Generalized Diagonal Pivoting Methods for Tridiagonal Systems without Interchanges

Jennifer B. Erway, Roumef M. Marcia, and Joseph A. Tyson

Abstract—It has been shown that a nonsingular symmetric tridiagonal linear system of the form $Tx = b$ can be solved in a backward-stable manner using diagonal pivoting methods, where the $LBL^T$ decomposition of $T$ is computed, i.e., $T = LBL^T$, where $L$ is unit lower triangular and $B$ is block diagonal with $1 \times 1$ and $2 \times 2$ blocks. In this paper, we generalize these methods for solving unsymmetric tridiagonal matrices. We present three algorithms that compute the $LBM^T$ factorization, where $L$ and $M$ are unit lower triangular and $B$ is block diagonal with $1 \times 1$ and $2 \times 2$ blocks. These algorithms are normwise backward stable and reduce to the $LBL^T$ factorization when applied to symmetric matrices. We demonstrate the robustness of the algorithms for the unsymmetric case using a wide range of well-conditioned and ill-conditioned linear systems. Numerical results suggest that these algorithms are comparable to Gaussian elimination with partial pivoting (GEPP). However, unlike GEPP, these algorithms do not require row interchanges, and thus, may be used in applications where row interchanges are not possible. In addition, substantial computational savings can be achieved by carefully managing the nonzero elements of the factors $L$, $B$, and $M$.

Index Terms—linear algebra, tridiagonal systems, diagonal pivoting, Gaussian elimination.

I. INTRODUCTION

A nonsingular tridiagonal linear system of the form

$$Tx = b,$$  
(1)

where $T \in \mathbb{R}^{n \times n}$ and $x$ and $b \in \mathbb{R}^n$, is often solved using matrix factorizations. If $T$ is symmetric and positive definite, then the Cholesky decomposition or the $LDL^T$ factorization, where $L$ is a lower triangular matrix and $D$ is a diagonal matrix, can be used to solve (1). If $T$ is symmetric but indefinite, then diagonal pivoting methods can be used to form the $LBL^T$ factorization, where $B$ is block diagonal with either $1 \times 1$ or $2 \times 2$ blocks, with row and/or column permutations (e.g., [1], [2], [3], [4]) or without (e.g., [2], [5], [6], [7]). Finally, if $T$ is unsymmetric, then (1) can be solved using Gaussian elimination with full pivoting or with partial pivoting (GEPP).

Based on previous works by the authors [8], [9], this paper generalizes diagonal pivoting methods for forming a backward-stable $LBM^T$ decomposition of a nonsingular tridiagonal matrix $T$ without interchanges, where $B$ is a block diagonal matrix with either $1 \times 1$ or $2 \times 2$ blocks and $L$ and $M$ are unit-lower tridiagonal matrices. In this paper, we present three algorithms, two of which do not require the full matrix to be known $a$ priori to form the factorization. We demonstrate that the resulting $L$, $B$, and $M$ factors from the algorithm can be used to solve the linear system $Tx = b$ with accuracy comparable to Gaussian elimination with partial pivoting (GEPP). However, unlike GEPP, the algorithms we present do not use row interchanges, making them particularly useful in look-ahead Lanczos methods [10] and composite-step bi-conjugate gradient methods [11] for solving unsymmetric linear systems.

II. DIAGONAL PIVOTING

Let $T \in \mathbb{R}^{n \times n}$ denote the unsymmetric nonsingular tridiagonal matrix

$$T = \begin{bmatrix} \alpha_1 & \gamma_2 & 0 & \cdots & 0 \\ \beta_2 & \alpha_2 & \gamma_3 & \cdots & 0 \\ 0 & \beta_3 & \alpha_3 & \cdots & 0 \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & \beta_n & \alpha_n \end{bmatrix}.$$  
(2)

We partition $T$ in the following manner:

$$T = \begin{bmatrix} d & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & d \end{bmatrix} \begin{bmatrix} B_1 & T_{12} \\ T_{21} & T_{22} \end{bmatrix}.$$  
(3)

The computation of the $LBM^T$ factorization, where $L$ and $M$ are unit lower triangular and $B$ is block diagonal with $1 \times 1$ and $2 \times 2$ blocks, involves choosing the dimension $(d = 1$ or $2)$ of the pivot $B_1$ at each stage:

$$T = \begin{bmatrix} I_d & 0 \\ T_{21} & B_1^{-1}I_{n-d} \end{bmatrix} \begin{bmatrix} B_1 & 0 \\ 0 & S_d \end{bmatrix} \begin{bmatrix} I_d & B_1^{-1}T_{12} \\ 0 & I_{n-d} \end{bmatrix},$$  
(4)

where $S_d = T_{22} - T_{21}B_1^{-1}T_{12} \in \mathbb{R}^{(n-d) \times (n-d)}$. It can be shown that an invertible $B_1$ exists for some $d$ and that the Schur complement $S_2$ is tridiagonal so that the factorization can be defined recursively. Specifically, for $d = 1$,

$$S_1 = T_{22} - \frac{\beta_2\gamma_2}{\alpha_1} e_1^{(n-1)}e_1^{(n-1)^T},$$  
(5)

where $e_1^{(n-1)}$ is the first column of the $(n - 1) \times (n - 1)$ identity matrix. For $d = 2$,

$$S_2 = T_{22} - \left( \frac{\alpha_1\beta_2\gamma_3}{\Delta} \right) e_1^{(n-2)}e_1^{(n-2)^T},$$

where

$$\Delta = \alpha_1\alpha_2 - \beta_2\gamma_2$$

is the determinant of the $2 \times 2$ pivot $B_1$ and $e_1^{(n-2)}$ is the first column of the $(n - 2) \times (n - 2)$ identity matrix. In both
cases $S_d$ and $T_{22}$ differ only in the $(1,1)$ entry. Thus, the LBM$i$ factorization can be recursively defined to obtain the matrices $L, B$, and $M$, all the pivoting methods in this paper can be completely described by looking at the first stage of the factorization.

III. SYMMETRIC TRIDIAGONAL PIVOTING

Diagonal pivoting methods have its origins in the seminal works of Bunch [5], Bunch and Parlett [1], and Bunch and Kaufman [2] and have since been extended and expanded (see e.g., [3], [12], [13]). The factorization methods in these works generally solve dense symmetric indefinite systems using row and column interchanges. However, for symmetric tridiagonal matrices, diagonal pivoting methods have been developed that do not require row or column interchanges. In this section, we review these diagonal pivoting methods. Specifically, we let $\gamma_j = |\beta_j|$ for $j = 2,3,\cdots,n$ in (2), and consider the following nonsingular symmetric tridiagonal matrix:

$$T = \begin{bmatrix} \alpha_1 & \beta_2 & 0 & \cdots & 0 \\ \beta_2 & \alpha_2 & \beta_3 & \cdots & 0 \\ 0 & \beta_3 & \alpha_3 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & \cdots & \beta_n \end{bmatrix}.$$  

(6)

The tridiagonal $T$ can be partitioned similar to as before (see (3)). However, in the symmetric case, $B_1$ and $T_{22}$ are symmetric and $T_{12} = T_{21}$.

A. Bunch pivoting strategy

The diagonal pivoting method of Bunch [5] uses a $1 \times 1$ pivot (i.e., $d = 1$) if the diagonal pivot is sufficiently large relative to the sub-diagonal entry. More precisely, if

$$\sigma = \max_{i,j} |T_{i,j}|,$$  

(7)

a $1 \times 1$ pivot is used in the first step if

$$|\alpha_1| \geq \kappa |\beta_2|^2.$$  

The choice of $\kappa$, which is a root of the equation $\kappa^2 + \kappa - 1 = 0$, balances the element growth in the Schur complement for both pivot sizes by equating the maximal element growth. Here

$$\kappa = \frac{\sqrt{5} - 1}{2} \approx 0.62.$$  

(8)

The Bunch pivoting strategy can be summarized as follows:

Algorithm B. This algorithm describes the first step in the pivoting strategy for symmetric tridiagonal matrices proposed in [5].

$$\kappa = (\sqrt{5} - 1)/2 \approx 0.62$$

$$\sigma = \max_{i,j} |T_{i,j}|$$

if $|\alpha_1| \geq \kappa |\beta_2|^2$

$d_B = 1$

else

$d_B = 2$

end

A recursive application of Algorithm B yields a factorization $T = LBL^T$, where $L$ is unit lower triangular and $B$ is block diagonal. (In fact, one can show that $L$ and $M$ are such that $L_{i,j} = 0$ for $i - j > 2$.)

B. Bunch-Kaufman pivoting strategy

The pivoting strategy of Bunch requires that the largest element of $T$ be known a priori. In some instances, we would like to be able to form the factorization as $T$ is formed. The pivoting strategy of Bunch and Kaufman [2] circumvents this requirement by defining $\sigma_1$, which is the largest element in magnitude “nearby” rather than the global maximum of $T$ in (7). More precisely, a $1 \times 1$ pivot is used in the first iteration if

$$|\sigma_1| \geq \kappa |\beta_2|^2,$$

where

$$\sigma_1 = \max\{|\alpha_2|,|\beta_2|,|\beta_3|\}.$$  

Note that this strategy is different from their well-known symmetric factorization using partial pivoting (see Sec. 4.2 of [2]). The Bunch-Kaufman pivoting strategy can be summarized as follows:

Algorithm BK. This algorithm is the tridiagonal pivoting strategy proposed in [2].

$$\kappa = (\sqrt{5} - 1)/2 \approx 0.62$$

$$\sigma_1 = \max\{|\alpha_2|,|\beta_2|,|\beta_3|\}$$

if $|\alpha_1| |\sigma_1| \geq \kappa |\beta_2|^2$, 

$d_{BK} = 1$

else

$d_{BK} = 2$

end

C. Bunch-Marcia pivoting strategy

Recently, Bunch and Marcia [6], [7] proposed diagonal pivoting methods for symmetric indefinite tridiagonal matrices based on keeping the elements of the tridiagonal matrix $L$ small, which is often an important property in many applications such as optimization. Here, we explicitly describe this pivoting strategy.

Let $L_1 = T_{21}B_{21}^{-1}$ in (4). If a $1 \times 1$ pivot is used, then the $(1,1)$ element of $L_1$ is

$$(L_1)_{1,1} = \frac{\beta_2}{\alpha_1}.$$  

(9)

If a $2 \times 2$ pivot is used, then the $(1,1)$ and $(1,2)$ elements of $L_1$ are

$$(L_1)_{1,1} = -\frac{\beta_2 \beta_3}{\Delta}, \quad \text{and} \quad (L_1)_{1,2} = \frac{\alpha_1 \beta_3}{\Delta}.$$  

(10)

With the elements of $L_1$ for a $2 \times 2$ pivot scaled by the constant $\kappa$ (described above from the Bunch pivoting strategy [5]), Algorithm BM chooses a $1 \times 1$ pivot if the $(L_1)_{1,1}$ in (9) is smaller than the largest entry in (10) in magnitude, i.e.,

$$\frac{|\beta_2|}{|\alpha_1|} \leq \max \kappa \left\{ \frac{|\beta_2 \beta_3|}{\Delta}, \frac{|\alpha_1 \beta_3|}{\Delta} \right\},$$  

(11)

and a $2 \times 2$ pivot is chosen otherwise. In other words, Algorithm BM chooses pivot sizes based on whichever leads
to smaller entries (in magnitude) in the computed factor \( L \).
In [7], an additional criterion is imposed that if
\[
|\alpha_1 \alpha_2| \geq \kappa |\beta_2|^2
\]
a 1 \times 1 pivot is chosen. This additional criterion guarantees
that if \( T \) is positive definite, the \( \text{LBL}^T \) factorization reduces to the \( \text{LDL}^T \) factorization. The choice of pivot size in the
first iteration is described as follows:

**Algorithm BM.** This algorithm is the simplified tridiagonal
pivoting strategy for symmetric tridiagonal matrices proposed in [7].
\[
\kappa = (\sqrt{5} - 1)/2 \approx 0.62
\]
\[
\Delta = \alpha_1 \alpha_2 - \beta_2^2
\]
if \(|\alpha_1 \alpha_2| \geq \kappa |\beta_2|^2\) or \(|\Delta| \leq \kappa |\alpha_1 \beta_2|\) or \(|\beta_2 |\Delta| \leq \kappa (|\alpha_1|^2 |\beta_3|)
\]
d\(_{BM} = 1
\]
else
\[
d_{BM} = 2
\]

Intuitively, Algorithm BM chooses a 1 \times 1 pivot if \( \alpha_1 \) is sufficiently large relative to the determinant of the
2 \times 2 pivot, i.e., a 1 \times 1 pivot is chosen if a 2 \times 2 pivot is relatively closer to being singular than \( \alpha_1 \) is to zero.

**IV. UNSYMMETRIC CASE GENERALIZATION**

In this section, we generalize the diagonal pivoting methods
for symmetric tridiagonal matrices described in Sec. III
to unsymmetric systems. We now describe these methods,
characterize relationships between the methods, and show
the three methods are normwise backward stable.

**A. Unsymmetric Bunch pivoting**

The Bunch pivoting strategy can be extended to unsymmetric
triangular matrices by choosing a 1 \times 1 pivot if
\[
|\alpha_1 |\sigma| = \kappa |\beta_2 \gamma_2|,
\]
where \( \sigma \) is as before in (7), and a 2 \times 2 pivot otherwise.
(We prove the backward stability of this pivoting strategy in Sec. IV-F.)

**Algorithm UB.** This algorithm describes the first step of
the unsymmetric Bunch pivoting strategy.
\[
\kappa = (\sqrt{5} - 1)/2 \approx 0.62
\]
\[
\sigma = \max_{i,j} \{|T_{i,j}|\}
\]
if \( \sigma |\alpha_1| \geq \kappa |\beta_2 \gamma_2|\)
\[
d_{UB} = 1
\]
else
\[
d_{UB} = 2
\]

**B. Unsymmetric Bunch-Kaufman pivoting**

In [8], we generalized the pivoting strategy of Bunch
and Kaufman for symmetric tridiagonal matrices (Algorithm
BK). This pivoting strategy chooses a 1 \times 1 pivot if the
(1, 1) diagonal entry is sufficiently large relative to the off-diagonals, i.e.,
\[
|\alpha_1 |\sigma| \geq \kappa |\beta_2 \gamma_2|,
\]
where
\[
\sigma_1 = \max\{|\alpha_2|, |\gamma_2|, |\beta_2|, |\gamma_1|, |\beta_3|\},
\]
and \( \kappa \) is as in Sec. 2.1. This algorithm can be viewed as
a generalization of the Bunch-Kaufman algorithm to the
unsymmetric case.

**Algorithm UBK.** This alternative algorithm determines
the size of the pivot for the first stage of the \( \text{LBM}^T \)
factorization applied to a tridiagonal matrix \( T \in \mathbb{R}^{n \times n} \).
\[
\kappa = (\sqrt{5} - 1)/2 \approx 0.62
\]
\[
\sigma_1 = \max\{|\alpha_2|, |\gamma_2|, |\beta_2|, |\gamma_1|, |\beta_3|\}
\]
if \( |\alpha_1 |\sigma_1 \geq \kappa |\beta_2 \gamma_2|\),
\[
d_{UB} = 1
\]
else
\[
d_{UB} = 2
\]

**C. Unsymmetric Bunch-Marcia pivoting**

The Bunch-Marcia pivoting strategy can be generalized to
unsymmetric tridiagonal matrices by the following. Let
\[
L_1 = T_{21} B_1^{-1} \quad \text{and} \quad M_1^T = B_1^{-1} T_{12}^T
\]
in (4). If a 1 \times 1 pivot is used, then the (1, 1) elements
of \( L_1 \) and \( M_1 \) are
\[
(L_1)_{1,1} = \frac{\beta_2}{\alpha_1} \quad \text{and} \quad (M_1)_{1,1} = \frac{\gamma_2}{\alpha_1}, \quad (13)
\]
If a 2 \times 2 pivot is used, then the (1, 1) and (1, 2) elements
of \( L_1 \) and \( M_1 \) are
\[
(L_1)_{1,1} = -\frac{\beta_2 \beta_3}{\Delta}, \quad (M_1)_{1,1} = -\frac{\gamma_2 \gamma_3}{\Delta}, \quad (14)
\]
\[
(L_1)_{1,2} = \frac{\alpha_1 \beta_3}{\Delta}, \quad (M_1)_{1,2} = \frac{\alpha_1 \gamma_3}{\Delta}
\]
With the elements of \( L_1 \) and \( M_1 \) for a 2 \times 2 pivot scaled
by the constant \( \kappa \) as before in (8), the unsymmetric
Bunch-Marcia pivoting strategy (Algorithm UBM) chooses a 1 \times 1
pivot if the both of the entries in (13) is smaller than the
largest entry in (14) in magnitude, i.e.,
\[
\max \left\{ \frac{|\beta_2|}{|\alpha_1|}, \frac{|\gamma_2|}{|\alpha_1|} \right\}
\]
\[
\leq \max \left\{ \frac{|\beta_2 \beta_3|}{|\Delta|}, \frac{|\alpha_1 \beta_3|}{|\Delta|}, \frac{|\gamma_2 \gamma_3|}{|\Delta|}, \frac{|\alpha_1 \gamma_3|}{|\Delta|} \right\},
\]
(15)
and a 2 \times 2 pivot is chosen otherwise. In other words,
Algorithm UBM chooses pivot sizes based on whichever
leads to smaller entries (in magnitude) in the computed
factors \( L \) and \( M \). In addition, we impose the criterion that
if
\[
|\alpha_1 |\sigma_1 \geq \kappa |\beta_2 \gamma_2|,
\]
(16)
a 1 \times 1 pivot is chosen. This additional criterion guarantees
that if \( T \) is positive definite, the \( \text{LBM}^T \) factorization reduces to the
\( \text{LDL}^T \) factorization. The choice of pivot size in the
first iteration is described as follows:

**Algorithm UBM.** This algorithm determines the size of the pivot for the first stage of the \( \text{LBM}^T \)
factorization applied to a tridiagonal matrix \( T \in \mathbb{R}^{n \times n} \).

(Advance online publication: 23 November 2010)
D. Relationships of the pivoting strategies

In this section, we summarize the relationship between the three pivoting strategies for unsymmetric matrices. In [8], it was shown that if the unsymmetric Bunch-Marcia pivoting strategy chooses a 1 × 1 pivot, i.e., $d_{UBM} = 1$, then the unsymmetric Bunch-Kaufman pivoting strategy also chooses a 1 × 1 pivot, i.e., $d_{UBK} = 1$. Conversely, if $d_{UBK} = 2$, then $d_{UBM} = 2$ as well. Finally, if $d_{UBM} = 2$ and $d_{UBK} = 1$, then the subsequent unsymmetric Bunch-Kaufman pivot size will also be 1 × 1. This implies that the difference between Algorithm UBK and Algorithm UBM is that sometimes all the possible pivot size choices for the three algorithms

\[ \Delta = \alpha_1 \alpha_2 - \beta_2 \gamma_2 \]

if $|\alpha_1 \alpha_2| \geq \kappa \beta_2 \gamma_2$ or $|\Delta| \max \{|\beta_2 \gamma_3|, |\alpha_1 \gamma_3|\} \leq \kappa |\alpha_1| \max \{|\beta_2 \beta_3|, |\alpha_1 \beta_3|\}

\[ d_{UBM} = 1 \]

else
\[ d_{UBM} = 2 \]
end

TABLE I: At any given iteration step, this table summarizes the possible pivot size choices for the three algorithms discussed in this section (Algorithms UB, UBK, and UBM).

<table>
<thead>
<tr>
<th>UB</th>
<th>UBK</th>
<th>UBM</th>
</tr>
</thead>
<tbody>
<tr>
<td>One $1 \times 1$</td>
<td>One $1 \times 1$</td>
<td>One $1 \times 1$</td>
</tr>
<tr>
<td>Two $1 \times 1$</td>
<td>Two $1 \times 1$</td>
<td>One $2 \times 2$</td>
</tr>
<tr>
<td>Two $1 \times 1$</td>
<td>One $2 \times 2$</td>
<td>One $2 \times 2$</td>
</tr>
</tbody>
</table>

Finally, if $T$ is positive definite, then
\[ |\alpha_1 \alpha_2| \geq |\beta_2 \gamma_2| \geq \kappa |\beta_2 \gamma_2|, \]
which implies that Algorithm UBM will always choose a 1 × 1 pivot. Similarly, Algorithms UB and UBK will always choose a 1 × 1 pivot because
\[ |\alpha_1 \sigma| \geq |\alpha_1 \sigma| \geq |\alpha_1 \alpha_2|. \]

This leads to the following result:

**Proposition 1.** If $T$ is a positive definite tridiagonal matrix, then the LDM factorization using Algorithms UB, UBK, or UBM reduces to the LDM factorization, where $L$ and $M$ are unit lower triangular and $D$ is diagonal.

E. Connections to the symmetric case

When applied to a symmetric tridiagonal matrix, the three algorithms for unsymmetric tridiagonal matrices described in Sec. IV reduce to the symmetric methods described in Sec. III.

The following two lemmas show that Algorithm UB and UBK are unsymmetric generalizations of Algorithm B and BK, respectively. In other words, on symmetric matrices Algorithms UB and UBK reduce to Algorithms B and BK, respectively.

**Lemma 2.** Let $T$ be a nonsingular symmetric matrix as in (6). Algorithm B chooses a 1 × 1 pivot if and only if Algorithm UB chooses a 1 × 1 pivot. Moreover, Algorithm BK chooses a 2 × 2 pivot if and only if Algorithm UBK chooses a 2 × 2 pivot.

**Proof.** When $\beta_2 = \gamma_2$, Algorithms B and UB are identical. □

**Lemma 3.** Let $T$ be a nonsingular symmetric matrix as in (6). Algorithm BK chooses a 1 × 1 pivot if and only if the Algorithm UBK chooses a 1 × 1 pivot. Moreover, Algorithm BK chooses a 2 × 2 pivot if and only if Algorithm UBK chooses a 2 × 2 pivot.

**Proof.** Substituting $\beta_2 = \gamma_2$ and $\beta_3 = \gamma_3$ into Algorithm UBK gives the result. □

The following lemma shows that Algorithm UBM is a generalization of Algorithm BM in the sense that on symmetric matrices, both algorithms choose the same pivots.

**Lemma 4.** Let $T$ be a nonsymmetric symmetric matrix as in (6). Algorithm BM chooses a 1 × 1 pivot if and only if Algorithm UBM chooses a 1 × 1 pivot. Algorithm BM chooses a 2 × 2 pivot if and only if Algorithm UBM chooses a 2 × 2 pivot.

**Proof.** If Algorithm BM chooses a 1 × 1 pivot then either $|\alpha_1 \alpha_2| \geq \kappa |\beta_2 \beta_3| or |\Delta| |\beta_2| \leq \kappa |\alpha_1| \max \{|\beta_2 \beta_3|, |\alpha_1 \beta_3|\}$. The latter condition is equivalent to either $|\Delta| |\beta_2| \leq \kappa |\alpha_1| |\beta_2 \beta_3| or |\Delta| |\beta_2| \leq \kappa |\alpha_1 \beta_3|$. Thus, if Algorithm BM chooses a 1 × 1 pivot then
\[ |\alpha_1 \alpha_2| \geq |\beta_2 \gamma_2| or |\Delta| \leq \kappa |\alpha_1 \beta_3| or |\beta_2 \Delta| \leq \kappa |\alpha_1 \beta_3|,

(Advance online publication: 23 November 2010)
satisfying the conditions for which Algorithm UBM will also choose a $1 \times 1$ pivot. Reversing this argument shows that if Algorithm UBM chooses a $1 \times 1$ pivot then Algorithm BM will do the same. The second statement of the lemma follows since it is the contrapositive of the first. □

The following proposition summarizes the lemmas in this section:

**Proposition 2.** If $T$ is a symmetric tridiagonal matrix, then Algorithms UB, UBK, and UBM reduce to Algorithms B, BK, and BM, respectively. If $T$ is also positive definite, then the LBM factorization reduces to the LDL factorization.

F. Backward stability result

In [8], we showed that Algorithms UBK and UBM imply a backward stability result that demonstrates that (a) the difference between computed factorization $\tilde{L}\tilde{B}\tilde{M}^T$ and the original tridiagonal matrix $T$ is small (i.e., it is of order machine precision), and (b) the computed solution $\hat{x}$ is the exact solution to a nearby problem. Using the relationship between the pivots of Algorithms UB and UBK and using arguments similar to that in [8], we can easily show the same results for Algorithm UBM. More formally, we state the following theorem:

**Theorem 1.** Assume the LBM factorization of the unsymmetric tridiagonal matrix $T \in \mathbb{R}^{n \times n}$ obtained using the pivoting strategy of Algorithm UB, UBK, or UBM yields the computed factorization $T \approx \tilde{L}\tilde{B}\tilde{M}^T$, and let $\hat{x}$ be the computed solution to $Tx = b$ obtained using the factorization. Assume that all linear systems $Ey = f$ involving $2 \times 2$ pivots $E$ are solved using the explicit inverse. Then

$$T - \tilde{L}\tilde{B}\tilde{M}^T = \Delta T_1,$$

and

$$(T + \Delta T_2)\hat{x} = b,$$

where

$$\|\Delta T_i\|_{\max} \leq cu\|T\|_{\max} + O(u^2), \quad i = 1, 2,$$

where $c$ is a constant and $u$ is the machine precision.

**Proof.** See [8].

Pictorially, the backward stability result of Theorem 1 is represented in Fig. 1.

V. NUMERICAL EXPERIMENTS

Numerical tests were run using MATLAB implementations of Algorithm UB, UBK, UBM, GEPP, and the MATLAB backslash command (“\”). (Specifically, Algorithms UB, UBK, and UBM were embedded in a code that used one forward substitution to solve with $L$, one back substitution to solve with $M^T$, and a solve with the block-diagonal factor $B$). We compared the performance of each code on 16 types of nonsingular tridiagonal linear systems of the form $Tx = b$. The test set of system matrices contains a wide range of difficulty. Many ill-conditioned matrices were chosen as part of the test set in order to compare algorithm robustness. (Ill-conditioned matrices are often challenging for matrix-factorization algorithms.) The test set of system matrices was taken from recent literature (specifically, [14] and [15]) on estimating condition numbers of ill-conditioned matrices—a task that can become increasingly difficult the more ill-conditioned the matrix is.

Table II contains a description of each tridiagonal matrix type in the test set. The first ten matrix types listed are based on test cases in [15]. Types 11-14 correspond to test cases found in [14] not found in [15], i.e., we eliminated redundant types. Finally, we include two additional tridiagonal matrices (Types 15-16) that can be generated using the MATLAB command gallery. For our test, $T$ was chosen to be a $100 \times 100$ matrix. The elements of $b$ were chosen from a uniform distribution on $[-1, 1]$.

One system matrix was generated from each matrix type, and together with a vector $b$, the same linear system of the form $Tx = b$ was solved by each algorithm. Table III shows the relative errors associated with each method. Columns 2-6 of the table contain the relative error

$$E_{rel} = \frac{\|T\hat{x} - b\|_2}{\|b\|_2},$$

where $\hat{x}$ is the computed solution by each solver. The final column gives the condition number of the system matrix $T$. Each row in the table corresponds to one linear system from one type; that is, row $i$ contains the relative errors associated with each solver on a linear system whose system matrix was from type $i$ in Table II. Although we only present one set of results in this paper, these results are representative of the many runs we performed.

Table III suggests that all five algorithms are comparable on a wide variety of linear systems. When the system matrix is well-conditioned (e.g., Types 1, 4, 5, 6, 8, 10, 14, 16), the algorithms have relative errors that are close to machine precision and that are comparable to one another. On the other hand, more significant differences occur when the system matrix is very ill-conditioned (Types 2, 3, 7, 11, 12 and 15). For example, on Type 13, Algorithms UB, UBK, and UBM perform particularly poorly, and the solutions are offered by GEPP and the MATLAB backslash command are also very poor, with an error near $10^{-1}$. In fact, due to the ill-conditioning of the system matrix, all methods failed to solve the linear system; however, all methods obtain relative errors within an order of magnitude of each other. Meanwhile,
### TABLE II: Tridiagonal matrix types used in the numerical experiments.

<table>
<thead>
<tr>
<th>Matrix type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Randomly generated matrix from a uniform distribution on $[-1,1]$.</td>
</tr>
<tr>
<td>2</td>
<td><code>gallery('randsvd',n,le15,2,1,1)</code> – Randomly generated matrix with condition number $\leq 15$ and one small singular value.</td>
</tr>
<tr>
<td>3</td>
<td><code>gallery('randsvd',n,le15,3,1,1)</code> – Randomly generated matrix with condition number $\leq 15$ and geometrically distributed singular values.</td>
</tr>
<tr>
<td>4</td>
<td>Toeplitz matrix $T$ with $T_{ii} = 10^8$ for $i = 1 \ldots n$, and the elements $T_{ij}$ for $i \neq j$ are randomly generated from a uniform distribution on $[-1,1]$.</td>
</tr>
<tr>
<td>5</td>
<td>Toeplitz matrix $T$ with $T_{ii} = 10^{-8}$ for $i = 1 \ldots n$, and the elements $T_{ij}$ for $i \neq j$ are randomly generated from a uniform distribution on $[-1,1]$.</td>
</tr>
<tr>
<td>6</td>
<td><code>gallery('lesp',n)</code> – Matrix with sensitive eigenvalues that are smoothly distributed in the approximate interval $[-2n - 3.5, -4.5]$.</td>
</tr>
<tr>
<td>7</td>
<td><code>gallery('dorr',n,le-4)</code> – Ill-conditioned, diagonally dominant matrix.</td>
</tr>
<tr>
<td>8</td>
<td>Randomly generated matrix from a uniform distribution on $[-1,1]$; the $50^\text{th}$ subdiagonal element is then multiplied by $10^{-50}$.</td>
</tr>
<tr>
<td>9</td>
<td>Matrix whose elements are all generated from a uniform normal distribution on $[-1,1]$; the lower diagonal are then multiplied by $10^{-50}$.</td>
</tr>
<tr>
<td>10</td>
<td>Main diagonal elements generated randomly from a uniform distribution on $[-1,1]$; off-diagonal elements each chosen with 50% probability as either zero or generated randomly from a uniform distribution on $[-1,1]$.</td>
</tr>
<tr>
<td>11</td>
<td><code>gallery('randsvd',n,le15,1,1,1)</code> – Randomly generated matrix with condition number $\leq 15$ and one large singular value.</td>
</tr>
<tr>
<td>12</td>
<td><code>gallery('randsvd',n,le15,4,1,1)</code> – Randomly generated matrix with condition number $\leq 15$ and arithmetically distributed singular values.</td>
</tr>
<tr>
<td>13</td>
<td>Toeplitz matrix $T$ with $T_{ii} = 64$ for $i = 1 \ldots n$, and the elements $T_{ij}$ for $i \neq j$ are randomly generated from a uniform distribution on $[-1,1]$.</td>
</tr>
<tr>
<td>14</td>
<td>Toeplitz matrix $T$ with $T_{ii} = 0$ for $i = 1 \ldots n$, and the elements $T_{ij}$ for $i \neq j$ are randomly generated from a uniform distribution on $[-1,1]$.</td>
</tr>
<tr>
<td>15</td>
<td><code>gallery('clement',n,0)</code> – Main diagonal elements are zero; eigenvalues include plus and minus the numbers $n - 1, n - 3, n - 5, \ldots 1$.</td>
</tr>
<tr>
<td>16</td>
<td><code>inv(gallery('kms',n,0.5))</code> – Inverse of a Kac-Murdoch-Szego Toeplitz $(A_{ij} = (0.5)^{</td>
</tr>
</tbody>
</table>

Types 9 and 13 have larger condition numbers and relative errors that slightly larger than machine precision; on these problems all algorithms have nearly identical relative errors (i.e., same order of magnitude).

### VI. CONCLUSIONS AND FUTURE WORK

Algorithms UB, UBK, and UBM presented in this paper were shown to compute a backward-stable LBM$^T$ decomposition of any nonsingular tridiagonal matrix without using row or column interchanges. In this paper, we showed that these algorithms can be viewed as generalizations of algorithms for symmetric tridiagonal matrices by comparing their choice of pivot sizes on symmetric tridiagonal matrices. Moreover, we established relationships between pivot choices made by each unsymmetric algorithms on general unsymmetric tridiagonal matrices. Numerical results presented in this paper on a wide range of linear systems suggest that the performance of these algorithms are comparable to GEPP and the MATLAB backslash command.

In addition to computing a backward-stable LBM$^T$ decomposition of $T$, these algorithms have minimal storage requirements. Specifically, $T$ is tridiagonal, and thus, can stored using three vectors. Moreover, updating the Schur complement requires updating only one nonzero component of $T$. The matrices $L$ and $M$ are unit-lower triangular with $L_{ij} = M_{ij} = 0$ for $i - j > 2$; thus, their entries can be stored in two vectors each. Finally, $B$ is block-diagonal with $1 \times 1$ or $2 \times 2$ blocks, requiring only three vectors for storage.

Of the three strategies, Algorithms UBK and UBM do not depend on knowing the largest entry of $T$ a priori, making them suitable to factorize $T$ as it is being formed. Thus, future work will focus on embedding Algorithms UBK and UBM into methods such as the look-ahead Lanczos methods [10] and composite-step bi-conjugate gradient methods [11] for solving unsymmetric linear systems.

### REFERENCES

<table>
<thead>
<tr>
<th>Matrix Type</th>
<th>Algorithm UB</th>
<th>Algorithm UK</th>
<th>Algorithm UBM</th>
<th>GEPP</th>
<th>MATLAB Backslash</th>
<th>Condition Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3.4866e-14</td>
<td>3.3034e-14</td>
<td>4.6098e-14</td>
<td>2.5466e-14</td>
<td>3.2133e-14</td>
<td>2.5701e+03</td>
</tr>
<tr>
<td>4</td>
<td>7.5491e-16</td>
<td>7.5491e-16</td>
<td>7.5491e-16</td>
<td>5.5892e-16</td>
<td>5.5892e-16</td>
<td>1.0000e+00</td>
</tr>
<tr>
<td>6</td>
<td>7.1602e-16</td>
<td>7.1602e-16</td>
<td>7.1602e-16</td>
<td>5.8452e-16</td>
<td>5.8452e-16</td>
<td>6.7485e+01</td>
</tr>
<tr>
<td>7</td>
<td>7.5256e+01</td>
<td>7.5256e+01</td>
<td>1.5790e+02</td>
<td>1.4851e+02</td>
<td>2.7842e+16</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>3.0088e-13</td>
<td>3.0088e-13</td>
<td>2.4821e-13</td>
<td>2.4463e-13</td>
<td>6.9653e+03</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>9.5730e-05</td>
<td>9.5730e-05</td>
<td>8.1332e-05</td>
<td>5.3370e-05</td>
<td>5.3370e-05</td>
<td>1.2529e+15</td>
</tr>
<tr>
<td>12</td>
<td>1.1328e-02</td>
<td>1.1149e-02</td>
<td>1.0650e-02</td>
<td>6.9575e-03</td>
<td>6.9521e-03</td>
<td>9.6309e+14</td>
</tr>
<tr>
<td>13</td>
<td>1.4695e-10</td>
<td>1.4695e-10</td>
<td>1.4576e-10</td>
<td>1.4576e-10</td>
<td>2.2052e+07</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>1.5023e-01</td>
<td>1.5023e-01</td>
<td>1.3084e-01</td>
<td>1.3084e-01</td>
<td>3.1446e+15</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>6.9333e-16</td>
<td>6.9333e-16</td>
<td>1.1264e-15</td>
<td>1.2741e-15</td>
<td>8.9813e+00</td>
<td></td>
</tr>
</tbody>
</table>