PHYM004 Final Project
Project F: Thermal Diffusion in 1-Dimension

Tom Joshi-Cale

25th March 2021

Abstract

The implicit Backwards Euler method was used to simulate temperature diffusion through either a spherically or cylindrically symmetric system by solving the 1-D temperature diffusion equation. A code was created which could model these systems with customisable boundary conditions, giving the temperature distribution at various 'snapshot' times. The code was found to match with literature for the cylindrical case, but predicts a 25% longer cooking time than literature for the spherical case.

1 Introduction

Figure 1: Three decorated eggs. From left to right: The Increggible Hulk, Keggvin the Minion, and Clark Keggt.

Eggs have a wide range of uses - incubating young, decoration for special occasions (Figure 1), and perhaps most commonly as a protein rich food. There are over 100 ways to cook an egg [1], and for each method, a 'perfectly cooked' egg needs to be a certain temperature. The difficulty in cooking eggs lies in knowing how long the egg needs to be cooked for, and to determine this cooking time a model of temperature diffusion throughout the egg would be useful. Such a model could also be easily modified for other scenarios, such as modelling the environmental temperature change caused by the presence of a radioactively decaying nuclear rod. An equation has been derived by C.D.H. Williams [2] (henceforth Williams) to calculate the cooking time of an egg, and a method for modelling the environmental temperature change caused by a decaying rod is given in L. Olsen-Kettle's *Numerical Solutions of Partial Differential Equations* [3] (henceforth Olsen-Kettle). The aim of this project is to create a code that can use this prior work to calculate either the cooking time of an egg of inputted properties, or how long a radioactive rod of certain properties would take to cool.

2 Background Theory

The heat diffusion equation in three dimensions is given by

$$\frac{1}{\kappa}\frac{\partial T(x,y,z,t)}{\partial t} = \nabla^2 T(x,y,z,t) + S(x,y,z,t), \tag{1}$$

where $T(r,t)$ is temperature in Kelvin as a function of position and time, κ is the thermal diffusivity of the material in cm^2 s^{-1} and $S(r,t)$ is a Source term in K cm^{-2}, representing any internal heating of the system.

2.1 Reducing to a 1-D problem

The spherical mode required to model the egg and the cylindrical mode for modelling the nuclear rod can be solved in the same manner once they are reduced to a 1D problem by exploiting symmetry.

Cylindrical Symmetry

For the nuclear rod, the problem has circular symmetry, so a 2-D problem in x and y can be reduced to a 1-D problem in the radial direction. The Laplacian operator ∇^2 can be re-written in cylindrical co-ordinates as

$$\nabla^2 = \frac{\partial^2}{\partial r^2} + \frac{1}{r}\frac{\partial}{\partial r} + \frac{1}{r^2}\frac{\partial^2}{\partial \phi^2}$$

but due to the circular symmetry, $\frac{\partial^2}{\partial \phi^2} = 0$. With some re-arrangement, Equation 1 can then be written as

$$\frac{1}{\kappa}\frac{\partial T(r,t)}{\partial t} - \frac{\partial^2 T(r,t)}{\partial r} - \frac{1}{r}\frac{\partial T(r,t)}{\partial r} = S(r,t) \tag{2}$$

and the Thermal Diffusion equation has been reduced to a 1-D radial equation.

Spherical Symmetry

In the spherical case, the Laplacian operator can be re-written in spherical co-ordinates as

$$\frac{1}{r^2}\frac{\partial}{\partial r}\left(r^2\frac{\partial}{\partial r}\right) + \frac{1}{r^2\sin\theta}\frac{\partial}{\partial\theta}\left(\sin\theta\frac{\partial}{\partial\theta}\right) + \frac{1}{r^2\sin^2\theta}\frac{\partial^2}{\partial\phi^2}.$$

Again, due to the symmetry of this case, the $\frac{\partial}{\partial\theta}$ and $\frac{\partial}{\partial\phi}$ terms will always be zero, so Equation 1 becomes

$$\frac{1}{\kappa}\frac{\partial T}{\partial t} - \frac{1}{r^2}\frac{\partial}{\partial r}\left(r^2\frac{\partial T}{\partial r}\right) = S(r,t),$$

which can have its second term expanded out to obtain

$$\frac{1}{\kappa}\frac{\partial T}{\partial t} - \frac{\partial^2 T}{\partial r^2} - \frac{2}{r}\frac{\partial T}{\partial r} = S(r,t), \tag{3}$$

which bares clear similarity to Equation 2 with the exception of the factor 2 in the third term. These two equations however can be solved in the same manner; by discretising the problem in space and time then solving using the Backwards Euler Method.

2.2 Discretising the Problem

To solve Equations 2 & 3 we must change the problem from a continuous problem to a discrete problem, with spacial and time steps. For a full explanation of how to discretise the problem, see Olsen-Kettle ch. 5.3.1. The discretised 1-D thermal diffusion equation can be written as

$$T^{k+1}_{j+1}\left[-s - \frac{s}{2j}\right] + T^{k+1}_{j-1}\left[-s + \frac{s}{2j}\right] + T^{k+1}_{j}\left[1 + 2s\right] = T^k_j + \kappa\Delta t S^k_j,$$

where T^k_j is a shorthand for $T(r = j\Delta r, t = k\Delta t)$, $j\Delta r$ is the j-th spacial step, $k\Delta t$ is the k-th time step and s is a dimensionless 'gain parameter' (for the case of the nuclear rod, $s = \frac{\kappa\Delta t}{(\Delta r)^2}$). This equation can be written in matrix form as

$$\underline{\mathbf{T}}^{k+1} = \underline{\underline{\mathbf{A}}}^{-1}\left[\underline{\mathbf{T}}^k + \kappa\Delta t\underline{\mathbf{S}}^k + \underline{\mathbf{b}}\right], \tag{4}$$

which is the equation to be solved by the code, where $\underline{\mathbf{T}}^{k+1}$ is a vector of Temperature at the next timestep, $\underline{\mathbf{T}}^k$ is a vector of temperature at the current time-step, $\underline{\mathbf{S}}^k$ is the source heating at the current time-step, $\underline{\underline{\mathbf{A}}}$ is a tri-diagonal matrix representing the system of simultaneous equations to be solved, and $\underline{\mathbf{b}}$ is an implementation of the boundary conditions for each problem.

Boundary Conditions for the Nuclear Rod

When solving Equation 2, a singularity for the $r = 0$ case should become immediately apparent. This problem is solved by implementing a Neumann boundary condition at $r = 0$, since temperature cannot flow into the $r = 0$ region,

$$\left.\frac{\partial T}{\partial r}\right|_{(r=0,\ t)} = 0,$$

which, after discretisation, becomes

$$\frac{T^k_1 - T^k_0}{\Delta t} = 0 \Rightarrow T^k_0 \approx T^k_1,$$

so we can make the assumption within the model that at all timesteps, the temperature at $r = 0$ is equal to the temperature at $r = 1$.

The other boundary condition for the nuclear rod ensures that at some critical distance r_c from the centre of the rod, the temperature will not change from the initial environmental temperature. Taking $r = r_c$ as the final spacial step of the simulation, and the intial environmental temperature as 300 K, we can write

$$T_{r_c}^k = 300\text{K}.$$

Applying these boundary conditions into Equation 4 means that the vector $\underline{\mathbf{b}}$ becomes

$$\underline{\mathbf{b}} = \begin{pmatrix} (-s + \frac{s}{2})T_0^{k+1} \\ 0 \\ \vdots \\ 0 \\ (-s - \frac{s}{2(n+1)})T_{r_c}^{k+1} \end{pmatrix} = \begin{pmatrix} (-\frac{s}{2})T_1^{k+1} \\ 0 \\ \vdots \\ 0 \\ (-s - \frac{s}{2(n+1)}) \quad 300 \end{pmatrix}.$$

For a more complete demonstration of how to calculate the boundary conditions for a Nuclear rod, consult Olsen-Kettle.

Boundary Conditions for an egg

To model the egg, the first assumption we make is that the egg is a spherical object, consisting of concentric spheres of different material (egg yolk, albumen etc.). This allows us to reduce the problem to 1 dimension using spherical symmetry.

The difference of a factor 2 in the third term between Equations 2 & 3 mean the boundary condition vector $\underline{\mathbf{b}}$ becomes:

$$\underline{\mathbf{b}} = \begin{pmatrix} (-s + \frac{s}{1})T_0^{k+1} \\ 0 \\ \vdots \\ 0 \\ (-s + \frac{s}{(n+1)})T_{r_c}^{k} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ (-s + \frac{s}{(n+1)})T_{\text{water}} \end{pmatrix}.$$

Source Heating

Some modelled systems may include a source of internal heating, while others will not. For an egg submerged in hot water, the egg will not generate any internal heat - but for the nuclear rod the radioactivity causes the rod to heat up both itself and its surroundings. This internal heating is dealt with by the $\underline{\mathbf{S}}^k$ term in Equation 4. For a system like the egg without any internal heating, this term is 0 at all points, but for a more complicated system like the rod, the Source term needs to be calculated.

Mathematically, the source term for the radioactive decay of the rod is given in Olsen-Kettle as:

$$S(r,t) = \begin{cases} T_{rod} e^{-t/\tau_0}/a^2 & \text{for } r \leq a \\ 0 & \text{elsewhere,} \end{cases}$$

where a is the radius of the rod, t is the time since simulation start, T_{rod} is the temperature difference of the rod above its surroundings at $t = 0$ and τ_0 is the half-life of the rod.

3 Method of Solution

The C program, `Thermal.c`, runs in two separate modes, spherical or cylindrical, which are selected by the user at run-time using command line arguments. The other conditions for the problem, such as the simulation time and composition of the simulated system, are set using `static const` values defined at the top of the code.

When the user selects a 'mode' to run the code in, a `switch` statement is used to set up the problem in the desired mode. Each `case` sets up the required simulation length and spacial resolution, the simulation time and time resolution, and all the required conditions to set up and solve Equation 4. This allows the inversion of $\underline{\underline{\mathbf{A}}}$ and the solving of Equation 4 to take place outside the `switch` statement.

Since the desired plot only requires storage of the temperature at each required time-step, a `gsl matrix` (see Section 3.1) is created which has enough memory for the Temperature at each position calculated at each snapshot time. The required data is written to the correct matrix column as it is calculated, so that it can be forgotten when the next time-step is calculated, so as to be as economical as possible with memory.

3.1 Vectors and Matrices

Since the solution to Equation 4 requires inverting the matrix $\underline{\underline{\mathbf{A}}}$, the GNU Scientific Library (GSL) Linear Algebra header files [4] were used, and $\underline{\underline{\mathbf{A}}}$ stored as a `gsl matrix` to more easily invert it using the LU Decomposition method outlined in the GSL documentation. Since the `gsl matrix` format needed to be used, it was also decided that all arrays within the code, such as that used to store each spacial step, would be stored in `gsl vector` format to maximise consistency and coherency throughout the code. To ensure that this didn't lead to a reduction of performance, a copy of the code was created which used arrays instead of vectors, and the execution times for both were compared for different values of spacial resolution (different numbers of elements in the array/vector). The comparison can be seen in Figure 2, which shows that there is no significant increase in execution time associated with using `gsl vector` data type over an array, which allows for the vector type to be used for consistency.

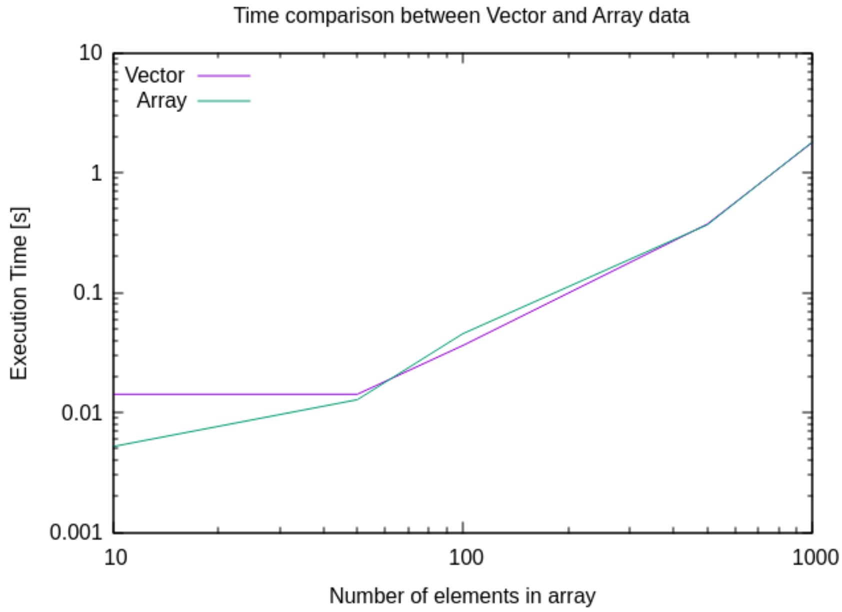
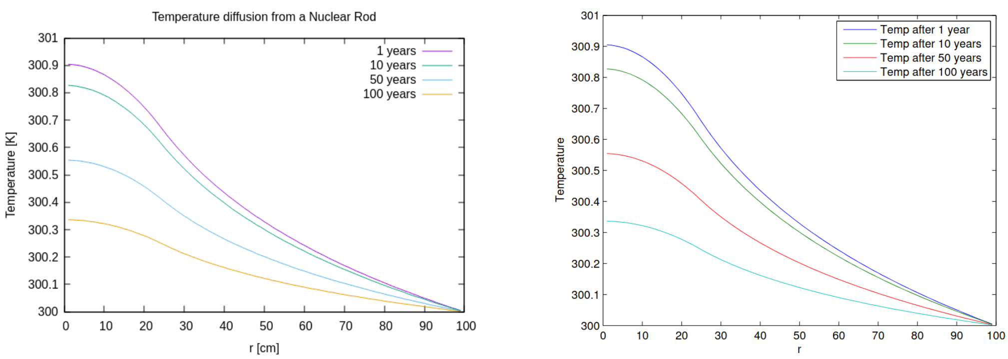


Figure 2: A comparison of execution time between using arrays and `gsl vector` datatypes throughout the code. This figure shows no clear advantage to using one or the other.

4 Results and Discussion

4.1 Nuclear Rod

Running the program for a nuclear rod with a thermal conductivity of 0.634 cm^2 s^{-1} (which matches the thermal conductivity used in Olsen-Kettle) and comparing against the graph produced by Olsen-Kettle as in Figure 3 shows us that the two codes produce a very similar output, comparison between the values produced by the two techniques tell us that the largest discrepency between `Thermal.c` and Olsen-Kettle is 0.0004K, telling us that the two codes produce functionally the same output. This output shows the rod starting hot and cooling over



(a) **(b)**

Figure 3: A comparison between **(a)** the temperature diffusion from the nuclear rod outputted by `Thermal.c` (left) and **(b)** from Olsen-Kettle's `Nuclear.m` (right). The largest discrepency between the two outputs is 0.0004 K

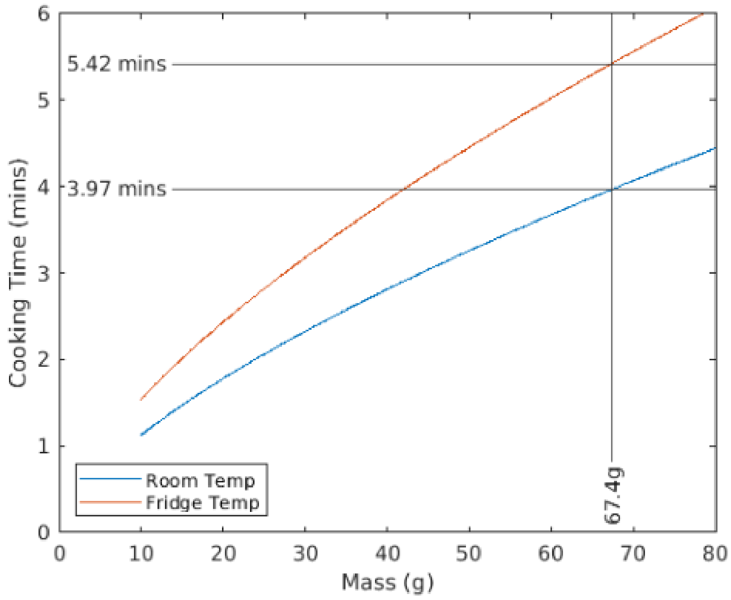


Figure 4: Plotted using Equation 15 from Williams, this graph shows the calculated time taken to cook eggs of different masses from a starting temperature of either 20^{o}C (blue) or 4^{o}C (orange).

time, as expected. The plot shows that even after 100 years, the nuclear rod is still heating its environment, but to only a third of the initial temperature. Indeed, as the half-life of the rod is 100 years, after 100 years we would expect the activity of the rod to have dropped by a half, and so the source heating will have dropped. It is of interest that the temperature the rod is heated to is a third of its initial temperature, and not a half as may be expected. A possible explanation for this is that the rod is not just operating under its own heating, but is also being cooled by the environment, which could be an explanation as to why the temperature in the rod is lower than expected.

4.2 Boiling Egg

The egg modelled by the program had a radius of 2.5cm and comprised of 2 concentric circles of egg yolk and egg white ($\kappa_{\text{yolk}} = 1.22 \times 10^{-3}$ cm^2 s^{-1}, $\kappa_{\text{white}} = 1.40 \times 10^{-3}$ cm^2 s^{-1}, calculated from values given in Williams), where the yolk-white boundary occurs at 69% of the radius of the egg (making the yolk 33% of the volume of the egg), or 1.73cm. These values correspond to an egg of mass 67.4 grams, which is about the same as a European Size 1 egg. Figure 4 suggests that an egg with a mass of 67.4g coming from the fridge (at 4^{o}C) should take 5.42 minutes to cook (where cooking is defined as the yolk-white boundary reaching a temperature of 63^{o}C), and Figure 5 shows that for the simulated model the egg is 'cooked' after between 6 and 7.5 minutes of cooking minutes of cooking. This suggests that the program is overestimating heating times compared to literature by a factor of ~1.25, or by 25%. It should be noted that

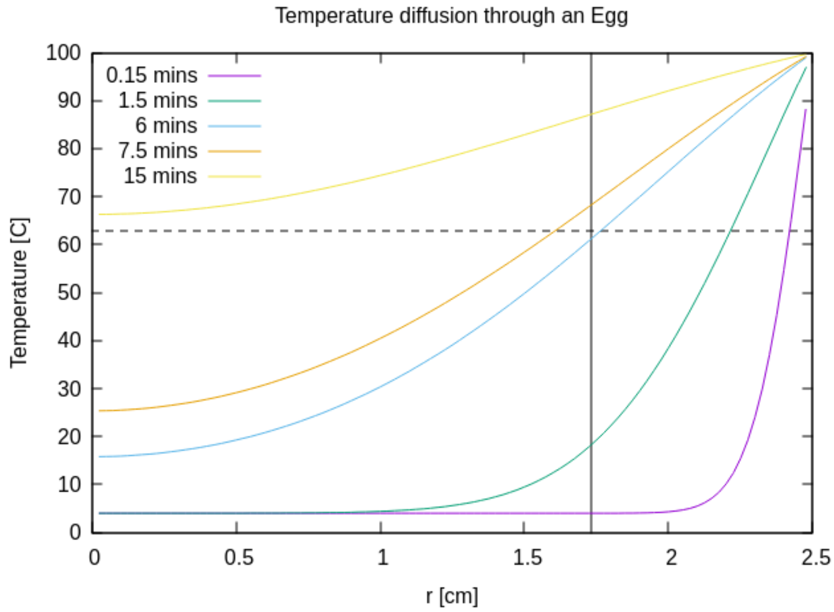


Figure 5: The temperature diffusion through an egg, initially at 4°C, immersed in boiling water. The vertical black line represents the Yolk-White boundary, which occurs at $r = 0.1.73$ cm, and the dotted horizontal line represents $T = 63^o$C, or the temperature at which we call the egg 'cooked'.

this is a comparison against an analytical model, not against experimentally obtained results, so it is unclear which of the two models is most accurate.

Williams also gives an equation for calculating the temperature throughout the egg at various times, which has been plotted against the output of `Thermal.c` in Figure 6. The Williams equation is arrived at through analytically solving the Temperature Diffusion equation, and truncating the infinite series recieved, whereas `Thermal.c` solves the equation numerically. This means that both methods only give approximate values, and without physical experimentation to compare to it is unclear which method gives the most accurate predictions.

The model for the egg could also be brought closer to reality by modelling the eggs as ovoids rather than spheres, and also by implementing the latent heat it would take to change the albumen of the egg from a liquid state to a gel.

Conclusions

`Thermal.c` was written to perform simulations of temperature diffusion through systems with cylindrical symmetry (nuclear rod) and spherical symmetry (boiling an egg). The outputs were compared to literature, and for the rod were found to match with the literature, but for the egg it was found to predict a longer cooking time by a factor of 1.25. It was also shown that `Thermal.c` underpredicts the temperature at longer times and radii compared to literature for the rod.

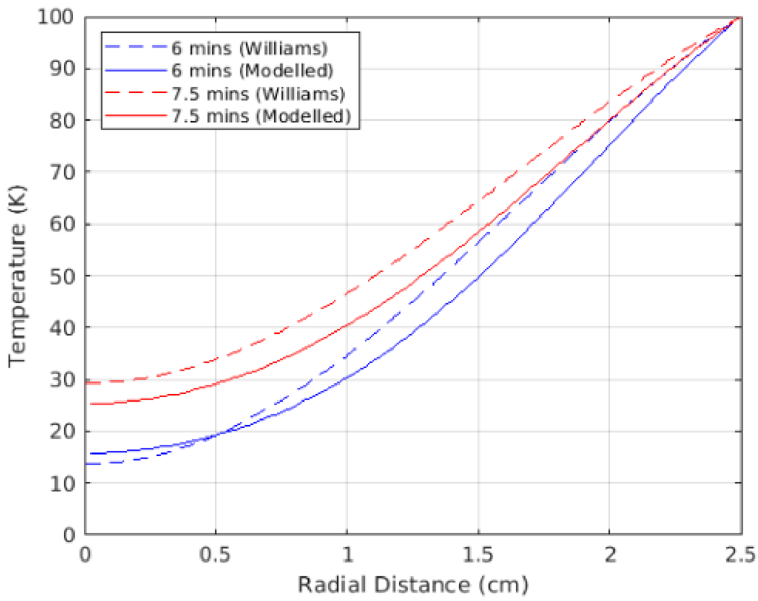


Figure 6: A comparison between the temperature calculated numerically by `Thermal.c`, and the temperature calculated analytically in Williams. It can be seen that the Williams equation predicts a higher temperature at longer times at larger radii than `Thermal.c`.

References

[1] J. Charles, ed., *100 Ways with Eggs: Boiled, baked, fried, scrambled and more!* Ryland Peters & Small, 2016.

[2] C. D. H. Williams, "Boiling an egg." `https://newton.ex.ac.uk/teaching/CDHW/egg/CW061201-1.pdf`, 2006. [Online; accessed: 23-March-2021].

[3] L. Olsen-Kettle, *Numerical solution of partial differential equations and code*, ch. 5.3.1. Earth Systems Science Computational Centre Publications, 2011.

[4] M. Galassi, *GNU scientific library reference manual*, ch. 8, 14. Network Theory Ltd., 3rd ed., 2009.