Journal of Computational and Applied Mathematics 252 (2013) 3–11

Contents lists available at SciVerse ScienceDirect

Journal of Computational and Applied Mathematics

journal homepage: www.elsevier.com/locate/cam

Non-symbolic algorithms for the inversion of tridiagonal matrices

J. Abderramán Marrero [a], M. Rachidi [b], V. Tomeo [c,*]

[a] Department of Mathematics Applied to Information Technologies, (ETSIT-UPM) Telecommunication Engineering School, Technical University of Madrid Avda Complutense s/n. Ciudad Universitaria, 28040 Madrid, Spain

[b] Group DEFA - Department of Mathematics and Informatics, Faculty of Sciences, University Moulay Ismail, B.P. 4010, Beni M'hamed, Meknes, Morocco

[c] Department of Algebra, (EUE-UCM) School of Statistics, University Complutense, Avda de Puerta de Hierro s/n. Ciudad Universitaria, 28040 Madrid, Spain

ARTICLE INFO

Article history:
Received 20 September 2011
Received in revised form 8 February 2012

MSC:
15A09
15A15
15A29
39A06
65F05
65Y20

Keywords:
Computational complexity
Difference equation
Inverse matrix
Numerical algorithm
Tridiagonal matrix

ABSTRACT

A representation for the entries of the inverse of general tridiagonal matrices is based on the determinants of their principal submatrices. It enables us to introduce, through the linear recurrence relations satisfied by such determinants, a simple algorithm for the entries of the inverse of any tridiagonal nonsingular matrix, reduced as well as unreduced. The numerical approach is preserved here, without invoking the symbolic computation. For tridiagonal diagonally dominant matrices, a scaling transformation on the recurrences allows us to give another algorithm to avoid overflow and underflow.

1. Introduction

Let $T = \{a_i, b_i, c_i\}$ $(1 \le i \le n)$ be an $n \times n$ tridiagonal nonsingular matrix, with $a_1 = c_n = 0$, where the $\{b_i\}$ are the coefficients of the principal diagonal and the $\{a_i\}$, $\{c_i\}$ are those of the lower and upper subdiagonals, respectively. Algorithms for the inversion of such matrices are frequently used. There are efficient packages for the numerical inversion of matrices based on Gaussian algorithms, with pivoting strategies, and for solving linear systems by using the Neville elimination, see e.g. [1]. But they require a great amount of memory and greater run times than other specific algorithms. Concerning the abundant literature about such simpler algorithms for the inversion of tridiagonal matrices, we can refer to [2–4], for example. In general, these specialized algorithms are applicable only in the case of tridiagonal unreduced matrices. Frequently, reduced matrices have been avoided because if an entry on the subdiagonals is null, then the routine can be applied in separate blocks. Indeed, just consider a scenario of tridiagonal strongly reduced matrices, which have numerous null entries in the subdiagonals. Therefore, complexity of such a method of inversion becomes significant. Numerical techniques have also been applied on linear systems with block tridiagonal matrices, see e.g. [5,6].

A first complete analysis on the inversion of tridiagonal nonsingular matrices, without imposing any condition on the coefficients, was introduced in [7]. Nevertheless, the resulting numerical algorithm breaks down when any (left or right) principal submatrix is singular. The symbolic computation recently established in this subject, see e.g. [8,9], overcomes

* Corresponding author. Tel.: +34 913944055.
E-mail addresses: jc.abderraman@upm.es (J. Abderramán Marrero), mu.rachidi@hotmail.fr (M. Rachidi), tomeo@estad.ucm.es (V. Tomeo).

0377-0427/$ – see front matter © 2012 Elsevier B.V. All rights reserved.
doi:10.1016/j.cam.2012.05.003

difficulties by considering symbolic parameters, which are adequately replaced in a posterior step of the algorithm. The computational complexity of the algorithms given in [7–9] is $O(n^2)$.

In the applied domain, the numerical approach is currently more spread out and usable than the symbolic one. Thus we try to go on with the numerical line from [7], by introducing a simple algorithm to obtain the entries of the inverse of any tridiagonal nonsingular matrix. There are some compact representations for the entries of the inverse of tridiagonal nonsingular matrices, special as well as general, see e.g. [10,11]. We propose a numerical algorithm by taking advantage of the representation based on the determinants of proper principal submatrices, see e.g. [9],

$$(T^{-1})_{ij} = \begin{cases} (-1)^{i+j} \left(\prod_{k=j+1}^{i} a_k \right) \dfrac{\det T_{j-1} \cdot \det T_{n-i}^{(i)}}{\det T} & \text{if } i > j, \\ (-1)^{i+j} \left(\prod_{k=i}^{j-1} c_k \right) \dfrac{\det T_{i-1} \cdot \det T_{n-j}^{(j)}}{\det T} & \text{if } i \leq j. \end{cases} \tag{1}$$

The submatrix T_{i-1} is the left principal one of order $i-1$. The submatrix $T_{n-j}^{(j)}$ is the right principal one of order $n-j$, which begins in the $(j+1)$-th row and column and finishes in the n-th row and column. We define here $\det T_0 = \det T_0^{(n)} = 1$. Representation (1) for the entries of the inverse of tridiagonal nonsingular matrices is a particular case of the closed representation for inverses of nonsingular Hessenberg matrices, see e.g. [12]. We can also obtain Expression (1) by using the companion decomposition, recently introduced in [13], on any tridiagonal nonsingular matrix T.

If in addition T is a symmetric matrix, then its inverse matrix is also a symmetric one, and its entries have the simpler representation,

$$(T^{-1})_{ij} = (T^{-1})_{ji} = (-1)^{M+m} \left(\prod_{k=m+1}^{M} a_k \right) \frac{\det T_{m-1} \cdot \det T_{n-M}^{(M)}}{\det T}, \tag{2}$$

with $M = \max\{i, j\}$, $m = \min\{i, j\}$.

The complexity for the inversion of tridiagonal nonsingular matrices is related to the obtainment of the determinants of all their principal submatrices. For a fast computation we have at our disposal the second order linear difference equations satisfied by such determinants; see also [14]. The linear recurrence relation for determinants of the left principal submatrices, with initial conditions $\det T_1 = b_1$, $\det T_2 = b_2 b_1 - a_2 c_1$, is

$$\det T_{k+2} = b_{k+2} \det T_{k+1} - a_{k+2} c_{k+1} \det T_k, \qquad (1 \leq k \leq n-2)\,. \tag{3}$$

For determinants of the right principal submatrices, the recurrence relation for $1 \leq k \leq n-2$, with initial conditions $\det T_1^{(n-1)} = b_n$, $\det T_2^{(n-2)} = b_{n-1} b_n - c_{n-1} a_n$, is

$$\det T_{k+2}^{(n-k-2)} = b_{n-k-1} \det T_{k+1}^{(n-k-1)} - c_{n-k-1} a_{n-k} \det T_k^{(n-k)}. \tag{4}$$

Just consider as we can directly obtain a particular entry of the inverse with $O(n)$ complexity. Although, overflow or underflow can appear in further computation of such recurrences. Thus, our algorithm works for values of the recurrences into the usage range. For example, in some diagonally dominant matrices, i.e. $|b_i| \geq |a_i| + |c_i|$, the solutions of the recurrences grow (or reduce) quickly in magnitude. Therefore other methods should be introduced, such as scaling transformations on the recurrences. We handle these difficulties by considering another algorithm.

The material of this paper is organized as follows. In Section 2, after analyzing difficulties of some current specialized numerical algorithms for the inversion of tridiagonal matrices, [7,9], we point out the features of the algorithm detailed in Appendix A. This algorithm permits us to compute the inverse of any tridiagonal nonsingular matrix of finite order. As an illustration, graphical comparisons of its run times with respect to the built-in function *inv*() of the *Matlab* ® package are given in Fig. 1. In Section 2.2 we check the algorithm of Appendix A on some current examples of tridiagonal matrices. As it was pointed out previously, some difficulties related to overflow and underflow appear in the inversion of various tridiagonal diagonally dominant matrices. We manage these difficulties by introducing in Section 2.3 scaling transformations in the linear recurrences involved. Therefore, an equivalent recursive algorithm is detailed in Appendix B. It permits us to avoid overflow and underflow. To check the complexity of the proposed algorithm from Appendix B with respect to those given in [7,9], a graphical comparison for the mean elapsed time in the inversion of some diagonally dominant matrices is finally provided.

2. Inversion of general tridiagonal matrices

2.1. Algorithms of inversion in the general case

An advance on specialized numerical algorithms for the inversion of general tridiagonal matrices was provided in [7], beyond the classical method using four vectors on unreduced matrices; see e.g. [9, Section 4.1]. This algorithm permits us to

cover the strictly nonsingular matrices, [13]; i.e. tridiagonal nonsingular matrices without singular principal submatrices. The new difficulty was the location in denominators of the involved formulae of null determinants of some principal submatrices.

We analyze this difficulty using an equivalent notation based on determinants. Thus, in [7] two vectors, $\vec{z}$, $\vec{y}$ with entries $z_i = \det T_i$, $y_j = \det T_{n-j+1}^{(j-1)}$, were introduced. There is no difficulty with the principal submatrices, but overflow and underflow should be considered.

The diagonal entries of the inverse matrix, $\phi_{j,j}$, were computed as

$$\phi_{j,j} = \frac{1}{b_j - a_j c_{j-1} \frac{z_{j-2}}{z_{j-1}} - a_{j+1} c_j \frac{y_{j+2}}{y_{j+1}}} = \frac{\det T_{j-1} \det T_{n-j}^{(j)}}{\det T}, \tag{5}$$

because the resulting expression in the denominator,

$$b_j \det T_{j-1} \det T_{n-j}^{(j)} - a_j c_{j-1} \det T_{j-2} \det T_{n-j}^{(j)} - a_{j+1} c_j \det T_{j-1} \det T_{n-j-1}^{(j+1)},$$

is simply the expansion of the determinant of T by its j-th row. The off-diagonal entries of the same column were computed in a recursive way, using the diagonal entries; see [7, Eqs. (4)–(5)]. For example Eq. (4) from [7] yields,

$$\phi_{i,j} = \begin{cases} -c_i \dfrac{z_{i-1}}{z_i} \phi_{i+1,j} = -c_i \dfrac{\det T_{i-1}}{\det T_i} \phi_{i+1,j} & \text{if } i < j, \\ -a_i \dfrac{y_{i+1}}{y_i} \phi_{i-1,j} = -a_i \dfrac{\det T_{n-i}^{(i)}}{\det T_{n-i+1}^{(i-1)}} \phi_{i-1,j} & \text{if } i > j. \end{cases} \tag{6}$$

It is clear that the algorithm breaks down if any principal submatrix of T is singular.

Another algorithm recently given in [9, Algorithm 4.1], carries the same difficulty with principal submatrices. It is initiated by building in a recursive way two vectors, $\vec{\varphi}$, $\vec{\theta}$, with entries,

$$\begin{aligned} \varphi_i &= -c_i \frac{\det T_{i-1}}{\det T_i}, \\ \theta_i &= -a_i \frac{\det T_{n-i}^{(i)}}{\det T_{n-i+1}^{(i-1)}}. \end{aligned} \tag{7}$$

Then, the diagonal and the off-diagonal entries of the inverse were computed with expressions equivalent to (5)–(6), respectively. Besides a minor flop count, the principal advantage of Algorithm 4.1 from [9] with respect to the one given in [7, Section 2], is that expressions (7) permit us to control overflow and underflow in the inversion of some tridiagonal strictly nonsingular matrices. Algorithm 4.1 from [9] also breaks down if any principal submatrix is singular. To overcome these difficulties the symbolic computation, already introduced in [8], was applied.

In summary, the lack of success of previous specialized numerical algorithms for the inversion of general tridiagonal matrices is related to the location of null denominators in the involved formulae. Thus, if we handle expressions (1)–(4) for an algorithm of inversion, we note that this drawback does not appear, because the only involved denominator is the determinant of a tridiagonal nonsingular matrix.

A simple non Gaussian algorithm to compute the inverse of any tridiagonal nonsingular matrix is proposed in Appendix A. It represents a continuation of the results from [7], because its steps 2–3 are equivalent to those first steps of the method given in [7]. The determinants are evaluated with the vector solutions of the recurrences (3)–(4).

Our algorithm does not break down when any principal submatrix is singular. In steps 4–5 the off-diagonal entries of the inverse matrix are directly computed with the aid of Expression (1).

In the reduced case, and for matrices with a large order n, products of the a_i or the c_i that appear in (1) must be considered. Thus, we build recursively two vectors, prod a and prod c, to avoid the unnecessary computation as well as the repetition of large products. It is important for the efficacy of the algorithm. As for the computations of $(T^{-1})_{i,j}$ in (1), we can make the following substitutions,

$$(-1)^{i+j} \left(\prod_{k=j+1}^{i} a_k \right) = \frac{\text{prod } a(i)}{\text{prod } a(j)} \quad (i > j); \qquad (-1)^{i+j} \left(\prod_{k=i}^{j-1} c_k \right) = \frac{\text{prod } c(j)}{\text{prod } c(i)} \quad (i < j).$$

The introduction of the previous ratios for both vectors prod a and prod c requires us to handle the null entries a_i and c_i in the reduced case. For this task, we introduce the input vectors asign a; for the positions of rows with null entries in the lower subdiagonal, and asign c; for the positions of columns with null entries in the upper subdiagonal, respectively. Thus, we avoid the computation of the (null) entries with null products in the denominators.

When an entry a_i or c_i of T is a null entry, then this tridiagonal nonsingular matrix is a 2×2 block matrix, with a null block element. It is well known that its inverse is also a 2×2 block one, with the same null block element. Just consider as

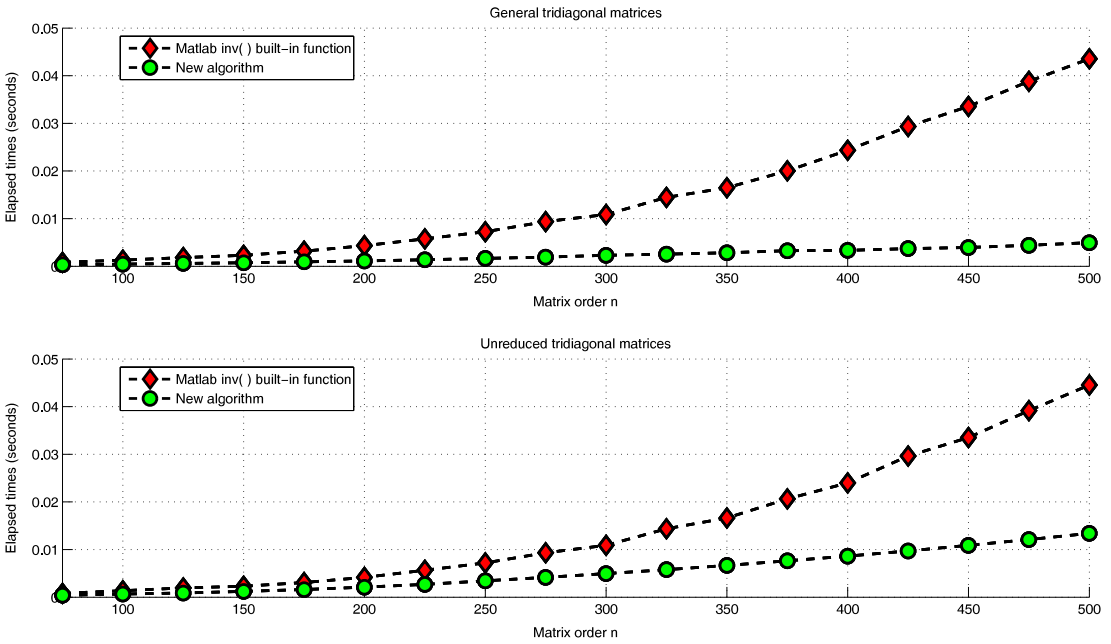

Fig. 1. Mean value of the elapsed time, 150 trials, in the computations of the inverses.

these null entries are not evaluated by our algorithm. We also take advantage of this fact in the step 1, when initializing T^{-1} as the null matrix of size n.

Therefore, the computational complexity decreases when the matrix T is a strongly reduced matrix; with many null entries a_i or c_i. Our algorithm is especially useful in this scenario, because the unnecessary computation of almost all the null entries of the inverse is avoided. On the other hand, for unreduced matrices the n^2 entries of the inverse are evaluated by considering the inputs asign $a = [1]$, asign $c = [1]$. For unreduced tridiagonal matrices the computational complexity is $O(n^2)$, with $4n^2 + O(n)$ flop counts.

We introduce in Fig. 1 graphics for the general as well as the unreduced case. To check the computational complexity for general tridiagonal matrices, the mean elapsed time of the algorithm of Appendix A is compared with respect to the built-in function *inv*() of the *Matlab* ® package. Here, the tridiagonal matrices are of order n with $75 \leq n \leq 500$, in steps of 25 units, and take random values from $[-5, 5]$. As we expected, the complexity of our algorithm decreases in the general case with respect to the unreduced one.

2.2. Current examples of tridiagonal matrices

Tridiagonal nonsymmetric matrices. We begin with tridiagonal nonsymmetric matrices (type 1 of the test matrices from [15]) with random entries from the interval $[-1; 1]$. We have checked matrices of this type up to $n = 1000$, and the algorithm works correctly in the usage range. An illustrative numerical situation, with $n = 7$, is given. We consider the unreduced case; asign $a = [1]$, asign $c = [1]$.

```
>> T =
          0.6294   0.8116        0        0        0        0        0
         -0.7460   0.2647   0.8268        0        0        0        0
               0  -0.8049   0.0938  -0.4430        0        0        0
               0        0   0.9150  -0.6848   0.9298        0        0
               0        0        0   0.9412  -0.0292   0.9143        0
               0        0        0        0   0.6006  -0.1565  -0.7162
               0        0        0        0        0   0.8315   0.5844

>> T^(-1) =
          0.8709  -0.6057   0.6789   0.4778   0.6671  -0.7072  -0.8667
          0.5568   0.4698  -0.5265  -0.3705  -0.5174   0.5484   0.6721
          0.6075   0.5125   0.7811   0.5497   0.7676  -0.8137  -0.9972
         -0.8830  -0.7450  -1.1354   0.7896   1.1026  -1.1687  -1.4323
         -1.2482  -1.0531  -1.6049   1.1161   0.0567  -0.0601  -0.0736
```

```
 0.8691  0.7333  1.1175 -0.7772 -0.0395  1.2012  1.4721
-1.2366 -1.0433 -1.5900  1.1058  0.0561 -1.7091 -0.3834
```

Tridiagonal reduced matrices. When the nonsingular matrices are also reduced matrices the algorithm works correctly. All numerical trials produce adequate outputs into the usage range. It also works in the limit of two-band triangular matrices. When the number of zeros in the subdiagonal increases, the complexity of the algorithm decreases. A numerical situation is provided with a random reduced matrix of order $n = 7$. Here, we take asign $a = [2, 4, 6, 7]$, asign $c = [3, 5]$.

```
>> T =
          0.9004 -1.0000       0        0       0       0       0
               0  0.5310       0        0       0       0       0
               0  1.0000 -0.0205  -1.0000       0       0       0
               0       0       0   0.4187       0       0       0
               0       0       0   1.0000  0.3594 -1.0000       0
               0       0       0        0       0 -0.7620 -1.0000
               0       0       0        0       0       0  0.9195

>> T^(-1) =
          1.1106   2.0916         0         0       0       0       0
               0   1.8832         0         0       0       0       0
               0  91.8653 -48.7805 -116.5046       0       0       0
               0       0         0    2.3883       0       0       0
               0       0         0   -6.6454  2.7824 -3.6515 -3.9711
               0       0         0         0       0 -1.3123 -1.4272
               0       0         0         0       0       0  1.0875
```

Tridiagonal symmetric matrices. For the computation of the inverse of tridiagonal symmetric matrices, step 5 of the algorithm of Appendix A can be suppressed, because pairs of symmetric entries of the inverse matrix can be evaluated at step 4. When a tridiagonal symmetric matrix is also a reduced one, both the tridiagonal matrix and its inverse have the same diagonal block structure. Hence, the computation of the inverse matrix is simple. For these type of matrices a numerical situation handled with our algorithm, with asign $a =$ asign $c = [2, 3, 5, 7]$, is given.

```
>> T =
         -0.9547       0       0       0       0       0       0
               0 -1.9182       0       0       0       0       0
               0       0  0.6465 -2.0000       0       0       0
               0       0 -2.0000 -1.2752       0       0       0
               0       0       0       0  0.5087 -1.0000       0
               0       0       0       0 -1.0000  1.3667       0
               0       0       0       0       0       0 -1.5909

>> T^(-1) =
         -1.0474       0       0       0       0       0       0
               0 -0.5213       0       0       0       0       0
               0       0  0.2643 -0.4146       0       0       0
               0       0 -0.4146 -0.1340       0       0       0
               0       0       0       0 -4.4845 -3.2813       0
               0       0       0       0 -3.2813 -1.6692       0
               0       0       0       0       0       0 -0.6286
```

Tridiagonal nonsingular matrices with singular principal submatrices. If a tridiagonal nonsingular matrix is not a strictly nonsingular one, some principal submatrices are singular. Then, new lines of zeros appear in the inverse matrix [15]. Our algorithm does not avoid the unnecessary evaluation of such zeros, although it can be improved. We give a numerical example for the inverse of a symmetric Toeplitz matrix of order $n = 8$ (type 8 of the test matrices from [15]) with $b_i = 0$, $a_i = c_i = 1$. We consider the unreduced case; asign $a = [1]$, asign $c = [1]$.

```
>> T^(-1) =
           0     1     0    -1     0     1     0    -1
           1     0     0     0     0     0     0     0
           0     0     0     1     0    -1     0     1
          -1     0     1     0     0     0     0     0
           0     0     0     0     0     1     0    -1
```

```
 1     0    -1     0     1     0     0     0
 0     0     0     0     0     0     0     1
-1     0     1     0    -1     0     1     0
```

Tridiagonal strictly diagonally dominant matrices. Finally, the strictly diagonally dominant matrices are tested. In particular, we check two Toeplitz matrices. Note as for tridiagonal Toeplitz-like matrices the solutions of the recurrences (3)–(4) are the same; i.e. $\det T_k = \det T_k^{(n-k)}$. First we consider here the inversion of symmetric Toeplitz matrices of type 7 from [15], with $b_i = 10^8$, $a_i = c_i = 1$. For matrices of order $n < 39$ the algorithm works correctly. The algorithm overflows during the computation of the determinants, when the order increases in such a way that $n \geq 39$. We also consider another nonsymmetric Toeplitz matrix, with $b_i = 10^{-3}$, $a_i = -10^{-4}$ $c_i = 10^{-5}$. For matrices of order $n < 67$ the algorithm works correctly. Nevertheless, when $n \geq 67$ the algorithm underflows.

2.3. Algorithm for avoiding overflow and underflow in diagonally dominant matrices

All the entries on the main diagonal of any tridiagonal diagonally dominant matrix are non null entries, when such matrix is nonsingular. For sufficiently large order n, it will enables us to avoid overflow and underflow of the recurrences and large products involved in the computation of the inverses of such matrices. To reach this goal, we introduce the following scaling transformations to obtain the solutions from the linear recurrences (3) and (4), respectively,

$$\det T_{k-1} = \left(\prod_{i=1}^{k-1} b_i\right) SL_{k-1}; \qquad \det T_{n-k}^{(k)} = \left(\prod_{i=k+1}^{n} b_i\right) SR_{n-k}. \tag{8}$$

The transformed recurrences, with initial conditions $SL_0 = SL_1 = 1$; $SR_0 = SR_1 = 1$, are,

$$SL_{k+2} = SL_{k+1} - \frac{a_{k+2}c_{k+1}}{b_{k+2}b_{k+1}} SL_k; \qquad SR_{k+2} = SR_{k+1} - \frac{a_{n-k}c_{n-k-1}}{b_{n-k}b_{n-k-1}} SR_k. \tag{9}$$

Note that, for strictly diagonally dominant matrices, when solutions of the linear recurrences (3) and (4) grow (or reduce) quickly in magnitude, although the transformed recurrences (8) and (9) have a slow variation from the given initial conditions. This aspect has a main role when handling overflow and underflow using such recurrences. After the introduction of the preceding transformations, the representation (1) for the entries of the inverse matrix yields

$$(T^{-1})_{ij} = \begin{cases} (-1)^{i+j} \left(\prod_{k=j+1}^{i} \left(\frac{a_k}{b_k}\right)\right) \frac{SL_{j-1} \cdot SR_{n-i}}{b_j \cdot SL_n} & \text{if } i > j, \\ (-1)^{i+j} \left(\prod_{k=i}^{j-1} \left(\frac{c_k}{b_k}\right)\right) \frac{SL_{i-1} \cdot SR_{n-j}}{b_j \cdot SL_n} & \text{if } i \leq j. \end{cases} \tag{10}$$

For sufficiently large n, Expression (10) does not solve these difficulties. In particular, we adapt the algorithm from Appendix A to the new representation (10) and the example of type 7 from [15] is checked. We note that the algorithm underflows, because the denominators involved in the products from (10) are bigger. Then, large products must also be avoided.

A recursive relation between two consecutive entries in the same row of the inverse matrix can be derived from (10),

$$(T^{-1})_{ij} = \begin{cases} -\left(\frac{a_{j+1} \cdot SL_{j-1}}{b_j \cdot SL_j}\right) (T^{-1})_{i,j+1} & \text{if } i > j, \\ -\left(\frac{c_{j-1} \cdot SR_{n-j}}{b_j \cdot SR_{n-j+1}}\right) (T^{-1})_{i,j-1} & \text{if } i < j. \end{cases} \tag{11}$$

Indeed, Expression (10) can be considered as a direct evaluation for the diagonal entries of the inverse and it does not bring problems, because in this scenario all terms of the denominators are non-null terms. Also, large products are not involved. That is, after the evaluation of the main diagonal, we can compute all the entries using a row-by-row procedure, via the recursive relations (11). Hence, the difficulties associated with overflow and underflow in the recurrences and the large products are controlled.

The preceding method is introduced in the alternative algorithm of Appendix B for the inversion of the tridiagonal diagonally dominant matrices. This algorithm gives adequate outputs even on matrices of a large order n. However, the algorithm of the Appendix A goes beyond of the usage range.

We check the algorithm of Appendix B with the numerical tests on strictly diagonally dominant matrices given in Section 2.2. Thus, for symmetric matrices of type 7 from [15], the algorithm works correctly; overflow is avoided for $n > 39$. Also, for nonsymmetric Toeplitz matrices with $b_i = 10^{-3}$, $a_i = -10^{-4}$, and $c_i = 10^{-5}$, the algorithm of Appendix B does not underflow for $n > 67$. Other tested strictly diagonally dominant matrices, where the algorithm of Appendix A overflows or underflows, have also given adequate outputs. A numerical illustration is given, with $n = 7$, and the vectors asign $a = [2]$, asign $c = [4, 6]$,

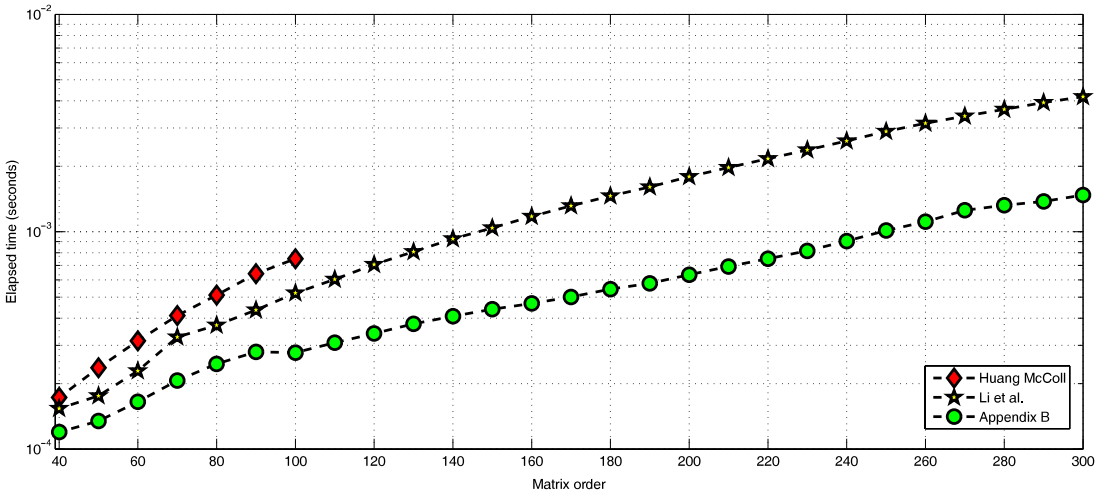

Fig. 2. Comparison of the mean value, in Log scale, for the elapsed times, 150 trials, in the computation of the inverses of some tridiagonal diagonally dominant matrices.

```
>> T =
     2     1     0     0     0     0     0
     0     4     1     0     0     0     0
     0    -1     6     0     0     0     0
     0     0    -2     8     2     0     0
     0     0     0    -2    10     0     0
     0     0     0     0     2    12     4
     0     0     0     0     0     6    14

>> T^(-1) =
    0.5000   -0.1200    0.0200         0         0         0         0
         0    0.2400   -0.0400         0         0         0         0
         0    0.0400    0.1600         0         0         0         0
         0    0.0095    0.0381    0.1190   -0.0238         0         0
         0    0.0019    0.0076    0.0238    0.0952         0         0
         0   -0.0004   -0.0015   -0.0046   -0.0185    0.0972   -0.0278
         0    0.0002    0.0006    0.0020    0.0079   -0.0417    0.0833
```

The computational complexity of our proposed algorithms depends on the subdiagonal entries of the involved tridiagonal matrix. For the unreduced case both algorithms requires $4n^2 + O(n)$ flop counts.

As the diagonally dominant matrices are also strictly nonsingular, we can compare the computational complexity of the algorithm of Appendix B with those given in [7,9]. For this task we handle random tridiagonal nonsingular matrices with non-null entries $b_i = 10^{-3}$, $a_i = -10^{-4} \cdot \text{randi}([-5; 5])$, and $c_i = 10^{-4} \cdot \text{randi}([-5; 5])$, where randi$([-5; 5])$ is a random integer from the given closed interval.

In Fig. 2 a comparison of the mean elapsed times, on 150 trials, is provided. In each trial, random matrices are taken in an increasing order, from $n = 40$ to $n = 300$, in steps of 10 units. For a finer comparison, the mean values are given in a Log scale. Outcomes show the robustness of the algorithms under study. Nevertheless, the algorithm from [7, Section 2], underflows when $n \geq 110$. As we expected, the algorithm from [9] and the one proposed in Appendix B avoid underflow.

Algorithm 4.1 from [9] obtains shorter elapsed times in the unreduced scenario. Although for general reduced diagonally dominant matrices, the algorithm of Appendix B has a minor complexity, as it can be observed in Fig. 2. Therefore, a combination of both numerical algorithms could be of interest for the inversion of tridiagonal strictly nonsingular matrices.

Acknowledgments

The authors express their sincere thanks to the reviewers for their constructive comments and remarks, resulting in the improvement of this paper. This work was partially supported by a research grant of the UPM-CAM in Madrid, Spain.

Appendix A. A numerical algorithm of inversion

Input:

- Order n and the components $\{a_i, b_i, c_i\}$ of the tridiagonal matrix T.

- A vector with the positions of rows with null entries in the lower subdiagonal, (asign a).
- The total number of null entries in the lower subdiagonal, (*numbera*).
- A vector with the positions of columns with null entries in the upper subdiagonal, (asign c).
- The total number of null entries in the upper subdiagonal, (*numberc*).

(For unreduced matrices we take number $= 1$ and asign $= 1$ for both subdiagonals).
Output: T^{-1} the inverse of the tridiagonal matrix T.

1. Initialize T^{-1} as the null matrix of size n.
2. Set the initial conditions. For $k = 3 : n + 1$; build the two vectors of principal determinants.
3. For $i = 1 : n$; evaluate $(T^{-1})_{ii}$, the entries of the main diagonal.
4. Evaluate the entries of the lower triangle of T^{-1}.
   (a) prod $a(1) = 1$. For $k = 2$:asign $a(1) - 1$; prod $a(k) = -a_k *$ prod $a(k-1)$.
   For $j = 1 : k - 1$; evaluate $(T^{-1})_{k,j}$.
   (b) For $m = 2$:*numbera*; prod a(asign $a(m-1)) = 1$.
   For $k =$ asign $a(m-1) + 1$:asign $a(m) - 1$: prod $a(k) = -a_k *$ prod $a(k-1)$;
   For $j =$ asign $a(m-1) : k - 1$; evaluate $(T^{-1})_{k,j}$.
   (c) prod a(asign a(*numbera*)) $= 1$;
   For $k =$ asign a(*numbera*) $+ 1 : n$; prod $a(k) = -a_k *$ prod $a(k-1)$;
   For $j =$ asign a(*numbera*) $: k - 1$; evaluate $(T^{-1})_{k,j}$.
5. Evaluate the entries of the upper triangle of T^{-1}.
   (a) prod $c(1) = 1$. For $k = 2$:asign $c(1) - 1$; prod $c(k) = -c_{k-1} *$ prod $c(k-1)$.
   For $i = 1 : k - 1$; evaluate $(T^{-1})_{i,k}$.
   (b) For $m = 2 :$ *numberc*; prod c(asign $c(m-1)) = 1$.
   For $k =$ asign $c(m-1) + 1$:asign $c(m) - 1$: prod $c(k) = -c_{k-1} *$ prod $c(k-1)$;
   For $i =$ asign $c(m-1) : k - 1$; evaluate $(T^{-1})_{i,k}$.
   (c) prod c(asign c(*numberc*)) $= 1$;
   For $k =$ asign c(*numberc*) $+ 1 : n$; prod $c(k) = -c_{k-1} *$ prod $c(k-1)$;
   For $i =$ asign c(*numberc*) $: k - 1$; evaluate $(T^{-1})_{i,k}$.

Appendix B. A numerical algorithm for the inversion of tridiagonal diagonally dominant matrices

Input:

- Order n and the components $\{a_i, b_i, c_i\}$ of the tridiagonal matrix T.
- A vector with the positions of rows with null entries in the lower subdiagonal, (asign a).
- The total number of null entries in the lower subdiagonal, (*numbera*).
- A vector with the positions of columns with null entries in the upper subdiagonal, (asign c).
- The total number of null entries in the upper subdiagonal, (*numberc*).

(For unreduced matrices we take number $= 1$ and asign $= 1$ for both subdiagonals).
Output: T^{-1} the inverse of the tridiagonal diagonally dominant matrix T.

1. Initialize T^{-1} as the null matrix of size n.
2. Set the initial conditions. For $k = 3 : n + 1$; build the two scaling vectors, (9).
3. For $i = 1 : n$; evaluate $(T^{-1})_{ii}$, the entries of the main diagonal, Eq. (10).
4. Evaluate the entries of the lower triangle of T^{-1}, relation (11).
   (a) For $k = 2$: asign $a(1) - 1$;
   For $j = k - 1 : -1 : 1$; evaluate $(T^{-1})_{k,j}$.
   (b) For $m = 2 :$ *numbera*;
   For $k =$ asign $a(m-1) + 1$: asign $a(m) - 1$:
   For $j = k - 1 : -1$: asign $a(m-1)$; evaluate $(T^{-1})_{k,j}$.
   (c) For $k =$ asign a(*numbera*) $+ 1 : n$;
   For $j = k - 1 : -1$: asign a(*numbera*); evaluate $(T^{-1})_{k,j}$.
5. Evaluate the entries of the upper triangle of T^{-1}, relation (11).
   (a) For $k = 2$: asign $c(1) - 1$;
   For $j = k$: asignc$(1) - 1$; evaluate $(T^{-1})_{k-1,j}$.
   (b) For $m = 2 :$ *numberc*;
   For $k =$ asign $c(m-1) + 1$: asign $c(m) - 1$;
   For $j = k$: asign $c(m-1)$; evaluate $(T^{-1})_{k-1,j}$.
   (c) For $k =$ asign c(*numberc*) $+ 1 : n$;
   For $j = k : n$; evaluate $(T^{-1})_{k-1,j}$.

References

[1] P. Alonso, J. Delgado, R. Gallego, J.M. Peña, Growth factors of pivoting strategies associated with Neville elimination, J. Comput. Appl. Math. 235 (2011) 1755–1762.
[2] B. Bukhberger, G.A. Emelýanenko, Methods of inverting tridiagonal matrices, USSR Comput. Math. Math. Phys. 13 (1973) 10–20.
[3] S.R. Vatsya, H.O. Pritchard, An explicit inverse of a tridiagonal matrix, Int. J. Comput. Math. 14 (1983) 295–304.
[4] T. Yamamoto, Y. Ikebe, Inversion of band matrices, Linear Algebra Appl. 24 (1979) 105–111.
[5] M.H. Koulaei, F. Toutounian, On computing of block ILU preconditioner for block tridiagonal systems, J. Comput. Appl. Math. 202 (2007) 248–257.
[6] B.V. Minchev, Some algorithms for solving special tridiagonal block Toeplitz linear systems, J. Comput. Appl. Math. 156 (2003) 179–200.
[7] Y. Huang, W.F. McColl, Analytical inversion of general tridiagonal matrices, J. Phys. A: Math. Gen. 30 (1997) 7919–7933.
[8] M. El-Mikkawy, A. Karawia, Inversion of general tridiagonal matrices, Appl. Math. Lett. 19 (2006) 712–720.
[9] Huo-Biao Li, T. Huang, X. Liu, Hong Li, On the inverses of general tridiagonal matrices, Linear Algebra Appl. 433 (2010) 965–983.
[10] C.M. da Fonseca, On the eigenvalues of some tridiagonal matrices, J. Comput. Appl. Math. 200 (2007) 283–286.
[11] R. Usmani, Inversion of a tridiagonal Jacobi matrix, Linear Algebra Appl. 212–213 (1994) 413–414.
[12] J. Abderramán Marrero, V. Tomeo, On the closed representation for the inverses of Hessenberg matrices, J. Comput. Appl. Math. 236 (2012) 2962–2970.
[13] J. Abderramán Marrero, M. Rachidi, Companion factorization in the general group $GL(n;\mathbb{C})$ and applications, Linear Algebra Appl. 434 (2011) 1261–1271.
[14] M. El-Mikkawy, A fast algorithm for evaluating nth order tri-diagonal determinants, J. Comput. Appl. Math. 166 (2004) 581–584.
[15] I.D. Dhillon, Reliable computation of the condition number of a tridiagonal matrix in $O(n)$ times, SIAM J. Matrix Anal. Appl. 19 (1998) 776–796.