



Accurate eigenvalues and exact zero Jordan blocks of totally nonnegative matrices

Plamen Koev¹

Received: 22 May 2018 / Revised: 13 November 2018 / Published online: 9 January 2019
© Springer-Verlag GmbH Germany, part of Springer Nature 2019

Abstract

We present a new algorithm which, given a bidiagonal decomposition of a totally nonnegative matrix, computes all its eigenvalues to high relative accuracy in floating point arithmetic in $O(n^3)$ time. It also computes exactly the Jordan blocks corresponding to zero eigenvalues in up to $O(n^4)$ time.

Mathematics Subject Classification 65F15 · 15A18

1 Introduction

A matrix is totally nonnegative (TN) if all of its minors are nonnegative [2,8,10]. All n eigenvalues of an $n \times n$ TN matrix are real and nonnegative and we present a new algorithms that computes all of them, including the zero ones, to high relative accuracy in floating point arithmetic. Namely, the eigenvalues, λ_i , and their computed counterparts, $\hat{\lambda}_i$, $i = 1, 2, \dots, n$, satisfy

$$|\lambda_i - \hat{\lambda}_i| \leq O(\varepsilon)\lambda_i,$$

where ε is the machine precision. The above inequality implies that the positive eigenvalues are computed with correct sign and most significant digits and the zero ones are computed exactly. Additionally, the Jordan blocks corresponding to eigenvalue 0 (which we call *zero Jordan blocks*) are computed exactly.

The class of TN matrices includes many famous notoriously ill conditioned matrices such as Pascal, Hilbert, certain Cauchy and Vandermonde matrices, etc. Conventional eigenvalue algorithms (e.g., LAPACK [1]) can fail to compute the tiny eigenvalues of ill conditioned matrices with any relative accuracy at all. This is due to a phenomenon

✉ Plamen Koev
plamen.koev@sjsu.edu

¹ Department of Mathematics, San Jose State University, San Jose, CA 95192, USA

known as *subtractive cancellation*—the loss of correct significant digits during subtraction.

The first step in our pursuit of accurate eigenvalues is to acknowledge that the matrix entries are a poor choice of parameters to represent a TN matrix in floating point arithmetic. Indeed, even tiny relative perturbations in the matrix entries (caused by merely storing the matrix in the computer) can cause enormous relative perturbations in the tiny eigenvalues and can completely destroy the zero ones.

For example, an ε perturbation in the (2,2) entry of the following 2×2 TN matrix changes the smallest eigenvalue from 0 to about $\varepsilon/2$ then to about ε :

$$\begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix} \rightarrow \begin{bmatrix} 1 & 1 \\ 1 & 1 + \varepsilon \end{bmatrix} \rightarrow \begin{bmatrix} 1 & 1 \\ 1 & 1 + 2\varepsilon \end{bmatrix}.$$

Since every TN matrix can be decomposed as a product of nonnegative bidiagonals [3, 6, 9, 15], we follow the ideas in the nonsingular case [12] and choose such a bidiagonal decomposition as the representation of the TN matrix. This choice is motivated by the fact that the entries of a bidiagonal decomposition of a TN matrix determine all eigenvalues to high relative accuracy and determine the sizes of the zero Jordan blocks exactly (see Sect. 12).

Breaking things down further, every bidiagonal matrix is a product of what we call *elementary bidiagonal matrices* (EBMs). The EBMs differ from the identity matrix in three entries only—one offdiagonal and its two immediately adjacent diagonal neighbors. A TN matrix is thus, in turn, also a product of EBMs (Sects. 2 and 3). Following the ideas of Cryer [3], multiplication by an EBM is the only type of transformation one needs in order to reduce the TN eigenvalue problem to the bidiagonal singular value problem (Sect. 8). The latter is then solved to high relative accuracy by using the result of Demmel and Kahan [5].

Finally, the sizes of the zero Jordan blocks are inferred from the ranks of the powers of the TN matrix (Sects. 9 and 10). This requires that we compute the bidiagonal decomposition of a product of TN matrices, which is easily achieved by multiplying one of the TN matrices by all the EBMs that comprise the other (Sect. 6).

The eigenvalue computation takes $O(n^3)$ time. The cost of computing the zero Jordan blocks is $O(n^3 z)$, where z is the size of the second largest Jordan block. The overall cost is thus bounded by $O(n^4)$ (Sect. 11).

The TN matrices are the closure of the set of *nonsingular* TN matrices. Algorithms for computing all eigenvalues of *nonsingular* TN matrices to high relative accuracy were presented in [12], thus the focus and contribution of this paper is on the *singular* case. We make no assumption of singularity—our technique works regardless. As much as we do follow the general framework of our earlier approach [12], there are several important theoretical and algorithmic novelties. First, the notion of a bidiagonal decomposition of a singular TN matrix is well defined and established via the process of Neville elimination (Sect. 3). For singular TN matrices, in the process of Neville elimination, we may occasionally be faced with the need to eliminate a nonzero entry with a zero one—a task commonly considered impossible and requiring pivoting—yet we show how to do just that. Second, the resulting lack of uniqueness in this process is well understood and shown to have no impact on our ability to compute

the eigenvalues accurately (Sect. 3.1). Third, we generalize the method of [12] and show that given the bidiagonal decomposition of a TN matrix, one can compute the bidiagonal decomposition of the product of that TN matrix and an EBM without performing any subtractions even when some of the factors or the EBM are singular (Sects. 4 and 5). The process is entirely subtraction free, thus subtractive cancellation is avoided and the relative accuracy is preserved. Finally, in what we believe to be the first example of a Jordan block being computed accurately in floating point arithmetic for a matrix of any type, we reveal *exactly* the zero Jordan structure of the TN matrix. With this we solve completely and to high relative accuracy the eigenvalue problem for the class of *irreducible* TN matrices, since their nonzero eigenvalues are distinct [7] and only the zero eigenvalues can have nontrivial Jordan blocks.

The paper is organized as follows. We introduce the main building block of a TN matrix—the EBM in Sect. 2 and review bidiagonal decompositions of TN matrices in Sect. 3. The two main building blocks of our algorithms—multiplication by an EBM on the right and the left are covered in Sects. 4 and 5. We show how to compute the product of TN matrices and how to swap a zero row or a zero column with the following or the previous one in Sects. 6 and 7, respectively. Cryer’s algorithm for reducing a TN matrix to tridiagonal form is in Sect. 8. The process for computing the rank of a TN matrix is in Sect. 9 and its zero Jordan structure in Sect. 10. We discuss the cost, perturbation theory, and accuracy of our algorithm in Sects. 11 and 12. We present numerical experiments in Sect. 13 and conclude with open problems in Sect. 14.

2 Elementary bidiagonal matrices

The main building block of a bidiagonal decomposition of a TN matrix is the *elementary bidiagonal matrix* (EBM)

$$J_i(x, y, z) \equiv \begin{bmatrix} 1 & & & & \\ & \ddots & & & \\ & & y & & \\ & & x & z & \\ & & & \ddots & \\ & & & & 1 \end{bmatrix}, \quad (1)$$

which differs from the identity only in the entries $x \geq 0$, $y \geq 0$, and $z \geq 0$ in positions $(i, i-1)$, $(i-1, i-1)$, and (i, i) , respectively. For reasons that will become apparent in Sect. 4, we also require that $z > 0$ if $i < n$.

We also extensively utilize the EBM

$$E_i(b, c) \equiv J_i(b, c, 1) = \begin{bmatrix} 1 & & & & \\ & \ddots & & & \\ & & c & & \\ & & b & 1 & \\ & & & \ddots & \\ & & & & 1 \end{bmatrix}, \quad (2)$$

where $b \geq 0$ and $c \in \{0, 1\}$.

Allowing for $c = 0$ in $E_i(b, c)$ is critical for the theoretical construction of the bidigonal decomposition of *singular* TN matrices using the process of Neville elimination, which we review in the next section.

For the rest of this section we focus on how the EBMs E_i function when one needs to eliminate a single nonzero (and thus positive) entry a_{ij} in a TN matrix using the previous row.

$$\begin{bmatrix} a_{11} & & & & \\ & \ddots & & & \\ & & a_{i-1,j} & \cdots & a_{i-1,k} \cdots \\ & & a_{ij} & \cdots & a_{ik} \\ & & & \vdots & \end{bmatrix},$$

If the entry immediately above, $a_{i-1,j}$, is positive, we subtract, in a typical fashion, $x = a_{ij}/a_{i-1,j}$ times row $i - 1$ from row i to obtain a matrix A' where $a'_{ij} = 0$. The resulting decomposition is $A = E_i(x, 1) \cdot A'$.

If $a_{i-1,j} = 0$, then we obviously can't eliminate the nonzero a_{ij} with a nonsingular EBM, but we can do so with a *singular* one. We observe first that $a_{i-1,j} = 0$ coupled with $a_{ij} > 0$ implies that the entire $(i-1)$ st row of A is zero (if some $a_{i-1,k} > 0$, $k > j$, the 2×2 minor consisting of rows $i - 1$ and i and columns j and k would be negative contradicting the total nonnegativity of A).

Thus we can swap rows $i - 1$ and i with the decomposition $A = E_i(1, 0) \cdot A'$. For example,

$$\begin{bmatrix} 0 & 0 & 0 \\ 1 & 1 & 1 \\ 0 & 3 & 4 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 \\ 1 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ 0 & 3 & 4 \end{bmatrix}. \quad (3)$$

Note that we do not want to use the obvious swap matrix

$$\begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

for elimination since its leading principal 2×2 minor is -1 and thus this matrix is not TN.

Additionally, we can use the matrix $E_i(1, 0)$ to swap only the a_{ij} entry to position $(i - 1, j)$ keeping the rest of the matrix intact. For example,

$$\begin{bmatrix} 0 & 0 & 0 \\ 1 & 1 & 1 \\ 0 & 3 & 4 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 \\ 1 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \underbrace{\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 1 \\ 0 & 3 & 4 \end{bmatrix}}_B. \quad (4)$$

This swap of a single entry does not necessarily preserve the TN structure of the resulting matrix B , but it does so in all situations we care about in Sects. 4 and 5 for the banded TN matrices we encounter there.

We conclude this section with two important properties of EBMs, which we will need later on:

$$E_i(b, c)E_j(e, g) = E_j(e, g)E_i(b, c), \quad \text{if } |i - j| > 1, \quad (5)$$

$$(E_i(b, 1))^{-1} = E_i(-b, 1). \quad (6)$$

3 Bidiagonal decompositions of TN matrices

Every TN matrix can be decomposed as a product of nonnegative bidiagonals using the classical approach of *Neville elimination*. This bidiagonal decomposition is unique for nonsingular TN matrices [9], but not necessarily so for singular ones. The lack of uniqueness causes no issues since *any* decomposition of a TN matrix as a product of nonnegative bidiagonals serves equally well as a starting point for our eigenvalue algorithm. We discuss this at the end of the section.

A bidiagonal decomposition of a TN matrix is obtained using Neville elimination as follows.

We use adjacent rows and columns to eliminate a matrix to diagonal form. We eliminate the first column using row operations, then the first row using column operations, and so on until we obtain a diagonal matrix

$$D = \text{diag}(d_1, d_2, \dots, d_n). \quad (7)$$

Say, we've reached position (i, j) , $i > j$, below the diagonal (so we are using row operations for elimination).

- If the i th row of A is zero, we move on without doing anything and use $E_i(0, 1) = I$ for our decomposition. We write $A = E_i(0, 1) \cdot A'$.
- Otherwise, if the $(i - 1)$ st row is 0, we swap rows $i - 1$ and i and write $A = E_i(1, 0) \cdot A'$.
- Finally, if both rows $i - 1$ and i are nonzero, let l and k be the smallest indices, such that $a_{i-1,l} \neq 0$ and $a_{ik} \neq 0$. We must have $l \leq k$ or the minor

$$\det \begin{pmatrix} a_{i-1,k} & a_{i-1,l} \\ a_{ik} & a_{il} \end{pmatrix} = -a_{ik}a_{i-1,l} < 0,$$

which would contradict the fact that A is TN.

- If $l < k$, we move on without doing anything and write $A = E_i(0, 1) \cdot A'$.
- Otherwise (if $l = k$), we eliminate the entry a_{ik} using row $i - 1$ with multiplier $b_{ij} = \frac{a_{i-1,k}}{a_{ik}}$ to create a zero in position (i, k) . We write $A = E_i(b_{ij}, 1) \cdot A'$.

This process results in a decomposition

$$A = E_n(b_{n1}, c_{n1}) \cdots E_n(b_{n,n-1}, c_{n,n-1}) \cdot D \cdot E_n^T(b_{n-1,n}, c_{n-1,n}) \cdots E_n^T(b_{1n}, c_{1n}). \quad (8)$$

Using the commutativity relationships (5) we can combine the EBMs into bidiagonal factors

$$\begin{aligned} L^{(i)} &= E_{n-i+1}(b_{n-i+1,1}, c_{n-i+1,1}) \cdots E_n(b_{ni}, c_{ni}) \\ &= \begin{bmatrix} 1 & & & & & & \\ & \ddots & & & & & \\ & & c_{n-i+1,1} & & & & \\ & & b_{n-i+1,1} & c_{n-i+2,2} & & & \\ & & & b_{n-i+2,2} & \ddots & & \\ & & & & \ddots & \ddots & \\ & & & & & b_{n-1,i-1} & c_{ni} \\ & & & & & b_{ni} & 1 \end{bmatrix}. \end{aligned} \quad (9)$$

and repeat the same with the upper bidiagonal factors to form upper bidiagonal matrices $U^{(i)}$. Note that the diagonal entries of every bidiagonal factor are zeros or ones with the exception of the bottom right corner entry, which must equal 1.

We obtain a bidiagonal decomposition

$$A = L^{(1)} L^{(2)} \cdots L^{(n-1)} D U^{(n-1)} U^{(n-2)} \cdots U^{(1)}. \quad (10)$$

In contrast with the nonsingular case, in the above decomposition, the diagonal factor D can have zero entries on the diagonal and each factor $L^{(i)}$ can have a zero on the diagonal immediately above any nontrivial offdiagonal entry. Similarly, $U^{(i)}$ can also have a zero on the diagonal immediately to the left of any nontrivial offdiagonal entry.

We represent the bidiagonal decomposition (10) of A as a set of two $n \times n$ square arrays containing the nontrivial entries b_{ij} , c_{ij} , and d_i as

$$\mathcal{BD}(A) = [B, C]$$

with the additional convention that $b_{ii} = d_i$, $i = 1, 2, \dots, n$, are the diagonal entries of D [as in (7)] and that the diagonal entries of the array C are unused. Despite being

unused, we set the diagonal entries of C to 1 in our examples, since all nontrivial entries of C are ones for nonsingular TN matrices and it is thus convenient in practice to pass C as a matrix of all ones to our algorithm in such cases.

For example, the bidiagonal decomposition resulting from the Neville elimination of the following singular TN matrix

$$\begin{aligned} A &= \begin{bmatrix} 1 & 0 & 2 \\ 0 & 0 & 0 \\ 3 & 0 & 10 \end{bmatrix} \\ &= \begin{bmatrix} 1 & & \\ 0 & & \\ 1 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 2 \\ 3 & 0 & 10 \\ 0 & 0 & 0 \end{bmatrix} \\ &= \begin{bmatrix} 1 & & \\ 0 & & \\ 1 & 1 \end{bmatrix} \begin{bmatrix} 1 & & \\ 3 & 1 & \\ 0 & 1 & \end{bmatrix} \begin{bmatrix} 1 & 0 & 2 \\ 0 & 0 & 4 \\ 0 & 0 & 0 \end{bmatrix} \\ &= \begin{bmatrix} 1 & & \\ 0 & & \\ 1 & 1 \end{bmatrix} \begin{bmatrix} 1 & & \\ 3 & 1 & \\ 0 & 1 & \end{bmatrix} \begin{bmatrix} 1 & 2 & 0 \\ 0 & 4 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} 1 & & \\ 0 & 1 & \\ 1 & 1 & \end{bmatrix} \\ &= \begin{bmatrix} 1 & & \\ 0 & & \\ 1 & 1 \end{bmatrix} \begin{bmatrix} 1 & & \\ 3 & 1 & \\ 0 & 1 & \end{bmatrix} \begin{bmatrix} 1 & & \\ & 4 & \\ & 0 & \end{bmatrix} \begin{bmatrix} 1 & 2 & \\ 1 & 1 & \\ 1 & 1 & \end{bmatrix} \begin{bmatrix} 1 & & \\ 0 & 1 & \\ 1 & 1 & \end{bmatrix} \end{aligned}$$

is encoded as $\mathcal{BD}(A) = [B, C]$, where

$$B = \begin{Bmatrix} 1 & 2 & 1 \\ 3 & 4 & 0 \\ 1 & 0 & 0 \end{Bmatrix}, \quad C = \begin{Bmatrix} 1 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 1 \end{Bmatrix}.$$

Finally, we note the obvious fact that

$$\mathcal{BD}(A^T) = [B^T, C^T]. \quad (11)$$

In all our further considerations, we assume that $\mathcal{BD}(A)$ is given as an input and is the representation of the TN matrix A . The matrix entries of A , while easily (and accurately) obtainable from $\mathcal{BD}(A)$, will not be needed or referenced.

3.1 Other bidiagonal decompositions

If a TN matrix is represented as a different product of nonnegative bidiagonals, we can first compute the decomposition (10) from that (other) decomposition using the method of Sect. 6. This computation is subtraction-free and thus any representation of a TN matrix as a product of nonnegative bidiagonals is an equally good starting point for our algorithm.

4 Multiplication by an EBM on the right

All computations in this paper utilize only one operation: multiplication by an EBM. These multiplications can occur on the left or on the right of the matrix and in this section we consider the latter. [Multiplication by the transpose of an EBM is the same type of operation because of (11).]

In what follows, we describe how to compute $\mathcal{BD}(A \cdot J_i(x, y, z))$ given $\mathcal{BD}(A)$ and x, y, z without performing any subtractions.

We start with

$$AJ_i(x, y, z) = L^{(1)}L^{(2)}L^{(3)}\dots L^{(n-1)}DU^{(n-1)}\dots U^{(2)}U^{(1)} \cdot J_i(x, y, z). \quad (12)$$

We consecutively chase the “bulge” $J_i(x, y, z)$ to the left of the decomposition (12), preserving the structure of the bidiagonal factors:

$$AJ_i = L^{(1)}\dots L^{(i-2)}L^{(i-1)}L^{(i)}\dots L^{(n-2)}L^{(n-1)}DU^{(n-1)}\dots U^{(3)}U^{(2)}\underline{U^{(1)}J_i} \quad (13)$$

$$= L^{(1)}\dots L^{(i-2)}L^{(i-1)}L^{(i)}\dots L^{(n-2)}L^{(n-1)}DU^{(n-1)}\dots U^{(3)}\underline{U^{(2)}J_i^{(1)}\mathcal{U}^{(1)}} \quad (14)$$

$$= L^{(1)}\dots L^{(i-2)}L^{(i-1)}L^{(i)}\dots L^{(n-2)}L^{(n-1)}DU^{(n-1)}\dots \underline{U^{(3)}J_i^{(2)}\mathcal{U}^{(2)}\mathcal{U}^{(1)}} \quad (15)$$

= ...

$$= L^{(1)}\dots L^{(i-2)}L^{(i-1)}L^{(i)}\dots L^{(n-2)}\underline{L^{(n-1)}D}\underline{J_i^{(n-1)}\mathcal{U}^{(n-1)}}\dots \mathcal{U}^{(2)}\mathcal{U}^{(1)} \quad (16)$$

$$= L^{(1)}\dots L^{(i-2)}L^{(i-1)}L^{(i)}\dots L^{(n-3)}\underline{L^{(n-2)}E_i^{(n)}}\underline{\mathcal{D}}\underline{\mathcal{U}^{(n-1)}}\dots \mathcal{U}^{(2)}\mathcal{U}^{(1)} \quad (17)$$

$$= L^{(1)}\dots L^{(i-2)}L^{(i-1)}L^{(i)}\dots \underline{L^{(n-3)}E_{i+2}^{(n+2)}}\underline{\mathcal{L}^{(n-2)}}\underline{\mathcal{L}^{(n-1)}}\underline{\mathcal{D}}\underline{\mathcal{U}^{(n-1)}}\dots \mathcal{U}^{(2)}\mathcal{U}^{(1)} \\ = \dots \quad (18)$$

$$= L^{(1)}\dots L^{(i-2)}\underline{L^{(i-1)}E_n^{(2n-i)}}\underline{\mathcal{L}^{(i+1)}}\dots \underline{\mathcal{L}^{(n-2)}}\underline{\mathcal{L}^{(n-1)}}\underline{\mathcal{D}}\underline{\mathcal{U}^{(n-1)}}\dots \mathcal{U}^{(2)}\mathcal{U}^{(1)}$$

$$= L^{(1)}\dots L^{(i-2)}\mathcal{L}^{(i-1)}\mathcal{L}^{(i)}\dots \mathcal{L}^{(n-2)}\mathcal{L}^{(n-1)}\mathcal{D}\mathcal{U}^{(n-1)}\dots \mathcal{U}^{(2)}\mathcal{U}^{(1)}. \quad (18)$$

The factors that are transformed at each step of the above transformations are underlined. The matrices $J_m^{(k)}$ and $E_i^{(k)}$ equal $J_m(x_k, y_k, z_k)$ and $E_i(x_k, y_k)$, respectively, for some x_k, y_k , and z_k . The matrices $\mathcal{L}^{(k)}, \mathcal{D}$, and $\mathcal{U}^{(k)}$, $k = 1, 2, \dots, n - 1$, are unit lower bidiagonal, diagonal, and unit upper diagonal, respectively, with the same pattern as $L^{(k)}$, D , and $U^{(k)}$.

We now explain how to obtain (14) from (13) and so on until we obtain (18).

4.1 Passing through the upper bidiagonal factors

We start with the transformations (13)–(15). At each step we are given matrices

$$U = \begin{bmatrix} d_1 & u_1 \\ & d_2 & u_2 \\ & & \ddots & \ddots \\ & & & d_{n-1} & u_{n-1} \\ & & & & d_n \end{bmatrix}, \quad u_j \geq 0, d_j \in \{0, 1\}, \quad j = 1, 2, \dots, n-1; \quad d_n = 1,$$

and $J_i(x, y, z)$, $x \geq 0, y \geq 0, z \geq 0$ with $z > 0$ if $i < n$. We need to compute matrices

$$\mathcal{U} = \begin{bmatrix} d'_1 & u'_1 \\ & d'_2 & u'_2 \\ & & \ddots & \ddots \\ & & & d'_{n-1} & u'_{n-1} \\ & & & & d'_n \end{bmatrix}, \quad u'_j \geq 0, d'_j \in \{0, 1\}, \quad j = 1, 2, \dots, n-1, \quad d'_n = 1,$$

and $J_i(x', y', z')$ such that $J_i(x', y', z') \cdot \mathcal{U} = U \cdot J_i(x, y, z)$, $x' \geq 0, y' \geq 0, z' \geq 0$, and $z' > 0$ if $i < n$. Namely,

$$\begin{bmatrix} \ddots & \ddots & & \\ & d'_{i-2} & u'_{i-2} & \\ & d'_{i-1}y' & y'u'_{i-1} & \\ & d'_{i-1}x' & d'_iz' + x'u'_{i-1} & u'_iz' \\ & & & \ddots \end{bmatrix} = \begin{bmatrix} \ddots & \ddots & & \\ & d_{i-2} & u_{i-2}y & \\ & d_{i-1}y + xu_{i-1} & u_{i-1}z & \\ & d_ix & d_iz & u_i \\ & & & \ddots \end{bmatrix}. \quad (19)$$

Comparing entries we get automatically $u'_{i-2} = u_{i-2}y$. For the rest of the entries we need to consider several cases as we, essentially, perform implicit Neville elimination on the matrix on the right of (19) to eliminate the entry d_ix .

- If $d_{i-1}y + xu_{i-1} \neq 0$, then

$$\begin{aligned} d'_{i-1} &= 1 \\ y' &= d_{i-1}y + xu_{i-1} \\ u'_{i-1} &= u_{i-1}z/y' \\ x' &= d_ix \end{aligned}$$

We have $d'_i z' = d_i z - x'u'_{i-1} = d_i z - \frac{d_i x u_{i-1} z}{d_{i-1} y + x u_{i-1}} = \frac{d_{i-1} d_i y z}{y'}$.

- If $d_{i-1} d_i y z \neq 0$, then $d'_i = 1$ and $z' = \frac{d_{i-1} d_i y z}{y'}$.
- If $d_{i-1} d_i y z = 0$, then $d'_i = 0$, $z' = 1$.
- If $d_{i-1} y + x u_{i-1} = 0$, then
 - If $d_i x = 0$, then $u'_{i-1} = u_{i-1} z$, $x' = 0$, and
 - if $u'_{i-1} \neq 0$, then $d'_{i-1} = 0$ and $y' = 1$;
 - otherwise, $d'_{i-1} = 1$ and $y' = 0$.
 - If $d_i x \neq 0$, then we must have $u_{i-1} z = 0$ (since $U \cdot J_i(x, y, z)$ is TN).¹ Thus $d'_{i-1} = 1$, $y' = 0$, $u'_{i-1} = 0$, and $x' = d_i x$.

In either case

- if $d_i = 1$, then $d'_i = 1$, $z' = z$;
- otherwise (if $d_i = 0$), then $d'_i = 0$, $z' = 1$.

Since the bottom right entry d_n of each bidiagonal factor U must equal 1, if $i = n$ and the above procedure results in $d'_n = 0$, $z' = 1$, we write instead $d'_n = 1$, $z' = 0$. We can do this for $i = n$, since in this case the entries d'_n and z' participate only in the (n, n) entry, $d'_n z' + x' u'_{n-1}$, on the left hand side of (19), and do so as a product.

Finally, $u'_i = u_i/z'$. This is the reason we need to have z strictly positive for $i < n$. For $i = n$ we allow $z = 0$ since then the operation $u'_i = u_i/z'$ won't have to be performed (the indices in the u 's only go to $n - 1$). Clearly, for $i < n$, $z > 0$ implies $z' > 0$.

The rest of the entries remain unchanged: $d'_j = d_j$ for $j \notin \{i-1, i\}$ and $u'_j = u_j$ for $j \notin \{i-2, i-1, i\}$.

4.2 Passing through the diagonal factor

Next, we turn our attention to the transformation (15)–(16).

- If $y d_{i-1} > 0$,

$$D \cdot J_i(x, y, z) = \begin{bmatrix} \ddots & & \\ & 1 & \\ & y d_{i-1} & \\ & x d_i & z d_i \\ & & \ddots \end{bmatrix} = E_i(x', y') \cdot \text{diag}(d_1, d_2, \dots, d_{i-2}, d_{i-1} y, d_i z, d_{i+1}, \dots, d_n),$$

where $x' = x d_i / (y d_{i-1})$, $y' = 1$.

¹ This is a situation where we need to “move” only the entry $d_i x$ to row $i - 1$ as in example (4). We can’t swap the entire zero row $i - 1$ with row i since a nonzero entry u_i would destroy the tridiagonal structure of the product $U \cdot J_i(x, y, z)$ on the right hand side of (19). Thus we move just the entry $d_i x$ one row up, leaving everything else in its place.

– Otherwise (if $y d_{i-1} = 0$), then there are two possibilities:

– If $x d_i > 0$, then

$$D \cdot J_i(x, y, z) = E_i(x', y') \cdot \text{diag}(d_1, d_2, \dots, d_{i-2}, 1, d_i z, d_{i+1}, \dots, d_n),$$

where $x' = x d_i$, $y' = 0$.

– Otherwise (i.e., if $x d_i = 0$),

$$D \cdot J_i(x, y, z) = \text{diag}(d_1, d_2, \dots, d_{i-2}, 0, d_i z, d_{i+1}, \dots, d_n)$$

and the procedure terminates (i.e., $x' = 0$, $y' = 1$, so that $E_i(x', y') = I$).

Either way, we have factored out an $E_i(x', y')$, where $x' \geq 0$ and $y' \in \{0, 1\}$.

4.3 Passing through the lower bidiagonal factors

The final stage are the transformations (16)–(18), where we start with the matrix E_i from the previous step. Each transformation is $E_{i+1}(x', y') \cdot \mathcal{L} = L \cdot E_i(x, y)$. Namely,

$$\begin{bmatrix} \ddots & & & \\ & d'_{i-1} & & \\ & y' b'_i & y' d'_i & \\ & x' b'_i & x' d'_i + b'_{i+1} & d'_{i+1} \\ & & & \ddots \end{bmatrix} = \begin{bmatrix} \ddots & & & \\ & d_{i-1} y & & \\ & b_i y + d_i x & d_i & \\ & b_{i+1} x & b_{i+1} & d_{i+1} \\ & & & \ddots \end{bmatrix}.$$

Comparing entries, we get $d'_{i-1} = d_{i-1} y$ and

– If $b_{i+1} x = 0$, then $b'_i = b_i y + d_i x$ and we are done ($E_{i+1}(x', y') = I$).
– Otherwise, we have the following two possibilities:

– If $b_i y + d_i x \neq 0$, then we set $y' = 1$ and obtain

$$\begin{aligned} b'_i &= b_i y + d_i x \\ x' &= \frac{b_{i+1} x}{b'_i} \\ d'_i &= d_i \\ b'_{i+1} &= b_{i+1} - x' d'_i = b_i b_{i+1} y / b'_i. \end{aligned}$$

– Otherwise (i.e., $b_i y + d_i x = 0$), since $b_{i+1} x \neq 0$, we must have $d_i = 0$ and thus we factor an $E_{i+1}(x, 0)$ from $L \cdot J(x, y, 1)$ as follows:

$$\begin{aligned}
L \cdot J(x, y, 1) &= \begin{bmatrix} \ddots & & \\ & d_{i-1}y & \\ & 0 & 0 \\ & b_{i+1}x & b_{i+1} & d_{i+1} \\ & & & \ddots \end{bmatrix} \\
&= \begin{bmatrix} \ddots & & \\ & 1 & & \\ & 0 & & \\ & x & 1 & \\ & & & \ddots \end{bmatrix} \begin{bmatrix} \ddots & & & \\ & d_{i-1}y & & \\ & b_{i+1} & 0 & \\ & b_{i+1} & d_{i+1} & \\ & & & \ddots \end{bmatrix}.
\end{aligned}$$

Thus $y' = 0$, $x' = x$, $b'_i = b_{i+1}$, and $d'_{i+1} = d_{i+1}$. Once again, just as in (4), we used the (singular) matrix $E_i(x, 0)$ to “eliminate” the entry $b_{i+1}x$ in position $(i+1, i-1)$.

4.4 Scaling a row or a column by a scalar d

Being able to multiply by an EBM $J_i(x, y, z)$ directly allows us to easily scale a TN matrix by any diagonal factor. To scale a single column, say the i th, by a scalar $d \geq 0$, we form the product with $J_{i+1}(0, d, 1)$ if $i < n$ or with $J_n(0, 1, d)$ if $i = n$. This distinction is needed because of the requirement that $z > 0$ if $i < n$ in the definition (1) of J_i .

In particular, we can set any column to 0 by picking $d = 0$. This is of particular importance in our eigenvalue algorithm, where we have to set a column to 0 whenever we encounter a zero row and vice versa.

To scale a row, we work on the transpose of a matrix and its corresponding bidiagonal decomposition (11).

5 Multiplication by an EBM on the left

Since we already know how to scale rows and columns, it suffices to show how to multiply by the EBM $E_i(b, c)$ on the left (since $J_i(x, y, z)$ is a product of an $E_i(b, c)$ and a diagonal factor).

Let A be TN with bidiagonal decomposition

$$A = L^{(1)} L^{(2)} \dots L^{(m-1)} D U^{(n-1)} U^{(n-2)} \dots U^{(1)}.$$

In what follows we show how to compute $\mathcal{BD}(E_i(b_1, d_1) \cdot A)$ given b_1 , d_1 , and $\mathcal{BD}(A)$ without performing any subtractions.

The relations (5) imply that $E_i(b_1, d_1)$ commutes with $L^{(1)}, L^{(2)}, \dots, L^{(n-i-1)}$, thus

$$\begin{aligned} & E_i(b_1, d_1) \cdot A \\ &= L^{(1)} L^{(2)} \cdots L^{(n-i-1)} E_i(b_1, d_1) L^{(n-i)} \cdots L^{(n-1)} D U^{(n-1)} U^{(n-2)} \cdots U^{(1)}. \end{aligned}$$

The goal is thus to obtain two new bidiagonal matrices $\mathcal{L}^{(n-i)}$ and $\mathcal{L}^{(n-i+1)}$ with the same structure as $L^{(n-i)}$ and $L^{(n-i+1)}$ such that

$$\mathcal{L}^{(n-i)} \mathcal{L}^{(n-i+1)} = E_i(b_1, d_1) L^{(n-i)} L^{(n-i+1)}. \quad (20)$$

We only demonstrate the case $i = 2$ since the cases $i > 2$ are completely analogous.

Let

$$L^{(n-2)} = \begin{bmatrix} 1 & & & & \\ 0 & d_2 & & & \\ b_2 & d_3 & & & \\ \ddots & \ddots & \ddots & & \\ & b_{n-1} & d_n & & \end{bmatrix}, \quad L^{(n-1)} = \begin{bmatrix} e_1 & & & & \\ c_1 & e_2 & & & \\ c_2 & e_3 & & & \\ \ddots & \ddots & \ddots & & \\ & c_{n-1} & e_n & & \end{bmatrix}$$

and

$$\mathcal{L}^{(n-2)} = \begin{bmatrix} 1 & & & & \\ 0 & d'_2 & & & \\ b'_2 & d'_3 & & & \\ \ddots & \ddots & \ddots & & \\ & b'_{n-1} & d'_n & & \end{bmatrix}, \quad \mathcal{L}^{(n-1)} = \begin{bmatrix} e'_1 & & & & \\ c'_1 & e'_2 & & & \\ c'_2 & e'_3 & & & \\ \ddots & \ddots & \ddots & & \\ & c'_{n-1} & e'_n & & \end{bmatrix}.$$

Then

$$E_2(b_1, d_1) \cdot L^{(n-2)} = \begin{bmatrix} d_1 & & & & \\ b_1 & d_2 & & & \\ b_2 & d_3 & & & \\ \ddots & \ddots & \ddots & & \\ & b_{n-1} & d_n & & \end{bmatrix}$$

and (20) becomes

$$\begin{aligned}
 & \left[\begin{array}{cccccc} e'_1 & & & & & \\ d'_2 c'_1 & d'_2 e'_2 & & & & \\ b'_2 c'_1 & b'_2 e'_2 + d'_3 c'_2 & d'_3 e'_3 & & & \\ \ddots & \ddots & \ddots & \ddots & & \\ & b'_{n-1} c'_{n-2} & b'_{n-1} e'_{n-1} + d'_n c'_{n-1} & d'_n e'_n & & \end{array} \right] \\
 & = \left[\begin{array}{cccccc} d_1 e_1 & & & & & \\ b_1 e_1 + c_1 d_2 & d_2 e_2 & & & & \\ b_2 c_1 & b_2 e_2 + d_3 c_2 & d_3 e_3 & & & \\ \ddots & \ddots & \ddots & \ddots & & \\ & b_{n-1} c_{n-2} & b_{n-1} e_{n-1} + d_n c_{n-1} & d_n e_n & & \end{array} \right]. \quad (21)
 \end{aligned}$$

We compare entries on both sides of (21) to obtain (since $b'_1 = 0, d'_1 = 1$)

$$\begin{aligned}
 e'_i &= \frac{e_i d_i}{d'_i} \\
 b'_i &= \frac{b_i c_{i-1}}{c'_{i-1}}, \\
 d'_{i+1} c'_i &= b_i e_i - b'_i e'_i + d_{i+1} c_i
 \end{aligned} \quad (22)$$

for $i = 1, 2, \dots$. We eliminate the subtraction in (22) by introducing auxiliary variables $g_i \equiv b_i e_i - b'_i e'_i$. Then $g_1 = e_1 b_1 - b'_1 e'_1 = e_1 b_1$ and

$$\begin{aligned}
 g_i &= b_i e_i - b'_i e'_i \\
 &= b_i e_i - \frac{b_i c_{i-1} e_i d_i}{c'_{i-1} d'_i} \\
 &= \frac{b_i e_i}{c'_{i-1} d'_i} (c'_{i-1} d'_i - c_{i-1} d_i) \\
 &= \frac{b_i e_i}{c'_{i-1} d'_i} (b_{i-1} e_{i-1} - b'_{i-1} c'_{i-1}) \\
 &= \frac{b_i e_i g_{i-1}}{c'_{i-1} d'_i}.
 \end{aligned}$$

Therefore we set $d'_1 = 1$ and iterate for $i = 1, 2, \dots$

– If $i = 1$ or $d'_i c'_{i-1} \neq 0$,

$$\begin{aligned}
 g_i &= \begin{cases} \frac{b_i e_i g_{i-1}}{c'_{i-1} d'_i}, & \text{if } i > 1 \\ b_1 e_1, & \text{otherwise} \end{cases} \\
 b'_i &= \begin{cases} \frac{b_i c_{i-1}}{c'_{i-1}}, & \text{if } i > 1 \\ 0, & \text{otherwise} \end{cases}
 \end{aligned}$$

$$\begin{aligned} e'_i &= \frac{e_i d_i}{d'_i} \\ d'_{i+1} c'_i &= d_{i+1} c_i + g_i. \end{aligned} \tag{23}$$

We resolve the last equation, (23), after the next case since we end up with the same one there as well.

– If $d'_i c'_{i-1} = 0$, then

- if $d'_i = 1$ we set $b'_i = 0$, $e'_i = e_i d_i$
- if $d'_i = 0$ we set $b'_i = 1$, $e'_i = 0$.

Either way, $g_i = b_i e_i - b'_i e'_i = b_i e_i$ and $d'_{i+1} c'_i = d_{i+1} c_i + g_i$.

In both cases above, we end up having to resolve the equation $d'_{i+1} c'_i = d_{i+1} c_i + g_i$ for d'_{i+1} and c'_i .

Since $d'_{i+1} \in \{0, 1\}$, if $d_{i+1} c_i + g_i \neq 0$, we must have $d'_{i+1} = 1$ and $c'_i = d_{i+1} c_i + g_i$.

If $d_{i+1} c_i + g_i = 0$ and $b_{i+1} c_i \neq 0$, we must have $c'_i \neq 0$ (since $b'_{i+1} c'_i = b_{i+1} c_i$), which means $d_{i+1} = 0$ and we can pick c'_i to be an arbitrary positive number. Choosing $c'_i = b_{i+1} c_i$ would imply $b'_{i+1} = 1$ in the following step, which means that we, essentially, eliminate the entry $b_{i+1} c_i > 0$ in position $(i+2, i)$ in (21) using a trasformation of type (4).

Finally, if $d_{i+1} c_i + g_i = 0$ and $b_{i+1} c_i = 0$, we choose $d'_{i+1} = 1$, $c'_i = 0$. This choice is in line with our goal of having ones on the diagonal and zeros on the subdiagonal of the L factors in $\mathcal{BD}(A)$ in the eigenvalue computation.

6 Product of TN matrices

Since a TN matrix is a product of EBM s, (8), we can compute the bidiagonal decomposition of a product of two TN matrices A and B by starting with $\mathcal{BD}(A)$ and using the methods of Sects. 4 and 5 to accumulate all the EBM factors of $\mathcal{BD}(B)$.

This approach also allows us to recover the decomposition (10) of a TN matrix A represented as *any* product of nonnegative bidiagonals. Since every nonnegative bidiagonal is a products of EBM s, we can start with the identity matrix (which is TN) and consecutively multiply by all EBM s that comprise A .

As we mentioned previously, this is the reason why all representations of a TN matrix as products of nonnegative bidiagonals are numerically equivalent.

The ability to compute the bidiagonal decomposition of a product of TN matrices will also be needed in the computation of the zero Jordan blocks in Sect. 10 below, where we will need the bidiagonal decompositions of the powers of the TN matrix.

7 Swapping a zero row or column with the following or the previous one

The ability to swap zero rows or zero columns with the following or the previous ones will be needed in the eigenvalue computation in Sect. 8 and the computation of the rank of a TN matrix in Sect. 9.

We can swap a zero column, i , in a TN matrix with the *following* column, $i + 1$, by forming the product $A \cdot E_{i+1}(1, 0)$ and then setting the $(i + 1)$ st column to 0. In other words, we add column $i + 1$ to (zero times) the zero column i , then set column $i + 1$ to zero, as for example in the following 3×3 TN matrix for $i = 2$:

$$\begin{bmatrix} 1 & 0 & 1 \\ 1 & 0 & 2 \\ 1 & 0 & 3 \end{bmatrix} \begin{bmatrix} 1 & 0 & \\ & 1 & 1 \end{bmatrix} \begin{bmatrix} 1 & 1 & 0 \\ & 1 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 2 & 0 \\ 1 & 3 & 0 \end{bmatrix}.$$

Both multiplications are handled as in Sect. 4.

We can swap a zero row, i , with the *previous* one by forming $E_i(1, 0) \cdot A$ as in the following example:

$$\begin{bmatrix} 0 & & \\ 1 & 1 & \\ & & 1 \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ 1 & 2 & 3 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 \\ 1 & 1 & 1 \\ 1 & 2 & 3 \end{bmatrix}.$$

8 Cryer's method

In [3] Cryer described an algorithm which reduces a TN matrix A to a symmetric tridiagonal matrix T with the same eigenvalues. It works by creating zeros in A below the first subdiagonal and above the first superdiagonal starting with the first column then proceeding with the first row and so on as follows.

If the entry $a_{ij} > 0$ (say below the diagonal) is to be zeroed out, then there are two possibilities:

1. If $a_{i-1,j} > 0$, we subtract an appropriate multiple of row $i - 1$ from row i to create a zero in position (i, j) . We complete the similarity by adding the same multiple of column i to column $i - 1$. This does not affect the zeros created earlier in this process and preserves the TN structure of A .
2. If $a_{i-1,j} = 0$, this means that the entire $(i - 1)$ st row of A is zero (since $a_{ij} > 0$). Column $i - 1$ can thus be set to zero without changing the eigenvalues. We then swap the $(i - 1)$ st row and the $(i - 1)$ st column with the i th ones, respectively.

We continue this process until we are left with a tridiagonal matrix.

In the language of $\mathcal{BD}(A)$ this algorithm works as follows. We need to eliminate all EBMs in (8) that do not belong to $L^{(n-1)}$, D , or $U^{(n-1)}$. Here “eliminate” means turn all those EBMs into identity matrices.

We start on the leftmost EBM in (8), then the rightmost, and alternate until we are left with just $L^{(n-1)}$, D , and $U^{(n-1)}$.

We explain how this process works to eliminate the first factor, $E_n(b_{n1}, c_{n1})$, in (8) with the rest of the process being analogous.

We have, $A = E_n(b_{n1}, c_{n1}) \cdot A'$, where $\mathcal{BD}(A')$ is obtained from $\mathcal{BD}(A)$ by setting $b_{n1} = 0$, $c_{n1} = 1$.

1. If $c_{n1} = 1$, we form the similarity

$$\begin{aligned} E_n(-b_{n1}, 1) \cdot A \cdot (E_n(-b_{n1}, 1))^{-1} &= E_n(-b_{n1}, 1) \cdot E_n(b_{n1}, 1) \cdot A' \cdot E_n(b_{n1}, 1) \\ &= A' \cdot E_n(b_{n1}, 1). \end{aligned}$$

The decomposition $\mathcal{BD}(A' \cdot E_n(b_{n1}, 1))$ is computed using the method of Sect. 4.

2. If $c_{n1} = 0$, this means that row $n - 1$ of A is 0. We proceed by setting column $n - 1$ to zero using the method of Sect. 4.4 which, despite not being a similarity, does not change the eigenvalues of A . We then swap rows $n - 1$ and n as well as columns $n - 1$ and n as in Sect. 7.

The critical observation here is that in both situations the bidiagonal decomposition of the newly obtained matrix is such that $b_{n1} = 0$, $c_{n1} = 1$, i.e., the factor $E_n(b_{n1}, c_{n1})$ has been eliminated—it now equals the identity matrix. This is obvious in the case $c_{n1} = 1$. When $c_{n1} = 0$, swapping rows $n - 1$ and n is achieved by forming the product

$$J_n^T(0, 1, 0) \cdot A = J_n^T(0, 1, 0) \cdot E_n(b_{n1}, 0) \cdot A' = J_n^T(1, b_{n1}, 0) \cdot A'.$$

We already have $b_{n1} = 0$, $c_{n1} = 1$ in $\mathcal{BD}(A')$, thus computing $\mathcal{BD}(J_n^T(1, b_{n1}, 0) \cdot A')$ as the transpose of $\mathcal{BD}((A')^T \cdot J_n(1, b_{n1}, 0))$ using the method of Sect. 4 does not affect the values of b_{n1} and c_{n1} .

At the end of this procedure we are left the bidiagonal decomposition of a tridiagonal matrix T with the same eigenvalues as A :

$$T = L^{(n-1)} D U^{(n-1)}. \quad (24)$$

Replacing the offdiagonal entries $t_{i,i+1}$ and $t_{i+1,i}$ of T by $\sqrt{t_{i,i+1} t_{i+1,i}}$ does not change its eigenvalues, but makes it symmetric. The matrix T is also nonnegative definite since all of its minors and, in particular, the principal ones are nonnegative. Its Cholesky factor is readily available from (24).

We compute the eigenvalues of T as the squares of the singular values of its Cholesky factor to high relative accuracy using the result of Demmel and Kahan [5].

9 The rank of a TN matrix

The rank of a TN matrix A can be recovered exactly from $\mathcal{BD}(A)$ by first reducing A to upper bidiagonal form as follows.

We consider the decomposition (8) and start eliminating the EBMs corresponding to the first column in B , $E_n(b_{n1}, c_{n1}), \dots, E_2(b_{21}, c_{21})$ (in that order). By “eliminating,” again, we mean turning these factors into identity matrices. The process does not change the rank of A .

If the current leftmost factor, say, $E_i(b_{i1}, c_{i1})$ is such that $c_{i1} = 1$, then this factor is nonsingular and replacing it with the identity matrix by setting $b_{i1} = 0$ does not change the rank of A .

If, in that factor, $c_{i1} = 0$, then the $(i - 1)$ st row of A is zero and we swap it with row i as in Sect. 7. The rank of A is unaffected, but again the factor $E_i(b_{i1}, c_{i1})$ is eliminated (for the same reason explained in Sect. 8.)

We then turn our attention to the first row of B . We eliminate only the last $n - 2$ rightmost factors $E_3^T(b_{13}, c_{13}), \dots, E_n^T(b_{1n}, c_{1n})$, starting with the last one in the same fashion as we did for the first column of B . We leave $E_2^T(b_{12}, c_{12})$ intact since attempting to eliminate it when $c_{12} = 0$ could result in some of the factors $E_n(b_{n1}, c_{n1}), \dots, E_2(b_{21}, c_{21})$ already eliminated in the previous step reappearing in the bidiagonal decomposition.

We proceed with the second row and column, etc., until we are left with an upper bidiagonal matrix, E , with bidiagonal decomposition $E = D \cdot U^{(n-1)}$. Computing its rank is now a trivial matter.

10 The zero Jordan blocks

The number of zero Jordan blocks of any matrix A is equal to $n - \text{rank}(A)$. For a TN matrix, we readily compute this number off $\mathcal{BD}(A)$ using the method from the previous section.

If we have more than one zero Jordan block, we form the sequence

$$\text{rank}(A), \text{rank}(A^2), \dots \quad (25)$$

by first computing $\mathcal{BD}(A^2), \mathcal{BD}(A^3)$, etc., and then the corresponding ranks. The difference

$$z_i \equiv \text{rank}(A^i) - \text{rank}(A^{i+1})$$

equals the number of zero Jordan blocks of size at least i . The largest i for which we need to compute z_i is thus the first one for which z_i becomes 1. Once that happens, the sizes of all zero Jordan blocks are directly determined. Since the computation of the rank is exact, the sizes of the Jordan blocks are revealed exactly. The maximum power of A that needs to be computed in the sequence (25) equals the size of the second largest zero Jordan block.

11 The cost of computing the eigenvalues and the zero Jordan blocks

The introduction of the new variables on the diagonal of the bidiagonal factors in $\mathcal{BD}(A)$ does add a tiny additional arithmetic cost to the algorithms in Sects. 4 and 5 compared to those in [12,13]. A careful examination of the new formulas readily reveals that the cost of our new eigenvalue algorithm in this paper does not exceed twice the cost of the eigenvalue algorithm in [12]. Therefore our new algorithm costs $O(n^3)$.

The cost of computing the zero Jordan blocks is $O(n^3 z)$, where z is the size of the second largest zero Jordan block. Since the second largest Jordan block can potentially

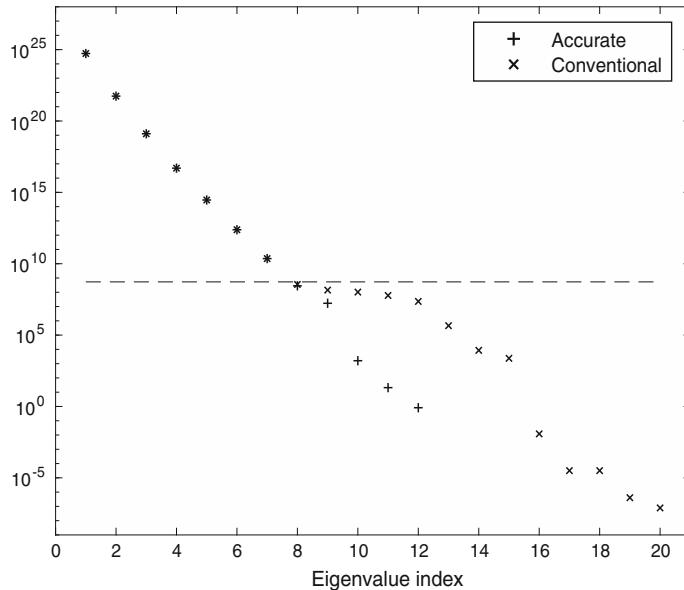


Fig. 1 The eigenvalues of a 20×20 Vandermonde matrix

be of size $O(n)$ (e.g., $\frac{n}{3}$), we have an upper bound of $O(n^4)$. Finding an algorithm to compute the zero Jordan blocks in $O(n^3)$ time remains an open problem.

12 Perturbation theory and accuracy of the algorithms

Following the same arguments as in [12] we can readily state that small relative perturbations in the entries of $\mathcal{BD}(A)$ cause small relative perturbations in the eigenvalues of A . The zero Jordan structure is unaffected by relative perturbations in $\mathcal{BD}(A)$ of any magnitude [7].

Additionally, the error bounds for the eigenvalues, proven in [12], apply directly to the singular case as well—the only difference here is the possibility for zeros on the diagonals of the factors $L^{(i)}$ and $U^{(i)}$ of (10), which does not yield any additional roundoff errors.

In particular, all eigenvalues of a TN matrix are computed to high relative accuracy. The zero eigenvalues and their corresponding Jordan blocks are revealed exactly.

13 Numerical experiments

We performed extensive numerical tests to verify the correctness of our eigenvalue algorithm. We present one illustrative example in Fig. 1 of the (singular) 20×20 Vandermonde matrix with nodes

$$1, 2, 2, 2, 2, 6, 7, 7, 7, 7, 7, 7, 13, 14, 15, 16, 17, 18, 19, 20.$$

Since the nodes 2 and 7 are repeated 4 and 6 times, respectively, the rank of the matrix is 12. We computed the eigenvalues using our new algorithm, a conventional algorithm (`eig` in MATLAB [14]), and in 60 decimal digit extended precision arithmetic. As expected, only the largest eigenvalues are computed accurately by `eig` and the zero eigenvalues are lost. In contrast, our new algorithm computed all nonzero eigenvalues to at least 14 correct decimal digits when compared with those computed in extended precision. No amount of extra precision will allow us to reliably produce the zero eigenvalues exactly using conventional algorithms, thus the only reason we know that our algorithm returned the correct number of zero eigenvalues (8) is because we know from theory that the rank of the matrix is 12.² This matrix has no nontrivial zero Jordan blocks.

We also tested the computation of zero Jordan blocks against known examples and found the output of our algorithm to be an exact match.

We conclude with a small 4×4 example,

$$A = \begin{bmatrix} 3 & 3 & 2 & 1 \\ 2 & 2 & 3 & 2 \\ 1 & 1 & 2 & 3 \\ 1 & 1 & 2 & 3 \end{bmatrix}, \quad \mathcal{BD}(A) = \begin{Bmatrix} 3 & 1 & \frac{2}{3} & 0.5 \\ \frac{2}{3} & \frac{5}{3} & 1 & 0.3 \\ 0.5 & 0.3 & 1.6 & 1 \\ 1 & 0 & 0 & 0 \end{Bmatrix}, \quad \begin{Bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 \end{Bmatrix},$$

for which we computed $\mathcal{BD}(A)$ exactly by hand. The entries $\frac{2}{3}$, $\frac{5}{3}$, 0.3, and 1.6 of $\mathcal{BD}(A)$ are not exact binary floating point numbers, which means that the TN matrix represented by the stored binary floating point version of $\mathcal{BD}(A)$ has nonzero eigenvalues which are slightly perturbed versions of the nonzero eigenvalues of A (the zero eigenvalues are unaffected by small relative perturbations in $\mathcal{BD}(A)$). Despite those perturbations and the further roundoff errors in the eigenvalue computation, the eigenvalues are computed by our algorithm as

$$\{7.828427124746190, 2.171572875253810, 0, 0\},$$

matching the exact nonzero eigenvalues, $5 \pm 2\sqrt{2}$, to 16 decimal digits. The zero ones are computed exactly and the zero Jordan block of size 2 is also correctly identified.

MATLAB implementations of the algorithms described here are available online [11].

14 Open problems

Several open problems remain. The most troubling is the potential $O(n^4)$ cost to reveal the zero Jordan blocks, which we would have ideally liked to be $O(n^3)$. Other open problems include extending the techniques of this paper to the SVD and other TN-preserving operations as well as finding explicit formulas for the bidiagonal decompositions of the classical TN matrices Vandermonde, Cauchy, etc., when they have repeated nodes and are thus singular.

² The zero eigenvalues are not depicted in the figure since it is log-scale.

Acknowledgements I thank James Demmel for suggesting the problem of accurate computations with TN matrices and for conjecturing correctly in [4] that accurate computations with them ought to be possible. I am also thankful for the accommodations during my sabbatical at the University of California, Berkeley, when a part of this research was conducted. I also thank the referees for the careful reading of the manuscript and for their suggestions, which improved the presentation of the material. This work was partially supported by the Woodward Fund for Applied Mathematics at San Jose State University. The Woodward Fund is a gift from the estate of Mrs. Marie Woodward in memory of her son, Henry Teynham Woodward. He was an alumnus of the Mathematics Department at San Jose State University and worked with research groups at NASA Ames.

References

1. Anderson, E., Bai, Z., Bischof, C., Blackford, S., Demmel, J., Dongarra, J., Du Croz, J., Greenbaum, A., Hammarling, S., McKenney, A., Sorensen, D.: LAPACK Users' Guide. Software Environ. Tools 9, 3rd edn. SIAM, Philadelphia (1999)
2. Ando, T.: Totally positive matrices. *Linear Algebra Appl.* **90**, 165–219 (1987)
3. Cryer, C.: Some properties of totally positive matrices. *Linear Algebra Appl.* **15**, 1–25 (1976)
4. Demmel, J.: Accurate singular value decompositions of structured matrices. *SIAM J. Matrix Anal. Appl.* **21**(2), 562–580 (1999)
5. Demmel, J., Kahan, W.: Accurate singular values of bidiagonal matrices. *SIAM J. Sci. Stat. Comput.* **11**(5), 873–912 (1990)
6. Fallat, S.M.: Bidiagonal factorizations of totally nonnegative matrices. *Am. Math. Mon.* **108**(8), 697–712 (2001)
7. Fallat, S.M., Gekhtman, M.I.: Jordan structures of totally nonnegative matrices. *Can. J. Math.* **57**(1), 82–98 (2005)
8. Fallat, S.M., Johnson, C.R.: *Totally Nonnegative Matrices*. Princeton Series in Applied Mathematics. Princeton University Press, Princeton (2011)
9. Gasca, M., Peña, J.M.: Total positivity and Neville elimination. *Linear Algebra Appl.* **165**, 25–44 (1992)
10. Karlin, S.: *Total Positivity*, vol. I. Stanford University Press, Stanford (1968)
11. Koev, P.: <http://www.math.sjsu.edu/~koev>. Accessed 7 Jan 2019
12. Koev, P.: Accurate eigenvalues and SVDs of totally nonnegative matrices. *SIAM J. Matrix Anal. Appl.* **27**(1), 1–23 (2005)
13. Koev, P.: Accurate computations with totally nonnegative matrices. *SIAM J. Matrix Anal. Appl.* **29**, 731–751 (2007)
14. The MathWorks Inc: MATLAB Reference Guide. The MathWorks Inc, Natick (1992)
15. Whitney, A.M.: A reduction theorem for totally positive matrices. *J. Anal. Math.* **2**, 88–92 (1952)

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.