Department of
Computer Science and Engineering

University of Dhaka

Title: Solving Systems of Linear Equations Using Iterative Methods

CSE 3212: Numerical Analysis Lab

Batch: 28/3rd Year 2nd Semester 2024

1 Objective(s)

- Implement Jacobi and Gauss-Seidel iterative algorithms for solving linear systems
- Analyze convergence behavior and computational efficiency
- Develop robust numerical software with proper error handling
- Compare theoretical predictions with practical implementations

2 Problem Description

Implement a comprehensive program that solves systems of linear equations using both Jacobi and Gauss-Seidel iterative methods. The program should accept user input for all necessary parameters, implement both algorithms, and provide detailed comparative analysis of their performance.

2.1 Program Specifications

The program shall accept the following inputs from the user:

- **Number of equations/variables** (n):
$$n \in \mathbb{Z}^+, \quad n \geq 2$$
- **Coefficient matrix A**:
$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix} \in \mathbb{R}^{n \times n}$$
- **Constants vector b**:
$$\mathbf{b} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix} \in \mathbb{R}^{n \times 1}$$
- **Initial guess vector $\mathbf{x}^{(0)}$**:
$$\mathbf{x}^{(0)} = \begin{bmatrix} x_1^{(0)} \\ x_2^{(0)} \\ \vdots \\ x_n^{(0)} \end{bmatrix} \in \mathbb{R}^{n \times 1}$$
- **Maximum number of iterations** ($K_{\max}$):
$$K_{\max} \in \mathbb{Z}^+, \quad K_{\max} > 0$$
- **Tolerance for convergence** (ϵ):
$$\epsilon \in \mathbb{R}^+, \quad \epsilon > 0$$

2.2 Algorithm Implementation

The program shall implement both iterative methods according to their mathematical formulations:

Jacobi Method:

$$x_i^{(k+1)} = \frac{1}{a_{ii}} \left(b_i - \sum_{j=1, j\neq i}^{n} a_{ij} x_j^{(k)} \right), \quad i = 1, 2, \ldots, n \tag{1}$$

Gauss-Seidel Method:

$$x_i^{(k+1)} = \frac{1}{a_{ii}} \left(b_i - \sum_{j=1}^{i-1} a_{ij} x_j^{(k+1)} - \sum_{j=i+1}^{n} a_{ij} x_j^{(k)} \right), \quad i = 1, 2, \ldots, n \tag{2}$$

2.3 Complete Input Example

User Input Sequence

The following demonstrates the complete input sequence for a 3×3 linear system:

```
Enter number of equations: 3

Enter coefficient matrix A (row-wise):
4  -1  0
-1  4  -1
0  -1  4

Enter constants vector b:
12
-1
5

Enter initial guess vector:
0
0
0

Enter maximum iterations: 50
Enter tolerance: 0.0001
```

Listing 1: Complete User Input Sequence

Mathematical Representation

The input corresponds to the following linear system:

Coefficient Matrix

$$\mathbf{A} = \begin{bmatrix} 4 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 4 \end{bmatrix}$$

Constants Vector

$$\mathbf{b} = \begin{bmatrix} 12 \\ -1 \\ 5 \end{bmatrix}$$

Initial Guess Vector

$$\mathbf{x}^{(0)} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

Complete System

The system of linear equations is:

$$
\begin{aligned}
4x_1 - x_2 + 0x_3 &= 12 \\
-x_1 + 4x_2 - x_3 &= -1 \\
0x_1 - x_2 + 4x_3 &= 5
\end{aligned}
$$

Iteration Parameters

- **Maximum iterations**: 50
- **Convergence tolerance**: 0.0001

2.4 Output Requirements

The program shall display the following results for both methods:

1. **Solution vector $\mathbf{x}^*$**:

   $$\mathbf{x}^* = \begin{bmatrix} x_1^* \\ x_2^* \\ \vdots \\ x_n^* \end{bmatrix}$$

   displayed with appropriate precision (at least 6 decimal places).

2. **Number of iterations** (k) required to reach convergence:

   $$k \leq K_{\max}$$

3. **Error metrics after each iteration**:
   - Absolute error of each variable x_i
   - Residual Error: $\|\mathbf{A}\mathbf{x}^{(k)} - \mathbf{b}\|_2$

4. **Convergence behavior comparison**:
   - Iteration count comparison
   - Convergence rate analysis
   - Computational efficiency remarks

2.5 Error Handling

The program must handle the following exceptional cases:

- **Non-convergence**: Display appropriate message when maximum iterations are reached without convergence
- **Division by zero**: Check for zero diagonal elements before iteration
- **Invalid input**: Provide clear error messages for malformed input
- **Memory limits**: Handle large systems within computational constraints

2.6 Testing Requirements

Test the implementation with the following cases:

Test Case 1:

$$\mathbf{A} = \begin{bmatrix} 4 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 4 \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} 12 \\ -1 \\ 5 \end{bmatrix}, \quad \mathbf{x}^{(0)} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

$$\text{Expected: } \mathbf{x}^* \approx \begin{bmatrix} 3.000 \\ 1.000 \\ 1.500 \end{bmatrix}$$

Test Case 2: Larger System

$$\mathbf{A} = \begin{bmatrix} 10 & -1 & 2 & 0 \\ -1 & 11 & -1 & 3 \\ 2 & -1 & 10 & -1 \\ 0 & 3 & -1 & 8 \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} 6 \\ 25 \\ -11 \\ 15 \end{bmatrix}$$

$$\text{Expected: } \mathbf{x}^* \approx \begin{bmatrix} 1.000 \\ 2.000 \\ -1.000 \\ 1.000 \end{bmatrix}$$

2.7 Grading Criteria

Table 1: Assessment Rubric

Criteria	**Description**	**Points**
Input/Output Module	Correct handling of all user inputs and formatted output	15
Jacobi Implementation	Accurate implementation of Jacobi iterative method with proper error calculation	30
Gauss-Seidel Implementation	Accurate implementation of Gauss-Seidel method with proper error calculation	30
Comparative Results	Meaningful comparison between both methods	10
Error Handling	Robust handling of edge cases and invalid inputs	15
Total		**100**