Appl. Math. J. Chinese Univ.
2022, 37(3): 340-349

Inverting a k-heptadiagonal matrix based on Doolitle LU factorization

Maryam Shams Solary   Mehran Rasouli

Abstract. The purpose of the present paper is to show a new numeric and symbolic algorithm for inverting a general nonsingular k-heptadiagonal matrix. This work is based on Doolitle LU factorization of the matrix. We obtain a series of recursive relationships then we use them for constructing a novel algorithm for inverting a k-heptadiagonal matrix. The computational cost of the algorithm is calculated. Some illustrative examples are given to demonstrate the effectiveness of the proposed method.

§1 Introduction

There are various methods for finding the inverse of band matrices (if these exist). Recently, many authors worked in this way such as [1, 2, 4, 6, 8]. The inverse of band matrices are necessary in solving many problems, such as, computing the condition number, parallel computing, telecommunication system analysis, investigating the decay rate of the inverse and solving a linear system whose coefficient matrix is banded, solving differential equations using finite differences, heat conduction and fluid flow problems.

In this paper, we obtain a novel algorithm for inversing a k-heptadiagonal matrix by using a lemma and recursive relationships. Our method is based on Doolitle LU factorization. The computational cost of the algorithm is shown in the next section.

The $n \times n$ general heptadiagonal matrix T_n takes the form:

$$T_n = \begin{bmatrix} a_1 & b_1 & c_1 & d_1 & 0 & \cdots & 0 \\ e_1 & a_2 & b_2 & c_2 & d_2 & \ddots & \vdots \\ f_1 & e_2 & \ddots & \ddots & \ddots & \ddots & 0 \\ g_1 & f_2 & \ddots & \ddots & \ddots & \ddots & d_{n-3} \\ 0 & g_2 & \ddots & \ddots & \ddots & \ddots & c_{n-2} \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & b_{n-1} \\ 0 & \cdots & 0 & g_{n-3} & f_{n-2} & e_{n-1} & a_n \end{bmatrix}. \tag{1}$$

Received: 2019-02-21.   Revised: 2020-01-14.
MR Subject Classification: 65F15, 15A18.
Keywords: k-Heptadiagonal matrices, LU factorization, algorithm, inverse of a matrix.
Digital Object Identifier(DOI): https://doi.org/10.1007/s11766-022-3763-8.

The $n \times n$ general k-heptadiagonal matrix $T_n^{(k)}$ takes the form:

$$T_n^{(k)} = \begin{bmatrix}
a_1 & 0 & \dots & 0 & b_1 & 0 & \dots & 0 & c_1 & 0 & \dots & 0 & d_1 & 0 & \dots & 0 \\
0 & a_2 & 0 & \dots & 0 & b_2 & 0 & \cdots & 0 & c_2 & 0 & \cdots & 0 & d_2 & \ddots & \vdots \\
\vdots & 0 & \ddots & \ddots & \dots & \ddots & \ddots & \ddots & \dots & \ddots & \ddots & \ddots & \dots & \ddots & \ddots & 0 \\
0 & \vdots & \ddots & \ddots & \ddots & \dots & \ddots & \ddots & \ddots & \dots & \ddots & \ddots & \ddots & \dots & \ddots & d_{n-3k} \\
e_1 & 0 & \vdots & \ddots & \ddots & \ddots & \dots & \ddots & \ddots & \ddots & \dots & \ddots & \ddots & \ddots & \dots & 0 \\
0 & e_2 & \ddots & \vdots & \ddots & \ddots & \ddots & \dots & \ddots & \ddots & \ddots & \dots & \ddots & \ddots & \ddots & \vdots \\
\vdots & 0 & \ddots & \ddots & \vdots & \ddots & \ddots & \ddots & \dots & \ddots & \ddots & \ddots & \dots & \ddots & \ddots & 0 \\
0 & \vdots & \ddots & \ddots & \ddots & \vdots & \ddots & \ddots & \ddots & \dots & \ddots & \ddots & \ddots & \dots & \ddots & c_{n-2k} \\
f_1 & 0 & \vdots & \ddots & \ddots & \ddots & \vdots & \ddots & \ddots & \ddots & \dots & \ddots & \ddots & \ddots & \dots & 0 \\
0 & f_2 & \ddots & \vdots & \ddots & \ddots & \ddots & \vdots & \ddots & \ddots & \ddots & \dots & \ddots & \ddots & \ddots & \vdots \\
\vdots & 0 & \ddots & \ddots & \vdots & \ddots & \ddots & \ddots & \vdots & \ddots & \ddots & \ddots & \dots & \ddots & \ddots & 0 \\
0 & \vdots & \ddots & \ddots & \ddots & \vdots & \ddots & \ddots & \ddots & \vdots & \ddots & \ddots & \ddots & \dots & \ddots & b_{n-k} \\
g_1 & 0 & \vdots & \ddots & \ddots & \ddots & \vdots & \ddots & \ddots & \ddots & \vdots & \ddots & \ddots & \ddots & \dots & 0 \\
0 & g_2 & \ddots & \vdots & \ddots & \ddots & \ddots & \vdots & \ddots & \ddots & \ddots & \vdots & \ddots & \ddots & \ddots & \vdots \\
\vdots & 0 & \ddots & \ddots & \vdots & \ddots & \ddots & \ddots & \vdots & \ddots & \ddots & \ddots & \vdots & \ddots & \ddots & 0 \\
0 & \cdots & 0 & g_{n-3k} & 0 & \cdots & 0 & f_{n-2k} & 0 & \cdots & 0 & e_{n-k} & 0 & \cdots & 0 & a_n
\end{bmatrix}, \tag{2}$$

where a_i, b_i, c_i, d_i, e_i, f_i and g_i $(i = 1, 2, \dots, n)$ are finite sequences of numbers such that $b_{n-k+1} = \cdots = b_n = 0$, $c_{n-2k+1} = \cdots = c_n = 0$, $d_{n-3k+1} = \cdots = d_n = 0$, $e_{n-k+1} = \cdots = e_n = 0$, $f_{n-2k+1} = \cdots = f_n = 0$, $g_{n-3k+1} = \cdots = g_n = 0$ and $1 \le k < \lfloor \frac{n}{3} \rfloor$ and $n \ge 4$.
For $\lfloor \frac{n}{3} \rfloor \le k < \lfloor \frac{n}{2} \rfloor$ the matrix $T_n^{(k)}$ is a k-pentadiagonal matrix, for $\lfloor \frac{n}{2} \rfloor \le k < n$ the matrix $T_n^{(k)}$ is a k-tridiagonal matrix and for $k \ge n$ is ordinary diagonal matrix.

§2 Main Result

In this section, based on following lemma and [2, 5, 9] we are going to consider the construction of a new computational algorithm for inverting any nonsingular k-heptadiagonal matrix.

Lemma 2.1. *If $D_k (k = 1, 2, \dots, n)$, leading principal minor of order k, of $T_n^{(k)}$ are not zeros, then the matrix $T_n^{(k)}$ has the only one Doolitle LU factorization as*

$$T_n^{(k)} = L_n^{(k)} U_n^{(k)}, \tag{3}$$

where

$$L_n^{(k)} = \begin{bmatrix}
1 & 0 & \cdots & & & & & & & & & & & & \cdots & 0 \\
0 & 1 & & & & & & & & & & & & & & \vdots \\
\vdots & 0 & \ddots & & & & & & & & & & & & & \\
0 & \vdots & \ddots & \ddots & & & & & & & & & & & & \\
x_1 & 0 & \vdots & \ddots & \ddots & & & & & & & & & & & \\
0 & x_2 & \ddots & \vdots & \ddots & \ddots & & & & & & & & & & \\
\vdots & 0 & \ddots & \ddots & \vdots & \ddots & \ddots & & & & & & & & & \\
0 & \vdots & \ddots & \ddots & \ddots & \vdots & \ddots & \ddots & & & & & & & & \\
y_1 & 0 & \vdots & \ddots & \ddots & \ddots & \vdots & \ddots & \ddots & & & & & & & \\
0 & y_2 & \ddots & \vdots & \ddots & \ddots & \ddots & \vdots & \ddots & \ddots & & & & & & \\
\vdots & 0 & \ddots & \ddots & \vdots & \ddots & \ddots & \ddots & \vdots & \ddots & \ddots & & & & & \\
0 & \vdots & \ddots & \ddots & \ddots & \vdots & \ddots & \ddots & \ddots & \vdots & \ddots & \ddots & & & & \\
z_1 & 0 & \vdots & \ddots & \ddots & \ddots & \vdots & \ddots & \ddots & \ddots & \vdots & \ddots & \ddots & & & \\
0 & z_2 & \ddots & \vdots & \ddots & \ddots & \ddots & \vdots & \ddots & \ddots & \ddots & \vdots & \ddots & \ddots & & \vdots \\
\vdots & \ddots & \ddots & \ddots & \vdots & \ddots & \ddots & \ddots & \vdots & \ddots & \ddots & \ddots & \vdots & \ddots & \ddots & 0 \\
0 & \cdots & 0 & z_{n-3k} & 0 & \cdots & 0 & y_{n-2k} & 0 & \cdots & 0 & x_{n-k} & 0 & \cdots & 0 & 1
\end{bmatrix}, \tag{4}$$

$$U_n^{(k)} = \begin{bmatrix}
m_1 & 0 & \dots & 0 & p_1 & 0 & \dots & 0 & t_1 & 0 & \dots & 0 & s_1 & 0 & \dots & 0 \\
0 & m_2 & 0 & \dots & 0 & p_2 & 0 & \dots & 0 & t_2 & 0 & \dots & 0 & s_2 & \ddots & \vdots \\
\vdots & & \ddots & \ddots & \dots & \ddots & \ddots & \ddots & \dots & \ddots & \ddots & \ddots & \dots & \ddots & \ddots & 0 \\
 & & & \ddots & \ddots & \dots & \ddots & \ddots & \ddots & \dots & \ddots & \ddots & \ddots & \dots & \ddots & s_{n-3k} \\
 & & & & \ddots & \ddots & \dots & \ddots & \ddots & \ddots & \dots & \ddots & \ddots & \ddots & \dots & 0 \\
 & & & & & \ddots & \ddots & \dots & \ddots & \ddots & \ddots & \dots & \ddots & \ddots & \ddots & \vdots \\
 & & & & & & \ddots & \ddots & \dots & \ddots & \ddots & \ddots & \dots & \ddots & \ddots & 0 \\
 & & & & & & & \ddots & \ddots & \dots & \ddots & \ddots & \ddots & \dots & \ddots & t_{n-2k} \\
 & & & & & & & & \ddots & \ddots & \dots & \ddots & \ddots & \ddots & \dots & 0 \\
 & & & & & & & & & \ddots & \ddots & \dots & \ddots & \ddots & \ddots & \vdots \\
 & & & & & & & & & & \ddots & \ddots & \dots & \ddots & \ddots & 0 \\
 & & & & & & & & & & & \ddots & \ddots & \dots & \ddots & p_{n-k} \\
 & & & & & & & & & & & & \ddots & \ddots & \dots & 0 \\
 & & & & & & & & & & & & & \ddots & \ddots & \vdots \\
\vdots & & & & & & & & & & & & & & \ddots & 0 \\
0 & \cdots & & & & & & & & & & & & \cdots & 0 & m_n
\end{bmatrix}, \tag{5}$$

and $det(T_n^{(k)}) = \prod_{i=1}^{n} m_i$.

According to (2), (3), (4) and (5), we can get the relations as

$$m_i = \begin{cases} a_i, & i = 1, 2, \dots, k \\ a_i - x_{i-k}p_{i-k}, & i = k+1, \dots, 2k \\ a_i - y_{i-2k}t_{i-2k} - x_{i-k}p_{i-k}, & i = 2k+1, \dots, 3k \\ a_i - z_{i-3k}s_{i-3k} - y_{i-2k}t_{i-2k} - x_{i-k}p_{i-k}, & i = 3k+1, \dots, n. \end{cases} \tag{6}$$

$$x_i = \begin{cases} \dfrac{e_i}{m_i}, & i = 1, 2, \dots, k \\ \dfrac{e_i - y_{i-k}p_{i-k}}{m_i}, & i = k+1, \dots, 2k \\ \dfrac{e_i - z_{i-2k}t_{i-2k} - y_{i-k}p_{i-k}}{m_i}, & i = 2k+1, \dots, n-k. \end{cases} \tag{7}$$

$$p_i = \begin{cases} b_i, & i = 1, 2, \dots, k \\ b_i - x_{i-k}t_{i-k}, & i = k+1, \dots, 2k \\ b_i - y_{i-2k}s_{i-2k} - x_{i-k}t_{i-k}, & i = 2k+1, \dots, n-k. \end{cases} \tag{8}$$

$$y_i = \begin{cases} \dfrac{f_i}{m_i}, & i = 1, 2, \dots, k \\ \dfrac{f_i - z_{i-k}p_{i-k}}{m_i}, & i = k+1, \dots, n-2k. \end{cases} \tag{9}$$

$$t_i = \begin{cases} c_i, & i = 1, 2, \dots, k \\ c_i - x_{i-k}s_{i-k}, & i = k+1, \dots, n-2k. \end{cases} \tag{10}$$

$$z_i = \frac{g_i}{m_i}, \qquad i = 1, 2, \dots, n-3k. \tag{11}$$

$$s_i = d_i, \qquad i = 1, 2, \dots, n-3k. \tag{12}$$

Let $T_n^{(k)}$ be a nonsingular matrix and $(T_n^{(k)})^{-1} = H = (H_1, H_2, \dots, H_n)^t$ where H_j is jth row of $(T_n^{(k)})^{-1}$ for $j = 1, 2, \dots, n$. Through the relation $T_n^{(k)}(T_n^{(k)})^{-1} = E$ (E is the $n \times n$ identity matrix), we can get $L_n^{(k)}(U_n^{(k)}H) = E$. Let

$$L_n^{(k)}X = E, \tag{13}$$

then

$$U_n^{(k)}H = X, \tag{14}$$

where $X = (X_1, \dots, X_n)^t$, $E = (E_1, \dots, E_n)^t$ and $H = (H_1, \dots, H_n)^t$. By (13) we have

$$X_i = \begin{cases} E_i, & i = 1, 2, \dots, k \\ E_i - x_{i-k}X_{i-k}, & i = k+1, \dots, 2k \\ E_i - x_{i-k}X_{i-k} - y_{i-2k}X_{i-2k}, & i = 2k+1, \dots, 3k \\ E_i - x_{i-k}X_{i-k} - y_{i-2k}X_{i-2k} - z_{i-3k}X_{i-3k}, & i = 3k+1, \dots, n. \end{cases} \tag{15}$$

Similarly, by (14) we can get

$$H_i = \begin{cases} \dfrac{X_i}{m_i}, & i = n, n-1, \dots, n-k+1 \\ \dfrac{X_i - p_iH_{i+k}}{m_i}, & i = n-k, \dots, n-2k+1 \\ \dfrac{X_i - t_iH_{i+2k} - p_iH_{i+k}}{m_i}, & i = n-2k, \dots, n-3k+1 \\ \dfrac{X_i - s_iH_{i+3k} - t_iH_{i+2k} - p_iH_{i+k}}{m_i}, & i = n-3k, \dots, 1. \end{cases} \tag{16}$$

Now, relations (6)-(12), (15) and (16) enable us to construct the algorithm.

Algorithm 2.2. ***Symbolic algorithm for inverting a k-heptadiagonal matrix.***
To find the inverse of a k-heptadiagonal matrix in (2)*, we may proceed as follows:*
INPUT: Order of matrix n*, value of* k *and the values of* a_i*,* b_i*,* c_i*,* d_i*,* e_i*,* f_i *and* g_i *for* $i = 1, \dots, n$ *and* $b_{n-k+1} = \cdots = b_n = 0$*,* $c_{n-2k+1} = \cdots = c_n = 0$*,* $d_{n-3k+1} = \cdots = d_n = 0$*,* $e_{n-k+1} = \cdots = e_n = 0$*,* $f_{n-2k+1} = \cdots = f_n = 0$ *and* $g_{n-3k+1} = \cdots = g_n = 0$*.*
OUTPUT:The inverse matrix H*.*

Step 1: ***For*** $i = 1, 2, \dots, k$ ***do***
Set: $m_i = a_i$.
If $m_i = 0$ ***then*** $m_i = u$ ***end if.***
$p_i = b_i,\ t_i = c_i,\ s_i = d_i.$
Compute and simplify:
$x_i = \dfrac{e_i}{a_i},\ y_i = \dfrac{f_i}{a_i},\ z_i = \dfrac{g_i}{a_i}.$
end do.

Step 2: ***For*** $i = k+1, k+2, \dots, 2k$ ***do***
Set $s_i = d_i$.
Compute and simplify:
$m_i = a_i - x_{i-k}p_{i-k}.$
If $m_i = 0$ ***then*** $m_i = u$ ***end if.***
$p_i = b_i - x_{i-k}t_{i-k},$
$x_i = \dfrac{e_i - y_{i-k}p_{i-k}}{m_i},$
$y_i = \dfrac{f_i - z_{i-k}p_{i-k}}{m_i},$
$t_i = c_i - x_{i-k}s_{i-k},$
$z_i = \dfrac{g_i}{m_i}.$
end do.

Step 3: ***For*** $i = 2k+1, 2k+2, \dots, 3k$ ***do***
Set:$s_i = d_i$.
Compute and simplify:
$m_i = a_i - y_{i-2k}t_{i-2k} - x_{i-k}p_{i-k}.$
If $m_i = 0$ ***then*** $m_i = u$ ***end do.***
$p_i = b_i - y_{i-2k}s_{i-2k} - x_{i-k}t_{i-k},$
$x_i = \dfrac{e_i - z_{i-2k}t_{i-2k} - y_{i-k}p_{i-k}}{m_i},$
$y_i = \dfrac{f_i - z_{i-k}p_{i-k}}{m_i},$
$t_i = c_i - x_{i-k}s_{i-k},$
$z_i = \dfrac{g_i}{m_i}.$
end do.

Step 4: ***For*** $i = 3k+1, 3k+2, \dots, n$ ***do***

Set: $s_i = d_i$.
$m_i = a_i - z_{i-3k}s_{i-3k} - y_{i-2k}t_{i-2k} - x_{i-k}p_{i-k}$.
If $m_i = 0$ **then** $m_i = u$ **end do.**
$p_i = b_i - y_{i-2k}s_{i-2k} - x_{i-k}t_{i-k}$,
$x_i = \dfrac{e_i - z_{i-2k}t_{i-2k} - y_{i-k}p_{i-k}}{m_i}$,
$y_i = \dfrac{f_i - z_{i-k}p_{i-k}}{m_i}$,
$t_i = c_i - x_{i-k}s_{i-k}$,
$z_i = \dfrac{g_i}{m_i}$.
end do.

Step 5: Substitute the actual value $u = 0$, *then compute* $det(T_n^{(k)}) = \prod_{i=1}^{n} m_i$. *If* $det(T_n^{(k)}) = 0$, *the matrix* $T_n^{(k)}$ *is singular, then OUTPUT ("Singular matrix"); stop.*

Step 6: **For** $i = 1, 2, \ldots, k$
do $X_i = E_i$.
end do.
For $i = k+1, k+2, \ldots, 2k$
do $X_i = E_i - x_{i-k}X_{i-k}$.
end do.
For $i = 2k+1, 2k+2, \ldots, 3k$
do $X_i = E_i - x_{i-k}X_{i-k} - y_{i-2k}X_{i-2k}$.
end do.
For $i = 3k+1, 3k+2, \ldots, n$
do $X_i = E_i - x_{i-k}X_{i-k} - y_{i-2k}X_{i-2k} - z_{i-3k}X_{i-3k}$.
end do.

Step 7: **For** $i = n, n-1, \ldots, n-k+1$
do $H_i = \dfrac{X_i}{m_i}$.
end do.
For $i = n-k, n-k-1, \ldots, n-2k+1$
do $H_i = \dfrac{X_i - p_iH_{i+k}}{m_i}$.
end do.
For $i = n-2k, n-2k-1, \ldots, n-3k+1$
do $H_i = \dfrac{X_i - t_iH_{i+2k} - p_iH_{i+k}}{m_i}$.
end do.
For $i = n-3k, n-3k-1, \ldots, 1$
do $H_i = \dfrac{X_i - s_iH_{i+3k} - t_iH_{i+2k} - p_iH_{i+k}}{m_i}$.
end do.

Step 8: The inverse matrix is $H|_{u=0}$.

The computational cost of the algorithm (2.2) is given in table below. According to the

Table 1. The computational cost of the algorithm (2.2).

Step	1	2	3	4	5	6	7
$\times, \backslash$	$3k$	$8k$	$13k$	$12n-36k$	n	$3n^2-6kn$	$4n^2-6kn$
$+, -(=)$	$4k$	$6k$	$9k$	$10n-30k$	0	$3n^2-5kn$	$3n^2-6kn$

table 1, it is obvious that the computational cost of the algorithm is $O(n^2)$.

§3 Illustrative examples

In this section we are going to consider two illustrative examples.

Example 3.1. *Find the inverse of the* 8×8 *matrix.*

$$T_8^{(2)} = \begin{bmatrix} 1 & 0 & -1 & 0 & 2 & 0 & 2 & 0 \\ 0 & 1 & 0 & 2 & 0 & 3 & 0 & 1 \\ 1 & 0 & 2 & 0 & 1 & 0 & -1 & 0 \\ 0 & 2 & 0 & 3 & 0 & 4 & 0 & 1 \\ -1 & 0 & -1 & 0 & 2 & 0 & 3 & 0 \\ 0 & 2 & 0 & 3 & 0 & 1 & 0 & 4 \\ 3 & 0 & -2 & 0 & -3 & 0 & 1 & 0 \\ 0 & 4 & 0 & -4 & 0 & 1 & 0 & 3 \end{bmatrix}.$$

Solution:

By applying 2.2 algorithm, we get:

•$\mathbf{m} = (1, 1, 3, -1, \frac{10}{3}, -3, \frac{19}{5}, 24)$.

•$det(T_8^{(2)}) = \prod_{i=1}^{8} m_i = 2736$.

$$X = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & -2 & 0 & 1 & 0 & 0 & 0 & 0 \\ \frac{1}{3} & 0 & \frac{2}{3} & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & 1 & 0 & 0 \\ -\frac{9}{5} & 0 & \frac{7}{5} & 0 & \frac{13}{5} & 0 & 1 & 0 \\ 0 & 20 & 0 & -\frac{49}{3} & 0 & \frac{13}{3} & 0 & 1 \end{bmatrix}$$

$$H = \begin{bmatrix} \frac{5}{19} & 0 & \frac{7}{38} & 0 & -\frac{3}{19} & 0 & \frac{5}{38} & 0 \\ 0 & -\frac{4}{3} & 0 & \frac{35}{36} & 0 & \frac{1}{36} & 0 & \frac{1}{12} \\ -\frac{12}{19} & 0 & \frac{25}{38} & 0 & \frac{11}{19} & 0 & \frac{7}{38} & 0 \\ 0 & -\frac{1}{2} & 0 & \frac{3}{8} & 0 & \frac{1}{8} & 0 & -\frac{1}{8} \\ \frac{10}{19} & 0 & -\frac{5}{38} & 0 & -\frac{6}{19} & 0 & -\frac{9}{38} & 0 \\ 0 & \frac{5}{6} & 0 & -\frac{25}{72} & 0 & -\frac{11}{72} & 0 & \frac{1}{24} \\ -\frac{9}{19} & 0 & \frac{7}{19} & 0 & \frac{13}{19} & 0 & \frac{5}{19} & 0 \\ 0 & \frac{5}{6} & 0 & -\frac{49}{72} & 0 & \frac{13}{72} & 0 & \frac{1}{24} \end{bmatrix}.$$

Example 3.2. *Find the inverse of the* 9×9 *matrix* $T_9^{(2)}$.

$$T_9^{(2)} = \begin{bmatrix} 2 & 0 & 1 & 0 & -1 & 0 & 2 & 0 & 0 \\ 0 & -1 & 0 & 2 & 0 & 1 & 0 & -2 & 0 \\ 1 & 0 & 1 & 0 & 2 & 0 & -2 & 0 & 4 \\ 0 & 1 & 0 & -2 & 0 & 1 & 0 & -1 & 0 \\ 1 & 0 & -1 & 0 & 1 & 0 & -1 & 0 & 2 \\ 0 & -1 & 0 & 2 & 0 & -1 & 0 & 3 & 0 \\ 2 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & -1 \\ 0 & -2 & 0 & -1 & 0 & -1 & 0 & -1 & 0 \\ 0 & 0 & 1 & 0 & 2 & 0 & -2 & 0 & 2 \end{bmatrix}.$$

Solution:

By applying 2.2 algorithm, we get:

•$\mathbf{m} = (2, -1, \frac{1}{2}, u, 9, -2, \frac{13}{9}, \frac{20-9u}{2u}, \frac{-5}{13})$.

•$det(T_9^{(2)}) = (\prod_{i=1}^{9} m_i)_{u=0} = (\frac{90u-200}{2})_{u=0} = -100.$

•$(T_9^{(2)})^{-1} = H|_{u=0}$.

$$X = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -\frac{1}{2} & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ -2 & 0 & 3 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ -\frac{5}{9} & 0 & -\frac{2}{3} & 0 & -\frac{2}{9} & 0 & 1 & 0 & 0 \\ 0 & -\frac{1}{2} & 0 & \frac{5}{u} & 0 & -\frac{1}{2}\frac{-10+3u}{u} & 0 & 1 & 0 \\ \frac{6}{13} & 0 & -\frac{11}{13} & 0 & \frac{5}{13} & 0 & -\frac{3}{13} & 0 & 1 \end{bmatrix}$$

$$H = \begin{bmatrix} \frac{12}{5} & 0 & \frac{-17}{5} & 0 & 2 & 0 & \frac{-6}{5} & 0 & \frac{21}{5} \\ 0 & \frac{4u-6}{20-9u} & 0 & \frac{-13}{20-9u} & 0 & \frac{6u-11}{20-9u} & 0 & \frac{u-8}{20-9u} & 0 \\ \frac{4}{5} & 0 & \frac{-4}{5} & 0 & 0 & 0 & \frac{-2}{5} & 0 & \frac{7}{5} \\ 0 & \frac{2}{20-9u} & 0 & \frac{-9}{20-9u} & 0 & \frac{-3}{20-9u} & 0 & \frac{-4}{20-9u} & 0 \\ -3 & 0 & 4 & 0 & -2 & 0 & 2 & 0 & 5 \\ 0 & \frac{10-7u}{20-9u} & 0 & \frac{25}{20-9u} & 0 & \frac{15-3u}{20-9u} & 0 & \frac{5u}{20-9u} & 0 \\ \frac{-19}{5} & 0 & \frac{29}{5} & 0 & -3 & 0 & \frac{12}{5} & 0 & \frac{-37}{5} \\ 0 & \frac{-u}{20-9u} & 0 & \frac{10}{20-9u} & 0 & \frac{10-3u}{20-9u} & 0 & \frac{2u}{20-9u} & 0 \\ \frac{-6}{5} & 0 & \frac{11}{5} & 0 & -1 & 0 & \frac{3}{5} & 0 & \frac{-13}{5} \end{bmatrix}_{u=0},$$

$$
= \begin{bmatrix}
\frac{12}{5} & 0 & \frac{-17}{5} & 0 & 2 & 0 & \frac{-6}{5} & 0 & \frac{21}{5} \\
0 & \frac{-3}{10} & 0 & \frac{-13}{20} & 0 & \frac{-11}{20} & 0 & \frac{-2}{5} & 0 \\
\frac{4}{5} & 0 & \frac{-4}{5} & 0 & 0 & 0 & \frac{-2}{5} & 0 & \frac{7}{5} \\
0 & \frac{1}{10} & 0 & \frac{-9}{20} & 0 & \frac{-3}{20} & 0 & \frac{-1}{5} & 0 \\
-3 & 0 & 4 & 0 & -2 & 0 & 2 & 0 & -5 \\
0 & \frac{1}{2} & 0 & \frac{5}{4} & 0 & \frac{3}{4} & 0 & 0 & 0 \\
\frac{-19}{5} & 0 & \frac{29}{5} & 0 & -3 & 0 & \frac{12}{5} & 0 & \frac{-37}{5} \\
0 & 0 & 0 & \frac{1}{2} & 0 & \frac{1}{2} & 0 & 0 & 0 \\
\frac{-6}{5} & 0 & \frac{11}{5} & 0 & -1 & 0 & \frac{3}{5} & 0 & \frac{-13}{5}
\end{bmatrix}.
$$

§4 Conclusion

In this work a new numeric and symbolic algorithm has been developed for finding the inverse of a general nonsingular k-heptadiagonal matrix. The algorithm is based on Doolitle LU factorization and it is reliable, computationally efficient. At last, we confirmed the validity of the algorithm by two illustrative examples and we wrote a MAPLE procedure of the algorithm.

References

[1] M El-Mikkawy. *A fast algorithm for evaluating nth order tri-diagonal determinants*, Journal of Computational and Applied Mathematics, 2004, 166(2): 581-584.

[2] M El-Mikkawy. *A Generalized Symbolic Thomas Algorithm*, Applied Mathematics, 2012, 3: 342-345.

[3] M El-Mikkawy, F Atlan. *A novel algorithm for inverting a general k-tridiagonal matrix*, Applied Mathematics Letters, 2014, 32: 41-47.

[4] M El-Mikkawy, A Karawia. *Inversion of general tridiagonal matrices*, Applied Mathematics Letters, 2006, 19(8): 712-720.

[5] J Jia, S Li. *Symbolic algorithms for the inverses of general k-tridiagonal matrices*, Computers and Mathematics with Applications, 2015, 70(12): 3032-3042.

[6] J Jia, S Li. *New algorithms for numerically solving a class of bordered tridiagonal systems of linear equations*, Computers and Mathematics with Applications, 2019, 78(1): 144-151.

[7] J Jia, T Sogabe, M El-Mikkawy. *Inversion of k-tridiagonal matrices with Toeplitz structure*, Computers and Mathematics with Applications, 2013, 65(1): 116-125.

[8] X Zhao, T Huang. *On the inverse of a general pentadiagonal matrix*, Applied Mathematics and Computation, 2008, 20(2): 639-646.

[9] Y Lin, X Lin. *A novel algorithm for inverting a k-pentadiagonal matrix*, The 2016 3rd International Conference on Systems and Informatics (ICSAI 2016), 2016, 578-582.

Department of mathematics, Payamenoor university, Tehran, Iran.
Email: shamssolary@pnu.ac.ir, shamssolary@gmail.com, rasouli.mehran1368@gmail.com