# Numerical Methods for the Solution of Partial Differential Equations

Luciano Rezzolla

*Institute for Theoretical Physics,*
*Frankfurt, Germany*

July 10, 2020

# Contents

## Acknowledgements

I am indebted to the several students who have helped me with the typing of the lectures notes into at TEXformat. They are too numerous to be reported here but my special thanks go to Olindo Zanotti for his help with the hyperbolic equations and to Gregor Leiler for his help with the parabolic equations and Chapter 3.6.

# Chapter 1

# Introduction

Let us consider a partial differential equation (PDE) of second-order in two dimensions $(x, y)$, which we can write generically as

$$a_{11}\frac{\partial^2 u}{\partial x^2} + 2a_{12}\frac{\partial^2 u}{\partial x \partial y} + a_{22}\frac{\partial^2 u}{\partial y^2} + f(x, y, u) = 0\,, \tag{1.1}$$

where $x, y$ are not all spatial coordinates and where we will assume the coefficients $a_{ij}$ to be functions of position only, i.e., $a_{ij} = a_{ij}(x, y)$. The PDE (1.1) is then said to be "linear with variable coefficients". On the other hand, the PDE (1.1) is said to be "quasi-linear " (or loosely speaking "nonlinear") if $a_{ij} = a_{ij}(x, y, u)$.

The traditional classification of partial differential equations is then based on the sign of the determinant $\Delta := a_{11}a_{22} - a_{12}^2$ that we can build with the coefficients of equation (1.1) and distinguishes three types of such equations. More specifically, equation (1.1) will be (see Table 1.1)

- (strictly) *hyperbolic* if $\Delta = 0$ has roots that are real and distinct.
- *parabolic* if $\Delta = 0$ has real but zero roots. will be
- *elliptic* if $\Delta = 0$ has complex roots.

Elliptic equations, on the other hand, describe *boundary value* problems, or **BVP**, since the space of relevant solutions $\Omega$ depends on the value that the solution takes on its boundaries $d\Omega$. Elliptic equations in physics are easily recognisable by the fact the solution does not depend on time coordinate $t$ and a prototype elliptic equation is in fact given by *Poisson equation* (*cf.* Table 1.1).

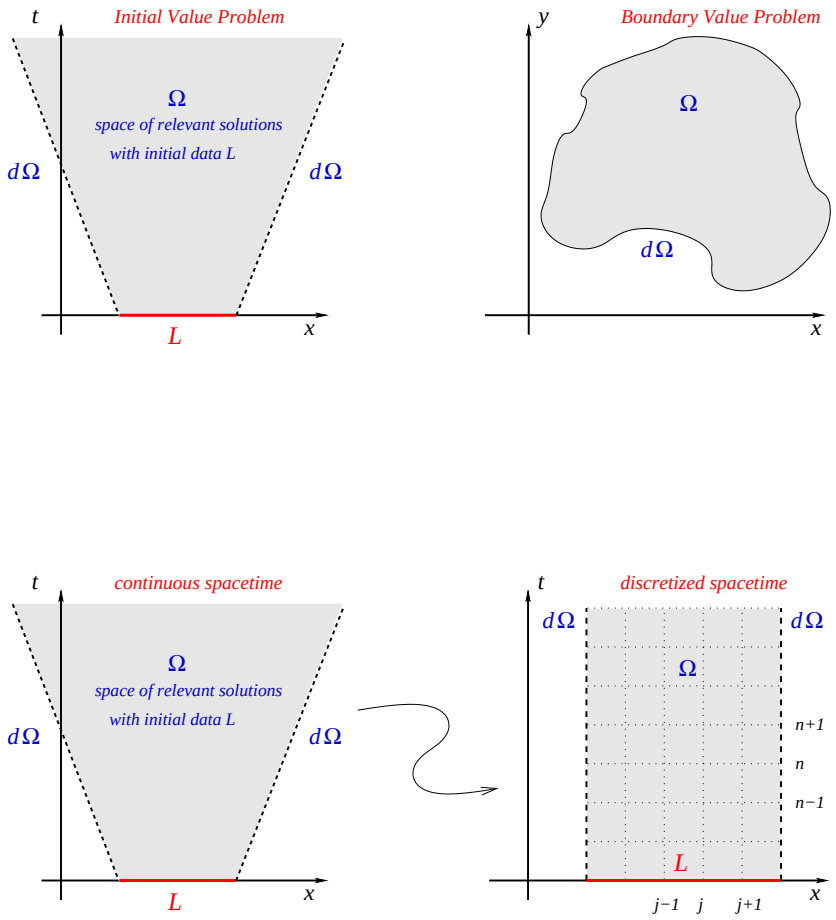


Figure 1.1: *Upper panel:* Schematic distinction between IVBPs and BVPs. *Lower Panel:* Schematic discretization of a hyperbolic IVBP

| *Type* | *Condition* | *Example (2 dimensions)* | |
|---|---|---|---|
| Hyperbolic | $a_{11}a_{22} - a_{12}^2 < 0$ | Wave equation: | $\frac{\partial^2 u}{\partial t^2} = v^2 \frac{\partial^2 u}{\partial x^2}$ |
| Parabolic | $a_{11}a_{22} - a_{12}^2 = 0$ | Diffusion equation: | $\frac{\partial u}{\partial t} = \frac{\partial}{\partial x}\left(D\frac{\partial u}{\partial x}\right)$ |
| Elliptic | $a_{11}a_{22} - a_{12}^2 > 0$ | Poisson equation: | $\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = \rho(x,y)$ |

Table 1.1: Schematic classification of a quasi-linear partial differential equation of second-order. For each class, a prototype equation is presented.

Hyperbolic and parabolic equations describe *initial value boundary* problems, or **IVBP**, since the space of relevant solutions $\Omega$ depends on the value that the solution $L$ (which we assume with compact support) takes on some initial time (see upper panel of Fig. 1.1). In practice, IVBP problems in physics are easily recognisable by the fact that the solution will depend on the time coordinate $t$. Very simple and useful examples of hyperbolic and parabolic equations are given by the *wave equation* and by the *diffusion equation*, respectively (*cf.* Table 1.1).

An important and physically-based difference between hyperbolic and parabolic equations becomes apparent by considering the "characteristic velocities" associated to them. These represent the velocities at which perturbations are propagated and have *finite* speeds in the case of hyperbolic equations, while these speeds are *infinite* in the case of parabolic equations. Since characteristic speeds are strictly real, they not defined for elliptic equations.

In this way, it is not difficult to appreciate that while both hyperbolic and parabolic equations describe time-dependent equations, the domain of dependence in a finite time for the two classes of equations can either be finite (as in the case of hyperbolic equations), or infinite (as in the case of parabolic equations).

## 1.1 The discretisation process

Given a set of partial differential equations of hyperbolic type, the corresponding *Cauchy* or *initial-value problem (IVP)* consists in finding a solution at an

arbitrary future time once the solution is known at an initial time, which is also referred to as the initial data. For simplicity, let us consider a well-posed initial-value problem in one spatial dimension and write it generically as (the generalisation to the multidimensional case is straightforward)

$$\mathcal{L}(u) - \mathcal{F} = 0\,, \tag{1.2}$$

where $u = u(x,t)$ is a *smooth* function in the two variables $x$ and $t$, $\mathcal{L}$ is a differential operator acting on $u$, and $\mathcal{F} = \mathcal{F}(u)$ is a function of $u$ only and not of its derivatives. To fix ideas, one could think that the generic expression (1.2) actually refers to an advection equation, so that $\mathcal{L}(u) = (\partial_t + v\partial_x)u$ and $\mathcal{F} = 0$.

Independently of the specific numerical method employed, the numerical solution of (1.2) consists of three "discretisation steps", *i.e.*,

- *Spacetime discretisation*: define a finite set of spacelike foliations of the spacetime "ordered" through the discrete time coordinate

$$t^n := t^0 + n\Delta t\,, \qquad\qquad n = 0,\, 1,\, \ldots,\, N_{\rm t}\,, \tag{1.3}$$

  where $\Delta t$ represents the separation between two spacelike foliations and can, in general, be a function of space and time. On each of such foliations, say the one at $t = t^n$, "order" the spatial positions through the discrete coordinates

$$x_j := x_0 + j\Delta x\,, \qquad\qquad j = 0,\, 1,\, \ldots,\, J\,, \tag{1.4}$$

  where we have simplified the notation, *i.e.*, $x^n_j \to x_j$ and where also $\Delta x$ can be a function of space and time. The set of spacetime points $\{x^n_k\}$ is also referred to as *gridpoints*. The points $x_0$ and $x_J$ mark the edges of the computational domain.

- *Variable discretisation*: replace the function $u(x,t)$ with a discrete set of values $\{u^n_j\}$ that approximate the *exact* pointwise values of $u$ at the gridpoints $\{x^n_j\}$, *i.e.*, $U^n_j$, through the gridfunction $\{u^n_j\}$ defined as

$$\{u^n_j\} \approx u\left(\{x^n_j\}\right) = u(x = x_j, t = t^n) =: \{U^n_j\}\,, \tag{1.5}$$

  with $n = 0,\, 1,\, \ldots,\, N_{\rm t}$, and $j = 0,\, 1,\, \ldots,\, J$. In this way, we can represent a generic solution $u(x,t)$ of the generic equation (1.2) in the continuum

spacetime with an infinite set of discretised solutions $\{u_j^n\}$, whose properties will depend both on the details of the discretisation (*i.e.*, on $\Delta t$ and $\Delta x$) and on the method used to discretise the differential operator (see below).

- *Operator discretisation*: replace the continuous differential operator $\mathcal{L}$ with a discretised one, $L_h$, that when applied to the gridfunction $\{u_j^n\}$ gives an approximation to $\mathcal{L}(u)$ in terms of algebraic combinations of the values $\{u_j^n\}$.

Through this discretisation process, the *continuum* initial-value problem (1.2) is replaced by the *discrete* initial-value problem

$$\mathcal{L}(u) - \mathcal{F} = 0 \qquad \longmapsto \qquad L_h\left(u_j^n\right) - F_h = 0\,, \tag{1.6}$$

that is, by a discrete representation of *both* the differential operator $\mathcal{L}$ and of the function $u$, where $h := \Delta x$. Note that the right-hand side of Eq. $(1.6)_2$ is zero because the differential operator acts on the numerical solution $u_j^n$, but this is no longer the case if the operator acts on the exact pointwise values of $u$ at the gridpoints $\{x_j^n\}$, *i.e.*,

$$L_h\left(U_j^n\right) - F_h \neq 0\,. \tag{1.7}$$

The amount by which it differs from zero is actually very important as it reflects the *error* made in the discretisation of the operator, whose significance will be clarified below [*cf.*, Eqs. (1.15), and (1.17)].

### 1.1.1 Spatial norms

A very useful tool in assessing the global properties of a discretised solution is offered by the (spatial) *discretised norms*. We recall that for a continuum function $u(x,t)$, smooth in the interval $x \in [a,b]$, the corresponding *p-norm* is defined as

$$\|u\|_p := \left(\frac{1}{(b-a)}\int_a^b |u(x,t)|^p dx\right)^{1/p}, \tag{1.8}$$

and has the same dimensions as the originating quantity $u(x,t)$. The extension of the definition (1.8) to a discretised space and time is straightforward and

yields the following discretised norms most commonly used

$$||u(t^n)||_1 = \frac{1}{J}\sum_{j=0}^{J}|u_j^n| = ||u(t^n)||\,, \qquad :: \text{ one–norm,} \tag{1.9}$$

$$||u(t^n)||_2 = \left(\frac{1}{J}\sum_{j=0}^{J}(u_j^n)^2\right)^{1/2}, \qquad :: \text{ two–norm,} \tag{1.10}$$

$$||u(t^n)||_p = \left(\frac{1}{J}\sum_{j=0}^{J}|u_j^n|^p\right)^{1/p}, \qquad :: \ p\text{–norm,} \tag{1.11}$$

$$||u(t^n)||_\infty = \max(|u_j^n|)\,, \qquad j = 0, \ldots J\,, \qquad :: \text{ infinity–norm.} \tag{1.12}$$

Note that the discretised two-norm $\|u\|_2$ effectively corresponds to a root mean square of the discretised solution $u_j^n$ and indeed it is often used as a measure of the average of the solution over the computational domain.

## 1.2 Numerical errors

Errors are an inevitable property of the numerical solution of a mathematical problem and their presence is not a nuisance as long as their origin is well understood and their behaviour matches the expected one. Like an experimental physicist, who has to determine all the sources of error in his measurements, so a computational physicist must determine all the contributions to his numerical solution that make it differ from the exact one, that is, the *numerical errors*. Three main errors will be discussed repeatedly in the following chapters and we briefly discuss them below.

### 1.2.1 Machine-precision error

The *machine-precision error* is a consequence of the fact that any machine will represent a rational number with a finite set of significant figures. It can be expressed in terms of the equality

$$\mathrm{fp}\,(1.0) = \mathrm{fp}\,(1.0) + \epsilon_{\mathrm{M}}\,, \tag{1.13}$$

where $\mathrm{fp}\,(1.0)$ is the floating-point representation of the number 1. Stated differently, the machine-precision error reflects the ability of the machine to dis-

tinguish two floating-point numbers and is therefore a genuine property of the machine.

### 1.2.2 Round-off error

The *round-off error* is the accumulation of machine-precision errors as a result of $N_{\text{FP}}$ floating-point operations. Because of the incoherent nature in which machine-precision errors add up, this error can be estimated to be

$$\epsilon_{\text{RO}} \approx \sqrt{N_{\text{FP}}}\epsilon_{\text{M}} \,. \tag{1.14}$$

When performing a numerical computation one should restrict the number of operations such that $\epsilon_{\text{RO}}$ is below the error at which the results need to be determined.

### 1.2.3 Truncation error

The *truncation error* (either *local* or *global*) is fundamentally different from the previous two types of errors in that it is entirely under human control and reflects the decision made in discretising the continuum problem. As we will discuss below, the truncation error is the most important tool to assess the correct discretisation of a system of partial differential equations, and it is therefore useful to dedicate a brief discussion to this concept.

For simplicity, we consider a hyperbolic system of partial differential equations in one dimension with a discretisation $\Delta t$ in time and $\Delta x$ in space. If $u(x,t)$ is the exact solution of the system (1.2) at $t = t^n$ and $x = x_j$, we can measure the difference between the exact solution of the continuum problem, $u(t^n, x_j)$, and its numerical counterpart $u_j^n$, that is, using Eqs. (1.2) and (1.6), we can define the *local truncation error* (or "residual") as [*cf.*, Eq. (1.6)]

$$(\epsilon^{(h)})_j^n := \left[L_{_h}\left(U_j^n\right) - F_{_h}\right] - \left[\mathcal{L}(u) - \mathcal{F}\right] = L_{_h}\left(U_j^n\right) - F_{_h} \,, \tag{1.15}$$

where $U_j^n := u(x = x_j, t = t^n)$ are the values of the continuum (exact) solution at the discretised locations in the spacetime [*cf.*, Eq. (1.5)]. In other words, the local truncation error measures the difference from zero when the discretised operators are applied to the exact solution. In this respect, it represents the

error we have selected when a specific mathematical choice has been made in the discretisation of the differential operator $\mathcal{L}$. In general, the local truncation error can be written as a combination of an error associated with the *time discretisation* and an error associated with the *spatial discretisation*, *i.e.*,

$$(\epsilon^{(h)})_j^n = \mathcal{O}\left(c_1 \Delta t^q + c_2 \Delta x^p\right) , \tag{1.16}$$

where $c_1$ and $c_2$ are assumed to be two constant coefficients. The discretised problem (1.3) is then said to have a *local order of accuracy* $r$, where $r = \min(p, q)$. Note that in the above definition we have assumed that $c_1 \sim c_2$; if the coefficients are very different, *e.g.*, $c_1 \ll c_2$, it is then possible that the order of accuracy is $p$ even if $p > q$. This is actually the case in most practical numerical simulations, where the time discretisation has a smaller order of accuracy, but where the use of small time-steps makes the spatial discretisation the dominant truncation error.

It is quite clear that the truncation error is totally under human judgement and its measure is essential to guarantee that the discretisation operation has been made properly and that the discretised problem is therefore a faithful representation of the continuum one, but for the truncation error. Let us elaborate further on this concept and simplify our notation a bit. In the vast majority of discretisation methods for hyperbolic problems, a constraint requires that the time and spatial discretisation are comparable,[1] *i.e.*, $\Delta x = \mathcal{O}(\Delta t)$, so that we do not need to distinguish between the space and time discretisation and just consider a generic discretisation interval $h := \Delta x \sim \Delta t$. Let us then drop the index $n$ referring to the time slice and indicate the local truncation error simply as

$$\epsilon_j^{(h)} = C h^{p_j} + \mathcal{O}(h^{p_j+1}) , \tag{1.17}$$

with $C$ a constant. Given the local numerical solution $u_j^{(h)}$ obtained with grid spacing $h$ and a discrete differential operator which is $p$-th order accurate at $x_j$, we can calculate the local error, $E_j^{(h)}$, as the difference between the exact and the numerical solution at $x_j^n$, *i.e.*,

$$E_j^{(h)} := U_j - u_j^{(h)} . \tag{1.18}$$

The local truncation error and the local error are clearly related and this relation is particularly simple to derive in the case of linear problem, where

$$E_j^{(h)} = (L_h)^{-1} \epsilon_j^{(h)} = C h^{\tilde{p}_j} + \mathcal{O}(h^{\tilde{p}_j+1}) , \tag{1.19}$$

[1] We will see below that this constraint is imposed by the *Courant–Friedrichs–Lewy (CFL) condition*

so that we can immediately obtain a measure of the local error simply in term of the spacing of the discretisation.[2] Note that the $E_j^{(h)}$ *measures* a numerical error and is thus proportional to $\tilde{p}_j$; in this sense it is fundamentally different from $\epsilon_j^{(h)}$, which represents instead a mathematical error and is proportional to $p_j$.

If a different solution is computed with a grid spacing $k < h$, it will then have, at the same time $t = t^n$ and spatial position $x_j$, a corresponding error $E_j^{(k)}$, so that we can introduce the *error ratio* as

$$R_j(h,k) := \frac{E_j^{(h)}}{E_j^{(k)}} = \frac{h^{\tilde{p}_j}}{k^{\tilde{p}_j}} + \mathcal{O}(h^{\tilde{p}_j+1}) \,, \tag{1.20}$$

from which we can compute the *numerical* local order of accuracy as

$$\tilde{p}_j := \frac{\log |R_j(h,k)|}{\log(h/k)} \,. \tag{1.21}$$

Note that $p$ and $\tilde{p}_j$ are conceptually similar but distinct. The first one represents the accuracy order in the continuum limit, while the second one is the accuracy order as *measured* from the numerical solution of the continuum problem at $x = x_j$. As we will comment later on, it is important to establish the relations between $p$ and $\tilde{p}_j$ as the resolution is changed. Assuming now that the two resolutions scale as $k = h/\gamma$, the error ratio (1.20) and the corresponding order of accuracy can be written respectively as

$$R_j(h, h/\gamma) = \gamma^{\tilde{p}_j} = 2^{\tilde{p}_j} \,, \quad \tilde{p}_j = \log |R_j(h, h/\gamma)| \,/\, \log(\gamma) = \log_2 |R_j| \,, \tag{1.22}$$

where the second equalities in (1.22) are written in the (rather common) case in which the grid spacing is simply halved, *i.e.*, $\gamma = 2$.

When defining the local error (1.18), we have assumed knowledge of the exact solution $U_j$, which, however, is in general not available. This does not represent a major obstacle and the local accuracy order can still be computed by simply employing a third (or more) numerical evaluation of the solution. We therefore exploit the fact that the difference between two numerical solutions does not depend on the actual exact solution and write

$$u_j^{(h)} - u_j^{(k)} = \left(U_j - E_j^{(h)}\right) - \left(U_j - E_j^{(k)}\right) = E_j^{(k)} - E_j^{(h)} \,, \tag{1.23}$$

[2] Of course, for Eq. (1.19) to be valid, the inverse operator $(L_h)^{-1}$ must not be singular for $h \to 0$; a similar relation can be found also in the case of nonlinear problems.

where, of course, the two solutions $u_j^{(h)}$ and $u_j^{(k)}$ should be evaluated at the same gridpoint $t = t^n, x = x_j$. If one of the numerical solutions is not available at such a point (*e.g.*, because the spacing used is not uniform) a suitable interpolation is needed and attention must be paid that the error it introduces is much smaller than either $E_j^{(h)}$ or $E_j^{(k)}$ in order not to spoil the measurement of the order of accuracy.

Using the definition (1.18) and three different numerical solutions $u_j^{(h)}$, $u_j^{(k)}$, $u_j^{(\ell)}$ with grid spacings $h, k$ and $\ell$ such that $\ell < k < h$, two different error ratios can then be defined as[3]

$$R_j(h,k;\ell) := \frac{u_j^{(h)} - u_j^{(\ell)}}{u_j^{(k)} - u_j^{(\ell)}} = \frac{E_j^{(h)} - E_j^{(\ell)}}{E_j^{(k)} - E_j^{(\ell)}} = \frac{h^{\tilde{p}_j} - \ell^{\tilde{p}_j}}{k^{\tilde{p}_j} - \ell^{\tilde{p}_j}}\,, \tag{1.24}$$

$$R_j(h,k,\ell) := \frac{u_j^{(h)} - u_j^{(k)}}{u_j^{(k)} - u_j^{(\ell)}} = \frac{E_j^{(h)} - E_j^{(k)}}{E_j^{(k)} - E_j^{(\ell)}} = \frac{h^{\tilde{p}_j} - k^{\tilde{p}_j}}{k^{\tilde{p}_j} - \ell^{\tilde{p}_j}}\,, \tag{1.25}$$

where in (1.24) we have taken the numerical solution $u_j^{(\ell)}$ with the associated error $E_j^{(\ell)}$ as the "reference" solution, since it is the one with the smallest error. Assuming again for concreteness the different resolutions have the same ratio $\gamma$, *i.e.*, that $k = h/\gamma$ and $\ell = k/\gamma = h/\gamma^2$, then the error ratios assume the simple expressions[4]

$$R_j(h, h/\gamma; h/\gamma^2) = \gamma^{\tilde{p}_j} + 1 = 2^{\tilde{p}_j} + 1\,, \qquad R_j(h, h/\gamma, h/\gamma^2) = \gamma^{\tilde{p}_j} = 2^{\tilde{p}_j}\,, \tag{1.26}$$

where, again, the second equalities in (1.26) refer to the case in which the grid spacing is halved. As a result, the corresponding orders of numerical accuracy can be computed equivalently as

$$\tilde{p}_j = \frac{\log|R_j(h, h/\gamma; h/\gamma^2) - 1|}{\log(\gamma)} = \log_2|R_j(h, h/2; h/4) - 1|\,, \tag{1.27}$$

or as

$$\tilde{p}_j = \frac{\log|R_j(h, h/\gamma, h/\gamma^2)|}{\log(\gamma)} = \log_2|R_j(h, h/2, h/4)|\,. \tag{1.28}$$

All of our considerations so far have been *"local"*, in the sense that both the truncation error $E_j^{(h)}$ and the order of accuracy $\tilde{p}_j$ have been computed at a

[3] Note the slight but important difference in the notation of Eqs. (1.24) and (1.25), *i.e.*, $R_j(h,k;\ell)$ and $R_j(h,k,\ell)$.

[4] If the resolutions are not in a constant ratio, a nonlinear equation needs to te solved via a root-finding algorithm.

representative spatial position $x = x_j$. Of course, such considerations should apply equally for any position in the computational grid and therefore also in a *"global"* sense, that is, when the truncation error and the order of accuracy are computed relative to a "global" measurement in terms of quantities that can be considered as spatial averages of the solution. Any representative spatially averaged measure can be used, *e.g.*, a volume integral of the solution, but particularly useful are the spatial norms introduced in the previous section. For any spatial norm, therefore, it will be possible to define a *global truncation error* as the extension of the definition (1.15), *i.e.*,

$$\epsilon^{(h)} := ||\epsilon_j^{(h)}|| = ||L_h\left(U_j^n\right) - F_h||\,, \tag{1.29}$$

as well as the *global error* as the extension of the definition (1.18), *i.e.*,

$$E^{(h)} := ||E_j^{(h)}|| = ||U_j - u_j^{(h)}||\,. \tag{1.30}$$

These global measurements can then be used to define a *global order of accuracy* as the extension of the definition (1.21), *i.e.*,

$$\tilde{p} := \frac{\log|R(h,k)|}{\log(h/k)}\,, \tag{1.31}$$

where now the *global* error ratio for two resolutions $h$ and $k$ is given by [*cf.*, (1.20)]

$$R(h,k) := \frac{E^{(h)}}{E^{(k)}} = \frac{h^{\tilde{p}}}{k^{\tilde{p}}} + \mathcal{O}(h^{\tilde{p}+1})\,, \tag{1.32}$$

and the errors $\epsilon(h)$ and $\epsilon(k)$ are computed via the norms (1.29). Following a logic which is identical to that followed before for the local order of accuracy, we can compute the global error ratio also when the exact solution is not known. More specifically, in this case, given three resolutions $h, k$ and $\ell$ with $k = h/\gamma$, $\ell = k/\gamma$, the *global orders of accuracy* can be computed equivalently as

$$\tilde{p} = \frac{\log|R(h,h/\gamma;h/\gamma^2) - 1|}{\log(\gamma)} = \log_2|R(h,h/2;h/4) - 1|\,, \tag{1.33}$$

or as

$$\tilde{p} = \frac{\log|R(h,h/\gamma,h/\gamma^2)|}{\log(\gamma)} = \log_2|R(h,h/2,h/4)|\,. \tag{1.34}$$

### 1.2.4 Consistency, convergence and stability

Many of the concepts and quantities introduced in the previous section represent the building blocks for two important definitions that will be presented here. Let us therefore go back to the hyperbolic partial differential equation [*cf.*, Eq. (1.2)] where, we recall, $\mathcal{L}$ is a quasi-linear differential operator and $\mathcal{F}$ is a generic source term that depends on $u$ but not on its derivatives. We indicate again with $L_{_h}$ the discretised representation of the continuum differential operator and with $\epsilon^{(h)}$ the associated global truncation error [*cf.*, Eq. (1.29)], which can be conveniently expressed as $\epsilon^{(h)} = Ch^p = \mathcal{O}(h^p)$, with $C$ a real constant coefficient [*cf.*, (1.17)]. The discretised representation $L_{_h}$ of the partial differential operator $\mathcal{L}(u)$ is then said to satisfy the global *consistency condition* if

$$\lim_{h\to 0} \epsilon^{(h)} = 0\,. \tag{1.35}$$

In addition, $L_{_h}$ is said to satisfy the global *convergence condition* if

$$\lim_{h\to 0} E^{(h)} = \lim_{h\to 0} Ch^p = 0\,, \tag{1.36}$$

where the first equality follows from taking the norm of Eq. (1.19), and where $p$ is then called the global convergence order. Note that local consistency conditions and local convergence can be defined in a very similar fashion by simply replacing $\epsilon^{(h)}$ with $\epsilon_j^{(h)}$ in Eq. (1.35), and $E^{(h)}$ with $E_j^{(h)}$ in Eq. (1.36), respectively. Of course, these changes would then lead to local convergence order $\tilde{p}_j$.

The convergence condition (1.36) can also be expressed in a different, more revealing way. Using in fact the definition of the numerical order of accuracy made in Eq. (1.21), the discretised operator $L_{_h}$ is said to be locally *convergent* if

$$\lim_{h\to 0} \tilde{p} := \frac{\log(E^{(h)})}{\log(Ch)} = p\,, \tag{1.37}$$

that is, if the accuracy order coincides with the convergence order, or, equivalently, if the measured numerical local truncation error coincides with the expected continuum one. Note that the convergence condition (1.36) is much more restrictive than the consistency one (1.35). While both require the local truncation error to decrease with increasing resolution and to vanish in the continuum limit, the convergence condition requires that this happens at a very precise rate, that is, the *convergence rate*. Therefore, consistency is a necessary condition for convergence, but not a sufficient one.

A few remarks are worth making. First, in practice, there will be a minimum resolution, $h_{\min}$, below which the truncation error will dominate over the others, *e.g.*, round-off error. Clearly, one should expect convergence only for $h < h_{\min}$ and the solution in this case is said to be in a convergent regime. Second, as already anticipated, the consistency and convergence conditions (1.35) and (1.36), which have been expressed above for the *global* truncation error, can be easily extended to the *local* truncation error following the logic behind expressions (1.15) and (1.21). Third, when validating the correct discretisation of a partial differential equation, the convergence condition (1.36) is verified by computing the numerical solution at different resolutions and by estimating the truncation error through the exact solution [*cf.*, Eq. (1.22)]. This is usually referred to as the *"convergence test"*, with two resolutions being sufficient.[5] If the exact solution is not known, it is sufficient to perform an additional measurement at a third resolution, comparing the three different truncation errors to estimate the order of accuracy [*cf.*, Eq. (1.24)]. This is usually referred to as the *"self-convergence test"*.

We conclude this section with an important theorem that brings together many of the concepts discussed so far and provides a unique interpretation for the interplay between consistency, convergence and stability. Indeed, the measurement of a convergent discretisation also has another important aspect, which requires, however, yet another definition. Let us consider again the discretised representation $L_h$ of the partial differential operator $\mathcal{L}(u)$ and recall that its application across a time interval $\Delta t$ introduces an associated truncation error $\epsilon(h)$. The evolution from time $t = 0$ to $t = t^n$ can then be thought of as the application $n$-times of the operator $L_h$ to the corresponding solution $u_j^m$ with $m = 1, \ldots, n$. The application of this operator should be such that the error accumulated does not grow unbounded and we express this requirement through the condition of numerical stability. More specifically, indicating with $L_h^n$ the $n$-th application of the operator $L_h$, the latter is said to be *numerically stable* if for each time $T = t^n$ there is a constant $C_s$ and a value $h_0$ such that

$$\|L_h^n\|_1 \leq C_s\,, \qquad \text{for all} \quad n\,h \leq T\,, \text{ and } h < h_0\,. \tag{1.38}$$

In essence, although our initial-value problem has been chosen to be well-posed, its discretisation can still lead to a solution that grows unbounded if an "unstable" numerical method is used. Hence, stability is a primary requirement for

[5] In practice, the truncation errors measured with the two resolutions are used to draw a straight line in a log-log plot of $\epsilon_j$ versus $h$, whose slope should match the expected one.

any discretised operator and the numerical solution of a well-posed initial-value problem is simply hopeless if performed with an unstable method.

Note that the operator is clearly stable if $\|L_h\|_1 \leq 1$, since $\|L_h^n\|_1 \leq \|L_h\|_1^n \leq 1$. In most practical situations, however, a certain growth is allowed, for instance if the solution intrinsically grows with time, so that the stability condition is enforced by requiring that $\|L_h\|_1 \leq 1 + \gamma h$, and

$$\|L_h\|_1^n \leq (1 + \gamma h)^n \leq e^{\gamma h n} \leq e^{\gamma T} . \tag{1.39}$$

Stated differently, the solution at later times is bounded to grow at most exponentially. With this definition in hand, we can state the following theorem

> **Theorem** *Given a well-posed initial-value linear problem and a finite-difference approximation to it that satisfies the consistency condition, stability is a necessary and sufficient condition for convergence.*

This theorem, known as the *Lax equivalence theorem* shows that for an initial-value problem which has been discretised with a consistent finite-difference operator (which we will introduce in detail in the next section), the concept of *stability* and *convergence* are interchangeable. In principle, therefore, proving that the numerical solution is convergent will not only validate that the discrete form of the equations represents a faithful representation of the continuum ones, but also that the solution will be bounded at all times. In practice, however, since the theorem strictly holds only for linear partial differential equations, it has a limited impact for most of the problems of physical interest. A proof of the Lax equivalence theorem can also be found in [11].

# Chapter 2

# Hyperbolic PDEs: Flux Conservative Formulation

It is often the case, when dealing with hyperbolic equations, that they can be formulated through conservation laws stating that a given quantity "$u$" is transported in space and time and is thus locally "conserved". The resulting "law of continuity" leads to equations which are called *conservative* and are of the type

$$\frac{\partial u}{\partial t} + \nabla \cdot \boldsymbol{F}(u) = 0\,, \tag{2.1}$$

where $u(\boldsymbol{x}, t)$ is the *density* of the conserved quantity, $\boldsymbol{F}$ the density flux and $\boldsymbol{x}$ a vector of spatial coordinates. In most of the physically relevant cases, the flux density $\boldsymbol{F}$ will not depend explicitly on $\boldsymbol{x}$ and $t$, but only implicitly through the density $u(\boldsymbol{x}, t)$, *i.e.*, $\boldsymbol{F} = \boldsymbol{F}(u(\boldsymbol{x}, t))$. The vector $\boldsymbol{F}$ is also called the *conserved flux* and takes this name from the fact that in the integral formulation of the conservation equation (2.1), the time variation of the integral of $u$ over a reference volume $V$ is indeed given by the net flux of $u$ across the surface enclosing $V$.

Generalising expression (2.1), we can consider a vector of densities $\boldsymbol{U}$ and write a set of conservation equations in the form

$$\frac{\partial \boldsymbol{U}}{\partial t} + \nabla \cdot \boldsymbol{F}(\boldsymbol{U}) = \boldsymbol{S}(\boldsymbol{U})\,. \tag{2.2}$$

Here, $\boldsymbol{S}(\boldsymbol{U})$ is a generic "source term" indicating the sources and sinks of the vector $\boldsymbol{U}$. The main property of the homogeneous equation (2.2) (*i.e.*, when $\boldsymbol{S}(\boldsymbol{U}) = 0$) is that the knowledge of the state-vector $\boldsymbol{U}(x, t)$ at a given point $x$

at time $t$ allows to determine the rate of flow, or flux, of each state variable at $(x, t)$.

Conservation laws of the form given by (2.2) can also be written as a quasi-linear form

$$\frac{\partial \boldsymbol{U}}{\partial t} + \boldsymbol{A}(U)\frac{\partial \boldsymbol{U}}{\partial x} = 0\,, \tag{2.3}$$

where $\boldsymbol{A}(U) := \partial \boldsymbol{F}/\partial \boldsymbol{U}$ is the Jacobian of the flux vector $\boldsymbol{F}(U)$.

The use of a conservation form of the equations is particularly important when dealing with problems admitting shocks or other discontinuities in the solution, *e.g.*, when solving the hydrodynamical equations. A non-conservative method, *i.e.*, a method in which the equations are not written in a conservative form, might give a numerical solution which appears perfectly reasonable but then yields incorrect results. A well-known example is offered by Burger's equation, *i.e.*, the momentum equation of an isothermal gas in which pressure gradients are neglected, and whose non-conservative representation fails dramatically in providing the correct shock speed if the initial conditions contain a discontinuity. Moreover, since the hydrodynamical equations follow from the physical principle of conservation of mass and energy-momentum, the most obvious choice for the set of variables to be evolved in time is that of the conserved quantities. It has been proved that non-conservative schemes do not converge to the correct solution if a shock wave is present in the flow, whereas conservative numerical methods, if convergent, do converge to the *weak solution* of the problem.

In the following, we will concentrate on numerical algorithms for the solution of hyperbolic partial differential equations written in the *conservative* form of equation (2.2). The advection and wave equations can be considered as prototypes of this class of equations in which with $\boldsymbol{S}(\boldsymbol{U}) = 0$ and will be used hereafter as our working examples.

# Chapter 3

# The advection equation in one dimension (1D)

A special class of conservative hyperbolic equations are the so called *advection equations*, in which the time derivative of the conserved quantity is proportional to its spatial derivative. In these cases, $\boldsymbol{F}(\boldsymbol{U})$ is diagonal and given by

$$\boldsymbol{F}(\boldsymbol{U}) = v\mathbf{I}\cdot\boldsymbol{U}\,, \tag{3.1}$$

where $\mathbf{I}$ is the identity matrix.

Because in this case the finite-differencing is simpler and the resulting algorithms are easily extended to more complex equations, we will use it as our "working example". More specifically, the advection equation for $u$ we will consider hereafter has, in 1D, the simple expression

$$\frac{\partial u}{\partial t} + v\frac{\partial u}{\partial x} = 0\,, \tag{3.2}$$

and admits the general analytic solution $u = f(x - vt)$, representing a wave moving in the positive $x$-direction if $v > 0$.

## 3.1 The 1D Upwind scheme: $\mathcal{O}(\Delta t, \Delta x)$

We will start making use of finite-difference techniques to derive a discrete representation of equation (3.2) by first considering the derivative in time. Taylor expanding the solution around $(x_j, t^n)$ we obtain

$$u(x_j, t^n + \Delta t) = u(x_j, t^n) + \frac{\partial u}{\partial t}(x_j, t^n)\Delta t + \mathcal{O}(\Delta t^2)\,, \tag{3.3}$$

or, equivalently,

$$u_j^{n+1} = u_j^n + \left.\frac{\partial u}{\partial t}\right|_j^n \Delta t + \mathcal{O}(\Delta t^2)\,. \tag{3.4}$$

Isolating the time derivative and dividing by $\Delta t$ we obtain

$$\left.\frac{\partial u}{\partial t}\right|_j^n = \frac{u_j^{n+1} - u_j^n}{\Delta t} + \mathcal{O}(\Delta t)\,. \tag{3.5}$$

Adopting a standard convention, we will consider the finite-difference representation of an $m$-th order *differential operator* $\partial^m/\partial x^m$ in the generic $x$-direction (where $x$ could either be a time or a spatial coordinate) to be of order $p$ if and only if

$$\left.\frac{\partial^m u}{\partial x^m}\right|_{x_j^n} = \left.\frac{\partial^m u}{\partial x^m}\right|_j^n = L_h(u_j^n) + \mathcal{O}(\Delta x^p)\,. \tag{3.6}$$

Of course, the time and spatial operators may have finite-difference representations with different orders of accuracy and in this case the overall order of the equation is determined by the differential operator with the largest truncation error.

Note also that while the truncation error is expressed for the differential operator, the numerical algorithms will not be expressed in terms of the differential operators and will therefore have different (usually smaller) truncation errors. This is clearly illustrated by the equations above, which show that the explicit solution (3.4) is of higher order than the finite-difference expression for the differential operator (3.5).

With this definition in mind, it is not difficult to realise that the finite-difference expression (3.5) for the time derivative is only first-order accurate in $\Delta t$. However, accuracy is not the most important requirement in numerical analysis and a first-order but stable scheme is greatly preferable to one which is higher order (*i.e.*, has a smaller truncation error) but is unstable.

In way similar to what we have done in (3.5) for the time derivative, we can derive a first-order, finite-difference approximation to the space derivative as

$$\left.\frac{\partial u}{\partial x}\right|_j^n = \frac{u_j^n - u_{j-1}^n}{\Delta x} + \mathcal{O}(\Delta x)\,. \tag{3.7}$$

While formally similar, the approximation (3.7) suffers of the ambiguity, not present in expression (3.5), that the first-order term in the Taylor expansion can be equally expressed in terms of $u_{j+1}^n$ and $u_j^n$, *i.e.*,

$$\left.\frac{\partial u}{\partial x}\right|_j^n = \frac{u_{j+1}^n - u_j^n}{\Delta x} + \mathcal{O}(\Delta x)\,. \tag{3.8}$$

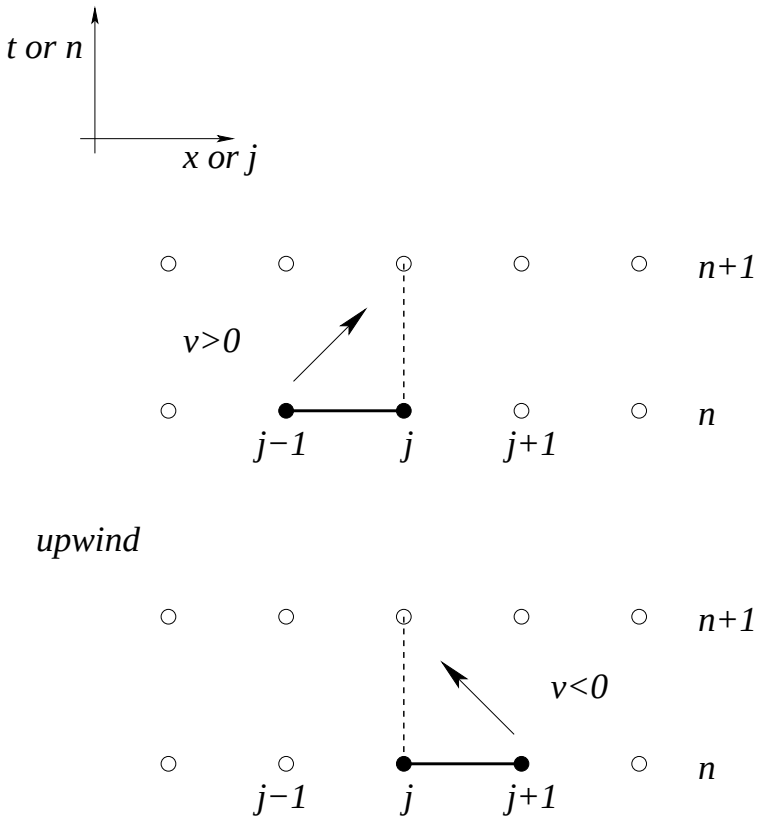


Figure 3.1: Schematic diagram of an UPWIND evolution scheme.

This ambiguity is the consequence of the first-order approximation which prevents a proper "centring" of the finite-difference stencil. However, and as long as we are concerned with an advection equation, this ambiguity is easily solved if we think that the differential equation will simply translate each point in the initial solution to the new position $x+v\Delta t$ over a time interval $\Delta t$. In this case, it is natural to select the points in the solution at the time-level $n$ that are "upwind" of the solution at the position $j$ and at the time-level $n+1$, as these are the ones causally connected with $u_j^{n+1}$. Depending then on the direction in which the solution is translated, and hence on the value of the advection velocity $v$, two different finite-difference representations can be given of equation (3.2)

and these are

$$\frac{u_j^{n+1}-u_j^n}{\Delta t} = -v\left(\frac{u_j^n - u_{j-1}^n}{\Delta x}\right) + \mathcal{O}(\Delta t, \Delta x)\,, \qquad \text{if } v>0\,, \tag{3.9}$$

$$\frac{u_j^{n+1}-u_j^n}{\Delta t} = -v\left(\frac{u_{j+1}^n - u_j^n}{\Delta x}\right) + \mathcal{O}(\Delta t, \Delta x)\,, \qquad \text{if } v<0\,, \tag{3.10}$$

respectively. As a result, the final finite-difference algorithms for determining the solution at the new time-level will have the form

$$u_j^{n+1} = u_j^n - \frac{v\Delta t}{\Delta x}(u_j^n - u_{j-1}^n) + \mathcal{O}(\Delta t^2, \Delta x \Delta t)\,, \qquad \text{if } v>0\,, \tag{3.11}$$

$$u_j^{n+1} = u_j^n - \frac{v\Delta t}{\Delta x}(u_{j+1}^n - u_j^n) + \mathcal{O}(\Delta t^2, \Delta x \Delta t)\,, \qquad \text{if } v<0\,. \tag{3.12}$$

More in general, for a system of linear hyperbolic equations with state vector $\mathbf{U}$ and flux-vector $\mathbf{F}$, the upwind scheme will take the form

$$\boldsymbol{U}_j^{n+1} = \boldsymbol{U}_j^n \pm \frac{\Delta t}{\Delta x}\left[\boldsymbol{F}_{j\mp 1}^n - \boldsymbol{F}_j^n\right] + \mathcal{O}(\Delta t^2, \Delta x \Delta t)\,, \tag{3.13}$$

where the $\pm$ sign should be chosen according to whether $v>0$ or $v<0$. The logic behind the choice of the stencil in an upwind method is is illustrated in Fig. 1.1 where we have shown a schematic diagram for the two possible values of the advection velocity.

The upwind scheme (as well as all of the others we will consider here) is an example of an *explicit* scheme, that is of a scheme where the solution at the new time-level $n+1$ can be calculated explicitly from the quantities that are already known at the previous time-level $n$. This is to be contrasted with an *implicit* scheme in which the finite-difference representations of the differential equation has, on the right-hand-side, terms at the new time-level $n+1$. These methods require in general the solution of a number of coupled algebraic equations and will not be discussed further here.

The upwind scheme is a stable one in the sense that the solution will not have exponentially growing modes. This can be seen through a *von Neumann stability analysis*, a useful tool which allows a first simple validation of a given numerical scheme. It is important to underline that the von-Neumann stability analysis is *local* in the sense that: *a)* it does not take into account boundary effects; *b)* it assumes that the coefficients of the finite-difference equations are sufficiently slowly varying to be considered constant in time and space (this is a reasonable assumptions if the equations are linear). Under these assumptions,

the solution can be seen as a sum of eigenmodes which at each grid point have the form

$$u_j^n = \xi^n e^{ikx_j}\,, \tag{3.14}$$

where $k$ is the spatial wave number and $\xi = \xi(k)$ is a *complex* number.

If we now consider the symbolic representation of the finite-difference equation as

$$u_j^{n+1} = \mathcal{T}(\Delta t^p, \Delta x^q) u_j^n\,, \tag{3.15}$$

with $\mathcal{T}(\Delta t^p, \Delta x^q)$ being the evolution operator of order $p$ in time and $q$ in space, it then becomes clear from (3.14) and (3.15) that the time evolution of a single eigenmode is nothing but a succession of integer powers of the complex number $\xi$ which is therefore named *amplification factor*. This naturally leads to a criterion of stability as the one for which the modulus of the amplification factor is always less or equal than 1, *i.e.*,

$$|\xi|^2 = \xi\xi^* \leq 1\,. \tag{3.16}$$

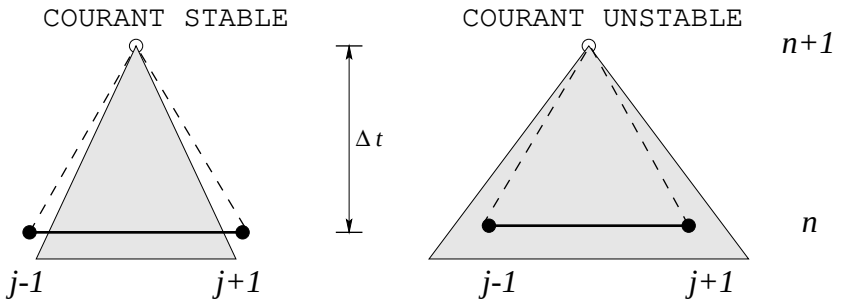


Figure 3.2: Schematic diagram of Courant stable and unstable choices of time-steps $\Delta t$. The two dashed lines limit the numerical domain of dependence of the solution at $x_j^{n+1}$, while the shaded area represents the physical domain of dependence. Stability is achieved when the first one is larger than the second one.

Using (3.14) in (3.11)–(3.12) we would obtain an amplification factor

$$\xi = 1 - |\alpha|\,(1 - \cos(k\Delta x)) - \mathrm{i}\alpha \sin(k\Delta x)\,, \tag{3.17}$$

where

$$\alpha := \frac{v\Delta t}{\Delta x}\,. \tag{3.18}$$

Its squared modulus $|\xi|^2 := \xi\xi^*$ is then

$$|\xi|^2 = 1 - 2\,|\alpha|\,(1 - |\alpha|)\,(1 - \cos(k\Delta x))\,, \tag{3.19}$$

so that the amplification factor (3.19) is less than one as long as the *Courant-Friedrichs-Löwy condition* (CFL condition)

$$|\alpha| \leq 1\,, \tag{3.20}$$

is satisfied (condition (3.20) is sometimes referred to simply as the Courant condition.). Note that in practice, the CFL condition (3.20) is used to determine the time-step $\Delta t$ once $v$ is known and $\Delta x$ has been chosen to achieve a certain accuracy, *i.e.*,

$$\Delta t = c_{\text{CFL}} \frac{\Delta x}{|v|}\,, \tag{3.21}$$

with $c_{\text{CFL}} < 1$ being the CFL factor. Expression (3.21) also allows a useful interpretation of the CFL condition.

From a *mathematical* point of view, the condition ensures that the numerical domain of dependence of the solution is *larger* than the physical one. From a *physical* point of view, on the other hand, the condition ensures that the propagation speed of any physical perturbation (*e.g.*, the sound speed, or the speed of light) is always smaller than the numerical one $v_{\text{N}} := \Delta x/\Delta t$, *i.e.*,

$$|v| = c_{\text{CFL}} \frac{\Delta x}{\Delta t} \leq v_{\text{N}} := \frac{\Delta x}{\Delta t}\,. \tag{3.22}$$

Equivalently, the CFL conditions prevents any physical signal to propagate for more than a fraction of a grid-zone during a single time-step (*cf.* Fig. 3.2)

As a final remark it should be noted that as described so far, the upwind method will yield satisfactory results only in the case in which the equations have an obvious transport character in one direction. However, in more general situations such as a wave equation, the upwind method will not be adequate and different expressions, based on finite-volume formulations of the equations will be needed [1, 4].

## 3.2 The 1D FTCS scheme: $\mathcal{O}(\Delta t, \Delta x^2)$

Let us consider again the advection equation (3.2) but we now finite difference with a more accurate approximation of the space derivative. To do this we can

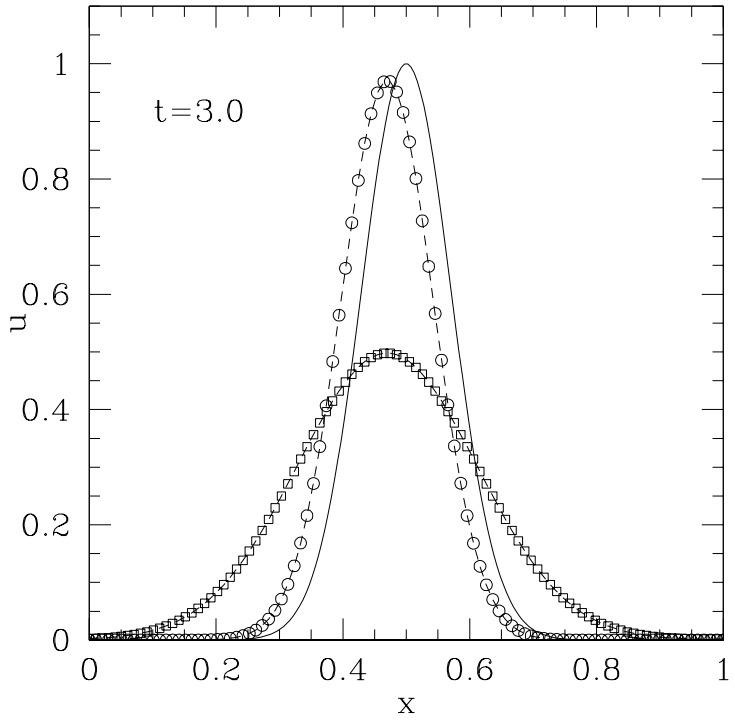


Figure 3.3: Time evolution of a Gaussian initially centred at $x = 0.5$ computed using an upwind scheme with $v = 10$ and 100 gridpoints. The analytic solution at time $t = 3$ is shown with a solid line the dashed lines are used to represent the numerical solution at the same time. Two different simulations are reported with the circles referring to a CFL factor $c_{\rm CFL} = 0.99$ and squares to a CFL factor $c_{\rm CFL} = 0.50$. Note how dissipation increases as the CFL is reduced.

calculate the two Taylor expansions in $x_j \pm \Delta x$

$$u(x_j + \Delta x, t^n) = u(x_j, t^n) + \frac{\partial u}{\partial x}(x_j, t^n)\Delta x + \frac{1}{2}\frac{\partial^2 u}{\partial x^2}(x_j, t^n)\Delta x^2 + \mathcal{O}(\Delta x^3)\,, \tag{3.23}$$

$$u(x_j - \Delta x, t^n) = u(x_j, t^n) - \frac{\partial u}{\partial x}(x_j, t^n)\Delta x + \frac{1}{2}\frac{\partial^2 u}{\partial x^2}(x_j, t^n)\Delta x^2 + \mathcal{O}(\Delta x^3)\,, \tag{3.24}$$

Subtracting now the two expressions and dividing by $2\Delta x$ we eliminate the first-order terms and obtain

$$\left.\frac{\partial u}{\partial x}\right|_j^n = \frac{u_{j+1}^n - u_{j-1}^n}{2\Delta x} + \mathcal{O}(\Delta x^2)\,, \tag{3.25}$$

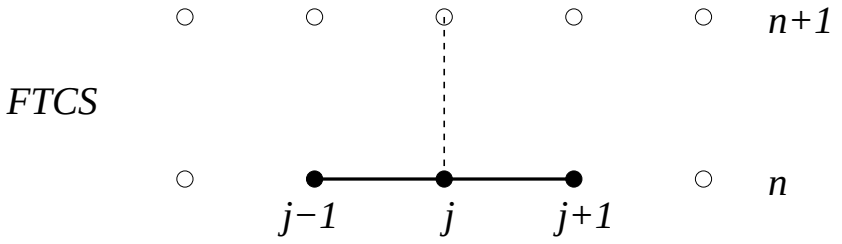


Figure 3.4: Schematic diagram of a FTCS evolution scheme.

Using now the second-order accurate operator (3.25) we can finite-difference equation (3.2) through the so called FTCS (Forward-Time-Centered-Space) scheme in which a first-order approximation is used for the time derivative, but a second order one for the spatial one. Using the a finite-difference notation, the FTCS is then expressed as

$$\frac{u_j^{n+1} - u_j^n}{\Delta t} = -v\left(\frac{u_{j+1}^n - u_{j-1}^n}{2\Delta x}\right) + \mathcal{O}(\Delta t, \Delta x^2)\,, \tag{3.26}$$

so that

$$u_j^{n+1} = u_j^n - \frac{\alpha}{2}(u_{j+1}^n - u_{j-1}^n) + \mathcal{O}(\Delta t^2, \Delta x^2 \Delta t)\,, \tag{3.27}$$

or more generically, for a system of linear hyperbolic equations

$$\boldsymbol{U}_j^{n+1} = \boldsymbol{U}_j^n - \frac{\Delta t}{2\Delta x}\left[\boldsymbol{F}_{j+1}^n - \boldsymbol{F}_{j-1}^n\right] + \mathcal{O}(\Delta t^2, \Delta x^2 \Delta t)\,, \tag{3.28}$$

The stencil for the finite- differencing (3.27) is shown symbolically in Fig. 3.4.

Disappointingly, the FTCS scheme is *unconditionally unstable*: *i.e.*, the numerical solution will be destroyed by numerical errors which will be certainly produced and grow exponentially. This is shown in Fig. 3.5 where we show the time evolution of a Gaussian using an FTCS scheme 100 gridpoints. The analytic solution at time $t = 0.3$ is shown with a solid line the dashed lines are used to represent the numerical solution at the same time. Note that the solution plotted here refers to a time which is 10 times smaller than the one in Fig. 3.3.

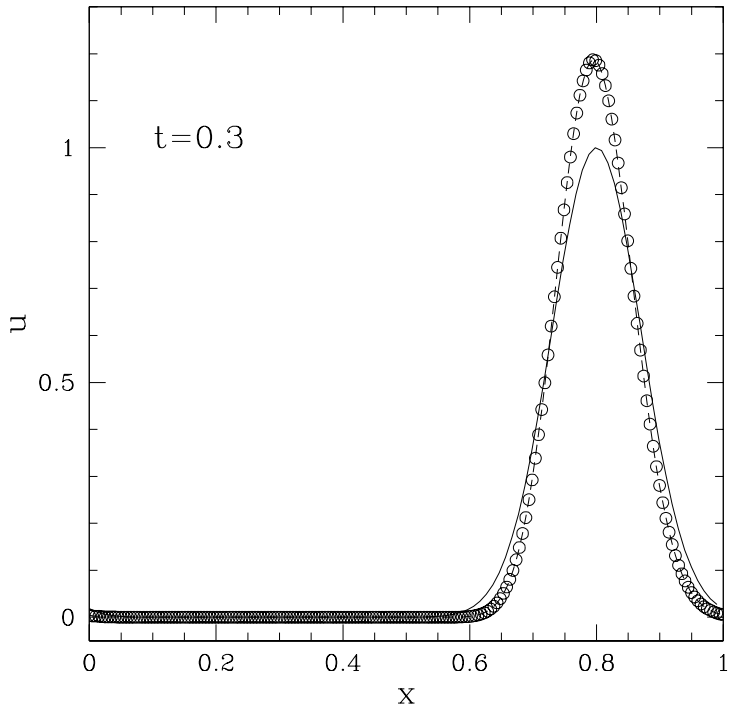


Figure 3.5: Time evolution of a Gaussian using an FTCS scheme with $v = 1$ and 100 gridpoints. The analytic solution at time $t = 0.3$ is shown with a solid line, while the dashed line is the numerical solution at the same time. Soon after $t \simeq 0.3$ the exponentially growing modes appear, rapidly destroying the solution.

Soon after $t \simeq 0.3$ the exponentially growing modes appear, rapidly destroying the solution.

Applying the definition (3.14) to equation (3.27) and few algebraic steps lead to an amplification factor

$$\xi = 1 - \mathrm{i}\alpha \sin(k\Delta x)\,. \tag{3.29}$$

whose squared modulus is

$$|\xi|^2 = 1 + (\alpha \sin(k\Delta x))^2 > 1\,, \tag{3.30}$$

thus proving the unconditional instability of the FTCS scheme. Because of this, the FTCS scheme is rarely used and will not produce satisfactory results but for

a very short timescale as compared to the typical crossing time of the physical problem under investigation.

A final aspect of the von-Neumann stability worth noticing is that it is a *necessary* but *not sufficient* condition for stability. In other words, a numerical scheme that appears stable with respect to a von-Neumann stability analysis might still be unstable.

## 3.3 The 1D Lax-Friedrichs scheme: $\mathcal{O}(\Delta t, \Delta x^2)$

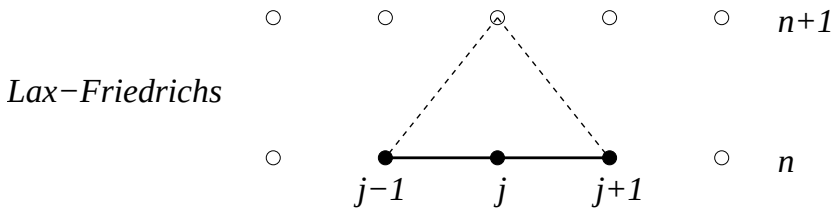


Figure 3.6: Schematic diagram of a Lax-Friedrichs evolution scheme.

A solution to the stability problems offered by the FTCS scheme was proposed by Lax and Friedrichs. The basic idea is very simple and is based on replacing, in the FTCS formula (3.27), the term $u_j^n$ with its spatial average, *i.e.*, $u_j^n = (u_{j+1}^n + u_{j-1}^n)/2$, so as to obtain for an advection equation

$$u_j^{n+1} = \frac{1}{2}(u_{j+1}^n + u_{j-1}^n) - \frac{\alpha}{2}(u_{j+1}^n - u_{j-1}^n) + \mathcal{O}(\Delta x^2)\,, \tag{3.31}$$

and, for a system of linear hyperbolic equations

$$\boldsymbol{U}_j^{n+1} = \frac{1}{2}(\boldsymbol{U}_{j+1}^n + \boldsymbol{U}_{j-1}^n) - \frac{\Delta t}{2\Delta x}\left[\boldsymbol{F}_{j+1}^n - \boldsymbol{F}_{j-1}^n\right] + \mathcal{O}(\Delta x^2)\,. \tag{3.32}$$

Note that the truncation error in equations (3.31) and (3.32) is reported to be $\mathcal{O}(\Delta x^2)$ and not $\mathcal{O}(\Delta t^2, \Delta x^2 \Delta t)$ because we are assuming that the CFL condition is satisfied and hence $\Delta t = \mathcal{O}(\Delta x)$. We will maintain this assumption hereafter.

The schematic diagram of a Lax-Friedrichs evolution scheme is shown in Fig. 3.6. Perhaps surprisingly, the algorithm (3.32) is now *conditionally stable* as can be verified through a von Neumann stability analysis. Proceeding as done for the FTCS scheme and using (3.14) in (3.32) we would obtain an amplification

factor whose modulus squared is

$$|\xi|^2 = 1 - \sin^2(k\Delta x)\left(1 - \alpha^2\right) , \tag{3.33}$$

which is $< 1$ as long as the CFL condition is satisfied.

Although not obvious, the correction introduced by the Lax-Friedrichs scheme is equivalent to the introduction of a *numerical dissipation* (viscosity). To see this, we rewrite (3.32) so that it clearly appears as a correction to (3.27):

$$\frac{u_j^{n+1} - u_j^n}{\Delta t} = -v\left(\frac{u_{j+1}^n - u_{j-1}^n}{2\Delta x}\right) + \frac{1}{2}\left(\frac{u_{j+1}^n - 2u_j^n + u_{j-1}^n}{\Delta t}\right) . \tag{3.34}$$

This is exactly the finite-difference representation of the equation

$$\frac{\partial u}{\partial t} + v\frac{\partial u}{\partial x} = \frac{1}{2}\left(\frac{\Delta x^2}{\Delta t}\right)\frac{\partial^2 u}{\partial x^2} , \tag{3.35}$$

where a diffusion term, $\propto \partial^2 u/\partial x^2$, has appeared on the right-hand-side. To prove this, we sum the two Taylor expansions (3.23)–(3.24) around $x_j$ to eliminate the first-order derivatives and obtain

$$\left.\frac{\partial^2 u}{\partial x^2}\right|_j^n = \frac{u_{j+1}^n - 2u_j^n + u_{j-1}^n}{\Delta x^2} + \mathcal{O}(\Delta x^2) , \tag{3.36}$$

where the sum has allowed us to cancel both the terms $\mathcal{O}(\Delta x)$ and $\mathcal{O}(\Delta x^3)$. Note that since the expression for the second derivative in (3.36) is $\mathcal{O}(\Delta x^2)$, it is appears multiplied by $\Delta x^2/\Delta t = \mathcal{O}(\Delta x)$ in equation (3.35), thus making the right-hand-side $\mathcal{O}(\Delta x^3)$ overall. The left-hand-side, on the other hand, is only $\mathcal{O}(\Delta x)$ (the time derivative is $\mathcal{O}(\Delta x)$, while the spatial derivative is $\mathcal{O}(\Delta x^2)$). As a result, the dissipative term goes to zero more rapidly than the intrinsic truncation error of the Lax-Friedrichs scheme, thus guaranteeing that the in the continuum limit the algorithm will converge to the correct solution of the advection equation.

A reasonable objection could be made for the fact that the Lax-Friedrichs scheme has changed the equation whose solution one is interested in [*i.e.*, Eq. (3.2)] into a new equation, in which a spurious numerical dissipation has been introduced [*i.e.*, Eq. (3.35)]. Unless $|v|\Delta t = \Delta x$, $|\xi| < 1$ and the amplitude of the wave is doomed to decrease (see Fig. 3.7).

However, such objection can be easily circumvented. As mentioned above, the dissipative term is always smaller than the truncation error thus guaranteeing the convergence to the correct solution. Furthermore, it is useful to bear

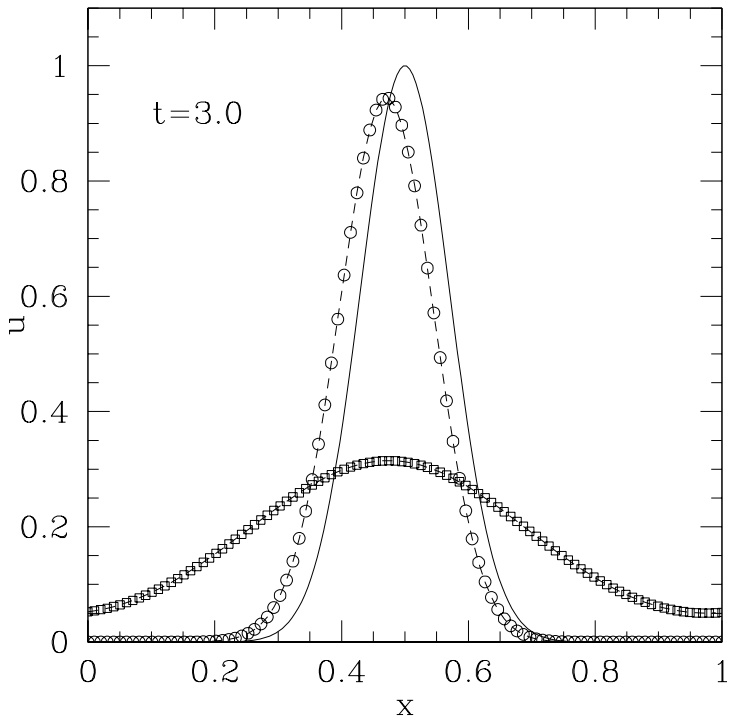


Figure 3.7: This is the same as in Fig. 3.3 but for a Lax-Friedrichs scheme. Note how the scheme is stable but also suffers from a considerable dissipation.

in mind that the key aspect in any numerical representation of a physical phenomenon is the determination of the length scale over which we need to achieve an accurate description. In a finite-difference approach, this length scale must necessarily encompass many grid points and for which $k\Delta x \ll 1$. In this case, expression (3.33) clearly shows that the amplification factor is very close to 1 and the effects of dissipation are therefore small. Note that this is true also for the FTCS scheme so that on these scales the stable and unstable schemes are equally accurate. On the very small scales however, which we are not of interest to us, $k\Delta x \sim 1$ and the stable and unstable schemes are radically different. The first one will be simply inaccurate, the second one will have exponentially growing errors which will rapidly destroy the whole solution. It is rather obvious that stability and inaccuracy are by far preferable to instability, especially if the accuracy is lost over wavelengths that are not of interest or when it can

be recovered easily by using more refined grids.

## 3.4 The 1D Leapfrog scheme: $\mathcal{O}(\Delta t^2, \Delta x^2)$

Both the FTCS and the Lax-Friedrichs are "one-level" schemes with first-order approximation for the time derivative and a second-order approximation for the spatial derivative. In those circumstances $v\Delta t$ should be taken significantly smaller than $\Delta x$ (to achieve the desired accuracy), well below the limit imposed by the Courant condition.

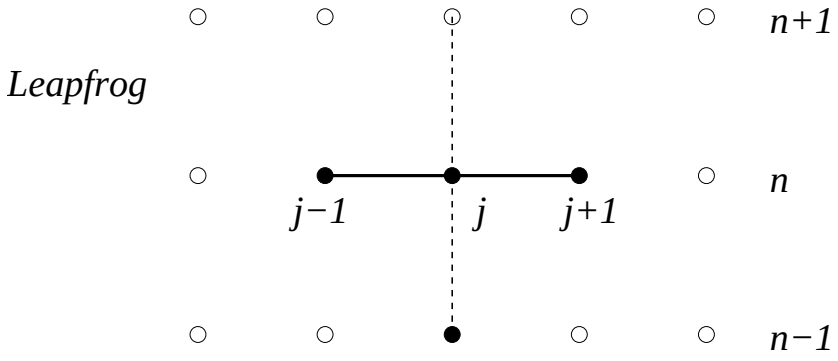


Figure 3.8: Schematic diagram of a Leapfrog evolution scheme.

Second-order accuracy in time can be obtained if we insert

$$\left.\frac{\partial u}{\partial t}\right|_j^n = \frac{u_j^{n+1} - u_j^{n-1}}{2\Delta t} + \mathcal{O}(\Delta t^2)\,, \tag{3.37}$$

in the FTCS scheme, to find the *Leapfrog* scheme

$$u_j^{n+1} = u_j^{n-1} - \alpha\left(u_{j+1}^n - u_{j-1}^n\right) + \mathcal{O}(\Delta x^2)\,, \tag{3.38}$$

where it should be noted that the factor 2 in $\Delta x$ cancels the equivalent factor 2 in $\Delta t$.

For a set of linear equations, the Leapfrog scheme simply becomes

$$\boldsymbol{U}_j^{n+1} = \boldsymbol{U}_j^{n-1} - \frac{\Delta t}{\Delta x}\left[\boldsymbol{F}_{j+1}^n - \boldsymbol{F}_{j-1}^n\right] + \mathcal{O}(\Delta x^2)\,, \tag{3.39}$$

and the schematic diagram of a Leapfrog evolution scheme is shown in Fig. 3.8.

Also for the case of a Leapfrog scheme there are a number of aspects that should be noticed:

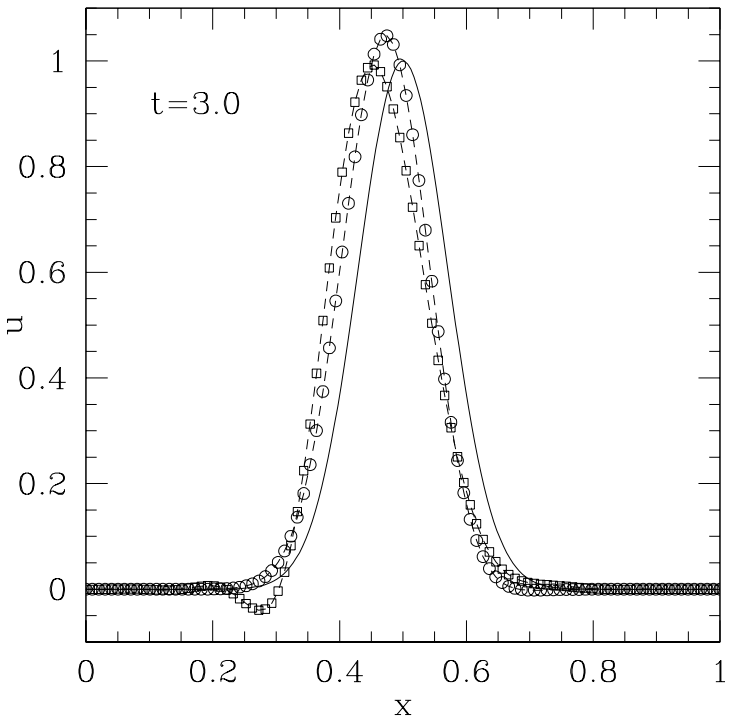


Figure 3.9: This is the same as in Fig. 3.3 but for a Leapfrog scheme. Note how the scheme is stable and does not suffers from a considerable dissipation even for low CFL factors. However, the presence of a little "dip" in the tail of the Gaussian for the case of $c_{\rm CFL} = 0.5$ is the result of the dispersive nature of the numerical scheme.

- In a Leapfrog scheme that is Courant stable, there is no amplitude dissipation (*i.e.*, $|\xi|^2 = 1$). In fact, a von Neumann stability analysis yields

$$\xi = -\mathrm{i}\alpha \sin(k\Delta x) \pm \sqrt{1 - [\alpha \sin(k\Delta x)]^2}\,, \tag{3.40}$$

  and so that

$$|\xi|^2 = \alpha^2 \sin^2(k\Delta x) + \{1 - [\alpha \sin(k\Delta x)]^2\} = 1 \qquad \forall\ \alpha \leq 1\,. \tag{3.41}$$

  As a result, the squared modulus of amplification factor is always 1, provided the CFL condition is satisfied (*cf.* Fig. 3.11).

- The Leapfrog scheme is a two-level scheme, requiring records of values at time-steps $n$ and $n-1$ to get values at time-step $n+1$. This is clear from

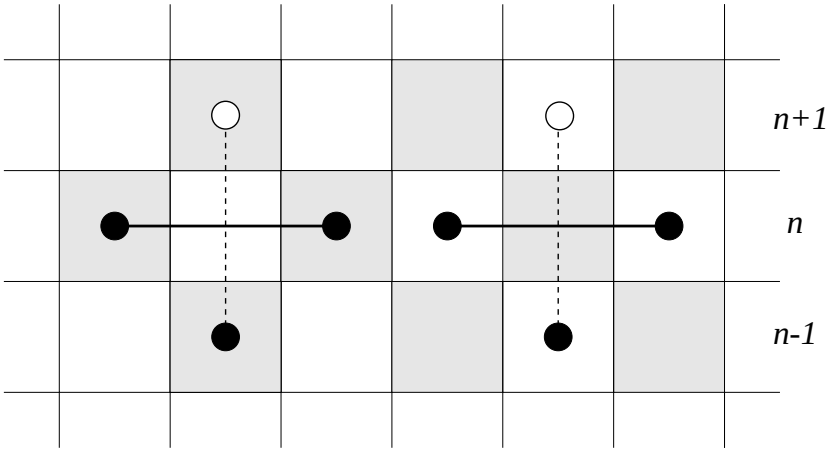


Figure 3.10: Schematic diagram of the decoupled grids in a Leapfrog evolution scheme.

expression (5.22) and cannot be avoided by means of algebraic manipulations.

- The major disadvantage of this scheme is that odd and even mesh points are completely decoupled (see Fig. 8).

  In principle, the solutions on the black and white squares are identical. In practice, however, their differences increase as the time progresses. This effect, which becomes evident only on timescales much longer then the crossing timescale, can be cured either by discarding one of the solutions or by adding a dissipative term of the type

$$\ldots + \epsilon(u^n_{j+1} - 2u^n_{j+1} + u^n_{j+1}) \,, \tag{3.42}$$

  in the right-hand-side of (5.17), where $\epsilon \ll 1$ is an adjustable coefficient.

## 3.5 The 1D Lax-Wendroff scheme: $\mathcal{O}(\Delta t^2, \Delta x^2)$

The Lax-Wendroff scheme is the second-order accurate extension of the Lax-Friedrichs scheme. As for the case of the Leapfrog scheme, in this case too we need two time-levels to obtain the solution at the new time-level.

There are a number of different ways of deriving the Lax-Wendroff scheme but it is probably useful to look at it as to a combination of the Lax-Friedrichs

scheme and of the Leapfrog scheme. In particular a Lax-Wendroff scheme can be obtained as

1. A Lax-Friedrichs scheme with half step:

$$\boldsymbol{U}_{j+\frac{1}{2}}^{n+\frac{1}{2}} = \frac{1}{2}\left[\boldsymbol{U}_{j+1}^{n} + \boldsymbol{U}_{j}^{n}\right] - \frac{\Delta t}{2\Delta x}\left[\boldsymbol{F}_{j+1}^{n} - \boldsymbol{F}_{j}^{n}\right] + \mathcal{O}(\Delta x^2)\,,$$

$$\boldsymbol{U}_{j-\frac{1}{2}}^{n+\frac{1}{2}} = \frac{1}{2}\left[\boldsymbol{U}_{j}^{n} + \boldsymbol{U}_{j-1}^{n}\right] - \frac{\Delta t}{2\Delta x}\left[\boldsymbol{F}_{j}^{n} - \boldsymbol{F}_{j-1}^{n}\right] + \mathcal{O}(\Delta x^2)\,,$$

   where $\Delta t/(2\Delta x)$ comes from having used a timestep $\Delta t/2$;

2. The evaluation of the fluxes $\mathbf{F}_{j\pm\frac{1}{2}}^{n+\frac{1}{2}}$ from the values of $\boldsymbol{U}_{j\pm\frac{1}{2}}^{n+\frac{1}{2}}$

3. A Leapfrog "half-step":

$$\boldsymbol{U}_{j}^{n+1} = \boldsymbol{U}_{j}^{n} - \frac{\Delta t}{\Delta x}\left[\boldsymbol{F}_{j+\frac{1}{2}}^{n+\frac{1}{2}} - \boldsymbol{F}_{j-\frac{1}{2}}^{n+\frac{1}{2}}\right] + \mathcal{O}(\Delta x^2)\,. \tag{3.43}$$

The schematic diagram of a Lax-Wendroff evolution scheme is shown in Fig. 3.11 and the application of this scheme to the advection equation (3.2) is straightforward. More specifically, the "half-step" values can be calculated as

$$u_{j\pm1/2}^{n+1/2} = \frac{1}{2}\left(u_{j}^{n} + u_{j\pm1}^{u}\right) \mp \frac{\alpha}{2}\left(u_{j\pm1}^{n} - u_{j}^{n}\right) + \mathcal{O}(\Delta x^2)\,, \tag{3.44}$$

so that the solution at the new time-level will then be

$$u_{j}^{n+1} = u_{j}^{n} - \alpha\left(u_{j+1/2}^{n+1/2} - u_{j-1/2}^{n+1/2}\right) + \mathcal{O}(\Delta x^2) \tag{3.45}$$

$$= u_{j}^{n} - \frac{\alpha}{2}\left(u_{j+1}^{n} - u_{j-1}^{n}\right) + \frac{\alpha^2}{2}\left(u_{j+1}^{n} - 2u_{j}^{n} + u_{j-1}^{n}\right) + \mathcal{O}(\Delta x^2)\,. \tag{3.46}$$

where expression (3.46) has been obtained after substituting (3.44) in (3.45).

Aspects of a Lax-Wendroff scheme worth noticing are:

- In the Lax-Wendroff scheme there might be some amplitude dissipation. In fact, a von-Neumann stability analysis yields

$$\xi = 1 - \mathrm{i}\alpha\sin(k\Delta x) - \alpha^2\left[1 - \cos(k\Delta x)\right]\,, \tag{3.47}$$

  so that the squared modulus of the amplification factor is

$$|\xi|^2 = 1 - \alpha^2(1-\alpha^2)\left[1 - \cos^2(k\Delta x)\right]\,. \tag{3.48}$$

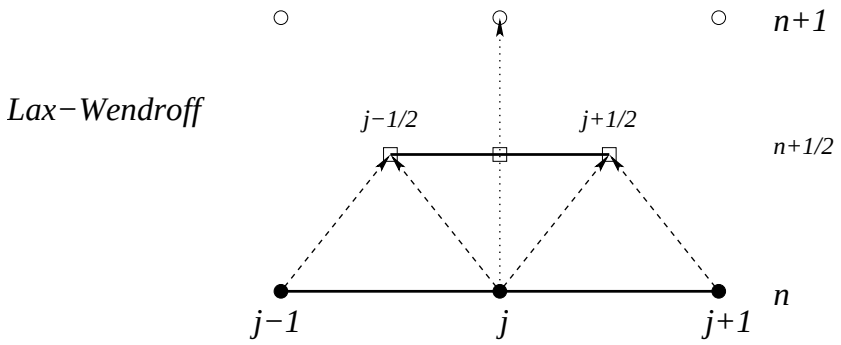


Figure 3.11: Schematic diagram of a Lax-Wendroff evolution scheme.

As a result, the von-Neumann stability criterion $|\xi|^2 \leq 1$ is satisfied as long as $\alpha^2 \leq 1$, or equivalently, as long as the CFL condition is satisfied. (*cf.* Fig. 10). It should be noticed, however, that unless $\alpha^2 = 1$, then $|\xi|^2 < 1$ and some amplitude dissipation is present. In this respect, the dissipative properties of the Lax-Friedrichs scheme are not completely lost in the Lax-Wendroff scheme but are much less severe (*cf.* Figs. 5 and 10).

- The Lax-Wendroff scheme is a two-level scheme, but can be recast in a one-level form by means of algebraic manipulations. This is clear from expressions (3.46) where quantities at time-levels $n$ and $n+1$ only appear.

## 3.6 The 1D ICN scheme: $\mathcal{O}(\Delta t^2, \Delta x^2)$

The idea behind the *iterative Crank-Nicolson* (ICN) scheme is that of transforming a stable implicit method, *i.e.*, the Crank-Nicolson (CN) scheme (see Sect. 8.5.2) into an explicit one through a series of iterations. To see how to do this in practice, consider differencing the advection equation (3.2) having a centred space derivative but with the time derivative being backward centred, *i.e.*,

$$\frac{u_j^{n+1} - u_j^n}{\Delta t} = -v \left( \frac{u_{j+1}^{n+1} - u_{j-1}^{n+1}}{2\Delta x} \right) . \tag{3.49}$$

This scheme is also known as "backward in time, centred in space" or BTCS (see Sect. 8.5.1) and has amplification factor

$$\xi = \frac{1}{1 + \mathrm{i}\alpha \sin k\Delta x} , \tag{3.50}$$

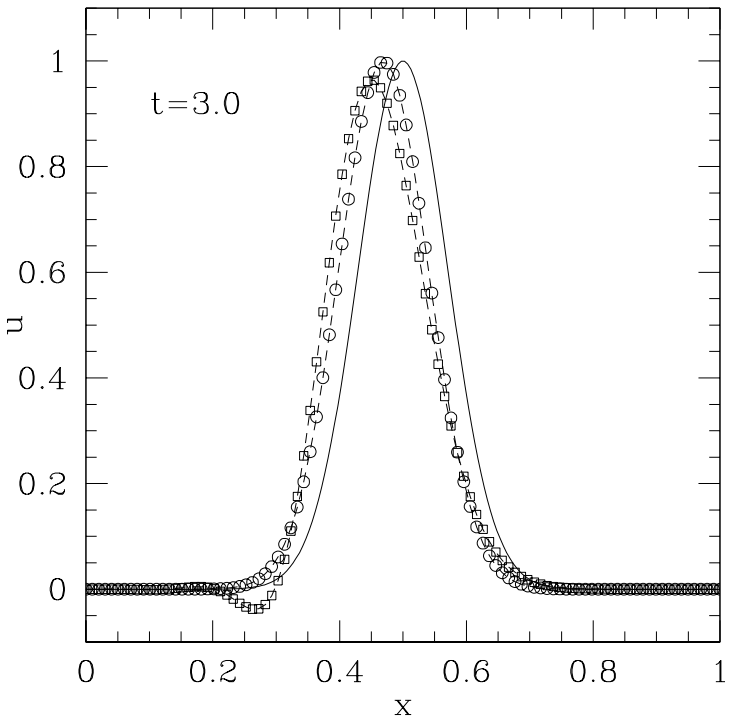


Figure 3.12: This is the same as in Fig. 3.3 but for a Lax-Wendroff scheme. Note how the scheme is stable and does not suffers from a considerable dissipation even for low CFL factors. However, the presence of a little "dip" in the tail of the Gaussian for the case of $c_{\rm CFL} = 0.5$ is the result of the dispersive nature of the numerical scheme.

so that $|\xi|^2 < 1$ for any choice of $\alpha$, thus making the method unconditionally stable.

The *Crank-Nicolson* (CN) scheme, instead, is a second-order accurate method obtained by averaging a BTCS and a FTCS method or, in other words, equations (3.26) and (3.49). Doing so one then finds

$$\xi = \frac{1 + {\rm i}\alpha \sin k\Delta x/2}{1 - {\rm i}\alpha \sin k\Delta x/2}\,. \tag{3.51}$$

so that the method is stable. Note that although one averages between an explicit and an implicit scheme, terms containing $u^{n+1}$ survive on the right-hand-side of equation (3.49), thus making the CN scheme implicit.

The first iteration of iterative Crank-Nicolson starts by calculating an inter-

mediate variable ${}^{(1)}\tilde{u}$ using equation (3.26):

$$\frac{{}^{(1)}\tilde{u}_j^{n+1} - u_j^n}{\Delta t} = -v\left(\frac{u_{j+1}^n - u_{j-1}^n}{2\Delta x}\right). \tag{3.52}$$

Then another intermediate variable ${}^{(1)}\bar{u}$ is formed by averaging:

$${}^{(1)}\bar{u}_j^{n+1/2} := \frac{1}{2}\left({}^{(1)}\tilde{u}_j^{n+1} + u_j^n\right). \tag{3.53}$$

Finally the timestep is completed by using equation (3.26) again with $\bar{u}$ on the right-hand side:

$$\frac{u_j^{n+1} - u_j^n}{\Delta t} = -v\left(\frac{{}^{(1)}\bar{u}_{j+1}^{n+1/2} - {}^{(1)}\bar{u}_{j-1}^{n+1/2}}{2\Delta x}\right). \tag{3.54}$$

Iterated Crank-Nicolson with *two iterations* is carried out in much the same way. After steps (3.52) and (3.53), we calculate

$$\frac{{}^{(2)}\tilde{u}_j^{n+1} - u_j^n}{\Delta t} = -v\left(\frac{{}^{(1)}\bar{u}_{j+1}^{n+1/2} - {}^{(1)}\bar{u}_{j-1}^{n+1/2}}{2\Delta x}\right), \tag{3.55}$$

$${}^{(2)}\bar{u}_j^{n+1/2} := \frac{1}{2}\left({}^{(2)}\tilde{u}_j^{n+1} + u_j^n\right). \tag{3.56}$$

Then the final step is computed analogously to equation (3.54):

$$\frac{u_j^{n+1} - u_j^n}{\Delta t} = -v\left(\frac{{}^{(2)}\bar{u}_{j+1}^{n+1/2} - {}^{(2)}\bar{u}_{j-1}^{n+1/2}}{2\Delta x}\right). \tag{3.57}$$

Further iterations can be carried out following the same logic.

To investigate the stability of these iterated schemes we compute the amplification factors relative to the different iterations to be

$${}^{(1)}\xi = 1 + 2\mathrm{i}\beta\,, \tag{3.58}$$

$${}^{(2)}\xi = 1 + 2\mathrm{i}\beta - 2\beta^2\,, \tag{3.59}$$

$${}^{(3)}\xi = 1 + 2\mathrm{i}\beta - 2\beta^2 - 2\mathrm{i}\beta^3\,, \tag{3.60}$$

$${}^{(4)}\xi = 1 + 2\mathrm{i}\beta - 2\beta^2 - 2\mathrm{i}\beta^3 + 2\beta^4\,, \tag{3.61}$$

where $\beta := (\alpha/2)\sin(k\Delta x)$, and ${}^{(1)}\xi$ corresponds to the FTCS scheme. Note that the amplification factors (3.58) correspond to those one would obtain by expanding equation (3.51) in powers of $\beta$.

Computing the squared moduli of (3.58) one encounters an alternating and recursive pattern. In particular, iterations 1 and 2 are unstable ($|\xi|^2 > 1$);

iterations 3 and 4 are stable ($|\xi|^2 < 1$) provided $\beta^2 \leq 1$; iterations 5 and 6 are also unstable; iterations 7 and 8 are stable provided $\beta^2 \leq 1$; and so on. Imposing the stability for all wavenumbers $k$, we obtain $\alpha^2/4 \leq 1$, or $\Delta t \leq 2\Delta x$ which is just the CFL condition [the factor 2 is inherited by the factor 2 in equation (3.26)].

In other words, while the magnitude of the amplification factor for the iterated Crank-Nicolson scheme does approach 1 as the number of iterations becomes infinite, the convergence is not monotonic. The magnitude oscillates above and below 1 with ever decreasing oscillations. All the iterations leading to $|\xi|^2$ above 1 are unstable, although the instability might be very slowly growing as the number of iterations increases. Because the truncation error is not modified by the number of iterations and is always $\mathcal{O}(\Delta t^2, \Delta x^2)$, a number of iterations larger than two is never useful; three iterations, in fact, would simply amount to a larger computational cost.

### 3.6.1 ICN as a $\theta$-method

In the ICN method the $M$-th average is made weighting equally the newly predicted solution ${}^{(M)}\tilde{u}_j^{n+1}$ and the solution at the "old" timelevel" $u^n$. This, however, can be seen as the special case of a more generic averaging of the type

$$ {}^{(M)}\bar{u}^{n+1/2} = \theta \, {}^{(M)}\tilde{u}^{n+1} + (1-\theta)u^n \,, \tag{3.62} $$

where $0 < \theta < 1$ is a constant coefficient. Predictor-corrector schemes using this type of averaging are part of a large class of algorithms named $\theta$-*methods* [11], and we refer to the ICN generalized in this way as to the "$\theta$-ICN" method.

A different and novel generalisation of the $\theta$-ICN can be obtained by *swapping* the averages between two subsequent corrector steps, so that in the $M$-th corrector step

$$ {}^{(M)}\bar{u}^{n+1/2} = (1-\theta) \, {}^{(M)}\tilde{u}^{n+1} + \theta u^n \,, \tag{3.63} $$

while in the $(M+1)$-th corrector step

$$ {}^{(M+1)}\bar{u}^{n+1/2} = \theta \, {}^{(M+1)}\tilde{u}^{n+1} + (1-\theta)u^n \,. \tag{3.64} $$

Note that as long as the number of iterations is even, the sequence in which the averages are computed is irrelevant. Indeed, the weights $\theta$ and $1-\theta$ in eqs. (3.63)–(3.64) could be inverted and all of the relations discussed hereafter for the swapped weighted averages would continue to hold after the transformation $\theta \to 1-\theta$.

### Constant Arithmetic Averages

Using a von-Neumann stability analysis, Teukolsky has shown that for a hyperbolic equation the ICN scheme with $M$ iterations has an amplification factor [14]

$$^{(M)}\xi = 1 + 2\sum_{n=1}^{M} (-\mathrm{i}\beta)^n \,, \tag{3.65}$$

where $\beta := v[\Delta t/(2\Delta x)]\sin(k\Delta x)$ [1]. More specifically, zero and one iterations yield an unconditionally unstable scheme, while two and three iterations a stable one provided that $\beta^2 \leq 1$; four and five iterations lead again to an unstable scheme and so on. Furthermore, because the scheme is second-order accurate from the first iteration on, Teukolsky's suggestion when using the ICN method for hyperbolic equations was that two iterations should be used *and no more* [14]. This is the number of iterations we will consider hereafter.

### Constant Weighted Averages

Performing the same stability analysis for a $\theta$-ICN is only slightly more complicated and truncating at two iterations the amplification factor is found to be

$$\xi = 1 - 2\mathrm{i}\beta - 4\beta^2\theta + 8\mathrm{i}\beta^3\theta^2 \quad , \tag{3.66}$$

where $\xi$ is a shorthand for $^{(2)}\xi$. The stability condition in this case translates into requiring that

$$16\beta^4\theta^4 - 4\beta^2\theta^2 - 2\theta + 1 \leq 0 \,, \tag{3.67}$$

or, equivalently, that for $\theta > 3/8$

$$\frac{\sqrt{\frac{1}{2} - \sqrt{2\theta - \frac{3}{4}}}}{2\theta} \leq \beta \leq \frac{\sqrt{\frac{1}{2} + \sqrt{2\theta - \frac{3}{4}}}}{2\theta} \,, \tag{3.68}$$

which reduces to $\beta^2 \leq 1$ when $\theta = 1/2$. Because the condition (3.68) must hold for every wavenumber $k$, we consider hereafter $\beta := v\Delta t/(2\Delta x)$ and show in the left panel of Fig. 3.13 the region of stability in the $(\theta, \beta)$ plane. The thick solid lines mark the limit at which $|\xi| = 1$, while the dotted contours indicate the different values of the amplification factor in the stable region.

A number of comments are worth making. Firstly, although the condition (3.68) allows for weighting coefficients $\theta < 1/2$, the $\theta$-ICN is stable only if

[1] Note that we define $\beta$ to have the opposite sign of the corresponding quantity defined in ref. [14]

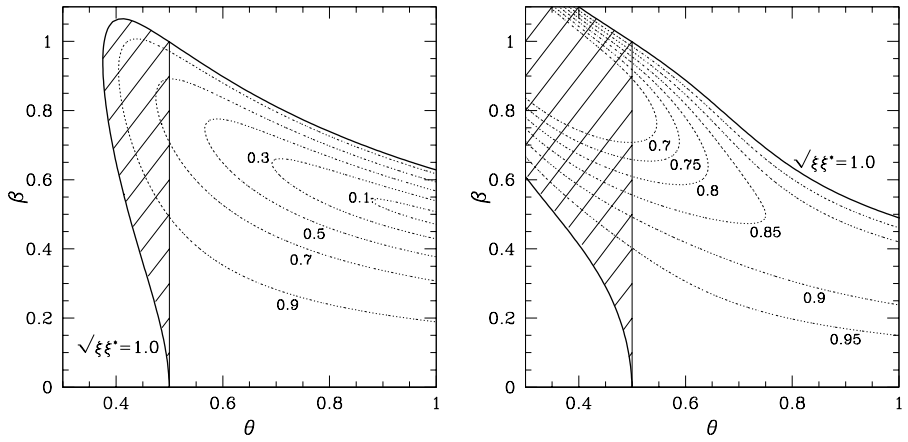


Figure 3.13: *Left panel:* stability region in the $(\theta, \beta)$ plane for the two-iterations $\theta$-ICN for the advection equation (3.2). Thick solid lines mark the limit at which $|\xi| = 1$, while the dotted contours indicate the values of the amplification factor in the stable region. The shaded area for $\theta < 1/2$ refers to solutions that are linearly unstable [16]. *Right panel:* same as in the left panel but when the averages between two corrections are swapped. Note that the amplification factor in this case is less sensitive on $\theta$ and always larger than the corresponding amplification factor in the left panel.

$\theta \geq 1/2$. This is a known property of the weighted Crank-Nicolson scheme [11] and inherited by the $\theta$-ICN. In essence, when $\theta \neq 1/2$ spurious solutions appear in the method [17] and these solutions are linearly unstable if $\theta < 1/2$, while they are stable for $\theta > 1/2$ [16]. For this reason we have shaded the area with $\theta < 1/2$ in the left panel of Fig. 3.13 to exclude it from the stability region. Secondly, the use of a weighting coefficient $\theta > 1/2$ will still lead to a stable scheme provided that the timestep (*i.e.,* $\beta$) is suitably decreased. Finally, as the contour lines in the left panel of Fig. 3.13 clearly show, the amplification factor can be very sensitive on $\theta$.

### Swapped weighted averages

The calculation of the stability of the $\theta$-ICN when the weighted averages are swapped as in eqs. (3.63) and (3.64) is somewhat more involved; after some lengthy but straightforward algebra we find the amplification factor to be

$$\xi \quad = \quad 1 - 2\mathrm{i}\beta - 4\beta^2\theta + 8\mathrm{i}\beta^3\theta(1-\theta)\,, \tag{3.69}$$

which differs from (3.66) only in that the $\theta^2$ coefficient of the $\mathcal{O}(\beta^3)$ term is replaced by $\theta(1-\theta)$. The stability requirement $|\xi| \leq 1$ is now expressed as

$$16\beta^4\theta^2(1-\theta)^2 - 4\beta^2\theta(2-3\theta) - 2\theta + 1 \leq 0\,. \tag{3.70}$$

Solving the condition (3.70) with respect to $\beta$ amounts then to requiring that

$$\beta \geq \frac{\sqrt{2 - 3\theta - \sqrt{4\theta - 11\theta^2 + 8\theta^3}}}{2(1-\theta)\sqrt{2\theta}}\,, \tag{3.71a}$$

$$\beta \leq \frac{\sqrt{2 - 3\theta + \sqrt{4\theta - 11\theta^2 + 8\theta^3}}}{2(1-\theta)\sqrt{2\theta}}\,, \tag{3.71b}$$

which is again equivalent to $\beta^2 \leq 1$ when $\theta = 1/2$. The corresponding region of stability is shown in right panel of Fig. 3.13 and should be compared with left panel of the same Figure. Note that the average-swapping has now considerably increased the amplification factor, which is always larger than the corresponding one for the $\theta$-ICN in the relevant region of stability (*i.e.,* for $1/2 \leq \theta \leq 1$ [2]).

[2] Of course, when the order of the swapped averages is inverted from the one shown in eqs. (3.63)–(3.64) the stability region will change into $0 \leq \theta \leq 1/2$.

## 3.7 Summary

In what follows I summarise the most salient aspects of the different finite-difference operators discussed so far and report, for each of them, the truncation error $\epsilon_{\text{T}}$, the amplification factor $|\xi|^2$ and the finite-difference representation of the advection equation 3.2. I recall that $\alpha := v\Delta t/\Delta x$

| *Method* | $\epsilon_{\text{T}}$ | $\lvert\xi\rvert^2$ for $(k\Delta x \ll 1)$ | finite-difference form |
|---|---|---|---|
| Upwind | $\mathcal{O}(\Delta t, \Delta x)$ | $1 - 2\lvert\alpha\rvert(1 - \lvert\alpha\rvert)\cos(k\Delta x)$ | $u_j^{n+1} = u_j^n \mp \alpha(u_{j\pm1}^n - u_j^n)$ |
| FTCS | $\mathcal{O}(\Delta t, \Delta x^2)$ | $1 + \sin^2(k\Delta x)\alpha^2$ | $u_j^{n+1} = u_j^n - \alpha(u_{j+1}^n - u_{j-1}^n)$ |
| Lax Friedrichs | $\mathcal{O}(\Delta t, \Delta x^2)$ | $1 - \sin^2(k\Delta x)(1 - \alpha^2)$ | $u_j^{n+1} = \frac{1}{2}(u_{j+1}^n + u_{j-1}^n)$ $-\frac{1}{2}\alpha(u_{j+1}^n - u_{j-1}^n)$ |
| Leapfrog | $\mathcal{O}(\Delta t^2, \Delta x^2)$ | 1 | $u_j^{n+1} = u_j^{n-1} - \alpha(u_{j+1}^n - u_{j-1}^n)$ |
| Lax Wendroff | $\mathcal{O}(\Delta t^2, \Delta x^2)$ | $1 - \alpha^2(1 - \alpha^2)\sin^2(k\Delta x)$ | $u_j^{n+1} = u_j^n - \frac{1}{2}\alpha(u_{j+1}^n - u_{j-1}^n)$ $+\frac{1}{2}\alpha^2(u_{j+1}^n - 2u_j^n + u_{j-1}^n)$ |

Table 3.1: Schematic summary of the finite-difference operators discussed so far.

### 3.7.1 Finite-difference stencils

In what follow I summarise the most used finite-difference stencils for derivatives of order 1 to 4

**Finite-difference stencils for $\partial \boldsymbol{u}/\partial \boldsymbol{x}$**

| *type* | Difference Stencil | LTE |
|---|---|---|
| forward | $(-u_j + u_{j+1})/h$ | $\mathcal{O}(h)$ |
| backward | $(-u_{j-1} + u_j)/h$ | $\mathcal{O}(h)$ |
| forward | $(-3u_j + 4u_{j+1} - u_{j+2})/h$ | $\mathcal{O}(h^2)$ |
| backward | $(u_{j-2} - 4u_{j-1} + 3u_j)/2h$ | $\mathcal{O}(h^2)$ |
| centered | $(-u_{j-1} + u_{j+1})/2h$ | $\mathcal{O}(h^2)$ |
| forward | $(-25u_j + 48u_{j+1} - 36u_{j+2} + 16u_{j+3} - 3u_{j+4})/12h$ | $\mathcal{O}(h^4)$ |
| backward | $(3u_{j-4} - 16u_{j-3} + 36u_{j-2} - 48u_{j-1} + 25u_j)/12h$ | $\mathcal{O}(h^4)$ |
| centered | $(u_{j-2} - 8u_{j-1} + 8u_{j+1} - u_{j+2})/12h$ | $\mathcal{O}(h^4)$ |

Table 3.2: Finite-difference stencils for $\partial u/\partial x$

**Finite-difference stencils for $\partial^2 u/\partial^2 x$**

| *type* | Difference Stencil | LTE |
|---|---|---|
| forward | $(u_j - 2u_{j+1} + u_{j+2})/h^2$ | $\mathcal{O}(h)$ |
| backward | $(u_{j-2} - 2u_{j-1} + u_j)/h^2$ | $\mathcal{O}(h)$ |
| forward | $(2u_j - 5u_{j+1} + 4u_{j+2} - u_{j+3})/h^2$ | $\mathcal{O}(h^2)$ |
| backward | $(-u_{j-3} + 4u_{j-2} - 5u_{j-1} + 2u_j)/h^2$ | $\mathcal{O}(h^2)$ |
| centered | $(u_{j-1} - 2u_j + u_{j+1})/h^2$ | $\mathcal{O}(h^2)$ |
| forward | $(45u_j - 154u_{j+1} + 214u_{j+2} - 156u_{j+3} + 61u_{j+4} - 10u_{j+5})/12h^2$ | $\mathcal{O}(h^4)$ |
| backward | $(-10u_{j-5} + 61u_{j-4} - 156u_{j-3} + 214u_{j-2} - 154u_{j-1} + 45u_j)/12h^2$ | $\mathcal{O}(h^4)$ |
| centered | $(-u_{j-2} + 16u_{j-1} - 30u_j + 16u_{j+1} - u_{j+2})/12h^2$ | $\mathcal{O}(h^4)$ |

Table 3.3: Finite-difference stencils for $\partial^2 u/\partial^2 x$

**Finite-difference stencils for $\partial^3 u/\partial^3 x$**

| *type* | Difference Stencil | LTE |
|---|---|---|
| forward | $(-u_j + 3u_{j+1} - 3u_{j+2} + u_{j+3})/h^3$ | $\mathcal{O}(h)$ |
| backward | $(-u_{j-3} + 3u_{j-2} - 3u_{j-1} + u_j)/h^3$ | $\mathcal{O}(h)$ |
| forward | $(-5u_j + 18u_{j+1} - 24u_{j+2} + 14u_{j+3} - 3u_{j+4})/2h^3$ | $\mathcal{O}(h^2)$ |
| backward | $(3u_{j-4} - 14u_{j-3} + 24u_{j-2} - 18u_{j-1} + 5u_j)/2h^3$ | $\mathcal{O}(h^2)$ |
| centered | $(-u_{j-2} + 2u_{j-1} - 2u_{j+1} + u_{j+2})/2h^3$ | $\mathcal{O}(h^2)$ |

Table 3.4: Finite-difference stencils for $\partial^3 u/\partial^3 x$

**Finite-difference stencils for $\partial^4 u/\partial^4 x$**

| *type* | Difference Stencil | LTE |
|---|---|---|
| forward | $(u_j - 4u_{j+1} + 6u_{j+2} - 4u_{j+3} + u_{j+4})/h^4$ | $\mathcal{O}(h)$ |
| backward | $(u_{j-4} - 4u_{j-3} + 6u_{j-2} - 4u_{j-1} + u_j)/h^4$ | $\mathcal{O}(h)$ |
| forward | $(3u_j - 14u_{j+1} + 26u_{j+2} - 24u_{j+3} + 11u_{j+4} - 2u_{j+5})/h^4$ | $\mathcal{O}(h^2)$ |
| backward | $(-2u_{j-5} + 11u_{j-4} - 24u_{j-3} + 26u_{j-2} - 14u_{j-1} + 3u_j)/h^4$ | $\mathcal{O}(h^2)$ |
| centered | $(u_{j-2} - 4u_{j-1} + 6u_j - 4u_{j+1} + u_{j+2})/h^4$ | $\mathcal{O}(h^2)$ |

Table 3.5: Finite-difference stencils for $\partial^4 u/\partial^4 x$

# Chapter 4

# Dissipation, Dispersion and Convergence

We will here discuss a number of problems that often emerge when using finite-difference techniques for the solution of hyperbolic partial differential equations. In stable numerical schemes, the impact of many of these problems can be suitably reduced by going to sufficiently high resolutions, but it is nevertheless important to have a simple and yet clear idea of what are the most common sources of these problems.

## 4.1 On the Origin of Dissipation and Dispersion

We have already seen in Chapter 3 how the Lax-Friedrichs scheme applied to a linear advection equation (3.2) yields the finite-difference expression

$$u_j^{n+1} = \frac{1}{2}(u_{j+1}^n + u_{j-1}^n) - \frac{\alpha}{2}(u_{j+1}^n - u_{j-1}^n) + \mathcal{O}(\Delta x^2)\,. \tag{4.1}$$

We have also mentioned how expression (4.1) can be rewritten as

$$u_j^{n+1} = u_j^n - \frac{\alpha}{2}(u_{j+1}^n - u_{j-1}^n) + \frac{1}{2}(u_{j+1}^n - 2u_j^n + u_{j-1}^n) + \mathcal{O}(\Delta x^2)\,, \tag{4.2}$$

to underline how the Lax-Friedrichs scheme effectively provides a first-order finite-difference representation of a non-conservative equation

$$\frac{\partial u}{\partial t} + v\frac{\partial u}{\partial x} = \varepsilon_{\text{LF}}\frac{\partial^2 u}{\partial x^2}\,, \tag{4.3}$$

that is an advection-diffusion equation in which a dissipative term

$$\varepsilon_{\text{LF}} := v\frac{\Delta x^2}{2\Delta t} = \alpha\frac{\Delta x}{2}\,, \tag{4.4}$$

is present. Given a computational domain of length $L$, this scheme will therefore have a typical diffusion timescale $\tau \simeq L^2/\varepsilon_{\text{LF}}$. Clearly, the larger the diffusion coefficient, the faster will the solution be completely smeared over the computational domain.

In a similar way, it is not difficult to realise that the upwind scheme

$$u_j^{n+1} = u_j^n - \alpha\left(u_j^n - u_{j-1}^n\right) + \mathcal{O}(\Delta x^2)\,, \tag{4.5}$$

provides a first-order accurate (in space) approximation to equation (3.2), but a second-order approximation to equation

$$\frac{\partial u}{\partial t} + v\frac{\partial u}{\partial x} = \varepsilon_{\text{UW}}\frac{\partial^2 u}{\partial x^2}\,, \tag{4.6}$$

where

$$\varepsilon_{\text{UW}} := \frac{v\Delta x}{2}\,. \tag{4.7}$$

Stated differently, also the upwind method reproduces at higher order an advection-diffusion equation with a dissipative term that is responsible for the gradual dissipation of the advected quantity $u$. This is shown in Fig. 4.2 for a wave packet (*i.e.*, a periodic function embedded in a Gaussian) propagating to the right and where it is important to notice how the different peaks in the packet are advected at the correct speed, although their amplitude is considerably diminished.

In Courant-limited implementations, $\alpha = |v|\Delta t/\Delta x < 1$, so that the ratio of the dissipation coefficients can be written as

$$\frac{\varepsilon_{\text{LF}}}{\varepsilon_{\text{UW}}} = \frac{1}{\alpha} \geq 1\,, \qquad \text{for} \quad \alpha \in [0,1]\,. \tag{4.8}$$

In other words, while the upwind and the Lax-Friedrichs methods are both dissipative, the latter is generically more dissipative despite being more accurate in space. This can be easily appreciated by comparing Figs. 3.3 and 3.7 but also provides an important rule: *a more accurate numerical scheme is not necessarily a preferable one.*

A bit of patience and a few lines of algebra would also show that the Lax-Wendroff scheme for the advection equation (3.2) [*cf.* Eq. (3.46)]

$$u_j^{n+1} = u_j^n - \alpha\left(u_{j+1}^n - u_{j-1}^n\right) + \frac{\alpha^2}{2}\left(u_{j+1}^n - 2u_j^n + u_{j-1}^n\right) + \mathcal{O}(\Delta x^2)\,. \tag{4.9}$$

provides a first-order accurate approximation to equation (3.2), a second-order approximation to an advection-diffusion equation with dissipation coefficient $\varepsilon_{\text{LW}}$, and a third-order approximation to equation

$$\frac{\partial u}{\partial t} + v\frac{\partial u}{\partial x} = \varepsilon_{\text{LW}}\frac{\partial^2 u}{\partial x^2} + \beta_{\text{LW}}\frac{\partial^3 u}{\partial x^3}\,, \tag{4.10}$$

where

$$\varepsilon_{\text{LW}} := \frac{\alpha v \Delta x}{2}\,, \qquad \beta_{\text{LW}} := -\frac{v\Delta x^2}{6}\left(1-\alpha^2\right)\,. \tag{4.11}$$

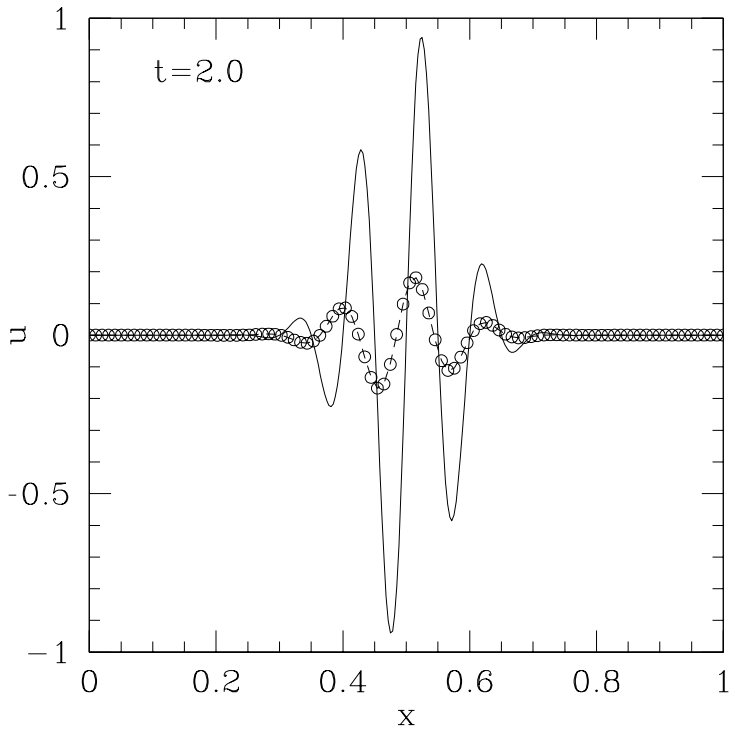


Figure 4.1: Time evolution of a wave-packet initially centred at $x = 0.5$ computed using a Lax-Friedrichs scheme with $C_{\text{CFL}} = 0.75$. The analytic solution at time $t = 2$ is shown with a solid line the dashed lines are used to represent the numerical solution at the same time. Note how dissipation reduces the amplitude of the wave-packet but does not change sensibly the propagation of the wave-packet.

As mentioned in Section 3, the Lax-Wendroff scheme retains some of the dissipative nature of the originating Lax-Friedrichs scheme and this is incorporated in the dissipative term proportional to $\varepsilon_{\text{LW}}$. Using expression (4.9), it is easy

to deduce the magnitude of this dissipation and compare it with the equivalent one produced with the Lax-Friedrichs scheme. A couple of lines of algebra show that

$$\varepsilon_{\text{LW}} = \alpha^2 \varepsilon_{\text{LF}} = \alpha^3 \frac{\Delta x}{2} \ll \varepsilon_{\text{LF}} \,, \tag{4.12}$$

thus emphasizing that the Lax-Wendroff scheme is considerably less dissipative than the corresponding Lax-Friedrichs.

The simplest way of quantifying the effects introduced by the right-hand-sides of equations (4.3), (4.6), and (4.10) is by using a single Fourier mode with angular frequency $\omega$ and wavenumber $k$, propagating in the positive $x$-direction, *i.e.*,

$$u(x,t) = e^{\mathrm{i}(kx-\omega t)} \,. \tag{4.13}$$

It is then easy to verify that in the continuum limit

$$\frac{\partial u}{\partial t} = -\mathrm{i}\omega u \,, \qquad \frac{\partial u}{\partial x} = \mathrm{i}k u \,, \qquad \frac{\partial^2 u}{\partial x^2} = -k^2 u \,, \qquad \frac{\partial^3 u}{\partial x^3} = -\mathrm{i}k^3 u \,. \tag{4.14}$$

In the case in which the finite-difference scheme provides an accurate approximation to a purely advection equation, the relations (4.14) lead to the obvious dispersion relation $\omega = vk$, so that the *numerical* mode $\tilde{u}(x,t)$ will have a solution

$$\tilde{u}(x,t) = e^{\mathrm{i}k(x-vt)} \,, \tag{4.15}$$

representing a mode propagating with *phase velocity* $c_{\text{p}} := \omega/k = v$, which coincides with the *group velocity* $c_{\text{g}} := \partial\omega/\partial k = v$.

However, it is simple to verify that the advection-diffusion equation approximated by the Lax-Friedrichs scheme (4.3), will have a corresponding solution

$$\tilde{u}(x,t) = e^{-\varepsilon_{\text{LF}} k^2 t} e^{\mathrm{i}k(x-vt)} \,, \tag{4.16}$$

thus having, besides the advective term, also an exponentially decaying mode over a timescale $\tau = \varepsilon_{\text{LF}} k^2$. Similarly, a few lines of algebra are sufficient to realise that the dissipative term does not couple with the advective one and, as a result, the phase and group velocities remain the same and $c_{\text{p}} = c_{\text{g}} = v$. This is clearly shown in Fig. 4.1 which shows how the wave packet is sensibly dissipated but, overall, maintains the correct group velocity.

Finally, it is possible to verify that the advection-diffusion equation approximated by the Lax-Wendroff scheme (4.10), will have a solution given by

$$\tilde{u}(x,t) = e^{-\varepsilon_{\text{LW}} k^2 t} e^{\mathrm{i}k\left[x-\left(v+\beta_{\text{LW}} k^2\right)t\right]} \,, \tag{4.17}$$

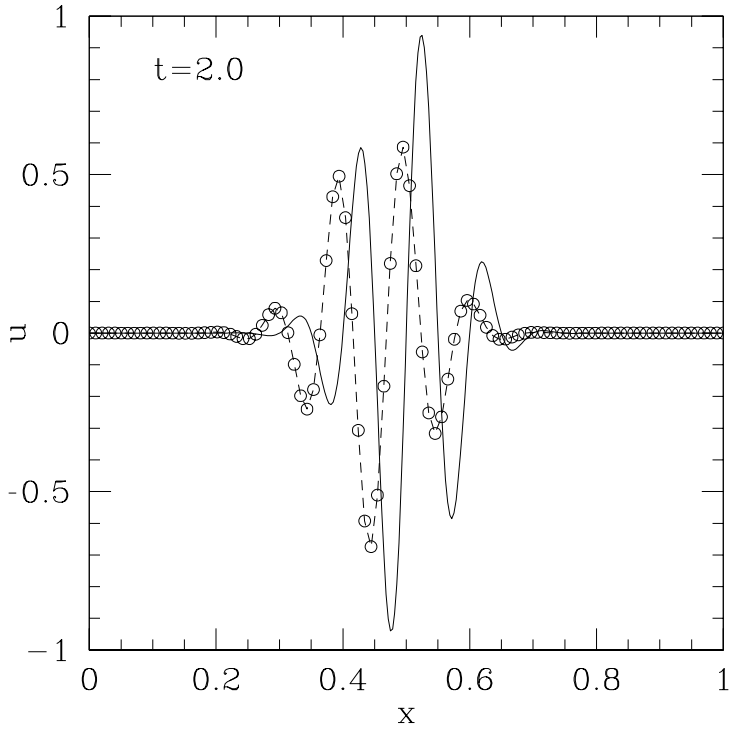


Figure 4.2: Time evolution of a wave-packet initially centred at $x = 0.5$ computed using a Lax-Wendroff scheme with $C_{\mathrm{CFL}} = 0.75$. The analytic solution at time $t = 2$ is shown with a solid line the dashed lines are used to represent the numerical solution at the same time. Note how the amplitude of the wave-packet is not drastically reduced but the group velocity suffers from a considerable error.

where, together with the advective and (smaller) exponentially decaying modes already encountered before, there appears also a *dispersive* term $\sim \beta_{\mathrm{LW}} k^2 t$ producing different propagation speeds for modes with different wavenumbers. This becomes apparent after calculating the phase and group velocities which are given by

$$c_{\mathrm{p}} = \frac{\omega}{k} = v + \beta_{\mathrm{LW}} k^2 \,, \qquad \text{and} \qquad c_{\mathrm{g}} = \frac{\partial \omega}{\partial k} = v + 3\beta_{\mathrm{LW}} k^2 \,, \tag{4.18}$$

and provides a simple interpretation of the results shown in Fig. 4.2.

## 4.2 Measuring Dissipation and Convergence

From what discussed so far it appears clear that one is often in the need of tools that allow a rapid comparison among different evolution schemes. One might be interested, for instance, in estimating which of two methods is less dissipative or whether an evolution scheme which is apparently stable will eventually turn out to be unstable. All of these features of a numerical method can be easily assessed through the use of the norms discussed in Section 1.1.1.

# Chapter 5

# The Wave Equation in 1D

The numerical solution of the wave equation offers a good example of how a higher-order (in space and time) PDE can be easily solved numerically through the solution of a system of coupled 1st-order PDEs.

In one spatial dimension (1D) the wave equation has the general form:

$$\frac{\partial^2 u}{\partial t^2} = v^2 \frac{\partial^2 u}{\partial x^2}\,, \tag{5.1}$$

where, for simplicity, we will assume that $v$ is constant (*i.e.*, $v \neq v(x)$), thus restricting our attention to linear problems. It is much more convenient to rewrite (5.1) as a system of coupled first-order conservative PDE. For this we set

$$r = v\frac{\partial u}{\partial x}\,, \tag{5.2}$$

$$s = \frac{\partial u}{\partial t}\,, \tag{5.3}$$

so that (5.1) can be rewritten as a system of three coupled, first-order differential equations

$$\begin{cases} \dfrac{\partial r}{\partial t} = v\dfrac{\partial s}{\partial x}\,, \\ \dfrac{\partial s}{\partial t} = v\dfrac{\partial r}{\partial x}\,, \\ \dfrac{\partial u}{\partial t} = s\,, \end{cases}$$

where it should be noted that the equations have the time derivative of *one* variable that is proportional to the space derivative of the *other* variable. This breaks the advective nature of the equation discussed in the previous Chapter and will prevent, for instance, the use of an upwind scheme.

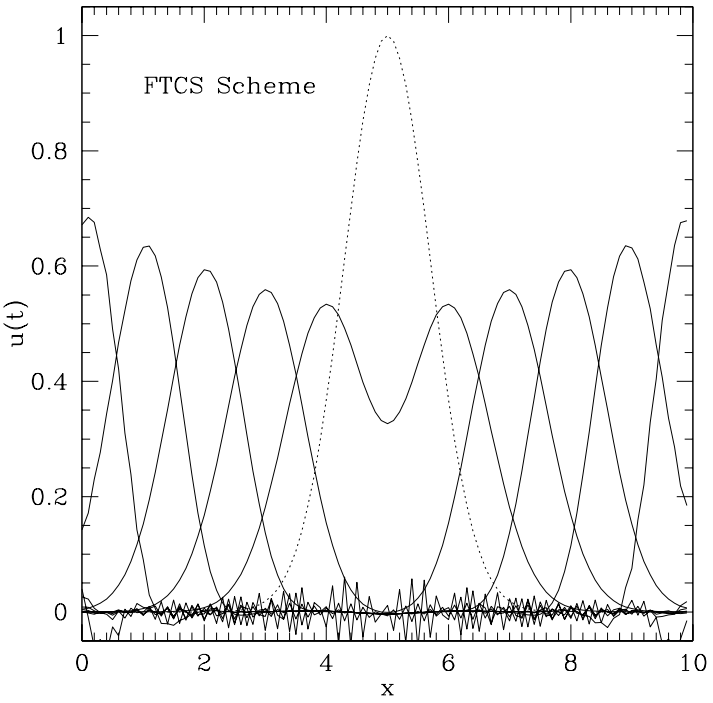


Figure 5.1: Plot of the time evolution of the wave equation when the FTCS scheme is used. The initial conditions were given by a Gaussian centered at $x = 5$ with unit variance and are shown with the dotted line. Note the growth of the wave crests and the appearance of short wavelength noise. When this happens, the numerical errors have grown to be comparable with the solution which will be rapidly destroyed.

In vector notation the system (5.4) can be symbolically written as

$$\frac{\partial \boldsymbol{U}}{\partial t} + \frac{\partial \boldsymbol{F}(\boldsymbol{U})}{\partial x} = 0\,, \tag{5.4}$$

where

$$\boldsymbol{U} = \begin{pmatrix} r \\ s \end{pmatrix}\,, \qquad \text{and} \qquad \boldsymbol{F}(\boldsymbol{U}) = \begin{pmatrix} 0 & -v \\ -v & 0 \end{pmatrix} \boldsymbol{U}\,. \tag{5.5}$$

## 5.1 The FTCS Scheme

As mentioned in the previous Chapter, the upwind method cannot be applied to the solution of the wave equation and the simplest, first-order in time method we can use for the solution of the wave equation is therefore given by the FTCS scheme. Applying it to the first-order system (5.4) and obtain

$$r_j^{n+1} = r_j^n + \frac{\alpha}{2}(s_{j+1}^n - s_{j-1}^n) + \mathcal{O}(\Delta x^2)\,, \tag{5.6}$$

$$s_j^{n+1} = s_j^n + \frac{\alpha}{2}(r_{j+1}^n - r_{j-1}^n) + \mathcal{O}(\Delta x^2)\,, \tag{5.7}$$

Once the value of $s_j^{n+1}$ has been calculated, the value of $u$ can be integrated in time according to equation (5.3) so that

$$u_j^{n+1} = u_j^n + \Delta t s_j^n + \mathcal{O}(\Delta x^2)\,, \tag{5.8}$$

where it should be noted that $u^{n+1}$ has the same truncation error of $r^{n+1}$ and $s^{n+1}$.

Of course, we do not expect that the FTCS scheme applied to the solution of the wave equation will provide a stable evolution and this is clearly shown in Fig. 5.1 which reports the solution of equations (5.6), (5.6) and (5.8) having as initial conditions a Gaussian centered at $x = 5$ with unit variance. Different lines show the solution at different times and is apparent how the initial profile splits in two part propagating in two opposite directions. During the evolution, however, the error grows (note that the peaks of the two packets increase with time) and in about one crossing time the short wavelength noise appears (this is shown by the small sharp peaks produced when the wave has left the numerical grid). When this happens, the numerical errors have grown to be comparable with the solution, which will be rapidly destroyed.

## 5.2 The Lax-Friedrichs Scheme

As done in the previous Section, we can apply the Lax-Friedrichs scheme to the solution of the wave equation through the first-order system (5.4) and easily obtain

$$r_j^{n+1} = \frac{1}{2}(r_{j+1}^n + r_{j-1}^n) + \frac{\alpha}{2}(s_{j+1}^n - s_{j-1}^n) + \mathcal{O}(\Delta x^2)\,, \tag{5.9}$$

$$s_j^{n+1} = \frac{1}{2}(s_{j+1}^n + s_{j-1}^n) + \frac{\alpha}{2}(r_{j+1}^n - r_{j-1}^n) + \mathcal{O}(\Delta x^2)\,, \tag{5.10}$$

Also in this case, once the value of $s_j^{n+1}$ has been calculated, the value for $u_j^{n+1}$ can be computed according to (5.8).

The solution of equations (5.9), (5.9) and (5.8) with the same initial data used in Fig. 5.1 is shown in Fig. 5.2. Note that we encounter here the same behaviour found in the solution of the advection equation and in particular it is apparent the progressive diffusion of the two travelling packets which spread over the numerical grid as they propagate. As expected, the evolution is not stable and no error growth is visible many crossing times after the wave has left the numerical grid.

## 5.3 The Leapfrog Scheme

We can adapt the Leapfrog scheme to equations (5.4) for the solution of the wave equation in one dimension, centring variables on appropriate half-mesh points

$$r_{j+\frac{1}{2}}^n := v\left.\frac{\partial u}{\partial x}\right|_{j+\frac{1}{2}}^n = v\frac{u_{j+1}^n - u_j^n}{\Delta x} + \mathcal{O}(\Delta x)\,, \tag{5.11}$$

$$s_j^{n+\frac{1}{2}} := \left.\frac{\partial u}{\partial t}\right|_j^{n+\frac{1}{2}} = \frac{u_j^{n+1} - u_j^n}{\Delta t} + \mathcal{O}(\Delta t)\,, \tag{5.12}$$

and then considering the Leapfrog representation of equations (5.4)

$$r_{j+\frac{1}{2}}^{n+1} = r_{j+\frac{1}{2}}^n + \alpha\left(s_{j+1}^{n+\frac{1}{2}} - s_j^{n+\frac{1}{2}}\right) + \mathcal{O}(\Delta x^2)\,, \tag{5.13}$$

$$s_j^{n+\frac{1}{2}} = s_j^{n-\frac{1}{2}} + \alpha\left(r_{j+\frac{1}{2}}^n - r_{j-\frac{1}{2}}^n\right) + \mathcal{O}(\Delta x^2)\,, \tag{5.14}$$

As in the previous examples, the new value for the wave variable $u$ is finally computed after the integration in time of $s$. Here however, to preserve the second-order accuracy in time it is necessary to average the time derivative $s$ between $n$ and $n+1$ to obtain

$$u_j^{n+1} = u_j^n + \frac{\Delta t}{2}(s_j^{n+1} + s_j^n) + \mathcal{O}(\Delta x^2) = u_j^n + \frac{\Delta t}{2}s_j^{n+1/2} + \mathcal{O}(\Delta x^2)\,. \tag{5.15}$$

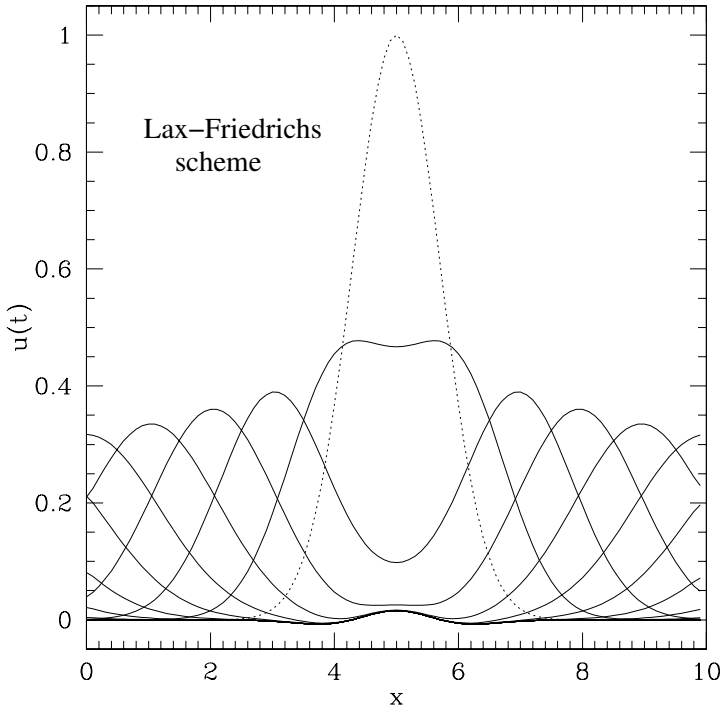


Figure 5.2: The same as in Fig. 5.1 but when the Lax-Friedrichs scheme is used. Note the absence of the late time instabilities but also the effects of the numerical diffusion that widens and lowers the wave fronts.

A simple substitution of (5.11) and (5.12) into (5.13) and (5.14) shows how the Leapfrog representation of the wave equation is nothing but its second-order differencing:

$$\frac{u_j^{n+1} - 2u_j^n + u_j^{n-1}}{\Delta t^2} = v^2 \left( \frac{u_{j+1}^n - 2u_j^n + u_{j-1}^n}{\Delta x^2} \right) + \mathcal{O}(\Delta t^2, \Delta x^2)\,, \qquad (5.16)$$

so that the solution at the new time-level is

$$u_j^{n+1} = \alpha^2 u_{j+1}^n + 2u_j^n \left(1 - \alpha^2\right) + \alpha^2 u_{j-1}^n - u_j^{n-1} + \mathcal{O}(\Delta x^4)\,. \qquad (5.17)$$

Note that as formulated in (5.17), the Leapfrog scheme has been effectively recast into a "one-level" scheme.

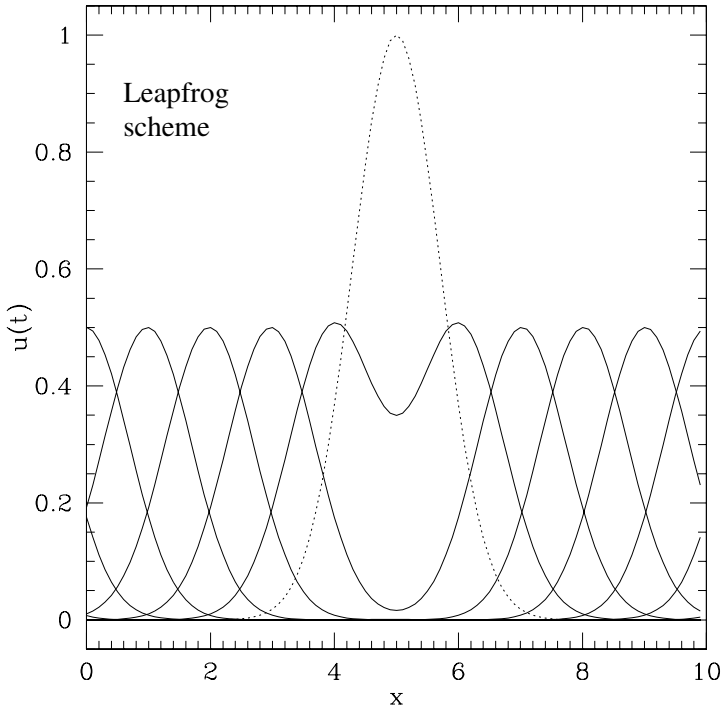


Figure 5.3: The same as in Fig. 5.1 but when the Leapfrog scheme is used. Note the absence of the late time instabilities and of the effects of the numerical diffusion.

The solution of equations (5.17) and (5.15) with the same initial data used in Fig. 5.1 is shown in Fig. 5.3. Note that we do not encounter here a significant amount of diffusion for the two travelling wave packets. As expected, the evolution is stable and no error growth is visible many crossing times after the wave has left the numerical grid.

## 5.4 The Lax-Wendroff Scheme

Also in the case, the application of this scheme to our system of equations (5.4) is straightforward. We can start with the time evolution of the variable $r$ to

obtain

$$r_j^{n+1} = r_j^n + \alpha \left( s_{j+1/2}^{n+1/2} - s_{j-1/2}^{n+1/2} \right) + \mathcal{O}(\Delta x^2) \,, \tag{5.18}$$

where the terms in the spatial derivatives are computed as

$$s_{j+1/2}^{n+1/2} = \frac{1}{2} \left( s_j^n + s_{j+1}^n \right) + \alpha \left( r_{j+1}^n - r_j^n \right) + \mathcal{O}(\Delta x^2) \,, \tag{5.19}$$

$$s_{j-1/2}^{n+1/2} = \frac{1}{2} \left( s_j^n + s_{j-1}^n \right) + \alpha \left( r_j^n - r_{j-1}^n \right) + \mathcal{O}(\Delta x^2) \,. \tag{5.20}$$

As done for the advection equation, it is convenient not to use equations (5.18) and (5.19) as two coupled but distinct equations and rather to combine them into two "one-level" evolution equations for $r$ and $s$

$$r_j^{n+1} = r_j^n + \alpha \left[ \frac{1}{2} (s_{j+1}^n - s_{j-1}^n) + \frac{\alpha}{2} (r_{j+1}^n - 2 r_j^n + r_{j-1}^n) \right] + \mathcal{O}(\Delta x^2) \,, \tag{5.21}$$

$$s_j^{n+1} = s_j^n + \alpha \left[ \frac{1}{2} (r_{j+1}^n - r_{j-1}^n) + \frac{\alpha}{2} (s_{j+1}^n - 2 s_j^n + s_{j-1}^n) \right] + \mathcal{O}(\Delta x^2) \,. \tag{5.22}$$

The solution of equations (5.21), (5.22) and (5.15) with the same initial data used in Fig. 5.1 is shown in Fig. 5.4. Note that we do not encounter here a significant amount of diffusion for the two travelling wave packets. As expected, the evolution is stable and no error growth is visible many crossing times after the wave has left the numerical grid.

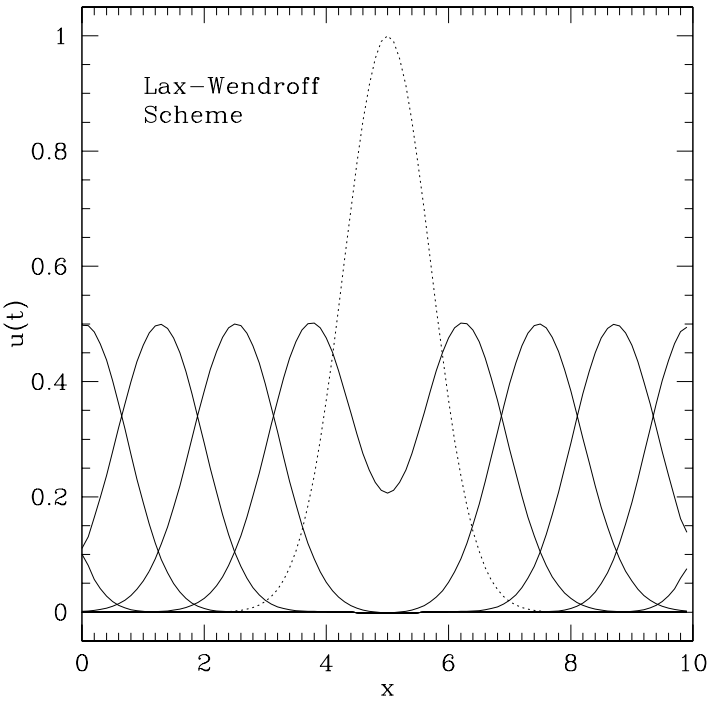


Figure 5.4: The same as in Fig. 5.1 but when the Lax-Wendroff scheme is used. Note the absence of the late time instabilities and of the effects of the numerical diffusion.

# Chapter 6

# Boundary Conditions

Unavoidable and common to all the numerical schemes discussed so far is the problem of treating the solution on the boundaries of the spatial grid as the time evolution proceeds. Let 1 be the first gridpoint and $J$ the last one. It is clear from equations (3.26), (5.16), (5.21) and (5.22) that the new solution at the boundaries of the spatial grid (*i.e.*, $u_1^{n+1}$,$u_J^{n+1}$) is undetermined as it requires the values $u_0^n$, $u_{J+1}^n$. The most natural boundary conditions for the evolution of a wave equation are the so called *Sommerfeld boundary conditions* (or *radiative boundary conditions*) which will be discussed in the following Section. Other boundary conditions of general interest are:

- ***Dirichlet-type*** boundary conditions: values of the relevant quantity are imposed at the boundaries of the numerical grid. These values can be either functions of time or be held constant (*cf.* boundary conditions for boundary value problems);

  –*"Periodic"* boundary conditions: assume that the numerical domain is topologically connected in a given direction; this is often used in cosmological simulations (and "videogames").

- ***von Neumann-type*** boundary conditions: values of the derivatives of the relevant quantity are imposed at the boundaries of the numerical grid. As for Dirichlet, these values can be either functions of time or be held constant (*cf.* boundary conditions for boundary value problems);

  –*"Reflecting"* boundary conditions: mimic the presence of a reflecting boundary, *i.e.*, of a boundary with zero transmission coefficient;

–*"Absorbing"* boundary conditions: mimic the presence of an absorbing boundary, *i.e.*, of a boundary with unit transmission coefficient;

## 6.1 Outgoing Wave BCs: the outer edge

A scalar wave outgoing in the positive $x$-direction is described by the advection equation:

$$\frac{\partial u}{\partial t} + v\frac{\partial u}{\partial x} = 0 \tag{6.1}$$

A finite-difference, first-order accurate representation of equation (6.1) which is centered in both time (at $n+\frac{1}{2}$) and in space (at $j+\frac{1}{2}$) is given by (see Fig. 3.11)

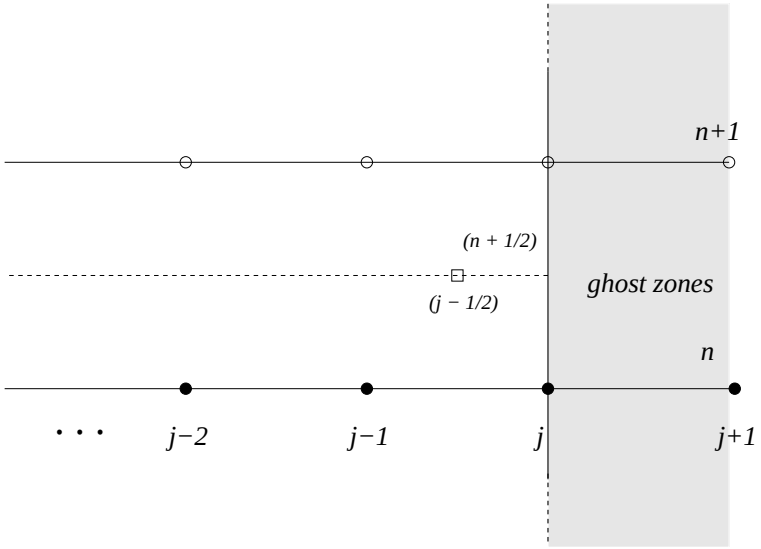


Figure 6.1: Schematic representation of the centring for a first-order, outgoing-wave Sommerfeld boundary conditions. An equivalent one can be drawn for an ingoing-wave.

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

and which leads to

$$u_{j+1}^{n+1}(1+\alpha) = u_j^{n+1}(-1+\alpha) + u_{j+1}^n(1-\alpha) + u_j^n(1+\alpha) \tag{6.2}$$

Expression (6.2) can also be written as

$$u_{j+1}^{n+1} = u_j^n - u_j^{n+1}Q + u_{j+1}^n Q\,, \tag{6.3}$$

where

$$Q := \frac{1-\alpha}{1+\alpha} = \frac{\Delta x - v\Delta t}{\Delta x + v\Delta t}\,. \tag{6.4}$$

The use of expression (6.3) for the outermost grid point where the wave is outgoing will provide first-order accurate and stable boundary conditions. Note, however, that (6.3) is a discrete representation of a physical condition which would transmit the wave without reflection. In practice, however, a certain amount of reflection is always produced (the transmission coefficient is never exactly one); the residual wave is then transmitted back in the numerical box. A few reflections are usually sufficient to reduce the wave content to values below the machine accuracy.

## 6.2 Ingoing Wave BCs: the inner edge

Similarly, a scalar wave outgoing in the negative $x$-direction (or ingoing in the positive one) is described by the advection equation:

$$\frac{\partial u}{\partial t} - v\frac{\partial u}{\partial x} = 0 \tag{6.5}$$

Following the same procedure discussed before, the algorithm becomes:

$$u_j^{n+1}\,(1+\alpha) = -u_{j+1}^{n+1}\,(1-\alpha) + u_{j+1}^{n}\,(1+\alpha) + u_j^n\,(1-\alpha)$$

Then

$$u_j^{n+1} = u_{j+1}^{n} - u_{j+1}^{n+1}Q + u_j^n Q\,, \tag{6.6}$$

where $Q$ is the same quantity as for the out-going wave. If we use equations (6.3) and (6.6) to evolve the solution at time-step $n+1$ at the boundary of our spatial grid, we are guaranteed that our profile will be completely transported away, whatever integration scheme we are adopting (Leapfrog, Lax-Wendroff etc.).

## 6.3 Periodic Boundary Conditions

These are very simple to impose and if $j$ is between 1 and $J$, they are given simply by

$$u_1^{n+1} = u_{J-1}^{n+1}, \qquad u_J^{n+1} = u_2^{n+1}\,, \tag{6.7}$$

In the case of a Gaussian leaving the center of the numerical grid, these boundary conditions effectively produce a reflection. The boundary conditions

(6.7) force to break the algorithm for the update scheme excluding the first and last points that need to be computed separately.

An alternative procedure consists of introducing a number of *"ghost"* gridpoints outside the computational domain of interest so that the solution is calculated using always the *same stencil* for $j = 1, 2, \ldots, J$ and exploiting the knowledge of the solution also at the ghost gridpoints, *e.g.*, 0 and $J + 1$.

In the case there is only one ghost gridpoint at either edge of the 1D grid, the boundary conditions are simply given by

$$u_0^{n+1} = u_J^{n+1}, \qquad u_{J+1}^{n+1} = u_1^{n+1}\,. \tag{6.8}$$

# Chapter 7

# The wave equation in two spatial dimensions (2D)

We will now extend the procedures studied so far to the case of a wave equation in two dimensions

$$\frac{\partial^2 u}{\partial t^2} = v^2 \left( \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} \right) . \tag{7.1}$$

As for the one-dimensional case, also in this case the wave equation can be reduced to the solution of a set of three first-order advection equations

$$\frac{\partial r}{\partial t} = v \frac{\partial s}{\partial x} , \tag{7.2}$$

$$\frac{\partial l}{\partial t} = v \frac{\partial s}{\partial y} , \tag{7.3}$$

$$\frac{\partial s}{\partial t} = v \left( \frac{\partial r}{\partial x} + \frac{\partial l}{\partial y} \right) , \tag{7.4}$$

once the following definitions have been made

$$r = v \frac{\partial u}{\partial x} , \tag{7.5}$$

$$l = v \frac{\partial u}{\partial y} , \tag{7.6}$$

$$s = \frac{\partial u}{\partial t} . \tag{7.7}$$

In vector notation the system can again be written as

$$\frac{\partial \boldsymbol{U}}{\partial t} + \nabla \boldsymbol{F}(\boldsymbol{U}) = 0\,, \tag{7.8}$$

where

$$\boldsymbol{U} = \begin{pmatrix} r \\ l \\ s \end{pmatrix}, \qquad \text{and} \qquad \boldsymbol{F}(\boldsymbol{U}) = \begin{pmatrix} -v & 0 & 0 \\ 0 & -v & 0 \\ 0 & 0 & -v \end{pmatrix} \cdot \boldsymbol{U} = -v \begin{pmatrix} r \\ l \\ s \end{pmatrix}, \tag{7.9}$$

provided we define

$$\nabla := \begin{pmatrix} 0 & 0 & \dfrac{\partial}{\partial x} \\ 0 & 0 & \dfrac{\partial}{\partial y} \\ \dfrac{\partial}{\partial x} & \dfrac{\partial}{\partial y} & 0 \end{pmatrix}. \tag{7.10}$$

The finite-difference notation should also be extended to account for the two spatial dimension and we will then assume that $u^n_{i,j} := u(x_i, y_j, t^n)$.

## 7.1 The Lax-Friedrichs Scheme

We can look at the system of equations (7.2) and (7.3) as a set of two equations to be integrated with the procedures so far developed in one-dimension. Furthermore, we need to solve for Eq. (7.4) which can be written as

$$\frac{\partial s}{\partial t} = -\frac{\partial F_{\rm x}}{\partial x} - \frac{\partial F_{\rm y}}{\partial y} \tag{7.11}$$

once we identify $F_{\rm x}$ with $-vr$ and $F_{\rm y}$ with $-vl$.

The Lax-Friedrichs scheme for this equation is just the generalisation of the 1D expressions discussed so far and yields

$$\begin{aligned} s^{n+1}_{i,j} &= \frac{1}{4}\left[s^n_{i+1,j} + s^n_{i-1,j} + s^n_{i,j+1} + s^n_{i,j-1}\right] - \frac{\Delta t}{2\Delta x}\left[(F^n_{\rm x})_{i+1,j} - (F^n_{\rm x})_{i-1,j}\right] \\ &\qquad - \frac{\Delta t}{2\Delta y}\left[(F^n_{\rm y})_{i,j+1} - (F^n_{\rm y})_{i,j-1}\right], \\ &= \frac{1}{4}\left[s^n_{i+1,j} + s^n_{i-1,j} + s^n_{i,j+1} + s^n_{i,j-1}\right] - \frac{v_{\rm x}\,\Delta t}{2}\left[\frac{r^n_{i+1,j} - r^n_{i-1,j}}{\Delta x}\right] \\ &\qquad - \frac{v_{\rm y}\,\Delta t}{2}\left[\frac{l^n_{i,j+1} - l^n_{i,j-1}}{\Delta y}\right], \end{aligned} \tag{7.12}$$

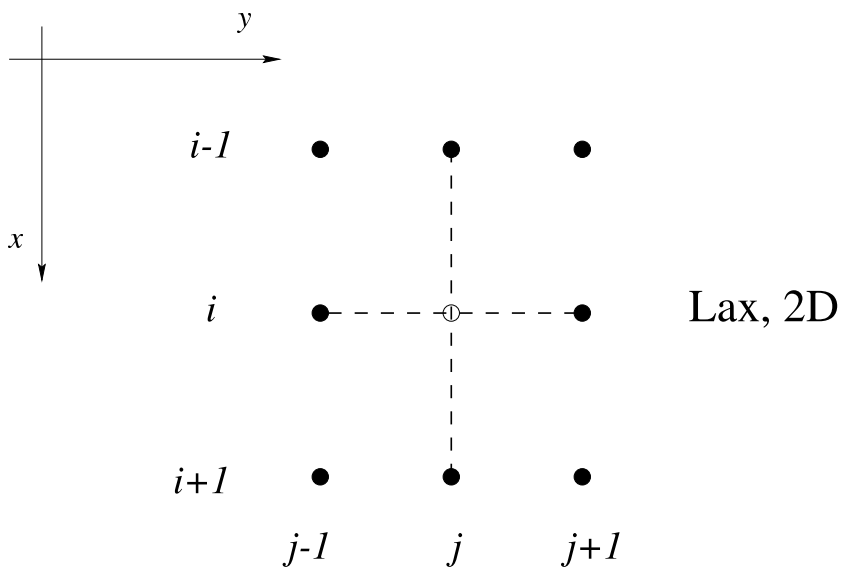


Figure 7.1: Schematic diagram of a Lax-Friedrichs evolution scheme in two dimensions. Note that the center of the cross-like stencil is not used in this case.

with the corresponding stencil being shown in Fig. 7.1 and where it should be noted that the center of the cross-like stencil is not used. A von-Neumann stability analysis can be performed also in 2D and it yields

$$\xi = \frac{1}{2}[\cos(k_{\rm x}\Delta x) + \cos(k_{\rm y}\Delta y)] - i[\alpha_{\rm x}\sin(k_{\rm x}\Delta x) + \alpha_{\rm y}\sin(k_{\rm y}\Delta y)]\,, \qquad (7.13)$$

where

$$\alpha_{\rm x} := \frac{v_{\rm x}\Delta t}{\Delta x}\,, \qquad\qquad \alpha_{\rm y} := \frac{v_{\rm y}\Delta t}{\Delta y}\,. \qquad (7.14)$$

Stability is therefore obtained if

$$\frac{1}{2} - (\alpha_{\rm y}^2 + \alpha_{\rm y}^2) \geq 0\,, \qquad (7.15)$$

or, equally, if

$$\Delta t \leq \frac{\Delta x}{\sqrt{2(v_{\rm x}^2 + v_{\rm y}^2)}}\,, \qquad (7.16)$$

Expression (7.16) represents the 2D extension of the CFL stability condition. In general, for a $N$-dimensional space, the CFL stability condition can be expressed as

$$\Delta t \leq \min\left(\frac{\Delta x_i}{N\tilde{v}}\right)\,, \qquad (7.17)$$

where $i = 1, ...N$ and $\tilde{v} := (\sum_{i=1}^{N} v_i^2)^{1/2}$. Note, in 2D, the appearance of an averaging coefficient $1/4$ multiplying the value of the function at the time-level $n$.

## 7.2 The Lax-Wendroff Scheme

The 2D generalisation of the one-dimensional scheme (3.43) is also straightforward and can be described as follows

1. Compute $r$, $l$ and $s$ at the half-time using a half-step Lax-Friedrichs scheme

$$r_{i+\frac{1}{2},j}^{n+\frac{1}{2}} = \frac{1}{2}\left[\left(r_{i+1,j}^{n} + r_{i,j}^{n}\right) + \alpha_x\left(s_{i+1,j}^{n} - s_{i,j}^{n}\right)\right], \tag{7.18}$$

$$r_{i-\frac{1}{2},j}^{n+\frac{1}{2}} = \frac{1}{2}\left[\left(r_{i,j}^{n} + r_{i-1,j}^{n}\right) + \alpha_x\left(s_{i,j}^{n} - s_{i-1,j}^{n}\right)\right], \tag{7.19}$$

$$l_{i,j+\frac{1}{2}}^{n+\frac{1}{2}} = \frac{1}{2}\left[\left(l_{i,j+1}^{n} + l_{i,j}^{n}\right) + \alpha_y\left(s_{i,j+1}^{n} - s_{i,j}^{n}\right)\right], \tag{7.20}$$

$$l_{i,j-\frac{1}{2}}^{n+\frac{1}{2}} = \frac{1}{2}\left[\left(l_{i,j}^{n} + l_{i,j-1}^{n}\right) + \alpha_y\left(s_{i,j}^{n} - s_{i,j-1}^{n}\right)\right], \tag{7.21}$$

$$s_{i+\frac{1}{2},j}^{n+\frac{1}{2}} = \frac{1}{2}\left[\left(s_{i+1,j}^{n} + s_{i,j}^{n}\right) + \alpha_x\left(r_{i+1,j}^{n} - r_{i,j}^{n}\right) + \frac{\alpha_y}{2}\left(l_{i,j+1}^{n} - l_{i,j-1}^{n}\right)\right], \tag{7.22}$$

$$s_{i-\frac{1}{2},j}^{n+\frac{1}{2}} = \frac{1}{2}\left[\left(s_{i,j}^{n} + s_{i-1,j}^{n}\right) + \alpha_x\left(r_{i,j}^{n} - r_{i-1,j}^{n}\right) + \frac{\alpha_y}{2}\left(l_{i,j+1}^{n} - l_{i,j-1}^{n}\right)\right], \tag{7.23}$$

$$s_{i,j+\frac{1}{2}}^{n+\frac{1}{2}} = \frac{1}{2}\left[\left(s_{i,j+1}^{n} + s_{i,j}^{n}\right) + \frac{\alpha_x}{2}\left(r_{i+1,j}^{n} - r_{i-1,j}^{n}\right) + \alpha_y\left(l_{i,j+1}^{n} - l_{i,j}^{n}\right)\right], \tag{7.24}$$

$$s_{i,j-\frac{1}{2}}^{n+\frac{1}{2}} = \frac{1}{2}\left[\left(s_{i,j}^{n} + s_{i,j-1}^{n}\right) + \frac{\alpha_x}{2}\left(r_{i+1,j}^{n} - r_{i-1,j}^{n}\right) + \alpha_y\left(l_{i,j}^{n} - l_{i,j-1}^{n}\right)\right], \tag{7.25}$$

where $\alpha_x := v\Delta t/\Delta x$ and $\alpha_y := v\Delta t/\Delta y$.

2. Evolve $r$, $l$ and $s$ to the time-level $n+1$ using a half-step Leapfrog scheme

$$r_{i,j}^{n+1} = r_{i,j}^{n} + \alpha_x \left( s_{i+\frac{1}{2},j}^{n+\frac{1}{2}} - s_{i-\frac{1}{2},j}^{n+\frac{1}{2}} \right) , \tag{7.26}$$

$$l_{i,j}^{n+1} = l_{i,j}^{n} + \alpha_y \left( s_{i,j+\frac{1}{2}}^{n+\frac{1}{2}} - s_{i,j-\frac{1}{2}}^{n+\frac{1}{2}} \right) , \tag{7.27}$$

$$s_{i,j}^{n+1} = s_{i,j}^{n} + \alpha_x \left( r_{i+\frac{1}{2},j}^{n+\frac{1}{2}} - r_{i-\frac{1}{2},j}^{n+\frac{1}{2}} \right) + \alpha_y \left( l_{i,j+\frac{1}{2}}^{n+\frac{1}{2}} - l_{i,j-\frac{1}{2}}^{n+\frac{1}{2}} \right) . \tag{7.28}$$

3. Update $u$ to the time-level $n+1$, *i.e.*,

$$u_{i,j}^{n+1} = u_{i,j}^{n} + \frac{\Delta t}{2} \left( s_{i,j}^{n+1} + s_{i,j}^{n} \right) . \tag{7.29}$$

Alternatively, steps 1. and 2. can be combined analytically to yield the direct integration of $r$, $l$ and $s$ from level $n$ to level $n+1$ as

$$r_{i,j}^{n+1} = r_{i,j}^{n} + \alpha_x \left[ \frac{1}{2} \left( s_{i+1,j}^{n} - s_{i-1,j}^{n} \right) + \frac{\alpha_x}{2} \left( r_{i+1,j}^{n} - 2r_{i,j}^{n} + r_{i-1,j}^{n} \right) \right] , \tag{7.30}$$

$$l_{i,j}^{n+1} = l_{i,j}^{n} + \alpha_y \left[ \frac{1}{2} \left( s_{i,j+1}^{n} - s_{i,j-1}^{n} \right) + \frac{\alpha_y}{2} \left( s_{i,j+1}^{n} - 2s_{i,j}^{n} + s_{i,j-1}^{n} \right) \right] , \tag{7.31}$$

$$\begin{aligned} s_{i,j}^{n+1} = s_{i,j}^{n} &+ \alpha_x \left[ \frac{1}{2} \left( r_{i+1,j}^{n} - r_{i-1,j}^{n} \right) + \frac{\alpha_x}{2} \left( s_{i+1,j}^{n} - 2s_{i,j}^{n} + s_{i-1,j}^{n} \right) \right] + \\ &\alpha_y \left[ \frac{1}{2} \left( l_{i,j+1}^{n} - l_{i,j-1}^{n} \right) + \frac{\alpha_y}{2} \left( s_{i,j+1}^{n} - 2s_{i,j}^{n} + s_{i,j-1}^{n} \right) \right] , \end{aligned} \tag{7.32}$$

Such a form of the equations is simpler to implement and allows a transparent distinction between the advective and the dissipative terms produced by the presence of second spatial-derivatives terms. More specifically, the system above can be written in a compact form as

$$r_{i,j}^{n+1} = r_{i,j}^{n} + \frac{\alpha_x}{2} D_x s_{i,j}^{n} + \frac{\alpha_x^2}{2} D_{xx} r_{i,j}^{n} , \tag{7.33}$$

$$l_{i,j}^{n+1} = l_{i,j}^{n} + \frac{\alpha_y}{2} D_y s_{i,j}^{n} + \frac{\alpha_y^2}{2} D_{yy} s_{i,j}^{n} , \tag{7.34}$$

$$s_{i,j}^{n+1} = s_{i,j}^{n} + \frac{\alpha_x}{2} D_x r_{i,j}^{n} + \frac{\alpha_y}{2} D_y l_{i,j}^{n} + \frac{\alpha_x^2}{2} D_{xx} s_{i,j}^{n} + \frac{\alpha_y^2}{2} D_{yy} s_{i,j}^{n} , \tag{7.35}$$

where we have omitted for compactness the spatial indices and we have used the compact notation

$$D_x\phi_{i,j} := \phi_{i+1,j} - \phi_{i-1,j}\,, \qquad D_y\phi_{i,j} := \phi_{i,j+1} - \phi_{i,j-1}\,, \tag{7.36}$$

$$D_{xx}\phi_{i,j} := \phi_{i+1,j} - 2\phi_{i,j} + \phi_{i-1,j}\,, \quad D_{yy}\phi_{i,j} := \phi_{i,j+1} - 2\phi_{i,j} + \phi_{i,j-1}\,. \tag{7.37}$$

Expressions (7.33)-(7.37) can be easily extended to higher spatial dimensions and readily implemented in a recursive loop.

## 7.3 The Leapfrog Scheme

The 2D generalisation of the one-dimensional scheme (5.16) is less straightforward, but not particularly difficult. As in one dimension, we can start by rewriting directly the finite-difference form of the wave equation as

$$\frac{u^{n+1}_{i,j} - 2u^n_{i,j} + u^{n-1}_{i,j}}{\Delta t^2} = v^2\left(\frac{u^n_{i+1,j} - 2u^n_{i,j} + u^n_{i-1,j}}{\Delta x^2}\right) + v^2\left(\frac{u^n_{i,j+1} - 2u^n_{i,j} + u^n_{i,j-1}}{\Delta y^2}\right)$$

so that, after some algebra, we obtain the explicit form

$$u^{n+1}_{i,j} = \alpha^2\left[u^n_{i+1,j} + u^n_{i-1,j} + u^n_{i,j+1} + u^n_{i,j-1}\right] + 2u^n_{i,j}(1-2\alpha^2) - u^{n-1}_{i,j}\,. \tag{7.38}$$

The stencil relative to the algorithm (7.38) is illustrated in Fig. 7.2.

Figs. 7.3 and 7.4 show the solution of the wave equation in 2D using the scheme (7.38) and imposing Sommerfeld outgoing-wave boundary conditions at the edges of the numerical grid.

Radically different appears the evolution when reflective boundary conditions are imposed, as it is illustrated in Figs 4. Note that the initial evolution (*i.e.*, for which the effects of the boundaries are negligible) is extremely similar to the one shown in Figs. 4, but becomes radically different when the wavefront has reached the outer boundary. As a result of the high (but not perfect!) reflectivity of the outer boundaries, the wave is "trapped" inside the numerical grid and bounces back and forth producing the characteristic interference patterns.

## 7.4 Boundary conditions in 2D

### 7.4.1 Outgoing-wave BCs

Also in 2D, the Outgoing-wave boundary conditions can be expressed by imposing that the at the boundary the solution is locally given by an advection

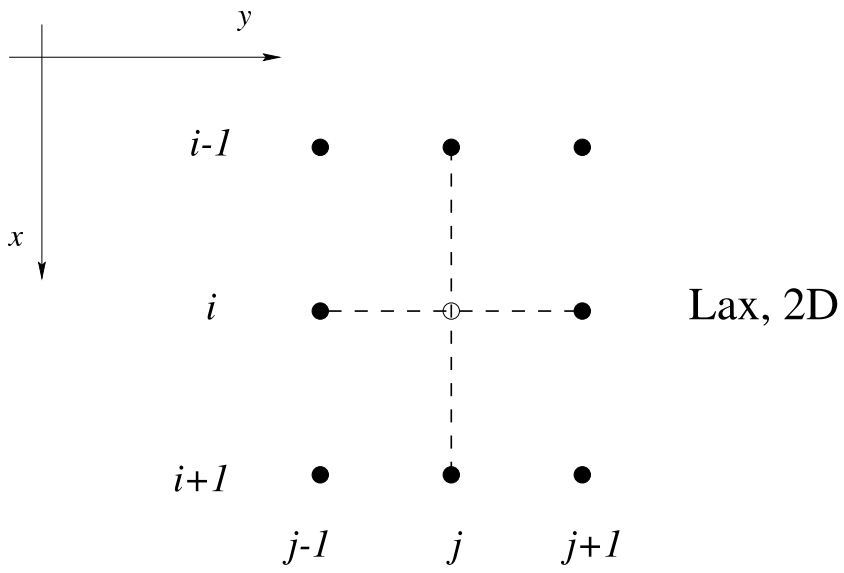


Figure 7.2: Schematic diagram of a Leapfrog evolution scheme in two dimensions. Note that the center of the cross-like stencil is used in this case both at the time-level $n$ (filled circle) and at the time level $n+1$ (filled square).

equation, whose finite-difference, first-order accurate representation can be obtained in a way which is logically similar to the one discussed in 1D. As a result, the Sommerfeld outgoing boundary conditions in the $x$-direction can then be expressed as

$$u_{1,j}^{n+1} = u_{2,j}^{n} + Q_x(u_{2,j}^{n+1} - u_{1,j}^{n})\,, \tag{7.39}$$

$$u_{M,j}^{n+1} = u_{M-1,j}^{n} + Q_x(u_{M-1,j}^{n+1} - u_{M,j}^{n})\,, \tag{7.40}$$

where we have assumed that $i = 1, 2, \ldots M$ and $j = 1, 2, \ldots N$. Similarly, the Sommerfeld outgoing boundary conditions in the $y$-direction can then be expressed as

$$u_{i,1}^{n+1} = u_{i,2}^{n} + Q_y(u_{i,2}^{n} - u_{i,1}^{n+1})\,, \tag{7.41}$$

$$u_{i,N}^{n+1} = u_{i,N-1}^{n} + Q_y(u_{i,N-1}^{n} - u_{i,N}^{n+1})\,, \tag{7.42}$$

where $i = 1, 2, \ldots M$ and

$$Q_x := \frac{1-\alpha}{1+\alpha} = \frac{\Delta x - v_x \Delta t}{\Delta x + v_x \Delta t}\,, \tag{7.43}$$

$$Q_y := \frac{1-\alpha}{1+\alpha} = \frac{\Delta y - v_y \Delta t}{\Delta y + v_y \Delta t}\,, \tag{7.44}$$

### 7.4.2 Periodic BCs

Also in this case, imposing the boundary conditions is logically the same as in the 1D case, so that the periodic boundary conditions for the outer edges in the $x$-direction can be simply expressed as

$$u_{1,j}^{n+1} = u_{M-1,j}^{n}\,, \qquad\qquad u_{M,j}^{n+1} = u_{2,j}^{n}\,, \tag{7.45}$$

with $j = 1, 2, \ldots N$. Similarly, the periodic boundary conditions for the outer edges in the $x$-direction can be simply expressed as

$$u_{i,1}^{n+1} = u_{i,N-1}^{n}\,, \qquad\qquad u_{i,N}^{n+1} = u_{i,2}^{n}\,, \tag{7.46}$$

with $i = 1, 2, \ldots M$.

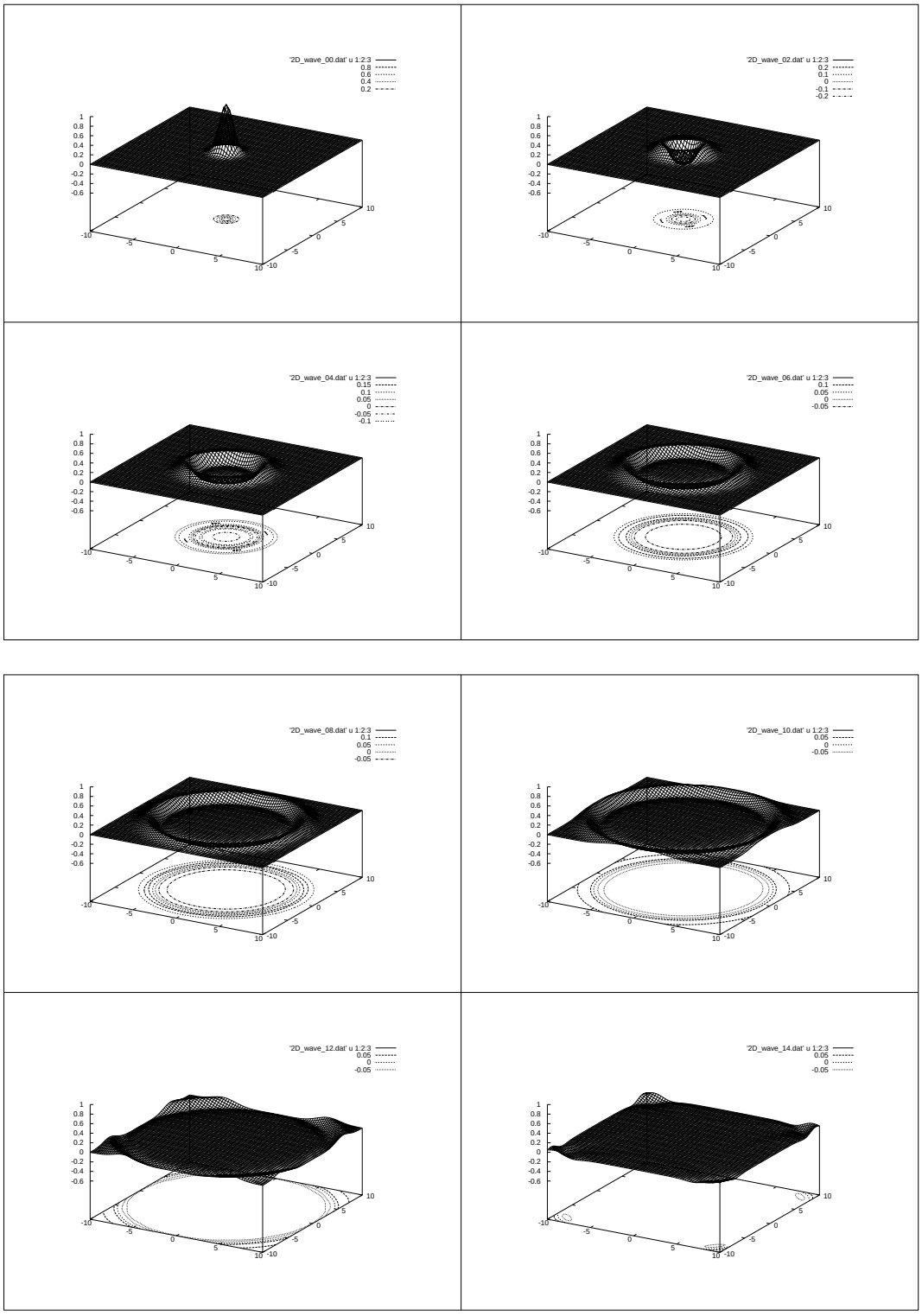


Figure 7.3: Plot of the time evolution of the wave equation when the Leapfrog scheme in 2D is used and Sommerfeld boundary conditions are imposed. Snapshots at increasing times are illustrated in a clockwise sequence.

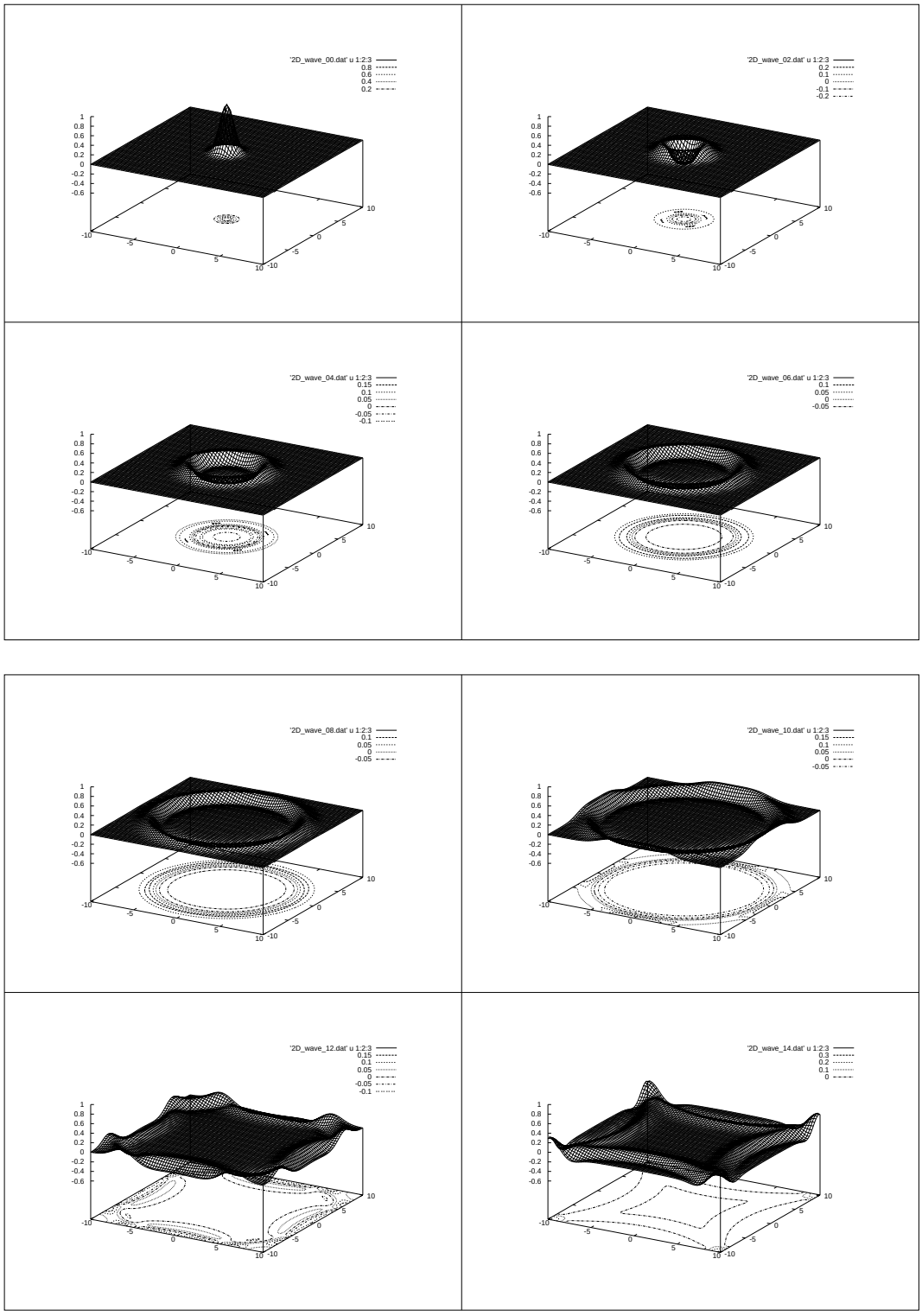


Figure 7.4: Plot of the time evolution of the wave equation when the Leapfrog scheme in 2D is used and Reflecting boundary conditions are applied. Snapshots at increasing times are illustrated in a clockwise sequence.

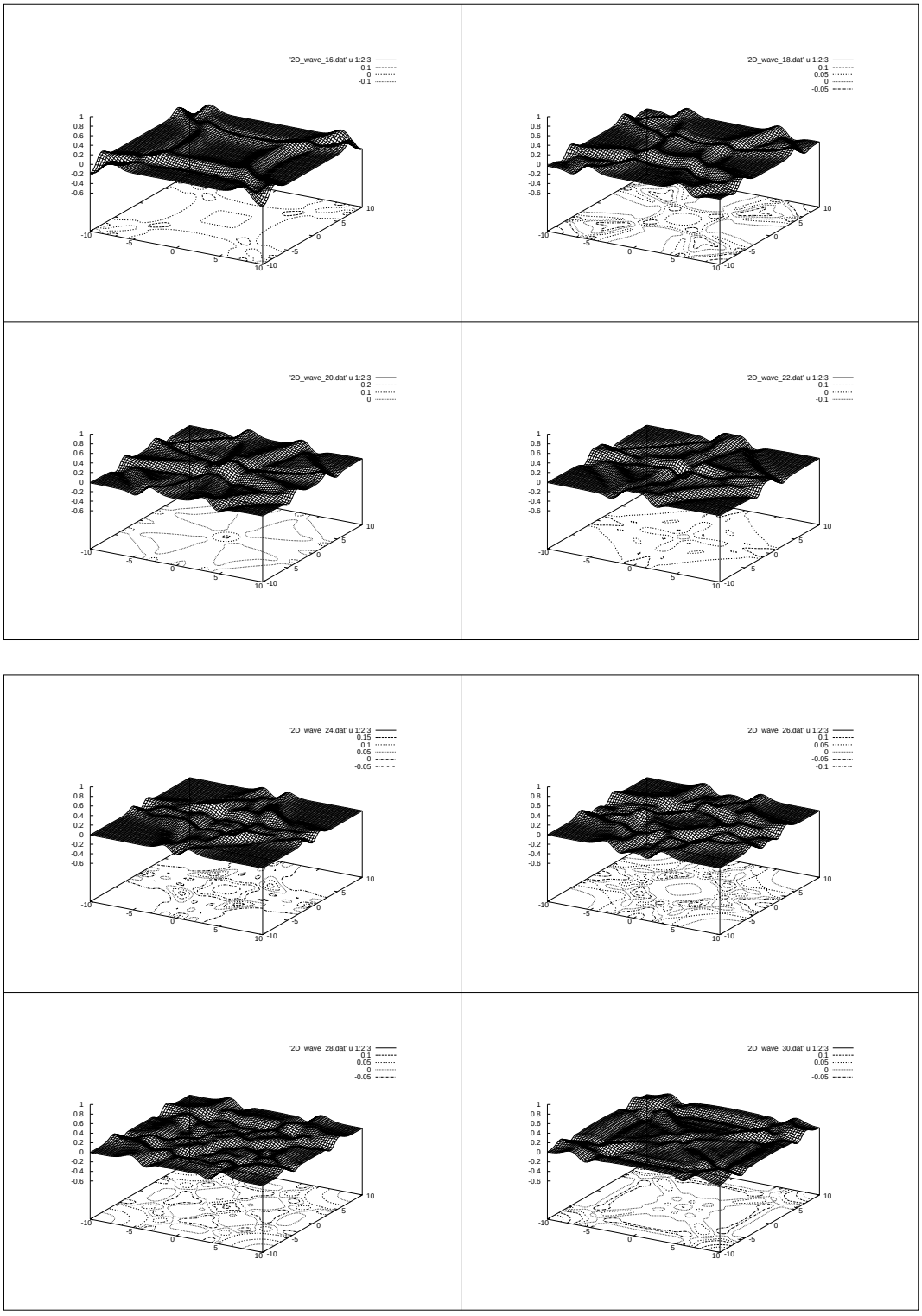


Figure 7.5: Plot of the time evolution of the wave equation when the Leapfrog scheme in 2D is used and Reflecting boundary conditions are applied.

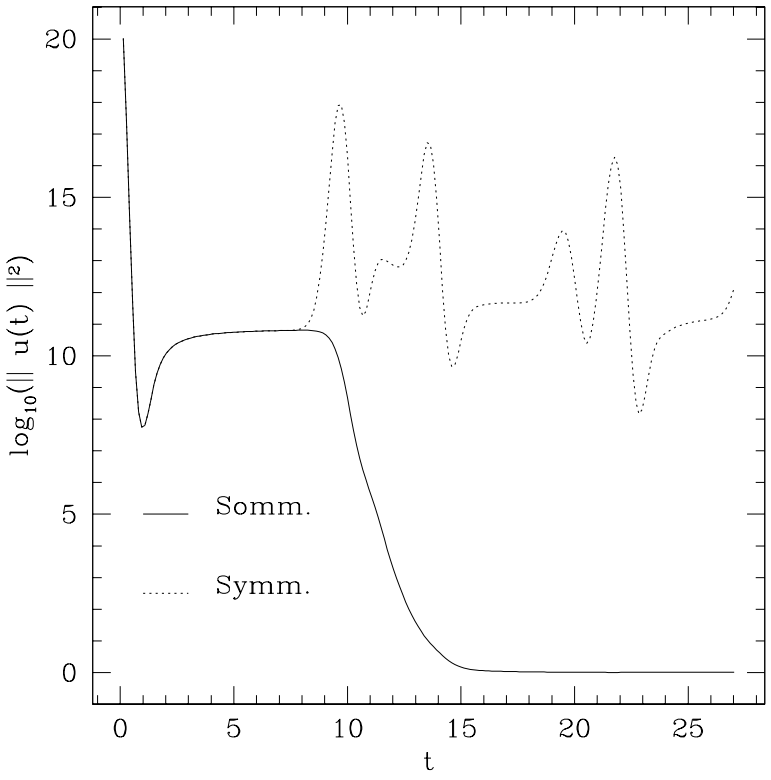


Figure 7.6: Plot of the time evolution of the 2-norm when the Leapfrog scheme in 2D is used. Note the radically different behaviour between Sommerfeld and reflecting boundary conditions.

# Chapter 8

# Parabolic PDEs

## 8.1 Diffusive problems

The inclusion of viscosity in the description of a fluid leads to non trivial complications in the numerical solution of the hydrodynamic equations. From an analytical point of view, the resulting equations are no longer purely hyperbolic PDEs, but rather mixed hyperbolic-parabolic PDEs. This means that the numerical method used to solve them must necessarily be able to cope with the parabolic part of the equations. It is therefore convenient to fully understand the prototypical parabolic equation, the one-dimensional diffusion equation, both analytically and numerically, before attempting to solve any mixed hyperbolic-parabolic PDE.

## 8.2 The diffusion equation in 1D

The description of processes like the heat conduction in a solid body or the spread of a dye in a motionless fluid is given by the one-dimensional *diffusion equation*

$$\frac{\partial u(x,t)}{\partial t} = D\frac{\partial^2 u(x,t)}{\partial x^2} \ . \tag{8.1}$$

Here $D > 0$ is a constant coefficient that determines the magnitude of the "diffusion" in the process under investigation (being given by the thermal conductivity and dye diffusion coefficient respectively in the above mentioned examples).

In what follows, some numerical methods will be presented to solve a simple diffusive problem in 1+1 dimensions distinguishing *explicit* methods from *im-*

*plicit* methods. Before that, however we present a semi-analytic solution of the model parabolic equation (8.1). is presented in Appendix 8.3.

## 8.3 Semi-analytical solution of the model parabolic equation

In this appendix we present details on the derivation of the semi-analytic solution to equation

$$\frac{\partial u(x,t)}{\partial t} = D\frac{\partial^2 u(x,t)}{\partial x^2}\,, \tag{8.2}$$

where $D$ is a constant coefficient. We will first consider homogeneous Dirichlet and then homogeneous Neumann boundary conditions. Because the initial value $u(x,0) = h(x)$ is also needed, we will consider two different initial profiles for the two cases. The solutions we will obtain are to be considered semi-analytical in the sense that they inovlve infinite series and integrals that could not always be evaluated analytically.

### 8.3.1 Homogeneous Dirichlet boundary conditions

Consider a generic problem for which equation (8.1) holds over a domain $[0, L]$. Suppose also that the boundary conditions could be written as *homogeneous* Dirichlet boundary conditions, *i.e.*, $u(0,t) = u(L,t) = 0$, and that at time $t_0 = 0$ the distribution of $u(x,t)$ is that shown in Figure 8.1, which could be written as

$$h(x) := u(x,0) = \begin{cases} 2x/L & \text{if} \quad 0 \le x \le L/2 \\ \\ -2x/L + 2 & \text{if} \quad L/2 < x \le L \end{cases} \tag{8.3}$$

while the boundary conditions are $u(0,t) = u(L,t) = 0$.

The equation could be solved by means of the separation-of-variables technique, *i.e.*, by searching for a solution of the form

$$u(x,t) = f(x)g(t)\,, \tag{8.4}$$

which allows us to write equation (8.2) as

$$f\frac{\partial g}{\partial t} = Dg\frac{\partial^2 f}{\partial x^2}\,. \tag{8.5}$$

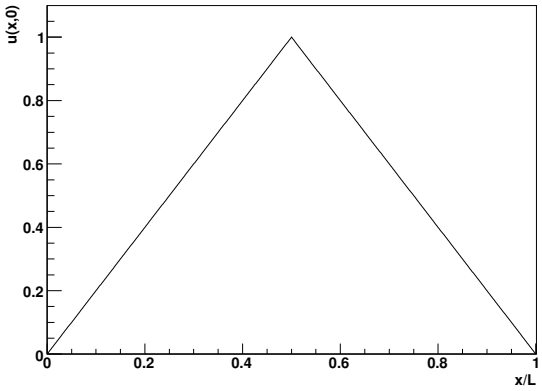


Figure 8.1: Initial value for the diffusive problem (8.1).

Multiplying both sides by $1/(fg)$ the result is

$$\frac{1}{g}\frac{\partial g}{\partial t} = D\frac{1}{f}\frac{\partial^2 f}{\partial x^2}\,. \tag{8.6}$$

The left-hand-side of (8.6) is a function of $t$ only, while the right-hand-side depends only on $x$. Because of that, their common value can only be a constant, with this constant being a negative number because otherwise $g \to \infty$ (and therefore $u \to \infty$) as $t \to \infty$. Thus the common value could be denoted as $-\lambda$ with $\lambda > 0$ and so (8.6) becomes

$$\frac{1}{g}\frac{\partial g}{\partial t} = -\lambda = D\frac{1}{f}\frac{\partial^2 f}{\partial x^2}\,. \tag{8.7}$$

Recalling that the initial condition has been written as $h(x)$, it is possible to write the solution as

$$u(x,t) = h(x)e^{-\lambda t}\,, \tag{8.8}$$

with the requirement that

$$-D\frac{\partial^2 f}{\partial x^2} = \lambda f\,. \tag{8.9}$$

The problem (8.9) is an *eigenvalue problem* for the differential operator $-D\,\partial^2/\partial x^2$ with *eigenvalue* $\lambda$ and *eigenfunction* $f(x)$. The eigenfunctions and eigenvalues will be determined imposing the boundary conditions.

The general solution to (8.9) can be written as

$$f(x) = Ae^{-\mathrm{i}kx} + Be^{\mathrm{i}kx}\,, \tag{8.10}$$

with $k := \sqrt{\lambda/D}$, and with $A$ and $B$ that are constants to be determined through the boundary conditions. Requiring that $f(0) = 0$, yields $B = -A$ and thus

$$f(x) = A\left(e^{-\mathrm{i}kx} - e^{\mathrm{i}kx}\right) = -2\mathrm{i}A\sin kx\,. \tag{8.11}$$

The second boundary condition $f(L) = 0$ allows to find the eigenvalues and the eigenfunctions (and the trivial solution $f(x) = 0$ as well). In fact $\sin(kL) = 0$ as soon as

$$kL = \sqrt{\frac{\lambda}{D}}L = m\pi\,, \quad m = 0, \pm1, \pm2, \pm3, \ldots \tag{8.12}$$

so that the $m$-th eigenvalue and the eigenfunction are

$$\lambda_m = D\left(\frac{m\pi}{L}\right)^2\,, \qquad f_m(x) = \sin\left(\frac{m\pi}{L}x\right). \tag{8.13}$$

The solution to (8.9) will therefore be a linear superposition of the eigenfunctions $f_m(x)$,

$$u(x,t) = \sum_{m=1}^{\infty} a_m \sin\left(\frac{m\pi}{L}x\right)\exp\left[-D\left(\frac{m\pi}{L}\right)^2 t\right]. \tag{8.14}$$

One last condition is still not satisfied, namely, that coming from the initial-value condition, and it is exactly this condition that allows to find the coefficients $a_m$ such that

$$u(x,0) = \sum_{m=1}^{\infty} a_m \sin\left(\frac{m\pi}{L}x\right) = h(x)\,. \tag{8.15}$$

This is a Fourier series on the interval $[0, L]$ of the initial value function $h(x)$ and its coefficients may easily be evaluated keeping in mind the orthogonality property of the trigonometric functions. It is not difficult to show that

$$\int_0^L \sin\left(\frac{m\pi}{L}x\right)\sin\left(\frac{k\pi}{L}x\right)\mathrm{d}x = \begin{cases} 0 & \text{if} \quad k \neq m,\, k = m = 0\,, \\ \\ L/2 & \text{if} \quad k = m\,, \end{cases} \tag{8.16}$$

which allows to compute the coefficients $a_m$ as

$$a_m = \frac{2}{L}\int_0^L h(x)\sin\left(\frac{m\pi}{L}x\right)\mathrm{d}x\,. \tag{8.17}$$

When using the very simple expression for $h(x)$ given by Eq. (8.3), the above computation leads to the final solution which therefore is

$$u(x,t) = \sum_{m=1}^{\infty} a_m \sin\left(\frac{m\pi}{L}x\right)\exp\left[-D\left(\frac{m\pi}{L}\right)^2 t\right], \qquad a_m = 8\frac{\sin(m\pi/2)}{m^2\pi^2}\,. \tag{8.18}$$

In other words, the analytic solution to the diffusion equation is a series of (trigonometric) Fourier modes, each with its own diffusion timescale $\tau_m = L^2/[D(m\pi)^2]$. Since $\tau_m \sim 1/m^2$, modes with smaller wavenumber (larger $m$) will be dissipated faster, while modes with larger wavenumber (smaller $m$) will be dissipated more slowly.

### 8.3.2 Homogeneous Neumann boundary conditions

Once equation (8.2) has been solved for homogeneous Dirichlet boundary conditions it is straightforward to solve it with homogeneous Neumann boundary conditions. In fact, the same procedure could be carried over to yield the correct solution.

Once again, let the mathematical domain be $x \in [0, L]$ for $t > 0$ and if

$$q(x,t) := \frac{\partial u}{\partial x} \tag{8.19}$$

the homogeneous Neumann boundary conditions are written as $q(0,t) = q(L,t) = 0$. Since the boundary conditions require the derivative to vanish, the initial condition is chosen so that this condition is satisfied at $t = 0$ as well. The initial condition will then be

$$h(x) := u(x,0) = 1 + 2\left(\frac{x}{L}\right)^3 - 3\left(\frac{x}{L}\right)^2 . \tag{8.20}$$

Everything that has been said in the previous case up to (8.10) still holds. The boundary conditions now require that

$$f'(x) := \frac{df}{dx} = \mathrm{i}k\left(Ae^{\mathrm{i}kx} - Be^{-ikx}\,,\right) \tag{8.21}$$

vanishes at the boundaries of the domain. From $f'(0) = 0$ follows that $A = B$ while $f'(L) = 0$ leads to the same eigenvalue $\lambda_m = D\left(m\pi/L\right)^2$ as in the previous case. The eigenfunction on the other hand changes since the general solution could be now written as

$$f(x) = A\left(e^{\mathrm{i}kx} + e^{-\mathrm{i}kx}\right) = 2A\cos\left(kx\right) \tag{8.22}$$

so that the eigenvalue and the eigenfunction in this case are

$$\lambda_m = D\left(\frac{m\pi}{L}\right)^2, \qquad f_m(x) = \cos\left(\frac{m\pi}{L}x\right). \tag{8.23}$$

To satisfy the initial condition it is necessary that

$$u(x,0) = \sum_{m=0}^{\infty} a_m \cos\left(\frac{m\pi}{L}x\right) = h(x) \tag{8.24}$$

where the sum now extends from 0 to $\infty$. This is because the orthogonality property of the eigenfunctions, which still holds and could once again be used to compute the coefficients $a_m$, now reads

$$\int_0^L \cos\left(\frac{m\pi}{L}x\right)\cos\left(\frac{k\pi}{L}x\right)\mathrm{d}x = \tag{8.25}$$

Because of this, the initial condition could be written as

$$h(x) = 1+2\left(\frac{x}{L}\right)^3-3\left(\frac{x}{L}\right)^2 = \frac{1}{2}+\sum_{m=1}^{\infty} a_m\cos\left(\frac{m\pi}{L}x\right), \quad a_m = 24\left(\frac{1-\cos(m\pi)}{m^4\pi^4}\right), \tag{8.26}$$

so that the complete solution is

$$u(x,t) = \frac{1}{2}+\sum_{m=1}^{\infty} a_m\cos\left(\frac{m\pi}{L}x\right)\exp\left[-D\left(\frac{m\pi}{L}\right)^2 t\right], \qquad a_m = 24\left(\frac{1-\cos(m\pi)}{m^4\pi^4}\right). \tag{8.27}$$

## 8.4 Explicit updating schemes

### 8.4.1 The FTCS method

The most straightforward way to finite-difference equation (8.1) is by the FTCS method, *i.e.*,

$$\frac{u_j^{n+1}-u_j^n}{\Delta t} = D\frac{u_{j+1}^n-2u_j^n+u_{j-1}^n}{\Delta x^2}+\mathcal{O}(\Delta t,\Delta x^2), \tag{8.28}$$

Unlike the case for a hyperbolic equation, where the FTCS method leads to an unconditionally unstable method, the presence of a second-order spatial derivative in the model parabolic equation (8.1) allows the FTCS method to be conditionally stable [10]. A von Neumann stability analysis leads in fact to the stability criterion

$$\gamma := 2D\frac{\Delta t}{\Delta x^2} \leq 1. \tag{8.29}$$

Recalling that the diffusion timescale over a lengthscale $L$ in Eq. (8.2) is given by $\tau \simeq L^2/D$, the condition (8.29) that lends itself to a physical interpretation: the maximum time step is, up to a numerical factor, the diffusion time across a cell of width $\Delta x$. This stability condition poses a serious limit in the use of the above scheme since the typical timescales of interest will require a number of timesteps which could be prohibitive in multidimensional calculations. The additional fact that the overall scheme is first-order accurate in time only strengthens the need for a different method.

### 8.4.2 The Du Fort-Frankel method and the $\theta$-method

With this objective in mind, it is not difficult to think of a way to avoid the reduced accuracy due to the forward-time finite differencing approach used in FTCS. A simple time-centered finite differencing

$$\frac{u_j^{n+1} - u_j^{n-1}}{2\Delta t} = D\frac{u_{j+1}^n - 2u_j^n + u_{j-1}^n}{\Delta x^2} \tag{8.30}$$

should grant second-order accuracy. Unfortunately, this method is unconditionally unstable. To overcome the stability problem, Du Fort and Frankel [12] suggested the following scheme

$$\frac{u_j^{n+1} - u_j^{n-1}}{2\Delta t} = D\frac{u_{j+1}^n - u_j^{n+1} - u_j^{n-1} + u_{j-1}^n}{\Delta x^2}\,, \tag{8.31}$$

which is obtained from (8.30) with the substitution of $u_j^n$ with $\frac{1}{2}(u_j^{n+1} + u_j^{n-1})$, that is, by taking the time average of $u$ at $x_j$. Writing this explicitly yields

$$u_j^{n+1} = \left(\frac{1-\gamma}{1+\gamma}\right) u_j^{n-1} + \left(\frac{\gamma}{1+\gamma}\right)\left(u_{j+1}^n + u_{j-1}^n\right) + \mathcal{O}(\Delta x^2)\,. \tag{8.32}$$

With this substitution, the method is still explicit and becomes unconditionally stable, but not without a price. A consistency analysis shows, in fact, that the Du Fort-Frankel method could be inconsistent. The local truncation error is [8]

$$\epsilon = \frac{\Delta t^2}{6}\frac{\partial^3 u}{\partial t^3}\bigg|_{j,n} - D\frac{\Delta x^2}{12}\frac{\partial^4 u}{\partial x^4}\bigg|_{j,n} + \left(\frac{\Delta t}{\Delta x}\right)^2 \frac{\partial^2 u}{\partial t^2}\bigg|_{j,n} + \ldots \tag{8.33}$$

$$= \mathcal{O}\left(\Delta t^2, \Delta x^2, \left(\frac{\Delta t}{\Delta x}\right)^2\right)\,, \tag{8.34}$$

which shows that if $\Delta t$ and $\Delta x$ tend to zero at the same rate, *i.e.*, $\Delta t = k\Delta x$ with $k$ being a constant, then the truncation error does not vanish for $\Delta t \to 0$ and $\Delta x \to 0$. Indeed, the solution obtained with this method will not the solution of (8.1), but effectively the solution of the equation

$$\frac{\partial u(x,t)}{\partial t} + k^2\frac{\partial^2 u(x,t)}{\partial t^2} = D\frac{\partial^2 u(x,t)}{\partial x^2}\,, \tag{8.35}$$

which is also known as the *"telegraph equation"* (see [9] for a discussion).

On the other hand, it is also clear from (8.33) that having a timestep $\Delta t = k\Delta x^{1+\varepsilon}$ with $\varepsilon > 0$ will assure the consistency of the method. Of course, the closer is $\varepsilon$ to 1, the smaller will have to be $\Delta x$ in order to achieve consistency. Moreover, accuracy requirements pose an additional constraint on $\varepsilon$. For a first

order-method it is necessary to have $\varepsilon = 1/2$, while to achieve second-order accuracy the requirement is $\varepsilon = 1$. It would be pointless and computationally inefficient to set $\varepsilon > 1$ since in this case the dominant contribution to the truncation error would be determined by the term $O\left(\Delta x^2\right)$ which acts as an upper limit to the overall accuracy order. This means that $\varepsilon$ is constrained to be in the interval $1/2 \leq \varepsilon \leq 1$.

The advantages of the Du Fort-Frankel method over the FTCS scheme should now be easily seen. To achieve first-order accuracy, a timestep $\Delta t = (\Delta x)^{3/2}$ is needed with the Du Fort-Frankel method, while the FTCS scheme requires $\Delta t \approx (\Delta x)^2$, hence smaller timesteps. On the other hand, if a timestep $\Delta t = (\Delta x)^2$ is used, then the Du Fort-Frankel method gains second-order accuracy. Finally, any desired accuracy between first and second order could be achieved with a timestep that is independent of the diffusion coefficient $D$. The only minor drawback of the Du Fort-Frankel scheme lies in the requirement of keeping track of an additional time level.

A generalisation of the Du Fort-Frankel scheme is also straightforward. In particular, when averaging $u_j^{n+1}$ and $u_j^{n-1}$, instead of weighting them equally, it is possible to average them with different weights. The resulting update scheme is therefore

$$\frac{u_j^{n+1} - u_j^{n-1}}{2\Delta t} = D\frac{u_{j+1}^n - 2\left(\theta u_j^{n+1} - (1-\theta)u_j^{n-1}\right) + u_{j-1}^n}{\Delta x^2}, \tag{8.36}$$

where $\theta$ is a variable parameter. In [8] it is shown that the local truncation error for this scheme is

$$\frac{\Delta t^2}{6}\frac{\partial^3 u}{\partial t^3}\bigg|_{j,n} - D\frac{\Delta x^2}{12}\frac{\partial^4 u}{\partial x^4}\bigg|_{j,n} + (2\theta - 1)\frac{2\Delta t}{\Delta x^2}\frac{\partial u}{\partial t}\bigg|_{j,n} + \tag{8.37}$$

$$\frac{\Delta t^2}{\Delta x^2}\frac{\partial^2 u}{\partial t^2}\bigg|_{j,n} + \mathcal{O}\left(\frac{\Delta t^3}{\Delta x^2}, \Delta t^4, \Delta x^4\right), \tag{8.38}$$

which clearly shows that consistency could be achieved for any value of $\theta$ but only if $\Delta t = k\Delta x^{2+\varepsilon}$ with $\varepsilon$ and $k$ being positive real numbers. If $\theta = 1/2$ , on the other hand, the scheme is actually the Du Fort-Frankel scheme [*cf.* expression (8.33)] with the consistency constraints already outlined above. It is therefore clear that, when solving equation (8.1), timestep considerations show that the only viable $\theta$-scheme is the $\theta = 1/2$ scheme, *i.e.*, the Du Fort-Frankel scheme.

### 8.4.3 ICN as a $\theta$-method

We next extend the stability analysis of the $\theta$-ICN discussed in Sect. 3.6.1 to the a parabolic partial differential equation and use as model equation the one-dimensional diffusion equation (8.1). Parabolic equations are commonly solved using implicit methods such as the Crank-Nicolson, which is unconditionally stable and thus removes the constraints on the timestep [*i.e.,* $\Delta t \approx \mathcal{O}(\Delta x^2)$] imposed by explicit schemes [10]. In multidimensional calculations, however, or when the set of equations is of mixed hyperbolic-parabolic type, implicit schemes can be cumbersome to implement since the resulting system of algebraic equations does no longer have simple and tridiagonal matrices of coefficients. In this case, the most convenient choice may be to use an explicit method such as the ICN.

Also in this case, the first step in our analysis is the derivation of a finite-difference representation of the right-hand-side of eq. (8.1) which, at second-order, has the form

$$\mathcal{L}_\Delta(u^n_{j,j\pm1}) = \frac{u^n_{j+1} - 2u^n_j + u^n_{j-1}}{\Delta x^2} + \mathcal{O}(\Delta x^2)\,. \tag{8.39}$$

#### Constant Arithmetic Averages

Next, we consider first the case with constant arithmetic averages (*i.e.,* $\theta = 1/2$) and the expression for the amplification factor after $M$-iterations is then purely real and given by

$$^{(M)}\xi = 1 + 2\sum_{n=1}^{M}(-\gamma)^n\ , \tag{8.40}$$

where $\gamma := (2D\Delta t/\Delta x^2)\sin^2(k\Delta x/2)$. Requiring now for stability that $\sqrt{\xi^2} \leq 1$ and bearing in mind that

$$-1 \leq \sum_{n=0}^{M}(-\gamma)^{n+1} \leq 0\,, \qquad \text{for} \quad \gamma \leq 1\,, \tag{8.41}$$

we find that the scheme is stable for *any* number of iterations provided that $\gamma \leq 1$. Furthermore, because the scheme is second-order accurate from the first iteration on, our suggestion when using the ICN method for parabolic equations is that one iteration should be used *and no more*. In this case, in particular, the ICN method coincides with a FTCS scheme [10].

Note that the stability condition $\gamma \leq 1$ introduces again a constraint on the timestep that must be $\Delta t \leq \Delta x^2/(2D)$ and thus $\mathcal{O}(\Delta x^2)$. As a result and at

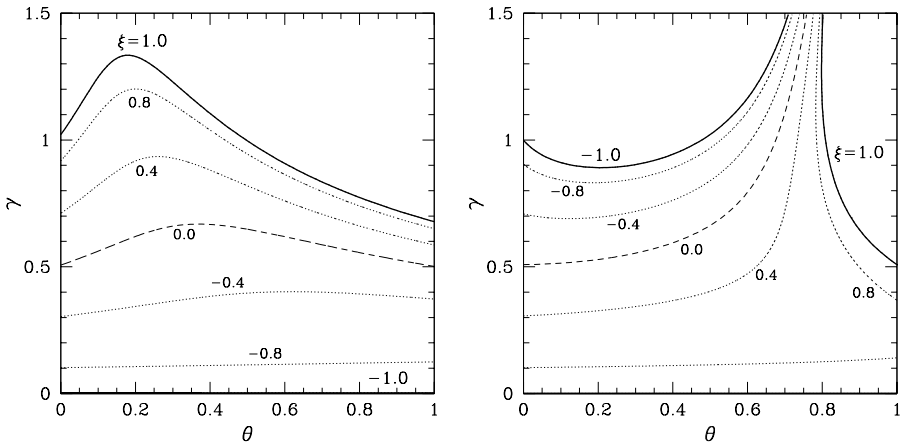


Figure 8.2: *Left panel:* stability region in the $(\theta, \gamma)$ plane for the two-iterations $\theta$-ICN for the diffusion equation (8.1). Thick solid lines mark the limit at which $\xi^2 = 1$, while the dotted contours indicate the values of the amplification factor in the stable region. *Right panel:* same as in the left panel but with swapping the averages between two corrections.

least in this respect, the ICN method does not seem to offer any advantage over other explicit methods for the solution of a parabolic equation [1].

### Constant Weighted Averages

We next consider the stability of the $\theta$-ICN method but focus our attention on a two-iterations scheme since this is the number of iterations needed in the solution of the parabolic part in a mixed hyperbolic-parabolic equation when, for instance, operator-splitting techniques are adopted [10]. In this case, the amplification factor is again purely real and given by

$$\xi \quad = \quad 1 - 2\gamma + 4\gamma^2\theta - 8\gamma^3\theta^2 \,, \tag{8.42}$$

so that stability is achieved if

$$0 \leq \gamma \left(1 - 2\theta\gamma + 4\theta^2\gamma^2\right) \leq 1 \,. \tag{8.43}$$

[1] Note that also the Dufort-Frankel method [12], usually described as unconditionally stable, does not escape the timestep constraint $\Delta t \approx \mathcal{O}(\Delta x^2)$ when a consistent second-order accurate solution is needed [8].

Since $\gamma > 0$ by definition, the left inequality is always satisfied, while the right one is true provided that, for $\gamma < 4/3$,

$$\frac{\gamma - \sqrt{\gamma(4-3\gamma)}}{4\gamma^2} \leq \theta \leq \frac{\gamma + \sqrt{\gamma(4-3\gamma)}}{4\gamma^2}\,. \tag{8.44}$$

The stability region described by the condition (8.44) is shown in the left panel of Fig. 8.2 for $\sin k\Delta x = 1$ and illustrates that the scheme is stable for any value $0 \leq \theta \leq 1$, and also that slightly larger timesteps can be taken when $\theta \simeq 0.2$.

### Swapped Weighted Averages

After some lengthy algebra the calculation of the amplification factor for the $\theta$-ICN method with swapped weighted averages yields

$$\xi = 1 - 2\gamma + 4\gamma^2\theta - 8\gamma^3\theta(1-\theta)\,, \tag{8.45}$$

and stability is then given by

$$-1 \leq 1 - 2\gamma + 4\gamma^2\theta - 8\gamma^3\theta(1-\theta) \leq 1\,. \tag{8.46}$$

Note that none of the two inequalities is always true and in order to obtain analytical expressions for the stable region we solve the condition (8.46) with respect to $\theta$ and obtain

$$\theta \leq \frac{2\gamma - 1 + \sqrt{4\gamma^2 - 4\gamma + 5}}{4\gamma}\,, \tag{8.47a}$$

$$\theta \leq \frac{\gamma(2\gamma-1) - \sqrt{\gamma\,(4\gamma^3 - 4\gamma^2 + 5\gamma - 4)}}{4\gamma^2}\,, \tag{8.47b}$$

$$\theta \geq \frac{\gamma(2\gamma-1) + \sqrt{\gamma\,(4\gamma^3 - 4\gamma^2 + 5\gamma - 4)}}{4\gamma^2}\,. \tag{8.47c}$$

The resulting stable region for $\sin k\Delta x = 1$ is plotted in the right panel of Fig. 8.2 and seems to suggest that arbitrarily large values of $\gamma$ could be considered when $\theta \gtrsim 0.6$ It should be noted, however, that the amplification factor is also severely reduced as larger values of $\gamma$ are used and indeed it is essentially zero in the limit $\theta \to 1$.

## 8.5 Implicit updating schemes

### 8.5.1 The BTCS method

It is common for explicit schemes to be only conditionally stable and in this respect the Du Fort-Frankel method is somewhat unusual. Implicit methods, on the other hand, do not share this property being typically unconditionally stable. A simple example of an implicit finite-differencing scheme can be obtained by considering a discretisation of equation (8.1) in the form of a "backward-time centered-space" (BTCS) scheme, namely

$$\frac{u_j^{n+1} - u_j^n}{\Delta t} = D\frac{u_{j+1}^{n+1} - 2u_j^{n+1} + u_{j-1}^{n+1}}{\Delta x^2} + \mathcal{O}(\Delta t, \Delta x^2)\,. \tag{8.48}$$

As a von-Neumann stability analysis shows [10], the amplification factor is given by

$$\xi = 1/\left[1 + 2\gamma \sin^2\left(\frac{k\Delta x}{2}\right)\right], \tag{8.49}$$

so that the finite-differencing (8.48) is unconditionally stable. This method is also called *backward time*. Rearranging the terms it is easy to obtain

$$-\gamma u_{j-1}^{n+1} + 2(1+\gamma)u_j^{n+1} - \gamma u_{j+1}^{n+1} = 2u_j^n\,, \tag{8.50}$$

which shows that to obtain $u$ at time level $n+1$ is necessary to solve a system of linear equations with a right-hand-side given by $u$ at time level $n$. Luckily, the system is *tridiagonal*, *i.e.*, only the nearest neighbours of the diagonal term are non zero, which allows the use of *sparse matrix* techniques (a matrix is called sparse if the number of non zero elements is small compared to the number of all the elements). The main disadvantage of this scheme, besides that of requiring the simultaneous solution of $N$ algebraic equations, is that it is only first-order accurate in time.

### 8.5.2 The Crank-Nicolson method

Combining the stability of an implicit method with the accuracy of a method that is second-order both in space and in time is possible and is achieved by averaging explicit FTCS and implicit BTCS schemes:

$$\frac{u_j^{n+1} - u_j^n}{\Delta t} = \frac{D}{2}\left[\frac{(u_{j+1}^{n+1} - 2u_j^{n+1} + u_{j-1}^{n+1}) + (u_{j+1}^n - 2u_j^n + u_{j-1}^n)}{\Delta x^2}\right] + \mathcal{O}(\Delta t^2, \Delta x^2)\,. \tag{8.51}$$

This scheme is called *Crank-Nicolson* and is second-order in time since both the left-hand-side and the right-hand-side are centered in $n+1/2$. A more compact and computer-ready representation of the algorithm is then given by

$$-\frac{\gamma}{4}u_{j-1}^{n+1}+\left(1+\frac{\gamma}{2}\right)u_j^{n+1}-\frac{\gamma}{4}u_{j+1}^{n+1}=\frac{\gamma}{4}u_{j-1}^{n}+\left(1-\frac{\gamma}{2}\right)u_j^{n}+\frac{\gamma}{4}u_{j+1}^{n}+\mathcal{O}(\Delta t^2,\Delta x^2)\,. \tag{8.52}$$

The amplification factor in this case is given by

$$\xi=\left[1-\gamma\sin^2\left(\frac{k\Delta x}{2}\right)\right]/\left[1+\gamma\sin^2\left(\frac{k\Delta x}{2}\right)\right]\,, \tag{8.53}$$

so that, as with the fully implicit BTCS scheme, the CN scheme is unconditionally stable. For this reason and for being higher order than BTCS, is the best choice for the solution of simple one-dimensional diffusive problems.

The disadvantage of this scheme with respect to an explicit scheme like the Du Fort-Frankel scheme lies in the fact that in more than one dimension the system of linear equation will no longer be tridiagonal, although it will still be sparse. The extension of the Du Fort-Frankel scheme, on the other hand, is straightforward and with the same constraints as in the one-dimensional case. Because of this and other problems which emerge in multidimensional applications, more powerful methods, like the *Alternating Direction Implicit* (ADI) have been developed. ADI embodies the powerful concept of *operator splitting* or *time splitting*, which requires a more detailed explanation and will not be given in these notes.

# Bibliography


[1] Leveque, R. J. 2002, *Finite Volume Methods for Hyperbolic Problems*, Cambridge University Press, Cambridge, UK.

[2] Potter, D 1973, *Computational Physics*, Wiley, New York, USA

[3] Press, W. H. et al., D 1992, *Numerical Recipes*, Cambridge University Press, Cambridge, UK.

[4] Toro, E. F. 1997, *Riemann Solvers and Numerical Methods for Fluid Dynamics*, Springer.

[5] Vesely, F. J. 1994, *Computational Physics: An Introduction*, Plenum, New York, USA

[6] E. C. Zachmanoglou and D. W. Thoe. *Introduction to Partial Differential Equations with Applications.* Dover Publications, Inc, 1986.

[7] A. Iserles. *A First Course in the Numerical Analysis of Differential Equations.* Cambridge University Press, 1996.

[8] G. D. Smith. *Numerical Solution of Partial Differential Equations: Finite Difference Methods.* Oxford University Press, third edition, 1986.

[9] L. Rezzolla and O. Zanotti *Relativistic Hydrodynamics.* Oxford University Press, 2013.

[10] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. *Numerical Recipes in Fortran 77 - The Art of Scientific Computing*, volume One. Cambridge University Press, second edition, 1997.

[11] R. D. Richtmyer and K. W. Morton. *Difference Methods for Initial-Value Problems.* Interscience - a division of John Wiley & Sons, second edition, 1967.

[12] E. C. Du Fort and S. P. Frankel. Stability conditions in the numerical treatment of parabolic differential equations. *Mathematical Tables and Other Aids to Computation*, 7(43):135–152, July 1953.

[13] R. J. LeVeque. *Finite-difference Methods for Differential Equations - Lecture Notes*. URL = ftp://amath.washington.edu/pub/rjl/papers/amath58X.ps.gz.

[14] S. A. Teukolsky. Stability of the iterated Crank-Nicolson method in numerical relativity. *Physical Review D*, 61(087501), 2000.

[15] J. Crank and P. Nicolson. A practical method for the numerical evaluation of solutions of partial differential equations of the heat-conduction type. *Proc. Camb. Philos. Soc.*, 43:50–67, 1947.

[16] G. J. Barclay, D. F. Griffiths, and D. J. Higham. Theta method dynamics. *LMS Journal of Computation and Mathematics*, 3:27–43, 2000.

[17] A. M. Stuart and A. T. Peplow. The dynamics of the theta method. *SIAM Journal on Scientific and Statistical Computing*, 12(6):1351–1372, 1991.

[18] R. J. LeVeque. *Numerical Methods for Conservation Laws*. Birkhäuser-Verlag, Basel, Switzerland, 1992.