Assignment 2 Partial Differential Equations:

INTRODUCTION

Solving partial differential equations is a vital part of physics, and can be used to describe a range of physical phenomena. As such, computational methods are useful in modelling physical systems. In this assignment it will be used to model capacitors electric field and potential and heat diffusion.

To solve these differentials the finite difference method is implemented using the Gauss-Seidel Method. This entails defining a grid of nodes, with initial boundary conditions. New values of the nodes are calculated using the finite difference equation, for Task 1 this is obtained through a Taylor expansion to find

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

Where V is the potential and h is the spacing between nodes in 2-D. For a system with no sinks or sources this is equivalent to the average of all 4 surrounding points. This process is iterated until a convergence condition is met, for the purposes of this assignment this was taken to be when the new average value of all the nodes only changes by a predefined amount.

TASK 1

In this section Eq.1 is used where $\rho = 0$. to solve Laplace's equation, $\nabla^2 V = 0$ in 2 dimensions. In FIG.1 we see what we obtain when the top edge is set to a positive constant and zero everywhere else, with interpolation set to 100, which smooths out the data.

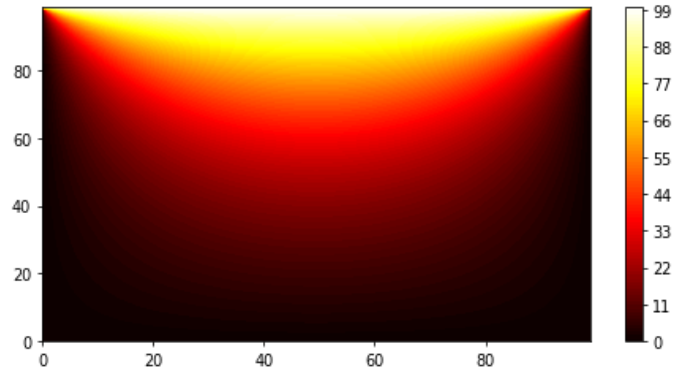

FIG. 1. Solution to the Laplace equation with boundary conditions of 100 at upper boundary and zero on all other sides.

This is as we would expect from a simple heat propagation. FIG.2 is an example of the same initial conditions with no interpolation, to show the effect interpolating has on the graph produced. We see that interpolating smooths out the sharp edges from the choice of grid density.

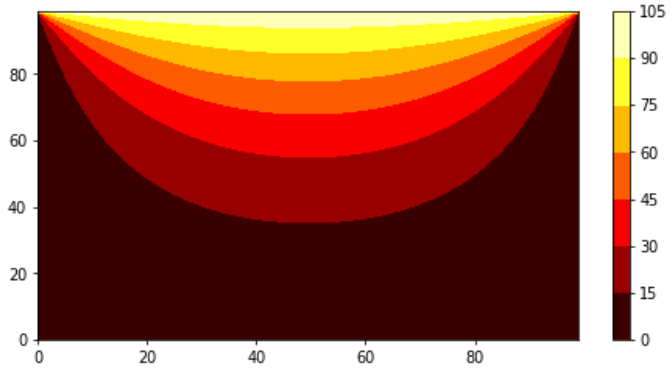

FIG. 2. Same solution and setup as in FIG.1 however no interpolation, to show the effect interpolation has on the final graph.

The effect of grid density on number of iterations needed to meet the convergence condition was investigated, we would expect this to increase proportional to N^2 for an $N \times N$ grid, as the number of calculations is directly proportional to the number of nodes. This is shown in FIG.3 below.

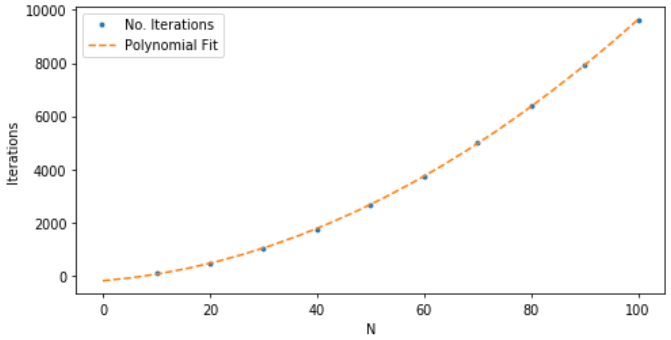

FIG. 3. Plot of Grid size vs Number of iterations, fitted with a second order polynomial fit in python, giving an R^2 value of 0.9999, indicating a good fit.

From this figure a 2nd order polynomial is observed, fitted using numpy.polyfit, which uses a least squares method. A 2nd order polynomial fit gives an R-Squared value from excel of 0.9999, supporting the N^2 hypothesis.

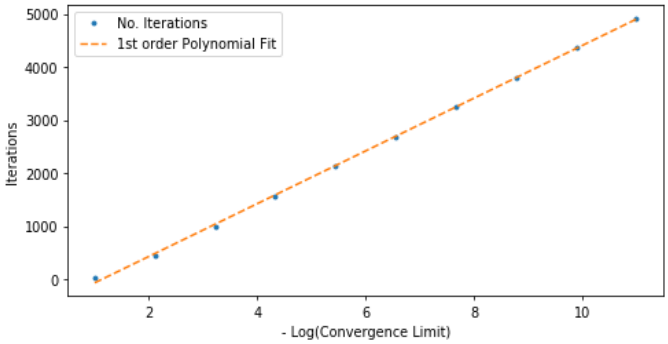

FIG. 4. Plot of -Log(Convergence Limit) vs Number of Iterations with a linear fit giving an R^2 value of 0.9994, indicating a good fit. This tells us the number of iterations scales logarithmically with grid size.

The effect of the convergence limit was then investigated, for this assignment the convergence limit was defined as minimum change between the mean value of the grid before and after a complete iteration through all grid points (excluding boundary conditions). A logarithmic relationship between convergence condition and number of iterations until met was demonstrated as seen in FIG. 4, where a linear fit with R-squared value 0.9994 can be seen when the negative log of the convergence limit is plotted against No. of iterations.

Between a convergence value of 1 and 0.1, where the number of iterations is low a power law distribution is observed with R-Squared value = 0.9787, which may explain the slight deviation from the logarithmic relationship seen in FIG.4 for the first entry at 0.1, as it transitions to a power law relationship, as evidenced in FIG.5

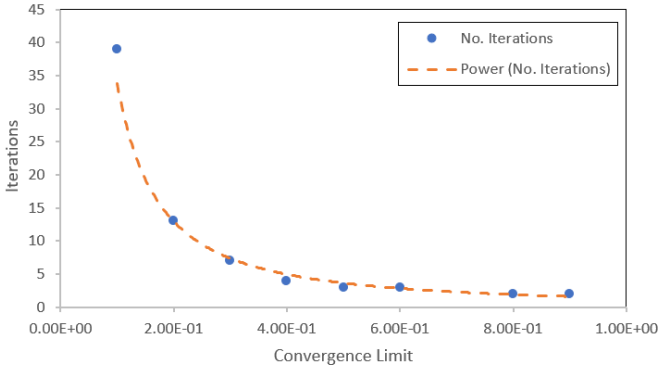

FIG. 5. Plot of Convergence Limit vs Number of iterations where the convergence limit is between 0.1 and 1, this gives a power law distribution with an R^2 value of 0.9787 indicating that it may be described by a power law. Plotted in excel.

PHYSICS PROBLEM 1

In this section the above methods are applied to a well known case of parallel plate capacitors to investigate the relationship for electric field and electric potential at the approximation of infinite length and small separation. By setting initial conditions that are left unchanged by iterations to represent the capacitors, using the method described in Task 1 and allowing the program to reach a convergence condition should result in a model of the capacitor system. In FIG.6 we can see the result from finite plates, where the arrows represent the electric field vectors, generated using the matplotlib library quiver plots. The electric field and potential show the form we would expect [1].

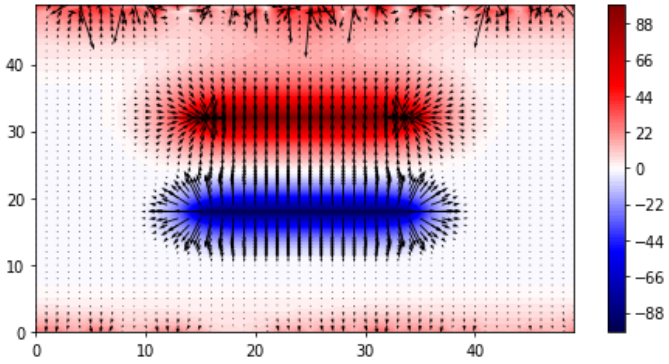

FIG. 6. Solution for finite length parallel plate capacitors, giving the solution we would expect, excluding some errors at the boundary due finite grid size.

To extend the program to model an infinite plate, periodic boundary conditions were applied in the direction parallel to the plate surface. We expect this to show a relationship $E = V/d$ where E is the magnitude of the electric field vector, V is the electric potential and d is separation between the plate, with the electric field vectors acting perpendicular to the plates. Outside the plates the electric field and hence the potential will go to zero.

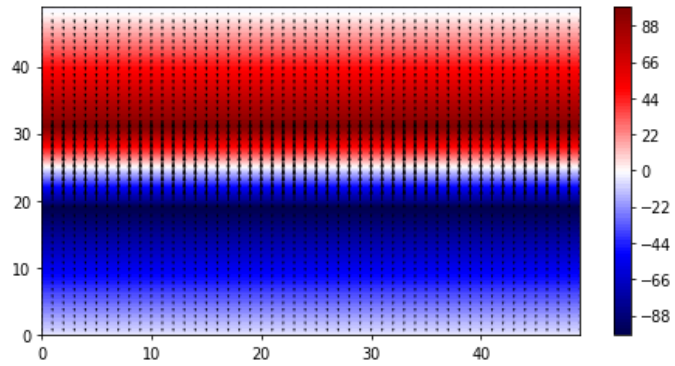

FIG. 7. Solution for the infinite parallel plate capacitors with same initial guess for all nodes. Between the plates we see the $E = V/d$ solution we expect, however we do not observe the zero electric field outside of the cavity, due to the initial guess.

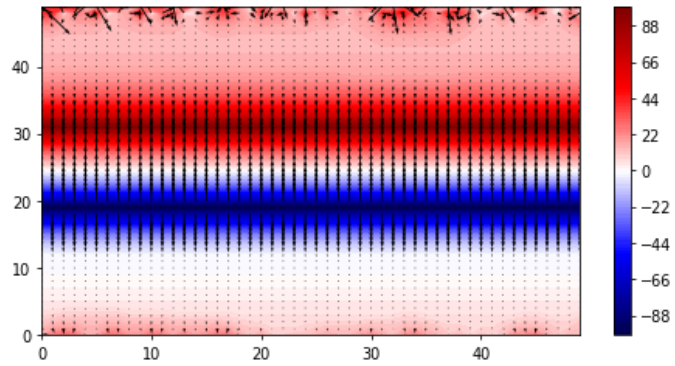

FIG. 8. Solution for the infinite parallel plate capacitors with randomised initial guess's for all nodes. Between the plates we see the $E = V/d$ solution we expect, and observe something similar to what we expect outside the cavity, with some error as it does not go to zero and anomalies at the boundaries.

In FIG.7 we see the result when a constant guess is used everywhere except the initial conditions (the plate). This results in the gradient between the two plates that we expected, however this gradient is continued outside of the capacitor cavity. This may be explained by treating the grid of homogeneous nodes as a stable system. When the initial conditions are defined, the electric potential propagates, in real cases there is an infinite set of vertical nodes, however in our system we are limited to a finite amount and not allowed to dissipate hence a potential difference is set up between the capacitor and the upper and lower boundaries resulting in the gradient seen in FIG.7. better way of dealing with the upper and lower boundary nodes could be defined, but by setting the initial guesses to random guesses we see a better approximation of the infinite plate solution as can be seen in FIG.8. Ignoring anomalies, particularly at the boundary we see in general that the external electric field does tend to zero, though does not equal zero, instead just a constant value.

PHYSICS PROBLEM 2: HOT POKER

In this problem a 1 dimensional iron poker is placed with one end in a 1000 degrees Celsius furnace. Using the finite approximation method as described above on the diffusion equation, Eq.2 below,

$$\alpha \nabla^2 \phi = \partial\phi/\partial t \tag{2}$$

we obtain the following.

$$\frac{\phi'(x_i) - \phi(x_i)}{\Delta t} = \frac{\alpha}{h^2}\left[\phi'(x_{i-1}) + \phi'(x_{i+1}) - 2\phi'(x_i)\right] \tag{3}$$

where α is the diffusion coefficient, ϕ is the temperature at time t, ϕ' is the temperature at time $t + \delta t$, δt is the timestep and h is the distance between the nodes. This gives us a set of simultaneous equations which from Exercise 1 we know can be solved simply through the use of matrices. We define a 1 dimensional set of nodes, of arbitrary size and expand Eq. 3 we obtain values of temperature outside of the boundary of our nodes. This is solved through the of "ghost" cells at either end, which are used to define our boundary condition. First, the solutions is to be found when there is no heat loss throughout the poker, for this we will define the average temperature between the first and second cells to be equal to the furnace temperature, this is known as a Dirichlet boundary condition [2]. At the other end we will define the difference between the penultimate cell and ghost cell temperature to be zero, i.e no heat flux between them, hence no heat will be lost, this is known as a Neumann boundary condition [2]. The equation obtained through these definitions is seen in Eq.4 for 4 nodes, this is easily expanded to N nodes, continuing the diagonal pattern. Using numpy.linalg.solve we obtain solutions for temperature throughout the poker for each time step. From Exercise 1 we know that computational time of this should go as N^2, where N is the number of nodes. and the resulting graph in FIG.9.

$$\begin{pmatrix} 1+3\gamma & -\gamma & 0 & 0 \\ -\gamma & 1+2\gamma & -\gamma & 0 \\ 0 & -\gamma & 1+2\gamma & -\gamma \\ 0 & 0 & -\gamma & 1+\gamma \end{pmatrix} \cdot \begin{pmatrix} \phi_1' \\ \phi_2' \\ \phi_3' \\ \phi_4' \end{pmatrix} = \begin{pmatrix} \phi_1 + 2\gamma T_H \\ \phi_2 \\ \phi_3 \\ \phi_4 \end{pmatrix} \tag{4}$$

where $\gamma = \frac{\alpha \delta t}{h^2}$ and T_H is the temperature of the hot resevoir.

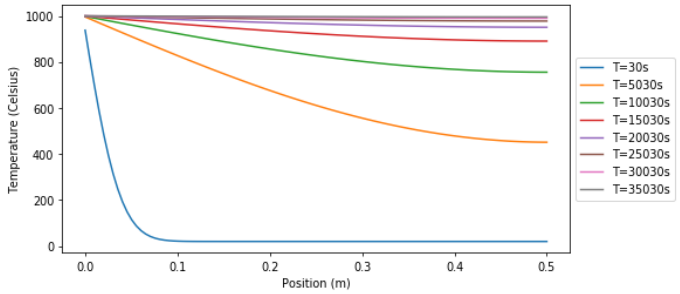

FIG. 9. Graph of Temperature vs Position for the iron poker at different times.

Fig.9 describes temperature as we would expect in the case of no heat loss. As time increases the temperature becomes more evenly distributed, until it all is heated to the same temperature after a time of around 30000s or so.

To define the condition where one of the poker is in the furnace and the other is in an ice bath at 0 degrees Celsius, we apply a similar logic to above, in this instance the average between the ghost cell and the adjacent cell is set to 0. This results in the following equation which is solved in the same way, resulting in FIG.10.

$$\begin{pmatrix} 1+3\gamma & -\gamma & 0 & 0 \\ -\gamma & 1+2\gamma & -\gamma & 0 \\ 0 & -\gamma & 1+2\gamma & -\gamma \\ 0 & 0 & -\gamma & 1+3\gamma \end{pmatrix} \cdot \begin{pmatrix} \phi_1' \\ \phi_2' \\ \phi_3' \\ \phi_4' \end{pmatrix} = \begin{pmatrix} \phi_1 + 2\gamma T_H \\ \phi_2 \\ \phi_3 \\ \phi_4 + 2\gamma T_C \end{pmatrix} \tag{5}$$

Where T_C is the temperature of the ice bath.

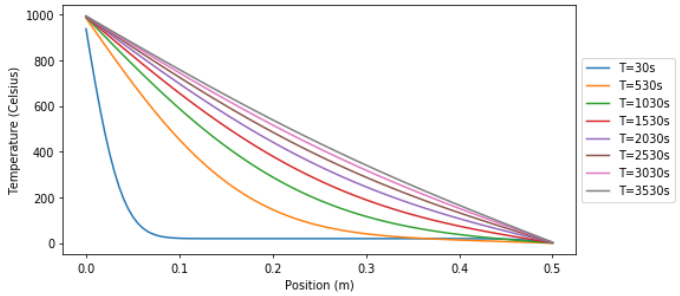

FIG. 10. Graph of Temperature vs Position for the iron poker at different times when ice bath placed at one end.

From this we see that as time progresses the temperature gradient becomes approximately constant, resulting in a linear distribution of temperature as time becomes large. This is what we would expect as it is essentially a potential difference between the two ends of the poker.

As an easier visualisation of what this looks like an animation of the heat distribution in time is included in the code, a snapshot of this can be seen below in FIG.11 taken at a time of 3950s.

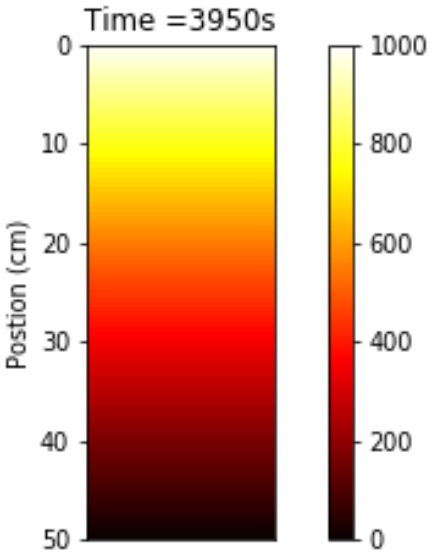

FIG. 11. Visualisation of Temperature in poker taken at 3950s, with ice bath at one end.

Here we clearly see the constant temperature gradient achieved after a long period of time.

CONCLUSION

In this exercise we demonstrated the power of using partial differential equations to model physical phenomena, including parallel plate capacitors and heat propagation. It was shown that the Gauss-Seidel method is a good method for implementing finite difference equations to a system, demonstrated by solving the Laplace equation. It was then shown that it is possible to solve the electric field and potential for parallel plates, though with some problems due to finite grid size and boundary conditions, to achieve similar results to those we expect. Lastly, using linear algebra to solve a more complex time and space dependent equation, the heat propagation of heat through an iron poker was found, with no heat loss and when an ice bath is used, and corresponds to the solutions we expect.

REFERENCES

[1] Nagel, James Org, Nageljr@ieee. (2011). Solving the Generalized Poisson Equation Using the Finite-Difference Method (FDM).

[2] Cheng, A. and D. T. Cheng (2005). Heritage and early history of the boundary element method, **Engineering Analysis with Boundary Elements**