Journal of Applied Analysis and Computation
Volume 12, Number 6, December 2022, 2299–2313
Website:http://www.jaac-online.com
DOI:10.11948/20210441

# ANALYTICALLY EXPLICIT INVERSE OF A KIND OF PERIODIC TRIDIAGONAL MATRIX USING A BACKWARD CONTINUED FRACTION APPROACH

Tim Hopkins[1] and Emrah Kılıç[2,†]

**Abstract** We present a fast algorithm for generating the inverse and the determinant of an extended, periodic, tridiagonal matrix. We use backward continued fractions to generate the elements of the inverse in closed form. By trading memory use against the cost of repeating the computation of certain quantities we are able to produce an effective procedure for a symbolic algebra implementation. We compare the performance of our Maple implementation with that of the standard Maple library procedures for matrix inversion and computation of the determinant on a set of illustrative example matrices.

**Keywords** Matrix inversion, LU-Factorization, backward continued fraction, almost periodic tridiagonal matrix.

**MSC(2010)** 15A09.

## 1. Introduction

Tridiagonal matrix structures occur frequently in both practical application areas and in the numerical approximation of equations resulting from the modelling of physical phenomena; see, for example, [3–6, 13, 20, 21].

Recently, attention has been given to methods for constructing inverses of both simple and periodic tridiagonal matrices algebraically as a means of finding algorithms that would prove efficient when implemented using symbolic (computer) algebra (SA) systems like Maple [19] and Mathematica [23] (for these methods see, for example, [10, 11, 14, 15, 17] and for other properties of both simple and periodic tridiagonal matrices such as eigenvalues, eigenvectors or factorizations see, for example, [2, 7, 8, 14]).

To compute the inverse of a simple tridiagonal matrix efficiently using floating-point arithmetic we would solve a set of $n$ right hand sides with the $j^{th}$ being $I_j$, the $j^{th}$ column of the unit matrix and the solution the $j^{th}$ column of the inverse. Using Gaussian Elimination this requires at least $5n^2 + O(n)$ floating-point operations increasing to $7n^2+O(n)$ if partial pivoting is required to provide numerical stability. Similar $O(n^2)$ algorithms also exist for periodic tridiagonal systems.

The situation is very different for SA systems where, generally, we are not interested in the use of floating-point numbers; here all the computations are performed

†The corresponding author: Email: ekilic@etu.edu.tr (E. Kılıç)
[1]Computing Laboratory, University of Kent, Canterbury, CT2 7NF, UK
[2]TOBB University of Economics and Technology, Mathematics Department, 06560 Ankara, Turkey

algebraically and, therefore, without any computation errors. This means that we do not need to worry about stability or the effects of the build up of errors due to the use of floating-point data; the fast algorithms, when used for algebraic computations, are acceptable for any non-singular tridiagonal matrix.

Closed forms for the elements of the inverse of a general tridiagonal matrix offer a number of possible advantages. First, they allow for the efficient computation of the inverse by taking account of the structure of the matrix. Second, they provide a means for just computing segments of the inverse without computing all the elements.

Comparing the efficiency of algorithms for SA systems is more complex than their floating-point counterparts. Generally a good comparative measure of efficiency for floating-point implementations may be obtained by using the number of basic arithmetic operations required to solve a problem of a particular size. This works because we can assume that each of the basic floating-point arithmetic operations executes at a fixed rate on a given processor and, provided that we have an 'average' operation mix, we can just compare the number of operations required to implement each algorithm. With SA systems the operands are general algebraic quantities and thus the execution time for a basic arithmetic operation depends on the complexity of the two operands. Thus, while operation counts are still useful when comparing SA algorithms, they do not have the same discriminating power as their floating-point cousins.

A periodic tridiagonal matrix, which may also be called a tridiagonal matrix with corners ([12, 22]), is of the form

$$F = \begin{bmatrix} x_1 & y_1 & & & & \delta_2 \\ z_1 & x_2 & y_2 & & & \\ & z_2 & x_3 & \ddots & & \\ & & \ddots & \ddots & \ddots & \\ & & & z_{n-2} & x_{n-1} & y_{n-1} \\ \mu_2 & & & & z_{n-1} & x_n \end{bmatrix}. \tag{1.1}$$

In [15] closed formulae for the elements of the inverse of (1.1) were obtained and it was shown that a Maple implementation of this closed form generally outperformed both the Maple Library procedures and an alternative algorithm proposed by [11] on a variety of example matrices.

In the current article we extend the complexity of the coefficient matrix to that given by $G$ where $G_{kk} = x_k$ for $1 \leq k \leq n$; $G_{kk+1} = y_k$ and $G_{k+1k} = z_k$ for $1 \leq k \leq n-1$; $G_{1n-1} = \delta_1$, $G_{1n} = \delta_2$, $G_{2n} = \delta_3$, $G_{n-11} = \mu_1$, $G_{n1} = \mu_2$, $G_{n2} = \mu_3$ and 0 otherwise. Clearly, this has the form

$$G = \begin{bmatrix} x_1 & y_1 & & & \delta_1 & \delta_2 \\ z_1 & x_2 & y_2 & & & \delta_3 \\ & z_2 & x_3 & \ddots & & \\ & & \ddots & \ddots & \ddots & \\ \mu_1 & & & z_{n-2} & x_{n-1} & y_{n-1} \\ \mu_2 & \mu_3 & & & z_{n-1} & x_n \end{bmatrix}. \tag{1.2}$$

The efficient computation of the determinant of multidiagonal matrices with and without corners is studied in [12].

In the following discussion we also assume that, in (1.2), none of the $x_i$, $y_i$ and $z_i$ values are zero and that the determinant of all the principal minors of the tridiagonal matrix, obtained by replacing all the corner elements of $G$ by zeros, are non-zero.

Our motivation for looking at this problem is the article by Ao and Sun [1] where matrices of a special case of (1.2) are used in the solution of Sturm-Liouville problems arising from applications involving heat conduction and vibrating strings with eigenproblem dependent boundary conditions.

As in [15], from the $LU$ decomposition of the matrix $G$ we generate the matrices $L^{-1}$ and $U^{-1}$ in terms of BCFs. We may then use these to formulate the elements of $G^{-1}$ explicitly along with $\det G$.

In Section 2 we provide basic definitions and results relating to the backward continued fractions that we use (for more details, see [16]) and we define a number of notational definitions that we use throughout the article. The $LU$ decomposition of the matrix $G$ using BCFs is given in Section 3 and the explicit forms of the elements of $G^{-1}$ are presented in Section 4. The proof of the closed forms of the inverse elements closely follows the method used in [15]; we, therefore, quote the explicit forms of the elements but only give a detailed proof of one part of the main result. An implementation of the algorithm in Maple is discussed in Section 5 along with efficiency comparisons with the Maple library routines for determining the inverse and determinant of a general matrix. Our conclusions are presented in Section 6.

# 2. Basic Definitions and Notations

## 2.1. Backward Continued Fractions

Following [17] we define a general backward continued fraction (BCF) as

$$a_n + \cfrac{b_{n-1}}{a_{n-1} + \cfrac{b_{n-2}}{a_{n-2} + \cfrac{}{\ddots \cfrac{}{+\cfrac{b_1}{a_1}}}}} \tag{2.1}$$

where $\{a_i\}_{i=1}^{n}$, $\{b_i\}_{i=1}^{n-1}$ are real and we may then rewrite this in compact form as

$$C^B = \left[a_1 + \frac{b_1}{a_2+}\frac{b_2}{a_3+}\cdots\frac{b_{n-1}}{a_n}\right]_B.$$

Also the $k^{th}$ backward convergent to $C^B$ defined in [17] is

$$C_k^B = \left[a_1 + \frac{b_1}{a_2+}\frac{b_2}{a_3+}\cdots\frac{b_{k-1}}{a_k}\right]_B$$

and

$$C_k^B = \frac{P_k\left(a_k, b_{k-1}\right)}{P_{k-1}\left(a_k, b_{k-1}\right)}, \quad k = 1, \ldots, n, \tag{2.2}$$

where $C_n^B = C^B$ and we also note $C_k^B! = C_k C_{k-1} \ldots C_1 = P_k\left(a_k, b_{k-1}\right)$.

$P_k(a_k, b_{k-1})$ is defined by the recurrence relation

$$P_k(a_k, b_{k-1}) = a_k P_{k-1}(a_k, b_{k-1}) + b_{k-1} P_{k-2}(a_k, b_{k-1}), \quad 2 \le k \le n, \tag{2.3}$$

with the initial conditions

$$P_0(a_k, b_{k-1}) = 1,$$
$$P_1(a_k, b_{k-1}) = a_1.$$

Specifically, we define a general BCF in terms of the entries of the matrix $G$ by setting $a_k = x_k$ and $b_k = -y_k z_k$ in (2.3). Throughout the remainder of this article we will refer to $\{P_k(x_k, -y_{k-1}z_{k-1})\}$ as $\{P_k\}$ whence the recurrence relation (2.3) becomes

$$P_k = x_k P_{k-1} - y_{k-1} z_{k-1} P_{k-2} \tag{2.4}$$

with initial conditions $P_0 = 1$ and $P_1 = x_1$.

## 2.2. Definitions and Notation

For convenience we collect together here a number of definitions that we will use throughout the article.

We assume that

$$z_0 = y_0 = C_0^B = 1 \tag{2.5}$$

and, in general, for any sequence $\{S_r\}$ used in the article we will assume that $S_0 = 1$ unless otherwise stated.

We define scaled versions of the given upper and lower diagonal elements

$$d_r = \frac{z_r}{C_r^B}; \qquad u_r = \frac{y_r}{C_r^B}; \qquad r = 1, 2, \ldots, n-2. \tag{2.6}$$

For $1 \le i \le 3$, we also use

$$\Omega_r = \begin{cases} \dfrac{x_1}{z_1}\delta_3 - \delta_2, & r = 1, \\ \dfrac{x_1}{y_1}\mu_3 - \mu_2, & r = 2, \end{cases} \qquad f_r(i) = \begin{cases} \delta_i, & r = 0, \\ \delta_i \displaystyle\prod_{k=1}^{r} \frac{z_k}{C_k^B}, & 1 \le r \le n-3, \end{cases} \tag{2.7}$$

$$g_r(i) = \begin{cases} \mu_i / x_1, & r = 0, \\ \mu_i \displaystyle\prod_{k=1}^{r+1} \frac{y_{k-1}}{C_k^B}, & 1 \le r \le n-3, \end{cases} \qquad m_r = \begin{cases} -g_0(2), & r = 0, \\ \dfrac{C_1^B}{y_1} g_r(3) - g_r(2), & 1 \le r \le n-3, \\ (-1)^{n-1} A_n / B_n, & r = n-2, \end{cases} \tag{2.8}$$

$$q_r = \begin{cases} -f_0(2), & r = 0, \\ \dfrac{C_1^B}{z_1} f_r(3) - f_r(2), & 1 \le r \le n-3, \end{cases} \tag{2.9}$$

$$T(n,m) = \sum_{k=m}^{n-2} \frac{y_{k-1}! d_{k-1}!}{C_k^B!} = \sum_{k=m-1}^{n-3} \frac{u_k! d_k!}{C_{k+1}^B}, \tag{2.10}$$

$$\begin{aligned} A_n &= z_{n-1} + \sum_{k=0}^{n-3} f_k(1)\, m_k - (-1)^n m_{n-3} y_{n-2} \\ &= z_{n-1} + \left( \delta_1 \Omega_2 T(n,2) - \frac{\delta_1 \mu_2}{x_1} \right) - (-1)^n \Omega_2 u_{n-2}!, \end{aligned} \tag{2.11}$$

$$\begin{aligned}\hat{A}_n &= y_{n-1} + \sum_{k=0}^{n-3} g_k(1)\, q_k - (-1)^n q_{n-3} d_{n-2} \\ &= y_{n-1} - \frac{\mu_1 \delta_2}{x_1} + \mu_1 \Omega_1 T(n,2) - (-1)^n \Omega_1 d_{n-2}!,\end{aligned} \tag{2.12}$$

$$\begin{aligned}B_n &= x_{n-1} - \sum_{k=0}^{n-4} f_k(1)\, g_k(1) - (d_{n-2} - (-1)^n g_{n-3}(1))\,(y_{n-2} - (-1)^n f_{n-3}(1)) \\ &= x_{n-1} - \delta_1 \mu_1 T(n,1) - d_{n-2} y_{n-2} + (-1)^n (\mu_1 u_{n-2}! + \delta_1 d_{n-2}!),\end{aligned} \tag{2.13}$$

$$U_{n,n} = x_n + (-1)^n m_{n-2} \hat{A}_n - \sum_{k=0}^{n-3} m_k q_k = x_n - \frac{A_n \hat{A}_n}{B_n} - \frac{\delta_2 \mu_2}{x_1} - \Omega_1 \Omega_2 T(n,2). \tag{2.14}$$

## 3. LU Factorization

In this section, we give the LU decomposition of the matrix $G$ via backward continued fractions. Suppose that the entries of $G$ are given, that is, we have $x_i$ for $1 \le i \le n$, $y_i$ and $z_i$ for $1 \le i \le n-1$, $\mu_i$ and $\delta_i$ for $1 \le i \le 3$.

We define the unit lower triangular matrix $L$ of order $n$ as

$$L_{ij} = \begin{cases} 1; & i = j, \\ d_j; & i = j+1, 1 \le j \le n-3, \\ d_{n-2} + (-1)^{n-1} g_{n-3}(1); & i = n-1, j = n-2, \\ (-1)^{j-1} g_{j-1}(1); & i = n-1, 1 \le j \le n-3, \\ (-1)^j m_{j-1}; & i = n, 1 \le j \le n-1, \\ 0; & \text{otherwise}, \end{cases} \tag{3.1}$$

where the $g_r$ and $m_r$ are defined by (2.8) and the $d_r$ by (2.6).

Define an upper triangular matrix $U$ of order $n$ by

$$U_{ij} = \begin{cases} C_i^B; & 1 \le i = j \le n-2, \\ y_j; & i = j+1, 1 \le j \le n-3, \\ (-1)^{i-1} f_{i-1}(1); & j = n-1, 1 \le i \le n-3, \\ y_{n-2} - (-1)^n f_{n-3}(1); & i = n-2, j = n-1, \\ -q_0; & i = 1, j = n, \\ (-1)^i q_{i-1}; & 2 \le i \le n-2, j = n, \\ \hat{A}_n; & i = n-1, j = n, \\ B_n; & i = n-1, j = n-1, \\ U_{nn}; & i = n, j = n, \\ 0; & \text{otherwise}, \end{cases} \tag{3.2}$$

where the $C_r^B$, $y_r$, $f_r$, $q_r$, $\hat{A}_n$, $B_n$ and $U_{nn}$ are defined by (2.2), (1.2), (2.7), (2.9), (2.12), (2.13) and (2.14) respectively.

Now we give the $LU$ decomposition of matrix $G$ via the following theorem

**Theorem 3.1.** *The LU decomposition of the matrix $G$ is given by*

$$G = L \cdot U,$$

*where $L$ and $U$ are given by* (3.1) *and* (3.2)*, respectively.*

Then we have the following corollary:

**Corollary 3.1.** *With the matrix $G$ and the backward continued fraction $C_n^B$ defined by* (1.2) *and* (2.2) *respectively with $a_k = x_k$ and $b_k = -y_k z_k$ we have, for $n > 1$,*

$$\det G = U_{n,n} B_n C_{n-2}^B!$$

*where $U_{n,n}$ and $B_n$ are defined by* (2.14) *and* (2.13) *respectively.*

**Corollary 3.2.** *For $n > 1$*

$$\det G = U_{n,n} B_n P_{n-2}, \tag{3.3}$$

*where $\{P_n\}$ is given by* (2.4)*.*

Thus when computing the elements of the inverse of $G$ we obtain the determinant almost for free (2 multiplies) since all the terms of the product, $U_{nn}$, $B_n$, and $P_{n-2}$ in (3.3), are required elsewhere in the computation of the inverse elements.

If only the determinant is required we make no claims as to the efficiency of using (3.3); it may be more efficient to use the pentadiagonal algorithm proposed in [12] and take account of the zero elements in the outer diagonals.

## 4. Inverse of the periodic tridiagonal matrix

Before we present $G^{-1}$ in terms of backward continued fractions, we derive the inverses of $L$ and $U$.

**Lemma 4.1.** *Let the unit lower triangular matrix $L$ have the form* (3.1) *then $Q = L^{-1}$ is defined by*

$$Q_{i,j} = \frac{(-1)^j}{d_{j-1}!} \begin{cases} (-1)^i d_{i-1}! & \text{if } 1 \le j \le i \le n-2, \\ \displaystyle\sum_{k=j-1}^{n-3} d_k! g_k(1) - (-1)^n d_{n-2}! & \text{if } i = n-1, \\ (-1)^n m_{n-2} \displaystyle\sum_{k=j-1}^{n-3} d_k! g_k(1) - \sum_{k=j-1}^{n-2} d_k! m_k & \text{if } i = n \ \&\ 1 \le j < n, \\ 1 & \text{if } i = j = n, \\ 0 & \text{if } i < j. \end{cases}$$

**Lemma 4.2.** *Let the $n \times n$ upper triangular matrix $U$ have the form* (3.2) *then $H = U^{-1}$ is defined by*

$$H_{i,j} = \frac{(-1)^i C_{i-1}^B!}{y_{i-1}!}$$

$$\times\begin{cases}\dfrac{-1}{B_nU_{n,n}}\displaystyle\sum_{t=i-1}^{n-3}\frac{y_t!}{C_{t+1}^B!}\left(\hat{A}_nf_t(1)+q_tB_n\right)+(-1)^n\dfrac{y_{n-2}!\hat{A}_n}{C_{n-2}^B!B_nU_{n,n}}; & \\ & j=n\ \&\ 1\le i\le n-1,\\ \dfrac{1}{B_n}\displaystyle\sum_{t=i-1}^{n-3}\frac{y_t!}{C_{t+1}^B!}f_t(1)-(-1)^n\dfrac{y_{n-2}!}{C_{n-2}^B!B_n}; & j=n-1\ \&\ 1\le i\le n-1,\\ (-1)^j\dfrac{y_{j-1}!}{C_j^B!}; & i\le j\le n-2,\\ 0; & i>j,\end{cases}$$

*and* $H_{n,n}=U_{n,n}^{-1}$.

Assume that $y_iz_i\neq 0$ for $1\le i\le n-1$. Let

$$\begin{aligned}&F(i_1,i_2,i_3,i_4,i_5,i_6,i_7;c_1,c_2)\\ =&\frac{(-1)^{i+j}}{u_{i-1}!d_{j-1}!B_n}\left[B_nT(n,i_1)+(\delta_1T(n,i_2)-(-1)^nu_{n-2}!)(\mu_1T(n,i_3)-(-1)^nd_{n-2}!)\right.\\ &-U_{n,n}^{-1}\left(\hat{A}_n\left[\delta_1T(n,i_4)-(-1)^nu_{n-2}!\right]+B_n(\Omega_1T(n,i_5)-c_1)\right)\\ &\times\left.(m_{n-2}\left[(-1)^n\mu_1T(n,i_6)-d_{n-2}!\right]-\Omega_2T(n,i_7)-c_2)\right].\end{aligned}\tag{4.1}$$

Then we have the following result for the inverse of the matrix $G$.

**Theorem 4.1.** *Denote the inverse of matrix $G$ of order $n$ by $W$. Then*

*I. For $1\le i=j\le n$. There are three subcases*

*(A) If $i=1$, then $W_{11}=F(1,1,1,1,2,1,2;\delta_2/x_1,-\mu_2/x_1)$.*
*(B) If $1<i<n$, then $W_{ii}=F(i,i,i,i,i,i,i;0,0)$.*
*(C) If $i=n$, then $W_{nn}=U_{n,n}^{-1}$.*

*II. For $1\le i<j\le n$, there are three subcases*

*(A) If $1\le i<j=n$, then $W_{in}=H_{in}$.*
*(B) If $1=i<j<n$, then $W_{1j}=F(j,1,j,1,2,j,j;\delta_2/x_1,0)$.*
*(C) If $1<i<j<n$, then $W_{ij}=F(j,i,j,i,i,j,j;0,0)$.*

*III. For $1\le j<i\le n$, there are three subcases*

*(A) If $1\le j<i=n$, then $W_{nj}=U_{n,n}^{-1}Q_{nj}$.*
*(B) If $1=j<i<n$, then $W_{i1}=F(i,i,1,i,i,1,2,0;-\mu_2/x_1)$.*
*(C) If $1<j<i<n$, then $W_{ij}=F(i,i,j,i,i,j,j;0,0)$.*

*Especially, we note*

$$W_{n-1,n-1}=\frac{1}{B_n}\left(1+\frac{\hat{A}_nA_n}{B_nU_{nn}}\right),$$

*which could also be derived from (I)A above. Here $F(-;-)$ is defined by* (4.1) *and $d_n,\hat{A}_n,A_n,B_n$, $T_n$, $u_r$, $U_{n,n}$ and $\Omega_i$ are all defined in Section 2.*

**Proof.** There are three separate cases to consider: the diagonal elements, the strictly lower triangular elements and the strictly upper triangular elements. Here, to conserve space, we just show the proof for the diagonal elements; the upper and lower triangular elements may be confirmed in a similar fashion.

Diagonal elements, $i = j$:

Since $H_{ij} = 0$ for $i > j$ and $W = (LU)^{-1} = HQ$, we have for $1 \leq i < n$

$$\begin{aligned}
W_{ii} =& \sum_{k=1}^{n} H_{i,k}Q_{k,i} = \sum_{k=i}^{n} H_{i,k}Q_{k,i} = \sum_{k=i}^{n-2} H_{i,k}Q_{k,i} + H_{i,n-1}Q_{n-1,i} + H_{i,n}Q_{n,i} \\
=& \sum_{k=i}^{n-2} (-1)^{i} \frac{C_{i-1}^{B}!}{y_{i-1}!} (-1)^{k} \frac{y_{k-1}!}{C_{k}^{B}!} \frac{(-1)^{i}}{d_{i-1}!} (-1)^{k} d_{k-1}! \\
&+ \frac{(-1)^{i} C_{i-1}^{B}!}{y_{i-1}!} \left( \frac{1}{B_n} \sum_{t=i-1}^{n-3} \frac{y_t!}{C_{t+1}^{B}!} f_t(1) - (-1)^{n} \frac{y_{n-2}!}{C_{n-2}^{B}! B_n} \right) \\
&\times \frac{(-1)^{i}}{d_{i-1}!} \left( \sum_{k=i-1}^{n-3} d_k! g_k(1) - (-1)^{n} d_{n-2}! \right) \\
&+ \frac{(-1)^{i} C_{i-1}^{B}!}{y_{i-1}!} \left( \frac{-1}{B_n U_{n,n}} \sum_{t=i-1}^{n-3} \frac{y_t!}{C_{t+1}^{B}!} \left( \hat{A}_n f_t(1) + q_t B_n \right) + (-1)^{n} \frac{y_{n-2}! \hat{A}_n}{C_{n-2}^{B}! B_n U_{n,n}} \right) \\
&\times \frac{(-1)^{i}}{d_{i-1}!} \left( (-1)^{n} m_{n-2} \sum_{k=i-1}^{n-3} d_k! g_k(1) - \sum_{k=i-1}^{n-2} d_k! m_k \right),
\end{aligned}$$

which, by taking $k-1$ instead of $k$ and using the definitions of a number of quantities, gives

$$\begin{aligned}
&\frac{T(n,i)}{u_{i-1}! d_{i-1}!} + \frac{1}{u_{i-1}! d_{i-1}! B_n} \left( \delta_1 T(n,i) - (-1)^{n} u_{n-2}! \right) \left( \mu_1 T(n,i) - (-1)^{n} d_{n-2}! \right) \\
&- \frac{1}{u_{i-1}! d_{i-1}! B_n U_{n,n}} \left( \delta_1 \hat{A}_n T(n,i) + B_n \sum_{t=i}^{n-2} \frac{u_{t-1}!}{C_t^{B}} q_{t-1} - (-1)^{n} u_{n-2}! \hat{A}_n \right) \\
&\times \left( \mu_1 (-1)^{n} m_{n-2} T(n,i) - \sum_{t=i}^{n-1} d_{t-1}! m_{t-1} \right).
\end{aligned}$$

We examine three subcases:

I) $i = 1$

$$\begin{aligned}
W_{11} =& T(n,1) + \frac{1}{B_n} \left( \delta_1 T(n,1) - (-1)^{n} u_{n-2}! \right) \left( \mu_1 T(n,1) - (-1)^{n} d_{n-2}! \right) \\
&- \frac{1}{B_n U_{n,n}} \left( \hat{A}_n \left( \delta_1 T(n,1) - (-1)^{n} u_{n-2}! \right) + B_n \left( -\frac{\delta_2}{x_1} + \Omega_1 T(n,2) \right) \right) \\
&\times \left( m_{n-2} \left[ (-1)^{n} \mu_1 T(n,1) - d_{n-2}! \right] + \frac{\mu_2}{x_1} - \Omega_2 T(n,2) \right),
\end{aligned}$$

as claimed.

II) $1 < i < n$

$$W_{ii} = \frac{1}{u_{i-1}! d_{i-1}! B_n} \left[ B_n T(n,i) + \left( \delta_1 T(n,i) - (-1)^{n} u_{n-2}! \right) \left( \mu_1 T(n,i) - (-1)^{n} d_{n-2}! \right) \right.$$

$$- \frac{1}{U_{n,n}} \left( \hat{A}_n \left( \delta_1 T(n,i) - (-1)^n u_{n-2}! \right) + B_n \Omega_1 T(n,i) \right)$$
$$\times \left( m_{n-2} \left[ \mu_1 (-1)^n T(n,i) - d_{n-2}! \right] - \Omega_2 T(n,i) \right).$$

III) $i = n$

$$W_{nn} = \sum_{k=1}^{n} H_{n,k} Q_{k,n} = \sum_{k=i}^{n} H_{n,k} Q_{k,n} = H_{n,n} Q_{n,n} = H_{n,n} = U_{n,n}^{-1}$$

as required.

We also note that we may obtain a compact definition for the case $i = n - 1$, i.e.,

$$W_{n-1,n-1} = \sum_{k=i}^{n} H_{n-1,k} Q_{k,n-1} = H_{n-1,n-1} + H_{n-1,n} Q_{n,n-1}$$
$$= \frac{1}{B_n} + \frac{\hat{A}_n A_n}{B_n B_n U_{nn}},$$

as claimed. □

# 5. An Efficient Maple Implementation

In this section we look at the efficiency of our proposed algorithm (KA) as a computational method, implemented in Maple, and compare its performance against a pair of Maple library procedures. We also use the Maple implementation to increase our confidence that the algorithm specified in this article does compute the inverse of the matrix $G$ as defined by (1.2).

We begin by looking at the basic arithmetic operation count and assume that add/subtract and multiply/divide may be considered comparable operations. As we noted in Section 1 this is a reasonable assumption for floating-point arithmetic but may not be so useful for algebraic computations. We also consider how we may trade savings in execution time against the use of extra storage.

## 5.1. The Implementation of the Proposed Algorithm

A pseudocode version of KA which attempts to minimize repeated computations as far as is reasonable is given as Algorithm 1. This requires eight temporary arrays using a total of $8n - 6$ extra elements to store reusable calculations. The implemented version uses $5n^2 + 20n - 14$ multiplications and $2n^2 + 2n + 12$ additions.

In addition, our current implementation of Algorithm 1 places the following restrictions on the structure of $G$:

1. $y_i,\ z_i \neq 0;\ i = 1, \ldots, n-1$, and
2. $x_1 \neq 0$, and
3. the determinants of all the principal minors of $G$ must be non-zero; i.e., $p_i \neq 0,\ i = 1, \ldots, n-1$. Condition 2. above is a consequence of this requirement.

We note that none of these restrictions taken on their own would signify that (1.2) was singular; however, each causes a division by zero at some point in the execution of our algorithm. We suspect that some or all of these restrictions could be removed by re-arranging the computation in some way but we could not discover how this may be achieved. All these conditions are checked by our implementation and an error flag is raised if any are not met.

For the Maple implementation it is vital to control the complexity of the algebraic expressions generated during the computation; failure to do this often leads to a dramatic increase in the number of terms in computed quantities and a corresponding increase in execution times. A good strategy appears to be to simplify all intermediate results in order, hopefully, to reduce the complexity of further calculations.

## 5.2. Choice of Sample Problems

Because a lower operation count does not translate directly into a more efficient symbolic algebra implementation (see Section 1 and [15] for more details). We are often unable to make a categorical choice that one implementation will always be more efficient over the whole problem class.

We have run our implementation on a wide variety of different matrices of the form (1.2) and this has allowed us to provide some indications of when our implementation may prove more efficient that the Maple Library procedures.

Three examples have been chosen to represent different classes of matrices and provide timing comparisons that illustrate the wide range of effectiveness of both methods of generating inverses.

### 5.2.1. Example Problems

**Example 5.1.** This example uses as its base the symmetric tridiagonal inverse of the Lehmer matrix [18] defined by

$$G_{ij} = \begin{cases} \dfrac{4i^3}{4i^2-1} & i = j,\ 1 \le i < n, \\ \dfrac{n^2}{2n-1} & i = j = n, \\ -\dfrac{i(i+1)}{2i+1} & j = i+1,\ 1 \le i \le n-1 \text{ and } j = i-1,\ 2 \le i \le n, \\ 0 & \text{otherwise} \end{cases} \tag{5.1}$$

which we have augmented with $\delta = [n/4, 3n/4, -n/4]$ and $\mu = [-n/3, 2n/3, n/3]$.

We use this as a representative of matrices whose elements are all rational numbers. The operation count may still not be a good efficiency metric as the different sizes of numerators and denominators mean that the time required for each arithmetic operation is not constant.

**Example 5.2.** This matrix is an extension to the one given in [11]

$$G_{ij} = \begin{cases} a & i = j,\ 1 \le i < n, \\ 1 & j = i+1,\ 1 \le i \le n-1, \\ -1 & j = i-1,\ 2 \le i \le n, \\ 0 & \text{otherwise} \end{cases} \tag{5.2}$$

with $\delta = [a/2, -a, -a/2]$ and $\mu = [-a/2, a, a/2]$.

We use this as an example of a simple algebraic matrix that is defined in terms of just a single free variable and rational numbers.

**Example 5.3.** This is an example matrix based on the tridiagonal inverse of the general KMS matrix [9, pp. E190–E191] and defined by

$$G_{ij} = \begin{cases} 1/(1-\sigma\rho) & i = j = 1, n, \\ (1+\sigma\rho)/(1-\sigma\rho) & i = j = 2, \ldots, n-1, \\ -\sigma/(1-\sigma\rho) & j = i-1,\ 2 \le i \le n, \\ -\rho/(1-\sigma\rho) & j = i+1,\ 1 \le i \le n-1, \\ 0 & \text{otherwise} \end{cases} \tag{5.3}$$

which we have augmented with $\delta = [-\rho, \sigma^2, -\sigma]$ and $\mu = [\sigma, \sigma\rho, \rho]$.

Here we have a matrix with more complex algebraic elements.

## 5.3. Performance of the Implementations

All the timings presented below are given in seconds and were achieved using Maple 2019 running under Ubutu 18.04 on an eight thread, four core, Intel(R) Core i7-8650U, 1.9Ghz CPU with 32GB of memory. The tests were all performed while the machine was very lightly loaded; i.e., it was not connected to the network and only a single window was open for executing the run-scripts (for more details see [15, Section 6.4]). All tests were executed, using our implementation of KA and the Maple library procedures, for multiple values of $n$ and both $\det G$ and the computed elements of $G^{-1}$ were compared for equality.

Sample execution times are quoted in Figure 1 for Examples 5.1–5.3 and Figure 2 gives the largest order of matrix that could be inverted utilising $< 10$ seconds of processor time on our test platform.

| Example 5.1 | | | Example 5.2 | | | Example 5.3 | | |
|---|---|---|---|---|---|---|---|---|
| $n$ | $MAPLE$ | $KA$ | $n$ | $MAPLE$ | $KA$ | $n$ | $MAPLE$ | $KA$ |
| 50 | 0.13 | 0.09 | 20 | 0.04 | 0.09 | 10 | 0.01 | 0.79 |
| 100 | 0.92 | 0.62 | 40 | 7.47 | 3.28 | 30 | 0.08 | 6.12 |
| 200 | 7.23 | 4.56 | 60 | 77.14 | 12.23 | 40 | 0.21 | 12.84 |

**Figure 1.** Comparison of execution times for MAPLE and KA on the test problems

| Example | $MAPLE$ | $KA$ |
|---|---|---|
| 5.1 | 250 | 300 |
| 5.2 | 50 | 60 |
| 5.3 | 140 | 40 |

**Figure 2.** Comparison of the maximum order of the test examples that can be inverted in less than 10 seconds.

In the case of matrices using just rational numbers our implementation outperformed the Maple Library procedures but, as illustrated by Example 5.1, this may still not reflect the apparent savings in operation counts. Example 5.2 provides an

example where the KA implementation does extremely well and Example 5.3 shows a case where the Maple routine.

These examples illustrate very well the difficulties associated with comparing performance between symbolic algebra algorithms. For Example 5.1 it would seem likely that the two implementations would be fairly even since, although the Maple library routines operate on a full matrix, many of the extra arithmetic operations will involve one or both of the operands being zero and, thus, complete very cheaply. The Example 5.1 KA timings show a saving of around a third over the Maple procedures and this is fairly typical for this type of matrix.

Examples 5.2 and 5.3 provide very different performance profiles for matrices with algebraic elements. The simpler algebraic matrix shows the increasing efficiency of KA as the order of the matrix increases with a factor six difference by $n = 60$. This contrasts sharply with Example 5.3, which has elements which are slightly more complex than Example 5.2, where the Maple procedures outperform KA by a factor of 60 by $n = 40$.

When the matrix elements are algebraic we have found that the two routines both exhibit a wide spectrum of performances. Thus, in a situation where exact inverses are required for a sequence of matrices of increasing order, and where the elements are of similar algebraic structure, it would be sensible to compare timings of the two implementations on some smaller orders and use these to inform a decision on a suitable choice for higher orders.

Finally, we note that our algorithm is unsuitable for use with floating-point arithmetic due to possible build up of rounding errors.

# 6. Conclusions and Future Work

We have used BCFs to generate explicit expressions for the elements of the inverse of the matrix, $G$, defined by (1.2). These expressions were then used to construct an algorithm and its efficient implementation in Maple to compute the inverse, $G^{-1}$. By exploiting the structure of the matrix we were able to generate a 'fast' algorithm for computing the inverse algebraically by reducing the order of complexity to $7n^2 + O(n)$.

Our performance results illustrate clearly the difficulty of identifying an effective implementation that will perform consistently well over the full range of algebraic problems. The operation count and the sparse regular pattern of the elements in the matrix would suggest that the proposed algorithm should perform better than the Maple library procedures; this is not borne out by our results. Our timing comparisons do, however, show that for some problems our proposed method could be a worthwhile alternative.

In the future we intend to study a number of other published algorithms for computing inverses of matrices with special structures to ascertain whether their implementations could prove competitive with the Maple routines for some classes of problems.

# References

[1] J. Ao and J. Sun, *Matrix representations of Sturm-Liouville problems with coupled eigenparameter-dependent boundary conditions*, Applied Mathematics and Computation, 2014, 244, 142–148.

**ALGORITHM 1:** Pseudocode for the KA using temporary storage to save reusable computations.

**Procedure** $KA(x, y, z, \delta, \mu)$;
**Input:** $x_{1..n}$, $y_{1..n-1}$, $z_{1..n-1}$, $\delta_{1..3}$, $\mu_{1..3}$
**Output:** $sing$, $det$, $A^{-1}$
#
$sing = FALSE; sg = -1;$
**if** $n \mod 2 == 0$ **then** $sg = 1;$
;
# Compute $D_i = d_i!$, $U_i = u_i!$ and $c_i$
# This requires the values of $z_i!$, $y_i!$ and $p_i$. However these
# values are not required in further calculations so they don't need to
be stored
$(p_1, p_2, yf, zf) = (x_1, 1, y_1, z_1)$; $(\Omega_1, \Omega_2) = (x_1\delta_3/z_1 - \delta_2,\ x_1\mu_3/y_1 - \mu_2);$
$(c_1, U_0, U_1, D_0, D_1) = (x_1, 1, y_1/x_1, 1, z_1/x_1);$
**for** $i = 2,\ n-2$ **do**
  $(yf, zf) = (yf \times y_i, zf \times z_i);$
  $p_0 = x_i p_1 - y_{i-1} z_{i-1} p_2;\ c_i = p_0/p_1;$
  $(D_i, U_i) = (zf/p_0, yf/p_0)$; $p_2 = p_1;\ p_1 = p_0;$
**end**
# We need $p_{n-2}$ for the determinant
$p_{n-2} = p_1;$
# Compute the $t_i$ for $i = 1, \ldots, n$
$t_{n-2..n} = (U_{n-3} D_{n-3}/c_{n-2}, 0, 0);$
**for** $i = n-3,\ 1,\ -1$ **do**
  $t_i = t_{i+1} + U_{i-1} D_{i-1}/c_i;$
**end**
# Compute $a = A_n$, $\hat{a} = \hat{A}_n$, $b = B_n$ and $u_{nn} = U_{nn}$
$a = z_{n-1} + \delta_1(\Omega_2 t_2 - \mu_2/x_1) - sg \times \Omega_2 U_{n-2};$
$\hat{a} = y_{n-1} - \mu_1\delta_2/x_1 + \Omega_1(\mu_1 t_2 - sg \times D_{n-2});$
$b = x_{n-1} - \delta_1\mu_1 t_1 - (z_{n-2} y_{n-2}/c_{n-2} - sg \times (\mu_1 U_{n-2} + \delta_1 D_{n-2}));$
$u_{nn} = x_n - \hat{a}a/b - \delta_2\mu_2/x_1 - \Omega_1\Omega_2 t_2;$
# Compute the determinant using (3.3)
$det = p_{n-2} u_{nn} b;$
**if** $det == 0$ **then return** $(TRUE, 0, NULL);$
;
# We require only element $n-2$ of the m array for further computation
$m_{n-2} = -sg \times a/b;\ c1 = \delta_2/x_1;\ c2 = \mu_2/x_1;$
# Precompute and store to save repeated calculations
# The storage used for the $c$ array may now be reused
# We use $v, w, e, g$, and $h$ as temporary arrays
$v_{1..n} = \delta_1 t_{1..n} - sg \times U_{n-2};\ w_{1..n} = \mu_1 t_{1..n} - sg \times D_{n-2};$
$e_{1..n-1} = b t_{1..n-1}\ \ s_{0..n-2} = b \times U_{0..n-2};$
$g_1 = (\hat{a}v_1 + b(\Omega_1 t_2 - c_1))/u_{nn};\ q = b\Omega_1;$
$g_{2..n-1} = (\hat{a}v_{2..n-1} + q t_{2..n-1})/u_{nn};$
$h_1 = sg \times m_{n-2} w_1 - \Omega_2 t_2 + c2;\ h_{2..n-1} = sg \times m_{n-2} w_{2..n-1} - \Omega_2 t_{2..n-1};$
# Form the inverse
# Diagonal elements
$A^{-1}_{11} = (e_1 + v_1 w_1 - g_1 h_1)/b;$
**for** $i = 2,\ n-1$ **do**
  $A^{-1}_{i,i} = (e_i + v_i w_i - g_i h_i)/(s_{i-1} D_{i-1});$
**end**
$A^{-1}_{nn} = 1/u_{nn};\ A^{-1}_{1,n} = g_1/b;\ A^{-1}_{n,1} = -h_1/u_{nn};$

**ALGORITHM 2:** Pseudocode for the KA using temporary storage to save reusable computations continued

```
# First row
A^{-1}_{1,2:n-1:2} = -(e_{2:n-1:2} + v_1 w_{2:n-1:2} - g_1 h_{2:n-1:2})/(D_{1:n-2:2} b);
A^{-1}_{1,3:n-1:2} =  (e_{3:n-1:2} + v_1 w_{3:n-1:2} - g_1 h_{3:n-1:2})/(D_{2:n-2:2} b);
# First column
A^{-1}_{2:n-1:2,1} = -(e_{2:n-1:2} + v_{2:n-1:2} w_1 - g_{2:n-1:2} h_1)/(s_{1:n-2:2});
A^{-1}_{3:n-1:2,1} =  (e_{3:n-1:2} + v_{3:n-1:2} w_1 - g_{3:n-1:2} h_1)/(s_{2:n-2:2});
# Upper triangle
k = -1;
for j = 3, n - 1 do
    T = k/(D_{j-1} b);
    A^{-1}_{2:j-1:2,j} = T/U_{1:j-2:2}(e_j + v_{2:j-1:2} w_j - g_{2:j-1:2} h_j);
    A^{-1}_{3:j-1:2,j} = -T/U_{2:j-2:2}(e_j + v_{3:j-1:2} w_j - g_{3:j-1:2} h_j);
    k = -k;
end
# Column n
A^{-1}_{2:n-1:2,n} = -g_{2:n-1:2}/(s_{1:n-2:2});
A^{-1}_{3:n-1:2,n} =  g_{3:n-1:2}/(s_{2:n-2:2});
# Lower triangle
k = -1;
for i = 3, n - 1 do
    T = k/_{,i-1};
    A^{-1}_{i,2:i-1:2} =  T/D_{1:i-1:2}(e_i + v_i w_{2:i-1:2} - g_i h_{2:i-1:2});
    A^{-1}_{i,3:i-1:2} = -T/D_{2:i-1:2}(e_i + v_i w_{3:i-1:2} - g_i h_{3:i-1:2});
    k = -k;
end
# Row n
A^{-1}_{n,2:n-1:2} =  h_{2:n-1:2}/(D_{1:n-2:2} u_{nn});
A^{-1}_{n,3:n-1:2} = -h_{3:n-1:2}/(D_{2:n-2:2} u_{nn});
return (FALSE, det, A^{-1})
```


[2] A. Ayzenberg, *Space of isospectral periodic tridiagonal matrices*, Algebr. Geom. Topol., 2020, 20, 2957–2994.

[3] M. G. Beker, G. Cella, R. DeSalvo, et al., *Improving the sensitivity of future GW observatories in the 1–10 Hz band: Newtonian and seismic noise*, General Relativity and Gravitation, 2011, 43(2), 623–656.

[4] A. Bunse-Gerstner, R. Byers and V. Mehrmann, *A chart of numerical methods for structured eigenvalue problems*, SIAM J. Matrix Anal. Appl., 1992, 13(2), 419–453.

[5] R. A. Bustos-Marún, E. A. Coronado and H. M. Pastawski, *Buffering plasmons in nanoparticle wave guides at the virtual-localized transition*, Phys. Rev. B, 2010, 82(3), 035434.

[6] B. K. Choudhury, *Diffusion of heat in multidimensional composite spherical body*, IMA J. Appl. Math., 2013, 78(3), 474–493.

[7] C. M. da Fonseca, *On the eigenvalues of some tridiagonal matrices*, J. Comput. Appl. Math., 2007, 200(1), 283–286.

[8] C. M. da Fonseca and V. Kowalenko, *Eigenpairs of a family of tridiagonal matrices: three decades later*, Acta Mathematica Hungarica, 2019, 160, 376–389.

[9] M. Dow, *Explicit inverses of Toeplitz and associated matrices*, ANZIAM Journal, 2008, 44, E185–E215.

[10] M. E. A. El-Mikkawy and F. Atlan, *A new recursive algorithm for inverting general k-tridiagonal matrices*, Appl. Math. Lett., 2015, 44, 34–39.

[11] M. A. El-Shehawey, G. A. El-Shreef and A. S. Al-Henawy, *Analytical inversion of general periodic tridiagonal matrices*, J. Math. Anal. Appl., 2008, 345, 123–134.

[12] K. Filipiak, A. Markiewicz and A. Sawikowski, *Determinants of multidiagonal matrices*, Electron. J. Linear Al., 2012, 25, 102–188.

[13] C. F. Fischer and R. A. Usmani, *Properties of some tridiagonal matrices and their application to boundary value problems*, SIAM J. Numer. Anal., 1969, 6(1), 127–142.

[14] Y. Fu, X. Jiang, Z. Jiang and S. Jhang, *Inverses and eigenpairs of tridiagonal Toeplitz matrix with opposite-bordered rows*, J. Appl. Anal. Comput., 2020, 10(4), 1599–1613.

[15] T. Hopkins and E. Kılıç, *An analytical approach: Explicit inverses of periodic tridiagonal matrices*, Journal of Computational and Applied Mathematics, 2018, 335, 207–226.

[16] W. B. Jones and W. J. Thron, *Continued fraction, analytic theory and applications*, Addison Wesley, Reading, MA, 1980.

[17] E. Kılıç, *Explicit formula for the inverse of a tridiagonal matrix by backward continued fractions*, Appl. Math. Comput., 2008, 197(1), 345–357.

[18] D. H. Lehmer, D. M. Smiley, M. F. Smiley and J. Williamson, *Solution to problem E710*, The American Mathematical Monthly, 1946, 53(9), 534–535.

[19] Maplesoft, *a division of Waterloo Maple Inc., Maple 2019*, Waterloo, Ontario, Canada.

[20] R. M. M. Mattheij and M. D. Smooke, *Estimates for the inverse of tridiagonal matrices arising in boundary-value problems*, Linear Algebra Appl., 1986, 73, 33–57.

[21] G. Meurant, *A review on the inverse of symmetric tridiagonal and block tridiagonal matrices*, SIAM J. Matrix Anal. Appl., 1992, 13(3), 707–728.

[22] L. G. Molinari, *Determinants of block tridiagonal matrices*, Linear Algebra Appl., 2008, 429, 2221–2226.

[23] Wolfram Research, Inc., *Mathematica, Version 12.0*. Champaign, IL, 2019.