PDE Diaries

A. Nieto

July 2020

1 The rules of the game

"We do not know what the rules of the game are; all we are allowed to do is to watch the playing. Of course, if we watch long enough we may eventually catch on to a few rules. The rules of the game are what we mean by fundamental physics" - Richard Feynman

The universe, as I want to believe it, moves according to some given rules, it follows a hidden pattern and it goes somewhere unknown. The partial differential equations (PDE) offer us a way to glimpse into those rules, when only as a coarse approximation. Any simulation we made reflects the universe in its conception, it moves on its own, following only the rules given by us at the beginning, but independently from our will and in some cases even beyond our prediction capabilities.

The name "PDE Diares" is quite misleading. I came up with this name on one of my holiday trips, the idea was to have a small notebook where I was suppose to solve some PDE exercises in a daily basis. As you can imagine after a short period of time the notebook became just a dead weight in my luggage, nevertheless the name stuck.

This text is actually a summary of some ideas I had while trying to learn how to code numerical simulations with Python. I'll start from the end, because as any other story, its development depends almost uniquely on the retrospective ideas of the author, on what it was already thought. My hope is that you will enjoy learning about *the rules of the game* as much as I did!

2 Disclaimer

The text, as it is today, is unfortunately not prepared as an introduction to numerical simulations, I'm assuming for the moment that the readers have some notion of how a numerical simulation goes, from the discretization of the PDEs to the post-processing visualization of the results. I'll eventually provide some external links I find adequate for beginners and some time in the future I'll try to include some basic stuff as well ... some day.

3 Nomenclature

I was thinking for a long time what would be the best way to shorten the already short nomenclature of the PDEs and discretization schemes. The risk I see by this approach is that eventually no one will understand what the extra-shortened nomenclature was supposed to mean, included me. We can give it a try anyway, for the sake of beauty!

3.1 PDE nomenclature

Let's start with some basic nomenclature, having the scalar field of an arbitrary quantity in function of time and 2D space $u(t, x, y)$, we express the change of u w.r.t. one of its variables t as:

$$\frac{\partial u}{\partial t} \to u_t$$
$$\frac{\partial u}{\partial x} \to u_x$$
$$\frac{\partial u}{\partial y} \to u_y$$

The second derivative w.r.t. a same variable is then written as:

$$\frac{\partial^2 u}{\partial x^2} \to u_{xx}$$

And finally the mixture of several partial derivatives w.r.t. to different variables can be expressed as:

$$\frac{\partial^2 u}{\partial x \partial y} \to u_{xy}$$

That way we can keep our PDEs nice and clean, but what about the discretization schemes?

3.2 Discretization schemes - "classic" nomenclature

This one is going to be trickier, as the discretized schemes relate on many upper- and sub-indexes to be understandable. The first step is to define the discretization scheme normally used for partial derivatives.

Take into account that these expressions are derived from the Taylor expansion by neglecting the H.O.T. (Higher Order Terms), which will lead irrevocably to a cumulative error and thus it counts only as an approximation of the derivative terms.

For a 1D scalar field of the quantity $u(x, t)$:

$$u_t \approx \frac{u_i^{n+1} - u_i^n}{\Delta t}$$
$$u_x \approx \frac{u_i^n - u_{i-1}^n}{\Delta x}$$

Where $n+1$ express a leap forward in time and $i-1$ a leap backward in x-space.

When dealing with 2D scalar fields of the type $u(t,x,y)$ it becomes necessary to express the leaps in y-space by using an extra sub-index, so that the previous expressions become:

$$u_t \approx \frac{u_{i,j}^{n+1} - u_{i,j}^{n}}{\Delta t} \tag{1a}$$

$$u_x \approx \frac{u_{i,j}^{n} - u_{i-1,j}^{n}}{\Delta x} \tag{1b}$$

$$u_y \approx \frac{u_{i,j}^{n} - u_{i,j-1}^{n}}{\Delta y} \tag{1c}$$

And the derivative of $u(t,x,y)$ w.r.t. y is expressed by using backward leaps in y-space as given in (1c).

For the second order derivatives w.r.t. the same variable we have:

$$u_{xx} \approx \frac{u_{i+1,j}^{n} - 2u_{i,j}^{n} + u_{i-1,j}^{n}}{\Delta x^2} \tag{2a}$$

$$u_{yy} \approx \frac{u_{i,j+1}^{n} - 2u_{i,j}^{n} + u_{i,j-1}^{n}}{\Delta y^2} \tag{2b}$$

I think you can start to see what a mess of indexes this nomenclature will produce when discretizing more complex PDEs. But all of these indexes are necessary for a good comprehension of the discretization schemes!

3.3 Discretization schemes - "short" nomenclature

What I propose next is a compromise between given information and readability, with the aim of making simpler and faster schemes, less prone to typos and more easy to correct.

We start by writing (1) as:

$$u_t \approx \frac{u_{oo}^{+} - u_{oo}}{\Delta t} \tag{3a}$$

$$u_x \approx \frac{u_{oo} - u_{-o}}{\Delta x} \tag{3b}$$

$$u_y \approx \frac{u_{oo} - u_{o-}}{\Delta y} \tag{3c}$$

And then we write (2) as:

$$u_{xx} \approx \frac{u_{+o} - 2u_{oo} + u_{-o}}{\Delta x^2} \tag{4a}$$

$$u_{yy} \approx \frac{u_{o+} - 2u_{oo} + u_{o-}}{\Delta y^2} \tag{4b}$$

So what had happened here? In summary:

- All the letters n were dropped, as we normally only used one instance of the quantity in the next time step, that instance is represented by the upper-index $+$.
- All the letters i, j and commas were dropped, as we already know we are dealing with a 2D space in function of x and y, and the former comes always first in the sub-indexes order.
- The backward and forward leaps in space are represented with the sub-indexes $-$ and $+$ respectively. The order here is important, as $u_{+o} \neq u_{o+}$, the former expressing a leap forward in x-space $(i+1)$ and the latter a leap forward in y-space $(j+1)$.
- When there are no leaps in space, the sub-index o is used as analog to $i+0$ or $j+0$.

That way we have cleared most of the writing while still holding a compressible nomenclature. We'll see in future examples how this "short" nomenclature will help us to quickly write and program our schemes.