# Homework 4: Relaxation

**Richard Ballantyne**[a,1]

[a]Dept. of Physics & Astronomy, Western Washington University, 516 High Street, Bellingham WA 98225; [1]E-mail: ballanr@wwu.edu

## 1. Introduction

The electric potential of two cocentric spheres, one inside the other, is a classic EM problem. One sphere is held at a voltage different than the other and usually the problem asks to solve for the potential at some distance away from the outer sphere. The purpose of this is to show that the electric potential drops as 1/r but never actually goes to 0 (as one cannot reach infinity). What then happens if we are interested in the interior region with a finite width between the the walls?

In this paper we investigate the electric potential of a charged plate ($V = 1$) in the middle of a square region ($V = 0$) and how varying the iterations, grid size, and initial conditions changes the precision/error of the numeric solution. We apply the Gauss-Seidel method to iteratively calculate a numeric solution. In Sec. 2 we describe the model and iterative Gauss-Seidel method. In Sec. 3 we discuss the influence that varying iterations, grid size, and initial conditions have on the precision/error and conclude in Sec. 4.

## 2. Model

**Laplace's Equation.** Laplace's equation is a second-order partial differential equation most known as the steady-state heat equation,

$$\nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} + \frac{\partial^2 f}{\partial z^2} = 0 \qquad [1]$$

where $\nabla^2$ (or $\Delta$ for the math community) is the Laplacian operator. However, this is not its only utility as the solutions to Laplace's equation are harmonic functions that are also particularly useful in describing the behavior of electric potentials,

$$\nabla^2 V = \frac{\partial^2 V}{\partial x^2} + \frac{\partial^2 V}{\partial y^2} + \frac{\partial^2 V}{\partial z^2} = 0 \qquad [2]$$

**Numeric Implementation.** Unlike ordinary differential equations, partial differential equations (PDEs) do not have a general algorithm like the Euler method or Runga-Kutta. Instead, each class of PDE has different algorithms specifically designed for them. An iterative approach that models the electric potential well is the relaxation method. The classic relaxation method is the Jacobi method which approximates the second derivatives in Eq. 2 as

$$\frac{\partial^2 V}{\partial x^2} \approx \frac{V_{i+1,j,k} + V_{i-1,j,k} - 2V_{i,j,k}}{(\Delta x)^2} \qquad [3]$$

and when inserted into Eq. 2 and solved for $V(i, j, k)$,

$$V_{i,j,k} = \frac{1}{6}\left[V_{i+1,j,k} + V_{i-1,j,k} + \cdots + V_{i,j,k-1} + V_{i,j,k+1}\right] \qquad [4]$$

The problem explored in this paper is two dimensional so Eqn. 4 becomes,

$$V_{i,j} = \frac{1}{4}\left[V_{i+1,j} + V_{i-1,j} + V_{i,j+1,} + V_{i,j-1}\right] \qquad [5]$$

This method iterates over each $V_{i,j}$ updating the values and placing them into a new array. This method is a little cumbersome as it requires two arrays for each pass and slows down the calculations.

In this paper a more efficient method (on the order of 2) called the Gauss-Seidel (GS) method is used. The GS method improves upon the Jacobi method as it updates values in place, propagating the change through one iteration of the entire grid. This removes the necessity to keep track of two arrays thereby reducing computation time.

The next section will discuss how the precision and errors of applying the GS method to an electric potential change with the number of iterations, grid size, and the initial conditions.

## 3. Results

**Iteration Variation.** In other numerical methods that involve time, we would investigate how changing the time-step, $\Delta t$, would change the precision in our numeric solution. However, in the GS method there is no time dependence but we can vary how many iterations over the grid there are. So, we change the number of iterations of the grid to see how the precision changes vs. the number of iterations.

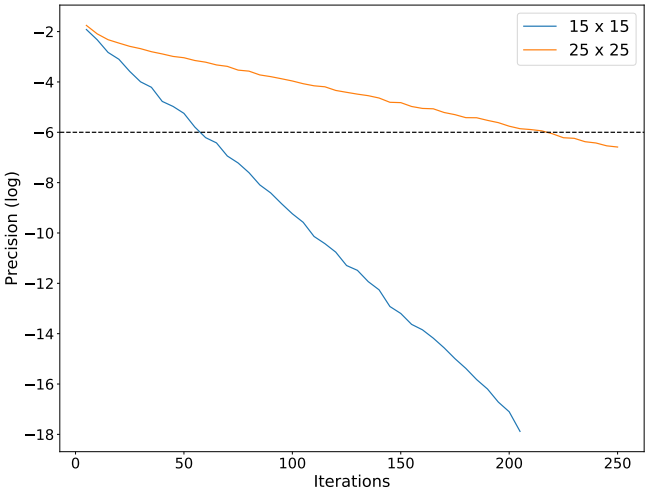


**Fig. 1.** Precision vs. Iteration testing. Two different grid sizes were tested to see how many iterations each took to get to a precision of $10^{-6}$. The $15\times15$ grid took $\sim$55 iterations while the $25\times25$ took $\sim$215.

Fig. 1 shows iterations vs. precision for two grids of size $15 \times 15$ (15-grid) and $25 \times 25$ (25-grid). For a precision of $10^{-6}$ (which will be the default precision threshold for this paper) we see that the 15-grid takes roughly 55 iterations to hit the desired precision while the 25-grid takes 215 iterations. We can see immediately that increasing the number of iterations

sharpens our precision at roughly a linear value and that it is also tied into the size of the grid in question.

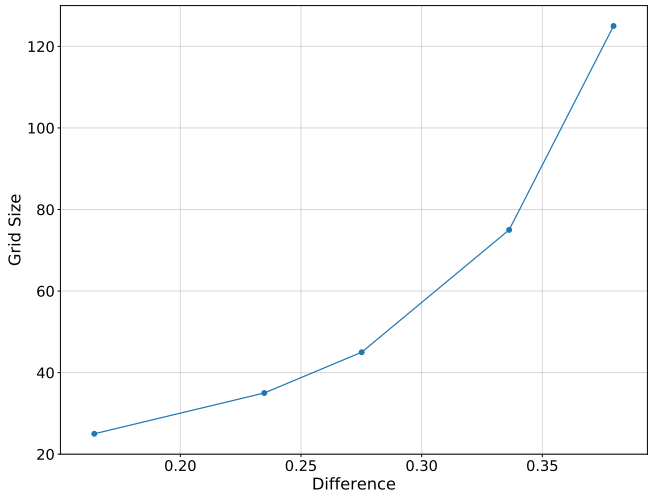


**Fig. 2.** Grid Size vs. Difference of interior plates after hitting the precision threshold of $10^{-6}$. Roughly follows a polynomial line of $0.5L^2$ with $L^2$ being the area of the grid.

**Grid Size Variation.** As seen above, the grid-size also affects the number of iterations required to reach the precision threshold. The 25-grid is 2.78 times larger (625 vs. 225) and also takes 3.91 times as many iterations (see Fig. 1) to reach the precision threshold than the 15-grid does. A rough estimate of the scaling would be $0.52L^2$ with $L^2$ being the area of the grid. This is very close to the expected value as the Jacobi method scales as $L^2$ and the GS is twice as fast so scales as $0.5L^2$.

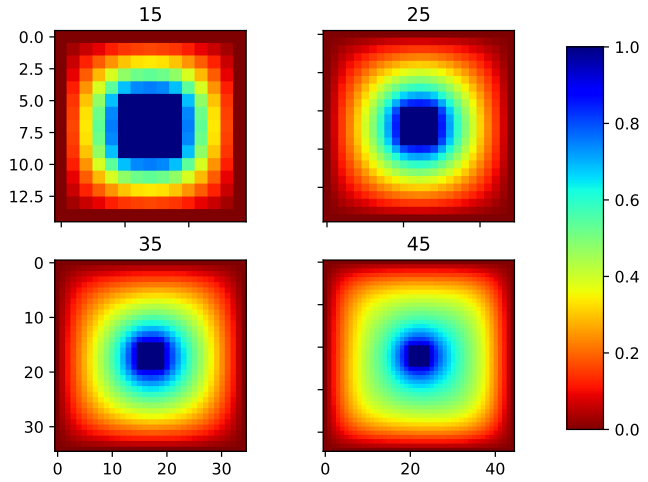


**Fig. 3.** Plots of grids 15, 25, 35, and 45 after 50 iterations. As the grid size gets larger each color region gets larger and more well defined.

To check how the grid-size affects the error we can take several grid sizes and compare how the middle $15 \times 15$ grid in each compares to our base 15-grid. In Fig. 2 is plotted the average difference in the center of five grid sizes: 25, 35, 45, 75, and 125. As the difference increases the grid size increases in a rough $x^2$ fashion. This means that as the grid size increases, the average value for the middle region is increasing as well. This adheres to our intuition as we know the potential falls off as 1/r (see Fig. 3). Because the boundaries are further away, the average value of the interior region of a larger grid will be larger than that of a smaller grid.

**Initial Conditions.** The GS routine used in this paper seeds the grid points that aren't a boundary condition or the middle plate to be a random value between 0 and 1. To test how the initial conditions of the grid affect the calculations, we can compare what happens for an array of random points, one of all zeros, and one of all ones. Fig. 4 shows that around a grid size of 35 is when the initial conditions start to visibly affect the number of iterations. The difference becomes important at a grid size of 75 as an initial condition of all ones takes about 500 more iterations to complete than all zeros or random values. At a grid size of 125 things are even more dramatic as random values now take several hundred more iterations than all zeros and all ones takes over a thousand more.

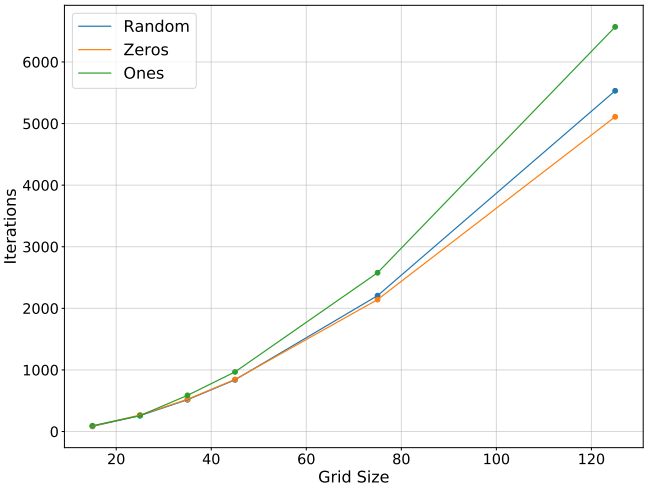


**Fig. 4.** Plot of the number of iterations vs. grid size for initial conditions of all random, all ones, and all zeros. As the grid size increases picking zero as the initial condition leads to least number of iterations.

## 4. Conclusion

For partial differential equations there is no one-size-fits-all numeric approach. For many potential equations, the relaxation method (here the GS method) works to approximate a numeric solution.

For this method there are three parameters that affect the precision and computational time required to achieve said precision: grid size, number of iterations, and initial conditions. To obtain a given precision a combination of the three parameters needs to be selected. However, because the number of iterations is a direct consequence of the initial conditions and the grid size (scaling roughly $0.5L^2$) so picking the grid size is the first priority. After that picking your initial conditions is straight-forward as setting them to be zeros takes fewer iterations to relax to the precision threshold than either random values or all ones.