

## Computer Lab Assignment 6

# Stationary States of the Heat Equation in 1D and 2D

No Matlab code will be provided for this computer lab. Instead you are asked to create all from scratch. Please save all files for future use.

### Stationary State of the 1D heat equation

Discretize the interval  $x=[0,L]$  in  $N$  sections using  $N+1$  equally spaced points in order to solve the 1D heat equation with the boundary conditions  $T(x=0)=100$  and  $T(x=L)=0$  and the initial condition  $T(x>0)=0$ . *Please note that Matlab only understands the notation  $T(i)$  but not  $T(x=0)$ ,  $T(x=L)$ , and  $T(x>0)$ .*

(1)  $N=25$  and  $L=10$  are good values to start but your code should work for any  $N$  and  $L$ . First you need to decide which Matlab index  $i$  corresponds to point  $x=0$  and which index corresponds to  $x=L$ . Then write two lines of code to set up a vector  $T$  to represent the specified initial conditions.

(2) Write the main loop that contains the different iterations to obtain the stationary state of the heat equation. (Please refer to the notes of the lecture 8 on PDEs and Heat Equation if something is unclear.) Inside the main loop introduce a second loop to update each element *in the interior* of the interval  $[0,L]$ . The update formula is

$$T_i^{new} = \frac{1}{2}[T_{i-1} + T_{i+1}]. \quad (*)$$

(3) Before entering the second loop introduce a second vector  $T^{new}=T$ . (This makes sure both vectors have the same size and the same values at the boundaries.) After completing the second loop, overwrite the temperature value in the vector  $T$  with the new values  $T^{new}$ .

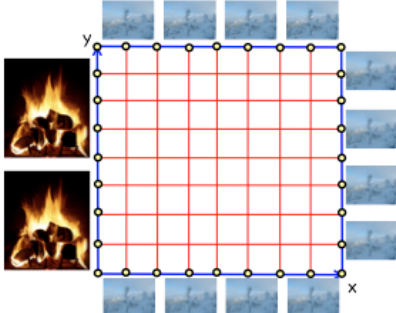
(4) Right below before closing the main loop, enter a command that plots the vector  $T$  followed by a `pause(0.1)` statement. Then run your code to see if it converges to the right stationary state: a straight line.

## Stationary State of the Heat Equation in 2D

Now we want to find the stationary temperature distribution for a 2D square with the following boundary conditions:

**Stationary state for  
different boundary conditions**

---


$$\frac{\partial T}{\partial t} = 0 \quad \frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2} = 0$$


Imagine you are renting a room of size  $L \times L$  that has three cold outside walls and one wall that is heated by your neighbor. (We are only considering heat conduction and neglect convection for the purpose of this exercise.) The outside temperature is kept fixed at  $0^\circ\text{C}$  and inside wall at  $100^\circ\text{C}$ . Divide the square into  $N \times N$  elements with  $N=51$ . This requires a 2D array  $T_{i,j}$  or in Matlab language  $T(i, j)$ .

Now convert the analytic form of the 2D heat equation given in the slide above into the discretized form using  $T_{i,j}$ . Derive the 2D version of equation (\*) given above.

$$T_{i,j}^{new} = \dots \left[ \dots \dots \dots \dots \right] \quad (**)$$

- (1) Write a code to set the boundary value of the 2D array  $T(i, j)$ .
- (2) Similar to part 1 write a main loop over iterations.
- (3) Now you need a nested pair of loop running over all inside points of the square shown above. At each point, you should set new value of  $T(i, j)$  according to equation (\*\*).
- (4) Now repeat the array copy statements from the 1D part section (4).
- (5) To plot the temperature distribution, please use
 

```
xx = 0:L/N:L;
yy = 0:L/N:L;
[X,Y] = meshgrid(yy,xx);
surf(X,Y,T);
view([+34.5 14]);
```

(6) At the very end of your code after e.g. 5000 iteration please make a contour plot using

```
pcolor(X,Y,T)
shading interp
hold on
contour(X,Y,T,30,'k');
```

(7) Run your new 2D code and see if it converges to a meaningful stationary temperature distribution.

(8) *Optional:* Instead of using a temporary array  $T^{new}$ , change the update formula (\*\*) so that you overwrite each element immediately:

$$T_{i,j} = \dots \left[ \dots \dots \dots \right]$$

Answer two questions:

- a) Does the code now converge faster or slower to the stationary temperature distribution?
- b) Does it make a difference in which order you update the elements? You could imagine interchanging the inside  $i$  and  $j$  loops, or running either one backwards.