
Finite Volume Numerical Methods for Hydrodynamics.

I - Discretization techniques for linear hyperbolic PDE



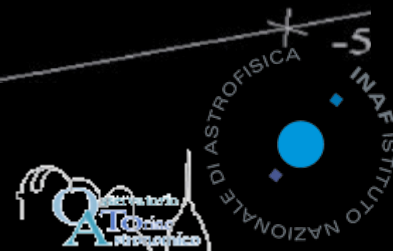
A. Mignone

Physics Department, Turin University

AU

UNIVERSITÀ
DEGLI STUDI
DI TORINO

ALMA UNIVERSITAS
TAURINENSIS



Lecture I - Outline

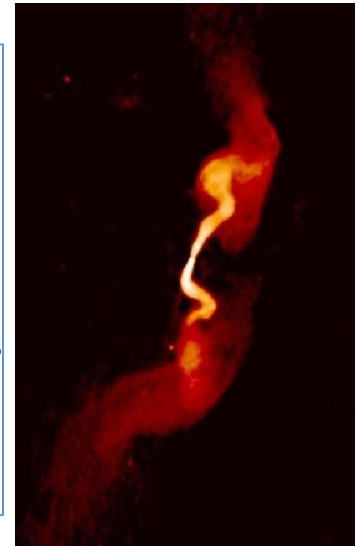
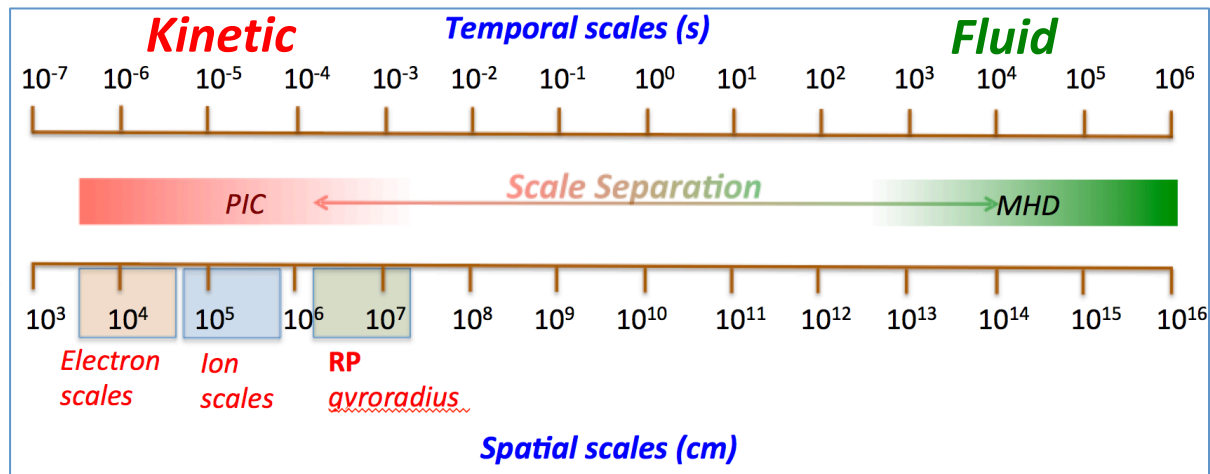
1. Scale separation in plasma astrophysics:
from kinetic to fluid description;
2. Basic discretization methods for hyperbolic PDEs;
3. Linear scalar hyperbolic PDEs;
4. Systems of linear hyperbolic PDEs
5. Nonlinear scalar PDE: Burger's equation;
6. Nonlinear systems - the Euler equations.

1. SCALE SEPARATION:

APPROACHING PLASMA ASTROPHYSICS AT DIFFERENT SCALES

Astrophysical Challenges: Scale Separation

- Astrophysical environments involve physical processes operating at *extremely different spatial* and *temporal scales*, and complex *interactions* between *plasmas* and *radiation*.
- Current computational *modeling* is still *largely fragmented* under the limited range of applicability of different models.



- A **large gap** stretches from theory to a clear interpretation of the observations of high-energy astrophysical sources.

The Scale Connection

- **Kinetic Description:** PIC codes are applicable to study small-scale kinetic effects:

$L \approx 10^4$ Km, $t \approx 10^1$ - 10^2 sec.

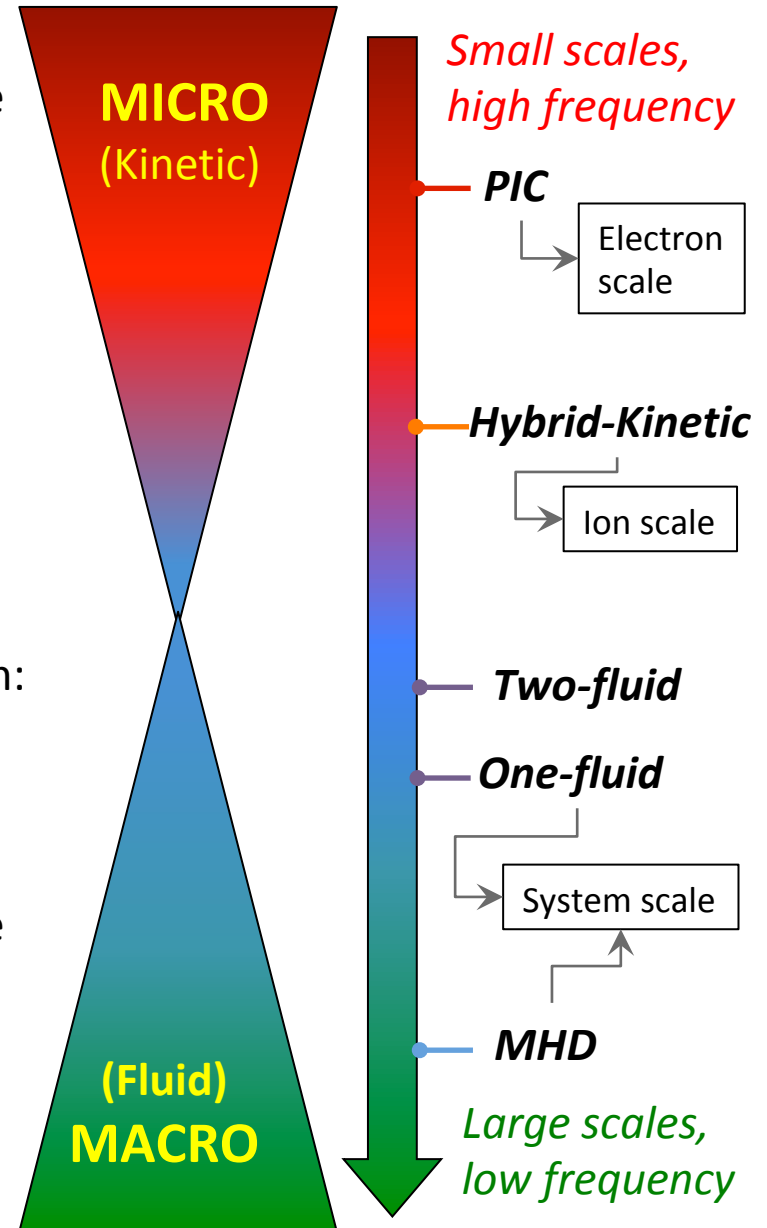
For typical astrophysical applications, these scales are several orders of magnitude smaller than the system size.

- **Hybrid Kinetic Models** (kinetic ions / electron fluid): must resolve the ion inertial length:

$L \leq 1$ AU, $t \approx 1$ hr.

- **Fluid models:** best approach to deal with large scale \rightarrow with **Magnetohydrodynamics** (MHD) only magnetic / light waves must be resolved \rightarrow

$L \approx$ Kpc, $t \approx 10^5$ yrs.



Classical Description

- Classical description:

$$m_i \ddot{\mathbf{r}}_i = e_i \left(\mathbf{E} + \frac{1}{c} \dot{\mathbf{r}}_i \times \mathbf{B} \right)$$

Individual particle motion

$$q(\mathbf{r}, t) = \sum_{i=1}^N e_i \delta[\mathbf{r} - \mathbf{r}_i(t)]$$

Charge and currents

$$\mathbf{J}(\mathbf{r}, t) = \sum_{i=1}^N e_i \mathbf{v} \delta[\mathbf{r} - \mathbf{r}_i(t)] \delta[\mathbf{v} - \mathbf{v}_i(t)]$$

$$\nabla \times \mathbf{E} = -\frac{1}{c} \frac{\partial \mathbf{B}}{\partial t}$$

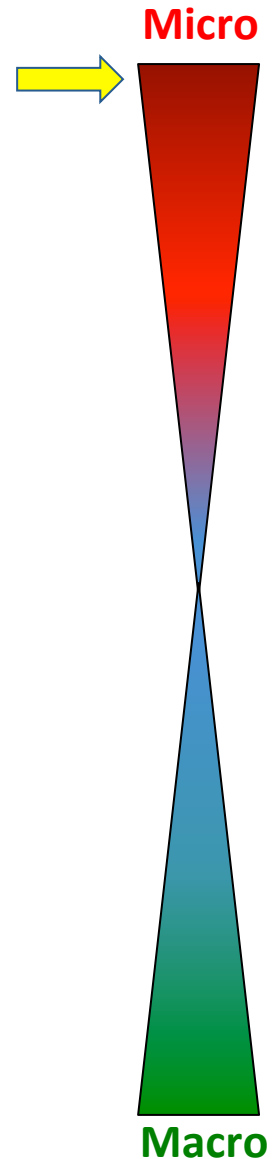
Maxwells' Equations

$$\nabla \times \mathbf{B} = \frac{1}{c} \frac{\partial \mathbf{E}}{\partial t} + \frac{4\pi}{c} \mathbf{J}$$

$$\nabla \cdot \mathbf{E} = 4\pi q$$

$$\nabla \cdot \mathbf{B} = 0$$

→ Not feasible !
(too many degrees of freedom)



Kinetic Description

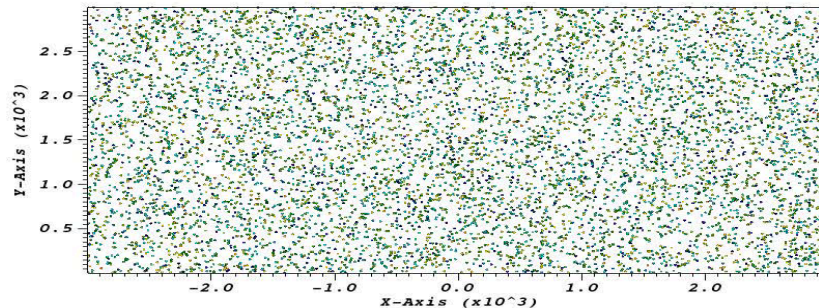
- Kinetic Description:

$$\frac{\partial f}{\partial t} + \mathbf{v} \cdot \nabla f + \frac{e_0}{m} (\mathbf{E} + \frac{1}{c} \mathbf{v} \times \mathbf{B}) \cdot \nabla_{\mathbf{v}} f = 0.$$

Vlasov Equation: $f(\mathbf{x}, \mathbf{v}, t)$ is the distribution function (for a given species) giving the number density per unit element of phase space

- Particle In Cell: (PIC) methods based on a *finite element approach*, but with moving and overlapping elements. Distribution function given by the superposition of several elements (“superparticles”):

$$f_s(x, v, t) = \sum_p f_p(x, v, t)$$

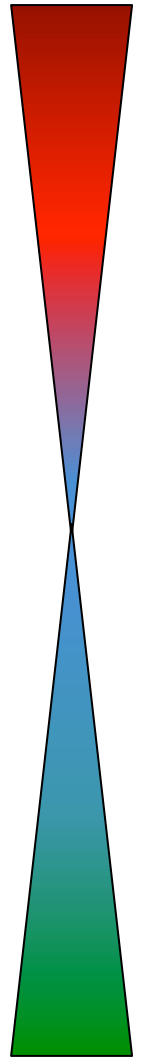


- Each element represents a large number of physical particles that are

Most consistent approach, but must resolve the plasma (electron) skin depth: $c/\omega_{pe} \sim 5.4 \times 10^5 \text{ cm } (n/\text{cm}^{-3})^{-1/2}$

Micro

Macro



The Fluid Approach

- In many astrophysical environments, the distribution function $f(x, v, t)$ is not a measurable quantity.
- The numerical solution of Vlasov-like equations presents huge difficulties since it involves six degrees of freedom;
- If we focus on the behavior of the system in ordinary space and not in the whole phase space, we may give up the information concerning the distribution of the velocities.
- This is achieved by averaging over the velocities themselves:

$$n(x, t) = \int f(x, v, t) d^3v \quad \rightarrow \quad \text{number density}$$

$$u(x, t) = \frac{1}{n} \int v f(x, v, t) d^3v \quad \rightarrow \quad \text{fluid velocity}$$

$$E(x, t) = \int \frac{1}{2} m v^2 f(x, v, t) d^3v \quad \rightarrow \quad \text{energy density}$$

The Fluid Approach

- The fluid approach treats the system as a continuous medium and considering the dynamics of a small volume of the fluid.
- Meaningful to model length scales much greater than mean free path or individual particle trajectories.
- “Fluid element”: small enough that any macroscopic quantity has a negligible variation across its dimension but large enough to contain many particles and so to be insensitive to particle fluctuations.
- Fluid equations involve only moments of the distribution function relating mean quantities. Knowledge of $f(x,v,t)$ is not needed*.
- Still: taking moments of the Vlasov equation lead to the appearance of a next higher order moment \rightarrow “loose end” \rightarrow Closure.

Two Fluid Description

- Fluid description: obtained by averaging distribution function over momentum space. Valid for $L \gg \lambda_{\text{mfp}}$ requiring the solution of highly nonlinear hyperbolic / parabolic P.D.E.

Two-Fluid Equations

$$\frac{\partial n_i}{\partial t} + \frac{1}{e} \nabla \cdot \mathbf{j}_i = 0$$

$$\frac{\partial n_e}{\partial t} - \frac{1}{e} \nabla \cdot \mathbf{j}_e = 0.$$

Continuity

$$n_i m_i \left[\frac{\partial \mathbf{v}_i}{\partial t} + (\mathbf{v}_i \cdot \nabla) \mathbf{v}_i \right] = -\nabla p_i + n_i e (\mathbf{E} + \mathbf{v}_i \times \mathbf{B}) + \mathbf{R}_{ei},$$

$$n_e m_e \left[\frac{\partial \mathbf{v}_e}{\partial t} + (\mathbf{v}_e \cdot \nabla) \mathbf{v}_e \right] = -\nabla p_e - n_e e (\mathbf{E} + \mathbf{v}_e \times \mathbf{B}) - \mathbf{R}_{ei},$$

Momentum

$$\frac{1}{\gamma - 1} n_i \left[\frac{\partial T_i}{\partial t} + (\mathbf{v}_i \cdot \nabla) T_i \right] = -p_i \nabla \cdot \mathbf{v}_i,$$

$$\frac{1}{\gamma - 1} n_e \left[\frac{\partial T_e}{\partial t} + (\mathbf{v}_e \cdot \nabla) T_e \right] = -p_e \nabla \cdot \mathbf{v}_e,$$

Energy

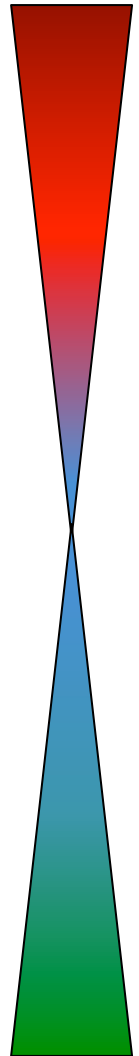
$$\frac{\partial \mathbf{B}}{\partial t} = -\nabla \times \mathbf{E},$$

$$\epsilon_o \frac{\partial \mathbf{E}}{\partial t} = \nabla \times \mathbf{B} / \mu_o + \mathbf{j}_i + \mathbf{j}_e.$$

Maxwell

Challenge: must resolve very separate scales (m_p/m_e)

Micro



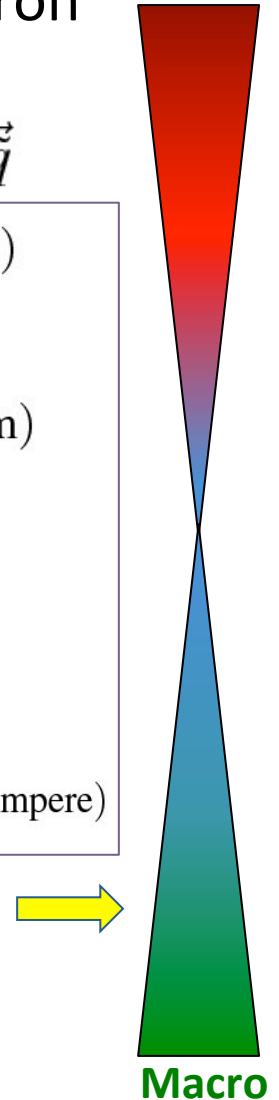
Macro

One-Fluid Model

- One Fluid description: based on averaging ions and electron equations.
- Set of 15 equations in 21 unknowns $\rho, P, q, \vec{U}, \vec{J}, \vec{E}, \vec{B}, \Pi, \vec{q}$

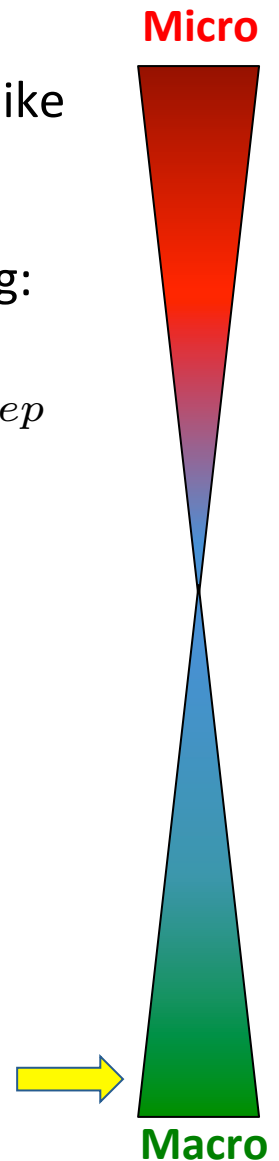
| | |
|---|--------------------|
| $\partial_t \rho + \partial_k (\rho U_k) = 0$ | (Continuity) |
| $\partial_t q + \partial_k J_k = 0$ | (Charge) |
| $\partial_t (\rho U_i) + \partial_k (\rho U_i U_k + P_{ik}) = q E_i + \frac{1}{c} (\vec{J} \times \vec{B})_i$ | (Momentum) |
| $\partial_t \left(\frac{1}{2} \rho U^2 + \frac{3}{2} P \right) + \partial_k \left[\left(\frac{1}{2} \rho U^2 + \frac{5}{2} P \right) U_k + \Pi_{ik} U_i + \tilde{q}_k + S_k \right] = 0$ | (Energy) |
| $E_i + \frac{1}{c} (\vec{U} \times \vec{B})_i - \frac{J_i}{\sigma} = \frac{m_e}{e^2 n_e} [\partial_t J_i + \partial_j (J_i U_k + J_k U_i)] - \frac{1}{n_e} \partial_k P_{ik}^{(e)} + \frac{1}{en_e c} (\vec{J} \times \vec{B})_i$ | (Ohm) |
| $\frac{1}{c} \partial_t \vec{B} = -\nabla \times \vec{E}$ | (Faraday) |
| $\frac{1}{c} \partial_t \vec{E} = \nabla \times \vec{B} - \frac{4\pi}{c} \vec{J}$ | (Maxwell – Ampere) |

- Closure must be found to express Π, \vec{q} in terms of macroscopic quantities. \mathbf{v}



MHD at Last !

- *MagnetoHydroDynamics* (MHD) treats the plasma like a conducting fluid and assigning macroscopic parameters to describe its particle-like interactions.
- Ideal MHD describes an electrically conducting single fluid, assuming:
 - *low frequency* $\omega \ll \omega_p, \quad \omega \ll \omega_c, \quad \omega \ll \nu_{pe}, \quad \omega \ll \nu_{ep}$
 - *large scales* $L \gg \frac{c}{\omega_p}, \quad L \gg R_c, \quad L \gg \lambda_{mfp},$
 - *Ignores electron mass* and finite Larmor radius effects;
 - Assume plasma is *strongly collisional* \rightarrow L.T.E., isotropy;
 - *Fields* and *fluid* fluctuate on the *same time* and *length scales*;
 - Neglect charge separation, electric force and displacement current.



Ideal MHD at Last

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{u}) = 0 \quad \text{≡ Continuity (Mass cons.)}$$

$$\rho \left(\frac{\partial \mathbf{u}}{\partial t} + \nabla \cdot [\mathbf{u} \mathbf{u}] \right) - \frac{\mathbf{B} \mathbf{B}}{4\pi} + \left(\nabla p + \frac{\mathbf{B}^2}{8\pi} \right) \times \mathbf{B} \quad \text{≡ Eq of Motion (Momentum cons.)}$$

$$\frac{\partial (E_{pe})}{\partial t} + \nabla \cdot \left[\left(E_{pe} \mathbf{u} + \frac{\mathbf{B}^2}{8\pi} \right) \mathbf{u} - \frac{(\mathbf{u} \cdot \mathbf{B})}{4\pi} \mathbf{B} \right] \quad \text{≡ Thermodynamic Law (Energy cons.)}$$

$$\frac{\partial \mathbf{B}}{\partial t} + \nabla \cdot (\mathbf{u} \mathbf{B} - \mathbf{B} \mathbf{u}) = 0 \quad \text{≡ Faraday (Mag. flux cons.)}$$

- MHD suitable for describing plasma at large scales;

- Good first approximation to much of the physics, even when some of the conditions are not met.

$$\mathbf{J} = \frac{c}{4\pi} \nabla \times \mathbf{B} \quad \text{(Ampere)}$$

$$\mathbf{E} + \frac{\mathbf{u}}{c} \times \mathbf{B} = 0 \quad \text{(Ohm)}$$

$$\nabla \cdot \mathbf{B} = 0 \quad \text{(Divergence - free)}$$

$$\rho_e = \rho_e(\rho, p) \quad \text{(EoS/Closure)}$$

- Draw some intuitive conclusions concerning plasma behavior without solving the equations in detail

- Fluid equations (except closure) are exact conservation laws;

Classification of PDEs

- Hyperbolic:
 - model the transport of some physical quantities; typically associated with wave propagation at finite speed.
- Parabolic:
 - model diffusion processes (infinite propagation speed): viscosity, thermal conduction, resistivity, radiation hydrodynamics, etc....
- Euler or MHD equations including viscous drag, thermal conduction or resistivity are of mixed type (hyperbolic/parabolic).
- Stable numerical discretization must be consistent with the nature of the underlying equations.

2. BASIC DISCRETIZATION METHODS FOR HYPERBOLIC PDE

Numerical Discretizations

- We consider our prototype first-order partial differential equation (PDE):

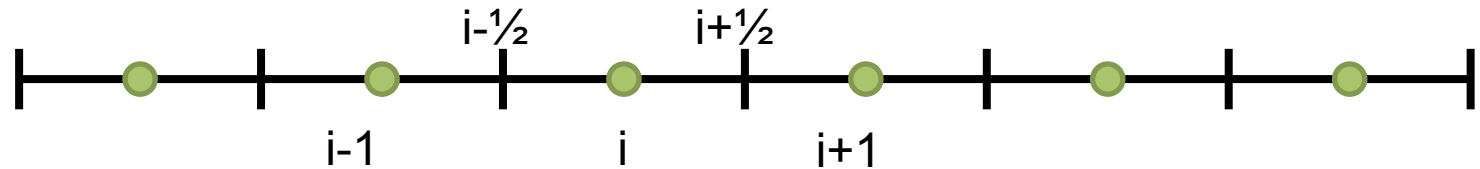
$$\frac{\partial U}{\partial t} + \frac{\partial F(U)}{\partial x} = 0$$

also known as a “Conservation Law”.

- Two popular methods for performing discretization:
 - Finite Differences (FD);
 - Finite Volumes (FV);
- For some problems, the resulting discretizations look identical, but they are distinct approaches;

Finite Difference Methods

- A finite-difference method stores the solution at specific points in space and time;



- Associated with each grid point is a function value,

$$U_i^n \equiv U(x_i, t^n)$$

- We replace the derivatives in our PDE with differences between neighbour points.

Finite Difference Methods

- From Taylor expansion of the function around (x_i, t^n) we obtain, e.g.
 - Forward derivative (in time):

$$\frac{\partial U(x, t)}{\partial t} = \frac{U_i^{n+1} - U_i^n}{\Delta t} - \frac{\Delta t}{2} \left(\frac{\partial^2 U}{\partial t^2} \right)^n + H.O.T.$$

or simply

$$\frac{\partial U(x, t)}{\partial t} \approx \frac{U_i^{n+1} - U_i^n}{\Delta t} + O(\Delta t)$$

- Central derivative (in space):

$$\frac{\partial U(x, t)}{\partial x} = \frac{U_{i+1}^n - U_{i-1}^n}{2\Delta x} - \frac{\Delta x^2}{6} \left(\frac{\partial^3 U}{\partial x^3} \right)_i + H.O.T.$$

or simply

$$\frac{\partial U(x, t)}{\partial x} \approx \frac{U_{i+1}^n - U_{i-1}^n}{2\Delta x} + O(\Delta x^2)$$

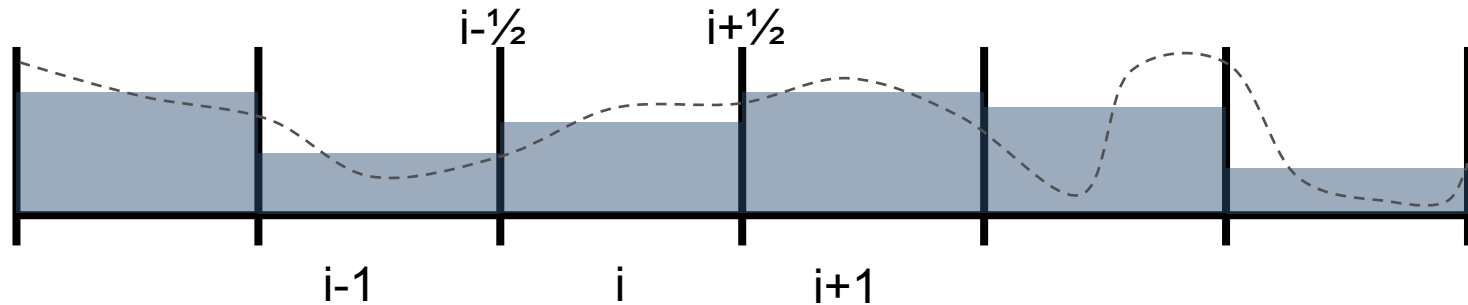
Truncation errors

Finite Volume Methods

- In a finite volume discretization, the unknowns are the spatial averages of the function itself:

$$\langle U \rangle_i^n = \frac{1}{\Delta x} \int_{x_{i-\frac{1}{2}}}^{x_{i+\frac{1}{2}}} U(x, t^n) dx$$

where $x_{i-\frac{1}{2}}$ and $x_{i+\frac{1}{2}}$ denote the location of the cell interfaces.



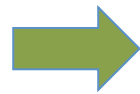
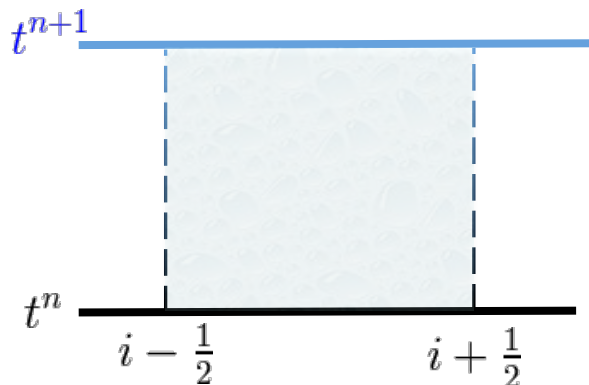
- The solution to the conservation law involves computing fluxes through the boundary of the control volumes

Finite Volume Formulation

- The *conservative form* of the equations provides the link between the *differential* form of the equation,

$$\frac{\partial U}{\partial t} + \frac{\partial F}{\partial x} = 0$$

and the *integral* form, obtained by integrating the equations over a time interval $\Delta t = t^{n+1} - t^n$ and cell size $\Delta x = x_{i+1/2} - x_{i-1/2}$:



$$\int_{t^n}^{t^{n+1}} \int_{x_{i-1/2}}^{x_{i+1/2}} \left(\frac{\partial U}{\partial t} + \frac{\partial F}{\partial x} \right) dt dx = 0$$

Finite Volume Formulation

- Spatial integration yields

$$\int_{t^n}^{t^{n+1}} \left[\Delta x \frac{d}{dt} \langle U \rangle_i + \left(F_{i+\frac{1}{2}} - F_{i-\frac{1}{2}} \right) \right] dt = 0$$

with $\langle U \rangle_i = \frac{1}{\Delta x} \int_{x_{i-\frac{1}{2}}}^{x_{i+\frac{1}{2}}} U(x, t) dx$ being a spatial average.

- Integration in time gives

$$\Delta x \left(\langle U \rangle_i^{n+1} - \langle U \rangle_i^n \right) + \Delta t \left(\tilde{F}_{i+\frac{1}{2}}^{n+\frac{1}{2}} - \tilde{F}_{i-\frac{1}{2}}^{n+\frac{1}{2}} \right) = 0$$

where $\tilde{F}_{i\pm\frac{1}{2}}^{n+\frac{1}{2}} = \frac{1}{\Delta t} \int_{t^n}^{t^{n+1}} F \left(U(x_{i\pm\frac{1}{2}}) \right) dt$ is a temporal average.

Finite Volume Formulation

- Rearranging terms:

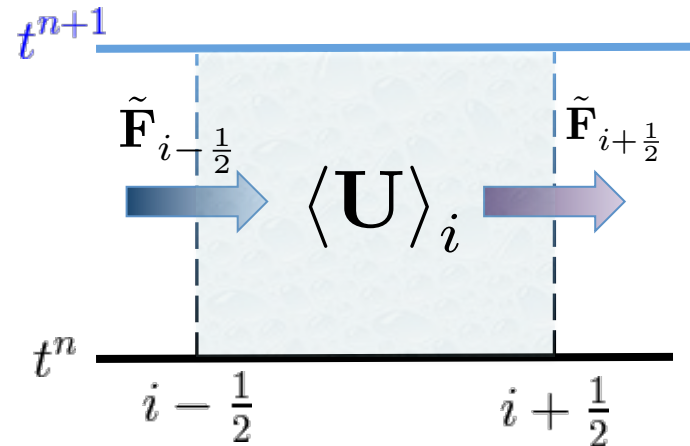
$$\langle U \rangle_i^{n+1} = \langle U \rangle_i^n - \frac{\Delta t}{\Delta x} \left(\tilde{F}_{i+\frac{1}{2}}^{n+\frac{1}{2}} - \tilde{F}_{i-\frac{1}{2}}^{n+\frac{1}{2}} \right)$$

Integral or Conservation form

where

$$\langle U \rangle_i^n = \frac{1}{\Delta x} \int_{x_{i-\frac{1}{2}}}^{x_{i+\frac{1}{2}}} U(x, t^n) dx$$

$$\tilde{F}_{i\pm\frac{1}{2}}^{n+\frac{1}{2}} = \frac{1}{\Delta t} \int_{t^n}^{t^{n+1}} F \left(U(x_{i\pm\frac{1}{2}}, t) \right) dt$$



- The conservation form is an exact relation, no approximation introduced;
- It provides an *integral* representation of the original differential equation.
- The integral form does not make use of partial derivatives!

Importance of Conservation Form

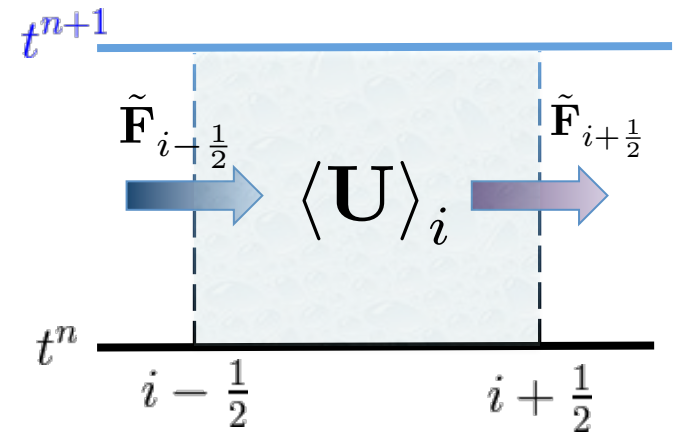
$$\langle U \rangle_i^{n+1} = \langle U \rangle_i^n - \frac{\Delta t}{\Delta x} \left(\tilde{F}_{i+\frac{1}{2}}^{n+\frac{1}{2}} - \tilde{F}_{i-\frac{1}{2}}^{n+\frac{1}{2}} \right)$$

- The conservation form ensure correct description of discontinuous waves in terms of speed and jumps;
- It guarantees global conservation properties (no mass / energy / momentum is created or destroyed unless a net flux exists);
- To second-order accuracy, a *finite difference* method and a *finite volume* method look essentially the same;
- Approximation introduced in the computation of the flux.

Flux computation: the Riemann Problem

- Since the solution is known only at t^n , some kind of approximation is required in order to evaluate the flux through the boundary:

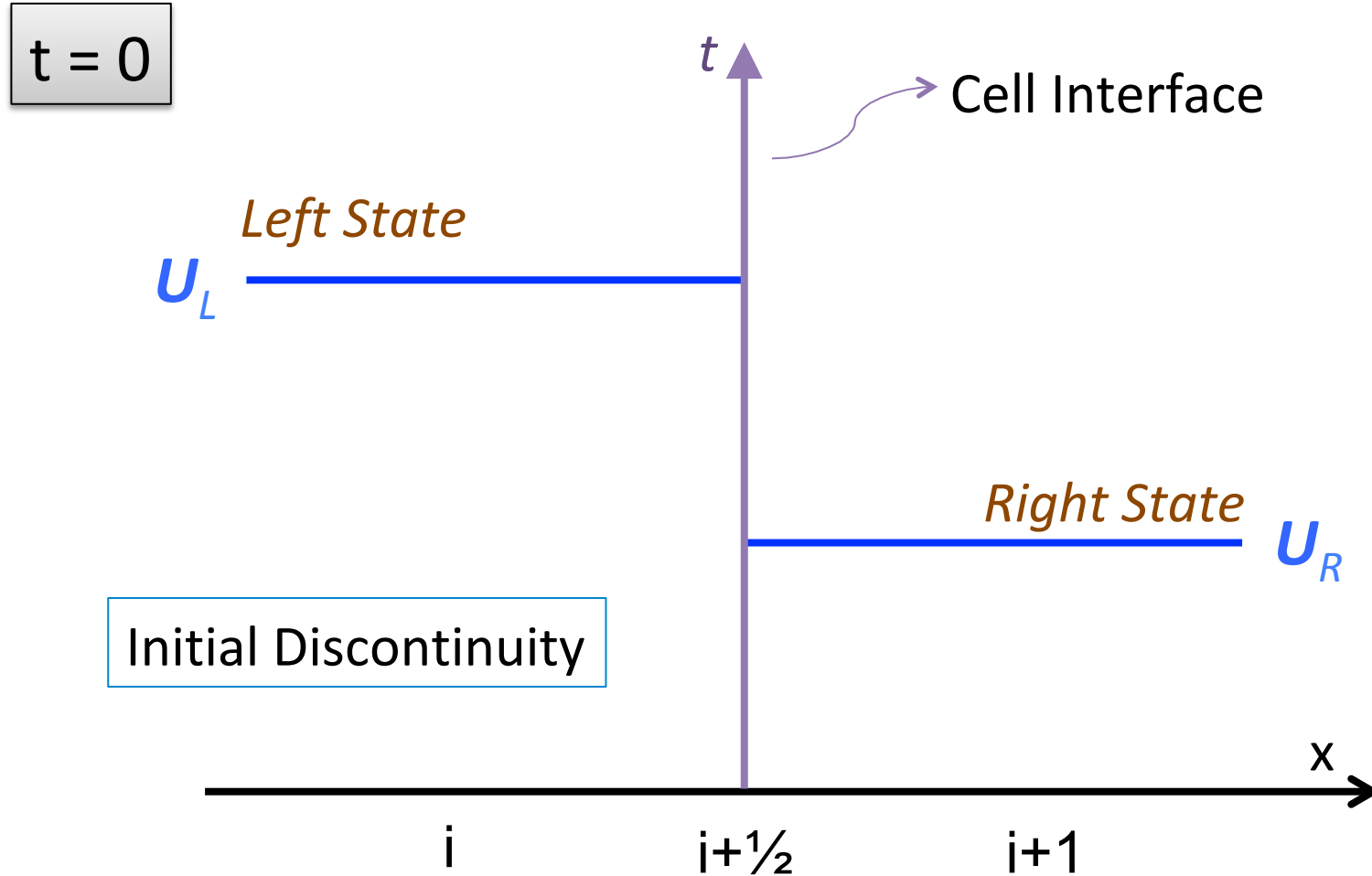
$$\tilde{F}_{i+\frac{1}{2}}^{n+\frac{1}{2}} = \frac{1}{\Delta t} \int_{t^n}^{t^{n+1}} F \left(U(x_{i+\frac{1}{2}}, t) \right) dt$$



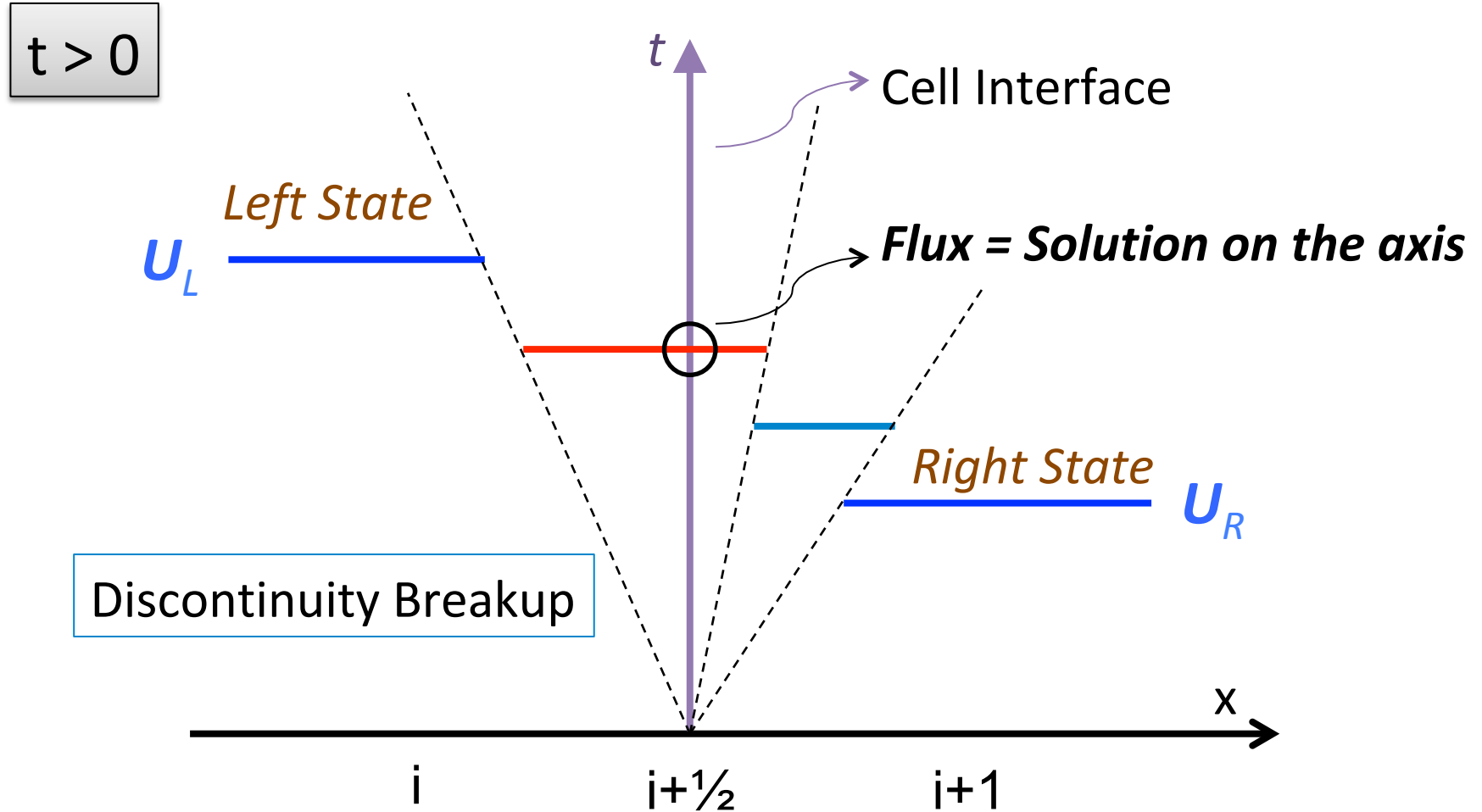
- This is achieved by solving the so-called “*Riemann Problem*”, i.e., the evolution of an initial discontinuity separating two constant states. The Riemann problem is defined by the initial condition:

$$U(x, 0) = \begin{cases} U_L & \text{for } x < x_{i+\frac{1}{2}} \\ U_R & \text{for } x > x_{i+\frac{1}{2}} \end{cases} \implies U(x_{i+\frac{1}{2}}, t > 0) = ?$$

The Riemann Problem



The Riemann Problem



3. THE LINEAR ADVECTION EQUATION: CONCEPTS AND DISCRETIZATIONS

The Advection Equation: Theory

- First order partial differential equation (PDE) in (x,t) :

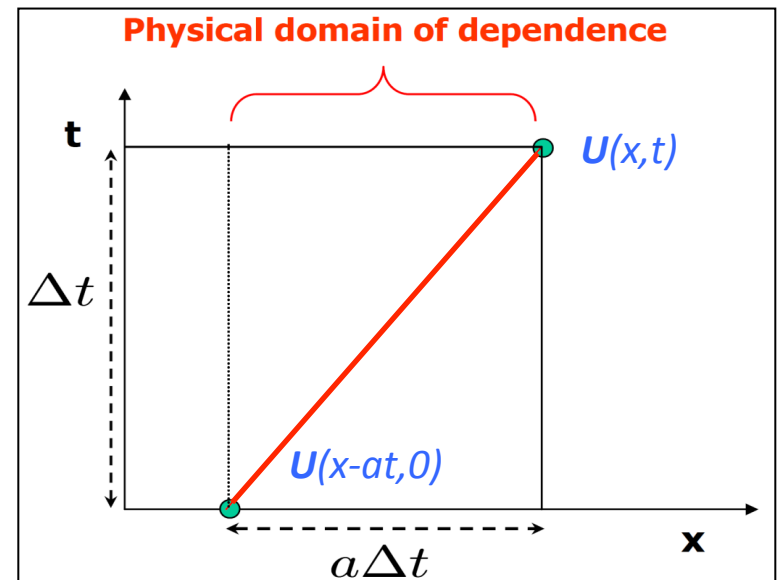
$$\frac{\partial U(x, t)}{\partial t} + a \frac{\partial U(x, t)}{\partial x} = 0$$

- Hyperbolic PDE: information propagates across domain at finite speed
→ method of characteristics

- Characteristic curves satisfy: $\frac{dx}{dt} = a$

- Along each characteristics:

$$\frac{dU}{dt} = \frac{\partial U}{\partial t} + \frac{dx}{dt} \frac{\partial U}{\partial x} = 0$$



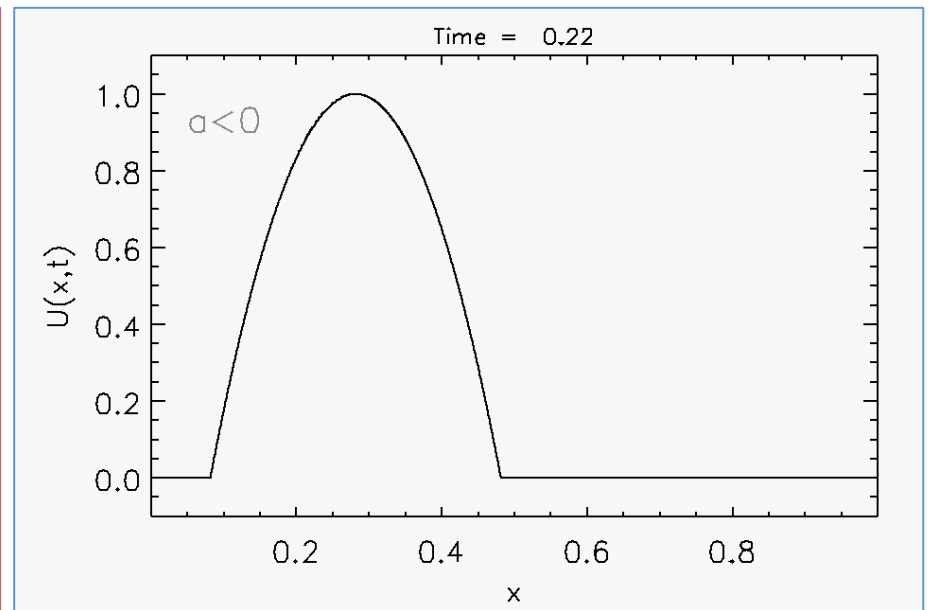
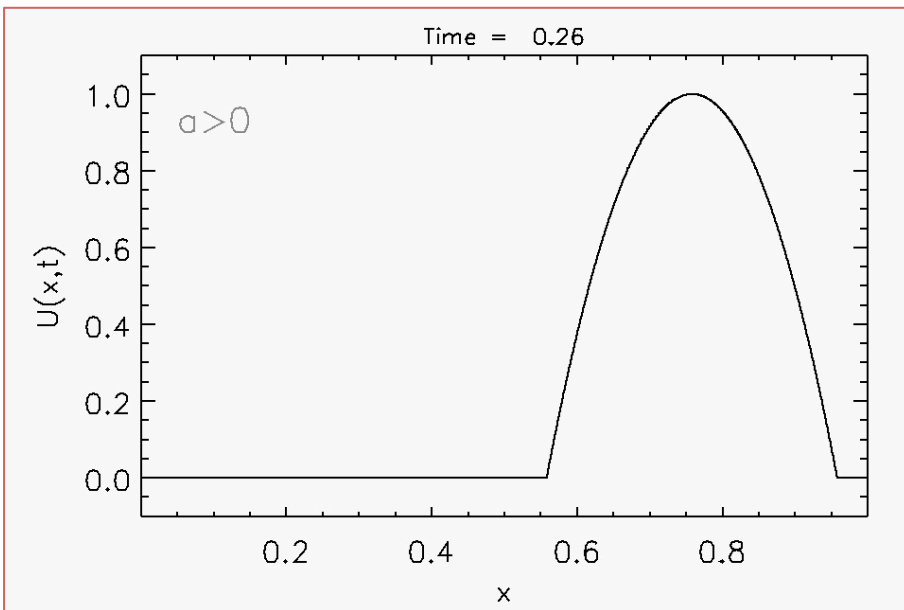
→ The solution is constant along characteristic curves.

The Advection Equation: Theory

- for constant a : the characteristics are straight parallel lines and the solution to the PDE is a uniform shift of the initial profile:

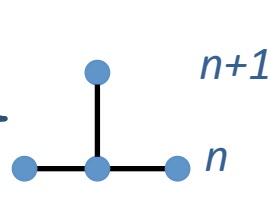
$$U(x, t) = U(x - at, 0)$$

- The solution shifts to the right (for $a > 0$) or to the left ($a < 0$):



Discretization: the FTCS Scheme

- Consider our model PDE
$$\frac{\partial U(x, t)}{\partial t} + a \frac{\partial U(x, t)}{\partial x} = 0$$

- Forward derivative in time:
$$\frac{\partial U}{\partial t} \approx \frac{U_i^{n+1} - U_i^n}{\Delta t} + O(\Delta t)$$
 - Centered derivative in space:
$$\frac{\partial U}{\partial x} \approx \frac{U_{i+1}^n - U_{i-1}^n}{2\Delta x} + O(\Delta x^2)$$
- 

- Putting all together and solving with respect to U^{n+1} gives

$$U_i^{n+1} = U_i^n - \frac{C}{2} (U_{i+1}^n - U_{i-1}^n)$$

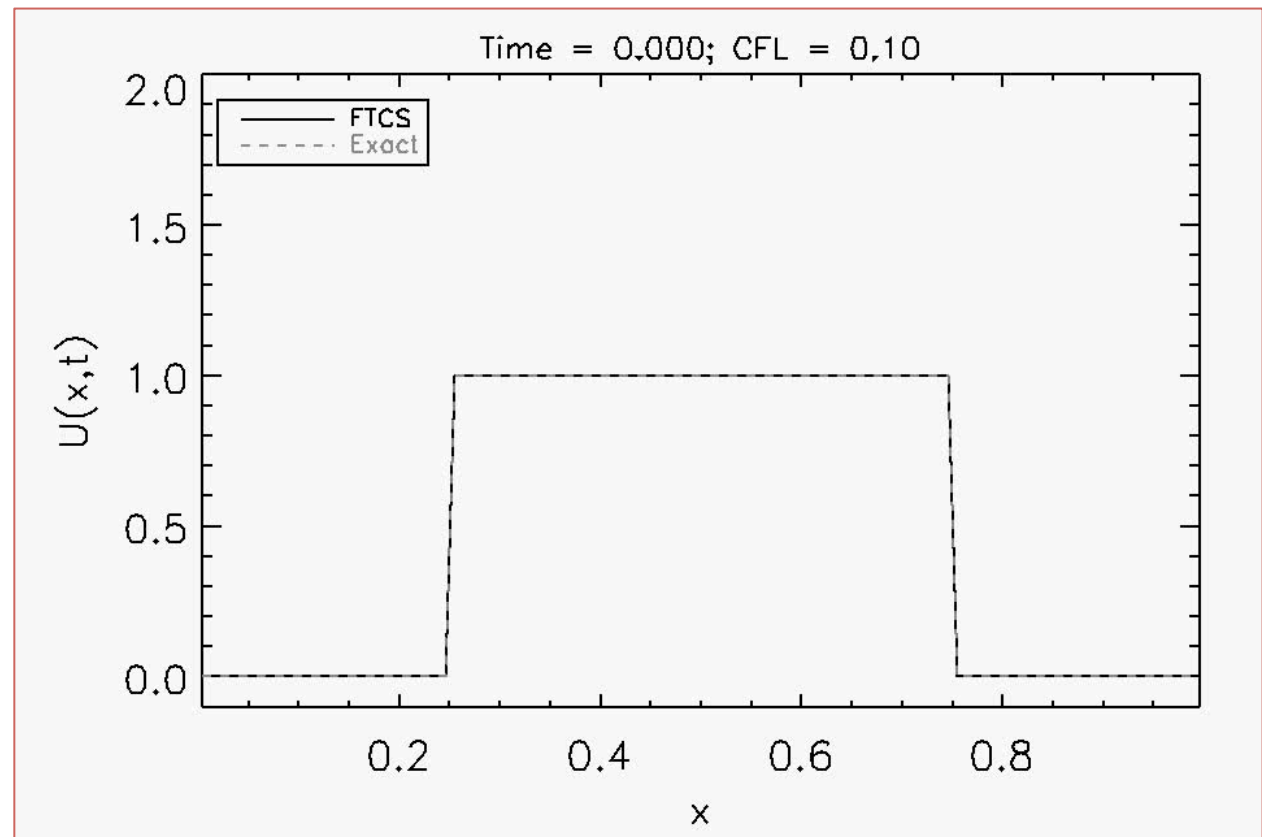
where $C = a \Delta t / \Delta x$ is the Courant-Friedrichs-Lewy (CFL) number.

- We call this method **FTCS** for **F**orward in **T**ime, **C**entered in **S**pace.
- It is an explicit method.

The FTCS Scheme

- At $t=0$, the initial condition is a square pulse with periodic boundary conditions:

$$\frac{\partial U}{\partial t} \approx \frac{U_i^{n+1} - U_i^n}{\Delta t} + O(\Delta t)$$
$$\frac{\partial U}{\partial x} \approx \frac{U_{i+1}^n - U_{i-1}^n}{2\Delta x} + O(\Delta x^2)$$



Something isn't right... why ?

FTCS: von Neumann Stability Analysis

- Let's perform an analysis of **FTCS** by expressing the solution as a Fourier series.
- Since the equation is linear, we only examine the behavior of a single mode. Consider a trial solution of the form:

$$U_i^n = A^n e^{Ii\theta}, \quad \theta = k\Delta x$$

- Plugging in the difference formula: $\frac{A^{n+1}}{A^n} = 1 - \frac{C}{2} (e^{I\theta} - e^{-I\theta})$

$$\implies \left| \frac{A^{n+1}}{A^n} \right|^2 = 1 + C^2 \sin^2 \theta \geq 1$$

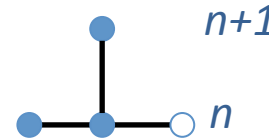
- Independently of the CFL number, all Fourier modes increase in magnitude as time advances.
- This method is **unconditionally unstable!**

Forward in Time, Backward in Space

- Let's try a difference approach. Consider the backward formula for the spatial derivative:

$$\frac{\partial U}{\partial x} \approx \frac{U_i^n - U_{i-1}^n}{\Delta x} + O(\Delta x) \implies \boxed{U_i^{n+1} = U_i^n - C (U_i^n - U_{i-1}^n)}$$

- The resulting scheme is called FTBS:



- Apply von Neumann stability analysis on the resulting discretized equation:

$$\left| \frac{A^{n+1}}{A^n} \right|^2 = 1 - 2C(1 - C)(1 - \cos \theta)$$

- Stability demands $\left| \frac{A^{n+1}}{A^n} \right| \leq 1 \implies 2C(1 - C) \geq 0$

- for $a < 0$ the method is unstable, but

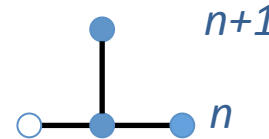
- for $a > 0$ the method is stable when $0 \leq C = a \Delta t / \Delta x \leq 1$.

Forward in Time, Forward in Space

- Repeating the same argument for the forward derivative

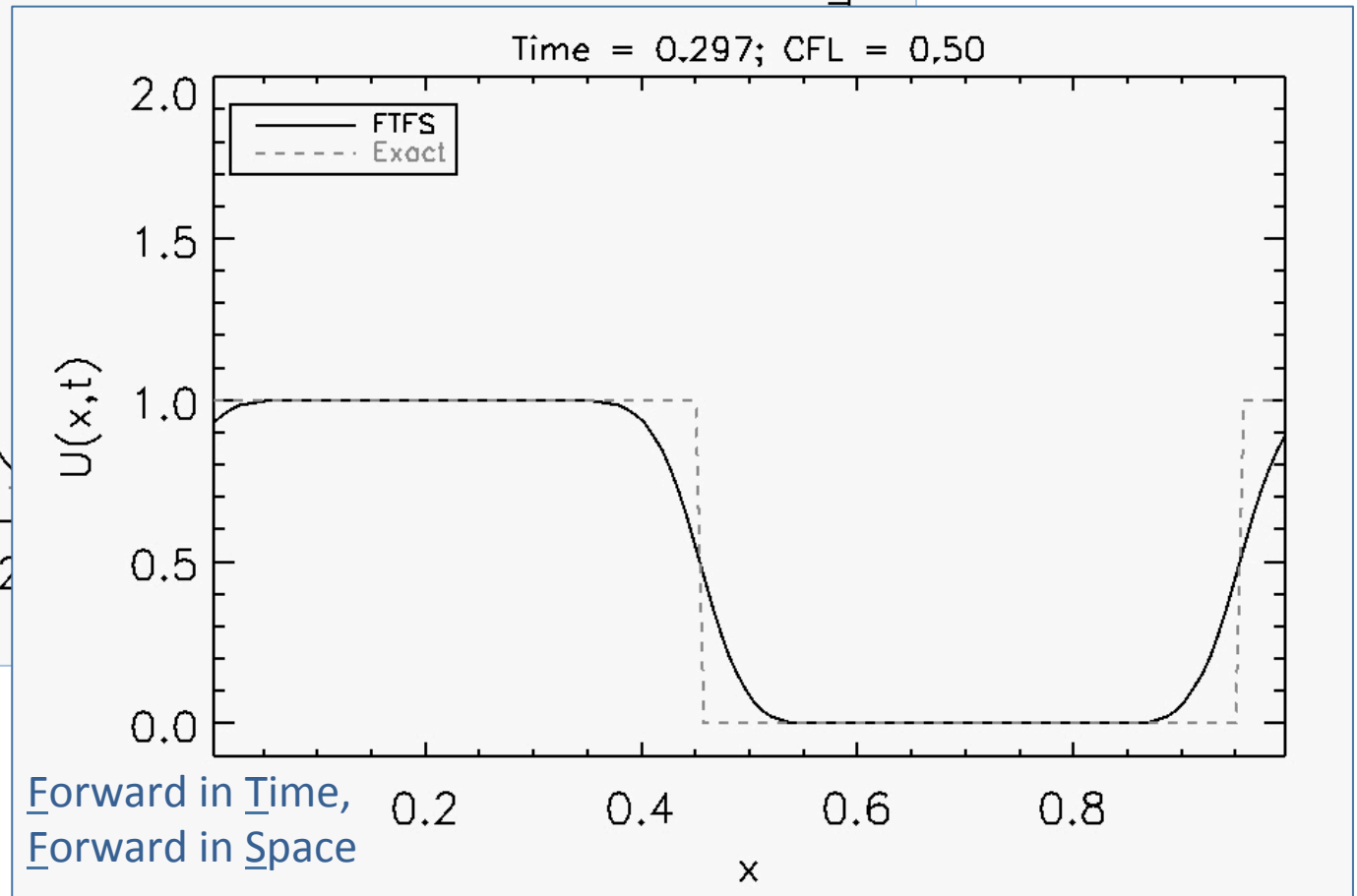
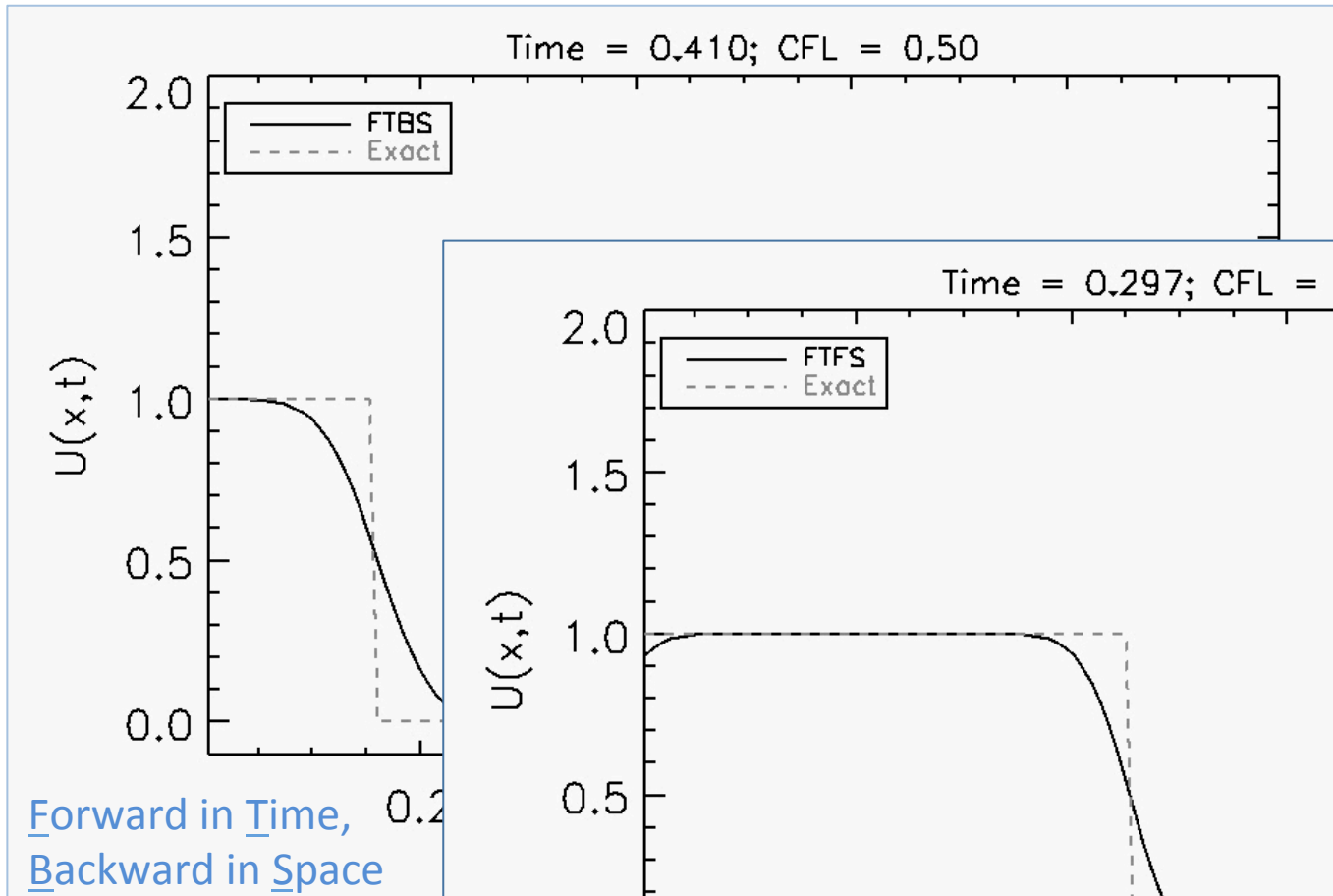
$$\frac{\partial U}{\partial x} \approx \frac{U_{i+1}^n - U_i^n}{\Delta x} + O(\Delta x) \implies \boxed{U_i^{n+1} = U_i^n - C (U_{i+1}^n - U_i^n)}$$

- The resulting scheme is called FTFS:



- Apply stability analysis yields $\left| \frac{A^{n+1}}{A^n} \right|^2 = 1 + 2C(1 - C)(1 - \cos \theta)$
- If $a > 0$ the method will always be unstable
- However, if $a < 0$ and $-1 \leq C = a \Delta t / \Delta x \leq 0$ then this method is stable;

Stable Discretizations: FTBS, FTFS



Stability: the CFL Condition

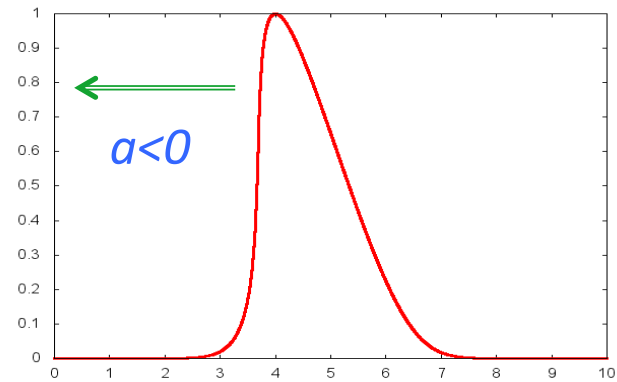
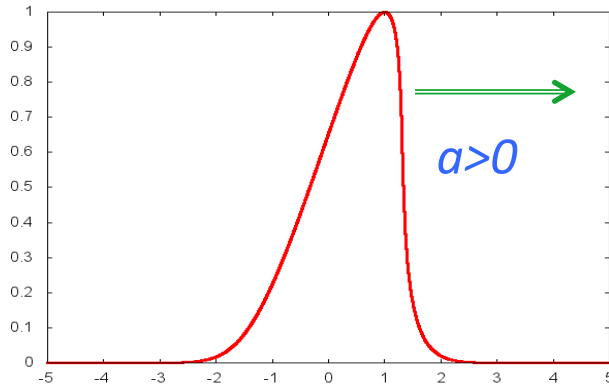
- Since the advection speed a is a parameter of the equation, Δx is fixed from the grid, the previous inequalities on $C=a\Delta t/\Delta x$ are stability constraints on the time step for explicit methods

$$\Delta t \leq \frac{\Delta x}{|a|}$$

- Δt cannot be arbitrarily large but, rather, less than the time taken to travel one grid cell (\rightarrow CFL condition).
- In the case of nonlinear equations, the speed can vary in the domain and the maximum of a should be considered instead.

The 1st Order Godunov Method

- Summarizing: the stable discretization makes use of the grid point where information is coming from:



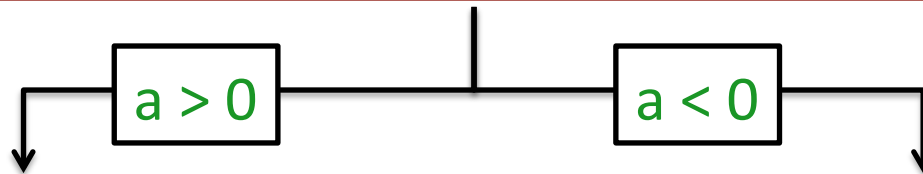
- 'Upwind':
$$\begin{cases} U_i^{n+1} = U_i^n - \frac{a\Delta t}{\Delta x} (U_i^n - U_{i-1}^n) & \text{for } a > 0 \\ U_i^{n+1} = U_i^n - \frac{a\Delta t}{\Delta x} (U_{i+1}^n - U_i^n) & \text{for } a < 0 \end{cases}$$

- This is also called the first-order Godunov method;

Conservative Form

- Define the “flux” function $F_{i+\frac{1}{2}}^n = \frac{a}{2} (U_{i+1}^n + U_i^n) - \frac{|a|}{2} (U_{i+1}^n - U_i^n)$ so that Godunov method can be cast in *conservative* form

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

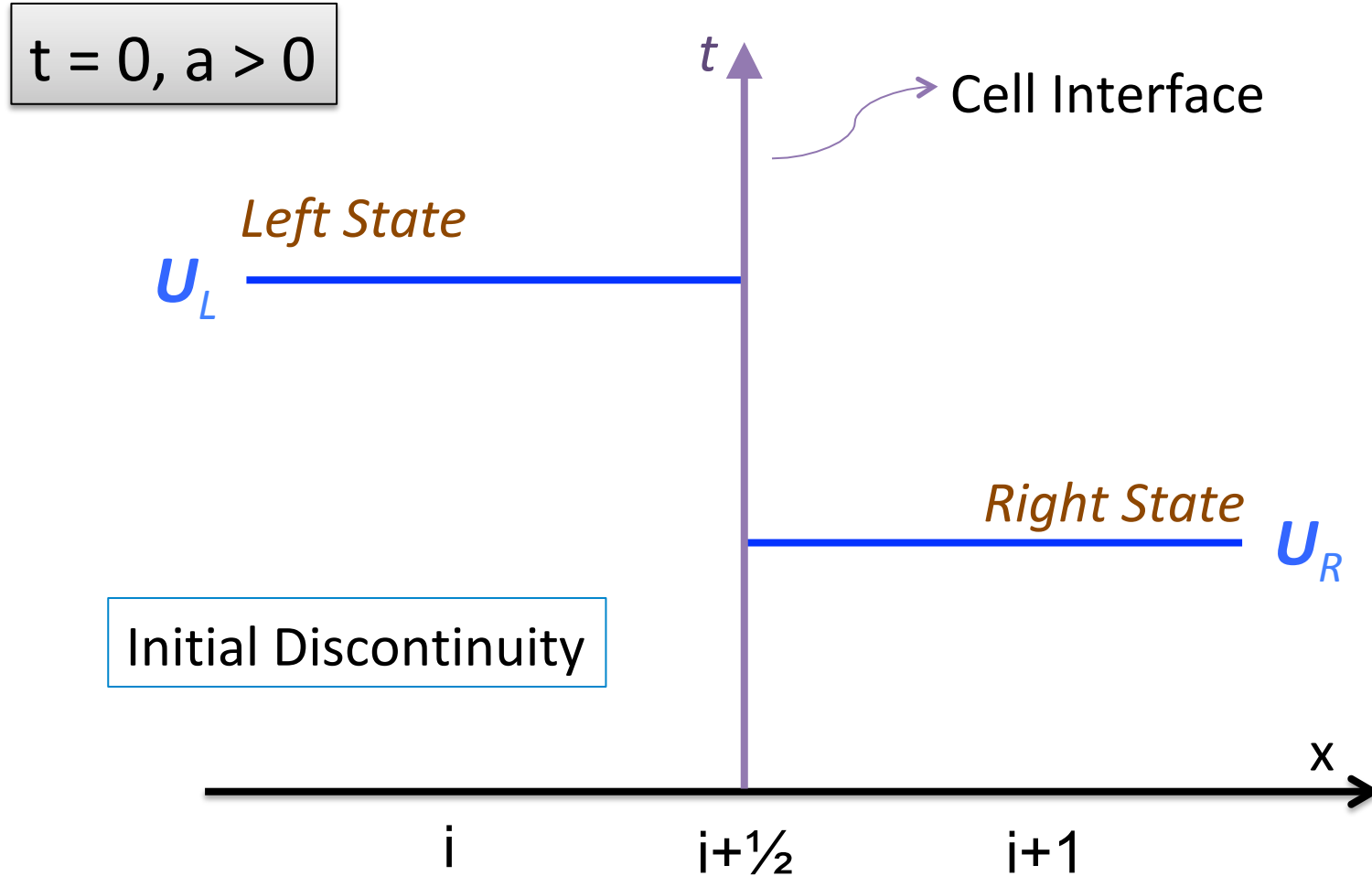


$$U_i^{n+1} = U_i^n - \frac{a\Delta t}{\Delta x} (U_i^n - U_{i-1}^n)$$

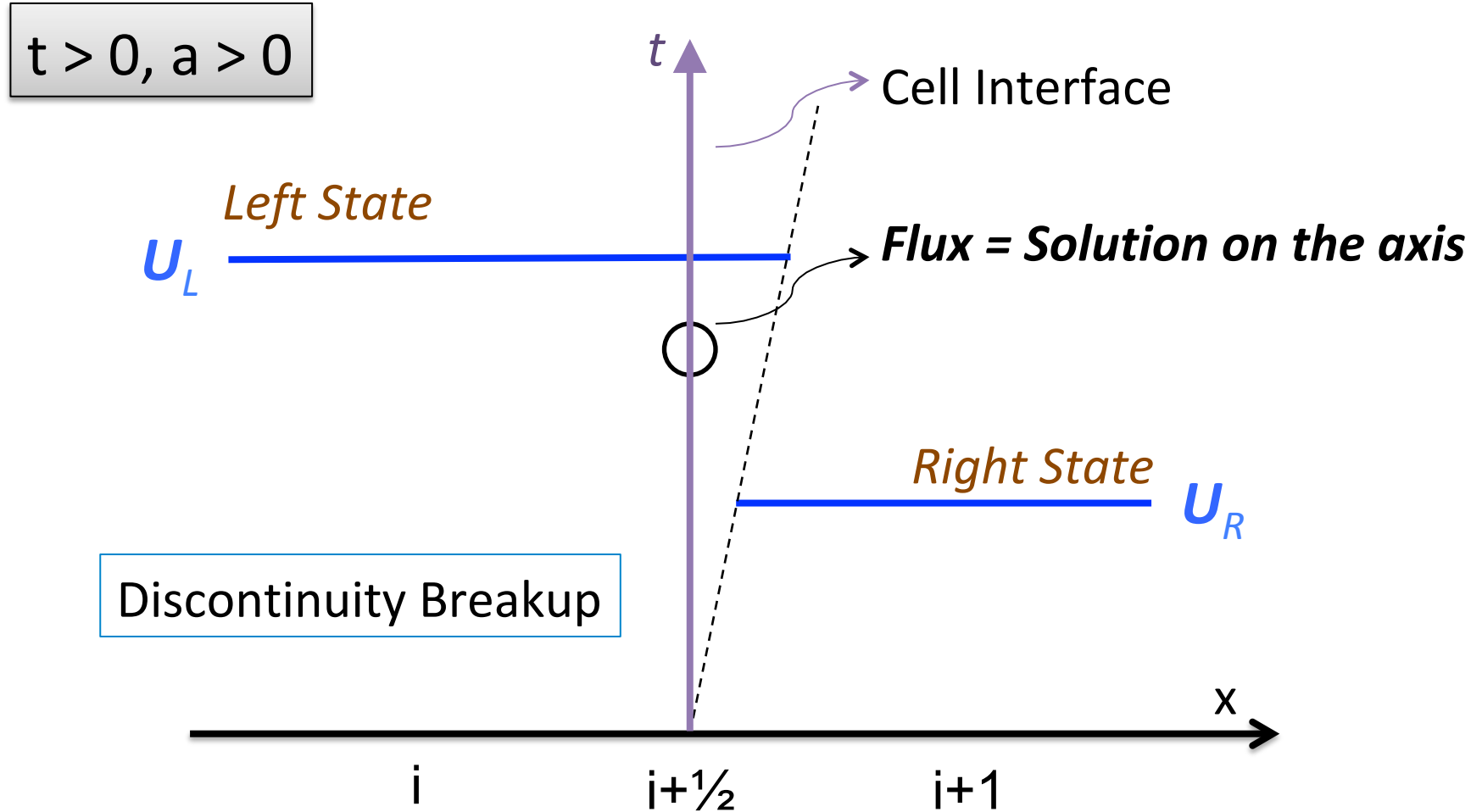
$$U_i^{n+1} = U_i^n - \frac{a\Delta t}{\Delta x} (U_{i+1}^n - U_i^n)$$

- The conservative form ensures a correct description of *discontinuities* in nonlinear systems, ensures global conservation properties and is the main building block in the development of high-order *finite volume* schemes.

The Riemann Problem



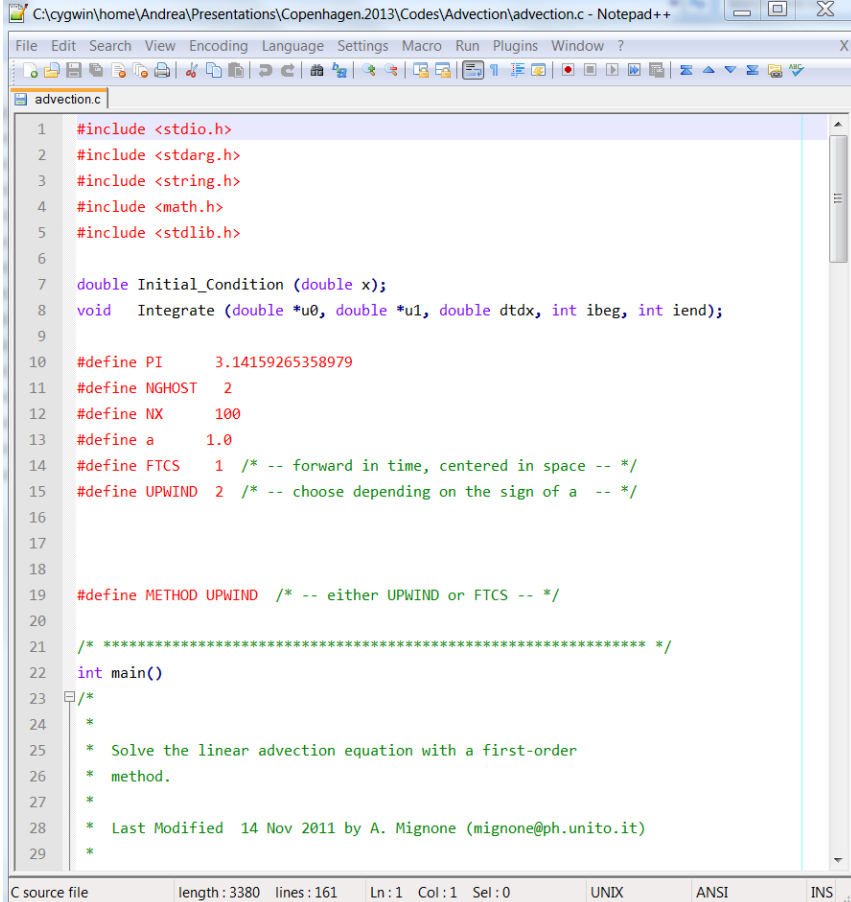
The Riemann Problem



Code Example

- File name: *advection.c*¹
- Purpose: solve the linear advection equation using the 1st-order Godunov method.
- Usage:

```
> gcc advection.c -lm -o advection
> ./advection
```
- Output: two-column ascii data files.
- Visualization: gnuplot (→ *advection.gp*).



```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <string.h>
4 #include <math.h>
5 #include <stdlib.h>
6
7 double Initial_Condition (double x);
8 void Integrate (double *u0, double *u1, double dtdx, int ibeg, int iend);
9
10 #define PI 3.14159265358979
11 #define NGHOST 2
12 #define NX 100
13 #define a 1.0
14 #define FTCS 1 /* -- forward in time, centered in space -- */
15 #define UPWIND 2 /* -- choose depending on the sign of a -- */
16
17
18
19 #define METHOD UPWIND /* -- either UPWIND or FTCS -- */
20
21 /* ***** */
22 int main()
23 /*
24 *
25 * Solve the linear advection equation with a first-order
26 * method.
27 *
28 * Last Modified 14 Nov 2011 by A. Mignone (mignone@ph.unito.it)
29 *
30 */
```

¹<http://personalpages.to.infn.it/~mignone/Astrosim2019/>

4. LINEAR SYSTEMS OF HYPERBOLIC CONSERVATION LAWS

System of Equations: Theory

- We turn our attention to the system of equations (PDE)

$$\frac{\partial \mathbf{q}}{\partial t} + A \cdot \frac{\partial \mathbf{q}}{\partial x} = 0$$

where $\mathbf{q} = \{q_1, q_2, \dots, q_m\}$ is the vector of unknowns. A is a $m \times m$ constant matrix.

- For example, for $m=3$, one has

$$\frac{\partial q_1}{\partial t} + A_{11} \frac{\partial q_1}{\partial x} + A_{12} \frac{\partial q_2}{\partial x} + A_{13} \frac{\partial q_3}{\partial x} = 0$$

$$\frac{\partial q_2}{\partial t} + A_{21} \frac{\partial q_1}{\partial x} + A_{22} \frac{\partial q_2}{\partial x} + A_{23} \frac{\partial q_3}{\partial x} = 0$$

$$\frac{\partial q_3}{\partial t} + A_{31} \frac{\partial q_1}{\partial x} + A_{32} \frac{\partial q_2}{\partial x} + A_{33} \frac{\partial q_3}{\partial x} = 0$$

System of Equations: Theory

- The system is hyperbolic if A has real eigenvalues, $\lambda^1 \leq \dots \leq \lambda^m$ and a complete set of linearly independent right and left eigenvectors \mathbf{r}^k and \mathbf{l}^k ($\mathbf{r}^j \cdot \mathbf{l}^k = \delta_{jk}$) such that

$$\begin{cases} A \cdot \mathbf{r}^k = \lambda^k \mathbf{r}^k \\ \mathbf{l}^k \cdot A = \mathbf{l}^k \lambda^k \end{cases} \quad \text{for } k = 1, \dots, m$$

- For convenience we define the matrices $\Lambda = \text{diag}(\lambda^k)$, and

$$R = \left(\mathbf{r}^1 | \mathbf{r}^2 | \dots | \mathbf{r}^m \right), \quad L = R^{-1} = \begin{pmatrix} \frac{1^1}{\mathbf{l}^1} \\ \frac{1^2}{\mathbf{l}^2} \\ \vdots \\ \frac{1^m}{\mathbf{l}^m} \end{pmatrix}$$

- So that $A \cdot R = R \cdot \Lambda$, $L \cdot A = \Lambda \cdot L$, $L \cdot R = R \cdot L = I$, $L \cdot A \cdot R = \Lambda$.

System of Equations: Theory

- The linear system can be reduced to a set of decoupled linear advection equations.
- Multiply the original system of PDE's by L on the left:

$$L \cdot \left(\frac{\partial \mathbf{q}}{\partial t} + A \cdot \frac{\partial \mathbf{q}}{\partial x} \right) = L \cdot \frac{\partial \mathbf{q}}{\partial t} + L \cdot A \cdot R \cdot L \cdot \frac{\partial \mathbf{q}}{\partial x} = 0$$

- Define the characteristic variables $w=L \cdot q$ so that

$$\frac{\partial w}{\partial t} + \Lambda \cdot \frac{\partial w}{\partial x} = 0$$

- Since Λ is diagonal, these equations are not coupled anymore.

System of Equations: Theory

- In this form, the system decouples into m independent advection equations for the characteristic variables:

$$\frac{\partial w}{\partial t} + \Lambda \cdot \frac{\partial w}{\partial x} = 0 \quad \Longrightarrow \quad \frac{\partial w^k}{\partial t} + \lambda^k \cdot \frac{\partial w^k}{\partial x} = 0$$

where $w^k = \mathbf{l}^k \cdot \mathbf{q}$ ($k=1,2,\dots,m$) is a characteristic variable.

- When $m=3$ one has, for instance:
$$\frac{\partial w^1}{\partial t} + \lambda^1 \frac{\partial w^1}{\partial x} = 0$$
$$\frac{\partial w^2}{\partial t} + \lambda^2 \frac{\partial w^2}{\partial x} = 0$$
$$\frac{\partial w^3}{\partial t} + \lambda^3 \frac{\partial w^3}{\partial x} = 0$$

System of Equations: Theory

- The m advection equations can be solved independently by applying the standard solution techniques developed for the scalar equation.
- In particular, one can write the exact analytical solution for the k -th characteristic field as

$$w^k(x, t) = w^k(x - \lambda^k t, 0)$$

i.e., the initial profile of w^k shifts with uniform velocity λ^k , and

$$w^k(x - \lambda^k t, 0) = \mathbf{l}^k \cdot \mathbf{q}(x - \lambda^k t, 0)$$

is the initial profile.

- The characteristics are thus constant along the curves $dx/dt = \lambda^k$

System of Equations: Exact Solution

- Once the solution in characteristic space is known, we can solve the original system via the inverse transformation

$$\mathbf{q}(x, t) = R \cdot \mathbf{w}(x, t) = \sum_{k=1}^{k=m} w^k(x, t) \mathbf{r}^k = \sum_{k=1}^{k=m} w^k(x - \lambda^k t, 0) \mathbf{r}^k$$

- The characteristic variables are thus the coefficients of the right eigenvector expansion of \mathbf{q} .
- The solution to the linear system reduces to a linear combination of m linear waves traveling with velocities λ^k .
- Expressing everything in terms of the original variables \mathbf{q} ,

$$\mathbf{q}(x, t) = \sum_{k=1}^{k=m} \mathbf{l}^k \cdot \mathbf{q}(x - \lambda^k t, 0) \mathbf{r}^k$$

Piecewise Discontinuous Data

- If \mathbf{q} is initially discontinuous, one or more characteristic variables will also have a discontinuity. Indeed, at $t = 0$,

$$w^k(x, 0) = \mathbf{l}^k \cdot \mathbf{q}(x, 0) = \begin{cases} w_L^k = \mathbf{l}^k \cdot \mathbf{q}_L & \text{if } x < x_{i+\frac{1}{2}} \\ w_R^k = \mathbf{l}^k \cdot \mathbf{q}_R & \text{if } x > x_{i+\frac{1}{2}} \end{cases}$$

- In other words, the initial jump $\mathbf{q}_R - \mathbf{q}_L$ is decomposed in several waves each propagating at the constant speed λ^k and corresponding to the eigenvectors of the Jacobian \mathbf{A} :

$$\mathbf{q}_R - \mathbf{q}_L = \alpha^1 \mathbf{r}^1 + \alpha^2 \mathbf{r}^2 + \dots + \alpha^m \mathbf{r}^m$$

where $\alpha^k = \mathbf{l}^k \cdot (\mathbf{q}_R - \mathbf{q}_L)$ are the wave strengths

Riemann Problem for Discontinuous Data

- For the linear case, the exact solution for each wave at the cell interface is:

$$w^k \left(x_{i+\frac{1}{2}}, t \right) = w^k \left(x_{i+\frac{1}{2}} - \lambda^k t, 0 \right) = \begin{cases} w_L^k & \text{if } \lambda^k > 0 \\ w_R^k & \text{if } \lambda^k < 0 \end{cases}$$

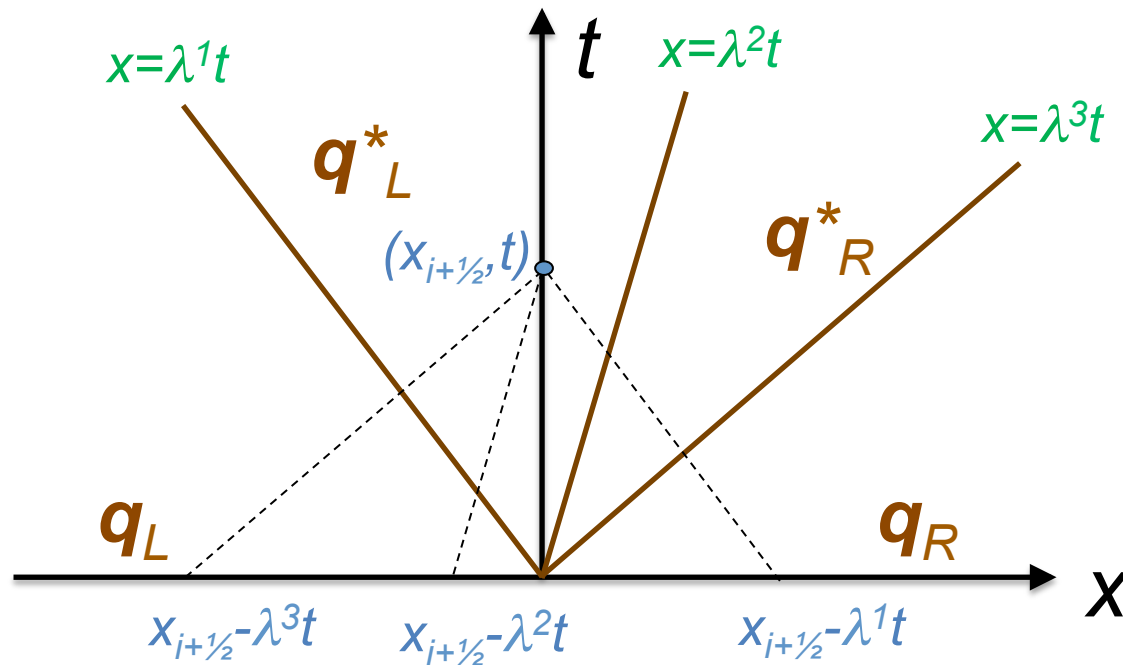
- The complete solution is found by adding all wave contributions:

$$\mathbf{q} \left(x_{i+\frac{1}{2}}, t \right) = \sum_{k:\lambda_k>0} w_L^k \mathbf{r}^k + \sum_{k:\lambda_k<0} w_R^k \mathbf{r}^k$$

- and the flux is finally computed as $\tilde{\mathbf{F}}_{i+\frac{1}{2}} = A \cdot \mathbf{q} \left(x_{i+\frac{1}{2}}, t \right)$

The Riemann Problem

$$\begin{aligned}\lambda^1 &< 0 \\ \lambda^2 &> 0 \\ \lambda^3 &> 0\end{aligned}$$



Point $(x_{i+1/2}, t)$ traces back to the right of the λ^1 characteristic emanating from the initial jump, but to the left of the other 2, so the solution is:

$$\mathbf{q} \left(x_{i+\frac{1}{2}}, t \right) = w_R^1 \mathbf{r}^1 + w_L^2 \mathbf{r}^2 + w_L^3 \mathbf{r}^3$$

Numerical Implementation

- We suppose the solution at time level n is known as q^n and we wish to compute the solution q^{n+1} at the next time level $n+1$.
- Our numerical scheme can be derived by working in the characteristic space and then transforming back:

$$q_i^{n+1} = \sum_k w_i^{k,n+1} r^k = q_i^n - \frac{\Delta t}{\Delta x} \left(F_{i+\frac{1}{2}}^n - F_{i-\frac{1}{2}}^n \right)$$

where

$$F_{i+\frac{1}{2}}^n = A \cdot \frac{q_{i+1}^n + q_i^n}{2} - \frac{1}{2} \sum_k |\lambda^k| l^k \cdot (q_{i+1}^n - q_i^n) r^k$$

is the *Godunov flux* for a linear system of advection equations.

Example: The Acoustic Wave Equations

- The acoustic wave equations can be derived from the Euler equations assuming small perturbations around a background constant state.
- Linearizing around a reference state $Q(x, t) = Q_0 + Q_1(x, t)$:

$$\frac{\partial Q_1}{\partial t} + A \frac{\partial Q_1}{\partial t} = 0, \quad A = \begin{pmatrix} u_0 & \rho_0 & 0 \\ 0 & u_0 & 1/\rho_0 \\ 0 & a^2 \rho_0 & u_0 \end{pmatrix}$$

where $Q_1 = (\rho_1, u_1, p_1)$ denotes perturbed quantities and $a = \sqrt{\frac{\Gamma p_0}{\rho_0}}$

- The spectral decomposition of this system is

$$\lambda = \left(u_0 - a, u_0, u_0 + a \right)$$

$$R = \begin{pmatrix} \frac{1}{a^2} & 1 & \frac{1}{a^2} \\ -\frac{1}{\rho_0 a^2} & 0 & \frac{1}{\rho_0 a} \\ 1 & 0 & 1 \end{pmatrix}, \quad L = \begin{pmatrix} 0 & -a\rho_0/2 & 1/2 \\ 1 & 0 & -1/a^2 \\ 0 & a\rho_0/2 & 1/2 \end{pmatrix}$$

- File name: *acoustic.c*¹
- Purpose: solve the 3x3 acoustic wave equations with the 1st-order Godunov method.
- Usage:

```
> gcc acoustic.c -lm -o acoustic
> ./acoustic
```
- Output: four-column ascii data files.
- Visualization: gnuplot (\rightarrow *acoustic.gp*).

```
7 #include <stdio.h>
8 #include <stdlib.h>
9 #include <string.h>
10 #include <math.h>
11 #include <stdlib.h>
12
13 void Initial_Condition (double, double *);
14 void Integrate (double **, double *, double, int, int);
15 int Output (double *, double **, int, int);
16 double **Array2D (int, int);
17
18 #define NGHOST 1 /* Number of ghost zones */
19 #define NVAR 3 /* Number of variables */
20 #define NX 1024 /* Number of zones (excluding ghost zones) */
21 #define GAMMA_EOS (5.0/3.0) /* Specific heat ratio */
22 /* ***** */
23 int main()
24 /*
25 *
26 *
27 ***** */
28 {
29     int i, nv, nstep;
30     int noutput = 100; /* Save noutput (+1) files */
31     int ibeg = NGHOST;
32     int iend = ibeg + NX - 1;
33     double xbeg = 0.0; /* start domain */
34     double xend = 1.0; /* end domain */
35     double tstop = 1.0; /* Final time */
36     double cfl = 0.9; /* Courant number */
37     double x[NX + 2*NGHOST], dx;
38     double **Q1; /* Solution array */
39     double Q0[NVAR]; /* Background reference state */
40     double t, dt, dtdx, a;
41
42     double rho0 = Q0[0] = 1.0; /* Background density */
43     double u0 = Q0[1] = 0.1; /* Background velocity */
44     double p0 = Q0[2] = 1.0/GAMMA_EOS; /* Background pressure */
45
46     Q1 = Array2D(NX + 2*NGHOST, NVAR);
47
48     /* -- 1. Make grid -- */
49
50     dx = (xend - xbeg)/(double)NX;
51     for (i = 0; i <= iend + NGHOST; i++){
52         x[i] = xbeg + (0.5 + i - ibeg)*dx;
53         Initial_Condition (x[i], Q1[i]);
54     }
55
56     /* -- 2. Start computation -- */
57
58     t = 0.0;
59     nstep = 0;
60     a = fabs(u0) + sqrt(GAMMA_EOS*p0/rho0); /* Maximum speed */
```

¹<http://personalpages.to.infn.it/~mignone/Astrosim2019/>

5. NONLINEAR SCALAR HYPERBOLIC PDE

Nonlinear Advection Equation

- We turn our attention to the scalar conservation law

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

- Where $f(u)$ is, in general, a nonlinear function of u .
- To gain some insights on the role played by nonlinear effects, we start by considering the inviscid Burger's equation:

$$\frac{\partial u}{\partial t} + \frac{\partial}{\partial x} \left(\frac{u^2}{2} \right) = 0$$

- This is the simplest nonlinear scalar hyperbolic PDE.

Nonlinear Advection Equation

- We can write Burger's equation also as $\frac{\partial u}{\partial t} + u \frac{\partial u}{\partial x} = 0$

- In this form, Burger's equation resembles the linear advection equation, except that the velocity is no longer constant but it is equal to the solution itself.
- The characteristic curve for this equation is

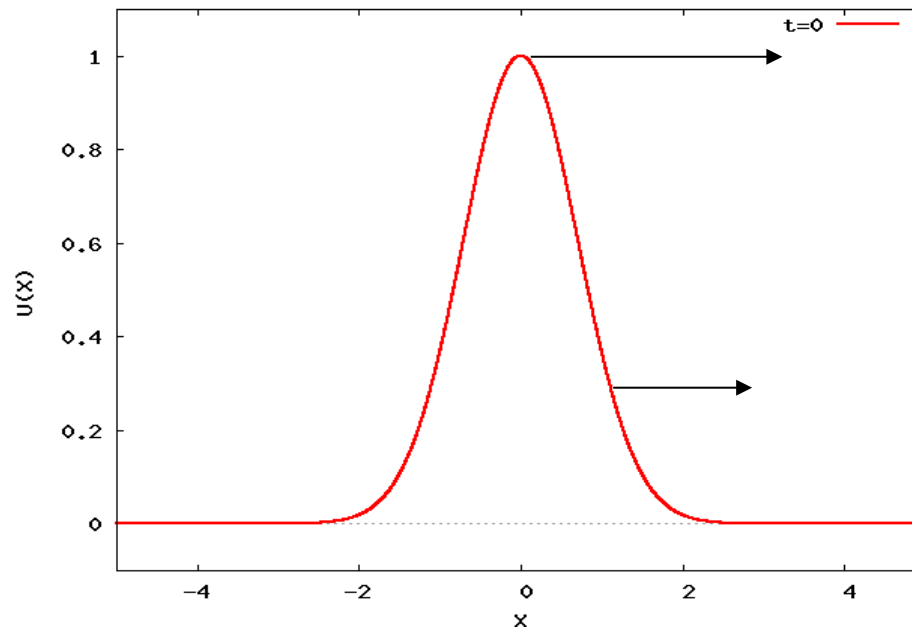
$$\frac{dx}{dt} = u(x, t) \quad \Longrightarrow \quad \frac{du}{dt} = \frac{\partial u}{\partial t} + \frac{\partial u}{\partial x} \frac{dx}{dt} = 0$$

- $\rightarrow u$ is constant along the curve $dx/dt = u(x, t) \rightarrow$ characteristics are again straight lines: values of u associated with some fluid element do not change as that element moves.

Nonlinear Advection Equation

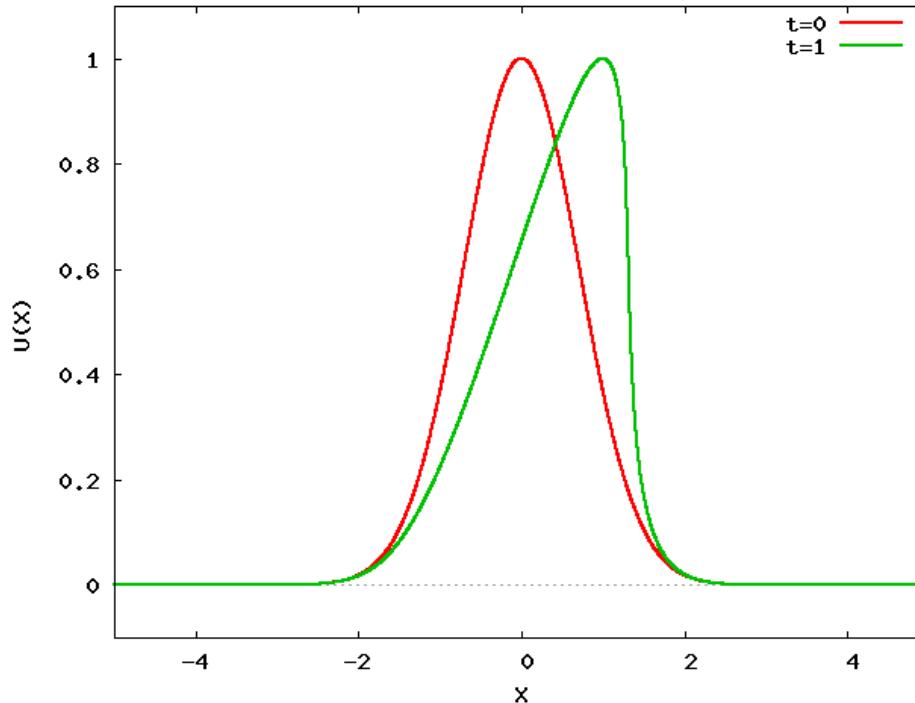
- From
$$\frac{\partial u}{\partial t} + u \frac{\partial u}{\partial x} = 0$$

one can predict that, higher values of u will propagate faster than lower values: this leads to a *wave steepening*, since upstream values will advance faster than downstream values.



Nonlinear Advection Equation

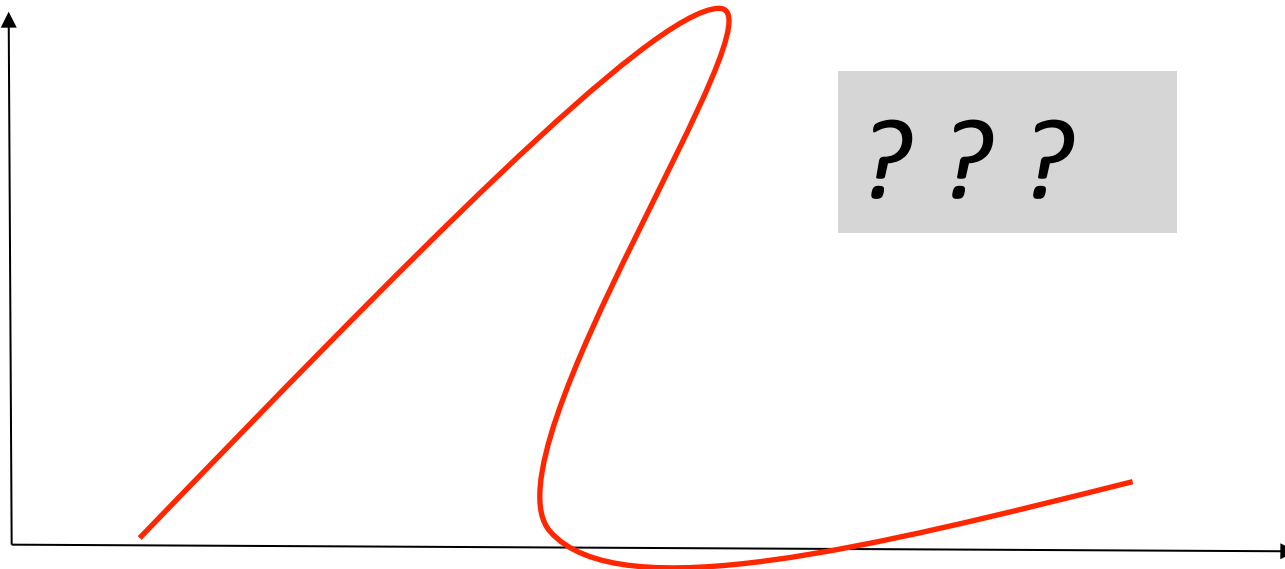
- Indeed, at $t=1$ the wave profile will look like:



- the wave steepens...

Nonlinear Advection Equation

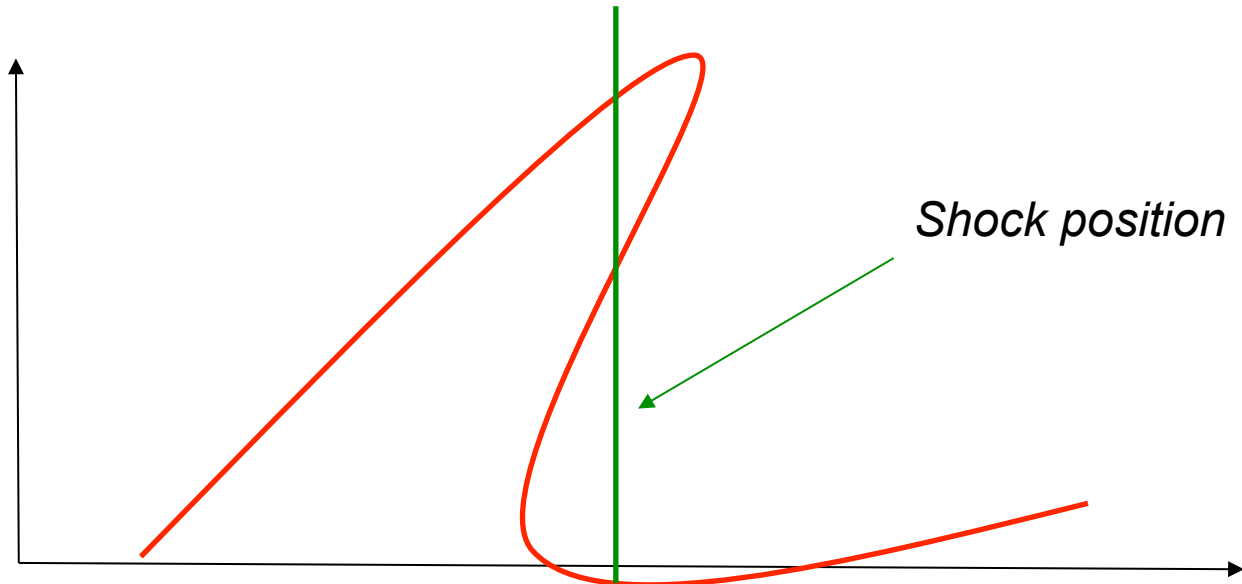
- If we wait more, we should get something like this:



- A multi-value functions?! → Clearly NOT physical !

Burger Equation: Shock Waves

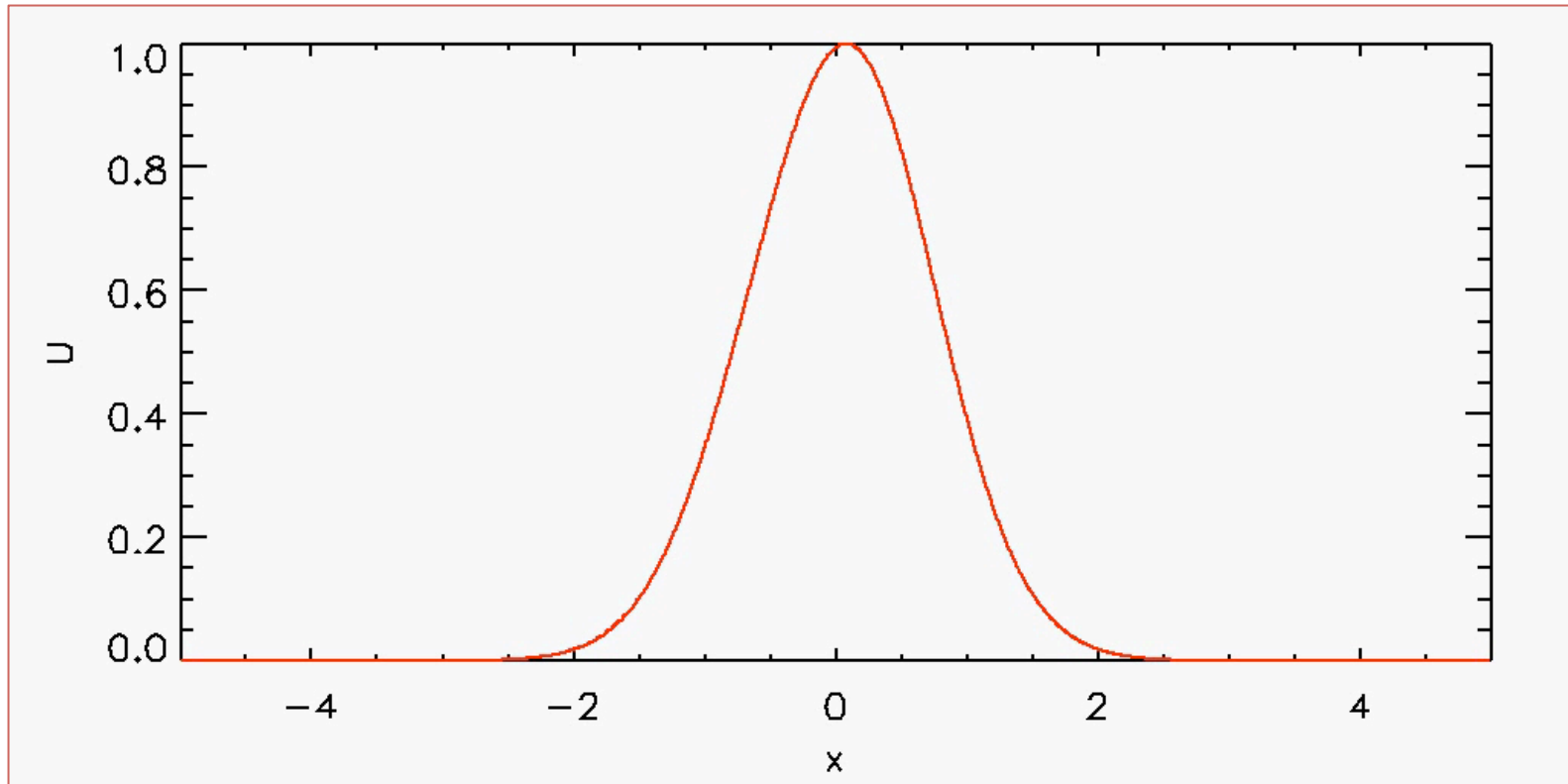
- The correct physical solution is to place a discontinuity there: a shock wave.



- Since the solution is no longer smooth, the differential form is not valid anymore and we need to consider the *integral form*.

Burger Equation: Shock Waves

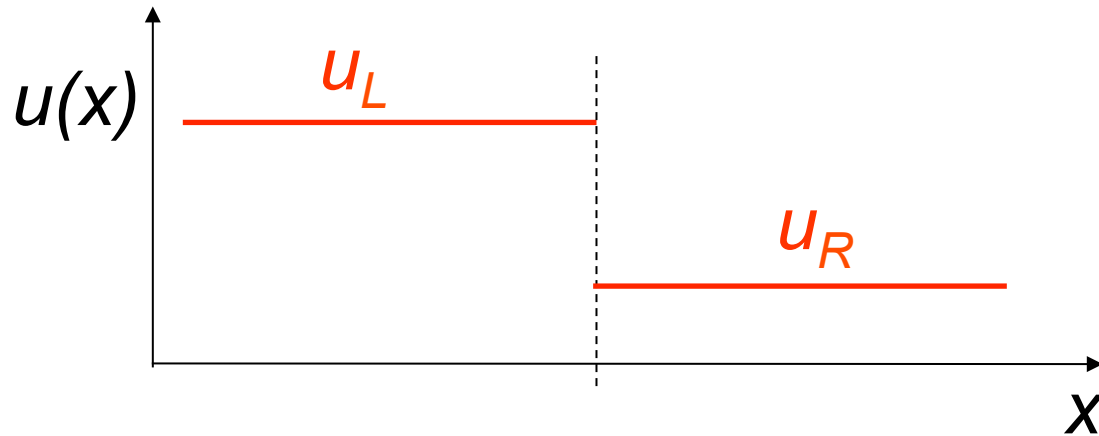
- This is how the solution should look like:



- Such solutions to the PDE are called *weak solutions*.

Burger Equation: Shock Waves

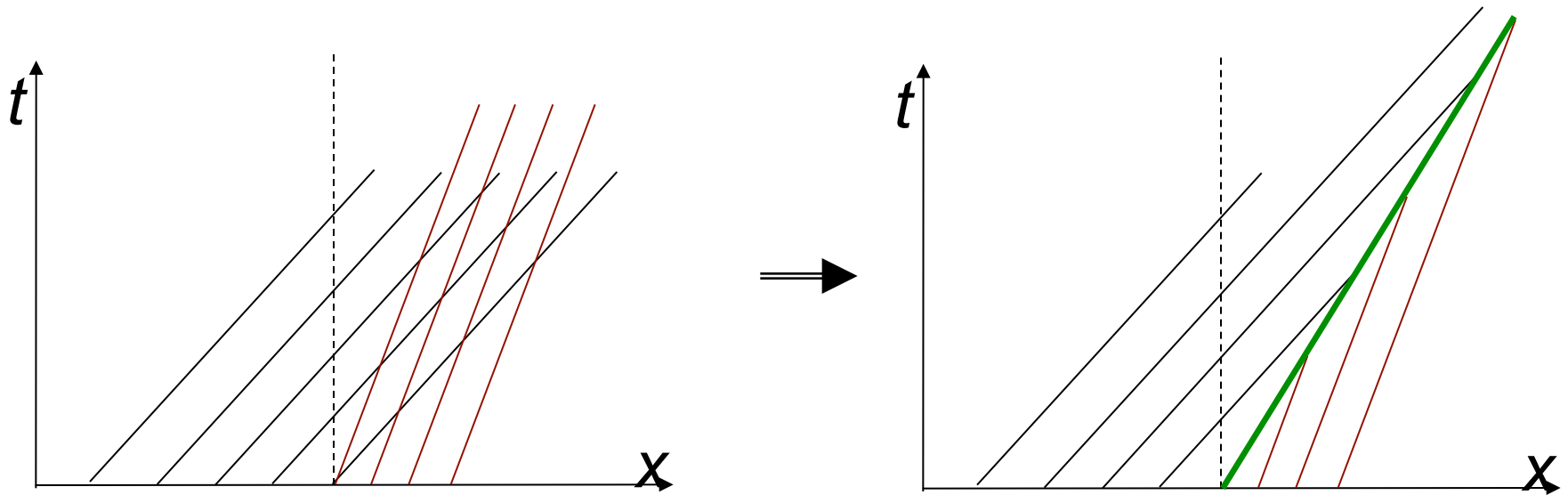
- Let's try to understand what happens by looking at the characteristics.
- Consider two states initially separated by a jump at an interface:



- Here, the characteristic velocities on the left are greater than those on the right.

Burger Equation: Shock Waves

- The characteristic will intersect, creating a *shock wave*:



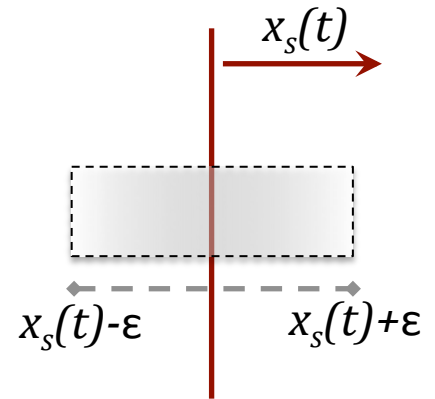
- The shock speed is such that $\lambda(u_L) > S > \lambda(u_R)$. This is called the entropy condition.

Shock Jump Conditions

- Consider a generic conservation law:
$$\frac{\partial u}{\partial t} + \frac{\partial f(u)}{\partial x} = 0$$

- Integrate it across a segment $[a, b] = [x_s(t) - \varepsilon, x_s(t) + \varepsilon]$ stretching across a discontinuity with position $x_s(t)$:

$$\int_{a(t)}^{b(t)} \partial_t u(x, t) dx + \int_{a(t)}^{b(t)} \partial_x f(u) dx = 0$$



- Using Leibniz rule for the first term, one obtains

$$\frac{d}{dt} \int_{a(t)}^{b(t)} u(x, t) dx - u(b, t) \dot{x}_s + u(a, t) \dot{x}_s + (f(b) - f(a)) = 0$$

- Taking the limit for $\varepsilon \rightarrow 0$,
$$\dot{x}_s (u_R - u_L) = f_R - f_L$$

- These are valid for a generic conservation laws and are also known as the Rankine-Hugoniot jump conditions.

Nonlinear Advection Equation

- In the case of Burger's equation we can immediately apply the Rankine-Hugoniot jump conditions, yielding

$$f(u_R) - f(u_L) = S(u_R - u_L)$$

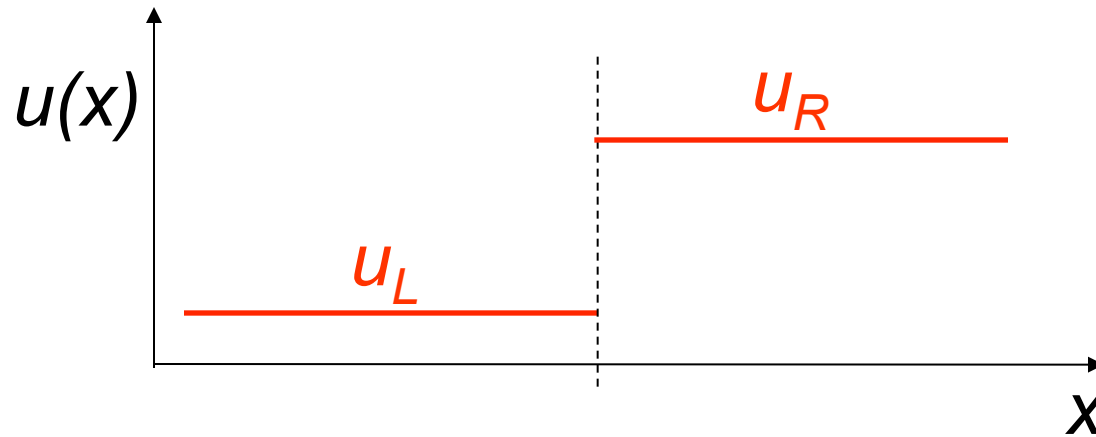
- For Burger's equation $f(u) = u^2/2$, one finds the shock speed as

$$S = \frac{u_L + u_R}{2}$$

- A shock wave is an abrupt discontinuous transition between two states ('upstream' and 'downstream') and it is best described by the integral representation.

Burger Equation: Rarefaction Waves

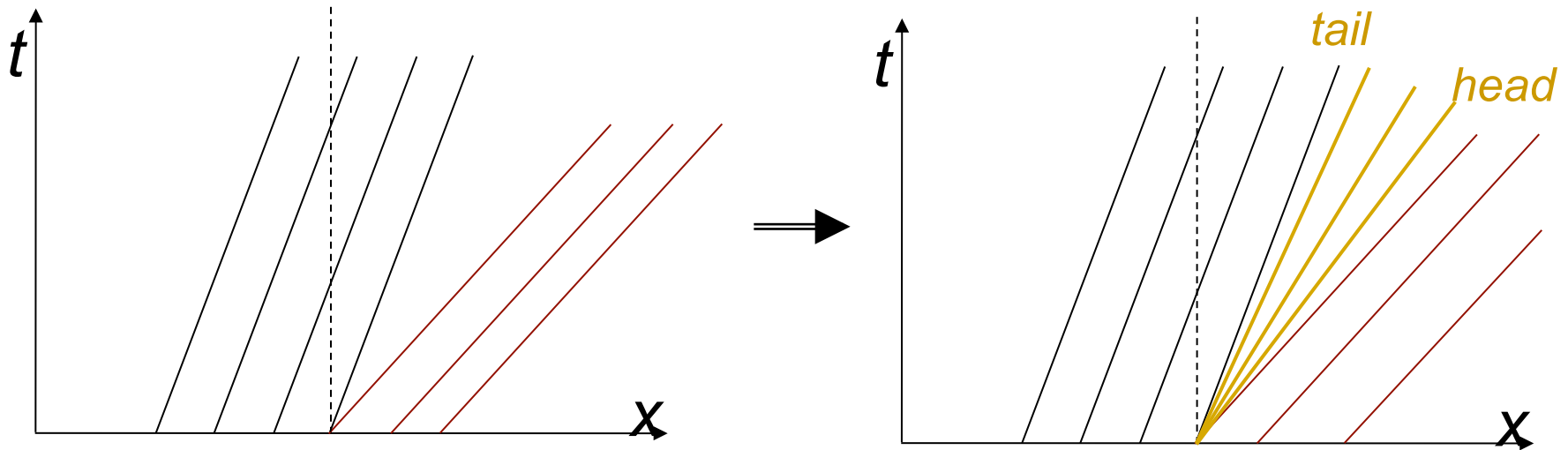
- Let's consider the opposite situation:



- Here, the characteristic velocities on the left are smaller than those on the right.

Burger Equation: Rarefaction Waves

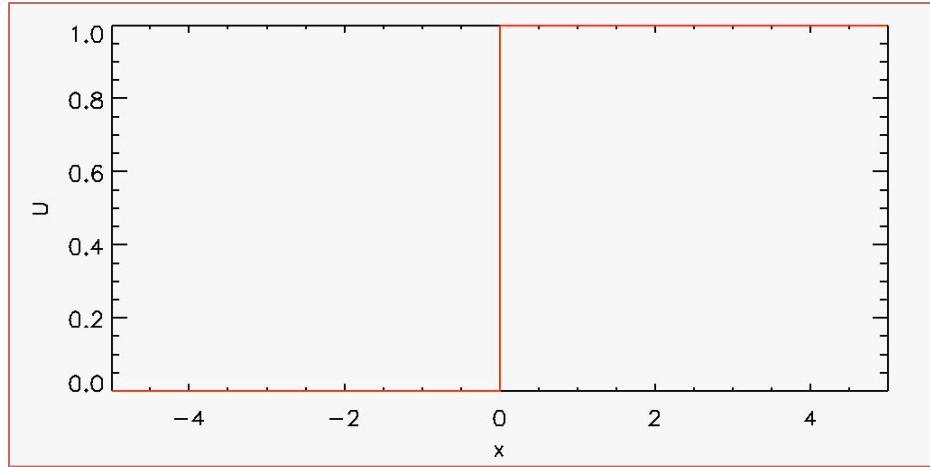
- Now the characteristics will diverge:



- Putting a shock wave between the two states would be incorrect, since it would violate the entropy condition. Instead, the proper solution is a rarefaction wave.

Burger Equation: Rarefaction Waves

- A rarefaction wave is a nonlinear wave that smoothly connects the left and the right state. It is an expansion wave.
- The solution can only be self-similar and takes on the range of values between u_L and u_R .
- The head of the rarefaction moves at the speed $\lambda(u_R)$, whereas the tail moves at the speed $\lambda(u_L)$.
- The general condition for a rarefaction wave is $\lambda(u_L) < \lambda(u_R)$
- Both rarefactions and shocks are present in the solutions to the Euler equation. Both waves are nonlinear.



Burger Equation: Riemann Solver

- These results can be used to write the general solution to the Riemann problem for Burger's equation:

- If $u_L > u_R$ the solution is a discontinuity (shock wave). In this case

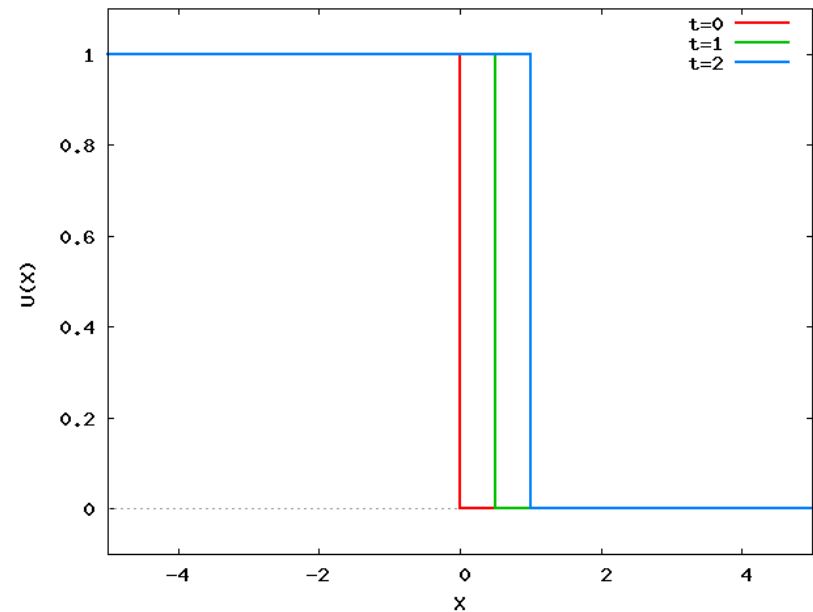
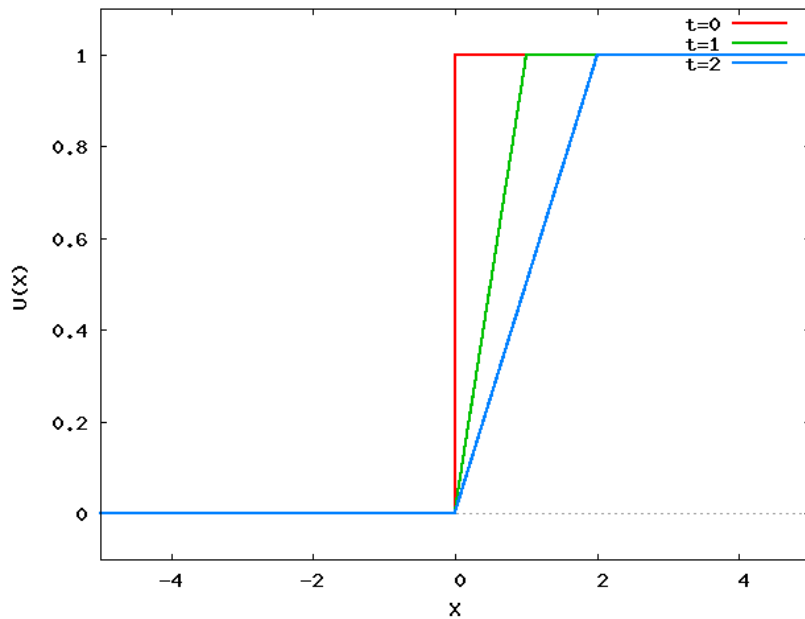
$$u(x, t) = \begin{cases} u_L & \text{if } x - St < 0 \\ u_R & \text{if } x - St > 0 \end{cases}, \quad S = \frac{u_L + u_R}{2}$$

- If $u_L < u_R$ the solution is a rarefaction wave. In this case

$$u(x, t) = \begin{cases} u_L & \text{if } x/t \leq u_L \\ x/t & \text{if } u_L < x/t < u_R \\ u_R & \text{if } x/t > u_R \end{cases}$$

Nonlinear Advection Equation

- Solutions look like



- for a rarefaction and a shock, respectively.

Code Example

- File name: *burger.c*¹
- Purpose: solve Burger's equation with 1st- or 2nd- order Godunov method.

- Usage:

```
> gcc burger.c -lm -o burger
> ./burger
```

- Output: two-column ascii data files "data.nnnn.out"
- Visualization: gnuplot (\rightarrow burger.gp).

```
7 #include <stdio.h>
8 #include <stdlib.h>
9 #include <string.h>
10 #include <math.h>
11 #include <stdlib.h>
12
13 double Initial_Condition (double);
14 void Integrate (double *, double, int, int);
15 int Output (double *, double *, int, int);
16
17 #define NGHOST 2
18 #define ORDER 2
19 #define NX 1600
20
21 /* ***** */
22 int main()
23 /*
24 *
25 *
26 *
27 * ***** */
28 {
29     int i, nstep, out_freq;
30     int ibeg = NGHOST; /* First active zone */
31     int iend = ibeg + NX - 1; /* Last active zone */
32     int noutput = 10; /* Number of desired output files (+1) */
33     double xbeg = -5.0; /* Start domain */
34     double xend = 5.0; /* End domain */
35     double tstop = 8.0; /* Final time */
36     double cfl = 0.9; /* Courant number */
37     double x[NX + 2*NGHOST], dx;
38     double u[NX + 2*NGHOST];
39     double t, dt, dtdx;
40     double umax;
41
42     /* -- 1. Generate grid -- */
43
44     dx = (xend - xbeg)/(double)NX;
45     for (i = 0; i <= iend + NGHOST; i++){
46         x[i] = xbeg + (0.5 + i - ibeg)*dx;
47         u[i] = Initial_Condition (x[i]);
48     }
49
50     /* -- 2. start computation -- */
51
52     t = 0.0; nstep = 0;
53     while (t <= tstop){
54
55         /* -- 2a. Set time step dt = Ca*dx/|u| -- */
56
57         umax = 0.0;
58         for (i = ibeg; i <= iend; i++){
59             if (fabs(u[i]) > umax) umax = fabs(u[i]);
60         }
61         dt = cfl*dx/umax;
```

¹<http://personalpages.to.infn.it/~mignone/Astrosim2019/>

6. NONLINEAR SYSTEMS OF CONSERVATION LAW

Nonlinear Systems

- Much of what is known about the numerical solution of hyperbolic systems of nonlinear equations comes from the results obtained in the linear case or simple nonlinear scalar equations.
- The key idea is to exploit the conservative form and assume the system can be locally “frozen” at each grid interface.
- However, this still requires the solution of the Riemann problem, which becomes increasingly difficult for complicated set of hyperbolic P.D.E.

Euler Equations

- System of conservation laws describing conservation of mass, momentum and energy:

$$\begin{array}{ll} \frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{v}) = 0 & \text{(mass)} \\ \frac{\partial (\rho \mathbf{v})}{\partial t} + \nabla \cdot [\rho \mathbf{v} \mathbf{v} + \mathbf{I} p] = 0 & \text{(momentum)} \\ \frac{\partial E}{\partial t} + \nabla \cdot [(E + p) \mathbf{v}] = 0 & \text{(energy)} \end{array}$$

- Total energy density E is the sum of thermal + Kinetic terms:

$$E = \rho \epsilon + \rho \frac{\mathbf{v}^2}{2}$$

- Closure requires an Equation of State (EoS).

For an ideal gas one has $\rho \epsilon = \frac{p}{\Gamma - 1}$

Euler Equations: Characteristic Structure

- The equations of gasdynamics can also be written in “quasi-linear” or primitive form. In 1D:

$$\frac{\partial \mathbf{V}}{\partial t} + A \cdot \frac{\partial \mathbf{V}}{\partial x} = 0, \quad A = \begin{pmatrix} v_x & \rho & 0 \\ 0 & v_x & 1/\rho \\ 0 & \rho c_s^2 & v_x \end{pmatrix}$$

where $\mathbf{V} = [\rho, v_x, p]$ is a vector of primitive variable, $c_s = (\gamma p / \rho)^{1/2}$ is the adiabatic speed of sound.

- It is called “quasi-linear” since, differently from the linear case where we had $A = \text{const}$, here $A = A(\mathbf{V})$.

Euler Equations: Characteristic Structure

- The quasi-linear form can be used to find the eigenvector decomposition of the matrix A :

$$\mathbf{r}^1 = \begin{pmatrix} 1 \\ -c_s/\rho \\ c_s^2 \end{pmatrix}, \quad \mathbf{r}^2 = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}, \quad \mathbf{r}^3 = \begin{pmatrix} 1 \\ c_s/\rho \\ c_s^2 \end{pmatrix}$$

- Associated to the eigenvalues:

$$\lambda^1 = v_x - c_s, \quad \lambda^2 = v_x, \quad \lambda^3 = v_x + c_s$$

- These are the characteristic speeds of the system, i.e., the speeds at which information propagates.
- Even if they're not rigorously constant, they tell us a lot about the structure of the solution.

Euler Equations: Riemann Problem

- By looking at the expressions for the right eigenvectors,

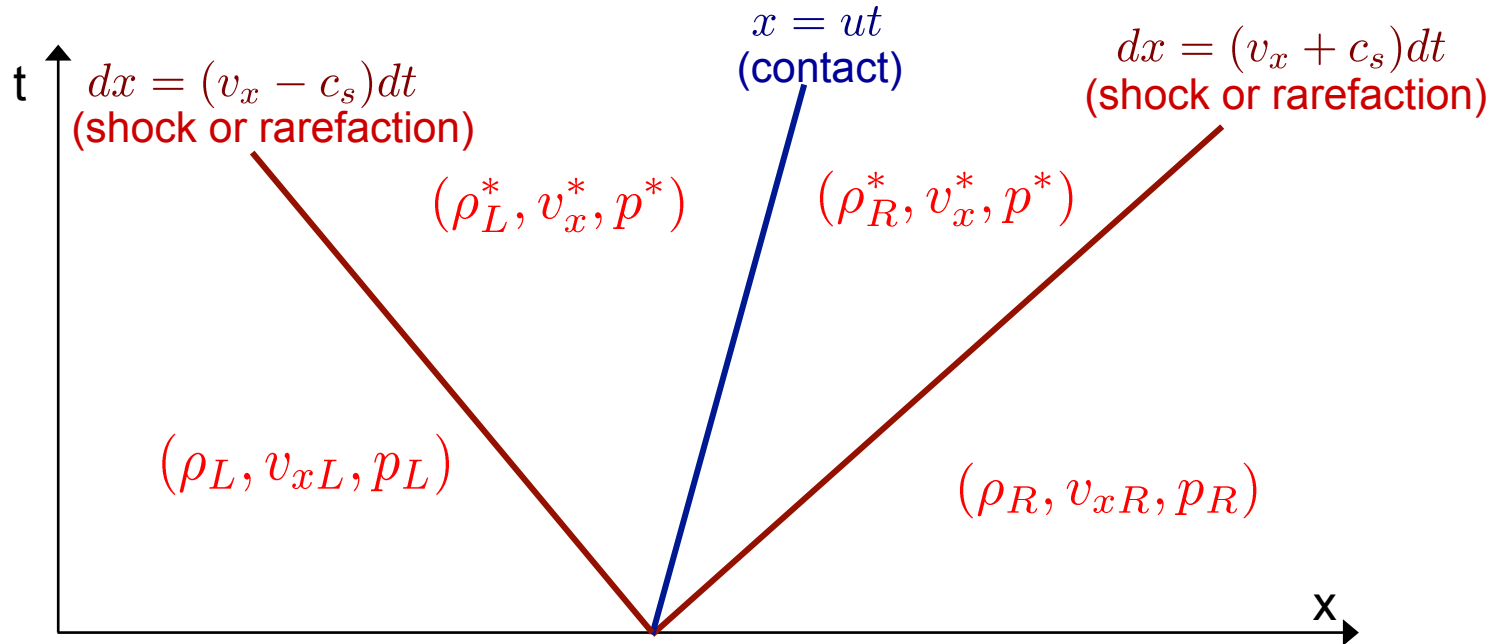
$$\mathbf{r}^1 = \begin{pmatrix} 1 \\ -c_s/\rho \\ c_s^2 \end{pmatrix}, \quad \mathbf{r}^2 = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}, \quad \mathbf{r}^3 = \begin{pmatrix} 1 \\ c_s/\rho \\ c_s^2 \end{pmatrix}$$

we see that across waves 1 and 3, all variables jump. These are nonlinear waves, either shocks or rarefaction waves.

- Across wave 2, only density jumps. Velocity and pressure are constant. This defines the [*contact discontinuity*](#).
- The characteristic curve associated with this linear wave is $dx/dt = u$, and it is a straight line. Since v_x is constant across this wave, the flow is neither converging or diverging.

Euler Equations: Riemann Problem

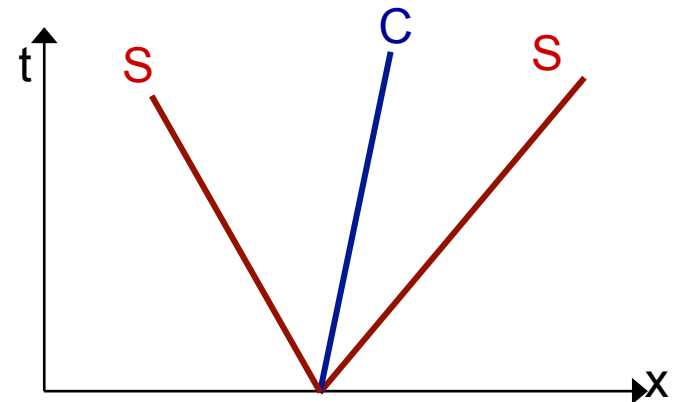
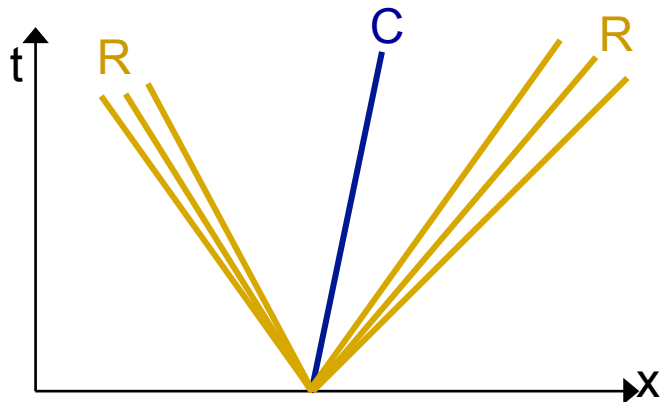
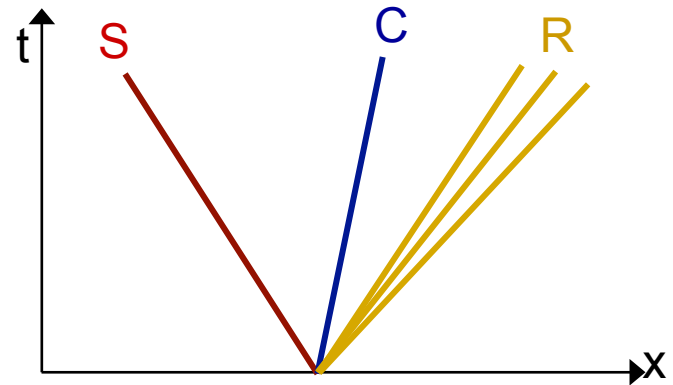
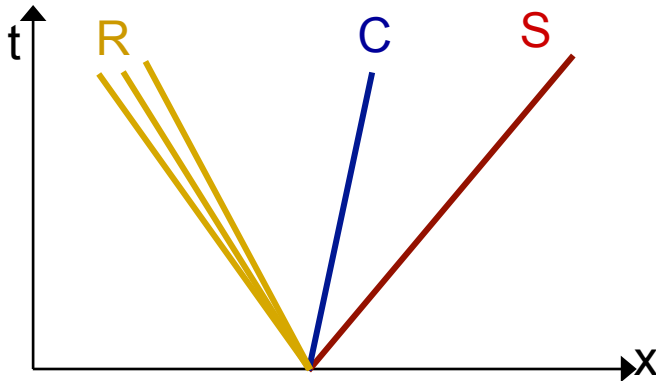
- The solution to the Riemann problem looks like



- The outer waves can be either shocks or rarefactions.
- The middle wave is always a contact discontinuity.
- In total one has 4 unknowns: $\rho_L^*, \rho_R^*, v_x^*, p^*$, since only density jumps across the contact discontinuity.

Possible Wave Patterns

- Depending on the initial discontinuity, a total of 4 patterns can emerge from the solution:



Exact Solution to the Riemann Problem

- For the Euler equations of gas-dynamics an exact solution to the Riemann problem exists (see the book by Toro, sec. 4.2) and it boils down to the following nonlinear algebraic equation for p^* :

$$f_L(p^*, \mathbf{W}_L) + f_R(p^*, \mathbf{W}_R) + u_R - u_L = 0$$

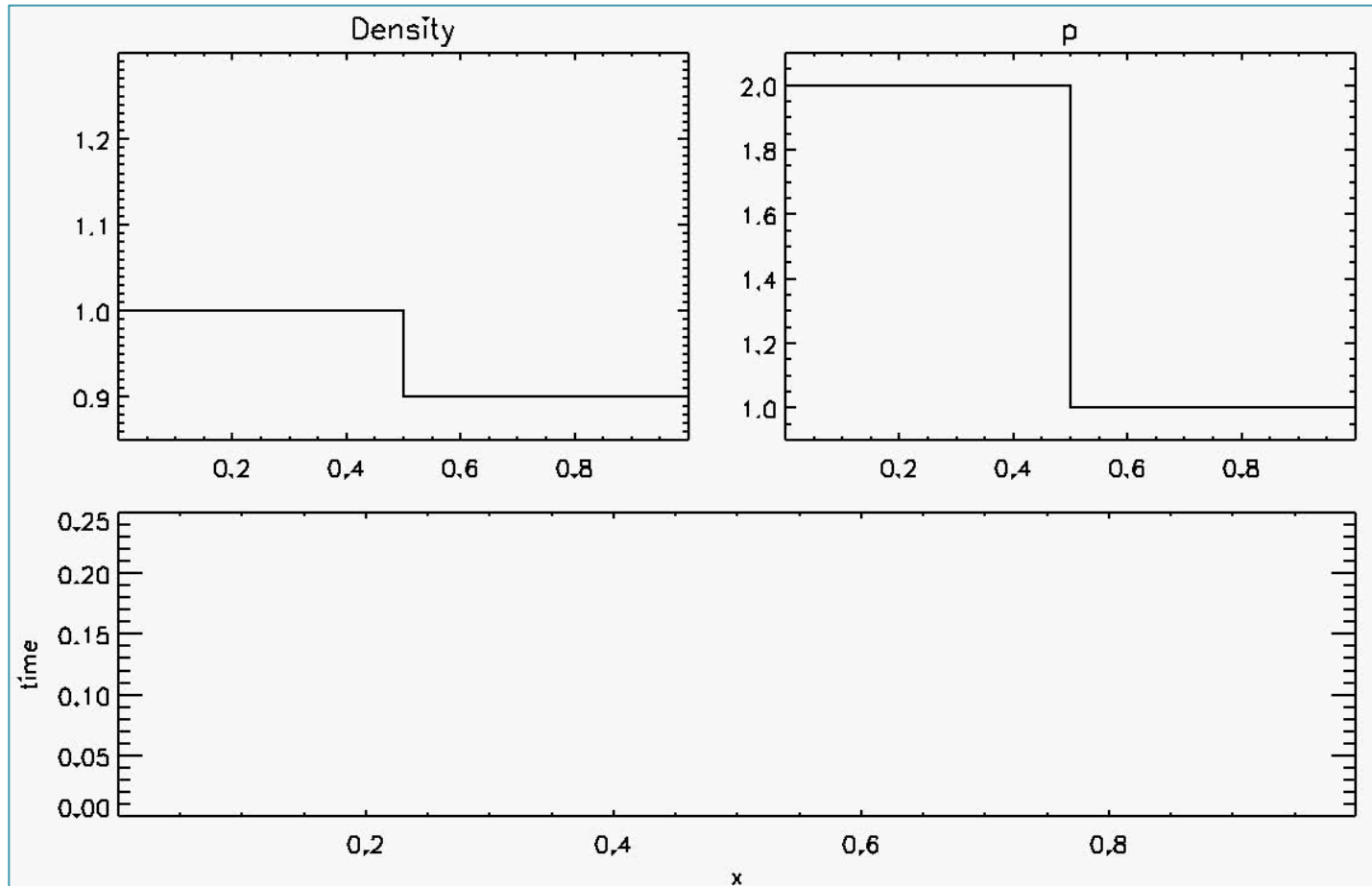
where

$$f_L(p, \mathbf{W}_L) = \begin{cases} (p - p_L) \left[\frac{A_L}{p + B_L} \right]^{\frac{1}{2}} & \text{if } p > p_L \text{ (shock) ,} \\ \frac{2a_L}{(\gamma-1)} \left[\left(\frac{p}{p_L} \right)^{\frac{\gamma-1}{2\gamma}} - 1 \right] & \text{if } p \leq p_L \text{ (rarefaction) ,} \end{cases}$$
$$f_R(p, \mathbf{W}_R) = \begin{cases} (p - p_R) \left[\frac{A_R}{p + B_R} \right]^{\frac{1}{2}} & \text{if } p > p_R \text{ (shock) ,} \\ \frac{2a_R}{(\gamma-1)} \left[\left(\frac{p}{p_R} \right)^{\frac{\gamma-1}{2\gamma}} - 1 \right] & \text{if } p \leq p_R \text{ (rarefaction) ,} \end{cases}$$

- The functions f_L and f_R governs relations across the left and right non-linear waves and serves to connect the unknown particle speed u^* to the known states L/R .

Euler Equations: Shock Tube Problem

- The decay of the discontinuity defines what is usually called the “shock tube problem”,

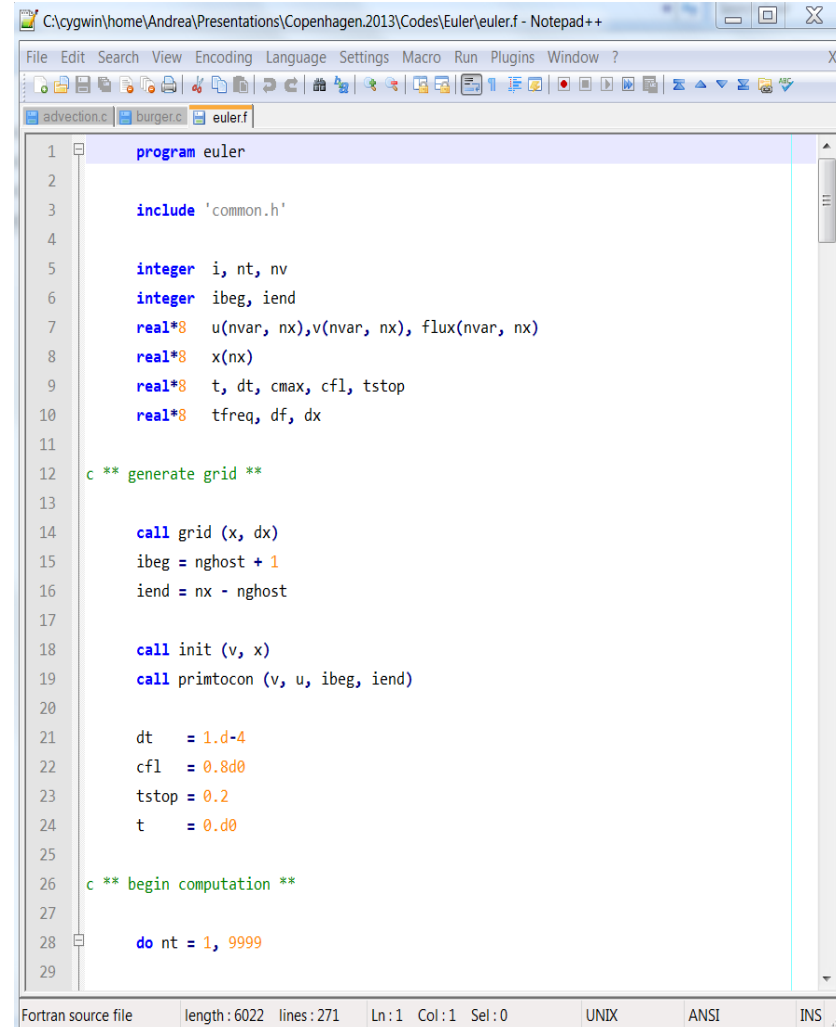


Code Example

- File name: euler.f¹
- Purpose: solve 1D Euler's equation using a 1st-order Lax-Friedrichs or HLLC method.
- Usage:

```
> gfortran -fdec-math euler.f -o euler  
> ./euler
```

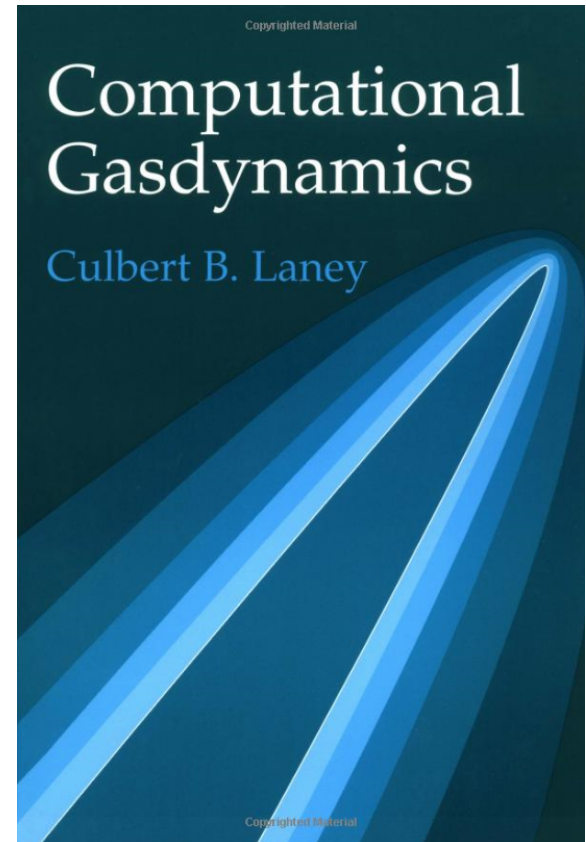
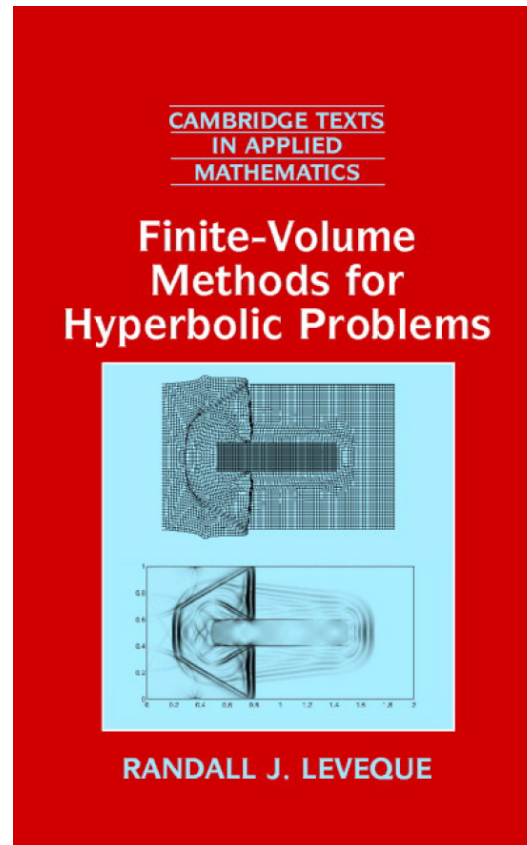
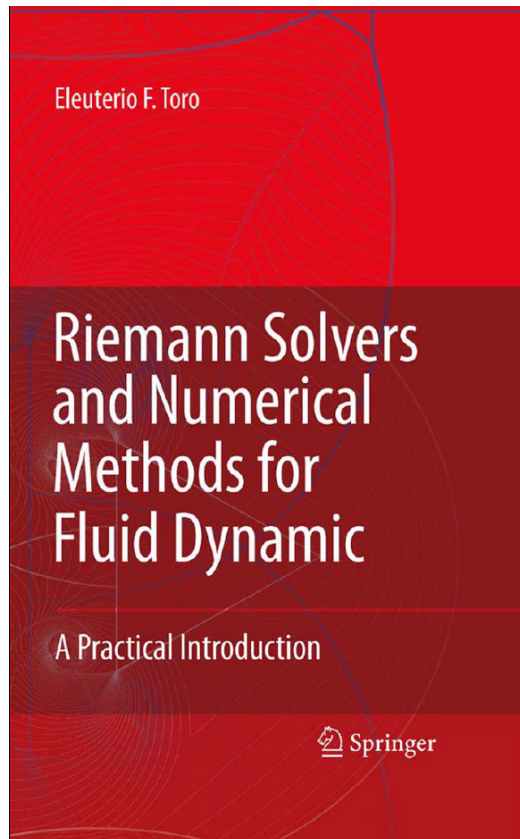
- Output:
4-column ascii data files "data.out"



```
1 program euler  
2  
3 include 'common.h'  
4  
5 integer i, nt, nv  
6 integer ibeg, iend  
7 real*8 u(nvar, nx), v(nvar, nx), flux(nvar, nx)  
8 real*8 x(nx)  
9 real*8 t, dt, cmax, cfl, tstop  
10 real*8 tfreq, df, dx  
11  
12 c ** generate grid **  
13  
14 call grid (x, dx)  
15 ibeg = nghost + 1  
16 iend = nx - nghost  
17  
18 call init (v, x)  
19 call primtocon (v, u, ibeg, iend)  
20  
21 dt = 1.d-4  
22 cfl = 0.8d0  
23 tstop = 0.2  
24 t = 0.d0  
25  
26 c ** begin computation **  
27  
28 do nt = 1, 9999  
29
```

¹<http://personalpages.to.infn.it/~mignone/Astrosim2019/>

Recommended Books



THE END
