Linear Systems – Part 2

Intensive Computation

Annalisa Massini - 2015/2016

Methods for solving linear equations

- **Direct methods**: find the ***exact*** solution in a finite number of steps
- **Iterative methods**: produce a sequence of approximate solutions hopefully converging to the exact solution

Cholesky Factorization Method

Cholesky factorization Method for solving $A\,x = b$

- A **direct** method
- The idea is:
  - every **positive definite matrix** A can be factored as $A=LL^T$

A **complex matrix** A is called **positive definite** if $\boxed{\Re[x^* Ax] > 0}$ for all nonzero complex vectors $x \in C$, where x^* denotes the conjugate transpose

If A is a **real matrix** then A is **positive definite** if $\boxed{x^T Ax > 0}$ where denotes x^T the transpose

Cholesky Factorization Method

Cholesky factorization Method for solving $A\,x = b$

- A **direct** method
- The idea is:
  - every **positive definite matrix** A can be factored as $A=LL^T$
  - where **L is lower triangular** with **positive diagonal elements**
  - L is called the Cholesky factor of A
  - L can be interpreted as ‘square root’ of a positive define matrix
- The method:
  - Provides an exact result (***ignoring roundoff***)
  - Is *computationally expensive*

Cholesky Factorization Method

We partition matrix *A* as *LL^T* as follows

$$\begin{bmatrix} a_{11} & A_{12}^T \\ A_{21} & A_{22} \end{bmatrix} = \begin{bmatrix} l_{11} & 0 \\ L_{21} & L_{22} \end{bmatrix} \begin{bmatrix} l_{11} & L_{21}^T \\ 0 & L_{22}^T \end{bmatrix}$$

$$= \begin{bmatrix} l_{11}^2 & l_{11} L_{21}^T \\ l_{11} L_{21} & L_{21} L_{21}^T + L_{22} L_{22}^T \end{bmatrix}$$

Cholesky Factorization Method

We have the following steps:

$$\begin{bmatrix} a_{11} & A_{12}^T \\ A_{21} & A_{22} \end{bmatrix} = \begin{bmatrix} l_{11} & 0 \\ L_{21} & L_{22} \end{bmatrix} \begin{bmatrix} l_{11} & L_{21}^T \\ 0 & L_{22}^T \end{bmatrix}$$

- **Step 1 -** Determine l_{11} and L_{21}

$$l_{11} = \sqrt{a_{11}} \qquad L_{21} = \frac{1}{l_{11}} A_{21}$$

- **Step 2 -** Compute L_{22} from

$$A_{22} - L_{21}L_{21}^T = L_{22}L_{22}^T$$

- This is a Cholesky factorization of order $n-1$

Cholesky Factorization Method

Proof that the algorithm works for positive definite A of order n

- if A is positive definite then $a_{11} > 0$

- if A is positive definite then

$$A_{22} - L_{21}L_{21}^T = A_{22} - \frac{1}{l_{11}} A_{21}A_{21}^T$$

is positive definite

- Hence we can apply again the factorization on the

Cholesky Factorization Method

Example

$$\begin{bmatrix} 25 & 15 & -5 \\ 15 & 18 & 0 \\ -5 & 0 & 11 \end{bmatrix} = \begin{bmatrix} l_{11} & 0 & 0 \\ l_{21} & l_{22} & 0 \\ l_{31} & l_{32} & l_{33} \end{bmatrix} \begin{bmatrix} l_{11} & l_{32} & l_{31} \\ 0 & l_{22} & l_{32} \\ 0 & 0 & l_{33} \end{bmatrix}$$

Cholesky Factorization Method

Example

$$\begin{bmatrix} 25 & 15 & -5 \\ 15 & 18 & 0 \\ -5 & 0 & 11 \end{bmatrix} = \begin{bmatrix} l_{11} & 0 & 0 \\ l_{21} & l_{22} & 0 \\ l_{31} & l_{32} & l_{33} \end{bmatrix} \begin{bmatrix} l_{11} & l_{32} & l_{31} \\ 0 & l_{22} & l_{32} \\ 0 & 0 & l_{33} \end{bmatrix}$$

First column of L $\quad l_{11} = \sqrt{a_{11}} = 5 \qquad L_{21} = \frac{1}{l_{11}} A_{21}$

$$\begin{bmatrix} 25 & 15 & -5 \\ 15 & 18 & 0 \\ -5 & 0 & 11 \end{bmatrix} = \begin{bmatrix} 5 & 0 & 0 \\ 3 & l_{22} & 0 \\ -1 & l_{32} & l_{33} \end{bmatrix} \begin{bmatrix} 5 & 3 & -1 \\ 0 & l_{22} & l_{32} \\ 0 & 0 & l_{33} \end{bmatrix}$$

Cholesky Factorization Method

Example

$$\begin{bmatrix} 25 & 15 & -5 \\ 15 & 18 & 0 \\ -5 & 0 & 11 \end{bmatrix} = \begin{bmatrix} l_{11} & 0 & 0 \\ l_{21} & l_{22} & 0 \\ l_{31} & l_{32} & l_{33} \end{bmatrix} \begin{bmatrix} l_{11} & l_{32} & l_{31} \\ 0 & l_{22} & l_{32} \\ 0 & 0 & l_{33} \end{bmatrix}$$

Second column of L – Apply the same procedure to

$A_{22} - L_{21}L_{21}^T$

$(= L_{22}L_{22}^T)$

$$A_{22} - L_{21}L_{21}^T = \begin{bmatrix} 18 & 0 \\ 0 & 11 \end{bmatrix} - \begin{bmatrix} 3 \\ -1 \end{bmatrix} \begin{bmatrix} 3 & -1 \end{bmatrix} = \begin{bmatrix} l_{22} & 0 \\ l_{32} & l_{33} \end{bmatrix} \begin{bmatrix} l_{22} & l_{32} \\ 0 & l_{33} \end{bmatrix}$$

$$l_{22} = \sqrt{a_{22}} = 3$$

$$L_{32} = \frac{1}{l_{22}} A_{32}$$

$$\begin{bmatrix} 9 & 3 \\ 3 & 10 \end{bmatrix} = \begin{bmatrix} 3 & 0 \\ 1 & l_{33} \end{bmatrix} \begin{bmatrix} 3 & 1 \\ 0 & l_{33} \end{bmatrix}$$

Cholesky Factorization Method

Example

$$\begin{bmatrix} 25 & 15 & -5 \\ 15 & 18 & 0 \\ -5 & 0 & 11 \end{bmatrix} = \begin{bmatrix} l_{11} & 0 & 0 \\ l_{21} & l_{22} & 0 \\ l_{31} & l_{32} & l_{33} \end{bmatrix} \begin{bmatrix} l_{11} & l_{32} & l_{31} \\ 0 & l_{22} & l_{32} \\ 0 & 0 & l_{33} \end{bmatrix}$$

Third column of L $\quad l_{33}^2 = 10 - 1 \qquad l_{33} = 3$

$$\begin{bmatrix} 25 & 15 & -5 \\ 15 & 18 & 0 \\ -5 & 0 & 11 \end{bmatrix} = \begin{bmatrix} 5 & 0 & 0 \\ 3 & 3 & 0 \\ -1 & 1 & 3 \end{bmatrix} \begin{bmatrix} 5 & 3 & -1 \\ 0 & 3 & 1 \\ 0 & 0 & 3 \end{bmatrix}$$

Cholesky Factorization Method

Solve $A\,x = b$ ***A positive definite of order n***

Step 1 $A = LL^T$ ➔ **LL^T factorization**

Step 2 $Ly = b$ ➔ Solve by forward substitution

Step 3 $L^Tx = y$ ➔ Solve by backward substitution

LL^T decomposition – Algorithm cost

We have:

- Multiplication and division operations for the factorization: ***$O(n^3)$***
- Multiplication and division operations for solving the triangular systems: ***$O(n^2)$***
- Addition and subtraction operations: ***$O(n^3)$***
- *Anyway usually we consider the number of floating point operations FLOPs*

Inverse of a positive definite matrix

Suppose A is positive definite with Cholesky factorization LL^T

- L is invertible (its diagonal elements are nonzero)

- $X = L^{-T}L^{-1}$ is a **right inverse** of A:

$$AX = LL^T L^{-T} L^{-1} = LL^{-1} = I$$

- $X = L^{-T}L^{-1}$ is a **left inverse** of A:

$$XA = L^{-T} L^{-1} L L^T = L^{-T} L^T = I$$

- hence ***A* is invertible with inverse**:

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

Cholesky for sparse matrices

- If A is very sparse, then *L* is often (but not always) sparse
- If L is sparse, the **cost of the factorization is less than (1/3)n^3**
- Exact cost depends on:
  - *n*
  - *#nonzero* elements
  - sparsity pattern
- Very large sets of equations ($n \sim 10^6$) are solved by exploiting sparsity

Cholesky for sparse matrices

- Consider a ***sparse systems***

$$\begin{bmatrix} 1 & a^T \\ a & I \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} b \\ c \end{bmatrix}$$

- Factorization

$$\begin{bmatrix} 1 & a^T \\ a & I \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ a & L_{22} \end{bmatrix} \begin{bmatrix} 1 & a^T \\ 0 & L_{22}^T \end{bmatrix}$$

Cholesky for sparse matrices

- Fill-in effect

$$\begin{bmatrix} * & * & * & * & * & * & * & * \\ * & * & & & & & & \\ * & & * & & & & & \\ * & & & * & & & & \\ * & & & & * & & & \\ * & & & & & * & & \\ * & & & & & & * & \\ * & & & & & & & * \end{bmatrix} = \begin{bmatrix} * & & & & & & & \\ * & * & & & & & & \\ * & * & * & & & & & \\ * & * & * & * & & & & \\ * & * & * & * & * & & & \\ * & * & * & * & * & * & & \\ * & * & * & * & * & * & * & \\ * & * & * & * & * & * & * & * \end{bmatrix} \times \begin{bmatrix} * & * & * & * & * & * & * & * \\ & * & * & * & * & * & * & * \\ & & * & * & * & * & * & * \\ & & & * & * & * & * & * \\ & & & & * & * & * & * \\ & & & & & * & * & * \\ & & & & & & * & * \\ & & & & & & & * \end{bmatrix}$$

- Factorization can produce a 100% fill-in

Cholesky for sparse matrices

- We can **reorder the equations** of the systems

$$\begin{bmatrix} 1 & a^T \\ a & I \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} b \\ c \end{bmatrix} \quad \Rightarrow \quad \begin{bmatrix} I & a \\ a^T & 1 \end{bmatrix} \begin{bmatrix} v \\ u \end{bmatrix} = \begin{bmatrix} c \\ b \end{bmatrix}$$

- Factorization

$$\begin{bmatrix} I & a \\ a^T & 1 \end{bmatrix} = \begin{bmatrix} I & 0 \\ a^T & \sqrt{1-a^T a} \end{bmatrix} \begin{bmatrix} I & a \\ 0 & \sqrt{1-a^T a} \end{bmatrix}$$

Cholesky for sparse matrices

- We can reorder the equation of the systems

$$\begin{bmatrix} 1 & a^T \\ a & I \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} b \\ c \end{bmatrix} \quad \Rightarrow \quad \begin{bmatrix} I & a \\ a^T & 1 \end{bmatrix} \begin{bmatrix} v \\ u \end{bmatrix} = \begin{bmatrix} c \\ b \end{bmatrix}$$

$$\begin{bmatrix} * & & & & & & & * \\ & * & & & & & & * \\ & & * & & & & & * \\ & & & * & & & & * \\ & & & & * & & & * \\ & & & & & * & & * \\ & & & & & & * & * \\ * & * & * & * & * & * & * & * \end{bmatrix} = \begin{bmatrix} * & & & & & & & \\ & * & & & & & & \\ & & * & & & & & \\ & & & * & & & & \\ & & & & * & & & \\ & & & & & * & & \\ & & & & & & * & \\ * & * & * & * & * & * & * & * \end{bmatrix} \times \begin{bmatrix} * & & & & & & & * \\ & * & & & & & & * \\ & & * & & & & & * \\ & & & * & & & & * \\ & & & & * & & & * \\ & & & & & * & & * \\ & & & & & & * & * \\ & & & & & & & * \end{bmatrix}$$

- Factorization with zero fill-in

Permutation matrices

Permutation matrix - identity matrix with rows reordered

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix} \qquad \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

The vector Ax is a permutation of x

$$\begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} x_2 \\ x_3 \\ x_1 \end{bmatrix}$$

$A^T x$ is the inverse permutation applied to x

$$\begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} x_3 \\ x_1 \\ x_2 \end{bmatrix}$$

$A^T A = A A^T = I$ then **A is invertible and $A^{-1} = A^T$**

Permutation matrices

Solving *Ax=b* when *A* is a permutation matrix

The solution of $Ax=b$ is vector $x = A^T b$

Example

$$\begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 3.5 \\ -3.2 \\ 8.0 \end{bmatrix} \qquad \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} 3.5 \\ -3.2 \\ 8.0 \end{bmatrix} = \begin{bmatrix} 8.0 \\ 3.5 \\ -3.2 \end{bmatrix}$$

The solution is $x=[8.0, 3.5, -3.2]^T$

Sparse Cholesky factorization

- If A is sparse and positive definite, it is usually factored as

$$A = PLL^TP^T$$

  - where ***P*** **is a permutation matrix** and ***L*** **is lower triangular with positive diagonal elements**

- ***Interpretation***: we permute the rows and columns of A and factor

$$P^TAP = LL^T$$

- Note that:
  - **choice of *P* greatly affects the sparsity *L***
  - many heuristic methods exist for ***selecting good permutation matrices*** *P*

Solving sparse positive definite eqs

- If A is sparse and positive definite, we solve $Ax = b$ via factorization

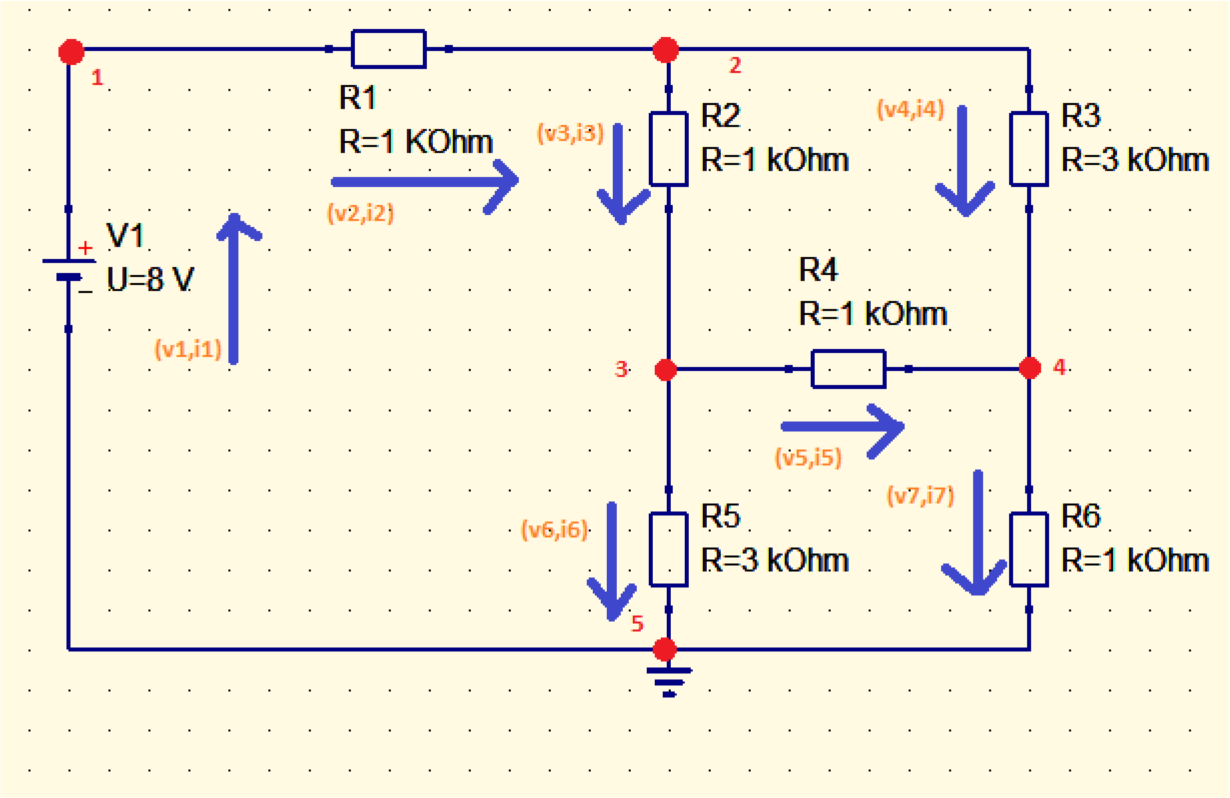
$$A = PLL^T P^T$$

- ***Algorithm***
  - $b' = P^T b$
  - solve $Lz = b'$ by forward substitution
  - solve $L^T y = z$ by backward substitution
  - 4. $x = Py$:

- That is **we solve $(P^T A P)y = b'$** using the Clolesky factorization of $P^T AP$

Example

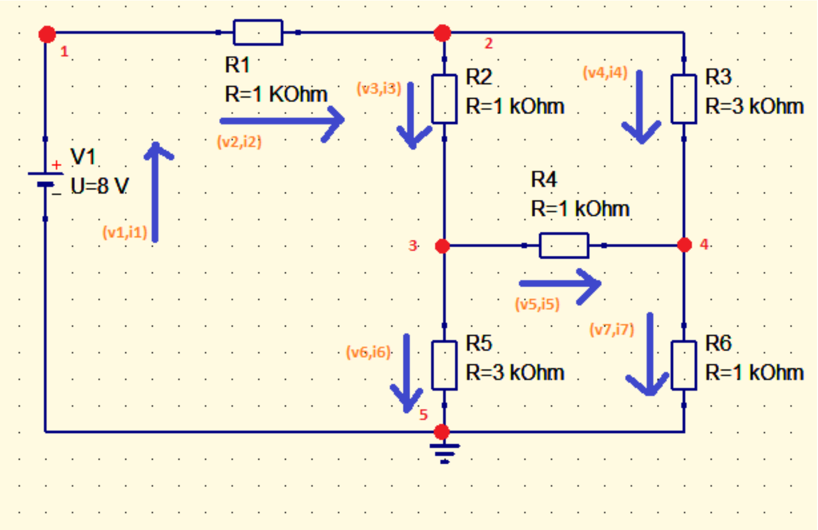
- Example of a sparse system
- Consider an electric circuit with only resistors



Example

To solve an electric circuit with only resistors, we can write the equations describing the circuit where:

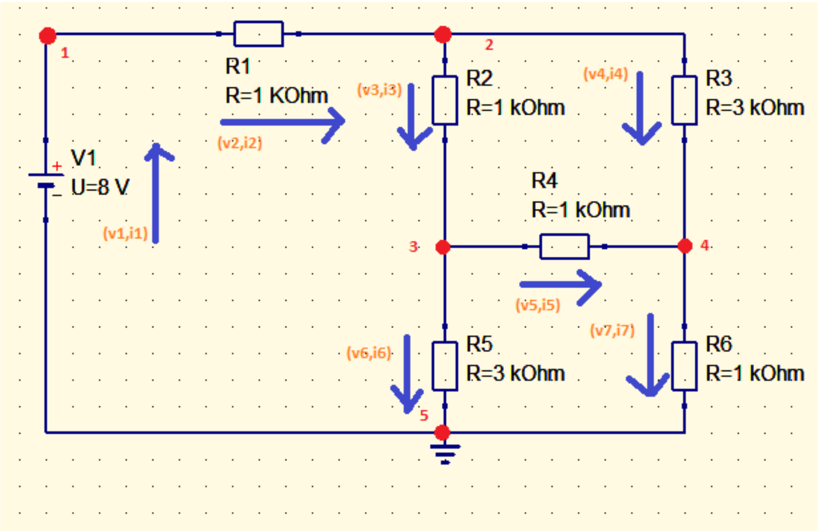
- Unknowns are
  - B branch currents (i)
  - N node voltages (e)
  - B branch voltages (v)
- Equations are
  - N+B Conservation Laws
  - B Constitutive Equations



For our circuit

- v1, v2, v3, v4, v5, v6, v7 ***branch voltages***
- î1, i2, i3, i4, i5, i6, i7 ***branch currents***
- ê1, e2, e3, e4 ***node voltages*** with voltage of node 5 equal to 0

Example



Branch Constitutive Equations

- -v1 = V 1
- v2 - R1 x i2 = 0
- v3 - R2 x i3 = 0
- v4 - R3 x i4 = 0
- v5 - R4 x i5 = 0
- v6 - R5 x i6 = 0
- v7 - R6 x i7 = 0

Kirchhoff's Current Law (KCL)

- -i1 + i2 = 0 (node 1)
- -i2 + i3 + i4 = 0 (node 2)
- -i3 + i5 + i6 = 0 (node 3)
- -i4 - i5 + i7 = 0 (node 4)

Kirchhoff's Voltage Law (KVL)

- v1 + e1 = 0
- v2 - e1 + e2 = 0
- v3 - e2 + e3 = 0
- v4 - e2 + e4 = 0
- v5 - e3 + e4 = 0
- v6 - e3 = 0
- v7 - e4 = 0

Example

- In matrix form

$$
\left[\begin{array}{ccccccc|ccccccc|cccc}
-1 & 1 & & & & & & & & & & & & & & & & \\
 & -1 & 1 & 1 & & & & & & & & & & & & & & \\
 & & -1 & & 1 & 1 & & & & & & & & & & & & \\
 & & & -1 & -1 & & 1 & & & & & & & & & & & \\
\hline
 & & & & & & & 1 & & & & & & & 1 & & & \\
 & & & & & & & & 1 & & & & & & -1 & 1 & & \\
 & & & & & & & & & 1 & & & & & & -1 & 1 & \\
 & & & & & & & & & & 1 & & & & & -1 & & 1 \\
 & & & & & & & & & & & 1 & & & & & -1 & 1 \\
 & & & & & & & & & & & & 1 & & & & -1 & \\
 & & & & & & & & & & & & & 1 & & & & -1 \\
\hline
 & & & & & & & -1 & & & & & & & & & & \\
-R_1 & & & & & & & & 1 & & & & & & & & & \\
 & -R_2 & & & & & & & & 1 & & & & & & & & \\
 & & -R_3 & & & & & & & & 1 & & & & & & & \\
 & & & -R_4 & & & & & & & & 1 & & & & & & \\
 & & & & -R_5 & & & & & & & & 1 & & & & & \\
 & & & & & -R_6 & & & & & & & & 1 & & & &
\end{array}\right]
\left[\begin{array}{c}
i_1 \\ i_2 \\ i_3 \\ i_4 \\ i_5 \\ i_6 \\ i_7 \\ \hline v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \\ v_6 \\ v_7 \\ \hline e_1 \\ e_2 \\ e_3 \\ e_4
\end{array}\right]
=
\left[\begin{array}{c}
0 \\ 0 \\ 0 \\ 0 \\ \hline 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ \hline V_1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0
\end{array}\right]
$$

Iterative methods

Iterative methods for solving $Ax = b$

- Begin with an **approximation** to the solution x_0
- Then provide a **series of improved approximations** $x_1, x_2, \ldots$
- That **converge** to the exact solution

Iterative methods

- The method can be **stopped** as soon as the approximations x_i have converged to an **acceptable precision** (which might also be something as 10^{-3})
- With a **direct method**, the process of elimination and back-substitution has to be carried right through to **completion**, or else **abandoned** altogether

Iterative methods

- The **main attraction of iterative methods** is that for certain problems (particularly those where the matrix A is large and sparse) they are **much faster than direct methods**

- On the other hand, **iterative methods can be unreliable**: for some problems they may exhibit very **slow convergence**, or they may **not converge** at all

The Jacobi Method

- Consider the system

$$
\begin{aligned}
a_{11}x_1 + a_{12}x_2 + a_{13}x_3 + \cdots + a_{1n}x_n &= b_1 \\
a_{21}x_1 + a_{22}x_2 + a_{23}x_3 + \cdots + a_{2n}x_n &= b_2 \\
\ldots\ldots\ldots\ldots\ldots\ldots \\
a_{n1}x_1 + a_{n2}x_2 + a_{n3}x_3 + \cdots + a_{nn}x_n &= b_n
\end{aligned}
$$

The Jacobi Method

$$
\begin{aligned}
a_{11}x_1 + a_{12}x_2 + a_{13}x_3 + \cdots + a_{1n}x_n &= b_1 \\
a_{21}x_1 + a_{22}x_2 + a_{23}x_3 + \cdots + a_{2n}x_n &= b_2 \\
\cdots\cdots\cdots\cdots\cdots \\
a_{n1}x_1 + a_{n2}x_2 + a_{n3}x_3 + \cdots + a_{nn}x_n &= b_n
\end{aligned}
$$

- **Rewrite the system** in the form:

$$
\begin{aligned}
x_1 &= -\frac{a_{12}}{a_{11}}x_2 - \frac{a_{13}}{a_{11}}x_3 - \cdots\cdots - \frac{a_{1n}}{a_{11}}x_n + \frac{b_1}{a_{11}} \\
x_2 &= -\frac{a_{21}}{a_{22}}x_1 - \frac{a_{23}}{a_{22}}x_3 - \cdots\cdots - \frac{a_{2n}}{a_{22}}x_n + \frac{b_2}{a_{22}} \\
&\cdots\cdots\cdots\cdots\cdots \\
x_n &= -\frac{a_{n1}}{a_{nn}}x_1 - \frac{a_{n3}}{a_{nn}}x_2 - \cdots - \frac{a_{nn-1}}{a_{nn}}x_{n-1} + \frac{b_n}{a_{nn}}
\end{aligned}
$$

The Jacobi Method

- Consider $x^{(0)} = (x_1^{(0)}, x_2^{(0)}, x_3^{(0)}, \cdots, x_n^{(0)})$ as an **initial approximation** of the solution and **substitute** $x_i^{(0)}$ in the eqs

$$x_1^{(1)} = -\frac{a_{12}}{a_{11}} x_2^{(0)} - \frac{a_{13}}{a_{11}} x_3^{(0)} - \cdots\cdots - \frac{a_{1n}}{a_{11}} x_n^{(0)} + \frac{b_1}{a_{11}}$$

$$x_2^{(1)} = -\frac{a_{21}}{a_{22}} x_1^{(0)} - \frac{a_{23}}{a_{22}} x_3^{(0)} - \cdots\cdots - \frac{a_{2n}}{a_{22}} x_n^{(0)} + \frac{b_2}{a_{22}}$$

$$\cdots\cdots\cdots\cdots\cdots$$

$$x_n^{(1)} = -\frac{a_{n1}}{a_{nn}} x_1^{(0)} - \frac{a_{n2}}{a_{nn}} x_2^{(0)} - \cdots - \frac{a_{nn-1}}{a_{nn}} x_{n-1}^{(0)} + \frac{b_n}{a_{nn}}$$

The Jacobi Method

- **Iterate** the substitution of $x^{(k)} = (x_1^{(k)}, x_2^{(k)}, x_3^{(k)}, \cdots, x_n^{(k)})$ such that at each step a **new solution approximation** is obtained

$$x_1^{(k+1)} = -\frac{a_{12}}{a_{11}} x_2^{(k)} - \frac{a_{13}}{a_{11}} x_3^{(k)} - \cdots\cdots - \frac{a_{1n}}{a_{11}} x_n^{(k)} + \frac{b_1}{a_{11}}$$

$$x_2^{(k+1)} = -\frac{a_{21}}{a_{22}} x_1^{(k)} - \frac{a_{23}}{a_{22}} x_3^{(k)} - \cdots\cdots - \frac{a_{2n}}{a_{22}} x_n^{(k)} + \frac{b_2}{a_{22}}$$

$$\cdots\cdots\cdots\cdots\cdots$$

$$x_n^{(k+1)} = -\frac{a_{n1}}{a_{nn}} x_1^{(k)} - \frac{a_{n2}}{a_{nn}} x_2^{(k)} - \cdots - \frac{a_{nn-1}}{a_{nn}} x_{n-1}^{(k)} + \frac{b_n}{a_{nn}}$$

The Jacobi Method

Example

- Consider the system

$$
\begin{aligned}
10x_1 - x_2 + 2x_3 &= 6 \\
-x_1 + 11x_2 - x_3 + 3x_4 &= 25 \\
2x_1 - x_2 + 10x_3 - x_4 &= -11 \\
3x_2 - x_3 + 8x_4 &= 15
\end{aligned}
$$

The Jacobi Method

Example

- Rewrite the system

$$
\begin{aligned}
x_1 &= \frac{1}{10}x_2 - \frac{1}{5}x_3 + \frac{3}{5} \\
x_2 &= \frac{1}{11}x_1 + \frac{1}{11}x_3 - \frac{3}{11}x_4 + \frac{25}{11} \\
x_3 &= -\frac{1}{5}x_1 + \frac{1}{10}x_2 + \frac{1}{10}x_4 - \frac{11}{10} \\
x_4 &= -\frac{3}{8}x_2 + \frac{1}{8}x_3 + \frac{15}{8}
\end{aligned}
$$

The Jacobi Method

Example

- Start iterations with $x^{(0)} = (0, 0, 0, 0)$

$$x_1^{(1)} = \frac{1}{10}x_2^{(0)} - \frac{1}{5}x_3^{(0)} + \frac{3}{5} = 0.6000$$

$$x_2^{(1)} = \frac{1}{11}x_1^{(0)} + \frac{1}{11}x_3^{(0)} - \frac{3}{11}x_4^{(0)} + \frac{25}{11} = 2.2727$$

$$x_3^{(1)} = -\frac{1}{5}x_1^{(0)} + \frac{1}{10}x_2^{(0)} + \frac{1}{10}x_4^{(0)} - \frac{11}{10} = -1.1000$$

$$x_4^{(1)} = -\frac{3}{8}x_2^{(0)} + \frac{1}{8}x_3^{(0)} + \frac{15}{8} = 1.8750$$

The Jacobi Method

Example

- Continuing with iterations, we obtain the sequence of approximations shown in the Table

k	0	1	2	3	4	5	6	7	8	9
$x_1^{(k)}$	0.0000	0.6000	1.0473	0.9326	1.0152	0.9890	1.0032	0.9981	1.0006	0.9997
$x_2^{(k)}$	0.0000	2.2727	1.7159	2.0533	1.9537	2.0114	1.9922	2.0023	1.9987	2.0004
$x_3^{(k)}$	0.0000	−1.1000	−0.8852	−1.0493	−0.9681	−1.0103	−0.9945	−1.0020	−0.9990	−1.0004
$x_4^{(k)}$	0.0000	1.8750	0.8852	1.1309	0.9739	1.0214	0.9944	1.0036	0.9989	1.0006

The Jacobi Method in Matrix Form

- Consider the system *Ax=b*

$$\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ b_4 \end{bmatrix}$$

- We **split *A* into**

$$A = \begin{bmatrix} a_{11} & 0 & \cdots & 0 \\ 0 & a_{22} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & a_{nn} \end{bmatrix} - \begin{bmatrix} 0 & \cdots & & 0 \\ -a_{21} & \cdots & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ -a_{n1} & \cdots & -a_{nn-1} & 0 \end{bmatrix} \begin{bmatrix} 0 & -a_{12} & \cdots & -a_{1n} \\ \vdots & \vdots & \cdots & \vdots \\ 0 & 0 & \ddots & -a_{n-1n} \\ 0 & 0 & \cdots & 0 \end{bmatrix} = D - (-L - U)$$

- That is ***Ax=b*** is transformed in ***(D-(-L-U))x=b***

The Jacobi Method in Matrix Form

- Assume D[-1] exists

$$D^{-1} = \begin{bmatrix} \frac{1}{a_{11}} & 0 & \cdots & 0 \\ 0 & \frac{1}{a_{22}} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \frac{1}{a_{nn}} \end{bmatrix}$$

- Then $x = D^{-1}(-L-U)x + D^{-1}b$
- The matrix form of Jacobi iterative method is

$$x^{(k+1)} = D^{-1}(-L-U)x^{(k)} + D^{-1}b \qquad k = 0,1,2,\ldots$$

The Jacobi Method

- We need a **stopping criterion**
- We are interested in the error $\boldsymbol{e}$ at each iteration between the true solution $\boldsymbol{x}$ and the approximation $\boldsymbol{x}^{(k)}$: $\boldsymbol{e}^{(k)} = \boldsymbol{x} - \boldsymbol{x}^{(k)}$
- Obviously, we don't usually know the true solution $\boldsymbol{x}$
- To better understand the behavior of an iterative method, we can consider a system $A\boldsymbol{x} = \boldsymbol{b}$ for which we *do* know the true solution and analyze how quickly the approximations are converging to the true solution

The Jacobi Method

- We can consider different ways of measuring the error:
  - $x^{(k+1)} - x^{(k)}$
  - $e^{(k)} = x - x^{(k)}$ where x is the exact solution
  - $\left\| e^{(k)} \right\|$
  - $\dfrac{\left\| x^{(k+1)} - x^{(k)} \right\|}{\left\| x^{(k+1)} \right\|}$

We consider norm l^2, that is:

$$\left\| x \right\|_2 = \sqrt{x_1^2 + x_2^2 + \cdots + x_n^2}$$

- We use one of the previous measures asking that it is $< \varepsilon$

The Jacobi Method

- **Theorem** The Jacobi method converges if the coefficient matrix *A* is a **strictly diagonally dominant matrix**
- An *nxn* matrix *A* is strictly diagonally dominant if the **absolute value** of each entry on the **main diagonal** is greater than the sum of the absolute values of the other entries in the same row:

$$
\begin{aligned}
&|a_{11}| > |a_{12}| + |a_{13}| + \cdots + |a_{1n}| \\
&|a_{22}| > |a_{21}| + |a_{23}| + \cdots + |a_{2n}| \\
&\ldots \\
&|a_{nn}| > |a_{n1}| + |a_{n3}| + \cdots + |a_{n,n-1}|
\end{aligned}
$$

Jacobi method – Algorithm cost

- Multiplication operations for each iteration: *n(n-1)* → ***O(n²)***
- If we do *k* iteration the cost is: *kn(n-1)*
- We also have the divisions with a_{ii} during the first iteration → the cost is kn^2
- To decide **between Gauss and Jacobi** methods we evaluate:

$$kn^2 < \frac{1}{3}n^3 \quad \Rightarrow \quad k < \frac{1}{3}n$$

Hence when using the ***Jacobi method*** *we need to* ***evaluate how many iterations*** *are needed before stopping*