Solving the diffusion equation analytically and numerically

Fredrik Hoftun
Faculty of Mathematics and Natural Sciences, University of Oslo

(Dated: December 17, 2021)

In this project we successfully computed the diffusion equation numerically and analytically. We modeled the distribution of temperature over an uniform rod where the initial temperature distribution was sine-wave shaped.

I. INTRODUCTION

Diffusion describes the net flow of something, like how salt in water spreads around. In this project we will solve the diffusion equation analytically and numerically for a uniform rod of length L. We will model the distribution of temperature over time.

II. METHODS

A. Theory

1. *Diffusion equation*

Diffusion is the net movement of something from high concentration to low concentration, described by Flick's law

$$J = -D\nabla c$$

where J is the diffusion flux, D is the diffusion coefficient, ∇ is the gradient-operator and c is concentration. The movement of diffusive particles is described by the continuity equation, also known as transport equation

$$\frac{\partial \rho}{\partial t} + \nabla \cdot J = 0$$

where ρ is the density, or concentration in fluids. Combining them we get the diffusion equation, a partial differential equation

$$\frac{\partial c(\vec{x},t)}{\partial t} = D\nabla^2 c(\vec{x},t)$$

Where $\vec{c}$ is the concentration vector.

2. *Analytical solution of the diffusion equation*

The one dimensional diffusion equation is analytically solvable with specific boundary and initial conditions. The simplest case is the use of Dirichlet boundary conditions, which we will use. We can reduce our diffusion equation by substituting in $u = c/D$ which gets us

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

We define u in space between 0 and $L = 1$, $x \in [0, L = 1]$, and use Dirichlet boundary conditions

$$u(0,t) = u(L,t) = 0, t \geq 0$$

with initial condition

$$u(x,0) = \sin \pi x, x \in (0, L)$$

We assume separation of variables $u(x,t) = U(x)T(t)$. Our equations then become

$$\frac{1}{T(t)}\frac{\partial T(t)}{\partial t} = \frac{1}{U(x)}\frac{\partial^2 U(x,t)}{\partial x^2} = -k^2$$

where k is a constant. The solutions to these equations are possible when $k^2 > 0$ (since the exponential doesn't satisfy the boundary condition) and are

$$U(x) = A\cos kx + B\sin kx)$$

$$T(t) = Ce^{-\lambda^2 t}$$

Applying the boundary condition we see that $A = 0$

$$U(x) = B\sin kx$$

where $k = n\pi, n \in \mathbb{N}$, giving us

$$U_n(x) = B_n \sin n\pi x$$

The general solution is then, with the use of Fourier coefficients

$$U(x,0) = \sum_{n=1}^{\infty} B_n \sin n\pi x$$

where $B_n = 2\int_0^{L=1} u(x,t=0)\sin n\pi x\,dx$. We now apply the boundary condition again $u(x,0) = \sin \pi x$. We can now find the Fourier coefficients

$$B_n = 2\int_0^1 \sin(\pi x)\sin(n\pi x)dx = \delta(n,1)$$

Or $B_1 = 1, B_2 = 0, ...B_n = 0, n = 2, 3, ...$ This gives us the final solution

$$u(x,t) = U(x)T(t) = \sin(\pi x)e^{-k^2 t} = \sin(\pi x)e^{-\pi^2 t}$$

3. Numerical solution of the diffusion equation

There are various ways to compute the partial differential equations numerically. We will use the simplest method, the explicit finite difference method. By the definition of the derivative and double derivative we can discretize the problem easily. Here p denotes the current step and p+1 denotes the next step, in either space or time.

$$\frac{\partial u}{\partial t} = \frac{u_i^{j+1} - u_i^j}{\Delta t} + O(\Delta t)$$

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

Plugging this into the diffusion equation and solving for u_i^{j+1}, the next time step, we get

$$u_i^{j+1} = u_i^j(1 - 2S) + S(u_{i+1}^j + u_{i-1}^j)$$

where $S = \Delta t / \Delta x^2 \leq 1/2$ is the stability criterion.

B. Implementation

The discretized method was vectorized and implemented in Python, with help from the modules NumPy and Matplotlib. It was implemented in a for-loop that ran over the time-steps. If the distribution was flat enough, determined by a tolerance value, the for-loop was terminated and the parameters were returned. We then used the final time to create 5 uniformly spaced times, to plot our figures. The number of data-points were directly related to the ratio between parameter and step-size, rounded up and added 1 for boundary. The number of spatial data points were $N_x = L/dx + 1$, while the number of temporal data-points were $N_t = T/dt + 1$, where T is the maximum time.

III. RESULTS

We ran 2 simulations where $dx = 0.1, 0.01$,$dt = 0.001, 0.00001$, $T = 1$s and $tol = 1e-5$. Both combinations satisfied the stability criterion. The plots are shown in figures 1 and 2 in the appendix [VII] below.

IV. DISCUSSION

From figures 1 and 2 we see from the mean squared error (MSE) that the numerical approximations were accurate. We see that figure 1 uses much fewer data-points than figure 2. This is due to the implementation.

V. CONCLUSION

In this project we have analytically and numerically solved the diffusion equation for an uniform rod of length $L = 1$ and temperature specified by the sine function. From the mean squared error we saw that the numerical method was accurate for both sets of values of dx and dt, even with few data-points, like where $dx = 0.1$ and $dt = 0.001$.

VI. REFERENCES

Links are available by pressing the reference. Link to GitHub repository.[1] Course page with project description.[2]

[1] F. Hoftun, Fys-stk4155 - project 3 (2021).

[2] M. Hjorth-Jensen, Applied data analysis and machine learning, fys-stk3155/4155 at the university of oslo, norway (2021).

VII. APPENDIX

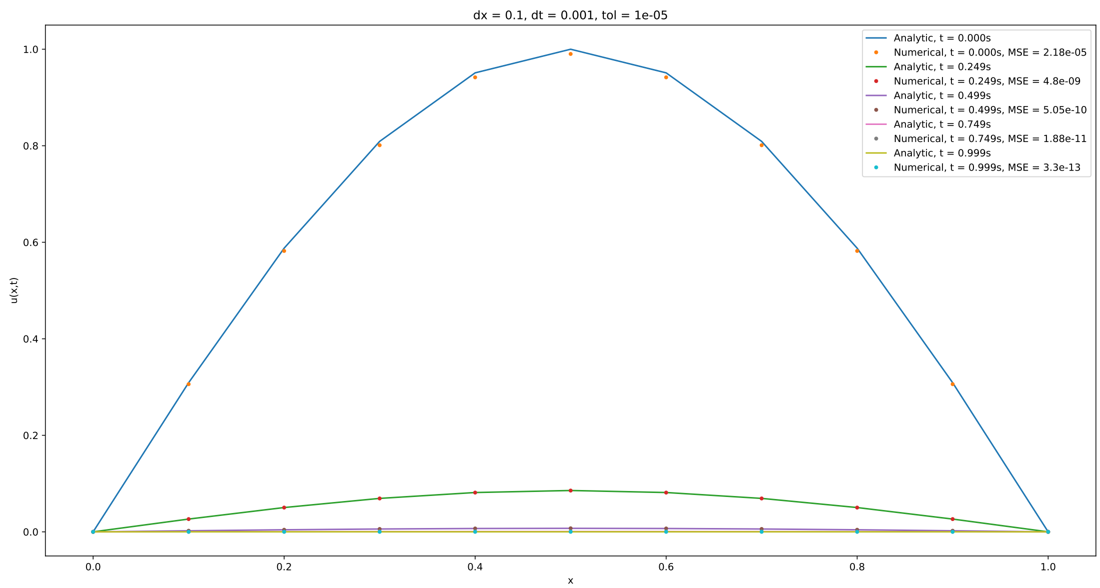

Figure 1. The figure shows the analytical and numerical solutions of our diffusion equation u(x,t) where $dx = 0.1$ and $dt = 0.001$.

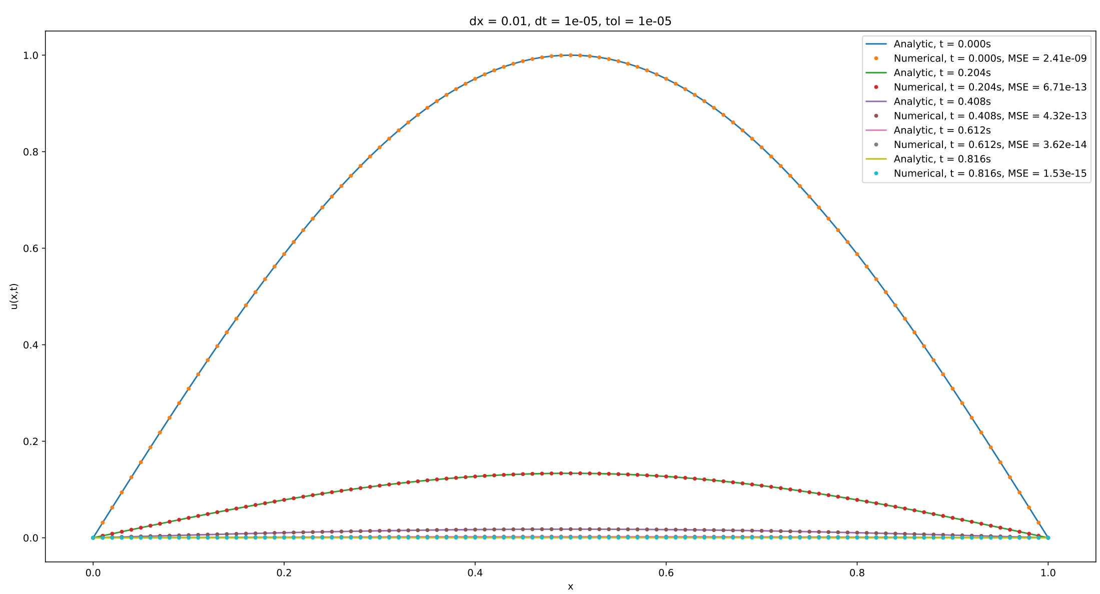

Figure 2. The figure shows the analytical and numerical solutions of our diffusion equation u(x,t) where $dx = 0.01$ and $dt = 0.00001$.