

Solving Laplace Equation Using Finite Difference Schemes and Iterative Methods

Author: Kane Anthony Norman (Undergraduate at Southern Methodist University ,
Dallas, TX: Math Department)

Course: MATH 4315, Dr. Geng

Date: May 7th, 2019

2. Project Description: We are given a two dimensional Laplace equation with Dirichlet boundary conditions. We will find an approximate solution using finite difference schemes and iterative methods.

In order to find an approximate solution we must follow three steps

1. Discretize the Laplace equation using finite difference schemes.
2. Solve the discretized form using Jacobian, Gauss-Seidel, and SOR iterative methods
3. Compare results. Observe how changes in h affect speed and accuracy.

Compare the convergence speed of the three iterative methods

3. Concepts and Theories:

I) Partial Differential Equation and Parameters:

Consider a square metal plate. The temperature of the plate $\phi(x, y)$ must satisfy the partial differential equation $\phi_{xx} + \phi_{yy} = 0$. We can consider the bottom edge of the plate to be our x-axis, where $0 \leq x \leq 1$. Also, we can consider the y-axis to be the left edge of the plate, where $0 \leq y \leq 1$.

II) Boundary Conditions:

Since this is a 2D boundary value problem, we will need 4 boundary conditions to guarantee a unique solution. The temperatures along the boundary of the plate satisfy Dirichlet boundary conditions. The temperature for top edge of the plate is given as $\phi(x, 1) = e^\pi \sin(\pi x)$. And the temperatures for the bottom, left, and right edges are given as

$$\phi(x, 0) = \sin(\pi x), \phi(0, y) = \phi(1, y) = 0 \text{ respectively.}$$

III) Computing Analytical Solution:

This is a 2D boundary value problem that can be solved using the eigenfunction expansion method; however, this method can be extremely difficult. Instead we will invoke an analytical approach.

In order to find an analytical solution, we must discretize the partial differential equation and use iterative methods.

IV) Discretization and Error

Recall $D_+y_i = \frac{y_{i+1} - y_i}{h}$ and $D_-y_i = \frac{y_i - y_{i-1}}{h}$. Thus : $D_+D_-y_i = \frac{y_{i+1} - 2y_i + y_{i-1}}{h^2} \approx y''(x)$. Notice this is an approximation since this discretization introduces some small error. Using Taylor's expansion we can see

$$y_{i+1} = y_i + hy_i' + \frac{h^2}{2}y_i'' + \frac{h^3}{3!}y_i''' + \frac{h^4}{4!}y_i^{(4)} + \frac{h^5}{5!}y_i'''' + O(h^6)$$

$$y_{i-1} = y_i - hy_i' + \frac{h^2}{2}y_i'' - \frac{h^3}{3!}y_i''' + \frac{h^4}{4!}y^{(4)} - \frac{h^5}{5!}y_i'' + O(h^6)$$

Thus,

$D_+D_-y_i = \frac{y_{i+1}-2y_i+y_{i-1}}{h^2} = y''(x) + \frac{h^2}{12}y^{(4)} + O(h^4)$. This means the approximation is second order accurate.

Using the previous information we can discretize $\phi_{xx} + \phi_{yy} = 0$ using finite difference methods. $\phi_{xx} + \phi_{yy} = 0$ can be approximated by $(D_+^x D_-^x + D_+^y D_-^y)w_{ij} = f_{ij} \Rightarrow$

$$\left(\frac{w_{i+1,j} - 2w_{ij} + w_{i-1,j}}{h^2} + \frac{w_{i,j+1} - 2w_{ij} + w_{i,j-1}}{h^2}\right) = f_{ij} \Rightarrow \frac{1}{h^2}(w_{i+1,j} - 2w_{ij} + w_{i-1,j} + w_{i,j+1} - 2w_{ij} + w_{i,j-1}) = f_{ij} \Rightarrow$$

$$\frac{1}{h^2}(-4w_{ij} + w_{i+1,j} + w_{i-1,j} + w_{i,j+1} + w_{i,j-1}) = f_{ij}$$
. This discretized form gives a linear system of equations $\frac{1}{h^2}Aw = f$. We can solve for w using iterative methods (.ie Jacobi, Gauss-Seidel, SOR). Our final result w will be a matrix, where $w_{ij} \approx \phi(x, y) = \sin(\pi x)e^{\pi y}$, $\{x = ih, y = jh\}$. This w is our analytical solution.

4.Numerical Implementation:

I) Discretized forms of Jacobi, Gauss-Seidel, and SOR

The following are the discretized forms of $\phi_{xx} + \phi_{yy} = 0$ for Jacobi, Gauss-Seidel, and SOR.

- Jacobi

$$\frac{1}{h^2}(-4w_{ij}^{(k+1)} + w_{i+1,j}^{(k)} + w_{i-1,j}^{(k)} + w_{i,j+1}^{(k)} + w_{i,j-1}^{(k)}) = f_{ij}$$

- Gauss-Seidel

$$\frac{1}{h^2}(-4w_{ij}^{(k+1)} + w_{i+1,j}^{(k)} + w_{i-1,j}^{(k+1)} + w_{i,j+1}^{(k)} + w_{i,j-1}^{(k+1)}) = f_{ij}$$

- SOR

$$w_{ij}^{(k+1)} = w_{ij}^{(k)} + \frac{1}{4}w^*(4w_{ij}^{(k)} - w_{i+1,j}^{(k)} - w_{i-1,j}^{(k+1)} - w_{i,j+1}^{(k)} - w_{i,j-1}^{(k+1)} - f_{ij}h^2)$$

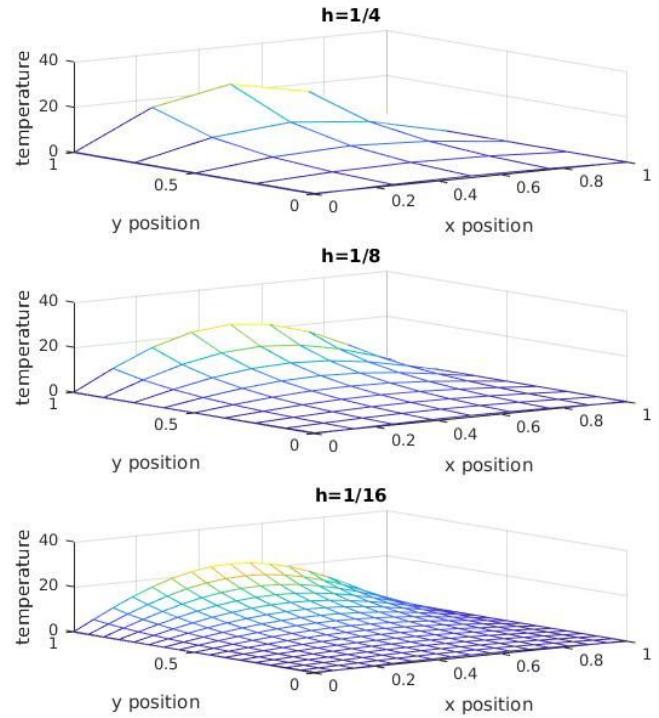
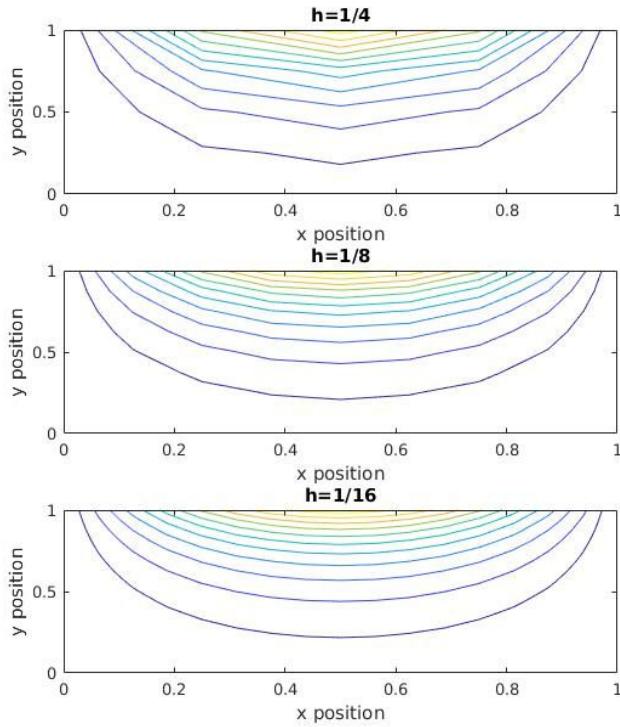
II) Comparison of the three iterative methods

Notice: For SOR and Gauss-Seidel, at iteration k+1 these two methods use the k+1 approximation of $w_{i-1,j}$ and $w_{i,j-1}$. For Jacobi, at iteration k+1 this method uses the k approximation of $w_{i-1,j}$ and $w_{i,j-1}$. This means for Jacobi we must store two versions of w , but for SOR and Gauss-Seidel only need to store one version of w . This makes SOR and Gauss-Seidel easier to compute.

5.Numerical Results and Implementations:

I. Dependence Upon h

In this project we use three different values of h , $\frac{1}{4}, \frac{1}{8}, \frac{1}{16}$. Below are contour and mesh plots



from using the Jacobi Method:

Notice that as h decreases, the plots become much smoother. This is an indication that the error of our approximation is decreasing as we decrease our value h . This is also evident in the following table:

Table 1: Error, Ratio, and Order Dependent Upon h

h	error	ratio	order
$\frac{1}{4}$	0.390641	-	-
$\frac{1}{8}$	0.101898	3.8336	1.938710
$\frac{1}{16}$	0.026246	3.3324	1.956950

Once again we can see that the error is decreasing as we decrease the value of h .

Decreasing the value of h helps reduce the error; however we must note that decreasing the value of h will also require more iterations. This is evident from the following table:

Table 2: Required Iterations Dependent Upon h

h	Jacobi	Gauss-Seidel	SOR
$\frac{1}{4}$	52	28	15
$\frac{1}{8}$	205	110	30
$\frac{1}{16}$	762	409	54

This observation can also be explained mathematically by evaluating $\rho(B)$ for each method.
 $\rho(B_J) = \cos(\pi h) \sim 1 - \frac{1}{2}\pi^2 h^2$

$$\rho(B_{GS}) = \cos^2(\pi h) \sim 1 - \pi^2 h^2$$

$$\rho(B_{w_*}) = \frac{2}{1 + \sqrt{1 - \rho(B_J)^2}} - 1 \sim 1 - 2\pi h$$

Notice that $\rho(B) \rightarrow 1$ as $h \rightarrow 0$. Thus as h decreases, the asymptotic rate of convergence will approach 1. If the asymptotic rate of converge is approximately one, this means we are converging very slowly. To summarize, we can conclude that using a large h will increase speed and decrease accuracy. Contrarily, using a small h will reduce speed and improve accuracy.

II. Convergence Speed of the Three Methods:

Refer to Table 2: *Required Iterations Dependent Upon h*. Notice that for any value h , SOR converges faster, then Gauss-Seidel, then Jacobi. This phenomenon can be explained by evaluating $\rho(B)$ for each method.

$$\rho(B_J) = \cos(\pi h) \sim 1 - \frac{1}{2}\pi^2 h^2$$

$$\rho(B_{GS}) = \cos^2(\pi h) \sim 1 - \pi^2 h^2$$

$$\rho(B_{w_*}) = \frac{2}{1 + \sqrt{1 - \rho(B_J)^2}} - 1 \sim 1 - 2\pi h$$

We can see that $\rho(B_{w_*}) < \rho(B_{GS}) < \rho(B_J)$, thus we have confirmed SOR converges fastest, followed by Gauss-Seidel, then Jacobi.

7. Concluding Remarks:

In this project we solved a two dimensional boundary value problem using finite difference schemes and iterative methods. We compared the convergence speed and error for different values of h . We found that as $h \rightarrow 0$ the error is reduced, but we compromise speed. We also observed that SOR is the fastest of the 3 iterative methods.