Lecture 03 - Solving systems of linear equations

Baojian Zhou

DATA830001, Numerical Computation
School of Data Science, Fudan University

Sep. 18th, 2024

Solving Systems of Linear Equations

This lecture aims at solving the following systems of linear equations

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + a_{13}x_3 + \cdots + a_{1n}x_n = b_1 \\ a_{21}x_1 + a_{22}x_2 + a_{23}x_3 + \cdots + a_{2n}x_n = b_2 \\ a_{31}x_1 + a_{32}x_2 + a_{33}x_3 + \cdots + a_{3n}x_n = b_3 \\ \vdots \\ a_{n1}x_1 + a_{n2}x_2 + a_{n3}x_3 + \cdots + a_{nn}x_n = b_n \end{cases}$$

It is written as

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}, \tag{1}$$

where $\boldsymbol{A} \in \mathbb{R}^{n\times n}$, $\boldsymbol{x} \in \mathbb{R}^n$, and $\boldsymbol{b} \in \mathbb{R}^n$.

Matrix Algebra

Given $\boldsymbol{A} = [\boldsymbol{a}_1, \boldsymbol{a}_2, \ldots, \boldsymbol{a}_n] \in \mathbb{R}^{n \times n}$

1. $\boldsymbol{a}_i \in \mathbb{R}^n$ a *column vector*.
2. *Transpose* of $\boldsymbol{A}$ is denoted by $\boldsymbol{A}^\top$ with each entry $\left(\boldsymbol{A}^\top\right)_{ij} = a_{ji}$.
3. We say $\boldsymbol{A}$ is *symmetric* if $\boldsymbol{A}^\top = \boldsymbol{A}$.
4. We say $\boldsymbol{I} = \boldsymbol{A}$ is an *identity matrix* $(\boldsymbol{A})_{ij} = 1$ if $i = j$; 0 otherwise.
5. We say $\boldsymbol{A}$ is positive definite if $\boldsymbol{x}^\top \boldsymbol{A}\boldsymbol{x} > 0, \forall \boldsymbol{x} \in \mathbb{R}^n \backslash \{0\}$.
6. We say $\boldsymbol{A}$ is diagonally dominant matrix, if $|a_{ii}| > \sum_{j \neq i}^{n} |a_{ij}|, i = 1, 2, \ldots, n$.
7. We say λ is an eigenvalue of $\boldsymbol{A}$ if $\boldsymbol{A}\boldsymbol{x} = \lambda \boldsymbol{x}$ given $\boldsymbol{x} \neq 0$ and we call $\boldsymbol{x}$ is an eigenvector.

Triangle matrix

A matrix of the form

$$\boldsymbol{L} = \begin{bmatrix} \ell_{1,1} & & & & 0 \\ \ell_{2,1} & \ell_{2,2} & & & \\ \ell_{3,1} & \ell_{3,2} & \ddots & & \\ \vdots & \vdots & \ddots & \ddots & \\ \ell_{n,1} & \ell_{n,2} & \dots & \ell_{n,n-1} & \ell_{n,n} \end{bmatrix}$$

is called a **lower triangular matrix**.

$$\boldsymbol{U} = \begin{bmatrix} u_{1,1} & u_{1,2} & u_{1,3} & \dots & u_{1,n} \\ & u_{2,2} & u_{2,3} & \dots & u_{2,n} \\ & & \ddots & \ddots & \vdots \\ & & & \ddots & u_{n-1,n} \\ 0 & & & & u_{n,n} \end{bmatrix}$$

is called an **upper triangular matrix**.

Properties of triangle matrix

We say $\boldsymbol{L}$ is unit lower triangular if $\ell_{ii} = 1$ for $i = 1, 2, \ldots, n$.

- The transpose of an upper triangular matrix is a lower triangular matrix and vice versa.

Properties of triangle matrix

We say $\boldsymbol{L}$ is unit lower triangular if $\ell_{ii} = 1$ for $i = 1, 2, \ldots, n$.

- The transpose of an upper triangular matrix is a lower triangular matrix and vice versa.
- A matrix which is both symmetric and triangular is diagonal.

Properties of triangle matrix

We say $\boldsymbol{L}$ is unit lower triangular if $\ell_{ii} = 1$ for $i = 1, 2, \ldots, n$.

- The transpose of an upper triangular matrix is a lower triangular matrix and vice versa.
- A matrix which is both symmetric and triangular is diagonal.
- If $\boldsymbol{U}$ is upper triangular and invertible, then $\boldsymbol{U}^{-1}$ is upper triangular.

Properties of triangle matrix

We say $\boldsymbol{L}$ is unit lower triangular if $\ell_{ii} = 1$ for $i = 1, 2, \ldots, n$.

- The transpose of an upper triangular matrix is a lower triangular matrix and vice versa.
- A matrix which is both symmetric and triangular is diagonal.
- If $\boldsymbol{U}$ is upper triangular and invertible, then $\boldsymbol{U}^{-1}$ is upper triangular.
- The inverse of a unit lower triangular matrix is a unit lower triangular.

Properties of triangle matrix

We say $\boldsymbol{L}$ is unit lower triangular if $\ell_{ii} = 1$ for $i = 1, 2, \ldots, n$.

- The transpose of an upper triangular matrix is a lower triangular matrix and vice versa.
- A matrix which is both symmetric and triangular is diagonal.
- If $\boldsymbol{U}$ is upper triangular and invertible, then $\boldsymbol{U}^{-1}$ is upper triangular.
- The inverse of a unit lower triangular matrix is a unit lower triangular.
- The product of two upper (lower) triangular matrices is upper (lower) triangular.

Properties of triangle matrix

We say $\boldsymbol{L}$ is unit lower triangular if $\ell_{ii} = 1$ for $i = 1, 2, \ldots, n$.

- The transpose of an upper triangular matrix is a lower triangular matrix and vice versa.
- A matrix which is both symmetric and triangular is diagonal.
- If $\boldsymbol{U}$ is upper triangular and invertible, then $\boldsymbol{U}^{-1}$ is upper triangular.
- The inverse of a unit lower triangular matrix is a unit lower triangular.
- The product of two upper (lower) triangular matrices is upper (lower) triangular.

Forward Substitution

The matrix equation $\boldsymbol{L}\mathrm{x} = \mathrm{b}$ can be written as a system of linear equations

$$\begin{aligned} \ell_{1,1}x_1 \quad & = b_1 \\ \ell_{2,1}x_1 + \ell_{2,2}x_2 \quad & = b_2 \\ \vdots \quad \vdots \quad \ddots \quad \vdots \\ \ell_{n,1}x_1 + \ell_{n,2}x_2 + \cdots + \ell_{n,n}x_n & = b_n \end{aligned}$$

The solution can be written as

$$\begin{aligned} x_1 &= \frac{b_1}{\ell_{1,1}}, \\ x_2 &= \frac{b_2 - \ell_{2,1}x_1}{\ell_{2,2}}, \\ &\vdots \\ x_n &= \frac{b_n - \sum_{i=1}^{n-1} \ell_{n,i}x_i}{\ell_{n,n}}. \end{aligned}$$

Inverse of matrix

Idea: The n columns of a nonsingular $n \times n$ matrix $\boldsymbol{A}$ form a basis for the whole space $\mathbb{R}^n$. Therefore, we can uniquely express any vector as a linear combination of them. In particular, the canonical unit vector with 1 in the j th entry and zeros elsewhere, written $\boldsymbol{e}_j$, can be expanded:

$$\boldsymbol{e}_j = \sum_{i=1}^{n} z_{ij} \boldsymbol{a}_i, \qquad j = 1, 2, \ldots, n$$

Let $\boldsymbol{Z}$ be the matrix with entries z_{ij}, and let $\boldsymbol{z}_j$ denote the j th column of $\boldsymbol{Z}$. Then the above can be written $\boldsymbol{e}_j = \boldsymbol{A}\boldsymbol{z}_j$. It can be written concisely as

$$\left[\begin{array}{c|c|c} \boldsymbol{e}_1 & \cdots & \boldsymbol{e}_n \end{array} \right] = \boldsymbol{I} = \boldsymbol{A}\boldsymbol{Z}, \qquad (2)$$

where $\boldsymbol{I}$ is the $n \times n$ matrix known as the identity. The matrix $\boldsymbol{Z}$ is the inverse of $\boldsymbol{A}$. Any square *nonsingular* matrix $\boldsymbol{A}$ has a unique inverse, written $\boldsymbol{A}^{-1}$, that satisfies $\boldsymbol{A}\boldsymbol{A}^{-1} = \boldsymbol{A}^{-1}\boldsymbol{A} = \boldsymbol{I}$.

Inverse of matrix

Theorem 1.1 (Equivalent properties of inverse matrix)

For $\boldsymbol{A} \in \mathbb{R}^{n\times n}$, the following conditions are equivalent:

1. *$\boldsymbol{A}$ has an inverse $\boldsymbol{A}^{-1}$*
2. $\text{rank}(\boldsymbol{A}) = n$
3. *range* $(\boldsymbol{A}) = \mathbb{R}^n$
4. $\text{null}(\boldsymbol{A}) = \{0\}$
5. *0 is not an eigenvalue of $\boldsymbol{A}$*
6. *0 is not a singular value of $\boldsymbol{A}$*
7. $\det(\boldsymbol{A}) \neq 0$
8. *The equation $\boldsymbol{Ax} = 0$ implies $\boldsymbol{x} = 0$*
9. *For each $\boldsymbol{b} \in \mathbb{R}^n$, there is exactly one $\boldsymbol{x} \in \mathbb{R}^n$ such that $\boldsymbol{Ax} = \boldsymbol{b}$*

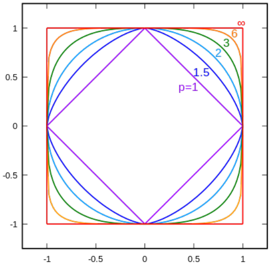
Norms in vector space

To measure vectors, we assign vectors nonnegative numbers.
When $p \geq 1$, we define p-norm of vector $\boldsymbol{x} \in \mathbb{R}^n$ as

$$\|\boldsymbol{x}\|_p \triangleq \left(\sum_{i=1}^{n} |x_i|^p \right)^{1/p} \tag{3}$$

Some useful properties:

- $\|\boldsymbol{x}\|_2 \leq \|\boldsymbol{x}\|_1 \leq \sqrt{n}\|\boldsymbol{x}\|_2$
- $\|\boldsymbol{x}\|_\infty \leq \|\boldsymbol{x}\|_2 \leq \sqrt{n}\|\boldsymbol{x}\|_\infty$
- $\|\boldsymbol{x}\|_\infty \leq \|\boldsymbol{x}\|_1 \leq n\|\boldsymbol{x}\|_\infty$
- Cauchy-Schwarz inequality: $|\langle \boldsymbol{x}, \boldsymbol{y} \rangle| \leq \|\boldsymbol{x}\|_2 \cdot \|\boldsymbol{y}\|_2$
- Triangle Inequality $\|\boldsymbol{x}+\boldsymbol{y}\|_p \leq \|\boldsymbol{x}\|_p + \|\boldsymbol{y}\|_p$



The values of coordinates (x_1, x_2), for which the p-norm $\|\boldsymbol{x}\|_p$ takes the value 1, i.e. $\|\boldsymbol{x}\|_p = 1$

Matrix Norm

Definition 1.2 (Matrix Norm)

Consider a vector space of matrices with m rows and n columns. A matrix norm is a function $\|\cdot\| : \mathbb{K}^{m\times n} \to \mathbb{R}$ that must satisfy the following properties: For all scalars $\alpha \in \mathbb{K}$ and matrices $\boldsymbol{A}, \boldsymbol{B} \in \mathbb{K}^{m\times n}$,

- Nonnegativity: $\|\boldsymbol{A}\| \geq 0$, unique zero: $\|\boldsymbol{A}\| = 0 \Leftrightarrow \boldsymbol{A} = 0_{m\times n}$
- Absolutely homogeneous: $\|\alpha \boldsymbol{A}\| = |\alpha| \cdot \|\boldsymbol{A}\|$
- Triangle inequality: $\|\boldsymbol{A} + \boldsymbol{B}\| \leq \|\boldsymbol{A}\| + \|\boldsymbol{B}\|$

- Forbenius norm: $\|\boldsymbol{A}\|_F := \left(\sum_{i,j=1}^{n} |a_{ij}|^2\right)^{1/2}$
- Operator norm: $\|\boldsymbol{A}\|_p = \sup_{\boldsymbol{x}\neq 0} \frac{\|\boldsymbol{A}\boldsymbol{x}\|_p}{\|\boldsymbol{x}\|_p}$, where $1 \leq p \leq \infty$

Matrix properties

Inverse of Simple Matrix

Given an invertible matrix $\boldsymbol{A} \in \mathbb{R}^{2\times 2}$, we have

$$\boldsymbol{A} = \begin{bmatrix} a & b \\ c & d \end{bmatrix}, \boldsymbol{A}^{-1} = \frac{1}{\det \boldsymbol{A}} \begin{bmatrix} d & -b \\ -c & a \end{bmatrix} = \frac{1}{ad - bc} \begin{bmatrix} d & -b \\ -c & a \end{bmatrix}.$$

Matrix Condition number

$$\begin{aligned}
\max_{\boldsymbol{e},\boldsymbol{b}\neq 0} \left\{ \frac{\left\|\boldsymbol{A}^{-1}\boldsymbol{e}\right\|}{\|\boldsymbol{e}\|} \frac{\|\boldsymbol{b}\|}{\|\boldsymbol{A}^{-1}\boldsymbol{b}\|} \right\} &= \max_{\boldsymbol{e}\neq 0} \left\{ \frac{\left\|\boldsymbol{A}^{-1}\boldsymbol{e}\right\|}{\|\boldsymbol{e}\|} \right\} \max_{\boldsymbol{b}\neq 0} \left\{ \frac{\|\boldsymbol{b}\|}{\|\boldsymbol{A}^{-1}\boldsymbol{b}\|} \right\} \\
&= \max_{\boldsymbol{e}\neq 0} \left\{ \frac{\left\|\boldsymbol{A}^{-1}\boldsymbol{e}\right\|}{\|\boldsymbol{e}\|} \right\} \max_{\boldsymbol{x}\neq 0} \left\{ \frac{\|\boldsymbol{A}\boldsymbol{x}\|}{\|\boldsymbol{x}\|} \right\} \\
&= \left\|\boldsymbol{A}^{-1}\right\| \cdot \|\boldsymbol{A}\|.
\end{aligned}$$

Eigenvalues of symmetric matrices

Definition 1.3 (Symmetric matrix)

Given any $\boldsymbol{A} \in \mathbb{R}^{n \times n}$, if $\boldsymbol{A}^\top = \boldsymbol{A}$, then we call $\boldsymbol{A}$ is a symmetric matrix.

Definition 1.4 (Orthonormal)

A set of vectors S is orthonormal if the elements of the set are unit vectors that are pairwise orthogonal. Let $S = \{\boldsymbol{u}, \boldsymbol{v}\}$, then $\|\boldsymbol{u}\| = 1, \|\boldsymbol{v}\| = 1$ and $\boldsymbol{u}^\top \boldsymbol{v} = 0$.

Theorem 1.5 (Real eigenvalues of $\boldsymbol{A}$)

Assume that $\boldsymbol{A} \in \mathbb{R}^{n \times n}$ is symmetric with real entries. Then the eigenvalues are real numbers, and the set of unit eigenvectors of $\boldsymbol{A}$ is an orthonormal set $S = \{\boldsymbol{w}_1, \ldots, \boldsymbol{w}_n\}$ that forms a basis of $\mathbb{R}^n$.

Symmetric positive-definite matrices

Definition 1.6 (Symmetric Positive-Definite (SPD))

The matrix $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ if $\boldsymbol{A}^\top = \boldsymbol{A}$. The matrix $\boldsymbol{A}$ is positive-definite if $\boldsymbol{x}^\top \boldsymbol{A}\boldsymbol{x} > 0$ for all vectors $\boldsymbol{x} \neq 0$.

Example:

- Show that $\boldsymbol{A} = \begin{bmatrix} 2 & 2 \\ 2 & 5 \end{bmatrix}$ is SPD: $\boldsymbol{A}$ is symmetric as $\boldsymbol{A}^\top = \boldsymbol{A}$ and applies definition

$$\begin{aligned} \boldsymbol{x}^\top \boldsymbol{A}\boldsymbol{x} &= \begin{bmatrix} x_1 & x_2 \end{bmatrix} \begin{bmatrix} 2 & 2 \\ 2 & 5 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \\ &= 2x_1^2 + 4x_1x_2 + 5x_2^2 = 2(x_1 + x_2)^2 + 3x_2^2. \end{aligned}$$

- $\boldsymbol{A} = \begin{bmatrix} 2 & 4 \\ 4 & 5 \end{bmatrix}$ is not SPD. One can show that $\boldsymbol{x}^\top \boldsymbol{A}\boldsymbol{x} = 2(x_1 + 2x_2)^2 - 3x_2^2$ with $x_1 = -2$ and $x_2 = 1$.

SPD matrices

Theorem 1.7

If $\boldsymbol{A} \in \mathbb{R}^{n\times}$ is a symmetric matrix, then $\boldsymbol{A}$ is positive-definite if and only if all of its eigenvalues are positive.

SPD matrices

Theorem 1.7

If $\boldsymbol{A} \in \mathbb{R}^{n\times}$ is a symmetric matrix, then $\boldsymbol{A}$ is positive-definite if and only if all of its eigenvalues are positive.

Proof.

Notice that if $\boldsymbol{A}$ is positive-definite and $\boldsymbol{A}\boldsymbol{v} = \lambda \boldsymbol{v}$ for a nonzero vector $\boldsymbol{v}$, then $0 < \boldsymbol{v}^\top \boldsymbol{A}\boldsymbol{v} = \boldsymbol{v}^\top(\lambda \boldsymbol{v}) = \lambda \|\boldsymbol{v}\|_2^2$. Hence, $\lambda > 0$.
On the other hand, if all eigenvalues of $\boldsymbol{A}$ are positive, then write any nonzero $\boldsymbol{x} = c_1 \boldsymbol{v}_1 + \ldots + c_n \boldsymbol{v}_n$ where each $\boldsymbol{v}_i$ are orthonormal unit vectors and not all c_i are zero. Then, we have

$$\begin{aligned} \boldsymbol{x}^\top \boldsymbol{A}\boldsymbol{x} &= (c_1 \boldsymbol{v}_1 + \cdots + c_n \boldsymbol{v}_n)^\top (c_1 \lambda_1 \boldsymbol{v}_1 + \cdots + c_n \lambda_n \boldsymbol{v}_n) \\ &= \lambda_1 \|\boldsymbol{c}_1\|_2^2 + \cdots + \lambda_n \|\boldsymbol{c}_n\|_2^2 > 0. \end{aligned}$$

■

Gaussian Elimination

The Naive Gaussian Elimination

- Add or subtract a multiple of one equation from another
- Multiply an equation by a nonzero constant

$$\left[\begin{array}{cccc|c} a_{11} & a_{12} & \dots & a_{1n} & b_1 \\ a_{21} & a_{22} & \dots & a_{2n} & b_2 \end{array}\right]$$

Subtract $\frac{a_{21}}{a_{11}}$ times row 1 from row 2, we have

$$\left[\begin{array}{cccc|c} a_{11} & a_{12} & \dots & a_{1n} & b_1 \\ 0 & a_{22} - \frac{a_{21}}{a_{11}} a_{12} & \dots & a_{2n} - \frac{a_{21}}{a_{11}} a_{1n} & b_2 - \frac{a_{21}}{a_{11}} b_1 \end{array}\right]$$

Repeat the above procedure until we find $\boldsymbol{U}$

Gaussian Elimination: LU Factorization

The idea of transforming $\boldsymbol{A}$ into an upper triangular $\boldsymbol{U}$ by introducing zeros below the diagonal by subtracting multiples of each row from subsequent rows. This process is equivalent to multiplying $\boldsymbol{A}$ by a sequence of lower triangular $\boldsymbol{L}_k$ on the left:

$$\underbrace{\boldsymbol{L}_{n-1} \cdots \boldsymbol{L}_2 \boldsymbol{L}_1}_{\boldsymbol{L}^{-1}} \boldsymbol{A} = \boldsymbol{U}.$$

Setting $\boldsymbol{L} = \boldsymbol{L}_1^{-1} \boldsymbol{L}_2^{-1} \cdots \boldsymbol{L}_{n-1}^{-1}$ gives $\boldsymbol{A} = \boldsymbol{L}\boldsymbol{U}$. Thus we obtain

$$\boldsymbol{A} = \boldsymbol{L}\boldsymbol{U}.$$

- $\boldsymbol{U}$ is upper triangular
- $\boldsymbol{L}$ is unit lower triangular (diagonals are all ones)

LU Factorization: An example

Let us consider $\boldsymbol{A} = \begin{bmatrix} 2 & 1 & 1 & 0 \\ 4 & 3 & 3 & 1 \\ 8 & 7 & 9 & 5 \\ 6 & 7 & 9 & 8 \end{bmatrix}$

First step of Gaussian elimination

$$\boldsymbol{L}_1\boldsymbol{A} = \begin{bmatrix} 1 & & & \\ -2 & 1 & & \\ -4 & & 1 & \\ -3 & & & 1 \end{bmatrix} \begin{bmatrix} 2 & 1 & 1 & 0 \\ 4 & 3 & 3 & 1 \\ 8 & 7 & 9 & 5 \\ 6 & 7 & 9 & 8 \end{bmatrix} = \begin{bmatrix} 2 & 1 & 1 & 0 \\ & 1 & 1 & 1 \\ & 3 & 5 & 5 \\ & 4 & 6 & 8 \end{bmatrix}$$

The second step

$$\boldsymbol{L}_2\boldsymbol{L}_1\boldsymbol{A} = \begin{bmatrix} 1 & & & \\ & 1 & & \\ & -3 & 1 & \\ & -4 & & 1 \end{bmatrix} \begin{bmatrix} 2 & 1 & 1 & 0 \\ & 1 & 1 & 1 \\ & 3 & 5 & 5 \\ & 4 & 6 & 8 \end{bmatrix} = \begin{bmatrix} 2 & 1 & 1 & 0 \\ & 1 & 1 & 1 \\ & & 2 & 2 \\ & & 2 & 4 \end{bmatrix}$$

LU Factorization: An example (continued)

$$L_3 L_2 L_1 A = \begin{bmatrix} 1 & & & \\ & 1 & & \\ & & 1 & \\ & & -1 & 1 \end{bmatrix} \begin{bmatrix} 2 & 1 & 1 & 0 \\ & 1 & 1 & 1 \\ & & 2 & 2 \\ & & 2 & 4 \end{bmatrix} = \begin{bmatrix} 2 & 1 & 1 & 0 \\ & 1 & 1 & 1 \\ & & 2 & 2 \\ & & & 2 \end{bmatrix} = U$$

Compute the product $L = L_1^{-1} L_2^{-1} \cdots L_{n-1}^{-1}$.

- L_i^{-1} is just L_i itself, but with each entry below the diagonal negated
- $L_1^{-1} L_2^{-1} \cdots L_{n-1}^{-1}$ is just the unit lower-triangular matrix with the nonzero subdiagonal entries of L_i^{-1} inserted in the appropriate places.

$$\underbrace{\begin{bmatrix} 1 & & & \\ -2 & 1 & & \\ -4 & & 1 & \\ -3 & & & 1 \end{bmatrix}^{-1}}_{L_1^{-1}} = \begin{bmatrix} 1 & & & \\ 2 & 1 & & \\ 4 & & 1 & \\ 3 & & & 1 \end{bmatrix}, \quad \underbrace{\begin{bmatrix} 2 & 1 & 1 & 0 \\ 4 & 3 & 3 & 1 \\ 8 & 7 & 9 & 5 \\ 6 & 7 & 9 & 8 \end{bmatrix}}_{A} = \underbrace{\begin{bmatrix} 1 & & & \\ 2 & 1 & & \\ 4 & 3 & 1 & \\ 3 & 4 & 1 & 1 \end{bmatrix}}_{L} \underbrace{\begin{bmatrix} 2 & 1 & 1 & 0 \\ & 1 & 1 & 1 \\ & & 2 & 2 \\ & & & 2 \end{bmatrix}}_{U}$$

Naive Gaussian Elimination

Algorithm 1 Naive Gaussian Elimination

$\boldsymbol{U} = \boldsymbol{A}, \boldsymbol{L} = \boldsymbol{I}$
for $k = 1$ to $n - 1$ **do**
    for $j = k + 1$ to n **do**
        $\ell_{jk} = u_{jk} / u_{kk}$
        $\boldsymbol{u}_{j,k:n} = \boldsymbol{u}_{j,k:n} - \ell_{jk} \boldsymbol{u}_{k,k:n}$
    end for
end for

- **Memory usage**
  To minimize memory use on the computer, both $\boldsymbol{L}$ and $\boldsymbol{U}$ can be written into the same array as $\boldsymbol{A}$.
- **Time complexity**

$$\sum_{k=1}^{n-1} \sum_{j=k+1}^{n} (n - k + 1) \sim \frac{2}{3} n^3 \text{ flops.}$$

Instability of Naive Guassian Elmination

Consider $\boldsymbol{A} = \begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix}$.

- $\boldsymbol{A}$ has full rank with $\kappa(A) = (3+\sqrt{5})/2 \approx 2.618$ in the $\ell_2 - \textit{norm}$.
- We cannot do Gaussian Elimination without swapping.

Consider $\boldsymbol{A} = \begin{bmatrix} 10^{-20} & 1 \\ 1 & 1 \end{bmatrix}$.

- 10^{20} times the first row is subtracted from the second row:

$$\boldsymbol{L} = \begin{bmatrix} 1 & 0 \\ 10^{20} & 1 \end{bmatrix}, \quad \boldsymbol{U} = \begin{bmatrix} 10^{-20} & 1 \\ 0 & 1 - 10^{20} \end{bmatrix}$$

- Suppose $\epsilon_{\text{mach}} \approx 10^{-16}$. The number $1 - 10^{20}$ will not be represented exactly; it will be rounded to the nearest floating point number. Suppose that this is exactly -10^{20}.

Instability of Naive Guassian Elmination

$$\tilde{\boldsymbol{L}} = \begin{bmatrix} 1 & 0 \\ 10^{20} & 1 \end{bmatrix}, \quad \tilde{\boldsymbol{U}} = \begin{bmatrix} 10^{-20} & 1 \\ 0 & -10^{20} \end{bmatrix} \Rightarrow \tilde{\boldsymbol{L}}\tilde{\boldsymbol{U}} = \begin{bmatrix} 10^{-20} & 1 \\ 1 & 0 \end{bmatrix}$$

- The above example gives $\boldsymbol{A} \neq \tilde{\boldsymbol{L}}\tilde{\boldsymbol{U}}$.
- The error could be large when we solve $\boldsymbol{Ax} = \boldsymbol{b}$. For example, with $\boldsymbol{b} = [1, 0]^\top$ we get $\tilde{\boldsymbol{x}} = [0, 1]^\top$, whereas the correct solution is $\boldsymbol{x} \approx [-1, 1]^\top$.

To summarize, we know

- For certain matrices, Naive Gaussian Elimination fails entirely because it attempts division by zero.
- Gaussian elimination, as presented so far, is unstable for solving general linear systems (from backward substitution)
- It is also unstable at forward substitution (image that x_{kk} is too small)

Pivoting

Pivot At step k of Gaussian elimination, multiples of row k are subtracted from rows $k+1, \dots, n$ of the working matrix $\boldsymbol{X}$ in order to introduce zeros in entry k of these rows. In this operation, row k, column k, and especially the entry x_{kk} play special roles. We call x_{kk} the **pivot**. From every entry in the submatrix $\boldsymbol{X}_{k+1:n,k:n}$ is subtracted the product of a number in row k and a number in column k, divided by x_{kk}

$$\begin{bmatrix} \times & \times & \times & \times & \times \\ & x_{kk} & \times & \times & \times \\ & \times & \times & \times & \times \\ & \times & \times & \times & \times \\ & \times & \times & \times & \times \end{bmatrix} \Longrightarrow \begin{bmatrix} \times & \times & \times & \times & \times \\ & x_{kk} & \times & \times & \times \\ & 0 & \times & \times & \times \\ & 0 & \times & \times & \times \\ & 0 & \times & \times & \times \end{bmatrix}$$

- Why the kth row and column must be chosen?

Pivoting

Pivot At step k of Gaussian elimination, multiples of row k are subtracted from rows $k+1, \dots, n$ of the working matrix $\boldsymbol{X}$ in order to introduce zeros in entry k of these rows. In this operation, row k, column k, and especially the entry x_{kk} play special roles. We call x_{kk} the **pivot**. From every entry in the submatrix $\boldsymbol{X}_{k+1:n,k:n}$ is subtracted the product of a number in row k and a number in column k, divided by x_{kk}

$$\begin{bmatrix} \times & \times & \times & \times & \times \\ & x_{kk} & \times & \times & \times \\ & \times & \times & \times & \times \\ & \times & \times & \times & \times \\ & \times & \times & \times & \times \end{bmatrix} \Longrightarrow \begin{bmatrix} \times & \times & \times & \times & \times \\ & x_{kk} & \times & \times & \times \\ & 0 & \times & \times & \times \\ & 0 & \times & \times & \times \\ & 0 & \times & \times & \times \end{bmatrix}$$

- Why the kth row and column must be chosen?
- We are free to choose any entry of $\boldsymbol{X}_{k:n,k:n}$ as the pivot, as long as it is nonzero.

Pivoting Strategies

- **Complete pivoting** If every entry of $X_{k:n,k:n}$ is considered as a possible pivot at step k, there are $O\left((n-k)^2\right)$ entries to be examined to determine the largest. Summing over n steps, the total cost of selecting pivots becomes $O\left(n^3\right)$ operations, adding significantly to the cost of Gaussian elimination.

Pivoting Strategies

- **Complete pivoting** If every entry of $X_{k:n,k:n}$ is considered as a possible pivot at step k, there are $O\left((n-k)^2\right)$ entries to be examined to determine the largest. Summing over n steps, the total cost of selecting pivots becomes $O\left(n^3\right)$ operations, adding significantly to the cost of Gaussian elimination.
- **Partial pivoting** Only rows are interchanged. The pivot at each step is chosen as the largest of the $n-k+1$ sub diagonal entries in column k, incurring a total cost of only $O(n-k)$ operations for selecting the pivot at each step, $O\left(n^2\right)$ operations overall.

$$\begin{bmatrix} \times & \times & \times & \times & \times \\ & \times & \times & \times & \times \\ & \times & \times & \times & \times \\ & x_{ik} & \times & \times & \times \\ & \times & \times & \times & \times \end{bmatrix} \xrightarrow{\boldsymbol{P_1}} \begin{bmatrix} \times & \times & \times & \times & \times \\ & x_{ik} & \times & \times & \times \\ & \times & \times & \times & \times \\ & \times & \times & \times & \times \\ & \times & \times & \times & \times \end{bmatrix} \xrightarrow{\boldsymbol{L_1}} \begin{bmatrix} \times & \times & \times & \times & \times \\ & x_{ik} & \times & \times & \times \\ & 0 & \times & \times & \times \\ & 0 & \times & \times & \times \\ & 0 & \times & \times & \times \end{bmatrix}$$

After $n-1$ steps, $\boldsymbol{A}$ becomes an upper-triangular matrix $\boldsymbol{U}$:

$$\boldsymbol{L}_{n-1}\boldsymbol{P}_{n-1}\cdots\boldsymbol{L}_2\boldsymbol{P}_2\boldsymbol{L}_1\boldsymbol{P}_1\boldsymbol{A}=\boldsymbol{U}$$

Partial pivoting: Example

Consider $\boldsymbol{A} = \begin{bmatrix} 2 & 1 & 1 & 0 \\ 4 & 3 & 3 & 1 \\ 8 & 7 & 9 & 5 \\ 6 & 7 & 9 & 8 \end{bmatrix}$

Interchange the first and third rows (left-multiplication by $\boldsymbol{P}_1$):

$$\begin{bmatrix} & & 1 & \\ & 1 & & \\ 1 & & & \\ & & & 1 \end{bmatrix} \begin{bmatrix} 2 & 1 & 1 & 0 \\ 4 & 3 & 3 & 1 \\ 8 & 7 & 9 & 5 \\ 6 & 7 & 9 & 8 \end{bmatrix} = \begin{bmatrix} 8 & 7 & 9 & 5 \\ 4 & 3 & 3 & 1 \\ 2 & 1 & 1 & 0 \\ 6 & 7 & 9 & 8 \end{bmatrix}$$

The elimination step now looks like this (left-multiplication by $\boldsymbol{L}_1$)

$$\begin{bmatrix} 1 & & & \\ -1/2 & 1 & & \\ -1/4 & & 1 & \\ -3/4 & & & 1 \end{bmatrix} \begin{bmatrix} 8 & 7 & 9 & 5 \\ 4 & 3 & 3 & 1 \\ 2 & 1 & 1 & 0 \\ 6 & 7 & 9 & 8 \end{bmatrix} = \begin{bmatrix} 8 & 7 & 9 & 5 \\ & -1/2 & -3/2 & -3/2 \\ & -3/4 & -5/4 & -5/4 \\ & 7/4 & 9/4 & 17/4 \end{bmatrix}$$

Partial pivoting: Example (Continued)

The second and fourth rows are interchanged (multiplication by $\boldsymbol{P}_2$)

$$\begin{bmatrix} 1 & & & \\ & & & 1 \\ & & 1 & \\ & 1 & & \end{bmatrix} \begin{bmatrix} 8 & 7 & 9 & 5 \\ & -1/2 & -3/2 & -3/2 \\ & -3/4 & -5/4 & -5/4 \\ & 7/4 & 9/4 & 17/4 \end{bmatrix} = \begin{bmatrix} 8 & 7 & 9 & 5 \\ & 7/4 & 9/4 & 17/4 \\ & -3/4 & -5/4 & -5/4 \\ & -1/2 & -3/2 & -3/2 \end{bmatrix}$$

Second elimination step (multiplication by $\boldsymbol{L}_2$)

$$\begin{bmatrix} 1 & & & \\ & 1 & & \\ & \frac{3}{7} & 1 & \\ & \frac{2}{7} & & 1 \end{bmatrix} \begin{bmatrix} 8 & 7 & 9 & 5 \\ & \frac{7}{4} & \frac{9}{4} & \frac{17}{4} \\ & -\frac{3}{4} & -\frac{5}{4} & -\frac{5}{4} \\ & -\frac{1}{2} & -\frac{3}{2} & -\frac{3}{2} \end{bmatrix} = \begin{bmatrix} 8 & 7 & 9 & 5 \\ & \frac{7}{4} & \frac{9}{4} & \frac{17}{4} \\ & & -\frac{2}{7} & \frac{4}{7} \\ & & -\frac{6}{7} & -\frac{2}{7} \end{bmatrix}$$

Partial pivoting: Example (Continued)

After $\boldsymbol{P}_3$

$$
\begin{bmatrix} 1 & & & \\ & 1 & & \\ & & & 1 \\ & & 1 & \end{bmatrix}
\begin{bmatrix} 8 & 7 & 9 & 5 \\ & \frac{7}{4} & \frac{9}{4} & \frac{17}{4} \\ & & -\frac{2}{7} & \frac{4}{7} \\ & & -\frac{6}{7} & -\frac{2}{7} \end{bmatrix}
=
\begin{bmatrix} 8 & 7 & 9 & 5 \\ & \frac{7}{4} & \frac{9}{4} & \frac{17}{4} \\ & & -\frac{6}{7} & -\frac{2}{7} \\ & & -\frac{2}{7} & \frac{4}{7} \end{bmatrix}
$$

After $\boldsymbol{L}_3$

$$
\begin{bmatrix} 1 & & & \\ & 1 & & \\ & & 1 & \\ & & -\frac{1}{3} & 1 \end{bmatrix}
\begin{bmatrix} 8 & 7 & 9 & 5 \\ & \frac{7}{4} & \frac{9}{4} & \frac{17}{4} \\ & & -\frac{6}{7} & -\frac{2}{7} \\ & & -\frac{2}{7} & \frac{4}{7} \end{bmatrix}
=
\begin{bmatrix} 8 & 7 & 9 & 5 \\ & \frac{7}{4} & \frac{9}{4} & \frac{17}{4} \\ & & -\frac{6}{7} & -\frac{2}{7} \\ & & & \frac{2}{3} \end{bmatrix}
$$

Partial pivoting: Example (Continued)

The final matrix $\boldsymbol{PA} = \boldsymbol{LU}$

$$\begin{bmatrix} & & 1 & \\ & & & 1 \\ & 1 & & \\ 1 & & & \end{bmatrix} \begin{bmatrix} 2 & 1 & 1 & 0 \\ 4 & 3 & 3 & 1 \\ 8 & 7 & 9 & 5 \\ 6 & 7 & 9 & 8 \end{bmatrix} = \begin{bmatrix} 1 & & & \\ \frac{3}{4} & 1 & & \\ \frac{1}{2} & -\frac{2}{7} & 1 & \\ \frac{1}{4} & -\frac{3}{7} & \frac{1}{3} & 1 \end{bmatrix} = \begin{bmatrix} 8 & 7 & 9 & 5 \\ & \frac{7}{4} & \frac{9}{4} & \frac{17}{4} \\ & & -\frac{6}{7} & -\frac{2}{7} \\ & & & \frac{2}{3} \end{bmatrix}$$

- All the subdiagonal entries of $\boldsymbol{L}$ are ≤ 1 in magnitude, a consequence of the property $|x_{kk}| = \max_j |x_{jk}|$ by this partial pivoting.
- $\boldsymbol{L}_3\boldsymbol{P}_3\boldsymbol{L}_2\boldsymbol{P}_2\boldsymbol{L}_1\boldsymbol{P}_1\boldsymbol{A} = \boldsymbol{U}$
- It is equivalent to $\boldsymbol{PA} = \boldsymbol{LU}$, why ?

Gaussian Elimination with Partial Pivoting

Algorithm 2 Gaussian Elimination with Partial Pivoting

$\boldsymbol{U} = \boldsymbol{A}, \boldsymbol{L} = \boldsymbol{I}, \boldsymbol{P} = \boldsymbol{I}$
for $k = 1$ to $n - 1$ **do**
  Select $i \geq k$ to maximize $|u_{ik}|$
  Interchange two rows $u_{k,k:n} \leftrightarrow u_{i,k:n}$
  $\ell_{k,1:k-1} \leftrightarrow \ell_{i,1:k-1}$
  $p_{k,:} \leftrightarrow p_{i,:}$
  for $j = k+1, k+2, \ldots, n$ **do**
    $\ell_{jk} = u_{jk}/u_{kk}$
    $u_{j,k:n} = u_{j,k:n} - \ell_{jk} u_{k,k:n}$
  end for
end for

- **Memory usage** One array as $\boldsymbol{A}$.
- **Time complexity** $\sum_{k=1}^{n-1} \sum_{j=k+1}^{n} (n - k + 1) \sim \frac{2}{3} n^3$ flops.

Cholesky Factorization

Theorem 2.1 (Cholesky Theorem on $\boldsymbol{L}\boldsymbol{L}^\top$-Factorization)

If $\boldsymbol{A}$ is a real, symmetric, and positive definite matrix, then it has a unique factorization, $\mathbf{A} = \mathbf{L}\mathbf{L}^\top$, in which $\mathbf{L}$ is lower triangular with a positive diagonal.

- The complexity of computing $\boldsymbol{L}$ is $\sim n^3/3$ flops
- $\boldsymbol{L}$ is called the *Cholesky factor* of $\boldsymbol{A}$
- It can be interpreted as the "square root" of a p.d. matrix
- It gives a practical method for testing positive definiteness

Intuition of iteration methods

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

where $\boldsymbol{A} \in \mathbb{R}^{n\times n}, \boldsymbol{x} \in \mathbb{R}^n, \boldsymbol{b} \in \mathbb{R}^n$. Suppose $\boldsymbol{x}^0$ be the initial guess of $\boldsymbol{x}^*$. We can measure the estimation error:

$$\boldsymbol{e}^0 \triangleq \boldsymbol{A}^{-1}\boldsymbol{b} - \boldsymbol{x}^0,$$

then the solution can be expressed as

$$\boldsymbol{x}^* = \boldsymbol{x}^0 + \boldsymbol{e}^0.$$

$\boldsymbol{e}^0$ is unknown (hard as the original). Fortunately, we know the residual:

$$\boldsymbol{r}^0 \triangleq \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}^0.$$

Notice that $\boldsymbol{A}\boldsymbol{x}^* = \boldsymbol{A}\boldsymbol{x}^0 + \boldsymbol{A}\boldsymbol{e}^0 \Leftrightarrow \boldsymbol{b} = \boldsymbol{A}\boldsymbol{x}^0 + \boldsymbol{A}\boldsymbol{e}^0$. Hence $\boldsymbol{r}^0 = \boldsymbol{A}\boldsymbol{e}^0$.

Intuition of iteration methods

Intuition: We already know that $\boldsymbol{r}^0 = \boldsymbol{A}\boldsymbol{e}^0$. If there is an $\boldsymbol{M}$ such that $\boldsymbol{M}^{-1}\boldsymbol{A}\boldsymbol{e}^0 \approx \boldsymbol{e}^0$, i.e., $\boldsymbol{M}^{-1}\boldsymbol{A} \approx \boldsymbol{I}$, then we can use $\boldsymbol{M}^{-1}\boldsymbol{r}^0$ to approximate $\boldsymbol{e}^0$. Let

$$\boldsymbol{M}\boldsymbol{z}^0 = \boldsymbol{r}^0 \Leftrightarrow \boldsymbol{z}^0 = \boldsymbol{M}^{-1}\boldsymbol{r}^0 \approx \boldsymbol{e}^0. \tag{4}$$

$\boldsymbol{M}^{-1}$ must be cheap!

We hope $\boldsymbol{x}^1 = \boldsymbol{x}^0 + \boldsymbol{z}^0$ is getting closer to $\boldsymbol{x}^*$. To summarize:

1. **Step 1**: Compute residual: $\boldsymbol{r}^0 = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}^0$
2. **Step 2**: Solve $\boldsymbol{M}\boldsymbol{z}^0 = \boldsymbol{r}^0$ and use $\boldsymbol{M}^{-1}\boldsymbol{r}^0$ to approximate $\boldsymbol{e}^0$
3. **Step 3**: Get a better solution: $\boldsymbol{x}^1 = \boldsymbol{x}^0 + \boldsymbol{z}^0$

Repeat the above steps until the stop condition is satisfied.

A general idea of iteration methods

We summarize the idea and write it as a linear iteration:

Algorithm 3 Iteration($\boldsymbol{x}^0, \boldsymbol{A}, \boldsymbol{b}, \boldsymbol{M}$)

$\boldsymbol{x}^0, \boldsymbol{A}, \boldsymbol{b}, \boldsymbol{M}$ be initial guesses
for $t = 0, 1, \ldots,$ **do**
    Com. Residual $\boldsymbol{r}^t = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}^t$
    Com. Approximate estimation error $\boldsymbol{z}^t = \boldsymbol{M}^{-1}\boldsymbol{r}^t$
    Update $\boldsymbol{x}^{t+1} = \boldsymbol{x}^t + \boldsymbol{z}^t$
end for
Return $\boldsymbol{x}^{t+1}$

- **Q1**: How do we design $\boldsymbol{M}$?
- **Q2**: When will this method converge under what condition?

Jacobi Method

Every i-th equation of $\boldsymbol{Ax} = \boldsymbol{b}$ is

$$\sum_{j=1}^{n} a_{ij} x_j = b_i.$$

To solve the problem, for each x_i at t-th iteration, assume other entries of $\boldsymbol{x}$ remain fixed. This gives

$$x_i^{t+1} = \frac{b_i - \sum_{j \neq i} a_{ij} x_j^t}{a_{ii}}.$$

Notice diagonal elements of $\boldsymbol{A}$ appears in denominator and $\sum_{j \neq i} a_{ij}$ can be decomposed into two parts $\sum_{j<i} a_{ij} + \sum_{j>i} a_{ij}$. Denote

$$\boldsymbol{A} = \boldsymbol{L} + \boldsymbol{D} + \boldsymbol{U}.$$

Jacobi Method

- In the matrix form, the Jacobi method is:

$$
\begin{aligned}
\boldsymbol{x}_0 &= \text{ initial vector} \\
\boldsymbol{x}_{t+1} &= \boldsymbol{D}^{-1}(\boldsymbol{b} - (\boldsymbol{L} + \boldsymbol{U})\boldsymbol{x}_t) \text{ for } t = 0, 1, 2, \ldots
\end{aligned}
$$

- In the form of Fixed-Point Iteration:

$$
\boldsymbol{x}_{t+1} = g(\boldsymbol{x}_t), \text{ where } g(\boldsymbol{x}_t) = \boldsymbol{D}^{-1}(\boldsymbol{b} - (\boldsymbol{L} + \boldsymbol{U})\boldsymbol{x}_t)
$$

To show that Jacobi is FPI, we have

$$
\begin{aligned}
\boldsymbol{A}\boldsymbol{x} &= \boldsymbol{b} \\
(\boldsymbol{D} + \boldsymbol{L} + \boldsymbol{U})\boldsymbol{x} &= \boldsymbol{b} \\
\boldsymbol{D}\boldsymbol{x} &= \boldsymbol{b} - (\boldsymbol{L} + \boldsymbol{U})\boldsymbol{x}.
\end{aligned}
$$

Jacobi Method

Jacobi method:

$$\boldsymbol{x}_0 = \text{ initial vector}$$
$$\boldsymbol{x}_{t+1} = \boldsymbol{D}^{-1}(\boldsymbol{b} - (\boldsymbol{L} + \boldsymbol{U})\boldsymbol{x}_t) \text{ for } t = 0, 1, 2, \ldots$$

Recall we have an iterative method

Algorithm 4 Iteration($\boldsymbol{x}^0, \boldsymbol{A}, \boldsymbol{b}, \boldsymbol{M}$)

1: $\boldsymbol{x}^0$ be initial guesses
2: **for** $t = 0, 1, \ldots,$ **do**
3: $\quad \boldsymbol{r}^t = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}^t$
4: $\quad \boldsymbol{z}^t = \boldsymbol{M}^{-1}\boldsymbol{r}^t$
5: $\quad \boldsymbol{x}^{t+1} = \boldsymbol{x}^t + \boldsymbol{z}^t$
6: **end for**
7: **Return** $\boldsymbol{x}^{t+1}$

Quiz: Find a suitable $\boldsymbol{M}$ and then show that Jacobi method is equivalent to Iteration method. (5 minutes)

Jacobi - An example

An example of the Jacobi method:

$$\begin{bmatrix} 3x_0 + x_1 = 5 \\ x_0 + 2x_1 = 5 \end{bmatrix} \text{ with an initial guess } \begin{bmatrix} x_0 \\ x_1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}.$$

$$\begin{bmatrix} x_0^0 \\ x_1^0 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

$$\begin{bmatrix} x_0^1 \\ x_1^1 \end{bmatrix} = \begin{bmatrix} \frac{5-x_0^0}{3} \\ \frac{5-x_1^0}{2} \end{bmatrix} = \begin{bmatrix} \frac{5}{3} \\ \frac{5}{2} \end{bmatrix}$$

$$\begin{bmatrix} x_0^2 \\ x_1^2 \end{bmatrix} = \begin{bmatrix} \frac{5-x_0^1}{3} \\ \frac{5-x_1^1}{2} \end{bmatrix} = \begin{bmatrix} \frac{5}{6} \\ \frac{5}{3} \end{bmatrix}$$

$$\begin{bmatrix} x_0^3 \\ x_1^3 \end{bmatrix} = \begin{bmatrix} \frac{5-x_0^2}{3} \\ \frac{5-x_1^2}{2} \end{bmatrix} = \begin{bmatrix} \frac{10}{9} \\ \frac{25}{12} \end{bmatrix}$$

Observation: The most recently updated values of the unknowns are not used at each step.

Gauss-Seidel

Gauss-Seidel: the most recently updated values of the unknowns are used at each step. Example:

$$\begin{bmatrix} 3 & 1 & -1 \\ 2 & 4 & 1 \\ -1 & 2 & 5 \end{bmatrix} \begin{bmatrix} u \\ v \\ w \end{bmatrix} = \begin{bmatrix} 4 \\ 1 \\ 1 \end{bmatrix}$$

The iteration goes to

$$\begin{aligned} u_{t+1} &= \frac{4 - v_t + w_t}{3} \\ v_{t+1} &= \frac{1 - 2u_{t+1} - w_t}{4} \\ w_{t+1} &= \frac{1 + u_{t+1} - 2v_{t+1}}{5}. \end{aligned}$$

$$x_i^{t+1} = \frac{b_i - \sum_{j<i} a_{ij} x_j^{t+1} - \sum_{j>i} a_{ij} x_j^t}{aii}, \text{ for } i = 1, 2, \ldots, n. \tag{5}$$

Gauss-Seidel

An alternative way: The most recently updated values of the unknowns are used at each step, even if the updating occurs in the current step.
Gauss-Seidel Method:

$$\boldsymbol{x}^0 = \text{ an initial vector}$$
$$\boldsymbol{x}^{t+1} = \boldsymbol{D}^{-1}\left(\boldsymbol{b} - \boldsymbol{U}\boldsymbol{x}^t - \boldsymbol{L}\boldsymbol{x}^{t+1}\right) \text{ for } k = 0, 1, 2, \ldots.$$

Algorithm 5 Iteration($\boldsymbol{x}^0, \boldsymbol{A}, \boldsymbol{b}, \boldsymbol{M}$)

1: $\boldsymbol{x}^0$ be initial guesses
2: **for** $t = 0, 1, \ldots,$ **do**
3: $\quad \boldsymbol{r}^t = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}^t$
4: $\quad \boldsymbol{z}^t = \boldsymbol{M}^{-1}\boldsymbol{r}^t$
5: $\quad \boldsymbol{x}^{t+1} = \boldsymbol{x}^t + \boldsymbol{z}^t$
6: **end for**
7: **Return** $\boldsymbol{x}^{t+1}$

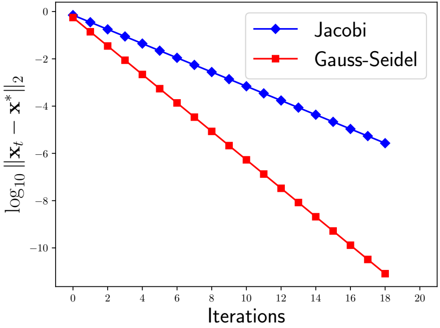
Quiz: Find a suitable $\boldsymbol{M}$ and then show that Gauss-Seidel method is equivalent to Iteration method. (5 minutes)

Comparison between Jacobi and Gauss-Seidel

An example:

$$\begin{bmatrix} 2x_0 - x_1 & = 1 \\ -1x_0 + 2x_1 & = 1 \end{bmatrix}, \text{ where } \boldsymbol{A} = \begin{bmatrix} 2 & -1 \\ -1 & 2 \end{bmatrix}, \boldsymbol{b} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

We use Jacobi and Gauss-Seidel with $\boldsymbol{x}_0 = [0, 0]^\top$.



Gauss-Seidel converges faster than Jacobi in this example.

Successive Over-Relaxation

Can we do better?
Idea of Successive Over-Relaxation: define each component of the new guess $\boldsymbol{x}^{t+1}$ as a weighted average of ω times the Gauss-Seidel formula and $1-\omega$ times the current guess $\boldsymbol{x}^{t}$.
SOR method:

$$x_i^{(t+1)} = (1-\omega)x_i^{(t)} + \frac{\omega}{a_{ii}}\left(b_i - \sum_{j<i} a_{ij}x_j^{(t+1)} - \sum_{j>i} a_{ij}x_j^{(t)}\right), \qquad (6)$$

where $\quad i = 1, 2, \ldots, n$.

- SOR method is equivalent to Iteration method when $\boldsymbol{M} = \frac{\boldsymbol{D}}{\omega} - \boldsymbol{L}$.