NUMERICAL METHODS

Finite Difference

Pedro Andrés Colombo

21317227

15/01/2014

Index

Finite Difference Method

Description

The basis for the finite difference method is the construction of a discrete grid, the replacement of the continuous derivatives in the governing partial differential equation with equivalent finite differences expressions and the rearrangement of the resulting algebraic equation into an algorithm [1] .

1D Advection Equation

1 - Introduction

1.1 - The equation that it is going to be solved is:

$$\frac{\partial u}{\partial t} + V\frac{\partial u}{\partial x} = 0$$

1.2 - Physical meaning

In this equation V is the known velocity (of the wind for instance) and u represents a passive scalar, for example the temperature (it is assumed that this scalar has no influence on the transport).

2 - Scheme: FTBS

Several schemes can be used to discretize this equation; if, for example, a scheme FTCS (forward in time, centered in space) is used with this equation, we get a solution that is unconditionally unstable. In a first approach we are going to use a scheme Forward in time and Backward in space (FTBS).

$$\frac{u_i^{n+1} - u_i^n}{\Delta t} + V\frac{u_i^n - u_{i-1}^n}{\Delta x} = 0$$

Transposing the previous equation we obtain:

$$u_i^{n+1} = u_i^n - V\frac{\Delta t}{\Delta x}(u_i^n - u_{i-1}^n)$$

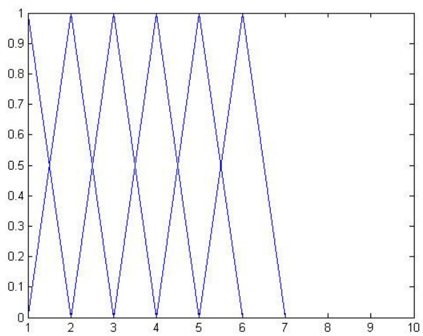
If we apply a Taylor's expansion to this equation we will find that exists a truncation error of $O(\Delta t^2, \Delta x^2)$

The code used for this case can be observed in Appendix - 1.

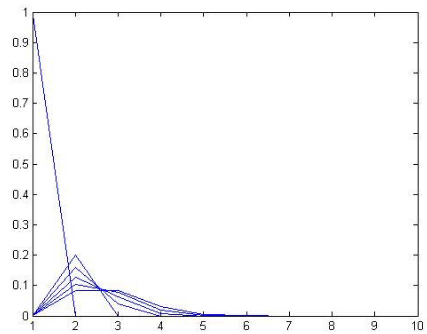
2.1. - General comments - Limits of the method

2.1.1 - We can define 3 different scenarios:

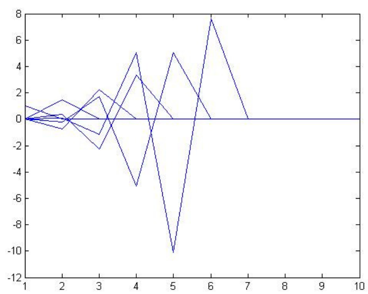
a) V=$\frac{\Delta x}{\Delta t}$=10 (c=V*$\frac{\Delta t}{\Delta x}$=1)



b)V=2 (c=0,2)



c) V=15 (c=1,5)



2.1.2 - Stability

A numerical solution scheme is said to be stable if it does not amplify errors that appear in the course of the numerical solution process.

We can observe in figure "c" that the scheme is no longer stable. This scheme is *conditionally stable*; is stable if and only if the physical velocity V is not bigger than the spreading velocity $\frac{\Delta x}{\Delta t}$ of the numerical method, in other words, $c \leq 1$ (stability condition as we will see below).

In the figure "a" we have c=1, so that is the analytical solution.

2.1.2.1 - Von Neumann stability analysis for the present scheme (based on the method for diffusion equation using the FTCS scheme from [4])

$$\frac{\xi_i^{n+1}-\xi_i^n}{\Delta t}=-V\frac{\xi_i^n-\xi_{i-1}^n}{\Delta x}$$

$$\xi_i^{n+1}=\xi_i^n-d(\xi_i^n-\xi_{i-1}^n)$$

Where $d = V\frac{\Delta t}{\Delta t}$. Each component can be written as

$$\xi_i^n = B^n e^{Ik_x(i\Delta x)}$$

Where B^n is the amplitude function at time-level n of the particular component whose wave number is k_x and $I = \sqrt{-1}$. The spatial domain is considered infinite in extent. Define the phase angle $\theta = k_x \Delta x$, giving

$$\xi_i^n = B^n e^{Ii\theta}$$

Similarly

$$\xi_{i\pm 1}^{n+1} = B^{n+1} e^{I(i\pm 1)\theta}$$

Substituting in the first equation

$$B^{n+1} e^{I(i)\theta} = B^n e^{I(i)\theta} - d(B^n e^{I(i)\theta} - B^n e^{I(i-1)\theta})$$

Cancelling the common terms $e^{I(i)\theta}$

$$B^{n+1} = B^n[1 - d(1 - e^{-I\theta})]$$

And defining the amplification factor G

$$B^{n+1} = GB^n$$

We find that

$$G = 1 - d(1 - e^{-I\theta})$$

We use the identity $\cos(\theta) - i\sin(\theta) = e^{-I\theta}$

$$G = 1 - d[1 - (\cos(\theta) - i\sin(\theta))]$$
$$G = 1 - d[1 - \cos(\theta)] + di\sin(\theta)]$$

If we remember that the way to get the length of a complex number is multiplying it by its complex conjugate, we find

$$G = 1 - d[1 - \cos(\theta)] + d^2\sin^2(\theta)]$$
$$G = 1 - d\{[1 - \cos(\theta)] + 2d^2[1 - \cos(2\theta)]\}$$

So we can see that, if the solutions are to remain bounded, we must have (note that $G = G(\theta)$)

$$\left|G\right| \le 1$$

$$-1 \le 1 - d\{[1-\cos(\theta)] + 2d^2[1-\cos(2\theta)]\} \le 1$$

Which must be satisfied for all possible θ. We will consider the critical values, those are when the values are maximum --> $\cos\theta = -1$ and $\cos(2\theta) = 1$ (for $\cos\theta = 1$ we have a trivial solution), which then gives the stability requirement

$$-1 \le 1 - 2d \le 1$$
$$0 \le 2 - 2d \le 2$$
$$0 \le 1 - d \le 1$$
$$-1 \le -d \le 0$$
$$1 \ge d \ge 0$$

So we can finally conclude that:

$$d = V\frac{\Delta t}{\Delta x} \le 1$$

This constant d that we found is usually found in the literature as the Courant number expressed by "c", as in the examples above. For d=1 we have the analytical solution.

2.1.3 - Diffusivity associated [(1)] :

A Taylor series expansion give us:

$$\frac{\partial u}{\partial t} + V\frac{\partial u}{\partial x} + 0.5\Delta t\frac{\partial^2 u}{\partial t^2} - 0.5\Delta x\frac{\partial^2 u}{\partial x^2} + O(\Delta t^2, \Delta x^2) = 0$$

If we consider

$\frac{\partial u}{\partial t} = -V\frac{\partial u}{\partial x}$, and $\frac{\partial^2 u}{\partial t^2} = -V^2\frac{\partial^2 u}{\partial x^2}$

The main equation can be rewrite as

$$\frac{\partial u}{\partial t} + V\frac{\partial u}{\partial x} - 0.5V\Delta x(1-c)\frac{\partial^2 u}{\partial x^2} + O(\Delta t^2, \Delta x^2) = 0$$

If we assume the truncation error, we can rewrite the equation as

$$\frac{\partial u}{\partial t} + V\frac{\partial u}{\partial x} - \alpha\frac{\partial^2 u}{\partial x^2} = 0$$

Therefore, the use of this scheme in the 1D advection equation introduce an artificial diffusivity $\alpha = 0.5V\Delta x(1-c)$. This can be easily observed in figure "b". Clearly, the artificial diffusivity is zero when c=1 (figure "a"). To avoid the artificial diffusivity we can set Δt , in order to obtain c=1.

3 - Scheme: Leapfrog

The leapfrog method consists in approximate both derivatives with a centered approximation

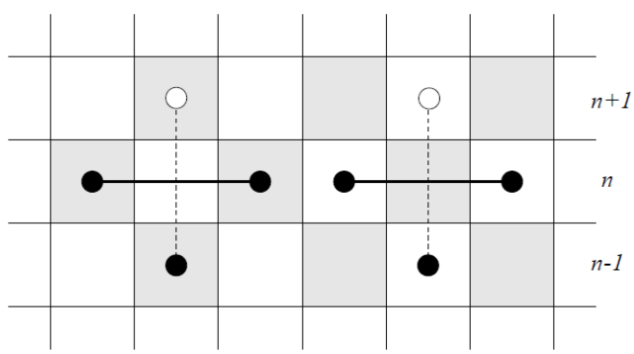
$$\frac{u_i^{n+1} - u_i^{n-1}}{2\Delta t} + V\frac{u_{i+1}^n - u_{i-1}^n}{2\Delta x} = 0$$

Transposing the previous equation we obtain:

$$u_i^{n+1} = u_i^{n-1} - V\frac{\Delta t}{\Delta x}(u_{i+1}^n - u_{i-1}^n)$$

If we apply a Taylor's expansion to this equation we will find that exists a truncation error of $O(\Delta t^2, \Delta x^2)$

In the following picture [(3)] we can see the structure of the scheme:



In the picture we can observe the major disadvantage of the method: odd and even mesh points are completely decoupled. We will see later, that a filter can be applied to sort this difficulty.

The code used for this scheme can be found in Appendix - 2.

3.1. - **General comments - Limits of the method**

3.1.1 - We can observe that to find the value of the function at one time step, it is necessary to know the value of the function at the previous two time steps. This is a problem, because with this scheme we need two values to start it; in addition to the physical initial condition, a computational initial condition is required. There are several alternatives to find this computational condition, and from then on, the leapfrog scheme can be used; however, the errors of the first step will persist.

The most important of these several alternatives are:

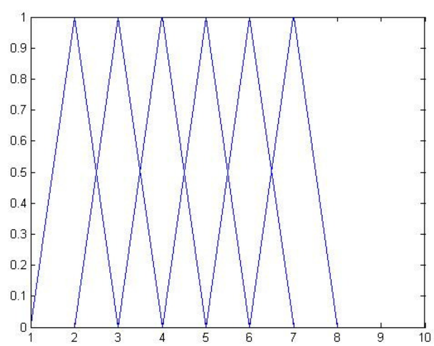
- Set the computational initial condition equal to the physic initial condition, this introduce errors of first order and that is why is not recommended.

- Use half of the initial time step for the forward time step, followed by leapfrog time steps. This will reduce the error introduced in the unstable first step.

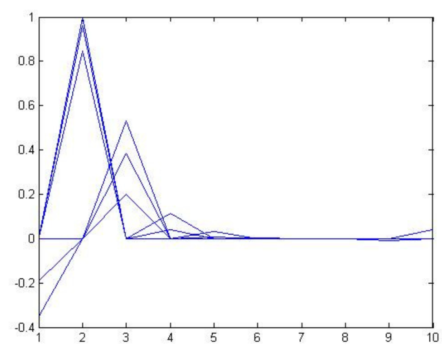
- Use another scheme for the first time step. Since this time step is only used once, the total error is still of order 2. We are going to use an Euler scheme to set our computational initial condition.

3.1.2 - We can define <u>3 different scenarios:</u>

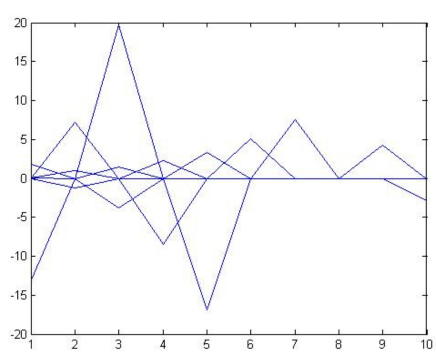
a) V= $\frac{\Delta x}{\Delta t}$ =10 (c=V* $\frac{\Delta t}{\Delta x}$ =1)



b)V=2 (c=0,2)



c) V=15 (c=1,5)



3.1.3 -<u>Stability</u>

We will proceed in the same way that in the other scheme. We can observe in figure "c" that the scheme is no longer stable. This scheme is *conditionally stable*; is stable if and only if the physical velocity V is not bigger than the spreading velocity $\frac{\Delta x}{\Delta t}$ of the numerical method, in other words, $c \leq 1$ (stability condition as we will see below).

If we perform the same analysis for this scheme than for the other scheme we will find the same stability condition, $c \leq 1$.

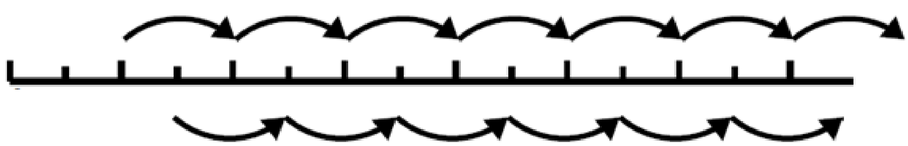
In the figure "a", c=1, so that is the analytical solution.

3.1.4 - Diffusivity associated

The diffusive problem remains in this scheme because the explanation that we did was not related with any particular scheme. This can be observed in the figure "b".

$$\frac{\partial u}{\partial t} + V\frac{\partial u}{\partial x} - \alpha\frac{\partial^2 u}{\partial x^2} = 0$$

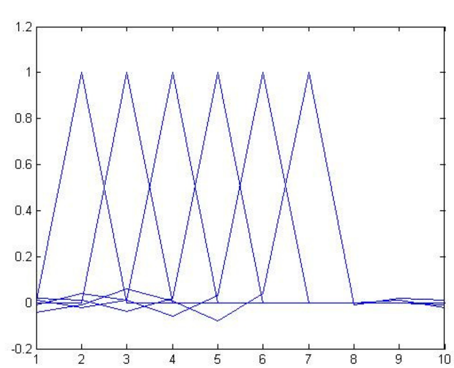
3.2 Robert-Asselin filter



One problem with this method is that odd and even time step are decoupled, as we can see in the picture above. So we introduce this filter that after each leapfrog step, it mixes solutions from the three consecutive time points t_{n-1}, t_n and t_{n+1}.

$$\overline{U^n} = U^n + \alpha(U^{n+1} - 2U^n + U^{n-1})$$

In the code we introduced the variable "concbRAF", and it is applied when we update the variables to perform the change in time at the end of the code. The chosen constant it is called "alpha" in the code, and for a value of 0.01 we get the following result



Appendix

1) 1D Advection Equation code - FTBS

```
clear;

%Define domain
len=100;
nl=10;
dx=len/nl;

%Time parameters
dt=1;
nt=5;

%Wind velocity
V=dx/dt;

%Initial condition
concb=zeros(nl,1);
concb(1)=1;
conca=zeros(nl,1);
plot(concb)

 %Time integration
 for it=1:nt
     conca(1)=0;
     for ix=2:nl
         conca(ix)=concb(ix)-V*dt/dx*(concb(ix)-concb(ix-1))
     end
     concb=conca;
     hold on
     plot(concb)
     hold off
 end
```

2) 1D Advection Equation code - Leapfrog scheme

```
clear;

%Define domain
len=100;
nl=10;
dx=len/nl;

%Time parameters
dt=1;
nt=5;

%Wind velocity
% V=dx/dt;
```

```
V=10;

%RAF paramether(if alpha=0 -> no filter)
alpha=0;
concbRAF=zeros(nl,1);

%Initial condition
conca=zeros(nl,1);
concb=zeros(nl,1);
concc=zeros(nl,1);
concb(1)=0;
concb(2)=1;
concc(1)=concb(1)+V*dt/dx*(concb(2)-concb(1));
plot(concb)

 %Time integration
 for it=1:nt;
     for ix=1:nl
         if ix<=(nl-1);
             ixp=ix+1;
         else
             ixp=1;
         end
         if ix>=2;
             ixm=ix-1;
         else
             ixm=nl;
         end
         conca(ix)=concc(ix)-V*dt/dx*(concb(ixp)-concb(ixm));
     end
     concbRAF=concb+alpha*(conca-2*concb+concc);
     concc=concbRAF;
     concb=conca;
hold on
plot(concb)
hold off
 end
```

Bibliography / References

(1) - Computational Techniques for Fluid Dynamics. Volume 1 - C.A.J. Fletcher - 2nd Edition 1991, Springer.

(2) http://www.youtube.com/watch?v=y2WaK7_iMRI - 12 Steps to Navier Stokes Equations - Lorena Barba (Boston University, USA).

(3) Numerical Methods for the Solution of Partial Differential Equations. Lecture Notes for the COMPSTAR School on Computational Astrophysics, 8-13/02/10, Caen, France Luciano Rezzolla

(4) Computational Fluids Dynamics - Patrick J. Roache -1972, Hermosa Publishers