Modern numerical methods

Németh Gábor

Date: 2020. 06. 17.

Contents

1 Introduction

In this project we solve the time dependent parabolic heat equation on a given square like plate, with the next boundary conditions:

- On the left side the temperature is given (T_0)
- On the bottom side there is a constant heat current (j)
- On the other two sides there is perfect heat insulation

We are looking for a function which tells us the value of the temperature at given time and space coordinates. Let's call that $T(x, y, t)$, beacuse we are working in two dimensions.

2 Methods

We can write the heat equation in the next form:

$$\frac{\partial T}{\partial t} = \nabla\big(D\nabla T\big) \tag{1}$$

In this equation the D thermal diffusivity, we can choose that as a constant value. We working only two dimensions, so the equation has this form:

$$\frac{\partial T(x,y,t)}{\partial t} = D \cdot \left(\frac{\partial^2}{\partial^2 x} + \frac{\partial^2}{\partial^2 y}\right) T(x,y,t) \tag{2}$$

This problem could be solved by discretization of the plate as a grid, which is described by two coordinates(x,y) or indexes (i,j).

$$x_i = i \cdot \Delta x \quad y_j = j \cdot \Delta y \tag{3}$$

The time evolution could be handled also by discretization, the time at the n-th step is:

$$t_n = n \cdot \Delta t \tag{4}$$

The partial differential in the two sides of 2. can be approximated by difference quotient. The temperature item in the right side of the equation can be approximated by first-order time forward difference quotient.

$$\left(\frac{\partial T}{\partial t}\right)^n_{i,j} = \left(\frac{T^{n+1}_{i,j} - T^n_{i,j}}{\Delta t}\right) \tag{5}$$

The second-order partial differential can be approximated by the central difference quotient:

$$\left(\frac{\partial^2 T}{\partial x^2}\right)^n_{i,j} = \frac{\left(T^n_{i+1,j} - 2 \cdot T^n_{i,j} + T^n_{i-1,j}\right)}{\Delta x^2} \tag{6}$$

$$\left(\frac{\partial^2 T}{\partial y^2}\right)_{i,j}^{n} = \frac{\left(T_{i,j+1}^{n} - 2 \cdot T_{i,j}^{n} + T_{i,j-1}^{n}\right)}{\Delta y^2} \tag{7}$$

We can choose the same value for the spatial differences for easier computation:

$$\Delta x = \Delta y = h \tag{8}$$

Whith the previous equations and a few simplification we get the next equation for the temperature at $n+1$ timestep, if we know the states at the n step.

$$T_{i,j}^{n+1} = T_{i,j}^{n} - 4F_0 T_{i,j}^{n} + F_0 \cdot \left(T_{i+1,j}^{n} + T_{i-1,j}^{n} + T_{i,j+1}^{n} + T_{i,j-1}^{n}\right) \tag{9}$$

Where the $F_0 = \frac{D\Delta t}{h^2}$ In every time step we compute the values on the grid and add to the previous time value. Because we know the initial state of the system we know it later on.

The stability condition is:

$$\left(1 - \frac{4\Delta t \cdot D}{h^2}\right) \geq 0 \tag{10}$$

We have to be careful to choose the good parameters and differences or there is no steady state.

The first boundary condition means that the first values on the left side always remain the same. The second is more complicated:

$$\frac{\partial T}{\partial y} = j \tag{11}$$

Where j is constant. It is necessary to set virtual node outside the boundary to make the boundary node into interior node. On the bottom the update of the nodes follows the next equation:

$$T_{i,j}^{n+1} = T_{i,j}^{n} - 4F_0 \cdot T_{i,j}^{n} + F_0\left(T_{i,j-1}^{n} + 2 \cdot T_{i,j}^{n} + T_{i,j+1}^{n} - 2 \cdot h \cdot j\right) \tag{12}$$

With this implementation the given parameters can be find in the source-code.

3 Discussion

With the forward Euler method we can compute the state of the system with this boundary conditions. We get the same state as the initial state, because the differeces between the points remains zero, because the linear decreasing initial values.

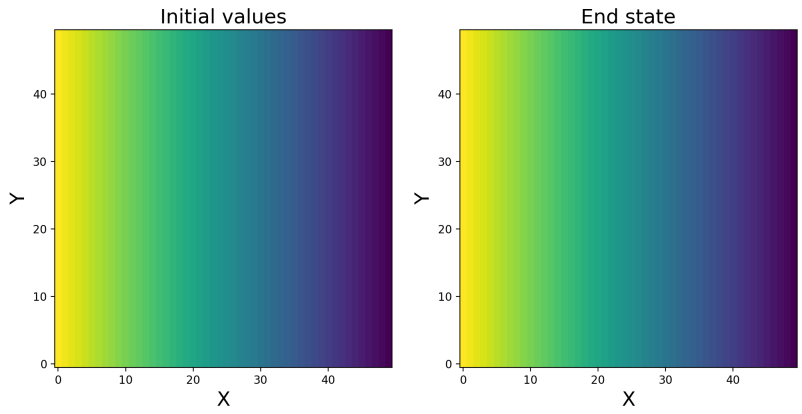

Figure 1: The initial values of the grid and the result after the computation.