Numerical Modeling in a nutshell

Viggo H. Hansteen

September 7, 2012

1 Introduction

When dealing with physical systems, and in particular for the study of stellar dynamics one is often required to solve *partial differential equations* (PDEs) of various kinds. The solution method selected will depend on the type of equation that is confronted, these can be divided into *hyperbolic*, *parabolic*, and *elliptic* type equations. Typical examples of the hyperbolic and parabolic equation are given by the one-dimesional wave equation and the diffusion equation respectively:

$$\frac{\partial^2 u}{\partial t^2} = v^2 \frac{\partial^2 u}{\partial x^2} \tag{1}$$

$$\frac{\partial u}{\partial t} = \frac{\partial}{\partial x}\left(D\frac{\partial u}{\partial x}\right) \tag{2}$$

These equations are often solved as *initial value problems*. On the other hand, a typical elliptic equation is the Poisson equation

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = \rho(x, y), \tag{3}$$

which, in contrast, is best solved as a *boundary value problem*.

In our case, as modellers of stellar atmospheres, we are largely interested in solving the equations of mass, momentum, and energy balance, along with the equations governing the evolution of the magnetic field, the transport equations for the radiation field, heat flux equations, etc.

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{u}) = 0$$

$$\frac{\partial \rho \mathbf{u}}{\partial t} + \nabla \cdot (\rho \mathbf{u})\mathbf{u} = -\nabla p_{\rm g} + \rho g + \dots$$

$$\dots$$

In sum, the MHD equations with additions and extensions depending on the problem to be solved and the regime which we wish to study.

1.1 Initial value problems

In initial value problems there are a number of questions one must answer as a starting point for solving the model equations:

- What are the dependent variables?
- What is the evolution equation for each variable? Are there any "non-hyperbolic" portions of the equations that would require alternate solution strategies.

- What is the highest time derivative?
- What boundary condition governs the evolution in time? These can be Dirichlet boundaries on the variables themselves; Neumann boundaries wich set the gradients normal to the boundary as functions of time; or outgoing wave boundaries.

The overriding computational concern in initial value problems is often stability, may reasonable looking solution strategies often do not work as we will see later when considering von Neumann stability analysis.

1.2 Boundary value problems

In boundary value problems the question raised are

- What are the variables?
- What equations are satisfied in the interior of the region of interest?
- What equations are satisfied by points on the boundary? Dirichlet, Neumann or something more complicated?

In this type of problem, in contrast to initial value problems, stability is relatively easy. What is more difficult and the major goal of setting solution strategies is the efficiency of the algorithm, both in terms of storage and computational load.

Since all the conditions on a boundary value problem must be satisfied "simultaneously" these problems often end being a question of solving a large number of simultaneous algebraic equations. When such equations are non-linear the strategy is usually to linearize and iterate.

As an example, consider the Poisson equation 3 and let us solve it by the *finite difference mehtod*. We represent the function $u(x, y)$ by its values at discrete points

$$\begin{aligned} x_j &= x_0 + j\Delta, \qquad j = 0, 1, \ldots, J \\ y_l &= y_0 + l\Delta, \qquad l = 0, 1, \ldots, L \end{aligned}$$

where Δ is the grid spacing. For equation 3 we substitute a finite difference representation

$$\frac{u_{j+1,l} - 2u_{j,l} + u_{j-1,l}}{\Delta^2} + \frac{u_{j,l+1} - 2u_{j,l} + u_{j,l-1}}{\Delta^2} = \rho_{j,l}$$

or equivalently

$$u_{j+1,l} + u_{j-1,l} + u_{j,l+1} + u_{j,l-1} - 4u_{j,l} = \Delta^2 \rho_{j,l} \tag{4}$$

To write this system of linear equations in vector form we need to make a vector of u. Let us number the two dimension of grid points in a single one-dimensional sequence by defining

$$i \equiv j(L+1) + 1 \qquad \text{for} \quad j = 0, 1, \ldots, J, \quad l = 0, 1, \ldots, L$$

We can now write

$$u_{i+L+1} + u_{i-(L+1)} + u_{i+1} + u_{i-1} - 4u_i = \Delta^2 \rho_i$$

which holds on the interior points. Taking care of the boundaries we can therefore write the set of equations as

$$\mathbf{A} \cdot \mathbf{u} = \mathbf{b} \tag{5}$$

The structure of the matrix **A** is so called *tridiagonal with fringes* and any general linear second order elliptic equation

$$a(x,y)\frac{\partial^2 u}{\partial x^2} + b(x,y)\frac{\partial u}{\partial x} + c(x,y)\frac{\partial^2 u}{\partial y^2} + d(x,y)\frac{\partial u}{\partial y}$$
$$+e(x,y)\frac{\partial^2 u}{\partial x \partial y} + f(x,y)u = g(x,y)$$

will give a matrix of similar structure except that the non-zero entries will not be constants.

There are three general approaches to solving equations 5: relaxation methods, "rapid" methods, and direct matrix mehtods.

Relaxation methods are based on splitting the matrix **A** into two parts

$$\mathbf{A} = \mathbf{E} - \mathbf{F}$$

where the matrix **E** is easily invertible. In which case eqations 5 becomes

$$\mathbf{E} \cdot \mathbf{u} = \mathbf{F} \cdot \mathbf{u} + \mathbf{b}$$

Relaxation comes in by choosing an initial guess $\mathbf{u}^{(0)}$ and then solving successively for iterates $\mathbf{u}^{(r)}$ from

$$\mathbf{E} \cdot \mathbf{u}^{(r+1)} = \mathbf{F} \cdot \mathbf{u}^{(r)} + \mathbf{b}.$$

Since **E** is chosen to be easily invertible, each iteration is fast.

The so called *rapid* methods apply only for a certain class of problems; with constant coefficients, or those that are seperable in the chose co-ordinates. We will not discuss them further here.

Matrix methods attempt to solve equation 5 directly. The structure of the matrix **A** will determine whether a direct solution is a viable option. In general it is required that the matrix be very sparse. If possible to implement then direct inversion is usually the best choice, as only multigrid relaxation mehtods can compete.

2 Solving the advection equation

The MHD equations are (mostly) hyperbolic and can be framed as flux conservation equations, or even more simply as advection equation. With this as a starting point, it is clear that we will be solving them as initial value problems. This means that the numerical algorithm used will have a general structure given by

```
<set up>;                ! read file containing parameters of run
<read initial model>;
t=t_start
do <while not finished>; ! check if t_end has been reached
  <t->t+dt>;             ! advance dependent variable time dt
  <output of result>;    ! print result if desired
  <determine dt>;        ! apply CFL criteria and/or equivalent
  t=t+dt                 ! prepare for next time step
enddo
<clean up>;
<exit>;
```

Aside from the "administrative" tasks such as reading the set up and initial model, writing the output of our calculations, our central goal is then to find a method for advancing the dependent variables from timestep t to time step $t + dt$.

Let us start with the flux conservation equation

$$\frac{\partial u}{\partial t} = -\frac{\partial F(u)}{\partial x}$$

where u represents any scalar and $F(u)$ the flux of this scalar. We will first study this equation in a particularly simple form, *i.e.* as the *advection equation*:

$$\frac{\partial u}{\partial t} + v\frac{\partial u}{\partial x} = 0, \tag{6}$$

here v is the (constant) flow velocity of the fluid represented by u. This equation has the general solution

$$u = f(x - vt)$$

such that the initial solution $u_0 = f(x, t = t_0)$ moves without changing shape to the right with velocity v.

Let us attempt to solve this equation by using a *finite difference* method. For these methods we will divide continuous space x and time t up into discrete positions and instants for example via

$$x_j = x_0 + j\Delta x \qquad t_n = t_0 + n\Delta t$$

while defining u_j^n as our scalar field at time t_n and position x_j:

$$u_j^n \equiv u(t_n, x_j).$$

This immediately leads to the simplest discretization possible of the terms in the advection equation:

$$\begin{aligned} \left.\frac{\partial u}{\partial t}\right|_{j,n} &= \frac{u_j^{n+1} - u_j^n}{\Delta t} + \mathcal{O}(\Delta t) \\ \left.\frac{\partial u}{\partial x}\right|_{j,n} &= \frac{u_{j+1}^n - u_{j-1}^n}{2\Delta x} + \mathcal{O}(\Delta x^2), \end{aligned}$$

and writing this approximation, named *Forward time, centered space* (FTCS), to advancing the variable u from time t_n to time t_{n+1}

$$u_j^{n+1} = u_j^n - \frac{v\Delta t}{2\Delta x}\left(u_{j+1}^n - u_{j-1}^n\right). \tag{7}$$

To quote *Numerical Recipes* "It's a fine example of an alogrithm that is easy to derive, takes little storage, and executes quickly. Too bad it doesn't work!". To find out why we will perform so called *von Neumann stability analysis*.

2.1 von Neumann stability analysis

von Neumanns stability analysis is local — this implies we assumethat the coefficients of the difference equations are varying slowly enough to be considered constant in space and time. In this case we can look for a solution to the difference equation we are considering of the form

$$u_j^n = \xi^n e^{ikj\Delta x} \quad \text{where} \quad k \in \mathbb{R} \quad \text{and} \quad \xi = \xi(k) \in \mathbb{C} \tag{8}$$

I.e. the time dependence of a singel eigenmode is successive integer powers of the complex number ξ. Therefore, the difference equation will be unstable if

$$|\,\xi(k)\,| > 1$$

and that eigenmode will grow without bound. The number ξ is called the *amplification factor* at a given wave number k.

Let use now find $\xi(k)$ for our difference equation 7. This can be simply done by substituting our expression for the amplification factor 8 into the difference formula, resulting in

$$\xi^{n+1}e^{ikj\Delta x} = \xi^n e^{ikj\Delta x} - \frac{v\Delta t}{2\Delta x}\left(\xi^n e^{ik(j+1)\Delta x} - \xi^n e^{ik(j-1)\Delta x}\right)$$

which, dividing by ξ^n we easily can simplify to

$$\begin{aligned}\xi(k) &= 1 - \frac{v\Delta t}{2\Delta x}\left(e^{ik\Delta x} - e^{-ik\Delta x}\right) \\ &= 1 - i\frac{v\Delta t}{\Delta x}\sin k\Delta x\end{aligned}$$

Note that

$$|\,\xi(k)\,| > 1 \quad \forall\, k$$

so the FTCS scheme is unconditionally unstable for all wavenumbers k.

Note that if the velocity v were a function of x and t we would write v_j^n in equation 7 but still treat it as a constant since the von Neumann analysis is a local analysis. In general, if the equations right hand side were non-linear in u then a von Neumann analysis would linearize by writing $u = u_0 + \delta u$ expanding to linear order in δu. Assuming that u_0 already satisfies the difference equation, the analysis would look for stability in the eigenmodes of δu. According to *Numerical Recipes*, the von Neumann method generally gives valid answers and is much easier to use than more "careful" methods.

2.1.1 Lax Method

We can "repair" the FTCS method by making a small change in the first term on the right hand side of equation 7,

$$u_j^n \quad \rightarrow \quad \frac{1}{2}\left(u_{j+1}^n + u_{j-1}^n\right)$$

e.g by replacing the value of u_j^n by the average of its nearest neighbors. This gives

$$u_j^{n+1} = \frac{1}{2}\left(u_{j+1}^n + u_{j-1}^n\right) - \frac{v\Delta t}{2\Delta x}\left(u_{j+1}^n - u_{j-1}^n\right), \tag{9}$$

which is the *Lax method*. Carrying out the von Neumann stability analysis gives an amplification factor

$$\xi(k) = \cos k\Delta x - i\frac{v\Delta t}{\Delta x}\sin k\Delta x \tag{10}$$

We can easily see that the stability condition $|\,\xi\,|^2 \leq 1$ is satisfied when

$$\frac{|\,v\,|\,\Delta t}{\Delta x} \leq 1. \tag{11}$$

This is the famous *Courant-Friedrichs-Lewy* (CFL or Courant) stability criterion.

We have manged to stabilize our numerical method, but this whole procedure may seem a bit obscure: what in the world have we actually done when replacing u_j^n by the average of its neighbors? Some light may be thrown on this by rewriting the Lax difference equation 9 as

$$\frac{u_j^{n+1} - u_j^n}{\Delta t} = -v\left(\frac{u_{j+1}^n - u_{j-1}^n}{2\Delta x}\right) + \frac{(\Delta x)^2}{2\Delta t}\left(\frac{u_{j+1}^n - 2u_j^n + u_{j-1}^n}{(\Delta x)^2}\right).$$

But, defining a diffusion coefficient $D \equiv (\Delta x)^2/2\Delta t$, we see that this is the FTCS representation of the following continuous PDE:

$$\frac{\partial u}{\partial t} = -v\frac{\partial u}{\partial x} + D\frac{\partial^2 u}{\partial x^2}.$$

In other words, in order to stabalise our solution we have added a diffusion term to our equation, *i.e.* a dissipative term that damps the unconditionally unstable wave modes. This is an example of *numerical dissipation* or *numerical viscosity*; note that unless $\mid v \mid \Delta t$ is exactly equal to Δx, the amplifaction factor $|\xi| < 1$ and the amplitude of a given wave decreases spuriously.

The reason we can accept this behaviour is because those wave modes we are (should be!) interested in are those that have $k\Delta x \ll 1$, but for those $|\xi| \simeq 1$ so these are not damped unduly. The short wavelengths where $k\Delta x \simeq 1$ will die away innocuously, but they are modes we cannot represent properly on our grid of limited resolution anyway.

2.2 Other types of error

Amplitude errors are the most serious type of errors for hyperbolic (and parabolic) type problems, but there are also other types of error that need to be considered, and that give considerable insight into the numerical solution of PDEs. Amongst these are *phase errors*, *non-linear instabilities* and *transport errors*.

2.2.1 Phase errors

Finite difference schemes can exhibit phase, or dispersion, errors. For example equation 10 can be rewritten as

$$\xi(k) = e^{-ik\Delta x} + i\left(1 - \frac{v\Delta t}{\Delta x}\right)\sin k\Delta x \tag{12}$$

Our initial state will be some superposition of modes with different k's. At each timestep these modes are multiplied by different phase factors given by the expression above, depending on k. If $\Delta t = \Delta x/v$, then the exact solution for each mode is only multiplied by $\exp(-ik\Delta x)$ and the solution $f(x - vt)$ is retained. However, if $v\Delta t/\Delta x$ is not exactly one, the phase relations of the modes change and the wave packet disperses. The dispersion becomes large as soon as $k\Delta x$ approaches unity.

2.2.2 Non-linear instabilities

Non-linear instabilities are those associated with hyperbolic PDEs which contain non-linear terms. A typical example is found in the Navier-Stokes equations for fluid flow where we find

$$\frac{\partial v}{\partial t} = -v\frac{\partial v}{\partial x} + \ldots$$

The non-linear term in v will cause a transfer of energy in Fourier space from long wavelenghts (small k) to shorter wavelengths (large k): *i.e* the wave steepens and eventually a shock structure develops. von Neumann analysis (and common sense) suggests that stability will depend on $k\Delta x$, we can only represent gradients up to a certain strength on a grid of finite resolution, and our solution will become unstable as we approach this limit. Energy flows from the small k modes to large k modes where it accumulates and swamps the energy contained in the longer wavelength modes of interest.

Of course, the development of shocks is a natural phenomena that we will be interested in modelling, and there are techniques that have been developed to convert the energy building up at the smallest scales to heat as nature does.

2.2.3 Transport errors

When studying waves, propagation errors in amplitude or phase are often the most worrisome. For advection equations, transport errors are of greater concern. For example in the Lax scheme, equation 9, a disturbance in the advected quantity u at grid point j propagates to $j+1$ and $j-1$ at the next timestep, while in reality, if v is positive, then only $j+1$ should be affected.

One obvious way to model the transport properties better is to use *upwind differencing*.

$$\frac{u_j^{n+1}-u_j^n}{\Delta t} = -v_j^n \begin{cases} \frac{u_j^n - u_{j-1}^n}{\Delta x}, & v_j^n > 0 \\ \frac{u_{j+1}^n - u_j^n}{\Delta x}, & v_j^n < 0 \end{cases}$$

Note that this scheme is only first-order accurate in the calculations of the spatial derivatives. Even so, this scheme is often preferrable to a centered scheme since it includes information on the actualy behaviour of the physical system to be modeled. A von Neumann analysis of the first order upwind scheme agian yield the CFL condition already encountered in equation 11.

2.3 2^{nd} order upwind; van Leers method

Considerable effort was expended during the mid 80's in finding good upwind transport schemes. These are well summarized in [**?**]. One of the 2^{nd} order upwind methods discussed there was developed by Bram van Leer and has been much used by researchers at ITA. To derive this method we begin with forming an integral form of the advection equation 6

$$\int_{x_j}^{x_{j+1}} u(t,x)dx\Big|_{t_n}^{t_n+1} + \int_{t_n}^{tn+1} vu(t,x)dt\Big|_{x_j}^{x_{j+1}} = 0$$

Discretizing this equation then gives us

$$(u_{j+1/2}^{n+1} - u_{j+1/2}^n)\Delta x + (\langle vu\rangle_{j+1} - \langle vu\rangle_j)\Delta t = 0$$

where $\langle vd\rangle_j$ is a time average and fills the role of a flux through the grid cell boundary at j. In the following a cell center position will be given by half integer locations, $j+1/2$ etc. Consider now the calculation of the flux at j. In the $1^s t$ order upwind scheme we simply write

$$\langle vu\rangle_j = \begin{cases} vu_{j-1/2}, & v \geq 0 \\ vu_{j+1/2}, & v < 0. \end{cases}$$

Let us improve the accuracy of this scheme by improving this approximation, if we write

$$\langle vu\rangle_j = \frac{1}{2} v\left[u(t_{n+1}, x_j) + u(t_n, x_j)\right],$$

where the function $u(t_{n+1}, x)$ is approximately equal to $u(t_n, x)$ advected over a distance $v\Delta t = \sigma\Delta x$, where $\sigma = v\Delta t/\Delta x$, *i.e.* we can write

$$u(t_{n+1}, x_j) = u(t_n, x_j - \sigma\Delta x)$$

This means we are averaging the function $u(t,x)$ at $x = x_j$ at the beginning of the timestep and at the end after advection. Let us now move back to the zone center by replacing $u(t_n, x_j)$ with $u(t_n, x_{j-1/2} + \frac{1}{2}\Delta x)$:

$$\begin{aligned} \langle vu\rangle_j = \quad & \tfrac{1}{2} v\left[u\left(t_n, x_{j-1/2} + \tfrac{1}{2}\Delta x - \sigma\Delta x\right)\right. \\ & \left. + u(t_n, x_{j-1/2} + \tfrac{1}{2}\Delta x)\right]. \end{aligned}$$

Expand the first term on the right hand side to first order to obtain

$$u_{j-1/2} + (\frac{1}{2}\Delta x - \sigma\Delta x)\Delta_{j-1/2}u/\Delta x$$

using the gradient of u centered at $j-1/2$, $\Delta_{j-1/2}u/\Delta x$. Substituting this back into the expression for the flux gives

$$\begin{aligned}
\langle vu\rangle_j &= \frac{1}{2}v\left[u_{j-1/2} + \frac{1}{2}\Delta x - \sigma\Delta x)\Delta_{j-1/2}u/\Delta x\right. \\
&\quad \left. u_{j-1/2}(\frac{1}{2}\Delta x)\Delta_{j-1/2}u/\Delta x\right]. \\
&= \frac{1}{2}v\left[2u_{j-1/2} + (\Delta x - \sigma\Delta x)\Delta_{j-1/2}u/\Delta x\right] \\
&= v\left[u_{j-1/2} + \frac{1}{2}(1-\sigma)\Delta_{j-1/2}u\right].
\end{aligned}$$

The accuracy of this scheme will depend on how we define the gradient of $u(t,x)$, $\Delta_{j-1/2}u$. van Leer tried several schemes for this approximation, but in any case our difference advection equation turns out to be

$$u^{n+1}_{j+1/2} = u^n_{j+1/2} - \sigma\left[u^n_{j+1/2} - u_{j-1/2} + \frac{1}{2}(1-\sigma)(\Delta_{j+1/2}u - \Delta_{j-1/2}u)\right]. \tag{13}$$

This scheme works very well when the condition of *monotinicity* is used in the definition of $\Delta_{j-1/2}u$: We require that a distribution $u(x,t)$ which locally increases or decreases monotonically before advection must maintain that quality after advection. This means that the function $u(t,x)$ implicitly defined in zone $j-1/2$ by $u_{j-1/2}$ adn $\Delta_{j-1/2}u$ must have values between those spanned by the neighboring functions $u(t,x)$ in zones $j-3/2$ and $j+1/2$, *i.e.* this means that we require

$$u_{j-3/2} \gtrless u_{j-1/2} + (\frac{1}{2}\Delta x - \sigma\Delta x)\Delta_{j-1/2}u/\Delta x \gtrless u_{j-1/2}$$

In the case of a local minimum or maximum in $u(t,x)$ the slope of $\Delta_{j-1/2}u$ is set equal to zero through the zone in which the extremum occurs. van Leer finds that the following definition for the slope satisfies this criterion:

$$\Delta_{j-1/2} = \begin{cases} 2\frac{(u_{j-1/2}-u_{j-3/2})(u_{j+1/2}-u_{j-1/2})}{(u_{j+1/2}-u_{j-3/2})}, & \text{when} \quad (u_{j-1/2}-u_{j-3/2})(u_{j+1/2}-u_{j-1/2}) > 0 \\ 0 & \text{otherwise} \end{cases}$$

2.4 2^{nd} order accuracy in time

When using a method that is 1^{st} order accurate in time and 2^{nd} order in space one generally has to make $v\Delta t$ significantly smaller than Δx, say by a factor of 5, in order to acheive the desired accuracy. In this case the CFL condition is not actually the limiting factor. It is, however possible to construct schemes that are 2^{nd} order accurate in time as well as in space, in which case one can push the timestep up to the stability limit. One example of such a scheme is the *staggered leapfrog* method which is defined as follows:

$$u^{n+1}_j - u^{n-1}_j = -\frac{\Delta t}{\Delta x}(F^n_{j+1} - F^n_{j-1}).$$

The name comes from the fact that the time levels in the time derivative term "leapfrog" over the time levels used in the space derivative term. Note that this method requires that one store both u^{n-1} and u^n in order to compute u^{n+1}. For our simple model equation 6 the staggered leapfrog takes the form

$$u^{n+1}_j - u^{n-1}_j = -\frac{v\Delta t}{\Delta x}(u_{j+1} - u_{j-1})$$

with amplification factor

$$\xi^2(k) - 1 = -2i\xi(k)\frac{v\Delta t}{\Delta x}\sin k\Delta x$$

with solution

$$\xi = -i\frac{v\Delta t}{\Delta x}\sin k\Delta x \pm \sqrt{1 - \left(\frac{v\Delta t}{\Delta x}\sin k\Delta x\right)^2}$$

So, again we find that the CFL criterion is required for stability. Note that $|\,\xi\,|^2 = 1$ fro any $v\Delta t \leq \Delta x$ and that therefore there is no amplitude dissipation for the staggered leapfrog method. Unfortunately, for equations more complicated than our simple model equation the method becomes unstable when gradients get large. This is related to teh fact that teh odd and even mesh points are completely decoupled like the white and black squares of a chessboard. This problem can be cured by coupling the two meshes through a numerical viscosity term, *e.g.* by adding a small coefficient ($\ll 1$) times $u^n_{j+1} - 2u^n_j + u^n_{j-1}$.

The *Two-Step Lax-Wendroff* scheme is a second-order time method that avoids large numerical dissipation and mesh drifting by defining intermediate values $u^{n+1/2}_{j+1/2}$. These are calculated by the Lax scheme

$$u^{n+1/2}_{j+1/2} = \frac{1}{2}(u^n_{j+1} + u^n_j) - \frac{\Delta t}{2\Delta x}(F^n_{j+1} - F^n_j) \tag{14}$$

and then using these variables to calucate the fluxes $F^{n+1/2}_{j+1/2}$ giving updated values

$$u^{n+1}_j = u^n_j - \frac{\Delta t}{\Delta x}(F^{n+1/2}_{j+1/2} - F^{n+1/2}_{j-1/2}) \tag{15}$$

The stability of this method can be illustrated for the case where $F = vu$ by substituting equation 14 into equation 15

$$\begin{aligned} u^{n+1}_j = u^n_j - \alpha\Bigg[&\frac{1}{2}(u^n_{j+1} + u^n_j) - \frac{1}{2}\alpha(u^n_{j+1} - u^n_j) \\ &-\frac{1}{2}(u^n_j + u^n_{j-1}) + \frac{1}{2}\alpha(u^n_j - u^n_{j-1})\Bigg] \end{aligned}$$

with

$$\alpha \equiv \frac{v\Delta t}{\Delta x}$$

Then

$$\xi(k) = 1 - i\alpha\sin k\Delta x - \alpha^2(1 - \cos k\Delta x)$$

and

$$|\,\xi\,|^2 = 1 - \alpha^2(1 - \alpha^2)(1 - \cos k\Delta x)^2$$

which gives the usual CFL condition for stability.

Except for $\alpha = 1$, $|\,\xi\,|^2 \leq 1$ so there is some amplitude damping. This effect is relatively small for wavelengths large compared with the mesh size Δx. In fact expanding for $k\Delta x \ll 1$ we find

$$|\,\xi\,|^2 = 1 - \alpha^2(1 - \alpha^2)\frac{(k\Delta x)^4}{4} + \ldots$$

and therefore that the departure from unity occurs only at fourth order in k. This should be contrasted with the Lax method which has

$$|\,\xi\,|^2 = 1 - (1 - \alpha^2)(k\Delta x)^2 + \ldots$$

for small $k\Delta x$.

3 The diffusion equation and implicit methods

Consider now a typical version of the *diffusion equation*

$$\frac{\partial u}{\partial t} = \frac{\partial}{\partial x}\left[D\frac{\partial u}{\partial x}\right]. \tag{16}$$

Equation 16 can actually be formulated as a conservative flux equation by setting the flux F to

$$F = -D\frac{\partial u}{\partial x},$$

but as we shall see, the appearance of the second derivate makes a large difference in the behaviour of this equation and in the numerical methods that are used to solve it. Let us first assume that we have $D > 0$, if not, the equation is physically unstable and all small perturbations in u will grow without bound. It is never possible to stably model a problem for which the governing equations themselves are unstable.

Let us first discretize equation 16 using the FTCS scheme and, for simplicity, that D is constant. We then get

$$\frac{u_j^{n+1} - u_j^n}{\Delta t} = D\left[\frac{u_{j+1}^n - 2u_j^n - u_{j-1}^n}{\Delta x^2}\right]$$

If we carry out von Neumann stability analyis on this equation we find

$$\xi(k) = 1 - \frac{4D\Delta t}{(\Delta x)^2}\sin^2\left(\frac{k\Delta x}{2}\right)$$

Thus, we find that the stability condition $|\xi|^2 \leq 1$ is satisfied when

$$\frac{2D\Delta t}{(\Delta x)^2} \leq 1$$

Assuming that the typical scale we are studying is given by λ then the diffusion time scale for this scale is

$$\frac{\lambda^2}{D} \sim \tau,$$

but since we have that $\lambda > \Delta x$ (since we *should* be studying phenomena that are (much) larger than the grid scale it follows that we will have to run of order $\lambda^2/(\Delta x)^2$ timesteps before we see evolution on the scales we are interested in. I.e. running this problem *explicitly*[1] is prohibited by the large number of timesteps needed to solve the problem.

So, let us reformulate the problem using a *Backward time, centered space* (BCTS) method. The difference equation then becomes

$$\frac{u_j^{n+1} - u_j^n}{\Delta t} = D\left[\frac{u_{j+1}^{n+1} - 2u_j^{n+1} - u_{j-1}^{n+1}}{\Delta x^2}\right] \tag{17}$$

Ignoring for a moment the question of how to solve this equation, define

$$\alpha \equiv \frac{D\Delta t}{(\Delta x)^2},$$

and carry out von Neumann stability analysis to find that

$$\xi(k) = \frac{1}{1 + 4\alpha\sin^2\left(\frac{k\Delta x}{2}\right)}$$

[1] An explicit solution is one where the unknown at timestep $(n+1)$, position j, is a function only of quantities at timestep n (or earlier). An *implicit* method is one where several quantities at $(n+1)$, i.e. at several positions appear in the same equation.

Ir is quite clear from this that

$$|\xi(k)|< 1 \quad \forall\, k$$

and that therefore this algorithm is unconditionally stable, no matter the wavenumber k or the length of the timestep Δt

Now turn to the problem of solving for u_j^{n+1}; moving all the unknowns in equation 17 to the left hand side gives

$$-\alpha u_{j-1}^{n+1} + (1+2\alpha)u_j^{n+1} - \alpha u_{j+1}^{n+1} = u_j^n \qquad j = 1, \ldots, J-1.$$

Supplemented with appropriate boundary conditions this is a tridiagonal system that is readily inverted by standard methods. We can use the BTCS method to derive the equilibrium solution

$$\frac{\partial^2 u}{\partial x^2} = 0$$

of this equation via relaxation with very long Δt since the system is unconditionally stable. This is a characteristic feature of implicit methods.

Of course, solving the diffusion equation with very long Δt will not necessarily give a very *accurate* time-dependent solution. If we want additional accuracy we can combine the stability of an implicit method with a method that is second order in space and time by forming the average of the BTCS and FTCS schemes:

$$\frac{u_j^{n+1} - u_j^n}{\Delta t} = \frac{D}{2}\left[\frac{u_{j+1}^{n+1} - 2u_j^{n+1} - u_{j-1}^{n+1} + u_{j+1}^n - 2u_j^n - u_{j-1}^n}{\Delta x^2}\right] \tag{18}$$

Both left and right hand sides are centered at timestep $n+\frac{1}{2}$ so the method is second-order accurate in time. The amplification factor is

$$\xi(k) = \frac{1 - 2\alpha \sin^2\left(\frac{k\Delta x}{2}\right)}{1 + 2\alpha \sin^2\left(\frac{k\Delta x}{2}\right)},$$

so this method, called *Crank-Nicolson* is also unconditionally stable for any Δt.

It is also common to variably center the timestep; instead of exactly at $n+\frac{1}{2}$, rather at $n+\theta$ where $0 \le \theta \le 1$, in which case the mehtod goes under the name of the 'θ'-mehtod.

3.1 Newton-Raphsons method

In greneral in implicit mehtods one is striving to solve a set of PDEs $\mathbf{f}$ which are functions of the unknowns at timestep $n+1$ (as well as the variables at timestep n or earlier) such that

$$\mathbf{f}(\rho_j^{n+1}, u_j^{n+1}, \ldots) = \mathbf{0}.$$

Define now the solution vector $\mathbf{x} \equiv (\rho^{n+1}, u^{n+1}, \ldots)$ and assume that we are close to the correct solution, *i.e.* that we can write

$$\mathbf{x} = \mathbf{x}' + \delta\mathbf{x}.$$

We can now expand around $\mathbf{x}$

$$\mathbf{f}(\mathbf{x}) = 0 = \mathbf{f}(\mathbf{x}') + \sum \frac{\partial f_i}{\partial x_j}\delta x_j + \mathcal{O}(\delta\mathbf{x}^2).$$

To arrive at the correct solution we must therefore solve for $\delta\mathbf{x}$

$$-\mathbf{f}(\mathbf{x}') = \sum \frac{\partial f_i}{\partial x_j}\delta x_j$$

iteratively at every timestep.