This project solves the heat equation in 2D using the finite difference numerical method.

The two-dimensional heat equation is given by:

$$\frac{\partial u}{\partial t} = \alpha \left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} \right), \tag{1}$$

- $u(x, y, t)$ is the temperature distribution,
- α is the thermal diffusivity,
- x, y are the spatial coordinates,
- t is time.

article amsmath

The two-dimensional heat equation is:

$$\frac{\partial u}{\partial t} = \alpha \left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} \right), \tag{2}$$

where $u(x, y, t)$ is the temperature, α is the thermal diffusivity, and x, y, t are spatial and temporal variables.

Using the finite difference method, discretize the domain into a grid with spacing Δx and Δy in the x and y directions, and a time step Δt. Let $u_{i,j}^n$ represent the temperature at grid point (i, j) and time step n. The spatial derivatives are approximated as:

$$\frac{\partial^2 u}{\partial x^2} \approx \frac{u_{i+1,j}^n - 2u_{i,j}^n + u_{i-1,j}^n}{\Delta x^2}, \tag{3}$$

$$\frac{\partial^2 u}{\partial y^2} \approx \frac{u_{i,j+1}^n - 2u_{i,j}^n + u_{i,j-1}^n}{\Delta y^2}. \tag{4}$$

The explicit time integration scheme is:

$$u_{i,j}^{n+1} = u_{i,j}^n + \alpha \Delta t \left(\frac{u_{i+1,j}^n - 2u_{i,j}^n + u_{i-1,j}^n}{\Delta x^2} + \frac{u_{i,j+1}^n - 2u_{i,j}^n + u_{i,j-1}^n}{\Delta y^2} \right). \tag{5}$$

]

The code starts with the grid size of x and y directions. the dx and dy which are the adjacent grid points of the spatial resolution. The alpha shows how fast heat diffuses in the material. The dt represents the time increment for each simulation step. The time steps indicates the total number of steps that will be simulated. After that, the if condition checks the that the simuation is stable to avoid errors in the results. Then, creating a two dimensional array

using numpy library to represent the temperature of two hundreds near the grid center. Then, writing a matplotlib function to add figures of the simulation. Moreover, a package is used to enable the storing of all graph simulations in one pdf file. Then, a for loop iterate over all steps of time for simulation. Then, creating a new variable u_copy to save a copy of u and a for loop to update every point gride of temperature using FDM method. Then, setting temperature for the edges of the grid to zero as this will simulate a system where heat will be lost at the boundaries. Finally, using matplotlib library to graph the simulations of the solutions after each seven time steps and saving the entire simulations to a single file.

]

The Finite Difference Method was used in this project as this method helps to make the solution simple. It is based on discretization on a straightforaward approach that replace the continuous differential equations with algebraic equations that can be solved numerically on a discrete grid. Furthermore, it is suitable for problems with rectangular domains like the heat equation. In addition, the FDM has the ability to show the grid of temperature values at each time step which makes it easy to graph the results and track the evolution of the solution.

]

The simulation results shows a relatively uniform distribution of temperature at time step 0 which means that the system is in its initial conditions. As time passes, the temperature spreads in the domain which reflects the diffusion process of heat equation. Regarding the spatial patterns, the patterns at first showed a distribution in temperature such as the circular shape however, as time passses, the shapes significantly changed which might be due to nonlinear interactions or boundary conditions. Also, the appearance of high-temperature locations raising over time might be due to concentrated heat sources or numerical instabilities in the simulation.