Relaxed forms of BBK algorithm and FBP algorithm for symmetric indefinite linear systems

Ting-Zhu Huang*, Liang Li

School of Applied Mathematics, University of Electronic Science and Technology of China, Chengdu, Sichuan, 610054, PR China

Received 15 March 2007; accepted 10 April 2007

Abstract

In this paper, we study the direct solvers for the linear system \( Ax = b \), where \( A \) is symmetric and indefinite. We discuss the so-called BBK algorithm and FBP algorithm and propose relaxed forms of them which provide options for fast pivot selection. We also present some numerical tests to show the efficiency of our algorithms.

Keywords: Symmetric indefinite matrices; Symmetric pivoting; Direct solvers; BBK; FBP

1. Introduction

When \( A \) is a symmetric indefinite matrix, there are three well-known algorithms to solve \( Ax = b \) directly [1]: Aasen’s algorithm [2], the Bunch–Kaufman algorithm [3] and the Bunch–Parlett algorithm [4]. Aasen’s algorithm decomposes \( A \) into \( PAP^T = LTL^T \), where \( P \) is a permutation matrix, \( L \) is a unit lower triangular matrix, and \( T \) is a symmetric tridiagonal matrix. The other two algorithms decompose \( A \) into \( PAP^T = LDL^T \), where \( D \) is a block diagonal with diagonal blocks of dimension 1 or 2.

Since \( LDL^T \) factorization without pivoting suffers great breakdown, all these algorithms take pivoting strategies into consideration. Bunch–Parlett algorithm searches the whole matrix to determine the pivot at each stage, and yields a backward stable factorization and bounded \( \|L\|_\infty \). However, people hesitate to use it because “too much” comparisons are involved (totally \( o(n^3) \) comparisons). Bunch–Kaufman algorithm searches only two columns of the matrix at each stage, and needs only \( o(n^2) \) comparisons. But this algorithm cannot bound \( \|L\|_\infty \), which may cause nonstability, according to the analysis in [5,6].

Recently, Ashcraft, Grimes and Lewis [5] developed the bounded Bunch–Kaufman algorithm (BBK) and fast Bunch–Parlett algorithm (FBP), which are very promising. Instead of searching the whole matrix or only the first two columns, they took a clever compromise, and searched for a “local maximum” off-diagonal entry \( a_{ij} \) that has

* Supported by the NCET of the Chinese Ministry of Education (NCET-04-0893), Scientific and Technological Key Project of Chinese Ministry of Education (107098), Sichuan province Foundation for Applied Basic Research and Leader of Sci. and Tech. (05JY029-068-2), and project of academic leader and group of UESTC.

* Corresponding author.

E-mail addresses: tzhuang@uestc.edu.cn, tingzhuhuang@126.com (T.-Z. Huang).
the magnitude of the largest one in the \( r \)th row and in the \( i \)th column. They found that finding a local maximum off-diagonal entry of the BBK algorithm usually requires far less than \( o(n^2) \) comparisons in practice. And their numerical tests on random matrices show that on an average, fewer than \( 2.5n \) comparisons suffice to find a suitable pair of columns. The fast Bunch–Parlett algorithm is a little slower than the BBK algorithm, because this algorithm needs to find the maximum diagonal entry first and this yields more column interchanges.

However, for some kinds of matrices like the ones studied in [5], the BBK algorithm and FBP algorithm can only find the pivot after searching the whole matrix. This situation is not expected, thus in this paper, we try to make this better by relaxing the pivot selection criteria.

This paper is organized as follows. In Section 2, we propose the RBBK algorithm and study its numerical stability. In Section 3, the RFBP is proposed and studied. And finally, in Section 4 we test our algorithms by some numerical examples.

2. A relaxed bounded Bunch–Kaufman algorithm

In order to bound \( \|L\| \) and to speed up the pivot searching process, we search for an “almost local maximum” off-diagonal entry instead of the “local maximum” one. Given parameters \( \alpha \) and \( \beta \) where \( 0 < \alpha \leq 1 \) and \( 0 < \beta \leq 1 \), the following algorithm describes the pivot selection. We denote by RBBK, the relaxed bounded Bunch–Kaufman algorithm.

\[ \gamma_1 = \text{maximum magnitude of any off-diagonal entry in column 1;} \]
\[ r = \text{row index of first entry of maximum magnitude in column 1;} \]
\[ \text{if } \gamma_1 = 0 \]
\[ \text{nothing necessary for column 1;} \]
\[ \text{elseif } |a_{11}| \geq \alpha \gamma_1 \]
\[ \text{use } a_{11} \text{ as the pivot; } s = 1; P = I; \]
\[ \text{else} \]
\[ \text{flag = 0; } i = 1; \gamma_i = \gamma_1; \]
\[ \text{while } (\text{flag} = 0) \]
\[ \gamma_r = \text{maximum magnitude of any off-diagonal entries in column } i; \]
\[ r = \text{row index of first entry of maximum magnitude in column } i; \]
\[ \text{if } |a_{rr}| \geq \alpha \gamma_r \]
\[ \text{use } a_{rr} \text{ as the pivot; } s = 1; \text{flag} = 1; \]
\[ P \text{ swaps rows 1 and } r; \]
\[ \text{elseif } \gamma_r \leq \beta \gamma_i \]
\[ \text{use } \begin{bmatrix} a_{ii} & a_{ri} \\ a_{ri} & a_{rr} \end{bmatrix} \text{ as the pivot; } s = 2; \text{flag} = 1; \]
\[ P \text{ swaps rows 1 and } i, 2 \text{ and } r; \]
\[ \text{else} \]
\[ i = r; \gamma_i = \gamma_r; \]
\[ \text{end} \]
\[ \text{end} \]

Remark. No irreversible pivot (0 for 1 \( \times \) 1 pivot or \( \begin{bmatrix} a_{ii} & a_{ri} \\ a_{ri} & a_{rr} \end{bmatrix} \) is irreversible for 2 \( \times \) 2 pivot) will be generated if \( A \) is reversible. Otherwise \( D \) in the \( LDL^T \) factorization has an irreversible block, which implies that \( A \) is irreversible.

In a typical step of \( LDL^T \) factorization, after eliminating \( k - 1 \) columns, we proceed with the reduced matrix \( A^{(k)} \) to \( A^{(k+1)} \) with a 1 \( \times \) 1 pivot and \( A^{(k+2)} \) with a 2 \( \times \) 2 pivot.

Theorem 2.1. Let \( \mu \) be the maximum magnitude of any entry in \( A^{(k)} \) and \( \mu' \) be the maximum magnitude of any entry in the new reduced matrix. If a 1 \( \times \) 1 pivot \( a_{rr} \) is generated, we have the following inequality
\[ \mu' \leq \mu \left( 1 + \frac{1}{\alpha} \right). \]  
\hspace{1cm} (1)

If a 2 \times 2 pivot
\[ \begin{bmatrix} a_{ii} & a_{ri} \\ a_{ri} & a_{rr} \end{bmatrix} \]  
\hspace{1cm} (2)
is generated, and if \( \beta > \alpha^2 \), the following inequality holds
\[ \mu' < \mu \left( 1 + \frac{\alpha + 2 \beta + \alpha \beta}{\beta - \alpha^2} \right). \]  
\hspace{1cm} (3)

Proof. If \( a_{rr} \) is the pivot, it holds that
\[ a_{ij}^{(k+1)} = a_{ij} - \frac{a_{ir}a_{rj}}{a_{rr}}, \quad i \neq r; \ j \neq r. \]

Thus,
\[ |a_{ij}^{(k+1)}| \leq |a_{ij}| + \left| \frac{a_{ir}a_{rj}}{a_{rr}} \right| \leq |a_{ij}| + \gamma_r \left| \frac{a_{ir}}{a_{rr}} \right|, \]
then, we have
\[ \mu' \leq \mu + \gamma_r \frac{\gamma_r}{|a_{rr}|} \leq \mu \left( 1 + \frac{1}{\alpha} \right). \]

If a 2 \times 2 pivot (2) is generated, for any element in \( A^{(k+2)} \)
\[ a_{ij}^{(k+2)} = a_{ij} - \left[ a_{ij} \ a_{ir} \right] \left[ a_{ii} \ a_{ir} \right]^{-1} \left[ a_{ij} \ a_{rj} \right] = a_{ij} - \frac{a_{ii} a_{ri} a_{rj} - a_{ri} (a_{ii} a_{rj} + a_{ir} a_{ij}) + a_{rr} a_{ij}}{a_{ii} a_{rr} - a_{ir}^2}, \quad i' \neq i, r; \ j \neq i, r, \]
and the following equalities and inequalities hold:
\[ |a_{ir}| = \gamma_i, \quad |a_{ii}| < \alpha \gamma_i, \quad |a_{rr}| < \alpha \gamma_r \quad \text{and} \quad \gamma_r \leq \frac{\gamma_i}{\beta}. \]

Then,
\[ |a_{ij}^{(k+2)}| \leq |a_{i'j}| + \left| \frac{a_{ii} a_{rj} a_{ij} - a_{ri} (a_{ii} a_{rj} + a_{ir} a_{ij}) + a_{rr} a_{ij}}{a_{ii} a_{rr} - a_{ir}^2} \right| \]
\[ < |a_{i'j}| + \left| \frac{a_{ii} |\gamma_r^2 + 2 \gamma_r \gamma_i^2 + |a_{rr}| \gamma_i^2}{\gamma_i^2(1 - \alpha^2/\beta)} \right| \]
and the result (3) follows. The proof is complete. \( \Box \)

We shall explore how the elements in \( L \) are bounded in the theorem below.

Theorem 2.2. If a 1 \times 1 pivot \( a_{rr} \) is generated, we have the following inequality
\[ |l_{ir}| \leq \frac{1}{\alpha}, \quad i \neq r. \]  
\hspace{1cm} (4)

If a 2 \times 2 pivot (2) is generated, and if \( \beta > \alpha^2 \), then
\[
\begin{align*}
|l_{i'j}| & \leq \frac{\alpha + 1}{\beta - \alpha^2}, \quad i' \neq i, r; \\
|l_{ir}| & \leq \frac{2\alpha}{\beta - \alpha^2}, \quad i' \neq i, r.
\end{align*}
\]  
\hspace{1cm} (5)
**Proof.** If $a_{ir}$ is the pivot, it is easily found that

$$l^{(k+1)}_{ir} = \frac{a_{ir}}{a_{rr}}, \quad i \neq r.$$  

Having the pivot selection criteria in mind, we immediately get (4).

If a $2 \times 2$ pivot (2) is generated, the entries of $L$ in columns $i$ and $r$ can be explicitly calculated as

$$\begin{bmatrix} l_{i'i} & l_{i'r} \end{bmatrix} = \begin{bmatrix} a_{ii} & a_{ir} \\ a_{ri} & a_{rr} \end{bmatrix}^{-1},$$

and followed by

$$l_{i'i} = \frac{a_{i'i}a_{rr} - a_{ri}a_{r'i}}{a_{ii}a_{rr} - a_{r'i}^2}, \quad i' \neq i, r,$$

$$l_{i'r} = \frac{a_{i'r}a_{ii} - a_{ri}a_{i'i}}{a_{ii}a_{rr} - a_{r'i}^2}, \quad i' \neq i, r,$$

and easily we obtain the bounds for them:

$$|l_{i'i}| \leq \frac{|a_{i'i}a_{rr}| + |a_{ri}a_{r'i}|}{|a_{ii}a_{rr} - a_{r'i}^2|} < \frac{\alpha \gamma_i \gamma_r + \gamma_i \gamma_r}{\gamma_i^2 - \alpha^2 \gamma_i \gamma_r} \leq \frac{\alpha \gamma_i^2 / \beta + \gamma_i^2 / \beta}{\gamma_i^2 (1 - \alpha^2 / \beta)} = \frac{\alpha + 1}{\beta - \alpha^2},$$

and

$$|l_{i'r}| \leq \frac{|a_{i'r}a_{ii}| + |a_{ri}a_{i'i}|}{|a_{ii}a_{rr} - a_{r'i}^2|} < \frac{\alpha \gamma_i \gamma_r + \alpha \gamma_i \gamma_r}{\gamma_i^2 - \alpha^2 \gamma_i \gamma_r} \leq \frac{\alpha \gamma_i^2 / \beta + \alpha \gamma_i^2 / \beta}{\gamma_i^2 (1 - \alpha^2 / \beta)} = \frac{2\alpha}{\beta - \alpha^2}.$$  

The proof is complete. \qed

From the two theorems above, we know that if the pivot is $1 \times 1$, then the lower triangular elements of $L$ are bounded by $\frac{1}{\alpha}$. When the pivot is $2 \times 2$, it is quite reasonable that we require that the elements of $L$ are also bounded by $\frac{1}{\alpha}$. Since $0 < \alpha \leq 1$, $\frac{\alpha + 1}{\beta - \alpha^2} \geq \frac{2\alpha}{\beta - \alpha^2}$. Thus, if we set

$$\frac{1}{\alpha} = \frac{\alpha + 1}{\beta - \alpha^2}, \quad (6)$$

then all the lower triangular elements of $L$ will be bounded by $\frac{1}{\alpha}$. Then from (6) we get that

$$\beta = 2\alpha^2 + \alpha. \quad (7)$$

It is obvious that a small value of $\alpha$ will result in fast pivot selection, but loose bound on $\|L\|_\infty$. We know that if $\beta \leq 1$, then $2\alpha^2 + \alpha \leq 1$, by simple computation, we get that $0 < \alpha \leq 0.5$. Especially, when $\alpha = 0.5$, $\beta = 1$, the RBBK algorithm reduces to the BBK algorithm, and all the lower triangular elements of $L$ are bounded by 2.
Table 1
Performance of the RBBK and BBK algorithms with different $\alpha$

<table>
<thead>
<tr>
<th>$\alpha$</th>
<th>0.1</th>
<th>0.2</th>
<th>0.3</th>
<th>0.4</th>
<th>0.5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Algorithm</td>
<td>RBBK</td>
<td>BBK</td>
<td>RBBK</td>
<td>BBK</td>
<td>RBBK</td>
</tr>
<tr>
<td>nc</td>
<td>1680</td>
<td>2014</td>
<td>1809</td>
<td>2219</td>
<td>1872</td>
</tr>
<tr>
<td>$1 \times 1$</td>
<td>531</td>
<td>574</td>
<td>368</td>
<td>414</td>
<td>238</td>
</tr>
<tr>
<td>$2 \times 2$</td>
<td>469</td>
<td>426</td>
<td>632</td>
<td>586</td>
<td>762</td>
</tr>
</tbody>
</table>

3. A relaxed Fast Bunch–Parlett algorithm

The pivoting strategy for FBP algorithm is also aimed to search for the “local maximum” off-diagonal entry $a_{ri}$ that has the magnitude largest in the $r$th row and in the $i$th column, but it needs to find out the largest magnitude of all the diagonal entries first. The relaxed form of this algorithm also searches for an “almost local maximum” off-diagonal entry just like the RBBK algorithm does. Here we denote the relaxed version of FBP algorithm as $RFBP$. Given parameters $\alpha$ and $\beta$, where $0 < \alpha \leq 1$ and $0 < \beta \leq 1$, $RFBP$ algorithm is described as follows.

Algorithm $RFBP$ (This Algorithm Determines the Pivot for the First Stage).

1. $a_{ss} = \text{the diagonal entry of largest magnitude in } A^{(k)}$;
2. $\gamma_s = \text{magnitude of largest off-diagonal entry in column } s$;
3. if $\gamma_s = 0$
   nothing necessary for the $s$th column;
4. elseif $|a_{ss}| \geq \alpha \gamma_s$
   use $a_{ss}$ as a $1 \times 1$ pivot;
5. else
   1. $i = s$; $\gamma_i = \gamma_s$; flag = 0;
   2. while (flag = 0)
      1. $r = \text{row index of first entry of maximum magnitude in column } i$;
      2. $\gamma_r = \text{maximum magnitude of any off-diagonal entry in column } r$;
      3. if $\gamma_i \geq \beta \gamma_r$
         use $\begin{bmatrix} a_{ii} & a_{ir} \\ a_{ri} & a_{rr} \end{bmatrix}$ as a $2 \times 2$ pivot; flag=1;
      4. else
         1. $i = r$; $\gamma_i = \gamma_r$;
         2. end
   3. end

A $1 \times 1$ pivot is found when the largest diagonal entry is a large enough fraction of the largest entry in its column. A $2 \times 2$ pivot is generated if $a_{ir}$ is the largest entry in column $i$ and is a large enough fraction of the largest off-diagonal entry in column $r$.

The bounds of the lower triangular elements of $L$ and the growth of the magnitude largest elements in the reduced matrices can be similarly analyzed as we do for the RBBK algorithm. We obtain that when $0 < \alpha \leq 0.5$ and $\beta$ is computed by (7), the elements of $L$ will be controlled by $\frac{1}{\alpha}$. $RFBP$ algorithm is slightly slower than RBBK because of extra work and more column interchanges.

4. Numerical examples

Usually with the same parameter $\alpha$ the relaxed forms of BBK algorithm and FBP algorithm will be faster than their original forms, if parameter $\beta$ is computed by (7). From the analysis in Section 2, we see that the triangular elements of $L$ are bounded by $\frac{1}{\alpha}$ for both the original and relaxed forms.
Table 2
Performance of the RFBP algorithm with different $\alpha$

<table>
<thead>
<tr>
<th>$\alpha$</th>
<th>0.1</th>
<th>0.2</th>
<th>0.3</th>
<th>0.4</th>
<th>0.5</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 x 1</td>
<td>1000</td>
<td>1000</td>
<td>1000</td>
<td>989</td>
<td>939</td>
</tr>
<tr>
<td>2 x 2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>11</td>
<td>61</td>
</tr>
</tbody>
</table>

Example 1. We first test the worst cases for the bounded Bunch–Kaufman and fast Bunch–Parlett algorithms given by Ashcraft, Grimes and Lewis in [5]. When $\alpha = 0.3$ the matrix family

$$\begin{pmatrix}
0 & 2 \\
4 & 0 & 3 \\
2 & 3 & 0
\end{pmatrix}, \begin{pmatrix}
0 & 6 & 6 & 2 \\
6 & 0 & 5 & 4 \\
6 & 0 & 3 & 0
\end{pmatrix}, \begin{pmatrix}
0 & 8 & 8 & 2 \\
8 & 0 & 7 & 6 \\
7 & 0 & 6 & 5 \\
5 & 0 & 4 & 3
\end{pmatrix}, \ldots$$

requires a search of the entire matrix at each step by the BBK algorithm. Set $\alpha = 0.3$ and $\beta = 0.39$, the RBBK algorithm searches only two columns at each step to determine the pivot.

Similarly, for the FBP algorithm, choose a matrix from the family

$$\begin{pmatrix}
1 & 2 \\
4 & 0 & 3 \\
2 & 3 & 0
\end{pmatrix}, \begin{pmatrix}
1 & 6 & 6 & 2 \\
6 & 0 & 5 & 4 \\
6 & 0 & 3 & 0
\end{pmatrix}, \begin{pmatrix}
1 & 8 & 0 & 2 \\
8 & 0 & 7 & 6 \\
7 & 0 & 6 & 5 \\
5 & 0 & 4 & 3
\end{pmatrix}, \ldots$$

The fast Bunch–Parlett algorithm will make a complete search at each step. With the same $\alpha$ and $\beta$, RFBP algorithm will terminate the searching process after searching two columns.

Example 2. Since we can’t deduce how much comparisons are needed for these algorithms in theory, we test 1000 100 x 100 symmetric random matrices generated by the SPRANDSYM\(^1\) function (with density of 0.7) in Matlab’s sparfun package. Setting $\alpha = 0.1, 0.2, 0.3, 0.4, 0.5$, and $\beta$ as the ones computed by (7), we record the total number of columns involved in comparison at the first stage of factorization for all the matrices as $nc$, number of 1 x 1 pivot and 2 x 2 pivot for each algorithm. The results are exhibited in the Tables 1 and 2.

It should be noted that when $\alpha = 0.5$, the RBBK algorithm is actually BBK algorithm and RFBP is FBP. We find that RBBK algorithm can bound the elements of $L$ to the same $\frac{1}{\alpha}$ as BBK does but requires much less comparisons, especially when $\alpha < 0.5$. However, since the RBFP algorithm will search the diagonal first, and if $\alpha$ is small, this algorithm usually results in a 1 x 1 pivot, and in this case RFBP and FBP will perform similarly.

References


---

\(^1\) R = SPRANDSYM (n, density) is a symmetric random, n-by-n, sparse matrix with approximately density x n x n nonzeros; each entry is the sum of one or more normally distributed random samples.