Project I

Instructions: This assignment may be completed in groups of up to 3 people. There are three methods to be coded. Although you may combine your code for the three methods into one m-file, each person in your team should take primary responsibility for coding up at least one of the three methods. If the other members of the team find errors in one person's code then it's fine to help that person get their code right, but each person should take primary responsibility for one method. Everyone is responsible for the write-up and for the explorations you choose to do in part (3). Your write-up along with the code should be submitted on gradescope by 11pm (Pacific Time) on Saturday October 31st. The matlab code should be in a state that it can be run without making any alterations. Your m-file should be fully commented, explaining what each piece of code does. All parameters (such as N and the parameter in your stopping criterion) should be at the top of the file where we can easily adjust them and rerun your code).

Problem to be Investigated

Suppose $g(x, y)$ is a known function of two variables. The Poisson equation for a function $u(x, y)$ is

$$\Delta u = g.$$

This equation arises in many different applications of physics including electricity, magnetism, fluid mechanics, gravity, heat, and soap films. We will consider the case where (x, y) is in the unit square $(0, 1) \times (0, 1)$ and u satisfies the boundary condition

$$u(x, 0) = u(x, 1) = u(0, y) = u(1, y) = 0, \qquad 0 \le x \le 1, \quad 0 \le y \le 1.$$

Let $\Omega = (1/5, 3/5) \times (1/4, 1/2)$ be a rectangle inside the unit square. We will consider the case when

$$g(x, y) = g_\Omega(x, y) = \begin{cases} 1 & (x, y) \in \Omega \\ 0 & \text{otherwise} \end{cases}$$

The purpose of this project is to investigate the performance of different iterative algorithms for estimating the solution u to this equation.

Numerical Formulation

Let N be a (large) integer and let $h = 1/(N + 1)$. Consider the grid of points

$$(x_{ij}, y_{ij}) = (ih, jh), \quad 0 \le i \le N + 1, \quad 0 \le j \le N + 1$$

in the unit square. Let u_{ij} be the value of u at the point (x_{ij}, y_{ij}). Recall, the Laplacian Δu is

$$\Delta u = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}.$$

We'll use a difference quotient to estimate the value of this at the point (x_{ij}, y_{ij}). We start by approximating the first order derivatives at the points in between the

grid points. These are

$$\begin{aligned}
\left.\frac{\partial u}{\partial x}\right|_{(x_{ij}-\frac{h}{2},y_{ij})} &\approx \frac{u_{i,j}-u_{i-1,j}}{h}\\
\left.\frac{\partial u}{\partial x}\right|_{(x_{ij}+\frac{h}{2},y_{ij})} &\approx \frac{u_{i+1,j}-u_{i,j}}{h}\\
\left.\frac{\partial u}{\partial y}\right|_{(x_{ij},y_{ij}-\frac{h}{2})} &\approx \frac{u_{i,j}-u_{i,j-1}}{h}\\
\left.\frac{\partial u}{\partial y}\right|_{(x_{ij},y_{ij}+\frac{h}{2})} &\approx \frac{u_{i,j+1}-u_{i,j}}{h}
\end{aligned}$$

Now we use the first order partials to approximate the second order partials at the grid points. We get:

$$\begin{aligned}
\left.\frac{\partial^2 u}{\partial x^2}\right|_{(x_{ij},y_{ij})} &\approx \frac{1}{h}\left(\left.\frac{\partial u}{\partial x}\right|_{(x_{ij}+\frac{h}{2},y_{ij})} - \left.\frac{\partial u}{\partial x}\right|_{(x_{ij}-\frac{h}{2},y_{ij})}\right)\\
&= \frac{u_{i+1,j}-2u_{i,j}+u_{i-1,j}}{h^2}.
\end{aligned}$$

Similarly

$$\left.\frac{\partial^2 u}{\partial y^2}\right|_{(x_{ij},y_{ij})} \approx \frac{u_{i,j+1}-2u_{i,j}+u_{i,j-1}}{h^2}.$$

Thus,

$$\Delta u(x_{ij},y_{ij}) \approx \frac{-4u_{i,j}+u_{i+1,j}+u_{i-1,j}+u_{i,j+1}+u_{i,j-1}}{h^2}.$$

With this approximation of the Laplacian, we expect the solution to Poisson's equation to be approximately equal to the solution to the following system of linear equations

$$\begin{aligned}
4u_{i,j}-u_{i+1,j}-u_{i-1,j}-u_{i,j+1}-u_{i,j-1} = b_{i,j} \quad 1\le i\le N, \quad 1\le j\le N,\\
u_{i,j}=0 \quad \text{otherwise}
\end{aligned}$$

where

$$b_{i,j} = -h^2 g(x_{ij},y_{ij}).$$

This is called the discretized Poisson equation. Let u_{ij}^N be the approximations for u_{ij} obtained by solving this system of equations.

Assignment

(1) By writing u_{ij}^N as a vector u^N, u^N is a solution to a system of equations of the form

$$A^N u^N = b^N.$$

Find the matrices A^N and vector b^N and write these down explicitly in the case when $N=3$. Notice, the matrix you get depends on how you write u_{ij}^N as a vector. Choose a way that will make it easy to code finding the product of A^N and a vector.

(2) Write a matlab script that generates the matrix A^N and vector b^N for any N and solves the system of equations using each of the following methods.
   - Jacobi method,
   - Gauss-Seidel with successive over-relaxation with any parameter ω, and
   - the conjugate gradient method.

   Be sure you code efficiently. utilizing the sparcity of A. Determine a stopping criterion and write your code so that the parameter associated with the stopping criterion can be easily adjusted.
(3) Experiment with different values of the relaxation parameter ω to find an optimal choice for the SOR method. Describe how you did this and how you decided what choice was optimal.
(4) Experiment with the three methods of solving the equations (using the optimal choice of ω). Use as large an N as you can. Produce pictures similar to Figure 7.5 on page 185 of the text but remember it is not just the *number* of iterations that is important but the efficiency of the algorithm, so you might also try plotting the residual norm against time. (The `tic` and `toc` Matlab commands may be helpful for this.)
(5) Use your best performing algorithm with a large value of N and plot your solution.
(6) For each method, tell me who it was in your group who took primary responsibility for coding it up. Also, briefly describe the contributions of each of you to the rest of the project.