Chapter 12

Partial Differential Equations

Outline of Section

- Classification of PDEs
- Solving Elliptic PDEs
- Solving Hyperbolic PDEs
- Solving Parabolic PDEs

12.1 Classification of PDEs

For partial differential equations (PDEs) we have partial derivatives with respect to more than one variable in each equation in the system. This is the most common form of equation encountered when, e.g., we are modelling the spatial (x) and dynamic (t) characteristics of a system, or when we are modelling a static system in more than one dimension, e.g., (x, y). We will consider the two-dimensional case as an example in this section of the course.

A general PDE can be classified through the coefficients multiplying the various derivatives.

Most of the PDEs of interest in physics are 2nd order linear PDEs. The general form of a 2nd order linear PDE involving two independent variables x and y with solution $u(x, y)$ takes the form

$$\boxed{A\,\frac{\partial^2 u}{\partial x^2} + B\,\frac{\partial^2 u}{\partial x \partial y} + C\,\frac{\partial^2 u}{\partial y^2} + f\left(u, x, y, \frac{\partial u}{\partial x}, \frac{\partial u}{\partial y}\right) = G(x, y),} \tag{12.1}$$

where $f(\ldots)$ is an arbitrary function involving anything other than 2nd derivatives (but involving u somehow). Note that the coefficients A, B and C of the 2nd derivatives can depend on x, y, u, $\partial u/\partial x$ and/or $\partial u/\partial y$. $G(x, y)$ is independent of u and, if present, turns (12.1) into an inhomogeneous PDE.

In analogy with conic sections (in geometry)

$$A\,x^2 + B\,xy + C\,y^2 + D\,x + E\,y + F = 0,$$

(where A, $\ldots$, F are constants) we define the **discriminant**

$$\boxed{Q = B^2 - 4\,A\,C,} \tag{12.2}$$

and mathematically classify PDEs as

$$\boxed{\begin{aligned} Q < 0 &\ : \text{Elliptic,} \\ Q = 0 &\ : \text{Parabolic,} \\ Q > 0 &\ : \text{Hyperbolic.} \end{aligned}} \tag{12.3}$$

For conic sections, $Q < 0$ yields an elliptical curve, $Q = 0$ a parabola and $Q > 0$ a hyperbola. The presence or absence of the inhomogeneous term $G(x, y)$ does not impact the elliptical/parabolic/hyperbolic classification of the PDE.

The classification can be done for 3 or more independent variables (e.g. $u(t, x, y, z)$) but is outside the scope of this course. Luckily, for the common PDEs occurring in physics, the classification based on 2 variables (either 1 time + 1 spatial, or 2 spatial) happens to be the same when you expand the problem into more spatial dimensions. We will not show this, but merely state it for a few examples.

In terms of a more physical picture, basic example of the three types are

(a) **Poisson Equation (Elliptic)**:

$$\nabla^2 u \equiv \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = \rho(x, y), \tag{12.4}$$

i.e. how a solution (potential) reacts to a source (charge density). If $\rho(x, y) = 0$ then the Poisson equation becomes the **Laplace Equation**. Both the homogeneous (Laplace) and inhomogeneous (Poisson) equations are 'elliptic'. With reference to the general equation (12.1), for Poisson's equation $A = 1$, $B = 0$, $C = 1$ hence $Q = -4 < 0$. The 3D version of the Poisson & Laplace equations, where

$$\nabla^2 = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2} + \frac{\partial^2}{\partial z^2},$$

are still elliptic PDEs.

(b) **Diffusion Equation (Parabolic)**:

$$\frac{\partial u}{\partial t} = \frac{\partial}{\partial x}\left(D\,\frac{\partial u}{\partial x}\right), \tag{12.5}$$

where $D \equiv D(x, t)$ is the diffusion coefficient. Here, $A = D$, $B = C = 0$ hence $Q = 0$. (Note that B is the coefficient of $\partial^2 u/\partial t \partial x$ and C pertains to $\partial^2 u/\partial t^2$.) Again, moving to 2 or 3 spatial dimensions we have

$$\frac{\partial u}{\partial t} = \nabla \cdot (D \nabla u)$$

which is still a parabolic PDE. The diffusion coefficient can even be a function of u, $D(u)$, such that

$$\frac{\partial u}{\partial t} = D(u)\nabla^2 u + \frac{\partial D(u)}{\partial u}\,|\nabla u|^2$$

and the PDE is non-linear. This is still a parabolic PDE though.

(c) **Wave Equation (Hyperbolic)**:

$$\frac{\partial^2 u}{\partial t^2} = v^2\,\frac{\partial^2 u}{\partial x^2}, \tag{12.6}$$

where v is the (constant) speed of a wave. Here, $A = v^2$, $B = 0$ and $C = -1$ hence $Q = 4v^2 > 0$. The 3D wave equation

$$\frac{\partial^2 u}{\partial t^2} = v^2\,\nabla^2 u,$$

is still a hyperbolic PDE. The wave speed v can depend on position and even on the wave displacement u and the PDE is still hyperbolic.

Types of physical problems – these are governed by a combination of the PDE and boundary conditions (BCs) and can be classified into the following.

- **Boundary value problems** are relevant to **elliptic** PDEs, as in the case of the Poisson equation where we are solving for the static solution $u(x, y)$ requiring constraints on the boundary. The BCs are specified on a **closed surface**. These problems effectively involve *integrating in from the boundaries* and are solved numerically by **relaxation methods**.
- **Initial value problems** are relevant to **hyperbolic** and **parabolic** PDEs, e.g., as in the diffusion and wave cases where the dynamical solution $u(t, x)$ is sought. In this case we require constraints on the initial state of the solution for all x and we solve this problem by *integrating forward in time* using a **marching method**. We may also impose conditions on the spatial boundaries in this case.

Types of boundary conditions –

u defined on boundaries	: Dirichlet conditions,
$\vec{\nabla}u$ defined on boundaries	: Neumann conditions,
both u and $\vec{\nabla}u$ defined on boundaries	: Cauchy conditions.
u or $\vec{\nabla}u$ applied on diff. parts of boundary	: mixed conditions.
$u(x_r) = u(x_l + L) = u(x_l)$	: Periodic BC (e.g. in x).
$u(-x) = u(x)$	: Reflective BC (e.g. about $x_l = 0$).

where, e.g., x_l and x_r denote the left and right edges of the domain. Since the scope of this section is PDEs with 2 independent variables (i.e. x, y or t, x), for 'boundary surface' we really mean a boundary curve. For elliptic problems, this closed curve could be arbitrarily shaped (e.g. outline of a potato). However we will limit ourselves to a rectangular shaped boundary, and to Cartesian coordinates. For hyperbolic/parabolic initial-value problems, Cauchy conditions are applied only at $t = t_0$. $t \to \infty$ is known as an **open boundary** with nothing being imposed/specified there. For $\vec{\nabla}u$ we mean $\partial u/\partial\zeta$ where $\zeta \to x, y, z$ as appropriate. We will not consider subtleties in defining boundaries and applying BCs when there are 3 (or more) independent variables. A reflective BC is equivalent to having, e.g., $\partial u/\partial x = 0$ at a boundary, so is a special case of a Neumann condition.

We have already seen Dirichlet conditions in section 11.1 and mixed conditions in 11.3 when solving boundary value problems for ODEs.

12.2 Solving Elliptic PDEs

12.2.1 Dirichlet boundary conditions

To solve an elliptic equation we take a similar approach to the finite difference method of section 11.2 by discretising the system onto a 2D lattice of points and using finite difference approximations to the partial derivatives. The lattice is also known as a **mesh** or a **grid**. As an example we start with Laplace's equation

$$\nabla^2 u(x, y) = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = 0, \tag{12.7}$$

with Dirichlet boundary conditions, i.e., u specified on the boundary. We will see that we obtain a matrix equation $\mathbf{A} \cdot \vec{u} = \vec{b}$ (as we did in section 11.3 for 1D boundary value problems). This can be solved for the values of u on the 2D grid (packaged into a vector $\vec{u}$) using a **relaxation method**; an iterative matrix solver such as Jacobi, Gauss-Seidel or SOR. The vector $\vec{b}$ holds the boundary conditions. We will see that there is a way to implement the Jacobi method without resorting to actual matrix algebra, but rather *directly* working on the 2D array of gridded values for u.

We consider a rectangular domain extending $0 \le x \le L_x$ and $0 \le y \le L_y$. There are $n_x + 2$ regularly-spaced points in the x-direction including the edges and similarly n_y+2 in the y-direction.[1]

[1] Notational note: "n_x", "n_y", etc. are what we use here for clarity, but you will often see the (horrible) alternative notation "nx" and "ny" in numerical analysis texts. Beware!

To keep things simple the grid spacing will be taken as h in both directions. (The numerical equations below can readily be generalised for different spacings, e.g., $\Delta x \neq \Delta y$. It does not change the approach.) The location of the grid points are defined as

$$x_i = i\,h \qquad (i = 0,\, 1,\, \ldots\,, n_x + 1) \qquad L_x = (n_x + 1)\,h \tag{12.8}$$

$$y_j = j\,h \qquad (j = 0,\, 1,\, \ldots\,, n_y + 1) \qquad L_y = (n_y + 1)\,h. \tag{12.9}$$

Thus $i = 0$ corresponds to the left-boundary, $i = n_x + 1$ the right-boundary. Similarly $j = 0$ and $j = n_y + 1$ correspond to the bottom and top boundaries, respectively. There are $m = n_x \times n_y$ 'interior' points.

Setting up the FD matrix equation – To illustrate the method we take $n_x = 3$ and $n_y = 3$ and discretise the system using the following scheme;

$$\begin{array}{ccccc}
C_{0,4} & C_{1,4} & C_{2,4} & C_{3,4} & C_{4,4} \\
* & * & * & * & * \\
C_{0,3} & u_{1,3} & u_{2,3} & u_{3,3} & C_{4,3} \\
* & \circ & \circ & \circ & * \\
C_{0,2} & u_{1,2} & u_{2,2} & u_{3,2} & C_{4,2} \\
* & \circ & \circ & \circ & * \\
C_{0,1} & u_{1,1} & u_{2,1} & u_{3,1} & C_{4,1} \\
* & \circ & \circ & \circ & * \\
C_{0,0} & C_{1,0} & C_{2,0} & C_{3,0} & C_{4,0} \\
* & * & * & * & *
\end{array} \tag{12.10}$$

where

$$u_{i,j} = u(x_i, y_j). \tag{12.11}$$

We need to solve for the nine unknown internal points and the boundary conditions are set as $u_{0,0} = C_{0,0}$, $u_{0,1} = C_{0,1}$, etc. For the second order partial derivatives in the Laplacian operator we apply the finite difference approximation seen in section 8.1, i.e., equation (8.9), which yields

$$\left.\frac{\partial^2 u}{\partial x^2}\right|_{i,j} = \frac{u_{i-1,j} - 2u_{i,j} + u_{i+1,j}}{h^2} \qquad \left\{\, \mathcal{O}(h^2) \,\right\}, \tag{12.12}$$

and

$$\left.\frac{\partial^2 u}{\partial y^2}\right|_{i,j} = \frac{u_{i,j-1} - 2u_{i,j} + u_{i,j+1}}{h^2} \qquad \left\{\, \mathcal{O}(h^2) \,\right\}, \tag{12.13}$$

giving the finite difference approximation to the Laplacian

$$\boxed{\nabla^2 u_{i,j} = \frac{u_{i-1,j} + u_{i,j-1} + u_{i+1,j} + u_{i,j+1} - 4u_{i,j}}{h^2} \qquad \left\{\, \mathcal{O}(h^2) \,\right\}.} \tag{12.14}$$

(The $u_{i,j}$ in (12.14) are FD approximations to the true solution, but we dispense with the $\tilde{u}$ notation here.) This Laplacian can be represented as a **pictorial operator** acting on each unknown grid point

$$\nabla^2 u_{i,j} = \frac{1}{h^2} \begin{bmatrix} & 1 & \\ 1 & -4 & 1 \\ & 1 & \end{bmatrix} u_{i,j}. \tag{12.15}$$

As depicted in figure 12.1, this visually shows how the 4 points to the left, right, above and below the centre point (where the Laplacian is being determined) are involved and gives their relative weights

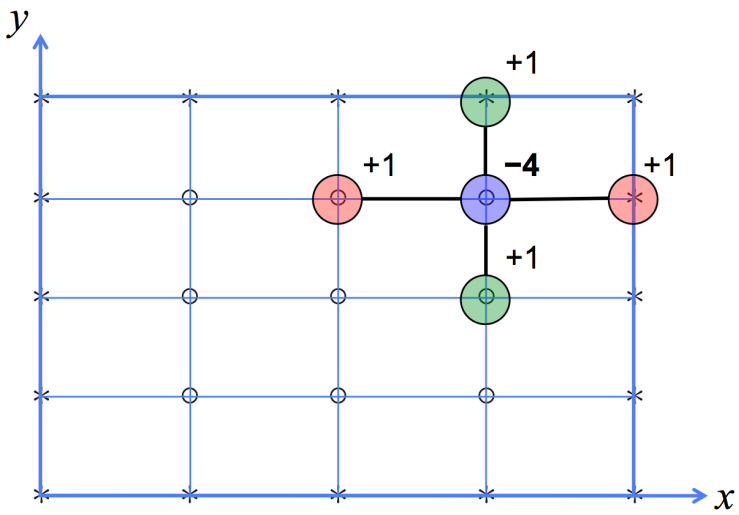

Figure 12.1: **FD stencil for the 2D Laplacian** $\nabla^2 u_{i,j}$. (Note that the $1/h^2$ factor is omitted in the indicated weights.)

(+1). The relative weight of the centre point is −4. This is also known as a **finite difference stencil**.

Equation (12.14) applied at each internal grid point forms one of $m = n_x \times n_y$ simultaneous equations. We can represent the system of simultaneous equations as a matrix equation involving an $m \times m = n_x n_y \times n_x n_y$ matrix (i.e. 9×9 in our example)

$$\left(\begin{array}{ccc|ccc|ccc} -4 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & -4 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & -4 & 0 & 0 & 1 & 0 & 0 & 0 \\ \hline 1 & 0 & 0 & -4 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & -4 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & -4 & 0 & 0 & 1 \\ \hline 0 & 0 & 0 & 1 & 0 & 0 & -4 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & -4 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & -4 \end{array}\right) \left(\begin{array}{c} u_{1,1} \\ u_{1,2} \\ u_{1,3} \\ \hline u_{2,1} \\ u_{2,2} \\ u_{2,3} \\ \hline u_{3,1} \\ u_{3,2} \\ u_{3,3} \end{array}\right) = - \left(\begin{array}{c} C_{0,1} + C_{1,0} \\ C_{0,2} \\ C_{0,3} + C_{1,4} \\ \hline C_{2,0} \\ 0 \\ C_{2,4} \\ \hline C_{3,0} + C_{4,1} \\ C_{4,2} \\ C_{4,3} + C_{3,4} \end{array}\right), \tag{12.16}$$

$$\text{i.e.} \qquad \mathbf{A} \cdot \vec{u} \;=\; \vec{b}. \tag{12.17}$$

Each unknown is one element of the solution vector $\vec{u}$. We use a 2D-array to 1D-array indexing scheme to package the unknowns into the vector. Our chosen mapping from the spatial indices i, j into the row index p of the solution vector $\vec{u}$ is

$$p = j + n_y \times (i - 1), \tag{12.18}$$

e.g. for our case with $n_x = n_y = 3$, unknown $u_{2,3}$ with $i = 2$ and $j = 3$ is located at row $p = 3 + 3 \times (2 - 1) = 6$. This mapping is not unique: We could equally as well have cycled over the x grid points first, and then the y points. (Nb: for 3 spatial dimensions, a 3D-array to 1D-array scheme – an extension of (12.18) – needs to be used.) The horizontal lines in (12.16) serve to visually vertically-partition the matrix and vectors into n_x blocks. Each block has n_y rows.

Solving the FD matrix equation – We can use the Jacobi method to solve this system since the spectral radius of the Jacobi update matrix $\mathbf{T}$ corresponding to $\mathbf{A}$ satisfies $\rho(\mathbf{T}) < 1$. [a]

Starting with some initial guess $\vec{u}^{\,0}$ we can iterate with

$$\vec{u}^{\,n+1} = -\mathbf{D}^{-1} \cdot (\mathbf{L} + \mathbf{U}) \cdot \vec{u}^{\,n} + \mathbf{D}^{-1} \cdot \vec{b}. \tag{12.19}$$

In this case $D^{-1}_{i,j} = -\delta_{i,j}\,\frac{1}{4}$ and we have

$$\vec{u}^{\,n+1} = \frac{1}{4}\mathbf{I} \cdot \left[(\mathbf{L} + \mathbf{U}) \cdot \vec{u}^{\,n} - \vec{b} \right]. \tag{12.20}$$

However this can be represented using the pictorial operator as

$$u^{n+1}_{i,j} = \frac{1}{4} \begin{bmatrix} & 1 & \\ 1 & 0 & 1 \\ & 1 & \end{bmatrix} u^{n}_{i,j}, \tag{12.21}$$

This pictorial operator averages over the nearest four neighbours for each of the internal points. At first sight, it seems that the boundary conditions have been forgotten since $\vec{b}$ is missing in (12.21). However, when updating an unknown on the edge of the internal domain (e.g. $u_{2,1}$) the pictorial operator automatically accesses the necessary boundary value(s) (i.e. $u_{2,0} = C_{2,0}$ in this case).

Thus the algorithm to solve the system using the Jacobi method in 'pictorial operator' form is

- Initialise all grid points including boundary values.
- Operate on all internal points with the pictorial operator, placing all $u^{n+1}_{i,j}$ values in a new array, i.e., leave $u^{n}_{i,j}$ values alone whilst scanning the pictorial operator over each grid point.
- Iterate until the solution has converged.

(See section 3.5, page 23 for details on checking for convergence). Note that Gauss-Seidel and SOR can also be implemented via a (different) pictorial operator scanning over the gridded values $u_{i,j}$ but there are extra considerations (not covered here).

Consistency and Accuracy – The order of accuracy of this FD scheme is $\mathcal{O}(h^2)$. For PDEs, the order is most readily determined from the modified differential equation (MDE). If Taylor expansions for $u_{i,j\pm1}$ and $u_{i\pm1,j}$ are substituted into (12.14), the FD equivalent of Laplace's equation, then the resulting MDE can be shown to be

$$\nabla^2 u = -\frac{1}{12}\left(\frac{\partial^4 u}{\partial x^4} + \frac{\partial^4 u}{\partial y^4} \right) h^2 + \mathcal{O}(h^4) \tag{12.22}$$

demonstrating **consistency** and **2nd-order accuracy**. The order is the rate at which the numerical solution converges to the true solution as the step sizes tend to zero.

Poisson – It is simple to show that the extension to the Poisson case

$$\nabla^2 u = \rho(x, y), \tag{12.23}$$

when using the Jacobi method in pictorial-operator form requires the minor modification

$$u_{i,j}^{n+1} = \frac{1}{4}\begin{bmatrix} & 1 & \\ 1 & 0 & 1 \\ & 1 & \end{bmatrix} u_{i,j}^{n} - \frac{h^2}{4}\rho_{i,j}, \tag{12.24}$$

where the source term is discretised on an identical grid.

Extension to 3D – The methods above can readily be extended to 3D. The unknowns then become $u_{i,j,k}$ where 'k' indexes the third dimension (e.g. z) of the grid. One needs the FD form of the Laplacian in 3D. The (Jacobi) pictorial operator would then be a 3D stencil, accessing the 6 nearest neighbouring grid points.

Physical interpretation of scheme – The pictorial operator shows how the centre point is locally coupled to 4 surrounding points. But these local couplings connect up giving a global coupling between all points; everything is interconnected. Hence we end up with a matrix equation to be solved. Physically, information at one point in space instantaneously propagates everywhere else. This is okay, as 'time' does not come into the equations, so there is no need to worry about causality.

[a] The FD matrix of the 2D (and 3D) Laplacian is an important example of a matrix that is not *strictly* diagonally dominant (since $|A_{ii}|$ equals $\sum_{j\neq i}|A_{ij}|$ for some rows) but does actually work with the Jacobi method. This is because the Jacobi update matrix (formed from $\mathbf{A}$) turns out to have a spectral radius less then unity. Similarly, the FD form of the Laplacian is suitable for Gauss-Seidel and SOR iterative matrix solvers.

12.2.2 Derivative boundary conditions

Consider the case where we have conditions on the derivatives of u at the boundaries

$$u'_{i,j} = Q_{i,j} \tag{12.25}$$

i.e. Neumann boundary conditions. As we did in section 11.2 we will approximate the derivative as a central difference at the boundary involving a fictitious point. Below, we illustrate derivative boundary conditions in y, i.e., applied at $y = 0$ and $y = L_y$. Dirichlet conditions are still used in x. The setup is as follows for $n_x = 3$, $n_y = 3$ interior points to be solved for:

$$\begin{array}{ccccc} & u'_{1,4}=Q_{1,4} & u'_{2,4}=Q_{2,4} & u'_{3,4}=Q_{3,4} & \\ & * & * & * & \\ C_{0,3} & u_{1,3} & u_{2,3} & u_{3,3} & C_{4,3} \\ * & \circ & \circ & \circ & * \\ C_{0,2} & u_{1,2} & u_{2,2} & u_{3,2} & C_{4,2} \\ * & \circ & \circ & \circ & * \\ C_{0,1} & u_{1,1} & u_{2,1} & u_{3,1} & C_{4,1} \\ * & \circ & \circ & \circ & * \\ & u'_{1,0}=Q_{1,0} & u'_{2,0}=Q_{2,0} & u'_{3,0}=Q_{3,0} & \\ & * & * & * & \end{array}, \tag{12.26}$$

(notice that we have omitted the corner points as they do not affect the solution at this order). We then create fictitious points next to the top and bottom boundaries

$$\begin{array}{ccccc} & u_{1,5} & u_{2,5} & u_{3,5} & \\ & * & * & * & \\ C_{0,4} & u_{1,4} & u_{2,4} & u_{3,4} & C_{4,4} \\ * & \circ & \circ & \circ & * \\ C_{0,3} & u_{1,3} & u_{2,3} & u_{3,3} & C_{4,3} \\ * & \circ & \circ & \circ & * \\ C_{0,2} & u_{1,2} & u_{2,2} & u_{3,2} & C_{4,2} \\ * & \circ & \circ & \circ & * \\ C_{0,1} & u_{1,1} & u_{2,1} & u_{3,1} & C_{4,1} \\ * & \circ & \circ & \circ & * \\ C_{0,0} & u_{1,0} & u_{2,0} & u_{3,0} & C_{4,0} \\ * & \circ & \circ & \circ & * \\ & u_{1,-1} & u_{2,-1} & u_{3,-1} & \\ & * & * & * & \end{array} . \tag{12.27}$$

This system has $m = n_x \times (n_y + 2)$ unknowns. The fictitious points then become the boundaries and can be initialised using the definition of the central difference scheme. For the lower boundary at $y = 0$ this yields

$$\left.\frac{\partial u}{\partial y}\right|_{i,0} = \frac{u_{i,1} - u_{i,-1}}{2h} = Q_{i,0} \qquad \text{giving} \qquad u_{i,-1} = u_{i,1} - 2h\, Q_{i,0}. \tag{12.28}$$

A similar condition needs to be applied at $y = L_y$ to give u_{i,n_y+2} in terms of Q_{i,n_y+1}. This system of simultaneous equations forms an $m \times m = n_x(n_y + 2) \times n_x(n_y + 2)$ matrix equation, which can then be solved iteratively as before, taking care to update the fictitious points along with the 'real' ones at each iteration.

12.2.3 Spectral methods – Solving the Poisson Equation by FFT

Methods using FTs to solve PDEs are known as **spectral methods**. As an example we consider the general Poisson equation

$$\nabla^2 u(\vec{x}) = \rho(\vec{x}), \tag{12.29}$$

where ρ is some source density and u is some potential that we are seeking to solve for. For electrostatic problems this specialises to

$$\nabla^2 \phi(\vec{x}) = -\rho_c(\vec{x})/\epsilon_o,$$

where ϕ and ρ_c are the electrostatic potential and charge-density, respectively, and for (Newtonian) gravitational problems

$$\nabla^2 \Phi(\vec{x}) = 4\pi G \rho(\vec{x})$$

where Φ is gravitational potential, G is Newton's constant and ρ is the mass density of matter.

The iterative method works from before works for any boundary conditions; periodic, fixed, etc. However, **for the case of periodic BCs**, using FFTs is much more efficient.

The exact analytical solution can easily be written in terms of FTs as follow. As we have seen the Laplacian is replaced by the factor $-|\vec{k}|^2$ in Fourier space so that exact solution in

Fourier space is

$$\tilde{u}(\vec{k}) = -\frac{\tilde{\rho}(\vec{k})}{|\vec{k}|^2}. \tag{12.30}$$

To obtain the exact solution in real (coordinate) space we just need to inverse transform the above

$$u(\vec{x}) = \mathcal{F}^{-1}\left[-\frac{\tilde{\rho}(\vec{k})}{|\vec{k}|^2}\right]. \tag{12.31}$$

In practice for general $\rho(\vec{x})$, tractable, analytic expressions for the Fourier transform integrals will be difficult (even impossible) to find.

Equipped with a discrete Fourier transform, the obvious way to solve (12.29) numerically (in, e.g., 2D) would seem to be to take an FFT of the gridded source density $\rho_{n,m}$ to get $\tilde{\rho}_{p,q}$, divide by $|\vec{k}_{p,q}|^2$ (where $(k_x)_p = p\pi/\Delta x$ and $(k_y)_q = q\pi/\Delta y$) and then take the backwards DFT to get $u_{n,m}$. However, this does not yield the same thing as solving the discretised PDE, as we did in section 12.2 (i.e. equation (12.24)). The reason is that we have not yet taken into account the finite difference approximation of the Laplacian.

12.2.4 Application to discrete system – 1-D

We discretise the system as usual with grid spacing h and use a second order finite difference scheme for the Laplacian

$$\frac{u_{j+1} - 2u_j + u_{j-1}}{h^2} = \rho_j. \tag{12.32}$$

Expanding each term through the backwards DFT we have

$$u_j = \frac{1}{N}\sum_p \tilde{u}_p e^{-i2\pi pj/N}, \tag{12.33}$$

and

$$u_{j\pm1} = \frac{1}{N}\sum_p \tilde{u}_p e^{-i2\pi p(j\pm1)/N} = \frac{1}{N}\sum_p \tilde{u}_p e^{\mp i2\pi p/N} e^{-i2\pi pj/N}. \tag{12.34}$$

Putting these into (12.32) gives

$$\frac{1}{N}\sum_p \tilde{u}_p e^{-i2\pi pj/N}\left[e^{+i2\pi p/N} + e^{-i2\pi p/N} - 2\right] = \frac{h^2}{N}\sum_p \tilde{\rho}_p e^{-i2\pi pj/N}. \tag{12.35}$$

Equating the bits inside the sum we have

$$\tilde{u}_p = \frac{h^2\tilde{\rho}_p}{\left[e^{+i2\pi p/N} + e^{-i2\pi p/N} - 2\right]}, \tag{12.36}$$

and since

$$e^{i\theta} = \cos\theta + i\sin\theta, \tag{12.37}$$

the denominator simplifies to

$$\left[e^{+i2\pi p/N} + e^{-i2\pi p/N} - 2\right] = \left[e^{+i\pi p/N} - e^{-i\pi p/N}\right]^2 = (2i)^2\sin^2(\pi p/N), \tag{12.38}$$

giving

$$\tilde{u}_p = -\frac{h^2\tilde{\rho}_p}{4\sin^2(\pi p/N)} \tag{12.39}$$

Notice that the 'monopole' $\tilde{u}_0$ diverges if $\tilde{\rho}_0 \neq 0$. (Note that $\tilde{\rho}_0$ is the average source density on the finite, periodically repeated, domain.) Physically, the potential is only defined to within a

constant so we can safely set $\tilde{\rho}_0 = 0$ to make the potential in real space zero when for a system where all the 'mass' is uniformly spread out. We then obtain the solution in coordinate space by taking the inverse DFT of (12.39)

$$u_j = -\frac{h^2}{4N} \sum_p \frac{\tilde{\rho}_p}{\sin^2(\pi p/N)} e^{-i2\pi pkj/N}. \tag{12.40}$$

In general the DFTs are replaced by black-box FFT routines from standard libraries and the algorithm can be summarised as

- Generate source lattice ρ_j.
- FFT source lattice $\rho_j \rightarrow \tilde{\rho}_p$.
- Solve for $\tilde{u}_p$ (and remove monopole).
- Inverse FFT $\tilde{u}_p \rightarrow u_j$.

The method can be easily extended to higher dimensions, e.g., 2D or 3D lattice.

12.3 Solving Hyperbolic PDEs

The wave equation involves second order derivatives in both time and space and is

$$\frac{\partial^2 u}{\partial t^2} - v^2 \frac{\partial^2 u}{\partial x^2} = 0, \tag{12.41}$$

in the 1D, linear case where v is the phase speed. We are looking to solve for $u(x,t)$ with $0 \le x \le L$ and $t_i \le t \le t_f$, i.e., an initial value problem. The boundary conditions for this system have to be provided for all x at t_i (i.e. the initial condition) and at the edges ($x = 0$, $x = L$) for $t_i < t \le t_f$. We can discretise the solutions using h as the spatial step and Δt as the time step. The locations of the grid points in (x,t) are defined as

$$x_i = i\,h \qquad (i = 0,\, 1,\, \ldots\,, n_x) \qquad h = L/n_x \tag{12.42}$$

$$t^j = t_i + j\,\Delta t \qquad (j = 0,\, 1,\, \ldots\,, n_t) \qquad \Delta t = (t_f - t_i)/n_t. \tag{12.43}$$

This approach is depicted in figure 12.2. The notation for $u(x_i, t^j)$ is

$$u_i^j = u(x_i, t^j)$$

i.e. the subscript & superscript denote the space and time index, respectively.

We could try to solve (12.41) numerically 'as is' by using second order finite difference approximations in both the space and time derivatives. We would effectively end up with a multi-point method since $\partial^2 u/\partial t^2$ would link 3 time steps; t^{j-1}, t^j and the next (unknown) time t^{j+1}. However, this is not the best way in this particular case. Some physical insight points to a better way, not involving the raw wave-equation.

Advection equations

This wave equation actually splits into two uncoupled 1st-order PDEs,

$$\frac{\partial f}{\partial t} + v\frac{\partial f}{\partial x} = 0 \quad , \quad \frac{\partial g}{\partial t} - v\frac{\partial g}{\partial x} = 0, \tag{12.44}$$

where $v \ge 0$ is the phase speed. They have general solutions $f(x,t) = F(x - vt)$ (arbitrary forward propagating disturbance) and $g(x,t) = G(x + vt)$ (arbitrary function propagating backwards), respectively. These are known as the **advection equations**, or more correctly the 'constant velocity advection' (CVA) equations. The term "convection" equation is also often used. The

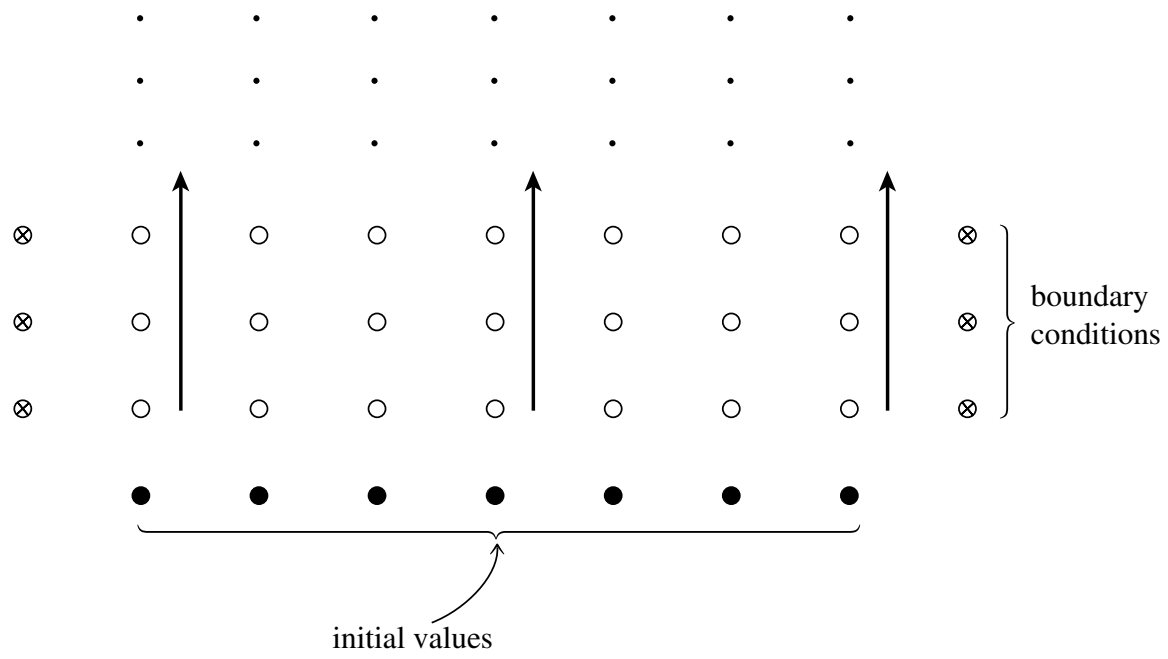

Figure 12.2: Initial value problem in (x,t) and the 1+1 discretisation scheme. The solution is integrated forward in time from an initial solution at a given "time slice", with boundary conditions given at the extremal values of x over all times. From Numerical Recipes.

general solution of the linear wave-equation is thus $u(x,t) = F(x-vt)+G(x+vt)$. In 3D the CVA equation looks like

$$\frac{\partial f}{\partial t} + \vec{v}\cdot\nabla f = 0.$$

The solution is $f(\vec{r},t) = f(t - \vec{v}\cdot\vec{r}/v^2)$, in a Cartesian coordinate system.

So the solution of the wave equations reduces to solving an advection equation for a vector of 2 functions. Therefore, most of the effort in numerically solving hyperbolic PDEs concentrates on solving the advection equation

$$\frac{\partial u}{\partial t} + v_x\frac{\partial u}{\partial x} = 0, \tag{12.45}$$

where now v_x is a velocity, so that its sign determines the direction of propagation. In practical terms, what we will be doing can be considered as an extension of the ODE methods (for initial value problems), seen in sections 8–9, which were used to solve

$$\frac{du}{dt} = f(t,u),$$

but where now the function f depends on the spatial gradient of u. We also have to integrate the solution forward in time, simultaneously across the whole spatial domain.

Coupled conservation-law + equation of motion

Physically, wave equations often arise from combining a *conservation law* (e.g. conservation of mass) with an *equation of motion*. For example, in the case of an acoustic wave in gas

$$\frac{\partial \rho}{\partial t} + \nabla\cdot(\rho\vec{u}) = 0, \qquad \text{(mass conservation law)} \tag{12.46}$$

$$\frac{\partial(\rho\vec{u})}{\partial t} + \vec{u}\cdot\nabla(\rho\vec{u}) = -\nabla P, \qquad \text{(eqn. of motion)} \tag{12.47}$$

where ρ is mass density, $\vec{u}$ is the fluid velocity and P is the pressure. These are coupled 1st-order PDEs. For example, using the adiabatic gas law ($PV^\gamma = \text{const.}$) which yields $P\rho^{-\gamma} = P_o\rho_o^{-\gamma}$ (where P_o and ρ_o are the unperturbed values in the absence of a wave) and then ignoring the non-linear term $\vec{u}\cdot\nabla(\rho\vec{u})$ yields the wave equation

$$\frac{\partial^2\rho}{\partial t^2} = \frac{\gamma P_o}{\rho_o}\nabla^2\rho$$

when the the adiabatic law is linearised. We identify $c = \sqrt{\gamma P_o/\rho_o}$ as the *sound speed*.

In 1D linearised versions of the mass conservation law and equation of motion are

$$\begin{aligned} \frac{\partial\rho}{\partial t} + \rho_o\frac{\partial u}{\partial x} &= 0, \\ \frac{\partial u}{\partial t} + \frac{c^2}{\rho_o}\frac{\partial\rho}{\partial x} &= 0, \end{aligned} \tag{12.48}$$

i.e. a set of coupled equations which can be combined into a single PDE of conservative form,

$$\frac{\partial\vec{U}}{\partial t} = -\frac{\partial(\mathbf{F}\cdot\vec{U})}{\partial x}, \tag{12.49}$$

where $\mathbf{F}$ is known as a **flux operator**. In our example

$$\vec{U} \equiv \begin{bmatrix} \rho(x,t) \\ u(x,t) \end{bmatrix} \qquad \text{and} \qquad \mathbf{F} \equiv \frac{1}{\rho_o}\begin{bmatrix} 0 & \rho_o^2 \\ c^2 & 0 \end{bmatrix}. \tag{12.50}$$

Equation (12.49) looks like an advection equation for a vector of functions, when we bear in mind that $\mathbf{F}$ is a matrix of constants so can come outside the spatial derivative. For EM waves in vacuum (plane waves propagating in the x-direction), $\vec{U} = [E_y, cB_z]^T$ and the off-diagonal elements of $\mathbf{F}$ would become c (speed of light in vacuum).

12.3.1 Upwind method

We take inspiration from the physics of advection and chose a one-sided finite difference approximation (FDA) for $\partial u/\partial x$ in the *direction from which information propagates*. For $v_x > 0$ we have

$$\left.\frac{\partial u}{\partial x}\right|_i^j = \frac{u_i^j - u_{i-1}^j}{h} \qquad \{\,\mathcal{O}(h)\,\}. \tag{12.51}$$

We chose the simple, forward difference scheme (FDS) for the partial time derivative

$$\left.\frac{\partial u}{\partial t}\right|_i^j = \frac{u_i^{j+1} - u_i^j}{\Delta t} \qquad \{\,\mathcal{O}(\Delta t)\,\}. \tag{12.52}$$

The resulting FDE equation for advection (for +ve v_x) is thus

$$\boxed{u_i^{j+1} = u_i^j - \frac{|v_x|\Delta t}{h}\left(u_i^j - u_{i-1}^j\right)}, \tag{12.53}$$

which is known as the **upwind method** (also known as the "donor cell method"). The factor

$$a = |v_x|\Delta t/h, \tag{12.54}$$

is known as the **advection number**, an important dimensionless parameter with respect to the numerical properties of FD methods for hyperbolic PDEs (as we shall see shortly).

Figure 12.3 shows the **finite difference stencil** corresponding to equation (12.53). Such stencils are used to visualise the points involved in the algorithm.

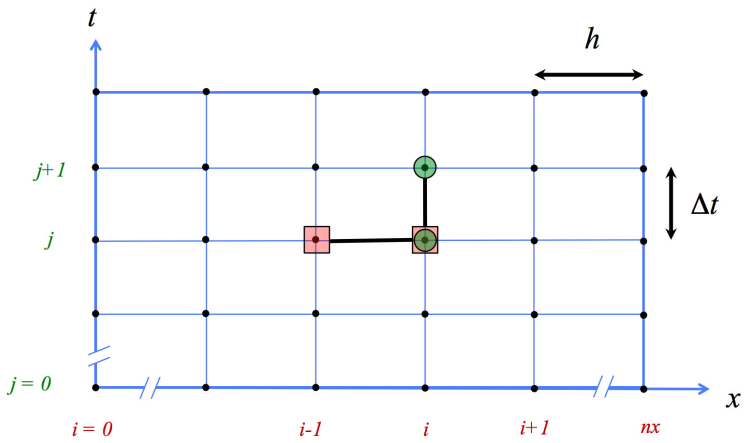

Figure 12.3: **FD stencil for upwind method.** Square and circle symbols denote contributions from $\partial u/\partial x$ and $\partial u/\partial t$ terms, respectively. The $v_x > 0$ case is shown.

For $v_x < 0$ we choose

$$\left.\frac{\partial u}{\partial x}\right|_i^j = \frac{u_{i+1}^j - u_i^j}{h} \qquad \{\,\mathcal{O}(h)\,\},$$

and thus the FDE is

$$u_i^{j+1} = u_i^j + a(u_{i+1}^j - u_i^j). \tag{12.55}$$

The upwind method is classed as an **explicit** scheme since $\partial u/\partial x$ depends on the known values $u(t^j)$ rather than the unknown ones $u(t^{j+1})$. Because the physical speed of information propagation is finite for hyperbolic PDEs, explicit schemes are generally best. Implicit schemes are more computationally intensive for the same accuracy. They generally offer no benefits for hyperbolic PDEs.

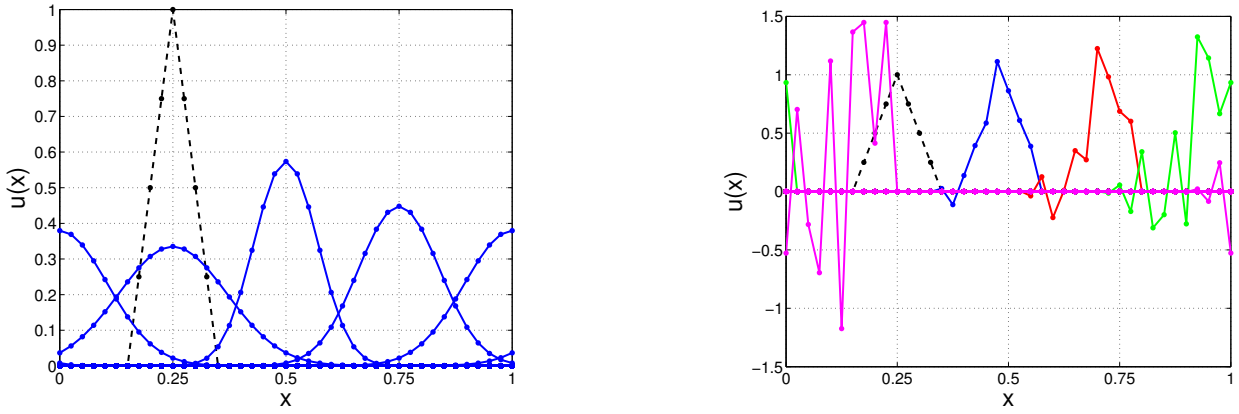

Figure 12.4: **Advection via the upwind method.** (Left) Advection number $a = 0.5$, and (Right) $a = 1.05$. The initial profile is a triangle (dotted black lines). Parameters: $v_x = +1$, $h = 1/40$, (left) $\Delta t = h/2$ and (right) $\Delta t = 1.05h$. The profile moves to the right and eventually returns to its starting location because of the periodic BCs. The 4 snapshots are at $t = 0.25, 0.5.\, 0.75, 1.0$. For $a = 0.5$, severe numerical diffusion can be seen. For $a = 1.05$ the upwind method is unstable.

Figure 12.4 shows the upwind method applied to an initially triangular profile, in the case of positive v_x. The spatial resolution is quite coarse ($n_x = 40$). For an advection number of $a = 0.5$ the profile unphysically spreads and blurs as it moves, due to inaccuracies of the upwind method.

For $a = 1.05$ the profile breaks up due to numerical instability. We will look more closely at these two issues (numerical diffusion and instability) below.

Consistency & accuracy – The modified differential equation obtained from (12.53) is

$$\frac{\partial u}{\partial t} + v_x \frac{\partial u}{\partial x} = -\frac{1}{2}\frac{\partial^2 u}{\partial t^2}\Delta t + \mathcal{O}(\Delta t^2) + \frac{1}{2}\frac{\partial^2 u}{\partial x^2} v_x h + \mathcal{O}(h^2). \tag{12.56}$$

The RHS vanishes as $\Delta t \to 0$ and $h \to 0$ which demonstrates that the upwind method is consistent with the advection PDE. The leading error terms show that it is 1st-order accurate in time and in space; it is a '$\boldsymbol{\mathcal{O}(\Delta t) + \mathcal{O}(h)}$ **scheme**'.

Numerical diffusion – The above MDE can be rewritten as

$$\begin{aligned} \frac{\partial u}{\partial t} + v_x \frac{\partial u}{\partial x} &= \frac{|v_x| h}{2}(1-a)\frac{\partial^2 u}{\partial x^2} + \mathcal{O}(\Delta t^2) + \mathcal{O}(h^2) \\ &\approx D_{\text{num}} \frac{\partial^2 u}{\partial x^2}, \end{aligned} \tag{12.57}$$

which is really interesting. It shows that the upwind method is really giving us the solution of the advection equation with some non-physical diffusion (the leading term on the RHS). This 'numerical diffusion' has a diffusion coefficient D_{num} that grows, the larger h is. Numerical diffusion causes a pulse with sharp features (e.g. top hat function) to artificially blur out as it propagates along. The higher order error terms further artificially modify the dynamics but are weaker than numerical diffusion.

Matrix equation – Practical implementation of (12.53) does not (and should not!) be done via matrices. However formulating the equations formed by the FDE applied at each spatial grid point, as a matrix equation is useful both conceptually and for, e.g., undertaking a stability analysis. We assume ***periodic spatial boundary conditions***, which is

$$u^j_{n_x} = u^j_0, \tag{12.58}$$

on our choice of spatial grid. Although there are $n_x + 1$ spatial points, periodic BCs reduces the number of unknowns to $m = n_x$. The matrix equation (for the $v_x > 0$ case) is then

$$\begin{pmatrix} u_1 \\ u_2 \\ u_3 \\ \vdots \\ u_{n_x} \end{pmatrix}^{j+1} = \begin{pmatrix} (1-a) & 0 & 0 & \dots & 0 & a \\ a & (1-a) & 0 & \dots & 0 & 0 \\ 0 & a & (1-a) & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \dots & a & (1-a) \end{pmatrix} \cdot \begin{pmatrix} u_1 \\ u_2 \\ u_3 \\ \vdots \\ u_{n_x} \end{pmatrix}^{j},$$

$$\vec{u}^{\,j+1} = \mathbf{T} \cdot \vec{u}^{\,j}, \tag{12.59}$$

where $\mathbf{T}$ is the update matrix (analogous to those seen with ODE FD methods in sections 2–9).

12.3.2 Stability analysis – von Neumann (Fourier) method

Von Neumann stability analysis checks whether the FDE causes a Fourier mode to grow exponentially in amplitude (instability) or not. It is arguably more physically intuitive than the matrix method (coming up soon), giving us insight about how the FD grid affects the phase speed and highlighting the presence of **numerical dispersion**. However, it only works for periodic boundary conditions.

The procedure is to substitute a Fourier mode for $u(x,t)$

$$\boxed{u_i^j = \exp\big[i(kx_i - \omega t^j)\big] = (g)^j \, \exp(ikx_i),} \tag{12.60}$$

into the FDE. This mode is sampled at discrete spatial and temporal points. Here the amplification factor g (a complex quantity) is raised to the power j (the time index). k and ω are the wave-number and angular frequency of the mode. The relation between u_i^j and adjacent points on the FD grid is

$$u_i^{j+1} = g\, u_i^j \qquad , \qquad u_{i\pm1}^j = u_i^j \, \exp(\pm i\, k\, h). \tag{12.61}$$

Substituting into the upwind method FDE (12.53) yields

$$g \cancel{u_i^j} = \cancel{u_i^j}(1-a) + \cancel{u_i^j}\, a\, e^{-ikh},$$

and then separating out into real and imaginary parts gives

$$g = (1-a) + a\cos(kh) \;-\; i\, a \sin(hk).$$

After some algebra we get

$$|g|^2 = 1 - 2a(1-a)[1-\cos(kh)],$$

and then using the identity $1-\cos(2\theta) = 2\sin^2\theta$ yields

$$|g|^2 = 1 - 4a(1-a)\sin^2(kh/2). \tag{12.62}$$

The maximum wave-number allowable on the grid is

$$k_{max} = \frac{\pi}{h} \qquad (\text{i.e. } \lambda_{min} = 2h\,), \tag{12.63}$$

so that $\sin^2(kh/2)$ ranges from 0 to 1. The condition for stability is

$$|g| \le 1,$$

i.e., $1 - 4a(1-a) \le 1$ which yields $a \ge 0$ and

$$\boxed{a \le 1 \qquad \text{or equivalently} \qquad \Delta t \le h/|v_x|.} \tag{12.64}$$

This is known as the **CFL condition** (named after Courant, Friedrich & Lewy). The CFL condition is equivalent to

$$|v_x| \le v_{\text{grid}}, \tag{12.65}$$

i.e. the physical, speed of propagation of information must be less than the *grid speed* $v_{\text{grid}} = h/\Delta t$. If violated, then the domain of dependence of a given grid point is physically larger than the FD scheme can reach, which leads to problems.

Note that in practice the time step needs to be quite a bit smaller than that given by the CFL condition to get good accuracy.

Referring back to the form of the Fourier mode (12.60), we will see in Section 5.2 that a discrete (and finite) set of k values are allowed on a spatial grid. For a given, allowed wavelength the numerical scheme determines the effective numerical wave frequency ω_{num} (since $\omega_{\text{num}} = i\,\ln g/\Delta t$) that can propagate. ω_{num} can differ from the true value of kv_x, especially at short wavelengths, leading to incorrect phase speed, a phenomenon known as **numerical dispersion**. If ω_{num} has an imaginary part, then there will be unphysical **numerical damping** (or growth!) of the Fourier mode.

12.3.3 Stability analysis – matrix method

This is the same as the approach used for coupled ODEs in section 8.6. In this context the coupled variables are the u_i's at each spatial grid point.

We can define $\tilde{\vec{u}}^j = \vec{u}^j + \vec{\epsilon}^j$ relating the FD approximation ($\tilde{\vec{u}}^j$) to the true solution of the exact PDE ($\vec{u}^j$) via an error ($\vec{\epsilon}^j$). Ignoring terms responsible for gradual, increasing loss of accuracy, as before, we are left with $\vec{\epsilon}^{\,j+1} = \mathbf{T} \cdot \vec{\epsilon}^j$ upon inserting $\tilde{\vec{u}}^j = \vec{u}^j + \vec{\epsilon}^j$ into the matrix version of the FDE (i.e. (12.59)).

For stability we require that the spectral radius of the update matrix does not exceed unity, i.e. , $\rho(\mathbf{T}) \leq 1$. Thus all eigenvalues λ_i of the update matrix must satisfy

$$|\lambda_i| \leq 1 \qquad (\text{all } i).$$

Again, the bounds of the eigenvalues can be assessed by Gerschgorin's theorem

$$|\lambda_i - T_{ii}| \leq \sum_{j \neq i} |T_{ij}|.$$

Example – For the upwind method (and periodic BCs) all rows of the matrix are 'similar' so

$$|\lambda - (1 - a)| \leq a,$$

where a is a positive real value. So λ lies within (or on the edge of) a disk of radius a in the complex plane, with this disk centred at $z_o = (1 - a)$, i.e., on the real axis. The 'right edge' of this disk is always at $z_{right} = +1$. The 'left edge' lies at $z_{left} = 1 - 2a$. For $a > 1$ we have $z_{left} < -1$ and so $|\lambda| > 1$ (i.e. instability) is possible. (Note that all other points on the circle edge break-out beyond unit distance from the origin when this happens, except the point exactly on the +ve real axis.) Hence $a \leq 1$ is required for $|\lambda| \leq 1$ and therefore stability.

The advantage of the matrix method over Von-Neumann's method is its ability to deal with (i) BCs other than periodic, (ii) cases where the phase speed changes across the grid.

12.3.4 FTCS method – (one not to use!)

This illustrates one sort of pitfall that exists when trying to come up with FD schemes for PDEs. To increase the accuracy of the spatial derivative, centred differencing might seem like a good idea;

$$\left.\frac{\partial u}{\partial x}\right|_i^j = \frac{u_{i+1}^j - u_{i-1}^j}{2h} \qquad \{\, \mathcal{O}(h^2) \,\}.$$

This leads to the innocuous looking FDE (for any v_x, +ve and -ve)

$$u_i^{j+1} = u_i^j - \frac{v_x \Delta t}{2h}\left(u_{i+1}^j - u_{i-1}^j\right), \tag{12.66}$$

known as the 'Forward Time Centred Space' (FTCS) method. It is a consistent, $\mathcal{O}(\Delta t)+\mathcal{O}(\Delta h^2)$ FD scheme. It may surprise you that this is **unconditionally unstable!** A stability analysis (by either method) yields the following impossible condition for stability $|v_x|\Delta t/(2h) \leq 0$. (Impossible for positive Δt and h.)

12.3.5 Lax method

This illustrates a second sort of pitfall that exists when trying to come up with FD schemes for PDEs; inadvertently coming up with an **inconsistent scheme**. The Lax method takes FTCS and substitutes a spatial average for the current point,

$$u_i^j = \frac{1}{2}\left(u_{i+1}^j + u_{i-1}^j\right),$$

giving the FDE

$$u_i^{j+1} = \frac{1}{2}\left(u_{i+1}^j + u_{i-1}^j\right) - \frac{v_x\,\Delta t}{2h}\left(u_{i+1}^j - u_{i-1}^j\right), \tag{12.67}$$

which turns out to have the usual CFL stability condition, $0 \le a \le 1$ (Remember $a = |v_x|\Delta t/h$.) However the MDE can be shown to be

$$\frac{\partial u}{\partial t} + v_x\frac{\partial u}{\partial x} = -\frac{1}{2}\frac{\partial^2 u}{\partial t^2}\,\Delta t - \frac{1}{6}\frac{\partial^3 u}{\partial x^3}\,v_x h^2 + \frac{1}{2}\frac{\partial^2 u}{\partial x^2}\,\frac{h^2}{\Delta t} + \mathcal{O}(\Delta t^2) + \mathcal{O}(h^2). \tag{12.68}$$

The third term on the RHS will not vanish as $h \to 0$ and $\Delta t \to 0$ arbitrarily. For it to vanish, h and Δt must be constrained as $\Delta t \propto h^p$ with $p < 2$. For $p > 2$ this term will diverge (swamping the intended PDE!). In practice this method will work, but suffers from numerical diffusion.

12.3.6 Lax-Wendroff

This is FTCS with a little extra diffusion added

$$\boxed{u_i^{j+1} = u_i^j - \frac{v_x\Delta t}{2h}\left(u_{i+1}^j - u_{i-1}^j\right) + \frac{(v_x\Delta t)^2}{2h^2}\left(u_{i+1}^j - 2u_i^j + u_{i-1}^j\right),} \tag{12.69}$$

which can be shown to be a consistent, $\mathcal{O}(\Delta t^2)+\mathcal{O}(\Delta h^2)$ scheme. It has the usual CFL condition $0 \le a \le 1$. Interestingly, adding in just the right amount of the FD diffusion operator makes this scheme higher order in Δt as well as stabilising the method. Also Lax-Wendroff doesn't have two different versions of the FDE, in contrast to the one-sided upwind method. The improved accuracy of the Lax-Wendroff method is demonstrated in figure 12.5.

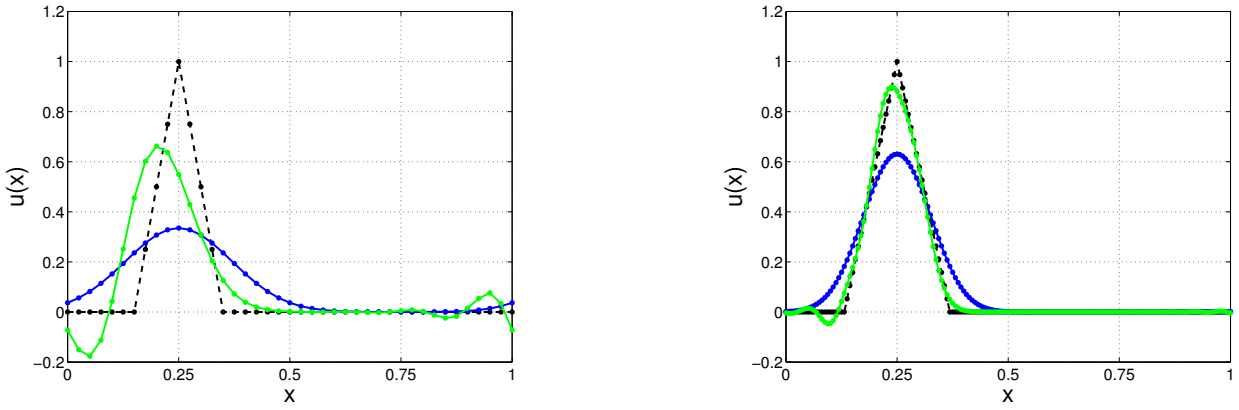

Figure 12.5: **Advection via the Lax-Wendroff method.** The same problem as in figure 12.4 is used. The Lax-Wendroff (green curves) and upwind method (blue curves) are compared after one transit time ($t = L/v_x = 1$) for advection number $a = 0.5$. (Left) $h = 1/40$. (Right) $h = 1/160$. Lax-Wendroff shows less numerical diffusion.

12.3.7 Coupled equations

The Lax-Wendroff and Lax methods can be applied to the coupled 1st-order PDEs seen earlier, e.g., equations (12.48) for ρ (mass density) and u (fluid velocity). Using the 'flux operator' approach and choosing the simpler Lax scheme to illustrate;

$$\vec{U}_i^{j+1} = \frac{1}{2}\left(\vec{U}_{i+1}^j + \vec{U}_{i-1}^j\right) - \mathbf{F}\cdot\left[\frac{\Delta t}{2h}\left(\vec{U}_{i+1}^j - \vec{U}_{i-1}^j\right)\right], \tag{12.70}$$

where $\vec{U} = [\rho, u]^T$, which splits into

$$\begin{aligned} \rho_i^{j+1} &= \tfrac{1}{2}\left(\rho_{i+1}^j + \rho_{i-1}^j\right) - \tfrac{\rho_o\,\Delta t}{2h}\left(u_{i+1}^j - u_{i-1}^j\right), \\ u_i^{j+1} &= \tfrac{1}{2}\left(u_{i+1}^j + u_{i-1}^j\right) - \tfrac{c^2\,\Delta t}{2\rho_o\,h}\left(\rho_{i+1}^j - \rho_{i-1}^j\right). \end{aligned} \tag{12.71}$$

12.4 Solving Parabolic PDEs

Consider the diffusion equation in one spatial dimension

$$\frac{\partial u}{\partial t} = D\,\frac{\partial^2 u}{\partial x^2}. \tag{12.72}$$

with D constant.

Explicit method

Explicit methods are ones that obtain the next time-step, i.e., u_i^{j+1} *only* using the solution at the current time step, i.e., u_i^j. Let us choose the first-order, forward difference scheme for the time derivative

$$\left.\frac{\partial u}{\partial t}\right|_i^j = \frac{u_i^{j+1} - u_i^j}{\Delta t} \qquad \{\,\mathcal{O}(\Delta t)\,\},$$

as we did with schemes for hyperbolic PDEs. For the spatial derivative let us choose the simplest approximation, the second order, central difference scheme,

$$\left.\frac{\partial^2 u}{\partial x^2}\right|_i^j = \frac{u_{i-1}^j - 2\,u_i^j + u_{i+1}^j}{h^2} \qquad \{\,\mathcal{O}(h^2)\,\}. \tag{12.73}$$

Inserting these approximations into the diffusion equation (12.72) we get

$$\frac{u_i^{j+1} - u_i^j}{\Delta t} = D\left(\frac{u_{i-1}^j - 2\,u_i^j + u_{i+1}^j}{h^2}\right), \tag{12.74}$$

giving the FDE

$$\boxed{u_i^{j+1} = (1 - 2\,d)\,u_i^j + d\,(u_{i-1}^j + u_{i+1}^j),} \tag{12.75}$$

where

$$d = \frac{D\,\Delta t}{h^2}, \tag{12.76}$$

is the **diffusion number**, a dimensionless parameter. An example of the explicit method for diffusion is shown in figure 12.6.

The FDE (12.75) is a consistent, $\mathcal{O}(\Delta t) + \mathcal{O}(h^2)$ scheme. As usual, these properties can be shown by the MDE method.

Either the Von-Neumann method (section 12.3.2) or the matrix method (section 12.3.3) can be used to show that the condition for the FDE above (12.75) to be stable is

$$\boxed{d = \frac{D\,\Delta t}{h^2} < \frac{1}{2} \qquad \text{or equivalently} \qquad \Delta t < \frac{h^2}{2D}.} \tag{12.77}$$

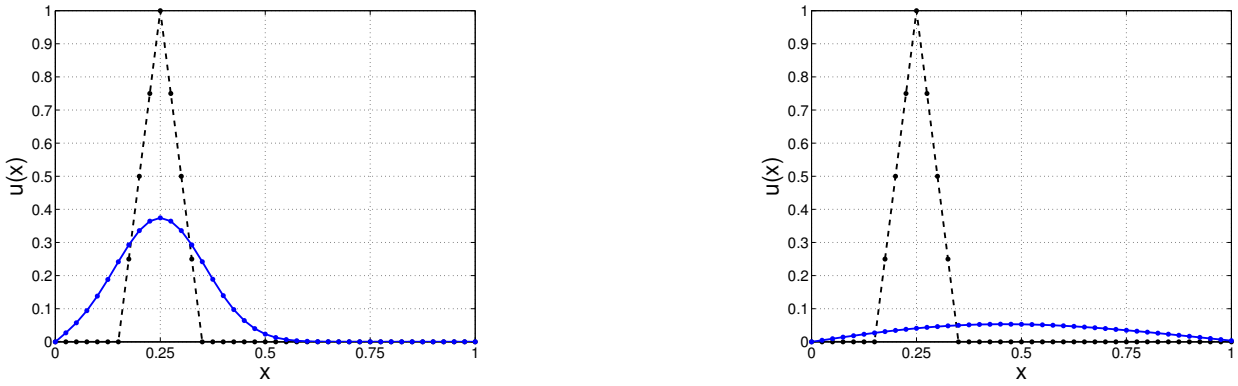

Figure 12.6: **Diffusion via the explicit method.** (Left) $t = 0.005$ and (Right) $t = 0.1$. The initial profile is a triangle (dotted black lines). Fixed (Dirichlet) boundary conditions are used; $u = 0$ on the boundaries. Parameters: $D = 1$, $h = 1/40$, $\Delta t = 0.25\tau_{\text{diff}-\text{grid}}$ (i.e. $d = 0.25$).

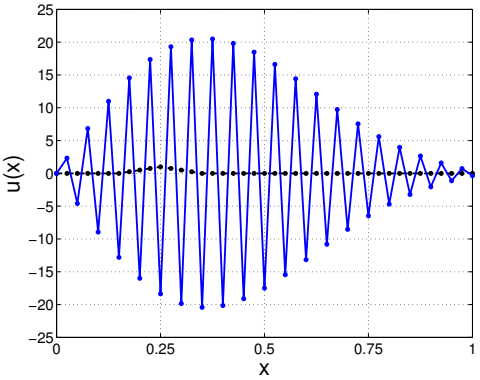

Figure 12.7: **Diffusion via the explicit method.** Numerical instability when the CFL condition is exceeded; shown at $t = 0.1$. Here $d = 0.51$. (The same system, and other parameters, are used as in figure 12.6.)

This bound has a physical implication in that it tells us that the maximum time step is limited to roughly half the **grid diffusion time**

$$\tau_{\text{diff}-\text{grid}} \equiv \frac{h^2}{D}, \tag{12.78}$$

which characterises the time it takes for information to propagate between sites on the grid. Figure 12.7 shows an example of the numerical instability that occurs if the CFL condition is violated.

This is actually quite restrictive. If the spatial scale of the problem is l (e.g. pulse width) then the physical diffusion time scale is $\tau_{_D} \sim l^2/D$. Given the CFL condition above, we have

$$\Delta t \leq \frac{\tau_{_D}}{2}\left(\frac{h}{l}\right)^2.$$

Since $h \ll l$ is required for good spatial resolution, many time steps are needed to observe the dynamics of the system (e.g. diffusive spreading of the pulse).

Crank-Nicolson Method - (Implicit Method)

To improve on this stringent constraint on the time-step and also increase the accuracy of the method, we go to a second order approximation for the time derivative. The Crank-Nicholson method is obtained by inserting a fictitious layer of grid points at $j+\frac{1}{2}$, i.e., half way between the original time steps. The second order, central difference scheme in time for the $j+1/2$ step is

$$\left.\frac{\partial u}{\partial t}\right|_i^{j+1/2} = \frac{u_i^{j+1}-u_i^j}{\Delta t} \qquad \left\{\,\mathcal{O}(\Delta t^2)\,\right\}. \tag{12.79}$$

For the spatial derivative we can take the average of those at times j and $j+1$

$$\left.\frac{\partial^2 u}{\partial x^2}\right|_i^{j+1/2} = \frac{1}{2}\left[\left.\frac{\partial^2 u}{\partial x^2}\right|_i^{j} + \left.\frac{\partial^2 u}{\partial x^2}\right|_i^{j+1}\right]. \tag{12.80}$$

Rearranging all $j+1$ terms onto the LHS we get the system

$$\boxed{(2+2\,d)\,u_i^{j+1} - d\,(u_{i-1}^{j+1}+u_{i+1}^{j+1}) = (2-2\,d)\,u_i^j + d\,\left(u_{i-1}^j+u_{i+1}^j\right),} \tag{12.81}$$

or in matrix form

$$\mathbf{M}\cdot\vec{u}^{\,j+1} = \vec{b}^{\,j}, \tag{12.82}$$

where

$$\mathbf{M} \equiv \begin{pmatrix} (2+2\,d) & -d & 0 & 0 & \cdot & \cdot & \cdot & 0 \\ -d & (2+2\,d) & -d & 0 & \cdot & \cdot & \cdot & 0 \\ 0 & -d & (2+2\,d) & -d & \cdot & \cdot & \cdot & 0 \\ & & & \cdot & & & & \\ & & & \cdot & & & & \\ & & & \cdot & & & & \\ 0 & 0 & \cdot & \cdot & \cdot & 0 & -d & (2+2\,d) \end{pmatrix}, \tag{12.83}$$

and

$$\vec{b}^{\,j} \equiv \begin{pmatrix} (2-2\,d) & d & 0 & 0 & \cdot & \cdot & \cdot & 0 \\ d & (2-2\,d) & d & 0 & \cdot & \cdot & \cdot & 0 \\ 0 & d & (2-2\,d) & d & \cdot & \cdot & \cdot & 0 \\ & & & \cdot & & & & \\ & & & \cdot & & & & \\ & & & \cdot & & & & \\ 0 & 0 & \cdot & \cdot & \cdot & 0 & d & (2-2\,d) \end{pmatrix} \cdot \vec{u}^{\,j} = \mathbf{A} \cdot \vec{u}^{\,j}. \qquad (12.84)$$

where Dirichlet BCs of $u_0^j = u_{n_x}^j = 0$ have been chosen in getting the above forms of $\mathbf{M}$ and $\vec{b}$. Hence the number of unknowns is $m = n_x - 1$ (i.e. $1 \leq i \leq n_x - 1$).

Crank-Nicholson is a consistent, $\mathcal{O}(\Delta t^2) + \mathcal{O}(h^2)$ scheme.

It can be shown that this method is actually stable for all values of d (i.e. any Δt for a given D and h).

To follow the dynamics of the system with acceptable accuracy without using an exceedingly small time step, we can solve equation (12.82) using an iterative matrix solver (Jacobi, Gauss-Seidel or SOR) at each time step.