Numerical methods
Bachelor in Physics

Practicum 7: Partial differential equations I: Basics and simple explicit methods

Contact: jeroen.mulkers@uantwerpen.be U.305

In the previous practicum, we have studied problems which can be expressed in ordinary differential equations (ODEs). However, there are many physical problems which need to be described by partial differential equations (PDEs), e.g. the Schrödinger equation in quantum mechanics, the Maxwell equations in electromagnetism, and the wave equation in optics.

In the next practicals, we will use a number of different techniques to solve PDEs numerically. All these methods are *finite difference* methods. But, let's start with a recapitulation of PDEs before solving them numerically.

1 Introduction to PDEs

1.1 Classification

When solving ODEs, we can use general methods, such as Runga-Kutta. This is no longer the case for PDEs. Different types of PDEs demand different solutions.

There are three archetypes of PDEs. The first one is the *one-dimensional diffusion equation*:

$$\frac{\partial}{\partial t}T(x,t) = \kappa \frac{\partial^2}{\partial x^2}T(x,t)$$

with diffusion coefficient κ. This equation describes many different diffusion processes, e.g. $T(x,t)$ could represent the temperature at position x at moment t. Another example is the 1D time dependent Schrödinger equation. This diffusion equation is a *parabolic* PDE.

The second important PDE is the *One-dimsenional wave equation*:

$$\frac{\partial^2 A}{\partial t^2} = c^2 \frac{\partial^2 A}{\partial x^2}$$

with amplitude $A(x,t)$ and propagation speed c. This equation is an example of a *hyperbolic* PDE.

The third PDE archetype is the *Poisson equation*, which is used in electrostatics. In two dimensions, the equation is given by

$$\frac{\partial^2\Phi}{\partial x^2} + \frac{\partial^2\Phi}{\partial y^2} = -\frac{1}{\epsilon_0}\rho(x,y)$$

with electrostatic potential $\Phi(x,y)$, charge density $\rho(x,y)$, and the permittivity of air ϵ_0. This equation is called *Laplace equation* if $\rho = 0$. The Poisson and the Laplace equations are examples of *elliptical* PDEs.

The end goal of this practicum, is not to study the classification of PDEs in depth. We will also not study the uniqueness of the solutions. Instead, we will study some specific examples for each type of PDE.

1.2 Initial value problems

Parabolic and hyperbolic PDEs, such as the diffusion equation and the wave equation, are usually solved for a given initial condition. E.g. in case of the differential equation, one can calculate the $T(x,t)$ for $t > 0$ if the initial temperature $T(x,t=0)$ is known. Analogously, in case of the wave equation, one can calculate the amplitude $A(x,t)$ for $t > 0$ for an initial amplitude $A(x,t=0)$ and an initial velocity $dA(x,t=0)/dt$ of a pulse.

On the other hand, it is possible to have boundary conditions which need to be taken into account. E.g. assume that we want to determine the temperature T in a certain region of space, let's say between $x = -L/2$ en $x = L/2$, then we could use the diffusion equation and impose the following boundary conditions:

$$T(x=-L/2,t) = T_a; \;\; T(x=L/2,t) = T_b$$

These are called *Dirichlet* boundary conditions. We could also solve the diffusion equation if the thermal flux, instead of the temperature itself, is known at the boundaries:

$$-\kappa \left.\frac{dT}{dx}\right|_{x=-L/2} = F_a; \;\; -\kappa \left.\frac{dT}{dx}\right|_{x=L/2} = F_b$$

These boundary conditions are called *Neumann* boundary conditions. A third type of boundary conditions are the *periodic boundary conditions* (the solution at both boundaries are equal):

$$\begin{aligned} T(x=-L/2,t) &= T(x=L/2,t) \\ \left.\frac{dT}{dx}\right|_{x=-L/2} &= \left.\frac{dT}{dx}\right|_{x=L/2} \end{aligned}$$

In order to solve such equations numerically, we will, similarly to ODE, discretize time $t_n = (n-1)\tau$ with time step τ and $n = 1, 2, \ldots$ We will also discretize space on a grid $x_i = (i-1)h - L/2$ with h the distance between the grid points and $i = 1, 2, \ldots$ Initial value problems are often solved with *marching* methods. Using the initial values, one can calculate the values for the next time step. This result can be used to calculate the next time step, and so on.

2 The diffusion equation

2.1 Forward Time Centered Space Scheme (FTCS)

We start with the diffusion equation, since this PDE is the easiest one to solve numerically:

$$\frac{\partial}{\partial t} T(x,t) = \kappa \frac{\partial^2}{\partial x^2} T(x,t)$$

with temperature $T(x,t)$ and thermal diffusion coefficient κ. We want solve the problem for initial condition $T(x, t=0) = \delta(x)$ and Dirichlet boundary conditions

$$T(x = -L/2, t) = T(x = L/2, t) = 0.$$

There are many ways to solve this problem analytically, but we will solve this problem numerically. We will use the following discretization of time and space:

$$T_i^n = T(x_i, t_n)$$

with $x_i = (i-1)h - L/2$ and $t_n = (n-1)\tau$. Index i corresponds with a grid point and index n with a time step. Note that $h = L/(N-1)$.

Using the forward difference approximation for the time derivative

$$\frac{\partial T(x,t)}{\partial t} \Rightarrow \frac{T_i^{n+1} - T_i^n}{\tau}$$

and the central difference approximation for the spatial derivative

$$\frac{\partial^2 T(x,t)}{\partial x^2} \Rightarrow \frac{T_{i+1}^n + T_{i-1}^n - 2T_i^n}{h^2}$$

we get the following discretized diffusion equation

$$\frac{T_i^{n+1} - T_i^n}{\tau} = \kappa \frac{T_{i+1}^n + T_{i-1}^n - 2T_i^n}{h^2}.$$

This yields the following iterative scheme to calculate the temperature at time step $n+1$:

$$T_i^{n+1} = T_i^n + \frac{\kappa\tau}{h^2}(T_{i+1}^n + T_{i-1}^n - 2T_i^n).$$

This method is called the *forward time centered space scheme* (FTCS) due to the used finite difference approximations. The FTCS is an *explicit* method since all terms that depend on n is on the rhs, while on the lhs, we only have the value of the temperature at time step $n+1$.

2.2 The FTCS program

The program dftcs.m solves the diffusion equation, using the FTCS. The algorithm can be described as follows:

- Initialize the parameters $(\tau, h, \ldots)$.
- Impose initial and boundary conditions
- Choose the number of time steps
- Iterate over the chosen number of time steps
  - Calculate the new temperature using the FTCS.
  - Store the temperature regularly
- Plot the temperature as a function of x and t.

The BC can be written down as

$$T_1^n = T_N^n = 0$$

for all n. It is impossible to implement the initial condition $T(x,0) = \delta(x)$. In order to approximate the delta function numerically, we use

$$\Delta(x) = \begin{cases} (x+h)/h^2 & \text{for } -h < x < 0 \\ (h-x)/h^2 & \text{for } 0 \leq< x < h \\ 0 & \text{else} \end{cases}$$

It holds true that

$$\lim_{h \to 0} \Delta(x) = \delta(x)$$

After discretization, we get

$$\Delta_i = \begin{cases} 1/h & \text{for } i = N/2 \\ 0 & \text{else} \end{cases}$$

In order to choose a suitable time step h, it is useful to note that the following Gaussian function

$$T_G(x,t) = \frac{1}{\sigma(t)\sqrt{2\pi}} \exp\left[\frac{-(x-x_0)^2}{2\sigma^2(t)}\right]$$

with an increasing standard deviation

$$\sigma(t) = \sqrt{2\kappa t}$$

is a solution of the diffusion equation. Call t_σ the time needed to increase the width σ from 0 to h, so

$$t_\sigma = \frac{h^2}{2\kappa}.$$

Intuitively, we can say that it is probably not a good idea to choose a time step much larger than t_σ

2.3 Tasks

Study the program dftcs.m and study carefully how the FTCS is implemented. Also, try to understand the implementation of the initial and boundary conditions.

1. Run the program for $N = 61$, $\kappa = 1$ and a different number of values for τ between 10^{-3} and 10^{-5}. Also, test the influence of different N for $\tau = 10^{-4}$. What do you notice? Relate this to t_σ. Study the result for initial condition $T(x, 0) = \delta(x - L/4)$.

2. Modify dftcs.m in order to use the following Neumann boundary conditions

$$\left.\frac{\partial T}{\partial x}\right|_{x=-L/2} = \left.\frac{\partial T}{\partial x}\right|_{x=L/2} = 0.$$

   Study the results for the initial conditions $T(x, 0) = \delta(x)$ and $T(x, 0) = \delta(x - L/4)$.

3. The DuFort-Frankel scheme (central difference approximation for both derivatives) uses the following discratisation of the the diffusion equation:

$$\frac{T_i^{n+1} - T_i^{n-1}}{2\tau} = \kappa \frac{T_{i+1}^n + T_{i-1}^n - (T_i^{n+1} + T_i^{n-1})}{h^2}$$

   Note that this scheme is not self-starting. Write a program which implements the DuFort-Frankel scheme. Use the FCTS for the first time step. Try multiple values for τ. Is this method stable? Is this method accurate?