AE227: Numerical fluid flow Assignment 3

Manish Sharma

April 16, 2025

Problem: Solving Steady State 1D Poisson's problem using Jacobi, GS and CG Iterative Methods

Consider the 1D Poisson's equation in the domain $\Omega = [0,\pi]$

$$-\frac{\partial^2 u}{\partial x^2} = f \quad \text{in} \quad \Omega \tag{1}$$

with the boundary conditions

$$u = g \quad \text{on } \partial\Omega \tag{2}$$

where f and g are given functions, and $\partial\Omega$ represents the boundary of Ω. Consider the case where f = sin(x), and g is given as,

$$g(x) = \begin{cases} 0 & \text{if } x = 0, \\ 0 & \text{if } x = \pi, \end{cases} \tag{3}$$

Consider four uniform meshes with the number of elements (Ni): $N_1 = 5$, $N_2 = 10$, $N_3 = 20$ and $N_4 = 40$. Discretize the weak (variational) form of Eq. 1 using continuous Galerkin FEM using linear(nodal) basis functions associated with the mesh points (as explained in class). Let u_h be the finite element solution. The discretizations for the four meshes result in four systems of linear equations (say A $\tilde{u}$ = b), where $\tilde{u}$ is the vector of coefficients for the solution uh. Solve the resulting linear systems for N_1, N_2, N_3, and N_4 using Jacobi, GS, and CG iterative methods. Take the initial guess as $\tilde{u}(0) = 1$ for all three iterative methods. The iterations should be continued until the relative change in the solution of the system of linear equations from one iteration to another is less than 10^{-10}. More precisely, stop the iterations when

$$\frac{||\tilde{u}^{k+1} - \tilde{u}^k||_2}{||\tilde{u}^k||_2} \leq 10^{-10} \tag{4}$$

Submit a short report, which has the following:

1 Start from discretizing the PDE(Eq. 1) using continuous Galerkin FEM using linear basis function, and derive the matrix A and rhs vector b for all mesh sizes (N_i) i.e. write the structure of A and b

The 1D Poisson's equation is given by:

$$-\frac{d^2u}{dx^2} = f \quad \text{in} \quad \Omega = [0, \pi], \tag{5}$$

with Dirichlet boundary conditions:

$$u = g \quad \text{on} \quad \partial\Omega, \tag{6}$$

where $f(x) = \sin(x)$, and the boundary conditions are:

$$g(x) = \begin{cases} 0, & \text{if } x = 0, \\ 0, & \text{if } x = \pi. \end{cases} \tag{7}$$

We discretize this problem using the continuous Galerkin Finite Element Method (FEM) with linear basis functions on uniform meshes with $N_i = 5, 10, 20, 40$ elements. The goal is to derive the structure of the stiffness matrix $\mathbf{A}$ and the right-hand side vector $\mathbf{b}$ for the resulting linear systems $\mathbf{A}\bar{u} = \mathbf{b}$.

1.1 Weak Form

To apply the FEM, we first derive the weak form of the Poisson's equation. Multiply both sides of Equation (1) by a test function $v \in H_0^1(\Omega)$ (where $H_0^1(\Omega)$ is the space of functions with square-integrable first derivatives and zero boundary values) and integrate over the domain:

$$-\int_0^\pi \frac{d^2u}{dx^2} v \, dx = \int_0^\pi f v \, dx. \tag{8}$$

Integrate the left-hand side by parts:

$$-\int_0^\pi \frac{d^2u}{dx^2} v \, dx = -\left[\frac{du}{dx} v\right]_0^\pi + \int_0^\pi \frac{du}{dx}\frac{dv}{dx} \, dx. \tag{9}$$

Since $v \in H_0^1(\Omega)$, we have $v(0) = v(\pi) = 0$, so the boundary term vanishes:

$$-\left[\frac{du}{dx} v\right]_0^\pi = 0. \tag{10}$$

Thus, the weak form is:

$$\int_0^\pi \frac{du}{dx}\frac{dv}{dx} \, dx = \int_0^\pi f v \, dx \quad \forall v \in H_0^1(\Omega). \tag{11}$$

1.2 Finite Element Discretization

We discretize the domain $\Omega = [0, \pi]$ into N elements, creating a uniform mesh with nodes $x_i = ih$, where $h = \pi/N$ and $i = 0, 1, \ldots, N$. The number of nodes is $N + 1$. We approximate the solution $u(x)$ in the finite element space $V_h \subset H^1(\Omega)$, defined as:

$$u_h(x) = \sum_{j=0}^{N} u_j \phi_j(x), \tag{12}$$

where $\phi_j(x)$ are piecewise linear basis functions, and u_j are the coefficients. The basis functions satisfy:

$$\phi_j(x_i) = \delta_{ij} = \begin{cases} 1, & \text{if } i = j, \\ 0, & \text{otherwise.} \end{cases} \tag{13}$$

For an element $[x_i, x_{i+1}]$, the local basis functions are:

$$\phi_i(x) = \frac{x_{i+1} - x}{h}, \quad \phi_{i+1}(x) = \frac{x - x_i}{h}, \quad x \in [x_i, x_{i+1}]. \tag{14}$$

Their derivatives are:

$$\frac{d\phi_i}{dx} = -\frac{1}{h}, \quad \frac{d\phi_{i+1}}{dx} = \frac{1}{h}. \tag{15}$$

To enforce the boundary conditions $u_h(0) = u_h(\pi) = 0$, we set $u_0 = u_N = 0$, reducing the unknowns to $u_1, u_2, \ldots, u_{N-1}$.

The discrete weak form is obtained by substituting u_h and test functions $v_h = \phi_i$ (for $i = 1, \ldots, N-1$) into Equation (11):

$$\int_0^\pi \frac{du_h}{dx} \frac{d\phi_i}{dx}\, dx = \int_0^\pi f\phi_i\, dx. \tag{16}$$

Substituting $u_h = \sum_{j=0}^N u_j \phi_j$, the left-hand side becomes:

$$\int_0^\pi \frac{d}{dx}\left(\sum_{j=0}^N u_j\phi_j\right) \frac{d\phi_i}{dx}\, dx = \sum_{j=0}^N u_j \int_0^\pi \frac{d\phi_j}{dx}\frac{d\phi_i}{dx}\, dx. \tag{17}$$

This leads to the linear system:

$$\mathbf{A}\bar{u} = \mathbf{b}, \tag{18}$$

where:

- $\mathbf{A}$ is the stiffness matrix with entries:

$$A_{ij} = \int_0^\pi \phi_j'\phi_i'\, dx, \tag{19}$$

- $\mathbf{b}$ is the load vector with entries:

$$b_i = \int_0^\pi f\phi_i\, dx, \tag{20}$$

- $\bar{u} = [u_1, u_2, \ldots, u_{N-1}]^T$ (after enforcing boundary conditions).

1.3 Computation of A

The stiffness matrix $\mathbf{A}$ is assembled element-wise. For an element $[x_k, x_{k+1}]$, the local stiffness matrix is:

$$K_e = \int_{x_k}^{x_{k+1}} \begin{bmatrix} \phi_k'\phi_k' & \phi_k'\phi_{k+1}' \\ \phi_{k+1}'\phi_k' & \phi_{k+1}'\phi_{k+1}' \end{bmatrix} dx. \tag{21}$$

Using the derivatives from Equation (15):

$$\phi_k' = -\frac{1}{h}, \quad \phi_{k+1}' = \frac{1}{h}, \tag{22}$$

we compute:

- $\int_{x_k}^{x_{k+1}} \frac{d\phi_k}{dx}\frac{d\phi_k}{dx}\, dx = \int_{x_k}^{x_{k+1}} \left(-\frac{1}{h}\right)\left(-\frac{1}{h}\right)\, dx = \int_{x_k}^{x_{k+1}} \frac{1}{h^2}\, dx = \frac{1}{h}$,
- $\int_{x_k}^{x_{k+1}} \frac{d\phi_k}{dx}\frac{d\phi_{k+1}}{dx}\, dx = \int_{x_k}^{x_{k+1}} \left(-\frac{1}{h}\right)\left(\frac{1}{h}\right)\, dx = -\frac{1}{h}$,
- $\int_{x_k}^{x_{k+1}} \frac{d\phi_{k+1}}{dx}\frac{d\phi_k}{dx}\, dx = -\frac{1}{h}$,
- $\int_{x_k}^{x_{k+1}} \frac{d\phi_{k+1}}{dx}\frac{d\phi_{k+1}}{dx}\, dx = \frac{1}{h}$.

Thus, the local stiffness matrix is:

$$K_e = \frac{1}{h}\begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}. \tag{23}$$

The global stiffness matrix $\mathbf{A}$ is assembled by summing contributions from all elements. For interior nodes $i = 1, \ldots, N-1$:

- Diagonal: $A_{ii} = \frac{1}{h} + \frac{1}{h} = \frac{2}{h}$ (contributions from elements $[x_{i-1}, x_i]$ and $[x_i, x_{i+1}]$),
- Off-diagonal: $A_{i,i+1} = A_{i+1,i} = -\frac{1}{h}$ (from element $[x_i, x_{i+1}]$),
- All other entries are zero.

For a reduced system (excluding boundary nodes), $\mathbf{A}$ is a tridiagonal matrix of size $(N-1)\times(N-1)$:

$$\mathbf{A} = \frac{1}{h}\begin{bmatrix} 2 & -1 & 0 & \cdots & 0 \\ -1 & 2 & -1 & \ddots & \vdots \\ 0 & -1 & 2 & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & -1 \\ 0 & \cdots & 0 & -1 & 2 \end{bmatrix}. \tag{24}$$

1.4 Computation of b

The load vector $\mathbf{b}$ is computed as:

$$b_i = \int_0^{\pi} f(x)\phi_i(x)\,dx = \int_{x_{i-1}}^{x_{i+1}} \sin(x)\phi_i(x)\,dx, \tag{25}$$

since $\phi_i(x)$ is non-zero only in elements $[x_{i-1}, x_i]$ and $[x_i, x_{i+1}]$. For element $[x_{i-1}, x_i]$, where $\phi_i(x) = \frac{x - x_{i-1}}{h}$:

$$b_i^{(1)} = \int_{x_{i-1}}^{x_i} \sin(x)\frac{x - x_{i-1}}{h}\,dx. \tag{26}$$

For element $[x_i, x_{i+1}]$, where $\phi_i(x) = \frac{x_{i+1} - x}{h}$:

$$b_i^{(2)} = \int_{x_i}^{x_{i+1}} \sin(x)\frac{x_{i+1} - x}{h}\,dx. \tag{27}$$

Thus:

$$b_i = b_i^{(1)} + b_i^{(2)}. \tag{28}$$

These integrals are typically evaluated numerically (e.g., using quadrature) due to the presence of $\sin(x)$. The resulting $\mathbf{b}$ is:

$$\mathbf{b} = [b_1, b_2, \ldots, b_{N-1}]^T, \tag{29}$$

where each b_i depends on the mesh size $h = \pi/N_i$.

1.5 Structure of A

For each mesh size:

- $N_i = 5$: $h = \pi/5$, $\mathbf{A}$ is 4×4:

$$\mathbf{A} = \frac{5}{\pi}\begin{bmatrix} 2 & -1 & 0 & 0 \\ -1 & 2 & -1 & 0 \\ 0 & -1 & 2 & -1 \\ 0 & 0 & -1 & 2 \end{bmatrix}.$$

- $N_i = 10$: $h = \pi/10$, $\mathbf{A}$ is 9×9, with the same tridiagonal structure.
- $N_i = 20$: $h = \pi/20$, $\mathbf{A}$ is 19×19.
- $N_i = 40$: $h = \pi/40$, $\mathbf{A}$ is 39×39.

1.6 Structure of b

For each mesh size:

- $N_i = 5$: **b** is 4×1, with entries b_i depending on $\sin(x)$ evaluated at nodes $x_i = i\pi/5$.
- $N_i = 10$: **b** is 9×1.
- $N_i = 20$: **b** is 19×1.
- $N_i = 40$: **b** is 39×1.

Each b_i is a linear combination of integrals over two adjacent elements, scaled by the mesh size h.

2 For a given method (e.g. Jacobi), plot the relative error (LHS of Eq. 4) versus the iteration index (k) for all three mesh sizes (h1,h2, and h3). Repeat the same for all four methods (a total of four figures). In the plot, the relative error (y-axis) should be in base-10 logarithmic scale, and the iteration count (x-axis) must be on a linear scale. Write the inferences from the plots based on your understanding.

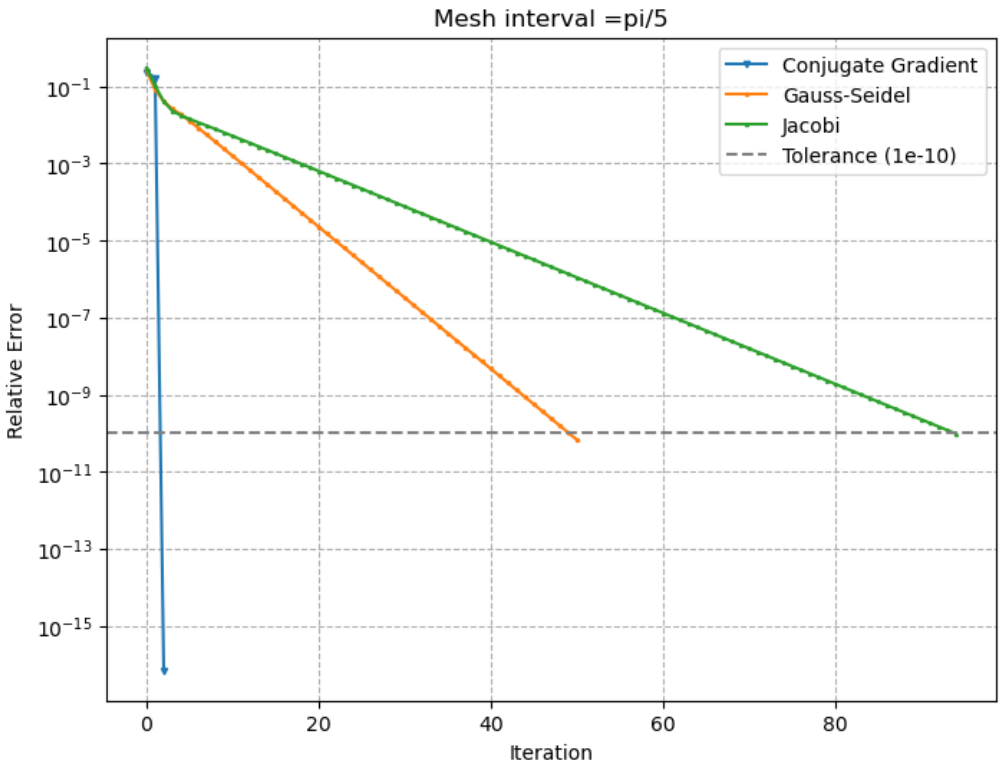

Figure 1: Jacobi Method

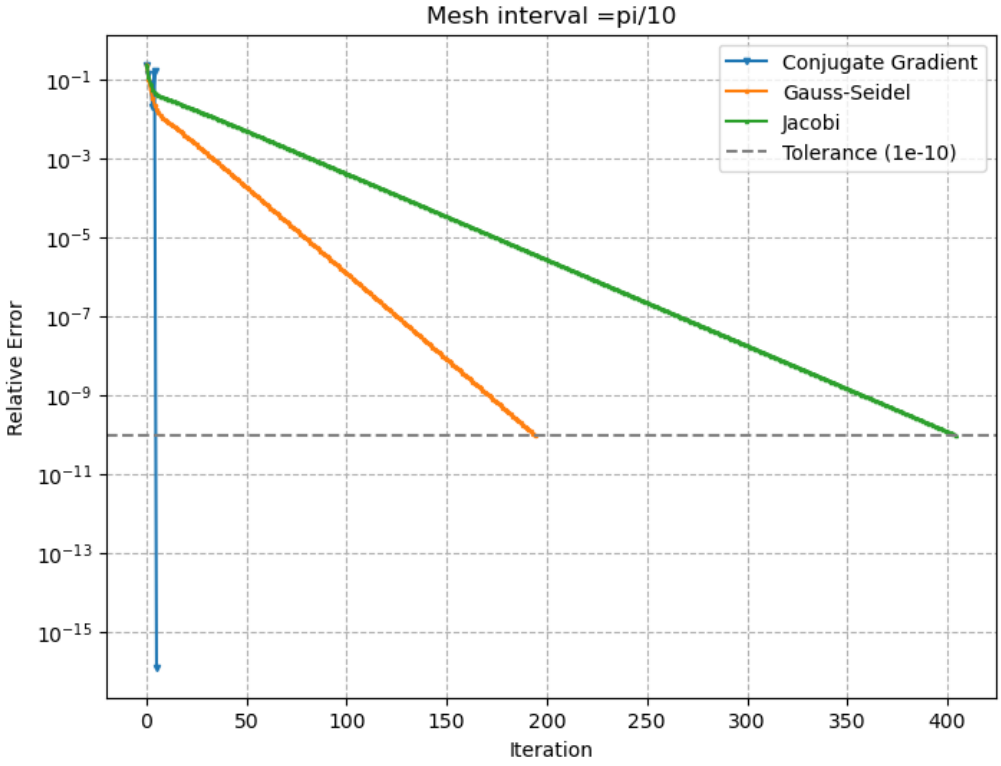

Figure 2: Gauss seidel method

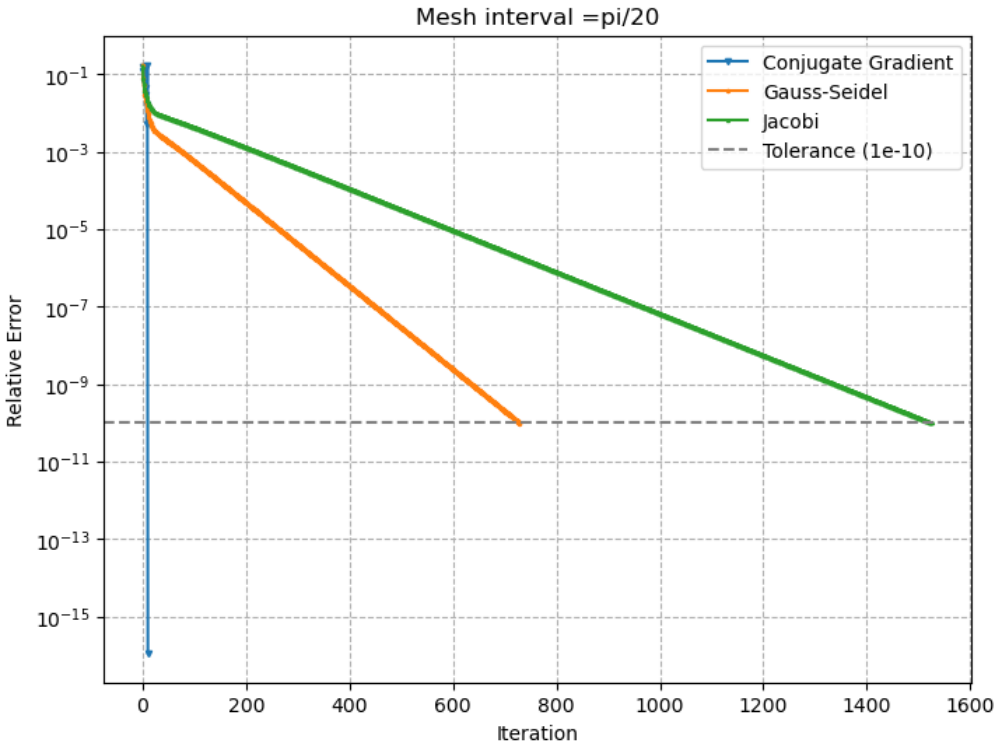

Figure 3: Steepest Descent Method

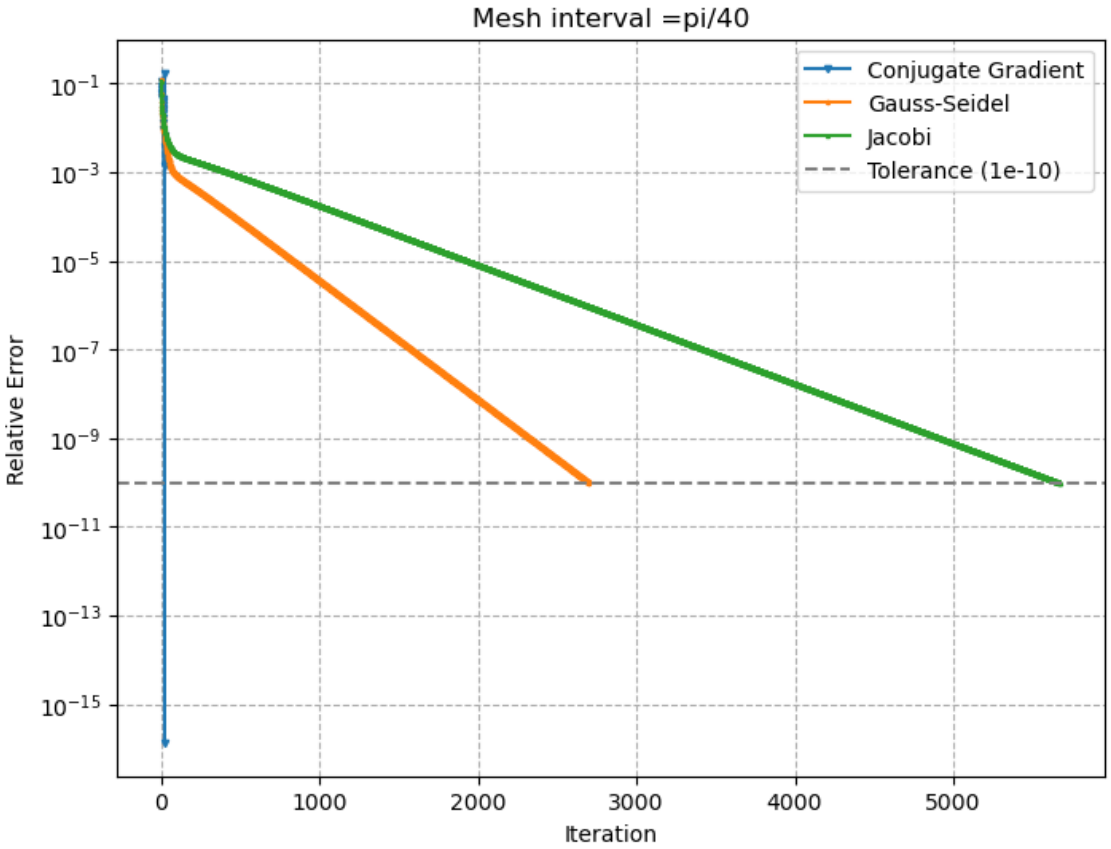

Figure 4: Conjugate Gradient Method

Inferences
The convergence behavior of the iterative methods is analyzed based on the plots:

1. **Conjugate Gradient** : Across all mesh sizes, the CG method converges significantly faster than Jacobi and Gauss-Seidel. The relative error decreases sharply, reaching the tolerance of 10^{-10} in far fewer iterations.

2. **Gauss-Seidel vs. Jacobi**: Gauss-Seidel generally converges faster than Jacobi for all N_i. The relative error for Gauss-Seidel decreases more rapidly, often requiring roughly half the iterations of Jacobi to reach the same error level. This is because Gauss-Seidel uses updated solution components immediately, improving convergence for the tridiagonal, diagonally dominant stiffness matrix.

3. **Effect of Mesh Size**: As N_i increases (i.e., $h = \pi/N_i$ decreases), the number of iterations required for convergence increases for all methods, particularly for Jacobi and Gauss-Seidel. This is due to the condition number of the stiffness matrix, which scales as $\mathcal{O}(h^{-2}) = \mathcal{O}(N_i^2)$. A larger condition number slows the convergence of stationary methods like Jacobi and Gauss-Seidel.

All methods reach the tolerance of 10^{-10}. However, CG achieves this in the fewest iterations, followed by Gauss-Seidel, then Jacobi. For finer meshes ($N = 20, 40$), the disparity in iteration counts becomes more pronounced, highlighting CG's efficiency for larger systems.

These observations confirm the advantages of CG for solving the Poisson equation's FEM system and illustrate the impact of mesh refinement.

3 In a table, report the number of iterations required to converge for Jacobi,GS, and CG methods for all four mesh sizes.

All the iteration counts are from (i+1) i.e 1,2,3.....n iteartions.

Mesh interval[h]	$\pi/5$	$\pi/10$	$\pi/20$	$\pi/40$
Jacobi	95	405	1526	5672
Gauss Seidel	51	195	730	2697
Conjugate Gradient	3	6	11	21

Table 1: Numnber of Iterations required coresponding to Jacobi,GS and CG methods

Convergence can be influenced by several factors

- Diagonal Dominance - Methods like Jacobi and Gauss-Seidel converge faster if the matrix **A** is diagonally dominant.
- Symmetry and Positive Definiteness - For methods like the Conjugate Gradient, **A** being symmetric and positive definite ensures faster convergence.
- Initial Guess: The choice of the initial guess $\mathbf{u}^{(0)}$ can influence the number of iterations needed for convergence. A guess closer to the actual solution can result in faster convergence.
- Condition Number - The condition number of the matrix **A** affects convergence. A lower condition number generally leads to faster convergence.
- Sparsity - Sparse matrices, where most elements are zero, can improve the efficiency and sometimes the convergence rate of iterative methods.

1. **Convergence of Jacobi:** Jacobi Method is the slowest method to solve a given linear system.
   - The Jacobi method converges faster if the matrix **A** is diagonally dominant. This means that for each row i,

     $$|a_{ii}| > \sum_{\substack{j=1 \\ j \neq i}}^{n} |a_{ij}|,$$

     where a_{ii} is the diagonal element and a_{ij} are the off-diagonal elements in row i. Diagonal dominance ensures that the influence of off-diagonal elements is small compared to the diagonal element.
   - The spectral radius $\rho(\mathbf{D}^{-1}(\mathbf{L}+\mathbf{U}))$ of the iteration matrix $\mathbf{D}^{-1}(\mathbf{L}+\mathbf{U})$ must be less than 1 for the method to converge, where **D** is the diagonal part of **A**, and **L** and **U** are the strictly lower and upper triangular parts of **A**, respectively.
2. **Convergence of Gauss Seidel:** It is updated method of Jacobi method. Unlike Jacobi method,Gauss seidel uses new values i.e most updated value for an unknown x_j for next iterations.

   $$|x_i^{k+1}| = \frac{1}{a_{ii}}(b_i - \sum_{j=1}^{i-1} |a_{ij} * x_j^{k+1}| - \sum_{j=i+1}^{n} |a_{ij} * x_j^{k}|,$$

   - The Gauss-Seidel method converges faster if the matrix **A** is diagonally dominant. This means that for each row i,

$$|a_{ii}| > \sum_{\substack{j=1 \\ j \neq i}}^{n} |a_{ij}|,$$

where a_{ii} is the diagonal element and a_{ij} are the off-diagonal elements in row i. Diagonal dominance ensures that the influence of off-diagonal elements is small compared to the diagonal element.

- The spectral radius $\rho((\mathbf{D}+\mathbf{L})^{-1}(\mathbf{U}))$ of the iteration matrix $(\mathbf{D}+\mathbf{L})^{-1}(\mathbf{U})$ must be less than 1 for the method to converge, where $\mathbf{D}$ is the diagonal part of $\mathbf{A}$, and $\mathbf{L}$ and $\mathbf{U}$ are the strictly lower and upper triangular parts of $\mathbf{A}$, respectively. Gauss seidel Method converges faster than Jacobi : Consider $\rho(\mathbf{G_J})$ and $\rho(\mathbf{G_G S})$ be spectral radius of jacobi and gauss seidel then we have relation

$$\rho(\mathbf{G_J})^2 = \rho(\mathbf{G_G S}) \tag{30}$$

This means that one Gauss-Seidel iteration is about as good as two Jacobi iterations; Gauss-Seidel converges twice as fast as Jacobi.

3. **Convergence of Conjugate gradient:**. The conjugate-gradient (CG) method is a simple variation on steepest descent that performs better because it has a memory.

The computation of a is organized a bit differently, but this difference is cosmetic. The line searches are still exact; the CG algorithm is an instance ofAlgorithm 7.4.14. **Initially $p \leftarrow r$, so the first step is steepest descent**. On subsequent steps there is a difference. Instead of $p \leftarrow r$, we have $p \leftarrow r + \beta p$. The residual or steepest descent direction still plays an important role in determining the new search direction, but now the old search direction also matters. This is the one point at which memory of past iterations is used. This slight change makes a huge difference.

$$\beta_k = \frac{(r^{k+1})^T r^{k+1}}{(r^k)^T r^k} \tag{31}$$

$$p^{k+1} = r^{k+1} + \beta_k p^k \tag{32}$$

We see that the CG algorithm is far superior to steepest descent.

The CG method generates search directions that are A -orthogonal (i.e., conjugate). This orthogonality ensures that each search direction is independent of the previous ones, leading to more efficient progress towards the solution.

$$p^k \perp p^{k-1} \perp p^{k-2} ..... \perp p^0 \tag{33}$$

$$(p^k)^T A p^i = 0 \tag{34}$$

This is called A- orthogonality, p^k is A- conjugate to all previous directions.

The CG algorithm, applied to an n x n positive definite system Ax = b, arrives at the exact solution in n or fewer steps.

4 Plot the L2 norm of the error in the solution (Eq.35) vs the mesh element size (h= π/Ni) on a base-10 logarithmic scale (log-log plot) for any one of the iterative methods. Also,report the average slope of the line thus plotted.

$$L_2(u_h) = (\int_\Omega (u - u_h)^2)^{\frac{1}{2}} \tag{35}$$

where,u_h is the finite element solution and analytical solution(u) is given by:

$$u = sin(x) \tag{36}$$

The full solution vector is reconstructed as:

$$u_h(x_i) = \begin{cases} 0, & \text{if } i = 0 \text{ or } i = N_i, \\ \bar{u}_{i-1}, & \text{if } i = 1, \ldots, N_i - 1, \end{cases} \tag{37}$$

where $\bar{u}$ contains the interior node values, and the boundary conditions $u_h(0) = u_h(\pi) = 0$ are enforced.

Within each element $[x_i, x_{i+1}]$, the solution is interpolated using linear basis functions:

$$u_h(x) = u_h(x_i)\phi_i(x) + u_h(x_{i+1})\phi_{i+1}(x), \quad x \in [x_i, x_{i+1}], \tag{38}$$

where:

$$\phi_i(x) = \frac{x_{i+1} - x}{h}, \quad \phi_{i+1}(x) = \frac{x - x_i}{h}. \tag{39}$$

The L2 error norm is computed by evaluating the integral:

$$L_2(u_h) = \left(\sum_{i=0}^{N_i - 1} \int_{x_i}^{x_{i+1}} \left(\sin(x) - u_h(x)\right)^2 \, dx \right)^{\frac{1}{2}}. \tag{40}$$

This integral is approximated numerically using adaptive quadrature to ensure high accuracy, as the integrand involves the smooth function $\sin(x)$ and the piecewise linear $u_h(x)$.

The mesh sizes are:

$$h = \frac{\pi}{N_i}, \quad N_i = 5, 10, 20, 40. \tag{41}$$

For each h, we compute $L_2(u_h)$, and plot $\log_{10}(L_2)$ versus $\log_{10}(h)$. The slope between consecutive points is calculated as:

$$slope_k = \frac{\log_{10}(L_2(h_{k+1})) - \log_{10}(L_2(h_k))}{\log_{10}(h_{k+1}) - \log_{10}(h_k)}, \tag{42}$$

4.1 Theoretical Expectation

For linear finite elements, the L2 error is expected to scale as:

$$\|u - u_h\|_{L_2} \leq Ch^2 \|u\|_{H^2}, \tag{43}$$

where C is a constant, and $\|u\|_{H^2}$ is the Sobolev norm of the analytical solution. Since $u(x) = \sin(x)$ is smooth ($u''(x) = -\sin(x)$), the second derivatives are bounded, and we expect:

$$L_2(u_h) \propto h^2. \tag{44}$$

Taking the base-10 logarithm:

$$\log_{10}(L_2) = 2\log_{10}(h) + \log_{10}(C'), \tag{45}$$

where C' is a constant. Thus, the log-log plot of $\log_{10}(L_2)$ versus $\log_{10}(h)$ should be a straight line with a slope of approximately 2.

4.2 Results

The log-log plot is presented in Figure 5, showing $\log_{10}(L_2)$ versus $\log_{10}(h)$ for the CG method.

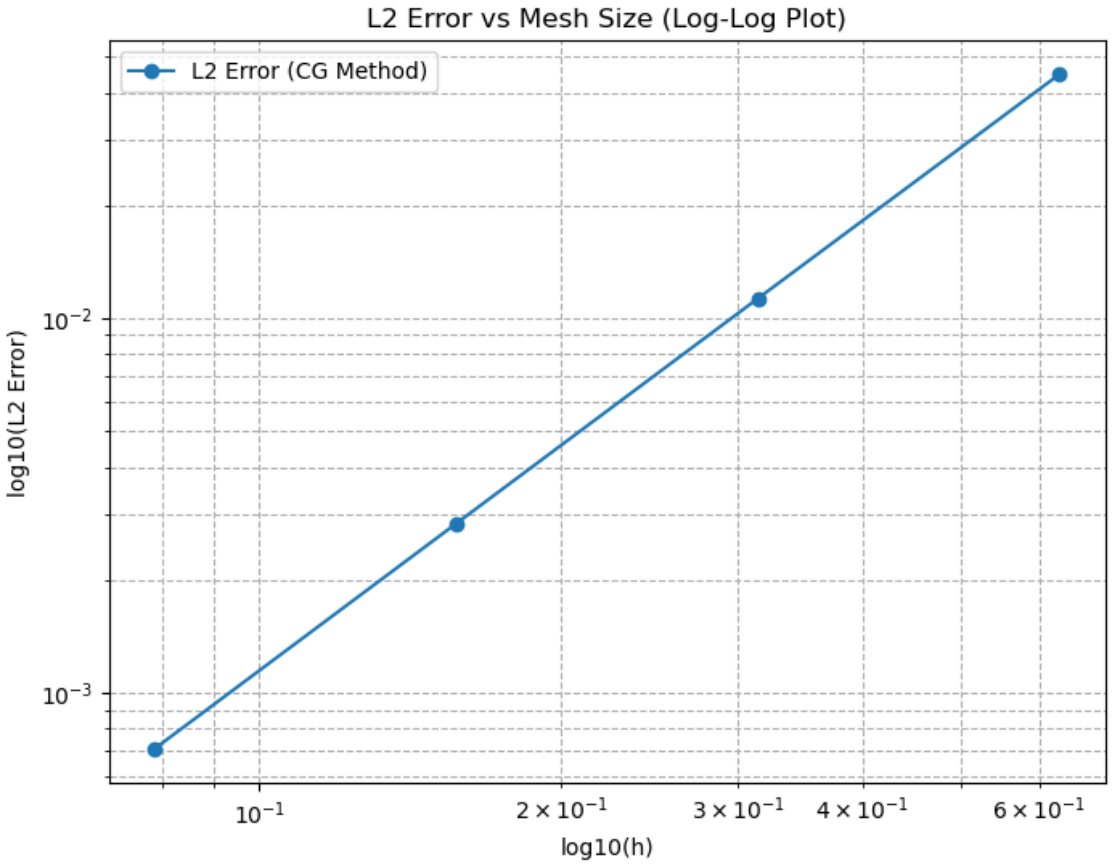

Figure 5: Log-log plot of L2 error versus mesh size h for the CG method, demonstrating quadratic convergence.

The average slope of the line, computed using Equation (8) is 1.996, is expected to be close to 2, confirming the quadratic convergence of the L2 error with respect to h.

5 Let the converged numerical solution corresponding to four mesh sizes N_1,N_2,N_3, and N_4 be u_{h1}, u_{h2}, u_{h3}, and u_{h4}, respectively. On a single figure, plot the analytical solution u, and numerical solutions,u_{h1}, u_{h2}, u_{h3}, and u_{h4} w.r.t. x for any one of the iterative methods.

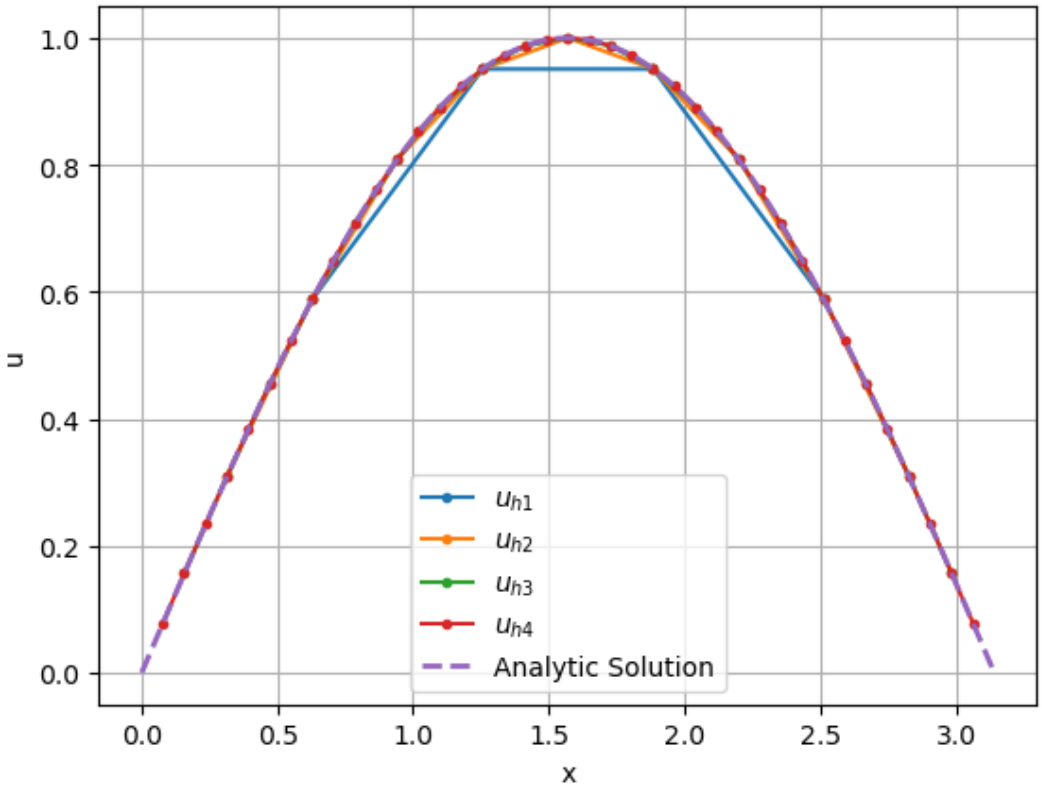

Figure 6: Numerical solution versus analytic solution

Convergence to Analytical Solution: As the mesh size N_i increases (i.e., element size $h = \pi/N_i$ decreases), the numerical solutions u_{hi} approach the analytical solution $u(x) = \sin(x)$. For $N_i = 5$, the solution u_{h1} shows noticeable deviations, particularly near the peak at $x = \pi/2$. In contrast, u_{h4} ($N_i = 40$) is nearly indistinguishable from the analytical solution, indicating improved accuracy with mesh refinement.