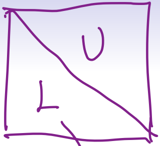
LU with Partial Pivoting



not including diagonal ($=1$)

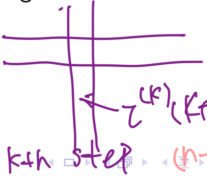
Find $\mathbf{L}, \mathbf{U}$ s.t. $\mathbf{PA} = \mathbf{LU}$ in MATLAB

```
for k=1:n-1
     [~ ,piv]=max(abs(A(k:n,k))); piv=piv-1+k;
     A([k piv],:)=A([piv k],:)  % swap row k and row piv
% If A(k,k)=0, then nothing to do
     if A(k,k)~ =0
         rows=k+1:n
         A(rows,k)=A(rows,k)/A(k,k) % compute τ^(k)(k+1:n)
         A(rows,rows)=A(rows,rows)-A(rows,k)*A(k,rows)
    end
end
```

In the above code, $A(k, k:n)$ represents $\mathbf{U}(k, k:n)$ and $A(k+1:n, k)$ represents $\mathbf{L}(k+1:n, k)$ (We already know the diagonal entries of $\mathbf{L}$ are 1)

$\mathcal{O}(n^2)$ comparisons for searching for the pivots

$\mathcal{O}(2n^3/3)$ flops



$\tau^{(k)}(k+1:n) \to \tilde{\tau}^{(k)}(k+1:n)$

kth step (n-1)th step

LU with Complete Pivoting

Complete Pivoting: Permute the largest entry of $A^{(k-1)}(k:n, k:n)$ in absolute value into the (k,k)-entry

- Require both row and column swaps

$$
\begin{aligned}
&(rowpiv(k), colpiv(k)) = \arg\max_{(i,j)\in[k,n]\times[k,n]} |\mathbf{A}^{(k-1)}(i,j)| \\
&\mathbf{A}^{(k-1)}(k, 1:n) \leftrightarrow \mathbf{A}^{(k-1)}(rowpiv(k), 1:n) \\
&\mathbf{A}^{(k-1)}(1:n, k) \leftrightarrow \mathbf{A}^{(k-1)}(1:n, colpiv(k))
\end{aligned}
$$

Then apply Gauss Transform to obtain $\mathbf{A}^{(k)}$ s.t. $\mathbf{A}^{(k)}(k+1:n, k) = \mathbf{0}$

The above Upper Triangularization gives

$$
\begin{aligned}
&\mathbf{A}^{(k)} = \mathbf{M}_k \mathbf{\Pi}_k \mathbf{A}^{(k-1)} \Gamma_k = \mathbf{M}_k \mathbf{\Pi}_k \cdots \mathbf{M}_1 \mathbf{\Pi}_1 \mathbf{A} \mathbf{\Gamma}_1 \cdots \mathbf{\Gamma}_k, \quad k = 1, \ldots, n-1 \\
&\mathbf{A}^{(n-1)} = \mathbf{U}
\end{aligned}
$$

LU with Complete Pivoting

- $\mathcal{O}(n^3)$ comparisons and $\mathcal{O}(\frac{2}{3}n^3)$ flops
  - Much more costly than partial pivoting
  - But lead to much smaller bound on growth factor, which reflects the safety of applying Gaussian elimination (cf. Section 3.4.5 in textbook)
- $\mathbf{P}\mathbf{A}\mathbf{Q}^T = \mathbf{L}\mathbf{U}$
  - $\mathbf{P} = \mathbf{\Pi}_{n-1} \cdots \mathbf{\Pi}_1$, where $\mathbf{\Pi}_k$ interchanges row k and row $rowpiv(k)$ of $\mathbf{I}$
  - $\mathbf{Q} = \mathbf{\Gamma}_{n-1} \cdots \mathbf{\Gamma}_1$, where $\mathbf{\Gamma}_k$ interchanges row k and row $colpiv(k)$ of $\mathbf{I}$
  - $\mathbf{U}$ is upper triangular, $\mathbf{L}$ is unit lower triangular with $|\ell_{ij}| \leq 1$

Solving Linear System via LU with Pivoting

$$\Rightarrow A = P^{-1} L U Q^{-T}$$

Solve $\mathbf{Ax} = \mathbf{b}$ using $\mathbf{PAQ}^T = \mathbf{LU}$

1. Solve $\mathbf{Lz} = \mathbf{Pb}$ for $\mathbf{z}$ (Forward Substitution $\mathcal{O}(n^2)$)
2. Solve $\mathbf{Uy} = \mathbf{z}$ for $\mathbf{y}$ (Back Substitution $\mathcal{O}(n^2)$)
3. Set $\mathbf{x} = \mathbf{Q}^T y$

$$Ax = b$$

$$P^{-1} L U Q^{-T} x = b$$

$$L \underbrace{U Q^{-T} x}_{z} = Pb$$

$$U \underbrace{Q^{-T} x}_{y} = z$$

$\mathbf{Q} = \mathbf{I}$ for partial pivoting

no need for step 3

Discussion

- When you call `lu(A)` or `A\b` in MATLAB, it always performs pivoting

- Apart from solving linear systems, LU decomposition is also used to
  - Compute $\mathbf{A}^{-1}$ (solve n linear systems): let $\mathbf{B} = \mathbf{A}^{-1}$

$$\mathbf{AB} = \mathbf{I} \quad \Longleftrightarrow \quad \mathbf{Ab}_i = \mathbf{e}_i, \; i = 1, \ldots, n$$

  - Compute $\det(\mathbf{A})$:

$$\det(\mathbf{A}) = \det(\mathbf{L})\det(\mathbf{U}) = \prod_{i=1}^{n} u_{ii}$$

- Another way of pivoting: Let the pivot be the element in $A^{(k-1)}(k:n, k:n)$ that has the maximal absolute value in both its row and its column (Rook Pivoting)

Matrix Computations
Chapter 2 Linear systems and LU decomposition
Section 2.3 Special Linear Systems and Other Decompositions

Jie Lu
ShanghaiTech University

LDM Decomposition

Given $\mathbf{A} \in \mathbb{R}^{n\times n}$, find matrices $\mathbf{L}, \mathbf{D}, \mathbf{M} \in \mathbb{R}^{n\times n}$ such that

$$\mathbf{A} = \mathbf{L}\mathbf{D}\mathbf{M}^T \qquad \text{(LDM decomposition)}$$

where

$\mathbf{L}$ is **unit** lower triangular

$\mathbf{D} = \mathrm{Diag}(d_1, \ldots, d_n)$

$\mathbf{M}$ is **unit** lower triangular $\quad M^T$ unit upper triangular

If $\mathbf{A} = \mathbf{L}\mathbf{U}$ is an LU decomposition, then the LDM decomposition uses the same $\mathbf{L}$ and sets

$$\mathbf{D} = \mathrm{Diag}(u_{11}, \ldots, u_{nn}), \qquad \mathbf{M} = \mathbf{U}^T\mathbf{D}^{-1} \quad \text{if } D^{-1} \text{ exists}$$

$$LDM^T = LD(U^TD^{-1})^T = LDD^{-T}U = LDD^{-1}U = LU$$

The existence of LDM decomposition follows that of LU decomposition

Solving LDM Decomposition

Examine $\mathbf{A} = \mathbf{LDM}^T$ column by column. For each $j = 1, \ldots, n$,

$$\mathbf{A}(:,j) = \mathbf{Ae}_j = \mathbf{Lv}$$
$$\mathbf{v} = \mathbf{DM}^T\mathbf{e}_j$$

Example: Let n=4 and find $\mathbf{v}$ with $j = 3$

$$D = \mathrm{Diag}(d_1, d_2, d_3, d_4) \qquad M^T = \begin{bmatrix} 1 & m_{21} & m_{31} & m_{41} \\ & 1 & m_{32} & m_{42} \\ & & 1 & m_{43} \\ & & & 1 \end{bmatrix}$$

$$v = DM^T e_3 = \begin{bmatrix} d_1 & & & \\ & d_2 & & \\ & & d_3 & \\ & & & d_4 \end{bmatrix} \begin{bmatrix} 1 & m_{21} & m_{31} & m_{41} \\ 0 & 1 & m_{32} & m_{42} \\ 0 & 0 & 1 & m_{43} \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}$$

$$= \begin{bmatrix} d_1 & d_1 m_{21} & d_1 m_{31} & d_1 m_{41} \\ 0 & d_2 & d_2 m_{32} & d_2 m_{42} \\ 0 & 0 & d_3 & d_3 m_{43} \\ 0 & 0 & 0 & d_4 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix} = \begin{bmatrix} d_1 m_{31} \\ d_2 m_{32} \\ d_3 \\ 0 \end{bmatrix}$$

Solving LDM Decomposition (cont'd)

$v = DM^T e_j$

Observations: For $i, j = 1, \ldots, n$,

$$v_i = d_i m_{ji}$$

ith entry of v

- For $i \geq j+1$, $v_i = 0$ because $m_{ji} = 0$
- For $i = j$, $v_j = d_j$ because $m_{jj} = 1$

Therefore, $\mathbf{A}(:, j) = \mathbf{L}\mathbf{v}$ can be partitioned as

unit lower triangular

$$\begin{bmatrix} \mathbf{A}(1:j, j) \\ \mathbf{A}(j+1:n, j) \end{bmatrix} = \begin{bmatrix} \mathbf{L}(1:j, 1:j) & \mathbf{0} \\ \mathbf{L}(j+1:n, 1:j) & \mathbf{L}(j+1:n, j+1:n) \end{bmatrix} \begin{bmatrix} \mathbf{v}(1:j) \\ \mathbf{0} \end{bmatrix}$$

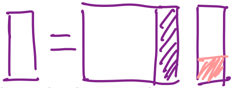
$$= \begin{bmatrix} \mathbf{L}(1:j, 1:j)\mathbf{v}(1:j) \\ \mathbf{L}(j+1:n, 1:j)\mathbf{v}(1:j) \end{bmatrix}$$

the first j columns of L

Solving LDM Decomposition (cont'd)

It follows from the above equation that

$$\text{1}\quad \mathbf{A}(1:j,j) = \mathbf{L}(1:j,1:j)\mathbf{v}(1:j)$$
$$\text{2}\quad \mathbf{A}(j+1:n,j) = \mathbf{L}(j+1:n,1:j)\mathbf{v}(1:j)$$



Idea: Recursively find each column of **L**, each row of **M**, and each diagonal entry of **D**

For $j = 1 : n$

no need for $j=1$

Step 1. Form $\mathbf{L}(1:j,1:j)$ using the columns $1,\ldots,j-1$ of **L** and $\mathbf{L}(j,j) = 1$

Step 2. Solve the ~~linear~~ triangular system $\mathbf{A}(1:j,j) = \mathbf{L}(1:j,1:j)\mathbf{v}(1:j)$ for $\mathbf{v}(1:j)$

Step 3. Compute $\mathbf{L}(j+1:n,j)$ according to (not needed for $j = n$) 2

$$\mathbf{L}(j+1:n,j) = (\mathbf{A}(j+1:n,j) - \mathbf{L}(j+1:n,1:j-1)\mathbf{v}(1:j-1))/\mathbf{v}(j)$$

Step 4. Set $d_j = v_j$, $m_{ji} = v_i/d_i$ for all $i = 1,\ldots,j-1$

$d_1, \ldots, d_{j-1}$ obtained from previous iterations

% Recall that $\mathbf{L}(1:j,j) = \begin{bmatrix}0 & \cdots & 0 & 1\end{bmatrix}^T$ and $\mathbf{M}(j,j:n) = \begin{bmatrix}1 & 0 & \cdots & 0\end{bmatrix}$

LDM Code

```
function [L,D,M]= LDMdecomposition(A)
n= size(A,1);
L= eye(n); d= zeros(n,1); M= eye(n);
v= zeros(n,1);
for j=1:n
```

Step 2

```
     v(1:j)= ForwardSubstitution(L(1:j,1:j),A(1:j,j));
     % solve A(1:j,j) = L(1:j,1:j)v(1:j) using forward
substitution
     d(j)= v(j);
     for i=1:j-1,
         M(j,i)= v(i)'/d(i);
     end;
```

Step 4

```
     L(j+1:n,j)= (A(j+1:n,j)-L(j+1:n,1:j-1)*v(1:j-1))/v(j);
end;
```

Step 3

```
D= diag(d);
```

- Complexity: $\mathcal{O}(2n^3/3)$ (same as the previous LU code)

LDL Decomposition for Symmetric Matrices

For any real symmetric matrix $\mathbf{A}$, i.e., $\mathbf{A} \in \mathbb{S}^{n \times n}$,

$$\mathbf{A} = \mathbf{L}\mathbf{D}\mathbf{L}^T \qquad \text{(LDL decomposition)}$$

where $\mathbf{L}$ is **unit** lower triangular and $\mathbf{D} = \mathrm{Diag}(d_1, \ldots, d_n)$

Theorem

If $\mathbf{A} \in \mathbb{S}^{n \times n}$ is nonsingular, then its LDL decomposition is unique. In addition, if $\mathbf{A} = \mathbf{L}\mathbf{D}\mathbf{M}^T$ is the LDM decomposition, then $\mathbf{L} = \mathbf{M}$.

Proof: From the LU decomposition Theorem, A has a unique LU decomposition $A = LU$.

$$L^{-1} A L^{-T} = \underbrace{L^{-1} L}_{} U L^{-T} = U L^{-T}$$

Symmetric ↑ upper triangular

$$(L^{-1} A L^{-T})^T = L^{-1} A^T L^{-T} = L^{-1} A L^{-T}$$

Therefore, UL^{-T} is diagonal. Let $D = UL^{-T}$.

$$A = LU = L(UL^{-T})L^T = LDL^T$$

unique

Solving LDL Decomposition

- In solving LDM decomposition, the key is to solve $\mathbf{A}(1:j,j) = \mathbf{L}(1:j,1:j)\mathbf{v}(1:j)$ for

$$\mathbf{v} = \mathbf{D}\mathbf{M}^T\mathbf{e}_j \;\Rightarrow\; v_i = d_i m_{ji}$$

  via forward substitution

- Now for LDL decomposition, we have $\mathbf{M} = \mathbf{L}$

$$v_i = d_i \ell_{ji}$$

- Finding $\mathbf{v}$ is much easier and no need for forward substitution
  - With the knowledge of the columns $1, \ldots, j-1$ of $\mathbf{L}$, we can easily find $v_i = d_i \ell_{ji}$, $i = 1, \ldots, j-1$
  - Then, find v_j by $v_j = \mathbf{A}(j,j) - \mathbf{L}(j, 1:j-1) * \mathbf{v}(1:j-1)$

LDL Code

```
function [L,D]= LDLdecomposition(A)
n= size(A,1);
L= eye(n); d= zeros(n,1); M= eye(n);
v= zeros(n,1);
for j=1:n
    v(1:j)= ForwardSubstitution(L(1:j,1:j),A(1:j,j));
    v(1:j-1)= L(j,1:j-1)'.*d(1:j-1);
    v(j)= A(j,j)- L(j,1:j-1)*v(1:j-1);
    d(j)= v(j);
    for i=1:j-1,
        M(j,i)= v(i)'/d(i);
    end;
    L(j+1:n,j)= (A(j+1:n,j)-L(j+1:n,1:j-1)*v(1:j-1))/v(j);
end;
D= diag(d);
```

new

- Complexity: $\mathcal{O}(n^3/3)$, half of LU or LDM