British Journal of Mathematics & Computer Science
4(9): 1246-1267, 2014

SCIENCEDOMAIN *international*
www.sciencedomain.org

SCIENCEDOMAIN

Algorithms for Solving Doubly Bordered Tridiagonal Linear Systems

Moawwad El-Mikkawy*[1] and Faiz Atlan[1]

[1] *Mathematics Department, Faculty of Science, Mansoura University, Mansoura 35516, Egypt.*

Original Research Article

Received: 31 December 2013
Accepted: 02 February 2014
Published: 27 February 2014

Abstract

In this paper we focus on solving doubly bordered tridiagonal linear systems of equations via transformation. It investigates numeric and symbolic algorithms for solving such systems. The computational cost of the algorithms are given. A MAPLE procedure based on these algorithms is listed. Some illustrative examples are introduced.

Keywords: Doubly bordered; Tridiagonal matrices; Partitioned matrices; Algorithm; LU factorization; MAPLE.

2010 Mathematics Subject Classification:15A09, 15A23, 15B05, 65F05, 65Fxx, 68W30.

1 Introduction and Basic Definitions

We begin this section by giving the following definitions:

Definition 1.1 [1]. An $n \times n$ matrix $A = (a_{ij})_{i,j=1}^{n}$ is said to have upper bandwidth p if p is the smallest integer such that:

$$a_{ij} = 0, \quad j - i \geq p,$$

and similarly, to have lower bandwidth q if q is the smallest integer such that:

$$a_{ij} = 0, \quad i - j \geq q.$$

The bandwidth, w for the matrix A is defined to be $w = p + q - 1$.

Definition 1.2 [2]. The symmetric matrix $A = (a_{ij})_{i,j=1}^{n}$ is called positive definite if and only if

$$\boldsymbol{x}^T A \boldsymbol{x} > 0, \quad \text{for all} \quad \boldsymbol{x} \in \mathbb{R}^n, \quad \boldsymbol{x} \neq 0.$$

Definition 1.3 [2]. An $n \times n$ matrix A is called diagonally dominant if

$$|a_{ii}| \geq \sum_{\substack{j=1, \\ j \neq i}}^{n} |a_{ij}|, \quad \text{holds for each } i = 1, 2, ..., n,$$

*Corresponding author: E-mail: m_elmikkawy@yahoo.com

and strictly diagonally dominant if

$$|a_{ii}| > \sum_{\substack{j=1,\\ j\neq i}}^{n} |a_{ij}|, \quad \text{holds for each } i = 1, 2, ..., n.$$

In many scientific and engineering applications, different special linear systems of equations arise. For such systems the coefficient matrix has special structure. Sparse matrices which contain a majority of zeros occur are often encountered. It is usually more efficient to solve these systems using tailor-made algorithms, much faster and with less storage than a full matrix. This can be achieved by taking advantage of the special structure of the coefficient matrix. Important examples are band matrices, and the most common cases are the matrices of tridiagonal type for which $p = q = w - 1 = 2$. Tridiagonal systems of linear equations take the form:

$$T\,\boldsymbol{x} = \boldsymbol{f}, \tag{1.1}$$

where

$$T = \begin{bmatrix} d_1 & a_1 & 0 & ... & 0 \\ b_1 & d_2 & a_2 & \ddots & \vdots \\ 0 & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & b_{n-2} & d_{n-1} & a_{n-1} \\ 0 & \dots & 0 & b_{n-1} & d_n \end{bmatrix}, \tag{1.2}$$

$\boldsymbol{x} = [x_1, x_2, ..., x_n]^T$ and $\boldsymbol{f} = [f_1, f_2, ..., f_n]^T$. The superscript T corresponds to the transpose operation. This type of matrices frequently appears in many applications. For example in parallel computing, telecommunication system analysis, solving differential equations using finite differences, heat conduction and fluid flow problems. A general $n \times n$ tridiagonal matrix of the form (1.2) can be stored in $3n - 2$ memory locations, rather than n^2 memory locations for a full matrix, by using three vectors $\mathbf{a} = [a_1, a_2, \dots, a_{n-1}]$, $\mathbf{b} = [b_1, b_2, \dots, b_{n-1}]$, and $\mathbf{d} = [d_1, d_2, \dots, d_n]$. This is always a good habit in computation in order to save memory space. To study tridiagonal matrices it is convenient to introduce vector $\mathbf{c}$ defined by ([3], [4]):

$$\mathbf{c} = [c_1, c_2, ..., c_n], \tag{1.3}$$

where

$$c_i = \begin{cases} d_1, & \text{for } i = 1 \\ d_i - \frac{a_{i-1}\, b_{i-1}}{c_{i-1}}, & \text{for } i = 2, 3, ..., n. \end{cases} \tag{1.4}$$

In [5] it is shown that the tridiagonal matrix (1.2) is positive definite if and only if $c_i > 0$, for all $i = 1, 2, ..., n$. This is an easy way to check weather a tridiagonal matrix is positive definite or not. For some important results concerning tridiagonal matrix the reader may refer to ([6], [7], [8], [5], [9], [10], [4], [11], [12], [13], [14], [15], [16], [17], [18], [19]). The motivation of the current paper is to derive algorithms for solving doubly bordered tridiagonal linear systems of two types of the forms:

$$\begin{bmatrix} d_1 & a_1 & 0 & \dots & \dots & 0 & p_1 \\ b_1 & d_2 & a_2 & \ddots & & \vdots & p_2 \\ 0 & b_2 & d_3 & a_3 & \ddots & \vdots & p_3 \\ \vdots & \ddots & \ddots & \ddots & \ddots & 0 & \vdots \\ \vdots & & \ddots & b_{n-3} & d_{n-2} & a_{n-2} & p_{n-2} \\ 0 & \dots & \dots & 0 & b_{n-2} & d_{n-1} & a_{n-1} \\ q_1 & q_2 & q_3 & \dots & q_{n-2} & b_{n-1} & d_n \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_{n-2} \\ x_{n-1} \\ x_n \end{bmatrix} = \begin{bmatrix} f_1 \\ f_2 \\ f_3 \\ \vdots \\ f_{n-2} \\ f_{n-1} \\ f_n \end{bmatrix}, \tag{1.5}$$

and

$$\begin{bmatrix} d_1 & a_1 & p_{n-2} & \dots & p_3 & p_2 & p_1 \\ b_1 & d_2 & a_2 & 0 & \dots & \dots & 0 \\ q_{n-2} & b_2 & d_3 & a_3 & \ddots & & \vdots \\ \vdots & 0 & \ddots & \ddots & \ddots & \ddots & \vdots \\ q_3 & \vdots & \ddots & b_{n-3} & d_{n-2} & a_{n-2} & 0 \\ q_2 & \vdots & & \ddots & b_{n-2} & d_{n-1} & a_{n-1} \\ q_1 & 0 & \dots & \dots & 0 & b_{n-1} & d_n \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_{n-2} \\ x_{n-1} \\ x_n \end{bmatrix} = \begin{bmatrix} g_1 \\ g_2 \\ g_3 \\ \vdots \\ g_{n-2} \\ g_{n-1} \\ g_n \end{bmatrix} \tag{1.6}$$

respectively. The linear systems (1.5) and (1.6) frequently occur in engineering computation and analysis, e.g. in computation of electric power system and in solution of partial differential equations, as referred in ([20], [21], [22], [23], [24], [25], [26], [27], [28], [29], [30]). The linear systems defined by (1.5) and (1.6) are called doubly bordered tridiagonal linear systems of type (A) and of type (B), respectively.

Throughout this paper, the word 'simplify' means simplify the expression under consideration to its simplest rational form.
The organization of the paper is as follows. The main results are given in Section 2 and Section 3. In Section 4, a MAPLE procedure is introduced. Some illustrative examples are given in Section 5.

2 Algorithms for Solving Doubly Bordered Tridiagonal Linear Systems of Type (A)

In this section, we are going to consider the construction of new algorithms for solving doubly bordered tridiagonal linear systems of type (A) via transformation. For this purpose it is convenient to introduce the vector **c** $= [c_1, c_2, ..., c_n]$ whose first $(n-1)$ components, $c_1, c_2, ..., c_{n-1}$ are given by (1.4). The system (1.5) can be completely described by the augmented matrix, G given by:

$$G := [T_n, \boldsymbol{f}] = \left[\begin{array}{ccccccc|c} d_1 & a_1 & 0 & \dots & \dots & 0 & p_1 & f_1 \\ b_1 & d_2 & a_2 & \ddots & & \vdots & p_2 & f_2 \\ 0 & b_2 & d_3 & a_3 & \ddots & \vdots & p_3 & f_3 \\ \vdots & \ddots & \ddots & \ddots & \ddots & 0 & \vdots & \vdots \\ \vdots & & \ddots & b_{n-3} & d_{n-2} & a_{n-2} & p_{n-2} & f_{n-2} \\ 0 & \dots & \dots & 0 & b_{n-2} & d_{n-1} & a_{n-1} & f_{n-1} \\ q_1 & q_2 & q_3 & \dots & q_{n-2} & b_{n-1} & d_n & f_n \end{array}\right]. \tag{2.1}$$

The matrix G contains all the information about the system (1.5) that is necessary to determine its solution, but in a compact form. Let $\mathbf{R}_i$ denotes the i-th row of the matrix G. Performing the following row operations on G, in the same order:

$\mathbf{R}_1 \longleftarrow \mathbf{R}_1/c_1$

For $i = 2, 3, ..., n-1$ do

$\mathbf{R}_i \longleftarrow \mathbf{R}_i - b_{i-1}\mathbf{R}_{i-1}$

$\mathbf{R}_i \longleftarrow \mathbf{R}_i/c_i$

End do.

For $j = 1, 2, ..., n-1$ do

$\mathbf{R}_n \longleftarrow \mathbf{R}_n - h_j\mathbf{R}_j$

End do.

$\mathbf{R}_n \longleftarrow \mathbf{R}_n/c_n.$

Then we have the resulting transformed linear system of the form:

$$\left[\begin{array}{cccccc|c} 1 & y_1 & 0 & \dots & 0 & v_1 & z_1 \\ 0 & 1 & y_2 & \ddots & \vdots & v_2 & z_2 \\ \vdots & \ddots & \ddots & \ddots & 0 & \vdots & \vdots \\ \vdots & & \ddots & 1 & y_{n-2} & v_{n-2} & z_{n-2} \\ \vdots & & & \ddots & 1 & v_{n-1} & z_{n-1} \\ 0 & \dots & & \dots & 0 & 1 & z_n \end{array}\right], \tag{2.2}$$

where

$$c_1 = d_1, \quad y_1 = \frac{a_1}{c_1}, \quad v_1 = \frac{p_1}{c_1}, \quad z_1 = \frac{f_1}{c_1}, \quad h_1 = q_1,$$

For $i = 2, 3, ..., n-2$ do

$$\begin{aligned}
&c_i = d_i - b_{i-1}\, y_{i-1},\\
&y_i = \tfrac{a_i}{c_i},\\
&v_i = \tfrac{1}{c_i}(p_i - b_{i-1}\, v_{i-1}),\\
&z_i = \tfrac{1}{c_i}(f_i - b_{i-1}\, z_{i-1}).\\
&h_i = q_i - h_{i-1}\, y_{i-1},
\end{aligned}$$

End do.

$$\begin{aligned}
&c_{n-1} = d_{n-1} - b_{n-2}\, y_{n-2},\\
&v_{n-1} = \frac{1}{c_{n-1}}(a_{n-1} - b_{n-2}\, v_{n-2}),\\
&z_{n-1} = \frac{1}{c_{n-1}}(f_{n-1} - b_{n-2}\, z_{n-2}),\\
&h_{n-1} = b_{n-1} - h_{n-2}\, y_{n-2},\\
&c_n = d_n - \sum_{i=1}^{n-1} h_i\, v_i,\\
&z_n = \frac{1}{c_n}(f_n - \sum_{i=1}^{n-1} h_i\, z_i).
\end{aligned} \tag{2.3}$$

It is well known that [1] the reduced system (2.2) has the same solution as the original linear system (1.5). At this stage, the determinant of the coefficient matrix in (1.5) can be computed using the following computational symbolic algorithm. It is an extension of the **DETGTRI** algorithm in [3] and the **PERTRI** algorithm in [10]. The parameter 's' in the algorithm is just a symbolic name. It is a dummy argument and its actual value is zero.

Algorithm 2.1. An algorithm for computing the determinant of doubly bordered tridiagonal matrices of type (A).

To compute the determinant of the coefficient matrix in (1.5), we may proceed as follows:

INPUT: Order of the coefficient matrix n and the components, $a_i,\, d_i,\, b_i,\, p_i,\, q_i$.
OUTPUT: The determinant of the coefficient matrix in (1.5).
Step 1: Set $c_1 = d_1$. If $c_1 = 0$ then $c_1 = s$ end if.
Step 2: For $i = 2, 3, ..., n-1$ do
Compute and simplify:
$c_i = d_i - \frac{a_{i-1}\, b_{i-1}}{c_{i-1}}$. If $c_i = 0$ then $c_i = s$ end if.
End do.
Step 3: Compute c_n using (2.3).
Step 4: Compute and simplify:
$P(s) = \prod_{r=1}^{n} c_r$.
Step 5: $\det(T_n) = P(0)$.

The Algorithm 2.1, will be referred to as **DETDBTRI-A** algorithm. The transformed system (2.2) is triangular and easy to solve by backward substitution. Consequently, the linear system (1.5) can be solved using the following algorithm:

Algorithm 2.2. Numeric algorithm for solving doubly bordered tridiagonal linear system of type (A).

To solve the linear system of the form (1.5), we may proceed as follows:

INPUT: Order of the coefficient matrix n and the components, $a_i,\ d_i,\ b_i,\ f_i,\ p_i,\ q_i$.
OUTPUT: The determinant of the coefficient matrix in (1.5) and the solution vector
$\boldsymbol{x} = [x_1, x_2, \dots, x_n]^T$.
Step 1: Set $c_1 = d_1$, $y_1 = \frac{a_1}{c_1}$, $v_1 = \frac{p_1}{c_1}$, $z_1 = \frac{f_1}{c_1}$, and $h_1 = q_1$.
Step 2: For $i = 2, 3, ..., n-2$ do
Compute and simplify:
$c_i = d_i - b_{i-1}\, y_{i-1}$,
$y_i = \dfrac{a_i}{c_i}$,
$v_i = \dfrac{1}{c_i}(p_i - b_{i-1}\, v_{i-1})$,
$z_i = \dfrac{1}{c_i}(f_i - b_{i-1}\, z_{i-1})$,
$h_i = q_i - h_{i-1} y_{i-1}$,
End do.
Step 3: Set $c_{n-1} = d_{n-1} - b_{n-2}\, y_{n-2}$,
$v_{n-1} = \dfrac{1}{c_{n-1}}(a_{n-1} - b_{n-2}\, v_{n-2})$,
$z_{n-1} = \dfrac{1}{c_{n-1}}(f_{n-1} - b_{n-2}\, z_{n-2})$,
$h_{n-1} = b_{n-1} - h_{n-2} y_{n-2}$.
Step 4: Set $c_n = d_n - \sum\limits_{i=1}^{n-1} h_i\, v_i$,
$z_n = \dfrac{1}{c_n}(f_n - \sum\limits_{i=1}^{n-1} h_i\, z_i)$.
Step 5: Use the **DETDBTRI-A** algorithm to check the non-singularity of the coefficient matrix of the system (1.5).
Step 6: If the determinant of the coefficient matrix in (1.5) equals zero, then Exiterror('No solutions') end if.
Step 7: Compute the solution vector $\boldsymbol{x} = [x_1, x_2, \dots, x_n]^T$ using
$x_n = z_n$
$x_{n-1} = z_{n-1} - v_{n-1}\, x_n$
Step 8: For $i = n-2, n-3, ..., 1$ do
$x_i = z_i - y_i\, x_{i+1} - v_i x_n$
End do.

The Algorithm 2.2, will be referred to as **TRANSDBTRI-AI** algorithm. The cost of the algorithm is $(11n - 16)$ multiplications/divisions and $(8n - 13)$ addtions/subtractions.
Note that the algorithm **TRANSDBTRI-AI** works properly only if $c_i \neq 0$ for all $i = 1, 2, ..., n$.

The following symbolic version algorithm is developed in order to remove the cases where the numeric algorithm **TRANSDBTRI-AI** fails.

Algorithm 2.3. Symbolic version algorithm for **TRANSDBTRI-AI** algorithm.

To solve the linear system of the form (1.5), we may proceed as follows:

INPUT: Order of the matrix n and the components, $a_i,\ d_i,\ b_i,\ f_i,\ p_i,\ q_i$.
OUTPUT: The determinant of the coefficient matrix in (1.5) and the solution vector $\boldsymbol{x} = [x_1, x_2, \ldots, x_n]^T$.
Step 1: Set $c_1 = d_1$. If $c_1 = 0$ then $c_1 = s$ end if. $y_1 = \frac{a_1}{c_1},\ \ v_1 = \frac{p_1}{c_1},\quad z_1 = \frac{f_1}{c_1}$, and $h_1 = q_1$.
Step 2: For $i = 2, 3, ..., n-2$ do
Compute and simplify:
$c_i = d_i - b_{i-1}\, y_{i-1}$. If $c_i = 0$ then $c_i = s$ end if.
$y_i = \dfrac{a_i}{c_i}$,
$v_i = \dfrac{1}{c_i}(p_i - b_{i-1}\, v_{i-1})$,
$z_i = \dfrac{1}{c_i}(f_i - b_{i-1}\, z_{i-1})$,
$h_i = q_i - h_{i-1} y_{i-1}$,
End do.
Step 3: Set $c_{n-1} = d_{n-1} - b_{n-2}\, y_{n-2}$. If $c_{n-1} = 0$ then $c_{n-1} = s$ end if.
$v_{n-1} = \dfrac{1}{c_{n-1}}(a_{n-1} - b_{n-2}\, v_{n-2})$,
$z_{n-1} = \dfrac{1}{c_{n-1}}(f_{n-1} - b_{n-2}\, z_{n-2})$,
$h_{n-1} = b_{n-1} - h_{n-2} y_{n-2}$.
Step 4: Set $c_n = d_n - \sum\limits_{i=1}^{n-1} h_i\, v_i$. If $c_n = 0$ then $c_n = s$ end if.
$z_n = \dfrac{1}{c_n}(f_n - \sum\limits_{i=1}^{n-1} h_i\, z_i)$.
Step 5: Use the **DETDBTRI-A** algorithm to check the non-singularity of the coefficient matrix of the system (1.5).
Step 6: If the determinant of the coefficient matrix in (1.5) equals zero, then Exiterror('No solutions') end if.
Step 7: Compute the solution vector $\boldsymbol{x} = [x_1, x_2, \ldots, x_n]^T$ using
$x_n = z_n$
$x_{n-1} = z_{n-1} - v_{n-1}\, x_n$
Step 8: For $i = n-2, n-3, ..., 1$ do
$x_i = z_i - y_i\, x_{i+1} - v_i x_n$
End do.
Step 9: Substitute $s = 0$ in all expressions of the solution vector $x_i, i = 1, 2, ..., n$. The Algorithm 2.3, will be referred to as **TRANSDBTRI-AII** algorithm. At this point it is worth mentioned that:

- The values $z_i,\ i = 1, 2, ..., n$ in (2.3) satisfy:

$$\begin{bmatrix} c_1 & 0 & \ldots & & \ldots & \ldots & 0 \\ b_1 & c_2 & \ddots & & & & \vdots \\ 0 & b_2 & c_3 & \ddots & & & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & & \vdots \\ \vdots & & \ddots & b_{n-3} & c_{n-2} & \ddots & \vdots \\ 0 & \ldots & \ldots & 0 & b_{n-2} & c_{n-1} & 0 \\ h_1 & h_2 & h_3 & \ldots & h_{n-2} & h_{n-1} & c_n \end{bmatrix} \begin{bmatrix} z_1 \\ z_2 \\ z_3 \\ \vdots \\ z_{n-2} \\ z_{n-1} \\ z_n \end{bmatrix} = \begin{bmatrix} f_1 \\ f_2 \\ f_3 \\ \vdots \\ f_{n-2} \\ f_{n-1} \\ f_n \end{bmatrix}. \tag{2.4}$$

• The solution vector, $\boldsymbol{x} = [x_1, x_2, ..., x_n]^T$ of the system (1.5) satisfies:

$$\begin{bmatrix} 1 & y_1 & 0 & \dots & 0 & v_1 \\ 0 & 1 & y_2 & \ddots & \vdots & v_2 \\ \vdots & \ddots & \ddots & \ddots & 0 & \vdots \\ \vdots & & \ddots & 1 & y_{n-2} & v_{n-2} \\ \vdots & & & \ddots & 1 & v_{n-1} \\ 0 & \dots & & \dots & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ \vdots \\ x_{n-2} \\ x_{n-1} \\ x_n \end{bmatrix} = \begin{bmatrix} z_1 \\ z_2 \\ \vdots \\ \vdots \\ z_{n-2} \\ z_{n-1} \\ z_n \end{bmatrix}. \tag{2.5}$$

Hence using (2.4) and (2.5), we get

$$\begin{bmatrix} c_1 & 0 & \dots & & \dots & \dots & 0 \\ b_1 & c_2 & \ddots & & & & \vdots \\ 0 & b_2 & c_3 & \ddots & & & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & & \vdots \\ \vdots & & \ddots & b_{n-3} & c_{n-2} & \ddots & \vdots \\ 0 & \dots & \dots & 0 & b_{n-2} & c_{n-1} & 0 \\ h_1 & h_2 & h_3 & \dots & h_{n-2} & h_{n-1} & c_n \end{bmatrix} \begin{bmatrix} 1 & y_1 & 0 & \dots & 0 & v_1 \\ 0 & 1 & y_2 & \ddots & \vdots & v_2 \\ \vdots & \ddots & \ddots & \ddots & 0 & \vdots \\ \vdots & & \ddots & 1 & y_{n-2} & v_{n-2} \\ \vdots & & & \ddots & 1 & v_{n-1} \\ 0 & \dots & & \dots & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ \vdots \\ x_{n-2} \\ x_{n-1} \\ x_n \end{bmatrix} = \begin{bmatrix} f_1 \\ f_2 \\ \vdots \\ \vdots \\ f_{n-2} \\ f_{n-1} \\ f_n \end{bmatrix}. \tag{2.6}$$

From (1.5) and (2.6), we obtain the Crout LU factorization [1] of the matrix in (1.5) in the form:

$$T_n = L_1 U_1$$

where

$$L_1 = \begin{bmatrix} c_1 & 0 & \dots & & \dots & \dots & 0 \\ b_1 & c_2 & \ddots & & & & \vdots \\ 0 & b_2 & c_3 & \ddots & & & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & & \vdots \\ \vdots & & \ddots & b_{n-3} & c_{n-2} & \ddots & \vdots \\ 0 & \dots & \dots & 0 & b_{n-2} & c_{n-1} & 0 \\ h_1 & h_2 & h_3 & \dots & h_{n-2} & h_{n-1} & c_n \end{bmatrix} \text{ and } U_1 = \begin{bmatrix} 1 & y_1 & 0 & \dots & 0 & v_1 \\ 0 & 1 & y_2 & \ddots & \vdots & v_2 \\ \vdots & \ddots & \ddots & \ddots & 0 & \vdots \\ \vdots & & \ddots & 1 & y_{n-2} & v_{n-2} \\ \vdots & & & \ddots & 1 & v_{n-1} \\ 0 & \dots & & \dots & 0 & 1 \end{bmatrix}. \tag{2.7}$$

The Doolittle LU factorization of the coefficient matrix in (1.5) is given by:

$$T_n = L_2 U_2$$

where

$$L_2 = \begin{bmatrix} 1 & 0 & \dots & & \dots & \dots & 0 \\ \frac{b_1}{c_1} & 1 & \ddots & & & & \vdots \\ 0 & \frac{b_2}{c_2} & 1 & \ddots & & & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & & \vdots \\ \vdots & & \ddots & \frac{b_{n-3}}{c_{n-3}} & 1 & \ddots & \vdots \\ 0 & \dots & \dots & 0 & \frac{b_{n-2}}{c_{n-2}} & 1 & 0 \\ \frac{h_1}{c_1} & \frac{h_2}{c_2} & \frac{h_3}{c_3} & \dots & \frac{h_{n-2}}{c_{n-2}} & \frac{h_{n-1}}{c_{n-1}} & 1 \end{bmatrix} \text{ and } U_2 = \begin{bmatrix} c_1 & a_1 & 0 & \dots & 0 & c_1 v_1 \\ 0 & c_2 & a_2 & \ddots & \vdots & c_2 v_2 \\ \vdots & \ddots & \ddots & \ddots & 0 & \vdots \\ \vdots & & \ddots & c_{n-2} & a_{n-2} & c_{n-2} v_{n-2} \\ \vdots & & & \ddots & c_{n-1} & c_{n-1} v_{n-1} \\ 0 & \dots & & \dots & 0 & c_n \end{bmatrix}. \tag{2.8}$$

- The values $v_i,\ i = 1, 2, ..., n-1$ in (2.3) satisfy:

$$\begin{bmatrix} c_1 & 0 & \dots & & \dots & 0 \\ b_1 & c_2 & \ddots & & & \vdots \\ 0 & b_2 & c_3 & \ddots & & \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & b_{n-3} & c_{n-2} & 0 \\ 0 & \dots & \dots & 0 & b_{n-2} & c_{n-1} \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ v_3 \\ \vdots \\ v_{n-2} \\ v_{n-1} \end{bmatrix} = \begin{bmatrix} p_1 \\ p_2 \\ p_3 \\ \vdots \\ p_{n-2} \\ a_{n-1} \end{bmatrix} \tag{2.9}$$

- The values $h_i,\ i = 1, 2, ..., n-1$ in (2.3) satisfy:

$$\begin{bmatrix} 1 & 0 & \dots & & \dots & 0 \\ y_1 & 1 & \ddots & & & \vdots \\ 0 & y_2 & 1 & \ddots & & \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & y_{n-3} & 1 & 0 \\ 0 & \dots & \dots & 0 & y_{n-2} & 1 \end{bmatrix} \begin{bmatrix} h_1 \\ h_2 \\ h_3 \\ \vdots \\ h_{n-2} \\ h_{n-1} \end{bmatrix} = \begin{bmatrix} q_1 \\ q_2 \\ q_3 \\ \vdots \\ q_{n-2} \\ b_{n-1} \end{bmatrix}. \tag{2.10}$$

Armed with (2.7)-(2.10) we may write the partitioned forms:

$$\left[\begin{array}{cccccc|c} d_1 & a_1 & 0 & \dots & ... & 0 & p_1 \\ b_1 & d_2 & a_2 & \ddots & & \vdots & p_2 \\ 0 & b_2 & d_3 & a_3 & \ddots & \vdots & p_3 \\ \vdots & \ddots & \ddots & \ddots & \ddots & 0 & \vdots \\ \vdots & & \ddots & b_{n-3} & d_{n-2} & a_{n-2} & p_{n-2} \\ 0 & \dots & \dots & 0 & b_{n-2} & d_{n-1} & a_{n-1} \\ \hline q_1 & q_2 & q_3 & \dots & q_{n-2} & b_{n-1} & d_n \end{array}\right]$$

$$= \left[\begin{array}{cccccc|c} c_1 & 0 & \dots & & \dots & 0 & 0 \\ b_1 & c_2 & \ddots & & & \vdots & \vdots \\ 0 & b_2 & c_3 & \ddots & & & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots & \vdots \\ \vdots & & \ddots & b_{n-3} & c_{n-2} & 0 & \vdots \\ 0 & \dots & \dots & 0 & b_{n-2} & c_{n-1} & 0 \\ \hline h_1 & h_2 & h_3 & \dots & h_{n-2} & h_{n-1} & c_n \end{array}\right] \left[\begin{array}{cccccc|c} 1 & y_1 & 0 & ... & ... & 0 & v_1 \\ 0 & 1 & y_2 & \ddots & & \vdots & v_2 \\ \vdots & \ddots & 1 & y_3 & \ddots & \vdots & v_3 \\ \vdots & & \ddots & \ddots & \ddots & 0 & \vdots \\ \vdots & & & \ddots & 1 & y_{n-2} & v_{n-2} \\ 0 & \dots & \dots & \dots & 0 & 1 & v_{n-1} \\ \hline 0 & \dots & \dots & & \dots & 0 & 1 \end{array}\right], \tag{2.11}$$

and

$$\left[\begin{array}{cccccc|c} d_1 & a_1 & 0 & \dots & ... & 0 & p_1 \\ b_1 & d_2 & a_2 & \ddots & & \vdots & p_2 \\ 0 & b_2 & d_3 & a_3 & \ddots & \vdots & p_3 \\ \vdots & \ddots & \ddots & \ddots & \ddots & 0 & \vdots \\ \vdots & & \ddots & b_{n-3} & d_{n-2} & a_{n-2} & p_{n-2} \\ 0 & \dots & \dots & 0 & b_{n-2} & d_{n-1} & a_{n-1} \\ \hline q_1 & q_2 & q_3 & \dots & q_{n-2} & b_{n-1} & d_n \end{array}\right] =$$

$$= \left[\begin{array}{cccccc|c} 1 & 0 & \dots & ... & ... & 0 & 0 \\ \frac{b_1}{c_1} & 1 & \ddots & & & \vdots & \vdots \\ 0 & \frac{b_2}{c_2} & 1 & \ddots & & & \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots & \vdots \\ \vdots & & \ddots & \frac{b_{n-3}}{c_{n-3}} & 1 & 0 & \vdots \\ 0 & \dots & \dots & 0 & \frac{b_{n-2}}{c_{n-2}} & 1 & 0 \\ \hline h_1 & h_2 & h_3 & \dots & h_{n-2} & h_{n-1} & 1 \end{array}\right] \left[\begin{array}{cccccc|c} c_1 & a_1 & 0 & ... & ... & 0 & v_1 \\ 0 & c_2 & a_2 & \ddots & & \vdots & v_2 \\ \vdots & \ddots & c_3 & a_3 & \ddots & \vdots & v_3 \\ \vdots & & \ddots & \ddots & \ddots & 0 & \vdots \\ \vdots & & & \ddots & c_{n-2} & a_{n-2} & v_{n-2} \\ 0 & \dots & \dots & \dots & 0 & c_{n-1} & v_{n-1} \\ \hline 0 & \dots & \dots & & \dots & 0 & c_n \end{array}\right]. \tag{2.12}$$

3 Algorithms for Solving Doubly Bordered Tridiagonal Linear Systems of Type (B)

In this section, we are going to consider the construction of new algorithms for solving doubly bordered tridiagonal linear systems of type (B) via transformation. For this purpose it is convenient to introduce the vector **e** $= [e_1, e_2, ..., e_n]$ whose components $e_n, e_{n-1}, ..., e_2$ are given by:

$$e_i = \begin{cases} d_n, & \text{for } i = n \\ d_i - \frac{a_i\, b_i}{e_{i+1}}, & \text{for } i = n-1, n-2, ..., 2. \end{cases} \tag{3.1}$$

The system (1.6) can be completely described by the augmented matrix, H given by:

$$H := [\hat{T}_n, \boldsymbol{g}] = \left[\begin{array}{ccccccc|c} d_1 & a_1 & p_{n-2} & \dots & p_3 & p_2 & p_1 & g_1 \\ b_1 & d_2 & a_2 & 0 & \dots & \dots & 0 & g_2 \\ q_{n-2} & b_2 & d_3 & a_3 & \ddots & & \vdots & g_3 \\ \vdots & 0 & \ddots & \ddots & \ddots & \ddots & \vdots & \vdots \\ q_3 & \vdots & \ddots & b_{n-3} & d_{n-2} & a_{n-2} & 0 & g_{n-2} \\ q_2 & \vdots & & \ddots & b_{n-2} & d_{n-1} & a_{n-1} & g_{n-1} \\ q_1 & 0 & \dots & \dots & 0 & b_{n-1} & d_n & g_n \end{array}\right]. \tag{3.2}$$

The matrix H contains all the information about the system (1.6) that is necessary to determine its solution, but in a compact form. Let $\mathbf{R}_i$ denotes the i-th row of the matrix H. Performing the following row operations on H, in the same order:

$\mathbf{R}_n \longleftarrow \mathbf{R}_n/e_n$
For $i = n-1, n-2, ..., 2$ do
    $\mathbf{R}_i \longleftarrow \mathbf{R}_i - a_i\mathbf{R}_{i+1}$
    $\mathbf{R}_i \longleftarrow \mathbf{R}_i/e_i$
End do.
For $j = 1, 2, ..., n-1$ do
    $\mathbf{R}_1 \longleftarrow \mathbf{R}_1 - v_j\mathbf{R}_{n-j+1}$
End do.
$\mathbf{R}_1 \longleftarrow \mathbf{R}_1/e_1.$
Then we have the resulting transformed linear system of the form:

$$\left[\begin{array}{ccccccc|c} 1 & 0 & \dots & & \dots & \dots & 0 & Z_1 \\ h_{n-1} & 1 & \ddots & & & & \vdots & Z_2 \\ h_{n-2} & Y_2 & 1 & \ddots & & & \vdots & Z_3 \\ \vdots & 0 & \ddots & \ddots & \ddots & & \vdots & \vdots \\ h_3 & \vdots & \ddots & Y_{n-3} & 1 & \ddots & \vdots & Z_{n-2} \\ h_2 & \vdots & & \ddots & Y_{n-2} & 1 & 0 & Z_{n-1} \\ h_1 & 0 & \dots & \dots & 0 & Y_{n-1} & 1 & Z_n \end{array}\right], \tag{3.3}$$

where

$$
\begin{aligned}
& e_n = d_n, \quad h_1 = \frac{q_1}{e_n}, \quad Z_n = \frac{g_n}{e_n}, \quad v_1 = p_1, \\
& \text{For } i = n-1, n-2, ..., 2 \quad \text{do} \\
& \qquad Y_i = \frac{b_i}{e_{i+1}}, \\
& \qquad e_i = d_i - a_i\, Y_i, \\
& \qquad Z_i = \frac{1}{e_i}(g_i - a_i\, Z_{i+1}). \\
& \text{End do.} \\
& \text{For } i = 2, 3, ..., n-2 \quad \text{do} \\
& \qquad h_i = \tfrac{1}{e_{n-i+1}}(q_i - h_{i-1}\, a_{n-i+1}), \\
& \qquad v_i = p_i - Y_{n-i+1}\, v_{i-1}, \\
& \text{End do.} \\
& h_{n-1} = \frac{1}{e_2}(b_1 - h_{n-2}\, a_2), \\
& v_{n-1} = a_1 - Y_2\, v_{n-2}, \\
& e_1 = d_1 - \sum_{r=1}^{n-1} h_r\, v_r, \\
& Z_1 = \frac{1}{e_1}(g_1 - \sum_{r=1}^{n-1} v_r\, Z_{n-r+1}).
\end{aligned}
\tag{3.4}
$$

At this stage, the determinant of the coefficient matrix in (1.6) can be computed using the following computational symbolic algorithm.

Algorithm 3.1. An algorithm for computing the determinant of doubly bordered tridiagonal matrices of type (B).

To compute the determinant of the coefficient matrix in (1.6), we may proceed as follows:

INPUT: Order of the coefficient matrix n and the components, a_i, d_i, b_i, p_i, q_i.
OUTPUT: The determinant of the coefficient matrix in (1.6).
Step 1: Set $e_n = d_n$. If $e_n = 0$ then $e_n = s$ end if.
Step 2: For $i = n-1, n-2, ..., 2$ do
Compute and simplify:
$e_i = d_i - \frac{a_i\, b_i}{e_{i+1}}$. If $e_i = 0$ then $e_i = s$ end if.
End do.
Step 3: Compute e_1 from (3.4).
Step 4: Compute and simplify:
$P(s) = \prod_{r=1}^{n} e_r$.
Step 5: $\det(\hat{T}_n) = P(0)$.

The Algorithm 3.1, will be referred to as **DETDBTRI-B** algorithm.

The reduced transformed system (3.3) is triangular and easy to solve by forward substitution. Consequently, the linear system (1.6) can be solved using the following symbolic algorithm:

Algorithm 3.2. Symbolic version algorithm for solving doubly bordered tridiagonal linear system of type (B).

To solve the linear system of the form (1.6), we may proceed as follows:

INPUT: Order of the matrix n and the components, $a_i,\ d_i,\ b_i,\ g_i,\ p_i,\ q_i$.
OUTPUT: The determinant of the coefficient matrix in (1.6) and the solution vector $\boldsymbol{x} = [x_1, x_2, \ldots, x_n]^T$.
Step 1: Set $e_n = d_n$. If $e_n = 0$ then $e_n = s$ end if. $h_1 = \frac{q_1}{e_n}$, $Z_1 = \frac{g_n}{e_n}$, and $v_1 = p_1$.
Step 2: For $i = n-1, n-2, ..., 2$ do
Compute and simplify:
$Y_i = \dfrac{b_i}{e_{i+1}}$,
$e_i = d_i - a_i\, Y_i$. If $e_i = 0$ then $e_i = s$ end if.
$Z_i = \dfrac{1}{e_i}(g_i - a_i\, Z_{i+1})$.
End do.
Step 3: For $i = 2, 3, ..., n-2$ do
Compute and simplify:
$h_i = \frac{1}{e_{n-i+1}}(q_i - a_{n-i+1}\, h_{i-1})$,
$v_i = p_i - Y_{n-i+1}\, v_{i-1}$,
End do.
Step 4: Set $h_{n-1} = \frac{1}{e_2}(b_1 - a_2\, h_{n-2})$,
$v_{n-1} = a_1 - Y_2\, v_{n-2}$.
Step 5: Set $e_1 = d_1 - \sum\limits_{r=1}^{n-1} h_r\, v_r$. If $e_1 = 0$ then $e_1 = s$ end if.
$Z_1 = \frac{1}{e_1}(g_1 - \sum\limits_{r=1}^{n-1} v_r\, Z_{n-r+1})$.
Step 6: Use the **DETDBTRI-B** algorithm to check the non-singularity of the coefficient matrix of the system (1.6).
Step 7: If the determinant of the coefficient matrix in (1.6) equals zero, then Exiterror('No solutions') end if.
Step 8: Compute the solution vector $\boldsymbol{x} = [x_1, x_2, \ldots, x_n]^T$ using
$x_1 = Z_1$
$x_2 = Z_2 - h_{n-1}\, x_1$
Step 9: For $i = 3, 4, ..., n$ do
$x_i = Z_i - Y_{i-1}\, x_{i-1} - h_{n-i+1} x_1$
End do.

Step 9: Substitute $s = 0$ in all expressions of the solution vector $x_i, i = 1, 2, ..., n$. The Algorithm 3.2, will be referred to as **TRANSDBTRI-BI** algorithm. The cost of this algorithm is exactly the same as the cost of the **TRANSDBTRI-AI** algorithm.

Similar to the situation with the doubly bordered tridiagonal linear systems of type (A), the following points are worth to mention:

• The solution vector, $\boldsymbol{x} = [x_1, x_2, ..., x_n]^T$ of the system (1.6) satisfies:

$$\begin{bmatrix} 1 & 0 & \dots & & \dots & \dots & 0 \\ h_{n-1} & 1 & \ddots & & & & \vdots \\ h_{n-2} & Y_2 & 1 & \ddots & & & \vdots \\ \vdots & 0 & \ddots & \ddots & \ddots & & \vdots \\ h_3 & \vdots & \ddots & Y_{n-3} & 1 & \ddots & \vdots \\ h_2 & \vdots & & \ddots & Y_{n-2} & 1 & 0 \\ h_1 & 0 & \dots & \dots & 0 & Y_{n-1} & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_{n-2} \\ x_{n-1} \\ x_n \end{bmatrix} = \begin{bmatrix} Z_1 \\ Z_2 \\ Z_3 \\ \vdots \\ Z_{n-2} \\ Z_{n-1} \\ Z_n \end{bmatrix}. \tag{3.5}$$

• The values $Z_i,\ i = 1, 2, ..., n$ in (3.4) satisfy:

$$\begin{bmatrix} e_1 & v_{n-1} & \dots & \dots & v_2 & v_1 \\ 0 & e_2 & a_2 & 0 & \dots & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & e_{n-2} & a_{n-2} & 0 \\ \vdots & & & \ddots & e_{n-1} & a_{n-1} \\ 0 & \dots & & \dots & 0 & e_n \end{bmatrix} \begin{bmatrix} Z_1 \\ Z_2 \\ Z_3 \\ \vdots \\ Z_{n-2} \\ Z_{n-1} \\ Z_n \end{bmatrix} = \begin{bmatrix} g_1 \\ g_2 \\ g_3 \\ \vdots \\ g_{n-2} \\ g_{n-1} \\ g_n \end{bmatrix}. \tag{3.6}$$

Hence using (3.5) and (3.6), we get

$$\begin{bmatrix} e_1 & v_{n-1} & \dots & \dots & v_2 & v_1 \\ 0 & e_2 & a_2 & 0 & \dots & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & e_{n-2} & a_{n-2} & 0 \\ \vdots & & & \ddots & e_{n-1} & a_{n-1} \\ 0 & \dots & & \dots & 0 & e_n \end{bmatrix} \begin{bmatrix} 1 & 0 & \dots & & \dots & \dots & 0 \\ h_{n-1} & 1 & \ddots & & & & \vdots \\ h_{n-2} & Y_2 & 1 & \ddots & & & \vdots \\ \vdots & 0 & \ddots & \ddots & \ddots & & \vdots \\ h_3 & \vdots & \ddots & Y_{n-3} & 1 & \ddots & \vdots \\ h_2 & \vdots & & \ddots & Y_{n-2} & 1 & 0 \\ h_1 & 0 & \dots & \dots & 0 & Y_{n-1} & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_{n-2} \\ x_{n-1} \\ x_n \end{bmatrix} = \begin{bmatrix} g_1 \\ g_2 \\ g_3 \\ \vdots \\ g_{n-2} \\ g_{n-1} \\ g_n \end{bmatrix}. \tag{3.7}$$

From (1.6) and (3.7), we obtain the Doolittle UL factorization [1] of the matrix in (1.6) in the form:

$$\hat{T}_n = U_1 L_1$$

where

$$U_1 = \begin{bmatrix} e_1 & v_{n-1} & \dots & \dots & v_2 & v_1 \\ 0 & e_2 & a_2 & 0 & \dots & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & e_{n-2} & a_{n-2} & 0 \\ \vdots & & & \ddots & e_{n-1} & a_{n-1} \\ 0 & \dots & & \dots & 0 & e_n \end{bmatrix} \quad \text{and} \quad L_1 = \begin{bmatrix} 1 & 0 & \dots & & \dots & \dots & 0 \\ h_{n-1} & 1 & \ddots & & & & \vdots \\ h_{n-2} & Y_2 & 1 & \ddots & & & \vdots \\ \vdots & 0 & \ddots & \ddots & \ddots & & \vdots \\ h_3 & \vdots & \ddots & Y_{n-3} & 1 & \ddots & \vdots \\ h_2 & \vdots & & \ddots & Y_{n-2} & 1 & 0 \\ h_1 & 0 & \dots & \dots & 0 & Y_{n-1} & 1 \end{bmatrix}. \tag{3.8}$$

The Crout UL factorization of the coefficient matrix in (1.6) is given by:

$$\hat{T}_n = U_2 L_2$$

where

$$U_2 = \begin{bmatrix} 1 & \frac{v_{n-1}}{e_2} & \dots & \dots & \frac{v_2}{e_{n-1}} & \frac{v_1}{e_n} \\ 0 & 1 & \frac{a_2}{e_3} & 0 & \dots & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & 1 & \frac{a_{n-2}}{e_{n-1}} & 0 \\ \vdots & & & \ddots & 1 & \frac{a_{n-1}}{e_n} \\ 0 & \dots & & \dots & 0 & 1 \end{bmatrix} \text{ and } L_2 = \begin{bmatrix} e_1 & 0 & \dots & & \dots & \dots & 0 \\ e_2h_{n-1} & e_2 & \ddots & & & & \vdots \\ e_3h_{n-2} & b_2 & e_3 & \ddots & & & \vdots \\ \vdots & 0 & \ddots & \ddots & \ddots & & \vdots \\ e_{n-2}h_3 & \vdots & \ddots & b_{n-3} & e_{n-2} & \ddots & \vdots \\ e_{n-1}h_2 & \vdots & & \ddots & b_{n-2} & e_{n-1} & 0 \\ e_nh_1 & 0 & \dots & \dots & 0 & b_{n-1} & e_n \end{bmatrix}. \tag{3.9}$$

• The values $v_i,\ i = 1, 2, ..., n-1$ in (3.4) satisfy:

$$\begin{bmatrix} 1 & 0 & \dots & & \dots & 0 \\ Y_{n-1} & 1 & \ddots & & & \vdots \\ 0 & Y_{n-2} & 1 & \ddots & & \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & Y_3 & 1 & 0 \\ 0 & \dots & \dots & 0 & Y_2 & 1 \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ v_3 \\ \vdots \\ v_{n-2} \\ v_{n-1} \end{bmatrix} = \begin{bmatrix} p_1 \\ p_2 \\ p_3 \\ \vdots \\ p_{n-2} \\ a_1 \end{bmatrix} \tag{3.10}$$

• The values $h_i,\ i = 1, 2, ..., n-1$ in (3.4) satisfy:

$$\begin{bmatrix} e_n & 0 & \dots & & \dots & 0 \\ a_{n-1} & e_{n-1} & \ddots & & & \vdots \\ 0 & a_{n-2} & 1 & \ddots & & \\ \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & & \ddots & a_3 & e_3 & 0 \\ 0 & \dots & \dots & 0 & a_2 & e_2 \end{bmatrix} \begin{bmatrix} h_1 \\ h_2 \\ h_3 \\ \vdots \\ h_{n-2} \\ h_{n-1} \end{bmatrix} = \begin{bmatrix} q_1 \\ q_2 \\ q_3 \\ \vdots \\ q_{n-2} \\ b_1 \end{bmatrix}. \tag{3.11}$$

Armed with (3.8)-(3.11) we have the partitioned form:

$$
\left[\begin{array}{c|cccccc}
d_1 & a_1 & p_{n-2} & \dots & p_3 & p_2 & p_1 \\ \hline
b_1 & d_2 & a_2 & 0 & \dots & \dots & 0 \\
q_{n-2} & b_2 & d_3 & a_3 & \ddots & & \vdots \\
\vdots & 0 & \ddots & \ddots & \ddots & \ddots & \vdots \\
q_3 & \vdots & \ddots & b_{n-3} & d_{n-2} & a_{n-2} & 0 \\
q_2 & \vdots & & \ddots & b_{n-2} & d_{n-1} & a_{n-1} \\
q_1 & 0 & \dots & \dots & 0 & b_{n-1} & d_n
\end{array}\right]
$$

$$
= \left[\begin{array}{c|cccccc}
e_1 & v_{n-1} & v_{n-2} & v_{n-3} & \dots & v_2 & v_1 \\ \hline
0 & e_2 & a_2 & 0 & \dots & \dots & 0 \\
\vdots & 0 & e_3 & a_3 & \ddots & & \vdots \\
\vdots & \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\
\vdots & & & \ddots & e_{n-2} & a_{n-2} & 0 \\
\vdots & \vdots & & & \ddots & e_{n-1} & a_{n-1} \\
0 & 0 & & \dots & \dots & 0 & e_n
\end{array}\right]
\left[\begin{array}{c|ccccc}
1 & 0 & \dots & \dots & \dots & 0 \\ \hline
h_{n-1} & 1 & 0 & \dots & \dots & 0 \\
h_{n-2} & Y_2 & 1 & \ddots & & \vdots \\
\vdots & 0 & \ddots & \ddots & \ddots & \vdots \\
h_2 & \vdots & \ddots & Y_{n-2} & 1 & 0 \\
h_1 & 0 & \dots & 0 & Y_{n-1} & 1
\end{array}\right],
\tag{3.12}
$$

and

$$
\left[\begin{array}{c|cccccc}
d_1 & a_1 & p_{n-2} & \dots & p_3 & p_2 & p_1 \\ \hline
b_1 & d_2 & a_2 & 0 & \dots & \dots & 0 \\
q_{n-2} & b_2 & d_3 & a_3 & \ddots & & \vdots \\
\vdots & 0 & \ddots & \ddots & \ddots & \ddots & \vdots \\
q_3 & \vdots & \ddots & b_{n-3} & d_{n-2} & a_{n-2} & 0 \\
q_2 & \vdots & & \ddots & b_{n-2} & d_{n-1} & a_{n-1} \\
q_1 & 0 & \dots & \dots & 0 & b_{n-1} & d_n
\end{array}\right]
$$

$$
= \left[\begin{array}{c|cccccc}
1 & \frac{v_{n-1}}{e_2} & \frac{v_{n-2}}{e_3} & \frac{v_{n-3}}{e_4} & \dots & \frac{v_2}{e_{n-1}} & \frac{v_1}{e_n} \\ \hline
0 & 1 & \frac{a_2}{e_3} & 0 & \dots & \dots & 0 \\
\vdots & 0 & 1 & \frac{a_3}{e_4} & \ddots & & \vdots \\
\vdots & \vdots & \ddots & \ddots & \ddots & \ddots & \vdots \\
\vdots & & & \ddots & 1 & \frac{a_{n-2}}{e_{n-1}} & 0 \\
\vdots & \vdots & & & \ddots & 1 & \frac{a_{n-1}}{e_n} \\
0 & 0 & & \dots & \dots & 0 & 1
\end{array}\right]
\left[\begin{array}{c|ccccc}
e_1 & 0 & \dots & \dots & \dots & 0 \\ \hline
e_2h_{n-1} & e_2 & 0 & \dots & \dots & 0 \\
e_3h_{n-2} & b_2 & e_3 & \ddots & & \vdots \\
\vdots & 0 & \ddots & \ddots & \ddots & \vdots \\
e_{n-1}h_2 & \vdots & \ddots & b_{n-2} & e_{n-1} & 0 \\
e_nh_1 & 0 & \dots & 0 & b_{n-1} & e_n
\end{array}\right],
\tag{3.13}
$$

4 Computer Program

In this section, we are going to introduce a MAPLE procedure for solving linear systems of doubly bordered tridiagonal types (A) and (B) in (1.5) and (1.6), respectively. The procedure is based on the algorithms **DETDBTRI-A**, **TRANSDBTRI-AII**, **DETDBTRI-B** and **TRANSDBTRI-BI**.

```
#####################################################
restart:
doublytritrans:= proc(d::vector, a::vector, b::vector,q::vector,p::vector,f::vector,
n::posint,indic::nonnegint)
local i,r:
global x,T,detA:
x:= vector(n):
#####################################################
#Apply TRANDBTRI-AII algorithm if indic =1#
if indic=1 then
        if d[1] = 0 then d[1]:=t fi:
        a[1]:=simplify(a[1]/d[1]): f[1]:=simplify(f[1]/d[1]): p[1]:=p[1]/d[1]:
        for i from 2 to n-2 do
            d[i] := simplify(d[i]-a[i-1]*b[i-1]);if d[i] = 0 then d[i] := t; fi:
            a[i] := simplify(a[i]/d[i]);
            p[i] := simplify((p[i] - p[i-1]*b[i-1])/d[i]);
            f[i] := simplify((f[i]-f[i-1]*b[i-1])/d[i]);
            q[i] := simplify(q[i] - q[i-1]*a[i-1]);
        od:
        d[n-1] := simplify(d[n-1]-a[n-2]*b[n-2]);
        if d[n-1] = 0 then d[n-1] := t; fi:
        p[n-1] := simplify((a[n-1] - p[n-2]*b[n-2])/d[n-1]):
        f[n-1] := simplify((f[n-1] - f[n-2]*b[n-2])/d[n-1]):
        q[n-1] := simplify((b[n-1] - q[n-2]*a[n-2])):
        d[n] := simplify(d[n]-sum(q[r]*p[r],r=1..n-1)):
        f[n] := simplify((f[n]-sum(q[r]*f[r],r=1..n-1))/d[n]):
        if d[n] = 0 then d[n] := t; fi:
        #To compute the determinant of the doubly bordered tridiagonal matrix#
        T := simplify(subs(t =0,simplify(product(d[r],r= 1..n)))):
        detA:= eval(T);
        if T = 0 then error("Singular Matrix"); fi;
        # To compute the Solution of the system X. #
        x[n]:=simplify(f[n]);
        x[n-1]:=simplify((f[n-1]-p[n-1]*x[n]));
        for i from n-2 by -1 to 1 do
            x[i]:=simplify((f[i]-a[i]*x[i+1]-p[i]*x[n]));
        od;
        eval(x);
#####################################################
#Apply TRANDBTRI-BI algorithm if indic = 2#
elif indic=2 then
        if d[n] = 0 then d[n]:=t fi;
        g[n]:=simplify(g[n]/d[n]): q[1]:=simplify(q[1]/d[n]):
        for i from n-1 by -1 to 2 do
            b[i] := simplify(b[i]/d[i+1]);
            d[i] := simplify(d[i]-b[i]*a[i]);if d[i] = 0 then d[i] := t; fi:
```

```
            g[i] := simplify((g[i]-g[i+1]*a[i])/d[i]);
        od:
        for i from 2 to n-2 do
            q[i] := simplify((q[i] - q[i-1]*a[n-i+1])/d[n-i+1]);
            p[i] := simplify(p[i] - p[i-1]*b[n-i+1]);
        od:
        q[n-1] := simplify((b[1] - q[n-2]*a[2])/d[2]):
        p[n-1] := simplify(a[1] - p[n-2]*b[2]):
        d[1] := simplify(d[1]-sum(q[r]*p[r],r=1..n-1)):
        if d[1] = 0 then d[1] := t; fi:
        g[1] := simplify((g[1]-sum(p[r]*f[n-r+1],r=1..n-1))/d[1]):
        #To compute the determinant of the doubly bordered tridiagonal matrix#
        T := simplify(subs(t =0,simplify(product(d[r],r= 1..n)))):
        detA:= eval(T);
        if T = 0 then error("Singular Matrix");fi;
        # To compute the Solution of the system X. #
        x[1]:=simplify(g[1]):
        x[2]:=simplify((g[2]-q[n-1]*x[1])):
        for i from 3 to n do
        x[i]:=simplify((g[i]-b[i-1]*x[i-1]-q[n-i+1]*x[1]));
        od:
        eval(x);
elif (indic<>1 and indic<>2) then
        print("Error. The value of indic is out of range"):
fi:
end proc:
```

5 Illustrative Examples

Example 5.1. *[17]. Consider the singly bordered tridiagonal matrix*

$$A = \begin{bmatrix} 2 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & p_1 \\ 1 & 2 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & p_2 \\ 0 & 1 & 2 & 1 & 0 & 0 & 0 & 0 & 0 & p_3 \\ 0 & 0 & 1 & 2 & 1 & 0 & 0 & 0 & 0 & p_4 \\ 0 & 0 & 0 & 1 & 2 & 1 & 0 & 0 & 0 & p_5 \\ 0 & 0 & 0 & 0 & 1 & 2 & 1 & 0 & 0 & p_6 \\ 0 & 0 & 0 & 0 & 0 & 1 & 2 & 1 & 0 & p_7 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 2 & 1 & p_8 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 2 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 2 \end{bmatrix},$$

find $\det(A)$.
Solution: *We have*
$n = 10$, *indic* $= 1$, **a** $= [1, 1, 1, 1, 1, 1, 1, 1, 1]$, **d** $= [2, 2, 2, 2, 2, 2, 2, 2, 2, 2]$, **b** $= [1, 1, 1, 1, 1, 1, 1, 1, 1]$, **p** $= [p_1, p_2, p_3, p_4, p_5, p_6, p_7, p_8, 0]^T$ *and* **q** $= [0, 0, 0, 0, 0, 0, 0, 0, 0]$. *By applying the **DETDBTRI-A** algorithm , we obtain*

- **c**$=[2, \frac{3}{2}, \frac{4}{3}, \frac{5}{4}, \frac{6}{5}, \frac{7}{6}, \frac{8}{7}, \frac{9}{8}, \frac{10}{9}, \frac{11}{10} + \frac{4}{5}p_8 - \frac{7}{10}p_7 + \frac{3}{5}p_6 - \frac{1}{2}p_5 + \frac{2}{5}p_4 - \frac{3}{10}p_3 + \frac{1}{5}p_2 - \frac{1}{10}p_1]$.
- *detA*$=\prod_{i=1}^{10} c_i = 11 + 8p_8 - 7p_7 + 6p_6 - 5p_5 + 4p_4 - 3p_3 + 2p_2 - p_1 = 11 + \sum_{i=1}^{8} (-1)^i \, i \, p_i$.

This result is in complete agreement with the result in [17].

Example 5.2. *Solve the doubly bordered tridiagonal linear system*

$$\begin{bmatrix} 5 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 4 \\ 2 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 12 \\ 0 & -2 & 5 & 2 & 0 & 0 & 0 & 0 & 0 & 7 \\ 0 & 0 & 1 & 2 & 7 & 0 & 0 & 0 & 0 & 2 \\ 0 & 0 & 0 & 3 & 10 & 2 & 0 & 0 & 0 & 5 \\ 0 & 0 & 0 & 0 & 1 & 15 & 3 & 0 & 0 & 3 \\ 0 & 0 & 0 & 0 & 0 & 9 & 2 & 5 & 0 & 6 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 7 & 2 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 3 & 4 & 2 \\ 3 & 2 & -2 & 7 & -6 & 1 & 4 & 5 & 1 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \\ x_8 \\ x_9 \\ x_{10} \end{bmatrix} = \begin{bmatrix} 5 \\ -5 \\ 8 \\ 12 \\ 13 \\ 22 \\ 19 \\ 24 \\ 16 \\ 34 \end{bmatrix}$$

Solution: *We have a doubly bordered tridiagonal linear systems of type (A) for which:*
$n = 10,$ *indic* $= 1$, **a** $= [2, 1, 2, 7, 2, 3, 5, 7, 2]$, **d** $= [5, 1, 5, 2, 10, 15, 2, 1, 4, 1]$, **b** $= [2, -2, 1, 3, 1, 9, 1, 3, 1]$, **p** $= [4, 12, 7, 2, 5, 3, 6, 2, 0]^T$, **q** $= [3, 2, -2, 7, -6, 1, 4, 5, 0]$ *and* **f** $= [5, -5, 8, 12, 13, 22, 19, 24, 16, 34]^T$.
By applying the ***TRANSDBTRI-AII****, we get*

- **c**=$[5, \frac{1}{5}, 15, \frac{28}{15}, \frac{-5}{4}, \frac{83}{5}, \frac{31}{83}, \frac{-384}{31}, \frac{729}{128}, \frac{-161620}{567}]$.
- *detA*=$\prod_{i=1}^{10} c_i = -4363740.$
- *The solution vector is* **x**=$[1, 2, 3, 2, 1, 1, 3, 2, 3, -1]^T$.

By using ***TRANSDBTRI-AI****, we obtained the same results.*

Example 5.3. *Solve the doubly bordered tridiagonal linear system*

$$\begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 5 \\ 1 & 1 & 12 & 0 & 0 & 0 & 0 & 0 & 0 & 3 \\ 0 & 9 & 2 & 5 & 0 & 0 & 0 & 0 & 0 & 2 \\ 0 & 0 & 3 & 15 & 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 2 & 3 & 10 & 0 & 0 & 0 & 5 \\ 0 & 0 & 0 & 0 & 7 & 1 & 2 & 0 & 0 & 2 \\ 0 & 0 & 0 & 0 & 0 & -5 & 2 & 2 & 0 & 7 \\ 0 & 0 & 0 & 0 & 0 & 0 & 2 & 1 & 1 & 12 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 5 & 2 & 4 \\ 3 & 2 & 1 & 7 & 5 & -2 & 4 & 2 & 1 & 5 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \\ x_8 \\ x_9 \\ x_{10} \end{bmatrix} = \begin{bmatrix} 6 \\ 16 \\ 14 \\ 35 \\ 2 \\ 8 \\ 12 \\ 15 \\ 10 \\ 33 \end{bmatrix}$$

Solution: *We have a doubly bordered tridiagonal linear systems of type (A) for which:*
$n = 10,$ *indic* $= 1$, **a** $= [1, 12, 5, 1, 10, 2, 2, 1, 4]$, **d** $= [1, 1, 2, 15, 3, 1, 2, 1, 2, 5]$, **b** $= [1, 9, 3, 2, 7, -5, 2, 5, 1]$, **p** $= [5, 3, 2, 1, 5, 2, 7, 12, 0]^T$, **q** $= [3, 2, 1, 7, 5, -2, 4, 2, 0]$ *and* **f** $= [6, 16, 14, 35, 2, 8, 12, 15, 10, 33]^T$.
By applying the ***TRANSDBTRI-AII****, we have*

- **c**=$[1, s, 2\,\frac{s-54}{s}, \frac{15}{2}\,\frac{s-108}{s-54}, \frac{1}{15}\,\frac{41\,s-4644}{s-108}, -\frac{1009\,s-108756}{41\,s-4644}, 24\,\frac{67\,s-7128}{1009\,s-108756}, -\frac{1}{6}\,\frac{607\,s-65988}{67\,s-7128},$
$8\,\frac{403\,s-43227}{607\,s-65988}, -\frac{1}{8}\,\frac{80797\,s-8813606}{403\,s-43227}]$.
- *detA*=$(\prod_{i=1}^{10} c_i)_{s=0} = (-323188 * s + 35254424)_{s=0} = 35254424.$
- *The solution vector is*
x $= [1, 0, 1, 2, 1, -1, 0, 0, 3, 1]^T$.

Example 5.4. *Solve the doubly bordered tridiagonal linear system*

$$\begin{bmatrix} 5 & 2 & 2 & 6 & 3 & 5 & 2 & 7 & 12 & 4 \\ 2 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 5 & -2 & 5 & 2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 4 & 0 & 1 & 2 & 7 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 3 & 10 & 2 & 0 & 0 & 0 & 0 \\ -6 & 0 & 0 & 0 & 1 & 15 & 3 & 0 & 0 & 0 \\ 7 & 0 & 0 & 0 & 0 & 9 & 2 & 5 & 0 & 0 \\ -2 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 7 & 0 \\ 2 & 0 & 0 & 0 & 0 & 0 & 0 & 3 & 1 & 1 \\ 3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \\ x_8 \\ x_9 \\ x_{10} \end{bmatrix} = \begin{bmatrix} 34 \\ 5 \\ 4 \\ 3 \\ 0 \\ 18 \\ 32 \\ 3 \\ 9 \\ 4 \end{bmatrix}$$

Solution: *We have a doubly bordered tridiagonal linear systems of type (B) for which:* $n = 10$, *indic* $= 2$, **a** $= [2, 1, 2, 7, 2, 3, 5, 7, 1]$, **d** $= [5, 1, 5, 2, 10, 15, 2, 1, 1, 1]$, **b** $= [2, -2, 1, 3, 1, 9, 1, 3, 1]$, **p** $= [4, 12, 7, 2, 5, 3, 6, 2, 0]^T$, **q** $= [3, 2, -2, 7, -6, 1, 4, 5, 0]$ *and* **g** $= [34, 5, 4, 3, 0, 18, 32, 3, 9, 4]^T$. *By applying the* ***TRANSDBTRI-BI****, we have*

- **e**=$[\frac{11}{3} * \frac{(104*s-1637)}{(32*s+43)}, 3 * \frac{(32*s+43)}{(88*s+107)}, \frac{(88*s+107)}{(4*s+11)}, \frac{-1}{2} * \frac{(4*s+11)}{(17*s+13)}, \frac{14}{3} * \frac{(17*s+13)}{(8*s+7)}, 3 * \frac{(8*s+7)}{(s+14)}, -3 * \frac{(s+14)}{(s-21)}, \frac{(s-21)}{s}, s, 1]$.
- *detT*=$(\prod_{i=1}^{10} e_i)_{s=0} = (24024 * s - 378147)_{s=0} = -378147$.
- *The solution vector is*
  x $= [1, 2, 1, -1, 0, 1, 3, 2, 0, 1]^T$.

6 Conclusions

The current article focuses on solving doubly bordered tridiagonal linear systems of equations via transformation. It investigates numeric and symbolic algorithms for solving such systems. MAPLE was used to simulate the problem considered. If n is large (n>500), it may be preferable to use iterative methods for solving such systems. Consequently, the efficiency of the new methods in this paper should be compared with existing methods such as static and dynamic tuning method or the GPU accelerated tridiagonal solver. This will be the subject of the next paper.

Acknowledgment

The authors are extremely thankful to the editorial team of SDI and anonymous referees for their useful comments and suggestions that have helped in improving the readability of this paper.

Competing interests

The authors declare that they have no competing interests.

References

[1] Burden R.L, Faires J.D. Numerical Analysis. Seventh ed., Books & Cole Publishing, Pacific Grove, CA. 2001.

[2] Demmel J.W. Applied Numerical Linear Algebra. SIAM, Philadelphia. 1997.

[3] El-Mikkawy M.E.A. A fast algorithm for evaluating nth order tri-diagonal determinants. J. Comput. Appl. Math. 2004;166:581-584.

[4] El-Mikkawy M.E.A, Karawia A. Inversion of general tridiagonal matrices. Appl. Math. Lett. 2006;19:712-720.

[5] El-Mikkawy M.E.A. A note on a three-term recurrence for a tridiagonal matrix. Appl. Math. Comput. 2003;139:503-511.

[6] Akin H. On 1D reversible cellular automata with reflective boundary over the prime field of order p. Internat. J. Modern Phys. C. 2012;23(1):1-13.

[7] Chant TF, Resascot DC. Generalized deflated block-elimination. SIAM J. Numer. Anal. 1986;23:913-924.

[8] El-Mikkawy M.E.A. An algorithm for solving tridiagonal systems. J. Inst. Math. Comput. Sci. Comput. Sci. Ser. 1991;4(2):205-210.

[9] El-Mikkawy M.E.A. On the inverse of a general tridiagonal matrix. Appl. Math. Comput. 2004;150:669-679.

[10] El-Mikkawy M.E.A. A new computational algorithm for solving periodic tri-diagonal linear systems. Appl. Math. Comput. 2005;161:691-696.

[11] El-Mikkawy M.E.A, Rahmo E. A new recursive algorithm for inverting general tridiagonal and anti-tridiagonal matrices. Appl. Math. Comput. 2008;204:368-372.

[12] El-Mikkawy M.E.A, Rahmo E. A new recursive algorithm for inverting general periodic pentadiagonal and anti-pentadiagonal matrices. Appl. Math. Comput. 2009;207:164-170.

[13] El-Mikkawy M.E.A. A generalized symbolic Thomas algorithm. Applied Mathematics. 2012;3(4):342-345. doi: 10.4236/am.2012.34052.

[14] Kavcic A, Moura JMF. Matrices with banded inverses: Inversion algorithms and factorization of Gauss-Markov processes, IEEE Trans. Inform. Theory. 2000;46(4):1495-1509.

[15] Li H.B, Huang T.Z, Liu X.P, Li H. On the inverses of general tridiagonal matrices. Linear Algebra Appl. 2010;433:965-983.

[16] Shapiro L.W. Positive definite matrices and catalan numbers. revisited, Proc. Amer. Math. Soc. 1984;90:488-496.

[17] Sogabe T. On a two-term recurrence for the determinant of a general matrix. Appl. Math. Comput. 2007;187:785-788.

[18] Sugimoto T. On an inverse formula of a tridiagonal matrix. Operators and Matrices. 2012;6(3):465-480. doi:10.7153/oam-06-30.

[19] Usmani R. Inversion of a tridiagonal Jacobi matrix. Linear Algebra Appl. 1994;212/213:413-414.

[20] Akbudak K, Kayaaslan E, Aykanat C. Analyzing and enhancing OSKI for sparse matrix-vector multiplication. Available on-line at www.prace-ri.eu

[21] Amodio P, Gladwelly I, Romanazzi G. Numerical solution of general bordered ABD linear systems by cyclic reduction. J. Numer. Anal. Ind. Appl. Math. 2006;1(1):5-12.

[22] Doedel E. Numerical analysis and control of bifurcation problems (I) Bifurcation in finite dimensions. Internat. J. Bifur. Chaos Appl. Sci. Engrg. 1991;1(3):493-520.

[23] Duff I.S, Scott JA. Stabilized bordered block diagonal forms for parallel sparse solvers. RAL-TR-2004-006. Available on-line http://www.numerical.rl.ac.uk/reports/reports.html.

[24] da Fonseca C.M. A note on the inversion of acyclic matrices. Int. J. Pure Appl. Math. 2006;31(3):307-317.

[25] Karawia A.A. New Symbolic Algorithms for Solving A General Bordered Tridiagonal Linear System. (2013). avaliable at http://arxiv.org/pdf/1303.0738v1.pdf.

[26] Martin A, Boyd I.D. Variant of the Thomas Algorithm for opposite-bordered tridiagonal systems of equations. Int. J. Numer. Meth. Biomed. Engng. 2010;26:752-759.

[27] Pajic S. Power system state estimation and contingency constrained optimal power flow a numerically robust implementation. Ph,D. Thesis. Worcester-Polytechnic Institute. 2007.

[28] Rashidinia J, Jalilian R. Non-Polynomial Spline Solution of Fourth-Order Obstacle Boundary-Value Problems. World Academy of Science, Engineering and Technology. 28 2009.

[29] Udala A, Reedera R, Velmrea E, Harrisonb P. Comparison of methods for solving the Schrodinger equation for multiquantum well heterostructure applications. Proc. Estonian Acad. Sci. Eng. 2006;12:246-261.

[30] Xing-Bo W. A new algorithm with its scilab implementation for solution of bordered tridiagonal linear equations. Open-source Software for Scientific Computation (OSSC), 2009 IEEE International Workshop on Computing & Processing (Hardware/Software). 2009:11-14.

Peer-review history:
The peer review history for this paper can be accessed here (Please copy paste the total link in your browser address bar)
www.sciencedomain.org/review-history.php?iid=448&id=6&aid=3854