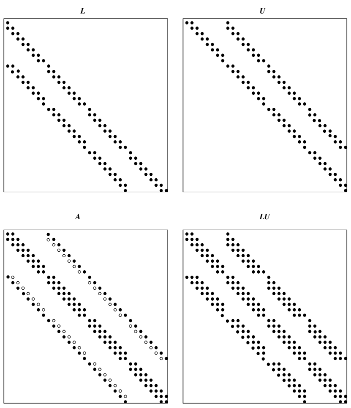
```
        if ( jrow .ne. k .or.
    +       luval(j) .eq. 0.0d0 ) goto 600
c---------------------- invert diagonal
        luval(j) = 1.0d0/luval(j)
c---------------------- zero work array
        do 210 i = j1, j2
           iw(ja(i)) = 0
 201    continue
 500 continue
c---------------------- normal return
     icode = 0
     return
c---------------------- zero pivot
 600 icode = k
     return
     end
```

Level of Fill and ILU(p)

- ILU(0) may fail for many realistic problems
- more accurate ILU preconditioners may be obtained by allowing more fill-in elements
- more accurate preconditioners cost more to construct, but may converge fast
- preconditioner accuracy is controlled by drop strategies
- level based drop strategy and threshold based drop strategy



The ILU(1) factorization for a five-point matrix

ILU(1) for Five-Point Matrix

ILU(0) only allows fill-in in the nonzero position of A.

ILU(1) allows one more line of fill-in in each of the L and U factors. This is achieved by viewing A with one more imaginary offdiagonal in each of the upper and lower triangular parts.

Higher accuracy can be obtained by allowing even more fill-in elements.

This approach is only applicable to structured matrices. For general sparse matrices, the concept of level of fill is needed.

Level of Fill

A typical update step in ILU is

$$a_{ij} = a_{ij} - a_{ik} \times a_{kj}$$

Let ϵ^k is attributed to any element whose level of fill is k, where $\epsilon < 1$. Initially, a nonzero element has a level of fill of one, a zero element has a level of fill of ∞. The update can viewed as

$$a_{ij} = \epsilon^{lv_{ij}} - \epsilon^{lv_{ik}} \times \epsilon^{lv_{kj}} = \epsilon^{lv_{ij}} - \epsilon^{lv_{ik}+lv_{kj}}$$

Define the new level of fill for ij as

$$lv_{ij} = \min\{lv_{ij}, lv_{ik} + lv_{kj}\}$$

ILU(p)

In favor of the definition of ILU(0), the level of fill is actually shifted by one. For the elements of A, initial definition is

$$lv_{ij} = \begin{cases} 0 & \text{if } a_{ij} \neq 0, \text{ or } i = j \\ \infty & \text{otherwise.} \end{cases}$$

The new level of fill with update is

$$lv_{ij} = \min\{lv_{ij}, lv_{ik} + lv_{kj} + 1\}$$

The zero pattern of ILU(p) is

$$P_p = \{(i, j) \mid lv_{ij} > p\}$$

Element whose level of fill is greater than p is dropped. Computation process consists of symbolic and numerical factorizations

Disadvantages of Level of Fill

- for general sparse matrices, storage and computational costs cannot be predicated in advance
- for indefinite matrices, the algorithm may drop large size elements and result in an inaccurate factorization
- the cost of updating the levels can be quite high
- threshold based algorithms can overcome these disadvantages to a certain degree

Threshold (Value) Based Strategies

- no sparsity pattern is prescribed in advance, drop small size elements as they are computed
- preconditioners can be constructed by using Gaussian elimination procedure and dropping small elements
- the total number of elements in each L and U factors can be limited. Storage cost is known in advance
- double dropping strategy results in ILUT

ILUT

```
1.  For i = 1,...,n do
2.    w := a_{i*}
3.    For k = 1,...,i-1 and when w_k ≠ 0 do
4.      w_k := w_k / a_{kk}
5.      Apply a dropping rule to w_k
6.      If w_k ≠ 0 then
7.        w := w - w_k * u_{k*}
8.      End if
9.    End do
10.   Apply a dropping rule to row w
11.   l_{ij} := w_j for j = 1,...,i-1
12.   u_{ij} := w_j for j = i,...,n
13.   w := 0
14. End do
```

Comments on ILUT(p, τ)

- small element w_k (with respect to τ and relative to a certain norm of the current row) are dropped in line 5
- in line 10, small elements are dropped again. A search algorithm is used to find out the largest p elements in size. These p elements are kept, the others are dropped again
- total storage is bounded by $2pn$
- τ controls computational cost, p controls memory cost

Tests with ILUT

GMRES(10) with ILUT($1, 10^{-4}$)

Matrix	Iters	Kflops	Residual	Error
F2DA	18	964	0.47E-03	0.41E-04
F3D	14	3414	0.11E-02	0.39E-03
ORS	6	341	0.13E+00	0.60E-04
F2DB	130	7167	0.45E-02	0.51E-03
FID	59	19112	0.19E+00	0.11E-03

GMRES(10) with ILUT($5, 10^{-4}$)

Matrix	Iters	Kflops	Residual	Error
F2DA	7	478	0.13E-02	0.90E-04
F3D	9	2855	0.58E-03	0.35E-03
ORS	4	270	0.92E-01	0.43E-04
F2DB	10	724	0.62E-03	0.26E-03
FID	40	14862	0.11E+00	0.11E-01

The cost of constructing preconditioner is not included

Comments on ILU Preconditioners

Preconditioners based on incomplete LU factorization of the coefficient matrix are usually inherently sequential, both in constructions and in applications

Certain degree of parallelism can be extracted by using level scheduling or multicoloring techniques

High accuracy ILU preconditioners are more robust and may be more efficient, but they have less parallelism due to increased coupling among nodes. Multilevel block ILU preconditioners are designed to achieve high accuracy and parallelism at the same time