Solving Linear System of Equations

Triangular Matrices

Backward substitution

$$\begin{pmatrix} U_{11} & U_{12} & \dots & U_{1n} \\ 0 & U_{22} & \dots & U_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & U_{nn} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} b \\ b_2 \\ \vdots \\ b_n \end{pmatrix}$$

Recall that we can also write $\boldsymbol{U}\,\boldsymbol{x} = \boldsymbol{b}$ as a linear combination of the columns of $\boldsymbol{U}$

$$\begin{bmatrix} U_{11} \\ 0 \\ \vdots \\ 0 \\ \vdots \\ 0 \end{bmatrix} x_1 + \begin{bmatrix} U_{12} \\ U_{22} \\ 0 \\ \vdots \\ 0 \end{bmatrix} x_2 + \dots + \begin{bmatrix} U_{1i} \\ U_{2i} \\ \vdots \\ U_{ii} \\ 0 \\ \vdots \\ 0 \end{bmatrix} x_i + \dots + \begin{bmatrix} U_{1n-1} \\ U_{2n-1} \\ \vdots \\ U_{i,n-1} \\ \vdots \\ U_{n-1,n-1} \\ 0 \end{bmatrix} x_{n-1} + \begin{bmatrix} U_{1n} \\ U_{2n} \\ \vdots \\ U_{in} \\ \vdots \\ U_{nn} \end{bmatrix} x_n = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix}$$

$$U_{nn} x_n = b_n \longrightarrow x_n = b_n / U_{nn}$$

$$\begin{bmatrix} u_{11} \\ 0 \\ \vdots \\ 0 \\ \vdots \\ 0 \end{bmatrix} x_1 + \begin{bmatrix} u_{12} \\ u_{22} \\ 0 \\ \vdots \\ \vdots \\ 0 \end{bmatrix} x_2 + \cdots + \begin{bmatrix} u_{1i} \\ u_{2i} \\ \vdots \\ u_{ii} \\ 0 \\ \vdots \\ 0 \end{bmatrix} x_i + \cdots + \begin{bmatrix} u_{1n-1} \\ u_{2n-1} \\ \vdots \\ u_{i,n-1} \\ \vdots \\ u_{n-1,n-1} \\ 0 \end{bmatrix} x_{n-1} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_i \\ \vdots \\ b_n \end{bmatrix} - \begin{bmatrix} u_{1n} \\ u_{2n} \\ \vdots \\ u_{in} \\ \vdots \\ u_{nn} \end{bmatrix} x_n$$

$$u_{n-1,n-1} x_{n-1} = b_{n-1} - u_{n-1,n} x_n$$

$$x_{n-1} = \frac{b_{n-1} - u_{n-1,n} x_n}{u_{n-1,n-1}}$$

$$u_{ii} x_i = b_i - \sum_{j=i+1}^{n} u_{ij} x_j$$

$$x_i = \frac{b_i - \sum_{j=i+1}^{n} u_{ij} x_j}{u_{ii}}$$

$$\begin{bmatrix} u_{11} \\ 0 \\ \vdots \\ 0 \\ \vdots \\ 0 \end{bmatrix} x_1 + \begin{bmatrix} u_{12} \\ u_{22} \\ 0 \\ \vdots \\ \vdots \\ 0 \end{bmatrix} x_2 + \cdots + \begin{bmatrix} u_{1i} \\ u_{2i} \\ \vdots \\ u_{ii} \\ 0 \\ \vdots \\ 0 \end{bmatrix} x_i \cdots = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_i \\ \vdots \\ b_n \end{bmatrix} - \begin{bmatrix} u_{1n} \\ u_{2n} \\ \vdots \\ u_{in} \\ \vdots \\ u_{nn} \end{bmatrix} x_n - \begin{bmatrix} u_{1n-1} \\ u_{2n-1} \\ \vdots \\ u_{i,n-1} \\ \vdots \\ u_{n-1,n-1} \\ 0 \end{bmatrix} x_{n-1} - \cdots$$

$$i = n, n-1, \cdots$$

Triangular Matrices

Forward-substitution for lower-triangular systems:

$$\begin{pmatrix} L_{11} & 0 & \dots & 0 \\ L_{21} & L_{22} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ L_{n1} & L_{n2} & \dots & L_{nn} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} b \\ b_2 \\ \vdots \\ b_n \end{pmatrix}$$

$$x_1 = b_1/L_{11} \qquad x_i = \frac{b_i - \sum_{j=1}^{i-1} L_{ij} x_j}{L_{ii}}, \qquad i = 2,3,\dots,n$$

same procedure for the forward substitution

Backward substitution

$$x_i = \frac{b_i - \sum_{j=i+1}^{n} U_{ij} x_j}{U_{ii}} \qquad i = n, n-1, n-2, \cdots, 1$$

	$i=n$	$i=n-1$	$i=n-2$	$\cdots$	$i=1\ (n-(n-1))$	TOTAL
# divisions	1	1	1	$\cdots$	1	n
# multiplications	0	1	2	$\cdots$	$(n-1)$	$\frac{1}{2}n(n-1)$
# subtractions/ additions	0	1	2	$\cdots$	$(n-1)$	$\frac{1}{2}n(n-1)$

$$\sum_{i=1}^{m} i = \frac{1}{2}m(m+1) \Longrightarrow \frac{1}{2}(n-1)(n-1+1) = \frac{1}{2}n(n-1)$$

$$\text{Total} = n + n(n-1) \Longrightarrow \text{Computational Complexity } O(n^2) \quad (n \to \infty)$$

Cost of solving triangular systems

$$x_n = b_n/U_{nn} \qquad x_i = \frac{b_i - \sum_{j=i+1}^{n} U_{ij}x_j}{U_{ii}}, \qquad i = n-1, n-2, \dots, 1$$

n divisions
$n(n-1)/2$ subtractions/additions
$n(n-1)/2$ multiplications

Computational complexity is $O(n^2)$

$$x_1 = b_1/L_{11} \qquad x_i = \frac{b_i - \sum_{j=1}^{i-1} L_{ij}x_j}{L_{ii}}, \qquad i = 2,3, \dots, n$$

n divisions
$n(n-1)/2$ subtractions/additions
$n(n-1)/2$ multiplications

Computational complexity is $O(n^2)$

Demo "Coding Back-substitution"

LU Factorization

$$\begin{pmatrix} 1 & 0 & \dots & 0 \\ L_{21} & 1 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ L_{n1} & L_{n2} & \dots & 1 \end{pmatrix} \begin{pmatrix} U_{11} & U_{12} & \dots & U_{1n} \\ 0 & U_{22} & \dots & U_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & U_{nn} \end{pmatrix} = \begin{pmatrix} A_{11} & A_{12} & \dots & A_{1n} \\ A_{21} & A_{22} & \dots & A_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ A_{n1} & A_{n2} & \dots & A_{nn} \end{pmatrix}$$

Assuming the LU factorization is know, we can solve the general system

① $A = LU \longrightarrow$?

$LUx = b$

② $Ly = b \longrightarrow$ solve for y $O(n^2)$

③ $Ux = y \longrightarrow$ solve for x $O(n^2)$

Iclicker question

Assume the $\boldsymbol{A = LU}$ factorization is known, yielding:

$$\boldsymbol{L} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0.5 & 1 & 0 & 0 \\ 0.5 & 1 & 1 & 0 \\ 0.5 & 0.5 & 0.5 & 1 \end{pmatrix} \quad \boldsymbol{U} = \begin{pmatrix} 2 & 8 & 4 & 1 \\ 0 & -2 & 1 & 2.5 \\ 0 & 0 & 3 & -1 \\ 0 & 0 & 0 & 0.75 \end{pmatrix}$$

Determine the solution $\boldsymbol{x}$ that satisfies $\boldsymbol{Ax = b}$, when $\boldsymbol{b} = \begin{pmatrix} 2 \\ 2 \\ 1 \\ 4 \end{pmatrix}$

$$\boldsymbol{L}\underbrace{\boldsymbol{Ux}}_{\boldsymbol{y}} = \boldsymbol{b}$$

First, solve the lower-triangular system $\boldsymbol{L\,y = b}$ for the variable $\boldsymbol{y}$

Then, solve the upper-triangular system $\boldsymbol{U\,x = y}$ for the variable $\boldsymbol{x}$

$$A = \begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ L_{21} & 1 \end{pmatrix} \begin{pmatrix} U_{11} & U_{12} \\ 0 & U_{22} \end{pmatrix}$$

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

① First row of U is the first row of A

$$U_{11} = A_{11} \qquad U_{12} = A_{12}$$

② $A_{21} = L_{21}U_{11} \rightarrow L_{21} = \frac{A_{21}}{U_{11}} = \frac{A_{21}}{A_{11}}$ (corresponding column of A divided by diagonal entry above)

③ $A_{22} = L_{21}U_{12} + U_{22}$

$$U_{22} = A_{22} - L_{21}U_{12} = A_{22} - \frac{A_{21}}{A_{11}}A_{12}$$

2x2 LU Factorization (simple example)

$$\begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ L_{21} & 1 \end{pmatrix} \begin{pmatrix} U_{11} & U_{12} \\ 0 & U_{22} \end{pmatrix}$$

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

$U_{11} = A_{21}/U_{11}$

2) $L_{21} = A_{21}/U_{11}$

3) $U_{22} = A_{22} - L_{21}U_{12}$

Seems quite simple! Can we generalize this for a $n \times n$ matrix $\boldsymbol{A}$?

Iclicker question

Which of the following statements about the $\boldsymbol{A = LU}$ factorization are true?

1) ☐ It can happen that A is invertible, but U is not. False
2) ☐ U has a diagonal that is full of ones. False
3) ☐ A factorization $A = LU$ does not always exist. True
4) ☐ L has a diagonal that is full of ones. True

A) 1 and 2
B) 1 and 3
C) 2 and 3
D) 2 and 4
E) 3 and 4

Iclicker question

Computing the Lower-Triangular Factor in LU

1 point

Consider the matrix

$$A = \begin{bmatrix} 2 & 3 \\ 1 & 4 \end{bmatrix}$$

and its corresponding LU factorization ($A = LU$), where the lower and upper triangular matrices given respectively by

$$L = \begin{bmatrix} 1 & 0 \\ l_{21} & 1 \end{bmatrix} \quad \text{and} \quad U = \begin{bmatrix} u_{11} & u_{12} \\ 0 & u_{22} \end{bmatrix}.$$

l_{21} = 0.5

u_{11} = 2

u_{12} = 3

u_{22} = $4 - \frac{(1)(3)}{2} = 2.5$

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

$$\begin{bmatrix} A_{11} & \underset{\sim}{a}_{12} \\ \underset{\sim}{a}_{21} & \underset{=}{A}_{22} \end{bmatrix} = \begin{bmatrix} 1 & \underset{\sim}{0} \\ \underset{\sim}{l}_{21} & \underset{=}{L}_{22} \end{bmatrix} \begin{bmatrix} U_{11} & \underset{\sim}{U}_{12} \\ \underset{\sim}{0} & \underset{=}{U}_{22} \end{bmatrix}$$

$$\begin{bmatrix} A_{11} & \underset{\sim}{a}_{12} \\ \underset{\sim}{a}_{21} & \underset{=}{A}_{22} \end{bmatrix} = \begin{bmatrix} U_{11} & \underset{\sim}{U}_{12} \\ U_{11}\,\underset{\sim}{l}_{21} & \underbrace{\underset{\sim}{l}_{21}\,\underset{\sim}{U}_{12}}_{\text{outer product!}} + \underset{=}{L}_{22}\,\underset{=}{U}_{22} \end{bmatrix}$$

$$\begin{bmatrix} A_{11} & \underset{\sim}{a}_{12} \\ \underset{\sim}{a}_{21} & \underline{\underline{A}}_{22} \end{bmatrix} = \begin{bmatrix} U_{11} & \underset{\sim}{u}_{12} \\ U_{11}\,\underset{\sim}{l}_{21} & \underset{\sim}{l}_{21}\,\underset{\sim}{u}_{12} + \underline{\underline{L}}_{22}\,\underline{\underline{U}}_{22} \end{bmatrix}$$

outer product!

- $A_{11} = U_{11}$, $\underset{\sim}{a}_{12} = \underset{\sim}{u}_{12}$ $\longrightarrow$ 1st row of $\underline{\underline{U}}$ is the first row of $\underline{\underline{A}}$!

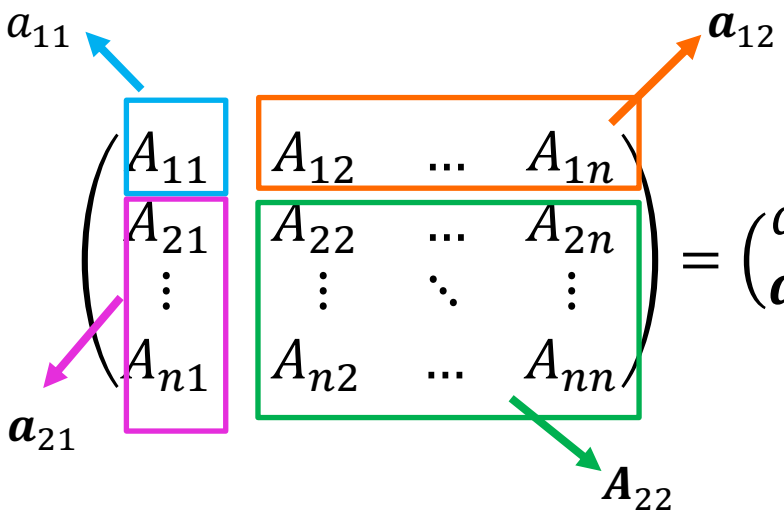
- $\underset{\sim}{a}_{21} = U_{11}\,\underset{\sim}{l}_{21} \Rightarrow \underset{\sim}{l}_{21} = \frac{1}{U_{11}}\,\underset{\sim}{a}_{21}$ $\longrightarrow$ take column of $\underline{\underline{A}}$ (not including diagonal entry) and divide by diagonal entry of $\underline{\underline{A}}$.

- $\underline{\underline{A}}_{22} = \underset{\sim}{l}_{21}\,\underset{\sim}{u}_{12} + \underline{\underline{L}}_{22}\,\underline{\underline{U}}_{22}$

$\underline{\underline{L}}_{22}\,\underline{\underline{U}}_{22} = \underline{\underline{A}}_{22} - \underset{\sim}{l}_{21}\,\underset{\sim}{u}_{12}$ $\longrightarrow$ what next? Repeat the same process for this smaller matrix! Recursion

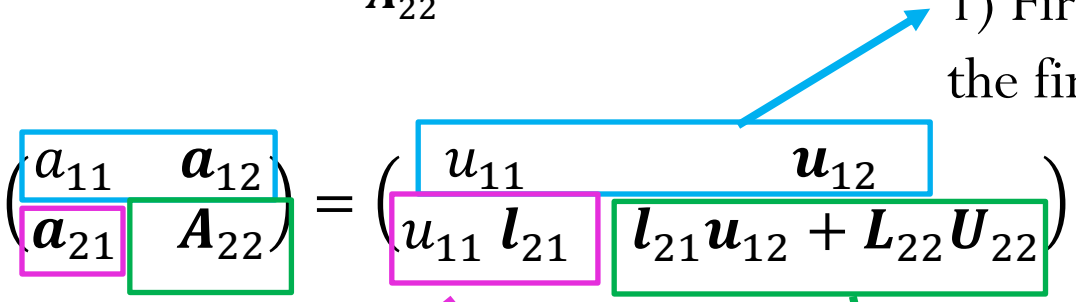
LU Factorization

a_{11}: scalar
$\boldsymbol{a}_{12}$: row vector $(1\times(n-1))$
$\boldsymbol{a}_{21}$: column vector $((n-1)\times 1)$
$\boldsymbol{A}_{22}$: matrix $((n-1)\times(n-1))$



$$\begin{pmatrix} A_{11} & A_{12} & \cdots & A_{1n} \\ A_{21} & A_{22} & \cdots & A_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ A_{n1} & A_{n2} & \cdots & A_{nn} \end{pmatrix} = \begin{pmatrix} a_{11} & \boldsymbol{a}_{12} \\ \boldsymbol{a}_{21} & \boldsymbol{A}_{22} \end{pmatrix} = \begin{pmatrix} 1 & \boldsymbol{0} \\ \boldsymbol{l}_{21} & \boldsymbol{L}_{22} \end{pmatrix} \begin{pmatrix} u_{11} & \boldsymbol{u}_{12} \\ \boldsymbol{0} & \boldsymbol{U}_{22} \end{pmatrix}$$

1) First row of $\boldsymbol{U}$ is the first row of $\boldsymbol{A}$



$$\begin{pmatrix} a_{11} & \boldsymbol{a}_{12} \\ \boldsymbol{a}_{21} & \boldsymbol{A}_{22} \end{pmatrix} = \begin{pmatrix} u_{11} & \boldsymbol{u}_{12} \\ u_{11}\,\boldsymbol{l}_{21} & \boldsymbol{l}_{21}\boldsymbol{u}_{12} + \boldsymbol{L}_{22}\boldsymbol{U}_{22} \end{pmatrix}$$

2) $\boldsymbol{l}_{21} = \frac{1}{u_{11}}\boldsymbol{a}_{21}$

First column of $\boldsymbol{L}$ is the first column of $\boldsymbol{A}/\ u_{11}$

3) $\boldsymbol{M} = \boldsymbol{L}_{22}\boldsymbol{U}_{22} = \boldsymbol{A}_{22} - \boldsymbol{l}_{21}\boldsymbol{u}_{12}$

Known!

Need another factorization!

$$M = \begin{pmatrix} 2 & 8 & 4 & 1 \\ 1 & 2 & 3 & 3 \\ 1 & 2 & 6 & 2 \\ 1 & 3 & 4 & 2 \end{pmatrix}$$

$$L = \begin{bmatrix} 1 & \emptyset & \emptyset & \emptyset \\ 0.5 & 1 & \emptyset & \emptyset \\ 0.5 & 1 & 1 & \emptyset \\ 0.5 & 0.5 & 0.5 & 1 \end{bmatrix} \qquad U = \begin{bmatrix} 2 & 8 & 4 & 1 \\ \emptyset & -2 & 1 & 2.5 \\ \emptyset & \emptyset & 3 & -1 \\ \emptyset & \emptyset & \emptyset & 0.75 \end{bmatrix}$$

$$\begin{bmatrix} 2 & 3 & 3 \\ 2 & 6 & 2 \\ 3 & 4 & 2 \end{bmatrix} - \begin{bmatrix} 4 & 2 & 0.5 \\ 4 & 2 & 0.5 \\ 4 & 2 & 0.5 \end{bmatrix} = \begin{bmatrix} -2 & 1 & 2.5 \\ -2 & 4 & 1.5 \\ -1 & 2 & 1.5 \end{bmatrix} \rightarrow \text{do it again!}$$

$$\begin{bmatrix} 4 & 1.5 \\ 2 & 1.5 \end{bmatrix} - \begin{bmatrix} 1 & 2.5 \\ 0.5 & 1.25 \end{bmatrix} = \begin{bmatrix} 3 & -1 \\ 1.5 & 0.25 \end{bmatrix} \rightarrow \text{do it again!}$$

$$0.25 - (-0.5) = 0.75$$

Example

$$M = \begin{pmatrix} 2 & 8 & 4 & 1 \\ 1 & 2 & 3 & 3 \\ 1 & 2 & 6 & 2 \\ 1 & 3 & 4 & 2 \end{pmatrix} \quad L = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0.5 & 0 & 0 & 0 \\ 0.5 & 0 & 0 & 0 \\ 0.5 & 0 & 0 & 0 \end{pmatrix} \quad U = \begin{pmatrix} 2 & 8 & 4 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

$$M = \begin{pmatrix} 2 & 8 & 4 & 1 \\ 1 & -2 & 1 & 2.5 \\ 1 & -2 & 4 & 1.5 \\ 1 & -1 & 2 & 1.5 \end{pmatrix} \quad L = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0.5 & 1 & 0 & 0 \\ 0.5 & 1 & 0 & 0 \\ 0.5 & 0.5 & 0 & 0 \end{pmatrix} \quad U = \begin{pmatrix} 2 & 8 & 4 & 1 \\ 0 & -2 & 1 & 2.5 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

$$M = \begin{pmatrix} 2 & 8 & 4 & 1 \\ 1 & -2 & 1 & 2.5 \\ 1 & -2 & 3 & -1 \\ 1 & -1 & 1.5 & 0.25 \end{pmatrix} \quad L = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0.5 & 1 & 0 & 0 \\ 0.5 & 1 & 1 & 0 \\ 0.5 & 0.5 & 0.5 & 0 \end{pmatrix} \quad U = \begin{pmatrix} 2 & 8 & 4 & 1 \\ 0 & -2 & 1 & 2.5 \\ 0 & 0 & 3 & -1 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

$$M = \begin{pmatrix} 2 & 8 & 4 & 1 \\ 1 & -2 & 1 & 2.5 \\ 1 & -2 & 3 & -1 \\ 1 & -1 & 1.5 & 0.75 \end{pmatrix} \quad L = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0.5 & 1 & 0 & 0 \\ 0.5 & 1 & 1 & 0 \\ 0.5 & 0.5 & 0.5 & 1 \end{pmatrix} \quad U = \begin{pmatrix} 2 & 8 & 4 & 1 \\ 0 & -2 & 1 & 2.5 \\ 0 & 0 & 3 & -1 \\ 0 & 0 & 0 & 0.75 \end{pmatrix}$$

LU Algorithm

```
A = np.array([[2.0,8,4,1],[1,2,3,3],[1,2,6,2],[1,3,4,2]])
```

```
## Algorithm 1
## Factorization using the block-format,
## creating new matrices L and U
## and not modifying A
print("LU factorization using Algorithm 1")
L = np.zeros((n,n))
U = np.zeros((n,n))
M = A.copy()
for i in range(n):
    U[i,i:] = M[i,i:]
    L[i:,i] = M[i:,i]/U[i,i]
    M[i+1:,i+1:] -= np.outer(L[i+1:,i],U[i,i+1:])
```

```
## Algorithm 2
## Factorization using the block-format
## Matrices L and U are stored in the input matrix
## that could be a copy of A or A itself
print("LU factorization using Algorithm 2")
M = A.copy()
for i in range(n):
    M[i+1:,i] = M[i+1:,i]/M[i,i]
    M[i+1:,i+1:] -= np.outer(M[i+1:,i],M[i,i+1:])
```

Demo “LU-Factorization”

LU Algorithm

```
A = np.array([[2.0,8,4,1],[1,2,3,3],[1,2,6,2],[1,3,4,2]])
```

```
## Algorithm 2
## Factorization using the block-format
## Matrices L and U are stored in the input matrix
## that could be a copy of A or A itself
print("LU factorization using Algorithm 2")
M = A.copy()
for i in range(n):
    M[i+1:,i] = M[i+1:,i]/M[i,i]
    M[i+1:,i+1:] -= np.outer(M[i+1:,i],M[i,i+1:])
```

For $j = i + 1 : n - 1$

$$M[j,i] = M[j,i]/M[i,i]$$

For $j = i + 1 : n - 1$

For $k = i + 1 : n - 1$

$$M[j,k] -= M[j,i] * M[i,k]$$

```
M = A.copy()
for i in range(n-1):
    for j in range(i+1,n):
        M[j,i] = M[j,i]/M[i,i]
        for k in range(i+1,n):
            M[j,k] -= M[j,i]*M[i,k]
```

Cost of LU factorization

```
## Algorithm 1
## Factorization using the block-format,
## creating new matrices L and U
## and not modifying A
print("LU factorization using Algorithm 1")
L = np.zeros((n,n))
U = np.zeros((n,n))
M = A.copy()
for i in range(n):
    U[i,i:] = M[i,i:]
    L[i:,i] = M[i:,i]/U[i,i]
    M[i+1:,i+1:] -= np.outer(L[i+1:,i],U[i,i+1:])
```

Side note:

$$\sum_{i=1}^{m} i = \frac{1}{2}m(m+1)$$

$$\sum_{i=1}^{m} i^2 = \frac{1}{6}m(m+1)(2m+1)$$

Number of divisions: $(n-1)+(n-2)+\cdots+1 = n(n-1)/2$

Number of multiplications $(n-1)^2+(n-2)^2+\ldots+(1)^2 = \frac{n^3}{3}-\frac{n^2}{2}+\frac{n}{6}$

Number of subtractions: $(n-1)^2+(n-2)^2+\ldots+(1)^2 = \frac{n^3}{3}-\frac{n^2}{2}+\frac{n}{6}$

Computational complexity is $O(n^3)$

Iclicker question

Which of the following statements are true about the LU factorization of an $n \times n$ matrix A, assuming LU factorization of A exists and not considering any row/column interchanges?

Select all that apply:

1) $A = LU$. True
2) LU factorization is exactly performing Gaussian elimination. True
3) We can solve for $LUx = b$ instead of solving $Ax = b$ to obtain x. True
4) L is a lower triangular matrix, and is exactly the lower part of A but with unit diagonal. False
5) U is an upper triangular matrix, and is exactly the upper part of A (including diagonal). False

A) 1,2,3
B) 1,2,3,5
C) 1,3
D) 1,2,3,4,5
E) 4,5

Solving linear systems

In general, we can solve a linear system of equations following the steps:

1) Factorize the matrix $\boldsymbol{A}$: $\boldsymbol{A = LU}$ (complexity $O(n^3)$)

2) Solve $\boldsymbol{L\,y = b}$ (complexity $O(n^2)$)

3) Solve $\boldsymbol{U\,x = y}$ (complexity $O(n^2)$)

But why should we decouple the factorization from the actual solve? (Remember from Linear Algebra, Gaussian Elimination does not decouple these two steps…)

Iclicker question

Let's assume that when solving the system of equations $\boldsymbol{K}\,\boldsymbol{U} = \boldsymbol{F}$, we observe the following:

- When the matrix $\boldsymbol{K}$ has dimensions (100,100), computing the LU factorization takes about 1 second and each solve (forward + backward substitution) takes about 0.01 seconds.

Estimate the total time it will take to find the response $\boldsymbol{U}$ corresponding to 10 different vectors $\boldsymbol{F}$ when the matrix $\boldsymbol{K}$ has dimensions (1000,1000)?

$A)$ ~10 *seconds*
$B)$ ~10^2 *seconds*
$C)$ ~10^3 *seconds*
$D)$ ~10^4 *seconds*
$E)$ ~10^5 *seconds*

$n = 10^2 \rightarrow$ 1 sec | 0.01 s
$n = 10^3 \rightarrow$? | ?

LU factorization

LU $t = \alpha n^3 \longrightarrow \frac{1\text{ sec}}{t_1} = \left(\frac{10^2}{10^3}\right)^3 \rightarrow \boxed{t_1 = 10^3 \text{ sec}}$

solve $t = \beta n^2 \longrightarrow \frac{t_2}{10^{-2}\text{ sec}} = \left(\frac{10^3}{10^2}\right)^2 \rightarrow \boxed{t_2 = 1\text{ sec}}$ each

TOTAL $= 10^3 + 10$

What can go wrong with the previous algorithm?

Demo "Little c"

$$\boldsymbol{M} = \begin{pmatrix} 2 & 8 & 4 & 1 \\ 1 & \mathbf{4} & 3 & 3 \\ 1 & 2 & 6 & 2 \\ 1 & 3 & 4 & 2 \end{pmatrix} \quad \boldsymbol{L} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0.5 & 0 & 0 & 0 \\ 0.5 & 0 & 0 & 0 \\ 0.5 & 0 & 0 & 0 \end{pmatrix} \quad \boldsymbol{U} = \begin{pmatrix} 2 & 8 & 4 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

$$\boldsymbol{l}_{21}\boldsymbol{u}_{12} = \begin{pmatrix} 4 & 2 & 0.5 \\ 4 & 2 & 0.5 \\ 4 & 2 & 0.5 \end{pmatrix} \quad \boldsymbol{M} - \boldsymbol{l}_{21}\boldsymbol{u}_{12} = \begin{pmatrix} 2 & 8 & 4 & 1 \\ 1 & \mathbf{0} & 1 & 2.5 \\ 1 & -2 & 4 & 1.5 \\ 1 & -1 & 2 & 1.5 \end{pmatrix}$$

The next update for the lower triangular matrix will result in a division by zero! LU factorization fails.

What can we do to get something like an LU factorization?

Pivoting

Approach:

1. Swap rows if there is a zero entry in the diagonal
2. Even better idea: Find the largest entry (by absolute value) and swap it to the top row.

The entry we divide by is called the pivot.

Swapping rows to get a bigger pivot is called (partial) pivoting.

$$\begin{pmatrix} a_{11} & \boldsymbol{a_{12}} \\ \boldsymbol{a_{21}} & \boldsymbol{A_{22}} \end{pmatrix} = \begin{pmatrix} u_{11} & \boldsymbol{u_{12}} \\ u_{11}\,\boldsymbol{l_{21}} & \boldsymbol{l_{21}u_{12} + L_{22}U_{22}} \end{pmatrix}$$

Find the largest entry (in magnitude)