Exercise 2

Simulating and visualising temperature distribution in a material

Lucas Piper - 14571773
Scientific Visualization and Virtual Reality
University of Amsterdam
14571773@uva.nl

ABSTRACT

This work proposes the simulation and visualization of a material's temperature distribution. The material at hand is the cross-section of a tube retrieved from Neal Wagner's cartoon "The Distillation of Human Knowledge" [Wagner(1978)]. The simulation consists of a continuous application of a two-dimensional Laplace equation using finite differences. Consequentially, the simulation grid resolution, the initial state and the end condition are also defined. Furthermore, this work aims to reflect upon how to visualize the temperature distribution in each simulation step.

KEYWORDS

Numeric simulation; Temperature distribution; Scientific visualization

1 INTRODUCTION

Most engineering fields are highly dependent on the behaviour of temperature within the components. As such, it's fundamental to know the range of temperatures that can be reached; whether or not temperature changes; if so, how it changes; and how is it distributed. Furthermore, it's often necessary to simulate these mechanisms, since this allows for a safer and less expensive way of acquiring an understanding on the matter.

Temperature distribution simulation can be achieved in multiple ways, namely through finite element methods (FEM), computer fluid dynamics (CFD), and, other numeric methods. When correctly implemented, these methods converge to similar solutions [Sławomir and Karol(2016)]. Moreover, it's also possible to represent a temperature distribution in distinct ways, more specifically, temperature distributions in a two-dimensional cross-sectional area.

Besides the possibility of using a third dimension to represent the temperature in a given point of the material (generating a continuous surface) [Min He and Feng(2021)], the use of color is also an important choice in these visualizations to avoid ambiguity.

Thus, with regards to the material provided in the aforementioned cartoon, it's necessary to reflect upon these decisions in order to answer the question *what is the best way to simulate and visualize temperature distribution?*

2 METHODS

The simulation is primarily implemented with Python and Numpy [Harris et al.(2020)]. The visualization and animation, on the other hand, are obtained with Matplotlib [Hunter(2007)]. This allowed for the representation of the material through an array that was later displayed as an animated 2D regular raster.

As suggested by Wagner, in order to reduce computational cost, the simulation is optimized by taking advantage of the vertical axis of symmetry. The horizontal axis is not used, because the boundary conditions of the problem are not the same in the bottom and the top of the cross-section. By using this optimization, the material and the boundary conditions can be represented by an array with dimensions $(9r+2) \times (\lceil 4.5r \rceil + 1)$, where r is the number of elements in the grid per inch of the material (described by the `resolution` variable in the code) and $\lceil x \rceil$ is the ceiling function evaluated at point x. For a resolution of $r = 13$, the simulation runs with an array of dimensions 119×60. Before displaying and animating the temperature distribution in the material, the array is first horizontally concatenated with a mirrored version in order to show the whole cross-sectional area. Note that for odd resolutions, it's necessary that the first column of the mirrored array is first dropped before the concatenation.

The simulation consists of a continuous application of a two-dimensional Laplace equation using finite differences on each element of the array [Belleman(2022)] that doesn't represent the boundary conditions. Nevertheless, the aforementioned optimization requires the formula to be different for elements in the last column of the array when the resolution r is even and when it's odd. As such, it's possible to identify two different 3 different cases:

$$T_{i,j}^{n+1} = \begin{cases} \frac{1}{4}(T_{i-1,j}^n + T_{i+1,j}^n + T_{i,j-1}^n + T_{i,j}^n) & C_1 \\ \frac{1}{4}(T_{i-1,j}^n + T_{i+1,j}^n + T_{i,j-1}^n + T_{i,j-1}^n) & C_2 \\ \frac{1}{4}(T_{i-1,j}^n + T_{i+1,j}^n + T_{i,j-1}^n + T_{i,j+1}^n) & otherwise \end{cases} \quad (1)$$

where:

- C_1: r is even and $j = \lceil 4.5r \rceil + 1$
- C_2: r is odd and $j = \lceil 4.5r \rceil + 1$

Besides reducing the simulation domain in half, a mask to easily identify which elements of the array should be changed in each iteration of the simulation and which ones represent boundary conditions (and should remain constant) is used.

Following once again the guidelines provided by the Wagner's comic, the initial condition is set to be uniform inside the material, the array elements are uniformly initialized with a temperature value roughly between 32°F and 212°F, i.e., 90°F (`init_temp`). The value itself is not as significant as other parameters, since it only affects the rate at which the simulation converges.

The metric used to represent the residuals of the simulation was the sum of absolute temperature differences between same elements of consecutive iterations (eq. 2). This metric does not represent the error in the full material, only on approximately half. However, it's still possible to identify a value for δ where the simulation no longer produces significant changes in the solution and in the

visualization. After some trial and error, the value found was about 10°F (`threshold`).

$$\delta = \sum_{i=1}^{9r+2} \sum_{j=1}^{\lceil 4.5r \rceil + 1} |T_{i,j}^{n+1} - T_{i,j}^{n}| \quad (2)$$

As for the visualization, after the array is initialized, it's necessary to create the Matplotlib figure to display it with all the relevant elements (axis, ticks, color map and color bar, labels and title). Then, at each iteration of the simulation, the array is updated, transformed into the full grid and updated in the figure. This pipeline can be represented in the following pseudo-code:

- Create array
- Create Matplotlib figure
- While $\delta >$ threshold:
  - Update array (eq. 1)
  - Get full array
  - Update figure

3 RESULTS

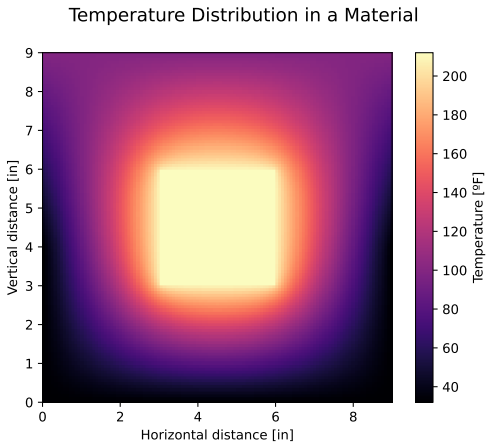

Figure 1: Visualization of the temperature distribution in a material (iteration 1275, δ=10.0)

Iteration	δ
1	7362.5
10	2326.5
100	615.5
500	84.8
1000	20.9
1275	10.0

Table 1

The final application allows the user to visualize the heat distribution in the material in every iteration while the simulation runs in the background in real-time. The application takes advantage of Matlab's `magma` color map to represent the temperature, given the its closeness with color maps used by some infrared thermography devices [Fries-Gaither(2009)].

The last iteration of the simulation is displayed in Figure 1. Furthermore, Figures 2, 3, 4, 5, 6 show the visualization animation during iteration no. 1, 10, 100, 500 and 1000, respectively. The residuals δ for these steps of the simulation can be examined in Table 1.

4 DISCUSSION

The visualization animation can provide some insight on how the heat is conducted throughout the material, namely the differences between the bottom and the top of the cross-sectional area, how boundary conditions can limit the heat flow and what role does the cross-section geometry play in the problem.

Furthermore, given the similarities in Figures 1 and 6, it becomes clear that the simulation is very close to converging when δ becomes less than the established threshold. It's possible to get additional understanding on how heat behaves in the material by tweaking the `init_temp`, `resolution` and `threshold` variables in the Python script. Finally, despite choosing the `magma` color map to represent temperature and prioritizing accuracy and familiarity with temperature representation, a different color map might also shed light on characteristics of the solution that were not as clear before.

5 CONCLUSION

Understanding temperature range and distribution inside components is fundamental in several major problems. Moreover, simulation and visualization are powerful tools that enable this understanding.

In this work, it's shown that a numeric method based on Laplace's equation and finite differences allows to simulate the temperature distribution inside a material, given the geometry and the boundary conditions. Using Python, Numpy and Matplotlib it's possible to run this simulation and, simultaneously, create a visualization that shows each simulation step.

REFERENCES

[Belleman(2022)] Robert Belleman. 2022. 1. Introduction. [PowerPoint Slides].

[Fries-Gaither(2009)] Jessica Fries-Gaither. 2009. Seeing Temperature Through Color. *Beyond Penguins and Polar Bears* (2009). https://beyondpenguins.ehe.osu.edu/issue/keeping-warm/seeing-temperature-through-color

[Harris et al.(2020)] Charles R. Harris, K. Jarrod Millman, Stéfan J. van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J. Smith, Robert Kern, Matti Picus, Stephan Hoyer, Marten H. van Kerkwijk, Matthew Brett, Allan Haldane, Jaime Fernández del Río, Mark Wiebe, Pearu Peterson, Pierre Gérard-Marchant, Kevin Sheppard, Tyler Reddy, Warren Weckesser, Hameer Abbasi, Christoph Gohlke, and Travis E. Oliphant. 2020. Array programming with NumPy. *Nature* 585, 7825 (Sept. 2020), 357–362. https://doi.org/10.1038/s41586-020-2649-2

[Hunter(2007)] J. D. Hunter. 2007. Matplotlib: A 2D graphics environment. *Computing in Science & Engineering* 9, 3 (2007), 90–95. https://doi.org/10.1109/MCSE.2007.55

[Min He and Feng(2021)] Wenpei Zheng Min He, Laibin Zhang and Yijing Feng. 2021. Investigation on a new inducer of pulsed eddy current thermography. *AIP Advances* (2021). https://doi.org/10.1063/1.4963894

[Sławomir and Karol(2016)] Grądziel Sławomir and Majewski Karol. 2016. Simulation of Temperature Distribution and Heat Transfer Coefficient in Internally Ribbed Tubes. *Procedia Engineering* 157 (2016), 44–49. https://doi.org/10.1016/j.proeng.2016.08.336 Selected Papers from IX International Conference on Computational Heat and Mass Transfer (ICCHMT2016).

[Wagner(1978)] Neal Wagner. 1978. The Distillation of Human Knowledge. In *A FORTRAN Coloring Book*, Roger E. Kaufman (Ed.). MIT Press, MIT.

APPENDIX

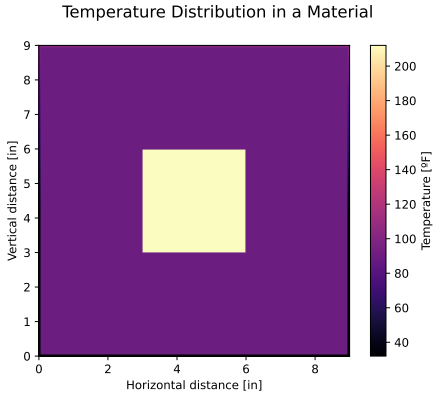

Figure 2: Visualization of the temperature distribution in a material (iteration 1, δ=7362.5)

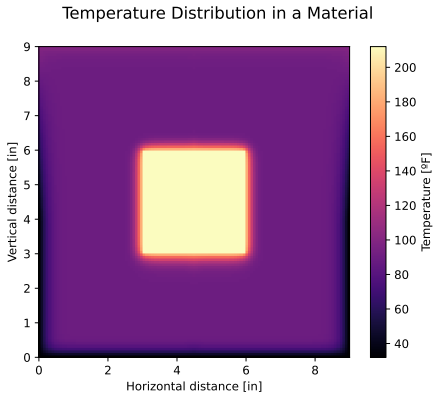

Figure 3: Visualization of the temperature distribution in a material (iteration 10, δ=2326.5)

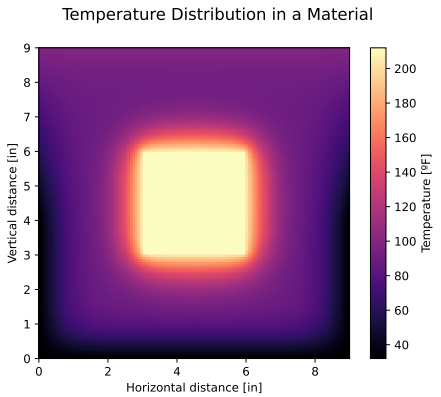

Figure 4: Visualization of the temperature distribution in a material (iteration 100, δ=615.5)

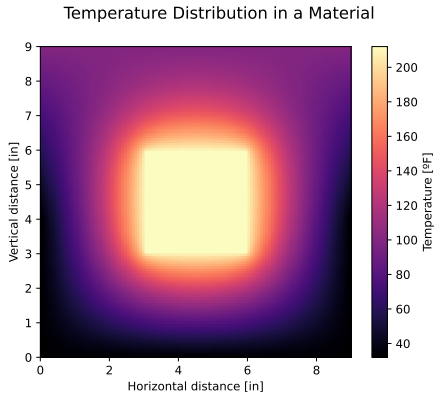

Figure 5: Visualization of the temperature distribution in a material (iteration 500, δ=84.8)

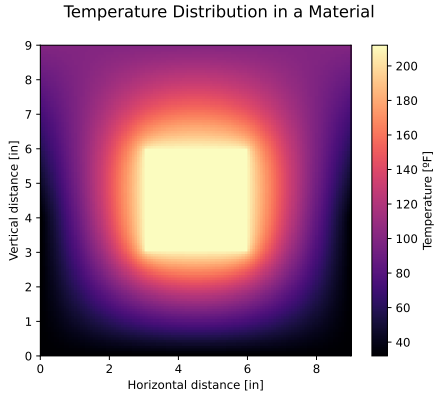

Figure 6: Visualization of the temperature distribution in a material (iteration 1000, δ=20.9)

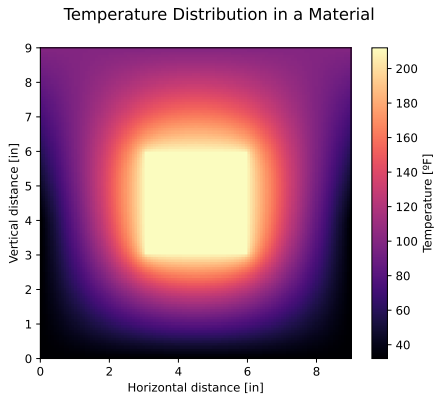

Figure 7: Visualization of the temperature distribution in a material (iteration 1275, δ=10.0)