Numerical Solution Methods for Poisson's Equation

Nagaprasad Rudrapatna

May 1, 2021

1 Introduction

In this report, we discuss numerical solution methods to solve Poisson's Equation in two dimensions. Poisson's Equation (PE) is a second-order linear partial differential equation (PDE) of the following form,

$$\nabla^2 \phi = S, \tag{1}$$

where ∇^2 is the Laplacian operator (in a Cartesian coordinate system). Given that the Laplacian is the sum of second partial derivatives of the function with respect to each independent variable, an equivalent formulation of Equation 1 is:

$$\frac{\partial^2 \phi}{\partial x^2} + \frac{\partial^2 \phi}{\partial y^2} = S. \tag{2}$$

Interestingly, this elliptic PDE can be adapted to solve fundamental problems in fluid dynamics, such as determining the pressure field in incompressible Navier-Stokes equations, and electrostatics, such as calculating the electric potential for a given charge distribution [1, 3]. Equation 1 is defined for both real and complex-valued functions ϕ and S on Euclidean space. We consider in particular Poisson's Equation for Gravity (PE-G),

$$\nabla^2 \phi = 4\pi G \rho(x, y), \tag{3}$$

where ϕ represents the unknown gravitational potential, G represents the gravitational constant[1], and ρ denotes the mass density as a function of two spatial variables.

Accordingly, the objective is to solve the discretized form of Equation 3 in a finite domain, i.e. $[-L, L] \times [-L, L]$, with Dirichlet boundary conditions and $\rho = \exp(-x^2 - y^2)$. The mass density is selected due to its rapid decaying behavior. As a consequence of this property, we expect L to be reasonably sized−limiting computational cost. The solution to this equation should describe the work per unit mass required to move an object (with mass density ρ) to a specific location from an initial position (assumed to be $(0, 0)$). Dirichlet boundary conditions (listed in Table 1) are imposed due to the physical nature of the problem. Explicitly, from Newton's laws, it is clear that the gravitational potential decreases as distance from the origin increases. So, as this distance approaches infinity, the gravitational potential approaches zero. Of course, this theoretical behavior cannot be exactly replicated computationally. Rather, we assume that the edges of the simulated region can be viewed as arbitrarily large. Carefully tuning the dimension L then becomes an important computational exercise (addressed in Section 3.2).

[1]The gravitational constant can be measured precisely to four significant digits: $6.674 \times 10^{-11}\ m^3 \cdot kg^{-1} \cdot s^{-2}$.

(x,y)	$\phi(x,y)$
(L,y)	0
$(-L,y)$	0
(x,L)	0
$(x,-L)$	0

Table 1: Boundary Conditions For The Discretized Form of Equation 3

2 Numerical Solution Methods

Researchers have proposed various numerical solution schemes for PE (and elliptic PDEs, in general), including finite element, multi-grid, and boundary integral equation methods. In this report, we utilize finite difference methods. In this approach, PE can be solved by the following procedure:

1. Convert the PDE into a linear system of $N-2$ equations[2] with $N-2$ unknowns—where N represents the number of x or y-points in a uniform grid—by applying a second-order centered difference formula, as shown below

$$\frac{\phi_{i+1,j}-2\phi_{i,j}+\phi_{i-1,j}}{\Delta x^2}+\frac{\phi_{i,j+1}-2\phi_{i,j}+\phi_{i,j-1}}{\Delta y^2}=S_{i,j},\ i,j\in[2,\ldots,N-1] \quad (4)$$

2. Simplify the discretized form, displayed below, with the assumption $h=\Delta x=\Delta y$

$$\phi_{i+1,j}+\phi_{i-1,j}+\phi_{i,j+1}+\phi_{i,j-1}-4\phi_{i,j}=h^2S_{i,j},\ i,j\in[2,\ldots,N-1] \quad (5)$$

$$\phi_{i,j}=\frac{1}{4}[\phi_{i+1,j}+\phi_{i-1,j}+\phi_{i,j+1}+\phi_{i,j-1}-h^2S_{i,j}],\ i,j\in[2,\ldots,N-1] \quad (6)$$

3. Solve Equation 6 using an iterative method

As one might expect, there are many iterative schemes to choose from; in this analysis, a two-dimensional successive over-relaxation (SOR) algorithm is applied to Equation 6. Accordingly, the following equation provides a succinct expression for the SOR update at the $(n+1)$-th step:

$$\phi^{n+1}_{i,j}=\frac{\beta}{4}[\phi^{n}_{i+1,j}+\phi^{n+1}_{i-1,j}+\phi^{n}_{i,j+1}+\phi^{n+1}_{i,j-1}-h^2S_{i,j}]+(1-\beta)\phi^{n}_{i,j},\ i,j\in[2,\ldots,N-1]. \quad (7)$$

It is well-known that each component of the new guess (ϕ^{n+1}) in SOR is a weighted average of the relaxation parameter, β, times the Gauss–Seidel (GS) formula and $(1-\beta)$ times the current guess (ϕ^n) [2]. By selecting $\beta=1$, the SOR formula reduces to the familiar GS form. This means the performance—in terms of rate of convergence and computational efficiency—of two iterative methods can be evaluated with the same *MATLAB* program.

However, it is easier to compare the performance of the two methods in the case where the exact solution of the PDE is known (as opposed to the discretized version of PE-G). In other words, to test the methods, it is beneficial to first analyze a (test)

[2]The number $N-2$ can be determined by calculating the length of the interval in Equation 4: $(N-1)-2+1=N-2$.

problem with a known solution. Consider PE for which the solution, ϕ, is $\exp(x^2 + y^2)$. Then, the right-hand side of Equation 1 is given by the Laplacian of ϕ:

$$\frac{\partial^2}{\partial x^2}[\exp(x^2 + y^2)] + \frac{\partial^2}{\partial y^2}[\exp(x^2 + y^2)] = \exp(x^2)[4x^2 \exp(y^2) + (4y^2 + 4)\exp(y^2)]. \quad (8)$$

The domain for the test problem is chosen as $[0, L] \times [0, L]$ for convenience.[3] Based on this choice of domain, the boundary conditions for the test problem (listed in Table 2) are different from those specified earlier for PE-G. Explicitly, these conditions can be determined by evaluating the known function ϕ along each edge of the boundary.

(x, y)	$\phi(x, y)$
$(0, y)$	$\exp(y^2)$
(L, y)	$\exp(L^2 + y^2)$
$(x, 0)$	$\exp(x^2)$
(x, L)	$\exp(x^2 + L^2)$

Table 2: Boundary Conditions For The Test Problem

While examining this test problem, the questions of interest are:

1. What is the optimal (in terms of the computational efficiency) relaxation parameter, $\beta_{optimal}$, in Equation 7?
2. How do the rates of convergence of GS and SOR compare?
3. What is the order of convergence?

3 Discussion of Results

3.1 Test Problem

3.1.1 Optimal Relaxation Parameter

Before addressing Question 1, we must first describe the relaxation parameter, β, in more detail. Technically, $0 < \beta < 2$. But when $0 < \beta < 1$, the method is often called successive under-relaxation (SUR). Whereas SUR takes the GS direction toward the solution and undershoots to try to increase stability, SOR takes the GS direction toward the solution and overshoots to try to speed convergence [2]. As convergence is a critical component of this report (while stability is not), we will restrict the range of β to $[1, 2)$.

Next, we will clarify what we mean by "computational efficiency". In this report, the metric used to gauge efficiency is the number of iterations. While it was our intention to also provide runtime comparisons (e.g. elapsed times in *MATLAB* using tic and toc), we ultimately decided that including these notoriously unreliable and non-reproducible metrics (i.e. without having access to our device, replicating the conditions needed to obtain similar elapsed times is effectively impossible) would only discredit our findings. However, the number of iterations was not the only consideration in selecting the optimal

[3] $[-L, L] \times [-L, L]$ could also be used here. It should be noted, however, that the reverse is not true, i.e. the domain used in the test problem is not applicable to PE-G due to physical constraints, i.e. the initial position of the object is fixed at the origin.

relaxation parameter, $\beta_{optimal}$. Two other factors were the magnitude of N, which controls the dimensions of the simulated region, and the error tolerance level.

Numerical experiments show that, given an "adequately" small tolerance level, the largest grid that can be generated (assuming the maximum number of iterations is 150000) is: $[257, 257] \times [257, 257]$. This is an important result because it highlights a subtle problem which can arise in SOR. If the tolerance level is too large, then the error in SOR can impact the order of convergence. So, the tolerance level must be sufficiently small to ensure that the error in SOR has negligible effect. As a result of this phenomenon, the error tolerance level decreased by a hundred, from 10^{-8} to 10^{-10} for this test problem.

There is another source of concern in regard to the magnitude of the tolerance level; if the tolerance is too small, then the algorithm will again fail to converge (due to the accumulation of rounding errors). In the case of the test problem, the minimum tolerance level (before rounding errors become significant) was found to be 10^{-12}. We addressed this trade-off in the size of the error tolerance level by selecting 10^{-10}. This tolerance level remained consistent during relaxation parameter experiments.

Having resolved any issues with tolerance levels, we are ready to begin testing various relaxation parameters. Instead of testing β values randomly, we can determine the optimal value by first identifying β values which could be used to construct a $[257, 257] \times [257, 257]$ uniform grid and then comparing the number of iterations required to achieve the task. The following table summarizes the experimental results.

β	Iterations: $[129, 129] \times [129, 129]$	Iterations: $[257, 257] \times [257, 257]$
1	27811	102003
1.3	21727	79803
1.4	20262	74454
1.45	19603	72047
1.5	18987	69785
1.51	18869	69364
1.52	18752	> 150000

Table 3: The Computational Cost of Different Relaxation Parameters

It is evident that $\beta_{optimal} = 1.51$ based on the second column of Table 3. Notice that SOR with $\beta = 1.51$ is only slightly more efficient than SOR $\beta = 1.5$. However, this improvement comes with essentially no disadvantages (as we have already taken care of any error tolerance-related issues). There may however be an argument to using $\beta = 1.52$. If the problem of interest does not require a large grid (i.e. smaller than $[257, 257] \times [257, 257]$) to achieve the intended level of accuracy, then selecting $\beta = 1.52$ may be justified due to the sheer difference in iteration counts. Explicitly, SOR with $\beta = 1.51$ requires approximately 3.7 times as many iterations as SOR with $\beta = 1.52$, albeit with a higher level of accuracy (comparing across rows of Table 3). For our problem of interest (PE-G), SOR with $\beta = 1.51$ is more appropriate because the dimension L may be significantly larger (and our simulated region should be at least as large as $[-L, L] \times [-L, L]$). It is worthwhile to mention that GS performs quite poorly in terms of computational efficiency, e.g. GS requires more than 100000 iterations to solve the test problem on a $[257, 257] \times [257, 257]$ uniform grid.

3.1.2 Rates of Convergence

Now that we have identified $\beta_{optimal}$, we can compare the rates of convergence of GS and SOR with $\beta_{optimal}$. Recall that, provided the rate is less than one, the iterative method converges. While examining the estimated spectral radii (using the estimation formula $\frac{r_{n+1}-r_n}{r_n-r_{n-1}}$), we notice an alternating pattern. The presence of an alternating pattern in the rate of convergence implies that convergence is not exactly linear. Rather, the methods converge at least linearly (i.e. error in the sequence of iterates is less than or equal to Cr^n) with rates r_{GS} and r_{SOR} respectively. We calculate $r_{GS} = \sqrt{0.999851269 \times 0.99984903} = 0.999850$ and $r_{SOR} = \sqrt{1.0004911 \times 0.9990365} = 0.999764$. Note that, as expected (based on Table 3), the rate of convergence of SOR with $\beta_{optimal}$ is smaller than the rate of convergence of GS.

3.1.3 Order of Convergence

In the previous section, we showed that GS and SOR with $\beta_{optimal}$ converge at least linearly. Moreover, we previously discussed how the error tolerance level was chosen so that the error in SOR would have a negligible effect on the order of convergence (i.e. convergence plot is reliable). Here, we will elaborate on the order of convergence. Theoretically, we expect the order of convergence to be $O(h^2)$. This is because the total error is roughly $O(h^2)$ when you consider the contributions from the second-order centered difference formula and the propagated error. We tested this hypothesis in the test problem because we can look at the absolute error in our approximate solution (since the exact solution is known). Figure 1 illustrates how the absolute error varies as a function of step size. Consider the two points in Figure 1. Let $(x_1, y_1) = (0.03125, 0.000806357)$ and $(x_2, y_2) = (0.0625, 0.00319872)$. Notice that $\frac{x_1}{x_2} = \frac{1}{2}$ and $\frac{y_1}{y_2} = 0.252087$. This means that decreasing the step size by a factor of two roughly leads to a quartering of the worst-case error at each step. In other words, the worst-case error is approximately divided by four at each step. The implication is that SOR (and GS[4]) is second-order accurate. Thus, our hypothesis is correct and the order of convergence is roughly $O(h^2)$.

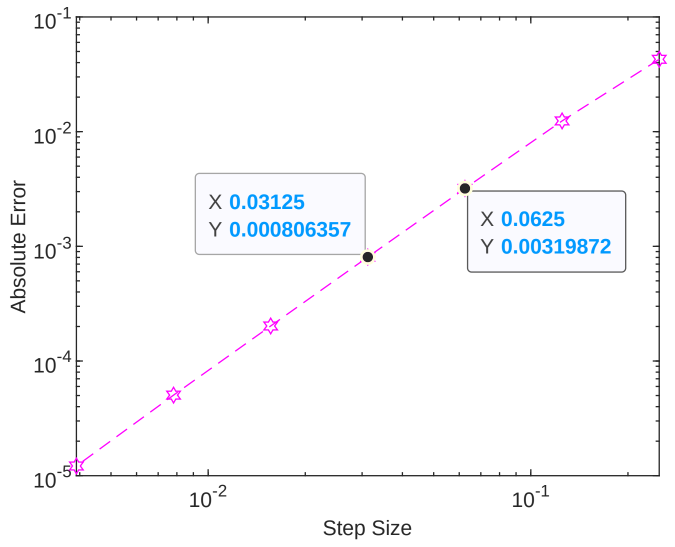


Figure 1: The absolute error in the approximate solution generated by SOR with $\beta_{optimal}$ is plotted against step size on a log-log plot.

[4]The convergence plot for GS is essentially identical.

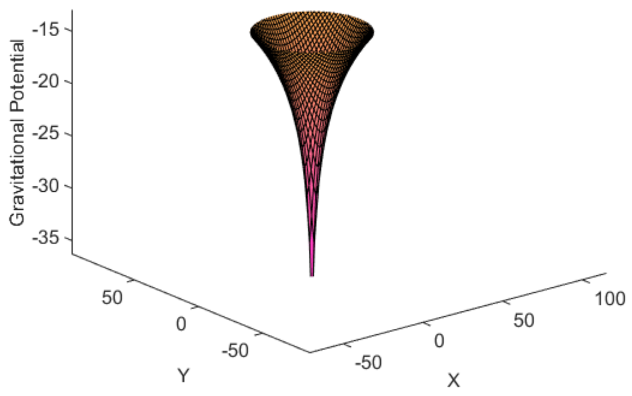
3.2 Application: Poisson's Equation for Gravity

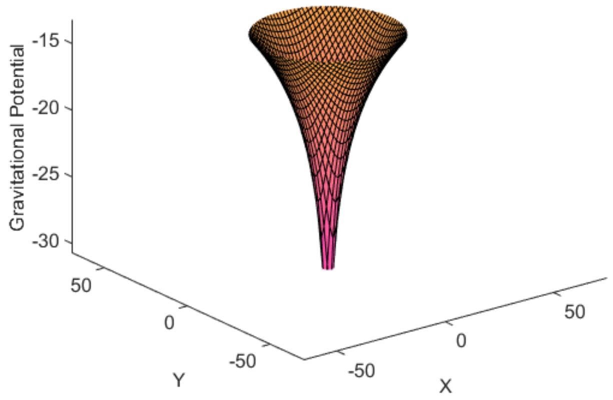
We have concluded our exploration of the test problem. Now, we will consider PE-G. We begin by revisiting the issue of error tolerance levels. Based on our numerical experiments, it seems that PE-G requires a larger tolerance level than the test problem. Specifically, we found that tolerance levels below 10^{-10} were too small (i.e. did not converge due to an accumulation of rounding errors). Furthermore, we learned that tolerance levels above 10^{-9} were too large (i.e. error in SOR significantly impacted order of convergence). So, we decided to use an error tolerance level of 10^{-9}.

As mentioned before, we tuned the dimension L according to convergence. The basic idea is: L should be large enough to approximate an infinite "box". To find the optimal dimension, we performed numerical experiments. We localized the optimal dimension to the interval $(150, 200)$ in our initial tests. Our experiments showed that $L = 200$ is sufficiently large since the order of convergence was approximately $O(h^2)$. On the other hand, with $L = 150$, the order of convergence was far away from $O(h^2)$. We further localized the optimal dimension to the interval $(190, 200)$ after subsequent tests. From there, we tested each integer L starting at 190 and found that the order of convergence moved closer and closer to $O(h^2)$ until $L = 196$. We concluded that the optimal dimension is 196 as the order of convergence most closely aligns with $O(h^2)$, i.e. decreasing the step size by a factor of two leads to a quartering of the worst-case error at each step.

Additionally, we examined the rates of convergence of GS and SOR with $\beta_{optimal}$. As in the test problem, there is evidence that GS and SOR each converge at least linearly with rates of $r_{GS} = 0.999851$ and $r_{SOR} = 0.999769$, respectively. Notice that these rates are very similar to those in the test problem. The rate of convergence of SOR with $\beta_{optimal}$ is smaller than the rate of convergence of GS.

Finally, we wanted to qualitatively assess how the approximate solutions (generated by GS and SOR with $\beta_{optimal}$) to PE-G in a finite setting match the physical behavior we expect. We expect the gravitational potential to decay as distance from the origin increases; thus, we expect to see behavior like $\frac{1}{(x^2+y^2)^{p/2}}$ for some integer p. Examining the surface plots in Figure 2, we see that our approximate solutions do not qualitatively agree with the expected behavior (the bottom plot displays $\frac{1}{(x^2+y^2)^{p/2}}$ for $p = 4$; however, the approximate solutions do not match for any p). This makes sense since the approximate solutions took $G = 1$. If the surface plots are all converted to the same scaling, they should agree.





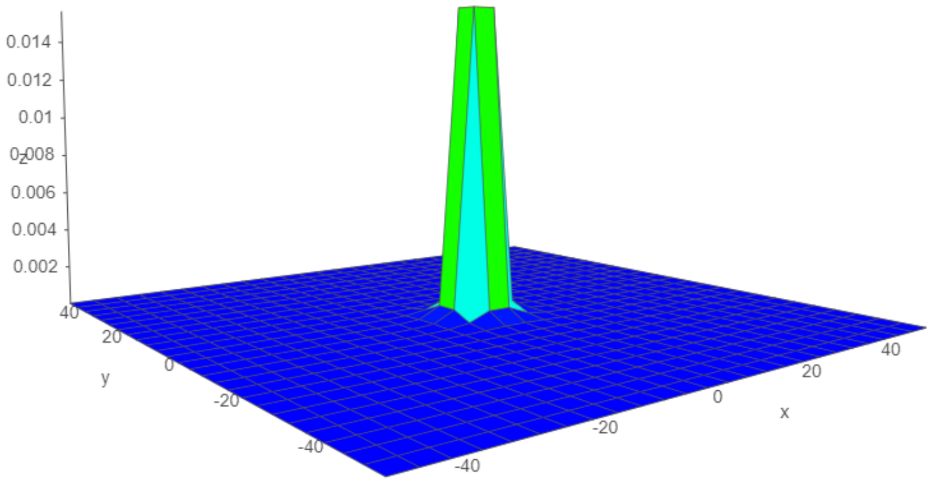


Figure 2: The solutions to the discretized form of PE-G are visualized in the top and middle surface plots. The optimal dimension $L = 196$ is utilized. Top: The solution generated by GS is displayed. Middle: The solution generated by SOR with $\beta_{optimal} = 1.51$ is displayed. Bottom: The expected behavior is displayed: $\frac{1}{(x^2+y^2)^2}$.

4 Conclusion

In this report, we determined the optimal relaxation parameter ($\beta_{optimal} = 1.51$) in the SOR update by analyzing the behavior of a test problem. Furthermore, we learned that decreasing the step size by a factor of two roughly leads to a quartering of the worst-case error at each step (in the test problem). This implies that the SOR algorithm is second-order accurate and that the order of convergence is roughly $O(h^2)$. We found that the order of convergence also applies in the case of Poisson's Equation for Gravity in a finite setting.

References

[1] *LaPlace's and Poisson's Equations.* URL: `http://hyperphysics.phy-astr.gsu.edu/hbase/electric/laplace.html` (visited on 04/28/2021).

[2] Timothy Sauer. *Numerical analysis.* Pearson Education, Inc., 2012, pp. 128–129.

[3] Benjamin Seibold. *A compact and fast Matlab code solving the incompressible Navier-Stokes equations on rectangular domains.* 2008. URL: `http://math.mit.edu/~gs/cse/codes/mit18086_navierstokes.pdf` (visited on 04/28/2021).