Chapter 03

IS5306: Numerical Solution of System of Linear Equations

MS. M.W.S. Randunu

Department of Interdisiplinary Studies,University of Ruhuna.

November 6, 2024

Introduction

Numerical methods for solving systems of linear equations are divided into two categories:

1. *Direct methods*
   Direct methods, such as Gaussian elimination and LU decomposition, provide exact solutions in a finite number of steps but may become impractical for very large systems due to computational and memory constraints.
2. *Iterative methods*
   On the other hand, iterative methods, like the Jacobi and Gauss-Seidel methods, provide approximate solutions by successively refining an initial guess, making them suitable for large and sparse systems where direct methods may be too costly.

Linear System of Equations

A system of n linear equations with n variables can be represented as:

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

where:

- $x_1, x_2, \ldots, x_n$ are the n unknowns (variables),
- a_{ij} are the coefficients of the system, with i representing the equation number and j the variable's index,
- $b_1, b_2, \ldots, b_n$ are the constants on the right-hand side of the equations.

A linear system can be transformed into a new system that is easier to solve, while maintaining the same solutions.

During this process, it is not necessary to rewrite the full equations or track the variables $x_1, x_2, \ldots, x_n$, as long as they stay in the same columns. Only the coefficients and the constants on the right-hand side change.

This is why we often represent a linear system with a matrix, which provides all the necessary information in a compact and computer-friendly form.

Matrices and vectors

A matrix is a rectangular array of numbers or other mathematical objects arranged in rows and columns. It has the general form:

$$\begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{pmatrix}$$

- The size of a matrix is defined by the number of rows and columns it contains. A matrix with m rows and n columns is called an $m \times n$ matrix, or m-by-n matrix, where m and n are called its dimensions.
- The individual items in a matrix are called its elements or entries.
- A square matrix is an $n \times n$ matrix. The main or principal diagonal of a square matrix consists of the elements a_{ii}, where $i = 1, 2, \ldots, n$, running from the upper left to the lower right.
- An $1 \times n$ matrix, such as $\mathbf{y} = \begin{pmatrix} y_1 & y_2 & \cdots & y_n \end{pmatrix}$, is called an n-dimensional row vector. Similarly, an $n \times 1$ matrix, such as $\mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix}$, is called an n-dimensional column vector.

Special Types of Square Matrices

- **Diagonal Matrix:**

$$D = \begin{pmatrix} d_1 & 0 & 0 & \cdots & 0 \\ 0 & d_2 & 0 & \cdots & 0 \\ 0 & 0 & d_3 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & d_n \end{pmatrix}$$

- **Identity Matrix:**

$$I = \begin{pmatrix} 1 & 0 & 0 & \cdots & 0 \\ 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \end{pmatrix}$$

- **Upper Triangular Matrix:**

$$A = \begin{pmatrix} a_{11} & a_{12} & a_{13} & \cdots & a_{1n} \\ 0 & a_{22} & a_{23} & \cdots & a_{2n} \\ 0 & 0 & a_{33} & \cdots & a_{3n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & a_{nn} \end{pmatrix}$$

- **Lower Triangular Matrix:**

$$L = \begin{pmatrix} a_{11} & 0 & 0 & \cdots & 0 \\ a_{21} & a_{22} & 0 & \cdots & 0 \\ a_{31} & a_{32} & a_{33} & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & a_{n3} & \cdots & a_{nn} \end{pmatrix}$$

Matrix form of a linear system

A system of n linear equations with n variables can be represented as:

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

In matrix form, this system can be written as:

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

where:

- A is an $n \times n$ coefficient matrix:

$$A = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{pmatrix},$$

- $\mathbf{x}$ is the vector of unknowns:

$$\mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix},$$

- $\mathbf{b}$ is the vector of constants:

$$\mathbf{b} = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{pmatrix}.$$

The Elimination Method

The following three operations are called the elementary row operations:

1. **Row Scaling:** Multiplying any row R_i of a matrix by a non-zero number k. This operation is denoted by:

$$kR_i \to R_i$$

2. **Row Addition:** Adding a multiple of a row R_i to another row R_j. This operation is denoted by:

$$R_j + kR_i \to R_j$$

3. **Row Interchange:** Interchanging the order of two rows R_i and R_j. This operation is denoted by:

$$R_i \leftrightarrow R_j$$

Example 1: Let

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

We can apply the following elementary row operations to the matrix A.

1. Multiply the second row of A by 2 and obtain a new matrix B.
2. Add -2 times the first row to the third row and obtain a new matrix C.
3. Interchange the order of the second and third rows and obtain a new matrix D:

Solving linear systems of Equations

Numerical methods for the solution of systems of linear equations are of two types:

1. **Direct Methods**
   - Gaussian Elimination Method with Backward Substitution
   - LU Factorization
2. **Iterative Methods**
   - Jacobi Method
   - Gauss-Seidel Method

Direct Methods

Direct techniques are methods that gives the exact answers to the system in a finite number of steps.

Gaussian Elimination with back substitution

Gaussian elimination is a method to solve a linear system $A\mathbf{x} = \mathbf{b}$ by transforming the matrix A into an upper triangular matrix using elementary row operations on the augmented matrix $[A|\mathbf{b}]$.

Remark: Applying elementary row operations to the augmented matrix $[A|\mathbf{b}]$ does not change the solution of the linear system $A\mathbf{x} = \mathbf{b}$.

Let A be an upper triangular matrix. Back-substitution is the process of solving the linear system $A\mathbf{x} = \mathbf{b}$ by finding the unknown x_n from the last equation, then substituting this value into the $(n-1)$th equation to find the unknown x_{n-1}, and so on, until all unknowns are determined.

Example 1: Use Gauss Elimination and back-substitution to solve the linear system.

$$
\begin{aligned}
x_1 + x_2 + 3x_4 &= 4 \\
2x_1 + x_2 - x_3 + x_4 &= 1 \\
3x_1 - x_2 - x_3 + 2x_4 &= -3 \\
-x_1 + 2x_2 + 3x_3 - x_4 &= 4
\end{aligned}
$$

We can write the above linear system in the form $A\mathbf{x} = \mathbf{b}$ as follows:

$$
\begin{pmatrix} 1 & 1 & 0 & 3 \\ 2 & 1 & -1 & 1 \\ 3 & -1 & -1 & 2 \\ -1 & 2 & 3 & -1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} = \begin{pmatrix} 4 \\ 1 \\ -3 \\ 4 \end{pmatrix}
$$

Next, we consider the augmented matrix for the system $[A|\mathbf{b}]$:

$$[A|\mathbf{b}] = \left(\begin{array}{cccc|c} 1 & 1 & 0 & 3 & 4 \\ 2 & 1 & -1 & 1 & 1 \\ 3 & -1 & -1 & 2 & -3 \\ -1 & 2 & 3 & -1 & 4 \end{array}\right)$$

We will apply elementary row operations to transform the matrix into an upper triangular form.

Step 1: Row Operations

1. Subtract $2R_1$ from R_2:

$$R_2 - 2R_1 \to R_2 \implies R_2 = (2-2\cdot 1, 1-2\cdot 1, -1-2\cdot 0, 1-2\cdot 3|1-2\cdot 4)$$

Resulting in:

$$\left(\begin{array}{cccc|c} 1 & 1 & 0 & 3 & 4 \\ 0 & -1 & -1 & -5 & -7 \\ 3 & -1 & -1 & 2 & -3 \\ -1 & 2 & 3 & -1 & 4 \end{array}\right)$$

2. Subtract $3R_1$ from R_3:

$$R_3 - 3R_1 \to R_3 \implies R_3 = (3-3\cdot1, -1-3\cdot1, -1-3\cdot0, 2-3\cdot3 | -3-3\cdot4)$$

Resulting in:

$$\begin{pmatrix} 1 & 1 & 0 & 3 & | & 4 \\ 0 & -1 & -1 & -5 & | & -7 \\ 0 & -4 & -1 & -7 & | & -15 \\ -1 & 2 & 3 & -1 & | & 4 \end{pmatrix}$$

3. Add R_1 to R_4:

$$R_4 + R_1 \to R_4 \implies R_4 = (-1+1, 2+1, 3+0, -1+3 | 4+4)$$

Resulting in:

$$\begin{pmatrix} 1 & 1 & 0 & 3 & | & 4 \\ 0 & -1 & -1 & -5 & | & -7 \\ 0 & -4 & -1 & -7 & | & -15 \\ 0 & 3 & 3 & 2 & | & 8 \end{pmatrix}$$

4. Next, we simplify rows to achieve upper triangular form. From R_3 and R_4, we can continue applying row operations:

$$R_3 + 4R_2 \to R_3$$

Resulting in:

$$\left(\begin{array}{cccc|c} 1 & 1 & 0 & 3 & 4 \\ 0 & -1 & -1 & -5 & -7 \\ 0 & 0 & 3 & 13 & 13 \\ 0 & 0 & 0 & -13 & -13 \end{array}\right)$$

Step 2: Back-Substitution

Now that we have an upper triangular matrix, we can use back-substitution to solve for the variables.

1. From the last row:

$$-13x_4 = -13 \implies x_4 = 1$$

2. Substitute x_4 into the third row:

$$3x_3 + 13(1) = 13 \implies 3x_3 = 0 \implies x_3 = 0$$

3. Substitute x_3 and x_4 into the second row:

$$-x_2 - 0 - 5(1) = -7 \implies -x_2 - 5 = -7 \implies x_2 = 2$$

4. Finally, substitute x_2, x_3, and x_4 into the first row:

$$x_1 + 2 + 0 + 3(1) = 4 \implies x_1 + 2 + 3 = 4 \implies x_1 = -1$$

The solution to the system is:

$$\begin{aligned} x_1 &= -1 \\ x_2 &= 2 \\ x_3 &= 0 \\ x_4 &= 1 \end{aligned}$$

Example 2: Consider the linear system represented by the following equations:

$$
\begin{aligned}
4x_1 + 5x_2 + 6x_3 &= 7 \\
7x_1 + 8x_2 + 9x_3 &= 8 \\
1x_1 + 2x_2 + 3x_3 &= 3
\end{aligned}
$$

Use Gauss Elimination and back-substitution to solve the above linear system.

Factorization method

LU Factorization

In this section, we factorize the coefficient matrix A into an upper triangular matrix U and a lower triangular matrix L, such that:

$$A = LU$$

By the factorization $A = LU$, we can rewrite the system of equations $A\mathbf{x} = \mathbf{b}$ as:

$$LU\mathbf{x} = \mathbf{b}$$

We introduce a new vector $\mathbf{y}$, where:

$$U\mathbf{x} = \mathbf{y}$$

Thus, the system becomes:

$$L\mathbf{y} = \mathbf{b}$$

To solve for $\mathbf{x}$:

1. First, solve $L\mathbf{y} = \mathbf{b}$ using forward substitution to find $\mathbf{y}$.
2. Then, solve $U\mathbf{x} = \mathbf{y}$ using backward substitution to find $\mathbf{x}$.

Therefore, solving $A\mathbf{x} = \mathbf{b}$ is reduced to solving two simpler triangular systems, which are more computationally efficient.

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

Algorithm

Step 1: Decomposition

Decompose the matrix A into L and U.Generally, the procedure is as follows:

1st row entries in U,

$$U_{ij} = \frac{a_{ij}}{l_{11}} \text{ where } i = 1, 2, ., n$$

1st column entries in L,

$$\ell_{i1} = \frac{a_{i1}}{u_{11}} \text{ where } i = 1, 2, ., n$$

$$\ell_{ii} u_{ii} = a_{ii} - \sum_{k=1}^{i-1} \ell_{ik} u_{ki} \text{ where } i = 1, 2, ...n - 1$$

$$\text{If } i > j, \text{ then } \ell_{ji} = \frac{a_{ji} - \sum_{k=1}^{i-1} \ell_{jk} u_{ki}}{u_{ii}}$$

$$\text{If } j > i, \text{ then } u_{ij} = \frac{a_{ij} - \sum_{k=1}^{i-1} \ell_{ik} u_{kj}}{\ell_{ii}}$$

$$\ell_{nn} u_{nn} = a_{nn} - \sum_{k=1}^{n-1} \ell_{nk} u_{kn}$$

Step 2: Forward Substitution
Solve $L\mathbf{y} = \mathbf{b}$ for $\mathbf{y}$ using forward substitution. This involves solving the system from top to bottom:

$$y_1 = \frac{b_1}{\ell_{11}}$$

$$y_2 = \frac{b_2 - \ell_{21} y_1}{\ell_{22}}$$

$$y_i = \frac{b_i - \sum_{j=1}^{i-1} \ell_{ij} y_j}{\ell_{ii}}$$

Step 3: Backward Substitution
Solve $U\mathbf{x} = \mathbf{y}$ for $\mathbf{x}$ using backward substitution. This involves solving the system from bottom to top:

$$x_n = \frac{y_n}{u_{nn}}$$

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

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

- Suppose that $\ell_{ii} = 1$ for $i = 1, 2, 3, ..., n$ then this method is called **Doolittle's LU Factorization.**
- Suppose $u_{ii} = 1$ for $i = 1, 2, \ldots, n$. In this case, the method is called **Crout's LU Factorization.**

Example:
Given a system of linear equations:

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

where:

$$A = \begin{pmatrix} 2 & 3 & 1 \\ 4 & 7 & 1 \\ -2 & 4 & -3 \end{pmatrix}, \quad \mathbf{b} = \begin{pmatrix} 1 \\ 3 \\ -1 \end{pmatrix}$$

Using Doolittle's LU factorization method, decompose matrix A into L and U:

$$L = \begin{pmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ -1 & 7 & 1 \end{pmatrix}, \quad U = \begin{pmatrix} 2 & 3 & 1 \\ 0 & 1 & -1 \\ 0 & 0 & 5 \end{pmatrix}$$

Now, solve $L\mathbf{y} = \mathbf{b}$:

$$L \begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix} = \begin{pmatrix} 1 \\ 3 \\ -1 \end{pmatrix}$$

Using forward substitution, solve for $\mathbf{y}$:

$$y_1 = 1, \quad y_2 = 1, \quad y_3 = -7$$

Next, solve $U\mathbf{x} = \mathbf{y}$:

$$U \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ -7 \end{pmatrix}$$

Using backward substitution, solve for $\mathbf{x}$:

$$x_3 = -7/5, \quad x_2 = -2/5, \quad x_1 = 9/5$$

Thus, the solution is:

$$\mathbf{x} = \begin{pmatrix} -7/5 \\ -2/5 \\ 9/5 \end{pmatrix}$$

Importance of Factorization

- **Efficiency**:factorization reduces the computational effort when solving multiple systems with the same coefficient matrix.
- **Versatility**: It can be used for both square and rectangular matrices, making it widely applicable in numerical solutions of linear systems.
- **Reusability**: Once the matrix A is factored into L and U, the factorization can be reused to solve multiple systems with different right-hand side vectors $\mathbf{b}$.

Importance of Direct Methods

1. **Exact Solutions:** Direct methods provide exact solutions in a finite number of steps.
2. **Wide Applicability:** These methods can be applied to a broad range of linear systems, including those arising from scientific computing, engineering simulations, and optimization problems.
3. **Computational Efficiency:** For small to moderate-sized systems, direct methods often offer better computational efficiency compared to iterative methods.
4. **Foundational Techniques:** Direct methods form the basis for many numerical algorithms and are often used to preprocess data for more complex numerical analysis techniques.
5. **Numerical Stability:** Well-implemented direct methods can exhibit favorable numerical stability properties.

Iterative Methods

Iterative methods provide an approximate solution to the system. An iterative technique to solve the system $Ax = b$ starts with an initial approximation $x^{(0)}$ and generates a sequence of vectors $\{x^{(k)}\}_{k=0}^{\infty}$ that converges to x. Iterative techniques reformulate the system $Ax = b$ into an equivalent system of the form:

$$x = Tx + c$$

for some fixed matrix T and vector c. After the initial vector $x^{(0)}$ is selected, the sequence of approximate solution vectors is generated by computing:

$$x^{(k)} = Tx^{(k-1)} + c \quad \text{for} \quad k = 1, 2, 3, \ldots$$

We will cover two iterative methods:

1. Jacobi Method
2. Gauss-Seidel Method

Convergence Criteria

For the iterative methods to converge:

- The matrix A should be diagonally dominant for the Jacobi and Gauss-Seidel methods.

Diagonal Dominance

A matrix is said to be diagonally dominant if, for every row of the matrix, the magnitude of the diagonal entry in a row is larger than or equal to the sum of the magnitudes of all the off-diagonal elements. More precisely, for all i:

$$|a_{ii}| > \sum_{j \neq i} |a_{ij}|$$

Example:

$$\begin{aligned} x_1 + 3x_2 + 5x_3 &= 8 \\ 3x_1 + x_2 + x_3 &= 5 \\ x_1 + 8x_2 - 5x_3 &= 3 \end{aligned}$$

If a system of equations is rearranged to achieve diagonal dominance, the matrix can then be solved iteratively. Now, a_{ii} is the coefficient with the largest magnitude in the i-th equation.

Jacobi Method

The Jacobi method solves each equation for a particular variable and uses these values to approximate the solution.

Algorithm:

s0.5cm

1. Rearrange the system $A\mathbf{x} = \mathbf{b}$ so that the diagonal elements of A are isolated.
2. Start with an initial guess $\mathbf{x}^{(0)}$.
3. Update each $x_i^{(k)}$ according to:

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

   for each $i = 1, 2, \ldots, n$.

4. Repeat until convergence.

Example 1:

$$
\begin{aligned}
E_1 &: \quad 10x_1 - x_2 + 2x_3 = 6 \\
E_2 &: \quad -x_1 + 11x_2 - x_3 + 3x_4 = 25 \\
E_3 &: \quad 2x_1 - x_2 + 10x_3 - x_4 = -11 \\
E_4 &: \quad 3x_2 - x_3 + 8x_4 = 15
\end{aligned}
$$

This system is already in diagonal dominance form. To convert $Ax = b$ to the form $x = Tx + c$, solve each equation E_i for x_i for $i = 1, 2, 3, 4$.

The first 10 iterations of the Jacobi method are given in the table below:

k	$x_1^{(k)}$	$x_2^{(k)}$	$x_3^{(k)}$	$x_4^{(k)}$
0	0.0000	0.0000	0.0000	0.0000
1	0.6000	2.2727	−1.1000	1.8750
2	1.0473	1.7159	−0.8052	0.8852
3	0.9326	2.0533	−1.0494	1.1309
4	1.0152	1.9537	−0.9681	0.9793
5	0.9890	2.0114	−1.0103	1.0214
6	1.0032	1.9922	−0.9945	0.9944
7	0.9981	2.0023	−1.0020	1.0036
8	1.0006	1.9987	−0.9940	0.9989
9	0.9997	2.0004	−1.0044	1.0006
10	1.0001	1.9998	−0.9998	0.9998

By considering the last two iterations ($x^{(9)}$ and $x^{(10)}$), we can see that convergence has occurred. Therefore, the required solution is:

$$x_1 = 1, \quad x_2 = 2, \quad x_3 = -1, \quad x_4 = 1$$

Matrix Form of Jacobi Method

The Jacobi method can also be written in matrix form. We split the matrix A into its diagonal, lower, and upper triangular parts:

$$A = D - L - U$$

where D is the diagonal matrix, L is the strictly lower triangular part, and U is the strictly upper triangular part. If D^{-1} exist that $a_{ii} \neq 0$ for each i, this leads to the iterative form:

$$x^{(k)} = D^{-1}(L + U)x^{(k-1)} + D^{-1}b$$

In practice, this equation is used for computation, while the theoretical form is:

$$x^{(k)} = Tx^{(k-1)} + c$$

where $T = D^{-1}(L + U)$ and $c = D^{-1}b$.

Note
The **infinity norm** (also called the *maximum norm* or *supremum norm*) is a vector norm defined as:

$$\|x\|_\infty = \max_{1 \leq i \leq n} |x_i|$$

For a vector $x = (x_1, x_2, \ldots, x_n) \in \mathbb{R}^n$, the infinity norm is given by:

$$\|x\|_\infty = \max\left(|x_1|, |x_2|, \ldots, |x_n|\right)$$

Example: For a vector $x = (3, -7, 5)$, the infinity norm is:

$$\|x\|_\infty = \max\left(|3|, |-7|, |5|\right) = 7.$$

The Error of the n^{th} computation is defined as

$$e_n = \frac{\|x^{(n)} - x^{(n-1)}\|_\infty}{\|x^{(n)}\|_\infty}$$

in infinity norm(l_∞).

Example 2: Consider the following system of equations:

$$\begin{aligned}
E_1: &\quad 10x_1 - x_2 + 2x_3 = 6,\\
E_2: &\quad -x_1 + 11x_2 - x_3 + 3x_4 = 25,\\
E_3: &\quad 2x_1 - x_2 + 10x_3 - x_4 = -11,\\
E_4: &\quad 3x_2 - x_3 + 8x_4 = 15.
\end{aligned}$$

Use the Jacobi method to solve this system, starting with an initial guess:

$$x^{(0)} = (0, 0, 0, 0).$$

Continue iterating until:

$$\left\| x^{(k)} - x^{(k-1)} \right\|_\infty < 10^{-3}.$$

Example 3: Use Jacobi method to solve the following linear system with Tolerance=TOL=10^{-3} in the l_∞ norm, that is

$$e_n = \frac{\|x^{(n)} - x^{(n-1)}\|_\infty}{\|x^{(n)}\|_\infty} < 10^{-3}$$

$$\begin{aligned} E_1: &\quad 10x_1 + 2x_2 + x_3 = 9, \\ E_2: &\quad x_1 + 10x_2 - x_3 = -22, \\ E_3: &\quad -2x_1 + 3x_2 + 10x_3 = 22. \end{aligned}$$

We initial guess:

$$x^{(0)} = (0, 0, 0).$$

The iterative equations can be derived as follows:

$$x_1 = \frac{9 - 2x_2 - x_3}{10},$$
$$x_2 = \frac{-22 - x_1 + x_3}{10},$$
$$x_3 = \frac{22 + 2x_1 - 3x_2}{10}.$$

Continue iterating until:

$$\frac{\|x^{(n)} - x^{(n-1)}\|_\infty}{\|x^{(n)}\|_\infty} < 10^{-3}$$

Now, we proceed with the iterations:

Iteration	$x_1^{(k)}$	$x_2^{(k)}$	$x_3^{(k)}$	$\|x^{(k)} - x^{(k-1)}\|_\infty$	Error(e_k)
0	0.0000	0.0000	0.0000	–	–
1	0.9000	−2.2000	2.2000	2.2000	1.0000
2	1.1200	−2.0700	3.0400	0.8400	0.2763
3	1.0100	−2.0080	3.04500	0.1100	0.0361
4	0.9971	−1.9965	3.0044	0.0406	0.0135
5	0.9989	−1.9993	2.9984	0.0060	0.0020
6	1.0000	−2.0001	2.9996	0.0012	0.0004

After the 6th iteration, we can see that:

$$\text{Convergence occurs: } \frac{\|x^{(6)} - x^{(5)}\|_\infty}{\|x^{(6)}\|_\infty} < 10^{-3}$$

Thus, the approximate solution is:

$$x_1 \approx 1, \quad x_2 \approx -2, \quad x_3 \approx 3.$$

The Jacobi iterative method is a simple and effective technique for approximating the solution to a system of linear equations, especially when the matrix A is diagonally dominant. The method converges when the successive approximations are close enough, achieving the required accuracy.

Gauss-Seidel Method

The Gauss-Seidel method improves upon the Jacobi method by using the most recently computed values in the iterative process. In Jacobi's method, all components of $x^{(k)}$ are calculated using the values from $x^{(k-1)}$. In contrast, the Gauss-Seidel method updates each component as soon as its value is computed, leading to potentially faster convergence.

Algorithm:

1. Start with an initial guess $\mathbf{x}^{(0)}$.
2. Update each $x_i^{(k)}$ using:

$$x_i^{(k)} = \frac{1}{a_{ii}} \left(b_i - \sum_{j<i} a_{ij} x_j^{(k)} - \sum_{j>i} a_{ij} x_j^{(k-1)} \right)$$

3. Repeat until convergence.

Example 3:
Solve the system:

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

Start with:

$$x^{(0)} = (0, 0, 0, 0).$$

Continue iterating until:

$$\frac{\|x^{(k)} - x^{(k-1)}\|_\infty}{\|x^{(k)}\|_\infty} < 10^{-3}.$$

k	$x_1^{(k)}$	$x_2^{(k)}$	$x_3^{(k)}$	$x_4^{(k)}$
0	0.0000	0.0000	0.0000	0.0000
1	0.6000	2.3227	-0.9873	0.8789
2	1.0302	2.0369	-1.0144	0.9844
3	1.0066	2.0035	-1.0025	0.9984
4	1.0009	2.0003	-1.0003	0.9999
5	1.0001	2.0000	-1.0000	1.0000

After several iterations, check:

$$\frac{\|x^{(k)} - x^{(k-1)}\|_\infty}{\|x^{(k)}\|_\infty} < 10^{-3}.$$

Once this condition is met, $x^{(5)}$ is accepted as a reasonable approximation of the solution.

Matrix Form

The Gauss-Seidel method can also be represented in matrix form as:

$$(D - L)x^{(k)} = Ux^{(k-1)} + b$$

If $(D - L)^{-1}$ exists, then that

$$x^{(k)} = (D - L)^{-1} Ux^{(k-1)} + (D - L)^{-1} b$$

where D, L, and U are the diagonal, lower triangular, and upper triangular parts of the coefficient matrix, respectively. For the

lower triangular matrix $(D - L)$ to be non-singular, it is necessary and sufficient that $a_{ii} \neq 0$, for each $i = 1, 2, 3, ..., n$.

Comparison Between Iterative Methods

- **Convergence Speed:** The Gauss-Seidel method converges more quickly than the Jacobi method due to its use of updated values immediately in calculations.
- **Memory Efficiency:** Iterative methods are typically more memory-efficient than direct methods, making them suitable for large-scale problems.
- **Error Correction:** Iterative methods benefit from inherent self-correcting properties, allowing for adjustments to errors made in previous iterations.

Comparison Between Direct and Iterative Methods

- **Efficiency for Large Systems:** For larger systems of equations, iterative methods are often faster than direct methods.
- **Direct Methods for Specific Structures:** Direct methods are particularly effective for solving tri-diagonal systems.
- **Round-off Errors:** Direct methods are prone to round-off errors.
- **Self-Correcting Nature:** Iterative methods correct errors automatically in subsequent iterations.
- **Sparse Coefficient Matrices:** When the coefficient matrix contains a significant number of zeros, iterative methods can leverage this sparsity for faster computations compared to direct methods.
- **Scalability:** Iterative methods can be more scalable for very large problems