Department of Mechanical Engineering

Spring 2017

Scientific Computing for
Mechanical Engineers

Diffusion Equation: 2D Linear Solver

Submitted By: Carlton Kenworthy (B02-2)

Date: May 5th, 2017

Instructors: Dr. Amit Amritkar and Dr. Andrea Prosperetti

Abstract

The diffusion equation is a second order partial differential equation that describes how particles or matter diffuse though a medium over time. The diffusion equation has applicability through many fields including weather analysis and heat transfer. The diffusion equation is dependent on a concentration to drive the diffusive flux, whether the concentration is heat or number of density of people infected with a virus in a population. This project will analyze the temperature of an aircraft engine block during startup. During startup, the engine will have a certain temperature. Immediately after startup, the cooling system will begin to pull heat from the engine. The engine will also begin to generate heat. This project will evaluate the engine temperate reaching a steady state balance due to engine heat generation and the cooling system.

Mathematical Description

To solve for the heat flow over time, the **two-dimensional diffusion equation** will be introduced as:

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

This will describe the heat flow. The heat flow will be evaluated on the engine over an area of:

$$a_x < x < b_x \text{ and } a_y < y < b_y$$

The cooling system will cool the boundaries giving the **boundary conditions** of:

$$\left.\frac{\partial u}{\partial x}\right|_{x=a_x}, u(x = b_x, y) = g_a(b_x) + \frac{y - a_y}{b_y - a_y}[f_a(b_x) - g_a(b_x)]$$

$$a_x = a_y = 0, b_x = b_y = 2\pi$$

The **initial conditions** at engine startup are:

$$U(x, y, t = 0) = 0$$

Discretization and Numerical Method

In order to evaluate the heat at certain time intervals, the equation must be converted to compute discrete form. The easiest way to discretize the

diffusion equation is the explicit method, which takes the form:

$$\frac{T_{i,j}^{k+1} - T_{i,j}^{k}}{\Delta t} = \alpha \left[\left(\frac{T_{i-1,j}^{k} - 2T_{i,j}^{k} + T_{i+1,j}^{k}}{\Delta x^2} \right) + \left(\frac{T_{i,j-1}^{k} - 2T_{i,j}^{k} + T_{i,j+1}^{k}}{\Delta y^2} \right) \right]$$

This method is used in the Diffusion_Explicit code, which takes the form:

```
T(i,j,t+1)=alpha*dt*((T(i+1,j,t)-2*T(i,j,t)+T(i-
1,j,t))/(dx^2)+(T(i,j+1,t)-2*T(i,j,t)+T(i,j-1,t))/(dy^2))+T(i,j,t);
```

Computer Specifications

The computer used for this computation is an Apple Laptop made in 2009. The specifications include:

- MacBook Pro (17-inch, Early 2009)
- Processor: 2.66 GHz Intel Core 2 Duo
- Memory: 4 GB 1067 MHz DDR3
- Graphics: NVIDIA GeForce 9400M 256 MB

Results

The code initializes a 2-D equally spaced grid of 41X41 size, where the length and width of the rectangle formed by the grids is 2*pi each in direction. The simulation is done for a time period of 2 seconds with a time increment of 10 milliseconds. Henceforth, the initial condition is defined as zero everywhere at the grid points at time t = 0. Moreover, the boundary conditions are defined as per the conditions provided in the problem statement. After defining the initial and boundary conditions, a simulation is done in time domain to plot the heat equation in 2-D space at each time

interval of 10 milliseconds to observe the behavior of the heat equation with time. As is observed, with time, the heat equation attains a steady state, with maximum temperature in the middle of the grid points and minimum temperature at the boundary points of the grid.

The solution reached a steady state around 1.2 seconds and continued with the heat concentrated in the center of the engine plate. This allows the engine to perform at a constant temperature after startup. Figure 1 shows the engine just after start up.

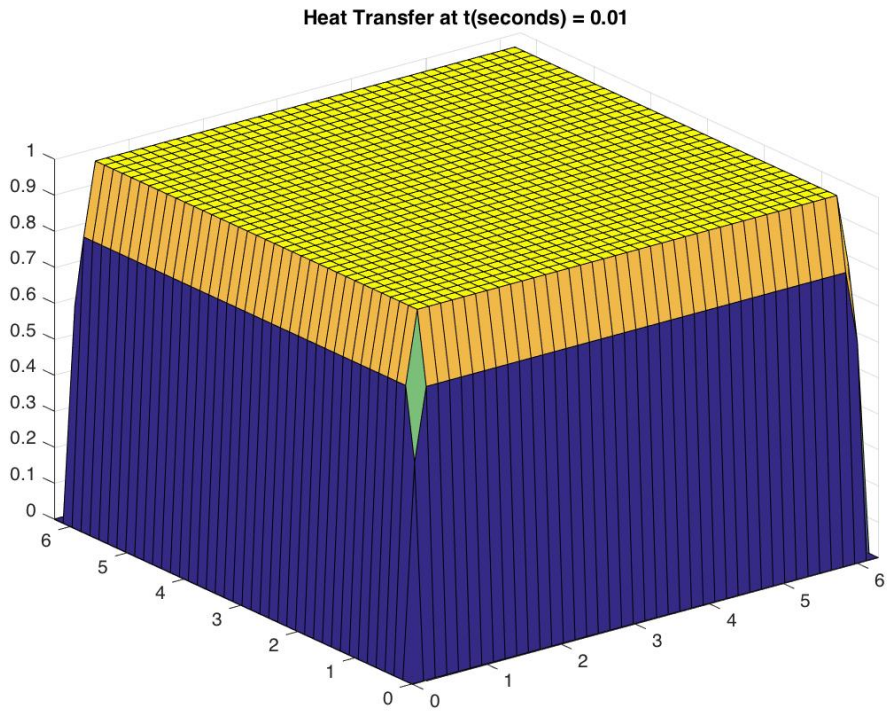

Figure 1: Temperature at .01 Seconds

This shows the engine at its initial temperature, and the cooling system beginning to expel heat at the sides of the plate. Figure 2 shows the heat at

0.5 seconds.

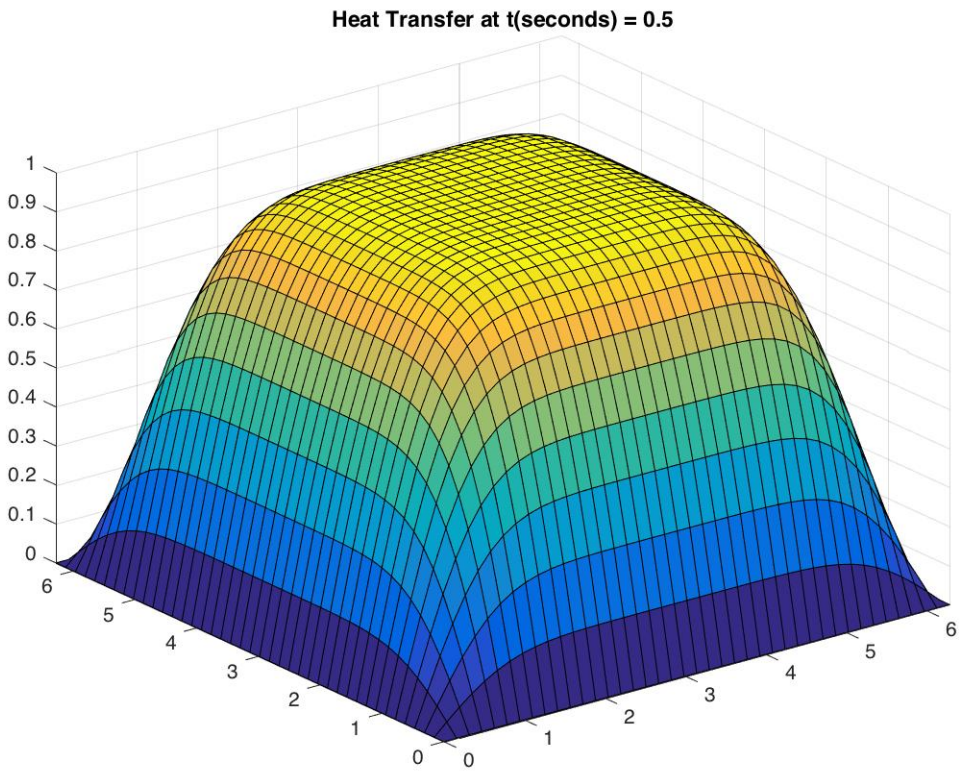

Figure 2: Temperature at. 5 seconds

After about 0.5 seconds, the heat flux begins to drop, and the engine begins to reach a stable position. Starting at 1.2 second, the engine has reached a steady state temperature as shown in Figure 3.

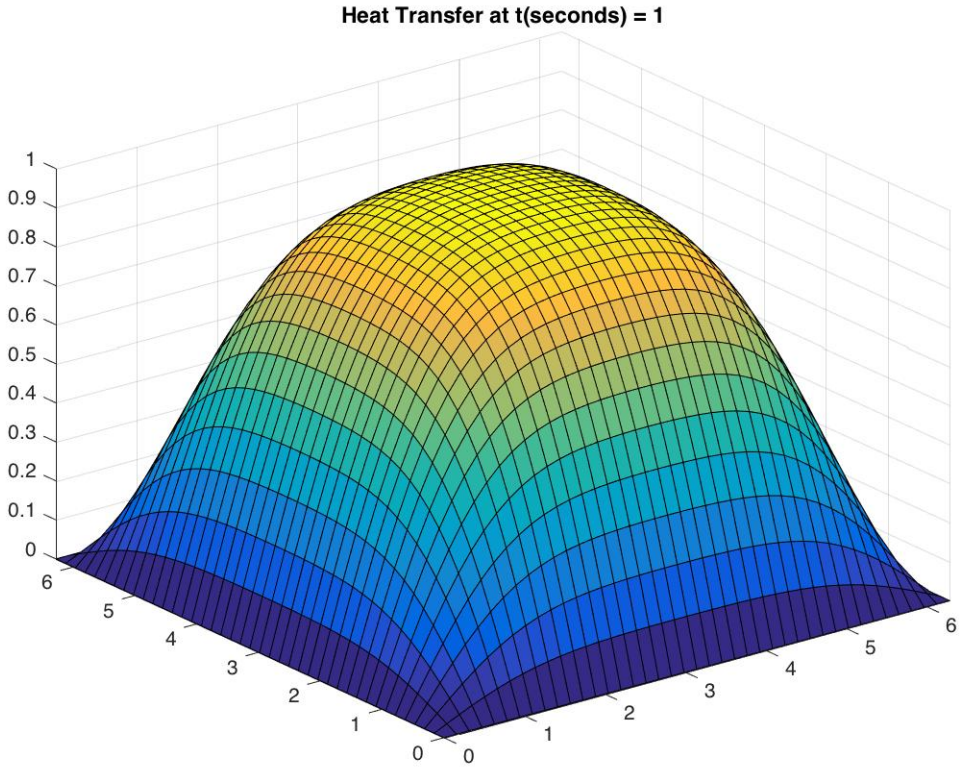

Figure 3 Temperature at 1 second

The computation continues to reach 2 seconds, but the temperature has reached a steady state, and looks very similar to 1 second. Figure 4 shows

the temperature distribution for the final steady state computation.

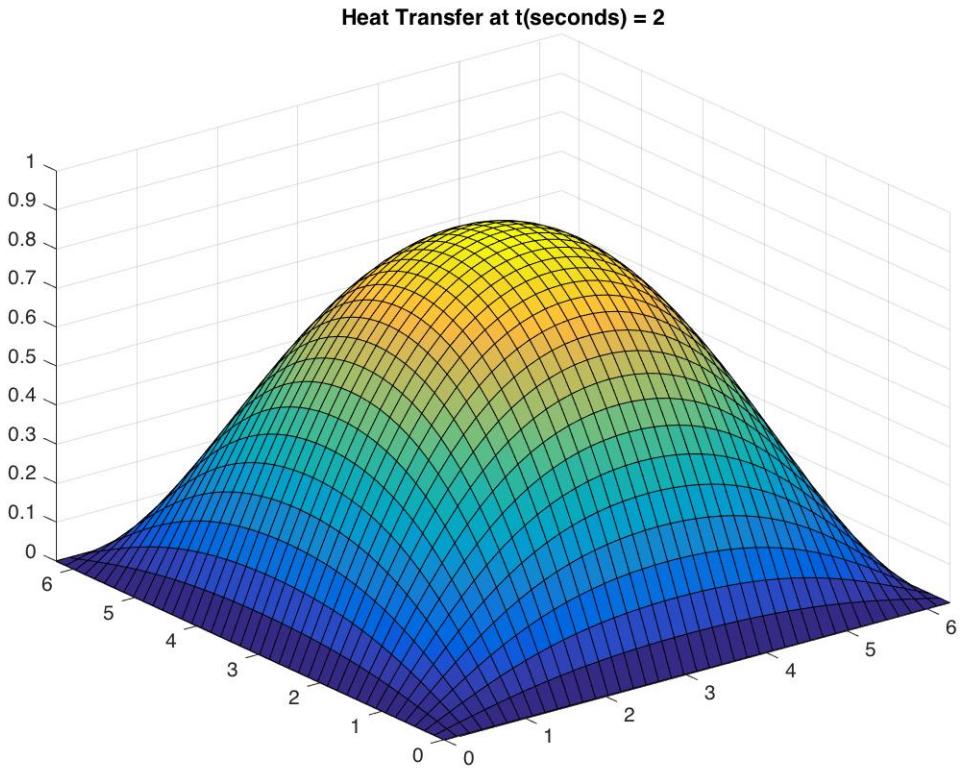

Figure 4 Steady State Temperature Distributions

In this code, the user can adjust the time evaluated and the time intervals by adjusting ‘t_f’ and ‘dt’, respectively. The user is encouraged to use a sufficiently low time step in order to properly evaluate the solution. If a small enough time step is not used, the solution will become unstable and an ‘unstable solution’ error will show.

Grid convergence of this can be adjusted by changing the number of grid points and the time intervals. This can be done by adjusting dx and dy, but also needs to adjust the grid size to compensate for the new grid points.

Conclussion

This project is a result of learning the different time discretization steps used to compute certain differential equations. Computers can not calculate derivatives, so computer coders must be able to adjust the equations into discrete time steps. This diffusion equation can often be very tricky to compute. Using the proper time step is very important to ensuring grid convergence and a stable solution. Whether computing weather information or temperature, the diffusion equation can accurately be solved by the computer code.