LU Factorization with pivoting.

Example: $Ax = \begin{bmatrix} c & 1 \\ -1 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix}$

start with an sol $x_{exact} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$

then we can solve it.

np.tril( ) 返回 下三角矩阵

np.triu( ) 返回 上三角矩阵

np.eye(m, n) 生成 m×n 对角矩阵.

$$\begin{bmatrix} A_{11} & A_{12} \cdots A_{1n} \\ A_{21} & A_{22} \cdots A_{2n} \\ \vdots & \\ A_{n1} & A_{n2} \cdots A_{nn} \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ L_{21} & L_{22} \end{bmatrix} \begin{bmatrix} U_{11} & U_{12} \\ 0 & U_{22} \end{bmatrix}$$

$$= \begin{bmatrix} U_{11} & U_{12} \\ L_{21}U_{11} & L_{21}U_{12} + L_{22}U_{22} \end{bmatrix}$$

$L_{22}U_{22} = a_{22} - L_{21}U_{12}$

if $= \begin{bmatrix} 0 & \square \\ \square & \square \end{bmatrix}$

$A_{11}' = 0$ what should we do?

Pivoting

1. Swap rows if there is zero entry in the diagonal.

2. Better: find the largest entry (by absolute value) and swap it to the top row.

the entry is called Pivot.

Sparse Matrices.

number of operations required

$A+B$: $O(n^2) \longrightarrow O(nnz(A)+nnz(B))$

$n \times n$ $n \times n$

number of nonzero elements

matrix with few non-zero entries.

$$A = \begin{bmatrix} 0 & 1.9 & 0 & -5.2 \\ 0.3 & 0 & 9.1 & 0 \\ 4.4 & 5.8 & 3.6 & 0 \\ 0 & 0 & 7.2 & 2.7 \end{bmatrix}$$

Dense (DNS): $A_{dense} = [$ 0 1.9 0 -5.2 | 0.3 0 9.1 0 | 4.4 5.8 3.6 0 | 0 0 7.2 2.7 $]$

row 1 | row 2 | row 3 | row 4

Coordinate Form (COO):

$$A = \begin{bmatrix} 0 & 1.9 & 0 & -5.2 \\ 0.3 & 0 & 9.1 & 0 \\ 4.4 & 5.8 & 3.6 & 0 \\ 0 & 0 & 7.2 & 2.7 \end{bmatrix}$$

(0, 1, 2, 3 label the columns and rows)

data = [1.9 -5.2 0.3 9.1 4.4 5.8 3.6 7.2 2.7]
<floats>

row = [0 0 1 1 2 2 2 3 3]

col = [1 3 0 2 0 1 2 2 3]
<int>.

order can be changed.

Compressed Sparse Row (CSR):

$$A = \begin{bmatrix} 0 & 1.9 & 0 & -5.2 \\ 0 & 0 & 0 & 0 \\ 4.4 & 5.8 & 3.6 & 0 \\ 0 & 0 & 7.2 & 2.7 \end{bmatrix}$$

0 → 2
1 → 0
2 → 3
3 → 2

data = [1.9 −5.2 4.4 5.8 3.6 7.2 2.7].

col = [1 3 0 1 2 2 3]

rowp = [0 2 2 5 7]

+2 +0 +3 +2

plus nnz row i.