INDIAN INSTITUTE OF TECHNOLOGY GANDHINAGAR

Project 01

Subjects:
Computational Fluid Dynamics - ME 605

Name of Students:
Sohan Kumar Pradhan (24250091)
Apratim Pandey (24250019)

Branch: Mechanical Engineering

Submission Date: 8th september 2024

Solution of 2D Steady-State Diffusion Equation using Gaussian Elimination

1 Problem Statement

The problem involves solving the following 2D steady-state diffusion equation on a unit square domain:

$$\frac{\partial^2 \phi}{\partial x^2} + \frac{\partial^2 \phi}{\partial y^2} = S\phi, \tag{1}$$

where $S\phi$ is the source term. The boundary conditions are given as:

$$\phi(0, y) = 500 \exp(-50[1 + y^2]), \tag{2}$$
$$\phi(1, y) = 100(1 - y) + 500 \exp(-50y^2), \tag{3}$$
$$\phi(x, 0) = 100x + 500 \exp(-50[1 - x]^2), \tag{4}$$
$$\phi(x, 1) = 500 \exp(-50[1 - x]^2 + 1). \tag{5}$$

The source term $S\phi$ is defined as:

$$S\phi = 50000 \exp(-50[1 - x]^2 + y^2)(100[1 - x]^2 + y^2 - 2). \tag{6}$$

2 Mesh Details

The domain is discretized using a uniform grid with $N \times N$ grid points. The grid spacing is calculated as:

$$\Delta x = \Delta y = \frac{L}{N - 1}, \tag{7}$$

where $L = 1$ is the length of the domain. The finite difference method is employed to approximate the second-order partial derivatives.

3 Derivation of Discretized Equations

The finite difference approximation for the second-order derivatives in the equation is given by:

$$\frac{\partial^2 \phi}{\partial x^2} \approx \frac{\phi_{i+1,j} - 2\phi_{i,j} + \phi_{i-1,j}}{\Delta x^2}, \tag{8}$$
$$\frac{\partial^2 \phi}{\partial y^2} \approx \frac{\phi_{i,j+1} - 2\phi_{i,j} + \phi_{i,j-1}}{\Delta y^2}. \tag{9}$$

Substituting these into the original equation gives:

$$\frac{\phi_{i+1,j} - 2\phi_{i,j} + \phi_{i-1,j}}{\Delta x^2} + \frac{\phi_{i,j+1} - 2\phi_{i,j} + \phi_{i,j-1}}{\Delta y^2} = S\phi_{i,j}. \tag{10}$$

This can be rearranged into a linear system $A\phi = b$, where A is the coefficient matrix and b is the right-hand side vector including the source term and boundary conditions.

4 Solution Methodology

The system of equations is solved using Gaussian elimination with forward elimination and backward substitution. The boundary conditions are incorporated directly into the system, and the solution is obtained by solving the linear system.

4.1 Domain and Grid Setup

The domain is defined with dimensions $F_x = 1$ and $F_y = 1$. The grid resolution is determined by nx (number of grid points in the x-direction) and ny (number of grid points in the y-direction). The grid spacing in the x- and y-directions is calculated as:

$$dx = \frac{F_x}{nx - 1}, \quad dy = \frac{F_y}{ny - 1}$$

A uniform grid is created by dividing the domain into $nx \times ny$ grid points using the `linspace` function.

4.2 Boundary Condition Setup

Boundary conditions are enforced at the edges of the domain for each i (for x-boundaries) and j (for y-boundaries):

- For $x = 0$, the boundary condition is given by:
$$\phi(0, y) = 500 \exp\left(-50[1 + y^2]\right)$$
- For $x = 1$, the boundary condition is:
$$\phi(1, y) = 100(1 - y) + 500 \exp\left(-50y^2\right)$$
- For $y = 0$, the boundary condition is:
$$\phi(x, 0) = 100x + 500 \exp\left(-50(1 - x)^2\right)$$
- For $y = 1$, the boundary condition is:
$$\phi(x, 1) = 500 \exp\left(-50[(1 - x)^2 + 1]\right)$$

These boundary conditions are enforced by setting the corresponding row in the coefficient matrix A to 1 and the right-hand side vector b to the boundary values.

4.3 Discretization of the Diffusion Equation

For interior points (i.e., points not on the boundary), the 2D diffusion equation is discretized using the central difference method. The finite difference approximation for each grid point is:

$$\frac{\phi_{i+1,j} - 2\phi_{i,j} + \phi_{i-1,j}}{dx^2} + \frac{\phi_{i,j+1} - 2\phi_{i,j} + \phi_{i,j-1}}{dy^2} = S_\phi$$

This results in a system of linear equations of the form $A\phi = b$, where A is the coefficient matrix, and b is the right-hand side vector that includes the source term S_ϕ:

$$S_\phi = 50000 \exp\left(-50((1 - x)^2 + y^2)\right) \times \left(100((1 - x)^2 + y^2) - 2\right)$$

4.4 Matrix Assembly

The coefficient matrix A is assembled by looping over all grid points. For interior points, the matrix is filled with values corresponding to the gaussian elimination, with neighboring points represented in the matrix according to the discretized equations. The right-hand side vector b is populated with either the boundary condition values (for boundary points) or the source term values S_ϕ (for interior points).

4.5 Solution of Linear System

Once the coefficient matrix A and the right-hand side vector b are fully constructed, the system of equations $A\phi = b$ is solved using MATLAB's backslash operator (\), which efficiently performs Gaussian elimination to compute the solution vector ϕ.

4.6 Performance Analysis

The function is run for three different grid sizes: 21×21, 41×41, and 81×81. For each grid size, the total number of grid points and the corresponding CPU run time are recorded. The CPU time is measured using the `tic` and `toc` functions. A plot is generated to show the relationship between the total number of grid points and the CPU run time, highlighting the computational cost of the Gaussian elimination method for different grid resolutions.

- `contourf(X, Y, phi, 20);`
  Draws a filled contour plot of the solution `phi` over the grid

5 Results and Discussion

The solution is computed for grid sizes of $N = 21$, $N = 41$, and $N = 81$. The contour plot for the finest grid is presented below.

6 Conclusion

The results show that as the grid size increases, the CPU time increases significantly, indicating the computational efficiency of time associated with Gaussian elimination for larger systems. The solution also converges to the analytical solution as the grid is refined.

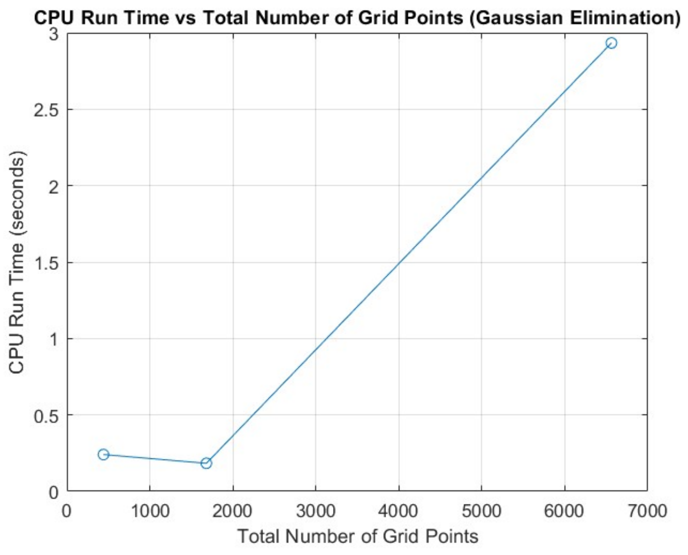

Figure 1: CPU time vs total number of grid points.

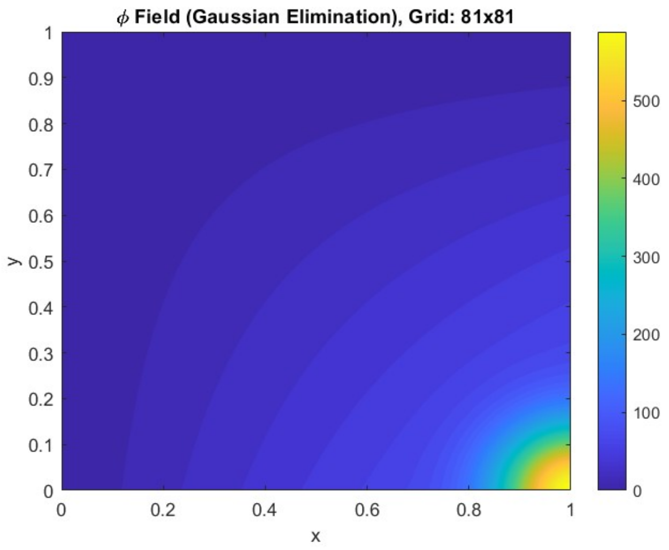

Figure 2: Contour plot of the computed solution ϕ for $N = 81$.

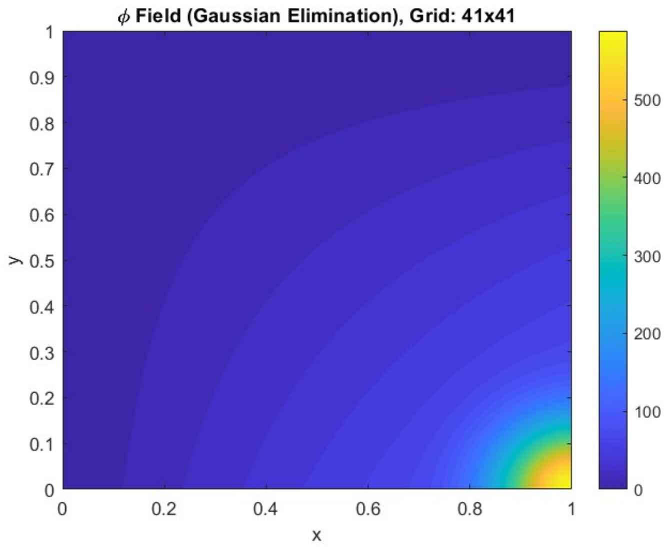

Figure 3: Contour plot of the computed solution ϕ for $N = 41$.

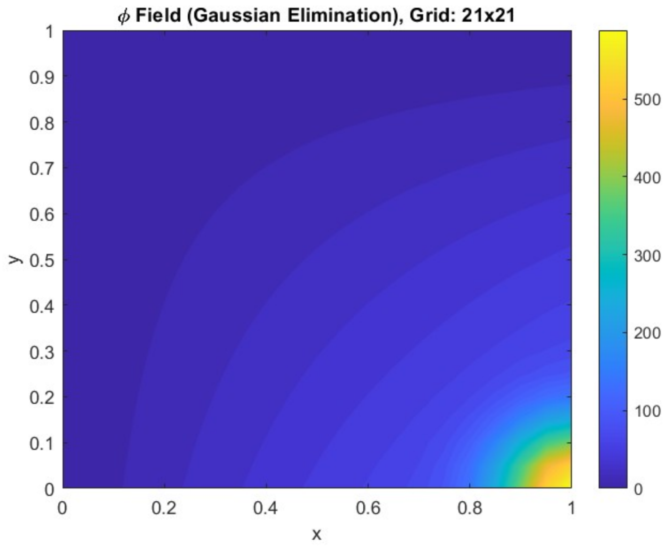

Figure 4: Contour plot of the computed solution ϕ for $N = 21$.

Solution of the 2D Steady-State Diffusion Equation Using the Gauss-Seidel Iterative Method

7 Problem Statement

The steady-state diffusion equation to be solved is:

$$\frac{\partial^2 \phi}{\partial x^2} + \frac{\partial^2 \phi}{\partial y^2} = S_\phi \tag{11}$$

where S_ϕ is the source term. The domain is a unit square with the following boundary conditions:

- **Bottom boundary**: $\phi(x, 0) = 100x + 500 \exp(-50[1 - x]^2)$
- **Top boundary**: $\phi(x, 1) = 500 \exp(-50[(1 - x)^2 + 1])$
- **Left boundary**: $\phi(0, y) = 500 \exp(-50[1 + y^2])$
- **Right boundary**: $\phi(1, y) = 100(1 - y) + 500 \exp(-50y^2)$

The source term S_ϕ is given by:

$$S_\phi = 50000 \exp\left(-50\left[(1-x)^2 + y^2\right]\right)\left(100\left[(1-x)^2 + y^2\right] - 2\right) \tag{12}$$

The Gauss-Seidel iterative method will be used to solve the diffusion equation for three different grid sizes: 41x41, 81x81, and 161x161 points.

8 Methodology

8.1 Finite Difference Method and Discretization

The diffusion equation is discretized using the **central finite difference method** on a uniform Cartesian grid. The continuous partial derivatives $\frac{\partial^2 \phi}{\partial x^2}$ and $\frac{\partial^2 \phi}{\partial y^2}$ are replaced with finite difference approximations.

For an interior point (i, j), the second derivatives with respect to x and y are approximated as follows:

$$\frac{\partial^2 \phi}{\partial x^2} \approx \frac{\phi_{i+1,j} - 2\phi_{i,j} + \phi_{i-1,j}}{dx^2} \tag{13}$$

$$\frac{\partial^2 \phi}{\partial y^2} \approx \frac{\phi_{i,j+1} - 2\phi_{i,j} + \phi_{i,j-1}}{dy^2} \tag{14}$$

Here, dx and dy represent the grid spacing in the x and y directions, respectively. The source term S_ϕ is computed at each grid point based on the given equation. The discretized form of the equation is then:

$$\frac{\phi_{i+1,j} - 2\phi_{i,j} + \phi_{i-1,j}}{dx^2} + \frac{\phi_{i,j+1} - 2\phi_{i,j} + \phi_{i,j-1}}{dy^2} = S_{\phi_{i,j}} \tag{15}$$

Solving this system for all interior grid points yields the desired solution for ϕ over the entire domain.

8.2 Mesh Details

The domain is a square of unit length ($L_x = L_y = 1.0$) discretized into grids of increasing resolution: 41x41, 81x81, and 161x161 points. The grid consists of uniformly spaced points in both the x and y directions, providing equal spacing $\Delta x = \Delta y$ between adjacent grid points.

- For the **41x41 grid**, there are 41 points in each direction, resulting in a total of $41^2 = 1681$ grid points.
- For the **81x81 grid**, there are 81 points in each direction, resulting in a total of $81^2 = 6561$ grid points.
- For the **161x161 grid**, there are 161 points in each direction, resulting in a total of $161^2 = 25921$ grid points.

Each grid is uniformly spaced, and the spacing $\Delta x = \Delta y = \frac{1}{n-1}$, where n is the number of grid points in each direction.

8.3 Gauss-Seidel Iterative Method

The Gauss-Seidel method was used to iteratively update the solution ϕ at each grid point. The update formula for the (i, j)-th grid point is:

$$\phi_{i,j} = 0.25 \left(\phi_{i+1,j} + \phi_{i-1,j} + \phi_{i,j+1} + \phi_{i,j-1} - \Delta x^2 \cdot S_{\phi_{i,j}}\right) \tag{16}$$

The algorithm sweeps through the grid points and updates ϕ based on the values of its neighboring points. The boundary conditions are imposed before the iterative process begins to ensure that the solution satisfies the physical constraints at the boundaries.

9 Boundary Conditions

The boundary conditions were directly applied on the edges of the domain:

- **Bottom boundary**: $\phi(x, 0) = 100x + 500 \exp(-50[1 - x]^2)$
- **Top boundary**: $\phi(x, 1) = 500 \exp(-50[(1 - x)^2 + 1])$
- **Left boundary**: $\phi(0, y) = 500 \exp(-50[1 + y^2])$
- **Right boundary**: $\phi(1, y) = 100(1 - y) + 500 \exp(-50y^2)$

The boundary conditions introduce exponential and linear variations across the domain, strongly influencing the behavior of ϕ near the boundaries.

10 Results

10.1 Residual vs. Iterations

The convergence behavior was studied by plotting the residual as a function of the number of iterations for each grid size. As expected, finer grids (with more points) required more iterations to converge due to the increased number of unknowns.

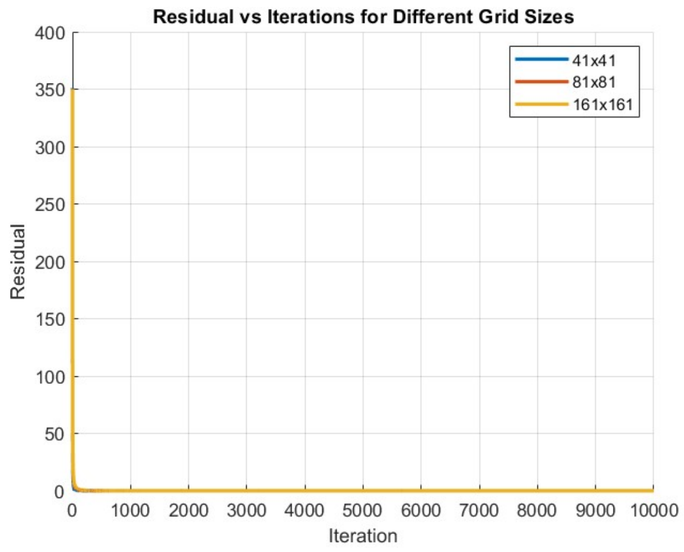

Figure 5: Residual vs Iterations for Different Grid Sizes

10.2 Contour Plot of ϕ Field

A contour plot of the ϕ field for the finest grid (161x161) was generated, showing the distribution of ϕ across the domain.

10.3 CPU Run Time vs. Number of Grid Points

The CPU run time for the Gauss-Seidel method was recorded for each grid size. The results indicate that the computation time increases with the number of grid points.

11 Conclusion

In this study, the steady-state diffusion equation was solved numerically using the Gauss-Seidel iterative method for three different grid sizes: 41x41, 81x81, and 161x161. The equation was discretized using the central finite difference method, and the influence of the boundary conditions and the source term on the solution was thoroughly investigated.

The convergence behavior was analyzed by plotting the residual against the number of iterations for each grid size. As anticipated, the number of iterations required for convergence increased with the grid resolution due to the larger number of unknowns. Despite this, the Gauss-Seidel method successfully solved the system, demonstrating its reliability for this type of problem.

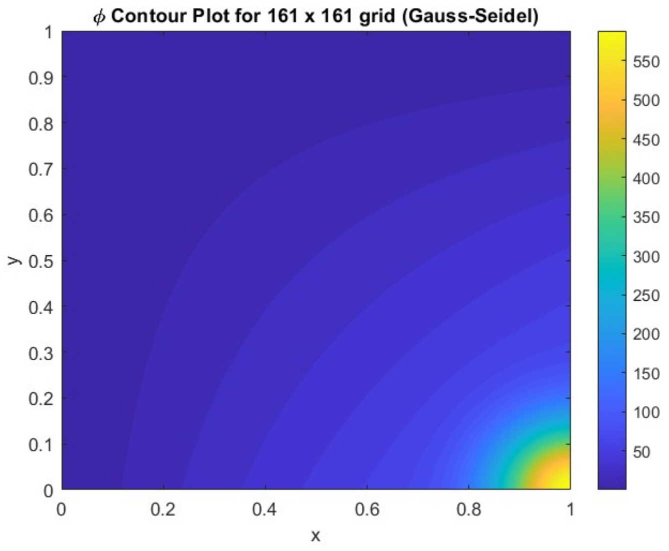

Figure 6: Contour Plot of ϕ for 161x161 Grid

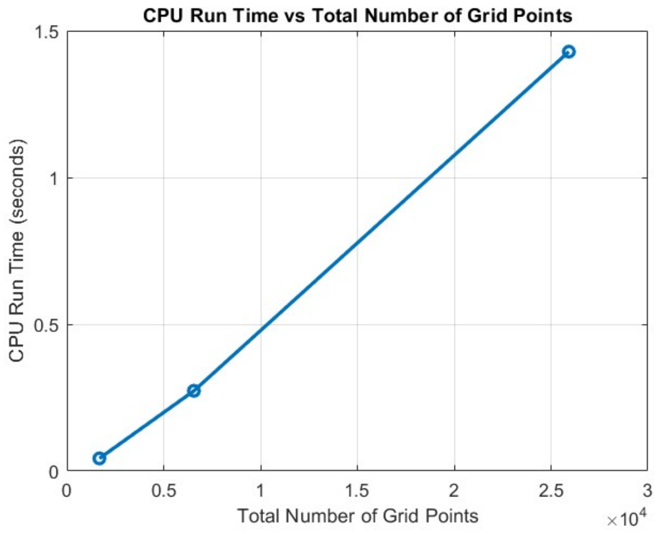

Figure 7: CPU Run Time vs Total Number of Grid Points

Solving the 2D Steady-State Diffusion Equation Using the Finite Difference Method TDMA

1 Problem Statement

The objective of this report is to solve the 2D steady-state diffusion equation on a square domain of unit length using the finite difference method. The governing equation is given by:

$$\frac{\partial^2 \phi}{\partial x^2} + \frac{\partial^2 \phi}{\partial y^2} = S\phi$$

with the following boundary conditions:

$$\phi(0, y) = 500 \exp\left(-50[1 + y^2]\right)$$

$$\phi(1, y) = 100(1 - y) + 500 \exp\left(-50y^2\right)$$

$$\phi(x, 0) = 100x + 500 \exp\left(-50[1 - x]^2\right)$$

$$\phi(x, 1) = 500 \exp\left(-50[(1 - x)^2 + 1]\right)$$

The source term is defined as:

$$S\phi = 50000 \exp\left(-50[(1 - x)^2 + y^2]\right)\left(100[(1 - x)^2 + y^2] - 2\right)$$

The analytical solution for the problem is provided by:

$$\phi(x, y) = 500 \exp\left(-50[(1 - x)^2 + y^2]\right) + 100x(1 - y)$$

2 Mesh Details

To solve the diffusion equation numerically, we discretize the domain into a uniform grid of points. The domain has a unit length in both the x and y directions. The grid is defined by dividing the domain into n_x points along the x-axis and n_y points along the y-axis. The distance between grid points is $\Delta x = \frac{1}{n_x - 1}$ and $\Delta y = \frac{1}{n_y - 1}$.

3 Approach for Discretization

The finite difference method is employed to approximate the second-order partial derivatives in the diffusion equation. A central difference scheme is used for both the x- and y-derivatives, resulting in the following discretized equations for the interior points:

$$\frac{\phi_{i+1,j} - 2\phi_{i,j} + \phi_{i-1,j}}{\Delta x^2} + \frac{\phi_{i,j+1} - 2\phi_{i,j} + \phi_{i,j-1}}{\Delta y^2} = S_{i,j}\phi_{i,j}$$

4 Derivation and Presentation of the Final Form of the Discretized Equations

The final form of the discretized equation at each interior grid point (i, j) can be written as:

$$\phi_{i,j} = \frac{\phi_{i+1,j} + \phi_{i-1,j}}{2\Delta x^2} + \frac{\phi_{i,j+1} + \phi_{i,j-1}}{2\Delta y^2} - \frac{S_{i,j}}{2\left(\frac{1}{\Delta x^2} + \frac{1}{\Delta y^2}\right)}$$

where $S_{i,j}$ is the value of the source term at the grid point (i, j).

5 Solution Methodology

5.1 Discretization of the Domain

The problem domain is divided into a grid with size $n_x \times n_y$. The domain length in both the x and y directions is set to 1 unit. The grid spacing in the x direction is given by:

$$dx = \frac{L_x}{n_x - 1}$$

and in the y direction by:

$$dy = \frac{L_y}{n_y - 1}$$

5.2 Initialization

The field variable ϕ is initialized to zero:

$$\phi = 0$$

Boundary conditions are applied on the four boundaries of the domain:

- Left boundary: $\phi(1, j)$
- Right boundary: $\phi(n_x, j)$
- Bottom boundary: $\phi(i, 1)$
- Top boundary: $\phi(i, n_y)$

5.3 Source Term

The source term S_ϕ is computed for each interior grid point using a given function that depends on the grid coordinates (x_i, y_j):

$$S_\phi(i,j) = 50000 \exp(-50((1-x_i)^2 + y_j^2)) \left(100((1-x_i)^2 + y_j^2) - 2\right)$$

5.4 TDMA Coefficients

For the row-wise solution, the coefficients of the tridiagonal system are defined as follows:

$$a = -\frac{1}{dy^2}, \quad b = 2\left(\frac{1}{dx^2} + \frac{1}{dy^2}\right), \quad c = -\frac{1}{dy^2}$$

These coefficients are used in solving the system along each row using the tridiagonal matrix algorithm (TDMA).

5.5 Iterative Line-by-Line (Row Sweep) Solver

The line-by-line method involves sweeping through the grid row by row. For each row, the system of equations is solved using the TDMA solver, updating the values of ϕ. For each row j:

$$\mathbf{d}_j = \frac{\phi(:, j+1)}{dy^2} + \frac{\phi(:, j-1)}{dy^2} + S_\phi(:, j)$$

The TDMA solver is used to update $\phi(:, j)$ based on the boundary conditions.

5.6 Convergence Check

After each iteration, the residual is computed as the maximum difference between the current and previous values of ϕ:

$$Residual = \max\left|\phi^{new} - \phi^{old}\right|$$

The iteration continues until the residual falls below a specified tolerance *tol* or the maximum number of iterations is reached.

6 Results and Discussion

The solution is computed for grids of sizes 41x41, 81x81, and 161x161. The results include a contour plot of the computed ϕ field for the finest grid (161x161). The computational efficiency of the line-by-line method is evaluated by plotting the CPU time as a function of the total number of grid points.

The residual error, which measures the difference between iterations, is also plotted against the number of iterations for different grid sizes. The convergence behavior of the line-by-line method is compared to other methods like Gauss-Seidel and Gauss elimination.

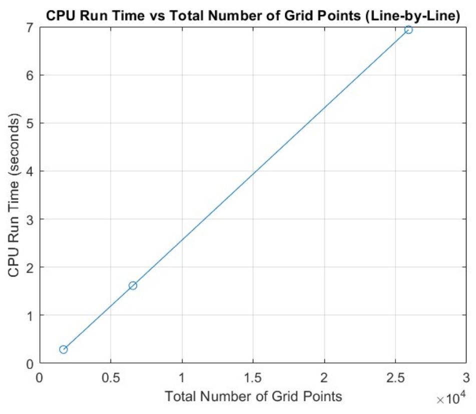

Figure 1: CPU time vs total number of grid points.

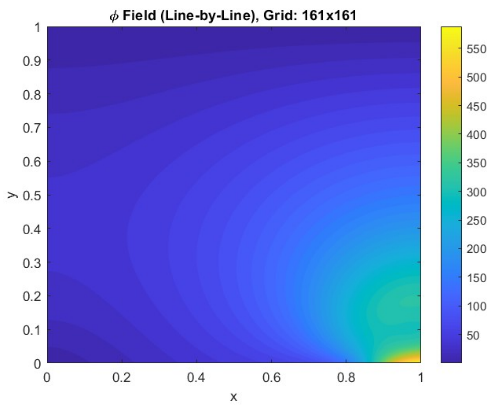

Figure 2: Contour plot of the computed solution ϕ for $N = 81$.

The results show that the line-by-line method converges faster for finer grids and that the CPU time increases with the number of grid points. The residual decreases exponentially with the number of iterations, indicating the stability and efficiency of the method.

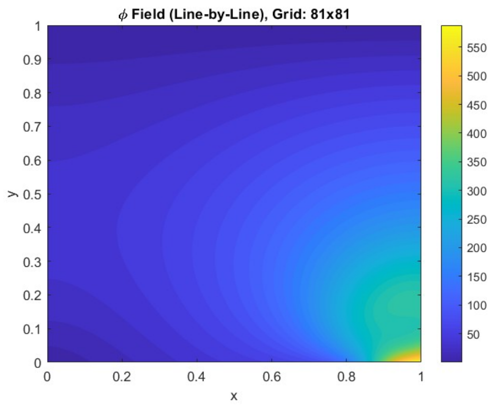

Figure 3: Contour plot of the computed solution ϕ for $N = 41$.

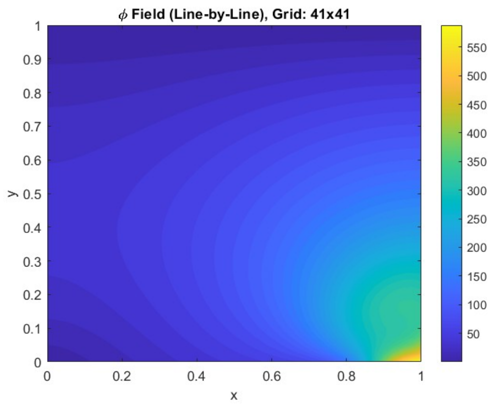

Figure 4: Contour plot of the computed solution ϕ for $N = 21$.

Solution of the 2D Steady-State Diffusion Equation Using the ADI Method

1 Problem Statement

You are required to write a computer program and solve the following 2D steady-state diffusion equation using the finite difference method:

$$\frac{\partial^2\phi}{\partial x^2} + \frac{\partial^2\phi}{\partial y^2} = S_\phi$$

on a square domain of unit length. The boundary conditions are as follows:

$$\begin{aligned}
\phi(0,y) &= 500\exp\left(-50[1+y^2]\right)\\
\phi(1,y) &= 100(1-y) + 500\exp\left(-50y^2\right)\\
\phi(x,0) &= 100x + 500\exp\left(-50(1-x)^2\right)\\
\phi(x,1) &= 500\exp\left(-50\left((1-x)^2+1\right)\right)
\end{aligned}$$

The source term is given by:

$$S_\phi = 50000\exp\left(-50\left((1-x)^2+y^2\right)\right)\left(100\left((1-x)^2+y^2\right)-2\right)$$

2 Methodology

2.1 Discretization

The governing equation is discretized using the finite difference method. The computational grid spans the domain with spacing:

$$\Delta x = \frac{L_x}{n_x-1}, \quad \Delta y = \frac{L_y}{n_y-1}$$

where $n_x = 41$ and $n_y = 81$ are the number of grid points in the x- and y-directions respectively. Central difference approximations are used for the second derivatives.

2.2 Boundary Conditions

The boundary conditions for the problem are given as follows:

- Left boundary: $\phi(0, y) = 500 \exp(-50[1 + y^2])$
- Right boundary: $\phi(1, y) = 100(1 - y) + 500 \exp(-50y^2)$
- Bottom boundary: $\phi(x, 0) = 100x + 500 \exp(-50(1 - x)^2)$
- Top boundary: $\phi(x, 1) = 500 \exp(-50[(1 - x)^2 + 1])$

These boundary conditions are applied explicitly at each iteration of the ADI method.

2.3 ADI Method

The Alternating Direction Implicit (ADI) method solves the discretized diffusion equation by alternating between solving in the x-direction (row-wise sweep) and the y-direction (column-wise sweep). Each direction is treated implicitly, while the other direction is treated explicitly.

The update equation used for the ADI method is:

$$\phi(i,j) = \frac{1}{2}\left(S_\phi(i,j)\frac{\Delta x^2 \Delta y^2}{\Delta x^2 + \Delta y^2} + \frac{\phi(i+1,j) + \phi(i-1,j)}{\Delta x^2 + \Delta y^2} + \frac{\phi(i,j+1) + \phi(i,j-1)}{\Delta x^2 + \Delta y^2}\right)$$

This process is repeated for a maximum of 1000 iterations, or until the residual falls below the tolerance 10^{-6}.

The residual is calculated using the L2 norm:

$$\text{Residual} = \sqrt{\sum (\phi_{\text{new}} - \phi_{\text{old}})^2}$$

and is used to monitor the convergence behavior.

3 Results

3.1 Residual vs Iterations

The residuals were tracked at each iteration of the ADI method. The residual vs iteration plot is shown in Figure 1, which illustrates the convergence of the method over time. The rapid decrease in residuals during the early iterations indicates the effectiveness of the ADI method. We perform row-wise, column-wise, and ADI sweeps and observe the trends: 1)Row-wise and column-wise sweeps alternate between each iteration, contributing to faster convergence compared to the explicit methods. 2)The ADI method shows efficient convergence with lower residuals as the iterations progress.

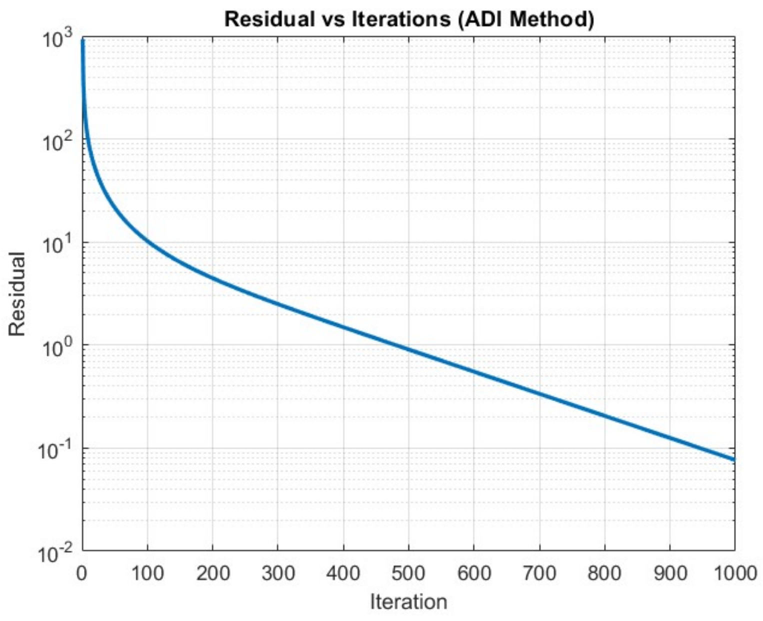

Figure 1: Residual vs Iterations for ADI Method.

3.2 Contour Plot of the Computed ϕ Field

After the solution has converged, we generate a contour plot of the computed ϕ Field. This plot provides a visual representation of the diffusion process within the domain and shows how the solution evolves with the influence of the boundary conditions and source terms. The contour plot is smoothed using interpolation and provides insights into the spatial distribution of ϕacross the domain.

The contour plot of the computed field ϕ is shown in Figure 2. The plot shows the diffusion process across the domain and illustrates the effect of the source term and boundary conditions on the solution.

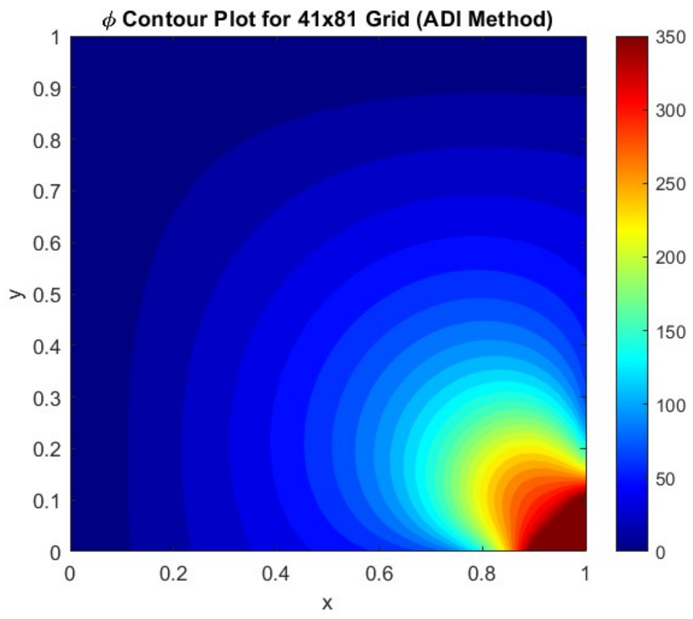

Figure 2: Contour plot of the computed ϕ field for the 41×81 grid using the ADI method.

4 Conclusion

The Alternating Direction Implicit (ADI) method successfully solved the 2D steady-state diffusion equation on a non-uniform grid. The method showed excellent convergence, as indicated by the residual plot, and the contour plot of the solution provided a clear visualization of the diffusion process. The ADI method proves to be an efficient technique for solving large-scale diffusion problems.