Bull. Iranian Math. Soc.
Vol. 40 (2014), No. 5, pp. 1327–1346
Online ISSN: 1735-8515

ILU AND IUL FACTORIZATIONS OBTAINED FROM FORWARD AND BACKWARD FACTORED APPROXIMATE INVERSE ALGORITHMS

A. RAFIEI

(Communicated by Nezam Mahdavi-Amiri)

Abstract. In this paper, an efficient dropping criterion has been used to compute the IUL factorization obtained from Backward Factored APproximate INVerse (BFAPINV) and ILU factorization obtained from Forward Factored APproximate INVerse (FFAPINV) algorithms. We use different drop tolerance parameters to compute the preconditioners. To study the effect of such a dropping on the quality of the ILU and IUL factorizations, we have used the preconditioners as the right preconditioners for several linear systems and then, the Krylov subspace methods have been used to solve the preconditioned systems. To avoid storing matrix A in two CSR and CSC formats, the linked lists trick has been used in the implementations. As the preprocessing, the multilevel nested dissection reordering has also been used.
Keywords: ILU factorization, IUL factorization, forward $FAPINV$ process, backward $FAPINV$ process, linked lists trick.
MSC(2010): Primary: 65F08, 65F10; Secondary: 65F50.

1. Introduction

Suppose that a matrix A is nonsymmetric. Also, suppose that $W = [w_1^T, \cdots, w_n^T]^T$ and $Z = [z_1, \cdots, z_n]$ are unit upper and lower triangular matrices, respectively and $D = diag(d_1, \cdots, d_n)$ is a diagonal matrix. If the matrices W, Z, D and A satisfy the relation

$$WAZ = D, \tag{1.1}$$

Article electronically published on October 27, 2014.
Received: 12 June 2012, Accepted: 2 October 2013.

then, matrices W, Z and D are the inverse factors of A. In [5], a procedure to compute the inverse factorization (1.1) has been presented which is termed the Backward Factored INVerse or BFINV process. If the entries of W and Z are dropped in this process, then the approximate inverse factorization

$$WAZ \approx D, \tag{1.2}$$

will be computed and the process will be termed Backward Factored APproximate INVerse or BFAPINV. Matrices W, Z and D obtained from this process are the approximate inverse factors of matrix A. In each step of BFAPINV process, a row of W and a column of Z are computed. Since at step j of this process, the $(n-j+1)$-st row and column of W and Z are computed, respectively, it is called a backward process.

Suppose that matrices W and Z are unit lower and upper triangular matrices and D is still diagonal. Also, suppose that the relation (1.1) still holds. In [10], another process which is termed the Forward Factored INVerse or FFINV has also been proposed to compute the inverse factors W, Z and D of a matrix A. If in this process the entries of W and Z are dropped, then the approximate inverse factorization (1.2) will be computed and the process will be termed the Forward Factored APproximate INVerse or FFAPINV. This process also computes W row wise and Z column wise. At step j of this process, the j-th row and column of W and Z are computed, respectively. This is why it is called a Forward process.

Existence of approximate inverse factors which are obtained from FFAPINV process has been studied in [7, 8] for M-matrices, $H-$matrices and also for positive definite matrices. All the observations, can also be extended for the existence of the approximate inverse factors which are obtained from BFAPINV process.

In [8], an ILU factorization of matrix A, which is obtained as by-product of FFAPINV process, has been presented in which L is a unit lower triangular and U is an upper triangular matrix. Matrices L, U and A satisfy the relation

$$A \approx LU.$$

We term this ILU factorization, ILUFF (ILU factorization obtained from Forward Factored APproximate INverse). The approximate INVerse factors W, Z and D and also matrices L and U satisfy the following relation

$$L \approx W^{-1}, \qquad U \approx DZ^{-1}.$$

It is also possible to obtain an IUL factorizatin of matrix A, as by-product of BFAPINV process such that

$$A \approx UL.$$

In this case, L is a lower triangular and U is a unit upper triangular matrix. We term this IUL factorizatin as IULBF (IUL factorization obtained from Backward Factored APproximate INVerse). The approximate INVerse factors W, Z and D and also matrices U and L satisfy the following relation

$$U \approx W^{-1}, \qquad L \approx DZ^{-1}.$$

Consider the linear system of equation of the form

$$Ax = b,$$

where the coefficient matrix $A \in \mathbb{R}^{n\times n}$ is nonsingular, large, sparse and nonsymmetric with $x, b \in \mathbb{R}^n$. An implicit preconditioner M for the above system is a matrix $M \approx A$. If the Krylov subspace methods [9] can not solve such a system in a proper number of iterations and if M is a good approximation of A, then it is better to solve the right preconditioned linear system

$$AM^{-1}u = b; \qquad M^{-1}u = x,$$

by the Krylov subspace methods. Both ILUFF and IULBF factorizations are examples of implicit preconditioners.

A crucial challenge for the ILU preconditioners is how to apply the dropping strategy. For the first time in [1, 2], Bollhöfer presented a safe dropping strategy for this type of preconditioners and he termed it the INVerse-based dropping strategy. In this paper, a type of INVerse-based dropping strategy for both ILUFF and IULBF preconditioners will be proposed. To study the effectiveness of such a strategy, we have generated some linear systems with the coefficient matrices taken from [3]. Then, both ILUFF and IULBF preconditioners have been computed by using this type of dropping strategy and we have used these two preconditioners as the right preconditioner for the systems. After that, two Krylov subspace methods Bicgstab and GMRES(30) [9] have been applied to solve the right preconditioned linear systems.

This paper is organized as follows. In section two, we first review the ILUFF preconditioner and then, in Proposition 2.3, an INVerse-based dropping strategy for this preconditioner will be presented. In section three, at first, the IULBF preconditioner is introduced and then, at the end of this section, Proposition 3.3 will give an INVerse-based dropping strategy for this preconditioner. In section four, numerical experiments will be presented.

In this paper, notations $A_{:,j}$ and $A_{j,:}$ are used for the j-th column and the j-th row of the matrix A, respectively.

2. **Forward factored INVerse process**

The following algorithm is the FFINV algorithm [7, 10] which computes the exact factorization (1.1). If we drop the entries of the vectors z_j and w_j in each step j, then at the end of this algorithm, the approximate factorization (1.2) will be computed instead. In this case, the algorithm is called FFAPINV algorithm.

Algorithm 1 (FFINV algorithm)

```
w_1 = e_1^T, z_1 = e_1, d_1 = a_11.
for j = 2 to n do
    w_j = e_j^T, z_j = e_j.
    for i = 1 to j − 1 do
        β_ji = (A_{j,:} z_i)/d_i   α_ij = (w_i A_{:,j})/d_i
        z_j = z_j − α_ij z_i, w_j = w_j − β_ji w_i
    end for
    d_j = w_j A_{:,j} {not positive definite}
    d_j = w_j A w_j^T {positive definite}
end for
Return W = [w_1^T, ..., w_n^T]^T, D = diag(d_i)_{1≤i≤n} and Z = [z_1, ..., z_n].
```

Suppose that a matrix A has the exact factorization

$$A = LDU.$$

In [8], it has been shown that L and U can be computed as a by-products of Algorithm 1 and for $i < j$

$$L_{ji} = \beta_{ji}, \qquad U_{ij} = \alpha_{ij}.$$

Algorithm 2, computes the ILUFF factorization of matrix A. In this algorithm, the pivot entries are computed from line 11 instead of line 10, when the matrix is positive definite. This will guarantee the existence of the ILU factorization.

Algorithm 2 (ILU factorization obtained from FFAPINV algorithm)

```
w_1 = e_1^T, z_1 = e_1, d_1 = a_11.
for j = 2 to n do
    w_j = e_j^T, z_j = e_j.
    for i = 1 to j − 1 do
        L_ji = (A_{j,:} z_i)/d_i   U_ij = (w_i A_{:,j})/d_i
        apply a dropping rule to L_ji and to U_ij
        z_j = z_j − ((w_i A_{:,j})/d_i) z_i,  w_j = w_j − ((A_{j,:} z_i)/d_i) w_i
        for all l ≤ i apply a dropping rule to z_lj and to w_jl
    end for
    d_j = w_j A_{:,j} {not positive definite}
    d_j = w_j A w_j^T {positive definite}
end for
Return L = (L_ji)_{1≤j,i≤n}, D = diag(d_i)_{1≤i≤n} and V = (V_ij)_{1≤i,j≤n}
```

Suppose that at each step j of Algorithm 2, the vectors $q^{(j)}$ and $p^{(j)}$ are defined as:

$$q^{(j)} = (\frac{w_1 A_{:,j}}{d_1}, \cdots, \frac{w_{j-1} A_{:,j}}{d_{j-1}}, 0, \cdots, 0)^T, \quad p^{(j)} = (\frac{A_{j,:} z_1}{d_1}, \cdots, \frac{A_{j,:} z_{j-1}}{d_{j-1}}, 0, \cdots, 0).$$

Also suppose that I indicates the identity matrix and e_j is the j-th column of this matrix. We define matrices Q_j and P_j as:

$$Q_j = I - q^{(j)} e_j^T, \qquad P_j = I - e_j p^{(j)}.$$

Consider $W^{(j)}$ and $Z^{(j)}$ as the computed W and Z matrices at the end of step j, and $W^{(j-1)}$ and $Z^{(j-1)}$ as the computed W and Z matrices at the end of step $j-1$ of Algorithm 2, respectively. Therefore, one can observe that

$$Z^{(j)} = Z^{(j-1)} Q_j - T_j, \qquad W^{(j)} = P_j W^{(j-1)} - G_j,$$

in which G_j and T_j are the error matrices produced by the dropping strategy. Suppose that ε_W and ε_Z are the drop tolerance parameters for matrices W and Z, respectively. Then, the following two dropping strategies can be used to drop the entries of the vectors z_j and w_j.

- **First strategy:** At each step j of Algorithm 2, entries z_{lj} and w_{jl}, for $l \leq i$, are dropped when

$$|z_{lj}| \leq \varepsilon_Z, \qquad |w_{jl}| \leq \varepsilon_W. \tag{2.1}$$

- **Second strategy:** At each step j of Algorithm 2, the whole vectors

$$z_j = e_j - \sum_{i=1}^{j-1} (\frac{w_i A_{:,j}}{d_i}) z_i, \qquad w_j = e_j^T - \sum_{i=1}^{j-1} (\frac{A_{j,:} z_i}{d_i}) w_i,$$

will be computed and then, the entries z_{lj} and w_{jl}, for $l \leq j$, that satisfy the dropping criteria (2.1) will be dropped.

For both dropping criteria, just the entries $(G_j)_{jl}$ and $(T_j)_{lj}$, for $l < j$, will probably be nonzero.

Proposition 2.1. *For $i \leq j$, the following two relations*

$$T_j Q_i = T_j, \qquad P_i G_j = G_j, \tag{2.2}$$

hold. Suppose that no dropping is applied to the entries of the matrices L and U in Algorithm 2. At the end of step j of this algorithm, suppose that U_j is the matrix that its first j columns are the already computed columns of matrix U and its last $n-j$ columns are the columns of the identity matrix. Also, let L_j be the matrix that its first j rows are the already computed rows of matrix L and its last $n-j$ rows are the rows of the identity matrix. Then,

$$U_j = Q_j^{-1} Q_{j-1}^{-1} \cdots Q_2^{-1}, \qquad L_j = P_2^{-1} \cdots P_{j-1}^{-1} P_j^{-1}, \tag{2.3}$$

and

$$I - Z^{(j)} U_j = \sum_{i=2}^{j} T_i, \qquad I - L_j W^{(j)} = \sum_{i=2}^{j} G_i. \tag{2.4}$$

Proof. Because of the pattern of the matrices P_i, G_j and T_j, Q_i, for $i \leq j$, the relation (2.2) is clear. Relation (2.3) will be proved by considering the fact that $Q_i^{-1} = I + q^{(i)} e_i^T$ and $P_i^{-1} = I + e_i p^{(i)}$, for $i \leq j$. From line

7 of Algorithm 2 and the first part of the proposition; we have

$$\begin{aligned} Z^{(j)} &= Z^{(j-1)}Q_j - T_j \\ &= (Z^{(j-1)} - T_j)Q_j \\ &= [Z^{(j-2)}Q_{j-1} - T_{j-1} - T_j]Q_j \\ &= [Z^{(j-2)} - T_{j-1} - T_j]Q_{j-1}Q_j \\ &= [Z^{(j-3)}Q_{j-2} - T_{j-2} - T_{j-1} - T_j]Q_{j-1}Q_j \\ &\vdots \\ &= [I - \sum_{i=2}^{j} T_i]Q_2Q_3\cdots Q_j. \end{aligned}$$

Thus, $Z^{(j)}U_j = I - \sum_{i=2}^{j} T_i$ and the first part of relation (2.4) has been proved. Similarly, the second part of this relation is proved. □

At each step j of Algorithm 2, let $\tilde{q}^{(j)}$ and $\tilde{p}^{(j)}$ be the dropped $q^{(j)}$ and $p^{(j)}$ vectors, respectively. Thus, there are vectors

$$f_j = (f_{1j}, \cdots, f_{j-1j}, 0, \cdots, 0)^T, \qquad h_j = (h_{j1}, \cdots, h_{jj-1}, 0, \cdots, 0),$$

such that

$$\tilde{q}^{(j)} = q^{(j)} - f_j, \qquad \tilde{p}^{(j)} = p^{(j)} - h_j. \tag{2.5}$$

We define matrices $\tilde{Q}_j$ and $\tilde{P}_j$ as:

$$\tilde{Q}_j = I - \tilde{q}^{(j)}e_j^T, \qquad \tilde{P}_j = I - e_j\tilde{p}^{(j)}. \tag{2.6}$$

Proposition 2.2. *At the end of step j of Algorithm 2, suppose that U_j is a matrix that its first j columns are the already computed and dropped columns of matrix U and its last $n-j$ columns are the columns of the identity matrix. Also, let L_j be a matrix that its first j rows are the already computed and dropped rows of matrix L and its last $n-j$ rows are the rows of the identity matrix. Then,*

$$U_j = \tilde{Q}_j^{-1}\tilde{Q}_{j-1}^{-1}\cdots\tilde{Q}_2^{-1}, \qquad L_j = \tilde{P}_2^{-1}\cdots\tilde{P}_{j-1}^{-1}\tilde{P}_j^{-1}, \tag{2.7}$$

and

$$I - Z^{(j)}U_j = \sum_{i=2}^{j} T_i + Z^{(j)}(\sum_{i=2}^{j} f_ie_i^T),\ I - L_jW^{(j)} = \sum_{i=2}^{j} G_i + (\sum_{i=2}^{j} e_ih_i)W^{(j)}. \tag{2.8}$$

Proof. From the pattern of the matrices $\tilde{Q}_i^{-1}$ and $\tilde{P}_i^{-1}$, for $i \leq j$, the relation (2.7) is clear. Let $\tilde{U}_j = \tilde{Q}_j^{-1}\tilde{Q}_{j-1}^{-1} \cdots \tilde{Q}_2^{-1}$. Relations (2.5) and (2.6) imply that for $i < j$

$$\tilde{Q}_i = Q_i + f_i e_i^T.$$

Since for $i \leq j$, $\tilde{Q}_i^{-1} = Q_i^{-1} - f_i e_i^T$; then

$$\tilde{U}_j = U_j - \sum_{i=2}^{j} f_i e_i^T, \tag{2.9}$$

in which U_j has been defined in (2.3). Proposition 2.1 and relation (2.9) give

$$I - Z^{(j)}\tilde{U}_j = I - Z^{(j)}[U_j - \sum_{i=2}^{j} f_i e_i^T] = \sum_{i=2}^{j} T_i + Z^{(j)}(\sum_{i=2}^{j} f_i e_i^T).$$

If in the previous relation we rename the matrix U_j by $\tilde{U}_j$, then the first part of relation (2.8) is proved. Similarly, the second part of this relation is proved. □

Proposition 2.3. *Let $\varepsilon_{U,Z}$ and $\varepsilon_{L,W}$ be the same drop tolerance parameters for matrices U, Z and for matrices L, W, respectively. Suppose that at each step j of Algorithm 2, entries L_{jk} and U_{kj}, for $k < j$, are dropped when the criteria*

$$|L_{jk}|\|W_{k,:}\|_1 \leq \varepsilon_{L,W}, \qquad |U_{kj}|\|Z_{:,k}\|_\infty \leq \varepsilon_{U,Z}, \tag{2.10}$$

are satisfied. For $1 \leq i \leq j \leq n$

- *if the first dropping strategy is applied to drop the entries of matrices Z and W, then*

$$|(I - ZU)_{ij}| \leq 2(j-i)\varepsilon_{U,Z}, \ |(I - LW)_{ji}| \leq 2(j-i)\varepsilon_{L,W}. \tag{2.11}$$

- *if the second dropping strategy is applied to drop the entries of matrices Z and W, then*

$$|(I - ZU)_{ij}| \leq (j-i+1)\varepsilon_{U,Z}, \ |(I - LW)_{ji}| \leq (j-i+1)\varepsilon_{L,W}. \tag{2.12}$$

Proof. From Proposition 2.2 and the dropping criteria in (2.10), one can write

$$\begin{aligned} |e_i^T(I-ZU)e_j| &\leq |e_i^T(\sum_{k=2}^{n} T_k)e_j| + |e_i^T Z(\sum_{k=2}^{n} f_k e_k^T)e_j| \\ &= |e_i^T(\sum_{k=2}^{n} T_k)e_j| + |Z_{i,:}f_j| \\ &\leq |e_i^T(\sum_{k=2}^{n} T_k)e_j| + \sum_{k=i}^{j-1} |f_{kj}| \|Z_{:,k}\|_\infty \\ &\leq |e_i^T(\sum_{k=2}^{n} T_k)e_j| + (j-i)\varepsilon_{U,Z}. \end{aligned}$$

If the first dropping strategy is used for matrix Z, then $|e_i^T(\sum_{k=2}^{n} T_k)e_j| \leq (j-i)\varepsilon_{U,Z}$ and if the second dropping strategy is used for this matrix, then $|e_i^T(\sum_{k=2}^{n} T_k)e_j| \leq \varepsilon_{U,Z}$. Therefore, the first parts of relations (2.11) and (2.12) have been proved. Similarly, the second parts of these two relations are proved. □

3. Backward cactored INVerse process

The following algorithm is the BFINV algorithm [5, 11] which computes the exact factorization (1.1). If we drop the entries of the vectors z_j and w_j in each step j, then at the end of this algorithm, the approximate factorization (1.2) will be computed instead. In this case, the algorithm is called BFAPINV algorithm.

Algorithm 3 (BFINV algorithm)

1. $w_n = e_n^T$, $z_n = e_n$, $d_n = a_{nn}$.
2. **for** $j = n-1$ to 1 **do**
3. $\quad w_j = e_j^T$, $z_j = e_j$.
4. $\quad$ **for** $i = j+1$ to n **do**
5. $\quad\quad \beta_{ji} = \frac{A_{j,:}z_i}{d_i}$ $\alpha_{ij} = \frac{w_i A_{:,j}}{d_i}$
6. $\quad\quad z_j = z_j - \alpha_{ij} z_i$, $w_j = w_j - \beta_{ji} w_i$
7. $\quad$ **end for**
8. $\quad d_j = w_j A_{:,j}$ {not positive definite}
9. $\quad d_j = w_j A w_j^T$ {positive definite}
10. **end for**
11. Return $W = [w_1^T, \ldots, w_n^T]^T$, $D = diag(d_i)_{1 \leq i \leq n}$ and $Z = [z_1, \ldots, z_n]$.

Suppose that a matrix A has the exact factorization

$$A = UDL. \tag{3.1}$$

The work in [8] can easily be extended and one can show that L and U in (3.1) are the by-products of Algorithm 3 and for $i > j$

$$U_{ji} = \beta_{ji}, \qquad L_{ij} = \alpha_{ij}.$$

The following algorithm computes the IULBF factorization of matrix A. In this algorithm, the pivot entries are computed from line 11, instead of line 10, when the matrix is positive definite. This will guarantee the existence of the IUL factorization.

Algorithm 4 (IUL factorization obtained from BFAPINV algorithm)

1. $w_n = e_n^T$, $z_n = e_n$, $d_n = a_{nn}$.
2. **for** $j = n-1$ to 1 **do**
3. $\quad w_j = e_j^T$, $z_j = e_j$.
4. $\quad$ **for** $i = j+1$ to n **do**
5. $\qquad U_{ji} = \frac{A_{j,:}z_i}{d_i}$ $L_{ij} = \frac{w_i A_{:,j}}{d_i}$
6. $\qquad$ *apply a dropping rule to* U_{ji} *and to* L_{ij}
7. $\qquad z_j = z_j - (\frac{w_i A_{:,j}}{d_i})z_i$, $w_j = w_j - (\frac{A_{j,:}z_i}{d_i})w_i$
8. $\qquad$ *for all* $l \geq i$ *apply a dropping rule to* z_{lj} *and to* w_{jl}
9. $\quad$ **end for**
10. $\quad d_j = w_j A_{:,j}$ {not positive definite}
11. $\quad d_j = w_j A w_j^T$ {positive definite}
12. **end for**
13. Return $U = (U_{ji})_{1\leq j,i \leq n}$, $D = diag(d_i)_{1 \leq i \leq n}$ and $L = (L_{ij})_{1 \leq i,j \leq n}$

Suppose that at each step j of Algorithm 4, the vectors $q^{(j)}$ and $p^{(j)}$ are defined as:

$$q^{(j)} = (0, \cdots, 0, \frac{A_{j,:}z_{j+1}}{d_{j+1}}, \cdots, \frac{A_{j,:}z_n}{d_n}), \quad p^{(j)} = (0, \cdots, 0, \frac{w_{j+1}A_{:,j}}{d_{j+1}}, \cdots, \frac{w_n A_{:,j}}{d_n})^T.$$

We define matrices Q_j and P_j as:

$$Q_j = I - e_j q^{(j)}, \qquad P_j = I - p^{(j)} e_j^T.$$

Consider $W^{(j)}$ and $Z^{(j)}$ as the computed W and Z matrices at the end of step j and $W^{(j+1)}$ and $Z^{(j+1)}$ as the computed W and Z matrices at the end of step $j+1$ of Algorithm 4, respectively. One can easily verify the relations

$$W^{(j)} = Q_j W^{(j+1)} - G_j, \qquad Z^{(j)} = Z^{(j+1)} P_j - T_j,$$

in which G_j and T_j are the error matrices produced by the dropping strategy. The following two dropping strategies can be used to drop the entries of the vectors z_j and w_j.

- **First strategy:** At each step j of Algorithm 4, entries z_{lj} and w_{jl}, for $l \geq i$, are dropped when the criteria (2.1) are satisfied.
- **Second strategy:** At each step j of Algorithm 4, the whole vectors
$$z_j = e_j - \sum_{i=j+1}^{n} (\frac{w_i A_{:,j}}{d_i}) z_i, \qquad w_j = e_j^T - \sum_{i=j+1}^{n} (\frac{A_{j,:} z_i}{d_i}) w_i,$$
will be computed and then, the entries z_{lj} and w_{jl}, for $l \geq j$, that satisfy the dropping criteria (2.1) will be dropped.

For both dropping criteria, just the entries $(G_j)_{jl}$ and $(T_j)_{lj}$, for $l > j$, will probably be nonzero.

Proposition 3.1. *For $i \geq j$, the following two relations*
$$Q_i G_j = G_j, \qquad T_j P_i = T_j,$$
hold. Suppose that no dropping is applied to the entries of the matrices L and U in Algorithm 4. At the end of step j of this algorithm, suppose that U_j is the matrix that its last j rows are the already computed rows of matrix U and its first $n-j$ rows are the rows of the identity matrix. Also, let L_j be the matrix that its last j columns are the already computed columns of matrix L and its first $n-j$ columns are the columns of the identity matrix. Then,
$$U_j = Q_{n-1}^{-1} \cdots Q_{j+1}^{-1} Q_j^{-1}, \qquad L_j = P_j^{-1} P_{j+1}^{-1} \cdots P_{n-1}^{-1},$$
and
$$I - U_j W^{(j)} = \sum_{i=j}^{n-1} G_i, \qquad I - Z^{(j)} L_j = \sum_{i=j}^{n-1} T_i.$$

Proof. The proof is similar to that of Proposition 2.1. □

At each step j of Algorithm 4, suppose that $\tilde{q}^{(j)}$ and $\tilde{p}^{(j)}$ are the dropped $q^{(j)}$ and $p^{(j)}$ vectors, respectively. Thus, there are vectors
$$f_j = (0, \cdots, 0, f_{jj+1}, \cdots, f_{jn}), \qquad h_j = (0, \cdots, 0, h_{j+1j}, \cdots, h_{nj})^T,$$
such that the relation (2.5) holds. We define matrices $\tilde{Q}_j$ and $\tilde{P}_j$ as:
$$\tilde{Q}_j = I - e_j \tilde{q}^{(j)}, \qquad \tilde{P}_j = I - \tilde{p}^{(j)} e_j^T.$$

Proposition 3.2. *At the end of step j of Algorithm 4, suppose that U_j is a matrix that its last j rows are the already computed and dropped rows of matrix U and its first $n-j$ rows are the rows of the identity matrix. Also, let L_j be a matrix that its last j columns are the already computed and dropped columns of matrix L and its first $n-j$ columns are the columns of the identity matrix. Then,*

$$U_j = \tilde{Q}_{n-1}^{-1} \cdots \tilde{Q}_{j+1}^{-1} \tilde{Q}_j^{-1}, \qquad L_j = \tilde{P}_j^{-1} \tilde{P}_{j+1}^{-1} \cdots \tilde{P}_{n-1}^{-1},$$

and

$$I - U_j W^{(j)} = \sum_{i=j}^{n-1} G_i + (\sum_{i=j}^{n-1} e_i f_i) W^{(j)}, \qquad I - Z^{(j)} L_j = \sum_{i=j}^{n-1} T_i + Z^{(j)} (\sum_{i=j}^{n-1} h_i e_i^T).$$

Proof. The proof is similar to that of Proposition 2.2. □

Proposition 3.3. *Let $\varepsilon_{U,W}$ and $\varepsilon_{L,Z}$ be the same drop tolerance parameters for matrices U, W and for matrices L, Z, respectively. Suppose that at each step j of Algorithm 4, entries L_{kj} and U_{jk}, for $k > j$, are dropped when the criteria*

$$|L_{kj}| \|Z_{:,k}\|_\infty \le \varepsilon_{L,Z}, \qquad |U_{jk}| \|W_{k,:}\|_1 \le \varepsilon_{U,W}, \tag{3.2}$$

are satisfied. For $1 \le j \le i \le n$

- *if the first dropping strategy is applied to drop the entries of matrices Z and W, then*

$$|(I - UW)_{ji}| \le 2(i-j)\varepsilon_{U,W}, \qquad |(I - ZL)_{ij}| \le 2(i-j)\varepsilon_{L,Z}.$$

- *if the second dropping strategy is applied to drop the entries of matrices Z and W, then*

$$|(I - UW)_{ji}| \le (i-j+1)\varepsilon_{U,W}, \qquad |(I - ZL)_{ij}| \le (i-j+1)\varepsilon_{L,Z}.$$

Proof. The proof is similar to that of Proposition 2.3. □

4. Numerical results

In this section, we report the results of Bicgtab and GMRES(30) methods to solve the right preconditioned linear systems. The preconditioners are the ILUFF and the IULBF. All the 35 test matrices have been taken from the University of Florida Sparse Matrix Collection [3]. All the matrices are just nonsymmetric and not positive definite. In all the experiments whenever a zero pivot has been occurred, then the pivot element has been replaced by the square root of the machine precision. All the experiments were done on a machine with one quad Intel(R)

CPU and 8 GB of RAM memory. We have written the codes of ILUFF and IULBF preconditioners in Fortran 77. In these two codes, we have just used the CSC format of matrix A. To access the CSR format of this matrix; the linked lists trick [6] has been exploited. We have used the multilevel nested dissection reordering [4] as the preprocessing for all the matrices to compute the ILUFF and IULBF preconditioners.

Table 1, presents the information of the test matrices and the results of the Krylov subspace methods to solve the original systems but not the preconditioned ones. In this table, n and nnz indicate the dimension and the number of nonzero entries of the matrix, respectively, and the column $Group/Matrix$ shows the group and the name of the matrix. It in this table is the number of iterations of the Krylov subspace method and $Itime$ is its iteration time in seconds.

In this table, a + means that the stopping criterion has not been satisfied in 10000 number of iterations. For all the systems, the stopping criterion has been considered as:

$$\frac{\| r_k \|_2}{\| r_0 \|_2} \leq 10^{-10},$$

in which r_k is the k-th residual vector of the system and r_0 is the initial residual vector. For all the systems, the initial guess is the zero vector and the right hand side vector is Ae where $e = [1, 1, ..., 1]^T$.

In Tables 2-5, properties of the preconditioners and the results of the Krylov subspace methods which solve the right preconditioned linear systems have been presented. In these tables, $Ptime$ is the preconditioning time which is also in seconds and density for both ILUFF and IULBF preconditioners, is defined as:

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

in which $nnz(L)$, $nnz(U)$ and $nnz(A)$ refer to the number of nonzero entries of matrices L, U and A, respectively. For all matrices, the D and U factors of the ILUFF preconditioner and the D and L factors of the IULBF preconditioner have been merged.

To compute the ILUFF preconditioner, $\varepsilon_{L,W}$ has been used as the same drop tolerance parameter for matrices L and W and $\varepsilon_{U,Z}$ as the same drop tolerance parameter for matrices U and Z. In Table 2, $\varepsilon_{L,W} = \varepsilon_{U,Z} = 0.01$ has been selected for all the test matrices but in Table 4, $\varepsilon_{L,W} = \varepsilon_{U,Z} = 0.1$ has been considered. The notations ILUFF(0.01) and ILUFF(0.1) refer to this selection of drop tolerance parameters for

TABLE 1. matrix properties and results of iterative methods with no preconditioning

Group/Matrix	Matrix properties		Bicgstab		GMRES(30)	
	n	nnz	It	Itime	It	Itime
Engwirda/airfoil_2d	14214	259688	+	+	+	+
Bourchtein/atmosmodd	1270432	8814880	625	45.59	919	208.82
Bourchtein/atmosmodj	1270432	8814880	629	45.82	2158	491.85
Lucifora/cell2	7055	30082	+	+	+	+
Muite/Chebyshev3	4101	36879	+	+	+	+
Watson/chem_master1	40401	201201	1033	0.819	+	+
Oberwolfach/chipcool0	20082	281150	+	+	+	+
Oberwolfach/chipcool1	20082	281150	+	+	+	+
IBM_Austin/coupled	11341	97193	4081	1.86	+	+
IBM_EDA/dc1	116835	766396	+	+	+	+
IBM_EDA/dc2	116835	766396	+	+	+	+
IBM_EDA/dc3	116835	766396	+	+	+	+
Sanghavi/ecl32	51993	380415	+	+	+	+
Averous/epb1	14734	95053	1033	0.51	1682	1.63
Averous/epb2	25228	175028	847	0.68	1338	2.76
Oberwolfach/flowmeter5	9669	67391	+	+	+	+
Norris/lung2	109460	492564	+	+	+	+
QLi/majorbasis	160000	1750416	255	2.59	216	3.81
Hamm/memplus	17758	99147	2899	1.51	4477	5.11
FEMLAB/poisson3Db	13514	352762	513	7.53	693	12.33
Rajat/rajat03	7602	32653	2457	0.319	+	+
Rajat/rajat31	4690002	20316253	+	+	+	+
HB/sherman3	5005	20033	+	+	+	+
IBM_EDA/trans4	116835	749800	+	+	+	+
IBM_EDA/trans5	116835	749800	+	+	+	+
Simon/venkat01	62424	1717792	+	+	+	+
Wang/wang3	26064	177168	429	0.25	608	1.10
Wang/wang4	26068	177196	671	0.36	+	+
Simon/raefsky5	6316	167178	+	+	4649	2.82
Simon/raefsky6	3402	130371	+	+	2643	1.12
Sandia/ASIC − 100ks	99190	578890	+	+	+	+
Hamm/hcircuit	105676	513072	+	+	+	+
Sandia/ASIC − 680ks	682712	1693767	+	+	201	23.32
Sandia/ASIC − 320ks	321671	1316085	3283	54.93	526	61.28
FEMLAB/poisson3Da	13514	352762	259	0.32	444	5.348

ILUFF preconditioner. The dropping criteria (2.10) has been applied to drop the entries of matrices L and U and the first dropping strategy has been used to drop the entries of matrices Z and W.

To compute the IULBF preconditioner in Tables 3 and 5, $\varepsilon_{L,Z}$ and $\varepsilon_{U,W}$ have been used as the same drop tolerance parameters for matrices L, Z and U, W, respectively. In Table 3, the notation IULBF(0.01) indicates that $\varepsilon_{U,W} = \varepsilon_{L,Z} = 0.01$ has been considered for all the test matrices and the notation IULBF(0.1) in Table 5 means that $\varepsilon_{U,W} = \varepsilon_{L,Z} = 0.1$ has been taken. The dropping criteria (3.2) has been exploited to drop the entries of matrices L, U and again the first dropping strategy has been considered to drop the entries of matrices Z and W.

In Tables 2-5, *It* is again the number of iterations of the Krylov subspace method and *Ttime* is the total time which is the preconditioning time plus the iteration time. This papameter is also in seconds. In these tables, a + indicates that the convergence criterion has not been satisfied in 2500 number of iterations.

Numerical results of Tables 2 and 3, indicate that the *density* and the *Ptime* of both ILUFF(0.01) and IULBF(0.01) preconditioners are more or less the same as each other. Matrices $dc1$, $dc2$, $dc3$, $trans4$ and $trans5$ are exceptions. These results also show that these two preconditioners have nearly made the Krylov subspace methods convergent in the same number of iterations and total time.

Numerical results of Tables 4 and 5, also show that *Ptime* and *density* of both ILUFF(0.1) and IULBF(0.1) preconditioners are more or less the same except for matrices $dc1$, $dc2$, $dc3$, $trans4$ and $trans5$. These results also indicate that both of these two preconditioners are useful to decrease the number of iterations of the Bicgstab and GMRES(30) methods.

5. Conclusion

In this paper, new dropping techniques for ILU and IUL factorizations, which are obtained as by-products of FFAPINV and BFAPINV processes, have been presented. These types of droppings are known as the INVerse-based dropping techniques. Numerical experiments on 35 test matrices indicate that when the new dropping strategies are used to compute both of the ILU and IUL factorizations, then they are equally effective to reduce the number of iterations of the Krylov subspace methods.

TABLE 2. Properties of the ILUFF(0.01) preconditioner and results of iterative methods

	ILUFF(0.01)					
			Bicgstab		GMRES(30)	
Matrix	*Ptime*	*density*	*It*	*Ttime*	*It*	*Ttime*
airfoil_2d	0.67	0.282	449	1.26	+	+
atmosmodd	6.06	1.04	369	71.83	414	162.68
atmosmodj	6.05	1.04	393	71.14	667	258.53
cell2	0.61	1.3	345	0.75	+	+
Chebyshev3	0.59	0.33	267	0.64	248	0.67
chem_master1	0.72	1.45	399	2.12	1402	8.76
chipcool0	0.70	0.71	227	1.36	355	2.13
chipcool1	0.72	0.71	187	1.25	341	2.12
coupled	0.68	0.64	147	0.80	138	0.85
dc1	54.38	0.72	787	63.06	289	59.25
dc2	53.52	0.72	213	55.91	149	56.22
dc3	55.00	0.72	1177	68.02	829	69.02
ecl32	0.8	0.83	483	3.24	+	+
epb1	0.65	1.18	359	1.08	494	1.55
epb2	0.69	1.09	155	1.05	155	1.28
flowmeter5	0.63	1.052	451	0.95	2058	2.72
lung2	0.86	1.08	347	3.79	367	5.83
majorbasis	1.25	0.58	45	2.23	44	2.50
memplus	0.63	0.39	585	1.08	522	1.44
poisson3Db	1.27	0.51	139	1.10	361	10.59
rajat03	0.61	0.79	431	0.73	473	0.86
rajat31	14.27	0.79	825	355.0	1038	1037.7
sherman3	0.63	1.26	391	0.72	1846	1.36
trans4	38.07	0.65	141	39.06	132	40.24
trans5	39.37	0.66	233	41.9	371	45.55
venkat01	1.36	0.75	75	2.85	70	2.95
wang3	0.69	1.36	183	1.13	228	1.58
wang4	0.68	1.18	197	1.13	265	1.66
raefsky5	0.62	0.42	11	0.63	10	0.65
raefsky6	0.61	0.20	17	0.61	12	0.62
ASIC_100ks	0.89	0.98	51	0.49	20	1.16
hcircuit	0.96	0.99	375	4.44	469	7.74
ASIC_680ks	2.09	0.60	7	2.55	5	2.64
ASIC_320ks	1.44	0.67	117	5.25	49	3.85
poisson3Da	0.68	0.52	139	1.10	182	1.40

TABLE 3. Properties of the IULBF(0.01) preconditioner and results of iterative methods

Matrix	IULBF(0.01)					
			Bicgstab		GMRES(30)	
	Ptime	*density*	*It*	*Ttime*	*It*	*Ttime*
airfoil_2d	0.62	0.28	423	1.15	+	+
atmosmodd	5.42	1.01	419	70.57	446	219.25
atmosmodj	5.67	1.00	459	83.83	473	235.82
*cell*2	0.61	1.11	+	+	+	+
*Chebyshev*3	0.6	0.44	399	0.67	298	0.72
*chem_master*1	0.88	1.27	529	2.42	+	+
*chipcool*0	0.69	0.71	217	1.29	293	2.08
*chipcool*1	0.72	0.71	199	1.27	292	2.15
coupled	0.63	0.54	+	+	72	3.27
*dc*1	0.90	0.59	1003	11.69	334	7.86
*dc*2	0.88	0.58	263	3.67	184	4.69
*dc*3	0.88	0.58	925	10.78	389	8.97
*ecl*32	0.78	0.81	583	3.83	+	+
*epb*1	0.64	1.01	391	1.05	493	1.77
*epb*2	0.69	0.92	135	0.97	143	1.36
*flowmeter*5	0.64	1.31	321	0.88	763	1.72
*lung*2	0.87	1.25	1741	16.61	+	+
majorbasis	1.1	0.39	51	2.05	46	2.51
memplus	0.62	0.41	479	1.01	304	1.34
*poisson*3*Db*	1.38	0.56	257	7.55	319	11.38
*rajat*03	0.58	0.63	2099	1.14	435	0.91
*rajat*31	13.49	0.63	+	+	+	+
*sherman*3	0.62	1.24	397	0.73	2018	1.66
*trans*4	0.79	0.37	145	2.07	136	3.28
*trans*5	0.81	0.37	257	3.11	423	8.72
*venkat*01	1.30	0.74	79	2.69	71	2.86
*wang*3	0.70	1.32	171	1.12	175	1.58
*wang*4	0.72	1.31	157	1.11	143	1.44
*raefsky*5	0.59	0.38	11	0.59	10	0.61
*raefsky*6	0.57	0.22	13	0.58	11	0.59
ASIC_100ks	1.06	0.87	101	2.26	73	2.99
hcircuit	0.78	0.87	+	+	+	+
ASIC_680ks	1.9	0.92	7	2.32	5	2.53
ASIC_320ks	1.2	0.38	51	2.44	76	5.03
*poisson*3*Da*	0.65	0.57	131	1.02	133	1.18

TABLE 4. Properties of the ILUFF(0.1) preconditioner and results of iterative methods

	ILUFF(0.1)					
			Bicgstab		GMRES(30)	
Matrix	*Ptime*	*density*	*It*	*Ttime*	*It*	*Ttime*
$airfoil\_2d$	0.67	0.26	419	1.22	+	+
$atmosmodd$	4.92	0.63	441	70.33	485	166.37
$atmosmodj$	4.81	0.63	423	67.57	761	267.36
$cell2$	0.62	0.94	+	+	+	+
$Chebyshev3$	0.61	0.33	287	0.66	345	0.72
$chem\_master1$	0.72	0.96	479	2.02	1327	7.42
$chipcool0$	0.69	0.34	293	1.24	503	2.28
$chipcool1$	0.69	0.34	231	1.19	493	2.24
$coupled$	0.66	0.48	159	0.77	168	0.83
$dc1$	52.00	0.64	+	+	273	56.11
$dc2$	51.02	0.63	277	53.94	167	53.72
$dc3$	51.02	0.64	759	58.97	742	62.77
$ecl32$	0.76	0.51	777	4.18	+	+
$epb1$	0.61	0.78	415	1.00	551	1.14
$epb2$	0.69	0.57	195	1.00	212	1.33
$flowmeter5$	0.60	0.74	521	0.90	1990	2.33
$lung2$	0.84	1.03	361	3.81	392	6.27
$majorbasis$	1.13	0.52	47	1.98	45	2.23
$memplus$	0.61	0.39	571	1.05	551	1.44
$poisson3Db$	0.97	0.17	313	5.68	481	10.66
$rajat03$	0.58	0.78	409	0.69	505	0.84
$rajat31$	15.51	0.77	897	405.91	1048	1240.50
$sherman3$	0.65	0.83	505	0.76	+	+
$trans4$	34.57	0.62	131	35.83	134	36.59
$trans5$	33.51	0.61	265	36.30	453	40.80
$venkat01$	0.97	0.34	101	2.34	93	2.54
$wang3$	0.73	0.84	203	1.11	268	1.59
$wang4$	0.75	0.55	239	1.08	344	1.72
$raefsky5$	0.75	0.19	13	0.75	11	0.75
$raefsky6$	0.70	0.14	15	0.71	12	0.71
$ASIC\_100ks$	0.86	0.78	51	1.31	23	1.15
$hcircuit$	0.89	0.75	481	4.81	649	9.13
$ASIC\_680ks$	2.35	0.60	7	2.80	6	2.97
$ASIC\_320ks$	1.51	0.65	961	31.44	51	4.02
$poisson3Da$	0.68	0.18	157	0.95	185	1.17

TABLE 5. Properties of the IULBF(0.1) preconditioner and results of iterative methods

	IULBF(0.1)					
			Bicgstab		GMRES(30)	
Matrix	*Ptime*	*density*	*It*	*Ttime*	*It*	*Ttime*
*airfoil*_ 2d	0.66	0.26	447	1.25	+	+
atmosmodd	4.69	0.62	453	66.30	451	222.82
atmosmodj	4.87	0.62	375	54.62	608	303.52
*cell*2	0.58	0.86	539	0.74	+	+
*Chebyshev*3	0.59	0.43	261	0.63	195	0.66
*chem*_ *master*1	0.66	0.86	575	1.96	+	+
*chipcool*0	0.66	0.33	255	1.13	393	2.19
*chipcool*1	0.68	0.33	239	1.13	349	2.10
coupled	0.59	0.40	689	1.05	+	+
*dc*1	0.83	0.52	+	+	704	15.51
*dc*2	0.86	0.52	337	4.12	281	6.26
*dc*3	0.88	0.51	1083	12.11	645	14.04
*ecl*32	0.75	0.50	615	3.51	+	+
*epb*1	0.69	0.7	413	1.04	567	1.85
*epb*2	0.67	0.61	171	0.96	165	1.36
*flowmeter*5	0.60	0.81	419	0.85	1045	1.85
*lung*2	0.88	1.11	+	+	+	+
majorbasis	1.07	0.27	53	1.98	50	2.70
memplus	0.61	0.37	1497	1.76	485	1.7
*poisson*3*Db*	1.02	0.18	299	6.04	402	10.68
*rajat*03	0.57	0.63	+	+	584	1.04
*rajat*31	14.39	0.63	+	+	+	+
*sherman*3	0.61	0.82	557	0.73	2123	1.63
*trans*4	0.86	0.34	253	3.22	187	4.47
*trans*5	0.85	0.34	477	5.27	530	11.15
*venkat*01	0.94	0.33	107	2.30	87	2.47
*wang*3	0.64	0.83	203	0.99	220	1.54
*wang*4	0.66	0.60	221	0.97	232	1.54
*raefsky*5	0.63	0.18	11	0.63	10	0.63
*raefsky*6	0.59	0.16	15	0.60	12	0.60
*ASIC*_ 100*ks*	0.82	0.57	+	+	73	2.01
hcircuit	0.83	0.51	305	3.08	+	+
*ASIC*_ 680*ks*	2.14	0.91	7	2.61	5	2.85
*ASIC*_ 320*ks*	1.27	0.38	53	2.59	77	5.53
*poisson*3*Da*	0.68	0.18	161	0.94	166	1.21

Acknowledgments

I would like to thank the referee for his/her helpful comments and suggestions which really improved the quality of the paper. I would also like to thank Professor Matthias Bollhöfer for providing the codes that I could apply the multilevel nested dissection reordering.

References

[1] M. Bollhöfer, A robust ILU with pivoting based on monitoring the growth of the INVerse factors, *Linear Algebra Appl.* **338** (2001) 201–218.

[2] M. Bollhöfer, A robust and efficient ILU that incorporates the growth of the INVerse triangular factors, *SIAM J. Sci. Comput.* **25** (2003), no. 1, 86–103.

[3] T. Davis, University of Florida Sparse Matrix Collection, `http://www.cise.ufl.edu/research/sparse/matrices`, Accessed 2011.

[4] 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 2010.

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

[6] N. Li, Y. Saad and E. Chow, Crout versions of ILU for general sparse matrices, *SIAM J. Sci. Comput.* **25** (2003), no. 2, 716–728.

[7] D. K. Salkuyeh, A Sparse Approximate INVerse Preconditioner for Nonsymmetric Positive Definite Matrices, *J. Appl. Math and Inform.* **28** (2010) no. 5-6 1131–1141.

[8] D. K. Salkuyeh, A. Rafiei and H. Roohani, ILU Preconditioning Based on the FAPINV Algorithm, *Opuscula Math.* Accepted.

[9] Y. Saad, Iterative Methods for Sparse Linear Systems, Second edition. Society for Industrial and Applied Mathematics, Philadelphia, 2003.

[10] J. Zhang, A Procedure for Computing Factored Approximate INVerse, M.Sc. dissertation, Department of Computer Science, University of Kentucky, 1999.

[11] 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, P.O. Box 9617976487, Sabzevar, Iran

E-mail address: `rafiei.am@gmail.com, a.rafiei@hsu.ac.ir`