Numerical Computations of Partial Differential Equations to Solve the Diffusion equation

Ilse Kuperus, Fredrik Leiros Nilsen, Fevzi Sankaya
(Dated: December 19, 2019)

In this article we explore and implement numerical computations of partial differential equations to solve the diffusion equation. First we find an analytical solution to the 1 D diffusion equation for given initial and boundary conditions. Then we numerically compute the 1 D case using three algorithms based on the Euler family of methods. We use an explicit scheme with the forward Euler method, implicit scheme with the backward Euler method and the Crank-Nicolson method which combines these two. We see that as $t \to \infty$ the 1 D solutions correspond with our analytical solution, the density $u(x,t)$ being a linear function. Afterwards, we extend our code to the 2 D diffusion case, using the same principles as we did in 1 D for the explicit scheme. For the 2 D implicit scheme we set up a system of linear equations which is then solved with Jacobi's iterative method. These methods allow us to make 2 D contour plots and animations of diffusion. We also find that parallelzing the 2 D code with OpenMP has an insignificant effect on our code run time. Lastly we apply our methods to a geological problem, namely describing the heat flow in the lithosphere saturated with radioactive elements. Here we find that the added radioactive enrichment gives an increase of temperature in the mantle of around 200°C depending on the depth, but with decay this falls to around 100°C after 3 Gyr.

INTRODUCTION

Partial differential equations are quite a common occurrence in physics, they are a method of describing a physical system that depend on multiple variables. In general, differential equations are used for many different physical systems. Solving differential equations analytically, especially partial differential equations is only possible in a few cases under certain conditions. Therefore the use of numerical methods is very useful as they allow us to solve many more equations than can be done analytically. The main principle behind solving these equations is discretization. The derivatives are solved by the use of iterative methods. The diffusion equation is a partial differential equation that describes the evolution of a physical property in time and space. It describes quantities such as the density of a gas in space over a given time, by this we can for instance simulate how a gas is distributed in an area over time.

In this article we will solve the diffusion equation numerically by using different iterative methods from the Euler family of methods. First we will find an analytical solution to the diffusion equation in 1 dimension. Afterwards, we will implement our numerical methods in 1 dimension, first with an explicit scheme, namely the forward Euler method. This however has stability drawbacks due to constraints placed on the step length we choose for the time and positions. Therefore, we will also look at the more numerically stable implicit schemes, such as the backwards Euler method, and lastly Crank-Nicolson method which combine both methods. We shall also extend our code for a 2 dimensional case using the explicit and implicit scheme. By parallelizing our algorithms we will try to achieve a speed up of code run time. Lastly we apply our methods to a geological case with a proposed subduction zone with radioactive enrichment in the mantle. Here we compute the temperature distribution in the lithosphere, and how this changes with extra radioactive enrichment.

For reproducibility the numerical code used for the results, plots and animations referenced in this article can be found on the website [1].

METHOD

Analytical Solution

In the case of the diffusion equation, we do have an analytical solution of the system. To find this in one dimension, we start with a separation of variables, setting

$$u(x,t) = F(x)G(t). \tag{1}$$

We then see that we can rewrite equation (7) as

$$F''(x)G(t) = F(x)G'(t) \implies \frac{F''(x)}{F(x)} = \frac{G'(t)}{G(t)}. \tag{2}$$

Since the right hand side is only dependent on t while the left is only dependent on x, $\frac{F''(x)}{F(x)}$ must be constant (while t is kept the same), and similarly for $\frac{G'(t)}{G(t)}$. Since these two sides are equal, we can then set them both to a constant, which we will call λ^2, so we get $F''(x) + \lambda^2 F(x) = 0$ and $G'(t) + \lambda^2 G(t) = 0$. We then have one first and one second order linear differential equation, with known solutions given by

$$\begin{aligned} F(x) &= C_1 \sin(\lambda x) + C_2 \cos(\lambda x) \\ G(t) &= C_3 \exp\left(-\lambda^2 t\right). \end{aligned} \tag{3}$$

We also get the static solution where $G'(t) = 0$ and $F''(x) = 0$ (so a linear solution). To continue from this solution, one has to know the boundary and initial conditions of the system. From the boundary conditions one will get a set of values for λ, called eigenvalues. In our case, we have $u(0,t) = 0$ and $u(L,t) = 1$ (with $x \in [0, L]$). This then gives one linear solution on the form $u(x,t) = \frac{x}{L}$. Since the full solution will be a sum of possible solution, we chose to rewrite our expression as $u(x,t) = \frac{x}{L} + F(x)G(t)$. We then see that $F(0) = F(1) = 0$. For this to be true, we see that $C_2 = 0$, and $\lambda = \frac{k\pi}{L}$, with $k = 1, 2, 3, \ldots$. The complete solution will then be a sum, on

the form

$$u(x,t) = \frac{x}{L} + \sum_{k=1}^{\infty} A_k \sin\left(\frac{k\pi}{L}x\right) e^{-(\frac{k\pi}{L})^2 t}. \tag{4}$$

To find the A_k-coefficients, we utilize that the last term of the solution take the form of a Fourier series. Since the initial conditions are $u(x,0) = 0$ for $x < L$, we see that the last term must equal $-\frac{x}{L}$. We can then find the coefficients by solving the integrals given by

$$A_k = \frac{2}{L}\int_0^L -\frac{x}{L}\sin\left(\frac{k\pi}{L}x\right) dx. \tag{5}$$

This then has the solution $A_k = 2\frac{1}{k\pi}\cos(k\pi) = 2\frac{(-1)^k}{k\pi}$. So we then get the full expression

$$u(x,t) = \frac{x}{L} + 2\sum_{k=1}^{\infty} \frac{(-1)^k}{k\pi}\sin\left(\frac{k\pi}{L}x\right) e^{-(\frac{k\pi}{L})^2 t} \tag{6}$$

Solving Partial Differential Equations Numerically

The diffusion equation describes the temporal evolution of a density u which can represents a quantity such as particle density, energy density or temperature. The scaled diffusion equation in one dimension is given by equation (7),

$$\frac{\partial^2 u(x,t)}{\partial x^2} = \frac{\partial u(x,t)}{\partial t} \tag{7}$$

In order to solve this equation numerically we need to discretize by Taylor expanding and approximating the derivatives using discrete steps in time and space. In addition, this partial differential equation needs two boundary conditions and one initial condition for it to be possible to solve. In our case we have the boundary conditions $u(0,t) = 0$ and $u(L,t) = 1$ and the initial conditions $u(x,0) = 0$ for $0 < x < L$. This can be interpreted as applying constant heat to one end of a rod, where our goal is to describe how the heat get distributed in space and time.

The diffusion equation can be numerically solved using different methods. There are three main methods which we will be implementing the forward Euler, the backward Euler and the Crank-Nicolsen method[2]. The first mentioned method is an explicit scheme, meaning that we iterate using what we know at the current step. The last two methods are implicit which involve solving an equation based on the current and next step of an iteration of the system. All three of the methods require discretization. First we discretize the length x (1 spatial dimension), We write this as $x = x_i = x_0 + i\Delta x\ \Delta, \quad x = \frac{x_n - x_0}{n} = \frac{1}{n}$ and time is discretized similarly $t = t_j = t_0 + j\Delta t, \quad \Delta t = \frac{t_m - t_0}{m} = \frac{t_m}{m}$. Here n and m are the number of steps (iterations) for the length and time respectively. By using Taylor expansion for the double derivative of x we get,

$$\frac{\partial^2 u}{\partial x^2} = \frac{u(x_i + \Delta x, t_j) + u(x_i - \Delta x, t_j) - 2u(x_i, t_i)}{\Delta x^2} + O(\Delta x^2)$$

$$\implies \frac{\partial^2 u}{\partial x^2} = \frac{u_{i+1,j} + u_{i-1,j} - 2u_{i,j}}{\Delta x^2} + O(\Delta x^2).$$

Similarly we use the derivative of t in Taylor expansion this expression depends on whether we use backwards or forwards Euler to find the expression. For forwards Euler we get

$$\frac{\partial u}{\partial t} = \frac{u(x_i, t_j + \Delta t) - u(x_i, t_j)}{\Delta t} + O(\Delta t)$$

$$\implies \frac{\partial u}{\partial t} = \frac{u_{i,j+1} - u_{i,j}}{\Delta t} + O(\Delta t)$$

For backwards Euler we get,

$$\frac{\partial u}{\partial t} = \frac{u(x_i, t_i) - u(x_i, t_i - \Delta t)}{\Delta t} + O(\Delta t)$$

$$\implies \frac{\partial u}{\partial t} = \frac{u_{i,j} - u_{i,j-1}}{\Delta t} + O(\Delta t)$$

For the explicit scheme we use forward Euler, this means that we have,

$$\frac{u_{i+1,j} + u_{i-1,j} - 2u_{i,j}}{\Delta x^2} + O(\Delta x^2) = \frac{u_{i,j+1} - u_{i,j}}{\Delta t} + O(\Delta t)$$

We want to find u for the next time step, meaning that we want to find $u_{i,j+1}$, from the equation we can see that this gives us the algorithm for our explicit scheme:

$$u_{i,j+1} = u_{i,j}(1 - 2\alpha) + \alpha(u_{i+1,j} + u_{i-1,j}), \tag{8}$$

where $\alpha = \frac{\Delta t}{\Delta x^2}$. The explicit scheme has stable solutions only if $\alpha = \frac{\Delta t}{\Delta x^2} \leq 1/2$, this means that if we want a good precision for the space steps, meaning small Δx, the time steps has to be chosen so that $\Delta t \leq 1/2\Delta x^2$. This means that the time step will be very small for good precision. What we also see for the explicit scheme is that we have truncation errors for both the time and the space step size, $O(\Delta x^2)$ and $O(\Delta t)$.

The implicit scheme uses backwards Euler.

$$\frac{u_{i+1,j} + u_{i-1,j} - 2u_{i,j}}{\Delta x^2} + O(\Delta x^2) = \frac{u_{i,j} - u_{i,j-1}}{\Delta t} + O(\Delta t)$$

The implicit scheme has truncation errors that are the same as the explicit scheme $O(\Delta t)$ and $O(\Delta x^2)$, however in contrast to the explicit scheme, the implicit scheme is stable for all Δx and Δt. Since the only difference now is that we use the backward Euler method we get the following algorithm for the implicit scheme [2],

$$u_{i,j-1} = -\alpha u_{i-1,j} + (1 - 2\alpha)u_{i,j} - \alpha u_{i+1,j}, \tag{9}$$

Which can be rewritten as a matrix and two vectors,

$$\hat{A} = \begin{bmatrix} 1+2\alpha & -\alpha & 0 & \dots & 0 \\ -\alpha & 1+2\alpha & -\alpha & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & 1+2\alpha \end{bmatrix}$$

$$\hat{V}_j = \begin{bmatrix} u_{1,j} \\ u_{2,j} \\ \vdots \\ u_{n,j} \end{bmatrix} \hat{V}_{j-1} = \begin{bmatrix} u_{1,j-1} \\ u_{2,j-1} \\ \vdots \\ u_{n,j-1} \end{bmatrix}$$

Then equation 9 can be written on the form $\hat{C}\hat{V}_j = \hat{V}_{j-1}$. To solve this for $\hat{V}_j$ we will implement the method for solving an matrix-vector equation using a tri-diagonal matrix [3].

The last method for solving the partial differential equation in 1 D is the Crank-Nicolson scheme. This method combines the explicit and the implicit scheme into a general approach. The Crank-Nicolson scheme starts with the forward Euler approach and then we Taylor expand $u(x, t+\Delta t)$, $u(x+\Delta x, t)$ and $u(x-\Delta x, t)$.

$$u(x+\Delta x,t) = u(x,t) + \frac{\partial u(x,t)}{\partial x}\Delta x + \frac{\partial^2 u(x,t)}{\partial x^2}\Delta x^2 + O(\Delta x^3)$$

$$u(x-\Delta x,t) = u(x,t) - \frac{\partial u(x,t)}{\partial x}\Delta x + \frac{\partial^2 u(x,t)}{\partial x^2}\Delta x^2 + O(\Delta x^3)$$

$$u(x,t+\Delta t) = u(x,t) + \frac{\partial u(x,t)}{\partial t}\Delta t + O(\Delta t^2)$$

In addition we need to Taylor expand $u(x+\Delta x, t+\Delta t)$ and $u(x-\Delta x, t+\Delta t)$ around $t+\Delta t/2$. By doing these steps we get that the equation for the Crank-Nicolson scheme is

$$-\alpha u_{i-1,j} + (2+2\alpha)u_{i,j} - \alpha u_{i+1,j} = \alpha u_{i-1,j-1} + (2-2\alpha)u_{i,j-1} + \alpha u_{i+1,j-1}$$

Where again $\alpha = \frac{\Delta t}{\Delta x^2}$. This scheme is stable for all Δt and Δx and has truncation errors $O(\Delta t^2)$ and $O(\Delta x^2)$. Thus we see that the explicit and implicit schemes have the same errors, but the explicit scheme has a limitation due to the stability requirement that the implicit scheme does not have, the Crank-Nicolson scheme has a smaller error than the two others and is stable for all steps.

To sum up, see table I for a quick rundown of each method.

Scheme	Truncation error	stability criteria
Explicit	$O(\Delta t)$ and $O(\Delta x^2)$	stable for $\alpha \leq 1/2$
Implicit	$O(\Delta t)$ and $O(\Delta x^2)$	stable for all $\Delta x, \Delta t$
Crank-Nicolson	$O(\Delta t^2)$ and $O(\Delta x^2)$	stable for all $\Delta x, \Delta t$

Table I: Table showing the truncation errors and stability criteria for the different schemes we have used.

Moving To 2 Dimensions

We can apply the same principles described in the previous section to make an algorithm for diffusion in 2 D. Our diffusion equation now looks like equation 10.

$$\frac{\partial u(x,y,t)}{\partial t} = \frac{\partial^2 u(x,y,t)}{\partial x^2} + \frac{\partial^2 u(x,y,t)}{\partial y^2} \tag{10}$$

we look at the explicit scheme where we again have to solve the equation by discretizing. By following the same steps such as Taylor expanding and rewriting we end up with the following algorithm:

$$u_{i,j}^{t+1} = \alpha(u_{i+1,j}^t + u_{i-1,j}^t + u_{i,j-1}^t + u_{i,j+1}^t) + (1-4\alpha)u_{i,j}^t$$

Here we assume that

$$\alpha \leq \frac{\Delta t}{\Delta x^2} = \frac{\Delta t}{\Delta y^2},$$

i.e $\Delta x = \Delta y$ where $x \in [0, L], y \in [0, L]$.

The initial and boundary conditions in 2 dimensions also has to be taken into account. We can extend our boundary conditions from 1 dimension, meaning that $u(0, y, t) = 0$ and $u(L, y, t) = 1$. We can also experiment with boundary conditions where for instance the left and right side is set to 1, i.e $u(0, y, t) = u(L, y, t) = 1$. Which means we have a matrix with ones on the left and right edge.

In order to implement the implicit scheme for two dimensions we need to use the Jacobi iteration method. For this method we also discretize equation (10), however now we have more unknown parameters, as we do not know the surrounding values of u^t. We model this by using a matrix approach, this matrix represents a grid where a movement in vertical direction is a movement in y-direction and a movement in horizontal direction is a movement in x-direction. The initial state of the matrix is a matrix with ones in the first and last column and zeroes everywhere else. To find the matrix at the next time step we then use the Jacobi iteration method. This method gives the diffusion in the next time step given by the diffusion in the earlier time step in both the x and y direction. For the Laplace equation the method find the solution by iterating

$$u_{i,j}^{l+1} = \frac{u_{i+1,j}^l + u_{i-1,j}^l + u_{i,j+1}^l + u_{i,j-1}^l}{4}$$ [4].

By using an initial guess for the $u_{i,j}$ where we have all the grid points known we can use this to find the matrix in the next time step. This step is repeated until a certain number of iterations is reached or until the matrix elements reach a set precision. Just like in the one dimensional case we use the implicit scheme because it does not have a set area where it is stable, it is stable for all Δt and Δx. This is an advantage when smaller time steps are wanted without decreasing the

steps in the x and y directions. To use this we first rewrite the discretized version of equation (10) to

$$u_{i,j}^t = \frac{1}{1+4\alpha}\left[\alpha(u_{i+1,j}^t + u_{i-1,j}^t + u_{i,j-1}^t + u_{i,j+1}^t) + u_{i,j}^{t-1}\right]. \tag{11}$$

And then iterate with Jacobi's method to find $u_{i,j}^t$.

Heat flow in the lithosphere

We will solve the heat equation for a physical situation, where we look at a geological situation relating to a proposed subduction zone of the coast of Norway. This subduction zone adds some radioactive elements to the mantle, which then produces some heat due to radioactive decay. This adds an extra term to our differential equation, so we now have the expression

$$\nabla(k\nabla T) + Q = \rho c_p \frac{\partial T}{\partial t}, \tag{12}$$

where k is the thermal conductivity, Q is the heat production, ρ is the density of the mantle and c_p is the specific heat capacity. The area in question goes from the surface, down to 120 km below the surface, and is 150 km wide. To solve this we will begin by scaling the equation. We start with a reduced length $\hat{l} = l/120\,\mathrm{km} = 8.33 \times 10^{-6} l\,\mathrm{m}^{-1}$. If we then divide both sides by k, we get the constant $\rho c_p/k$ on the left hand side. We will here use the values $\rho = 3.5 \times 10^3\,\frac{\mathrm{kg}}{\mathrm{m}^3}$, $k = 2.5\frac{\mathrm{W}}{\mathrm{m\cdot K}}$ and $c_p = 1000\frac{\mathrm{J}}{\mathrm{kg\cdot K}}$. We then see that the constant $\rho c_p/k = 1.4 \times 10^6\,\frac{\mathrm{s}}{\mathrm{m}^2}$. To scale the equation we want the value to be 1. We already have a reduced unit for length, so now we need one for time ($\hat{t} = t/\alpha$) such that $1.4 \times 10^6 \cdot (1.2 \times 10^5\,\mathrm{m})^2/\alpha \cdot \mathrm{s/m}^2 = 1$. This then gives $\alpha = 2.016\,\mathrm{s} = 0.6392\,\mathrm{Gyr}$, which is then our new unit of time. Finally we need to scale the temperature. Here, we have that the temperature at a depth of 120 km is set to $1300^\circ\mathrm{C} = 1573\,\mathrm{K}$. So we set $\hat{T} = T/1573\,\mathrm{K}$. At the surface, we will set a boundary condition of 8° C, and for the boundary conditions at the side we will model the system as being in contact with a reservoir at constant temperature, increasing linearly from 8° C at the surface to 1300° C at 120 km. We also model the system as having a with of 150 km.

With the equations scaled, we will then look at three cases. First we will look at a case where W has a non zero value, but does not change with time, modelling the system before radioactive enrichment. The value of W then represents the heat production from radio active elements that are present in the lithosphere in general. Here, we have $W = 1.4\,\mu\mathrm{W/m}^3$ from a depth of 0 km to 20 km, $W = 0.35\,\mu\mathrm{W/m}^3$ from a depth of 20 km to 40 km and $W = 0.05\,\mu\mathrm{W/m}^3$ from a depth of 40 km to 120 km. The first models the upper crust, the second the lower crust and the third the mantel. We divide this by k to get the values $5.6 \times 10^{-7}\,\mathrm{K/m}^2$, $1.4 \times 10^{-7}\,\mathrm{K/m}^2$ and $2 \times 10^{-8}\,\mathrm{K/m}^2$ respectively. In reduced units we then get 5.13, 1.28 and 0.18 for the three values for W.

For the second system we will model the system with radioactive enrichment, but without any decay (time-dependency in W). To to this we will add an extra $0.5\,\mu\mathrm{W/m}^3$ to the whole mantle. Divided by k and in reduced units this is then 1.83.

For the third situation we will add radioactive decay to the $0.5\,\mu\mathrm{W/m}^3$ from the previous case. We model the additional heat as being produced 40 % by U, with a half-life of 4.47 Gy, 20 % by Th with a half-life of 14.0 Gy and 20 % by K with a half-life of 1.25 Gy. In reduced units the half-lives become 6.99, 21.9 and 1.96 respectively.

To compare the results, we will be using the analytical solution with no radioactivity $W = 0$, and in thermal equilibrium at the start of the simulation, so the temperature only depends on depth, and increases linearly with depth. We will then plot the difference between our three numerical simulations and this analytical steady case.

For the simulations in the geological case we will run all with $n = 100$. For the first two we will then run until $t = 0.2$ Gyr, with $\alpha = 0.4$. For the final simulation, with radioactive decay modeled, we will run until $t = 3.0$ Gyr, to see the effects of the radioactive decay.

To simulate this system we will use the implicit method described in equation (11). However now we have the extra term given by $Q(t, x, y)$. By involving this in the discretization we see that this adds an extra term of $Q(t, x, y)\Delta t$ inside the parenthesis.

RESULTS

By using the numerical partial differential equation solver on the diffusion equation with set boundary conditions we obtain results for the different time steps. We preform the partial differential equation solver on a one dimensional case for explicit scheme, implicit scheme and Crank-Nicolson. Then we extend to two dimensions for the implicit scheme and the explicit scheme. Then lastly we use this solver to solve a physical problem.

One dimension

We have solved the diffusion equation using the explicit scheme in one dimension first. In order to solve this equation we have used the boundary conditions $u(0, t) = 0, \quad u(L, t) = 1$ and initial condition is a state where all the points are zero except for the last point. In one dimension this becomes a vector that changes in time, in order to show how the diffusion changes over time we have plotted the vectors for each time step. We have two figures with 100 points so $\Delta x = 1/100$ and Δt is chosen so that $\alpha \leq 1/2$

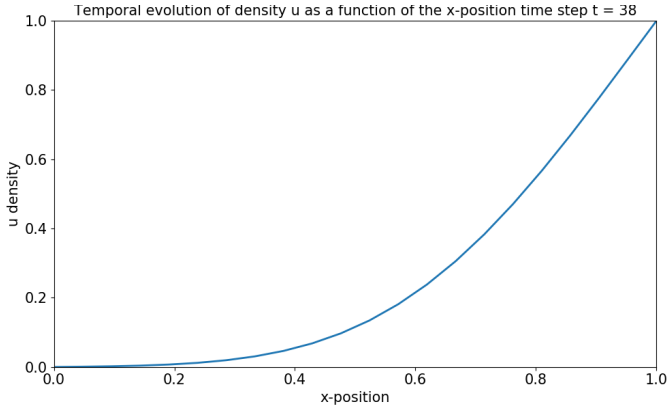


Figure 1: Density distribution when using the explicit scheme at time t_1

Figure 1 is a frame from the time step evolution of the explicit scheme. This figure is a frame from the 38th time step. After 38 time steps the graph is continuous.

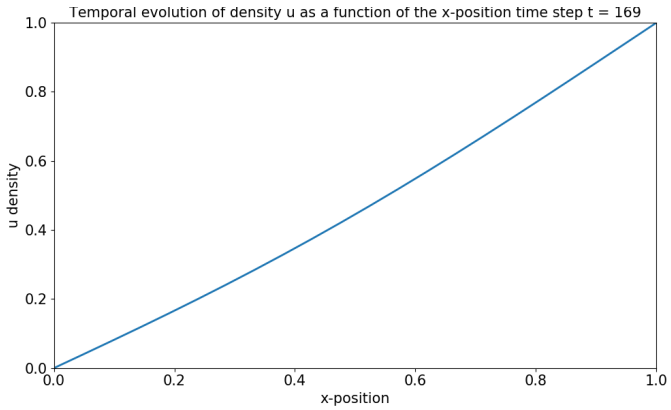


Figure 2: Density distribution using the explicit scheme at time t_2

In figure 2 we see the diffusion at another time step. This graph shows time step 169. We see that at this later time the distribution is very close to linear.

Both the implicit scheme and the Crank-Nicolson method yield a similar result as the explicit scheme where the distribution goes from zero everywhere to a continuous curve to in the end being a linear graph where the distribution is equal everywhere.

We have explored the errors in the different methods above in the methods section, however we can now use the results to see how the different methods compare in our calculations.

Table II: Deviation from the analytical solution for different time steps for all three different methods with n=20.

Method	Explicit	Implicit	Crank-Nicolson
$t_1 = 38$	0.04503	0.00145	0.00038
$t_2 = 169$	0.05178	0.00065	8.0164e-05

Table II shows the average difference between the analytical and the different methods at time step 38 and time step 169. This table shows that the error is much smaller when using Crank-Nicolson than when using either implicit or explicit.

Two Dimensions

In two dimensions we have made contour plots for both implicit and explicit scheme. These contour plots show how the density get distributed over time.

For the 2D explicit solution we have made a GIF (see [5]) of a contour plot showing how $u(x, y, t)$ evolves in time with boundary conditions $u(0, y, t) = u(L, y, t) = 1, \quad u(x, 0, t) = u(x, L, t) = 0$. the initial conditions were 0 everywhere except the boundaries.

We have also made an animation with the implicit method with slightly different boundary conditions $u(x, 0, t) = u(x, L, t) = 1, \quad u(0, y, t) = u(L, y, t) = 0$, see figure 3 and 4 for contour plots at two different times. The same initial conditions were used as with the explicit method.

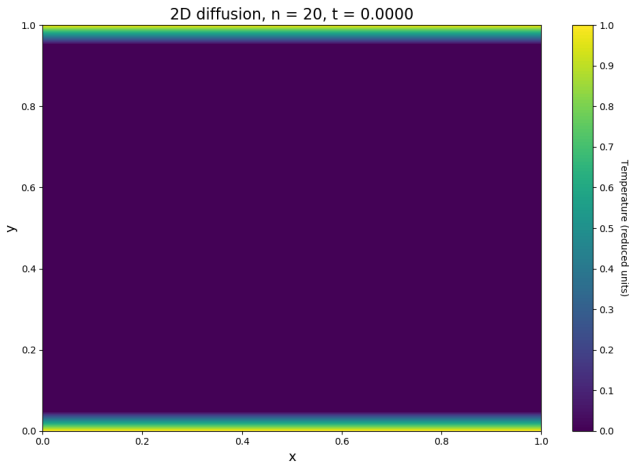


Figure 3: Implicit scheme two dimensions, n = 20, at time step 0.

From figure 3 we see that the the diffusion is the same everywhere except from the edges due to the initial conditions.

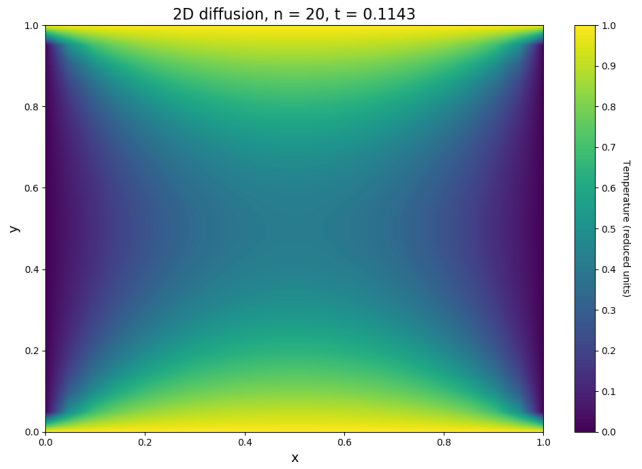


Figure 4: Implicit scheme two dimensions, n=20, at time step 182

In figure 4 we see that the density is more evenly spread. We can also see that the edges still have the same value (boundary conditions hold) as the figure illustrating an earlier time step.

Parallelization with OpenMP

We have also tried to parallelize our code to see how much of an effect it has on the code run time. Figure 5 shows us the time spent on a parallelized algorithm for the 2 D explicit scheme. It is worth mentioning that this is the result of merely one run. We also get a similar result for the implicit scheme.

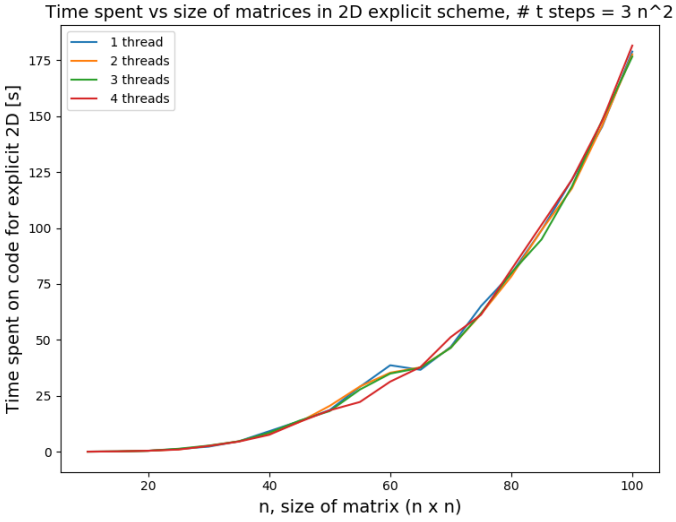


Figure 5: Plot showing the time spent on parallelized 2D explicit scheme for different values of n and different number of threads

Geological case

When we plotted the results in the first case, and plotted the difference from the analytical case with extra heat production from radioactivity we got the results in Figure 6.

With added radioactive enrichment in the mantle for case 2, we got the results shown in Figure 7.

For the third and final case with radioactive decay modeled, we got the results in Figure 8. All these plots are at the bottom of the article, as they take up a large amount of space.

DISCUSSION

One dimension

We study our results (figures 1 and 2) and compare with our analytic expression. We see that $t \to \infty \implies u(x,t) \to \frac{x}{L}$ because we have an exponential time decay. In other words this result is what we expect, a linear density function and since our functions are reduced $L = 1 \implies u(x, t \to \infty) = x$ Which we see holds true as we continue iterating in time. When we look at the function before $t \to \infty$ it is more difficult to tell what kind of function we are looking at, which is also the case with the analytical expression which has a term which is the sum of the product between an alternating sign, a sinus-expression and en exponential decay function.

From table II we see that Crank-Nicolson scheme is a more accurate method of finding the different values of u as this table shows that the difference between Crank-Nicolson results and the analytical results are much lower than both the implicit and the explicit scheme. This implies that this method is a better method because it has a much lower error. We can also see from this table that the error is smaller at a larger time step for Crank-Nicolson, but around the same for higher time steps for implicit and explicit scheme. This is as expected since the errors scale as seen in table I. We see that the truncation error is smaller for the Crank-Nicolsen scheme due to the error's proportionality to Δt^2 as opposed to Δt with the other two schemes.

In addition there is the matter of algorithmic stability. The explicit scheme is simple to implement, but unstable if the requirement $\alpha = \frac{\Delta t}{\Delta x^2} \leq 1/2$ is not met, meaning that for the scheme to work as intended we need to have $t \geq 2n^2$. If we for instance have $n = 100$ points in space we need at least $t = 2 \cdot 10^4$ time steps just to get results that make sense. This is clearly a waste of CPU time as we see from the results in figure 2, where $u(x,t)$ reaches the $t \to \infty$ limit after merely 138 time steps. In other words, equilibrium is already reached and nothing interesting happens for the majority of the time steps, meaning that an implicit scheme or the Crank-Nicolson scheme is much more efficient due to having the luxury of choosing how many time steps we want.

All in all for this one dimensional case we see that the "best" method ordered from worst to best is explicit (forward Euler), implicit (Backward Euler) then the Crank-Nicolson method.

Something to keep in mind for eventual future work is that we could have plotted all the methods at t_1 and t_2 but this would be 6 plots where each pair would show us practically the same thing. Therefore, to avoid redundancies we took a less graphical approach an analyzed the errors and stability criteria instead.

Two Dimensional case

We can see that in figure 3, which is at the very beginning of the time, the system is zero all places apart from two of the edges. This is determined by the boundary and initial conditions, we have here chosen to let two opposite ends have the value 1 at all times. Figure 4 shows the same system at a later time. that as time goes on the diffusion is more evenly spread. This means that in a physical case, where there are for instance particles in a room, the particles are more spread after a time period than they are in the beginning. We can from these figures also see that even though the value in the middle of the matrix changes the boundaries stay the same. This is as we expect from setting defined boundary conditions. From the animations (found here[5]) we can see that the change in the matrices is a gradual process over time. We see that the results for the implicit and explicit scheme in two dimensions hardly differ, and the time it takes for the methods to be performed is not far apart either, however due to the lack of stability constraints the implicit scheme might be preferable over the explicit scheme. By using only the implicit scheme the time steps and the steps in x and y direction can be chosen more flexibly.

The results obtained seem to be consistent with the physical nature of diffusion. We can for example see this two dimensional system as a simple heat distribution system where two of the edges are heated, the edges with boundary condition 1, while the room is cold. The solution of the diffusion equation then represents how the temperature in the room changes over time. Many other physical systems can also be solved with a partial differential equation solver for the diffusion equation. A more in depth and thorough example on heat flow in the lithosphere shows how this can be applies to real life situations.

We also take a look at the run time of our parallelized code (Figure 5). When solving the partial differential equations numerically it is limited how much we can parallelize the code. In other words we can only parallelize the operations done on the matrix elements. Unfortunately, it seems like parallelization has little to no effect. In some cases, it even seems like it has the opposite effect, slowing the code down as seen by the red line (4 threads) having the longest run time when $n = 100$. This might be due to the fact that some of the threads finish before the others making the others wait. Since both the explicit and implicit method is an iterative method we can not parallelize the parts that take time such as the time loop or the Jacobi iterations because the next step is dependant on the previous. There is however some deviancy going on around $n = 60$ where 4 threads gives us the shortest run time. For future work it is worth exploring this, there might be something wrong with our implementation of OpenMP.

Geological case

From our results in the first case, illustrated in Figure 6, we see that the temperature in the crust can be over 100°C higher with radioactive elements than without. We also see that with the added heat production, the system reach a roughly steady state in 0.1 to 0.2 Gyr. We also see that with only this heat production the temperature is changed more in the crust than the mantle, which is as expected given that we modeled the heat production as far higher in the crust than the mantle.

In the second case, with added radioactive enrichment, illustrated in Figure 7, in the mantle we see that the biggest change from the linear case is in the middle, which indicates that the effects from the mantle have a far more important role than in the first case. This is also as expected, given that the heat production in the mantle is now higher than the lower crust, and the area is far larger, so the total power output will be larger. This result then resembles that one would have gotten with a constant power production in the whole simulation area. If we look at the centre of the simulation, we see that the temperature reaches around 250°C higher than the solution for no radioactivity. Compared with Figure 6 where this value is around 50°C, we see that the added radioactive enrichment can increase the temperature by about 200°C in the mantle in our simulations.

In the third case with added radioactive decay, we see that we get to about the same temperature levels as in the second case in the first 0.3 Gyr, however after 1 Gyr the temperature have gone down significantly and the largest increase from the linear case is around 200°C, So for a proposed radioactive enrichment 1 Gyr ago one should se a increase of around 100°C (when comparing with Figure 6) from the temperature without radioactive enrichment. After 3 Gyr it has decreased even further. This is again as one would expect given that the radioactive elements have half lives on the order of magnitude of 1 to 10 Gyr. So in the first 0.2 Gyr the temperature should change roughly as if there were no half life, but after several Gyr the effects of radioactive becomes significant. If we again look at the centre of the simulation, we see that after 3 Gyr the temperature has gone down to 150°C over the linear case, meaning that after 3 Gyr has passed the temperature in the mantle has decreased by about 100°C due to radioactive decay.

In our simulations we modeled the geological system as being in contact with a system with a temperature constant in time, where the temperature increased linearly with depth. This is not quite accurate, as the temperature of the simulated lithosphere will affect the temperature of the parts around it, these non-simulated parts will probably also have some radioactivity so the temperature profile is not entirely linear. In future work this could then be improved upon, perhaps by varying the radioactive enrichment with width, and not just depth as done here. One could then also experiment with other boundary conditions like periodical boundary conditions, and test the ef-

fects of varying these conditions. The effects of the boundary conditions are especially apparent in the second and third case.

CONCLUSION

We have thus explored the diffusion equation by regarding it as a differential equation that we can discretize in order to solve numerically. Numerical implementations make it possible to solve differential equations that can not be solved analytically. In addition using a numerical method can make it easier to solve larger systems in less time than is possible analytically. We have explored different methods of solving the differential equations focusing on the Euler family, in order to optimize the methods and reduce the error there is a possibility for solving the diffusion equation using other differential equation solvers. Using more complicated solvers might reduce the error, but might also be more time consuming that the methods used in this article. We have also seen that as the explicit scheme is not stable in all cases, it is beneficial to use the implicit scheme in most cases.

After having explored the different methods for solving the diffusion equation with partial differential equation solvers we have applied these solvers on a geological system. In this system we have modeled the temperature in the lithosphere with and without radioactive enrichment. Here we found that by adding the radioactive elements had a considerable effect on the temperature in the mantle, where it increased by as much as $200°C$ above the temperature without radioactive enrichment. When considering radioactive decay, this sunk to around $100°C$ after about 3 Gyr. In future explorations it can be of interest to see if this system can be solved using different PDE solvers in an effort to see if the deviation from the analytical results can be reduced. To summarize we can say that the use of different solvers when solving partial differential equations is very useful for making relatively simple models of real physical systems numerically.

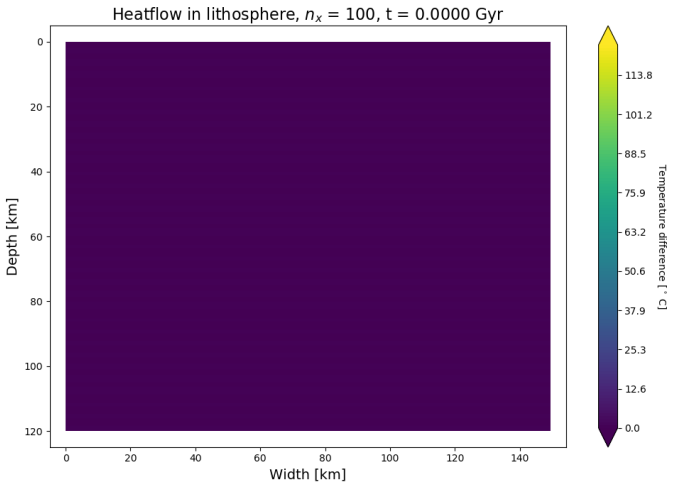
[1] "https://github.com/ilsekup/FYS3150/tree/master/Project%205."

[2] Hjort-Jensen M., "Computational Physics: Lecture Notes Fall 2015," *Lecture notes*, pp. 303–315, 2015.

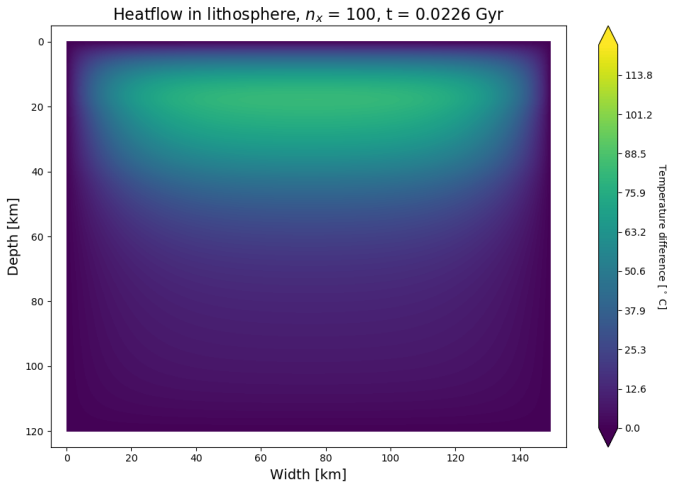
[3] Kuperus, I and Nilsen, F. L and Sankaya, F., "Numerical Solver for Second Order Differential Equations," *Project 1*, pp. 1–6, 2019.

[4] Stuart Dalziel, "Partial Differential Equations; http://www.damtp.cam.ac.uk/lab/people/sd/lectures/nummeth98/pdes.htm," 1998.

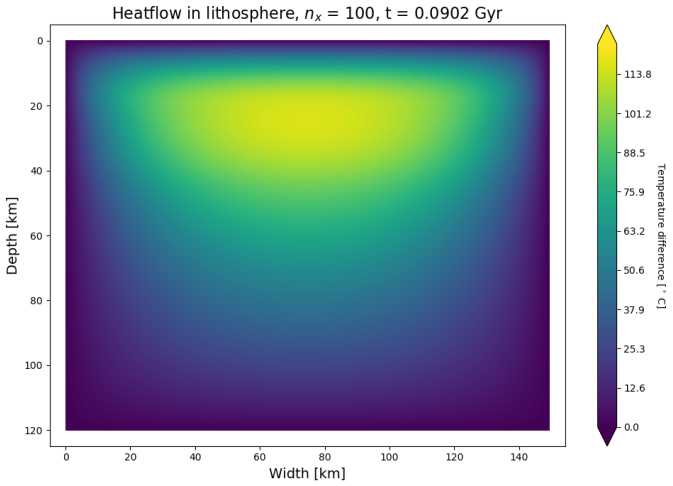
[5] "https://github.com/ilsekup/FYS3150/tree/master/Project%205/Results."



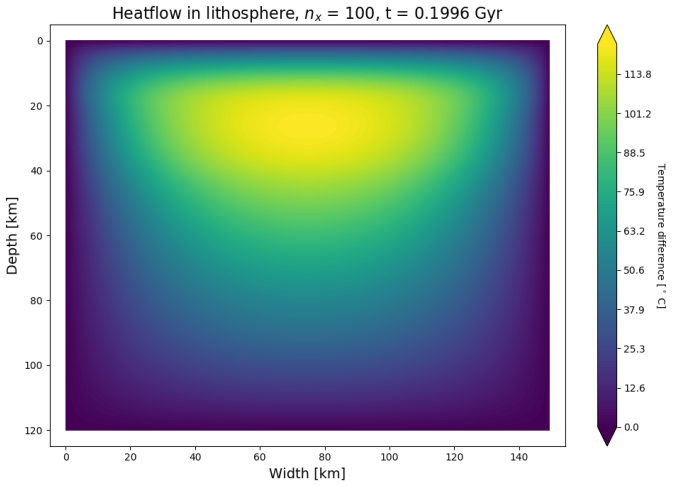
(a) At $t = 0$



(b) At $t = 0.0226\,\mathrm{Gyr}$

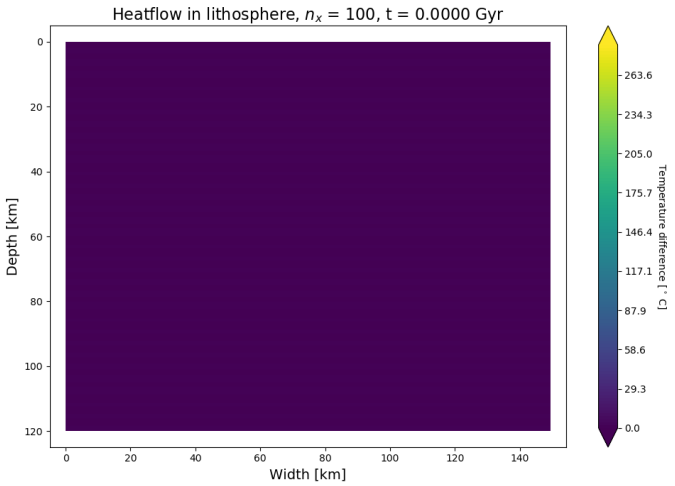


(c) At $t = 0.0902\,\mathrm{Gyr}$

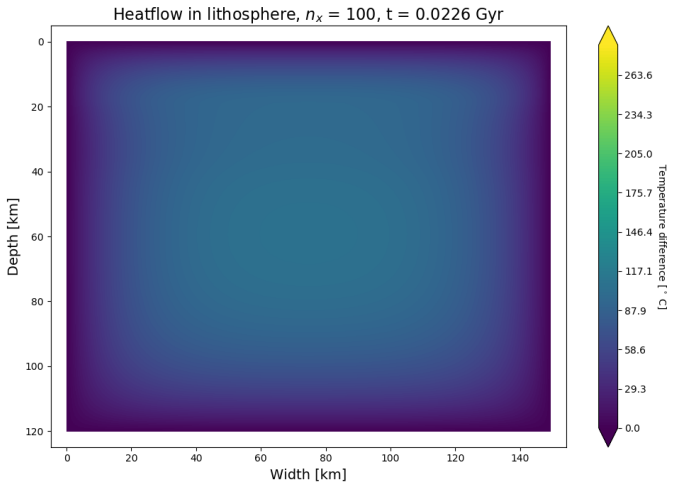


(d) At $t = 0.1996\,\mathrm{Gyr}$

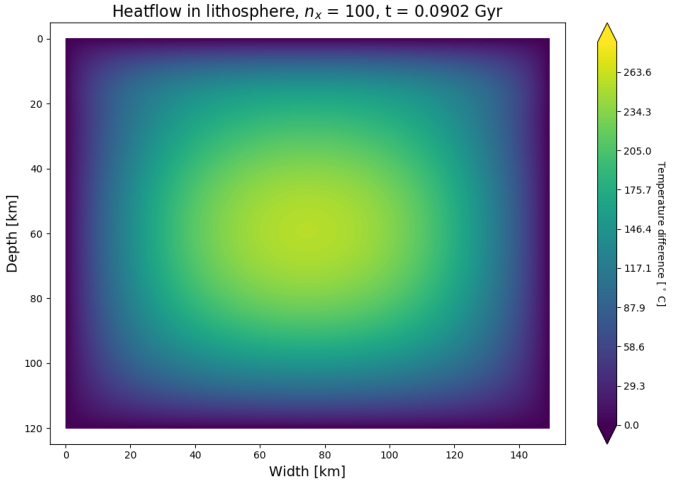
Figure 6: The temperature difference from the linear stationary solution without any radioactivity for the first case.



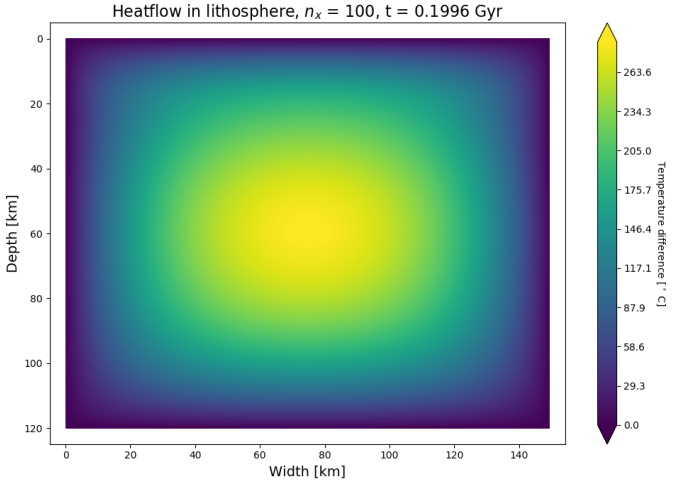
(a) At $t = 0$



(b) At $t = 0.0226\,\mathrm{Gyr}$

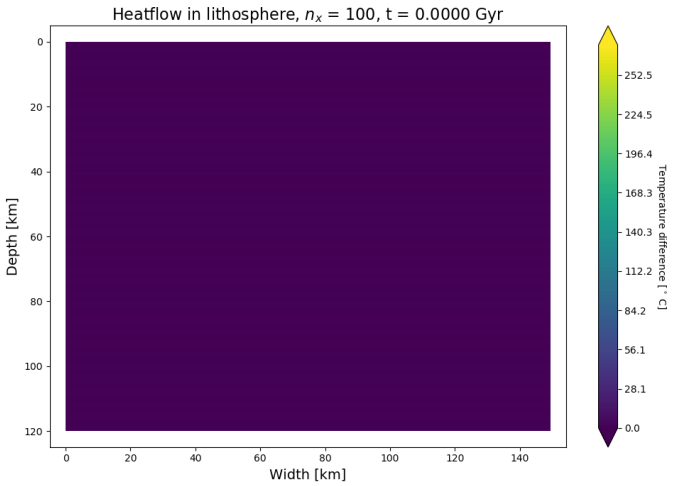


(c) At $t = 0.0902\,\mathrm{Gyr}$

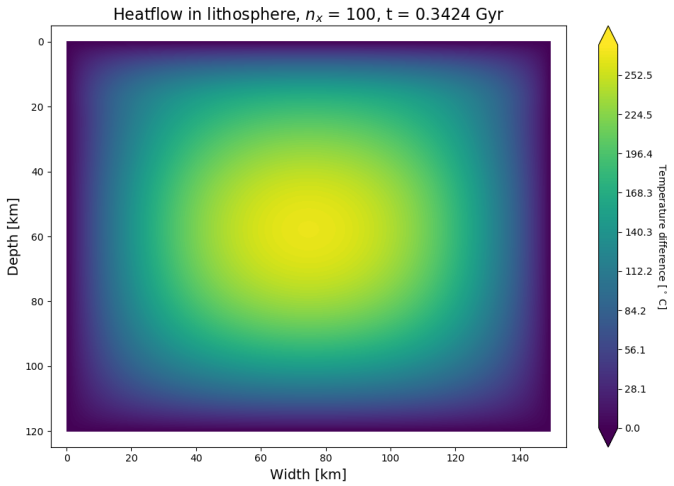


(d) At $t = 0.1996\,\mathrm{Gyr}$

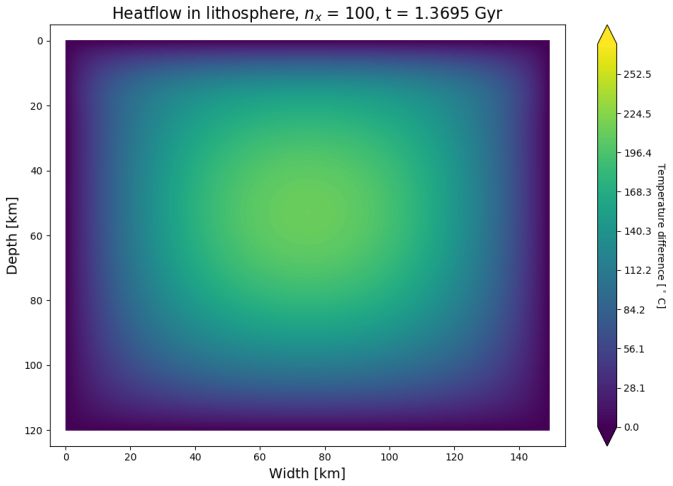
Figure 7: The temperature difference from the linear stationary solution without any radioactivity for the second case.



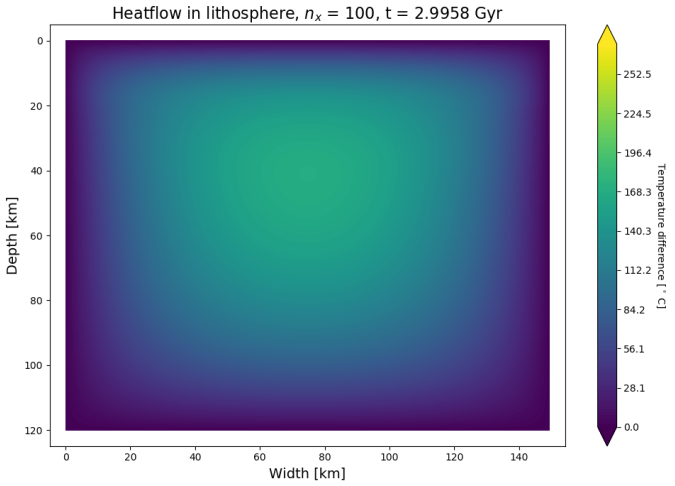
(a) At $t = 0$



(b) At $t = 0.3427\,\mathrm{Gyr}$



(c) At $t = 1.3695\,\mathrm{Gyr}$



(d) At $t = 2.9958\,\mathrm{Gyr}$

Figure 8: The temperature difference from the linear stationary solution without any radioactivity for the third case.