Journal of Applied Mathematics and Physics, 2015, 3, 862-873
Published Online July 2015 in SciRes. http://www.scirp.org/journal/jamp
http://dx.doi.org/10.4236/jamp.2015.37107

Scientific Research Publishing

New Algorithms for Solving Bordered *k*-Tridiagonal Linear Systems

Moawwad El-Mikkawy*, Faiz Atlan

Mathematics Department, Faculty of Science, Mansoura University, Mansoura, Egypt
Email: *m_elmikkawy@yahoo.com, faizatlan11@yahoo.com

Received 24 May 2015; accepted 12 July 2015; published 15 July 2015

Abstract

The present article is mainly devoted for solving bordered *k*-tridiagonal linear systems of equations. Two efficient and reliable symbolic algorithms for solving such systems are constructed. The computational cost of the algorithms is obtained. Some illustrative examples are given.

Keywords

Bordered *k*-Tridiagonal Matrices, Partitioned Matrices, Algorithm, LU Factorization, MAPLE

1. Introduction

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 tridiagonal matrices. Tridiagonal systems of linear equations take the form:

$$T_n \boldsymbol{x} = \boldsymbol{g}, \tag{1}$$

where

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

*Corresponding author.

How to cite this paper: El-Mikkawy, M. and Atlan, F. (2015) New Algorithms for Solving Bordered *k*-Tridiagonal Linear Systems. *Journal of Applied Mathematics and Physics*, **3**, 862-873. http://dx.doi.org/10.4236/jamp.2015.37107

$\boldsymbol{x}=[x_1,x_2,\cdots,x_n]^{\mathrm{T}}$ and $\boldsymbol{g}=[g_1,g_2,\cdots,g_n]^{\mathrm{T}}$. 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 (2) can be stored in $3n-2$ memory locations, rather than n^2 memory locations for a full matrix, by using three vectors $\boldsymbol{a}=[a_1,a_2,\cdots,a_{n-1}]$, $\boldsymbol{b}=[b_1,b_2,\cdots,b_{n-1}]$, and $\boldsymbol{d}=[d_1,d_2,\cdots,d_n]$. This is always a good habit in computation in order to save memory space. To study tridiagonal matrices it is convenient to introduce a vector $\boldsymbol{e}$ defined by [1] [2]:

$$\boldsymbol{e}=[e_1,e_2,\cdots,e_n], \tag{3}$$

where

$$e_i=\begin{cases} d_1, & \text{for } i=1 \\ d_i-\dfrac{a_{i-1}b_{i-1}}{e_{i-1}}, & \text{for } i=2,3,\cdots,n. \end{cases} \tag{4}$$

For some important results concerning tridiagonal matrix the reader may refer to [2]-[18]. The motivation of the current paper is to derive algorithms for solving bordered k-tridiagonal linear systems of the form:

$$A_n^{(k)}\boldsymbol{x}=\boldsymbol{f}, \tag{5}$$

with

$$A_n^{(k)}=\begin{bmatrix} d_1 & 0 & \cdots & 0 & a_1 & 0 & \cdots & & 0 & p_1 \\ 0 & d_2 & 0 & \cdots & 0 & a_2 & \ddots & & \vdots & p_2 \\ \vdots & \ddots & \ddots & \ddots & \cdots & \ddots & \ddots & & \vdots & \vdots \\ \vdots & & \ddots & \ddots & \cdots & \ddots & & \ddots & 0 & \vdots \\ 0 & \cdots & \ddots & \ddots & \ddots & \cdots & & \ddots & a_{n-k-1} & p_{n-k-1} \\ b_1 & \ddots & \cdots & \ddots & \ddots & \ddots & & \ddots & \ddots & a_{n-k} \\ 0 & b_2 & \ddots & \cdots & \ddots & \ddots & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & \cdots & & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & b_{n-k-1} & \ddots & \cdots & & \ddots & d_{n-1} & 0 \\ q_1 & q_2 & \cdots & q_{n-k-1} & b_{n-k} & 0 & \cdots & \cdots & 0 & d_n \end{bmatrix}. \tag{6}$$

where $1\le k<n$, $\boldsymbol{x}=[x_1,x_2,\cdots,x_n]^{\mathrm{T}}$ and $\boldsymbol{f}=[f_1,f_2,\cdots,f_n]^{\mathrm{T}}$.

The linear systems (5) for $k=1$, 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 [19]-[28].

Throughout this paper, $\lfloor x \rfloor$ denotes the greatest integer less than or equal to x. Also, the word "simplify" means simplify the algebraic 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. Some illustrative examples are given in Section 4. A conclusion is given in Section 5.

2. Solving the System (5) via *k*-Tridiagonal Solvers

In this section, we are going to formulate a new symbolic algorithm, based on the Sherman-Morrison-Woodbury formula [29], for solving bordered k-tridiagonal linear system of the form (5). By doing this, the solution of the system (5) reduces to solving three k-tridiagonal linear systems by using k-tridiagonal solvers such as these presented in [12] [30].

Let us first note that the coefficient matrix, $A_n^{(k)}$ of the system (5) can be written as:

$$A_n^{(k)}=T_n^{(k)}+UV^T, \tag{7}$$

where $T_n^{(k)}$, U and V are given by:

$$T_n^{(k)} = \begin{bmatrix} d_1 & 0 & \cdots & 0 & a_1 & 0 & \cdots & 0 \\ 0 & d_2 & 0 & \cdots & 0 & a_2 & \ddots & \vdots \\ \vdots & 0 & \ddots & \ddots & \cdots & \ddots & \ddots & 0 \\ 0 & \cdots & \ddots & \ddots & \ddots & \cdots & \ddots & a_{n-k} \\ b_1 & 0 & \cdots & \ddots & \ddots & \ddots & \cdots & 0 \\ 0 & b_2 & \ddots & \cdots & \ddots & \ddots & 0 & \vdots \\ \vdots & \ddots & \ddots & \ddots & \cdots & 0 & d_{n-1} & 0 \\ 0 & \cdots & 0 & b_{n-k} & 0 & \cdots & 0 & d_n \end{bmatrix}, \tag{8}$$

$$U = \begin{bmatrix} p_1 & 0 \\ p_2 & 0 \\ \vdots & \vdots \\ p_{n-k-1} & 0 \\ 0 & 0 \\ \vdots & \vdots \\ 0 & 0 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} u_1 & u_2 \end{bmatrix} \tag{9}$$

and

$$V = \begin{bmatrix} 0 & q_1 \\ 0 & q_2 \\ \vdots & \vdots \\ 0 & q_{n-k-1} \\ 0 & 0 \\ \vdots & \vdots \\ 0 & 0 \\ 1 & 0 \end{bmatrix} = \begin{bmatrix} v_1 & v_2 \end{bmatrix}. \tag{10}$$

By applying the Sherman-Morrison-Woodbury formula to $A_n^{(k)}$ in (7), we get:

$$\left(A_n^{(k)}\right)^{-1} = \left(T_n^{(k)} + UV^T\right)^{-1} = \left(T_n^{(k)}\right)^{-1} - \left(T_n^{(k)}\right)^{-1} U \left(I + V^T \left(T_n^{(k)}\right)^{-1} U\right)^{-1} V^T \left(T_n^{(k)}\right)^{-1}, \tag{11}$$

and

$$\det\left(A_n^{(k)}\right) = \det\left(T_n^{(k)}\right) \cdot \det\left(I + V^T \left(T_n^{(k)}\right)^{-1} U\right), \tag{12}$$

provided that the matrix $T_n^{(k)}$ in (8) is invertible.

By making use of (7)-(12), we see that the solution of the bordered *k*-tridiagonal system (5) reduces to solving three *k*-tridiagonal linear systems by using *k*-tridiagonal solvers. Consequently, we may formulate the following symbolic algorithm for solving the linear system (5).

Algorithm 2.1. An algorithm for solving bordered *k*-tridiagonal linear systems.

To solve bordered *k*-tridiagonal linear systems of the form (5), we may proceed as follows:

INPUT: The entries of the matrix $T_n^{(k)}$ in (8) and the vectors *V*, *U* and ***f***.

OUTPUT: The solution vector $\boldsymbol{x} = [x_1, x_2, \cdots, x_n]^{\mathrm{T}}$.

Step 1: Use the ***k*-DETGTRI** algorithm [12] to check the non-singularity of the matrix $T_n^{(k)}$ in (8).

Step 2: If $\det\left(T_n^{(k)}\right) = 0$, then Exiterror ("Failure") end if.

Step 3: Solve the three linear systems of *k*-tridiagonal type:

$$T_n^{(k)} \boldsymbol{y} = \boldsymbol{f}\ , \quad T_n^{(k)} \boldsymbol{z}_1 = \boldsymbol{u}_1, \quad \text{and} \quad T_n^{(k)} \boldsymbol{z}_2 = \boldsymbol{u}_2,$$

by using, for example, the k-Thomas solver in [12] then construct $\boldsymbol{z}=\left[z_1\ z_2\right]$.

Step 4: Compute the 2×2 matrix, H using $H=\left(I+V^{\mathrm{T}}\boldsymbol{z}\right)$.

If $\det(H)=0$ then Exiterror ("Failure") end if.

Step 5: Compute $\boldsymbol{x}=\boldsymbol{y}-\boldsymbol{z}H^{-1}V^{T}\boldsymbol{y}$ to get the solution vector $\boldsymbol{x}$.

The computational cost of the algorithm is $O(n)$. The Algorithm 2.1, will be referred to as **DB-kTRI1** algorithm. Parallel computations of the three linear systems in **Step 3** are available for heterogeneous environments.

It should be noted that the algorithm presented in [28] is a special case of the **DB-kTRI1** algorithm when $k=1$.

3. Solving the System (5) Using the LU Factorization and Partition

In this section, we are going to consider the construction of a new algorithm for solving linear systems of equations of bordered k-tridiagonal type (5) by using partition. For this purpose it is convenient to introduce a vector $\boldsymbol{c}=\left[c_1,c_2,\cdots,c_n\right]$, whose first $(n-1)$ components, $c_1,c_2,\cdots,c_{n-1}$ are given by:

$$c_i=\begin{cases}d_i, & \text{for } i=1,2,\cdots,k\\ d_i-b_{i-k}\,y_{i-k}, & \text{for } i=k+1,k+2,\cdots,n-1\end{cases} \tag{13}$$

where $y_i=\dfrac{a_i}{c_i}$. The last component, c_n of the vector $\boldsymbol{c}$ will be computed later on.

Consider the Doolittle LU factorization [31] of the coefficient matrix $A_n^{(k)}$ in (6).

$$\left[\begin{array}{ccccccccc|c}
d_1 & 0 & \cdots & 0 & a_1 & 0 & \cdots & & 0 & p_1\\
0 & d_2 & 0 & \cdots & 0 & a_2 & \ddots & & \vdots & p_2\\
\vdots & \ddots & \ddots & \ddots & \cdots & \ddots & \ddots & & \vdots & \vdots\\
\vdots & & \ddots & \ddots & \cdots & \ddots & & \ddots & 0 & \vdots\\
0 & \cdots & \ddots & \ddots & \ddots & \cdots & & \ddots & a_{n-k-1} & p_{n-k-1}\\
b_1 & \ddots & \cdots & \ddots & \ddots & \ddots & & \ddots & \ddots & a_{n-k}\\
0 & b_2 & \ddots & \cdots & \ddots & \ddots & \ddots & \ddots & \ddots & 0\\
\vdots & \ddots & \ddots & \ddots & \cdots & & \ddots & \ddots & \ddots & \vdots\\
0 & \cdots & 0 & b_{n-k-1} & \ddots & \cdots & & \ddots & d_{n-1} & 0\\
\hline
q_1 & q_2 & \cdots & q_{n-k-1} & b_{n-k} & 0 & \cdots & \cdots & 0 & d_n
\end{array}\right]=\left[\begin{array}{c|c}\hat{T}_{n-1}^{(k)} & \boldsymbol{p}\\ \hline \boldsymbol{q}^T & d_n\end{array}\right]$$

$$=\left[\begin{array}{ccccccccc|c}
1 & 0 & \cdots & & & & \cdots & & 0 & 0\\
0 & 1 & \ddots & & & & & & & \vdots\\
\vdots & 0 & \ddots & \ddots & & & & & & \\
0 & \vdots & \ddots & \ddots & \ddots & & & & & \\
\frac{b_1}{c_1} & \ddots & \vdots & \ddots & \ddots & \ddots & & & & \\
0 & \frac{b_2}{c_2} & \ddots & \cdots & \ddots & \ddots & \ddots & & & \vdots\\
\vdots & \ddots & \ddots & \ddots & \cdots & \ddots & \ddots & \ddots & & \vdots\\
0 & \cdots & 0 & \frac{b_{n-k-1}}{c_{n-k-1}} & \ddots & & \ddots & \ddots & 0 & 0\\
\hline
h_1 & h_2 & h_3 & \cdots & & \cdots & \cdots & h_{n-1} & 1 & 1
\end{array}\right]$$

$$\left[\begin{array}{ccccccccc|c}
c_1 & 0 & \cdots & 0 & a_1 & 0 & \cdots & & 0 & v_1\\
0 & c_2 & 0 & \cdots & 0 & a_2 & \ddots & & \vdots & v_2\\
\vdots & 0 & \ddots & \ddots & \cdots & \ddots & \ddots & & 0 & v_3\\
 & & \ddots & \ddots & \cdots & \ddots & \ddots & & a_{n-k-1} & \vdots\\
 & & & \ddots & \ddots & \cdots & \ddots & \ddots & \ddots & \\
 & & & & \ddots & \ddots & \cdots & \ddots & \ddots & \vdots\\
 & & & & & \ddots & \ddots & \cdots & \ddots & \vdots\\
 & & & & & & \ddots & \ddots & \ddots & \vdots\\
\vdots & & & & & & & \ddots & c_{n-1} & v_{n-1}\\
\hline
0 & \cdots & & & & \cdots & \cdots & & 0 & c_n
\end{array}\right], \tag{14}$$

where $\boldsymbol{p}=\left[p_1,p_2,\cdots,p_{n-k-1},a_{n-k},0,\cdots,0\right]^{\mathrm{T}}$, $\boldsymbol{q}^{\mathrm{T}}=\left[q_1,q_2,\cdots,q_{n-k-1},b_{n-k},0,\cdots,0\right]$, $\boldsymbol{p},\boldsymbol{q}\in\mathbb{R}^{(n-1)\times 1}$ and $\hat{T}_{n-1}^{(k)}$ is the $(n-1)$ th leading principal submatrix of $A_n^{(k)}$.

Equation (14) can be rewritten in the form:

$$\left[\begin{array}{c|c} \hat{T}_{n-1}^{(k)} & \boldsymbol{p} \\ \hline \boldsymbol{q}^T & d_n \end{array}\right] = \left[\begin{array}{c|c} L_{n-1}^{(k)} & \boldsymbol{0} \\ \hline \boldsymbol{h}^T & 1 \end{array}\right]\left[\begin{array}{c|c} U_{n-1}^{(k)} & \boldsymbol{v} \\ \hline \boldsymbol{0}^T & c_n \end{array}\right], \tag{15}$$

where

$$L_{n-1}^{(k)} = \begin{bmatrix} 1 & 0 & \cdots & & & \cdots & 0 \\ 0 & 1 & \ddots & & & & \vdots \\ \vdots & 0 & \ddots & \ddots & & & \\ 0 & \vdots & \ddots & \ddots & \ddots & & \\ \dfrac{b_1}{c_1} & \ddots & \vdots & \ddots & \ddots & \ddots & \\ 0 & \dfrac{b_2}{c_2} & \ddots & \cdots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \cdots & 0 & \ddots & 0 \\ 0 & \cdots & 0 & \dfrac{b_{n-k-1}}{c_{n-k-1}} & 0 & \cdots & 0 & 1 \end{bmatrix}, \; U_{n-1}^{(k)} = \begin{bmatrix} c_1 & 0 & \cdots & 0 & a_1 & 0 & \cdots & 0 \\ 0 & c_2 & 0 & \cdots & 0 & a_2 & \ddots & \vdots \\ \vdots & 0 & \ddots & \ddots & \cdots & \ddots & \ddots & 0 \\ & & \ddots & \ddots & \ddots & \cdots & \ddots & a_{n-k-1} \\ & & & \ddots & \ddots & \ddots & \cdots & 0 \\ & & & & \ddots & \ddots & 0 & \vdots \\ & & & & & \ddots & c_{n-2} & 0 \\ & & & & & \cdots & 0 & c_{n-1} \end{bmatrix}, \tag{16}$$

$\boldsymbol{v} = [v_1, v_2, \cdots, v_{n-1}]^{\mathrm{T}}$ and $\boldsymbol{h}^{\mathrm{T}} = [h_1, h_2, \cdots, h_{n-1}]$, $\boldsymbol{v}, \boldsymbol{h} \in \mathbb{R}^{(n-1)\times 1}$.

From (15), we see that the following four matrix equations must be satisfied.

$$L_{n-1}^{(k)} U_{n-1}^{(k)} = \hat{T}_{n-1}^{(k)}, \tag{17}$$

$$L_{n-1}^{(k)} \boldsymbol{v} = \boldsymbol{p}, \tag{18}$$

$$\boldsymbol{h}^{\mathrm{T}} U_{n-1}^{(k)} = \boldsymbol{q}^{\mathrm{T}}, \tag{19}$$

and

$$\boldsymbol{h}^{\mathrm{T}} \boldsymbol{v} + c_n = d_n. \tag{20}$$

Two cases will be considered:

CASE(I): $k \le \left\lfloor \dfrac{n}{2} \right\rfloor$:

In this case, solving (18) and (19) for $\boldsymbol{h}$ and $\boldsymbol{v}$ respectively, yields:

$$h_i = \begin{cases} \dfrac{q_i}{c_i} & \text{if } i = 1, 2, \cdots, k, \\ \dfrac{1}{c_i}\left(q_i - h_{i-k} a_{i-k}\right) & \text{if } i = k+1, k+2, \cdots, n-k-1, \\ \dfrac{1}{c_i}\left(b_i - h_{i-k} a_{i-k}\right) & \text{if } i = n-k \\ -\dfrac{a_{i-k}}{c_i} h_{i-k} & \text{if } i = n-k+1, n-k+2, \cdots, n-1, \end{cases} \tag{21}$$

$$v_i = \begin{cases} p_i & \text{if } i = 1, 2, \cdots, k, \\ p_i - \dfrac{b_{i-k}}{c_{i-k}} v_{i-k} & \text{if } i = k+1, k+2, \cdots, n-k-1, \\ a_i - \dfrac{b_{i-k}}{c_{i-k}} v_{i-k} & \text{if } i = n-k \\ -\dfrac{b_{i-k}}{c_{i-k}} v_{i-k} & \text{if } i = n-k+1, n-k+2, \cdots, n-1. \end{cases} \tag{22}$$

CASE(II): $k > \left\lfloor \frac{n}{2} \right\rfloor$:

In this case, solving (18) and (19) for *h* and *v* respectively, gives:

$$h_i = \begin{cases} \frac{q_i}{c_i} & \text{if } i = 1,2,\cdots,n-k-1, \\ \frac{b_i}{c_i} & \text{if } i = n-k \\ 0 & \text{if } i = n-k+1, n-k+2,\cdots,k, \\ -\frac{a_{i-k}}{c_i} h_{i-k} & \text{if } i = k+1, k+2,\cdots,n-1, \end{cases} \tag{23}$$

$$v_i = \begin{cases} p_i & \text{if } i = 1,2,\cdots,n-k-1, \\ a_i & \text{if } i = n-k \\ 0 & \text{if } i = n-k+1, n-k+2,\cdots,k \\ -\frac{b_{i-k}}{c_{i-k}} v_{i-k} & \text{if } i = k+1, k+2,\cdots,n-1. \end{cases} \tag{24}$$

In both cases, we have from (20),

$$c_n = d_n - \sum_{r=1}^{n-1} h_r v_r. \tag{25}$$

At this stage, the determinant of the coefficient matrix in (6) can be computed using the following computational symbolic algorithm.

Algorithm 3.1. An algorithm for computing the determinant of bordered *k*-tridiagonal matrices.

To compute the determinant of a bordered *k*-tridiagonal matrix in (6), we may proceed as follows:

INPUT: Order of the matrix *n*, the value of *k* and the components, a_i, d_i, b_i, p_i, q_i.

OUTPUT: The determinant of the matrix $A_n^{(k)}$ in (6).

Step 1: Compute c_i, $i = 1,2,\cdots,n-1$ using (13).

If $c_i = 0$ for any $i \le n$, set $c_i = t$ (*t* is just a symbolic name) and continue to compute $c_{i+1}, c_{i+2}, \cdots, c_{n-1}$, in its simplest rational forms, in terms of *t* using (13).

Step 2: Compute c_n using (25).

Step 3: Simplify $P(t) = \prod_{r=1}^{n} c_r$ to obtain $\det\left(A_n^{(k)}\right) = P(0)$.

The Algorithm 3.1, will be referred to as **DB-kDETGTRI** algorithm.

Remarks:

1) The **DETGTRI** algorithm in [1] is a special case of **DB-kDETGTRI** algorithm when $p_i = q_i = 0$, $i = 1,2,\cdots,n-k-1,$ and $k = 1$.

2) The ***k*-DETGTRI** algorithm in [12] is a special case of **DB-kDETGTRI** algorithm when $p_i = q_i = 0$ $i = 1,2,\cdots,n-k-1,$ and $1 \le k < n$.

3) The **PERTRI** algorithm in [8] is a special case of **DB-kDETGTRI** algorithm when $p_1 = t$, $q_1 = s$ and $p_i = q_i = 0$, $i = 2,3,\cdots,n-k-1$, and $k = 1$.

4) The **DETSGCM** algorithm in [32] is a special case of **DB-kDETGTRI** algorithm when $d_i = -\frac{\beta_i}{\alpha_i}$, $i = 1,2,\cdots,n-1,$ $d_n = \frac{-\beta_n - a_1}{\alpha_n}$, $a_i = \frac{1}{\alpha_i}$, $i = 1,2,\cdots,n-k,$ $b_{i-1} = \frac{\gamma_i}{\alpha_i}$, $i = 2,3,\cdots,n-k,$ $b_{n-k} = \frac{\gamma_n - a_2}{\alpha_n}$, $q_i = -\frac{a_{n+1-i}}{\alpha_n}$, $i = 1,2,\cdots,n-k-1,$ and $p_i = 0$, $i = 1,2,\cdots,n-k-1$, and $k = 1$. (See also [33]).

Now the linear system in (5), can be rewritten in partitioned form as:

$$\left[\begin{array}{c|c} L_{n-1}^{(k)} & \mathbf{0} \\ \hline \boldsymbol{h}^{\mathrm{T}} & 1 \end{array}\right]\left[\begin{array}{c|c} U_{n-1}^{(k)} & \boldsymbol{v} \\ \hline \mathbf{0}^{\mathrm{T}} & c_n \end{array}\right]\left[\begin{array}{c} X_1 \\ \hline X_2 \end{array}\right]=\left[\begin{array}{c} F_1 \\ \hline F_2 \end{array}\right], \tag{26}$$

where $X_1=[x_1,x_2,\cdots,x_{n-1}]^{\mathrm{T}}$, $X_2=[x_n]=x_n$, $F_1=[f_1,f_2,\cdots,f_{n-1}]^{\mathrm{T}}$ and $F_2=[f_n]=f_n$.

To solve the linear system (26) it is equivalent to solve the two standard linear systems:

$$\left[\begin{array}{c|c} L_{n-1}^{(k)} & \mathbf{0} \\ \hline \boldsymbol{h}^{T} & 1 \end{array}\right]\left[\begin{array}{c} Z_1 \\ \hline Z_2 \end{array}\right]=\left[\begin{array}{c} F_1 \\ \hline F_2 \end{array}\right], \tag{27}$$

and

$$\left[\begin{array}{c|c} U_{n-1}^{(k)} & \boldsymbol{v} \\ \hline \mathbf{0}^{T} & c_n \end{array}\right]\left[\begin{array}{c} X_1 \\ \hline X_2 \end{array}\right]=\left[\begin{array}{c} Z_1 \\ \hline Z_2 \end{array}\right], \tag{28}$$

where $Z_1=[z_1,z_2,\cdots,z_{n-1}]^{\mathrm{T}}$ and $Z_2=[z_n]=z_n$. The linear systems (27) and (28) can be solved directly by using forward and backward substitution respectively.

In conclusion, we may now formulate a second symbolic algorithm for solving the bordered *k*-tridiagonal linear system (5) as follows:

Algorithm 3.2. A symbolic algorithm for solving bordered *k*-tridiagonal linear systems using partition.

To solve a general bordered *k*-tridiagonal linear system of the form (5), we may proceed as follows:

INPUT: Order of the matrix *n*, the value of *k* and the components, a_i, d_i, b_i, p_i, q_i and f_i.

OUTPUT: The determinant of the matrix $A_n^{(k)}$ in (6) and the solution vector $\boldsymbol{x}=[x_1,x_2,\cdots,x_n]^{\mathrm{T}}$.

Step 1: If $k\le\left\lfloor\frac{n}{2}\right\rfloor$ **then**

For $i=1,2,\cdots,k$ **do**

Set $c_i=d_i$, **If** $c_i=0$ **then** $c_i=t$ **End if**

$y_i=\frac{b_i}{c_i}$, $z_i=f_i$, $h_i=\frac{q_i}{c_i}$ and $v_i=p_i$.

End do

For $i=k+1,k+2,\cdots,n-k-1$ **do**

Compute and simplify:

$c_i=d_i-a_{i-k}y_{i-k}$. **If** $c_i=0$. **then** $c_i=t$ **End if**

$y_i=\frac{b_i}{c_i}$,

$h_i=\frac{1}{c_i}\left(q_i-h_{i-k}a_{i-k}\right)$.

$v_i=p_i-v_{i-k}y_{i-k}$.

End do.

For $i=n-k,n-k+1,\cdots,n-1$ **do**

Compute and simplify:

$c_i=d_i-a_{i-k}y_{i-k}$. **If** $c_i=0$ **then** $c_i=t$ **End if**

End do.

$h_{n-k}=\frac{1}{c_{n-k}}\left(b_{n-k}-h_{n-2k}a_{n-2k}\right)$.

$v_{n-k}=a_{n-k}-v_{n-2k}y_{n-2k}$.

For $i=n-k+1,n-k+2,\cdots,n-1$ **do**

Compute and simplify:

$h_i=-h_{i-k}a_{i-k}/c_i$.

$v_i=-v_{i-k}y_{i-k}$.

End do.

Else

For $i = 1, 2, \cdots, k$ **do**
Set $c_i = d_i$, **If** $c_i = 0$ **then** $c_i = t$ **End if**
$z_i = f_i$.
End do
For $i = k+1, k+2, \cdots, n-1$ **do**
Compute and simplify:

$$y_{i-k} = \frac{b_{i-k}}{c_{i-k}},$$

$c_i = d_i - a_{i-k}\, y_{i-k}$. **If** $c_i = 0$ **then** $c_i = t$ **End if**
End do.
For $i = 1, 2, \cdots, n-k-1$ **do**
Compute and simplify:

$$h_i = \frac{q_i}{c_i}$$

$v_i = p_i$.
End do.
$h_{n-k} = y_{n-k}$.
$v_{n-k} = a_{n-k}$.
For $i = n-k+1, n-k+2, \cdots, k$ **do**
Compute and simplify:
$h_i = 0$.
$v_i = 0$.
End do.
For $i = k+1, k+2, \cdots, n-1$ **do**
Compute and simplify:
$h_i = -h_{i-k}\, a_{i-k} / c_i$.
$v_i = -v_{i-k}\, y_{i-k}$.
End do.
End if

Step 2: Compute and simplify:

$$c_n = d_n - \sum_{r=1}^{n-1} h_r v_r.$$

Step 3: Use the **DB-kDETGTRI** algorithm to check the non-singularity of the coefficient matrix of the system (5).

Step 4: If the determinant of the coefficient matrix in (5) equals zero, **then** Exiterror ("No solutions") **End if**.

Step 5: Compute the solution vector $\boldsymbol{x} = [x_1, x_2, \cdots, x_n]^T$ using

For $i = k+1, k+2, \cdots, n-1$ **do**
Compute and simplify:
$z_i = f_i - z_{i-k}\, y_{i-k}$.
End do.

$$z_n = f_n - \sum_{r=1}^{n-1} h_r\, z_r,$$

$$x_n = \frac{z_n}{c_n}$$

Step 6: For $i = n-1, n-2, \cdots, n-k$ **do**
Compute and simplify:
$x_i = (z_i - v_i x_n) / c_i$
End do.
For $i = n-k-1, n-k-2, \cdots, 1$ **do**
Compute and simplify:

$x_i = \left(z_i - a_i x_{i+k} - v_i x_n\right)/c_i$

End do.

Step 7: Substitute $t = 0$ in all expressions of the solution vector x_i, $i = 1, 2, \cdots, n$.

Concerning the computational cost of Algorithm 3.2, we have: For $k \le \left\lfloor \frac{n}{2} \right\rfloor$, the computational cost is $11n - 6k - 10$ multiplications/divisions and $8n - 7k - 6$ additions/subtractions. The computational cost for the case $k > \left\lfloor \frac{n}{2} \right\rfloor$, is $12n - 8k - 11$ multiplications/divisions and $6n - 3k - 6$ additions/subtractions. The Algorithm 2.3, will be referred to as **DB-kTRI2** algorithm.

Remarks:

- The **DB-kTRI2** algorithm is a natural generalization of the algorithms in [30] and [34].
- The last component, a_{n-k} of the vector $\boldsymbol{a}$ is also the (n-k)th component of the vector $\boldsymbol{p}$.
- The last component, b_{n-k} of the vector $\boldsymbol{b}$ is also the (n-k)th component of the vector $\boldsymbol{q}$.

A MAPLE procedure, based on the algorithm **DB-kDETGTRI** and **DB-kTRI2**, is available upon request from the authors.

4. Illustrative Examples

Example 4.1. Solve the bordered k-tridiagonal linear system

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

Solution: We have: $n = 10$, $k = 3$, $\boldsymbol{a} = [1, -1, 4, 7, 3, 2, 4]$, $\boldsymbol{d} = [1, 1, -2, 1, 5, -1, 1, 2, -1, 2]$, $\boldsymbol{b} = [1, 3, -1, 2, 5, 7, -3]$, $\boldsymbol{p} = [4, 8, 2, -1, 1, 3]^{\mathrm{T}}$, $\boldsymbol{q} = [3, 2, 1, -1, 1, 4]$ and $\boldsymbol{f} = [6, 8, 4, 8, 12, 3, 7, 7, 6, 9]^{\mathrm{T}}$.

By applying the **DB-kTRI1** algorithm, we get

- $\boldsymbol{c} = \left[1, 1, -2, t, 8, -3, \frac{(t-14)}{t}, \frac{1}{8}, \frac{11}{3}, \frac{14(t-2)}{(t-14)}\right]$.
- $\det\left(T_n^{(k)}\right) = \left(\prod_{i=1}^{10} c_i\right)_{t=0} = (308 * t - 616)_{t=0} = -616$ Thus $T_n^{(k)}$ is nonsingular, (**Steps 1, 2**).
- $\boldsymbol{y} = \left[\frac{25}{2}, -37, \frac{4}{11}, \frac{-13}{2}, -45, \frac{13}{11}, \frac{2}{7}, 116, \frac{25}{11}, \frac{69}{14}\right]^{\mathrm{T}}$,

  $\boldsymbol{z} = [z_1 \; z_2] = \begin{bmatrix} \frac{3}{2} & -38 & -\frac{7}{11} & \frac{5}{2} & -46 & \frac{2}{11} & \frac{-5}{7} & 115 & \frac{14}{11} & -\frac{15}{14} \\ 1 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & \frac{1}{2} \end{bmatrix}^{\mathrm{T}}$, (**Step 3**).
- $H = I + V^T z = \begin{bmatrix} -\frac{1}{14} & \frac{1}{2} \\ -\frac{1319}{11} & 5 \end{bmatrix}$, (**Step 4**).

- The solution vector is $\boldsymbol{x} = \boldsymbol{y} - \boldsymbol{z}H^{-1}V^T\boldsymbol{y} = [1,1,1,1,1,1,1,1,1,1]^{\mathrm{T}}$, (**Step 5**).

Example 4.2. Solve the bordered *k*-tridiagonal linear system

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 3 \\ 0 & 5 & 0 & 0 & 0 & 3 & 0 & 0 & 0 & 1 \\ 0 & 0 & 2 & 0 & 0 & 0 & -2 & 0 & 0 & 7 \\ 0 & 0 & 0 & -3 & 0 & 0 & 0 & 11 & 0 & -2 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 4 \\ 0 & 2 & 0 & 0 & 0 & 7 & 0 & 0 & 0 & 9 \\ 0 & 0 & 3 & 0 & 0 & 0 & 6 & 0 & 0 & 0 \\ 0 & 0 & 0 & 4 & 0 & 0 & 0 & -2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 5 & 0 & 0 & 0 & 1 & 0 \\ 1 & 7 & -3 & 2 & -2 & 6 & 0 & 0 & 0 & 11 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \\ x_8 \\ x_9 \\ x_{10} \end{bmatrix} = \begin{bmatrix} 5 \\ 7 \\ 7 \\ 31 \\ 7 \\ 23 \\ 9 \\ -6 \\ 6 \\ 19 \end{bmatrix}$$

Solution: We have: $n = 10$, $k = 4$, $\boldsymbol{a} = [1,3,-2,11,1,9]$, $\boldsymbol{d} = [1,5,2,-3,1,7,6,-2,1,11]$ $\boldsymbol{b} = [1,2,3,4,5,6]$, $\boldsymbol{p} = [3,1,7,-2,4]^{\mathrm{T}}$, $\boldsymbol{q} = [1,7,-3,2,-2]$ and $\boldsymbol{f} = [5,7,7,31,7,23,9,-6,6,19]^{\mathrm{T}}$.

By applying the **DB-kTRI2** algorithm, we have

- $\boldsymbol{c} = \left[1,5,2,-3,t,\frac{29}{5},9,\frac{38}{3},\frac{t-5}{t},\frac{1}{551}\frac{6139t-29042}{t-5}\right]$.
- $\det\left(A_n^{(k)}\right) = \left(\prod_{i=1}^{10} c_i\right)_{t=0} = (-221004*t+1045512)_{t=0} = 1045512$. Hence $A_n^{(k)}$ is nonsingular, (**Steps 3, 4**).
- The solution vector is $x = [1,0,1,0,1,2,1,3,1,1]^{\mathrm{T}}$, (**Steps 5-7**).

Example 4.3. Solve the bordered *k*-tridiagonal linear system

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 3 \\ 0 & 4 & 0 & 0 & 0 & 0 & 0 & 2 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & -1 & 7 \\ 0 & 0 & 0 & -3 & 0 & 0 & 0 & 0 & 0 & -2 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 7 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 6 & 0 & 0 & 0 \\ 0 & 2 & 0 & 4 & 0 & 0 & 0 & -2 & 0 & 0 \\ 0 & 0 & 3 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 7 & -3 & 4 & 0 & 0 & 0 & 0 & 0 & 11 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \\ x_8 \\ x_9 \\ x_{10} \end{bmatrix} = \begin{bmatrix} 10 \\ 9 \\ 20 \\ -6 \\ 1 \\ 7 \\ 1 \\ 0 \\ 1 \\ 41 \end{bmatrix},$$

Solution: We have: $n = 10$, $k = 6$, $\boldsymbol{a} = [1,2,-1,-2]$, $\boldsymbol{d} = [1,4,1,-3,1,7,6,-2,1,11]$, $\boldsymbol{b} = [1,2,3,4]$, $\boldsymbol{p} = [3,1,7]^{\mathrm{T}}$, $\boldsymbol{q} = [1,7,-3]$ and $\boldsymbol{f} = [10,9,20,-6,1,7,1,0,1,41]^{\mathrm{T}}$.

By applying the **DB-kTRI1** algorithm, we get

- $\boldsymbol{c} = \left[1,4,1,-3,1,7,5,-3,4,\frac{25}{3}\right]$.
- $\det\left(T_n^{(k)}\right) = \prod_{i=1}^{10} c_i = 42000$. Thus $T_n^{(k)}$ is nonsingular, (**Steps 1, 2**).
- $\boldsymbol{y} = \left[\frac{59}{5},\frac{3}{2},\frac{21}{4},-\frac{16}{25},1,1,-\frac{9}{5},\frac{3}{2},-\frac{59}{4},\frac{99}{25}\right]^{\mathrm{T}}$,

$$\boldsymbol{z} = [z_1\ z_2] = \begin{bmatrix} \frac{18}{5} & \frac{1}{6} & \frac{7}{4} & 0 & 0 & 0 & -\frac{3}{5} & \frac{1}{6} & -\frac{21}{4} & 0 \\ 0 & 0 & 0 & -\frac{2}{25} & 0 & 0 & 0 & 0 & 0 & \frac{3}{25} \end{bmatrix}^{\mathrm{T}},$$ (**Step 3**).

- $H = I + V^{\mathrm{T}} \boldsymbol{z} = \begin{bmatrix} 1 & \frac{3}{25} \\ -\frac{29}{60} & 1 \end{bmatrix}$, (**Step 4**).
- The solution vector is $\boldsymbol{x} = \boldsymbol{y} - \boldsymbol{z} H^{-1} V^{T} \boldsymbol{y} = [1,1,0,0,1,1,0,1,1,3]^{\mathrm{T}}$, (**Step 5**).

By applying the **DB-kTRI2** algorithm, we have

- $\boldsymbol{c} = \left[1,4,1,-3,1,7,5,-3,4,\frac{529}{60}\right]$.
- $\det\left(A_n^{(k)}\right) = \prod_{i=1}^{10} c_i = 44436$. Hence $A_n^{(k)}$ is nonsingular, (**Steps 3, 4**).
- The solution vector is $\boldsymbol{x} = [1,1,0,0,1,1,0,1,1,3]^{\mathrm{T}}$, (**Steps 5-7**).

5. Conclusion

In this paper we have derived two symbolic algorithms (**DB-kTRI1** and **DB-kTRI2**) for solving bordered *k*-tridiagonal linear systems. The cost of each algorithm is $O(n)$. Our algorithm does not require any simplifying assumptions. To the best of our knowledge, this is the first study to show how to solve bordered *k*-tridiagonal linear systems. Finally, three examples are given for the sake of illustration.

References

[1] El-Mikkawy, M.E.A. (2004) A Fast Algorithm for Evaluating nth Order Tri-Diagonal Determinants. *Journal of Computational and Applied Mathematics*, **166**, 581-584. http://dx.doi.org/10.1016/j.cam.2003.08.044

[2] El-Mikkawy, M.E.A. and Karawia, A. (2006) Inversion of General Tridiagonal Matrices. *Applied Mathematics Letters*, **19**, 712-720. http://dx.doi.org/10.1016/j.aml.2005.11.012

[3] Akin, H. (2012) On 1D Reversible Cellular Automata with Reflective Boundary over the Prime Field of Order p. *International Journal of Modern Physics C*, **23**, 1-13. http://dx.doi.org/10.1142/s0129183111017020

[4] Chant, T.F. and Resascot, D.C. (1986) Generalized Deflated Block-Elimination. *SIAM Journal on Numerical Analysis*, **23**, 913-924. http://dx.doi.org/10.1137/0723059

[5] El-Mikkawy, M.E.A. (1991) An Algorithm for Solving Tridiagonal Systems. *Journal of Institute of Mathematics and Computer Sciences. Computer Sciences Series*, **4**, 205-210.

[6] El-Mikkawy, M.E.A. (2003) A Note on a Three-Term Recurrence for a Tridiagonal Matrix. *Applied Mathematics and Computation*, **139**, 503-511. http://dx.doi.org/10.1016/S0096-3003(02)00212-6

[7] El-Mikkawy, M.E.A. (2004) On the Inverse of a General Tridiagonal Matrix. *Applied Mathematics and Computation*, **150**, 669-679. http://dx.doi.org/10.1016/S0096-3003(03)00298-4

[8] El-Mikkawy, M.E.A. (2005) A New Computational Algorithm for Solving Periodic Tri-Diagonal Linear Systems. *Applied Mathematics and Computation*, **161**, 691-696. http://dx.doi.org/10.1016/j.amc.2003.12.114

[9] El-Mikkawy, M.E.A. and Rahmo, E.-D. (2008) A New Recursive Algorithm for Inverting General Tridiagonal and Anti-Tridiagonal Matrices. *Applied Mathematics and Computation*, **204**, 368-372. http://dx.doi.org/10.1016/j.amc.2008.06.053

[10] El-Mikkawy, M.E.A. and Rahmo, E.-D. (2009) A New Recursive Algorithm for Inverting General Periodic Pentadiagonal and Anti-Pentadiagonal Matrices. *Applied Mathematics and Computation*, **207**, 164-170. http://dx.doi.org/10.1016/j.amc.2008.10.010

[11] El-Mikkawy, M.E.A. and Rahmo, E. (2010) Symbolic Algorithm for Inverting Cyclic Pentadiagonal Matrices Recursively—Derivation and Implementation. *Computers & Mathematics with Applications*, **59**, 1386-1396. http://dx.doi.org/10.1016/j.camwa.2009.12.020

[12] El-Mikkawy, M.E.A. (2012) A Generalized Symbolic Thomas Algorithm. *Applied Mathematics*, **3**, 342-345. http://dx.doi.org/10.4236/am.2012.34052

[13] Kavcic, A. and Moura, J.M.F. (2000) Matrices with Banded Inverses: Inversion Algorithms and Factorization of Gauss-Markov Processes. *IEEE Transactions on Information Theory*, **46**, 1495-1509. http://dx.doi.org/10.1109/18.954748

[14] Li, H.B., Huang, T.Z., Liu X.P. and Li, H. (2010) On the Inverses of General Tridiagonal Matrices. *Linear Algebra and Its Applications*, **433**, 965-983. http://dx.doi.org/10.1016/j.laa.2010.04.042

[15] Shapiro, L.W. (1984) Positive Definite Matrices and Catalan Numbers, Revisited. *Proceedings of the American Mathematical Society*, **90**, 488-496. http://dx.doi.org/10.1090/S0002-9939-1984-0728375-5

[16] Sogabe, T. (2007) On a Two-Term Recurrence for the Determinant of a General Matrix. *Applied Mathematics and Computation*, **187**, 785-788. http://dx.doi.org/10.1016/j.amc.2006.08.156

[17] Sugimoto, T. (2012) On an Inverse Formula of a Tridiagonal Matrix. *Operators and Matrices*, **6**, 465-480. http://dx.doi.org/10.7153/oam-06-30

[18] Usmani, R. (1994) Inversion of a Tridiagonal Jacobi Matrix. *Linear Algebra and Its Applications*, **212**/**213**, 413-414. http://dx.doi.org/10.1016/0024-3795(94)90414-6

[19] Akbudak, K., Kayaaslan, E. and Aykanat, C. (2012) Analyzing and Enhancing OSKI for Sparse Matrix-Vector Multiplication. www.prace-ri.eu

[20] Amodio, P., Gladwelly, I. and Romanazzi, G. (2006) Numerical Solution of General Bordered ABD Linear Systems by Cyclic Reduction. *Journal of Numerical Analysis, Industrial and Applied Mathematics*, **1**, 5-12.

[21] Doedel, E. (1991) Numerical Analysis and Control of Bifurcation Problems (I): Bifurcation in Finite Dimensions. *International Journal of Bifurcation and Chaos in Applied Sciences and Engineering*, **1**, 493-520. http://dx.doi.org/10.1142/S0218127491000397

[22] Duff, I.S. and Scott, J.A. (2004) Stabilized Bordered Block Diagonal Forms for Parallel Sparse Solvers, RAL-TR-2004-006. http://www.numerical.rl.ac.uk/reports/reports.html

[23] da Fonseca, C.M. (2006) A Note on the Inversion of Acyclic Matrices. *International Journal of Pure and Applied Mathematics*, **31**, 307-317.

[24] Martin, A. and Boyd, I.D. (2010) Variant of the Thomas Algorithm for Opposite-Bordered Tridiagonal Systems of Equations. *International Journal for Numerical Methods in Biomedical Engineering*, **26**, 752-759. http://dx.doi.org/10.1002/cnm.1172

[25] Pajic, S. (2007) Power System State Estimation and Contingency Constrained Optimal Power Flow a Numerically Robust Implementation. PhD Thesis, Worcester-Polytechnic Institute, Worcester.

[26] Rashidinia, J. and Jalilian, R. (2009) Non-Polynomial Spline Solution of Fourth-Order Obstacle Boundary-Value Problems. World Academy of Science, Engineering and Technology. *International Scholarly and Scientific Research & Innovation*, **3**, 167-173.

[27] Udala, A., Reedera, R., Velmrea, E. and Harrisonb, P. (2006) Comparison of Methods for Solving the Schrodinger Equation for Multiquantum Well Heterostructure Applications. *Proceedings of the Estonian Academy of Sciences, Engineering*, **12**, 246-261.

[28] Wang, X.B. (2009) A New Algorithm with Its Scilab Implementation for Solution of Bordered Tridiagonal Linear Equations. 2009 *IEEE International Workshop on Open-Source Software for Scientific Computation* (*OSSC*), Guiyang, 18-20 September 2009, 11-14.

[29] Golub, G. and Van Loan, C. (1996) Matrix Computations. Third Edition, The Johns Hopkins University Press, Baltimore and London.

[30] El-Mikkawy, M.E.A. and Atlan, F. (2014) Algorithms for Solving Doubly Bordered Tridiagonal Linear Systems. *British Journal of Mathematics & Computer Science*, **4**, 1246-1267. http://dx.doi.org/10.9734/BJMCS/2014/8835

[31] Burden, R.L. and Faires, J.D. (2001) Numerical Analysis. Seventh Edition, Books & Cole Publishing, Pacific Grove.

[32] Karawia, A.A. (2013) Symbolic Algorithm for Solving Comrade Linear Systems Based on a Modified Stair-Diagonal Approach. *Applied Mathematics Letters*, **26**, 913-918. http://dx.doi.org/10.1016/j.aml.2012.10.019

[33] Karawia, A.A. (2012) A New Recursive Algorithm for Inverting a General Comrade Matrix. CoRR abs/1210.4662.

[34] Karawia, A.A. and Rizvi, Q.M. (2013) On Solving a General Bordered Tridiagonal Linear System. *International Journal of Mathematics and Mathematical Sciences*, **33**, 1160-1163.