontrivial (meaning nonzero and offdiagonal) entries, at most two in each block. In particular, $\mathsf{L}$ has less than $n-1$ nontrivial entries if and only if $\mathsf{D}$ ends with a $2\times 2$ block.

Now we will look more closely at the block entries and how they relate to the entries of $\mathsf{T}$. Recall that property (iv) of Lemma 2.1 revealed that $\mathsf{D}_j$ has at most the top left entry not being also an entry of $\mathsf{T}$. We will denote this qualifying feature of $\mathsf{D}_j$ as $d_i$, where $i = 1 + \sum_{k=1}^{j-1}\mathrm{size}(k)$. Then we have

$$\mathsf{D}_j = \begin{cases} d_i, & \mathrm{size}(j) = 1, \\ \begin{bmatrix} d_i & e_i \\ e_i & c_{i+1} \end{bmatrix}, & \mathrm{size}(j) = 2. \end{cases} \tag{2.4}$$

It was already stated that $\mathsf{L}$ has at most $n-1$ nontrivial entries. Depending on the structure, we will use letters $k$ and $\ell$ to refer to them, more precisely

$$\mathsf{L}_j = \begin{cases} \ell_i, & \mathrm{size}(j) = \mathrm{size}(j+1) = 1, \\ \begin{bmatrix} \ell_i \\ 0 \end{bmatrix} & \mathrm{size}(j) = 1,\ \mathrm{size}(j+1) = 2, \\ \begin{bmatrix} k_i & \ell_{i+1} \end{bmatrix}, & \mathrm{size}(j) = 2,\ \mathrm{size}(j+1) = 1, \\ \begin{bmatrix} k_i & \ell_{i+1} \\ 0 & 0 \end{bmatrix}, & \mathrm{size}(j) = \mathrm{size}(j+1) = 2. \end{cases} \tag{2.5}$$

With our definition, for each index $i$ there exists either a $d_i$ or a $c_i$, and either a $k_i$ or an $\ell_i$, but never both. We have two reasons for distinguishing between $d$'s and $c$'s for $\mathsf{D}$, and between the $k$'s and $\ell$'s for $\mathsf{L}$ instead of using $d_1,\dots,d_n$ for the diagonal and $\ell_1,\dots,\ell_{n-\mathrm{size}(N)}$ for the nontrivial entries in $\mathsf{L}$ (as in [8]). First, the respective two quantities have very differing semantics, e.g., a $c_i$ is also a diagonal entry of $\mathsf{T}$, but a $d_i$ is not. This distinction will become more pronounced as we go on. The second reason is clarity of presentation. Employing block factorizations inevitably involves case distinctions to deal with the block structure, which can become confusing at points. Separating $\mathrm{diag}(\mathsf{D})$ into $d_i$'s and $c_i$'s, and $\mathsf{L}$ into $k_i$'s and $\ell_i$'s lets formulae carry the block structure implicitly, whereas using just $d_i$ and $\ell_i$ does not.

Henceforward, our treatment is based on entries (indices $i$) instead of blocks $\mathsf{D}_j$ and block numbers $j$. We will also make a slight but crucial adjustment in terminology in denoting the $\mathsf{D}_j$'s as *pivots* from now on and use *block* synonymously to $2\times 2$ pivot or $2\times 2$ block; a $\mathsf{D}_j$ with $\mathrm{size}(j) = 1$ is just a $1\times 1$ or *single* pivot (but *not* a block anymore). Thus, we can categorize the role of an index $i \in \{1,\dots,n\}$ into exactly one of either

ETNA
Kent State University
http://etna.math.kent.edu



- being *single*, with data $d_i$ and $\ell_i$ (the latter only if $i < n$);
- *starting a block*, with data $d_i$ and $k_i$ (the latter only if $i < n-1$);
- *ending a block*, with data $c_i$ and $\ell_i$ (the latter only if $i < n$).

The determinants of $2\times2$ pivots will play an important role, so we let

$$\Delta_i \;:=\; d_i c_{i+1} - e_i^2 \tag{2.6}$$

for each $i$ that starts a block. Based on these we define for $i = 1,\dots,n-1$ the quantity

$$\mathrm{inv}_{\mathsf{D}}(i) \;:=\; \begin{cases} 1/d_i, & \text{if } i \text{ is single,}\\ 0, & \text{if } i \text{ starts a block,}\\ d_{i-1}/\Delta_{i-1}, & \text{if } i \text{ ends a block.}\end{cases} \tag{2.7}$$

This definition is always proper if the factorization exists. Using Lemma 2.1, the relation between the diagonals of $\mathsf{D}$ and $\mathsf{T}$ can now be stated compactly as

$$\mathsf{T}(i,i) \;\equiv\; \mathsf{D}(i,i) + e_{i-1}^2\mathrm{inv}_{\mathsf{D}}(i-1), \tag{2.8}$$

which includes the special case $\mathsf{T}(1,1) = \mathsf{D}(1,1) = d_1$ if we assume quantities with "out-of-range" indices to be zero. Concerning $\mathsf{L}$, point (i) in Lemma 2.1 gives the characterization

$$\begin{cases} k_i \;=\; -e_{i+1}e_i/\Delta_i, & \text{if } i \text{ starts a block and } i < n-1,\\ \ell_i \;=\; e_i\mathrm{inv}_{\mathsf{D}}(i), & \text{if } i \text{ does not start a block.}\end{cases} \tag{2.9}$$

We close our introduction to $2\times2$ pivots with two more examples.

EXAMPLE 2.2. The structural properties (2.4) and (2.5) do not guarantee the product $\mathsf{LDL}^*$ to be a tridiagonal matrix. Consider the factors from Example 1.2 and change $\mathsf{L}$ to $\widetilde{\mathsf{L}}$ just by replacing $\ell_2$ with $\ell_2(1+\epsilon)$ for some $\epsilon \neq 0$. This results in

$$(\widetilde{\mathsf{L}}\mathsf{D}\widetilde{\mathsf{L}}^*)(3,1) \;=\; \ell_2\epsilon \;\neq\; 0.$$

Hence, relative perturbations to the nontrivial entries of $\mathsf{L}$ can destroy the tridiagonal structure. This happens only if the perturbations are uncorrelated. For the case above, we should have perturbed $k_1$ to $k_1(1+\epsilon)$ at the same time to retain the structure.

EXAMPLE 2.3. We mentioned that the factor $\mathsf{L}$ can have less than $n-1$ nontrivial entries if (and only if) a block ends at $n$. The following $3\times3$ problem demonstrates this:

$$\begin{bmatrix} 1 & 2 & \\ 2 & 3 & 5\\ & 5 & 4\end{bmatrix} = \begin{bmatrix} 1 & & \\ 2 & 1 & \\ & & 1\end{bmatrix}\begin{bmatrix} 1 & & \\ & -1 & 5\\ & 5 & 4\end{bmatrix}\begin{bmatrix} 1 & 2 & \\ & 1 & \\ & & 1\end{bmatrix} =:\; \mathsf{LDL}^*.$$

**2.2. A representation for block factorizations.** To define a specific block factorization, we first need a means to capture the block structure, i.e., the "type" of each index. In order to achieve minimality of information, we just just collect the indices where a $2\times2$ pivot *ends* in a set $\Omega \subseteq \{2,\dots,n\}$. By construction, $i \in \Omega$ implies $i-1 \notin \Omega$ and $i+1 \notin \Omega$, and therefore $\Omega$ cannot contain more than $\lfloor n/2 \rfloor$ elements.

DEFINITION 2.4. *The* standard representation *for a top-down BF* $\mathsf{T} = \mathsf{LDL}^*$ *is given by*

- $\Omega$, *the block structure (indices where a* $2\times2$ *pivot ends),*
- $e_1,\dots,e_{n-1}$, *the offdiagonal elements of* $\mathsf{T}$,
- $\mathsf{D}(1,1),\dots,\mathsf{D}(n,n)$, *the diagonal entries of* $\mathsf{D}$.

ETNA
Kent State University
http://etna.math.kent.edu



Note that this in fact generalizes the $e$–representation for non-blocked decompositions discussed in [18], as for the case that only single pivots are employed ($\Omega = \emptyset$), the same data items are kept. Using the set $\Omega$ one can tell whether a $\mathsf{D}(i,i)$ is actually a $d_i$ (if $i \notin \Omega$) or a $c_i$ (if $i \in \Omega$).

In [18] it proved very useful to distinguish between *primary* data items, which are actually stored and may be perturbed independently from each other to prove mixed stability, and *secondary* data, which are computed from the primary quantities and "inherit" their perturbations. With respect to the above standard representation for block factorizations, the quantities $\Delta_i$, $\mathrm{inv}_{\mathsf{D}}(i)$, as well as the entries $k_i$ and $\ell_i$ of $\mathsf{L}$, are secondary data and can be derived according to (2.6), (2.7) and (2.9), respectively.

REMARK 2.5 (Preserving tridiagonal form). It is natural to ask why none of the entries of $\mathsf{L}$ is used in our representation. Indeed, why not represent a block factorization using the nontrivial entries of $\mathsf{D}$ and $\mathsf{L}$, as it is usually done for standard bidiagonal decompositions? Such a representation would effectively contain *five* numbers for each $2 \times 2$ pivot (except the $N$th), namely $d_i$, $c_{i+1}$, $e_i$, $k_i$, and $\ell_{i+1}$. But these quantities are not independent, since they have to obey (2.9). Thus, basing a componentwise error analysis on such a representation would be fatal, as uncorrelated perturbations to all five data items at once will in general cause (2.9) not to be satisfied any more. The effect was already exhibited in Example 2.2: loss of tridiagonal structure, due to fill-in in $\mathsf{L}_j\mathsf{D}_j$.

One possible alternative to Definition 2.4 would be to use all entries from $\mathsf{D}$, but only the $\ell_i$ of $\mathsf{L}$, i.e., the data $\{d_i, \ell_{i-1} \,|\, i \notin \Omega\}$ and $\{e_{i-1}, c_i \,|\, i \in \Omega\}$. We prefer the data in Definition 2.4 because the offdiagonal entries remain unchanged by shifting. Thus, maximal use of the offdiagonal elements is good for efficiency and reduces the number of perturbations to specify in the mixed error analysis of the shifting algorithm. This will save us a lot of work on the following pages.

**2.3. The connection between blocked and non-blocked.** Allowing $2 \times 2$ pivots in $\mathsf{D}$ forfeits uniqueness, in the sense that multiple block structures may be possible for the same symmetric tridiagonal matrix, including for example using no blocks at all. Let $\mathsf{T}$ have a block factorization $\mathsf{T} = \mathsf{LDL}^*$ as before but also a non-blocked lower bidiagonal factorization $\mathsf{T} = \hat{\mathsf{L}}\hat{\mathsf{D}}\hat{\mathsf{L}}^*$ with diagonal $\hat{\mathsf{D}}$. Using (2.8) and (2.9), the following relations between the elements are easily derived by induction on $i$:

$$\hat{d}_i \equiv \begin{cases} \Delta_{i-1}/d_{i-1}, & \text{if } i \text{ ends a block in } \mathsf{D}, \\ d_i, & \text{otherwise}, \end{cases}$$

$$\hat{\ell}_i \equiv \begin{cases} -k_i/\ell_{i+1}, & \text{if } i \neq n-1 \text{ starts a block in } \mathsf{D}, \\ e_{n-1}/d_{n-1}, & \text{if } i = n-1 \text{ starts a block in } \mathsf{D}, \\ \ell_i, & \text{otherwise}. \end{cases}$$

Clearly, the only case where a blocked $\mathsf{T} = \mathsf{LDL}^*$ exists but a non-blocked $\mathsf{T} = \hat{\mathsf{L}}\hat{\mathsf{D}}\hat{\mathsf{L}}^*$ does not, occurs when a block-starting $d_i$ from $\mathsf{D}$ is zero. These relations were another reason for us to keep $c_i$ and $k_i$ separate from $d_i$ and $\ell_i$, as only the latter are identical to the corresponding data in a non-blocked factorization of the same matrix.

**2.4. When to choose a 2-by-2 pivot and when not.** For factorizing $\mathsf{T} = \mathsf{LDL}^*$—with or without shift and not regarding how $\mathsf{T}$ is represented—the motivation for allowing a $2 \times 2$ pivot in $\mathsf{D}$ covering indices $i$ and $i+1$ is to avoid what would otherwise become a very "large" single pivot $d_{i+1}$. How to gauge what is "large" depends on the application. In the past, a variety of schemes have been devised to evaluate when selecting a 2×2 pivot; see, e.g., [7, 11].

ETNA
Kent State University
http://etna.math.kent.edu



They all essentially try to avoid global element growth, that is, they would compare $d_{i+1}$ to the norm $\|\mathsf{T}\|$ or to a quantity of comparable magnitude, such as the spectral diameter.

Alternatively, one can evaluate the *local* element growth caused by the potential single pivot $d_{i+1}$, by comparing it directly to the concerned diagonal entry $\mathsf{T}(i+1, i+1)$. This would become the lower right entry $c_{i+1}$ of a block, should one be chosen. If not, the single pivot $d_{i+1}$ is given by

$$\mathsf{T}(i+1, i+1) \;=\; d_{i+1} + e_i^2/d_i.$$

Hence, if $d_{i+1}$ exceeds $\mathsf{T}(i+1, i+1)$ in magnitude by far, there has to be a cancellation between $d_{i+1}$ and $e_i^2/d_i$, so $e_i^2/d_i$ must also exceed $\mathsf{T}(i+1, i+1)$ in magnitude. The latter is preferable for a test, since it does not require to compute $d_{i+1}$. All in all, this motivates that each $2\times 2$ pivot $\begin{bmatrix} d_i & e_i \\ e_i & c_{i+1} \end{bmatrix}$ in $\mathsf{D}$ should fulfill

$$|d_i c_{i+1}| \;<\; K_\square e_i^2 \tag{2.10}$$

for some constant $K_\square \in (0, 1)$.

None of the above-mentioned pivoting strategies aiming at global element growth would ever choose a $2\times 2$ pivot violating (2.10) because, basically, avoiding local element growth requires the most $2\times 2$ pivots. Since the error analysis to come does only require that each selected $2\times 2$ pivot obeys (2.10), it carries over to other (more lax) pivoting schemes.

Requirement (2.10) is closely linked to the computation of the block determinants $\Delta_i$. Recall that for any real numbers $x$ and $y$, the condition number for the addition or subtraction $x \pm y$ is given by

$$\kappa_\pm(x, y) = \frac{|x| + |y|}{|x \pm y|} \tag{2.11}$$

because perturbing $x$ to $x(1+\xi)$ and $y$ to $y(1+\eta)$ leads to a result $(x \pm y)(1+\zeta)$, where $|\zeta| \le \kappa_\pm(x, y) \cdot \max\{|\xi|, |\eta|\}$. Thus, if (2.10) is fulfilled then

$$\kappa_-(d_i c_{i+1}, e_i^2) \;<\; \frac{1 + K_\square}{1 - K_\square} \;=:\; \kappa_\Delta \tag{2.12}$$

is a worst-case bound for the condition number of the final subtraction in the computation of $\Delta_i = d_i c_{i+1} - e_i^2$. Even the lax choice $K_\square = 1/4$ results in a benign bound of $\kappa_\Delta = 5/3$. As about every use of a block factorization will have to refer to the block determinants at some point, being able to compute them stably is crucial. The meanings of $K_\square$ and $\kappa_\Delta$ will remain in effect throughout this paper.

In practice, the test (2.10) will be executed in a floating-point context, and therefore the quantities $d_i c_{i+1}$ and $e_i^2$ will not be available exactly, but only as floating-point approximations $\mathrm{fl}(d_i c_{i+1})$ and $\mathrm{fl}(e_i^2)$. More generally, we may desire that the condition still holds when the involved quantities $d_i$, $c_{i+1}$ and $e_i^2$ are changed by a limited perturbation, and it should do so for the same constant $K_\square$, so that the theoretical error bounds we are going to derive remain valid. There is a simple recipe to achieve this, namely that for an evaluation of condition (2.10) in practice we plug in a value for $K_\square$ that is slightly smaller ($0.999K_\square$, say) than the one used to derive the error bounds. Details on this matter can be found in [16]; from now on we will just assume that it is done properly. Then, once we have established (2.10) we may conclude that this relation, as well as (2.12), also hold for the computed quantities:

$$|\mathrm{fl}(d_i c_{i+1})| \;<\; K_\square \mathrm{fl}(e_i^2) \qquad \text{and} \qquad \kappa_-(\mathrm{fl}(d_i c_{i+1}), \mathrm{fl}(e_i^2)) \;<\; \kappa_\Delta. \tag{2.13}$$

ETNA
Kent State University
http://etna.math.kent.edu



**3. Stationary block factorization.** The purpose of this section is to develop an analogue to `dstqds` for block factorizations, i.e., to compute

$$\mathsf{T} - \tau \;=\; \mathsf{T}^+, \quad \text{where} \quad \mathsf{T} = \mathsf{LDL}^*,\ \mathsf{T}^+ = \mathsf{L}^+\mathsf{D}^+(\mathsf{L}^+)^* \;\in\; \mathbb{R}^{n\times n} \tag{3.1}$$

and $\mathsf{D}$, $\mathsf{D}^+$ are block-diagonal (with bandwidth one) each.

We call $\mathsf{LDL}^*$ the *source* and $\mathsf{L}^+\mathsf{D}^+(\mathsf{L}^+)^*$ the *target* of the process. Our treatment is based on the standard representation, that is, we assume to get the data $\big(\Omega, \{d_i\}, \{c_j\}, \{e_k\}\big)$ as input for $\mathsf{LDL}^*$, and the outputs $\big(\Omega^+, \{d_i^+\}, \{c_j^+\}, \{e_k\}\big)$ will define $\mathsf{L}^+\mathsf{D}^+(\mathsf{L}^+)^*$.

The goal is an algorithm that allows a componentwise mixed relative error analysis. Hence, we need to find suitable perturbations of the input and output data items, of the form

$$\begin{aligned}
&d_i \rightsquigarrow \widetilde{d}_i, \quad i \notin \Omega, && c_i \rightsquigarrow \widetilde{c}_i, \quad i \in \Omega,\\
&e_i \rightsquigarrow \widetilde{e}_i, \quad i = 1,\dots,n-1, &&\\
&d_i^+ \rightsquigarrow \widetilde{d}_i^+, \quad i \notin \Omega^+, && c_i^+ \rightsquigarrow \widetilde{c}_i^+, \quad i \in \Omega^+,
\end{aligned}$$

such that the thus perturbed matrices satisfy $\widetilde{\mathsf{L}}\widetilde{\mathsf{D}}\widetilde{\mathsf{L}}^* - \tau = \widetilde{\mathsf{L}}^+\widetilde{\mathsf{D}}^+(\widetilde{\mathsf{L}}^+)^*$ exactly. Note that the perturbations to $d_i$, $c_i$, $d_i^+$, $c_i^+$ can be compactly stated as

$$\mathsf{D}(i,i) \rightsquigarrow \widetilde{\mathsf{D}}(i,i), \quad \mathsf{D}^+(i,i) \rightsquigarrow \widetilde{\mathsf{D}}^+(i,i), \quad i = 1,\dots,n.$$

**Auxiliaries.** Just as in standard `dstqds` [18, Section 5] we introduce auxiliary *adjustment* quantities

$$s_i \;:=\; \mathsf{D}^+(i,i) - \mathsf{D}(i,i) + \tau, \qquad i = 1,\dots,n. \tag{3.2}$$

However, for block factorizations these do not allow for a recursive formulation of the factorization process like in [18, Remark 5.1] except if the block structures $\Omega$ and $\Omega^+$ are identical. Furthermore, the way to compute $s_{i+1}$ is no longer unique, but depends on the local structure at hand, meaning the four true values of $i \in \Omega$, $i \in \Omega^+$, $i+1 \in \Omega$, $i+1 \in \Omega^+$. With (2.8) we have

$$s_{i+1} \;=\; e_i^2\big(\mathrm{inv}_{\mathsf{D}}(i) - \mathrm{inv}_{\mathsf{D}^+}(i)\big), \qquad i = 1,\dots,n-1.$$

The definition (2.7) of $\mathrm{inv}_{\mathsf{D}}(i)$ and $\mathrm{inv}_{\mathsf{D}^+}(i)$ yields nine possible cases to be considered (not sixteen, because neither $\Omega$ nor $\Omega^+$ contain two consecutive indices); they are compiled in Table 3.1.

**Pictogram notation.** The error analysis to come is based on considering these cases separately, but we will have to jump between them occasionally. It is error-prone to differentiate the cases based solely on the mathematical definition. The pictograms introduced in the table will help us identify the cases by their structural characteristics, which are just the sets $\Omega$ and $\Omega^+$ and the pattern they induce for the distribution of the diagonal entries of $\mathsf{D}$ and $\mathsf{D}^+$ into $d_i$s and $c_j$s.

DEFINITION 3.1 (Pictograms). *The top line in the pictogram represents the structure in $\mathsf{D}$ and the bottom line the structure in $\mathsf{D}^+$. We use ● to represent a $d_i$ (i.e., the start of a block, or a single pivot) and ○ for a $c_j$, with a connecting line to further mark a block.*

For an example, consider case S6, defined by $i \in \Omega$, $i \notin \Omega^+$, $i+1 \notin \Omega^+$. Note that the pictogram for case S6 has nothing in its lower left corner. The reason is that this case does not specify if $i-1$ belongs to $\Omega^+$ or not, because it has no impact on the definition of $s_{i+1}$.

**ETNA**
Kent State University
http://etna.math.kent.edu



TABLE 3.1
*Standard formulae and alternative formulae for the next adjustment $s_{i+1}$. The alternative formulae use previous auxiliaries in the form of already computed quantities $s_j - \tau$, $j \le i$. See Definition 3.1 for the pictograms in the third column.*

| Case | Description | $\overset{i}{\downarrow}$ | $s_{i+1}$ | alternative formulae for $s_{i+1}$ |
|---|---|---|---|---|
| S1 | $i, i+1 \notin \Omega$<br>$i, i+1 \notin \Omega^+$ | ● ●<br>● ● | $\dfrac{e_i^2}{d_i} - \dfrac{e_i^2}{d_i^+}$ | $\dfrac{e_i^2(s_i - \tau)}{d_i d_i^+}$ |
| S2 | $i+1 \in \Omega$<br>$i+1 \in \Omega^+$ | ●—○<br>●—○ | $0$ | |
| S3 | $i \in \Omega$<br>$i \in \Omega^+$ | ●—○ ●<br>●—○ ● | $\dfrac{e_i^2 d_{i-1}}{\Delta_{i-1}} - \dfrac{e_i^2 d_{i-1}^+}{\Delta_{i-1}^+}$ | $\dfrac{e_i^2\left[e_{i-1}^2(s_{i-1} - \tau) - d_{i-1}d_{i-1}^+\tau\right]}{\Delta_{i-1}\Delta_{i-1}^+}$ |
| S4 | $i+1 \in \Omega$<br>$i, i+1 \notin \Omega^+$ | ●—○<br>● ● | $-\dfrac{e_i^2}{d_i^+}$ | |
| S5 | $i, i+1 \notin \Omega$<br>$i+1 \in \Omega^+$ | ● ●<br>●—○ | $\dfrac{e_i^2}{d_i}$ | |
| S6 | $i \in \Omega$<br>$i, i+1 \notin \Omega^+$ | ●—○ ●<br>● ● | $\dfrac{e_i^2 d_{i-1}}{\Delta_{i-1}} - \dfrac{e_i^2}{d_i^+}$ | $\dfrac{e_i^2\left[d_{i-1}(s_i - \tau) + e_{i-1}^2\right]}{\Delta_{i-1}d_i^+}$ |
| S7 | $i, i+1 \notin \Omega$<br>$i \in \Omega^+$ | ● ●<br>●—○ ● | $\dfrac{e_i^2}{d_i} - \dfrac{e_i^2 d_{i-1}^+}{\Delta_{i-1}^+}$ | $\dfrac{e_i^2\left[d_{i-1}^+(s_i - \tau) - e_{i-1}^2\right]}{d_i\Delta_{i-1}^+}$ |
| S8 | $i \in \Omega$<br>$i+1 \in \Omega^+$ | ●—○ ●<br>●—○ | $\dfrac{e_i^2 d_{i-1}}{\Delta_{i-1}}$ | |
| S9 | $i+1 \in \Omega$<br>$i \in \Omega^+$ | ●—○<br>●—○ ● | $-\dfrac{e_i^2 d_{i-1}^+}{\Delta_{i-1}^+}$ | |

Also keep in mind that a ● at the right end might stand for the start of a block, but this, too, has no effect on how $s_{i+1}$ is defined.

The cases S1, S3, S6 and S7 are special because they do not allow one to compute $s_{i+1}$ using just multiplications and divisions. It is possible to rewrite their definitions such that previously computed auxiliaries can be utilized for computing $s_{i+1}$; the resulting alternative definitions are also collected in Table 3.1. Let us illustrate their derivation for the case S3:

$$
\begin{aligned}
s_{i+1} &= e_i^2\left(\frac{d_{i-1}}{\Delta_{i-1}} - \frac{d_{i-1}^+}{\Delta_{i-1}^+}\right) && \text{from Table 3.1,}\\
&= e_i^2\,\frac{d_{i-1}(d_{i-1}^+c_i^+ - e_{i-1}^2) - d_{i-1}^+(d_{i-1}c_i - e_{i-1}^2)}{\Delta_{i-1}\Delta_{i-1}^+} && \text{by (2.6),}\\
&= e_i^2\,\frac{e_{i-1}^2(d_{i-1}^+ - d_{i-1}) - d_{i-1}d_{i-1}^+\tau}{\Delta_{i-1}\Delta_{i-1}^+} && \text{as } c_i^+ = c_i - \tau,\\
&= e_i^2\,\frac{e_{i-1}^2(s_{i-1} - \tau) - d_{i-1}d_{i-1}^+\tau}{\Delta_{i-1}\Delta_{i-1}^+} && \text{by (3.2).}
\end{aligned}
$$

Note that standard `dstqds` is completely subsumed in case S1 and the respective (standard and alternative) formulae are identical.

**Allowing perturbations in the auxiliaries and the shifts.** The task to compute a block factorization will prove to be much harder than in standard `dstqds`. To make the problem manageable, we allow for two further perturbations:

ETNA
Kent State University
http://etna.math.kent.edu



$s_1 := 0$
**for** $i = 1$ **to** $n-1$ **do**
  $\mathsf{D}^+(i,i) := \mathsf{D}(i,i) + (s_i - \tau)$
  *// Compute block determinants*
  **if** $i \in \Omega$ **then**
    $\Delta_{i-1} := d_{i-1}c_i - e_{i-1}^2$
  **if** $i \in \Omega^+$ **then**
    $\Delta_{i-1}^+ := d_{i-1}^+ c_i^+ - e_{i-1}^2$
  *// Compute next auxiliary* $s_{i+1}$
  $s_{i+1} := \ldots$
**endfor**
$\mathsf{D}^+(n,n) := \mathsf{D}(n,n) + (s_n - \tau)$

ALGORITHM 3.1: *Template for the stationary block factorization.*

(i) The computed auxiliaries $s_i$ need not to fulfill the respective relation from Table 3.1 exactly for the perturbed data. Instead, we will be content if a small relative perturbation of $s_i$ has this property. This relaxation makes it meaningful to see the auxiliaries as secondary data and denote by $\widetilde{s}_i$ the correct value for the perturbed data; this is achieved by replacing everything in any definition by its perturbed counterpart, e.g.,

$$\widetilde{s}_{i+1} \;=\; \widetilde{e}_i^2\big(\widetilde{d}_{i-1}/\widetilde{\Delta}_{i-1} - 1/\widetilde{d}_i^+\big) \;=\; \widetilde{e}_i^2\big(\widetilde{d}_{i-1}/(\widetilde{d}_{i-1}\widetilde{c}_i - \widetilde{e}_{i-1}^2) - 1/\widetilde{d}_i^+\big)$$

for case S6.

(ii) The effective shift for any index $i$ may be perturbed, that is, instead for (3.1) above we actually strive for

$$\widetilde{\mathsf{L}}\widetilde{\mathsf{D}}\widetilde{\mathsf{L}}^* - \operatorname{diag}(\widetilde{\tau}_i) \;=\; \widetilde{\mathsf{L}}^+\widetilde{\mathsf{D}}^+(\widetilde{\mathsf{L}}^+)^*,$$

allowing different shifts for the diagonal entries. This is mainly a notational convenience because perturbed shifts might also be written as an outer (multiplicative) perturbation,

$$\widetilde{\mathsf{L}}\widetilde{\mathsf{D}}\widetilde{\mathsf{L}}^* - \operatorname{diag}(\widetilde{\tau}_i) \;=\; \widetilde{\mathsf{L}}^+\widetilde{\mathsf{D}}^+(\widetilde{\mathsf{L}}^+)^* \quad\Leftrightarrow\quad \mathsf{Y}\widetilde{\mathsf{L}}\widetilde{\mathsf{D}}\widetilde{\mathsf{L}}^*\mathsf{Y} - \tau \;=\; \mathsf{Y}\widetilde{\mathsf{L}}^+\widetilde{\mathsf{D}}^+(\widetilde{\mathsf{L}}^+)^*\mathsf{Y},$$

where $\mathsf{Y} = \operatorname{diag}((\widetilde{\tau}_1/\tau)^{-1/2}, \ldots, (\widetilde{\tau}_n/\tau)^{-1/2})$. Multiplicative perturbations are known to be perfectly conditioned with respect to their effect on eigenvalues and invariant subspaces [1, 6].

With these two simplifications, the relations to be fulfilled by the perturbation are just

$$\widetilde{\mathsf{D}}^+(i,i) \overset{!}{=} \widetilde{\mathsf{D}}(i,i) + \widetilde{s}_i - \widetilde{\tau}_i, \quad i = 1, \ldots, n, \tag{3.3}$$

as our standard representation allows to reuse the offdiagonals $e_i$ for the target. (Here and in the following we use the symbol $\overset{!}{=}$ for “should be equal to”.)

**Determining the new block structure.** Based on the auxiliaries $s_i$ and assuming the block structure $\Omega^+$ is known, the computation of $\mathsf{L}^+\mathsf{D}^+(\mathsf{L}^+)^*$ can proceed in a manner similar to standard `dstqds`, using Algorithm 3.1 as an algorithmic template.

One application where the block structure $\Omega^+$ would be known beforehand is when a non-blocked factorization $\mathsf{L}^+\mathsf{D}^+(\mathsf{L}^+)^*$ is desired, i.e., with $\mathsf{D}^+$ being diagonal or, equivalently, $\Omega^+ = \emptyset$. In Section 3.4 we will present a customized algorithm just for this purpose.

In general, however, we want to determine a suitable block structure on the fly, for example with the intent to minimize (local) element growth. Then Algorithm 3.1 needs to be augmented with suitable tests to set up $\Omega^+$. The natural position for such a test is right after a $d_i^+$ has been computed in line 3 to decide if this should start a new $2\times 2$ pivot in the target factorization, that is, if $i+1$ should be added to $\Omega^+$. The concrete shape and form of the test has to depend on the block structure $\Omega$ in the source. We will develop and discuss a couple of usable situation-specific tests on the following pages. But regardless of how they are implemented, we should keep in mind that a $2\times 2$ pivot may only be chosen if (2.10) is fulfilled for $\mathsf{T}^+$, that is,

$$|d_i^+ c_{i+1}^+| \;<\; K_\square e_i^2. \tag{3.4}$$

Except for the choice of $\Omega^+$, the prominent point left open in Algorithm 3.1 is the computation of $s_{i+1}$ in line 10. We have already indicated that this has to depend on the actual case at hand as determined by $\Omega$ and $\Omega^+$ and that there are alternative formulae for some cases. Doing it wrong can make componentwise mixed relative stability impossible to achieve.

We will tackle the nine cases for computing $s_{i+1}$ from Table 3.1 by partitioning them into groups which can then be considered one at a time:

(i) Case S1 has already been dealt with in [18] for standard `dstqds` based on an $e$–representation. Accordingly, the alternative formulae from Table 3.1 can be used, and the error analysis for $e$–representations from [18, Theorem 5.3] does apply, cf. Figure 1.1.

(ii) Cases S2 and S3 state that a block in $\mathsf{D}$ corresponds to one in $\mathsf{D}^+$. This will be our first challenge in Section 3.2.

(iii) In Section 3.3 we will deal extensively with cases S4 and S6. Those constitute what we call *breaking a block*: single pivots in $\mathsf{D}^+$ where $\mathsf{D}$ has a block.

(iv) The largest chunk will be to tackle S5 and S7–S9 in Section 3.5. They have in common that a block in $\mathsf{D}^+$ is (or has just been) introduced where $\mathsf{D}$ does not have one—we call this *creating a block*. A special role will fall to S8 and S9, where blocks in $\mathsf{D}$ and $\mathsf{D}^+$ do *overlap*, because once these two cases start to alternate and form an *overlap sequence*, the worst-case relative perturbation bounds will depend on the length of the sequence. We are not yet able to overcome this problem completely, but it can be controlled in a practicable way.

We will present, for each of (ii)–(iv), a computational sequence tailored just for the computation of the $s_i$ concerned. These are intended to be used as plugin for the template in Algorithm 3.1 and will be accompanied by a complete relative error analysis covering all data involved. The final algorithm and its error analysis in Section 3.6 can then be built by composition. Due to the amount of technical details involved and for the sake of a more fluent presentation, we moved the full proofs of the error analysis into the appendix.

**3.1. Preparations for the error analysis.** We assume the standard model for floating-point arithmetic, that is, for a floating-point operation which is well-defined (no underflow or overflow), the exact result $z$ and the number $x$ computed on the machine can be related as

$$x = z(1+\gamma) = z/(1+\delta), \quad |\gamma|, |\delta| \le \epsilon_\diamond, \tag{3.5}$$

with *machine epsilon* $\epsilon_\diamond$. Most modern architectures adhere to the IEEE 754 standard for floating-point arithmetic [13, 14]. For IEEE `double precision` with 53-bit significands and eleven-bit exponents the above model is fulfilled with $\epsilon_\diamond = 2^{-53} \approx 1.1\cdot 10^{-16}$. For more information on binary floating-point arithmetic and the IEEE standard see [9, 12, 15].

We assume the reader to be familiar with the concept of mixed relative perturbation analysis and only recall the customized notation for sharp first-order analysis introduced in [18] and the rules for propagating perturbations.

DEFINITION 3.2 ([18, Definition 3.2]). *For any $p, n \in \mathbb{R}^{\ge 0}$ and $0 \le pn\epsilon_\diamond < 1$, we define*

$$\epsilon^{[p]}(n) \;:=\; \textit{placeholder for a quantity } \alpha \textit{ with } |\alpha| \le \frac{n\epsilon_\diamond}{1-pn\epsilon_\diamond} = n\epsilon_\diamond + p(n\epsilon_\diamond)^2 + \mathcal{O}(\epsilon_\diamond^3).$$

*We abbreviate* $\epsilon^{[1]}(n) =: \epsilon(n)$ *and write* $\epsilon_\diamond$ *instead of* $\epsilon^{[0]}(1)$ *if in placeholder context (on the right-hand side of an "$\doteq$").*

By "placeholder" we mean that occurrences of $\epsilon^{[\cdot]}(\cdot)$-terms should be interpreted similarly to the traditional $\mathcal{O}(\cdot)$ notation. To pronounce relations that deal with unspecified quantities like these we will write "$\doteq$" instead of "$=$". The expressions become unambiguous if interpreted from left to right.

THEOREM 3.3 (Rules for running error analysis [18, Theorem 3.3 and Corollary 3.4]). *Let* $p, n, q, m \in \mathbb{R}^{\geq 0}$ *and* $R = \max\{1, p, q\}$. *Then*

$$\begin{aligned}\big(1+\epsilon^{[p]}(n)\big)\big(1+\epsilon^{[q]}(m)\big) &\doteq 1+\epsilon^{[R]}(n+m),\\ \big(1+\epsilon^{[p]}(n)\big)^{-1} &\doteq 1+\epsilon^{[p+1]}(n),\\ \big(1+\epsilon^{[p]}(n)\big)^{1/2} &\doteq 1+\epsilon^{[2p+2]}(\tfrac{1}{2}n),\end{aligned}$$

*provided that* $0 \leq sk\epsilon_\diamond < 1$ *for each quantity* $\epsilon^{[s]}(k)$.

*Let* $m_i \in \mathbb{R}$ *and* $s_i \in \{-1, 1\}$, $i = 1, \ldots, n$, *be given. Then*

$$\prod_{i=1}^{n}(1+m_i\epsilon_\diamond)^{s_i} \doteq 1+\epsilon(D), \quad \textit{where } D := \sum_{i=1}^{n}|m_i|, \tag{3.6}$$

*provided* $D\epsilon_\diamond < 1$.

For any quantity $a$ we will normally use $\widetilde{a}$ for its perturbed counterpart, and we reserve the letter $\varrho$ for the associated relative perturbation factors:

$$\widetilde{a} = a \cdot \varrho(a).$$

*We do not use a special notation to distinguish floating-point numbers from "exact" ones.* In the remainder of this paper, unadorned symbols $d_i$, $e_i$, $c_i^+$, $\Delta_i, \ldots$ in an algorithm or its accompanying error analysis always refer to numbers as they are stored in the machine.

Recall our use of the term secondary data for anything (meaningful) which can be derived from a representation's primary data; so far we have already introduced the block determinants $\Delta_i, i+1 \in \Omega$, and the auxiliaries $s_i$ as such. Secondary data also have a natural counterpart under the influence of a perturbation, namely the value one obtains if every primary data occurrence in a definition is replaced by the perturbed version. We will extend the $\widetilde{\ }$-notation and $\varrho$ to refer to perturbed secondary data as well. Hence, the determinants for the $2\times 2$ blocks in $\widetilde{\mathsf{D}}$ are

$$\widetilde{\Delta}_i = \widetilde{d}_i\widetilde{c}_{i+1} - \widetilde{e}_i^2 = \Delta_i\varrho(\Delta_i), \quad i+1 \in \Omega,$$

etc. Note that, although our lax use of the $\widetilde{\ }$-notation might suggest otherwise, there still remains the subtle point that we can choose primary perturbations like $d_i \rightsquigarrow \widetilde{d}_i$ freely, whereas $\Delta_i \rightsquigarrow \widetilde{\Delta}_i$ is an immediate consequence once all perturbations to the primary data are fixed.

Concerning the offdiagonal elements $e_i$, for a shifted factorization based on our standard representation only their squares $e_i^2$ will ever be needed, so assume we have them as

$$\mathrm{fl}(e_i^2) = e_i^2(1+\varepsilon_i), \quad |\varepsilon_i| \leq \epsilon_\diamond, \qquad i = 1, \ldots, n-1. \tag{3.7}$$

**Induced perturbations for the block determinants.** It will be necessary to relate the block determinants $\Delta_i$ and $\Delta_i^+$ as computed in lines 6 and 8 of Algorithm 3.1 to the exact ones $\widetilde{\Delta}_i$ and $\widetilde{\Delta}_i^+$ for the perturbed matrices. Based on the floating-point model (3.5) and on (3.7), we can state

$$\begin{aligned}\Delta_i(1+\beta_\Delta) &= d_ic_{i+1}(1+\alpha_\Delta) - e_i^2(1+\varepsilon_i), \quad \text{for } i+1 \in \Omega,\\ \Delta_i^+(1+\beta_\Delta^+) &= d_i^+c_{i+1}^+(1+\alpha_\Delta^+) - e_i^2(1+\varepsilon_i), \quad \text{for } i+1 \in \Omega^+,\end{aligned} \tag{3.8}$$

with suitable perturbations $|\alpha_\Delta|, |\alpha_\Delta^+|, |\beta_\Delta|, |\beta_\Delta^+| \le \epsilon_\diamond$. Those will of course depend on $i$, but it is not necessary to make this dependency explicit. We will ensure that for all $2\times 2$ pivots, the condition of computing its determinant is bounded by $\kappa_\Delta$ from (2.13). Then we obtain a connection between the computed value $\Delta_i$ and the exact value $\widetilde{\Delta}_i$ (corresponding to perturbed primary data) as follows. Take the identities $\widetilde{\Delta}_i = \Delta_i\varrho(\Delta_i)$, etc., to rewrite the exact value $\widetilde{\Delta}_i = \widetilde{d}_i\widetilde{c}_{i+1} - \widetilde{e}_i^2$ as

$$\begin{aligned}\widetilde{\Delta}_i &= d_i\varrho(d_i)\cdot c_{i+1}\varrho(c_{i+1}) \;-\; e_i^2\varrho(e_i^2)\\ &= \underbrace{d_ic_{i+1}(1+\alpha_\Delta)F}_{=:x}\cdot\underbrace{\frac{\varrho(d_i)\varrho(c_{i+1})}{(1+\alpha_\Delta)F}}_{=:1+\xi} \;-\; \underbrace{e_i^2(1+\varepsilon_i)F}_{=:y}\cdot\underbrace{\frac{\varrho(e_i^2)}{(1+\varepsilon_i)F}}_{=:1+\eta},\end{aligned}$$

where the latter equality holds for each $F\neq 0$ and, in particular, for $\alpha_\Delta$ and $\varepsilon_i$ from the computations $\mathrm{fl}(d_ic_{i+1}) = d_ic_{i+1}(1+\alpha_\Delta)$ and $\mathrm{fl}(e_i^2) = e_i^2(1+\varepsilon_i)$ in (3.8) and (3.7), respectively. Now insert the values $x$, $y$ and the perturbations $\xi$, $\eta$, as defined via the braces, into (2.11) to obtain

$$\begin{aligned}\widetilde{\Delta}_i &= (x-y)\cdot(1+\gamma) \quad \text{for some } |\gamma| \le \kappa_-(x,y)\cdot\max\{|\xi|,|\eta|\}\\ &= (1+\beta_\Delta)\Delta_iF\cdot(1+\gamma),\end{aligned}$$

due to (3.8). Taking into account the invariance of the condition with respect to scaling, (2.13) yields

$$\kappa_-(x,y) \;=\; \kappa_-(x/F, y/F) \;=\; \kappa_-(\mathrm{fl}(d_ic_{i+1}), \mathrm{fl}(e_i^2)) \;<\; \kappa_\Delta.$$

To summarize, for $i+1\in\Omega$ we obtain

$$\begin{aligned}\varrho(\Delta_i) \quad &= \quad (1+\beta_\Delta)F(1+\gamma), \quad \text{where}\\ |\gamma| \quad &\le \quad \kappa_\Delta\cdot\max\left\{\left|\frac{\varrho(d_i)\varrho(c_{i+1})}{(1+\alpha_\Delta)F} - 1\right|, \left|\frac{\varrho(e_i^2)}{(1+\varepsilon_i)F} - 1\right|\right\},\end{aligned} \tag{3.9}$$

and $F\neq 0$ may be chosen freely. For $i+1\in\Omega^+$ the same argument yields an analogous bound for $\varrho(\Delta_i^+)$ with the same $\kappa_\Delta$, $\varrho(e_i^2)$ and $\varepsilon_i$.

**Perturbations for the auxiliaries.** Except for few special cases, we will perturb the data influencing $s_i$, and maybe also the shift $\widetilde{\tau}_i$, just so that

$$s_i - \tau \;=\; \widetilde{s}_i - \widetilde{\tau}_i \tag{3.10}$$

holds. This means the exact difference of the quantities $s_i$ and $\tau$ (which are floating-point numbers stored in the machine) equals the exact difference of the perturbed data $\widetilde{s}_i$ and $\widetilde{\tau}_i$ (which in general will not be representable as floating-point numbers).

Provided the relation (3.10) holds, there is an obvious way to perturb the diagonal data of $\mathsf{D}$ and $\mathsf{D}^+$ such that (3.3) is achieved. Assume the computation in line 3 of Algorithm 3.1 obeys

$$\mathsf{D}^+(i,i)(1+\delta_i^+) \;=\; \mathsf{D}(i,i) + (s_i-\tau)\big/(1+\sigma_i), \qquad \big|\delta_i^+\big|, \big|\sigma_i\big| \le \epsilon_\diamond, \tag{3.11}$$

where $\sigma_i$ accounts for the subtraction and will retain this meaning throughout the paper, and $\delta_i^+$ accounts for the error in the addition. This notation, too, will hold from now on. Multiply by $1+\sigma_i$ and, noting (3.10), there is an obvious way to go,

$$\varrho(\mathsf{D}(i,i)) \;:=\; 1+\sigma_i, \qquad \varrho(\mathsf{D}^+(i,i)) \;:=\; (1+\delta_i^+)(1+\sigma_i). \tag{3.12}$$

**ETNA**
Kent State University
http://etna.math.kent.edu



These will serve as our default perturbation for $\mathsf{D}$ and $\mathsf{D}^+$.

To attain the relation $s_i - \tau = \widetilde{s}_i - \widetilde{\tau}_i$ in the first place, there are basically two ways. The obvious one is to choose $e_{i-1} \rightsquigarrow \widetilde{e}_{i-1}$ just so that the computed $s_i$ becomes exact, i.e., $s_i = \widetilde{s}_i$. Then there is even no need to touch the shift, as $\widetilde{\tau}_i := \tau$ will do the trick. An alternative is made possible if $s_i$ is not too large in magnitude compared to the shift, e.g.,

$$|s_i| \;\leq\; R|\tau|$$

for some parameter $R$. *Then we can achieve* $s_i - \tau = \widetilde{s}_i - \widetilde{\tau}_i$ *for every choice of* $e_{i-1} \rightsquigarrow \widetilde{e}_{i-1}$ by moving any "excess" from $\widetilde{s}_i$ to $\widetilde{\tau}_i$, in the form

$$\widetilde{\tau}_i - \tau \;=\; \widetilde{s}_i - s_i \qquad \Longrightarrow \qquad \big|\varrho(\tau_i) - 1\big| \;\leq\; R\big|\varrho(s_i) - 1\big|, \tag{3.13}$$

defining $\widetilde{\tau}_i$. This provides us with one additional degree of freedom in the choice of $\widetilde{e}_{i-1}$, which can be used to fix some other critical computation. Note that, effectively, we did cast a relative perturbation of one quantity ($\widetilde{s}_i$) as an absolute one and then wrote it again as a relative one, but for a different quantity ($\tau$). This technique will be a crucial ingredient for the error analysis.

This closes the general preparations. Note that, whenever (3.12) can be used, all that remains to be done is to specify fitting perturbations $e_i \rightsquigarrow \widetilde{e}_i$ for the offdiagonal data.

**3.2. Keep a block.** Block factorizations are mostly harmless as long as the block structure is not changed. With respect to Table 3.1 this comprises the cases S1, S2 and S3. For S1 a single pivot in the source corresponds to a single pivot in the target—this corresponds to standard `dstqds` for an $e$–representation, and the error bounds from Figure 1.1 do apply. In this section we will deal with the cases S2 and S3: a block in $\mathsf{D}$ is reproduced in $\mathsf{D}^+$, that is, we *keep* the block.

We begin by devising a criterion to determine when a block should be kept. Assume we have a block in the source covering indices $i$ and $i+1$, that is, $i+1 \in \Omega$. The option of keeping the block does only present itself if we did not already choose a block in the target at $i-1$, so assume $i \notin \Omega^+$. Considering again Table 3.1, the choice between keeping the block or not corresponds to the choice between the cases S2 or S4.

It is desirable to keep the structure, but this is not always possible. At least, we have to ensure that each block in $\mathsf{D}^+$ satisfies condition (3.4). The data $\mathsf{T}(i+1, i+1) = c_{i+1}$ are readily available in our standard representation in this situation, so just one extra addition gives $\mathsf{T}^+(i+i, i+1) = c_{i+1} - \tau$. This yields the following test.

> **Block Criterion I (BC–I).** ●—○
> ● ?
> ↑
> $i$
>
> Fix a parameter $K_1 \leq K_{\Box}$. Keep the block if
>
> $$|d_i^+(c_{i+1} - \tau)| \;<\; K_1 e_i^2.$$

Now focus on the case that a block $i+1 \in \Omega$ is indeed kept, that is, we assume the following situation (cf. Definition 3.1 for the pictogram):

In: $s_i = \widetilde{s}_i$ $\qquad$ $i$ ↓ ●—○ ● / ●—○ ● $\qquad$ Out: $s_{i+2} = \widetilde{s}_{i+2}$.

As depicted, we require $s_i$ as input and assume it to be exact with respect to the perturbed data; the produced output $s_{i+2}$ shall have the same property. The computation will go through case S2 in Table 3.1, followed by case S3 (for $s_{i+2}$ instead of $s_{i+1}$), but to ensure stability we

**ETNA**
Kent State University
http://etna.math.kent.edu



$$\boxed{\begin{array}{lrcl} 1. & s_{i+1} & := & 0 \\ 2. & x & := & e_i^2(s_i - \tau) - d_i d_i^+ \tau \\ 3. & s_{i+2} & := & e_{i+1}^2\, x \big/ \big(\Delta_i\, \Delta_i^+\big) \end{array}}$$

ALGORITHM 3.2: *Keep a block.*

need to take the alternative formula for the latter. This leads to the computational sequence in Algorithm 3.2, to be integrated with our algorithmic template in Algorithm 3.1.

Recall our goal: We need to find perturbations to the primary data involved, namely $d_i$, $d_i^+$, $c_{i+1}$, $c_{i+1}^+$, $e_i$ and $e_{i+1}$, and optionally also the shifts $\tau_i$, $\tau_{i+1}$, such that the relation (3.3) is fulfilled for $i$ and $i+1$, with respect to the exact adjustments $\widetilde{s}_i$, $\widetilde{s}_{i+1}$ for the perturbed data. Combined with the In/Out-specification above, this boils down to achieving

$$\widetilde{d}_i^+ \overset{!}{=} \widetilde{d}_i + (s_i - \widetilde{\tau}_i), \qquad \text{since } \widetilde{s}_i = s_i, \tag{3.14}$$

$$\widetilde{c}_{i+1}^+ \overset{!}{=} \widetilde{c}_{i+1} - \widetilde{\tau}_{i+1}, \qquad \text{as } s_{i+1} = 0, \tag{3.15}$$

$$\widetilde{s}_{i+2} \overset{!}{=} s_{i+2}. \tag{3.16}$$

The following lemma states that these relations can in fact be obtained with suitable perturbations, without touching the shift. The proof for the lemma is given in Appendix A.1.

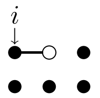
LEMMA 3.4. *If a block in the source is reproduced in the target, $i+1 \in \Omega \cap \Omega^+$, let in Algorithm 3.1 the auxiliaries $s_{i+1}$ and $s_{i+2}$ be computed as in Algorithm 3.2. Then we can find perturbations*

$$\begin{aligned} d_i &\rightsquigarrow \widetilde{d}_i &&\doteq \epsilon(1), & d_i^+ &\rightsquigarrow \widetilde{d}_i^+ &&\doteq \epsilon(2), \\ c_{i+1} &\rightsquigarrow \widetilde{c}_{i+1} &&\doteq \epsilon(1), & c_{i+1}^+ &\rightsquigarrow \widetilde{c}_{i+1}^+ &&\doteq \epsilon(2), \\ e_i &\rightsquigarrow \widetilde{e}_i &&\doteq \epsilon^{[4]}(3), & e_{i+1} &\rightsquigarrow \widetilde{e}_{i+1} &&\doteq \epsilon^{[4]}(\tfrac{7}{2} + \tfrac{7}{2}\kappa_\Delta), \end{aligned}$$

*such that*

$$\widetilde{d}_i^+ \;=\; \widetilde{d}_i + s_i - \tau, \quad \widetilde{c}_{i+1}^+ \;=\; \widetilde{c}_{i+1} - \tau, \quad \textit{and} \quad s_{i+2} \;=\; \widetilde{s}_{i+2}.$$

**3.3. Break a block.** In the previous section we dealt with the situation that a block in the source $\mathsf{D}$ is kept for the target $\mathsf{D}^+$. Now we will consider how to break a block without any overlap, that is, without creating a new block ending within $\{i, i+1, i+2\}$:

In: $s_i = \widetilde{s}_i$  Out: $s_{i+2} = \widetilde{s}_{i+2}$.

As depicted, we require $s_i$ as input and assume it to be exact with respect to the perturbed data, and we deliver $s_{i+2}$ enjoying the same property. We do not require that the decision to refrain from adding either one of $i$, $i+1$ or $i+2$ to $\Omega^+$ has been taken because BC–I is not fulfilled, but we assume

$$d_i^+ \neq 0 \quad \text{and} \quad d_{i+1}^+ \neq 0,$$

as otherwise the factorization would not be possible.

With respect to Table 3.1, the computation will go through S4 for $s_{i+1}$, followed by S6 for $s_{i+2}$. There are different ways to compute $s_{i+2}$. From case S6 we get the formula

$$s_{i+2} \;=\; \frac{e_{i+1}^2}{\Delta_i d_{i+1}^+}\big[\underbrace{d_i d_{i+1}^+ - \Delta_i}_{=:\;x}\big],$$

**ETNA**
Kent State University
http://etna.math.kent.edu



1. $s_{i+1} \ := \ -e_i^2/d_i^+$
2. **if** $|s_{i+1}| \le R_{\text{brk}}|\tau|$ **or** $\operatorname{sign}(d_i) \ne \operatorname{sign}(d_i^+)$ **then**
3. $\quad x \ := \ d_i(s_{i+1} - \tau) + e_i^2$ *// Formula I*
4. **else**
5. $\quad x \ := \ -s_{i+1}(s_i - \tau) - d_i\tau$ *// Formula II*
6. $s_{i+2} \ := \ e_{i+1}^2\, x \,/\, \big(\Delta_i d_{i+1}^+\big)$

ALGORITHM 3.3: *Break a block according to a given threshold* $R_{\text{brk}} > 1$.

revealing the intermediate quantity $x$ whose stable computation is a critical ingredient for breaking a block. This meaning of $x$ will remain in effect until the end of Section 3.4.

The alternative formula from Table 3.1 for case S6 shows that $x$ can also be written as

$$x \ = \ d_i(s_{i+1} - \tau) + e_i^2. \tag{3.17}$$

Two points should be noted. For having the slightest chance of finding a perturbation for $e_{i+1}$ such that $s_{i+2}$ becomes exact, we must compute $x$ stably, meaning that the computed $x$ should be only a small relative perturbation away from the exact $\widetilde{x}$ for perturbed data. Second, neither one of the two formulae for computing $x$ introduced above is always stable. This remains true even if one would assume that Block Criterion I was not fulfilled. These points make the task of breaking a block special, because they effectively force us to include a branch in the computation.

Computing $x$ as in (3.17) is advantageous, because it reuses the quantity $s_{i+1} - \tau$, which is required to compute $d_{i+1}^+$ anyway (line 3 in Algorithm 3.1). But for a relative error analysis, this approach is problematic because it uses $e_i$ explicitly again, although $s_{i+1}$ does already depend on $e_i$. An alternative formulation for $x$ is

$$x \ = \ \frac{e_i^2}{d_i^+}(s_i - \tau) - d_i\tau \ = \ -s_{i+1}(s_i - \tau) - d_i\tau.$$

This one is easily derived from (3.17), making use of the two identities $s_i - \tau = d_i^+ - d_i$ and $s_{i+1} = -e_i^2/d_i^+$. Again the outermost subtraction involved cannot bode well for all configurations, but at least the double dependency of $e_i$ is removed.

Both formulae to compute $x$ that were just introduced have their uses. They are integrated in Algorithm 3.3, together with a test for deciding which formula to take. Again we state only the parts relevant for computing the auxiliaries $s_{i+1}$ and $s_{i+2}$ and assume the rest is done according to Algorithm 3.1.

Although we also have investigated other ways to compute $s_{i+2}$, we do not see how to lead a mixed relative error analysis to completion if only one formula (I, II or another one) is used for all cases. However, we will show that the combination of Formulae I and II, together with the test in line 2 in Algorithm 3.3, leads to success. This result will be summarized in Lemma 3.5, and to arrive there we proceed as follows (see Appendix A.2 for the details):

(a) If $|s_{i+1}| \le R_{\text{brk}}|\tau|$ for some fixed $R_{\text{brk}} > 1$, then $s_{i+1}$ is not much larger than $\tau$ in magnitude, and then Formula I is fine because we can employ the technique surmised in (3.13) to modify the shift, opening up the freedom to perturb $e_i$ to control the subtraction involved in computing $x$ in line 3 of Algorithm 3.3. Lemma A.2 will deal with this case.

(b) If $|s_{i+1}| > R_{\text{brk}}|\tau|$ and $\operatorname{sign}(d_i) \ne \operatorname{sign}(d_i^+)$, then due to $R_{\text{brk}} > 1$ and $s_{i+1} = -e_i^2/d_i^+$, $\operatorname{sign}(s_{i+1} - \tau) = \operatorname{sign}(s_{i+1}) = -\operatorname{sign}(d_i^+)$ . Thus, $\operatorname{sign}(d_i) \ne \operatorname{sign}(d_i^+)$ implies that $d_i(s_{i+1} - \tau)$ and $e_i^2$ are nonnegative, and therefore the final addition in line 3 of Algorithm 3.3 is well-conditioned:

$$\kappa[x] \ := \ \kappa_+\big(\mathrm{fl}(d_i(s_{i+1} - \tau)), \mathrm{fl}(e_i^2)\big) \ = \ 1,$$

**ETNA**
**Kent State University**
**http://etna.math.kent.edu**



where $\kappa_+$ is defined according to (2.11). Lemma A.1 will show that Formula I is fine in this case because we are free to fix $\varrho(e_i)$ for any choice of $\varrho(d_i)$ such that $s_{i+1} = \widetilde{s}_{i+1}$ holds, while still being able to control the effect on $x$.

(c) Finally, Lemma A.3 will establish that Formula II is fine for the case $|s_{i+1}| > R_{\mathrm{brk}}|\tau|$.

These are just three out of a variety of conditions we studied while investigating how to break a block. Similarly to (b) one can show that Formula II is fine if the computation of $x$ in line 5 of Algorithm 3.3 is well-conditioned. However, just the three conditions stated above are included in Algorithm 3.3, so we will present only them in detail.

The following Lemma 3.5 summarizes the error analysis for breaking a block according to Algorithm 3.3 by taking the worst case bounds from Lemmas A.1–A.3. Indeed, as $|s_{i+1}| \gg |\tau|$ excludes the possibility of harmful cancellation between $s_{i+1}$ and $\tau$, just (a) and (c) alone would suffice to cover all input configurations, so one could even drop (b), and Lemma A.1 is not strictly required to break a block stably. We use it in a supporting role to alleviate some of the rather largish error bounds from Lemma A.3 in practice.

LEMMA 3.5. *For the case that a block in* $\mathsf{D}$ *is broken without overlap, i.e.,* $i+1 \in \Omega$ *and* $\Omega^+ \cap \{i-1, i, i+1\} = \emptyset$, *let in Algorithm 3.1 the auxiliaries* $s_{i+1}$ *and* $s_{i+2}$ *be computed as in Algorithm 3.3, with a parameter* $R_{\mathrm{brk}} > 1$. *Let* $R^* := (R_{\mathrm{brk}} - 1)^{-1}$. *Then there is a perturbation*

$$\begin{aligned}
d_i &\rightsquigarrow \widetilde{d}_i \doteq \epsilon(1), & d_i^+ &\rightsquigarrow \widetilde{d}_i^+ \doteq \epsilon(2),\\
c_{i+1} &\rightsquigarrow \widetilde{c}_{i+1} \doteq \epsilon(3+2R^*), & d_{i+1}^+ &\rightsquigarrow \widetilde{d}_{i+1}^+ \doteq \epsilon(4+2R^*),\\
e_i &\rightsquigarrow \widetilde{e}_i \doteq \epsilon^{[4]}(3), & e_{i+1} &\rightsquigarrow \widetilde{e}_{i+1} \doteq \epsilon^{[6]}(\tfrac{11}{2} + 2\kappa_\Delta + (\kappa_\Delta + 1)R^*),\\
\tau &\rightsquigarrow \widetilde{\tau}_{i+1} \doteq \epsilon(4R_{\mathrm{brk}}),
\end{aligned}$$

*such that*

$$\widetilde{d}_i^+ = \widetilde{d}_i + s_i - \tau, \quad \widetilde{d}_{i+1}^+ = \widetilde{c}_{i+1} + \widetilde{s}_{i+1} - \widetilde{\tau}_{i+1}, \quad \textit{and} \quad s_{i+2} = \widetilde{s}_{i+2}.$$

*The computed adjustment* $s_{i+1}$ *will satisfy* $\widetilde{s}_{i+1} \doteq s_{i+1}(1+\epsilon(4))$.

**3.4. Blocked to non-blocked factorization.** A crucial application of breaking blocks is when we desire a non-blocked target factorization $\mathsf{L}^+\mathsf{D}^+(\mathsf{L}^+)^*$ with $\mathsf{D}^+$ being diagonal, i.e., $\Omega^+ = \emptyset$. Computation of a general (blocked to blocked) factorization is expensive due to the conditionals involved. For an MR$^3$ algorithm based on block factorizations as representations for inner nodes, there is really no need to employ blocks during bisection or computing eigenvectors, as then the element growth in the target has no effect on accuracy. Algorithm 3.4 provides a non-blocked factorization, making use of the results from the previous pages.

THEOREM 3.6 (Error analysis for blocked to non-blocked `dstqds`). *Let Algorithm 3.4 be executed without underflow or overflow in an environment that satisfies the floating-point model (3.5), and let all blocks in* $\mathsf{D}$ *satisfy (2.12).*

*Then there are perturbations to the inputs and outputs such that*

$$\mathsf{LDL}^* - \mathrm{diag}(\widetilde{\tau}_i) = \mathsf{L}^+\mathsf{D}^+(\mathsf{L}^+)^*$$

*holds exactly. The perturbations can be bounded depending on the parameters* $R_{\mathrm{brk}}$ *and* $K_\square$ *according to Lemma 3.5. The specific choices* $R_{\mathrm{brk}} = 3$, $K_\square = 1/8$ *lead to the following bounds; only first-order bounds are shown, i.e., an entry* $p$ *stands for a bound* $p\epsilon_\diamond + \mathcal{O}(\epsilon_\diamond^2)$.

| $\mathsf{LDL}^*$ | |
|---|---|
| $d_i \rightsquigarrow \widetilde{d}_i$ | 1 |
| $c_i \rightsquigarrow \widetilde{c}_i$ | 4 |

| $\mathsf{L}^+\mathsf{D}^+(\mathsf{L}^+)^*$ | $i \notin \Omega$ | $i \in \Omega$ |
|---|---|---|
| $d_i^+ \rightsquigarrow \widetilde{d}_i^+$ | 2 | 5 |

| | $i \notin \Omega$ | $i \in \Omega$ |
|---|---|---|
| $e_i \rightsquigarrow \widetilde{e}_i$ | 3 | 10 |
| $\tau \rightsquigarrow \widetilde{\tau}_i$ | 0 | 12 |

ETNA
Kent State University
http://etna.math.kent.edu



Input: $\Omega$, shift $\tau$, $\{\mathsf{D}(i,i)\} = \{d_i \,|\, i \notin \Omega\} \cup \{c_i \,|\, i \in \Omega\}$, $\{e_1, \dots, e_{n-1}\}$
Output: $\{d_1^+, \dots, d_n^+\}$
Parameter: $R_{\mathrm{brk}} > 1$
Notes: The offdiagonal elements $e_i$ are reused to represent $\mathsf{T}^+$. Computing the block determinants $\Delta_i$, $i+1 \in \Omega$, is not shown; these should have been cached beforehand.

```
s_1 := 0
for i = 1 to n − 1 do
    d_i^+ := D(i,i) + (s_i − τ)
    if i + 1 ∈ Ω then                               // initiate breaking the block
        s_{i+1} := −e_i^2 / d_i^+                   // S4
    elseif i ∉ Ω then                               // standard dstqds
        s_{i+1} := e_i^2 (s_i − τ)/(d_i d_i^+)      // S1
    else                                            // finish breaking the block
        if |s_i| ≤ R_brk|τ| or sign(d_{i−1}) ≠ sign(d_{i−1}^+) then
            x := d_{i−1}(s_i − τ) + e_{i−1}^2       // Formula I
        else
            x := −s_i(s_{i−1} − τ) − d_{i−1}τ       // Formula II
        s_{i+1} := e_i^2 x / (Δ_{i−1} d_i^+)        // S6
endfor
d_n^+ := D(n,n) + (s_n − τ)
```

ALGORITHM 3.4: *Factorize blocked to non-blocked: Given a block factorization* $\mathsf{T} = \mathsf{LDL}^* \in \mathbb{R}^{n \times n}$ *in standard representation, compute data for non-blocked* $\mathsf{L}^+\mathsf{D}^+(\mathsf{L}^+)^* = \mathsf{T}^+$ *such that* $\mathsf{L}^+\mathsf{D}^+(\mathsf{L}^+)^* = \mathsf{LDL}^* - \tau$.

*Proof.* One just has to note that the bounds in Lemma 3.5 are already large enough to encompass the error bounds from Figure 1.1 for non-blocked `dstqds` ($e$–representation). The specific parameter settings $R_{\mathrm{brk}} = 3$, $K_\square = 1/8$ imply $R^* = 0.5$ and $\kappa_\Delta = 9/7$; then the indicated bounds are immediate. □

REMARK 3.7 (Breakdowns in Algorithm 3.4). Since no $2 \times 2$ pivots are allowed in the target, the factorization may break down if a $d_i^+$ becomes zero. This can be handled analogously to standard `dstqds`, cf. [18, Section 8] for details.

REMARK 3.8 (Optimizing Algorithm 3.4). We formulated Algorithm 3.4 with the intention to maximize clarity, but in this form it is quite inefficient, due to the many conditionals involved. An alternative design could use a nested loop structure:

```
while i < n do
    while i + 1 ∉ Ω do
        ... // do normal dstqds
    endwhile
    // break the block
    ...
    i := i + 2
endwhile
```

This reduces the number of extra conditionals to one per block in the source, which is minimal, since we cannot avoid having to select between Formulae I and II for stable computation.

**3.5. Creating blocks and handling overlap.** In this section we will analyze how to *create* new blocks at a position where the source does not have one. We will start with discussing criteria to decide when this is sensible.

**ETNA**
Kent State University
http://etna.math.kent.edu



**When to create a new block in $\mathsf{D}^+$.** Assume the factorization process did just compute $d_i^+$ at an index $i$ where no block starts in the source, that is, $i+1 \notin \Omega$. This leaves two cases to consider, since $\mathsf{D}$ might still have a block ending at index $i$ (cf. Definition 3.1 for the pictograms):

$$\begin{array}{cc} \overset{i}{\downarrow} & \\ \bullet & \bullet \\ \bullet & ? \end{array} \qquad \text{or} \qquad \begin{array}{ccc} & \overset{i}{\downarrow} & \\ \bullet\!\!- & \circ & \bullet \\ & \bullet & ? \end{array}$$

Here one could try to test for (2.10) directly by tentatively computing the diagonal element $\mathsf{T}^+(i+1,i+1) = c_{i+1}^+$. This would require to first determine $s_{i+1}$ according to either case S5 or S8 in Table 3.1, and then check if

$$\left| d_i^+ \left( d_{i+1} + (s_{i+1} - \tau) \right) \right| \;<\; K_\square e_i^2 \tag{3.18}$$

holds. This approach has the advantage of being to-the-point. A $2\times 2$ pivot is chosen aggressively in every situation where it makes sense according to the basic condition (2.10). However, there are two drawbacks. If the test should indicate *not* to choose a $2\times 2$ pivot, we end up in case S1 or S6 instead. Then the computed $s_{i+1}$ becomes invalid; its computation—including at least one division—as well as the two additions to get $c_{i+1}^+$, would have been wasted. The second drawback is more serious: we need additional assumptions about created blocks for the error analysis to succeed, and the direct test (3.18) does not provide them.

Due to these reasons we use the following stronger criterion for determining if a block should be introduced where the source does not have one.

**Block Criterion II (BC–II).**

$$\begin{array}{cc} & \bullet \\ \bullet & ? \\ \underset{i}{\uparrow} & \end{array}$$

Fix a parameter $K_2 \le K_\square/3$. Choose a $2\times 2$ pivot if

$$|d_i^+| \cdot \max\left\{|\tau|, |d_{i+1}|\right\} \;<\; K_2 e_i^2 \quad \text{and} \quad \begin{cases} |d_i^+| < K_2 |d_i| & \text{if } i \notin \Omega, \\ |d_i^+ d_{i-1}| < K_2 |\Delta_{i-1}| & \text{if } i \in \Omega. \end{cases}$$

To motivate this criterion, suppose a created block satisfies it in exact arithmetic. Then the expressions for $s_{i+1}$ in cases S5 or S8 from Table 3.1 reveal

$$c_{i+1}^+ \;=\; \begin{cases} d_{i+1} + e_i^2/d_i - \tau, & \text{if } i \notin \Omega, \\ d_{i+1} + e_i^2 d_{i-1}/\Delta_{i-1} - \tau, & \text{otherwise}, \end{cases} \qquad \Longrightarrow \qquad |d_i^+ c_{i+1}^+| < K_\square e_i^2.$$

Assuming the safeguard measures concerning $K_\square$ mentioned at the end of Section 2.4 have been heeded, the choice $K_2 \le K_\square/3$ ensures that the above property will hold for the perturbed quantities as well, so indeed the computed block data will even have the stronger properties (2.13) again.

Deploying BC–II instead of (3.18) has the advantage that

$$|\Delta_i^+| \;>\; (1 - K_\square) e_i^2 \;>\; \frac{3(1-K_\square)}{K_\square} |d_i^+ d_{i+1}| \tag{3.19}$$

will hold for any chosen block. For example, with $K_\square = .25$ this means the "hidden" pivot $\Delta_i^+/d_i^+$ would have been at least nine times larger in magnitude than $d_{i+1}$ (or infinite if $d_i^+ = 0$); so the choice to create a block was well-founded.

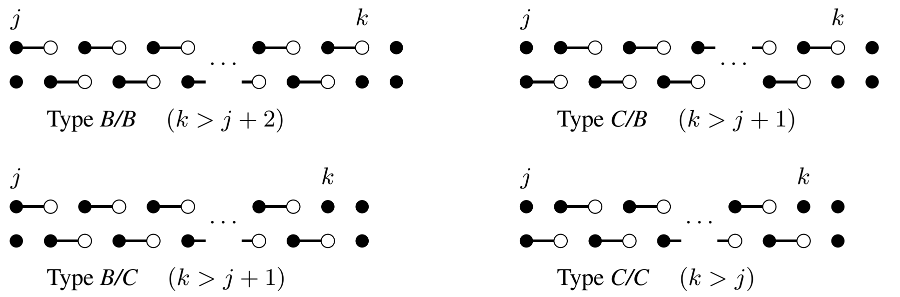
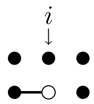


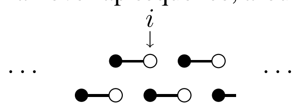

FIG. 3.1. *The four types of overlap sequences, classified as to whether they start/end by breaking a block (B) or creating a new one (C); cf. Definition 3.1 for the pictograms.*

**Overlap with blocks in $\mathsf{D}$.** A newly created block in the target can *overlap* with blocks in the source if $i-1 \in \Omega$ or $i+1 \in \Omega$. There is only one situation where this does not happen:




One could call this the "clean" or "pristine" creation of a new block. It is symmetrical to breaking a block. Indeed, this can be realized based on the less restrictive test (3.18) in a way completely analogous to breaking a block in Section 3.3, including a necessary branch in the computation. Our implementation does indeed include this special treatment, but we have chosen not to present its error analysis here, since it does not convey anything new compared to breaking a block. Furthermore, as it turns out, if the more restrictive Block Criterion II is employed, the computational branch is not necessary.

There is a fundamental problem involved with overlapping blocks. It arises when $\mathsf{D}$ and $\mathsf{D}^+$ each have a sequence of consecutive blocks that are out-of-sync, in the sense that $i \in \Omega \Leftrightarrow i \notin \Omega^+$. With respect to Table 3.1, this means alternating between S8 and S9. We call this phenomenon an *overlap sequence*. The essential problem with it is that we cannot formulate the factorization as a recursive process like one can for standard `dstqds`, cf. [18, Remark 5.1]. As a consequence the perturbation bounds to attain mixed relative stability will grow with the length of the sequence, at least in general. To explain why this is so, consider the following section of an overlap sequence, around $i \in \Omega$:




Any perturbation strategy has to fix $\widetilde{e}_i$ to control $\widetilde{s}_{i+1}$ such that (3.3) is fulfilled. As $i \in \Omega$, this makes $\varrho(e_i)$ depend on $\varrho(\Delta_{i-1})$. Now $e_i$ contributes to $\Delta_i^+$, so $\varrho(\Delta_i^+)$ will depend on $\varrho(e_i)$ and therefore, $\varrho(e_{i+1})$ will have to depend on $\varrho(e_i)$, too, forming a cycle.

Because of this interdependency we can deal with overlap sequences only by considering them *en bloc*, starting from the last index $j$ with $j \notin \Omega \cup \Omega^+$ and up to the next $k$ with $k+1 \notin \Omega \cup \Omega^+$. Then each sequence can start and end by either creating a block or breaking one. This leaves four basic types of overlap sequences; see Figure 3.1. The constraints shown on $j$ and $k$ stem from requiring that at least one new block from $\mathsf{D}^+$ be contained. Note that type *C/C* includes $k = j+1$, the creation of a new block without any overlap, which was introduced already. We kept this case as the error analysis to follow covers it seamlessly.

The computation of the adjustments $s_{j+1}, \ldots, s_{k+1}$ for any of the four types can proceed as summarized in Algorithm 3.5. It uses the standard formulae from Table 3.1 for the

```
1.  if j+1 ∈ Ω then                          // begin by breaking a block
2.        s_{j+1} := −e_j^2 / d_j^+
3.  else                                     // begin by creating a block
4.        s_{j+1} := e_j^2 / d_j
5.  for i = j+1 to k−1 do
6.        if i ∈ Ω then
7.              s_{i+1} := e_i^2 d_{i−1} / Δ_{i−1}
8.        else
9.              s_{i+1} := −e_i^2 d_{i−1}^+ / Δ_{i−1}^+
10. endfor
11. if k ∈ Ω then                            // end with breaking a block
12.       s_{k+1} := e_k^2 [(s_k − τ)d_{k−1} + e_{k−1}^2] / (Δ_{k−1} d_k^+)
13. else                                     // end with creating a block
14.       s_{k+1} := e_k^2 [(s_k − τ)d_{k−1}^+ − e_{k−1}^2] / (d_k Δ_{k−1}^+)
```

ALGORITHM 3.5: *Compute adjustments for an overlap sequence.*

auxiliaries up to $s_k$ and the alternative formulae from Table 3.1 for $s_{k+1}$.

A complete error analysis of this sequence is given in Appendix A.3. In the following we will just highlight the key points.

The computation as such is simple, but the mentioned interdependency means that for the offdiagonal elements the best bound we obtain is, up to first order,

$$|\varrho(e_i) - 1| \;\le\; \left(\frac{5}{2} + (2\kappa_\Delta + 1)\frac{\kappa_\Delta^{i-j+1} - 1}{\kappa_\Delta - 1}\right)\cdot\epsilon_\diamond \quad \text{for } j < i < k; \tag{3.20}$$

cf. (A.23) in Appendix A.3.2.

This dependence on the length of the overlap sequence is far too strong to make the result of much practical use. For example, with the moderate bound $\kappa_\Delta = 3/2$ (which we get with $K_\square = 1/5$) and a sequence of length $k - j = 6$, the perturbation to $e_{k-1}^2$ alone would already exceed $100\epsilon_\diamond$.

But there is light at the horizon in the form of choice. After all, the bound above can be monitored during a computation. Should it grow too large, we can simply choose not to take a $2\times 2$ pivot, thus capping the sequence. The only situation where we absolutely *must* take a $2\times 2$ pivot is if $d_i^+$ becomes zero. But this is actually a favorable situation, because it causes $s_{i+2}$ to be zero as well (cf. line 9 in Algorithm 3.5 for $i+1$). This effectively lifts any restriction on $\widetilde{e}_{i+1}$ and therefore cuts the dependency chain.

The full error analysis in Appendix A.3.2 does include this optimization not only for $d_i^+ = 0$ (thus $s_{i+2} = 0$), but more generally for any situation where an $|s_i|$ becomes "small". The analysis reveals that this effectively resets the recurrence, that is, the exponent to $\kappa_\Delta$ in (3.20). The resulting error bounds are compiled in Theorem 3.9 in Section 3.6.

**3.6. The complete algorithm.** Algorithm 3.6 summarizes the work on the previous pages for stationary block factorizations, with support for changing the block structure.

The basic design remains identical to the template Algorithm 3.1, except that we have omitted statements to compute the block determinants $\Delta_i$ and $\Delta_i^+$. Essentially it is the straightforward combination of standard `dstqds` for an $e$–representation, cf. Figure 1.1, and the computational sequences that we have considered for keeping (Algorithm 3.2), breaking (Algorithm 3.3) or creating (Algorithm 3.5) blocks. However, this fact is obscured somewhat due to the integration of a control mechanism for overlap sequences.

The outermost loop in Algorithm 3.6 is composed of two parts. The "first" half, lines 3–21, handles the factorization as long as the block structure is not changed. With respect to

**ETNA**
Kent State University
http://etna.math.kent.edu



Input: $\Omega$, shift $\tau$, $\{d_i \mid i \notin \Omega\}$, $\{c_i \mid i \in \Omega\}$, $\{e_1, \ldots, e_{n-1}\}$
Output: $\Omega^+$, $\{d_i^+ \mid i \notin \Omega^+\}$, $\{c_i^+ \mid i \in \Omega^+\}$
Parameters: $R_{\mathrm{brk}} > 1$, $R_{\mathrm{osq}} > 0$, $b_{\max} \geq 0$, $K_\square < 1/3$ as in (2.10),
$K_1 \leq K_\square$, $K_2 \leq K_\square/3$ for BC–I and BC–II

$\Omega^+ := \emptyset$, $i := 1$, $s_1 := 0$
**while** $i < n$ **do**
  $d_i^+ := d_i + (s_i - \tau)$
  **if** $i + 1 \in \Omega$ **then**
    **if** BC–I is fulfilled **then** *// keep the block*
      $\Omega^+ := \Omega^+ \cup \{i+1\}$
      $s_{i+1} := 0$ *// S2*
      **if** $i < n-1$ **then**
        $c_{i+1}^+ := c_{i+1} - \tau$
        $x := e_i^2(s_i - \tau) - d_i d_i^+ \tau$
        $s_{i+2} := e_{i+1}^2\, x \big/ (\Delta_i\, \Delta_i^+)$ *// S3*
        $i := i + 1$
    **else** *// initiate breaking the block*
      $s_{i+1} := -e_i^2 / d_i^+$ *// S4*
  **else**
    **if** BC–II is fulfilled **then** *// initiate creating a new block*
      $\Omega^+ := \Omega^+ \cup \{i+1\}$
      $s_{i+1} := e_i^2 / d_i$ *// S5*
    **else** *// standard `dstqds`*
      $s_{i+1} := e_i^2(s_i - \tau)/(d_i d_i^+)$ *// S1*
  $i := i + 1$
  $b := 0$ *// counts number of created blocks in an overlap sequence*
  **while** $i < n$ **and** $(i \in \Omega) \not\Leftrightarrow (i \notin \Omega^+)$ **do**
    $\mathsf{D}^+(i) := \mathsf{D}(i) + (s_i - \tau)$
    **if** $|s_i| \leq R_{\mathrm{osq}}|\tau|$ **then**
      $b := 0$ *// reset counter*
    **if** $i \in \Omega^+$ **then**
      **if** $i + 1 \in \Omega$ **then** *// continue sequence*
        $s_{i+1} := -e_i^2 d_{i-1}^+ / \Delta_{i-1}^+$ *// S9*
      **else** *// end by create*
        $s_{i+1} := e_i^2\big(d_{i-1}^+(s_i - \tau) - e_{i-1}^2\big)\big/\big(\Delta_{i-1}^+ d_i\big)$ *// S7*
    **else**
      **if** BC–II is fulfilled **and**
        $\Big(b < b_{\max}$ **or** $i = n-1$ **or** $|d_i^+| e_{i+1}^2 \leq (1 - K_\square) R_{\mathrm{osq}} |\tau| e_i^2\Big)$ **then**
        $\Omega^+ := \Omega^+ \cup \{i+1\}$ *// create next block in the sequence*
        $b := b + 1$
        $s_{i+1} := e_i^2 d_{i-1} / \Delta_{i-1}$ *// S8*
      **else** *// end by break or clean break*
        **if** $i - 1 \in \Omega^+$ **or** $|s_i| \leq R_{\mathrm{brk}}|\tau|$ **or** $\mathrm{sign}(d_{i-1}) \neq \mathrm{sign}(d_{i-1}^+)$ **then**
          $x := d_{i-1}(s_i - \tau) + e_{i-1}^2$
        **else**
          $x := -s_i(s_{i-1} - \tau) - d_{i-1}\tau$
        $s_{i+1} := e_i^2\, x \big/ (\Delta_{i-1} d_i^+)$ *// S6*
    $i := i + 1$
  **endwhile**
**endwhile**
$\mathsf{D}^+(n) := \mathsf{D}(n) + (s_n - \tau)$

ALGORITHM 3.6: *Blocked `dstqds`: Compute* $\mathsf{L}^+\mathsf{D}^+(\mathsf{L}^+)^* = \mathsf{LDL}^* - \tau$ *for block factorizations.*

**ETNA**
Kent State University
http://etna.math.kent.edu



Table 3.1 this encompasses cases S1–S5. There is nothing new here.

The "second" half, lines 22–45, is essentially one inner loop to handle any block structure changes from source to target. As such it concerns cases S6–S9 from Table 3.1. This loop will only be entered if either one of the statements in line 14 or 18 were executed to initiate a structure change, i.e., breaking or creating a block, respectively. The former one of those, together with lines 39–43 incorporates breaking a block from Algorithm 3.3 as well as end-by-break for an overlap sequence (line 12 from Algorithm 3.5), due to the extra test $i-1 \in \Omega^+$ in line 39. Note that Algorithm 3.4 is fully contained in Algorithm 3.6; the former can be revealed by fixing the tests for creating a new block in lines 5, 16 and 33 to evaluate to false.

The main new ingredient is the counter $b$, which is increased by one for each created block in an overlap sequence. It is reset to zero in line 26 whenever the current adjustment $s_i$ is not much larger than the shift, where the specifics are controlled by a parameter $R_{\mathrm{osq}}$. This integrates the optimized handling of overlap sequences that was mentioned at the end of the last section.

A new block may only be created if the complex test in lines 33–34 is passed. Besides checking for BC–II, creating a block in the target is only allowed if we can control the error bounds for the overlap sequence. To this end, one of three conditions has to be met:

(1) $b < b_{\max}$: The length of the sequence is still deemed acceptable.

(2) $i = n-1$: For a block that is created at the end it is easy to give very benign perturbation bounds to attain mixed relative stability, since no $s_{i+2}$ has to be computed.

(3) Because we require BC–II, the stronger property (3.19) will hold. Then fulfillment of the test $|d_i^+|e_{i+1}^2 \le (1-K_\square)R_{\mathrm{osq}}|\tau|e_i^2$ implies $|s_{i+2}| \le R_{\mathrm{osq}}|\tau|$ if the sequence contains $i+2$, i.e., we know that the test in line 25 will subsequently cause the counter to be reset.

Note that condition (3) permits the choice of a $2\times 2$ pivot whenever a tiny $d_i^+$ is encountered; in particular, the condition is always fulfilled if $d_i^+ = 0$. Hence, the factorization cannot break down, even for $b_{\max} = 0$.

The following result summarizes the componentwise mixed relative error analysis for Algorithm 3.6.

THEOREM 3.9 (Error analysis for blocked `dstqds`). *Let Algorithm 3.6 be executed without underflow or overflow in an environment that satisfies the floating-point model (3.5). Then there are perturbations to the inputs and outputs such that*

$$\mathsf{LDL}^* - \mathrm{diag}(\widetilde{\tau}_i) \;=\; \mathsf{L}^+\mathsf{D}^+(\mathsf{L}^+)^*$$

*holds exactly, with bounds to the individual perturbations given in the following table (cf. Definition 3.1 for the pictograms).*


| | | | | | |
|---|---|---|---|---|---|
| $\mathsf{D}(i)$ | 1 | 1 | $3+2R^*$ | 1 | 1 |
| $\mathsf{D}^+(i)$ | 2 | 2 | $4+2R^*$ | 2 | 2 |
| $\tau_i$ | 0 | 0 | $4R_{\mathrm{brk}}$ | $\max\{R_{\mathrm{osq}}(4+Q),\, 21+3Q\}$ | 0 |
| $e_i$ | 3 | $\frac{7}{2}(\kappa_\Delta+1)$ | $\frac{11}{2}+2\kappa_\Delta+(\kappa_\Delta+1)R^*$ | $\frac{15}{2}+\frac{3}{2}Q$ | $\frac{5}{2}+\frac{1}{2}Q$ |

*Only first-order bounds are shown, i.e., an entry $p$ stands for a bound $p\epsilon_\diamond + \mathcal{O}(\epsilon_\diamond^2)$, and*

$$R^* \;:=\; \frac{1}{R_{\mathrm{brk}}-1}, \qquad Q \;:=\; (2+4\kappa_\Delta)\frac{\kappa_\Delta^{2(B+1)}-1}{\kappa_\Delta-1},$$

ETNA
Kent State University
http://etna.math.kent.edu



*where* $0 \le B \le b_{\max}$ *is the maximal value attained by the counter* $b$ *during the computation.*

*Proof*. The bounds are obtained by combining the results for non-blocked `dstqds` from Figure 1.1 with Lemmata 3.4, 3.5, A.4 and A.5. □

Recall that our main objective was not to provide the sharpest possible error bounds, but to devise an algorithm for stationary block factorizations for which we can give componentwise relative error bounds in the first place.

However, there is no denying the fact that the bounds are quite large compared to standard `dstqds`. This is mainly due to the problems with overlap sequences, manifested by the parameter $Q$ (for $b_{\max} = 0$ and with $R_{\text{brk}} = 3$, $K_\square = 1/8$ as in Theorem 3.6 we could bound $|\varrho(e_i) - 1|$ by $10\epsilon_\diamond$). One could fear that such large componentwise errors would overshadow the benefits of using $2\times 2$ pivots regarding control of the local element growth. But keep in mind that the bounds are of a worst-case nature only. The experiments in the next section will show that the accuracy of the delivered results is far better than what these bounds would suggest.

Note also that the error bounds might be reduced significantly by taking smaller $K_\square$ values. As an example, $K_\square = 0.001$ leads to $\kappa_\Delta \approx 1.002$, and therefore $Q$ in Theorem 3.6 grows rather moderately for small $b_{\max}$, $Q \approx 12(B+1)$ for $B \le 10$. However, this choice also implies that many fewer $2\times 2$ pivots will be used (cf. BC–I and BC–II), thus increasing the admissible element growth, e.g., to $|\Delta_i^+/d_i^+| \gtrapprox 3000 d_{i+1}$ in (3.19). This may be harmful if blocked factorizations are used in the context of the MR$^3$ algorithm.

**4. Numerical results.** Block factorizations can be used as components for the tridiagonal MR$^3$ algorithm as well as in adapted solution strategies for BSVD, the singular value decomposition of an upper bidiagonal matrix. This section evaluates the potential gain from doing so.

In [16, Chapter 3] we have shown how, contrary to the established general consensus, BSVD can indeed be solved reliably by applying a just slightly modified MR$^3$ algorithm to the Golub–Kahan matrix (cf. Example 1.1). We have developed a prototype (meaning experimental and non-optimized) implementation of this algorithm, which can be configured to employ standard twisted factorizations (`XMR-TGK`) or top-down block factorizations (`BMR-TGK`). For testing we used a testset containing 19240 bidiagonal matrices $\mathsf{B}$. Those were produced by generating tridiagonal matrices of various standard types (including all types used in [2]) in dimensions up to 100, possibly self-gluing them, then scaling, splitting, shifting for positive definiteness and finally taking the Cholesky factors of the remaining unreduced subblocks. Details of the generation process can be found in [16, pp. 111 ff]. The test results are shown on the left-hand side of Table 4.1.

For TSEP, we have implemented our own custom MR$^3$ algorithm, which incorporates some new techniques that we have developed. These techniques and the implementation are presented in detail in [17]. Here we only want to assess the impact that block factorizations can have for TSEP. We do also know how to do a progressive transformation (`dqds`) with blocks, that is, the transition $\mathsf{LDL}^* - \tau =: \mathsf{U}^+\mathsf{R}^+(\mathsf{U}^+)^*$ from a lower to an upper (block) bidiagonal decomposition. For reasons of space this topic is not discussed in the present paper, but this feature has been integrated in the implementation.

This code, too, is configurable in what type of representation to use. We have compared three variations: using standard twisted factorizations (`XMR`), using only $\mathsf{LDL}^*$ block factorizations (`BMR-LDL`), or with the abovementioned twisted block factorizations (`BMR`, the default). We also used a synthetic testset here, but since this is a complete and production-level TSEP-solver no preprocessing steps need to be applied to the generated tridiagonal matrices, and the set is larger (base matrices were generated for all dimensions up to 200 before gluing). For a detailed description of the generation process see [17, Section 5]. The test results

ETNA
Kent State University
http://etna.math.kent.edu



TABLE 4.1

*Orthogonality and residual norms of the* $\mathsf{T}_{\mathrm{GK}}$*-based* BSVD *prototype solvers* `XMR-TGK` *(with twisted factorizations) and* `BMR-TGK` *(block factorizations), and the release* TSEP *solvers* `XMR` *(with twisted factorizations),* `BMR-LDL` *(*$\mathsf{LDL}^*$ *block factorizations) and* `BMR` *(twisted block factorizations). The lines below* MAX *give the percentages of test cases with maximum residual and loss of orthogonality, resp., in the indicated ranges.*

| **BSVD**-solvers (19240 testcases) | | | **TSEP**-solvers (116874 testcases) | | |
|---|---|---|---|---|---|
| `XMR-TGK` | `BMR-TGK` | | `XMR` | `BMR-LDL` | `BMR` |
| **Orthogonality levels** | | | | | |
| $\max\{\lvert\mathsf{U}^*\mathsf{U}-\mathsf{I}\rvert, \lvert\mathsf{V}^*\mathsf{V}-\mathsf{I}\rvert\}\ /\ n\epsilon_\diamond$ | | | $\max\{\lvert\mathsf{Q}^*\mathsf{Q}-\mathsf{I}\rvert\}\ /\ n\epsilon_\diamond$ | | |
| 5.34 | 4.41 | AVG | 234.19 | 68.63 | 1.44 |
| 1.38 | 1.69 | MED | 0.49 | 0.42 | 0.48 |
| 3095 | 788 | MAX | $9.68 \cdot 10^6$ | $7.86 \cdot 10^6$ | 9473 |
| 92.59 % | 91.04 % | $0 \ldots 10$ | 98.724 % | 98.872 % | 98.894 % |
| 7.04 % | 8.73 % | $10 \ldots 100$ | 1.119 % | 1.070 % | 1.070 % |
| 0.12 % | 0.12 % | $100 \ldots 200$ | 0.022 % | 0.022 % | 0.009 % |
| 0.11 % | 0.08 % | $200 \ldots 500$ | 0.027 % | 0.026 % | 0.013 % |
| 0.07 % | 0.03 % | $500 \ldots 10^3$ | 0.027 % | 0.005 % | 0.009 % |
| 0.06 % | | $> 10^3$ | 0.080 % | 0.005 % | 0.006 % |
| **Residual norms** | | | | | |
| $\max\{\lVert\mathsf{B}\bar{\mathsf{v}}_i-\bar{\mathsf{u}}_i\bar{\sigma}_i\rVert, \lVert\mathsf{B}^*\bar{\mathsf{u}}_i-\bar{\mathsf{v}}_i\bar{\sigma}_i\rVert\}\ /\ \lVert\mathsf{B}\rVert n\epsilon_\diamond$ | | | $\max\{\lVert\mathsf{T}\mathsf{q}_i-\mathsf{q}_i\lambda_i\rVert\}\ /\ \lVert\mathsf{T}\rVert n\epsilon_\diamond$ | | |
| 0.45 | 0.44 | AVG | 0.63 | 0.63 | 0.63 |
| 0.13 | 0.13 | MED | 0.13 | 0.13 | 0.13 |
| 118 | 4.67 | MAX | 13.48 | 13.48 | 13.48 |
| 84.96 % | 85.11 % | $0 \ldots 1$ | 80.807 % | 80.813 % | 80.807 % |
| 15.03 % | 14.89 % | $1 \ldots 10$ | 19.192 % | 19.186 % | 19.192 % |
| | | $10 \ldots 100$ | 0.001 % | 0.001 % | 0.001 % |
| 0.01 % | | $> 100$ | | | |

for the three configurations are shown on the right-hand side of Table 4.1.

For both problems BSVD and TSEP, the methods that use block factorizations do so in basically the same manner, by replacing them with the standard twisted factorizations as representations at the nodes in MR$^3$'s representation tree. For the execution of shifts, that is, to construct block factorizations for the representations at child nodes, Algorithm 3.6 is deployed with parameters $R_{\mathrm{brk}} = 5$, $R_{\mathrm{osq}} = n/4$, $b_{\max} = 4$, $K_\square = 1/8$, $K_1 = K_\square$, $K_2 = K_\square/3.01$. Sturm counts for bisection are done using the customized blocked to non-blocked factorization from Algorithm 3.4 with parameter $R_{\mathrm{brk}} = 8$. To construct the non-blocked factorizations that are needed for computing accurate eigenvectors, the same instantiation of Algorithm 3.4 was employed, together with a progressive analogue.

Due to the synthetic nature of the testsets containing many extreme cases that are unlikely in practice, we can say that all five solvers, with or without blocks, handle their task reasonably well. However, the results in Table 4.1 also clearly show that allowing $2\times2$ pivots can substantially improve the level of accuracy and stability of MR$^3$-based methods.

There are a couple of noteworthy observations to be made from the numbers. Residual norms are generally unproblematic, which is a typical feature of MR$^3$ (but not of all applications of MR$^3$ to solve BSVD, cf. [16] for counterexamples). That the residual statistics are almost identical for the three TSEP-solvers further shows that residual norms produced by MR$^3$ are far less sensitive to changes in the algorithm than orthogonality is. Concerning orthogonality, it is interesting to note that the introduction of top-down block factorizations in BMR−LDL does give a strong improvement, but the real jump comes from allowing twisted block factorizations. This shows how crucial the one additional degree of freedom in choosing the twist index can be to adapt to the problem at hand and avoid element growth where it arises.

What the results in Table 4.1 do not show is the possible performance impact of block factorizations. Due to the many conditionals involved, Algorithm 3.6 is far more expensive than a non-blocked factorization, so one could fear that we are just paying for reliability with speed. In [17] we have compared execution times of our release TSEP-solver BMR with the MR$^3$ and Divide-and-Conquer implementations DSTEGR and DSTEDC in the LAPACK library. Those results refute the fears concerning efficiency for block factorizations. The main reason for this is that the bulk of the work is actually to perform bisection and RQI steps, for which we only need the much faster Algorithm 3.4.

**Acknowledgements.** The authors want to thank Osni Marques and Christof Vömel for providing us the Pract matrices, Beresford Parlett for pointing us to block factorizations in the first place and for many fruitful discussions, and the referees for their valuable suggestions, which helped us to improve the presentation.

REFERENCES

[1] J. W. DEMMEL, *Applied Numerical Linear Algebra*, SIAM, Philadelphia, 1997.
[2] J. W. DEMMEL, O. A. MARQUES, B. N. PARLETT, AND C. VÖMEL, *A testing infrastructure for LAPACK's symmetric tridiagonal eigensolvers*, ACM Trans. Math. Software, 35 (2009), pp. 1–13. (Also available as LAPACK Working Note #182).
[3] I. S. DHILLON, *A new $O(n^2)$ algorithm for the symmetric tridiagonal eigenvalue/eigenvector problem*, Ph.D. Thesis, Computer Science Division, University of California, Berkeley, 1997.
[4] I. S. DHILLON AND B. N. PARLETT, *Multiple representations to compute orthogonal eigenvectors of symmetric tridiagonal matrices*, Linear Algebra Appl., 387 (2004), pp. 1–28.
[5] ———, *Orthogonal eigenvectors and relative gaps*, SIAM J. Matrix Anal. Appl., 25 (2004), pp. 858–899.
[6] S. C. EISENSTAT AND I. C. F. IPSEN, *Relative perturbation results for eigenvalues and eigenvectors of diagonalisable matrices*, BIT, 38 (1998), pp. 502–509.
[7] H.-R. FANG AND D. P. O'LEARY, *Stable factorizations of symmetric tridiagonal and triadic matrices*, SIAM J. Matrix Anal. Appl., 28 (2006), pp. 576–595.
[8] C. M. A. FERREIRA AND L. MIRANIAN, *(no title)*. Notes on qd-algorithms using blocks, 2005.
[9] D. GOLDBERG, *What every computer scientist should know about floating-point arithmetic*, ACM Comput. Surveys, 23 (1991), pp. 5–48.
[10] N. J. HIGHAM, *Stability of the diagonal pivoting method with partial pivoting*, SIAM J. Matrix Anal. Appl., 18 (1997), pp. 52–65.
[11] ———, *Stability of block $LDL^T$ factorization of a symmetric tridiagonal matrix*, Linear Algebra Appl., 287 (1999), pp. 181–189.
[12] ———, *Accuracy and Stability of Numerical Algorithms*, 2nd ed., SIAM, Philadelphia, 2002.
[13] IEEE, *IEEE Standard 754-1985 for binary floating-point arithmetic*, Aug. 1985.
[14] ———, *IEEE Standard 754-2008 for floating-point arithmetic*, Aug. 2008.
[15] W. KAHAN, *Lecture notes on the status of IEEE standard 754 for binary floating point arithmetic*, Tech. Report, EECS, UC Berkeley, 1996. http://www.cs.berkeley.edu/∼wkahan/ieee754status/IEEE754.PDF.
[16] P. R. WILLEMS, *On MR$^3$-type Algorithms for the Tridiagonal Symmetric Eigenproblem and the Bidiagonal SVD*, Ph.D. Thesis, Fachbereich Mathematik und Naturwissenschaften, Bergische Universität Wuppertal, Wuppertal, Germany, 2010.
[17] P. R. WILLEMS AND B. LANG, *A framework for the MR$^3$ algorithm: theory and implementation*, Preprint BUW-SC 2011/2, Fachbereich Mathematik und Naturwissenschaften, Bergische Universität Wuppertal, Wuppertal, Germany, 2011.
[18] ———, *Twisted factorizations and qd-type transformations for the MR$^3$ algorithm—new representations and analysis*, Preprint BUW-SC 2011/3, Bergische Universität Wuppertal, Fachbereich Mathematik und Naturwissenschaften, Wuppertal, Germany, 2011.

**Appendix A. The details of the error analyses.**

In the follwing we provide the more technical details to complete the analyses in Sections 3.2, 3.3 and 3.5.

**A.1. Completing the analysis of keeping a block.** In this section we prove Lemma 3.4 by specifying the necessary perturbations.

*Proof*. Concerning the intermediate $x$ and $s_{i+2}$, under the assumption that the execution environment obeys the floating-point model (3.5), we have

$$x(1+\beta_x) \;=\; e_i^2(s_i-\tau)\frac{1+\varepsilon_i}{(1+\sigma_i)(1+\alpha_1)} - d_i d_i^+\tau(1+\alpha_2), \tag{A.1}$$

$$s_{i+2}(1+\beta_s) \;=\; e_{i+1}^2(1+\varepsilon_{i+1})\,x\big/\big(\Delta_i\,\Delta_i^+\big), \tag{A.2}$$

where $|\alpha_1|, |\beta_x| \le \epsilon_\diamond$, $\alpha_2 \doteq \epsilon(2)$, $\beta_s \doteq \epsilon(3)$ and $\varepsilon_i$, $\varepsilon_{i+1}$, $\sigma_i$ stem from (3.7) and (3.11). Note that the bounds for $\alpha_2$ and $\beta_s$ were obtained by combining two (three) terms $1+\xi_j$ or $(1+\xi_j)^{-1}$ with $|\xi_j| \le \epsilon_\diamond$ according to (3.6). In the following we will frequently invoke Theorem 3.3 without further notice.

The shift need not be perturbed: $\widetilde{\tau}_i = \widetilde{\tau}_{i+1} = \tau$. Then, as per requirement $s_i = \widetilde{s}_i$ and trivially also $s_{i+1} = 0 = \widetilde{s}_{i+1}$, the default perturbations defined in (3.12) can be employed for $d_i$, $d_i^+$, $c_{i+1}$ and $c_{i+1}^+$, to the effect

$$\begin{aligned} \widetilde{d}_i &:= d_i(1+\sigma_i), & \widetilde{c}_{i+1} &:= c_{i+1}(1+\sigma_{i+1}),\\ \widetilde{d}_i^+ &:= d_i^+(1+\delta_i^+)(1+\sigma_i), & \widetilde{c}_{i+1}^+ &:= c_{i+1}^+(1+\delta_{i+1}^+)(1+\sigma_{i+1}). \end{aligned} \tag{A.3}$$

So far this gives us (3.14) and (3.15), but we still have to assure (3.16). We have not touched either $e_i$ or $e_{i+1}$ and can perturb them freely to achieve this.

As a first step, perturb $e_i \rightsquigarrow \widetilde{e}_i$ to control the subtraction involved in computing $x$. The goal is to get the exact intermediate for perturbed data, $\widetilde{x}$, to be a small relative perturbation of the computed value, $x$. We multiply (A.1) with $(1+\sigma_i)^2(1+\delta_i^+)/(1+\alpha_2)$ and substitute (A.3) to obtain

$$x(1+\beta_x)\frac{(1+\sigma_i)^2(1+\delta_i^+)}{1+\alpha_2} \;=\; e_i^2(s_i-\tau)\frac{1+\varepsilon_i}{(1+\sigma_i)(1+\alpha_1)}\frac{(1+\sigma_i)^2(1+\delta_i^+)}{1+\alpha_2} - \widetilde{d}_i\widetilde{d}_i^+\tau.$$

This tells us how to achieve

$$\widetilde{x} \;=\; \widetilde{e}_i^2(s_i-\tau) - \widetilde{d}_i\widetilde{d}_i^+\tau,$$

namely by setting

$$\widetilde{e}_i \;:=\; e_i\cdot\sqrt{\frac{(1+\varepsilon_i)(1+\sigma_i)(1+\delta_i^+)}{(1+\alpha_1)(1+\alpha_2)}} \;\doteq\; e_i\cdot\big(1+\epsilon^{[4]}(3)\big), \tag{A.4}$$

which gives

$$\widetilde{x} \;=\; x\frac{(1+\beta_x)(1+\sigma_i)^2(1+\delta_i^+)}{1+\alpha_2} \;\doteq\; x\cdot\big(1+\epsilon(6)\big).$$

Finally, we perturb $e_{i+1} \rightsquigarrow \widetilde{e}_{i+1}$ to get $\widetilde{s}_{i+2} = s_{i+2}$. As the computation of $s_{i+2}$ involves the block determinants $\Delta_i$ and $\Delta_i^+$, we have to control the perturbation's effect on them. Just

ETNA
Kent State University
http://etna.math.kent.edu



for this purpose did we craft the tool (3.9). Invoking (3.9) with $F := (1+\sigma_i)/(1+\alpha_2)$ and substituting the values $\varrho(d_i)$, $\varrho(c_{i+1})$ and $\varrho(e_i)$ from (A.3) and (A.4) yields

$$\widetilde{\Delta}_i \;=\; \Delta_i(1+\beta_\Delta)(1+\sigma_i)(1+\gamma)/(1+\alpha_2),$$

with

$$|\gamma| \;\le\; \kappa_\Delta \cdot \max\left\{ \left|\frac{(1+\sigma_{i+1})(1+\alpha_2)}{1+\alpha_\Delta} - 1\right|, \left|\frac{1+\delta_i^+}{1+\alpha_1} - 1\right| \right\} \;\doteq\; \kappa_\Delta \epsilon(4).$$

In analogous fashion we obtain a bound for $\varrho(\Delta_i^+)$, using (3.9) with $F^+ := (1+\sigma_i)(1+\delta_i^+)$:

$$\widetilde{\Delta}_i^+ \;=\; \Delta_i^+(1+\beta_\Delta^+)(1+\sigma_i)(1+\delta_i^+)(1+\gamma^+), \qquad \gamma^+ \;\doteq\; \kappa_\Delta\epsilon(3).$$

The purpose for factoring out $1+\alpha_2$ from $\widetilde{\Delta}_i$ was to cancel out with the one from $\varrho(x)$ in

$$\begin{aligned}
\widetilde{e}_{i+1} \;&:=\; e_{i+1}\sqrt{\frac{1+\varepsilon_{i+1}}{1+\beta_s}\varrho(\Delta_i)\varrho(\Delta_i^+)/\varrho(x)} \\
&=\; e_{i+1}\sqrt{\frac{(1+\beta_\Delta)(1+\beta_\Delta^+)}{(1+\beta_x)(1+\beta_s)}(1+\varepsilon_{i+1})(1+\gamma)(1+\gamma^+)} \\
&\doteq\; e_{i+1}\sqrt{(1+\epsilon(7))(1+\epsilon(7\kappa_\Delta))} \\
&\doteq\; e_{i+1}\big(1+\epsilon^{[4]}(\tfrac{7}{2}+\tfrac{7}{2}\kappa_\Delta)\big).
\end{aligned}$$

Inserting everything into (A.2) we obtain the desired relation

$$\widetilde{s}_{i+2} \;=\; \widetilde{e}_{i+1}^{\,2}\big(\widetilde{e}_i^{\,2}(s_i-\tau) - \widetilde{d}_i\widetilde{d}_i^+\tau\big)/\big(\widetilde{\Delta}_i\widetilde{\Delta}_i^+\big) \;=\; s_{i+2}. \qquad \square$$

**A.2. Completing the analysis of breaking a block.** To analyze the breaking of a block, let us first identify the effects of executing Algorithm 3.3 in floating-point arithmetic:

$$s_{i+1}(1+\beta_s) \;=\; -e_i^2(1+\varepsilon_i)/d_i^+, \tag{A.5}$$

$$x(1+\beta_x) \;=\; \begin{cases} \dfrac{d_i(s_{i+1}-\tau)}{(1+\alpha_1)(1+\sigma_{i+1})} + e_i^2(1+\varepsilon_i), & \text{for Formula I,} \\[2ex] -\dfrac{s_{i+1}(s_i-\tau)}{(1+\alpha_2)(1+\sigma_i)} - d_i\tau(1+\alpha_3), & \text{for Formula II,} \end{cases} \tag{A.6}$$

$$s_{i+2}(1+\beta_s') \;=\; e_{i+1}^2(1+\varepsilon_{i+1})\,x\,/\,\big(\Delta_i d_{i+1}^+\big), \tag{A.7}$$

where $|\alpha_1|$, $|\alpha_2|$, $|\alpha_3|$, $|\beta_x|$, $|\beta_s| \le \epsilon_\diamond$, $\beta_s' \doteq \epsilon(3)$, and $\varepsilon_i$, $\varepsilon_{i+1}$, $\sigma_i$, $\sigma_{i+1}$ stem from (3.7) and (3.11).

The error analyses in the three following lemmas have in common that the shift for the index $i$ is not perturbed, $\widetilde{\tau}_i := \tau$, and that the default perturbation (3.12) is deployed for $d_i$ and $d_i^+$,

$$\widetilde{d}_i \;:=\; d_i(1+\sigma_i), \quad \widetilde{d}_i^+ \;:=\; d_i^+(1+\delta_i^+)(1+\sigma_i). \tag{A.8}$$

One main goal will always be to prove that the computed $x$ has a small relative distance to the exact one for the perturbed data,

$$\widetilde{x} \;=\; x\varrho(x) \;=\; \widetilde{d}_i(\widetilde{s}_{i+1} - \widetilde{\tau}_{i+1}) + \widetilde{e}_i^{\,2} \;=\; -\widetilde{s}_{i+1}(\widetilde{s}_i - \tau) - \widetilde{d}_i\widetilde{\tau}_{i+1}.$$

The reason is that if we have an acceptable bound on $|\varrho(x)-1|$ then based on (A.7) we can attain $\widetilde{s}_{i+2}=s_{i+2}$ by defining $\widetilde{e}_{i+1}$ according to

$$\text{(A.9)}\quad \widetilde{e}_{i+1} \;:=\; e_{i+1}\sqrt{\frac{1+\varepsilon_{i+1}}{1+\beta_s'}\cdot\frac{\varrho(\Delta_i)\varrho(d_{i+1}^+)}{\varrho(x)}} \;\doteq\; e_{i+1}\cdot(1+\epsilon^{[6]}(2))\sqrt{\frac{\varrho(\Delta_i)\varrho(d_{i+1}^+)}{\varrho(x)}}.$$

So, what is left to do is to perturb three data items and the shift for the index $i+1$, namely

$$c_{i+1}\rightsquigarrow\widetilde{c}_{i+1},\quad d_{i+1}^+\rightsquigarrow\widetilde{d}_{i+1}^+,\quad e_i\rightsquigarrow\widetilde{e}_i,\quad \tau\rightsquigarrow\widetilde{\tau}_{i+1},$$

in order to ensure (3.3) for $i+1$, that is,

$$\text{(A.10)}\qquad \widetilde{d}_{i+1}^+ \;\overset{!}{=}\; \widetilde{c}_{i+1}+\widetilde{s}_{i+1}-\widetilde{\tau}_{i+1}.$$

With these preparations done, the following three lemmas can be considered separately, as they do not depend on each other.

LEMMA A.1. *For Formula I, let* $\kappa[x]=\kappa_+\big(\mathrm{fl}(d_i(s_{i+1}-\tau)),\mathrm{fl}(e_i^2)\big)$. *Then we can find a perturbation*

$$\begin{aligned} c_{i+1}\rightsquigarrow\widetilde{c}_{i+1} &\;\doteq\; \epsilon(1), & e_i\rightsquigarrow\widetilde{e}_i &\;\doteq\; \epsilon^{[4]}(2),\\ d_{i+1}^+\rightsquigarrow\widetilde{d}_{i+1}^+ &\;\doteq\; \epsilon(2), & e_{i+1}\rightsquigarrow\widetilde{e}_{i+1} &\;\doteq\; \epsilon^{[6]}(4+\kappa[x]+\kappa_\Delta), \end{aligned}$$

*such that (A.10) is fulfilled with* $\widetilde{\tau}_{i+1}=\tau$. *The computed adjustment* $s_{i+1}$ *will be exact, that is,* $s_{i+1}=\widetilde{s}_{i+1}$.

*Proof.* Based on (A.5) we attain $\widetilde{s}_{i+1}=s_{i+1}$ by setting

$$\text{(A.11)}\quad \widetilde{e}_i \;:=\; e_i\sqrt{\frac{\varrho(d_i^+)(1+\varepsilon_i)}{1+\beta_s}} \;=\; e_i\sqrt{\frac{(1+\varepsilon_i)(1+\sigma_i)(1+\delta_i^+)}{1+\beta_s}} \;\doteq\; e_i\cdot(1+\epsilon^{[4]}(2)).$$

Inserting the default perturbations (3.12) for $c_{i+1}$, $d_{i+1}^+$ into (3.11) gives us

$$\widetilde{d}_{i+1}^+ \;=\; \widetilde{c}_{i+1}+s_{i+1}-\tau,$$

from which we obtain (A.10) by using $s_{i+1}=\widetilde{s}_{i+1}$ and fixing $\widetilde{\tau}_{i+1}\;:=\;\tau$. For the intermediate $x$ we make use of $\widetilde{s}_{i+1}=s_{i+1}$, $\widetilde{\tau}_{i+1}=\tau$, (A.8) and (A.11) to obtain

$$\widetilde{x} \;=\; \widetilde{d}_i(s_{i+1}-\tau)+\widetilde{e}_i^2 \;=\; d_i(s_{i+1}-\tau)(1+\sigma_i)+e_i^2\frac{(1+\varepsilon_i)(1+\sigma_i)(1+\delta_i^+)}{1+\beta_s}.$$

Now we cast this in terms of (A.6) and use the definition of $\kappa[x]$ to see that the perturbation's effect on $x$ can be controlled as

$$\varrho(x) \;=\; (1+\beta_x)(1+\sigma_i)(1+\xi),\qquad \xi\;\doteq\;\epsilon(2\kappa[x]).$$

Since $\varrho(d_i)$ and $\varrho(e_i^2)$ have the common factor $F\;:=\;1+\sigma_i$, (3.9) lets us conclude that the computed $\Delta_i$ relates to the exact block determinant for perturbed data according to $\widetilde{\Delta}_i=\Delta_i\varrho(\Delta_i)$, where

$$\varrho(\Delta_i) \;=\; (1+\beta_\Delta)(1+\sigma_i)(1+\gamma),\quad \gamma\;\doteq\;\epsilon(2\kappa_\Delta).$$

Finally, plug the obtained $\varrho(\widetilde{d}_i)$, $\varrho(x)$ and $\varrho(\Delta_i)$ into (A.9) and cancel terms to determine $\widetilde{e}_{i+1}$ such that $\widetilde{s}_{i+2} = s_{i+2}$ holds. $\square$

LEMMA A.2. *For Formula I, let $|s_{i+1}| \leq R_{\mathrm{brk}}|\tau|$ for a parameter $R_{\mathrm{brk}} > 1$. Then there is a perturbation*

$$\begin{aligned} c_{i+1} \rightsquigarrow \widetilde{c}_{i+1} &\doteq \epsilon(1), & e_i \rightsquigarrow \widetilde{e}_i &\doteq \epsilon^{[4]}(2),\\ d^+_{i+1} \rightsquigarrow \widetilde{d}^+_{i+1} &\doteq \epsilon(2), & e_{i+1} \rightsquigarrow \widetilde{e}_{i+1} &\doteq \epsilon^{[6]}(\tfrac{9}{2} + \tfrac{1}{2}\kappa_\Delta),\\ \tau \rightsquigarrow \widetilde{\tau}_{i+1} &\doteq \epsilon(4R_{\mathrm{brk}}), \end{aligned}$$

*such that (A.10) is fulfilled. The computed adjustment $s_{i+1}$ will satisfy*

$$\widetilde{s}_{i+1} \doteq s_{i+1}(1 + \epsilon(4)).$$

*Proof*. We cannot assume that the computation of $x$ is well-conditioned. Therefore we choose the perturbation $e_i \rightsquigarrow \widetilde{e}_i$ specifically to safeguard the subtraction involved in computing $x$, namely as

$$\widetilde{e}_i := e_i \cdot \sqrt{(1+\varepsilon_i)(1+\sigma_i)(1+\sigma_{i+1})(1+\alpha_1)} \doteq e_i \cdot (1 + \epsilon^{[4]}(2)).$$

This gives us

$$x(1+\beta_x)(1+\sigma_i)(1+\sigma_{i+1})(1+\alpha_1) = \widetilde{d}_i(s_{i+1} - \tau) + \widetilde{e}_i^2, \tag{A.12}$$

and together with (A.5) and (A.8) we obtain

$$\widetilde{s}_{i+1} = -\frac{\widetilde{e}_i^2}{\widetilde{d}_i^+} = s_{i+1}\frac{(1+\sigma_{i+1})(1+\alpha_1)(1+\beta_s)}{(1+\delta_i^+)} \doteq s_{i+1}(1+\epsilon(4)).$$

Now employ the precondition and invoke (3.13) to define $\widetilde{\tau}_{i+1}$ satisfying

$$\varrho(\tau_{i+1}) \doteq 1 + \epsilon(4R_{\mathrm{brk}}),$$

such that $s_{i+1} - \tau = \widetilde{s}_{i+1} - \widetilde{\tau}_{i+1}$. Together with (A.12) this gives

$$\widetilde{x} = x(1+\beta_x)(1+\sigma_i)(1+\sigma_{i+1})(1+\alpha_1), \tag{A.13}$$

and together with the default perturbations (3.12) for $c_{i+1}$, $d^+_{i+1}$, as well as (3.11), we get the desired relation (A.10).

Concerning the block determinant $\Delta_i$, note that $\varrho(d_i)\varrho(c_{i+1})$ and $\varrho(e_i^2)$ have the common factor $F = (1+\sigma_i)(1+\sigma_{i+1})$, so our tool (3.9) gives

$$\varrho(\Delta_i) \doteq (1+\beta_\Delta)(1+\sigma_i)(1+\sigma_{i+1})(1+\epsilon(\kappa_\Delta)). \tag{A.14}$$

Invoke (A.9), insert (A.14), (3.12) and (A.13), and cancel terms to find the right perturbation for $e_{i+1}$. $\square$

LEMMA A.3. *For Formula II, let $|s_{i+1}| > R_{\mathrm{brk}}|\tau|$ for a parameter $R_{\mathrm{brk}} > 1$ and define $R^* := (R_{\mathrm{brk}} - 1)^{-1}$. Then there is a perturbation*

$$\begin{aligned} c_{i+1} \rightsquigarrow \widetilde{c}_{i+1} &\doteq \epsilon(3 + 2R^*), & e_i \rightsquigarrow \widetilde{e}_i &\doteq \epsilon^{[4]}(3),\\ d^+_{i+1} \rightsquigarrow \widetilde{d}^+_{i+1} &\doteq \epsilon(4 + 2R^*), & e_{i+1} \rightsquigarrow \widetilde{e}_{i+1} &\doteq \epsilon^{[6]}(\tfrac{11}{2} + 2\kappa_\Delta + (\kappa_\Delta + 1)R^*), \end{aligned}$$

*such that (A.10) is fulfilled with $\widetilde{\tau}_{i+1} = \tau$. The computed adjustment $s_{i+1}$ will satisfy*

$$\widetilde{s}_{i+1} \doteq s_{i+1}(1 + \epsilon(2)).$$

*Proof.* Again, our first action is to perturb $e_i$ to safeguard the computation of $x$. Setting

$$\widetilde{e}_i \;:=\; e_i \sqrt{\frac{(1+\varepsilon_i)(1+\sigma_i)(1+\delta_i^+)}{(1+\alpha_2)(1+\alpha_3)(1+\beta_s)}} \;\doteq\; e_i\,(1+\epsilon^{[4]}(3))$$

we reach that goal, because in concert with (A.5) and (A.6) it gives us

$$\widetilde{s}_{i+1} \;=\; -\frac{\widetilde{e}_i^2}{\widetilde{d}_i^+} \;=\; s_{i+1}\frac{1}{(1+\alpha_2)(1+\alpha_3)} \;\doteq\; s_{i+1}\,(1+\epsilon(2)),$$

as well as

$$x(1+\beta_x)(1+\sigma_i)/(1+\alpha_3) \;=\; \frac{\widetilde{e}_i^2}{\widetilde{d}_i^+}(s_i-\tau) - \widetilde{d}_i\tau \;=\; \widetilde{x}. \tag{A.15}$$

The precondition implies $|s_{i+1}| < (1+R^*)|s_{i+1}-\tau|$. Hence,

$$\widetilde{s}_{i+1}-\tau \;=\; s_{i+1}(1+\alpha_2)^{-1}(1+\alpha_3)^{-1}-\tau \;=\; (s_{i+1}-\tau)(1+\zeta)$$

with

$$\zeta \;=\; \left((1+\alpha_2)^{-1}(1+\alpha_3)^{-1}-1\right)\frac{s_{i+1}}{s_{i+1}-\tau} \;\doteq\; \epsilon(2+2R^*).$$

Thus, together with $\widetilde{\tau}_{i+1}=\tau$ and (3.11) we can achieve the desired relation (A.10) through

$$\begin{aligned} \widetilde{c}_{i+1} &\;:=\; c_{i+1}\cdot(1+\sigma_{i+1})(1+\zeta) &&\doteq\; c_{i+1}\cdot(1+\epsilon(3+2R^*)),\\ \widetilde{d}_{i+1}^+ &\;:=\; d_{i+1}^+\,(1+\delta_{i+1}^+)(1+\sigma_{i+1})(1+\zeta) &&\doteq\; d_{i+1}^+\,(1+\epsilon(4+2R^*)). \end{aligned} \tag{A.16}$$

Concerning the block determinant, invoke (3.9) with $F:=1+\sigma_i$ for

$$\varrho(\Delta_i) \;=\; (1+\beta_\Delta)(1+\sigma_i)(1+\gamma), \qquad \gamma \;\doteq\; \kappa_\Delta\epsilon(4+2R^*). \tag{A.17}$$

Use (A.9) together with (A.17), (A.16) and (A.15) for the definition of $\widetilde{e}_{i+1}$. ∎

**A.3. Completing the analysis of creating blocks and handling overlap.** The meaning of the indices $j$ and $k$ to denote the beginning and end of the overlap sequence will remain in effect during this section, thus freeing $i$ to be used as running index.

We can summarize the effects of executing Algorithm 3.5 in floating-point arithmetic by stating that, for $i=j,\dots,k$,

$$s_{i+1} \;=\; (1+\varepsilon_i)(1+\alpha_i)\cdot\begin{cases} e_i^2/(\dots) & \text{with } \alpha_i \doteq \epsilon(1), \quad \text{if } i=j,\\ e_i^2\cdot(\dots) & \text{with } \alpha_i \doteq \epsilon(2), \quad \text{if } j<i<k,\\ e_k^2\,x\,/(\dots) & \text{with } \alpha_i \doteq \epsilon(4), \quad \text{if } i=k, \end{cases} \tag{A.18}$$

introducing the intermediate quantity

$$x \;=\; \begin{cases} (s_k-\tau)d_{k-1}(1+\beta_k)/(1+\sigma_k) \;+\; e_{k-1}^2(1+\varepsilon_{k-1}), & \text{if } k\in\Omega,\\ (s_k-\tau)d_{k-1}^+(1+\beta_k)/(1+\sigma_k) \;-\; e_{k-1}^2(1+\varepsilon_{k-1}), & \text{if } k\in\Omega^+, \end{cases} \tag{A.19}$$

where $|\beta_k| \le \epsilon_\diamond$ and $\varepsilon_i$, $\sigma_k$ are those from (3.7) and (3.11), respectively. Note that the rounding error from the outermost addition in computing $x$ contributes to $\alpha_k$ and that we use the same names $x$ and $\beta_k$ for both types of endings.

ETNA
Kent State University
http://etna.math.kent.edu



The error analysis for Algorithm 3.5 will proceed as follows. We will first focus on the start and the repeating middle part up to $s_k$. This is the hardest part, as it involves dealing with perturbation bounds that depend on the length of the sequence (i.e., the distance to $j$). Once that is covered, we can deal with the two types of endings and wrap up.

As was already hinted at, the main challenge is that the required perturbations of $e_i$ and $\Delta_i$ depend on $e_{i-1}$ and $\Delta_{i-1}$, respectively. With the intent of handling these interdependencies more fluently, we define numbers $p_i$ and $q_i$ to be minimal such that

$$\varrho(e_i^2)/(1+\varepsilon_i) \doteq 1+\epsilon(p_i), \qquad i=j,\dots,k,$$

and

$$\begin{aligned} \varrho(\Delta_i) &\doteq 1+\epsilon(q_i), \quad \text{for } i+1\in\Omega, \\ \varrho(\Delta_i^+) &\doteq 1+\epsilon(q_i), \quad \text{for } i+1\in\Omega^+ \end{aligned} \qquad i=j,\dots,k-1.$$

For the diagonal data of $\mathsf{D}$ and $\mathsf{D}^+$ we deploy the default perturbations (3.12). Furthermore, we will have to use the tool (3.9) so often that we will not always state the reference.

**A.3.1. Two beginnings.** The "right" perturbation $e_j \rightsquigarrow \widetilde{e}_j$ to get $s_{j+1}$ to be exact for the perturbed data is given by

$$\text{(A.20)} \qquad \varrho(e_j^2) := (1+\varepsilon_j)(1+\alpha_j)\cdot\begin{cases}\varrho(d_j^+), & \text{if } j+1\in\Omega,\\ \varrho(d_j), & \text{if } j+1\in\Omega^+\end{cases} \qquad \Longrightarrow \quad p_j\le 3.$$

The default perturbations (3.12) for this case are

$$\begin{aligned} \widetilde{d}_j &= d_j(1+\sigma_j), & \varrho(D) &= (1+\sigma_{j+1}),\\ \widetilde{d}_j^+ &= d_j^+(1+\delta_j^+)(1+\sigma_j), & \varrho(D^+) &= (1+\delta_{j+1}^+)(1+\sigma_{j+1}), \end{aligned}$$

where either $D=c_{j+1}$, $D^+=d_{j+1}^+$ if $j+1\in\Omega$, or $D=d_{j+1}$, $D^+=c_{j+1}^+$ if $j+1\in\Omega^+$. Hence, depending on the situation we can invoke (3.9) with $F:=1+\sigma_{j+1}$, or the analogous formula for $\varrho(\Delta^+)$ with $F^+:=1+\sigma_{j+1}$ to reveal the perturbation's effect on the first block determinant of the sequence to be

$$\text{(A.21)} \qquad \begin{cases} \varrho(\Delta_j) \doteq 1+\epsilon(2+2\kappa_\Delta), & \text{if } j+1\in\Omega,\\ \varrho(\Delta_j^+) \doteq 1+\epsilon(2+4\kappa_\Delta), & \text{if } j+1\in\Omega^+\end{cases} \qquad \Longrightarrow \quad q_j\le 2+4\kappa_\Delta.$$

**A.3.2. The middle part.** For $i=j+1:k-1$, the perturbation $e_i\rightsquigarrow\widetilde{e}_i$ to get $s_{i+1}=\widetilde{s}_{i+1}$ is

$$\text{(A.22)} \quad \varrho(e_i^2) := (1+\varepsilon_i)(1+\alpha_i)\cdot\begin{cases}\varrho(\Delta_{i-1})/\varrho(d_{i-1}), & i\in\Omega,\\ \varrho(\Delta_{i-1}^+)/\varrho(d_{i-1}^+), & i\in\Omega^+\end{cases} \qquad \Longrightarrow \quad p_i\le q_{i-1}+4.$$

Concerning the block determinants, it is not hard to realize that the maxima in (3.9) are then attained with the $e_i^2$-terms. Hence, if those are perturbed as just specified, we will have

$$q_i \le 1+\kappa_\Delta p_i.$$

Combining this with (A.22) we see that the $q_i$ obey the recurrence $q_i\le b+r\cdot q_{i-1}$ with $b=1+4\kappa_\Delta$ and $r=\kappa_\Delta$, and therefore

$$q_i \le b\cdot\phi_{i-j-1}(r)+q_j\cdot r^{i-j} \le \max\{b,q_j\}\cdot\phi_{i-j}(r) \quad \text{for } j<i<k,$$

**ETNA**
Kent State University
http://etna.math.kent.edu



where

$$\phi_m(z) \;:=\; \sum_{i=0}^{m} z^i \;=\; \begin{cases} m+1, & \text{if } z=1,\\ (z^{m+1}-1)/(z-1), & \text{otherwise.}\end{cases}$$

Together with (A.22) and the bound on $q_j$ from (A.21) we obtain

$$p_i \le 4 + (2+4\kappa_\Delta)\cdot\phi_{i-j}(\kappa_\Delta), \quad j<i<k, \tag{A.23}$$

from which (3.20) follows by noting that $\varrho(e_i^2) \doteq 1+\epsilon(1+p_i)$ due to (A.20).

In the following we will show that the recurrence is effectively "reset" when a $d_i^+$ becomes zero (yielding $s_{i+2}=0$ in line 9 of Algorithm 3.5) or, more generally, for any situation where an $|s_i|$ becomes "small". The idea here was used before, namely that if, for any $i$, the adjustment $s_i$ is not much larger in magnitude than the shift $\tau$, then the latter can be perturbed to provide freedom in the choice for $\widetilde{e}_{i-1}$.

We will limit this optimization to the middle part. So, assume we have for an $i$ with $j<i<k$ that $|s_{i+1}| \le R_{\mathrm{osq}}|\tau|$ holds for a parameter $R_{\mathrm{osq}}>0$. Then perturb $e_i \rightsquigarrow \widetilde{e}_i$ instead of (A.22) simply as

$$\widetilde{e}_i^2 \;:=\; e_i^2(1+\varepsilon_i) \tag{A.24}$$

and employ (3.13) to define $\widetilde{\tau}_{i+1}$ such that $s_{i+1}-\tau = \widetilde{s}_{i+1}-\widetilde{\tau}_{i+1}$ is upheld. Look at the middle line of (A.18) to see that the above choice for $\widetilde{e}_i$ implies

$$\varrho(s_{i+1}) \;=\; (1+\alpha_i)^{-1}\cdot\begin{cases} \varrho(d_{i-1})/\varrho(\Delta_{i-1}), & i\in\Omega,\\ \varrho(d_{i-1}^+)/\varrho(\Delta_{i-1}^+), & i\in\Omega^+.\end{cases}$$

Hence, the necessary perturbation $\widetilde{\tau}_{i+1}=\tau\varrho(\tau_i)$ to the shift from (3.13) can be bounded as

$$\big|\varrho(\tau_{i+1})-1\big| \;\le\; R_{\mathrm{osq}}\big|\varrho(s_{i+1})-1\big| \;\le\; R_{\mathrm{osq}}\,\epsilon^{[2]}(4+q_{i-1}).$$

This effectively resets the recurrence, since (A.24) gives $p_i=0$ and $q_i \le 1+5\kappa_\Delta$. The parameter $R_{\mathrm{osq}}$ is yet unspecified. Indeed, we can choose it freely; its sole purpose is to control $\varrho(\tau_i)$. The following lemma summarizes where we stand.

LEMMA A.4. *Let in Algorithm 3.1 the adjustments $s_{j+1},\dots,s_k$ for an overlap sequence be computed as in Algorithm 3.5. Fix a parameter $R_{\mathrm{osq}}>0$ and define for $i=j,\dots,k-1$*

$$h(i) \;:=\; i-\max\big\{m \,\big|\, m=j \text{ or } j<m\le i \text{ and } |s_{m+1}|\le R_{\mathrm{osq}}|\tau|\big\}.$$

*Then there is a perturbation such that (3.3) is fulfilled for $i=j+1,\dots,k$. With suitable numbers*

$$q_i \le (2+4\kappa_\Delta)\phi_{h(i)}(\kappa_\Delta), \quad j\le i<k,$$

*the individual perturbations can be bounded as follows:*

$$\begin{aligned}
&\mathsf{D}(i,i)\rightsquigarrow\widetilde{\mathsf{D}}(i,i) \doteq \epsilon(1), \qquad \mathsf{D}^+(i,i)\rightsquigarrow\widetilde{\mathsf{D}}^+(i,i) \doteq \epsilon(2),\\
&e_j\rightsquigarrow\widetilde{e}_j \doteq \epsilon^{[4]}(2), \qquad e_i\rightsquigarrow\widetilde{e}_i \doteq \epsilon^{[4]}(\tfrac{5}{2}+\tfrac{1}{2}q_{i-1}), \quad j<i<k,\\
&\widetilde{\tau}_j=\tau, \qquad \widetilde{\tau}_{j+1}=\tau, \qquad \tau\rightsquigarrow\widetilde{\tau}_i \doteq R_{\mathrm{osq}}\,\epsilon^{[2]}(4+q_{i-2}), \quad j+1<i\le k.
\end{aligned}$$

*The implied perturbations for the secondary quantities will obey*

$$\left.\begin{array}{ll} i+1\in\Omega: & |\varrho(\Delta_i)-1| \\ i+1\in\Omega^+: & |\varrho(\Delta_i^+)-1| \end{array}\right\} \le \epsilon(q_i), \quad j\le i<k,$$

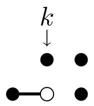
$$\widetilde{s}_{j+1}=s_{j+1}, \qquad |\varrho(s_{i+1})-1| \le \epsilon^{[2]}(4+q_{i-1}), \quad j<i<k.$$

The main purpose of the preceding analysis was not only to provide worst-case perturbation bounds, but also to give us the means to control the computation of an overlap sequence. Whenever a $d_i^+$ becomes small enough in magnitude that it causes $|s_{i+2}|$ to become small as well with respect to $|\tau|$ (cf. Section 3.6 for a detailed test to detect this after computing $d_i^+$), we can choose a $2\times2$ pivot while keeping a tight control over the error bounds. Otherwise, we retain the option of keeping track of the current length of the sequence (better: the current $h(i)$) and if this exceeds some threshold just choose a single pivot, thus ending the sequence and capping the bounds.

The challenging part is done; what remains is to deal with the two possible ways for the sequence to end. For the following we have to rely on the stronger Block Criterion II, and in particular (3.19).

**A.3.3. An end by creation: $k\in\Omega^+$.** We assume that $s_k-\tau$ is exact, i.e., $s_k-\tau=\widetilde{s}_k-\widetilde{\tau}_k$. Lemma A.4 does deliver this. What remains to be done is just to define $e_k \rightsquigarrow \widetilde{e}_k$ such that $s_{k+1}$ becomes exact, $s_{k+1}=\widetilde{s}_{k+1}$. The following picture summarizes the situation (cf. Definition 3.1 for the pictogram):

In: $s_k-\tau=\widetilde{s}_k-\widetilde{\tau}_k$




Out: $s_{k+1}=\widetilde{s}_{k+1}$.

Note that we may have $k=j+1$.

The task might be impossible should the computation of the intermediate $x$ in line 14 of Algorithm 3.5 involve cancellation. We will show that this computation cannot be too badly conditioned if the choice for a $2\times2$ pivot at $k-1$ was based on BC–II.

Depending on the situation at hand, $s_k$ is defined in one of two ways:

$$s_k \;=\; \begin{cases} e_{k-1}^2/d_{k-1}, & \text{if } k-1\notin\Omega, \\ e_{k-1}^2 d_{k-2}/\Delta_{k-2} & \text{if } k-1\in\Omega. \end{cases}$$

Thus, fulfillment of the second condition in BC–II implies $|s_k|<(K_\square/3)e_{k-1}^2/|d_{k-1}^+|$, which yields

$$|s_k-\tau||d_{k-1}^+| \;\le\; 2\max\big\{|s_k|,|\tau|\big\}|d_{k-1}^+| \;<\; \tfrac{2}{3}K_\square e_{k-1}^2.$$

Because of the safeguard on $K_\square$ to anticipate rounding errors, we can safely conclude that the computation of $x$ in line 14 of Algorithm 3.5 is governed by

$$\kappa_-\big(\mathrm{fl}((s_k-\tau)d_{k-1}^+),\mathrm{fl}(e_{k-1}^2)\big) \;<\; \frac{3+2K_\square}{3-2K_\square} \;=:\; \kappa[x]. \tag{A.25}$$

Hence we can control the rounding errors expressed in the lower line of (A.19) as

$$\widetilde{x}=(s_k-\tau)\widetilde{d}_{k-1}^+-\widetilde{e}_{k-1}^2=x\varrho(x) \quad \text{with} \quad |\varrho(x)-1|\le\kappa[x]\,\epsilon\big(\max\{4,p_{k-1}\}\big). \tag{A.26}$$

Finally,

$$\varrho(e_k^2) \;:=\; (1+\varepsilon_k)\varrho(\Delta_{k-1}^+)\varrho(d_k)(1+\alpha_k)/\varrho(x) \tag{A.27}$$

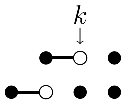
makes $s_{k+1}$ exact, as desired.

**ETNA**
**Kent State University**
**http://etna.math.kent.edu**



**A.3.4. An end by breaking:** $k \in \Omega$**.** The situation at hand is

In: $s_k - \tau = \widetilde{s}_k - \widetilde{\tau}_k$




Out: $s_{k+1} = \widetilde{s}_{k+1}$.

This is very similar to breaking a block from Section 3.3. Indeed, the way $s_{k+1}$ is computed in line 12 of Algorithm 3.5 corresponds to Formula I in Algorithm 3.3. The difference is that, here, we have overlap at the left end: $k-1 \in \Omega^+$. This will prove to be beneficial, because here we can assume that the created block covering indices $\{k-2, k-1\}$ satisfies BC–II, which will allow us to show that the single computational branch does work for all configurations. Note that we must have $k > j+1$, so that $s_k$ is defined as

$$s_k \;=\; -e_{k-1}^2 d_{k-2}^+ \big/ \Delta_{k-2}^+.$$

For the analysis, we consider two cases, depending on the parameter

$$\overline{R} \;:=\; (1 - 2K_\square)^{-1},$$

which is larger than one only if $K_\square < 1/2$ holds, and we assume it does.

**Case 1:** $|s_k| \le \overline{R}\,|\tau|$. We employ the perturbations from Lemma A.4 but throw away those for $e_{k-1}$ and $\tau_k$. We will redefine them manually to still retain $s_k - \tau = \widetilde{s}_k - \widetilde{\tau}_k$. With the intent of controlling $x$, we perturb $e_{k-1}$ instead as

$$\varrho(e_{k-1}^2) \;:=\; (1+\varepsilon_{k-1})\varrho(d_{k-1})(1+\sigma_k)\big/(1+\beta_k),$$

yielding

$$\widetilde{s}_k \;=\; s_k\varrho(s_k) \;=\; s_k\varrho(d_{k-2}^+)\varrho(d_{k-1})\big[\varrho(\Delta_{k-2}^+)(1+\alpha_{k-1})(1+\beta_k)\big]^{-1}(1+\sigma_k).$$

Now invoke (3.13) to move the relative perturbations to the shift. This defines $\tau \rightsquigarrow \widetilde{\tau}_k$ with

$$\big|\varrho(\tau_k) - 1\big| \;\le\; \overline{R}\cdot|\varrho(s_k) - 1| \;\le\; \overline{R}\,\epsilon^{[2]}(7 + q_{k-2}),$$

and gives the desired $s_k - \tau = \widetilde{s}_k - \widetilde{\tau}_k$. According to the upper line in (A.19) we therefore have

$$\text{(A.28)}\quad \widetilde{x} \;=\; (s_k - \tau)\widetilde{d}_{k-1} + \widetilde{e}_{k-1}^2 \;=\; x\frac{\varrho(d_{k-1})(1+\sigma_k)}{1+\beta_k} \quad\Longrightarrow\quad |\varrho(x) - 1| \;\le\; \epsilon(3).$$

**Case 2:** $|s_k| > \overline{R}\,|\tau|$. In this case we can keep the permutations for $e_{k-1}$ and $\tau_k$ from Lemma A.4; therefore we already have $s_k - \tau = \widetilde{s}_k - \widetilde{\tau}_k$. If BC–II is fulfilled for the created block ending at $k-1 \in \Omega^+$, then our definition of $\overline{R}$ and (3.19) for $i = k-2$ give

$$|s_k - \tau||d_{k-1}| \;<\; 2(1 - K_\square)|s_k||d_{k-1}| \;<\; \tfrac{2}{3}K_\square e_{k-1}^2.$$

In fact, the safeguard on $K_\square$ allows us to relax this relation by a small perturbation on each side, so that we may assume the computation of $x$ to be controlled by

$$\kappa_+\big(\mathrm{fl}((s_k - \tau)d_{k-1}), \mathrm{fl}(e_{k-1}^2)\big) \;<\; \frac{3 + 2K_\square}{3 - 2K_\square} \;=\; \kappa[x],$$

with the same $\kappa[x]$ as in (A.25). In fact, this was the motivation for our definition of $\overline{R}$. Thus,

$$\text{(A.29)}\quad \widetilde{x} = (s_k - \tau)\widetilde{d}_{k-1} + \widetilde{e}_{k-1}^2 = x\varrho(x) \quad \text{with} \quad |\varrho(x) - 1| \le \kappa[x]\,\epsilon(\max\{3, p_{k-1}\}).$$

ETNA
Kent State University
http://etna.math.kent.edu



In both cases the analysis revealed that the computed $x$ is close to the exact value for computed data, with a bound on $\varrho(x)$. Thus we achieve $s_{k+1} = \widetilde{s}_{k+1}$ with the perturbation

$$\widetilde{e}_k^2 \ := \ e_k^2(1+\varepsilon_k)\varrho(\Delta_{k-1})\varrho(d_k^+)(1+\alpha_k)\big/\varrho(x). \tag{A.30}$$

LEMMA A.5. *Let in Algorithm 3.1 the adjustments $s_{j+1},\dots,s_{k+1}$ for an overlap sequence be computed as in Algorithm 3.5. Furthermore, let all $2\times2$ pivots in $\mathsf{D}^+$ satisfy Block Criterion II with a parameter $K_\square < 1/3$.*

*Then there is a perturbation such that (3.3) holds for $j \le i \le k$ and the computed $s_{k+1}$ is exact. The componentwise bounds from Lemma A.4 do apply, with the following adjustments and extensions:*

$$\mathsf{D}(k,k) \rightsquigarrow \widetilde{\mathsf{D}}(k,k) \ \doteq \ \epsilon(1), \qquad \mathsf{D}^+(k,k) \rightsquigarrow \widetilde{\mathsf{D}}^+(k,k) \ \doteq \ \epsilon(2).$$

| *if $k = j+1$:* | *if $k > j+1$:* |
|---|---|
| $e_k \rightsquigarrow \widetilde{e}_k \ \doteq \ \epsilon^{[6]}(7+\frac{1}{2}q_{k-1})$, | $e_k \rightsquigarrow \widetilde{e}_k \ \doteq \ \epsilon^{[6]}(\frac{15}{2}+\frac{1}{2}q_{k-1}+q_{k-2})$, |
| $\tau_k \ = \ \tau$, | $\tau_k \rightsquigarrow \widetilde{\tau}_k \ \doteq \ \epsilon^{[2]}(\max\{21+3q_{k-2},\, R_{\mathrm{osq}}(4+q_{k-2})\})$, |
| $\widetilde{s}_k \ = \ s_k$, | $s_k \rightsquigarrow \widetilde{s}_k \ \doteq \ \epsilon^{[2]}(7+q_{k-2})$. |

*Proof.* The subtle point is that we might have redefined $\widetilde{e}_{k-1}$ if ending with breaking a block. But it is easy to verify that the bounds to $\varrho(e_{k-1})$ and $\varrho(\Delta_{k-1})$ from Lemma A.4 do still apply, just $\varrho(s_k)$ can increase from $1+\epsilon^{[2]}(4+q_{k-2})$ to $1+\epsilon^{[2]}(7+q_{k-2})$.

For the final $e_k \rightsquigarrow \widetilde{e}_k$, note that regardless of the type of ending, the perturbation's effect on the intermediate $x$ as stated in (A.26), (A.28) and (A.29) allows the uniform bound $|\varrho(x)-1| \le \kappa[x]\,\epsilon(\max\{4, p_{k-1}\})$ with $\kappa[x] = (3+2K_\square)\big/(3-2K_\square)$. The prerequisite $K_\square < 1/3$ gives $\kappa[x] < 2$ and $\overline{R} < 3$. The latter determines the bound on $\varrho(\tau_k)$, and (A.27) and (A.30) lead to the stated bound for $\varrho(e_k)$; recall that the analysis leading to Lemma A.4 revealed $p_j \le 3$ and $p_i \le q_{i-1}+4, i > j$. $\square$