SI231 Matrix Analysis and Computations
Topic 2: Linear Systems

Ziping Zhao

Spring Term 2020–2021

School of Information Science and Technology
ShanghaiTech University, Shanghai, China

Topic 2: Linear Systems

- direct methods for general linear systems
- direct methods for special (structured) linear systems
- iterative methods for linear systems
- other topics on linear systems

Main Results

- a matrix $\mathbf{A} \in \mathbb{R}^{n \times n}$ is said to have an LU decomposition/factorization if it can be factored as
$$\mathbf{A} = \mathbf{L}\mathbf{U},$$
where $\mathbf{L} \in \mathbb{R}^{n \times n}$ is lower triangular; $\mathbf{U} \in \mathbb{R}^{n \times n}$ is upper triangular
  - does not always exist
  - pivoting: there exists a permutation matrix $\mathbf{P} \in \mathbb{R}^{n \times n}$ such that $\mathbf{PA} = \mathbf{LU}$

- LDL decomposition/factorization: if $\mathbf{A} \in \mathbb{S}^{n \times n}$ has an LU decomposition, then $\mathbf{U} = \mathbf{D}\mathbf{L}^T$ where $\mathbf{D}$ is diagonal

- Cholesky decomposition/factorization: if $\mathbf{A} \in \mathbb{S}^{n \times n}$ is PD, it can always be factored as
$$\mathbf{A} = \mathbf{G}\mathbf{G}^T,$$
where $\mathbf{G}$ is lower triangular.

The System of Linear Equations

Consider the system of linear equations

$$\mathbf{A}\mathbf{x} = \mathbf{b},$$

where $\mathbf{A} \in \mathbb{R}^{n \times n}$ and $\mathbf{b} \in \mathbb{R}^n$ are given, and $\mathbf{x} \in \mathbb{R}^n$ is the solution to the system.

- a linear square system
- $\mathbf{A}$ will be assumed to be nonsingular (unless specified)
- we consider the real case for convenience; extension to the complex case is simple
  - if $\mathbf{A}$ is real and $\mathbf{b}$ is complex
    - get the real and complex part of the solution separately
  - if $\mathbf{A}$ is complex
    - rewrite decomposition routine to use complex arithmetic (more complicated code, fewer operations)
    - solve real and imaginary parts of matrix separately (utilizes same code, costs twice as many operations/storage space)

Solving the Linear System

Problem: compute the solution to $\mathbf{Ax} = \mathbf{b}$ in a numerically efficient manner.

- the problem is easy if $\mathbf{A}^{-1}$ is known
  - but computing $\mathbf{A}^{-1}$ also costs computations...
  - do you know how to compute $\mathbf{A}^{-1}$ efficiently?
- here, $\mathbf{A}$ is assumed to be a general nonsingular matrix.
  - the problem may become easy in some special cases, e.g., diagonal $\mathbf{A}$, lower triangular $\mathbf{A}$, upper triangular $\mathbf{A}$, orthogonal $\mathbf{A}$, permutation matrices $\mathbf{A}$, Toeplitz $\mathbf{A}$, circulant $\mathbf{A}$, sparse $\mathbf{A}$.

Solving Some "Easy" Linear Systems

- diagonal matrices $\mathbf{A}$ ($a_{ij} = 0$ if $i \neq j$): n flops

$$\mathbf{x} = \mathbf{A}^{-1}\mathbf{b} = [b_1/a_{11}, \ldots, b_n/a_{nn},]$$

- lower triangular matrices $\mathbf{A}$ ($a_{ij} = 0$ if $i < j$): n^2 flops with forward substitution
- upper triangular matrices $\mathbf{A}$ ($a_{ij} = 0$ if $i > j$): n^2 flops with backward substitution
- orthogonal matrices $\mathbf{A}^{-1} = \mathbf{A}^T$
  - compute $\mathbf{x} = \mathbf{A}^T\mathbf{b}$ for general $\mathbf{A}$ in $2n^2$ flops
  - less with structure, e.g., if $\mathbf{A} = \mathbf{I} - 2\mathbf{a}\mathbf{a}^T$ with $\|\mathbf{a}\|^2 = 1$, we can compute $\mathbf{x} = \mathbf{A}^T\mathbf{b} = \mathbf{b} - 2(\mathbf{a}^T\mathbf{b})\mathbf{a}$ in $4n$ flops
- permutation matrices $\mathbf{A}^{-1} = \mathbf{A}^T$
  Example:

$$\mathbf{A} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix}, \qquad \mathbf{A}^{-1} = \mathbf{A}^T = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}$$

  compute $\mathbf{x} = \mathbf{A}^T\mathbf{b}$ in 0 flops

Direct Methods for General Linear Systems

LU Decomposition

LU decomposition: given $\mathbf{A} \in \mathbb{R}^{n\times n}$, find two matrices $\mathbf{L}, \mathbf{U} \in \mathbb{R}^{n\times n}$ such that

$$\mathbf{A} = \mathbf{L}\mathbf{U},$$

where

$\mathbf{L} \in \mathbb{R}^{n\times n}$ is unit lower triangular (lower triangular with unit diagonal elements (i.e., $\ell_{ii} = 1$ for all i));
$\mathbf{U} \in \mathbb{R}^{n\times n}$ is upper triangular.

Idea: **Suppose** that $\mathbf{A}$ has an LU decomposition. Then, solving $\mathbf{A}\mathbf{x} = \mathbf{b}$ can be recast as two linear system problems:

1. solve $\mathbf{L}\mathbf{z} = \mathbf{b}$ for $\mathbf{z}$, and then
2. solve $\mathbf{U}\mathbf{x} = \mathbf{z}$ for $\mathbf{x}$.

Questions:

1. how to solve $\mathbf{L}\mathbf{z} = \mathbf{b}$, and then $\mathbf{U}\mathbf{x} = \mathbf{z}$?
2. how to perform $\mathbf{A} = \mathbf{L}\mathbf{U}$? Does LU decomposition exist?

Forward Substitution

Example: a 3×3 lower triangular system $\mathbf{Lz} = \mathbf{b}$

$$\begin{bmatrix} \ell_{11} & 0 & 0 \\ \ell_{21} & \ell_{22} & 0 \\ \ell_{31} & \ell_{32} & \ell_{33} \end{bmatrix} \begin{bmatrix} z_1 \\ z_2 \\ z_3 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix}.$$

If $\ell_{11}, \ell_{22}, \ell_{33} \neq 0$, then z_1, z_2, z_3 can be solved by

$$\begin{aligned} z_1 &= b_1/\ell_{11} \\ z_2 &= (b_2 - \ell_{21}z_1)/\ell_{22} \\ z_3 &= (b_3 - \ell_{31}z_1 - \ell_{32}z_2)/\ell_{33} \end{aligned}$$

Forward Substitution

Forward substitution for solving $\mathbf{L}\mathbf{z} = \mathbf{b}$:

$$z_i = \left(b_i - \sum_{j=1}^{i-1} \ell_{ij} z_j \right) \Big/ \ell_{ii}, \qquad \text{for } i = 1, 2, \ldots, n.$$

Forward substitution in MATLAB form:

```
function z= for_subs(L,b)
n= length(b);
z= zeros(n,1);
z(1)= b(1)/L(1,1);
for i=2:1:n
     z(i)= (b(i)-L(i,1:i-1)*z(1:i-1))/L(i,i);
end;
```

- complexity: $\mathcal{O}(n^2)$ (n^2 multiplications/divisions $+\, n^2 - n$ additions/subtractions)

Backward Substitution

Example: a 3×3 upper triangular system $\mathbf{U}\mathbf{x} = \mathbf{z}$

$$\begin{bmatrix} u_{11} & u_{12} & u_{13} \\ 0 & u_{22} & u_{23} \\ 0 & 0 & u_{33} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} z_1 \\ z_2 \\ z_3 \end{bmatrix}.$$

If $u_{11}, u_{22}, u_{33} \neq 0$, then x_1, x_2, x_3 can be solved by, in sequence,

$$\begin{aligned} x_3 &= z_3/u_{33} \\ x_2 &= (z_2 - u_{23}x_3)/u_{22} \\ x_1 &= (z_1 - u_{12}x_2 - u_{13}x_3)/u_{11} \end{aligned}$$

Backward Substitution

Backward substitution for solving $\mathbf{U}\mathbf{x} = \mathbf{z}$:

$$x_i = \left(z_i - \sum_{j=i+1}^{n} u_{ij} x_j \right) \Big/ u_{ii}, \qquad \text{for } i = n, n-1, \ldots, 1.$$

Backward substitution in MATLAB form:

```
function x= back_subs(U,z)
n= length(z);
x= zeros(n,1);
x(n)= z(n)/U(n,n);
for i= n-1:-1:1,
     x(i)= ( z(i)- U(i,i+1:n)*x(i+1:n) )/U(i,i);
end;
```

- complexity: $\mathcal{O}(n^2)$ (n^2 multiplications/divisions $+\ n^2 - n$ additions/subtractions)

Gauss Transformations: the Key Building Block for LU

Observation: given $\mathbf{x} \in \mathbb{R}^n$ that has $x_k \neq 0$, $1 \leq k \leq n$,

$$\underbrace{\begin{bmatrix} 1 & & & & & \\ & \ddots & & & & \\ & & 1 & & & \\ & & -\frac{x_{k+1}}{x_k} & 1 & & \\ & & \vdots & & \ddots & \\ & & -\frac{x_n}{x_k} & & & 1 \end{bmatrix}}_{=\mathbf{M}} \begin{bmatrix} x_1 \\ \vdots \\ x_k \\ x_{k+1} \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} x_1 \\ \vdots \\ x_k \\ 0 \\ \vdots \\ 0 \end{bmatrix}.$$

The above $\mathbf{M}$ also satisfies

$$\mathbf{M}\mathbf{y} = \mathbf{y}, \quad \text{for any } \mathbf{y} = [\ y_1, \ldots, y_{k-1}, 0, \ldots, 0\ ]^T,\ y_i \in \mathbb{R}.$$

Characterization of a Gauss transformation $\mathbf{M}$ (an outer-product form):

$$\mathbf{M} = \mathbf{I} - \boldsymbol{\tau}\mathbf{e}_k^T, \qquad \boldsymbol{\tau} = [\ 0, \ldots, 0, x_{k+1}/x_k, \ldots, x_n/x_k\ ]^T.$$

where $\boldsymbol{\tau}$ is called Gauss vector with $x_{k+1}/x_k, \ldots, x_n/x_k$ called multipliers.

Finding $\mathbf{U}$ by Gauss Elimination

Problem: find Gauss transformations $\mathbf{M}_1, \ldots, \mathbf{M}_{n-1} \in \mathbb{R}^{n \times n}$ such that

$$\mathbf{M}_{n-1} \cdots \mathbf{M}_2 \mathbf{M}_1 \mathbf{A} = \mathbf{U}, \quad \mathbf{U} \text{ being upper triangular.}$$

Step 1: choose $\mathbf{M}_1$ such that $\mathbf{M}_1 \mathbf{a}_1 = [\ a_{11}, 0, \ldots, 0\ ]^T$

- **if** $a_{11} \neq 0$, then we can choose

$$\mathbf{M}_1 = \mathbf{I} - \boldsymbol{\tau}^{(1)} \mathbf{e}_1^T, \qquad \boldsymbol{\tau}^{(1)} = [\ 0, a_{21}/a_{11}, \ldots, a_{n1}/a_{11}\ ]^T.$$

- result:

$$\mathbf{M}_1 \mathbf{A} = \mathbf{A} - \boldsymbol{\tau}^{(1)} \mathbf{e}_1^T \mathbf{A} = \begin{bmatrix} a_{11} & \times & \ldots & \times \\ 0 & \times & \ldots & \times \\ \vdots & \vdots & & \vdots \\ 0 & \times & \ldots & \times \end{bmatrix}$$

Finding $\mathbf{U}$ by Gauss Elimination

Step 2: let $\mathbf{A}^{(1)} = \mathbf{M}_1\mathbf{A}$. Choose $\mathbf{M}_2$ such that $\mathbf{M}_2\mathbf{a}_2^{(1)} = [\ a_{12}^{(1)}, a_{22}^{(1)}, 0, \ldots, 0\ ]^T$.

- **if** $a_{22}^{(1)} \neq 0$, then we can choose

$$\mathbf{M}_2 = \mathbf{I} - \boldsymbol{\tau}^{(2)}\mathbf{e}_2^T, \qquad \boldsymbol{\tau}^{(2)} = [\ 0, 0, a_{32}^{(1)}/a_{22}^{(1)}, \ldots, a_{n,2}^{(1)}/a_{22}^{(1)}\ ]^T.$$

- result:

$$\mathbf{M}_2\mathbf{A}^{(1)} = \mathbf{A} - \boldsymbol{\tau}^{(2)}\mathbf{e}_2^T\mathbf{A} = \begin{bmatrix} a_{11}^{(1)} & a_{12}^{(1)} & \times & \ldots & \times \\ 0 & a_{22}^{(1)} & \times & \ldots & \times \\ \vdots & 0 & \times & & \times \\ \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & \times & \ldots & \times \end{bmatrix}$$

Finding $\mathbf{U}$ by Gauss Elimination

Let $\mathbf{A}^{(k)} = \mathbf{M}_k \mathbf{A}^{(k-1)}$, $\mathbf{A}^{(0)} = \mathbf{A}$. Note $\mathbf{A}^{(k)} = \mathbf{M}_k \cdots \mathbf{M}_2 \mathbf{M}_1 \mathbf{A}$.

Step k: Choose $\mathbf{M}_k$ such that $\mathbf{M}_k \mathbf{a}_k^{(k-1)} = [\, a_{1k}^{(k-1)}, \ldots, a_{kk}^{(k-1)}, 0, \ldots, 0 \,]^T$.

- **if** $a_{kk}^{(k-1)} \neq 0$, then

$$\mathbf{M}_k = \mathbf{I} - \boldsymbol{\tau}^{(k)} \mathbf{e}_k^T, \qquad \boldsymbol{\tau}^{(k)} = [\, 0, \ldots, 0, a_{k+1,k}^{(k-1)}/a_{kk}^{(k-1)}, \ldots, a_{n,k}^{(k-1)}/a_{kk}^{(k-1)} \,]^T,$$

- result:

$$\mathbf{A}^{(k)} = \mathbf{M}_k \mathbf{A}^{(k-1)} = \mathbf{A} - \boldsymbol{\tau}^{(k)} \mathbf{e}_k^T \mathbf{A} = \begin{bmatrix} a_{11}^{(k-1)} & \cdots & a_{1k}^{(k-1)} & \times & \cdots & \times \\ 0 & \ddots & \vdots & \vdots & & \vdots \\ \vdots & & a_{kk}^{(k-1)} & \vdots & & \vdots \\ \vdots & & 0 & \times & & \times \\ \vdots & & \vdots & \vdots & & \vdots \\ 0 & \cdots & 0 & \times & \cdots & \times \end{bmatrix}$$

  – $\mathbf{A}^{(n-1)} = \mathbf{U}$ is upper triangular

Where is $\mathbf{L}$?

We have seen that under the assumption of $a_{kk}^{(k-1)} \neq 0$ for all k,

$$\mathbf{U} = \mathbf{M}_{n-1} \cdots \mathbf{M}_2 \mathbf{M}_1 \mathbf{A} \text{ is upper triangular.}$$

But where is $\mathbf{L}$?

Property 2.1. Let $\mathbf{A}, \mathbf{B} \in \mathbb{R}^{n \times n}$ be lower triangular. Then, $\mathbf{AB}$ is lower triangular. Also, if $\mathbf{A}$, $\mathbf{B}$ have unit diagonal entries, then $\mathbf{AB}$ has unit diagonal entries.

Property 2.2. If $\mathbf{A} \in \mathbb{R}^{n \times n}$ is lower triangular, then $\det(\mathbf{A}) = \prod_{i=1}^{n} a_{ii}$.

Property 2.3. Let $\mathbf{A} \in \mathbb{R}^{n \times n}$ be nonsingular lower triangular. Then, $\mathbf{A}^{-1}$ is lower triangular with $[\mathbf{A}^{-1}]_{ii} = 1/a_{ii}$.

Suppose that every $\mathbf{M}_k$ is invertible. Then,

$$\mathbf{L} = \mathbf{M}_1^{-1} \mathbf{M}_2^{-1} \cdots \mathbf{M}_{n-1}^{-1}$$

satisfies $\mathbf{A} = \mathbf{LU}$, and is lower triangular with unit diagonal entries.

A Naive Implementation of LU (Don't Use It)

```
function [L,U]= my_naive_lu(A)
n= size(A,1);
L= eye(n); t= zeros(n,1); U= A;
for k=1:1:n-1,
      rows= k+1:n;
      t(rows)= U(rows,k)/U(k,k);
      M= eye(n); M(rows,k)= -t(rows);
      U= M*U;              % compute A^(k) = M_k A^(k-1)
      L= L*inv(M);          % to eventually obtain L = M_1^{-1} M_2^{-1} ... M_{n-1}^{-1}
end;
```

Weaknesses:

- the above code treats each $\mathbf{A}^{(k)} = \mathbf{M}_k \mathbf{A}^{(k-1)}$ as a general matrix multiplication process, which takes $\mathcal{O}(n^3)$ flops. It does not utilize structures of $\mathbf{M}_k$.
- (more serious) to compute $\mathbf{L}$, the above code calls inverse $n-1$ times. If the problem is to solve $\mathbf{A}\mathbf{x} = \mathbf{b}$, then why not just call inverse once for $\mathbf{A}$?

Computing $\mathbf{L}$

Fact: $\mathbf{M}_k^{-1} = \mathbf{I} + \boldsymbol{\tau}^{(k)}\mathbf{e}_k^T$.

Verification by definition: by noting $[\boldsymbol{\tau}^{(k)}]_k = 0$,

$$\begin{aligned}(\mathbf{I} + \boldsymbol{\tau}^{(k)}\mathbf{e}_k^T)\mathbf{M}_k &= (\mathbf{I} + \boldsymbol{\tau}^{(k)}\mathbf{e}_k^T)(\mathbf{I} - \boldsymbol{\tau}^{(k)}\mathbf{e}_k^T) \\ &= \mathbf{I} + \boldsymbol{\tau}^{(k)}\mathbf{e}_k^T - \boldsymbol{\tau}^{(k)}\mathbf{e}_k^T + \boldsymbol{\tau}^{(k)}\underbrace{\mathbf{e}_k^T\boldsymbol{\tau}^{(k)}}_{=0}\mathbf{e}_k^T = \mathbf{I}.\end{aligned}$$

can also be verified by matrix inversion lemma (cf. Topic 1)

By the same spirit ($\mathbf{e}_j^T\boldsymbol{\tau}^{(k)} = 0$ for $j \le k$), it can be verified that

$$\begin{aligned}\mathbf{L} &= \mathbf{M}_1^{-1}\mathbf{M}_2^{-1}\ldots\mathbf{M}_{n-1}^{-1} = \left(\mathbf{I} + \boldsymbol{\tau}^{(1)}\mathbf{e}_1^T\right)\left(\mathbf{I} + \boldsymbol{\tau}^{(2)}\mathbf{e}_2^T\right)\ldots\left(\mathbf{I} + \boldsymbol{\tau}^{(n-1)}\mathbf{e}_{(n-1)}^T\right) \\ &= \mathbf{I} + \boldsymbol{\tau}^{(1)}\mathbf{e}_1^T + \boldsymbol{\tau}^{(2)}\mathbf{e}_2^T \ldots + \boldsymbol{\tau}^{(n-1)}\mathbf{e}_{(n-1)}^T = \mathbf{I} + \sum_{k=1}^{n-1}\boldsymbol{\tau}^{(k)}\mathbf{e}_k^T\end{aligned}$$

A More Mature LU Code (Still Not the LU inside MATLAB)

```
function [L,U]= my_lu(A)
n= size(A,1);
L= eye(n); t= zeros(n,1); U= A;
for k=1:1:n-1,
     rows= k+1:n;
     t(rows)= U(rows,k)/U(k,k);
     U(rows,rows)= U(rows,rows)- t(rows)*U(k,rows);
     U(rows,k)= 0;
     L(rows,k)= t(rows);
end;
```

- complexity: $\mathcal{O}(2n^3/3)$

$$\sum_{k=1}^{n-1}\left(\sum_{\text{rows=k+1}}^{n} 1 + 2\sum_{\text{rows=k+1}}^{n}\sum_{\text{rows=k+1}}^{n} 1\right) = \sum_{k=1}^{n-1}\left(n-k+2(n-k)^2\right) = 2n^3/3 + \mathcal{O}(n^2)$$

- works as long as $a_{kk}^{(k-1)}$—the so-called **pivots**—are all nonzero

Existence and Uniqueness of LU Decomposition

Theorem 2.1. A matrix $\mathbf{A} \in \mathbb{R}^{n \times n}$ has an LU decomposition if every principal submatrix $\mathbf{A}_{\{1,\dots,k\}}$ satisfies

$$\det(\mathbf{A}_{\{1,\dots,k\}}) \neq 0,$$

for $k = 1, 2, \dots, n-1$. If the LU decomposition of $\mathbf{A}$ exists and $\mathbf{A}$ is nonsingular, then $(\mathbf{L}, \mathbf{U})$ is unique.

- the proof is essentially about when $a_{kk}^{(k-1)} \neq 0$.
- see Theorem 3.2.1 in **[Golub-van-Loan'13]**

Discussion

- the LU algorithm described above requires nonzero pivots, $a_{kk}^{(k-1)} \neq 0$ for all k.
- Gauss elimination is known to be numerically unstable when a pivot is close to zero, i.e., $\left|a_{kk}^{(k-1)}\right| \ll 1$

  Example (effect of roundoff error): Solve the linear system $\mathbf{Ax} = \mathbf{b}$ via LU decomposition, where

$$\mathbf{A} = \begin{bmatrix} 0.02 & 61.3 \\ 3.43 & -8.5 \end{bmatrix} \text{ and } \mathbf{b} = \begin{bmatrix} 61.5 \\ 25.8 \end{bmatrix}.$$

  Keep 3 significant figures.
- examine the main step in Gauss elimination

$$a_{ij}^{(k)} = -[\mathbf{M}_k]_{ik} a_{kj}^{(k-1)} + a_{ij}^{(k-1)}$$

  any roundoff error in the computation of $a_{kj}^{(k-1)}$ is amplified by multiplier $[\mathbf{M}_k]_{ik}$
- **pivoting:** to ensured that the multipliers are small, at each Gauss elimination step, interchange/permutate the rows of $\mathbf{A}^{(k)}$ to obtain better pivots.
  – when you call `lu(A)` or `A\b` in MATLAB, it always perform pivoting

LU Decomposition with Partial Pivoting

- **pivoting:** when eliminating elements in $\mathbf{a}_k^{(k-1)}$, find an integer p, $k \leq p \leq n$, s.t.

$$\left|a_{pk}^{(k-1)}\right| = \max_{k \leq i \leq n} \left|a_{ik}^{(k-1)}\right|.$$

  and then interchange rows p and k of $\mathbf{A}^{(k-1)}$.
- requires $\mathcal{O}(n^2)$ comparisons to determine the appropriate row interchanges
- $[\mathbf{M}_k]_{ik} = -a_{ik}/a_{kk}$, then $\left|[\mathbf{M}_k]_{ik}\right| \leq 1$ for $k = 1, \ldots, n-1$ and $i = k+1, \ldots, n$.

LU decomposition with partial pivoting: given $\mathbf{A} \in \mathbb{R}^{n \times n}$, find three matrices $\mathbf{L}, \mathbf{U}, \mathbf{P} \in \mathbb{R}^{n \times n}$ such that

$$\mathbf{PA} = \mathbf{LU}$$

where

$\mathbf{P} \in \mathbb{R}^{n \times n}$ is a permutation matrix;
$\mathbf{L} \in \mathbb{R}^{n \times n}$ is unit lower triangular with $|\ell_{ij}| \leq 1$;
$\mathbf{U} \in \mathbb{R}^{n \times n}$ is upper triangular.

Questions: how to perform $\mathbf{PA} = \mathbf{LU}$?

Finding $\mathbf{U}$ by Gauss Elimination with Partial Pivoting

Problem: find interchange permutations $\mathbf{\Pi}_1, \mathbf{\Pi}_2, \ldots, \mathbf{\Pi}_{n-1} \in \mathbb{R}^{n\times n}$ and Gauss transformations $\mathbf{M}_1, \mathbf{M}_2, \ldots, \mathbf{M}_{n-1} \in \mathbb{R}^{n\times n}$ such that

$$\mathbf{M}_{n-1}\mathbf{\Pi}_{n-1}\cdots\mathbf{M}_2\mathbf{\Pi}_2\mathbf{M}_1\mathbf{\Pi}_1\mathbf{A} = \mathbf{U}, \quad \mathbf{U} \text{ being upper triangular,}$$

and no multipliers in $\mathbf{M}_k$ for $k = 1, \ldots, n-1$ is larger than one in absolute value.

Where is $\mathbf{P}$ and Where is $\mathbf{L}$?

Fact: since each permutation matrix $\mathbf{\Pi}_k$ at most interchanges row k with row p, where $p > k$, there is no difference between applying all of the row interchanges "up front" and applying $\mathbf{\Pi}_k$ immediately before applying $\mathbf{M}_k$ for each k. It follows that

$$\tilde{\mathbf{M}}_{n-1} \cdots \tilde{\mathbf{M}}_2 \tilde{\mathbf{M}}_1 \mathbf{\Pi}_{n-1} \cdots \mathbf{\Pi}_2 \mathbf{\Pi}_1 \mathbf{A} = \mathbf{U}, \quad \mathbf{U} \text{ being upper triangular,} \qquad (\star)$$

where $\tilde{\mathbf{M}}_k$'s are new Gauss transformations related to $\mathbf{M}_k$.

From $(\star)$, we have

- $\mathbf{P} = \mathbf{\Pi}_{n-1} \cdots \mathbf{\Pi}_2 \mathbf{\Pi}_1$ (product of all interchange permutation matrices)
- $\mathbf{L} = \tilde{\mathbf{M}}_1^{-1} \tilde{\mathbf{M}}_2^{-1} \cdots \tilde{\mathbf{M}}_{n-1}^{-1}$ where

$$\begin{aligned} \tilde{\mathbf{M}}_k &= (\mathbf{\Pi}_{n-1} \cdots \mathbf{\Pi}_{k+1}) \mathbf{M}_k (\mathbf{\Pi}_{k+1} \cdots \mathbf{\Pi}_{n-1}) \\ &= (\mathbf{\Pi}_{n-1} \cdots \mathbf{\Pi}_{k+1}) (\mathbf{I} - \boldsymbol{\tau}^{(k)} \mathbf{e}_k^T) (\mathbf{\Pi}_{k+1} \cdots \mathbf{\Pi}_{n-1}) = \mathbf{I} - \tilde{\boldsymbol{\tau}}^{(k)} \mathbf{e}_k^T \end{aligned}$$

  which is unit lower triagular with $\tilde{\boldsymbol{\tau}}^{(k)} = (\mathbf{\Pi}_{n-1} \cdots \mathbf{\Pi}_{k+1}) \boldsymbol{\tau}^{(k)}$ and hence $\tilde{\mathbf{M}}_k^{-1} = \mathbf{I} + \tilde{\boldsymbol{\tau}}^{(k)} \mathbf{e}_k^T$. Then, $\mathbf{L} = \tilde{\mathbf{M}}_1^{-1} \tilde{\mathbf{M}}_2^{-1} \cdots \tilde{\mathbf{M}}_{n-1}^{-1} = \mathbf{I} + \sum_{k=1}^{n-1} \tilde{\boldsymbol{\tau}}^{(k)} \mathbf{e}_k^T$.

Where is $\mathbf{P}$ and Where is $\mathbf{L}$?

Proof: moving $\boldsymbol{\Pi}_k$ to the far-right-hand side ($\boldsymbol{\Pi}_k$'s are symmetric and hence $\boldsymbol{\Pi}_k^2 = \mathbf{I}$)

$$\begin{aligned}
\mathbf{U} &= \mathbf{M}_{n-1}\boldsymbol{\Pi}_{n-1}\mathbf{M}_{n-2}\boldsymbol{\Pi}_{n-2}\cdots\boldsymbol{\Pi}_3\mathbf{M}_2\boldsymbol{\Pi}_2\mathbf{M}_1\boldsymbol{\Pi}_1\mathbf{A}\\
&= \mathbf{M}_{n-1}\boldsymbol{\Pi}_{n-1}\mathbf{M}_{n-2}(\boldsymbol{\Pi}_{n-1}\boldsymbol{\Pi}_{n-1})\boldsymbol{\Pi}_{n-2}\cdots\boldsymbol{\Pi}_3\mathbf{M}_2(\boldsymbol{\Pi}_3\boldsymbol{\Pi}_3)\boldsymbol{\Pi}_2\mathbf{M}_1(\boldsymbol{\Pi}_2\boldsymbol{\Pi}_2)\boldsymbol{\Pi}_1\mathbf{A}\\
&= \mathbf{M}_{n-1}(\boldsymbol{\Pi}_{n-1}\mathbf{M}_{n-2}\boldsymbol{\Pi}_{n-1})\boldsymbol{\Pi}_{n-1}(\boldsymbol{\Pi}_{n-2}\quad\cdots\quad\mathbf{M}_2\boldsymbol{\Pi}_3)\boldsymbol{\Pi}_3(\boldsymbol{\Pi}_2\mathbf{M}_1\boldsymbol{\Pi}_2)\boldsymbol{\Pi}_2\boldsymbol{\Pi}_1\mathbf{A}\\
&= \mathbf{M}_{n-1}(\boldsymbol{\Pi}_{n-1}\mathbf{M}_{n-2}\boldsymbol{\Pi}_{n-1})\boldsymbol{\Pi}_{n-1}(\boldsymbol{\Pi}_{n-2}\quad\cdots\\
&\qquad\qquad\mathbf{M}_2\boldsymbol{\Pi}_3)(\boldsymbol{\Pi}_4\boldsymbol{\Pi}_4)\boldsymbol{\Pi}_3(\boldsymbol{\Pi}_2\mathbf{M}_1\boldsymbol{\Pi}_2)(\boldsymbol{\Pi}_3\boldsymbol{\Pi}_3)\boldsymbol{\Pi}_2\boldsymbol{\Pi}_1\mathbf{A}\\
&= \mathbf{M}_{n-1}(\boldsymbol{\Pi}_{n-1}\mathbf{M}_{n-2}\boldsymbol{\Pi}_{n-1})(\boldsymbol{\Pi}_{n-1}\boldsymbol{\Pi}_{n-2}\quad\cdots\\
&\qquad\qquad\mathbf{M}_2\boldsymbol{\Pi}_3\boldsymbol{\Pi}_4)\boldsymbol{\Pi}_4(\boldsymbol{\Pi}_3\boldsymbol{\Pi}_2\mathbf{M}_1\boldsymbol{\Pi}_2\boldsymbol{\Pi}_3)\boldsymbol{\Pi}_3\boldsymbol{\Pi}_2\boldsymbol{\Pi}_1\mathbf{A}\\
&= \ldots\\
&= \underbrace{\mathbf{M}_{n-1}}_{\tilde{\mathbf{M}}_{n-1}}\underbrace{(\boldsymbol{\Pi}_{n-1}\mathbf{M}_{n-2}\boldsymbol{\Pi}_{n-1})}_{\tilde{\mathbf{M}}_{n-2}}\underbrace{(\boldsymbol{\Pi}_{n-1}\boldsymbol{\Pi}_{n-2}}_{\tilde{\mathbf{M}}_{n-3}}\cdots\\
&\qquad\underbrace{\mathbf{M}_2\boldsymbol{\Pi}_3\boldsymbol{\Pi}_4\cdots\boldsymbol{\Pi}_{n-1})}_{\tilde{\mathbf{M}}_2}\underbrace{(\boldsymbol{\Pi}_{n-1}\cdots\boldsymbol{\Pi}_3\boldsymbol{\Pi}_2\mathbf{M}_1\boldsymbol{\Pi}_2\boldsymbol{\Pi}_3\cdots\boldsymbol{\Pi}_{n-1})}_{\tilde{\mathbf{M}}_1}\boldsymbol{\Pi}_{n-1}\cdots\boldsymbol{\Pi}_3\boldsymbol{\Pi}_2\boldsymbol{\Pi}_1\mathbf{A}
\end{aligned}$$

The LU with Partial Pivoting Code

```
function [P,L,U]= my_lu_pivoting(A)
n= size(A,1);
P= eye(n); L= eye(n); t= zeros(n,1); U= A;
for k=1:1:n-1,
     p= argmax(U(k:n,k));         % pivoting
     P(k,:)  ⟷  P(p,:);          % to form the permutation matrix
     U(k,k:n) ⟷  U(p,k:n);       % interchange rows in A^(k)
     L(k,1:k-1) ⟷  L(p,1:k-1);   % interchange the mutipliers
     rows= k+1:n;
     t(rows)= U(rows,k)/U(k,k);
     U(rows,rows)= U(rows,rows)- t(rows)*U(k,rows);
     U(rows,k)= 0;
     L(rows,k)= t(rows);
end;
```

- complexity: $\mathcal{O}(2n^3/3)$
- Reiteration: If row k and p are interchanged to create the kth pivot, the multipliers $[\ell_{k1}, \ldots, \ell_{k,k-1}]$ and $[\ell_{p1}, \ldots, \ell_{p,k-1}]$ trade places in the formation of $\mathbf{L}$.

Discussion

- to summarize, procedures for solving a linear system $\mathbf{Ax} = \mathbf{b}$ by LU decomposition
  - LU decomposition: decompose $\mathbf{A}$ as $\mathbf{PA} = \mathbf{LU}$ ($2n^3/3$ flops).
  - Permutation: $\mathbf{Pb}$ (0 flops).
  - Forward substitution: solve $\mathbf{Lz} = \mathbf{Pb}$ (n^2 flops).
  - Backward substitution: solve $\mathbf{Ux} = \mathbf{z}$ (n^2 flops).

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

Example (effect of roundoff error): Solve the linear system $\mathbf{Ax} = \mathbf{b}$ via LU decomposition with partial pivoting, where

$$\mathbf{A} = \begin{bmatrix} 0.02 & 61.3 \\ 3.43 & -8.5 \end{bmatrix} \text{ and } \mathbf{b} = \begin{bmatrix} 61.5 \\ 25.8 \end{bmatrix}.$$

Keep 3 significant figures.

LU Decomposition with Complete Pivoting

- **complete/full pivoting:** when eliminating elements in $\mathbf{a}_k^{(k-1)}$, find an integer p, q, $k \le p, q \le n$, s.t.

$$\left|a_{pq}^{(k-1)}\right| = \max_{k \le i,j \le n} \left|a_{ij}^{(k-1)}\right|.$$

  and then interchange rows p and k and then columns q and k of $\mathbf{A}^{(k-1)}$.
- $[\mathbf{M}_k]_{ik} = -a_{ik}/a_{kk}$, then $\left|[\mathbf{M}_k]_{ik}\right| \le 1$ for $k = 1, \ldots, n-1$ and $i = k+1, \ldots, n$.

LU decomposition with comlete pivoting: given $\mathbf{A} \in \mathbb{R}^{n \times n}$, find matrices $\mathbf{L}, \mathbf{U}, \mathbf{P}, \mathbf{Q} \in \mathbb{R}^{n \times n}$ such that

$$\mathbf{PAQ} = \mathbf{LU}$$

where

$\mathbf{P}, \mathbf{Q} \in \mathbb{R}^{n \times n}$ is a permutation matrix;
$\mathbf{L} \in \mathbb{R}^{n \times n}$ is unit lower triangular with $|\ell_{ij}| \le 1$;
$\mathbf{U} \in \mathbb{R}^{n \times n}$ is upper triangular.

Questions: how to perform $\mathbf{PAQ} = \mathbf{LU}$?

LU Decomposition with Complete Pivoting

Finding $\mathbf{U}$ by Gauss elimination with complete pivoting

Problem: find interchange permutations $\mathbf{\Pi}_1, \mathbf{\Pi}_2, \ldots, \mathbf{\Pi}_{n-1}, \mathbf{\Gamma}_1, \mathbf{\Gamma}_2, \ldots, \mathbf{\Gamma}_{n-1} \in \mathbb{R}^{n\times n}$ and Gauss transformations $\mathbf{M}_1, \mathbf{M}_2, \ldots, \mathbf{M}_{n-1} \in \mathbb{R}^{n\times n}$ such that

$$\mathbf{M}_{n-1}\mathbf{\Pi}_{n-1}\cdots\mathbf{M}_2\mathbf{\Pi}_2\mathbf{M}_1\mathbf{\Pi}_1\mathbf{A}\mathbf{\Gamma}_1\mathbf{\Gamma}_2\cdots\mathbf{\Gamma}_{n-1} = \mathbf{U}, \quad \mathbf{U} \text{ being upper triangular,}$$

and no multipliers in $\mathbf{M}_k$ for $k = 1, \ldots, n-1$ is larger than one in absolute value.

Where is $\mathbf{P}$, $\mathbf{Q}$, and $\mathbf{L}$?

- $\mathbf{L}$, $\mathbf{P}$ is defined as the same with LU factorization with partial pivotings
- $\mathbf{Q} = \mathbf{\Gamma}_1\mathbf{\Gamma}_2\cdots\mathbf{\Gamma}_{n-1}$
- LU decomposition with complete pivoting $\mathbf{PAQ} = \mathbf{LU}$ (more stable, higher cost)

Discussion

- besides solving $\mathbf{Ax} = \mathbf{b}$, LU decomposition (with pivoting) can also be used to
  - compute $\mathbf{A}^{-1}$: let $\mathbf{B} = \mathbf{A}^{-1}$.

$$\mathbf{AB} = \mathbf{I} \quad \Longleftrightarrow \quad \mathbf{A}\mathbf{b}_i = \mathbf{e}_i,\ i = 1, \ldots, n \text{ (i.e., solve } n \text{ linear systems).}$$

  - compute $\det(\mathbf{A})$: $\det(\mathbf{A}) = \det(\mathbf{L})\det(\mathbf{U}) = \prod_{i=1}^{n} u_{ii}$ (cf. Property 2.2).

- I have learned solving $\mathbf{Ax} = \mathbf{b}$ by reducing the augmented matrix $[\mathbf{A} \mid \mathbf{b}]$ to a row echelon form based on Gauss elimination followed by backward substitution in elementary linear algebra. Why LU decomposition?
  - reducing the augmented matrix $[\mathbf{A} \mid \mathbf{b}]$ to a row echelon form: $\mathcal{O}(n^3)$
  - LU decomposition $\mathbf{PA} = \mathbf{LU}$: $\mathcal{O}(n^3)$
  - what if you have a series of linear systems, i.e., $\mathbf{A}\mathbf{x}_i = \mathbf{b}_i$ for $i = 1, \ldots, N$?

Direct Methods for Special Linear Systems

LDM Decomposition

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

$$\mathbf{A} = \mathbf{L}\mathbf{D}\mathbf{M}^T,$$

where

$\mathbf{L}$, $\mathbf{M}$ is unit lower triangular, i.e., lower triangular with unit diagonal elements;
$\mathbf{D} = \mathrm{Diag}(d_1, \ldots, d_n)$.

- a different way of writing the LU decomposition: if $\mathbf{A} = \mathbf{L}\mathbf{U}$ is the LU decomposition, then the same $\mathbf{L}$,

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

  form the LDM decomposition.
- $\mathbf{D}$ is the matrix of pivots. $\mathbf{U}$ is a row scailing of $\mathbf{M}^T$.
- the existence of LDM decomposition follows that of LU.
- a.k.a. LDU decomposition

Solving LDM Decomposition

Notation: $\mathbf{A}_{i:j,k:l}$ denotes a submatrix of $\mathbf{A}$ obtained by keeping $i, i+1, \ldots, j$ rows and $k, k+1, \ldots, l$ columns of $\mathbf{A}$.

Idea: examine $\mathbf{A} = \mathbf{LDM}^T$ column by column:

$$\mathbf{a}_j = \mathbf{A}_j = \mathbf{A}\mathbf{e}_j = \mathbf{L}\mathbf{v}, \tag{$\star$}$$

where $1 \leq j \leq n$,

$$\mathbf{v} = \mathbf{DM}^T\mathbf{e}_j.$$

Observations:

1. $(\star)$ can be expanded as

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

2. $v_i = d_i m_{ji}$, specifically, $v_i = d_i$ since $m_{ij} = 1,\ i = j$;
3. $v_i = 0, i = j+1, \ldots, n$;

(can also analyze $\mathbf{A} = \mathbf{LDM}^T$ row by row defining $\mathbf{e}_i^T\mathbf{A} = \mathbf{u}^T\mathbf{M}^T$ and $\mathbf{u}^T = \mathbf{e}_i^T\mathbf{LD}$)

Solving LDM Decomposition

Recall from the last page that

$$\overbrace{\begin{bmatrix} \times & \times & \cdots & \cdots \\ \times & \times & \cdots & \cdots \\ \vdots & \vdots & \ddots & \\ \vdots & \vdots & & \ddots \end{bmatrix}}^{\mathbf{A}} = \overbrace{\begin{bmatrix} 1 & & & \\ \times & 1 & & \\ \vdots & ? & \ddots & \\ \vdots & ? & & \ddots \end{bmatrix}}^{\mathbf{L}} \overbrace{\begin{bmatrix} v_1 \\ v_2 \\ 0 \\ \vdots \end{bmatrix}}^{\mathbf{v}}$$

$$\mathbf{A}_{1:j,j} = \mathbf{L}_{1:j,1:j}\mathbf{v}_{1:j}$$
$$\mathbf{A}_{j+1:n,j} = \mathbf{L}_{j+1:n,1:j}\mathbf{v}_{1:j}$$

$$v_i = d_i m_{ji} \ (v_j = d_j)$$

$$\overbrace{\begin{bmatrix} v_1 \\ v_2 \\ 0 \\ \vdots \end{bmatrix}}^{\mathbf{v}} = \overbrace{\begin{bmatrix} \times & & \\ & ? & \\ & & \\ & & \ddots \end{bmatrix}}^{\mathbf{D}} \overbrace{\begin{bmatrix} 1 & ? & \\ & 1 & \\ & & \\ & & \ddots \end{bmatrix}}^{\mathbf{M}^T}$$

Problem: suppose that $\mathbf{L}_{1:n,1:j-1}$, the first $j-1$ columns of $\mathbf{L}$, is known. Find $\mathbf{L}_{j+1:n,j}$ (the jth column of $\mathbf{L}$), d_j, and the $\left[\mathbf{M}^T\right]_{1:j-1,j}$ (the jth column of $\mathbf{M}^T$).

1. $\mathbf{L}_{1:j,1:j}$ is known (why?); solve $\mathbf{A}_{1:j,j} = \mathbf{L}_{1:j,1:j}\mathbf{v}_{1:j}$ for $\mathbf{v}_{1:j}$
2. obtain $\mathbf{L}_{j+1:n,j}$, d_j, and $\left[\mathbf{M}^T\right]_{1:j-1,j}$
   - $\mathbf{L}_{j+1:n,j} = (\mathbf{A}_{j+1:n,j} - \mathbf{L}_{j+1:n,1:j-1}\mathbf{v}_{1:j-1})/v_j$.
   - (bonus) $d_j = v_j$, $m_{ji} = v_i/d_i$ for $i = 1, \ldots, j-1$.

An LDM Decomposition Code

```
function [L,D,M]= my_ldm(A)
n= size(A,1);
L= eye(n); d= zeros(n,1); M= eye(n);
v= zeros(n,1);
for j=1:n,
     % solve A_{1:j,j} = L_{1:j,1:j} v_{1:j} by forward substitution
     v(1:j)= for_subs(L(1:j,1:j),A(1:j,j));
     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);
```

- complexity: $\mathcal{O}(2n^3/3)$ (same as the previous LU code)
- the LDM is not normally used in practice for solving general linear systems
- however, LDM decomposition is much more interesting when $\mathbf{A}$ is symmetric

LDL Decomposition for Symmetric Matrices

If $\mathbf{A}$ is symmetric, then the LDM decomposition may be reduced to

$$\mathbf{A} = \mathbf{L}\mathbf{D}\mathbf{L}^T.$$

Theorem 2.2. If $\mathbf{A} = \mathbf{L}\mathbf{D}\mathbf{M}^T$ is the LDM decomposition of a nonsingular symmetric $\mathbf{A}$, then $\mathbf{L} = \mathbf{M}$.

Solving LDL:

- recall that in the previous LDM decomposition, the key is to find the unknown

$$\mathbf{v} = \mathbf{D}\mathbf{M}^T\mathbf{e}_j$$

  by solving $\mathbf{A}_{1:j,j} = \mathbf{L}_{1:j,1:j}\mathbf{v}_{1:j}$ via forward substitution.
- Finding $\mathbf{v}$ is much easier and there is no need to run forward substitution.
  - (exploit the symmetry property) since $\mathbf{M} = \mathbf{L}$,

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

    All the elements, except for v_j, are known.
  - $a_{jj} = \mathbf{L}_{j,1:j}\mathbf{v}_{1:j} = \mathbf{L}_{j,1:j-1}\mathbf{v}_{1:j-1} + v_j$

An LDL Decomposition Code

```
function [L,D]= my_ldl(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)= for_subs(L(1:j,1:j),A(1:j,j));
     v(1:j-1)= L(j,1:j-1)'.*d(1:j-1); % replace for_subs.
     v(j)= A(j,j)- L(j,1:j-1)*v(1:j-1); % replace for_subs.
     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);
```

- complexity: $\mathcal{O}(n^3/3)$, half of LU or LDM
- LDL is used to solve symmetric linear systems

Cholesky Factorization for PD Matrices

- a matrix $\mathbf{A} \in \mathbb{S}^n$ is said to be positive semidefinite (PSD) if

$$\mathbf{x}^T\mathbf{A}\mathbf{x} \geq 0, \quad \text{for all } \mathbf{x} \in \mathbb{R}^n;$$

  and positive definite (PD) if

$$\mathbf{x}^T\mathbf{A}\mathbf{x} > 0, \quad \text{for all } \mathbf{x} \in \mathbb{R}^n \text{ with } \mathbf{x} \neq \mathbf{0}$$

Cholesky factorization: given a PD $\mathbf{A} \in \mathbb{S}^n$, factorize $\mathbf{A}$ as

$$\mathbf{A} = \mathbf{G}\mathbf{G}^T,$$

where $\mathbf{G} \in \mathbb{R}^{n\times n}$ is lower triangular with positive diagonal elements.

Cholesky Factorization for PD Matrices

Theorem 2.3. If $\mathbf{A} \in \mathbb{S}^n$ is PD, then there exists a unique lower triangular $\mathbf{G} \in \mathbb{R}^{n\times n}$ with positive diagonal elements such that $\mathbf{A} = \mathbf{G}\mathbf{G}^T$.

- idea: if $\mathbf{A}$ is symmetric and PD, then its LDL decomposition

$$\mathbf{A} = \mathbf{L}\mathbf{D}\mathbf{L}^T$$

has $d_i > 0$ for all $i = 1, \ldots, n$ (as an exercise, verify this). Putting $\mathbf{G} = \mathbf{L}\mathbf{D}^{\frac{1}{2}}$ yields the Cholesky factorization.

Solving Cholesky factorization:

- (exploit the symmetry) the key is to find the unknown

$$\mathbf{v} = \mathbf{G}^T\mathbf{e}_j \quad \text{or} \quad v_i = g_{ji}.$$

All the elements, except for v_j, are known.

- (exploit the positive-definiteness property)

$$a_{jj} = \mathbf{G}_{j,1:j}\mathbf{v}_{1:j} = \mathbf{G}_{j,1:j-1}\mathbf{v}_{1:j-1} + g_{jj}v_j = \mathbf{G}_{j,1:j-1}\mathbf{G}_{j,1:j-1}^T + g_{jj}^2$$

A Cholesky Factorization Code

```
function [G]= my_Cholesky(A)
n= size(A,1);
G= zeros(n,n);
v= zeros(n,1);
for j=1:n,
     v(1:j-1)= G(j,1:j-1);
     v(j)= sqrt(A(j,j)- v(1:j-1)'*v(1:j-1));
     G(j,j)= v(j);
     G(j+1:n,j)= (A(j+1:n,j)-G(j+1:n,1:j-1)*v(1:j-1))/v(j);
end;
```

- computing procedure is similar to LDL
- can be computed in $\mathcal{O}(n^3/3)$, no pivoting required, numerically very stable
- Cholesky decomposition is used to solve PD linear systems

LU Decomposition for Banded Matrices

For a banded matrix $\mathbf{A} \in \mathbb{R}^{n \times n}$,

- lower bandwidth p if $a_{ij} = 0$ whenever $i > j + p$
- upper bandwidth q if $a_{ij} = 0$ whenever $j > i + q$

Theorem 2.4. Suppose $\mathbf{A} \in \mathbb{R}^{n \times n}$ has an LU factorization $\mathbf{A} = \mathbf{LU}$. If $\mathbf{A}$ has upper bandwidth q and lower bandwidth p, then $\mathbf{U}$ has upper bandwidth q and $\mathbf{L}$ has lower bandwidth p.

Proof: cf. Theorem 4.3.1 in **[Golub-van-Loan'13]** for details

- $\mathbf{L}$ inheritates the lower bandwidth of $\mathbf{A}$

- $\mathbf{U}$ inheritates the upper bandwidth of $\mathbf{A}$

Banded LU factorization with partial pivoting: the upper bandwidth of $\mathbf{U}$ is $p + q$
cf. Theorem 4.3.2 in **[Golub-van-Loan'13]** for details

Iterative Methods for Linear Systems

Iterative/Indirect Methods for Linear Systems

- solving linear systems via LU requires $\mathcal{O}(n^3)$
- $\mathcal{O}(n^3)$ is too much for large-scale linear systems
- the motivation behind iterative methods is to seek less expensive ways to find an (approximate) linear system solution
- note: see also the ideas of handling large-scale LS problems forthcoming in Topic 3, which is relevant to the context here

The Key Insight of Iterative Methods

- assume $a_{ii} \neq 0$ for all i

- observe

$$\mathbf{b} = \mathbf{A}\mathbf{x} \quad \Longleftrightarrow \quad b_i = a_{ii}x_i + \sum_{j \neq i} a_{ij}x_j, \quad i = 1, \ldots, n$$

$$\Longleftrightarrow \quad x_i = \left(b_i - \sum_{j \neq i} a_{ij}x_j \right) / a_{ii}, \quad i = 1, \ldots, n \qquad (\dagger)$$

- idea: find an $\mathbf{x}$ that fulfils the equations in $(\dagger)$

Jacobi Iterations

input: a starting point $\mathbf{x}^{(0)}$
for $k = 0, 1, 2, \ldots$

$$x_i^{(k+1)} = \left(b_i - \sum_{j \neq i} a_{ij} x_j^{(k)}\right) / a_{ii}, \text{ for } i = 1, \ldots, n$$

end

- complexity per iteration: $\mathcal{O}(n^2)$ for dense $\mathbf{A}$, $\mathcal{O}(\mathrm{nnz}(\mathbf{A}))$ for sparse $\mathbf{A}$
- the Jacobi update step can be computed in a parallel or distributed fashion
  - same idea appeared in distributed power control in 2G or 3G wireless networks
- a natural idea, heuristic at first glance
- does the Jacobi iterations converge to the linear system solution?
  - it does not, in general
  - it does if the diagonal elements a_{ii}'s are "dominant" compared to the off-diagonal elements; see Theorem 11.2.2 in **[Golub-van-Loan'13]** for details

Gauss-Seidel (G-S) Iterations

input: a starting point $\mathbf{x}^{(0)}$
for $k = 0, 1, 2, \dots$
for $i = 1, 2, \dots, n$

$$x_i^{(k+1)} = \left(b_i - \sum_{j=1}^{i-1} a_{ij} x_j^{(k+1)} - \sum_{j=i+1}^{n} a_{ij} x_j^{(k)} \right) / a_{ii}$$

end
end

- use the most recently available $\mathbf{x}$ to perform update
- sequential, cannot be computed in a distributed or parallel manner
- guaranteed to converge to the linear system solution if
  - $\mathbf{A}$ has diagonally dominant characteristics (similar to the Jacobi iterations)
  - $\mathbf{A}$ is symmetric PD; see Theorem 11.2.3 in **[Golub-van-Loan'13]**

Other Topics on Linear Systems

Underdetermined Systems

Problem: If $\mathbf{A} \in \mathbb{R}^{m \times n}$ with $m < n$, $\mathrm{rank}(\mathbf{A}) = m$, and $\mathbf{b} \in \mathbb{R}^m$, find $\mathbf{x} \in \mathbb{R}^n$ s.t.

$$\mathbf{A}\mathbf{x} = \mathbf{b}.$$

Such a linear system is said to be underdetermined.

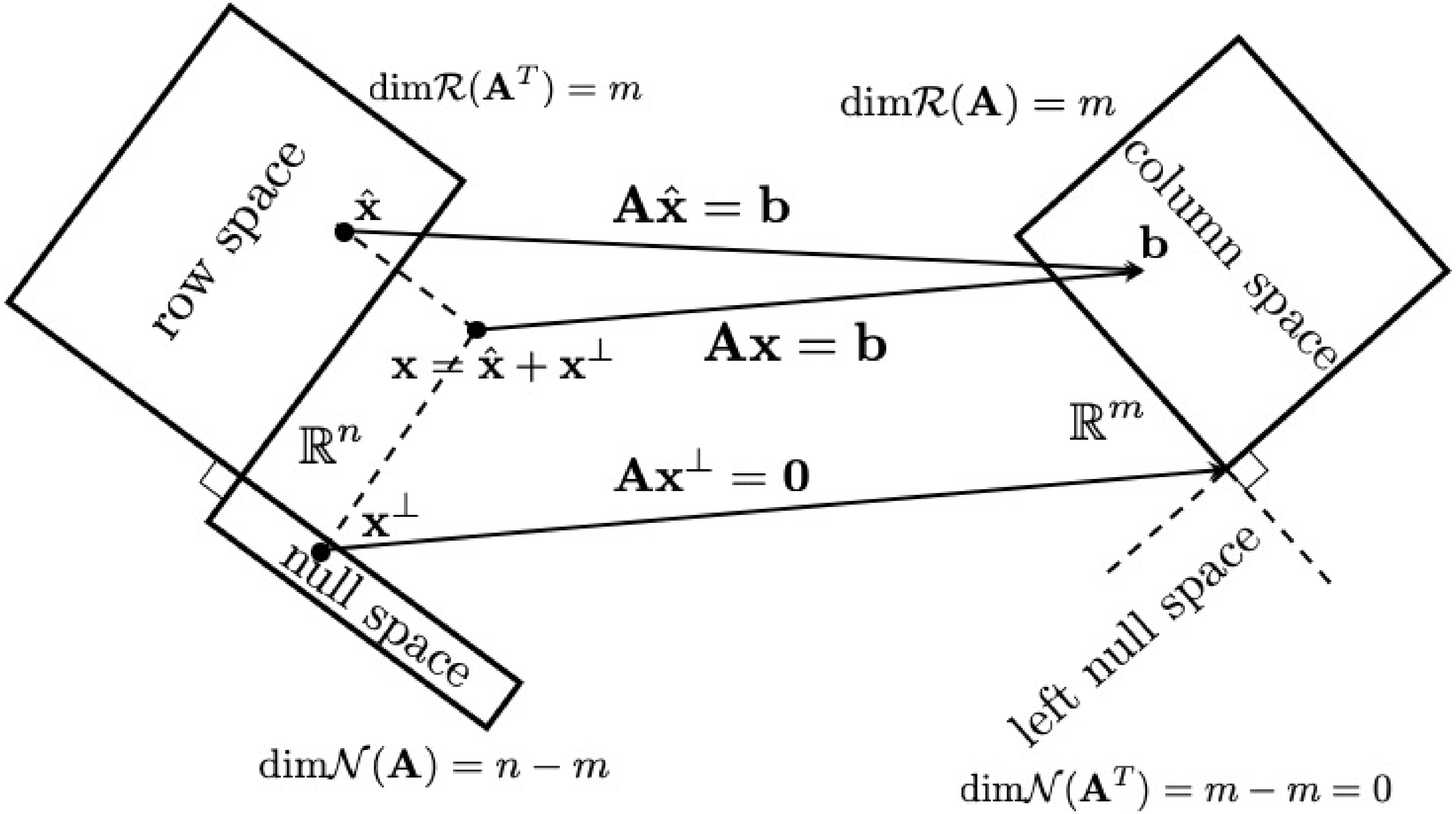
- an underdetermined linear system has infinite number of solutions given by

$$\mathbf{x} = \hat{\mathbf{x}} + \mathbf{x}^{\perp} = \hat{\mathbf{x}} + \mathbf{F}\mathbf{v} \quad \text{with} \quad \mathbf{v} \in \mathbb{R}^{n-m},$$

  where $\hat{\mathbf{x}} \in \mathcal{R}(\mathbf{A}^T)$ is (any) particular solution and special solutions $\mathbf{x}^{\perp} \in \mathcal{N}(\mathbf{A})$ with columns of $\mathbf{F} \in \mathbb{R}^{n \times (n-m)}$ spans $\mathcal{N}(\mathbf{A})$.

- several numerical methods for computing $\mathbf{F}$ (rectangular LU decomposition, QR factorization (cf. Topic 4), ...)
- solution to smallest ℓ_2 norm: $\mathbf{x}^{\perp} = \mathbf{0}$, i.e., $\mathbf{v} = \mathbf{0}$, cf. Topic 7
- solution to smallest ℓ_0 norm: can we find a sparsest solution $\mathbf{x}$? cf. Topic 8

Underdetermined Systems



Solving Underdetermined Systems by Rectangular LU

A rectangular LU decomposition of $\mathbf{A}$ is

$$\mathbf{A} = \mathbf{L}\big[\mathbf{U}_1 \mid \mathbf{U}_2\big]$$

where $\mathbf{L}$ is unit lower triangular, $\mathbf{U}_1 \in \mathbb{R}^{m\times m}$ is nonsingular and uppertriangular, and $\mathbf{U}_2 \in \mathbb{R}^{m\times(n-m)}$.

- note

  $$\mathbf{A}\mathbf{x} = \mathbf{L}\big[\mathbf{U}_1 \mid \mathbf{U}_2\big]\begin{bmatrix}\mathbf{x}_1\\ \mathbf{x}_2\end{bmatrix} = \mathbf{L}(\mathbf{U}_1\mathbf{x}_1 + \mathbf{U}_2\mathbf{x}_2) = \mathbf{b}$$

  which can be solved by first solving $\mathbf{L}\mathbf{z} = \mathbf{b}$ and then solving $\mathbf{U}_1\mathbf{x}_1 = \mathbf{z} - \mathbf{U}_2\mathbf{x}_2$ given a specific $\mathbf{x}_2 \in \mathbb{R}^{n-m}$.

  $$\mathbf{x}_1 = \mathbf{U}_1^{-1}\mathbf{L}^{-1}\mathbf{b} - \mathbf{U}_1^{-1}\mathbf{U}_2\mathbf{x}_2$$

  $$\mathbf{x} = \begin{bmatrix}\mathbf{x}_1\\ \mathbf{x}_2\end{bmatrix} = \begin{bmatrix}\mathbf{U}_1^{-1}\mathbf{L}^{-1}\mathbf{b} - \mathbf{U}_1^{-1}\mathbf{U}_2\mathbf{x}_2\\ \mathbf{x}_2\end{bmatrix} = \begin{bmatrix}\mathbf{U}_1^{-1}\mathbf{L}^{-1}\mathbf{b}\\ \mathbf{0}\end{bmatrix} + \begin{bmatrix}-\mathbf{U}_1^{-1}\mathbf{U}_2\\ \mathbf{I}\end{bmatrix}\mathbf{x}_2$$

- one possible solution is to set $\hat{\mathbf{x}} = \begin{bmatrix}\mathbf{U}_1^{-1}\mathbf{L}^{-1}\mathbf{b}\\ \mathbf{0}\end{bmatrix}$, $\mathbf{F} = \begin{bmatrix}-\mathbf{U}_1^{-1}\mathbf{U}_2\\ \mathbf{I}\end{bmatrix}$, and $\mathbf{v} = \mathbf{x}_2$.

Sensitivity Analysis of Linear Systems

- Scenario:
  - let $\mathbf{A} \in \mathbb{R}^{n \times n}$ be nonsingular, and $\mathbf{y} \in \mathbb{R}^n$. Let $\mathbf{x}$ be the solution to
  $$\mathbf{A}\mathbf{x} = \mathbf{b}.$$
  - consider a perturbed version of the above system: $\hat{\mathbf{A}} = \mathbf{A} + \Delta\mathbf{A}, \hat{\mathbf{y}} = \mathbf{y} + \Delta\mathbf{y}$, where $\Delta\mathbf{A}$ and $\Delta\mathbf{y}$ are errors. Let $\hat{\mathbf{x}}$ be a solution to the perturbed system
  $$\hat{\mathbf{A}}\hat{\mathbf{x}} = \hat{\mathbf{b}}.$$
- Problem: analyze how the solution error $\|\hat{\mathbf{x}} - \mathbf{x}\|_2$ scales with $\Delta\mathbf{A}$ and $\Delta\mathbf{y}$
- remark: $\Delta\mathbf{A}$ and $\Delta\mathbf{y}$ may be floating point errors, measurement errors, etc
- forthcoming in Topic 6 Singular Value Decomposition.

References

[Golub-van-Loan'13] G. H. Golub and C. F. Van Loan, *Matrix Computations*, 4th edition, JHU Press, 2013.