ISSN: 1017-060X (Print)

ISSN: 1735-8515 (Online)

Bulletin of the

Iranian Mathematical Society

Vol. 43 (2017), No. 5, pp. 1417–1456

Title:

Complete pivoting strategy for the IUL preconditioner obtained from backward factored APproximate INVerse process

Author(s):

A. Rafiei and M. Bollhöfer

Published by the Iranian Mathematical Society
http://bims.ims.ir

Bull. Iranian Math. Soc.
Vol. 43 (2017), No. 5, pp. 1417–1456
Online ISSN: 1735-8515

COMPLETE PIVOTING STRATEGY FOR THE IUL PRECONDITIONER OBTAINED FROM BACKWARD FACTORED APPROXIMATE INVERSE PROCESS

A. RAFIEI* AND M. BOLLHÖFER

(Communicated by Davod Khojasteh Salkuyeh)

Abstract. In this paper, we use a complete pivoting strategy to compute the IUL preconditioner obtained as the by-product of the Backward Factored APproximate INVerse process. This pivoting is based on the complete pivoting strategy of the Backward IJK version of Gaussian Elimination process. There is a parameter α to control the complete pivoting process. We have studied the effect of different values of α on the quality of the IUL preconditioner. For the numerical experiments section, the IUL factorization which is coupled with the complete pivoting is compared to the ILUTP and to the left-looking version of RIF which is coupled with the complete pivoting strategy. As the preprocessing, we have applied the maximum weighted matching coupled with the Reverse Cuthill-Mckee (RCM) and multilevel nested dissection reordering.
Keywords: Backward factored APproximate INVerse, IUL preconditioner, backward IJK version of Gaussian elimination, complete pivoting, ILUTP, left-looking RIF with pivoting.
MSC(2010): Primary: 65F10 ; Secondary: 65F50, 65F08.

1. Introduction

One can use the explicit and implicit preconditioner M for the linear system of equations of the form

$$Ax = b, \tag{1.1}$$

where the coefficient matrix $A \in \mathbb{R}^{n\times n}$ is nonsingular, large, sparse and nonsymmetric and also $x, b \in \mathbb{R}^n$. An explicit preconditioner M for system (1.1) is an approximation of the matrix A^{-1}. We can use this preconditioner to change the original system (1.1) to the right or left preconditioned systems and then, solve the preconditioned system by one of the Krylov subspace methods [17].

Article electronically published on 31 October, 2017.
Received: 2 January 2016, Accepted: 1 July 2016.
*Corresponding author.

In this case, we only need matrix-vector products which is really suitable for parallel architecture.

In 1993, Luo presented the **B**ackward **F**actored **INV**erse or BFINV algorithm which computes the inverse factorization of A in the form of

$$A^{-1} = \bar{Z}\bar{D}^{-1}\bar{W}, \tag{1.2}$$

where $\bar{W}$ and $\bar{Z}^T$ are unit upper triangular matrices and $\bar{D}$ is a diagonal matrix [11]. By applying a dropping rule for the entries of the $\bar{W}$ and $\bar{Z}$ matrices in the BFINV algorithm, the explicit preconditioner M is computed as

$$A^{-1} \approx M = ZD^{-1}W, \tag{1.3}$$

where $W \approx \bar{W}$, $D \approx \bar{D}$, $Z \approx \bar{Z}$ and the process is termed as the **B**ackward **F**actored **AP**proximate **INV**erse or BFAPINV. The implementation details to compute this explicit preconditioner can be found in [23].

In 1999, Zhang presented the **F**orward **F**actored **INV**erse or FFINV algorithm which computes the factorization (1.2). In this case, matrices $\bar{Z}$ and $\bar{W}$ are unit upper and unit lower triangular, respectively and $\bar{D}$ is again a diagonal matrix [22]. Using a dropping rule in this algorithm will compute the explicit preconditioner (1.3) and the process is termed as the **F**orward **F**actored **AP**proximate **INV**erse or FFAPINV [13].

In [20], the authors could find a relation between the FFINV algorithm and the left-looking version of the A-biconjugation process of Benzi and Tůma [1]. Based on this relation they showed that the explicit preconditioner (1.3) which is computed from the FFAPINV algorithm is exactly the left-looking version of the AINV preconditioner.

An implicit preconditioner for the system (1.1) is an approximation of matrix A. This preconditioner can also be used as the right or left preconditioner. When using the Krylov subspace methods to solve this preconditioned system, we face the forward and backward solving which are the bottle necks in the parallel implementation of implicit preconditioners in recent years. Solving such a problem is so crutial to apply an implicit preconditioner on parallel machines [9]. In [13], we could compute an implicit preconditioner M as the by-product of the BFAPINV process. This preconditioner is in the form of

$$A \approx M = UDL, \tag{1.4}$$

where U and L^T are unit upper triangular matrices and D is a diagonal matrix. This preconditioner is an incomplete UDL factorization. We have merged the factors D and L of this factorization and then, have termed it as the IULBF. This notation refers to the IUL factorization obtained from **B**ackward **F**actored approximate inverse process. In the factorizations (1.3) and (1.4), $L^{-1} \approx Z$ and $U^{-1} \approx W$.

Working with the FFAPINV process also gives us the chance to have an implicit preconditioner

$$A \approx M = LDU, \tag{1.5}$$

as the by-product. In [19], we have termed this preconditioner as the ILUFF which refers to the ILU preconditioner obtained from the **F**orward **F**actored approximate inverse process. In [2,15], the authors showed that one can compute an ILU preconditioner in the form of (1.5) as the by-product of the AINV preconditioner. This preconditioner was called RIF or **R**obust **I**ncomplete **F**actorization and has the left- and the right-looking versions. From the results presented in [20], one can easily verify that the ILUFF can be converted to the left-looking version of RIF and vice versa. In [16], we have implemented a type of complete pivoting strategy for the left-looking version of RIF which can also be considered as the complete pivoting strategy for the ILUFF preconditioner.

By applying the dropping strategy in the Forward form of the IJK version of Gaussian Elimination process one can compute an implicit ILU preconditioner for the system (1.1) [14,16]. In a sequential architecture, the preconditioning time of this ILU is less than the preconditioning time of the explicit preconditioners BFAPINV, FFAPINV and AINV. There is also a backward form of the IJK version of Gaussian elimination process. If we apply dropping in this backward form, then we compute an implicit IUL preconditioner M as in (1.4). Since the whole parts of the Schur-Complement matrices are explicitly available, then it is possible to apply the complete pivoting strategy in the backward form of this version of Gaussian elimination process.

As in [20], can we find a relation between the BFINV and the right-looking A-biconjugation process? Or more precisely, is the BFAPINV preconditioner another version of right-looking AINV preconditioner? The answer is no, since the factors of these two preconditioners are computed in a completely different way.There is a version of right-looking AINV in which the factors can be computed independently, but this is not possible in the BFAPINV preconditioner and the computation of the factors of this preconditioner can not be seperated [12,13]. This indicates that the right-looking version of RIF is also quite different from the IULBF preconditioner. In [12], we have implemented the complete pivoting strategy for the right-looking RIF preconditioner. The main purpose of this paper is to apply a complete pivoting strategy for the IULBF preconditioner. This pivoting will be based on the complete pivoting strategy of the Backward IJK version Gaussian elimination process.

In section 2 of this paper, we first review the Backward form of the IJK version of Gaussian elimination process and then, present its complete pivoting strategy. In section 3, we recall the BFINV algorithm and show that the computed $\bar{W}$, $\bar{D}$ and $\bar{Z}$ factors in this algorithm can implicitly generate the last column and the last row of the Schur-Complement matrices which are

Algorithm 1 (Backward IJK version of Gaussian Elimination process)

Input: $A \in \mathbb{R}^{n\times n}$.
Output: $A = \bar{U}\bar{D}\bar{L}$.

$\bar{U} = \bar{L} = I_n$, $\bar{S}^{(n)} = A$
for $i = n$ to 1 **do**
  $\bar{d}_{ii} = \bar{q}_i^{(i-1)} = \bar{p}_i^{(i-1)} = (\bar{S}^{(i)})_{ii}$
  for $j = i-1$ to 1 **do**
    $\bar{q}_i^{(j-1)} = (\bar{S}^{(i)})_{ij}$, $\bar{p}_i^{(j-1)} = (\bar{S}^{(i)})_{ji}$
    $\bar{L}_{ij} = \frac{\bar{q}_i^{(j-1)}}{\bar{d}_{ii}}$, $\bar{U}_{ji} = \frac{\bar{p}_i^{(j-1)}}{\bar{d}_{ii}}$
  end for
  for $j = i-1$ to 1 **do**
    for $k = i-1$ to 1 **do**
      $(\bar{S}^{(i-1)})_{jk} = (\bar{S}^{(i)})_{jk} - \bar{U}_{ji}\bar{d}_{ii}\bar{L}_{ik}$
    end for
  end for
end for
Return $\bar{U} = (\bar{U}_{ji})_{1\le j,i\le n}$, $\bar{D} = diag(\bar{d}_{ii})_{1\le i\le n}$ and $\bar{L} = (\bar{L}_{ij})_{1\le i,j\le n}$.

computed in the Backward form of the IJK version of Gaussian elimination process. Based on this connection, a complete pivoting strategy for the IULBF preconditioner is proposed in section 4. In section 5, we have reported the numerical results and the implementation details.

2. Backward IJK version of Gaussian elimination process

Algorithm 1, computes the exact factorization

$$A = \bar{U}\bar{D}\bar{L}, \tag{2.1}$$

where $\bar{U}$ and $\bar{L}^T$ are unit upper triangular matrices and $\bar{D}$ is a diagonal matrix. In this algorithm, matrices $\bar{U}$ and $\bar{L}$ are computed column-wise and row-wise, respectively. This algorithm is termed as a backward form since at the end of its i-th step, the columns n to i of matrix $\bar{U}$, the rows n to i of matrix $\bar{L}$ and the diagonal entries $\bar{d}_{jj}$, for $j \ge i$, are computed. At the end of this step, the relation (2.2) holds. For $j \ge i$, the vectors $[\bar{h}_j^T, 1, \overbrace{0, \cdots, 0}^{n-j}]^T$ and $[\bar{g}_j, 1, \overbrace{0, \cdots, 0}^{n-j}]$ in (2.2), are the j-th column of matrix $\bar{U}$ and the j-th row of matrix $\bar{L}$, respectively. In this relation, $\bar{h}_j \in \mathbb{R}^{(j-1)\times 1}$ and $\bar{g}_j \in \mathbb{R}^{1\times(j-1)}$, for $j \ge i$. The submatrix $(\bar{S}^{(i-1)})_{j,k\le i-1}$ is the associated Schur-Complement matrix. The computing pattern of matrices $\bar{U}$, $\bar{D}$, $\bar{L}$ and Schur-Complement can be found in Figure 1. Since the whole Schur-Complement matrices are

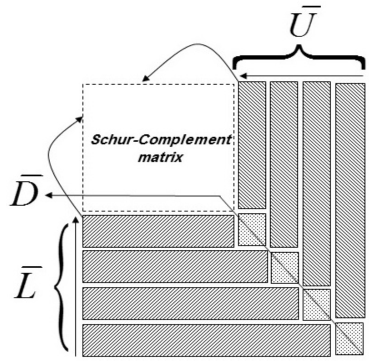


FIGURE 1. *Computing matrices $\bar{U}$, $\bar{D}$, $\bar{L}$ and Schur-Complement in the Backward* IJK *version of Gaussian Elimination process*

available in this algorithm, then we can apply the complete pivoting strategy.

$$A = \underbrace{\begin{bmatrix} 1 & & & & & & \\ & \ddots & & & & & \\ & & 1 & \bar{h}_i & \bar{h}_{i+1} & \cdots & \bar{h}_n \\ & & & 1 & & & \\ & & & & 1 & & \\ & & & & & \ddots & \\ & & & & & & 1 \end{bmatrix}}_{\bar{U}} \times \underbrace{\begin{bmatrix} (\bar{S}^{(i-1)})_{j,k\le i-1} & & & & \\ & \bar{d}_{ii} & & & \\ & & \bar{d}_{i+1,i+1} & & \\ & & & \ddots & \\ & & & & \bar{d}_{nn} \end{bmatrix}}_{\bar{S}^{(i-1)}}$$

$$(2.2) \qquad \times \underbrace{\begin{bmatrix} 1 & & & & & \\ & \ddots & & & & \\ & & 1 & & & \\ & \bar{g}_i & & 1 & & \\ & \bar{g}_{i+1} & & & 1 & \\ & \vdots & & & & \ddots \\ & \bar{g}_n & & & & & 1 \end{bmatrix}}_{\bar{L}},$$

Algorithm 2, is the Backward form of the IJK version of Gaussian elimination process which is coupled with the complete pivoting and dropping. At the end of step $i+1$ of this algorithm, the incomplete factorization 2.4 is computed. For $j \ge i+1$, matrices Π_j and Σ_j are the row and the column permutation matrices associated to step j. Also, $\Pi = \Pi_{i+1}\ \Pi_{i+2}\ ...\ \Pi_n$ and $\Sigma = \Sigma_n\ ...\ \Sigma_{i+2}\ \Sigma_{i+1}$. The submatrix $(S^{(i)})_{j,k\le i}$ is the approximate Schur-Complement matrix. At the end of this algorithm, the matrices U, D, L, Π and Σ will be computed such that

$$(2.3) \qquad \Pi A \Sigma \approx UDL.$$

For $j \geq i+1$, the vectors $[h_j^T, 1, \overbrace{0, \cdots, 0}^{n-j}]^T$ and $[g_j, 1, \overbrace{0, \cdots, 0}^{n-j}]$ in (2.4), are the already computed columns and rows of matrices U and L, respectively and the entries d_{jj} are the diagonal elements of D.

$$\underbrace{\Pi_{i+1}\ \Pi_{i+2}\ \ldots\ \Pi_n}_{\Pi}\ A\ \underbrace{\Sigma_n\ \ldots\ \Sigma_{i+2}\ \Sigma_{i+1}}_{\Sigma} \approx \underbrace{\begin{bmatrix} 1 & & & & & & \\ & \ddots & & & & & \\ & & 1 & h_{i+1} & h_{i+2} & \cdots & h_n \\ & & & 1 & & & \\ & & & & 1 & & \\ & & & & & \ddots & \\ & & & & & & 1 \end{bmatrix}}_{U} \times$$

$$\underbrace{\begin{bmatrix} (S^{(i)})_{j,k\leq i} & & & & \\ & d_{i+1,i+1} & & & \\ & & d_{i+2,i+2} & & \\ & & & \ddots & \\ & & & & d_{nn} \end{bmatrix}}_{S^{(i)}} \underbrace{\begin{bmatrix} 1 & & & & \\ & \ddots & & & \\ & & 1 & & \\ & g_{i+1} & & 1 & \\ & g_{i+2} & & & 1 \\ & \vdots & & & & \ddots \\ & g_n & & & & & 1 \end{bmatrix}}_{L}, \tag{2.4}$$

In this relation, $h_j \in \mathbb{R}^{(j-1)\times 1}$ and $g_j \in \mathbb{R}^{1\times(j-1)}$, for $j \geq i+1$.

Here, we explain the i-th step of Algorithm 2. At the beginning of this step, the elements m_i and n_i are set equal to zero. These two elements will be the number of row and column pivoting at the end of this step. The two logical parameters $satisfied\_\ p$ and $satisfied\_\ q$ are initialized as $false$ in line 4 of the algorithm. When $satisfied\_\ p$ ($satisfied\_\ q$) is $false$, then this indicates that we should apply the row (column) pivoting. Since $satisfied\_\ p$ is $false$, then the internal *while* loop will be run. In lines 6-8 of the algorithm, the vector $(p_i^{(0)}, p_i^{(1)}, \cdots, p_i^{(i-1)})^T$ is obtained which is the last column of the approximate Schur-Complement matrix $(S^{(i)})_{j,k\leq i}$. Suppose that $|p_i^{(k-1)}| = \max_{m\leq i} |p_i^{(m-1)}|$. In line 9 of the algorithm, we check whether the row pivoting criterion

$$|p_i^{(i-1)}| < \alpha |p_i^{(k-1)}|, \tag{2.5}$$

is satisfied for $\alpha \in (0,1]$. In (2.5), α is a parameter which controls the pivoting process. If this criterion is satisfied, then the lines 10-14 of the algorithm will be run. In these lines, m_i is incremented by one, $\pi_{m_i}^{(i)}$ is set equal to the identity matrix and $satisfied\_\ q$ is set to $false$ which means that after the row pivoting one should also apply the column pivoting. Also, the rows i and k of matrices $U - I$ and $\pi_{m_i}^{(i)}$ and the entries $p_i^{(i-1)}$ and $p_i^{(k-1)}$ are interchanged in these lines and matrices $S^{(i)}$ and Π are updated. After the row pivoting, $satisfied\_\ p$ is set to $true$ in line 16 of the algorithm. The vector $(q_i^{(0)}, q_i^{(1)}, \cdots, q_i^{(i-1)})$ which is the last row of the approximate Schur-Complement matrix $(S^{(i)})_{j,k\leq i}$

is obtained in lines 17-19. Since the parameter $satisfied\_q$ is $false$, then the lines 21-27 of the algorithm are run. Suppose that $|q_i^{(l-1)}| = \max_{m\leq i} |q_i^{(m-1)}|$. In line 21 of the algorithm, the column pivoting criterion

$$|q_i^{(i-1)}| < \alpha |q_i^{(l-1)}|, \tag{2.6}$$

is checked. In (2.6), $\alpha \in (0, 1]$ is again the pivoting parameter. If this criterion is satisfied, then the parameter n_i is incremented by one, the matrix $\sigma_{n_i}^{(i)}$ is initialized as the identity matrix and $satisfied\_p$ is set to $false$ which indicates that after the column pivoting we should again apply the row pivoting strategy. Also, the columns i and l of matrices $L-I$ and $\sigma_{n_i}^{(i)}$ and the elements $q_i^{(i-1)}$ and $q_i^{(l-1)}$ are interchanged and matrices $S^{(i)}$ and Σ are updated. After the column pivoting strategy, the parameter $satisfied\_q$ is set to $true$ in line 29 of the algorithm and the internal $while$ loop will be run again. This will be continued until a desired pivot element will be obtained. In line 31 of the algorithm, the (i, i) entry of the approximate Schur-Complement matrix $(S^{(i)})_{j,k\leq i}$ is defined as the (i, i) entry of matrix D. In lines 32-35 of the algorithm, the i-th column of matrix U and the i-th row of matrix L are computed and dropped. The dropping criterion is checked in line 34 of the algorithm. In lines 36-40 of the algorithm, the new approximate Schur-Complement matrix $(S^{(i-1)})_{j,k\leq i-1}$ is obtained.

If we define $\Pi_i = \pi_{m_i}^{(i)} \pi_{m_i-1}^{(i)} \cdots \pi_1^{(i)}$, $\Sigma_i = \sigma_1^{(i)} \cdots \sigma_{n_i-1}^{(i)} \sigma_{n_i}^{(i)}$ and if we consider $[h_i^T, 1, \overbrace{0, \cdots, 0}^{n-i}]^T$ and $[g_i, 1, \overbrace{0, \cdots, 0}^{n-i}]$ as the i-th column of matrix U and the i-th row of matrix L, respectively, then at the end of step i of Algorithm 2, the relation

$$\underbrace{\Pi_i\ \Pi_{i+1} \ldots \Pi_n}_{\Pi}\ A\ \underbrace{\Sigma_n \ldots \Sigma_{i+1}\ \Sigma_i}_{\Sigma} \approx \underbrace{\begin{bmatrix} 1 & & & & & & \\ & \ddots & & h_i & h_{i+1} & \cdots & h_n \\ & & 1 & & & & \\ & & & 1 & & & \\ & & & & 1 & & \\ & & & & & \ddots & \\ & & & & & & 1 \end{bmatrix}}_{U} \times$$

$$\underbrace{\begin{bmatrix} (S^{(i-1)})_{j,k\leq i-1} & & & & \\ & d_{ii} & & & \\ & & d_{i+1,i+1} & & \\ & & & \ddots & \\ & & & & d_{nn} \end{bmatrix}}_{S^{(i-1)}} \underbrace{\begin{bmatrix} 1 & & & & & \\ & \ddots & & & & \\ & & 1 & & & \\ & g_i & & 1 & & \\ & g_{i+1} & & & 1 & \\ & \vdots & & & & \ddots & \\ & g_n & & & & & 1 \end{bmatrix}}_{L},$$

holds.

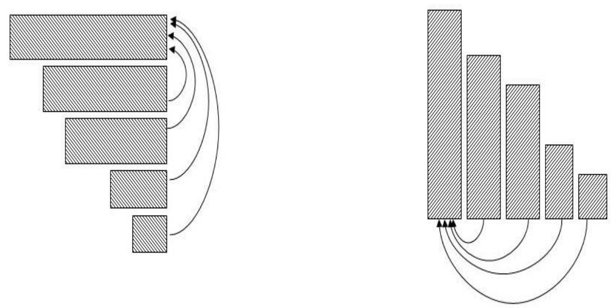
Algorithm 2 (Backward IJK version of Gaussian Elimination process with complete pivoting and dropping)

Input: $A \in \mathbb{R}^{n\times n}$, τ_l and $\tau_u \in (0,1)$ be the drop tolerances for L and U matrices and prescribe a pivoting tolerance $\alpha \in (0,1]$
Output: $\Pi A \Sigma \approx UDL$

$U = L = \Pi = \Sigma = I_n$, $S^{(n)} = A$
for $i = n$ to 1 **do**
  $m_i = n_i = 0$
  $satisfied\_\,p = false$, $satisfied\_\,q = false$
  while not $satisfied\_\,p$ **do**
    for $j = i$ to 1 **do**
      $p_i^{(j-1)} = e_j^T\ S^{(i)}\ e_i$
    end for
    if $|p_i^{(i-1)}| < \alpha\ \max_{m\leq i} |p_i^{(m-1)}|$ **then**
      $m_i = m_i + 1,\ \pi_{m_i}^{(i)} = I_n$
      $satisfied\_\,q = false$, choose k such that $|p_i^{(k-1)}| = \max_{m\leq i} |p_i^{(m-1)}|$
      Interchange the rows i and k of $U - I$ and $\pi_{m_i}^{(i)}$ and the elements $p_i^{(i-1)}$ and $p_i^{(k-1)}$
      $S^{(i)} = \pi_{m_i}^{(i)}\ S^{(i)}$
      $\Pi = \pi_{m_i}^{(i)}\ \Pi$
    end if
    $satisfied\_\,p = true$
    for $j = i$ to 1 **do**
      $q_i^{(j-1)} = e_i^T\ S^{(i)}\ e_j$
    end for
    if not $satisfied\_\,q$ **then**
      if $|q_i^{(i-1)}| < \alpha\ \max_{m\leq i} |q_i^{(m-1)}|$ **then**
        $n_i = n_i + 1,\ \sigma_{n_i}^{(i)} = I_n$
        $satisfied\_\,p = false$, choose l such that $|q_i^{(l-1)}| = \max_{m\leq i} |q_i^{(m-1)}|$
        Interchange the columns i and l of $L - I$ and $\sigma_{n_i}^{(i)}$ and the elements $q_i^{(i-1)}$ and $q_i^{(l-1)}$
        $S^{(i)} = S^{(i)}\ \sigma_{n_i}^{(i)}$
        $\Sigma = \Sigma\ \sigma_{n_i}^{(i)}$
      end if
    end if
    $satisfied\_\,q = true$
  end while
  $d_{ii} = e_i^T\ S^{(i)}\ e_i$ {*Consider that* $e_i^T\ S^{(i)}\ e_i = p_i^{(i-1)} = q_i^{(i-1)}$}
  for $j = i-1$ to 1 **do**
    $L_{ij} = \frac{q_i^{(j-1)}}{d_{ii}}$, $U_{ji} = \frac{p_i^{(j-1)}}{d_{ii}}$
    If $|L_{ij}| < \tau_l$, then set $L_{ij} = 0$. Also if $|U_{ji}| < \tau_u$, then set $U_{ji} = 0$
  end for
  for $j = i-1$ to 1 **do**
    for $k = i-1$ to 1 **do**
      $(S^{(i-1)})_{jk} = (S^{(i)})_{jk} - U_{ji} d_{ii} L_{ik}$
    end for
  end for
end for
Return $L = (L_{ij})_{1\leq i,j\leq n}$, $D = diag(d_{ii})_{1\leq i\leq n}$, $U = (U_{ji})_{1\leq j,i\leq n}$, Π and Σ

3. Backward Factored APproximate INVerse process

Algorithm 3, computes the factorization (1.2). This algorithm is termed as a backward form since at the end of its i-th step, for $j \geq i$, the vectors $\bar{w}_j^{(n-j)}$



(a) (b)

FIGURE 2. (a) *Pattern of the update for the rows of matrix* $\bar{W}$ *in Algorithm* 3. (b) *Pattern of the update for the columns of matrix* $\bar{Z}$ *in Algorithm* 3

which are the rows n to i of matrix $\bar{W}$, the vectors $\bar{z}_j^{(n-j)}$ which are the columns n to i of matrix $\bar{Z}$ and the entries $\bar{d}_{jj}$ are computed.

Algorithm 3 (BFINV algorithm)

Input: $A \in \mathbb{R}^{n \times n}$
Output: $A^{-1} = \bar{Z}\bar{D}^{-1}\bar{W}$

1. **for** $i = n$ to 1 **do**
2. $\quad \bar{w}_i^{(0)} = e_i^T$, $\bar{z}_i^{(0)} = e_i$.
3. $\quad$ **for** $j = i + 1$ to n **do**
4. $\quad\quad \bar{p}_j^{(i-1)} = e_i^T A \bar{z}_j^{(n-j)}$ $\bar{q}_j^{(i-1)} = \bar{w}_j^{(n-j)} A e_i$
5. $\quad\quad \bar{z}_i^{(j-i)} = \bar{z}_i^{(j-i-1)} - \frac{\bar{q}_j^{(i-1)}}{\bar{d}_{jj}} \bar{z}_j^{(n-j)}$, $\bar{w}_i^{(j-i)} = \bar{w}_i^{(j-i-1)} - \frac{\bar{p}_j^{(i-1)}}{\bar{d}_{jj}} \bar{w}_j^{(n-j)}$
6. $\quad$ **end for**
7. $\quad \bar{d}_{ii} = \bar{w}_i^{(n-i)} A e_i$
8. **end for**
9. Return $\bar{W} = [(\bar{w}_1^{(n-1)})^T, (\bar{w}_2^{(n-2)})^T, \ldots, (\bar{w}_n^{(0)})^T]^T$, $\bar{D} = diag(\bar{d}_{ii})_{1 \le i \le n}$ and $\bar{Z} = [\bar{z}_1^{(n-1)}, \bar{z}_2^{(n-2)}, \ldots, \bar{z}_n^{(0)}]$.

Consider step i of Algorithm 3. In the internal j loop of this step, a linear combination of the already obtained columns $\bar{z}_j^{(n-j)}$, for $j \geq i+1$, will compute the column $\bar{z}_i^{(n-i)}$ of matrix $\bar{Z}$. Also, a linear combination of the already obtained rows $\bar{w}_j^{(n-j)}$, for $j \geq i+1$, are used to compute the row $\bar{w}_i^{(n-i)}$ of matrix $\bar{W}$. In Figure 2, we have drawn a pattern for computing the rows of matrix $\bar{W}$ and the columns of matrix $\bar{Z}$ in this algorithm.

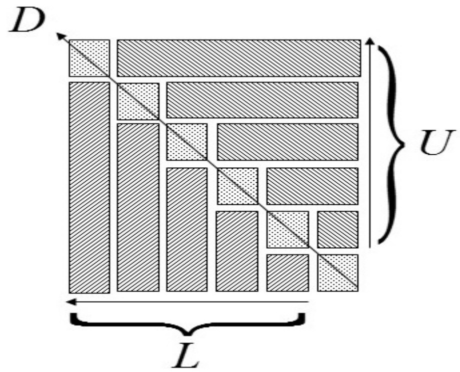


FIGURE 3. *Computing pattern of the U, D and L factors in Algorithm 4*

Recall that at the beginning of step $i+1$ of Algorithm 1, the Schur-Complement matrices $(\bar{S}^{(j)})_{k,l\leq j}$, for $j \geq i+1$, were already obtained. There is the relation

$$(\bar{S}^{(j)})_{ij} = e_i^T A \bar{z}_j^{(n-j)}, \qquad (\bar{S}^{(j)})_{ji} = (\bar{w}_j^{(n-j)}) A e_i, \qquad j \geq i+1,$$

which connects the Schur-Complement matrices in Algorithm 1 to the columns and rows of matrices $\bar{Z}$ and $\bar{W}$ of Algorithm 3. More details can be found in [13]. Therefore, we can use the relation

$$\bar{U}_{ij} = \frac{e_i^T A \bar{z}_j^{(n-j)}}{\bar{d}_{jj}}, \qquad \bar{L}_{ji} = \frac{(\bar{w}_j^{(n-j)}) A e_i}{\bar{d}_{jj}}, \qquad j \geq i+1, \tag{3.1}$$

to compute the i-th row and the i-th column of matrices $\bar{U}$ and $\bar{L}$ in (2.1).

Since we use the dropping strategy in line 8 of Algorithm 4, then the matrices $W = [(w_1^{(n-1)})^T, \cdots, (w_n^{(0)})^T]^T$, $Z = [z_1^{(n-1)}, \cdots, z_n^{(0)}]$ and $D = diag(d_{ii})_{1\leq i\leq n}$ are computed which are the approximations of the matrices $\bar{W} = [(\bar{w}_1^{(n-1)})^T, \cdots, (\bar{w}_n^{(0)})^T]^T$, $\bar{Z} = [\bar{z}_1^{(n-1)}, \cdots, \bar{z}_n^{(0)}]$ and $\bar{D} = diag(\bar{d}_{ii})_{1\leq i\leq n}$ computed in Algorithm 3. The incomplete factorization in (1.4) is also computed as the by-product of Algorithm 4. Based on the two relations in (3.1), the entries of matrices U and L are computed in lines 4 and 5 of this algorithm. After merging the factors D and L, this incomplete factorization is termed as the IULBF preconditioner [13]. The factors U and L of this preconditioner are computed row-wise and column-wise, respectively. The computation of these two factors does not depend on each other. Figure 3, shows the pattern of the computation for matrices U, D and L of this preconditioner.

Algorithm 4 (IULBF preconditioner obtained from BFAPINV process)

Input: $A \in \mathbb{R}^{n\times n}$ and $\tau_l, \tau_u, \tau_z, \tau_w \in (0,1)$ be drop tolerance parameters
Output: $A \approx UDL$

1. **for** $i = n$ to 1 **do**
2. $\quad w_i^{(0)} = e_i^T$, $z_i^{(0)} = e_i$.
3. $\quad$ **for** $j = i+1$ to n **do**
4. $\qquad p_j^{(i-1)} = e_i^T A z_j^{(n-j)}$ $q_j^{(i-1)} = w_j^{(n-j)} A e_i$
5. $\qquad U_{ij} = \frac{p_j^{(i-1)}}{d_{jj}}$ $L_{ji} = \frac{q_j^{(i-1)}}{d_{jj}}$
6. $\qquad$ If $|L_{ji}| < \tau_l$, then set $L_{ji} = 0$. Also if $|U_{ij}| < \tau_u$, then set $U_{ij} = 0$
7. $\qquad z_i^{(j-i)} = z_i^{(j-i-1)} - \frac{q_j^{(i-1)}}{d_{jj}} z_j^{(n-j)}$, $w_i^{(j-i)} = w_i^{(j-i-1)} - \frac{p_j^{(i-1)}}{d_{jj}} w_j^{(n-j)}$
8. $\qquad$ For all $l \geq j$, if $|z_{li}^{(j-i)}| < \tau_z$ and $|w_{il}^{(j-i)}| < \tau_w$, then set $z_{li}^{(j-i)} = 0$ and $w_{il}^{(j-i)} = 0$
9. $\quad$ **end for**
10. $\quad d_{ii} = w_i^{(n-i)} A e_i$
11. **end for**
12. Return $U = (U_{ij})_{1\leq i,j\leq n}$, $D = diag(d_{ii})_{1\leq i\leq n}$ and $L = (L_{ji})_{1\leq j,i\leq n}$.

At the beginning of step i of Algorithm 1, the Schur-Complement matrix $(\bar{S}^{(i)})_{j,k\leq i}$ is available. Also, at the end of step i of Algorithm 3, the row $\bar{w}_i^{(n-i)}$ and the column $\bar{z}_i^{(n-i)}$ have been computed. The relation

$$(\bar{S}^{(i)})_{ji} = \bar{p}_i^{(j-1)} = e_j^T A \bar{z}_i^{(n-i)}, \qquad (\bar{S}^{(i)})_{ij} = \bar{q}_i^{(j-1)} = (\bar{w}_i^{(n-i)}) A e_j, \quad j \leq i, \tag{3.2}$$

enables us to only obtain the last column and the last row of the Schur-Complement matrix $(\bar{S}^{(i)})_{j,k\leq i}$ [7]. Therefore, this relation also connects the two Algorithms 1 and 3. This relation will help us in Algorithm 5 to extend the complete pivoting strategy of the Backward form of the IJK version of Gaussian Elimination process to the complete pivoting strategy for the IULBF preconditioner.

4. Complete pivoting strategy for the IULBF preconditioner

In Algorithm 5, we use a complete pivoting strategy to obtain the incomplete factorization (2.3). We term this incomplete factorization as the IULBF preconditioner with complete pivoting strategy. The pivoting strategy of this algorithm is based on the complete pivoting strategy of the Backward IJK version of Gaussian elimination process.

At the end of step $i+1$ of this algorithm, suppose that $\Pi = \Pi_{i+1}\Pi_{i+2}\cdots\Pi_n$ and $\Sigma = \Sigma_n \cdots \Sigma_{i+2}\Sigma_{i+1}$ where Π_j and Σ_j, for $j \geq i+1$, are the row and the column permutation matrices associated to step j of this algorithm. Also, consider that the columns n to $i+1$ of matrix L, the rows n to $i+1$ of matrix U and the entries d_{jj}, for $j \geq i+1$, have already been computed. Here, we explain the step i of this algorithm. In line 2, we initialize the parameters m_i, n_i and $iter$. At the end of this step, m_i and n_i will be the number of row and column pivoting strategies, respectively. The parameter $iter$ will help us in line 12 to compute the pivot entry. In line 3, the two logical variables $satisfied\_\, p$ and

$satisfied\_q$ are set equal to $false$. When $satisfied\_p$ ($satisfied\_q$) is $false$, then we need to apply the row (column) pivoting. Since $satisfied\_p$ is $false$, then the algorithm will enter the internal *while* loop. In line 5, the parameter *iter* is incremented by one. In lines 6-11 of the algorithm, the column vector $z_i^{(n-i)}$ is computed. As we explained before, at the end of step $i+1$ of Algorithm 2, the relation (2.4) holds and therefore, the approximate Schur-Complement matrix $(S^{(i)})_{j,k\leq i}$ is available. In lines 12-15 of Algorithm 5, the relation

$$(S^{(i)})_{ji} \approx p_i^{(j-1)} = e_j^T(\Pi A\Sigma)z_i^{(n-i)}, \qquad j \leq i, \tag{4.1}$$

enables us to implicitly approximate the last column of the approximate Schur-Complement matrix $(S^{(i)})_{j,k\leq i}$. This relation has been written based on the first part of relation (3.2). We have mentioned in line 12 that if only *iter* is equal to 1, then $(S^{(i)})_{ii}$ can be approximated from (4.1). In lines 16-22 of the algorithm, we are applying the row pivoting strategy. Suppose that $|p_i^{(k-1)}| = \max_{m\leq i}|p_i^{(m-1)}|$. In these lines, we first check whether the row pivoting criterion (2.5) is satisfied. If yes, then m_i is incremented by one, the matrix $\pi_{m_i}^{(i)}$ is initialized as the identity matrix and then, the rows i and k of this matrix will be interchanged. Also, $satisfied\_q$ is set to $false$, the entries $p_i^{(i-1)}$ and $p_i^{(k-1)}$ are interchanged and the matrix Π is updated by $\pi_{m_i}^{(i)}$. The lines 16-22 of Algorithm 5 are the same as the lines 9-15 of Algorithm 2, except that in Algorithm 5, there is no need to update the matrix $S^{(i)}$ and to interchange the rows i and k of matrix $U - I$. After the row pivoting strategy, we set $satisfied\_p$ to *true* in line 23 of Algorithm 5. In line 24 of this algorithm, we check whether the column pivoting is needed. Since $satisfied\_q$ is $false$, then the lines 25-43 of the algorithm will be run. In lines 25-30, the row vector $w_i^{(n-i)}$ is computed. In line 31, we set the pivot entry $q_i^{(i-1)}$ equal to the entry $p_i^{(i-1)}$ which was an approximation for the (i,i) entry of $(S^{(i)})_{j,k\leq i}$. In lines 32-34, we use the relation

$$(S^{(i)})_{ij} \approx q_i^{(j-1)} = w_i^{(n-i)}(\Pi A\Sigma)e_j, \qquad j < i,$$

to implicitly approximate the rest of the entries of the last row of the approximate Schur-Complement matrix $(S^{(i)})_{j,k\leq i}$. This relation is proposed based on the second part of relation (3.2). The column pivoting strategy is applied in lines 35-41 of the algorithm. Suppose that $|q_i^{(l-1)}| = \max_{m\leq i}|q_i^{(m-1)}|$. In these lines, we first test whether the column pivoting criterion (2.6) is satisfied. If yes, then n_i is incremented by one, $\sigma_{n_i}^{(i)}$ is initialized as the identity matrix and then, the columns i and l of this matrix will be interchanged. Also, the parameter $satisfied\_p$ is set to $false$, the elements $q_i^{(i-1)}$ and $q_i^{(l-1)}$ are interchanged and the matrix Σ will be updated by $\sigma_{n_i}^{(i)}$. Comparing the lines 35-41 of Algorithm 5 by the lines 21-27 of Algorithm 2 indicates that there are differences between the column pivoting strategies of the two algorithms.

Algorithm 5 (**IULBF preconditioner coupled with complete pivoting strategy**)

Input: Let $A \in \mathbb{R}^{n\times n}$, $U = L = \Pi = \Sigma = I_n$, $\tau_w, \tau_z, \tau_l, \tau_u \in (0,1)$ be drop tolerances and prescribe a pivoting tolerance $\alpha \in (0,1]$.
Output: $\Pi A \Sigma \approx UDL$.

for $i = n$ to 1 **do**
  $m_i = n_i = iter = 0$
  $satisfied\_p = satisfied\_q = false$
  while not $satisfied\_p$ **do**
    $iter = iter + 1$
    $z_i^{(0)} = e_i$
    for $j = i+1$ to n **do**
      $q_j^{(i-1)} = w_j^{(n-j)}(\Pi A \Sigma)e_i$
      $z_i^{(j-i)} = z_i^{(j-i-1)} - (\frac{q_j^{(i-1)}}{d_{jj}})z_j^{(n-j)}$
      For all $l \geq j$, if $|z_{li}^{(j-i)}| < \tau_z$, then set $z_{li}^{(j-i)} = 0$
    end for
    If $iter = 1$, then set $p_i^{(i-1)} = e_i^T(\Pi A\Sigma)z_i^{(n-i)}$. Otherwise set $p_i^{(i-1)} = q_i^{(i-1)}$
    for $j = i-1$ to 1 **do**
      $p_i^{(j-1)} = e_j^T(\Pi A \Sigma)z_i^{(n-i)}$
    end for
    if $|p_i^{(i-1)}| < \alpha \ \max_{m\leq i}|p_i^{(m-1)}|$ **then**
      $m_i = m_i + 1$, $\pi_{m_i}^{(i)} = I_n$.
      $satisfied\_q = false$
      Choose k such that $|p_i^{(k-1)}| = \max_{m\leq i}|p_i^{(m-1)}|$.
      Interchange the rows i and k of $\pi_{m_i}^{(i)}$ and the elements $p_i^{(i-1)}$ and $p_i^{(k-1)}$
      $\Pi = \pi_{m_i}^{(i)}\Pi$
    end if
    $satisfied\_p = true$
    if not $satisfied\_q$ **then**
      $w_i^{(0)} = e_i^T$
      for $j = i+1$ to n **do**
        $p_j^{(i-1)} = e_i^T(\Pi A \Sigma)z_j^{(n-j)}$
        $w_i^{(j-i)} = w_i^{(j-i-1)} - (\frac{p_j^{(i-1)}}{d_{jj}})w_j^{(n-j)}$
        For all $l \geq j$, if $|w_{il}^{(j-i)}| < \tau_w$, then set $w_{il}^{(j-i)} = 0$
      end for
      $q_i^{(i-1)} = p_i^{(i-1)}$
      for $j = i-1$ to 1 **do**
        $q_i^{(j-1)} = w_i^{(n-i)}(\Pi A \Sigma)e_j$
      end for
      if $|q_i^{(i-1)}| < \alpha \ \max_{m\leq i}|q_i^{(m-1)}|$ **then**
        $n_i = n_i + 1$, $\sigma_{n_i}^{(i)} = I_n$
        $satisfied\_p = false$
        Choose l such that $|q_i^{(l-1)}| = \max_{m\leq i}|q_i^{(m-1)}|$
        Interchange the columns i and l of $\sigma_{n_i}^{(i)}$ and the elements $q_i^{(i-1)}$ and $q_i^{(l-1)}$
        $\Sigma = \Sigma\sigma_{n_i}^{(i)}$
      end if
      $satisfied\_q = true$
    end if
  end while
  $d_{ii} = p_i^{(i-1)}$
  for $j = i+1$ to n **do**
    $L_{ji} = \frac{q_j^{(i-1)}}{d_{jj}}$, $U_{ij} = \frac{p_j^{(i-1)}}{d_{jj}}$
    If $|L_{ji}| < \tau_l$, then set $L_{ji} = 0$. Also if $|U_{ij}| < \tau_u$, then set $U_{ij} = 0$.
  end for
end for
Return $L = (L_{ji})_{1\leq j,i\leq n}$, $D = diag(d_{ii})_{1\leq i\leq n}$, $U = (U_{ij})_{1\leq i,j\leq n}$, Π and Σ.

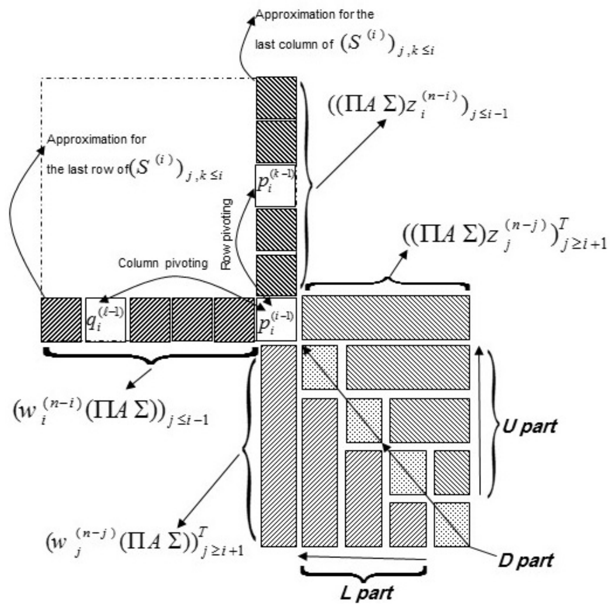


FIGURE 4. *Row and column pivoting strategies in step i of Algorithm 5*

Despite the column pivoting strategy of Algorithm 2, there is no need to interchange the columns i and l of matrix $L - I$ and to update matrix $S^{(i)}$ in the column pivoting strategy of Algorithm 5. After the column pivoting strategy, the parameter $satisfied\_q$ is set to $true$ in line 42 of Algorithm 5. Since $satisfied\_p$ is $false$, then the internal *while* loop should be run one more time. At the end of this loop, we set the (i, i) entry of matrix D equal to the element $p_i^{(i-1)}$ in line 45 of the algorithm. The i-th column of matrix L and the i-th row of matrix U are computed as the by-products in lines 46-49 of the algorithm.

In Figure 4, we have drawn a pattern for the row and the column pivoting strategies in step i of Algorithm 5.

5. Numerical results and implementation details

In this section, we report the results of numerical experiments to study the effectiveness of complete pivoting on the quality of the IULBF preconditioner. We have also presented some comparison between the three preconditioners ILUTP [17], left-looking RIF with complete pivoting [16] and IULBF with complete pivoting. This comparison is based on the results for 165 artificial linear systems where the coefficient matrices have been downloaded from [4].

We have proposed the names of these matrices in Table 1. The solution of the systems are the vectors $e = [1, \cdots, 1]^T$ and the right hand side vectors are $b = Ae$. We have applied all the preconditioners as the right preconditioner for these linear systems and then have solved the preconditioned systems by the GMRES(30) method. The code of GMRES can be found in [18]. For all the systems, the initial solution is taken as the zero vector and the stopping criterion is satisfied when the relative residual is less than 10^{-8}. For the original linear systems we have considered 5000 as the maximum number of iterations of the GMRES(30) method while for the preconditioned systems this value has been set to 2500. We have written the codes of plain IULBF and IULBF with complete pivoting strategy in Fortran 77.

We have considered the following details in the implementation of Algorithms 4 and 5.

- Matrix A is stored in CSR and CSC formats.
- Matrices Z and W are stored in CSC and CSR formats, respectively. This item is associated to line 7 of Algorithm 4 and to lines 9 and 28 of Algorithm 5.
- To break the complexity of these two algorithms, we need to access matrices Z and W row-wise and column-wise, respectively. For this aim, we have also stored matrices Z and W in dynamic sparse row and dynamic sparse column formats, respectively. For more details about these two formats see [1].
- The arrays *invpermw* and *permw* are used to store the information of matrices Π and Π^T, respectively. Also, the arrays *sigmaz* and *invsigmaz* are used to consider the information of the matrices Σ and Σ^T, respectively.

The first, third and fourth items are essential for the efficient implementation of line 4 of Algorithm 4 and lines 8 and 27 of Algorithm 5. These items will shorten the running time of these two algorithms.

The code of left-looking RIF with complete pivoting is also in Fortran 77. The code of ILUTP is available in [18]. All the numerical experiments have been run on a computing server with 30 GB of RAM. For plain IULBF, IULBF with complete pivoting and left-looking RIF with complete pivoting we have applied the multilevel nested dissection reordering [3, 10] while for ILUTP the RCM [3, 8] has been used as the reordering. This is why we have used the notations Metis and RCM in the title of Figures 5-13 and 17-25. For all the linear systems the maximum weighted matching process [6] has been coupled with the reorderings. This is the reason we have mentioned MC64 in the title of all figures. This process is available in the MC64 package of the HSL library [21].

TABLE 1. All the test matrices

Matrix Name		Matrix Name		Matrix Name	
af23560	airfoil_2d	ASIC_100ks	ASIC_680ks	ASIC_320ks	atmosmodd
atmosmodj	barrier2-9	bcircuit	cage13	cage14	cavity05
cavity10	cavity11	cavity12	cavity13	cavity16	cavity17
cavity18	cavity19	cavity20	cell1	cell2	Chebyshev3
chem_master1	chipcool0	chipcool1	circuit5M_dc	circuit_1	circuit_2
Circuit_3	comsol	coupled	crashbasis	cryg10000	ecl32
epb1	epb2	epb3	ex24	ex29	ex31
ex36	ex37	ex40	flowmeter5	Freescale1	FEM_3D_thermal1
FEM_3D_thermal2	garon1	hvdc2	hcircuit	ibm_matrix_2	jan99jac040sc
jan99jac100sc	jan99jac120sc	Kaufhold	language	lung2	matrix-9
matrix-new_3	memchip	memplus	mult_dcop_01	mult_dcop_02	mult_dcop_03
nmos3	ohne2	para-4	poisson3Da	poisson3Db	poli_large
powersim	rajat03	rajat15	rajat16	rajat18	rajat20
rajat21	rajat22	rajat25	rajat27	rajat28	raefsky1
raefsky2	raefsky5	raefsky6	Raj1	sme3Da	stomach
scircuit	sherman3	shyy161	swang1	swang2	thermal
thermomech_dK	tmt_unsym	torso2	torso3	trans4	trans5
transient	Transport	TSOPF_RS_b39_c7	TSOPF_RS_b39_c19	TSOPF_RS_b39_c30	TSOPF_RS_b162_c1
TSOPF_RS_b162_c3	TSOPF_RS_b162_c4	TSOPF_RS_b2052_c1	TSOPF_RS_b300_c2	TSOPF_RS_b300_c3	TSOPF_RS_b678_c1
utm5940	venkat01	venkat25	venkat50	wang3	wang4
Zhao1	bp_200	bp_400	bp_600	bp_1000	bp_1200
bp_1400	bp_1600	fs_541_1	fs_541_2	fs_541_3	fs_541_4
fs_680_1	fs_680_2	fs_680_3	fs_760_1	fs_760_2	fs_760_3
gemat11	gemat12	gre_512	gre_1107	lns_511	lns_3937
lnsp_511	lnsp3937	mahindas	nnc666	nnc1374	orani678
orsirr_1	orsirr_2	orsreg_1	pores_2	pores_3	psmigr_2
psmigr_3	sherman2	sherman4	sherman5	steam2	west0655
west0989	west1505	west2021			

The density of all preconditioners is defined as

$$density = \frac{nnz(L) + nnz(U)}{nnz(A)},$$

where $nnz(L)$, $nnz(U)$ and $nnz(A)$ are the number of nonzero entries of matrices L, U and A, respectively.

We have seperated the numerical experiments of this paper to two parts. In the next subsection we explain the first part.

5.1. **First part of experiments.** For all 165 linear systems we have considered $\tau_w = \tau_z = 0.1$ and $\tau_l = \tau_u = 0.001$ and have computed the plain IULBF preconditioner. In Tables 3 and 4, and in Figures 5 and 6, the notation IULBF(0.1,0.001) refers to this case.

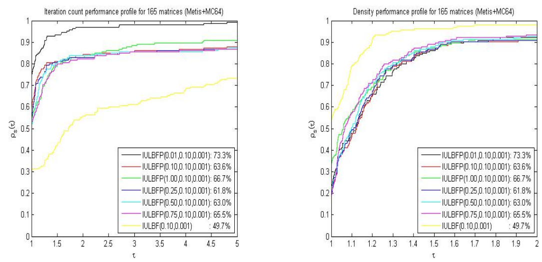
For all the linear systems, we have set $\tau_w = \tau_z = 0.1$ and $\tau_l = \tau_u = 0.001$ and then computed the IULBF with complete pivoting for $\alpha =$0.01, 0.1, 0.25, 0.5, 0.75, 1.0. In Tables 3 and 4 and in Figures 5-7 and 14-16, we have used the notation IULBFP(α,0.1,0.001) for these cases. For these preconditioners we have plotted the number of iterations, density, preconditioning time, total time, total number of row and total number of column pivoting performance profiles in Figures 5-7. As in [5], we here review the concept of performance profile for these parameters. Consider S as the set of all preconditioners IULBF(0.1,0.001) and IULBFP(α,0.1,0.001), for $\alpha =$0.01, 0.1, 0.25, 0.5, 0.75, 1.0. Also let p be one of the 165 test linear systems. If $s \in S$, then the performance ratio $r_{p,s}$ is defined as

$$r_{p,s} = \frac{t_{p,s}}{\min\{t_{p,s} | s \in S\}}, \tag{5.1}$$

where $t_{p,s}$ is the required preconditioning time to compute the preconditioner s for system p. The distributed function for the performance ratio is

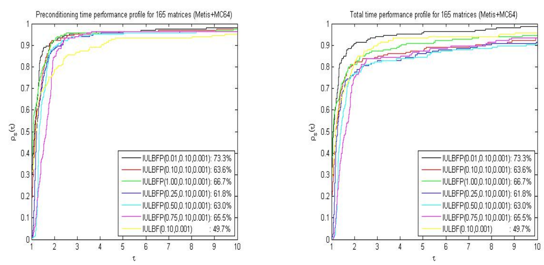
$$\rho_s(\tau) = \frac{1}{165} size(\{p \in P | r_{p,s} \le \tau\}), \tag{5.2}$$

where P is the set of all linear systems. This distributed function is known as the performance profile of the preconditioning time associated to s. As it is claimed in [5], if P is suitably large, then the preconditioners with larger $\rho_s(\tau)$ need the less preconditioning time than the other preconditioners.



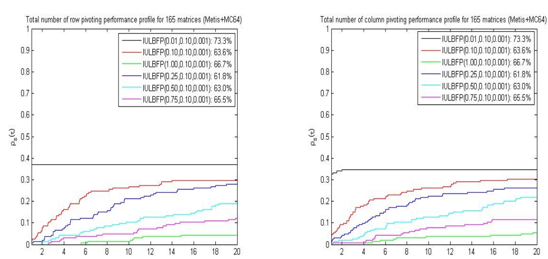
(a) (b)

FIGURE 5. (a) *Number of iterations performance profile for* IULBFP(α,0.1,0.001) *and* IULBF(0.1,0.001). (b) *Density performance profile for* IULBFP(α,0.1,0.001) *and* IULBF(0.1,0.001)



(a) (b)

FIGURE 6. (a) *Preconditioning time performance profile for* IULBFP(α,0.1,0.001) *and* IULBF(0.1,0.001). (b) *Total time performance profile for* IULBFP(α,0.1,0.001) *and* IULBF(0.1,0.001)



(a) (b)

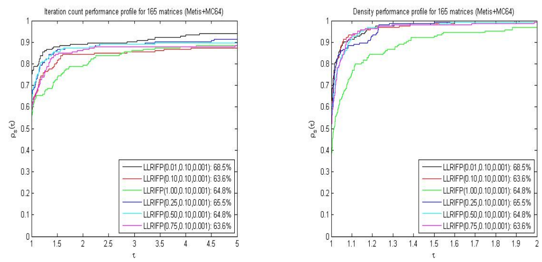
FIGURE 7. (a) *Total number of row pivoting performance profile for* IULBFP(α,0.1,0.001). (b) *Total number of column pivoting performance profile for* IULBFP(α,0.1,0.001)

If in (5.1) we replace $t_{p,s}$ by the density, total time, total number of row and total number of column pivoting, then $\rho_s(\tau)$ in (5.2) will be the associated performance profile of these parameters. We define the GMRES(30) method which is coupled with one of the preconditioners IULBF(0.1,0.001) and IULBFP(α,0.1,0.001), for α =0.01, 0.1, 0.25, 0.5, 0.75, 1.0 as a solver. Consider S_1 as the set of these solvers. For $s \in S_1$, if in (5.1) we replace the preconditioning time $t_{p,s}$ by the number of iterations of the solver s, then $r_{p,s}$ will be the performance ratio for the number of iterations and $\rho_s(\tau)$ in (5.2) will be the performance profile associated to the number of iterations. It should be mentioned that the larger number of iteration performance profile for a solver s is preferred since it indicates that the less number of iterations is required. In Figures 5-7, one can also find the associated performance profile plots for IULBF(0.1,0.001) preconditioner.

In Figures 5 and 6, we have reported the percentage of the solved right preconditioned systems by each of the preconditioners. From these figures, one can come to the following observations. For $\alpha = 0.01, 0.1, \ldots, 1.0$, all of the preconditioners IULBFP(α,0.1,0.001), make the GMRES(30) method convergent in less number of iterations than the IULBF(0.1,0.001) preconditioner. The choice of $\alpha = 0.01$ gives less number of iterations of the GMRES(30) method while it needs less total number of row and less total number of column pivoting than the other choices of α.

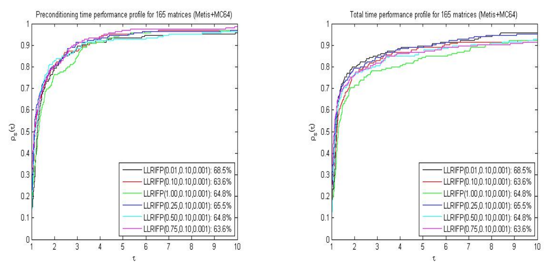
The density and preconditioning time of IULBFP(α,0.1,0.001), for $\alpha = 0.01, 0.1, \ldots, 1.0$, are more or less the same while the IULBF(0.1,0.001) is the most sparse preconditioner. The IULBFP(0.01,0.1,0.001) has the least total time among all preconditioners. From these figures we can say that all of the preconditioners IULBFP(α,0.1,0.001) for different values of α, have better quality than the IULBF(0.1,0.001) at reducing the number of iterations while the best choice of α is 0.01.

As it is mentioned in [16], the left-looking version of RIF preconditioner is in the form of $A \approx M = LDU$ and also needs to compute the upper triangular factors Z and W such that $A^{-1} \approx ZD^{-1}W^T$. For all the 165 linear systems, we have also computed this preconditioner which is coupled with complete pivoting strategy. To compute this preconditioner the drop tolerance parameters τ_w and τ_z have been set equal to 0.1 and the drop tolerance parameters τ_l and τ_u have been considered as 0.001. The complete pivoting strategy for this preconditioner also depends on a parameter α. We have set this parameter equal to α =0.01, 0.1, 0.25, 0.5, 0.75, 1.0. The preconditioning time, density, number of iterations, total time, total number of row and column pivoting performance profiles can be found in Figures 8-10. These performance profiles are computed when we define S to be the set of all preconditioners LLRIFP(α,0.1,0.001), for $\alpha = 0.01, 0.1, 0.25, 0.5, 0.75, 1.0$ and S_1 to be the set of all these preconditioners which are coupled with GMRES(30).



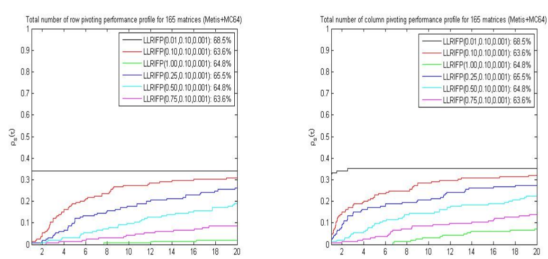
(a) (b)

FIGURE 8. (a) *Number of iterations performance profile for* LLRIFP(α,0.1,0.001). (b) *Density performance profile for* LLRIFP(α,0.1,0.001)



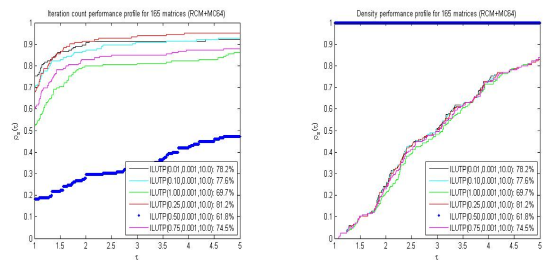
(a) (b)

FIGURE 9. (a) *Preconditioning time performance profile for* LLRIFP(α,0.1,0.001). (b) *Total time performance profile for* LLRIFP(α,0.1,0.001)



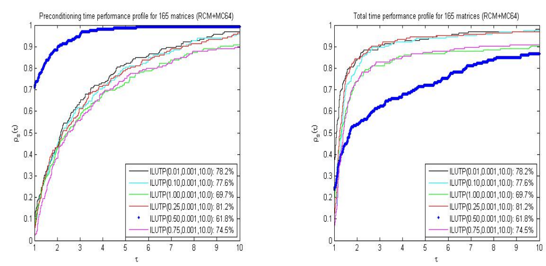
(a) (b)

FIGURE 10. (a) *Total number of row pivoting performance profile for* LLRIFP(α,0.1,0.001). (b) *Total number of column pivoting performance profile for* LLRIFP(α,0.1,0.001)



(a) (b)

FIGURE 11. (a) *Number of iterations performance profile for* ILUTP(*permtol*,0.001,10). (b) *Density performance profile for* ILUTP(*permtol*,0.001,10)



(a) (b)

FIGURE 12. (a) *Preconditioning time performance profile for* ILUTP(*permtol*,0.001,10). (b) *Total time performance profile for* ILUTP(*permtol*,0.001,10)

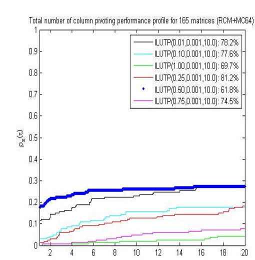
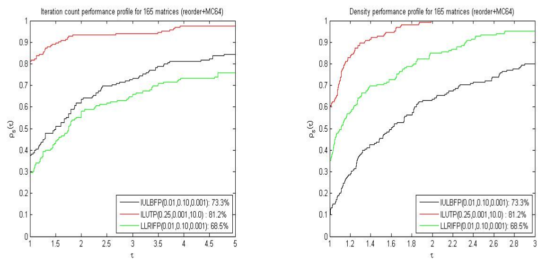


FIGURE 13. *Total number of pivoting performance profile for* ILUTP(*permtol*,0.001,10) preconditioner

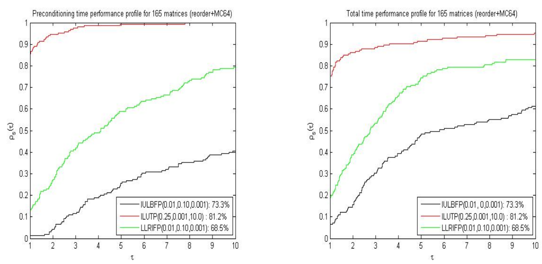
In Tables 3 and 4 and in Figures 8-10 and 14-16 the notation LLRIFP(α,0.1,0.001), for $\alpha = 0.01, 0.1, 0.25, 0.5, 0.75, 1.0$, indicates the left-looking version of RIF coupled with complete pivoting which uses $\tau_z = \tau_w = 0.1$ and $\tau_l = \tau_u = 0.001$ as the drop tolerance parameters and α as the pivoting parameter. In these figures, for each of the preconditioners we have also presented the percentage of the solved linear systems. Figure 8 shows that LLRIFP(0.01,0.1,0.001) gives the less number of iterations of the GMRES(30) method. It also indicates that the density of the preconditioners LLRIFP(α,0.1,0.001), for $\alpha = 0.01, 0.1, 0.25, 0.5, 0.75$ are nearly the same while LLRIFP(1.0,0.1,0.001) is the most dense preconditioner. One can observe in Figure 9 that there is not a great difference between the preconditioning time (total time) of all of the preconditioners LLRIFP(α,0.1,0.001), for $\alpha = 0.01, 0.1, \cdots, 1.0$. From the graphs in Figure 10 it can be concluded that the choice of $\alpha = 0.01$ generates the less total number of row and the less total number of column pivoting than the other choices of α. From the three Figures 8-10 it can be said that the choice of $\alpha = 0.01$ is the most effective value than the other choices of α for the left-looking RIF with complete pivoting.

The ILUTP preconditioner has three parameters to be set. They are τ which is the drop tolerance parameter for its L and U factors, the $lfil$ which is the total number of elements that should be kept in each row of L and U factors and the *permtol* which is the column pivoting parameter. This preconditioner only applies the column pivoting strategy. To compute this preconditioner we have selected $\tau = 0.001$, $lfil = 10$ and *permtol* equal to $0.01, 0.1, 0.25, 0.5, 0.75, 1.0$. In Tables 3 and 4 and in Figures 11-13 and 14-16, the notation ILUTP(*permtol*,0.001,10) refer to these cases. In Figures 11-13, there are the performance profile plots for the number of iterations, density, preconditioning time, total time and total number of column pivoting associated to the preconditioners ILUTP(*permtol*,0.001,10), for $permtol = 0.01, 0.1, 0.25, 0.5, 0.75, 1.0$. These plots can be obtained when S in (5.1) consists of ILUTP(*permtol*,0.001,10), for *permtol* =0.01, 0.1, 0.25, 0.5, 0.75, 1.0 and S_1 to be the set of all these preconditioners which are coupled with GMRES(30). In the legend of these figures one can also see the percentage of the solved right preconditioned systems associated to each preconditioner. From these figures one can conclude the following information. It is hard to see any great difference between the density of the preconditioners ILUTP(*permtol*,0.001,10), for $permtol = 0.01, 0.1, 0.25, 0.75, 1.0$ while the choice of $permtol = 0.5$ generates the most dense ILUTP preconditioner. The worst number of iterations and total time are due to the choice $permtol = 0.5$ while the least preconditioning time is associated to this value of *permtol*. The choice of $permtol = 0.25$ seems to give the best number of iterations of the GMRES(30) method.



(a) (b)

FIGURE 14. (a) *Number of iterations performance profile for* ILUTP, IULBFP *and* LLRIFP. (b) *Density performance profile for* ILUTP, IULBFP *and* LLRIFP



(a) (b)

FIGURE 15. (a) *Preconditioning time performance profile for* ILUTP, IULBFP *and* LLRIFP. (b) *Total time performance profile for* ILUTP, IULBFP *and* LLRIFP

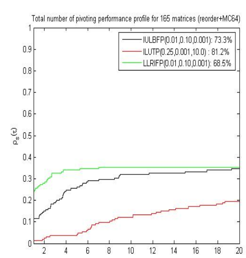


FIGURE 16. *Total number of pivoting performance profile for* ILUTP, IULBFP *and* LLRIFP

Except for the value $permtol = 0.5$, for the other choices of $permtol$, the preconditioning time of the preconditioners are more or less the same. The *permtol* = 0.5 and *permtol* = 1.0 need the least and the most total number of column pivoting, respectively.

In Figures 14 and 15 we have compared the number of iterations, density, preconditioning and total time performance profiles of the preconditioners ILUTP(0.25,0.001,10), IULBFP(0.01,0.1,0.001) and LLRIFP(0.01,0.1,0.001). We have summed the total number of row and column pivoting for IULBF with complete pivoting and for left-looking RIF with complete pivoting. This parameter is termed as the total number of pivoting associated to these preconditioners. In Figure 16, one can see the total number of pivoting performance profile for these two preconditioners and also the column pivoting performance profile for ILUTP(0.25,0.001,10). From Figures 14-16 we can consider the following results. The ILUTP(0.25,0.001,10) gives the best number of iterations of the GMRES(30) method than the other preconditioners. IULBFP(0.01,0.1,0.001) makes GMRES(30) method convergent in better number of iterations than the LLRIFP(0.01,0.1,0.001). The IULBFP(0.01,0.1,0.001) is the most dense one while ILUTP(0.25,0.001,10) is the most sparse preconditioner.
ILUTP(0.25,0.001,10) is the fastest preconditioner in terms of preconditioning time while IULBFP(0.01,0.1,0.001) is the slowest one. This situation also happens for the total time of the GMRES(30) method. The lines in Figure 16 indicate that among the three preconditioners ILUTP(0.25,0.001,10), IULBFP(0.01,0.1,0.001) and LLRIFP(0.01,0.1,0.001), the first one is computed by using the most number of total pivoting while the third one is obtained by the least number of total pivoting. The line associated to the total number of pivoting for the IULBFP(0.01,0.1,0.001) lies in between the lines associated to the two other preconditioners. In the legend of the Figures 14-16, we have also repeated the percentage of the solved preconditioned systems by each of the preconditioners. From the results of these figures we can say that among the three preconditioners ILUTP(0.25,0.001,10), IULBFP(0.01,0.1,0.001) and LLRIFP(0.01,0.1,0.001), the first one is the most effective one at reducing the number of iterations of GMRES(30) method while it needs the most total number of pivoting. Despite the fact that the quality of the IULBFP(0.01,0.1,0.001) preconditioner is not as well as the first one but it needs less total number of pivoting. Although IULBFP(0.01,0.1,0.001) is computed by using more total pivoting than LLRIFP(0.01,0.1,0.001) but it has a better quality at reducing the number of iterations of the GMRES(30) method.

For a better comparison of the four preconditioners ILUTP(0.25,0.001,10), IULBFP(0.01,0.1,0.001), LLRIFP(0.01,0.1,0.001) and IULBF(0.1,0.001), we have selected a subset of test matrices. The information of these matrices and the results of GMRES(30) method to solve the original systems can be found in Table 2. In this table, n and nnz are the dimension and the number of nonzero entries of the matrix and It and $Itime$ are the number of iterations and iteration time of the GMRES(30) method. $Itime$ is in seconds. A + sign in this

Table 2. A subset of test matrices

Matrix Name	*Matrix properties*		*GMRES(30)*	
	n	nnz	*It*	*Itime*
$af23560$	23560	484256	+	+
$atmosmodd$	1270432	8814880	808	46.85
$atmosmodj$	1270432	8814880	1615	93.27
$cage14$	1505785	27130349	19	1.61
$cavity13$	2597	76367	+	+
$cavity19$	4562	138187	+	+
$cavity20$	4562	138187	+	+
$circuit5M\_dc$	3523317	19194193	60	12.40
$Freescale1$	3428755	18920347	+	+
$hvdc2$	189860	1347273	+	+
$hcircuit$	105676	513072	+	+
$language$	399130	1216334	30	0.72
$memchip$	2707524	14810202	+	+
$ohne2$	181343	11063545	+	+
$para-4$	153226	5326228	+	+
$rajat15$	37261	443573	+	+
$rajat28$	87190	607235	+	+
$Raj1$	263743	1302464	+	+
tmt_unsym	917825	4584801	+	+
$trans4$	116835	766396	+	+
$trans5$	116835	766396	+	+
$Transport$	1602111	23500731	+	+
$venkat01$	62424	1717792	+	+
$venkat25$	62424	1717792	+	+
$venkat50$	62424	1717792	+	+
bp_1400	822	4790	+	+
bp_1600	822	4841	+	+
fs_760_2	760	5976	+	+
fs_760_3	760	5976	+	+
$gemat12$	4929	33111	+	+
lns_3937	3937	25407	+	+
$lnsp3937$	3937	25407	+	+
$sherman2$	1080	23094	+	+
$sherman4$	1104	3786	558	0.04
$sherman5$	3312	20793	+	+
$west1505$	1505	5445	+	+
$west2021$	2021	7353	+	+

table is used when the stopping criterion has not been satisfied in 5000 number of iterations. In Table 3, there are the properties of the preconditioners. In this table, *density* and $Prtime$ are the density and preconditioning time of the preconditioners. $Prtime$ is in seconds. In this table, Tot_piv is the summation of the total number of row and column pivoting. For ILUTP(0.25,0.001,10), this is only the total number of column pivoting.

In this paragraph, we discuss about the numerical results in Table 3. What we are concluding is something on average. From the results of this table we can say that in terms of preconditioning time, the ILUTP(0.25,0.001,10) is the fastest preconditioner for all the matrices while for most of the matrices, IULBF(0.1,0.001) is the slowest one. For 22 matrices LLRIFP(0.01,0.1,0.001)

TABLE 3. Properties of the preconditioners for 37 matrices

Matrix Name	ILUTP(0.25, 0.001, 10)			LLRIFP(0.01, 0.1, 0.001)			IULBFP(0.01, 0.1, 0.001)			IULBF(0.1, 0.001)	
	density	Prtime	Tot_piv	density	Prtime	Tot_piv	density	Prtime	Tot_piv	density	Prtime
af23560	0.964	0.119	304	2.383	4.883	79	2.490	1.770	15	2.093	14.925
atmosmodd	2.877	1.622	0	2.249	6.029	0	2.225	3.363	0	2.405	5.061
atmosmodj	2.877	1.613	0	2.249	6.152	0	2.225	3.442	0	2.405	5.071
cage14	1.036	9.286	0	1.061	11.797	0	1.063	4.916	0	1.149	11.398
cavity13	0.574	0.033	56	1.506	0.325	8	3.354	0.339	3	1.600	0.544
cavity19	0.571	0.037	22	1.359	0.283	1	2.656	0.300	0	2.014	1.224
cavity20	0.611	0.029	13	1.422	0.298	0	2.807	0.310	0	1.837	1.265
circuit5M_dc	0.548	0.844	6	0.698	4.710	0	0.762	4.070	0	0.716	3.578
Freescale1	0.766	1.064	410071	0.814	7.704	34858	0.940	6.239	9307	0.995	4.899
hvdc2	1.404	0.192	16766	1.318	0.788	113	2.000	0.783	416	2.197	6.521
hcircuit	1.320	0.040	2941	1.185	0.124	0	2.044	0.506	0	2.015	0.377
language	1.042	0.123	0	1.245	8.364	0	1.377	1.540	0	1.338	0.795
memchip	1.276	1.095	93820	1.146	5.320	0	1.211	6.247	0	1.232	4.070
ohne2	0.248	0.670	3906	0.233	9.379	1	0.262	2.845	23	0.280	5.496
$para-4$	0.377	0.384	14310	0.345	3.323	49	0.867	7.057	186	0.527	24.112
rajat15	0.854	0.079	2813	0.688	0.271	185	1.241	0.358	126	1.214	0.626
rajat28	1.189	0.268	411	1.028	5.670	6	1.647	13.083	42	1.562	7.351
Raj1	2.170	0.739	13180	1.286	5.718	7	1.485	5.537	135	1.509	16.080
tmt_unsym	3.995	0.780	0	2.011	2.408	0	2.010	2.374	0	2.311	2.559
trans4	0.586	3.290	4561	0.711	59.990	0	0.925	144.114	897	0.991	105.469
trans5	0.676	3.508	8596	0.735	52.093	0	5.388	131.005	24	4.990	102.824
Transport	1.356	2.844	0	0.832	12.511	0	0.832	7.323	0	1.076	12.401
venkat01	0.719	0.218	0	1.739	1.653	0	1.724	0.698	0	1.934	1.313
venkat25	0.719	0.280	0	1.945	2.436	0	1.909	0.882	0	2.160	1.747
venkat50	0.719	0.277	0	1.949	2.414	0	1.914	0.946	0	2.166	1.786
bp_1400	1.557	0.011	51	1.759	0.024	9	3.461	0.286	4	2.903	0.251
bp_1600	1.446	0.012	24	1.585	0.012	4	3.191	0.286	4	2.784	0.253
fs_760_2	1.133	0.011	123	1.037	0.016	10	1.302	0.224	59	1.845	0.255
fs_760_3	1.861	0.010	307	1.516	0.005	35	1.903	0.225	91	2.447	0.264
gemat12	1.523	0.021	536	1.371	0.013	30	2.212	0.224	64	2.868	0.377
lns_3937	2.348	0.017	241	2.309	0.129	8	4.399	0.713	5	2.569	0.448
lnsp3937	2.347	0.019	225	2.233	0.133	7	2.895	0.333	22	2.846	0.414
sherman2	0.632	0.004	23	0.664	0.007	0	0.802	0.287	0	0.658	0.247
sherman4	2.898	0.021	0	2.085	0.016	0	2.111	0.248	0	2.352	0.240
sherman5	1.343	0.010	0	1.249	0.003	0	1.701	0.242	0	1.520	0.254
west1505	1.598	0.003	17	1.481	0.002	2	3.507	0.220	2	3.080	0.254
west2021	1.744	0.014	27	1.483	0.016	7	3.177	0.227	1	2.773	0.269

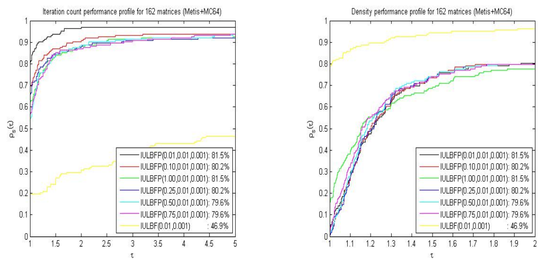
TABLE 4. Properties of the $GMRES(30)$ method to solve the right preconditioned systems for 37 matrices

$Matrix\ Name$	$ILUTP(0.25, 0.001, 10)$		$LLRIFP(0.01, 0.1, 0.001)$		$IULBFP(0.01, 0.1, 0.001)$		$IULBF(0.1, 0.001)$	
	It	$Ttime$	It	$Ttime$	It	$Ttime$	It	$Ttime$
$af23560$	2063	4.197	+	+	+	+	+	+
$atmosmodd$	78	12.698	140	31.591	186	26.960	230	32.734
$atmosmodj$	67	12.535	172	41.762	197	27.563	309	42.778
$cage14$	9	14.904	7	14.497	7	6.469	12	13.652
$cavity13$	1300	0.303	+	+	148	0.439	+	+
$cavity19$	589	0.262	+	+	154	0.458	+	+
$cavity20$	521	0.225	164	0.420	149	0.470	+	+
$circuit5M\_dc$	3	11.098	5	6.519	5	5.265	9	5.605
$Freescale1$	+	+	+	+	+	+	+	+
$hvdc2$	+	+	+	+	+	+	+	+
$hcircuit$	23	0.187	67	0.725	110	1.477	730	6.493
$language$	8	0.326	6	8.557	5	1.679	9	1.023
$memchip$	14	5.108	33	14.911	40	14.986	+	+
$ohne2$	+	+	+	+	+	+	+	+
$para-4$	+	+	+	+	+	+	+	+
$rajat15$	387	1.276	+	+	+	+	+	+
$rajat28$	16	0.366	54	6.069	19	13.215	89	7.968
$Raj1$	694	16.185	+	+	+	+	+	+
tmt_unsym	+	+	+	+	+	+	+	+
$trans4$	38	3.530	41	60.355	84	144.793	65	106.004
$trans5$	79	4.047	121	53.261	147	133.530	150	104.874
$Transport$	326	80.931	+	+	+	+	+	+
$venkat01$	25	0.410	22	2.034	20	0.996	43	1.778
$venkat25$	286	2.343	303	8.222	279	4.792	844	14.428
$venkat50$	427	3.450	527	12.742	494	7.953	1736	23.013
bp_1400	29	0.020	+	+	23	0.298	+	+
bp_1600	28	0.029	61	0.004	18	0.294	+	0.451
fs_760_2	27	0.019	+	+	93	0.244	+	+
fs_760_3	+	+	+	+	+	+	+	+
$gemat12$	+	+	+	+	+	+	+	+
lns_3937	7	0.028	341	0.284	147	0.786	+	+
$lnsp3937$	6	0.032	28	0.156	10	0.346	+	+
$sherman2$	10	0.023	20	0.023	19	0.302	27	0.263
$sherman4$	12	0.036	27	0.000	29	0.261	53	0.259
$sherman5$	19	0.019	33	0.016	28	0.262	111	0.287
$west1505$	7	0.014	60	0.020	16	0.226	+	+
$west2021$	7	0.028	+	+	16	0.238	+	+

is faster than IULBFP(0.01,0.1,0.001) in terms of preconditioning time and for the 15 other matrices this is vice versa. For most of the test matrices, the ILUTP(0.25,0.001,10) is the most sparse preconditioner. For all of the test matrices, the total number of pivoting associated to ILUTP(0.25,0.001,10) preconditioner is more than the total number of pivoting associated to the two other preconditioners. For 11 matrices, the total number of pivoting for IULBFP(0.01,0.1,0.001) is bigger than the total number of pivoting for LLRIFP(0.01,0.1,0.001) and for 8 other matrices this is vice versa. For the rest of other matrices, there is not a total number of pivoting associated to these two preconditioners or the total number of pivoting of these two preconditioners are equal. All these observation emphasize the results obtained from Figures 14-16.

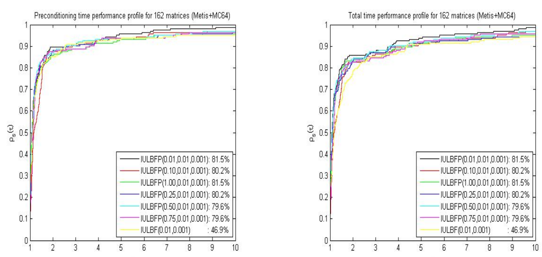
In Table 4, there are the information of GMRES(30) method to solve the right preconditioned systems. In this table, It is the iteration count and $Ttime$ is the total time which is the summation of preconditioning time and the iteration time. This parameter is also in seconds. A + sign in this table, indicates that the stopping criterion has not been satisfied in 2500 number of iterations. From this table we can say that for most of the test matrices, ILUTP(0.25,0.001,10) gives better number of iterations of GMRES(30) method than the two other preconditioners. The results in this table show that for 17 matrices, the IULBFP(0.01,0.1,0.001) makes the GMRES(30) method convergent in less number of iterations than the LLRIFP(0.01,0.1,0.001) and for 7 other matrices this is vice versa. For the rest of other matrices, these two preconditioners can not make the GMRES(30) method convergent or the number of iterations associated to these two preconditioners are equal. By comparing the data in the columns IULBFP(0.01,0.1,0.001) and IULBF(0.1,0.001) we can see that for almost all of the matrices, the number of iterations of the IULBFP(0.01,0.1,0.001) is much better than the number of iterations of the IULBF(0.1,0.001) preconditioner. If we summarize our consideration from Table 4, we can say that on average, the quality of the IULBFP(0.01,0.1,0.001) preconditioner is way better than the quality of the LLRIFP(0.01,0.1,0.001) and IULBF(0.1,0.001) preconditioners but not as well as the quality of the ILUTP(0.25,0.001,10) preconditioner. This was also a consideration we could get from Figure 14.

5.2. **Second part of experiments.** In this part of the numerical experiments, we have set $\tau_z = \tau_w = 0.01$ and $\tau_l = \tau_u = 0.001$ and $\alpha =$0.01, 0.1, 0.25, 0.5, 0.75, 1.0 for the IULBF, IULBF with complete pivoting and for the left-looking RIF with complete pivoting. For the ILUTP, τ has been set to 0.001 and $lfil = 15$ and $permtol$ will be 0.01, 0.1, 0.25, 0.5, 0.75, 1.0. In Figures 17-19 and in Tables 5 and 6, the notations IULBF(0.01,0.001), IULBFP(α,0.01,0.001), LLRIFP(α,0.01,0.001) and ILUTP($permtol$,0.001,15) refer to these cases.



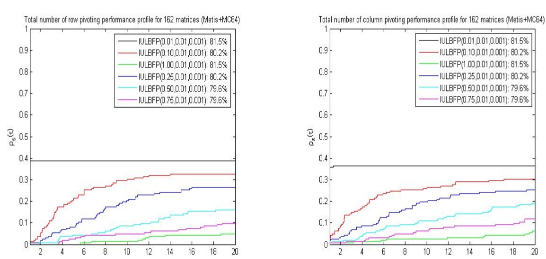
(a) (b)

FIGURE 17. (a) *Number of iterations performance profile for* IULBFP(α,0.01,0.001) *and* IULBF(0.01,0.001). (b) *Density performance profile for*IULBFP(α,0.01,0.001) *and* IULBF(0.01,0.001)



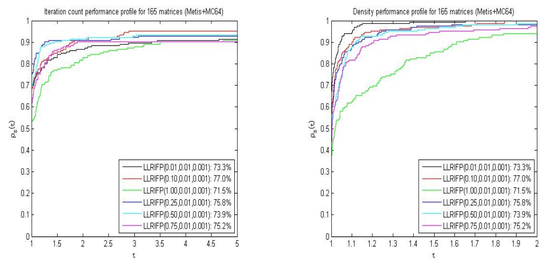
(a) (b)

FIGURE 18. (a) *Preconditioning time performance profile for* IULBFP(α,0.01,0.001) *and* IULBF(0.01,0.001). (b) *Total time performance profile for* IULBFP(α,0.01,0.001) *and* IULBF(0.01,0.001)



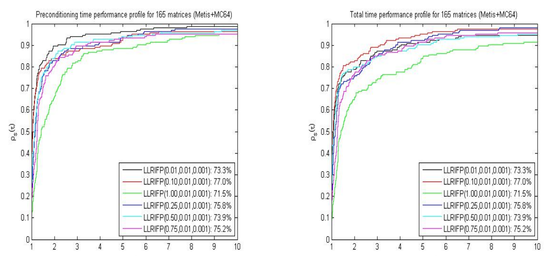
(a) (b)

FIGURE 19. (a) *Total number of row pivoting performance profile for* IULBFP(α,0.01,0.001). (b) *Total number of column pivoting performance profile for* IULBFP(α,0.01,0.001)



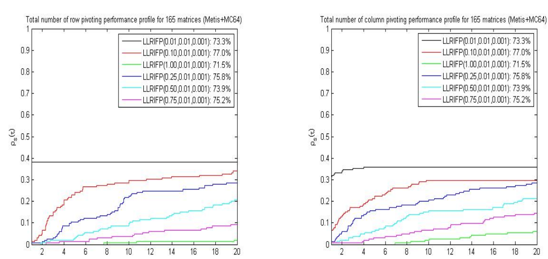
(a) (b)

FIGURE 20. (a) *Number of iterations performance profile for* LLRIFP(α,0.01,0.001). (b) *Density performance profile for* LLRIFP(α,0.01,0.001)



(a) (b)

FIGURE 21. (a) *Preconditioning time performance profile for* LLRIFP(α,0.01,0.001). (b) *Total time performance profile for* LLRIFP(α,0.01,0.001)



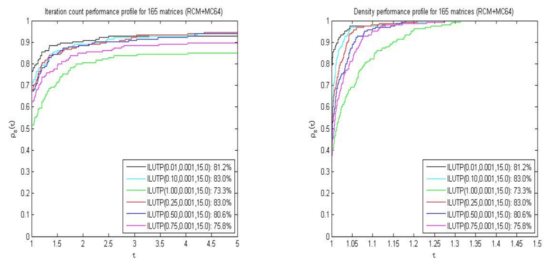
(a) (b)

FIGURE 22. (a) *Total number of row pivoting performance profile for* LLRIFP(α,0.01,0.001). (b) *Total number of column pivoting performance profile for* LLRIFP(α,0.01,0.001)

All these preconditioners have been applied as the right preconditioner for linear systems and then, the preconditioned systems were solved by the GMRES(30) method. In Figures 17-19, the performance profile plots associated to the preconditioners IULBF(0.01,0.001) and IULBFP(α,0.01,0.001) for $\alpha = 0.01$, 0.1, $\cdots$, 1.0 have been compared. For the three matrices $Freescale1$, *memchip* and $rajat21$, it was not possible to compute the IULBF(0.01,0.001) preconditioner. Therefore, all the performance profile figures are due to the numerical tests on 162 matrices. In the legend of these figures, we have written the percentage of the solved right preconditioned systems. Figure 17 indicates that the best preconditioner is IULBFP(0.01,0.01,0.001) at reducing the number of iterations of the GMRES(30) method while the worst one is IULBF(0.01,0.001). This figure also shows that the most sparse preconditioner is IULBF(0.01,0.001) and the other preconditioners have nearly the same density. From Figure 18, we can not say anything special about the preconditioning time (total time). In Figure 19, one can see that the least total number of row and column pivoting are due to the IULBFP(0.01,0.01,0.001). From Figures 17-19, we can claim that the choice of $\alpha = 0.01$ gives better preconditioner than the other choices of α.

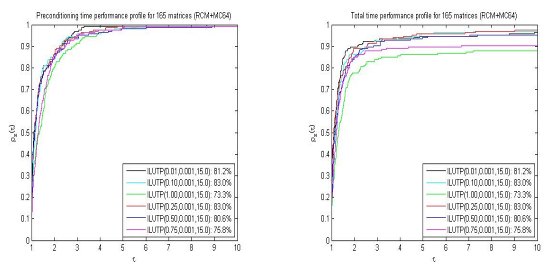
In Figures 20-22 and for all the 165 linear systems, we have drawn the performance profile graphs of the preconditioners LLRIFP(α,0.01,0.001) for α =0.01, 0.1, …, 1.0. The percentage of the solved systems have also been reported. The (a) part of Figure 20, shows that LLRIFP(0.1,0.01,0.001) has the least number of iterations of the GMRES(30) method than the other preconditioners. The (b) part of this figure indicates that the most and the least dense preconditioners are LLRIFP(0.01,0.01,0.001) and LLRIFP(1.0,0.01,0.001), respectively. From Figure 21, we can see that the preconditioners LLRIFP(0.01,0.01,0.001) and LLRIFP(1.0,0.01,0.001) are the fastest and the slowest preconditioners, respectively in terms of preconditioning time while the second preconditioner also has the highest total time. The (b) part of this figure shows that the total time of the LLRIFP(0.1,0.01,0.001) preconditioner is less than the total time of the other preconditioners. With respect to the percentage of the solved systems, number of iterations and total time, we can conclude from Figures 20-22 that the choice of $\alpha = 0.1$ gives better results of the left-looking RIF with complete pivoting.

Figures 23-26 are due to the performance profile lines of the preconditioners ILUTP(*permtol*,0.001,15) for *permtol* =0.01, 0.1, …, 1.0. In these figures, we have also presented the percentage of the solved systems by each preconditioner. From the (a) part of Figure 23 and with respect to the percentage of the solved systems, it is really hard to select the best preconditioner among the three preconditioners ILUTP(0.01,0.001,15), ILUTP(0.1,0.001,15) and ILUTP(0.25,0.001,15) in terms of the number of iterations of the GMRES(30) method. We have considered a parameter *count* for each of these three preconditioners.



(a) (b)

FIGURE 23. (a) *Number of iterations performance profile for* ILUTP (*permtol*,0.001,15). (b) *Density performance profile for* ILUTP (*permtol*,0.001,15)



(a) (b)

FIGURE 24. (a) *Preconditioning time performance profile for* ILUTP(*permtol*,0.001,15). (b) *Total time performance profile for* ILUTP(*permtol*,0.001,15)

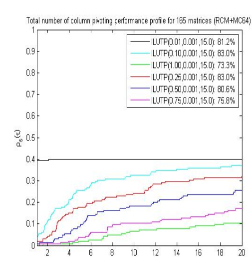
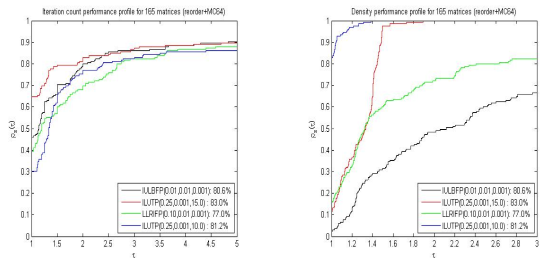
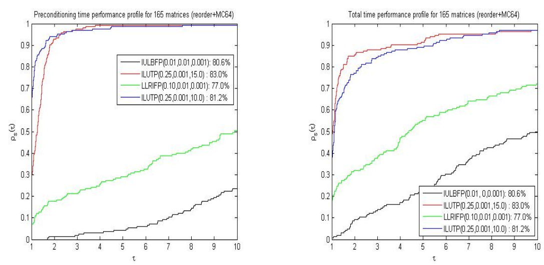


FIGURE 25. *Total number of pivoting performance profile for* ILUTP (*permtol*,0.001,15) preconditioner



(a) (b)

FIGURE 26. (a) *Number of iterations performance profile for* ILUTP, IULBFP *and* LLRIFP. (b) *Density performance profile for* ILUTP, IULBFP *and* LLRIFP



(a) (b)

FIGURE 27. (a) *Preconditioning time performance profile for* ILUTP, IULBFP *and* LLRIFP. (b) *Total time performance profile for* ILUTP, IULBFP *and* LLRIFP

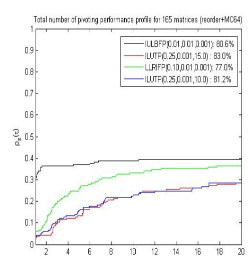


FIGURE 28. *Total number of pivoting performance profile for* ILUTP, IULBFP *and* LLRIFP

If for a system, the number of iterations of for example ILUTP(0.25,0.001,15) is less than the number of iterations of the two other preconditioners, then

we have incremented the *count* of ILUTP(0.25,0.001,15) by one. For the two other preconditioners ILUTP(0.01,0.001,15) and ILUTP(0.1,0.001,15) we have done the same and we have computed their *count* parameter. The *count* of ILUTP(0.25,0.001,15), ILUTP(0.1,0.001,15) and ILUTP(0.01,0.001,15) were 23, 5 and 19, respectively. This indicates that the best preconditioner is ILUTP(0.25,0.001,15) at reducing the number of iterations of the GMRES(30) method.
From the (b) part of Figure 23 we can only say that the choice of $permtol = 1.0$ generates the most dense preconditioner. As it is clear from Figure 24, all of the preconditioners ILUTP(*permtol*,0.001,15) for *permtol* =0.01, 0.1, $\cdots$, 1.0, have more or less the same preconditioning time while ILUTP(1.0,0.001,15) needs the highest toltal time. From the graphs in Figure 25 we see that ILUTP(0.01,0.001,15) and ILUTP(1.0,0.001,15) are computed by using the most and the least total number of column pivoting. All these observation define the $permtol = 0.25$ as the best parameter for ILUTP preconditioner.

In Figures 26-28, we have compared the number of iterations, density, preconditioning time, total time and total number of pivoting performance profile lines of the preconditioners ILUTP(0.25,0.001,10), ILUTP(0.25,0.001,15), IULBFP(0.01,0.01,0.001) and LLRIFP(0.1,0.01,0.001). The (a) part of Figure 26 shows that the IULBFP(0.01,0.01,0.001) makes the GMRES(30) convergent in a better number of iterations than the ILUTP(0.25,0.001,10) and LLRIFP(0.1,0.01,0.001) preconditioners. From this part of the figure we can see that the number of iterations of the two preconditioners IULBFP(0.01,0.01,0.001) and ILUTP(0.25,0.001,15) are comparable but we can not claim which one is a better preconditioner, although the percentage of the solved systems by the first preconditioner is less than the percentage of the solved systems by the second one.
From the (b) part of this figure it is obvious that IULBFP(0.01,0.01,0.001) is the most dense preconditioner while both of preconditioners ILUTP(0.25,0.001,10) and ILUTP(0.25,0.001,15) are the most sparse ones. Figure 27 indicates that the IULBFP(0.01,0.01,0.001) is computed by the highest preconditioning time and it solves the systems by the highest total time than the other preconditioners. Both of the preconditioners ILUTP(0.25,0.001.10) and ILUTP(0.25,0.001,15) seem to have the least preconditioning and total time. In Figure 28, the lines associated to each of the four preconditioners say that the IULBFP(0.01,0.01,0.001) preconditioner is computed by using the least total pivoting while the preconditioners ILUTP(0.25,0.001,10) and IULTP(0.25,0.001,15) need the most total pivoting than the other preconditioners.

In Table 5, we have presented the density and preconditioning time of ILUTP(0.25,0.001,15), IULBFP(0.01,0.01,0.001), LLRIFP(0.1,0.01,0.001) and IULBF(0.01,0.001) for 35 of the test linear systems. For the first three preconditioners, we have also reported the total number of pivoting in this table. In

Table 6, the number of iterations of the GMRES(30) method and the total time for each of these preconditioners have been reported. The notations *density*, *Prtime*, *Tot_piv*, *It* and *Ttime* in these two tables have the same definition as in Tables 3 and 4.

We can say the following from the results of Table 5. For almost all of the 35 matrices, the preconditioning time of the ILUTP(0.25,0.001,15) is less than the preconditioning time of the other preconditioners. For most of the test matrices, the density of the IULBFP(0.01,0.01,0.001) preconditioner is bigger than the density of the three other preconditioners. For 15 matrices, the preconditioning time of this preconditioner is less than the preconditioning time of the LLRIFP(0.1,0.01,0.001) while for the other test matrices this is vice versa. For 24 matrices, the total pivoting of the ILUTP(0.25,0.001,15) is bigger than the total pivoting of the IULBFP(0.01,0.01,0.001) and LLRIFP((0.1,0.01,0.001) preconditioners. For the rest of 11 other matrices, all these three preconditioners are computed without any pivoting.

What we can observe from the information of Table 6 is presented here. For 16 matrices, the number of iterations of the ILUTP(0.25,0.001,15) is less than the number of iterations of the IULBFP(0.01,0.01,0.001) while for 12 other matrices this is vice versa. For the rest of 7 other matrices, both preconditioners have the same effect on the number of iterations of the GMRES(30) method. The data in this table show that for 15 matrices the IULBFP(0.01,0.01,0.001) has a better effect than LLRIFP(0.1,0.01,0.001) at reducing the number of iterations of GMRES(30) method while for 8 other matrices, the second preconditioner gives better number of iterations than the first one. For the rest of 12 other matrices, both preconditioners have the same effect on the number of iterations of the GMRES(30) method.

If we compare the data associated to the IULBFP(0.01,0.01,0.001) in Table 6 by the information in the column ILUTP(0.25,0.001,10) in Table 4, we see that for 19 matrices, the number of iterations of the IULBFP(0.01,0.01,0.001) is less than the number of iterations of the ILUTP(0.25,0.001,10) while for 8 other matrices, the second preconditioner makes the GMRES(30) method convergent in less number of iterations than the first one. From the results of these two tables, we can also verify that for the rest of 8 other matrices, both of these two preconditioners behave the same on the number of iteartions of the GMRES(30) method.

If we summarize our consideration from Figures 17-28 and by analyzing the data in Tables 4, 5 and 6, it can be concluded that the quality of the IULBFP(0.01,0.01,0.001) preconditioner is better than the quality of preconditioners ILUTP(0.25,0.001,10), LLRIFP(0.1,0.01,0.001) and IULBF(0.01,0.001) at reducing the number of iterations of the GMRES(30) method. We should also mention that ILUTP(0.25,0.001,15) is somewhat better than the preconditioner IULBFP(0.01,0.01,0.001).

TABLE 5. Properties of the preconditioners for 35 matrices

Matrix Name	$ILUTP(0.25, 0.001, 15)$			$LLRIFP(0.1, 0.01, 0.001)$			$IULBFP(0.01, 0.01, 0.001)$			$IULBF(0.01, 0.001)$	
	density	*Prtime*	*Tot_piv*	*density*	*Prtime*	*Tot_piv*	*density*	*Prtime*	*Tot_piv*	*density*	*Prtime*
af23560	1.431	0.159	170	4.068	8.822	88	4.231	4.263	13	2.525	17.716
atmosmodd	4.299	2.487	0	3.828	22.232	0	3.588	11.212	0	3.263	16.815
atmosmodj	4.299	2.493	0	3.820	22.175	0	3.590	11.282	0	3.266	16.762
cage14	1.438	11.709	0	2.296	58.460	0	2.425	26.964	0	1.581	39.535
cavity13	0.807	0.031	39	1.986	1.000	13	6.608	1.883	2	1.855	0.724
cavity19	0.803	0.048	19	2.106	1.832	2	5.641	1.734	0	2.503	1.838
cavity20	0.859	0.043	21	2.181	1.996	4	6.019	1.785	0	1.876	1.288
circuit5M_dc	0.549	0.546	6	0.723	4.287	0	0.819	4.976	0	0.788	4.422
hvdc2	1.631	0.203	17216	1.566	2.242	3286	3.477	2.226	536	2.974	46.654
hcircuit	1.361	0.035	2948	1.258	0.227	135	3.183	0.804	0	2.673	0.751
language	1.077	0.112	0	1.276	26.526	0	1.384	1.691	0	1.362	0.961
ohne2	0.342	0.786	3889	0.423	38.327	1257	0.460	11.673	17	1.432	6121.329
para − 4	0.493	0.434	14372	0.576	41.419	2966	1.060	86.264	97	0.718	67.487
rajat15	1.001	0.068	2865	0.938	1.053	1134	1.624	0.815	135	18.719	147.199
rajat28	1.420	0.475	423	1.149	6.493	72	2.434	25.515	39	1.853	14.492
Raj1	2.717	0.731	14076	1.569	11.878	1493	1.952	20.674	154	10.585	215.588
tmt_unsym	5.772	1.204	0	3.150	9.407	0	3.050	5.977	0	2.996	4.901
trans4	0.636	3.429	4543	0.713	59.507	2	1.008	196.423	897	1.109	132.826
trans5	0.743	3.267	8865	0.741	58.230	587	6.056	202.573	22	5.154	168.001
Transport	1.992	5.056	0	1.661	60.064	0	1.636	26.806	0	1.541	33.681
venkat01	1.072	0.315	0	3.048	8.093	0	3.167	4.269	0	2.497	3.326
venkat25	1.073	0.369	0	3.674	16.737	0	4.082	9.647	0	2.787	5.161
venkat50	1.073	0.388	0	3.689	17.139	0	4.107	7.816	0	2.786	5.170
bp_1400	2.011	0.015	43	1.930	0.017	11	5.363	0.246	3	3.230	0.236
bp_1600	1.766	0.008	21	1.746	0.005	6	4.353	0.240	2	3.622	0.230
fs_760_2	1.315	0.009	148	1.166	0.010	57	1.634	0.221	62	2.317	0.240
fs_760_3	2.541	0.015	356	1.961	0.014	123	3.075	0.251	180	3.362	0.249
gemat12	1.740	0.018	514	1.620	0.047	208	3.889	0.341	74	3.419	0.425
lns_3937	3.278	0.021	213	3.454	0.343	34	7.755	1.774	4	2.966	0.460
lnsp3937	3.295	0.021	222	3.584	0.436	56	4.731	0.723	21	3.149	0.469
sherman2	0.814	0.015	29	0.793	0.018	12	0.967	0.239	0	0.823	0.241
sherman4	4.022	0.015	0	3.107	0.012	0	3.313	0.219	0	3.095	0.226
sherman5	1.843	0.023	0	1.909	0.042	0	3.092	0.242	0	2.372	0.260
west1505	1.681	0.011	17	1.565	0.012	12	4.074	0.225	6	3.833	0.245
west2021	1.833	0.013	30	1.569	0.022	26	3.598	0.235	5	3.036	0.240

TABLE 6. Properties of the $GMRES(30)$ method to solve the right preconditioned systems for 35 matrices

Matrix Name	ILUTP(0.25, 0.001, 15)		LLRIFP(0.1, 0.01, 0.001)		IULBFP(0.01, 0.01, 0.001)		IULBF(0.01, 0.001)	
	It	Ttime	It	Ttime	It	Ttime	It	Ttime
af23560	1295	2.743	+	+	66	4.604	+	+
atmosmodd	53	11.704	109	39.405	105	30.818	216	56.756
atmosmodj	50	10.506	111	41.001	118	30.678	331	79.328
cage14	7	13.358	6	60.464	6	29.433	9	42.043
cavity13	262	0.103	29	1.023	18	1.909	+	+
cavity19	242	0.144	23	1.852	26	1.779	+	+
cavity20	171	0.114	24	1.999	25	1.831	+	+
circuit5M_dc	3	1.331	3	5.200	3	5.845	13	7.688
hvdc2	+	+	+	+	+	+	+	+
hcircuit	17	0.129	21	0.371	30	1.099	1982	18.519
language	8	0.291	4	26.656	4	1.820	8	1.194
ohne2	341	11.918	+	+	+	+	+	+
para − 4	336	7.173	+	+	+	+	+	+
rajat15	255	0.801	+	+	+	+	+	+
rajat28	15	0.554	40	6.734	11	25.608	115	15.611
Raj1	+	+	+	+	+	+	+	+
tmt_unsym	+	+	+	+	+	+	+	+
trans4	25	3.584	6	59.538	40	196.753	150	134.252
trans5	31	3.495	10	58.298	36	203.136	+	+
Transport	183	48.860	989	279.868	919	364.955	+	+
venkat01	15	0.429	13	8.465	13	4.598	37	3.833
venkat25	119	1.299	100	20.162	90	12.182	994	19.641
venkat50	179	1.902	159	21.512	135	11.007	1964	33.775
bp_1400	24	0.022	19	0.006	8	0.252	+	+
bp_1600	16	0.013	9	0.004	8	0.250	+	+
fs_760_2	15	0.013	308	0.009	22	0.233	+	+
fs_760_3	+	+	+	+	+	+	+	+
gemat12	+	+	+	+	+	+	+	+
lns_3937	4	0.025	11	0.365	6	1.789	+	+
lnsp3937	5	0.035	17	0.444	5	0.729	+	+
sherman2	7	0.025	9	0.027	8	0.243	48	0.263
sherman4	9	0.033	15	0.007	17	0.239	87	0.245
sherman5	14	0.040	16	0.064	17	0.263	240	0.324
west1505	4	0.031	11	0.003	7	0.234	+	+
west2021	4	0.018	10	0.016	6	0.250	+	+

6. Conclusion

In this paper, we presented a complete pivoting strategy for the IUL preconditioner obtained as the by-product of the backward factored approximate inverse process. This preconditioner is termed as IULBFP. The pivoting process for this preconditioner depends on a parameter α. We have used the values 0.01, 0.1, 0.25, 0.5, 0.75 and 1.0 as α and then have applied the computed IULBFP as the right preconditioner for linear systems. The preconditioned systems have been solved by the GMRES(30) method. As the preprocessing, the multilevel nested dissection reordering has been coupled with the maximum weighted matching. The numerical results show that when we use different drop tolerance parameters to compute this preconditioner, the choice of $\alpha = 0.01$ gives better results at reducing the number of iterations while it needs the less total number of pivoting than the other choices of α. We have also prepared the same atmosphere for the left-looking version of RIF with complete pivoting to know if we can have the best value of α. The results show that the choices $\alpha = 0.01$ and $\alpha = 0.1$ are the most effective ones for this preconditioner when the multilevel nested dissection reordering and the maximum weighted matching are used as the preprocessing.

In the numerical experiments we have also used the ILUTP which is coupled with the RCM reordering and maximum weighted matching. This preconditioner has also been applied as the right preconditioner for linear systems and then the preconditioned systems have been solved by GMRES(30) method. For this preconditioner, we have fixed the drop tolerance parameter and have played around with the number of fill-in entries in L and U factors and have applied the same pivoting parameters as IULBF with complete pivoting and left-looking RIF with complete pivoting. The results show that the pivoting parameter 0.25 is the best option for this preconditioner.

As part of the numerical experiments, we have also compared the three preconditioners ILUTP, IULBF with complete pivoting and left-looking RIF with complete pivoting. For each of these preconditioners its associated best value of pivoting parameter has been used. The results show that by tuning the drop tolerance parameters, the quality of the IULBF with complete pivoting can be comparable to the quality of ILUTP at reducing the number of iterations of the GMRES(30) method. But this is not true for left-looking RIF with complete pivoting. The preconditioning time, total time and the density of ILUTP is way better than these parameters associated to the IULBF with complete pivoting and left-looking RIF coupled with complete pivoting. From the numerical tests, we could find that the ILUTP needs more total number of pivoting than the two other preconditioners. In terms of number of iterations, IULBF with complete pivoting seems to be better than the left-looking RIF with complete pivoting while it applies more total pivoting. The numerical results of this paper, also show that IULBF with complete pivoting is much more robust than plain IULBF at reducing the number of iterations of the GMRES(30) method.

Acknowledgements

The authors are really thankful for the comments of the two referees which really improved the paper. The numerical experiments section of the paper was done when the first author was on a short sabbatical leave in Technische Universität Braunschweig. This stay was financially supported by the Niels Henrik Abel Board in Berlin.

The first author is so thankful to Tanja Schenk in the institute of computational mathematics of Technische Universität Braunschweig. His special thanks goes to Kathrin Huter in the international office of this university for her great favors. At the end, he really thanks Lena Koch in the International Mathematical Union, CDC section in Berlin.

References

[1] M. Benzi and M. Tůma, A sparse approximate inverse preconditioner for nonsymmetric linear systems, *SIAM J. Sci. Comput.* **19** (1998), no. 3, 968–994.

[2] M. Benzi and M. Tůma, A robust incomplete factorization preconditioner for positive definite matrices, *Numer. Linear Algebra Appl.* **10** (2003), no. 5-6, 385–400.

[3] M. Bollhöfer, ILUPACK software package, http://www.icm.tu-bs.de/∼bolle/ilupack. Accessed 2015.

[4] T. Davis, University of Florida Sparse Matrix Collection, http://www.cise.ufl.edu/research/sparse/matrices. Accessed 2015.

[5] E.D. Dolan and J.J. More, Benchmarking optimization software with performance profile, *Math. Program. Ser. A* **91** (2002) 201–213.

[6] I.S. Duff and J. Koster, The design and use of algorithms for permuting large entries to the diagonal of sparse matrices, *SIAM J. Matrix Anal. Appl.* **20** (1999), no. 4, 889–901.

[7] F.R. Fazel, Computing the IUL factorization coupled by the complete pivoting as the by-product of BFAPINV process, M.Sc. Dissertation, Department of Mathematics and Computer Sciences, Hakim Sabzevari University, in Persian, 2015.

[8] J.A. George and J.W.H. Liu, Computer Solution of Large Sparse Positive Definite Systems, Prentice-Hall, 1981.

[9] T. Huckle, Personal communications , 2016.

[10] G. Karypis and V. Kumar, METIS, a Software Package for Partitioning Unstructured Graphs and Computing Fill-Reduced Orderings of Sparse Matrices, http://glaros.dtc.umn.edu/gkhome/views/metis. Accessed 2015.

[11] J.G. Luo, A new class of decomposition for inverting asymmetric and indefinite matrices, *Comput. Math. Appl.* **25**, no. 4, (1993) 95–104.

[12] A. Rafiei, A complete pivoting strategy for the right-looking Robust Incomplete Factorization preconditioner, *Comput. Math. Appl.* **64** (2012), no. 8, 2682–2694.

[13] A. Rafiei, ILU and IUL factorizations obtained from Forward and Backward Factored APproximate Inverse algorithms, *Bull. Iranian Math. Soc.* **40** (2014), no. 5, 1327–1346.

[14] A. Rafiei, Left-looking version of AINV preconditioner with complete pivoting strategy, *Linear Algebra Appl.* **445** (2014) 103–126.

[15] A. Rafiei and M. Bollhöfer, Extension of inverse-based dropping techniques for ILU preconditioners, http://www.digibib.tu-bs.de/?docid=00035624.

[16] A. Rafiei, B. Tolue and M. Bollhöfer, Complete pivoting strategy for the left-looking Robust Incomplete Factorization preconditioner, *Comput. Math. Appl.* **67** (2014), no. 11, 2055–2070.
[17] Y. Saad, Iterative Methods for Sparse Linear Systems, PWS Publishing, New York, 1996.
[18] Y. Saad, Sparskit and sparse examples, http://www-users.cs.umn.edu/∼saad/software. Accessed 2015.
[19] D.K. Salkuyeh, A. Rafiei and H. Roohani, ILU preconditioning based on the FAPINV algorithm, *Opuscula Math.* **35** (2015), no. 2, 235–250.
[20] D.K. Salkuyeh and H. Roohani, On the relation between the AINV and the FAPINV algorithms, *Int. J. Math. Math. Sci.* **2009** (2009), Article ID 179481, 6 pages.
[21] The HSL Mathematical Software Library, http://www.hsl.rl.ac.uk. Accessed 2015.
[22] J. Zhang, A procedure for computing factored approximate inverse, M.Sc. Dissertation, Department of Computer Science, University of Kentucky, 1999.
[23] J. Zhang, A sparse approximate inverse preconditioner for parallel preconditioning of general sparse matrices, *Appl. Math. Comput.* **130** (2002), no. 1, 63–85.

(Amin Rafiei) Department of Applied Mathematics, Hakim Sabzevari University, Sabzevar, Iran.
E-mail address: rafiei.am@gmail.com, a.rafiei@hsu.ac.ir, amin.rafiei@tu-bs.de.

(Matthias Bollhöfer) Institute computational Mathematics, Technische Universität Braunschweig, D-38106 Braunschweig, Germany.
E-mail address: m.bollhoefer@tu-bs.de.