Computational Modeling
Project 4

Maroua Ouahbi
Constructor University

Electric Potential:

In electrostatics, the electric potential V at a point in space represents the amount of potential energy per unit charge at that point. It is defined as the work required to move a test charge from infinity to a specific point in an electric field. For regions where there are no external charges, the electric potential satisfies Laplace's equation:

$$\nabla^2 V = 0$$

This partial differential equation governs the behavior of the potential in regions with no charge, except where boundary conditions are specified. The boundary conditions in this problem are as follows:

- The long sides of a rectangular region are held at a fixed potential of 10 V.
- The short sides of the rectangle are grounded, i.e., they are held at 0 V.
- A point charge is placed at the center of the rectangle, influencing the potential at nearby grid points.

Given that Laplace's equation does not have a simple analytical solution for general geometries like the rectangle described, numerical methods are employed to approximate the solution. In this project, we solve Laplace's equation within a rectangular region using three iterative methods:

1. **Jacobi Method**
2. **Gauss-Seidel Method**
3. **Successive Over-Relaxation (SOR) Method**

Methods and Code Overview

We begin by discretizing the region into a grid and setting initial boundary conditions. The methods we use for solving the Laplace equation are as follows:

1. **Jacobi Method**: An iterative method where the potential at each grid point is updated based on the average of its four neighboring points (up, down, left, and right).
2. **Gauss-Seidel Method**: Similar to the Jacobi method, but in this method, the grid points are updated sequentially using the most recent values of their neighbors, which leads to faster convergence.
3. **Successive Over-Relaxation (SOR)**: An enhancement of the Gauss-Seidel method that introduces a relaxation factor ω. This factor helps speed up convergence by controlling the weighting between the old and new values of the potential:

1. **Jacobi Method:**

This is an iterative method for solving the Laplace equation. In each iteration, the potential at each grid point is updated by averaging the potentials of its four neighboring points (up, down, left, right). The process continues until the potential values converge (when the changes between iterations become very small).

2. **Gauss-Jacobi Method**:

Similar to the Jacobi method, but it updates all grid points simultaneously using the potential values from the previous iteration. This can be more efficient than the Jacobi method in certain cases.

3. **Successive Over-Relaxation (SOR)**:

This method is an enhancement of the Gauss-Jacobi method. It introduces a relaxation factor ω, which is a weight between the old value and the newly computed value. This helps to accelerate the convergence, allowing the solution to reach the correct value faster.

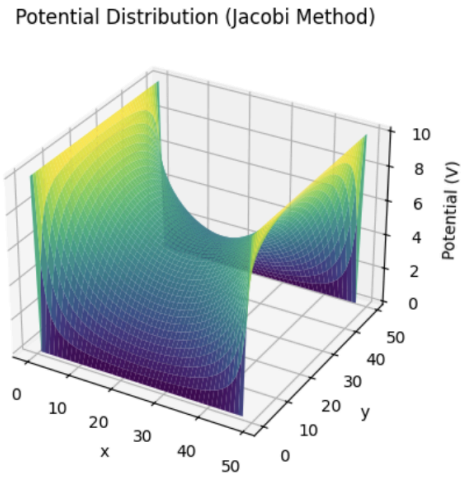
CODE RUN-THROUGH

The code starts by setting up a grid where the boundary conditions are defined: the long sides of the rectangle are fixed at 10 V, the short sides at 0 V, and a charge is placed in the center to influence the potential. Three different methods are used to solve the problem. The Jacobi Method updates the potential at each point by averaging the values from its neighbors, while the Gauss-Seidel Method speeds up the process by using the latest updates during calculations. The SOR Method goes a step further by adding a relaxation factor to improve convergence even more. Each method runs until the potential values stop changing significantly, and the results are visualized with 3D surface plots to show how the potential varies across the rectangle.

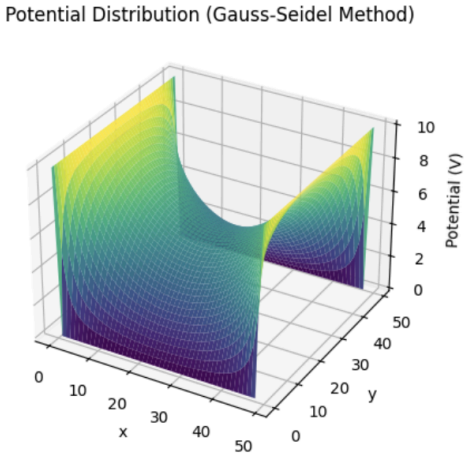
RESULTS & ANALYSIS

After running the Jacobi, Gauss-Seidel, and SOR methods, we observed that each method converged at different rates:

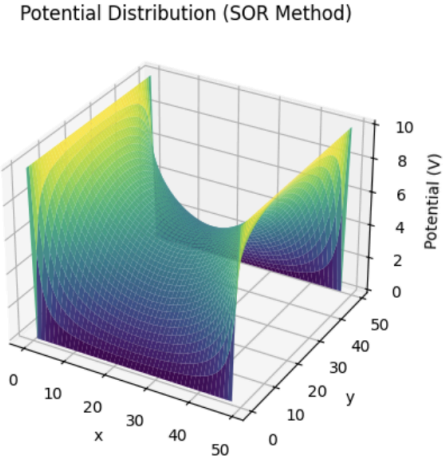
- **Jacobi Method**: This method was slow to converge, requiring many iterations to achieve a solution with the specified tolerance. This is expected since it updates each grid point simultaneously using values from the previous iteration.



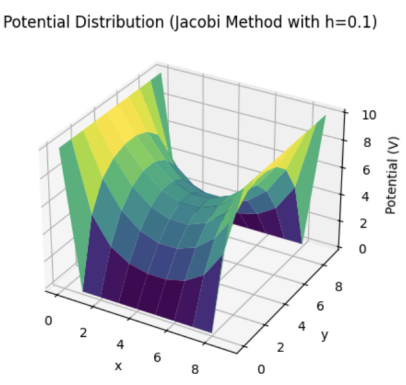
- **Gauss-Seidel Method**: The Gauss-Seidel method showed faster convergence compared to Jacobi, as it updates the grid points sequentially using the most recent values.

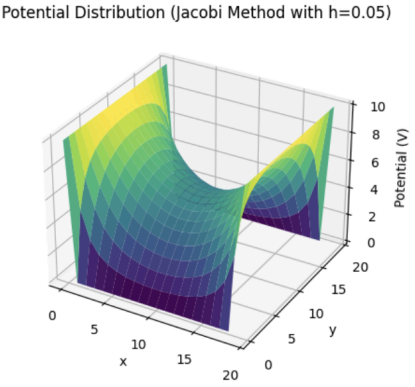


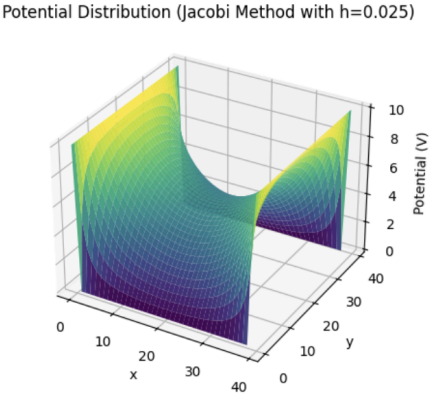
- **Successive Over-Relaxation (SOR)**: The SOR method, with an optimized relaxation factor $\omega=1.5$, converged the fastest among the three methods. The use of the relaxation factor accelerated the convergence by weighting the updates between old and new values.



We also investigated the effect of grid spacing (h) on the accuracy of the results. As the grid spacing decreased (the grid resolution increased), the solution became more accurate. However, the computational resources required also increased, highlighting the trade-off between accuracy and efficiency. Smaller grid spacing leads to a higher number of grid points and, consequently, more computational work for each iteration.







CONCLUSION

In this project, we calculated the electric potential within a rectangular region, considering boundary conditions and a central charge, using iterative methods like Jacobi, Gauss-Jacobi, and Successive Over-Relaxation (SOR). We found that the convergence of the solution depended heavily on the number of iterations, with different methods converging at different rates. By visualizing the potential with surface and 3D plots, we were able to observe how the potential varied across the grid.

We also explored how the grid spacing impacted the accuracy of the results. Smaller grid steps improved the precision of the solution, but also required more computational resources. This highlights the trade-off between accuracy and efficiency when using numerical methods to solve electrostatic problems.

Overall, this project provides valuable insights into solving Laplace's equation using numerical methods, emphasizing the importance of choosing the right method and grid spacing for accurate and efficient results. Future work could explore other optimization techniques to improve convergence rates and computational efficiency.