Computational Physics Exercise 2: Partial Differential Equations

Paul Hayes
Level 6 Computational Physics, School of Physics, University of Bristol.
(Dated: March 20, 2018)

I. NUMERICAL SOLUTIONS OF LAPLACE'S EQUATION

The Poisson equation ($\nabla^2 V = \rho$) in one dimension can be expanded in a Taylor series about the point x_i to give:

$$V(x) = V(x_i) + \delta x \frac{dV(x_i)}{dx} + \frac{\delta x^2}{2}\frac{d^2V(x_i)}{dx^2} + ..., \quad (1)$$

after adding values at $\delta x = \pm h$, rearranging and generalising to two dimensions gives:

$$V(x_i, y_j) = \frac{1}{4}(V(x_{i-1}, y_j) + V(x_{i+1}, y_j) + V(x_i, y_{j-1}) + V(x_i, y_{j+1})), \quad (2)$$

where ρ has been set to zero, for the Laplace equation $\nabla^2 V = 0$. In this case the potential on a grid of nodes separated by δx is the average of its four closest neighbours[1].

A. Method

To solve the Laplace equation, two algorithms may be used: the Gauss-Seidel and Jacobi methods. In Gauss-Seidel there exists one copy of the potential array and values at each node are updated continually using partially old and partially new values. In Jacobi a new value is calculated at each node based on the previous array of nodes, which uses two copies of the potential.

The boundaries of the grid of nodes require special consideration. At the corners there are only 2 closest neighbours and the edges 3. Therefore in the code this required if statements to not calculate nodes that did not exist.

B. Results and Discussion

Comparison of the Gauss-Seidel and Jacobi methods.

FIG.1 shows the relationship between the number of iterations needed to reach convergence and the density of grid used. The Gauss-Seidel method reaches convergence in less iterations than the Jacobi for each grid density measured, with a linear relationship measured in both cases. However, the expected relationship is n^2 [2], showing disagreement. FIG. 2 shows a log_2 graph. If it indeed n^2, it would be linear, but instead it behaves like n is linear. However, the number of iterations required for Gauss-Seidel is half that of the Jacobi for each grid density, which is in agreement of the expected result [2].

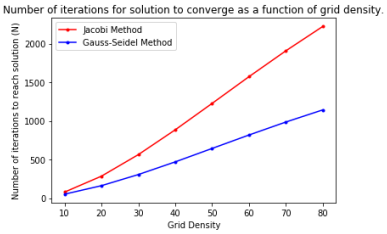

FIG. 1. Number of iterations to reach convergence as a function of grid density.

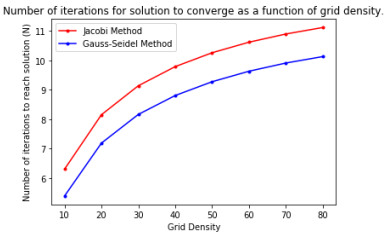

FIG. 2. The log base 2 of number of iterations to reach convergence as a function of grid density.

The number of iterations of the finite difference formula needed to reach convergence as a function of the tolerance condition is seen in FIG.3. For a given tolerance value, the Gauss-Seidel method reaches convergence in fewer iterations than the Jacobi method. An inverse relationship between number of iterations and tolerance is seen. The tolerance value used throughout is chosen to be at a higher value to reduce the computational time needed to reach convergence, but not too high at the cost of accuracy. A value of 0.1 was chosen.

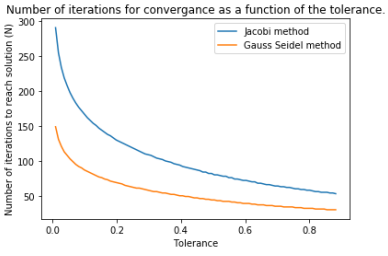

FIG. 3. Number of iterations to reach convergence as a function of tolerence.

In FIG.4, the time to reach convergence as a function of grid density is measured. In this case, time increases as n^4, which is in agreement with the expected value [2]. This is shown by FIG.5, a plot of the log_4 of the number of iterations. This

shows a relatively linear relationship, confirming this theory. Not only does The Gauss-Seidel method reach convergence in fewer iterations, but also in less time. It is clear that as the grid density increases too high, the time will increase dramatically, thus using the two methods for scientific computing is impractical.

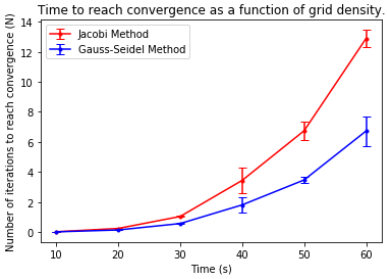

FIG. 4. Time to reach convergence as a function of grid density.

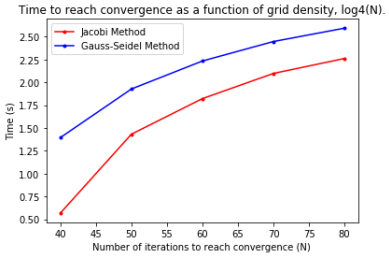

FIG. 5. Log base 4 of time to reach convergence as a function of grid density.

Test cases

Dirichlet boundary conditions are used in each of the following test cases. This is where the boundaries V(x=0,y) = V(x,y=0) = V(x=xmax,y) = V(x,y=ymax) = 0[3]. The Gauss-Seidel method was used as it reached convergence in less time, less iterations and a smaller tolerance was needed.

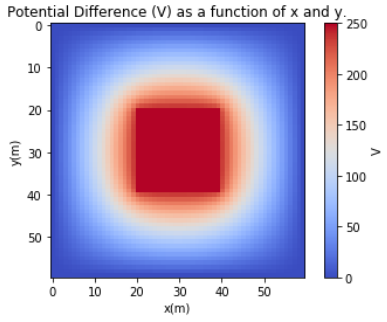

FIG. 6. Potential difference of a square plate capacitor.

FIG.6 and 7 show a square plate capacitor of 250V. The potential difference goes to zero at the edges and is constant over the plate, as expected. This shows that the algorithm is working effectively.

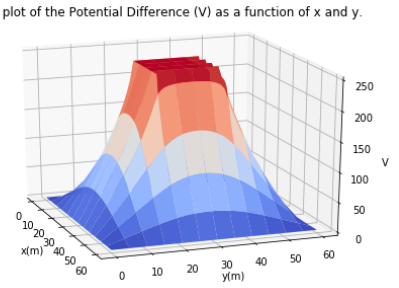

FIG. 7. Potential difference in 3D as a function of position of a square plate capacitor.

For the case of a point charge, this is difficult to interpret. To define a point charge, a single point on the grid of nodes is given a voltage. However, a point charge in this case cannot be interpreted as it must have a physical size, as seen in FIG.5. Indeed, as the size of the point charge approaches zero, its potential goes to infinity. This cannot be solved numerically. Therefore, FIG.8 does not represent a point charge.

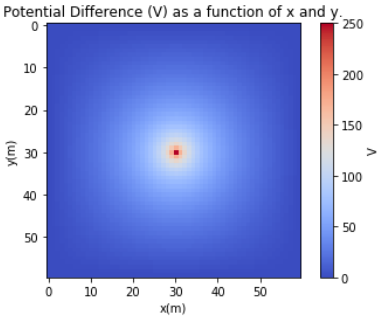

FIG. 8. Potential difference of a 'point source' charge.

Parallel plate capacitor

FIG.9 and FIG.10 show the potential difference and Electric field strength from a finite extent parallel plate capacitor of potential difference 500V. Dirichlet boundary conditions have been used, where the potential at the boundaries is set to 0. This is equivalent to saying the boundaries are infinitely far away from the capacitor. The potential indeed goes to zero at the boundaries, with the potential difference dropping off gradually outside the parallel plate and inside a constant gradient is observed.

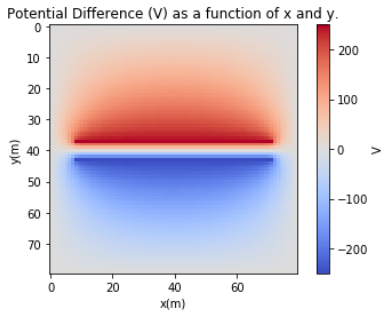

FIG. 9. Potential difference of a finite extent parallel plate capacitor.

The electric field in FIG.10 is constant between the capacitors, but at the edges there is a discontinuity where the Electric field spikes, and 'leaks' out. This expected as at the edges there will be points where there field is not uniform.

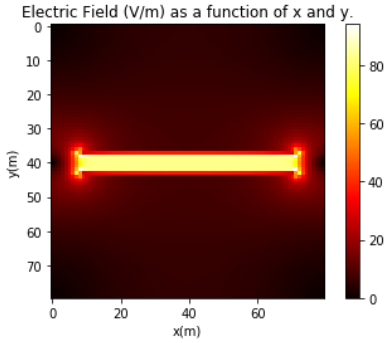

FIG. 10. Electric field strength of a parallel plate capacitor.

FIG.11 and 12 show the electric field as a function of position in 2D for a parallel plate capacitor of infinite extent. The Potential difference between the two capacitors is 500V, and the spacing 3m. Theoretically this should give a value for the Electric Field, from E=V/D, where D is the distance between the plates, of 83.3 V/m. The experimental results confirm this value, as seen clearly by the maximum values in FIG.11 and 12. Furthermore, the electric field must be equal to zero outside the plates. This is also seen in the two figures.

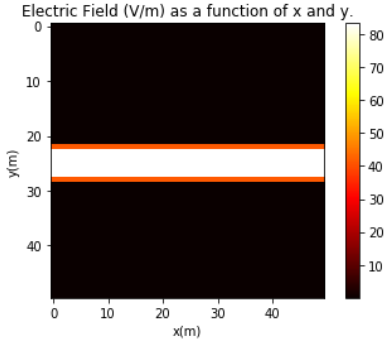

FIG. 11. Electric field of an infinite parallel plate capacitor.

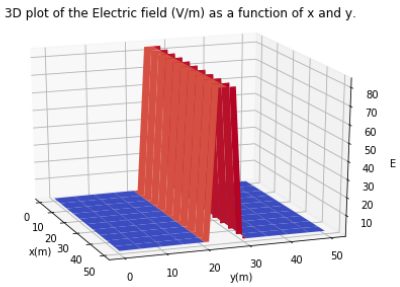

FIG. 12. 3D representation of the electric field of an infinite parallel plate capacitor.

II. DIFFUSION

To solve the diffusion equation,

$$\alpha\nabla^2\phi = \delta\phi/\delta t, \tag{3}$$

finite difference equations are used to propagate values at each node forward in time from known initial conditions. This finite difference formula is,

$$\frac{\phi`(x_i) - \phi(x_i)}{\delta t} = \frac{\alpha[\phi(x_{i-1}) + \phi(x_{i+1}) - 2\phi(x_i)]}{h^2}, \tag{4}$$

where ϕ at three neighbouring points at time t are used to find the value $\phi`$ at $t + \delta t$. However this forward-time form is usually unstable for initial condition problems. Therefore, a backward time formula is used in the form,

$$\frac{\phi`(x_i) - \phi(x_i)}{\delta t} = \frac{\alpha[\phi`(x_{i-1}) + \phi`(x_{i+1}) - 2\phi`(x_i)]}{h^2}, \tag{5}$$

which can be solved as a simultaneous equation by writing it as a matrix.

A. Method

The equation that is solved simultaneously using LU decomposition is,

$$A\phi` = \phi, \tag{6}$$

where A is a matrix of coefficients, $\phi`$ are the values of the temperature at nodes along a poker of length 0.5m at $t + \delta t$ and ϕ are the values of the temperature along the poker at time t. The matrix A is of the form,

$$\begin{pmatrix} 1 & 0 & 0 & 0 & ... & 0 \\ -a & b & -a & 0 & ... & 0 \\ 0 & -a & b & -a & ... & 0 \\ 0 & 0 & -a & b & ... & 0 \\ ... & ... & ... & ... & ... & 0 \\ 0 & 0 & 0 & 0 & ... & 1 \end{pmatrix}$$

, where $-a = -\frac{\alpha\delta t}{h^2}$, $b = \frac{2\alpha\delta t}{h^2} - 1$, h is the distance between nodes and $alpha$ is the thermal diffusivity.

Two cases were considered, the first one end of a poker initially in a furnace at 1000^oC and the other at 20^oC. In this case no heat loss was allowed at the lower temperature end. To solve this, the last value in the array of temperatures after each iteration was set to equal the value of the point next to it. This allowed the temperature of the end point to increase without any loss.

The second case, one end was again kept in a furnace at the same temperature, except the other was submerged in an ice bath at 0^oC.

B. Results and Discussion

FIG.13 shows varying temperature distributions along the poker after being left for different periods of time. For this particular case, the initial temperatures were 1000^oC at one end and 20^oC at the other. The 1000^oC end is left in a furnace at this temperature.

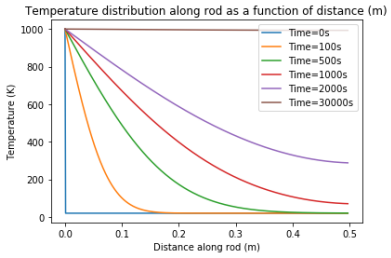

FIG. 13. Temperature distribution along a rod with one end fixed at 1000^oC and the other at initial temperature 20^oC.

If the poker is left for a large enough time, the temperature distribution along it should approach the furnace's temperature. This is shown in both FIG.13 and FIG.14, where in the first the lines approach this temperature and in the second it approaches a constant distribution of temperatures. Indeed, it was found that after 76023 seconds the temperature along the whole rod was within 0.01^oC of 1000^oC.

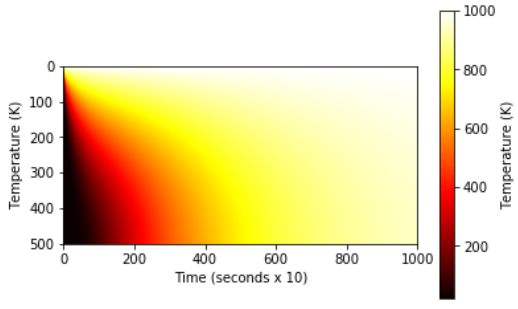

FIG. 14. Temperature distribution represented as a heat map for case 1.

For the second case, one end of the poker was again kept in a furnace at 1000^oC, but the other end was submerged in an ice bath at 0^oC. Theoretically, after enough time has passed, the temperature distribution along the rod should approach a constant gradient.

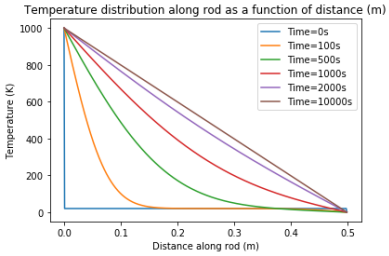

FIG. 15. Temperature distribution along a poker as a function of position for a number of times.

FIG.15 shows that the temperature distribution approaches a straight line as a function of the position across the rod. Furthermore, FIG.16 also shows that it approaches a constant gradient between the two temperatures as time passes. Indeed, it took 8339 seconds for the rod to reach equilibrium along the rod, within 1^oC.

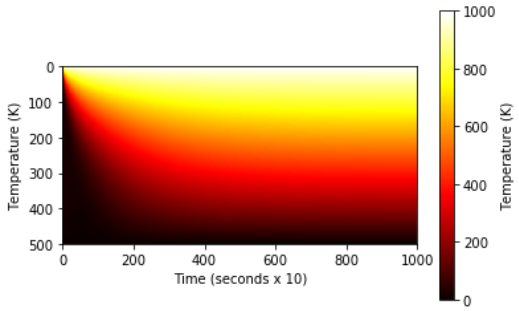

FIG. 16. Temperature distribution represented as a heat map.

III. CONCLUSIONS

In conclusion, the Gauss-Seidel method was convergence to given solution in less time and less number of iterations for a given grid density than The Jacobi. A lower tolerance condition could also be used. Although, for accurate scientific research where high resolution and grid densities may be required, these algorithms are too computationally draining and thus are impractical. However, the algorithm was able to accurately model an infinite parallel plate capacitor, so is effective for such applications. In solving the diffusion equation, the methods used accurately described what was expected physically.

REFERENCES

[1] Dr. Eric Ayars, Computational Physics With Python, California State University, Chico (2013)
[2] Mary L. Boas, Mathematical Methods in the Physical Sciences, third edition.
[3] Hansen, Per Brinch, Numerical Solution of Laplace's Equation (1992). Electrical Engineering and Computer Science Technical Reports.Paper 168.