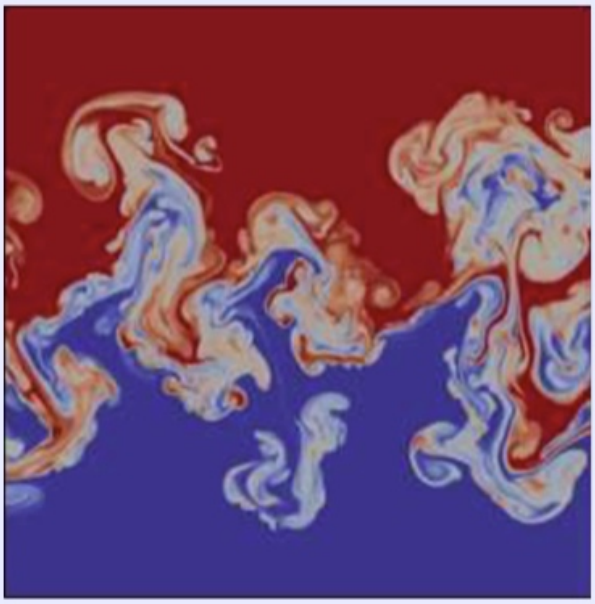
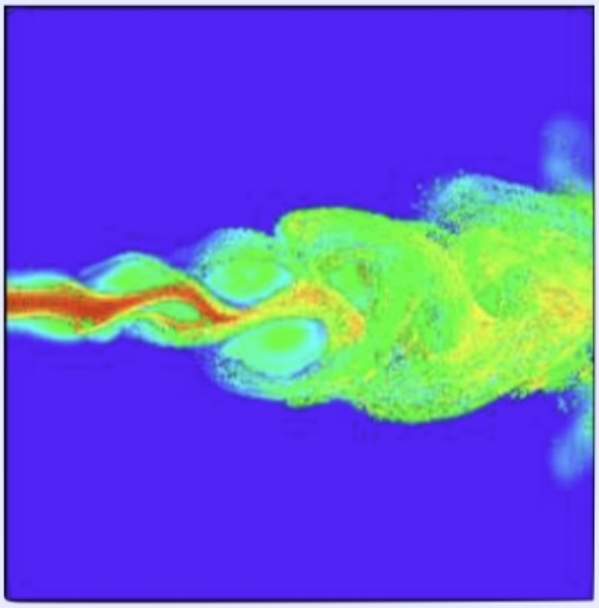
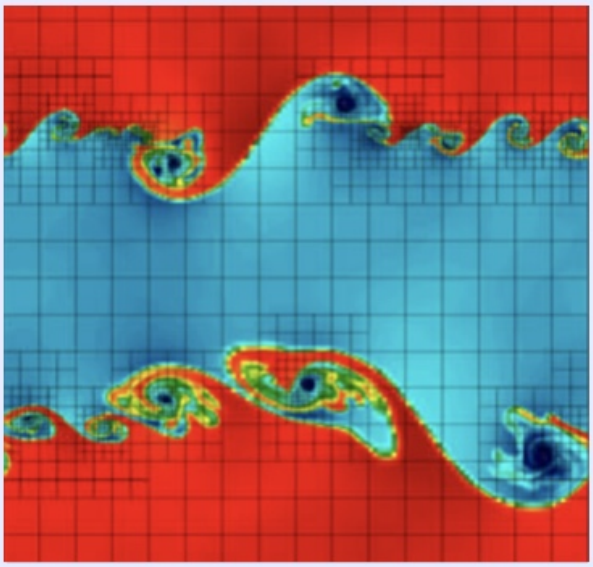
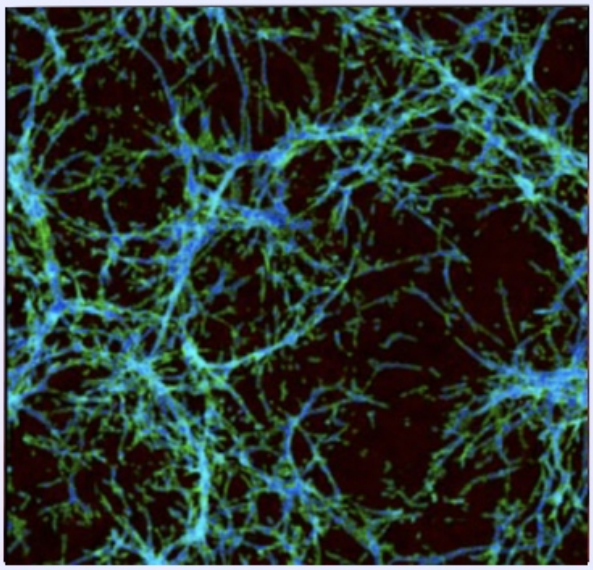
Numerical Hydrodynamics

am introductory treatment of numerically
solving PDEs with applications
to numerical hydrodynamics

Frank C. van den Bosch

These lecture notes are constantly being updated, extended and improved

This is Version F2021:v0.1

Last Modified: Jul 20, 2021

CONTENTS

CHAPTER 1

Solving PDEs with Finite Difference Methods

Having discussed the theory of fluid dynamics, we now focus on how to actually solve the **partial differential equations** (PDEs) that describe the time-evolution of the various hydrodynamical quantities. Solving PDEs analytically is complicated (see Appendix E for some background) and typically only possible for highly idealized flows. In general, we are forced to solve the PDEs numerically. How this is done is the topic of this and subsequent chapters.

The first thing to realize is that a computer will only solve a discrete representation of the actual PDE. We call this approximation a **finite difference approximation** (FDA). In addition, we need to represent the physical data (i.e., our hydrodynamical quantities of interest) in a certain finite physical region of interest, called the **computational domain**). Typically this is done by subdividing the computational domain using a *computational mesh*; the data is specified at a finite number of grid points or cells. Throughout these chapters on numerical hydrodynamics we will consider regular, cartesian meshes, but more complicated, even time-dependent meshes can be adopted as well. There are two different 'methods' (or 'philosophies') for how to formulate the problem. On the one hand, one can think of the data as literally being placed at the grid *points*. This yields a **finite difference formulation**. Alternatively, one can envision the data being spread our over the mesh cell (or 'zone'), yielding a **finite volume formulation**. In practice. finite difference formulations are a bit faster, and easier to comprehend, which is why much of what follows adopts the finite difference formulation. In Chapter 3, though, we will transition to finite volume formulation, which is the formulation most often adopted in modern computational fluid dynamics.

Before proceeding with discussing finite difference formulations of our hydrodynamical equations on a mesh, we point out that **particle based methods** have also been developed. Smoothed Particle Hydrodynamics (SPH) is the main example, which is quite often used in astrophysics (but rarely in engineering problems). SPH has the advantage of being Lagrangian, which is desirable in certain applications, but it is typically much harder to proof that the SPH scheme converges to the actual physical

equations being modelled. In what follows we therefore adhere to Eulerian methods based on computational domains that are discretized on a mesh.

In numerical hydrodynamics it is common and most convenient to write the hydrodynamical equations in **conservative form**

$$\frac{\partial q}{\partial t} + \nabla \cdot \vec{f}(q) = S$$

Here $q(\vec{x}, t)$ is a state variable of the fluid (either scalar or vector), $\vec{f}(q)$ describes the flux of q, and S describes the various sources and/or sinks of q. For an **ideal** (i.e., **inviscid** and **non-conductive**) fluid, *which is what we consider throughout these chapters on numerical hydrodynamics*, the continuity, momentum and energy equations in conservative form are:

$$\begin{aligned}
\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \vec{u}) &= 0 \\
\frac{\partial \rho \vec{u}}{\partial t} + \nabla \cdot (\rho \vec{u} \otimes \vec{u} + P) &= -\rho \nabla \Phi \\
\frac{\partial \rho(\frac{1}{2}u^2 + \Phi + \varepsilon)}{\partial t} + \nabla \cdot \left[\left(\frac{1}{2}u^2 + \Phi + \varepsilon + \frac{P}{\rho} \right) \rho \vec{u} \right] &= \rho \frac{\partial \Phi}{\partial t} - \mathcal{L}
\end{aligned}$$

The latter, the energy equation, can be rewritten as

$$\frac{\partial \rho(\frac{1}{2}u^2 + \varepsilon)}{\partial t} + \nabla \cdot \left[\left(\frac{1}{2}u^2 + \varepsilon + \frac{P}{\rho} \right) \rho \vec{u} \right] = -\rho \vec{u} \cdot \nabla \Phi - \mathcal{L}$$

In order to see this, use that

$$\begin{aligned}
\rho \frac{\partial \Phi}{\partial t} - \frac{\partial \rho \Phi}{\partial t} - \frac{\partial \rho \Phi u_k}{\partial x_k} &= -\Phi \frac{\partial \rho}{\partial t} - \nabla \cdot (\rho \Phi \vec{u}) \\
&= -\Phi \frac{\partial \rho}{\partial t} - \Phi \nabla \cdot \rho \vec{u} - \rho \vec{u} \cdot \nabla \Phi \\
&= -\Phi \left[\frac{\partial \rho}{\partial t} + \nabla \cdot \rho \vec{u} \right] - \rho \vec{u} \cdot \nabla \Phi = -\rho \vec{u} \cdot \nabla \Phi
\end{aligned}$$

Thus, ignoring radiation ($\mathcal{L} = 0$) and gravity ($\Phi = 0$), in addition to viscosity and conduction, the equations of hydrodynamics in conservative form are

$$\frac{\partial \vec{q}}{\partial t} + \nabla \cdot \vec{f} = 0 \qquad \text{or,} \qquad \frac{\partial q_i}{\partial t} + \frac{\partial f_i}{\partial q_i} = 0$$

where

$$\vec{q} = \begin{pmatrix} \rho \\ \rho\vec{u} \\ \frac{1}{2}\rho u^2 + \rho\varepsilon \end{pmatrix} \qquad \text{and} \qquad \vec{f}(\vec{q}) = \begin{pmatrix} \rho\vec{u} \\ \rho\vec{u} \otimes \vec{u} + P \\ (\frac{1}{2}u^2 + \varepsilon + P/\rho)\,\rho\vec{u} \end{pmatrix}$$

Depending on the number of spatial dimensions, this is a set of 3-5 coupled PDEs, and our goal is to come up with a scheme how to numerically solve these. Note that we are considering a highly oversimplified case, ignoring gravity, radiation, viscosity and conduction. As discussed later, adding viscosity and conduction changes the character of the PDE, while gravity and radiation can be added as source and/or sink terms, something we will not cover in these lecture notes.

Typically, one can numerically solve this set of PDEs using the following procedure:

1. Define a spatial grid, $\vec{x}_i$, where here the index refers to the grid point.
2. Specify initial conditions for $\vec{q}(\vec{x}, t)$ at $t = 0$ at all $\vec{x}_i$, and specify suitable boundary conditions on the finite computational domain used.
3. Compute ρ, $\vec{u}$ and ε at all $\vec{x}_i$.
4. Compute the fluxes $\vec{f}(\vec{q})$ at all $\vec{x}_i$.
5. Take a small step in time, Δt, and compute the new $\vec{q}(\vec{x}_i, t + \Delta t)$.
6. Go back to [3]

Although this sounds simple, there are many things that can go wrong. By far the most tricky part is step [5], as we will illustrate in what follows.

Let us start with some basics. The above *set* of PDEs is **hyperbolic**. Mathematically this means the following. Consider the Jacobian matrix of the flux function

$$F_{ij}(\vec{q}) \equiv \frac{\partial f_i}{\partial q_j}$$

The set of PDEs is hyperbolic if, for each value of $\vec{q}$, *all* the eigenvalues of $\mathbf{F}$ are **real**, and $\mathbf{F}$ is **diagonizable**.

In a system described by a hyperbolic PDE, information travels at a finite speed (c_s in the example here). Information is not transmitted until the wave arrives. The smoothness of the solution to a hyperbolic PDE depends on the smoothness of the initial and boundary conditions. For instance, if there is a jump in the data at the start or at the boundaries, then the jump will propagate as a shock in the solution. If, in addition, the PDE is nonlinear (which is the case for the Euler equations), then shocks may develop even though the initial conditions and the boundary conditions are smooth.

If we were to consider the **Navier-Stokes equation**, rather than the Euler equations, then we add 'non-ideal' terms (referring to the fact that these terms are absent for an ideal fluid) related to **viscosity** and **conduction**. These terms come with a second-order spatial derivative, describing diffusive processes. Such terms are called **parabolic**, and they have a different impact on the PDE. See the box on the next page.

Hence, we are faced with the problem of simultaneously solving a **hyperbolic** set of 3-5 PDEs, some of which are **non-linear**, and some of which may contain **parabolic** terms. This is a formidable problem, and one could easily devote an entire course to it. Readers who want to get more indepth information are referred to the textbooks listed at the start of this section on numerical hydrodynamics.

The Nature of Hyperbolic and Parabolic PDEs

Consider the first-order PDE

$$\frac{\partial \rho}{\partial t} + v\frac{\partial \rho}{\partial x} = 0$$

As we will see later on, this is the 1D linear advection equation, and it describes how density features are being propagated (i.e., advected) with constant speed v. To get some insight, let's consider the formal solution

$$\rho = \rho_0 + \rho_1 \mathrm{e}^{i(kx-\omega t)}$$

which consists of a constant part, ρ_0, plus a wave-like perturbation. Substituting this in the PDE yields

$$\omega = kv \qquad \Rightarrow \qquad \rho(x,t) = \rho_0 + \rho_1 \mathrm{e}^{ik(x-vt)}$$

This is a travelling wave, and ω is always real. We also see that the group velocity $\partial\omega/\partial k = v$ is constant, and independent of k; all modes propagate at the same speed, and the wave-solution is thus non-dispersive. This is characteristic of a **hyperbolic PDE**. Next consider the **heat equation**

$$\frac{\partial T}{\partial t} = \kappa \frac{\partial^2 T}{\partial x^2}$$

This equation describes how heat diffuses in a 1D system with diffusion coefficient κ. Substituting the same formal solution as above, we obtain that

$$\omega = -i\kappa k^2 \qquad \Rightarrow \qquad \rho(x,t) = \rho_0 + \rho_1 \mathrm{e}^{ikx}\mathrm{e}^{-\kappa k^2 t}$$

Note that ω is now imaginary, and that the solution has an exponentially decaying term. This describes how the perturbation will 'die out' over time due to dissipation. This is characteristic of a **parabolic PDE**.

In what follows, rather than tackling the problem of numerically solving the Euler or Navier-Stokes equations, we focus on a few simple limiting cases that result in fewer equations, which will give valuable insight to aspects of the various numerical schemes. In particular, we restrict ourselves to the **1D** case, and continue to assume an **ideal** fluid for which we can ignore **radiation** and **gravity**. The hydrodynamic equations now reduce to the following set of 3 PDEs:

$$\begin{aligned} \frac{\partial \rho}{\partial t} + \frac{\partial}{\partial x}(\rho u) &= 0 \\ \frac{\partial}{\partial t}(\rho u) + \frac{\partial}{\partial x}(\rho u u + P) &= 0 \\ \frac{\partial}{\partial t}\left(\frac{1}{2}\rho u^2 + \rho\varepsilon\right) + \frac{\partial}{\partial x}\left(\left[\frac{1}{2}\rho u^2 + \rho\varepsilon + P\right] u\right) &= 0 \end{aligned}$$

If we now, in addition, assume **constant pressure**, P, and a **constant flow velocity**, u, then this system reduces to 2 separate, linear PDEs given by

$$\frac{\partial \rho}{\partial t} + u\frac{\partial \rho}{\partial x} = 0 \qquad \text{and} \qquad \frac{\partial \varepsilon}{\partial t} + u\frac{\partial \varepsilon}{\partial x} = 0$$

These are identical equations, known as the **linear advection equation**. They describe the **passive transport** of the quantities ρ and ε in the flow with constant velocity u. This equation has a well-known, rather trivial solution: If the initial ($t = 0$) conditions are given by $\rho_0(x)$ and $\varepsilon_0(x)$, the the general solutions are

$$\rho(x,t) = \rho_0(x - ut) \qquad \text{and} \qquad \varepsilon(x,t) = \varepsilon_0(x - ut)$$

Since the analytical solution to this linear advection equation is (trivially) known, it is an ideal equation on which to test our numerical scheme(s). As we will see, even this extremely simple example will proof to be surprisingly difficult to solve numerically.

The Courant-Friedrich-Lewy condition: Due to the finite travelling speed of waves, hyperbolic PDEs have a **finite physical domain of dependence**. This implies that any scheme used to solve a hyperbolic PDE must obey the Courant-Friedrich-Lewy (CFL) condition (often called the Courant condition, for short). In particular, if u is the speed at which information propagates, and grid cells are interspaced by Δx, then the time-step Δt used in the numerical integration must obey

$$\boxed{|u\,\Delta t/\Delta x| \equiv |\alpha_{\rm c}| \leq 1}$$

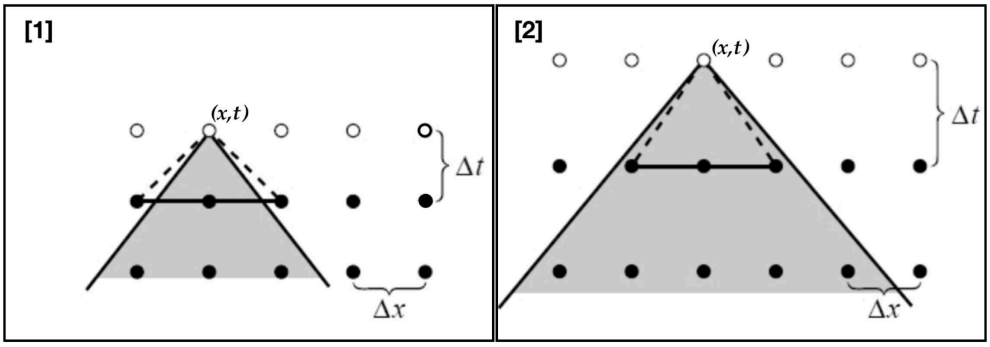


Figure 1: *Filled and open dots indicate the grid points (horizontal) and time steps (vertical). The grey region is the physical domain of dependence for the* (x,t)*-grid point, and the CFL-condition states that the numerical domain of dependence must contain this physical domain. The triangle formed by the two dashed lines and solid black line indicates this numerical domain in the case of the* **explicit Euler scheme** *of integration discussed below. It relies on the properties of the neighboring grid points at the previous time-step. The time-step in panel [1] DOES meet the CFL-criterion, while that in panel [2] does NOT.*

Here the parameter α_c is often called the CFL, or Courant, parameter. Note that this CFL condition is *necessary* for stability, but not *sufficient*. In other words, obeying the CFL condition does not guarantee stability, as we will see shortly.

The principle behind the CFL condition is simple: if a wave is moving across a discrete spatial grid and we want to compute its amplitude at discrete time steps of equal duration, Δt, then this duration must be less than the time for the wave to travel to adjacent grid points. As a corollary, when the grid point separation is reduced, the upper limit for the time step also decreases. In essence, *the numerical domain of dependence of any point in space and time must include the analytical domain of dependence* (wherein the initial conditions have an effect on the exact value of the solution at that point) to assure that the scheme can access the information required to form the solution. This is illustrated in Fig. 25.

IMPORTANT: change in notation: From here on out, following standard notation in many textbooks on this topic, we are going to write the function to be solved as $u(x,t)$, without implying that this u is velocity (as has been our notation thus far). Although this may cause some confusion, it is in line with the standard notation

in computational hydrodynamics and much of the literature on solving differential equations. In this new notation our linear advection equation is given by

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

where v is now the constant advection speed, and u is the property being advected. Throughout we adopt a discretization in time and space given by

$$\begin{aligned} t^n &= t_0 + n\Delta t \\ x_i &= x_0 + i\Delta x \end{aligned}$$

Note that subscripts indicate the spatial index, while superscripts are used to refer to the temporal index. Hence, u_i^{n+1} refers to the value of u at grid location i at time-step $n+1$, etc. The key to numerically solving differential equations is find how to express derivatives in terms of the discretized quantities. This requires a **finite difference scheme**. Using Taylor series expansion, we have that

$$\begin{aligned} u_{i+1} &\equiv u(x_i + \Delta x) = u(x_i) + \Delta x \frac{\partial u}{\partial x}(x_i) + \frac{(\Delta x)^2}{2}\frac{\partial^2 u}{\partial x^2}(x_i) + \frac{(\Delta x)^3}{6}\frac{\partial^3 u}{\partial x^3}(x_i) + \mathcal{O}(\Delta x^4) \\ u_i &\equiv u(x_i) \\ u_{i-1} &\equiv u(x_i - \Delta x) = u(x_i) - \Delta x \frac{\partial u}{\partial x}(x_i) + \frac{(\Delta x)^2}{2}\frac{\partial^2 u}{\partial x^2}(x_i) - \frac{(\Delta x)^3}{6}\frac{\partial^3 u}{\partial x^3}(x_i) + \mathcal{O}(\Delta x^4) \end{aligned}$$

By subtracting the first two expression, and dividing by Δx, we obtain the following finite difference approximation for the first derivative

$$\boxed{u_i' \approx \frac{u_{i+1} - u_i}{\Delta x} - \frac{1}{2}\frac{\partial^2 u}{\partial x^2}(x_i)\Delta x}$$

The first term is the finite difference approximation (FDA) for the first derivative, and is known as the **forward difference**. The second term gives the **truncation error**, which shows that this FDA is first-order accurate (in Δx).

We can obtain an alternative FDA for the first derivative by subtracting the latter two expressions, and again dividing by Δx:

$$\boxed{u_i' \approx \frac{u_i - u_{i-1}}{\Delta x} - \frac{1}{2}\frac{\partial^2 u}{\partial x^2}(x_i)\Delta x}$$

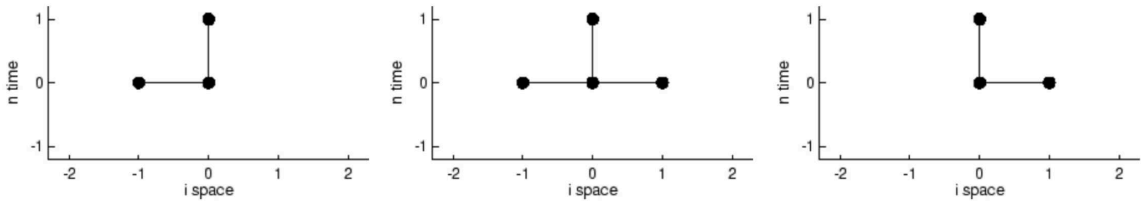


Figure 2: *Stencil diagrams for the Backward-Space FTBS scheme (left-hand panel), the Central-Space FTCS scheme (middle panel), and the Forward-Space FTFS scheme (right-hand panel). All of these are examples of explicite Euler integration schemes.*

which is also first-order accurate in Δx (i.e., the numerical error in u' is proportional to Δx). This finite difference scheme is known as **backward difference**.

Combining the two Taylor series approximations, and subtracting one from the other, yields yet another FDA for the first derivative, given by

$$\boxed{u'_i \approx \frac{u_{i+1} - u_{i-1}}{2\Delta x} - \frac{1}{3}\frac{\partial^3 u}{\partial x^3}(x_i)(\Delta x)^2}$$

which is known as the **centred difference** scheme. Note that this FDA is second-order accurate in Δx.

Using the same approach, one can also obtain FDAs for higher-order derivatives. For example, by *adding* the two Taylor series expressions above, we find that

$$\boxed{u''_i \approx \frac{u_{i+1} - 2u_i + u_{i-1}}{(\Delta x)^2} - \frac{1}{12}\frac{\partial^4 u}{\partial x^4}(x_i)(\Delta x)^2}$$

Similarly, by folding in Taylor series expressions for u_{i+2} and u_{i-2}, one can obtain higher-order FDAs. For example, the first derivative can then be written as

$$\boxed{u'_i \approx \frac{-u_{i+2} + 8u_{i+1} - 8u_{i-1} + u_{i-2}}{12\Delta x}}$$

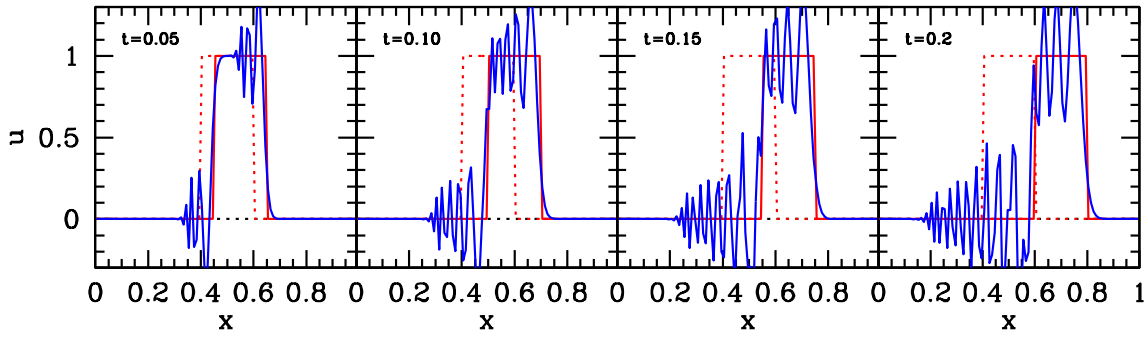


Figure 3: *The results of using the FTCS scheme to propate the initial conditions, indicated by the red dotted lines, using the 1D linear advection equation. Despite the fact that* $\alpha_c = v\Delta t/\Delta x = 0.1$*, thus obeying the CFL-condition, the numerically solution (in blue) develops growing oscillations, a manifestation of its inherent unstable nature. The red solid lines in each panel show the corresponding analytical solutions. These results are based on a linear spatial grid using 100 spatial cells over the domain* $[0, 1]$*, with a time step* $\Delta t = 0.001$*. The advection velocity is* $v = 1.0$*.*

which is fourth-order accurate in Δx. Typically, higher-order is better (if stable, see next chapter), but it also is computationally more expensive (slower).

Now let us return to our linear advection equation. Since we only know $u(x, t)$ in the past, but not in the future, we have to use the backward difference scheme to approximate $\partial\rho/\partial t$. For the spatial derivative it seems natural to pick the centred difference scheme, which is higher order than the forward or backward difference schemes. Hence, we have that

$$\frac{\partial u}{\partial t} \to \frac{u_i^{n+1} - u_i^n}{\Delta t} \qquad \frac{\partial u}{\partial x} \to \frac{u_{i+1}^n - u_{i-1}^n}{2\Delta x}$$

which allows us to write

$$\boxed{u_i^{n+1} = u_i^n - v\,\frac{\Delta t}{2\Delta x}\,\left(u_{i+1}^n - u_{i-1}^n\right)}$$

This scheme for solving the linear advection equation numerically is called the **explicit Euler scheme** or **FTCS-scheme** for **F**orward-**T**ime-**C**entral-**S**pace.

We can define similar explicit schemes based on the forward and backward difference

schemes. In particular, we have the **FTBS** (Forward-Time-Backward-Space) scheme

$$\boxed{u_i^{n+1} = u_i^n - v\,\frac{\Delta t}{\Delta x}\,\left(u_i^n - u_{i-1}^n\right)}$$

and the **FTFS** (Forward-Time-Forward-Space) scheme

$$\boxed{u_i^{n+1} = u_i^n - v\,\frac{\Delta t}{\Delta x}\,\left(u_{i+1}^n - u_i^n\right)}$$

It is useful to illustrate the dependencies in these schemes using so-called **stencil diagrams**. These are depicted for the FTBS, FTCS and FTFS schemes in Fig. 26.

Figure 27 shows the results of the FTCS-scheme applied to a simple initial condition in which $u = 1$ for $0.4 \leq x \leq 0.6$ and zero otherwise (red, dotted lines). These conditions are advected with a constant, uniform velocity $v = 1.0$. The blue curves show the results obtained using the FTCS scheme with $\Delta x = 0.01$ (i.e., the domain $[0, 1]$ is discretized using 100 spatial cells) and $\Delta t = 0.001$. Results are shown after 50, 100, 150 and 200 time-steps, as indicated. The solid, red curves in the four panels show the corresponding analytical solution, which simply correspond to a horizontal displacement of the initial conditions.

Despite the fact that the CFL-condition is easily met ($|v\,\Delta t/\Delta x = 0.1$), the solution develops large oscillations that grow with time, rendering this scheme useless.

Let's now try another scheme. Let's pick the **FTBS** scheme, whose stencil is given by the left-hand panel of Fig. 26. The results are shown in Fig. 28. Surprisingly, this scheme, which is also known as the **upwind** or **donor-cell** scheme yields very different solutions. The solutions are smooth (no growing oscillations), but they are substantial 'smeared out', as if **diffusion** is present.

For completeness, Fig. 29 shows the same results but for the **FTFS** scheme (stencil shown in right-hand panel of Fig. 26). This scheme is even more unstable as the FTCS scheme, with huge oscillations developing rapidly.

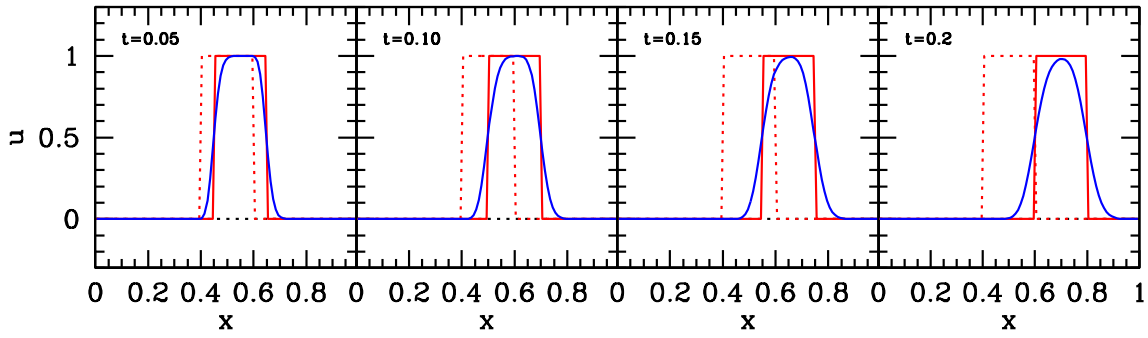


Figure 4: *Same as Figure 27, but for the FCBS scheme (stencil in left-hand panel of Fig. 26). This scheme is stable, yielding smooth solutions, but it suffers from significant numerical diffusion.*

Before we delve into other integration schemes, and an indepth analysis of why subtly different schemes perform so dramatically differently, we first take a closer look at the 1D **conservation equation**

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

where $f = f(u)$ is the flux associated with u (which, in the case of linear advection is give by $f = vu$). We can write this equation in **integral form** as

$$\int_{x_{i-\frac{1}{2}}}^{x_{i+\frac{1}{2}}} \mathrm{d}x \int_{t^n}^{t^{n+1}} \mathrm{d}t \left[\frac{\partial u}{\partial t} + \frac{\partial f}{\partial x}\right] = 0$$

where the integration limits are the boundaries of cell i, which we denote by $x_{i-\frac{1}{2}}$ and $x_{i+\frac{1}{2}}$, and the boundaries of the time step $t^n \rightarrow t^{n+1}$. If we now consider u as being constant over a cell, and the flux is assumed constant during a time step, we can write this as

$$\int_{x_{i-\frac{1}{2}}}^{x_{i+\frac{1}{2}}} \mathrm{d}x \left[u(x, t^{n+1}) - u(x, t^n)\right] + \int_{t^n}^{t^{n+1}} \mathrm{d}t \left[f(x_{i+\frac{1}{2}}, t) - f(x_{i-\frac{1}{2}}, t)\right] =$$

$$u(x_i, t^{n+1})\Delta x - u(x_i, t^n)\Delta x + f_{i+\frac{1}{2}}^{n+\frac{1}{2}}\Delta t - f_{i-\frac{1}{2}}^{n+\frac{1}{2}}\Delta t = 0$$

This yields the following **update formula in conservation form:**

$$\boxed{u_i^{n+1} = u_i^n - \frac{\Delta t}{\Delta x}\left(f_{i+\frac{1}{2}}^n - f_{i-\frac{1}{2}}^n\right)}$$

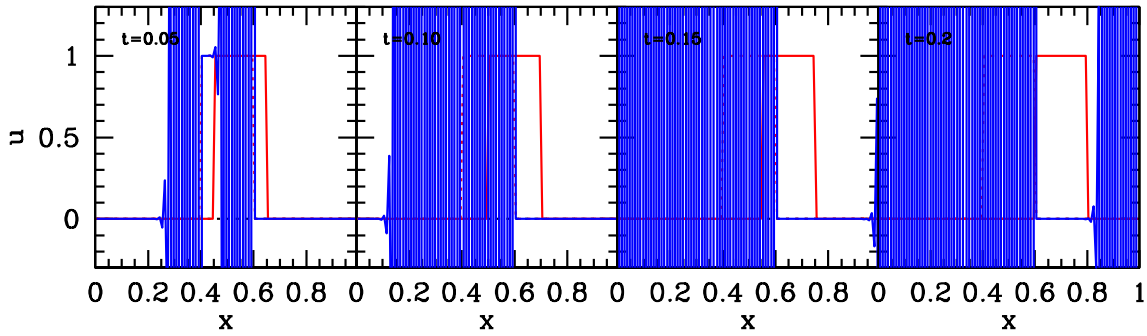


Figure 5: *Same as Figure 27, but for the FCFS scheme (stencil in right-hand panel of Fig. 26). Clearly, this scheme is utterly unstable and completely useless.*

This is called 'conservation form' because it expresses that property u in cell i only changes due to a flux of u through its boundaries. With this formulation, we can describe any integration scheme by simply specifying the flux $f^n_{i+\frac{1}{2}}$. For example, the three integration schemes discussed thus far are specified by $f^n_{i+\frac{1}{2}} = f(u^n_i)$ (FTBS), $f^n_{i+\frac{1}{2}} = \frac{1}{2}[f(u^n_{i+1}) + f(u^n_i)]$ (FTCS), and $f^n_{i+\frac{1}{2}} = f(u^n_{i+1})$ (FTFS). See also the Table on the next page.

So far we have considered three integration schemes, which are all examples of explicit Euler schemes. When a direct computation of the dependent variables can be made in terms of known quantities, the computation is said to be **explicit**. In contrast, when the dependent variables are defined by coupled sets of equations, and either a matrix or iterative technique is needed to obtain the solution, the numerical method is said to be **implicit**. An example of an implicit scheme is the following FDA of the heat equation (see Problem Set 3)

$$\frac{u^{n+1}_i - u^n_i}{\Delta t} = \kappa \left(\frac{u^{n+1}_{i+1} - 2u^{n+1}_i + u^{n+1}_{i-1}}{\Delta x^2} \right)$$

Note that the second-order spatial derivative on the rhs is evaluated at time t^{n+1}, rather than t^n, which is what makes this scheme implicit. Since explicit schemes are much easier to code up, we will not consider any implicit schemes in these lecture notes. We emphasize, though, the sometimes implicit schemes are powerful alternatives.

Neither of the three (explicit) schemes considered thus far are satisfactory; the FTCS

and FTFS schemes are unstable, yielding oscillations that grow rapidly, while the FTBS scheme suffers from a large amount of **numerical dissipation**. Fortunately, there are many alternative schemes, both explicit and implicit. The Table on the next page lists 7 explicit finite difference methods that have been suggested in the literature. In addition to the three Euler methods discussed above, this includes the **Lax-Friedrichs** method, which is basically a FTCS-method with an added **artificial viscosity** term. Similar to the FTCS method, it is second order accurate in space, and first-order accurate in time. The performance of this method is evident from the upper-right panel in Fig. 30. Clearly, the artificial viscosity suppresses the onset of growing oscillations, which is good, but the numerical diffusion is much worse than in the FCBS upwind (or donor-cell) scheme, rendering this method not very useful, except when the initial conditions are very smooth.

All the finite-difference methods encountered thus far are first-order accurate in time. The table also lists three schemes that are **second-order** accurate in both space and time, i.e., with an error that is $\mathcal{O}(\Delta x^2, \Delta t^2)$. The first of these is the **Lax-Wendroff** method. As is evident from the lower-left panel of Fig. 30 it results in some oscillations, but these don't grow much beyond a certain point (unlike, for example, the first-order Euler-FTCS scheme shown in the upper-left panel). Another, similar scheme, is the **Beam-Warming** method, whose performance, shown in the lower-middle panel of Fig 30, is only marginally better. Finally, the lower-right panel shows the performace of the **Fromm** method, which is basically the average of the Lax-Wendroff and Beam-Warming schemes, i.e., $f^n_{i+\frac{1}{2},\mathrm{Fromm}} = \frac{1}{2}[f^n_{i+\frac{1}{2},\mathrm{LW}} + f^n_{i+\frac{1}{2},\mathrm{BW}}]$. As is evident, this is clearly the most successfull method encountered thus far.

Explicit Finite Difference Methods for 1D Linear Advection Problem

Euler FTBS	$f^n_{i+\frac{1}{2}} = f(u^n_i)$ $u^{n+1}_i = u^n_i - \alpha_{\rm c}[u^n_i - u^n_{i-1}]$
Euler FTCS	$f^n_{i+\frac{1}{2}} = \frac{1}{2}[f(u^n_{i+1}) + f(u^n_i)]$ $u^{n+1}_i = u^n_i - \frac{\alpha_{\rm c}}{2}[u^n_{i+1} - u^n_{i-1}]$
Euler FTFS	$f^n_{i+\frac{1}{2}} = f(u^n_{i+1})$ $u^{n+1}_i = u^n_i - \alpha_{\rm c}[u^n_{i+1} - u^n_i]$
Lax-Friedrichs	$f^n_{i+\frac{1}{2}} = \frac{1}{2}[f(u^n_{i+1}) + f(u^n_i)] - \frac{1}{2}\frac{\Delta x}{\Delta t}[u^n_{i+1} - u^n_i]$ $u^{n+1}_i = u^n_i - \frac{\alpha_{\rm c}}{2}[u^n_{i+1} - u^n_{i-1}] + \frac{1}{2}[u^n_{i+1} - 2u^n_i + u^n_{i-1}]$
Lax-Wendroff	$f^n_{i+\frac{1}{2}} = \frac{1}{2}[f(u^n_{i+1}) + f(u^n_i)] - \frac{v^2}{2}\frac{\Delta t}{\Delta x}[u^n_{i+1} - u^n_i]$ $u^{n+1}_i = u^n_i - \frac{\alpha_{\rm c}}{2}[u^n_{i+1} - u^n_{i-1}] + \frac{\alpha^2_{\rm c}}{2}[u^n_{i+1} - 2u^n_i + u^n_{i-1}]$
Beam-Warming	$f^n_{i+\frac{1}{2}} = \frac{1}{2}[3f(u^n_i) - f(u^n_{i-1})] - \frac{v^2}{2}\frac{\Delta t}{\Delta x}[u^n_i - u^n_{i-1}]$ $u^{n+1}_i = u^n_i - \frac{\alpha_{\rm c}}{2}[3u^n_i - 4u^n_{i-1} + u^n_{i-2}] + \frac{\alpha^2_{\rm c}}{2}[u^n_i - 2u^n_{i-1} + u^n_{i-2}]$
Fromm	$f^n_{i+\frac{1}{2}} = \frac{1}{4}[f(u^n_{i+1}) + 4f(u^n_i) - f(u^n_{i-1})] - \frac{v^2}{4}\frac{\Delta t}{\Delta x}[u^n_{i+1} - u^n_i - u^n_{i-1} + u^n_{i-2}]$ $u^{n+1}_i = u^n_i - \frac{\alpha_{\rm c}}{4}[u^n_{i+1} + 3u^n_i - 5u^n_{i-1} + u^n_{i-2}] + \frac{\alpha^2_{\rm c}}{4}[u^n_{i+1} - u^n_i - u^n_{i-1} + u^n_{i-2}]$

Table listing all the explicit integration schemes discussed in the text. For each entry the first line indicates the flux, while the second line indicate the conservative update formula for the 1D linear advection equation, for which $f(u) = vu$, with v the constant advection speed. The parameter $\alpha_{\rm c}$ is the Courant (or CFL) parameter given by $\alpha_{\rm c} = v\,\Delta t/\Delta x$.

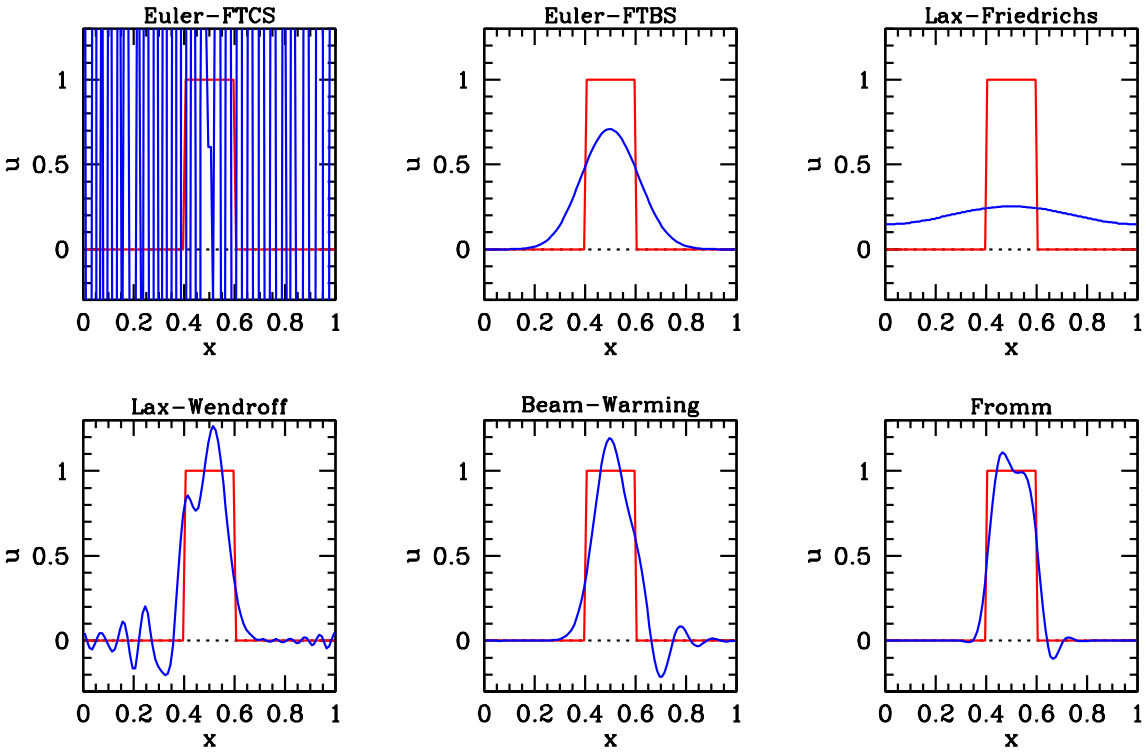


Figure 6: *The result of using 6 different explicit integration schemes, as indicated, to propagate the initial conditions indicated by red using the 1D linear advection equation with a constant velocity* $v = 1.0$. *All schemes use a Courant parameter* $\alpha_c = 0.1$, *and 100 grid points to sample* $u(x)$ *over the* x*-interval* $[0, 1]$, *assuming periodic boundary conditions. The blue curves show the results after 1000 time steps of* $\Delta t = 0.001$, *which covers exactly one full period.*

CHAPTER 2

Consistency, Stability, and Convergence

In the previous chapter we have explored a variety of finite difference methods, applied to a super-simple PDE, namely the 1D linear advection equation (whose analytical solution is trivially known). And we have seen that each method has its own shortcomings. In some cases, large oscillations develop, that continue to grow, in other cases, numerical diffusion washes away any sharp features.

In this chapter we address the question how one can test/assess the performance of finite difference schemes. We start by introducing some relevant terms:

Consistency: A numerical scheme is consistent if its *discrete* operator (with finite differences) converges towards the *continuous* operator (with derivatives) of the PDE in the limit $\Delta t, \Delta x \to 0$ (which is the limit of vanishing truncation error).

Stability: A numerical scheme is stable if numerical noise, from initial conditions, round-off errors, etc., does not grow.

Convergence: A numerical scheme converges if its solution converges towards the real solution of the PDE in the limit $\Delta t, \Delta x \to 0$.

The three 'aspects' of a numerical scheme are related through what is known as **Lax's equivalance theorem:** It states that *for a consistent finite difference method, for a well-posed linear initial value problem, the method is convergent if and only if it is stable.*

The theorem is important since it allows one to assess **convergence** of a finite difference method, which is ultimately what one is interested in, by establishing whether the method is **consistent** and **stable**. Consistency, the requirement that the finite difference scheme approximates the correct PDE is straightforward to verify, and stability is typically much easier to test than convergence. In the remainder of this chapter we discuss how to test for stability, and we introduce the concept of the

modified equation, which is useful to develop a feeling for the behavior of a finite difference method.

Truncation error: As we have seen in the previous chapter, the finite differences are typically obtained using Taylor series expansion up to some order in Δx and/or Δt. This introduces truncation errors, errors that derive from the fact that the series is truncated at some finite order. The forward and backward Euler schemes are first order in both space and time, we write $\mathcal{O}(\Delta t, \Delta x)$, the FTCS and Lax-Friedrichs schemes are first order in time, but second order in space, i.e., $\mathcal{O}(\Delta t, \Delta x^2)$, and the Lax-Wendroff, Beam-Warming and Fromm methods are all second order in both space and time, i.e., $\mathcal{O}(\Delta t^2, \Delta x^2)$. Typically higher order yields better accuracy, if stable. Or, put differently, one can achieve the same accuracy but using a coarser grid/mesh.

As we have seen in the previous chapter, the first-order method that appears stable (the upwind/donor cell methods) yields smeared solutions, while the second-order methods (Lax-Wendroff, Beam-Warming and Fromm) give rise to oscillations. This qualitively different behavior of first and second order methods is typical and can be understood using an analysis of what is called the **modified equation**. Recall that the discrete equation used (i.e., the finite difference scheme adopted) is to approximate the original PDE (in the cases discussed thus far, the 1D linear advection equation). However, the discrete equation may be an *even better approximation* of a modified version of the original PDE (one that corresponds to a higher order of the truncation error). Analyzing this modified equation gives valuable insight into the qualitative behavior of the numerical scheme in question.

As an example, consider the Euler FTCS method, which replaces the actual PDE

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

with the following discrete equation

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

Using Taylor series expansion in time up to second order, we have that

$$u_i^{n+1} = u_i^n + \Delta t \left(\frac{\partial u}{\partial t}\right) + \frac{(\Delta t)^2}{2} \left(\frac{\partial^2 u}{\partial t^2}\right) + \mathcal{O}(\Delta t^3)$$

which implies that

$$\frac{u_i^{n+1} - u_i^n}{\Delta t} = \frac{\partial u}{\partial t} + \frac{\Delta t}{2}\left(\frac{\partial^2 u}{\partial t^2}\right) + \mathcal{O}(\Delta t^2)$$

Similarly, using Taylor series expansion in space up to second order, we have that

$$u_{i+1}^n = u_i^n + \Delta x\left(\frac{\partial u}{\partial x}\right) + \frac{(\Delta x)^2}{2}\left(\frac{\partial^2 u}{\partial x^2}\right) + \mathcal{O}(\Delta x^3)$$

$$u_{i-1}^n = u_i^n - \Delta x\left(\frac{\partial u}{\partial x}\right) + \frac{(\Delta x)^2}{2}\left(\frac{\partial^2 u}{\partial x^2}\right) + \mathcal{O}(\Delta x^3)$$

which implies that

$$\frac{u_{i+1}^n - u_{i-1}^n}{2\Delta x} = \frac{\partial u}{\partial x} + \mathcal{O}(\Delta x^2)$$

Hence, our **modified equation** is

$$\frac{\partial u}{\partial t} + v\,\frac{\partial u}{\partial x} = -\frac{\Delta t}{2}\frac{\partial^2 u}{\partial t^2} + \mathcal{O}(\Delta t^2, \Delta x^2)$$

Using that

$$\frac{\partial^2 u}{\partial t^2} = \frac{\partial}{\partial t}\left(\frac{\partial u}{\partial t}\right) = -v\,\frac{\partial}{\partial t}\left(\frac{\partial u}{\partial x}\right) = -v\,\frac{\partial}{\partial x}\left(\frac{\partial u}{\partial t}\right) = v^2\frac{\partial^2 u}{\partial x^2}$$

where in the second and final step we have used the original PDE to relate the temporal derivate to the spatial derivative. Using this, we can write our **modified equation** as

$$\frac{\partial u}{\partial t} + v\,\frac{\partial u}{\partial x} = -v^2\frac{\Delta t}{2}\frac{\partial^2 u}{\partial x^2} + \mathcal{O}(\Delta t^2, \Delta x^2)$$

Note that the first term on the right-hand side is a **diffusion term**, with a diffusion coefficient

$$\mathcal{D} = -v^2\frac{\Delta t}{2}$$

Hence, to second order, the discrete equation of the 1D linear advection equation based on the FTCS method, actually solves what is known as an **advection-diffusion equation**. But, most importantly, the corresponding diffusion coeffient is negative. This implies that the FTCS scheme is **unconditionally unstable**; i.e., there are no Δx and/or Δt for which the FTCS method will yield a stable solution of the 1D linear advection equation.

Let us now apply the same method to the FTBS scheme, whose discrete equation for the 1D linear advection equation is given by

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

Using Taylor series expansions as above, one finds that

$$\frac{u_i^n - u_{i-1}^n}{\Delta x} = \frac{\partial u}{\partial x} - \frac{\Delta x}{2}\frac{\partial^2 u}{\partial x^2} + \mathcal{O}(\Delta x^2)$$

Hence, our **modified equation** is

$$\frac{\partial u}{\partial t} + v\,\frac{\partial u}{\partial x} = -\frac{\Delta t}{2}\frac{\partial^2 u}{\partial t^2} + v\,\frac{\Delta x}{2}\frac{\partial^2 u}{\partial x^2} + \mathcal{O}(\Delta t^2, \Delta x^2)$$

which can be recast in the **advection-diffusion equation** form with

$$\mathcal{D} = v\,\frac{\Delta x}{2} - v^2\,\frac{\Delta t}{2} = v\,\frac{\Delta x}{2}\left(1 - v\,\frac{\Delta t}{\Delta x}\right) = v\,\frac{\Delta x}{2}\,(1 - \alpha_{\mathrm{c}})$$

Thus we see that we can achieve **stable diffusion** (meaning $\mathcal{D} > 0$) if $v > 0$ and $\alpha_{\mathrm{c}} < 1$ (the latter is the CFL-condition, which has to be satisfied anyways). This explains the diffuse nature of the FTBS scheme (see Fig. 28). It also shows that if $v < 0$ one needs to use the FTFS scheme, to achieve similar stability. The **upwind** or **donor cell** scheme is generic term to refer to the FTBS (FTFS) scheme if $v > 0$ ($v < 0$). Or, put differently, in the upwind method the *spatial differencing is performed using grid points on the side from which information flows.*

The student is encouraged to apply this method to other finite difference schemes. For example, applying it to the Lax-Friedrichs method yields once again a modified equation of the advection-diffusion form, but this time with a diffusion coefficient

$$\mathcal{D} = \frac{\Delta x^2}{2\Delta t}(1 - \alpha_{\mathrm{c}}^2)$$

which results in stable diffusion ($\mathcal{D} > 0$) as long as the CFL-criterion is satisfied.

For Lax-Wendroff and Beam-Warming one obtains modified equations of the form

$$\frac{\partial u}{\partial t} + v\,\frac{\partial u}{\partial x} = \eta\,\frac{\partial^3 u}{\partial x^3} + \mathcal{O}(\Delta t^3, \Delta x^3)$$

with

$$\eta = \frac{v\Delta x^2}{6}\left(\alpha_c^2 - 1\right) \qquad \text{Lax-Wendroff}$$
$$\eta = \frac{v\Delta x^2}{6}\left(2 - 3\alpha_c + \alpha_c^2\right) \qquad \text{Beam-Warming}$$

In order to understand the behavior of the explicit, second-order schemes, consider the modified equation

$$\frac{\partial u}{\partial t} + v\,\frac{\partial u}{\partial x} = \eta\,\frac{\partial^3 u}{\partial x^3}$$

Applying this to a linear wave with frequency ω and wave number k, i.e., $u \propto \exp[\pm i(kx - wt)]$, yields a **dispersion relation**

$$-i\omega + i\,v\,k = -i\,\eta\,k^3 \qquad \Rightarrow \qquad \omega = v\,k + \eta\,k^3$$

and thus a **group velocity**

$$c_g \equiv \frac{\partial \omega}{\partial k} = v + 3\eta k^2$$

Hence, different waves move with a different group velocity, which means that the solution of the above equation is **dispersive**. Numerical noise due to (for example) the truncation error can be written as a Fourier series, and different modes will propagate at different speeds. In particular, if we satisfy the CFL criterion, such that $|\alpha_c| < 1$, then we see that $\eta < 0$ for the Lax-Wendroff scheme. This in turn implies that $c_g < v$, and thus that the noise-modes will start to trail with respect to the advection. This explains why the noise in this scheme is present downwind (see lower-left panel of Fig. 30). In the case of the Beam-Warming method, we have that $\eta > 0$ (if CFL-criterion is satisfied), and thus $c_g > v$; indeed, for this method the noise-induced oscillations lead the advection (see lower middle panel of Fig. 30).

We now turn our attention to the stability of finite difference schemes. In the case where the original PDE is linear, one can assess the stability of the finite difference method using a **von Neumann stability analysis**. This analysis models the numerical noise as a Fourier series, and investigates whether the amplitude of the Fourier modes will grow or not. To see how this works, consider once again the 1D linear advection equation, as discretized by the FTCS method:

$$u_i^{n+1} = u_i^n - \frac{\alpha_c}{2}\left[u_{i+1}^n - u_{i-1}^n\right]$$

Since the underlying PDE is linear, the numerical noise, which is what is added to the actual solution, also obeys the above equation. The von Neumann stability analysis therefore starts by writing the present solution as a Fourier series (representing the numerical noise), i.e.,

$$u_i^n = \sum_k A_k^n \exp(-ikx_i)$$

where we have assumed period boundary conditions, such that we have a discrete sum of modes. Substitution in the above equation yields

$$\begin{aligned} A_k^{n+1} &= A_k^n \left[1 - \frac{\alpha_c}{2}\exp(-ik\Delta x) + \frac{\alpha_c}{2}\exp(+ik\Delta x)\right] \\ &= A_k^n \left[1 + i\alpha_c \sin(k\Delta x)\right] \end{aligned}$$

where we have used that $\sin x = (e^{ix} - e^{-ix})/2i$. The evolution of the mode amplitudes is thus given by

$$\zeta^2 \equiv \frac{|A_k^{n+1}|^2}{|A_k^n|^2} = 1 + \alpha_c^2 \sin^2(k\Delta x)$$

As is evident, we have that $\zeta > 1$, for all k. Hence, for any k the mode amplitude will grow, indicating that the FTCS method is inherently, **unconditionally unstable**.

Now let's apply the same analysis to the upwind scheme (FTBS), for which the discrete equation is given by

$$u_i^{n+1} = u_i^n - \alpha_c \left[u_i^n - u_{i-1}^n\right]$$

Substituting the Fourier series yields

$$\begin{aligned} A_k^{n+1} &= A_k^n \left[1 - \alpha_c + \alpha_c \exp(+ik\Delta x)\right] \\ &= A_k^n \left[1 - \alpha_c + \alpha_c \cos(k\Delta x) + i\alpha_c \sin(k\Delta x)\right] \end{aligned}$$

After a bit of algebra, one finds that the evolution of the mode amplitudes is thus given by

$$\zeta^2 \equiv \frac{|A_k^{n+1}|^2}{|A_k^n|^2} = 1 - 2\alpha_c(1-\alpha_c)[1 - \cos(k\Delta x)]$$

Upon inspection, this has $\zeta < 1$ if $\alpha_c < 1$; Hence, *the upwind scheme is stable as long as the CFL condition is satisfied.* Note, though, that the fact that $\zeta < 1$ implies not only that the numerical noise will not grow, but also that the actual solution will decline with time. Pure advection, which is what the actual PDE describes, show have $\zeta = 1$, i.e., solutions only move, they don't grow or decay with time. Hence, the

fact that our FDA has $\zeta < 1$ is not physical; rather, this represents the numerical diffusion that is present in the upwind scheme.

In Problem Set 3, the students will perform a similar von Neumann stability analysis for an explicit FDA of the heat equation.

CHAPTER 3

Reconstruction and Slope Limiters

In the previous two chapters we discussed how numerically solving the equations of hydrodynamics means that we have to develop a FDA of the PDE (typically hyperbolic, potentially with non-ideal parabolic terms). We discussed how we can obtain insight as to the behavior of the FDA by examining the corresponding modified equation, and by performing a von Neumann stability analysis.

We have compared various FDA schemes to solve the 1D linear advection equation, but found all of them to have serious shortcomings. These became especially apparent when we examined the advection of ICs that contained discontinuities. The first-order FDA schemes were too diffusive and dissipative, while the second order schemes gave rise to spurious over- and undershoots. The latter can be fatal whenever the property to be advected is inherently positive (i.e., mass density). In fact, this relates to an important theorem due to Godunov,

Godunov Theorem: *there are no linear higher-order schemes for treating linear advection that retain positivity of the solution.*

So how are we to proceed? The solution, originally proposed by Dutch astrophysicist Bram van Leer from Leiden, is to use a *non-linear* scheme to treat *linear* advection. This is sometimes called **non-linear hybridization**. These non-linear schemes, though, are based on the **finite volume formulation**, rather than the finite difference formulation adopted thus far. We can transition to the finite volume formulation by taking the **integral form** of the linear advection equation. This is obtained by integrating the **advection equation in conservative form**,

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

over each cell, which is bounded by $x_{i-\frac{1}{2}} \equiv x_i - \Delta x/2$ and $x_{i+\frac{1}{2}} \equiv x_i + \Delta x/2$, and by t^n and $t^{n+1} = t^n + \Delta t$. Note that, in the case of linear advection considered here $f = vu$ with v the constant advection speed, and u the property that is advected. However, what follows is valid for any flux f.

The above equation in integral form is simply

$$\int_{x_{i-\frac{1}{2}}}^{x_{i+\frac{1}{2}}} \mathrm{d}x \int_{t^n}^{t^{n+1}} \mathrm{d}t \left[\frac{\partial u}{\partial t} + \frac{\partial f}{\partial x}\right] = 0$$

which reduces to

$$\int_{x_{i-\frac{1}{2}}}^{x_{i+\frac{1}{2}}} \mathrm{d}x \left[u(x,t^{n+1}) - u(x,t^n)\right] + \int_{t^n}^{t^{n+1}} \mathrm{d}t \left[f(x_{i+\frac{1}{2}},t) - f(x_{i-\frac{1}{2}},t)\right] = 0$$

If we now introduce the cell-averaged quantities

$$U_i^n \equiv \frac{1}{\Delta x}\int_{x_{i-\frac{1}{2}}}^{x_{i+\frac{1}{2}}} u(x,t^n)\mathrm{d}x\,, \qquad U_i^{n+1} \equiv \frac{1}{\Delta x}\int_{x_{i-\frac{1}{2}}}^{x_{i+\frac{1}{2}}} u(x,t^{n+1})\mathrm{d}x$$

and

$$F_{i-\frac{1}{2}}^{n+\frac{1}{2}} \equiv \frac{1}{\Delta t}\int_{t^n}^{t^{n+1}} f(x_{i-\frac{1}{2}},t)\mathrm{d}t\,, \qquad F_{i+\frac{1}{2}}^{n+\frac{1}{2}} \equiv \frac{1}{\Delta t}\int_{t^n}^{t^{n+1}} f(x_{i+\frac{1}{2}},t)\mathrm{d}t$$

then the update formula for the advection equation becomes

$$\boxed{U_i^{n+1} = U_i^n - \frac{\Delta t}{\Delta x}\left(F_{i+\frac{1}{2}}^{n+\frac{1}{2}} - F_{i-\frac{1}{2}}^{n+\frac{1}{2}}\right)}$$

This is similar to the update formula for the conservation equation that we derived in chapter 1, expect that here the quantities are volume averaged. Note that this equation is <u>exact</u> (it is not a numerical scheme), as long as the U and F involved are computed using the above integrations.

Computing the precise fluxes, though, requires knowledge of $u(x,t)$ over each cell, and at each time. This is easy to see within the context of the linear advection equation: Let $u(x,t^n)$ be the continuous description of u at time t^n. Then, the amount of u advected to the neighboring downwind cell in a timestep Δt is simply given by $\Delta u = \int_{x+\frac{1}{2}-v\Delta t}^{x+\frac{1}{2}} u(x,t^n)\mathrm{d}x$ (assuming that $v\Delta t < \Delta x$), and the time-averaged flux

through the corresponding cell face is $F^{n+\frac{1}{2}}_{i+\frac{1}{2}} = \Delta u/\Delta t$. If the continuous $u(x,t^n)$ is known, this flux can be computed, and the advection equation (in integral form) can be solved exactly. However, because of the discrete nature of sampling, we only know u at finite positions x_i, and the best we can hope to do is to *approximate* Δu, and thus the corresponding flux. Once such approximations are introduced, the update formula becomes a numerical scheme, called a **Godunov scheme**.

One ingredient of such a scheme is **reconstruction**, which refers to a method to reconstruct the continuous $u(x,t^n)$ from the discrete $u^n_i = u(x_i,t^n)$. In Godunovs first order method, it is assumed that $u(x,t)$ is **piecewise constant**; i.e., $u(x,t^n) = U^n_i$ for $x_{i-\frac{1}{2}} \le x \le x_{i+\frac{1}{2}}$. Obviously this is basically the simplest, lowest-order approximation one can make. Godunov then used the fact that the discontinuities between adjacent cells, if interpreted as real, basically consistitute what is called a **Riemann problem**, for which a solution can (often) be found analytically. This then allows one to compute the fluxes $F^{n+\frac{1}{2}}_{i\pm\frac{1}{2}}$, which in turn allows one to update $U^n_i \to U^{n+1}_i$. We will discuss the Riemann problem, and (approximate) Riemann solvers in Chapter 5. Here, we will apply this first-order Godunov scheme to our 1D linear advection equation.

The left-hand panel of Fig. 31 shows the condition for some particular $u(x_i)$ at time t^n. Only 5 cells are shown, for the sake of clarity. The cells are assumed to have a constant distribution of u (i.e., we have made the piecewise constant assumption). Let us now focus on cell i, which straddles a discontinuity in u. Advection with a constant $v > 0$ simply implies that in a time step Δt the piecewise constant profile of $u(x)$ shifts right-ward by an amount $v\Delta t$. This right-ward shift is indicated in the right-hand panel of Fig. 31 by the dashed lines. At the end of the time-step, i.e., at time t^{n+1}, we once again want the fluid to be represented in a piecewise constant fashion over the cells. This is accomplished, for cell i, by integrating $u(x)$ under the dashed lines from $x_{i-\frac{1}{2}}$ to $x_{i+\frac{1}{2}}$ and dividing it by Δx to obtain the new cell-averaged value U^{n+1}_i. The new U_i thus obtained are indicated by the solid lines in the right-hand panel. The U^{n+1}_i differs from U^n_i because some amount of u has flown into cell i from cell $i-1$ (indicated by the light-gray shading), and some amount of u has flown from i into cell $i+1$ (indicated by the dark-gray shading). The corresponding time-averaged fluxes obey

$$\Delta t\, F^{n+\frac{1}{2}}_{i-\frac{1}{2}} = (v\,\Delta t)\, U^n_{i-1}\,, \qquad \text{and} \qquad \Delta t\, F^{n+\frac{1}{2}}_{i+\frac{1}{2}} = (v\,\Delta t)\, U^n_i$$

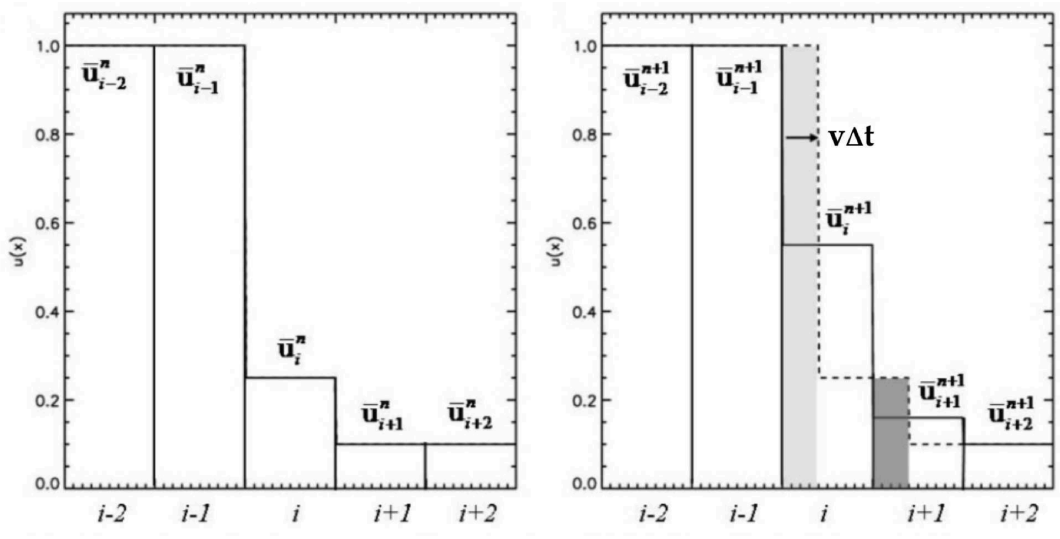


Figure 7: *A single time step in the linear advection of a fluid modelled using piecewise constant reconstruction. The left-hand panel (a) shows the conditions at time t^n. The right-hand panel (b) shows the slabs of fluid after they have been advected for a time Δt (dashed lines) as well as the final profile of $u(x)$ at the end of the time step (solid lines). The total amount of fluid entering (leaving) cell i is shaded light-gray (dark-gray). [Figure adapted from Prof. D. Balsara's lecture notes on "Numerical PDE Techniques for Scientists and Engineers"].*

By invoking conservation of u, we then have that

$$\Delta x\, U_i^{n+1} = \Delta x\, U_i^n + \Delta t\, F_{i-\frac{1}{2}}^{n+\frac{1}{2}} - \Delta t\, F_{i+\frac{1}{2}}^{n+\frac{1}{2}}$$

which implies that

$$U_i^{n+1} = U_i^n - v\frac{\Delta t}{\Delta x}\left(U_i^n - U_{i-1}^n\right) = U_i^n - \alpha_{\rm c}\left(U_i^n - U_{i-1}^n\right)$$

Note that this is exactly the **first-order accurate FTBS upwind (or donor-cell) scheme** from Chapter 1, but now applied to the volume average quantities.

So, one might wonder, what is so 'special' about this **Godunov scheme**? Well, the ingenious aspect of Godunov's method is that is yields an upwind scheme for a general, non-linear system of hyperbolic PDEs. For a linear system of equations, upwind schemes can only be used if all velocities of all waves in the problem (recall

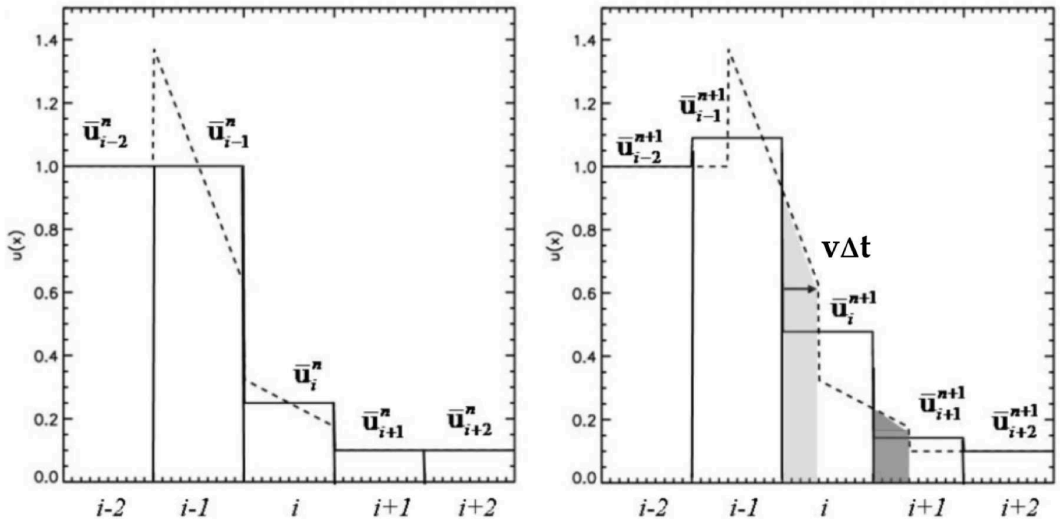


Figure 8: *Same as Fig. 31, but this time piecewise linear reconstruction is used, based on right-sided slopes (thus giving rise to the Lax-Wendroff scheme). Note how the linear reconstruction has introduced a new, higher-than-before, extremum in cell* $i+1$*, which is ultimately responsible for the spurious oscillations characteristic of second-order schemes. As discussed in the text, the solution is to develop a Total Variation Diminishing (TVD) scheme with the use of slope-limiters. [Figure adapted from Prof. D. Balsara's lecture notes on "Numerical PDE Techniques for Scientists and Engineers"].*

that hyperbolic PDEs describe travelling waves) have the same sign. If mixed signs are present, one can typically split the flux $F(u)$ in two components: F^+ and F^- which correspond to the fluxes in opposite directions. This is called **Flux Vector Splitting**. The linearity of the PDE(s) then assures that the solution of the PDE is simply given by the sum of the PDEs for F^+ and F^- separately. However, for a non-linear system (we will encounter such systems in the next chapter) this approach will not work. This is where Godunov's method really brings its value to bear.

For now, though, we apply it to the 1D linear advection equation, in which case it simply becomes identical to the first-order accurate FTBS scheme. And as we have already seen, this scheme suffers from a large amount of numerical diffusion. But, within the Godunov scheme, we can now try to overcome this by going to higher-order. In terms of **reconstruction**, this implies going beyond piecewise constant reconstruction.

The logical next-order step in reconstruction is to assume that within each cell $u(x)$ follows a linear profile, with a slope that is determined by the values of U at its neighboring cells. This is called **piecewise linear reconstruction**. As always, we have three choices for the slope: a right-sided finite difference $\delta U_i^n = U_{i+1}^n - U_i^n$, a left-sided difference $\delta U_i^n = U_i^n - U_{i-1}^n$ and a central difference $\delta U_i^n = (U_{i+1}^n - U_{i-1}^n)/2$. In what follows we shall refer to δU_i^n as the **slope**, eventhough it really is only an **undivided difference**.

The left-hand panel of Fig. 32 shows the same mesh function as in Fig 31, but this time the dashed lines indicate the reconstructed profile based on the right-sided slopes. For cells $i-2$, $i+1$ and $i+2$, this right-sided slope is zero, and the reconstruction is thus identical to that for the piecewise constant case. However, for cells $i-1$ and i reconstruction has endowed the cells with a non-zero slope. For example, the profile of $u(x)$ in cell i is given by

$$u_i^n(x) = U_i^n + \frac{\delta U_i^n}{\Delta x}(x - x_i)$$

where x_i is the central position of cell i. It is easy to see (the student should do this), that upon substitution of this profile in the integral expression for U_i^n, one obtains that $U_i^n = u_i^n$, as required.

Advecting the fluid *with second-order accuracy* is equivalent to shifting the *piecewise linear profile* rightwards by a distance $v\Delta t$. The resulting, shifted profile is shown in the right-hand panel of Fig. 32. As in Fig. 31, the light-gray and dark-gray shaded regions indicate the amount of u that is entering cell i from cell $i-1$, and leaving cell i towards $i+1$, respectively. With a little algebra, one finds that the associated time-averaged fluxes obey

$$\Delta t\, F_{i-\frac{1}{2}}^{n+\frac{1}{2}} = (v\Delta t)\left[U_{i-1}^n + \frac{1}{2}(1-\alpha_{\rm c})\delta U_{i-1}^n\right]$$

and

$$\Delta t\, F_{i+\frac{1}{2}}^{n+\frac{1}{2}} = (v\Delta t)\left[U_{i}^n + \frac{1}{2}(1-\alpha_{\rm c})\delta U_{i}^n\right]$$

As before, invoking conservation of u then implies that

$$\boxed{U_i^{n+1} = U_i^n - \alpha_{\rm c}\left(U_i^n - U_{i-1}^n\right) - \frac{\alpha_{\rm c}}{2}(1-\alpha_{\rm c})\left(\delta U_i^n - \delta U_{i-1}^n\right)}$$

A comparison with the update formula in the piecewise constant case, we see that we have added an extra term proportional to $(\Delta t/\Delta x)^2$ that depends on the slopes. Hence, this is indeed a second-order scheme. By substituting the expressions for the right-sided bias adopted here, the update formula becomes identical to that of the **Lax-Wendroff scheme** that we encountered in Chapter 1, but with u_i replaced by U_i. Similarly, it is easy to show that using the left-sided slopes, yields an update formula equal to that for the **Beam-Warming scheme**, while the central slopes yield an update formula identical to **Fromm's scheme**.

Piecewise Constant Reconstruction	$\rightarrow$	Upwind scheme
Piecewise Linear Reconstruction		
+ right-sided slopes	$\rightarrow$	Lax-Wendroff scheme
+ central slopes	$\rightarrow$	Fromm scheme
+ left-sided slopes	$\rightarrow$	Beam-Warming scheme

Finite volume reconstruction methods and their link to finite difference schemes.

As we have seen in Chapter 1, these second-order accurate schemes all give rise to large oscillations; large over- and undershoots. And as we know from **Godunov's theorem**, these schemes are not positivity-conserving. Fig. 32 makes it clear where these problems come from. Advection of the linearly reconstructed $u(x)$ has caused a spurious overshoot in cell $i-1$ at time t^{n+1}. Upon inspection, it is clear that this overshoot arises because our reconstruction has introduced values for $u(x)$ that are higher than any u_i present at t^n. Once such an unphysical extremum has been introduced, it has a tendency to grow in subsequent time-steps. Using the centered slopes would cause a similar overshoot (albeit somewhat smaller), while the left-sided slopes will result in an *under*shoot in cell $i+1$.

This insight shows us that the over- and under-shoots have their origin in the fact that the linear reconstruction introduces new extrema that were not present initially. The solution, which was originally suggested by Bram van Leer, is to *limit* the piecewise linear profile within each cell such that no new extrema are introduced. This is accomplished by introducing **slope-limiters** (or, very similar, **flux-limiters**). The idea is simple: limit the slopes δU_i^n, such that no new extrema are introduced. Over the years, many different slope-limiters have been introduced by the computational

fluid dynamics community. All of these use some combination of the left- and right-sided slopes defined above. An incomplete list of slope-limiters is presented in the Table below.

An incomplete list of Slope Limiters

van Leer	$\delta U_i^n = \ Q(\delta_{\rm L}, \delta_{\rm R}) \, \frac{\delta_{\rm L} \, \delta_{\rm R}}{\lvert\delta_{\rm L}\rvert + \lvert\delta_{\rm R}\rvert}$
MinMod	$\delta U_i^n = \frac{1}{2} Q(\delta_{\rm L}, \delta_{\rm R}) \min(\lvert\delta_{\rm L}\rvert, \lvert\delta_{\rm R}\rvert)$
Monotized Central	$\delta U_i^n = \frac{1}{2} Q(\delta_{\rm L}, \delta_{\rm R}) \min(\frac{1}{2}\lvert\delta_{\rm L} + \delta_{\rm R}\rvert, 2\lvert\delta_{\rm L}\rvert, 2\lvert\delta_{\rm R}\rvert)$
Superbee	$\delta U_i^n = \frac{1}{2} Q(\delta_{\rm L}, \delta_{\rm R}) \max\left[\min(2\lvert\delta_{\rm L}\rvert, \lvert\delta_{\rm R}\rvert), \min(\lvert\delta_{\rm L}\rvert, 2\lvert\delta_{\rm R}\rvert)\right]$

Here $\delta_{\rm L}$ *and* $\delta_{\rm R}$ *are the left- and right-sided slopes, and* $Q(\delta_{\rm L}, \delta_{\rm R}) = [\mathrm{sgn}(\delta_{\rm L}) + \mathrm{sgn}(\delta_{\rm R})]$ *with* $\mathrm{sgn}(x)$ *the sign-function, defined as* $+1$ *for* $x \geq 0$ *and* -1 *for* $x < 0$.

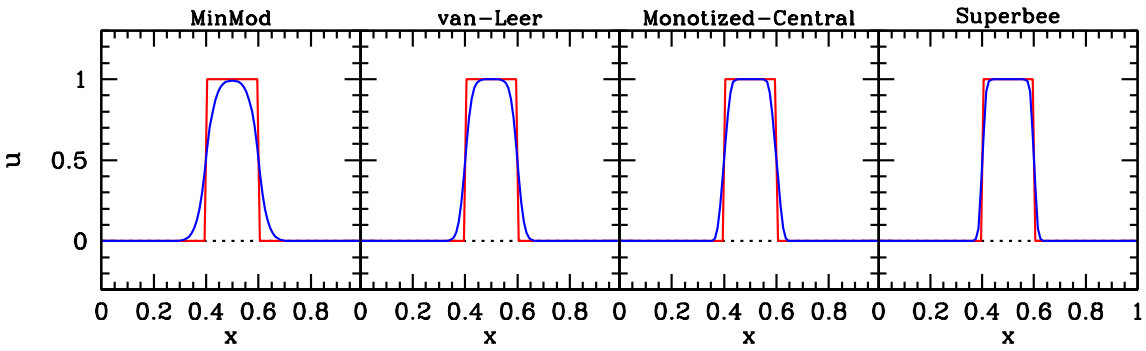


Figure 9: *The result of using Piecewise Linear Reconstruction combined with 4 different slope limiters (as indicated at the top of each panel). to linearly advect the initial conditions shown in red. As in Fig. 30, the advection speed is* $v = 1.0$*, and 100 grid points are used to sample* $u(x)$ *over the* x*-interval* $[0, 1]$*, assuming periodic boundary conditions. The Courant parameter* $\alpha_c = 0.5$*. The blue curves show the results after 1000 time steps of* $\Delta t = 0.001$*, which covers exactly one full period. Note the drastic improvement compared to the finite difference schemes used in Fig. 30!!*

Fig. 33 shows the results of applying our Piecewise Linear Reconstruction with four different slope-limiters (as indicated) to the 1D linear advection of initial conditions indicated by the red top-hat. The blue curves show the results obtained after one period (using periodic boundary conditions) using an advection speed $v = 1.0$, a mesh with 100 grid points on the domain $x \in [0, 1]$, and a Courant parameter $\alpha_c = 0.5$. As can be seen, all limiters produce oscillation-free propogation of the top-hat profile, and with a numerical diffusion that is much smaller than in the case of the upwind finite difference scheme used in Chapter 1 (i.e., compare Fig. 33 to the results in Fig. 30). Clearly, by using a **finite volume formulation** with **non-linear hybridizatrion** in the form of **piecewise linear reconstruction** with the use of **slope limiters** has drastically improved our ability to advect discontinuous features in the fluid.

What is it that makes these slope-limiters so succcessful? In short, the reason is that they are **total variation dimishing**, or **TVD** for short. The total variation, TV of a discrete set $U = U_1, U_2, ..., U_N$ is defined as

$$\boxed{TV(U^n) \equiv \sum_{i=1}^{N} \left| U_{i+1}^n - U_i^n \right|}$$

and an integration scheme is said to be **TVD** iff the total variation does NOT increase with time, i.e.,

$$\boxed{\textbf{TVD} \qquad \Leftrightarrow \qquad TV(U^{n+1}) \leq TV(U^n)}$$

Clearly, whenever a scheme introduces spurious oscillations, the TV will go up, violating the **TVD**-condition. Or, put differently, if a scheme is **TVD**, then it will not allow for the formation of spurious over- and/or under-shoots. Readers interested in finding a quick method to test whether a scheme is **TVD** are referred to the paper *"High Resolution schemes for Conservation Laws"* by Harten (1983) in the Journal of Computational Physics. Here we merely point out that the schemes used in this chapter are all **TVD**.

As we have seen, using reconstruction combined with slope-limiters yields integration schemes that do very well for our 1D linear advection problem. A natural extention to even high-order can be achieved by using high-order in the reconstruction. For example, one could use piecewise parabolic reconstruction, which is third-order accurate in space, and indeed, such schemes have been developed and are in use. It is important, though, to realize that as long as one uses a "piecewise" reconstruction, that discontinuities between neighboring cells persist. And it is these discustinuities that typically cause problems.

This begs the question: why can't we use a continuous reconstruction, i.e., connect all the u_i $(i = 1, 2, ..., N)$ using say an N^{th}-order polynomial. That would assure smoothness and differentiability across the entire computational domain. However, this is not an option, for the simple reason that, as we will see in the next Chapter, discontinuities can be real. An obvious example is a shock, which is a natural outcome of the Euler equations due to its **non-linear** character. Using continuous reconstruction would fail to capture such discontinuities.

As we will see, the solution is to use **Godunov schemes** that rely on use piecewise reconstruction (be it constant, linear or parabolic) and Riemann solvers to compute the fluxes across the resulting discontinuities between adjacent cells. Before we examine this approach in detail, though, we will first take a closer look at non-linearity.

CHAPTER 4

Burgers' Equation & Method of Characteristics

In the previous chapters we examined a number of different numerical schemes to solve the 1D linear advection equation. We derived this equation from the set of hydro-equations by ignoring gravity and radiation, and by assuming constant pressure and velocity (clearly a highly simplified case).

The linear advection equation is an ideal test case because its solution is trivially known (or can be derived using the **method of characteristics** discussed below). In this chapter we are going to consider another equation, which appears very similar to the linear advection equation, except that it is non-linear. As before, we consider the 1D case, and we ignore radiation and gravity. But rather than assuming both P and $\vec{u}$ to be constant, we only assume a constant pressure. This implies that the continuity equation is given by

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

while the momentum equation reduces to

$$\frac{\partial \rho u}{\partial t} + \frac{\partial}{\partial x}\left[\rho u u + P\right] = \rho \frac{\partial u}{\partial t} + u \frac{\partial \rho}{\partial t} + \rho u \frac{\partial u}{\partial x} + u \frac{\partial \rho u}{\partial x} = 0$$

where we have used that $\partial P / \partial x = 0$. Multiplying the continuity equation with u and subtracting this from the momentum equation yields

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

This equation is known as **Burgers' equation**. Unlike the similar looking advection equation, this is a **non-linear** equation. In fact, it is one of the few non-linear PDEs for which an analytical solution can be found for a select few ICs (see below). The importance of Burgers' equation is that it highlights the quintessential non-linearity of the Euler equations.

Note that the above form of Burgers' equation is not in conservative form. Rather this form is called **quasi-linear**. However, it is trivial to recast Burgers' equation in conservative form:

$$\boxed{\frac{\partial u}{\partial t} + \frac{\partial \frac{1}{2}u^2}{\partial x} = 0}$$

Let's devise finite difference upwind schemes for both (assuming $u > 0$). The results are shown in the table below.

quasi-linear	$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x}$	$u_i^{n+1} = u_i^n - \frac{\Delta t}{\Delta x}\, u_i^n \left[u_i^n - u_{i-1}^n\right]$
conservative	$\frac{\partial u}{\partial t} + \frac{\partial \frac{1}{2}u^2}{\partial x}$	$u_i^{n+1} = u_i^n - \frac{\Delta t}{2\Delta x} \left[(u_i^n)^2 - (u_{i-1}^n)^2\right]$

Forms of Burgers' equation and the corresponding numerical upwind scheme.

Let's use these two schemes to numerically solve Burgers' equation on the domain $x \in [0, 1]$ (using periodic boundary conditions) for an initial velocity field $u(x, 0)$ given by a Gaussian centered at $x = 0.5$, and with a dispersion equal to $\sigma = 0.1$. The initial density is assumed to be uniform. The results for a Courant parameter $\alpha_c = 0.5$ are shown in Fig. 34, where the initial conditions are shown in red, the results from the quasi-linear scheme in magenta (dashed) and the results from the conservative scheme in blue (solid). Note how, in the region where $\partial u/\partial x > 0$ a **rarefaction wave** develops, causing a reduction in the density. Over time, this rarefied region grows larger and larger. In the region where $\partial u/\partial x < 0$ a **compression wave** forms, which steepens over time. Because of the non-linear nature of the Euler equations such waves steepen to give rise to shocks, representing discontinuities in flow speed.

Note, though, that at late times the numerical schemes based on the conservative and quasi-linear forms of Burgers' equation yield different predictions for the location of this shock. As it turns out, and as we demonstrate explicitely below, the correct prediction is that coming from the conservative form. This highlights the importance of using a conservative scheme, which is expressed by the following theorem:

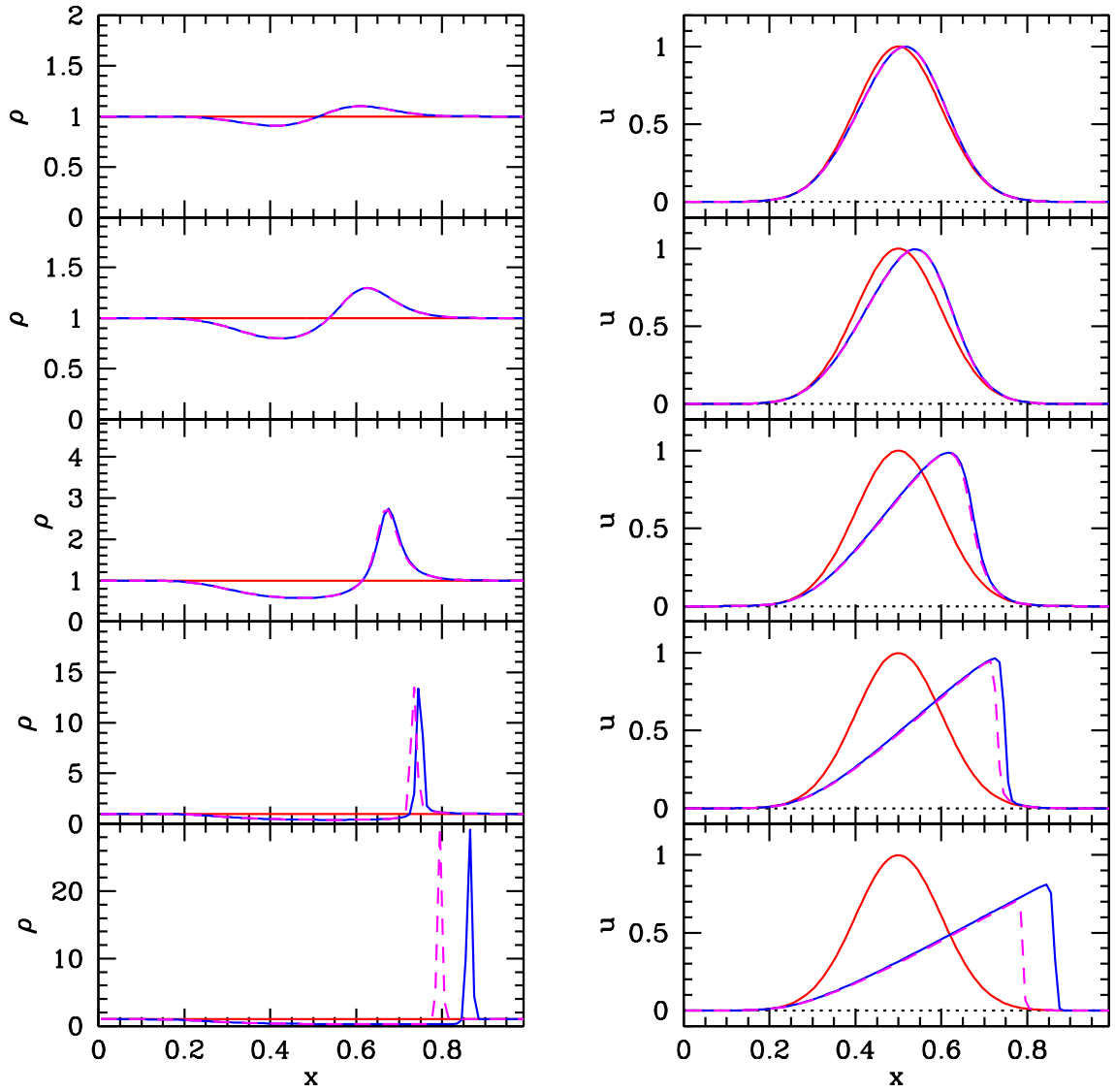


Figure 10: *Evolution as governed by Burgers' equation for an initial, uniform density with the 1D velocity field given by the red Gaussian. Left and right-hand panels show the evolution in density and velocity, respectively. Red lines indicate the initial conditions, while blue (solid) and magenta (dashed) lines indicate the numerical results obtained using the conservative and quasi-linear equations, respectively. Both are solved using the upwind scheme with a Courant parameter* $\alpha_c = 0.5$*, and sampling the* $x = [0, 1]$ *domain using 100 grid points. Note how a shock develops due to the non-linear nature of Burgers' equation, but that the location of the shock differs in the two schemes. Only the conservative scheme yields the correct answer.*

Lax-Wendroff theorem: *If the numerical solution of a conservative scheme converges, it converges towards a weak solution.*

In mathematics, a weak solution (also called a generalized solution) to an ordinary or partial differential equation is a function for which the derivatives may not all exist but which is nonetheless deemed to satisfy the equation in some precisely defined sense. Practically, what the Lax-Wendroff theorem indicates is that the use of equations in conservation form assures that shocks move at the correct speed, and thus converge to the correct location on the mesh.

In order to develop some understanding of the shock and rarefaction, we are going to solve Burgers' equation analytically using the 'method of characteristics', which is a powerful method to solve hyperbolic PDEs.

Method of Characteristics: The idea behind the method of characteristics is to find curves, called *chararacteristics* or *characteristic curves*, along which the PDE becomes an ODE. Once the ODE corresponding to the characteristics is found, these can be solved along the characteristic and transformed into a solution of the PDE. Rather than going through a detailed description of the **method of characteristics**, which can be found in any good textbook on differential calculus[1], we are going to give an example; we will use the method of characteristics to solve the 1D **Burgers equation** which, as discussed above, describes the non-linear evolution of the velocity field in a case without gravity, without radiation, and with a constant pressure (no pressure gradients).

Let the ICs of Burgers' equation be given by the initial velocity field $u(x,0) = f(x)$. Now consider an 'observer' moving with the flow (i.e., an observer 'riding' a fluid element). Let $x(t)$ be the trajectory of this observer. At $t = 0$ the observer is located at x_0 and has a velocity $u_0 = f(x_0)$. We want to know how the velocity of the observer changes as function of time, i.e., along this trajectory. Hence, we want to know

$$\frac{\mathrm{d}u}{\mathrm{d}t} = \frac{\mathrm{d}}{\mathrm{d}t}u(x(t),t) = \frac{\partial u}{\partial t} + \frac{\partial u}{\partial x}\,\frac{\mathrm{d}x}{\mathrm{d}t}$$

[1] or, for an elementary introduction, see https://www.youtube.com/watch?v=tNP286WZw3o

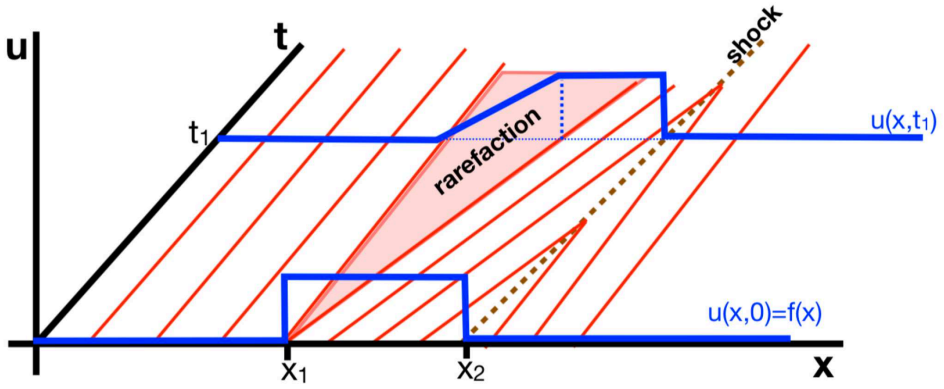


Figure 11: *Solving the Burgers equation for the initial conditions indicated by the blue curve at $t = 0$ using the method of characteristics. The red lines are characteristics; lines along which the velocity remains fixed to the intial value. Not the formation of a rarefaction fan, where the method of characteristics fails to provide a solution, and the formation of a shock where-ever characeristics collide together.*

We see that this equation is equal to Burgers' equation that we seek to solve if $\mathrm{d}x/\mathrm{d}t = u(x,t)$. And in that case we thus have that $\mathrm{d}u/\mathrm{d}t = 0$. Hence, we see that solving the Burgers equation (a quasi-linear, first-order PDE), is equivalent to solving the ODE $\mathrm{d}u/\mathrm{d}t = 0$ along characteristic curves (characteristics) given by $\mathrm{d}x/\mathrm{d}t = u(x,t)$. The solution is simple: $u(x,t) = u_0(x_0,0) = f(x_0)$ where $x_0 = x - u_0 t$. Hence, this can be solved implicitly: for given x and t find the x_0 that solves $x_0 = x - f(x_0)t$. Then, the instantaneous velocity at (x,t) is given by $f(x_0)$.

Fig. 35 illustrates an example. The blue curve indicates the initial conditions; i.e., the initial velocity as function of position x at $t = 0$. It shows a sudden jump (increase) in velocity at x_1 and a sudden decrease at $x = x_2$. The red lines are characteristics, i.e., lines along which the velocity remains constant; their slope is the inverse of the initial velocity at the location x_0 where they cross the $t = 0$ axis. From point x_1, a **rarefraction fan** emenates, which corresponds to a region where the density will decline since neighboring elements spread apart. The method of characteristics does not give a solution in this regime, simply because no characteristics enter here...the solution in this regime turns out to be a linear interpolation between the beginning and end-point of the fan at a given t. From point x_2 a **shock** emenates. Here characteristic from $x_0 < x_2$ 'merge' with characteristics emerging from $x_0 > x_2$. When two characteristics meet, they stop and a discontinuity in the solution emerges (which manifests as a shock).

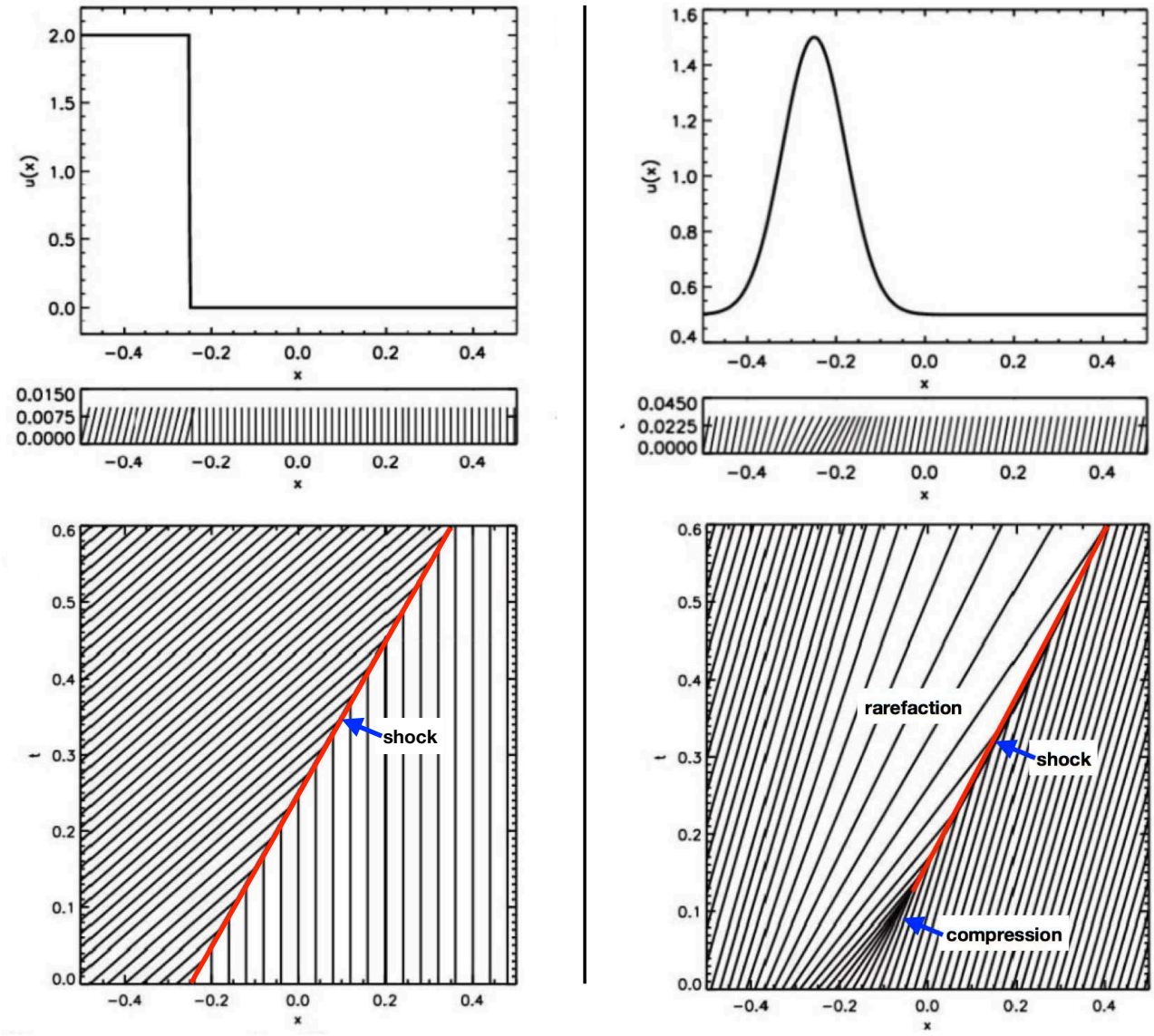


Figure 12: *Initial conditions of $u(x)$ (top panels), and the corresponding characteristics (bottom panels). Note the formation of shocks, and, in the right-hand panel, of a rarefaction wave. Clearly, characteristics give valuable insight into the solution of a hyperbolic PDE.*

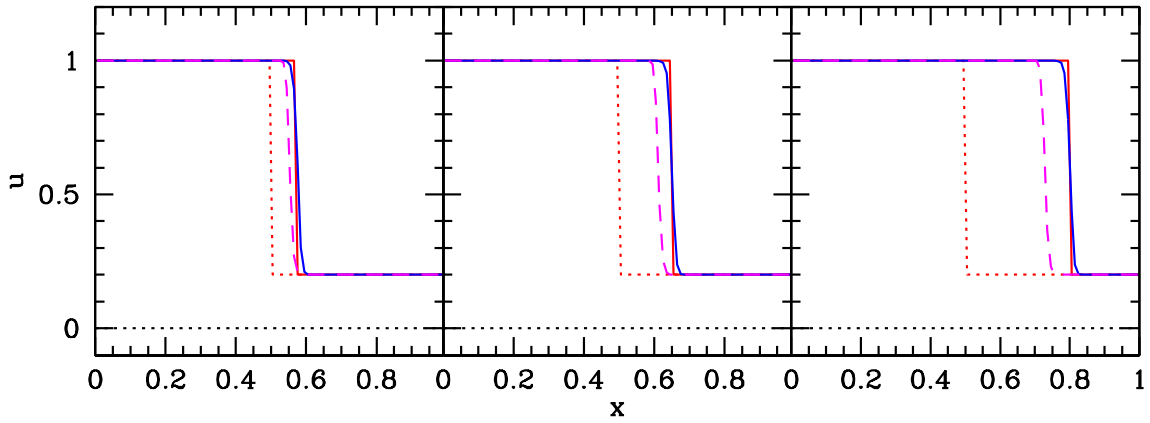


Figure 13: *Evolution of a shock wave in velocity. The initial discontinuity in* $u(x)$ *at* $x = 0.5$ *introduces a shock wave which propagates to the right. The solid red curve panels show the analytical solution (a shock propagating at* $u_{\rm shock} = [u(x < 0.5) + u(x > 0.5)]/2$*), while the red dotted curve shows the ICs. As in Fig. 34, the blue and magenta curves indicate the numerical solutions obtained using conservative and quasi-linear schemes, respectively. Note how the latter fails to reproduce the correct shock speed.*

This is further illustrated in Fig. 36. Upper panels show the initial conditions, with the little bar under the panel indicating with little line-segments the velocity (as reflected by the slope of the line-segment) as function of position. The lower panels plot the characteristics (in a t vs. x plot). Where characteristics merge, a shock forms. It is apparent from the left-hand panels, that the shock in this case will propagate with a speed that is simply the median of the upwind and downwind material, i.e., $u_{\rm shock} = [u(x_2+) + u(x_2-)]/2$. The right-hand panels show the characteristics in the case of the Gaussian ICs also considered in Fig. 34; note how one can see the formation of both a **rarefaction fan** as well as a **shock**.

The example shown in the left-hand panels of Fig. 36 presents us with a situation in which the shock speed is known analytically. We can use this as a test-case to determine which of our numerical schemes (quasi-linear vs. conservative) best reproduces this. We set up ICs in which $u(x) = 1.0$ for $x < 0.5$ and $u(x) = 0.2$ for $x > 0.5$. We solve this numerically using both schemes (with $\alpha_{\rm c} = 0.5$), and compare the outcome to our analytical solution (the shock is moving right-ward with a speed $v_{\rm shock} = (1.0 + 0.2)/2 = 0.6$). The results are shown in Fig. 37. Note

how the solutions from the conservative scheme (in blue) nicely overlap with the analytical solution (in red), while that from the quasi-linear scheme (in magenta) trails behind. This demonstrates the **Lax-Wendroff theorem**, and makes it clear that conservative schemes are required to correctly model the propogation of shocks.

Shock-formation is a natural outcome of the non-linear behavior of the **Burgers' equation**. And since Burgers' equation is a simplified case of the general **Euler equations**, we are to be prepared to deal with shocks and discontinuities in our attempt to numerically solve the hydrodynamic equations. Practically, as we have seen here, this means that we need to consider the Euler equations in conservative form.
%hfill

CHAPTER 5

The Riemann Problem & Godunov Schemes

Thus far, rather than trying to numerically solve the full set of hydrodynamics equations, we instead considered two very special, much simpler cases, namely the **linear advection equation**, and the **non-linear Burgers' equation**, both in 1D. We derived these equation from the set of hydro-equations by assuming an ideal fluid, ignoring gravity and radiation, by assuming constant pressure, and, in the case of the advection equation, also constant velocity. Clearly these are highly simplified cases, but they have the advantage that analytical solution exist, thus allowing us to test our numerical schemes.

We have seen, though, that numerically solving even these super-simple PDEs using finite difference schemes is far from trivial. First order schemes, if stable, suffer from significant numerical diffusion, while second order schemes have a tendency to develop oscillations. As we will discuss in this chapter, and briefly touched upon in Chapter 3, the way forward is to use **Godunov schemes** with **Riemann solvers**.

A **Godunov scheme** is a **Finite Volume Formulation** for solving **conservation laws**, that relies on **reconstruction**. The order of the scheme is related to the method of reconstruction used: **piece-wise constant reconstruction** yields Godunov's first-order scheme. In Chapter 3 we applied this to the linear advection problem, and obtained exactly the first-order Euler FTBS upwind scheme. The power of the first-order Godunov scheme, though, is that it is an upwind scheme for a general, linear or non-linear, system of hyperbolic PDEs; not only for the linear advection equation. If one uses **piece-wise linear reconstruction**, one obtains a scheme that is second-order accurate. As **Godunov's theorem** states, though, a linear higher-order scheme does not preserve positivity, and we have seen the consequences of that in Chapter 1 in the form of large over- and undershoots. As we discussed in Chapter 3, the solution for that is to use a non-linear scheme that involves **slope-limiters**. Applying that to the linear advection equation yields very satisfactory results indeed. And one can in principle take this to even higher accuracy (third-order) by using **piece-wise parabolic reconstruction**, which is something we won't discuss in any detail in these lecture notes (for the interested reader, see Colella & Woodward, 1984).

In this Chapter we are going to see how to apply **Godunov schemes** to the (1D) Euler equations. This will involve **Riemnann solvers**, which are numerical schemes for solving **Riemann problems**, which describes the evolution of a discontinuity in fluid properties. We will discuss how to solve a Riemann problem, apply it to the **SOD shock tube**, and then end by briefly discussing **approximate Riemann solvers**.

Let us first give a brief review of the basics behind the **Godunov scheme**. In the absence of **source/sink terms** (i.e., gravity and radiation), and ignoring viscocity and conduction (which add **parabolic terms**), the hydrodynamic equations reduce to a set of **hyperbolic** PDEs that can be written in **conservation form** as

$$\frac{\partial \vec{u}}{\partial t} + \nabla \cdot \vec{f}(\vec{u}) = 0$$

(see Chapter 1). The update-formula for this equation, in the Finite Volume formulation, is given by

$$\boxed{U_i^{n+1} = U_i^n - \frac{\Delta t}{\Delta x}\left(F_{i+\frac{1}{2}}^{n+\frac{1}{2}} - F_{i-\frac{1}{2}}^{n+\frac{1}{2}}\right)}$$

where the cell-averaged quantities are defined as

$$U_i^n \equiv \frac{1}{\Delta x}\int_{x_{i-\frac{1}{2}}}^{x_{i+\frac{1}{2}}} u(x,t^n)\mathrm{d}x\,, \qquad U_i^{n+1} \equiv \frac{1}{\Delta x}\int_{x_{i-\frac{1}{2}}}^{x_{i+\frac{1}{2}}} u(x,t^{n+1})\mathrm{d}x$$

and

$$F_{i-\frac{1}{2}}^{n+\frac{1}{2}} \equiv \frac{1}{\Delta t}\int_{t^n}^{t^{n+1}} f(x_{i-\frac{1}{2}},t)\mathrm{d}t\,, \qquad F_{i+\frac{1}{2}}^{n+\frac{1}{2}} \equiv \frac{1}{\Delta t}\int_{t^n}^{t^{n+1}} f(x_{i+\frac{1}{2}},t)\mathrm{d}t$$

Reconstruction basically means that one models the continuous $u(x,t)$ from the discrete u_i^n on the mesh. This means that, for $x_{i-\frac{1}{2}} \le x \le x_{i+\frac{1}{2}}$, one assumes that

$$u(x,t^n) = u_i^n \qquad \text{piecewise constant}$$

$$u(x,t^n) = u_i^n + \frac{\delta U_i^n}{\Delta x}(x - x_i) \qquad \text{piecewise linear}$$

In the latter δU_i^n is a **slope**, which can be computed centered, left-sided or right-sided. Once such a reconstruction scheme is adopted, one can compute the U_i^n using the integral expression given above. Next, the Godunov schemes use (approximate) Riemann solvers to infer the (time-averaged) fluxes $F_{i-\frac{1}{2}}^{n+\frac{1}{2}}$ and $F_{i+\frac{1}{2}}^{n+\frac{1}{2}}$. The idea is that reconstruction (be it piecewise constant, piecewise linear or piecewise parabolic) leaves discontinuities between adjacent cells. Godunov' insight was to treat these as 'real' and to solve them analytically as Riemann problems. That implies that one now has, at each cell-interface, a solution for $u(x,t)$, which can be integrated over time to infer $F_{i-\frac{1}{2}}^{n+\frac{1}{2}}$ and $F_{i+\frac{1}{2}}^{n+\frac{1}{2}}$. Next, one uses the update formula to compute U_i^{n+1}, and one proceeds cell-by-cell, and time-step by time-step. In what follows we take a closer look at this Riemann problem and how it may be solved.

Riemann Problem: A Riemann problem, named after the mathematician Bernhard Riemann, is a specific initial value problem composed of a conservation equation together with piecewise constant initial data which has a single discontinuity in the domain of interest. Let L and R denote the states to the left and right of the discontuity. Each of these states is described by three quantities. These can be the **conserved quantities** ρ, ρu and $E = \frac{1}{2}\rho u^2 + \rho\varepsilon$, or what are called the **primitive variables** ρ, u, and P.

The solution of the Riemann problem, i.e., the time-evolution of this discontinuous initial state, can comprise

- 0 or 1 contact discontinuitieas (also called entropy jumps)
- 0, 1 or 2 shocks
- 0, 1 or 2 rarefaction waves (or fans)

but, the total number of shocks *plus* rarefaction fans cannot exceed two. All these shocks, entropy jumps and rarefaction waves appear as **characteristics** in the solution. In particular, the velocities of the features are given by the eigenvalues of the Jacobian matrix of the flux function (called the **characteristic matrix**), which is given by

$$A_{ij}(\vec{q}) \equiv \frac{\partial f_i}{\partial q_j}$$

where $\vec{f}(\vec{q})$ is the flux in the Euler equations in conservative form.

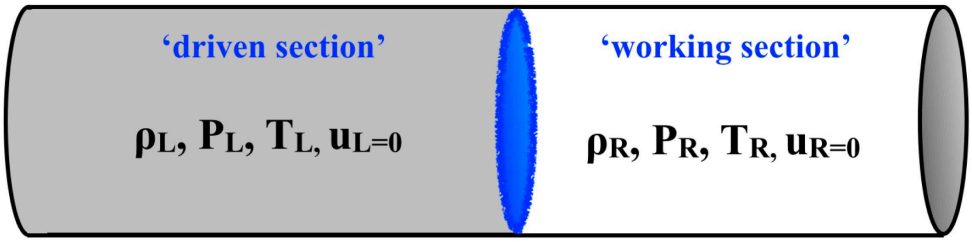


Figure 14: *Initial conditions for the Sod shock tube. The left region has the higher pressure (i.e., $P_L > P_R$) and is therefore called the driven section, while the region on the right is called the working section. The two regions are initially separated by a diaphragm (in blue), which is instantaneously removed at $t = 0$. Both the fluid on the left and right are assumed to be ideal fluids with an ideal equation of state.*

The solution for a completely general Riemann problem can be tedious, and will not be discussed here. Rather, we will look at a famous special case, the **Sod shock tube problem**, named after Gary Sod who discussed this case in 1978. It is a famous example of a 1D Riemann problem for which the solution is analytical, and which is often used as a typical test-case for numerical hydro-codes.

The shock tube is a long one-dimensional tube, closed at its ends and initially divided into two equal size regions by a thin diagragm (see Fig. 38). Each region is filled with the same gas (assumed to have an ideal equation of state), but with different thermodynamic parameters (pressure, density and temperature). The gas to the left, called the **driven section**, has a higher pressure than that to the right, called the **working section** (i.e., $P_L > P_R$), and both gases are initially at rest (i.e., $u_L = u_R = 0$). At $t = 0$, the diagragm, which we consider located at $x = x_0$ is instantaneously removed, resulting in a high speed flow, which propagates into the working section. The high-pressure gas originally in the driven section expands, creating an **expansion or rarefaction wave**, and flows into the working section, pushing the gas of this part. The rarefaction is a continuous process and takes place inside a well-defined region, called the **rarefaction fan**, which grows in width with time (see also Chapter 4). The **compression** of the low-pressure gas results in a **shock wave** propagating into the working section. The expanded fluid (originally part of the driven section) is separated from the compressed gas (originally part of the working section) by a **contact discontinuity**, across which there is a jump in entropy. The velocities and pressures on both sides of the contact discotinuity, though, are identical (otherwise it would be a shock).

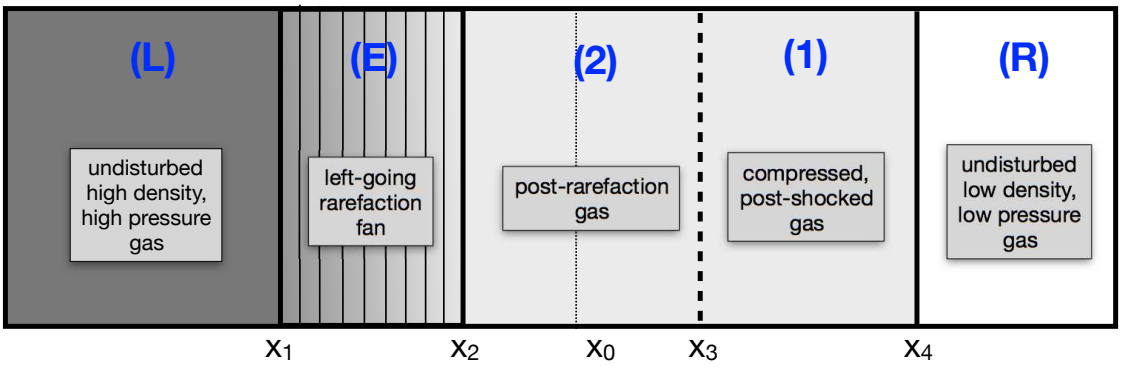


Figure 15: *Illustration of the different zones present in the SOD shock tube. The original diaphragm, which was removed at $t = 0$, was located at x_0 indicated by the dotted line. The solid line at x_4 marks the location of the right-going shock, while the dashed line at x_3 corresponds to a contact discontuity. The region marked (E), between x_1 and x_2, indicates the left-going rarefaction fan. Regions (L) and (R) are not yet affected by the removal of the diaphragm and thus reflect the initial conditions Left and Right of x_0.*

Fig. 39 illustrates the different zones at a time before either the shock wave or the rarefaction fan has been able to reach the end of the tube. Hence, the regions to the far left and far right are still in their original, undisturbed states, to which we refer as the 'L' and 'R' states, respectively. In between, we distinguish three different zones; a rarefaction fan 'E', a region of gas (region '2') that originally came from the driven section but has been rarefied due to expansion, and a region with gas (region '1') that originally belonged to the working section but that has been compressed (it has been overrun by the shock wave). Note that regions 1 and 2 are separated by a contact discontinuity (aka entropy jump).

Our goal is to compute $\rho(x,t)$, $u(x,t)$, and $P(x,t)$ in each zone, as well as the locations x_1, x_2, x_3 and x_4 of the boundaries between each of these zones. This is a typical Riemann problem. It can be solved using the **method of characteristics**, but since we are focussing on numerical hydrodynamics here, we are not going to give the detailed derivation; interested readers are referred to textbooks on this topic. Another useful resource is the paper by Lora-Clavijo et al. 2013, Rev. Mex. de Fisica, 29-50, which gives a detailed description of exact solutions to 1D Riemann problems.

However, even without the method of characteristics, we can use our physical insight developed in these lectures notes to obtain most of the solution. This involves the following steps:

[1] First we realize that we can infer the conditions in region '1' from the known conditions in region 'R' using the **Ranking-Hugoniot jump conditions** for a non-radiative shock. If we refer to the **Mach number** of the shock (to be derived below) as $\mathcal{M}_{\rm s} \equiv u_{\rm s}/c_{\rm s,R}$, with $c_{\rm s,R} = \sqrt{\gamma P_{\rm R}/\rho_{\rm R}}$ the sound speed in region 'R', then we have that

$$\begin{aligned} P_1 &= P_{\rm R}\left[\frac{2\gamma}{\gamma+1}\mathcal{M}_{\rm s}^2 - \frac{\gamma-1}{\gamma+1}\right] \\ \rho_1 &= \rho_{\rm R}\left[\frac{2}{\gamma+1}\frac{1}{\mathcal{M}_{\rm s}^2} + \frac{\gamma-1}{\gamma+1}\right]^{-1} \\ u_1 &= \frac{2}{\gamma+1}\left[\mathcal{M}_{\rm s} - \frac{1}{\mathcal{M}_{\rm s}}\right] \end{aligned}$$

Note that for the latter, one first needs to convert to the rest-frame of the shock, in which the velocities in regions 'R' and '1' are given by $u'_{\rm R} = u_{\rm R} - u_{\rm s} = -u_{\rm s}$ and $u'_1 = u_1 + u_{\rm s}$. One then solves for u'_1 and converts to u_1. Finally, if needed one can infer the temperature T_1 from $T_{\rm R}$ using the corresponding RH jump condition according to which $T_1/T_{\rm R} = (P_1\rho_1/P_{\rm R}\rho_{\rm R})$.

[2] Having established the properties in zone '1', the next step is to infer the properties in zone '2'. Here we use that the velocity and pressure are constant across a contact discontinuity to infer that $P_2 = P_1$ and $u_2 = u_1$. For the density, we need to link it to $\rho_{\rm L}$, which we can do using the fact that rarefraction is an adiabatic process, for which $P \propto \rho^\gamma$. Hence, we have that $\rho_2 = \rho_{\rm L}(P_2/P_{\rm L})^{1/\gamma}$.

[3] What remains is to compute the shock speed, $u_{\rm s}$, or its related Mach number, $\mathcal{M}_{\rm s}$. This step is not analytical, though. Using insight that can be gained from the method of characteristics, not discussed here, one can infer that the Mach number is a solution to the following implicit, non-linear equation, which needs to be solved numerically using a root finder:

$$\mathcal{M}_{\rm s} - \frac{1}{\mathcal{M}_{\rm s}} = c_{\rm s,L}\frac{\gamma+1}{\gamma-1}\left\{1 - \left[\frac{P_{\rm R}}{P_{\rm L}}\left(\frac{2\gamma}{\gamma+1}\mathcal{M}_{\rm s}^2 - \frac{\gamma-1}{\gamma+1}\right)\right]^{\frac{\gamma-1}{2\gamma}}\right\}$$

with $c_{\rm s,L} = \sqrt{\gamma P_{\rm L}/\rho_{\rm L}}$ the sound speed in region 'L'. Once the value of $\mathcal{M}_{\rm s}$ has been

determined, it can be used in steps [1] and [2] to infer all the parameters of (uniform) zones 1 and 2.

[**4**] To determine the internal structure of the rarefraction fan, one once again has to rely on the method of characteristics. Without any derivation, we simply give the solution:

$$\begin{aligned} u(x) &= \frac{2}{\gamma+1}\left(c_{\rm s,L} + \frac{x-x_0}{t}\right) \\ c_{\rm s}(x) &= c_{\rm s,L} - \frac{1}{2}(\gamma-1)u(x) \\ P(x) &= P_{\rm L}\left[\frac{c_{\rm s}(x)}{c_{\rm s,L}}\right]^{\frac{2\gamma}{\gamma-1}} \\ \rho(x) &= \gamma\frac{P(x)}{c_{\rm s}^2(x)} \end{aligned}$$

[**5**] Finally we need to determine the locations of the zone boundaries, indicated by x_1, x_2, x_3 and x_4 (see Fig. 39). The shock wave is propagating with speed $u_{\rm s} = \mathcal{M}_{\rm s}c_{\rm s,R}$. The contact discontinuity is propagating with a speed $u_2 = u_1$. The far left-edge of the rarefaction wave is propogating with the sound-speed in zone L. And finally, from the method of characteristics, one infers that the right-edge of the rarefaction zone is propagating with speed $u_2 + c_{\rm s,2}$ in the positive direction. Hence, we have that

$$\begin{aligned} x_1 &= x_0 - c_{\rm s,L}t & (1)\\ x_2 &= x_0 + (u_2 - c_{\rm s,2})t & (2)\\ x_3 &= x_0 + u_2 t & (3)\\ x_4 &= x_0 + u_{\rm s}t & (4) \end{aligned}$$

which completes the 'analytical' solution to the Sod shock tube problem. Note that the word analytical is in single quotation marks. This is to highlight that the solution is not trully analytical, in that it involves a numerical root-finding step!

Fig. 40 shows this analytical solution at $t = 0.2$ for a Sod shock tube problem with $\gamma = 1.4$ and the following (unitless) initial conditions:

$$\begin{aligned} \rho_{\rm L} = 8.0 \qquad & \rho_{\rm R} = 1.0 \\ P_{\rm L} = 10/\gamma \qquad & P_{\rm R} = 1/\gamma \\ u_{\rm L} = 0.0 \qquad & u_{\rm R} = 0.0 \end{aligned}$$

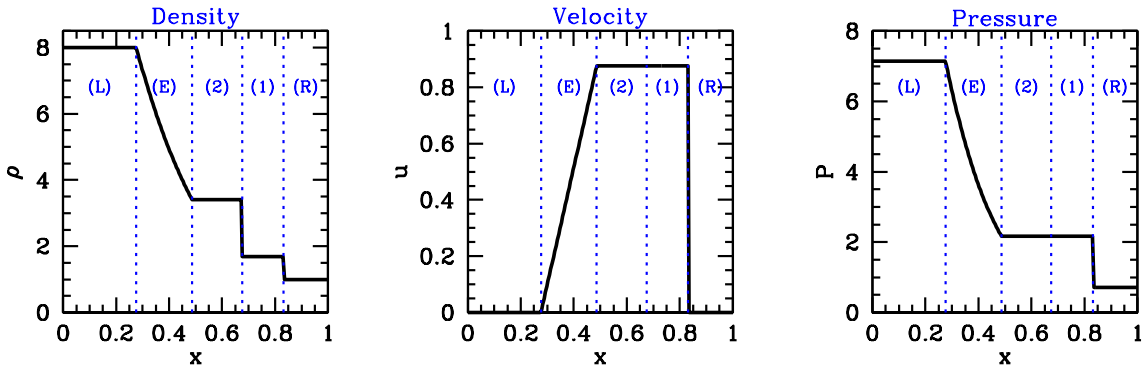


Figure 16: *Analytical solution to the SOD shock tube problem at* $t = 0.2$*. Note that the pressure and velocity are unchanged across the discontinuity (entropy jump) at* x_3*, while the density is clearly discontinuous.*

In what follows, we develop a simple 1D numerical hydro code to integrate this same Sod shock tube problem, which we can then compare with our 'analytical' solution. The code will use different Godunov schemes to do so.

As discussed above, Godunov's method, and its higher order modifications, require solving the Riemann problem at every cell boundary and for every time step. This amounts to calculating the solution in the regions between the left- and right-moving waves (i.e., zones 'E', '1', and '2' in the case of the Sod shock tube), as well as the speeds of the various waves (shock wave(s), rarefraction wave(s), and entropy jumps) involved. The solution of the general Riemann problem cannot be given in a closed analytic form, even for 1D Newtonian flows (recall, that even for the Sod shock tube a numerical root finding step is required). What can be done is to find the answer numerically, to any required accuracy, and in this sense the Riemann problem is said to have been solved exactly, even though the actual solution is not analytical.

However, mainly because of the iterations needed to solve the Riemann problem, the Godunov scheme as originally envisioned by Godunov, which involves using an exact Riemann solver at every cell-interface, is typically far too slow to be practical. For that reason, several **approximate Riemann solvers** have been developed. These can be divided in *approximate-state* Riemann solvers, which use an approximation for the Riemann states and compute the corresponding flux, and *approximate-flux* Riemann solvers, which approximate the numerical flux directly.

Here we highlight one of these approximate Riemann solvers; the **HLL(E) method**, after Harten, Lax & van Leer, who proposed this method in 1989, and which was later improved by Einfeldt (1988). The HLL(E) method is an approximate-flux Riemann solver, which assumes that the Riemann solution consists of just two waves separating three constant states; the original L and R states which border an intermediate 'HLL' state. It is assumed that, after the decay of the initial discontinuity of the local Riemann problem, the waves propagate in opposite directions with velocities $S_{\rm L}$ and $S_{\rm R}$, generating a single state (assumed constant) between them. $S_{\rm L}$ and $S_{\rm R}$ are the smallest and the largest of the signal speeds arising from the solution of the Riemann problem. The simplest choice is to take the smallest and the largest among the eigenvalues of the Jacobian matrix $\partial f_i/\partial q_j$ evaluated at some intermediate (between L and R) state. For the 1D Euler equation that we consider here, one obtains reasonable results if one simply approximates these eigenvalues as $S_{\rm L} = u_{\rm L} - c_{\rm s,L}$ and $S_{\rm R} = u_{\rm R} + c_{\rm s,R}$, where $u_{\rm L}$ and $u_{\rm R}$ are the initial fluid velocities in the L and R states, and $c_{\rm s,L}$ and $c_{\rm s,R}$ are the corresponding sound speeds. Without going into any detail, the HLL(E) flux to be used in the Godunov scheme is given by

$$\boxed{\vec{F}_{i-\frac{1}{2}}^{\,n+\frac{1}{2}} = \frac{S_{\rm R}\vec{f}_{\rm L} - S_{\rm L}\vec{f}_{\rm R} + S_{\rm L}\,S_{\rm R}\,(\vec{q}_{\rm R} - \vec{q}_{\rm L})}{S_{\rm R} - S_{\rm L}}}$$

Here we have made it explicit that the flux is a vector, where each element refers to the corresponding elements of $\vec{q}$ and $\vec{f}(\vec{q})$ of the Euler equation in conservation form. Note that the L and R states here, refer to mesh cells $i-1$ and i, respectiveley. In the case of the $F_{i+\frac{1}{2}}^{n+\frac{1}{2}}$ flux, which is needed in the Godunov scheme together with $F_{i-\frac{1}{2}}^{n+\frac{1}{2}}$, the L and R states refer to mesh cells i and $i+1$.

A simple 1D hydro-code: We are now ready to write our own simple 1D numerical hydro-code (adopting an adiabatic EoS), which we can test against the (analytical) solution of the Sod shock tube problem examined above. What follows are some of the steps that you may want to follow in writing your own code:

- Define an array $q(1:\mathrm{Nx}, 0:\mathrm{Nt}, 1:3)$ to store the discrete values of the vector $\vec{q} = (\rho, \rho u, E)^{\mathrm{t}}$ of conserved quantities on the spatial mesh x_i with $i = 1, .., \mathrm{Nx}$ and at discrete time t^n with $n = 0, 1, ..., \mathrm{Nt}$.

- Write a subroutine that computes the primary variables, $\rho(1:\mathrm{Nx})$, $u(1:\mathrm{Nx})$ and $P(1:\mathrm{Nx})$, given the conserved variables $\vec{q}$. This requires computing the pressure, which follows from $E = P/(\gamma - 1) - \frac{1}{2}\rho u^2$. Also compute the local sound speed $c_{\mathrm{s}}(1:\mathrm{Nx}) = \sqrt{\gamma P/\rho}$ (which is needed in the HLL(E) scheme).

- Write a subroutine that, given the primary variables, computes the time step $\Delta t = \alpha_{\mathrm{c}}(\Delta x/|v^n_{\mathrm{max}}|)$. Here $\alpha_{\mathrm{c}} < 1$ is the user-supplied value for the Courant parameter, and $|v^n_{\mathrm{max}}| = \mathrm{MAX}_i[|u^n_i| + c_{\mathrm{s}}(x_i)]$ denotes the maximum velocity present throughout the entire computational domain at time t^n. Since v_{max} can change with time, this means that different time steps typically adopt a different value for Δt.

- Write a subroutine that, given the array $q(1:\mathrm{Nx}, n, 1:3)$ computes the corresponding fluxes $\vec{f}(\vec{q})$ at time t^n, and store these in $f(1:\mathrm{Nx}, 1:3)$.

- Each time step (i) compute the primary variables, (ii) compute the time step, Δt, (iii) compute the fluxes $f(1:\mathrm{Nx}, 1:3)$, (iv) compute the Godunov fluxes $F^{n+\frac{1}{2}}_{i-\frac{1}{2}}$ and $F^{n+\frac{1}{2}}_{i+\frac{1}{2}}$; (this depends on the scheme used), and (v) update q using the Godunov update scheme:

$$q(i, n+1, 1:3) = q(i, n, 1:3) - \frac{\Delta t}{\Delta x}\left[F^{n+\frac{1}{2}}_{i+\frac{1}{2}}(1:3) - F^{n+\frac{1}{2}}_{i-\frac{1}{2}}(1:3)\right]$$

- Loop over time steps until the total integration time exceeds the user-defined time, and output the mesh of primary variables at the required times.

Fig. 41 shows the outcome of such a program for three different numerical schemes applied to the Sod shock tube problem. All methods start from the same ICs as discussed above (i.e., those used to make Fig. 40), and are propagated forward using time-steps that are computed using a Courant parameter $\alpha_{\mathrm{c}} = 0.8$ until $t = 0.2$. The

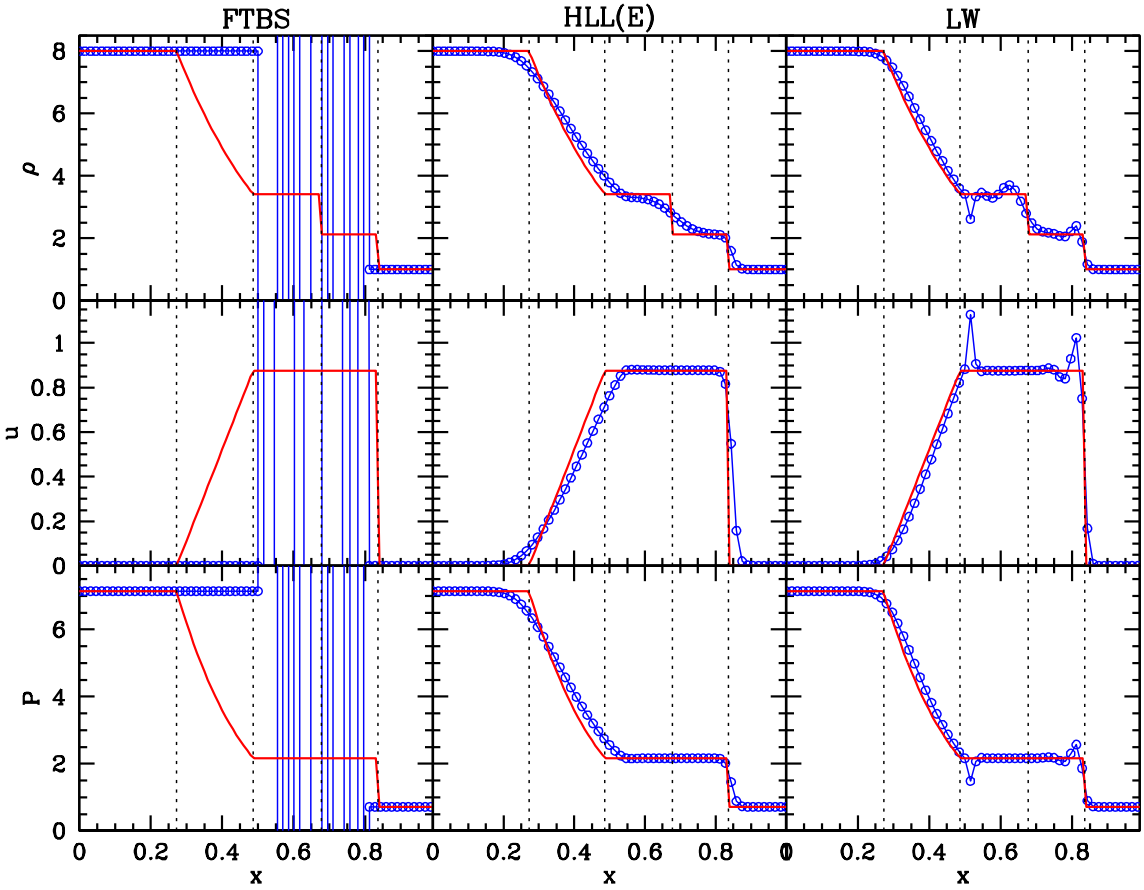


Figure 17: *Numerical integration of Sod's shock tube problem. From top to bottom the panels show the density, velocity and pressure as function of position. The 'analytical' results at* $t = 0.2$ *are indicated in red (these are identical to those shown in Fig. 40). In blue are the results from three different numerical schemes; from left to right, these are the first-order FTBS scheme, the first-order Godunov scheme with the approximate Riemann solver of HLL(E), and the second-order predictor-corrector scheme of Lax-Wendroff. All schemes adopted a spatial grid of 65 points on the domain* $x \in [0, 1]$*, and a Courant parameter* $\alpha_c = 0.8$*.*

results (in blue, open circles) are compared to the analytical solution (in red). In each column, panels from top to bottom show the density, velocity, and pressure.

The first scheme, shown in the left-hand panels, is the standard **Euler FTBS** scheme, which simply sets $F_{i-\frac{1}{2}}^{n+\frac{1}{2}} = f(q_{i-1}^n)$ (cf. Table at the end of Chapter 1). Although this scheme reduces to the stable upwind scheme in the case of the 1D linear advection equation, clearly in this more complicated case it failes miserably. The reason is easy to understand. In the Sod shock tube problem, there are multiple waves moving in different directions (forward moving shock and entropy jump, and a backward moving rarefraction wave). Hence, there is no single direction in the flow, and the FTBS scheme cannot be an upwind scheme for all these waves. For some it is a downwind scheme (equivalent to FTFS), and such a scheme is unconditionally unstable. This explains the drastic failure of this scheme. In the case of the advection equation there is only a single wave, and the FTBS (FTFS) scheme acts as an upwind scheme if $u > 0$ ($u < 0$).

What is needed, therefore, is a Godunov-scheme, which is an upwind scheme for the general, non-linear case. The middle panel shows an example, in which the fluxes are computed using the **HLL(E)** scheme discussed above. This scheme is first-order (i.e., it relies on piecewise constant reconstruction) and as such suffers from numerical diffusion, which is clearly apparent (i.e., the discontinuities in the solution are 'blurred'). Nevertheless, the scheme is stable, and captures the salient features of the analytical solution.

Finally,the right-hand panels show the results for a second-order Godunov scheme, based on the **Lax-Wendroff** fluxes. This scheme uses piecewise linear reconstruction (see Chapter 3), and is second-order in both space and time. The latter arises because this scheme uses a **predictor** and **corrector** step, according to:

$$\vec{q}_{i+\frac{1}{2}} = \frac{\vec{q}_i^n + \vec{q}_{i+1}^n}{2} - \frac{\Delta t}{2\Delta x}\left[\vec{f}(\vec{q}_{i+1}^n) - \vec{f}(\vec{q}_i^n)\right]$$

$$\vec{q}_i^{n+1} = \vec{q}_i^n - \frac{\Delta t}{\Delta x}\left[\vec{f}(\vec{q}_{i+\frac{1}{2}}) - \vec{f}(\vec{q}_{i-\frac{1}{2}})\right]$$

It uses an intermediate step, and it is apparent from combining the two steps that the final update formula is $\mathcal{Q}(\Delta t^2)$. It is left as an exersize for the student to show that this scheme reduces to the LW-scheme highlighted in Chapter 1 for the linear advection equation (i.e., when $f = vu$ with v the constant advection speed and u the

quantity that is advected). The higher-order accuracy of this scheme is better able to capture the discontinuities in the analytical solution of the Sod shock tube, but, as expected from **Godunov's theorem**, the scheme is not positivity conserving and introduces artificial over- and under-shoots.

As we have seen in Chapter 3, such over- and undershoots can be prevented by using slope (or flux) limiters. An example of such a scheme is the **MUSCL** scheme developed by van Leer, which is based on piecewise linear reconstruction combined with slope limiters. This higher-order reconstruction scheme implies that at the cell interfaces, one now has to solve so-called *generalized* Riemann problems; i.e., a discontinuity separating two linear (rather than constant) states. These are not as easy to solve as the standard Riemann problem. Hence, one typically resorts to approximate Riemann solvers.

Before closing our discussion of computational hydrodynamics, we briefly address a few loose ends. The schemes discussed in the previous four chapters are by no means exhaustive. A quick literature study will reveal many more schemes, combining higher-order reconstruction schemes with a wide variety of limiters (to assure TVD) and approximate Riemann solvers. Another topic we haven't really discussed is that of source terms. In our discussion of computational hydrodynamics we have restricted our discussion to **homogeneous** conservation laws. However, most astrophysical situations involve source and sink terms, especially in the form of gravity and radiation. Adding these terms to the hydro-equations makes the conservation laws **inhomogeneous**. How to deal with this numerically is not trivial, and an ongoing topic of investigation. The most obvious inclusion of a source term into a Godunov scheme adds extra complications to the Riemann problem and calculation of the corresponding intercell fluxes. The reader interested in learning more about these topics is referred to the excellent textbook *Riemann Solvers and Numerical Methods for Fluid Dynamics* by E.F. Toro, which presents a detailed discussion of the material covered here and much more, including alternative approximate Riemann solvers such as those by Roe and Osher, Flux Vector Splitting methods, higher-order TVD schemes, methods to include source terms, and the extension to multiple dimensions.

Finally, as already eluded to in Chapter 1, we have focussed exclusively on Eulerian schemes. Many hydro-simulations in astrophysics make use of the Lagrangian SPH method. It is left as an exersize for the interested reader to research this different methodology. Hopefully the material provided here will facilitate such a self-study.

Appendix

Differential Equations

The equations of fluid dynamics are all differential equations. In order to provide the necessary background, this appendix gives a very brief overview of the basics.

A differential equation is an **ordinary differential equation (ODE)** if the unknown function depends on only 1 independent variable.

If the unknown function depends on two or more independent variables, then the differential equation is a **partial differential equation (PDE)**.

The **order** of a differential equation is that of the highest derivative appearing in the equation.

Consider the following examples:

$$\begin{aligned} [\mathbf{a}] \quad & \tfrac{\mathrm{d}u}{\mathrm{d}x} = 2x^2 - 4 \\ [\mathbf{b}] \quad & \mathrm{e}^u \tfrac{\mathrm{d}^2u}{\mathrm{d}x^2} + 3\left(\tfrac{\mathrm{d}u}{\mathrm{d}x}\right)^4 = 2 \\ [\mathbf{c}] \quad & \tfrac{\partial^2 u}{\partial t^2} - 4\tfrac{\partial^2 u}{\partial x^2} = 0 \end{aligned}$$

Equation [**a**] is an ODE of order 1, equation [**b**] is an ODE of order 2, and equation [**c**] is a PDE of order 2.

In what follows we shall often use the following **shorthand notation:**

$$u' \equiv \frac{\mathrm{d}u}{\mathrm{d}x}\,, \qquad u'' \equiv \frac{\mathrm{d}^2u}{\mathrm{d}x^2}\,, \qquad u^{(n)} \equiv \frac{\mathrm{d}^n u}{\mathrm{d}x^n}\,.$$

When the independent variable is time, we often use a dot rather than a hyphen, i.e., $\dot{u} = \mathrm{d}u/\mathrm{d}t$, $\ddot{u} = \mathrm{d}^2u/\mathrm{d}t^2$, etc.

When dealing with PDEs, we use the following shorthand:

$$u_{,x} \equiv \frac{\partial u}{\partial x}\,, \qquad u_{,xy} \equiv \frac{\partial^2 u}{\partial x \partial y}\,, \qquad u_{,tt} \equiv \frac{\partial^2 u}{\partial^2 t}\,,$$

etc. Consider the following examples

$$\begin{aligned} \nabla^2 u = 0 \quad &\leftrightarrow \quad u_{,xx} + u_{,yy} + u_{,zz} = 0 \\ \nabla(\nabla \cdot u) + \nabla^2 u + u = 0 \quad &\leftrightarrow \quad u_{k,ki} + u_{i,jj} + u_i = 0 \end{aligned}$$

Note that in the latter we have adopted the Einstein summation convention.

A differential equation along with subsidiary conditions on the unknown function and its derivatives, all given at the same value of the independent variable, consistitute an **initial value problem**.

If the subsidiary conditions are given at more than one value of the independent variable, the problem is a **boundary value problem** and the conditions are **boundary conditions**.

There are three broad classes of boundary conditions:

- **Dirichlet boundary conditions:** The value of the dependent variable is specified on the boundary.
- **Neumann boundary conditions:** The normal derivative of the dependent variable is specified on the boundary.
- **Cauchy boundary conditions:** Both the value and the normal derivative of the dependent variable are specified on the boundary.

Cauchy boundary conditions are analogous to the initial conditions for a second-order ODE. These are given at one end of the interval only.

Linear and non-linear PDEs: A linear PDE is one that is of first degree in all of its field variables and partial derivatives.

Consider the following examples:

$$\begin{aligned}
&[\mathbf{a}] \quad \tfrac{\partial u}{\partial x} + \tfrac{\partial u}{\partial y} = 0 \\
&[\mathbf{b}] \quad \tfrac{\partial u}{\partial x} + \left(\tfrac{\partial u}{\partial y}\right)^2 = 0 \\
&[\mathbf{c}] \quad \tfrac{\partial^2 u}{\partial x^2} + \tfrac{\partial^2 u}{\partial y^2} = x + y \\
&[\mathbf{d}] \quad \tfrac{\partial u}{\partial x} + \tfrac{\partial u}{\partial y} = u^2 \\
&[\mathbf{e}] \quad \tfrac{\partial^2 u}{\partial x^2} + u\,\tfrac{\partial^2 u}{\partial y^2} = 0
\end{aligned}$$

Equations **[a]** and **[c]** are linear, while **[b]**, **[d]** and **[e]** are all non-linear.

We can write the above equations in **operator notation** as:

$$\begin{aligned}
&[\mathbf{a}] \quad L(u) = 0 \quad \text{with} \quad L := \frac{\partial}{\partial x} + \frac{\partial}{\partial y} \\
&[\mathbf{b}] \quad L(u) = 0 \quad \text{with} \quad L := \frac{\partial}{\partial x} + \left(\frac{\partial}{\partial y}\right)^2 \\
&[\mathbf{c}] \quad L(u) = x + y \quad \text{with} \quad L := \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2} \\
&[\mathbf{d}] \quad L(u) = 0 \quad \text{with} \quad L := \frac{\partial}{\partial x} + \frac{\partial}{\partial y} - u^2 \\
&[\mathbf{e}] \quad L(u) = 0 \quad \text{with} \quad L := \frac{\partial^2}{\partial x^2} + u\,\frac{\partial^2}{\partial y^2} = 0
\end{aligned}$$

Homogeneous and non-homogeneous PDEs: Let L be a **linear** operator. Then, a linear PDE can be written in the form

$$L(u) = f(x_1, x_2, x_3, ..., x_n, t)$$

The PDE is said to be homogeneous iff $f(x_1, x_2, x_3, ..., x_n, t) = 0$. Thus, in the examples above, equation **[a]** is homogeneous, while **[c]** is non-homogeneous (aka inhomogeneous).

In (hydro-)dynamics, we typically encounter three types of second-order PDEs, classified as **elliptic**, **hyperbolic**, and **parabolic**. Each type has certain characteristics that help determine if a particular finite element approach is appropriate to the problem being described by the PDE. Interestingly, just knowing the type of PDE can give us insight into how smooth the solution is, how fast information propagates, and the effect of initial and boundary conditions.

Consider a second-order PDE for the unknown function $u(x, y)$ of the form

$$a\,u_{,xx} + b\,u_{,xy} + c\,u_{,yy} + d\,u_{,x} + e\,u_{,y} + f\,u + g = 0$$

where each of a, b,...,g are allowed to be functions of x and/or y.

Elliptic: The above PDE is called elliptic if $b^2 - 4ac < 0$.
An example is the 2D **Poisson equation** $u_{,xx} + u_{,yy} = f$ (which has $a = c = 1$ and $b = 0$). The solutions of elliptic PDEs are always smooth, and boundary data at any point affect the solution at all points in the domain. There is no temporal propagation, yet elliptic PDEs convey the effect of objects on each other. Newtonian mechanics is an example of this, which is why the Poisson equation is elliptic.

Parabolic: The above PDE is called parabolic if $b^2 - 4ac = 0$.
An example is the **heat equation** $u_{,t} = u_{,xx}$ (which has $a = 1$ and $b = c = 0$) which describes heat flow in a 1D system. Parabolic PDEs are usually time dependent and represent diffusion-like processes (i.e., dissipation, convection). Solutions are smooth in space but may possess singularities.

Hyperbolic: The above PDE is called hyperbolic if $b^2 - 4ac > 0$.
An example is the **wave equation** $u_{,xx} - \frac{1}{c_s^2}\,u_{,tt} = f$ (which has $b = 0$, $a = 1$ and $c = -1/c_s^2 < 0$). In a system modeled with a hyperbolic PDE, information travels at a finite speed referred to as the wavespeed (c_s in the example here). Information is not transmitted until the wave arrives. The smoothness of the solution to a hyperbolic PDE depends on the smoothness of the initial and boundary conditions. For instance, if there is a jump in the data at the start or at the boundaries, then the jump will propagate as a shock in the solution. If, in addition, the PDE is nonlinear, then shocks may develop even though the initial conditions and the boundary conditions are smooth.

Finally, since solving PDEs can often be reduced to solving (sets) of ODEs, a few words about solving the latter. Problems involving ODEs can always be reduced to a set of first-order ODEs! For example, the 2nd order ODE

$$\frac{\mathrm{d}^2u}{\mathrm{d}x^2} + s(x)\,\frac{\mathrm{d}u}{\mathrm{d}x} = t(x)$$

can be rewritten as two first-order ODEs

$$\frac{\mathrm{d}u}{\mathrm{d}x} = v(x)\,, \qquad \frac{\mathrm{d}v}{\mathrm{d}x} = t(x) - s(x)v(x)$$

Consider the general n^{th}-order initial value problem

$$\frac{\mathrm{d}^nu}{\mathrm{d}x^n} = a_{n-1}(x)\,\frac{\mathrm{d}^{n-1}}{\mathrm{d}x^{n-1}} + \ldots\ldots + a_1(x)\,\frac{\mathrm{d}u}{\mathrm{d}x} + a_0(x)\,u(x) + f(x)$$

with $u(0) = c_0$, $u'(0) = c_1$, $u''(0) = c_2$, ... ,$u^{(n-1)}(0) = c_{n-1}$ as the initial values. In general, this can be written in matrix form as

$$\mathbf{u}' = \mathbf{A}(x)\,\mathbf{u}(x) + \mathbf{f}(x)$$

with the initial values given by $\mathbf{u}(0) = \mathbf{c}$. Here the elements of $\mathbf{u}$ are given by $u_1 = u(x)$, $u_2 = u'(x)$, ..., $u_n = u^{(n-1)}(x)$. Theses are interrelated with the elements of $\mathbf{u}'$ by the equations $u_1' = u_2$, $u_2' = u_3$, ... ,$u_{n-1}' = u_n$, $u_n' = u^{(n)}(x)$. The matrices $\mathbf{A}(x)$ and $\mathbf{f}(x)$ are related to $a_i(x)$ and $f(x)$ according to

$$\mathbf{A}(x) = \begin{pmatrix} 0 & 1 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & 0 & 0 & \cdots & 1 \\ a_0(x) & a_1(x) & a_2(x) & a_3(x) & \cdots & a_{n-1}(x) \end{pmatrix}$$

and

$$\mathbf{f}(x) = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ f(x) \end{pmatrix}$$

Hence, solving an ODE of order N reduces to solving a set of N coupled first-order differential equations for the functions u_i $(i = 1, 2, ..., N)$ having the general form

$$\frac{\mathrm{d}u_i}{\mathrm{d}x} = f_i(x, u_1, u_2, ..., u_n)$$

where the functions f_i on the rhs are known.