MECE 5397: Scientific Computing for Mechanical Engineers

Dr. Andrea Prosperetti

Dr. Amit Amritkar

**Solving two-dimensional Poisson’s Equation on a Rectangle using Gauss-Seidel and Successive Over-Relaxation Methods**

**Project ID: APc1-3**

Nidal Kiwai Chaban

1278641

Table of Contents

[Abstract 2](#_Toc481613843)

[Mathematical Statement of the Problem 3](#_Toc481613844)

[Discretization of the Equations 4](#_Toc481613845)

[Discretizing Poisson Equation 4](#_Toc481613846)

[Description of Numerical Methods Used 5](#_Toc481613847)

[Gauss-Seidel Method 5](#_Toc481613848)

[Successive Over Relaxation 6](#_Toc481613849)

[Technical Specifications of Computer Used 7](#_Toc481613850)

[Results 7](#_Toc481613851)

[Graphs 7](#_Toc481613852)

[Parameters used in simulations 8](#_Toc481613853)

[Effect of Number of Points Used For Discretization 9](#_Toc481613854)

[Verification 12](#_Toc481613855)

# Table of Figures

[Figure 1-Contour Plot 8](#_Toc481613959)

[Figure 2-Surface Plot 9](#_Toc481613960)

[Figure 3-Surface plot of 100x100 mesh for Gauss-Seidel 11](#_Toc481613961)

[Figure 4-Contour plot of 100x100 mesh for Gauss-Seidel 11](#_Toc481613962)

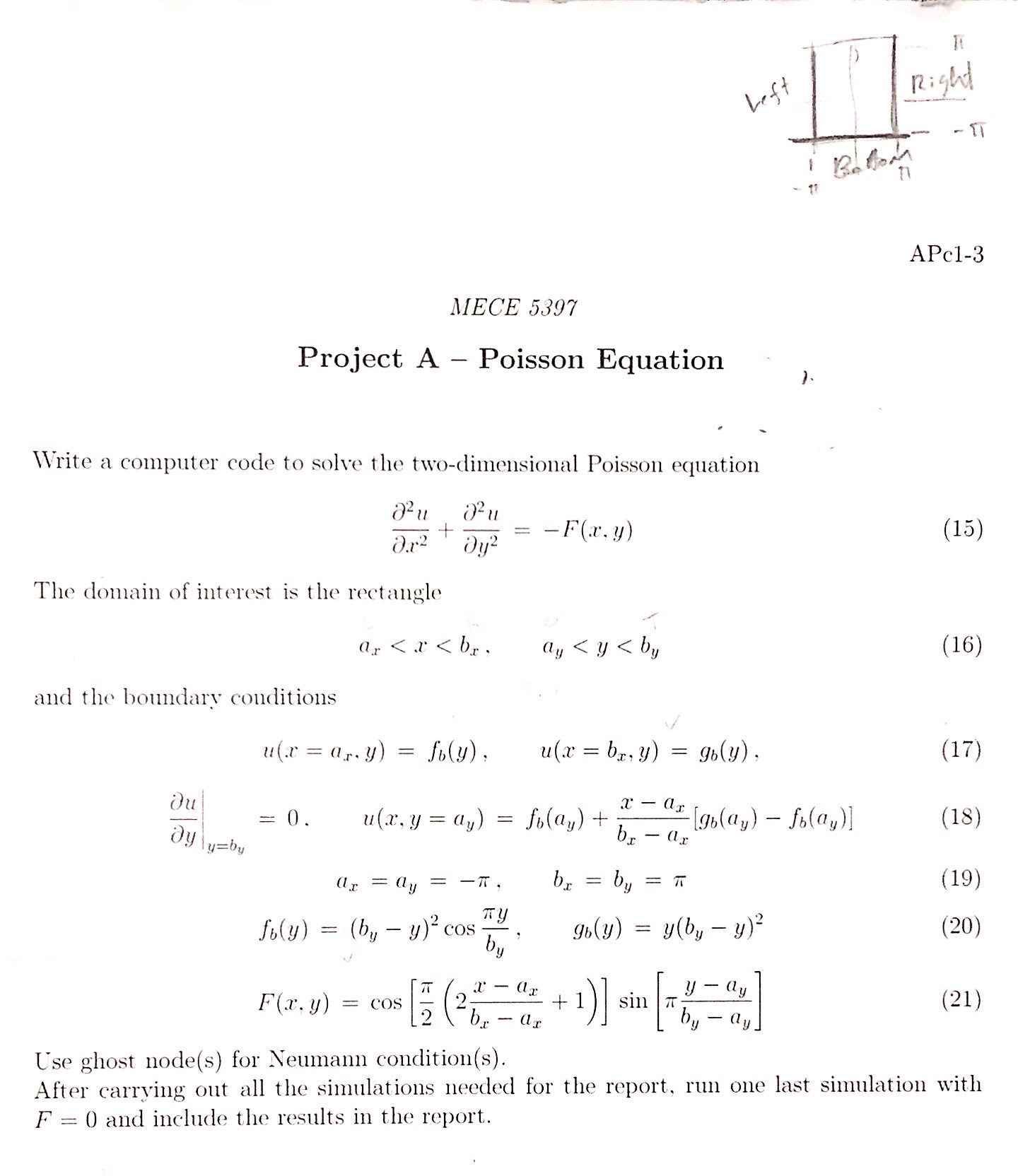
[Figure 5-Surface Plot of 200 x 200 mesh of SOR 12](#_Toc481613963)

[Figure 6-Contour Plot of 200x200 mesh for SOR 12](#_Toc481613964)

# Abstract

Poisson’s equation in 2D is a very popular elliptic partial differential equation used very widely in mechanical engineering and theoretical physics. It describes the potential field caused by a given density distribution when the potential field is known. As part of this project, iterative approaches as Gauss-Seidel and Successive Over-Relaxation were used to solve this equation with set boundary conditions, several codes were created; these codes are optimized (as much as possible to the author’s knowledge) and commented thoroughly. Also, the codes include a checkpoint functionality and each has a counterpart code with a restart functionality in the case that the original code is stopped meanwhile it is running. Also, a separate code for verification and grid independence study was created using the Successive Over-Relaxation method. The project results show a steady and converging function for different mesh sizes and iterations with different orders of convergence as well as depiction of the discrepancies in the use of coarser vs finer meshes in approximation.

# Mathematical Statement of the Problem



# Discretization of the Equations

## Discretizing Poisson Equation

Rearranging for ui,j

For Gauss-Seidel Method

For Successive Over Relaxation Method

# Description of Numerical Methods Used

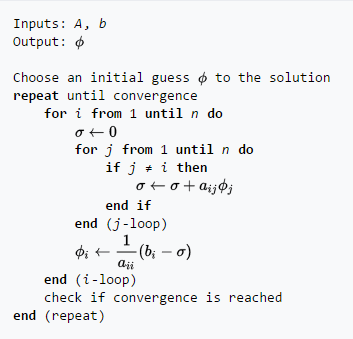
## Gauss-Seidel Method

The Gauss-Seidel Method is an iterative linear technique for solving square systems. It is defined by the form

C:\Users\nkiwaich\Pictures\Capture.PNG

where the procedure is generally continued until the changes between x and the next iteration of x are below a set tolerance.

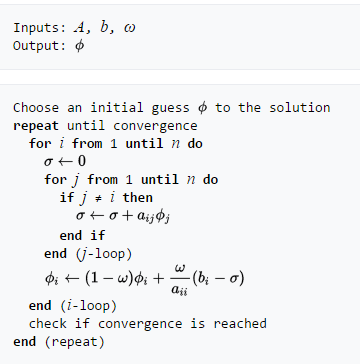
Convergence for Gauss-Seidel method are dependent on the matrix A. It converges if A is symmetric positive-definite or A is strictly diagonally dominant. Even if conditions are not satisfied, Gauss-Seidel converges sometimes.



## Successive Over Relaxation

The successive over-relaxation (SOR) is a variant of the Gauss-Seidel method for solving a linear system of equations, resulting in faster convergence. C:\Users\nkiwaich\Pictures\4.PNG where w is the relaxation factor, thusC:\Users\nkiwaich\Pictures\Capture1.PNG

Where w is the choice for relaxation, and depends of the coefficient matrix. Usually, w is greater than 0 and smaller than 2.



# Technical Specifications of Computer Used

* Processor – Intel Core i7-3770S CPU @ 3.10GHz
* RAM - 8 GB
* Hard Drive - 500 GB
* Graphics Card - any with DisplayPort/HDMI or DVI support - desktop only
* Monitor – Dell OptiPlex widescreen LCD with DisplayPort/HDMI or DVI support

# Results

## Graphs

For the Project given, we obtained a contour plot as shown

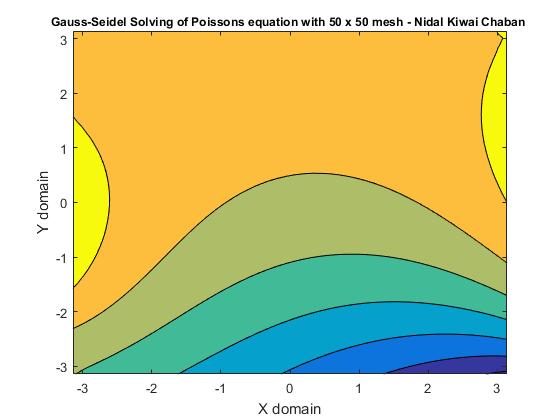


Figure 1-Contour Plot

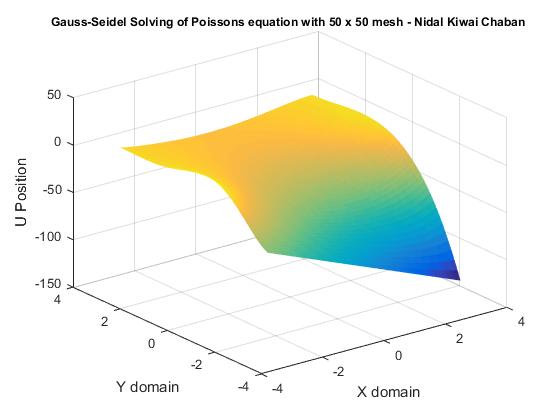


Figure 2-Surface Plot

## Parameters used in simulations

* Input Parameters
  + Number of nodes n for n x n mesh size
  + ax = ay = -π
  + bx = by = π
  + Boundary Conditions as stated
* Output Parameters
  + Number of iterations
  + Elapsed time
  + Mean of u (for grid independence)

## Effect of Number of Points Used For Discretization

For Gauss-Seidel

|  |  |  |
| --- | --- | --- |
| Gauss-Seidel Method | | |
| Mesh size | Iterations for tol=1e-6 | Elapsed time (seconds) |
| 10x10 | 34 | .037481 |
| 20x20 | 769 | .5911 |
| 50x50 | 4992 | 5.66963 |
| 100x100 | 20976 | 34.6258 |
| 200x200 | Excessive time consumed | |
| 1000x1000 |
| 5000x5000 |

For Successive Over-Relaxation

|  |  |  |
| --- | --- | --- |
| Successive Over-Relaxation Method | | |
| Mesh size | Iterations for tol=1e-6 | Elapsed time (seconds) |
| 10x10 | 8 | .008049 |
| 20x20 | 269 | .158379 |
| 50x50 | 1811 | 1.503082 |
| 100x100 | 7547 | 12.2652 |
| 200x200 | 33060 | 164.638 |
| 1000x1000 | Excessive time consumed | |
| 5000x5000 |

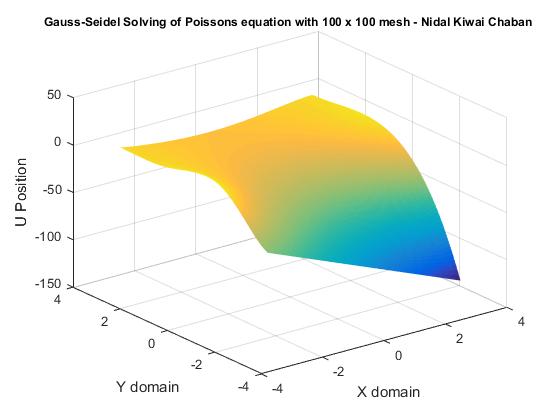


Figure 3-Surface plot of 100x100 mesh for Gauss-Seidel

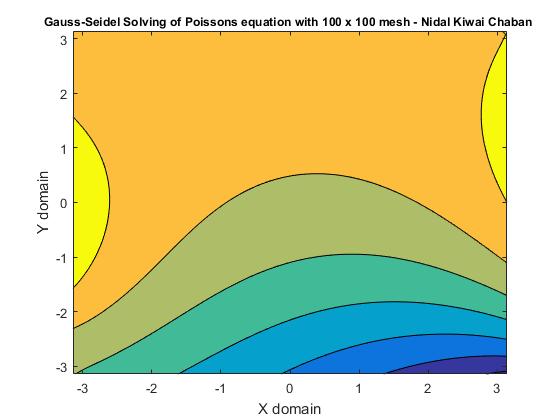


Figure 4-Contour plot of 100x100 mesh for Gauss-Seidel

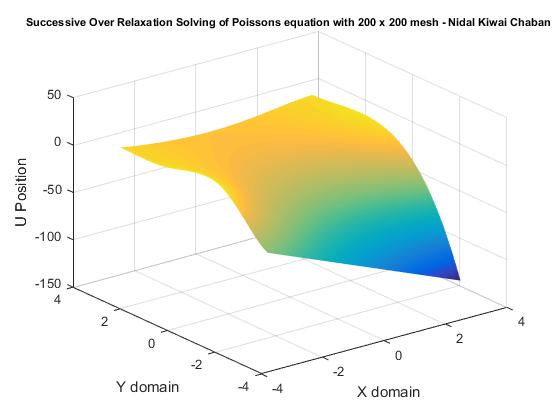


Figure 5-Surface Plot of 200 x 200 mesh of SOR

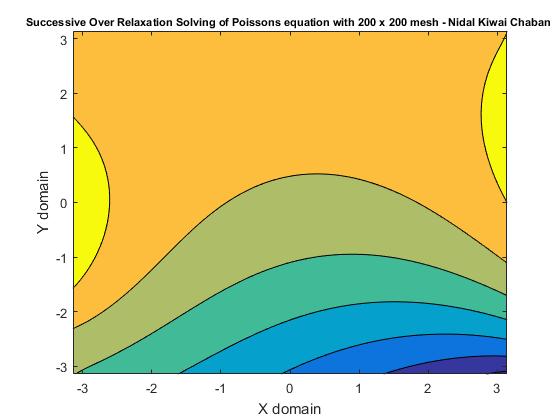


Figure 6-Contour Plot of 200x200 mesh for SOR

As it is shown, the finer the mesh is, the more accurate and the more iterations are required to come to convergence. This is due to the need for every node to come within tolerance to be considered a satisfactory simulation run.

## Verification

In order to verify that the approximations are correct, a comparison of the error between results is done with a fixed iteration method, this method will demonstrate that as iteration for specific size of meshes increase, the accuracy in results will increase as well. For this part, Successive Over-Relaxation method will be used since it converges faster and a wider range of iterations are able to be used.

|  |  |  |
| --- | --- | --- |
| Mesh Size: 30 | | |
| Iterations | **Elapsed Time** | **Error of u(i) and u(i-1)** |
| 10 | .007264 | 5.4605 |
| 100 | .013694 | 9.6395e-01 |
| 1000 | .085238 | 8.6302e-05 |
| 5000 | .405092 | -0.000 |

|  |  |  |
| --- | --- | --- |
| Mesh Size: 70 | | |
| Iterations | **Elapsed Time** | **Error of u(i) and u(i-1)** |
| 10 | .010761 | 2.2038e01 |
| 100 | .054540 | 3.9100e00 |
| 1000 | .487224 | 1.5703e01 |
| 5000 | 2.424058 | 3.9048e-04 |
| 10000 | 4.695324 | 7.4827e-07 |

|  |  |  |
| --- | --- | --- |
| Mesh Size: 200 | | |
| Iterations | **Elapsed Time** | **Error of u(i) and u(i-1)** |
| 100 | .402067 | 3.0773e01 |
| 1000 | 4.010405 | 2.6782e02 |
| 10000 | 40.235330 | 3.6238e-01 |
| 20000 | 79.6598 | 5.8127e-03 |

As expected, for each mesh size, as iteration number increases, the biggest error between u and the previous u value becomes smaller and smaller. Although having a coarser mesh make error smaller for larger numbers of iterations, it doesn’t represent the entire equation as good as having a larger mesh. The following graphs show 10000 iterations for a 20x20 mesh and a 60x60 mesh to show this discrepancy. For the first table, it shows that curves are steeper for smaller mesh sizes; while for the second table, color gradients are smoother for greater mesh sizes.

|  |  |
| --- | --- |
|  |  |
| 20x20 mesh | 60x60 mesh |

|  |  |
| --- | --- |
|  |  |
| 20x20 mesh | 60x60 mesh |