MONTE CARLO SOLUTION TO LAPLACE'S EQUATION ON A RECTANGULAR DOMAIN

MATT CASSINI, MARISSAH MCNEIL, AND MOISES RAMOS

Abstract. We use the Tour Du Wino Method, an algorithm based on Monte-Carlo simulation, to approximate a solution to Laplace's Equation on a rectangular domain with Dirichlet boundary conditions

1. Introduction

We seek to solve Laplace's Equation

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = 0 \tag{1}$$

on a rectangular domain $0 \leq x \leq 7, \quad 0 \leq y \leq 9$ with boundary conditions $u(0, y) = u(7, y) = u(x, 0) = 0$ and $u(x, 9) = 12$

2. Background

In electrostatics the following relationship is known where E represents an electric field, V the electric potential, ρ the charge density and ϵ the permittivity of the material.

$$E = -\nabla V$$

it is also notable that the electric field has the following properties physically

$$\nabla E = \frac{\rho}{\epsilon}$$

if we do the following to our original equation we see an interesting result

$$\nabla E = -\nabla \nabla V$$

which means

$$\nabla^2 V = -\frac{\rho}{\epsilon}$$

If the charge density is zero or in other words if we are in a charge free region or space we arrive at the problem we are interested in Laplace's Equation

$$\nabla^2 V = 0$$

Date: January 8, 2023.
Math 450H Final Project.

3. Implementation

We programmed a Monte-Carlo simulation using the Tour Du Wino method that utilizes a 2D random walker on a uniform Cartesian grid that records the boundary point on which it lands. It then repeats this process m times starting at this same point. It then averages the value of the boundary condition at each point recorded and assigns this to the value of the solution at the given point.[1]

The proof of this is very straightforward. The expected value for some point a where R(a) is the value it will take is

$E[R(a)] = \frac{1}{4}(E[R(b)] + E[R(c)] + E[R(d)] + E[R(e)])$

where b,c,d, and e are the neighboring points.

This can be rewritten as

$$u_{ij} = \frac{1}{4}(u_{i+1,j} + u_{i-1,j} + u_{i,j+1} + u_{i,j-1}) \tag{2}$$

which can be rearranged as

$$u_{i+1,j} + u_{i-1,j} + u_{i,j+1} + u_{i,j-1} - 4u_{ij} = 0$$

The centered finite difference approximations for the second partial derivatives are

$$u_{xx} = \frac{u_{i+1,j} + u_{i-1,j} - 2u_{i,j}}{h^2} \qquad u_{yy} = \frac{u_{i,j+1} + u_{i,j-1} - u_{ij}}{h^2}$$

It is now clear the above formulation can be expressed as $h^2 u_{xx} + h^2 u_{yy} = 0$ and dividing away h^2 leaves us $u_{xx} + u_{yy} = 0$ which is then Laplace's Eq.

Our initial algorithm was

Algorithm 1 Original Tour Du Wino

```
Initialize pos1 and pos2 as the first interior coordinate pair
Generate random number r =∼ U(0, 1)
if r ≤ 0.25 then
    pos1 ← pos1 + 1
else if r ≤ 0.5 then
    pos1 ← pos1 − 1
else if r ≤ 0.75 then
    pos2 ← pos2 + 1
else
    pos2 ← pos2 − 1
end if
if pos1 or pos2 == 1 or n then
    Save and continue to next starting point
else
    Repeat
end if
Repeat for all starting points
```

This was incredibly slow so we parallelized the algorithm and have Algorithm 2

Algorithm 2 Improved Tour Du Wino

Initalize $pos1$ and $pos2$ as all the coordinate pairs in $2 : n-1$, repeated m times
Create 2 $1 \times n^2m$ arrays, r, to store coordinates for each walker
Generate $1 \times n^2m$ array $\sim U(0,1)$
if $r_i \leq 0.25$ **then**
    $pos1_i \leftarrow pos1_i + 1$
else if $r_i \leq 0.5$ **then**
    $pos1_i \leftarrow pos1_i - 1$
else if $r_i \leq 0.75$ **then**
    $pos2_i \leftarrow pos2_i + 1$
else
    $pos2_i \leftarrow pos2_i - 1$
end if
if $pos1_i$ or $pos2_i == (1 \text{ or } n)$ **then**
    Remove $pos1_i$ and $pos2_i$ from list
    Decrease length of random number array
end if
Repeat until the array is empty

This algorithm has improved efficiency as it can take advantage of parallel computing while immediately deleting all "completed" walkers from the arrays so that it doesn't waste time.

Our Matlab code is submitted as a separate document to avoid cluttering the report.

4. Analytical Solution

If possible, before using a numerical solution to your problem it is best to have an idea of what your solution is supposed to look like. To reiterate the problem

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = 0$$

on a rectangular domain $0 \leq x \leq 7, \quad 0 \leq y \leq 9$ with boundary conditions $u(0,y) = u(7,y) = u(x,0) = 0$ and $u(x,9) = 12$

To solve this analytically, we used separation of variables by hand. When the boundary conditions were applied, the following analytical solution was found

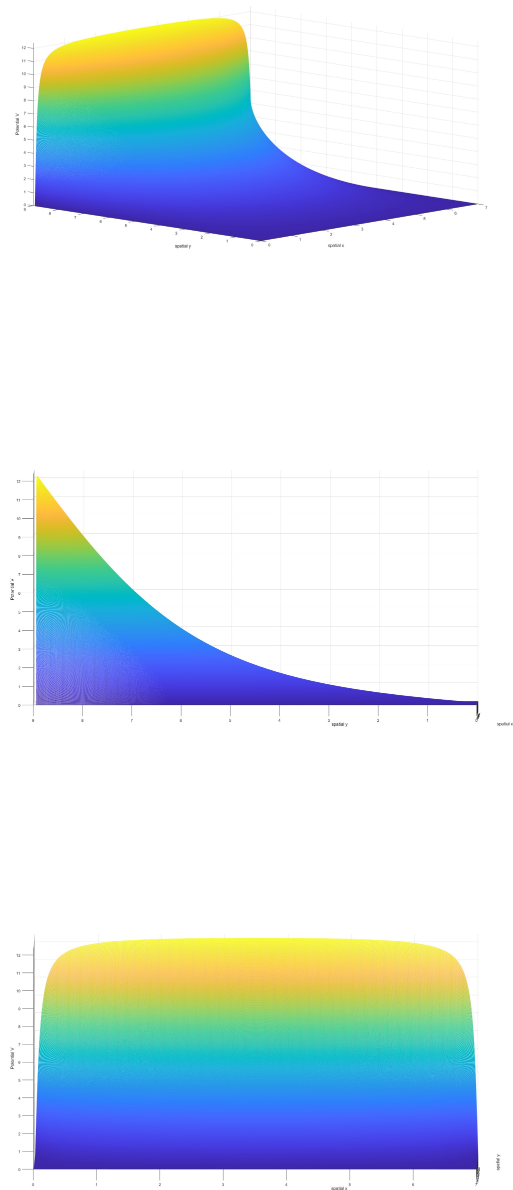
$$u(x,y) = \sum_{n=1}^{\infty} \frac{48}{n\pi \sinh(9\sqrt{\lambda})} \sin(x\sqrt{\lambda}) \sinh(y\sqrt{\lambda}) \tag{3}$$

Where $\lambda = \frac{n^2\pi^2}{7^2}$

4.1. **Analytical Solution Plot.**

Here the analytical solution is displayed from several angles

FIGURE 1. Analytical Solution



4.2. **Analytical Solution Equipotential Lines.** Here we several plots with a different amount of equipotential lines with spatial y as the horizontal axis and spatial x as the vertical axis, which gives an idea of the behavior of the electric potential when it is constant.

FIGURE 2. 5 Equipotential Lines

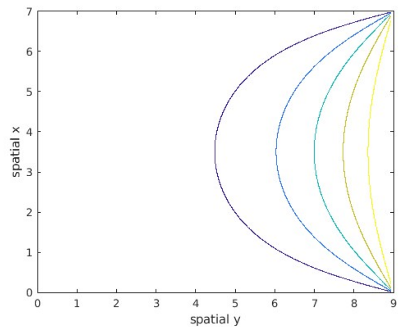


FIGURE 3. 15 Equipotential Lines

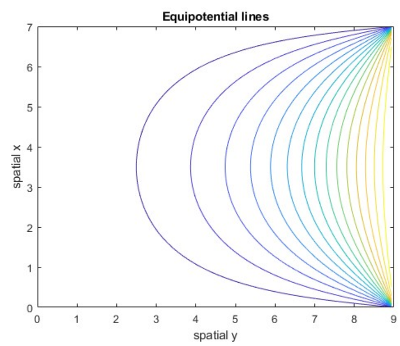
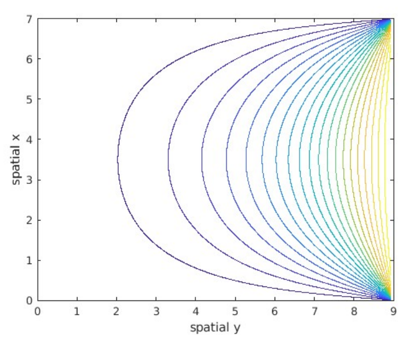


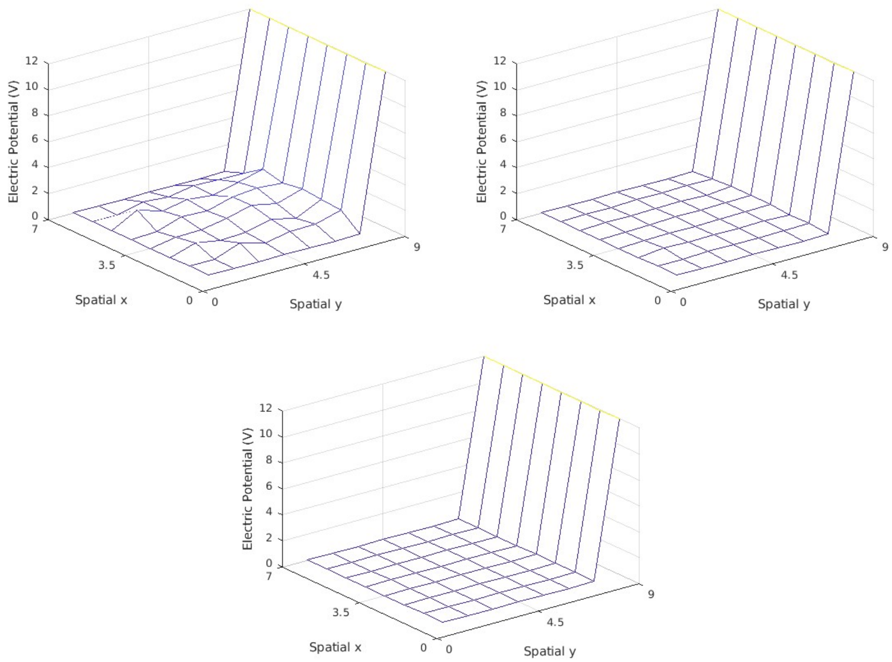
FIGURE 4. 20 Equipotential Lines



5. Computational Results

5.1. **Convergence Towards Analytical Solution.** For a small grid size n, the solution has significant error regardless of number of trials, m.

FIGURE 5. Solution with $m = 100$, $m = 1000$, and $m = 10000$



Clearly, a larger grid size is necessary to see results. By setting $n = m = 350$, we obtain Figure 6 above which looks very similar to 1.

FIGURE 6. Numerical solution with a grid size of 350x350 and 350 realizations

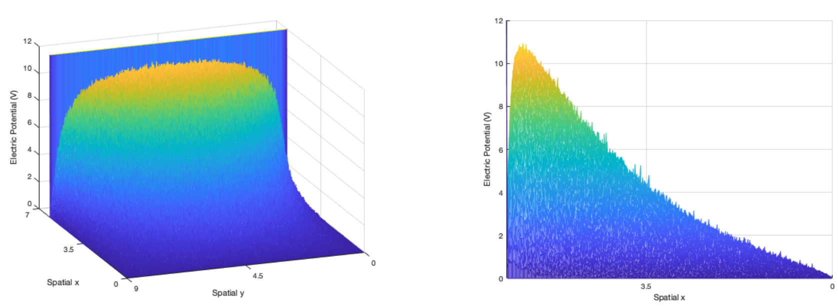
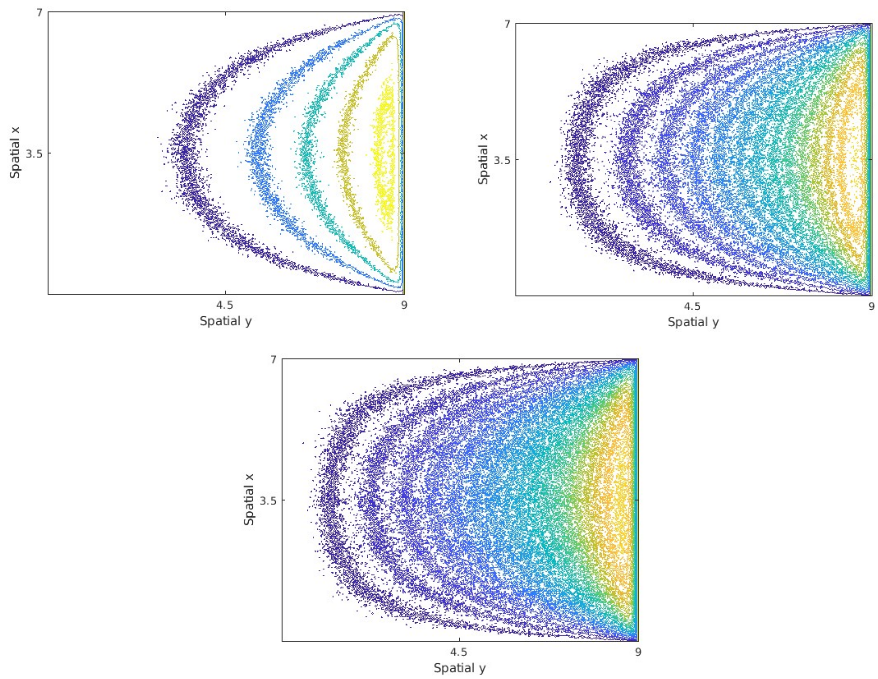
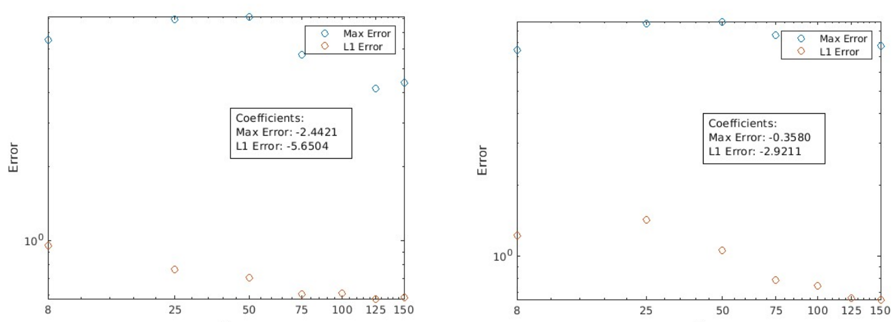


Figure 7. Equipotential lines of the numerical solution with 5, 15, and 20 contour lines



We can see in Figure 8 that the error decreases at a superlinear rate as the grid size is increased.

Figure 8. Error Convergence as n increases with $m = 100$ and $m = 1000$



As the grid size increases, the error decreases. Despite the large number of trials having a slower convergence, they both still converge towards the analytical solution.

5.2. **Computation Time.** We tested various different grid sizes and number of trials as seen above. We have found the computation time to be $O(n^4m^2)$ where n is the size of the grid and m is the number of trials.

For $n = 8$ and $m = 10,000$, the computation time was 0.47 seconds. For the final solution plot above, we used $n = m = 350$ and found the computation time to be 553 minutes or approximately 9.2 hours.

6. Discussion

We have found that the Tour Du Wino is effective in solving Laplace's Equation on a rectangular domain with Dirichlet boundary conditions which allows for a simulation of electric potential from a line charge diffusing through a rectangular domain. An important topic to cover is what occurs at the corners of our solution. Near the corner where the 12 V and 0 V boundary conditions meet, the analytic solution behaves reasonably and smoothly increases from 0 to 12 as y increases. The numerical solution instantly jumps to 12 along the boundary while immediately next to the boundary it rises similarly to the analytic solution.

As far as contribution to the project, Matt Cassini wrote the code to implement the Tour Du Wino algorithm and the error convergence, Moises Ramos derived and plotted the analytical solution and made the random walker animation, and Marissah McNeil wrote the code for the equipotential lines.

References

[1] S.J. Farlow. *Partial Differential Equations for Scientists and Engineers*. Dover Books on Mathematics. Dover Publications, 2012.

Department of Mathematical Sciences, New Jersey Institute of Technology, University Heights, Newark, NJ 07102
Email address: mc225@njit.edu

Department of Mathematical Sciences, New Jersey Institute of Technology, University Heights, Newark, NJ 07102
Email address: mm2458@njit.edu

Department of Mathematical Sciences, New Jersey Institute of Technology, University Heights, Newark, NJ 07102
Email address: mlr4@njit.edu