PDEs – Astrophysical Fluid Dynamics II

Lecture 12, Computational Astrophysics (ASTR660)

Hsiang-Yi Karen Yang, NTHU, 12/01/2022

Announcements

- ***HW5*** will be posted on eLearn TODAY. ***Due at 14:20 on 12/8 (Thu)***. Late submission within one week will receive ***75%*** of the credit
- Please help to provide comments/feedback to improve this course! The questionnaire can be found via this link or by scanning this QR code:

Previous lecture...

- A system can be described as a fluid if it continuously flows and is ***collisional***, i.e., if the ***Knudsen number*** is small:

$$K_n \equiv \frac{\lambda}{L} = \frac{mean\ free\ path}{typical\ scale} \ll 1 \qquad \text{or} \qquad \lambda \ll L$$

- Hydrodynamic equations are conservation laws for mass, momentum, and energy:

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \boldsymbol{v}) = 0 \qquad \text{Continuity equation}$$

$$\frac{\partial \rho \boldsymbol{v}}{\partial t} + \nabla \cdot (\rho \boldsymbol{v}\boldsymbol{v} + P \cdot \boldsymbol{I}) = 0 \qquad \text{Momentum equation}$$

$$\frac{\partial \rho E}{\partial t} + \nabla \cdot [(\rho E + P)\boldsymbol{v}] = 0 \qquad \text{Total energy equation}$$

Previous lecture…

- The *equation of state (EOS)* of ideal gas is often used as a *closure* for the hydrodynamic equations
- Hydrodynamic equations can be described in different views
  - *Eulerian*: stand still as fluid moves by
  - *Lagrangian*: move with the fluid
- Fluid properties & phenomena:
  - Solution for linear perturbations: *sound waves*
  - Solutions for nonlinear waves: *shocks, rarefactions, and contact discontinuities*
  - Fluid instabilities: *Kelvin-Helmholtz instability, Rayleigh-Taylor instability, Richtmyer-Meshkov instability*

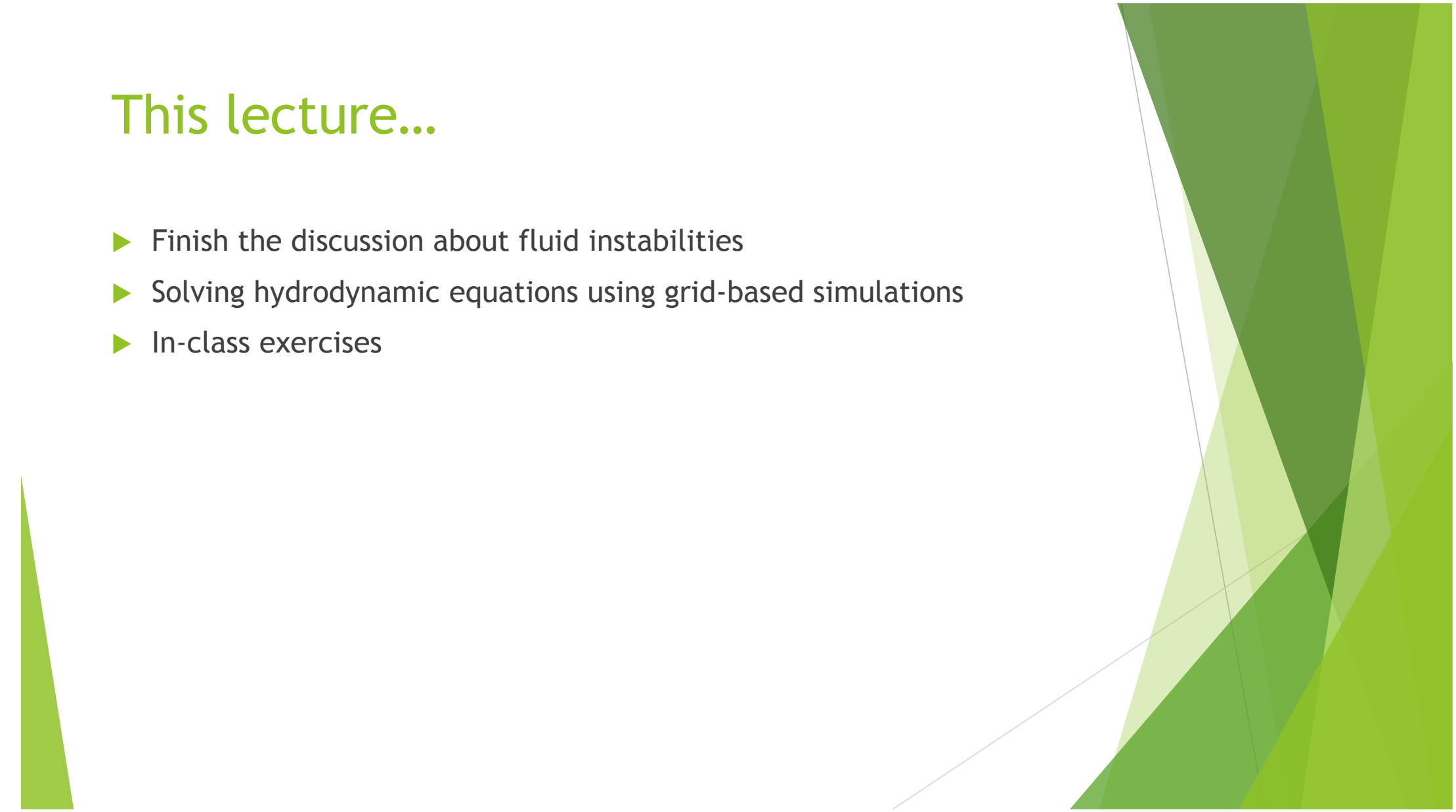
$$u = \rho\epsilon = \frac{P}{\gamma - 1} = \frac{nk_B T}{\gamma - 1}$$

Convective derivative

$$\frac{D}{Dt} \equiv \frac{\partial}{\partial t} + \boldsymbol{v} \cdot \nabla$$

This lecture…

- Finish the discussion about fluid instabilities
- Solving hydrodynamic equations using grid-based simulations
- In-class exercises



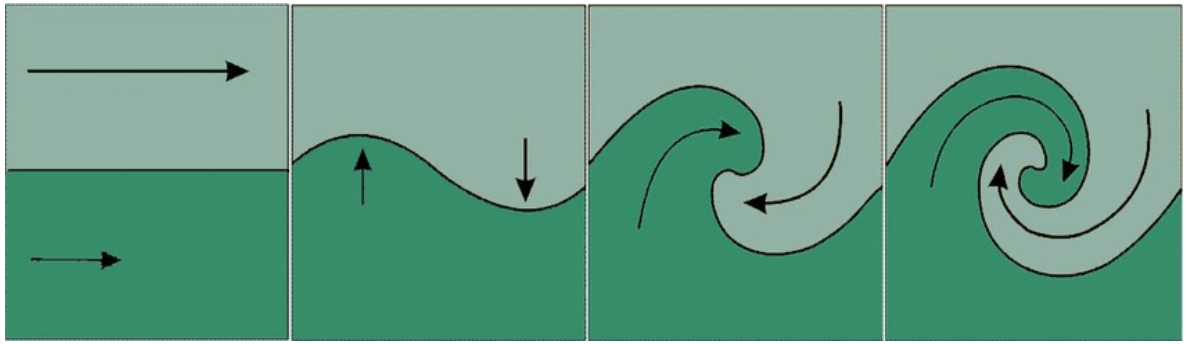
Fluid properties and phenomena

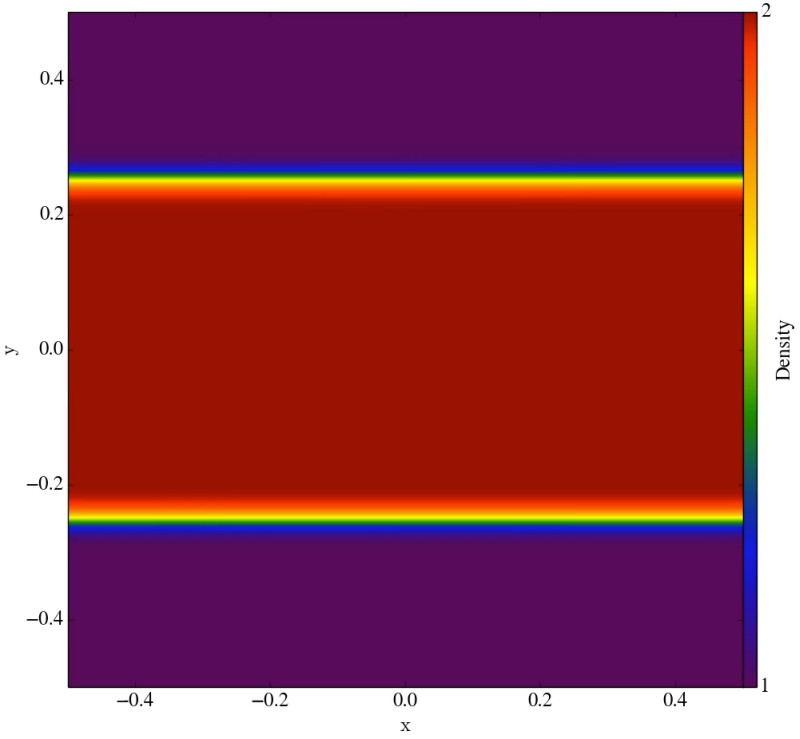
Fluid instabilities

- A fluid is ***unstable*** if small perturbations grow with time rather than damping out
- Many different types of instabilities associated with different forcing mechanisms, flow regimes, geometries, etc
- Key questions regarding fluid instabilities:
  - What is the criterion for instability?
  - How fast do linear perturbations grow? Is the growth rate a function of wavenumber?
  - What is the nonlinear development of perturbations like?
  - What additional effects can stabilize the fluid?

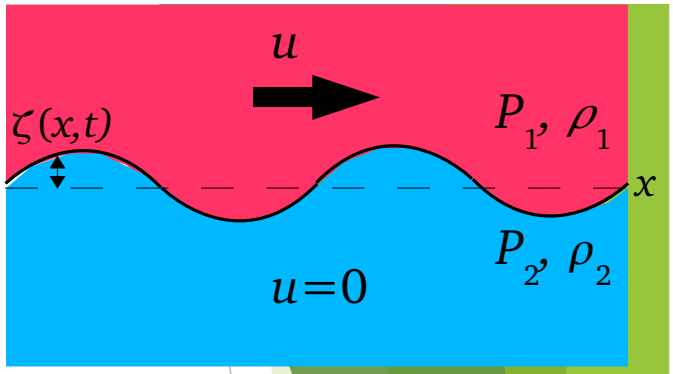
Kelvin-Helmholtz instability

- Arises in ***shear flow along a contact discontinuity***
- Often produces ***wave/ripple-like*** structures





Kelvin-Helmholtz instability



- Let $y = \zeta(x,t)$ be the displacement of the surface
- Linearized Euler equations for an incompressible ($\nabla \cdot v = 0$) fluid are:

$$\nabla \cdot \boldsymbol{\delta u} = 0$$

$$\frac{\partial \boldsymbol{\delta u}}{\partial t} + u\frac{\partial \boldsymbol{\delta u}}{\partial x} = -\frac{1}{\rho}\nabla \delta P$$

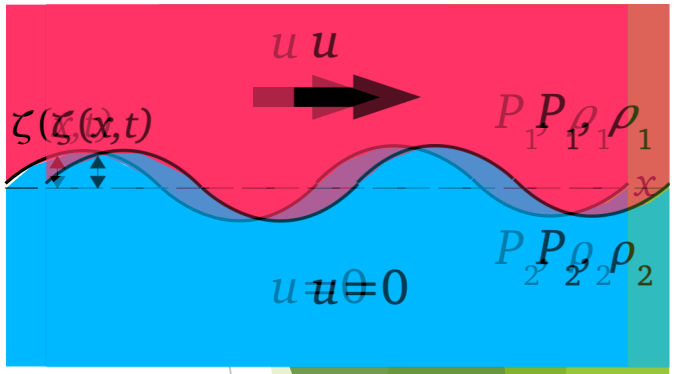
- Thus δP satisfies $\nabla^2 \delta P = 0$. Look for solution $\delta P = f(y)e^{i(kx-\omega t)}$:

$$\frac{d^2 f}{dy^2} - k^2 f = 0 \quad \rightarrow \quad f(y) \propto e^{\pm ky}$$

- Let $y > 0$ on side 1: $\delta P_1 \propto e^{i(kx-\omega t)}e^{-ky}$

$$\delta u_y = \frac{k\,\delta P_1}{i\,\rho_1 (ku-\omega)}$$

Kelvin-Helmholtz instability



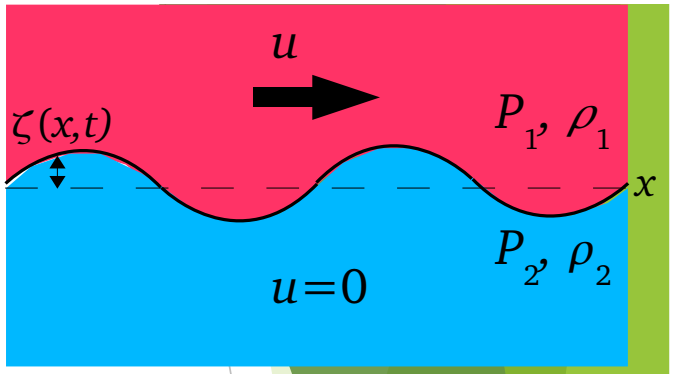
- Note that (from the movement of $\zeta(x,t)$): $\frac{\partial \zeta}{\partial t} = \delta u_y - u\frac{\partial \zeta}{\partial x}$
- If $\delta u_y = i\zeta(ku-\omega)$, $\zeta \propto e^{i(kx-\omega t)}$ is solution to the above equation. Thus

$$\delta P_1 = -\zeta \rho_1 \frac{(ku-\omega)^2}{k}$$

- Similarly for side 2: $\delta P_2 = \zeta \rho_2 \frac{\omega^2}{k}$
- Requiring $\delta P_1 = \delta P_2$ on the boundary gives: $\omega = ku\frac{\rho_1 \pm i\sqrt{\rho_1\rho_2}}{\rho_1+\rho_2}$

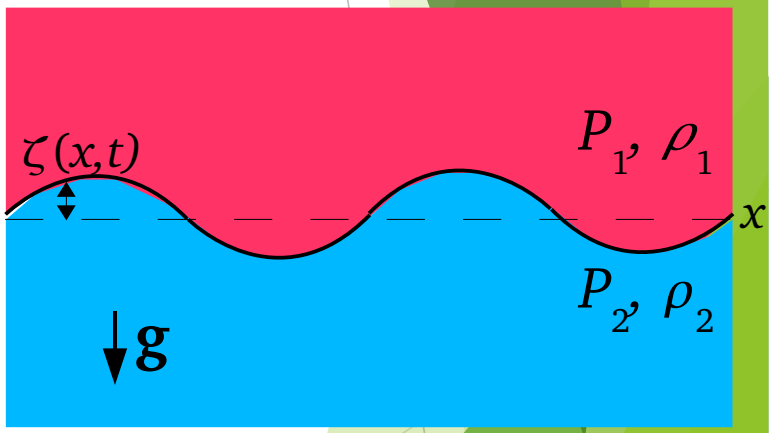
Kelvin-Helmholtz instability

- So we have $\zeta \propto e^{i(kx-\omega t)}$, $\omega = ku\dfrac{\rho_1 \pm i\sqrt{\rho_1\rho_2}}{\rho_1+\rho_2}$
- There are always frequencies with a positive imaginary part => ***all modes are unstable***
- Modes with larger *k* or ***shorter wavelengths grow fastest***
- If there is some ***surface tension*** on the interface (e.g., magnetic field, viscosity) as a restoring force, small-wavelength perturbations can be stabilized



Rayleigh-Taylor instability

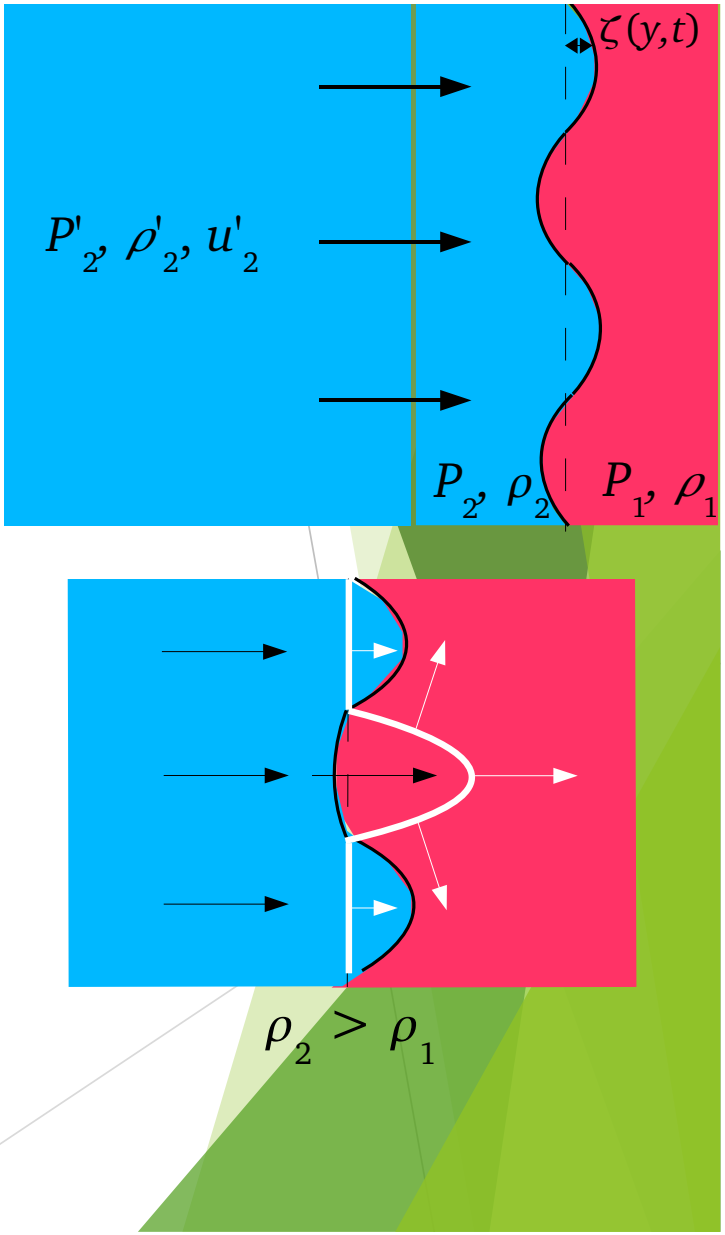
- Arises when a ***denser fluid overlies a less dense fluid***
- Often produces ***plume/finger-like*** structures
- Again, one can consider two incompressible fluids with sinusoidal perturbation at the interface. Both are at rest, and $\rho_1 > \rho_2$. A constant gravitational field $\boldsymbol{g}$ is present.
- This configuration is always unstable if there is no surface tension



Rayleigh-Taylor instability & non-linear development

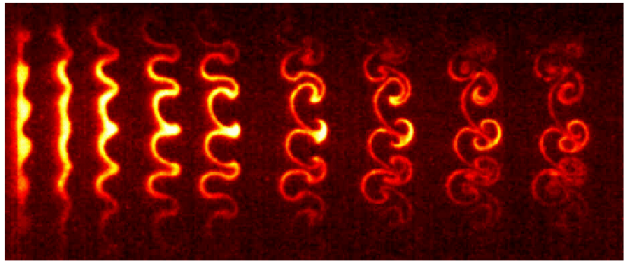
Richtmyer-Meshkov instability

- Arises when a ***shock passes through a perturbed interface between fluids of different densities***
- What happens?
  - Different parts of the shock hit the interface at different times
  - The parts that have reached the interface will travel at a different speed than other parts
  - Multiple intersecting shocks will form on the other side of the interface
  - The interface will also stretch in a way similar to Rayleigh-Taylor instability

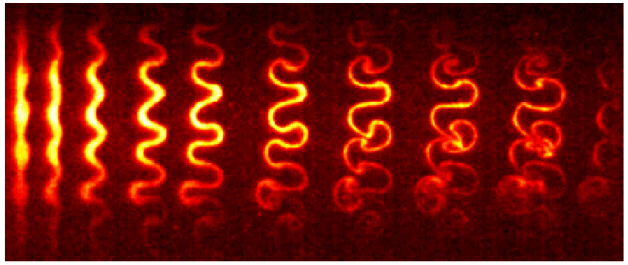


Richtmyer-Meshkov instability

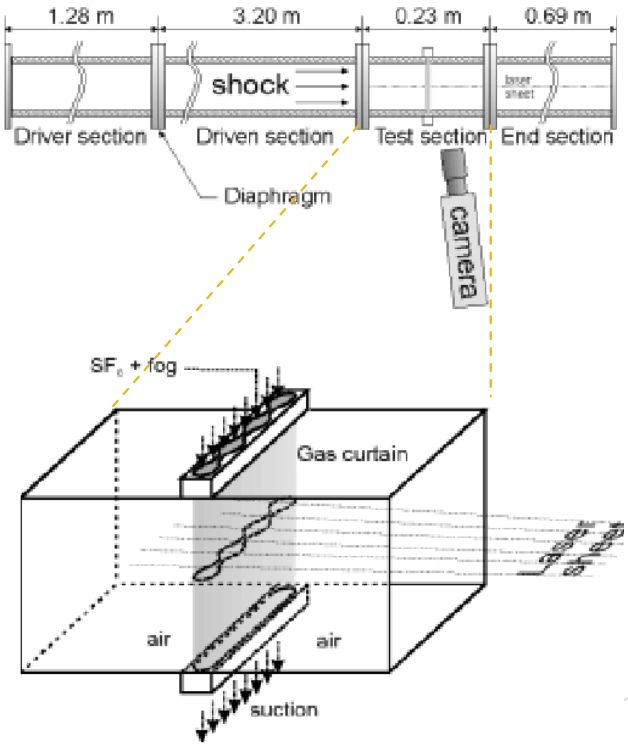
- Often produces ***mushroom-like*** structures
- Demonstration of Richtmyer-Meshkov instability using the gas-curtain experiment:



Downstream Mushrooms

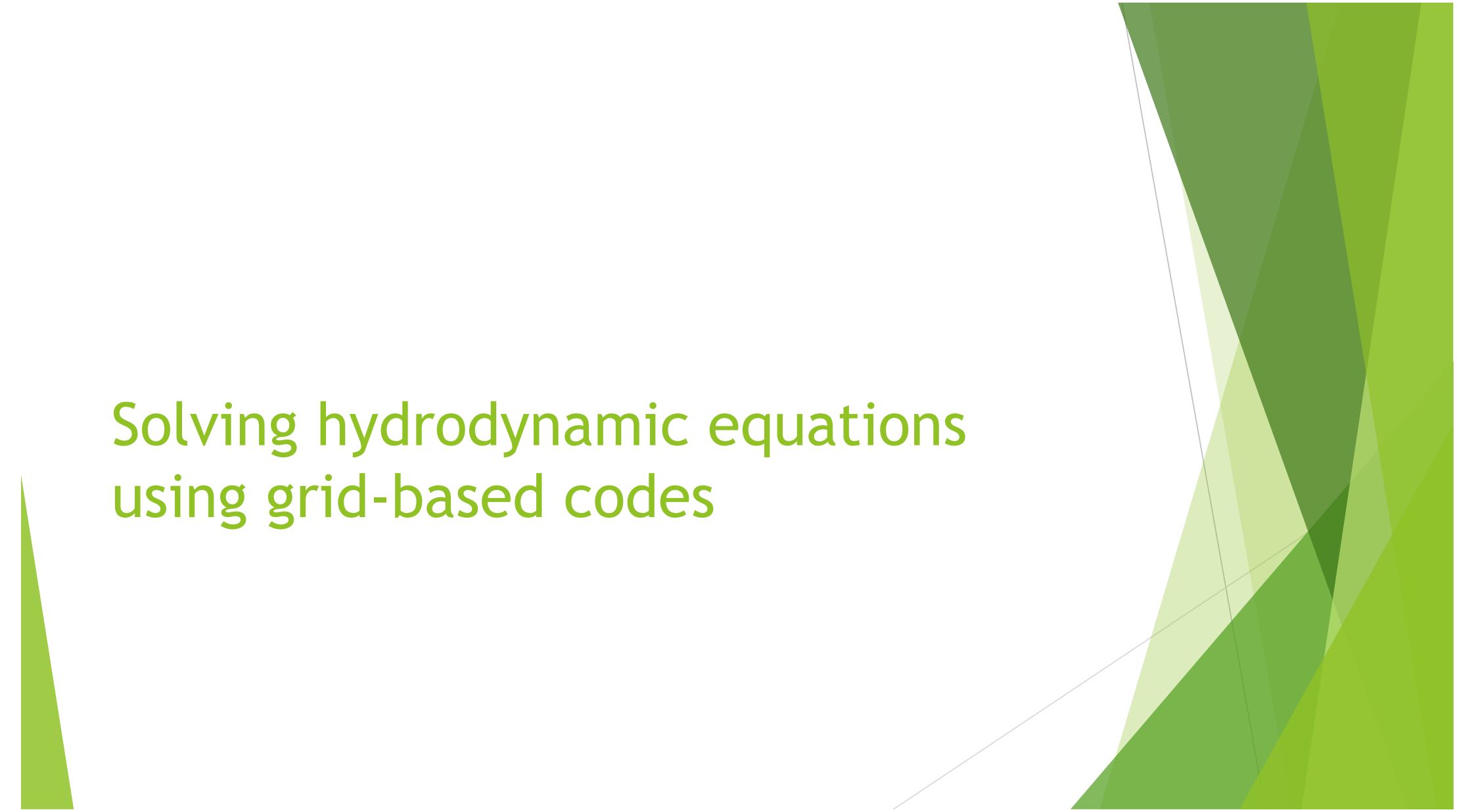


Sinuous Mode

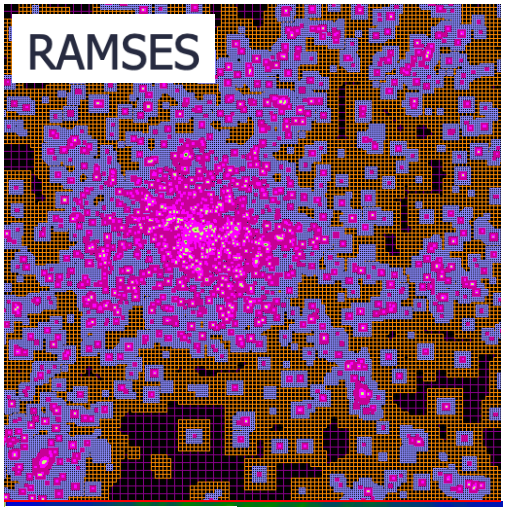


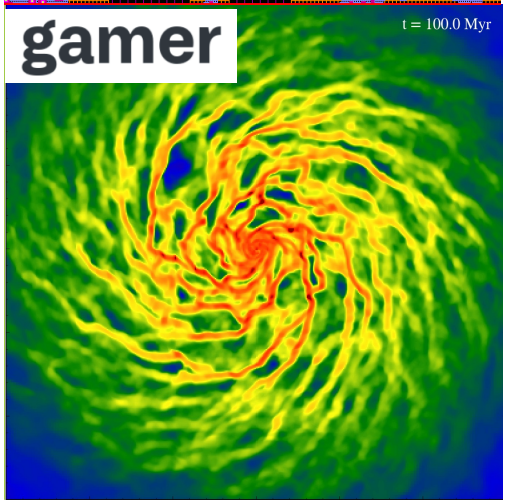
Rightley, Benjamin, & Vorobieff (1997)

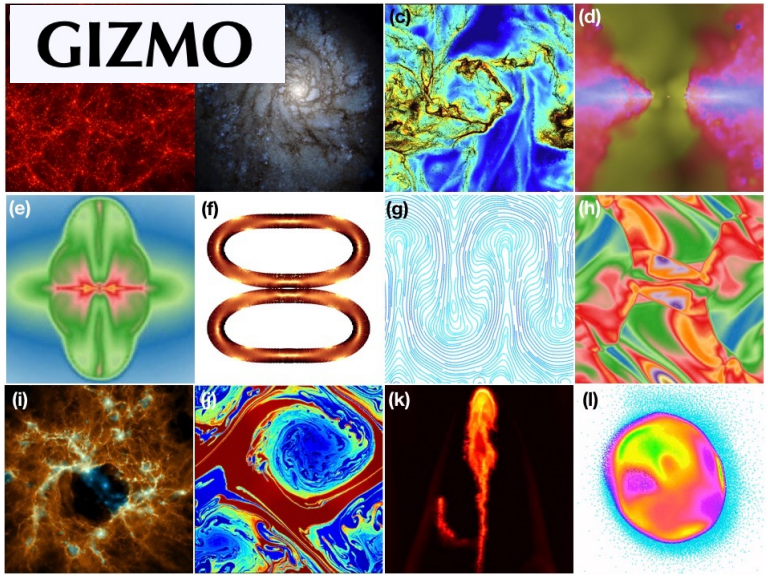
Solving hydrodynamic equations using grid-based codes

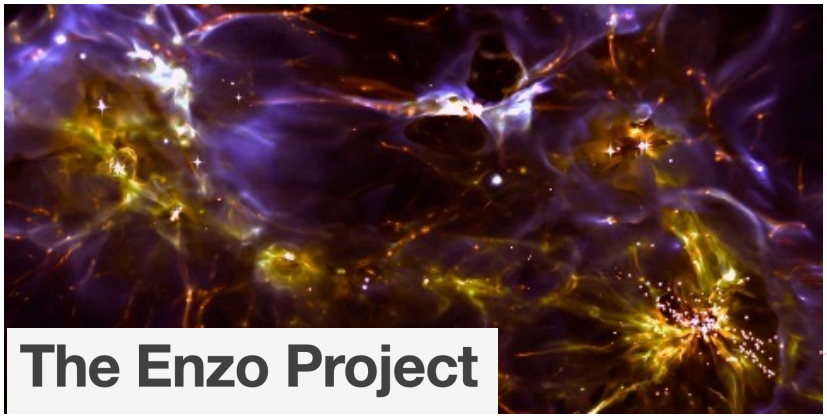


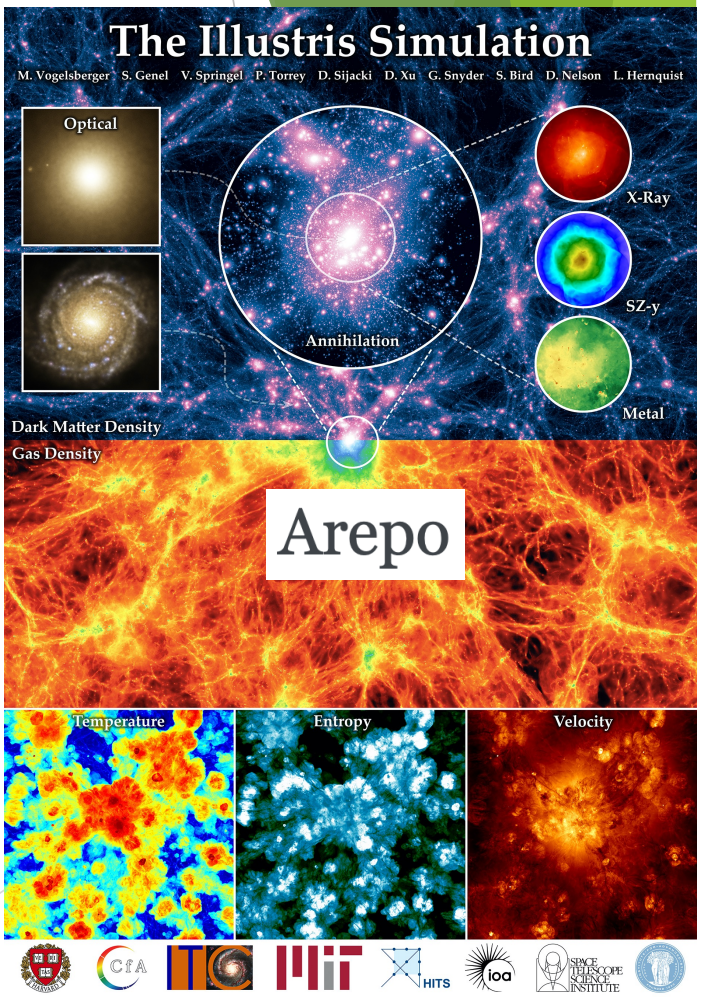
Grid-based astrophysical hydrodynamic codes











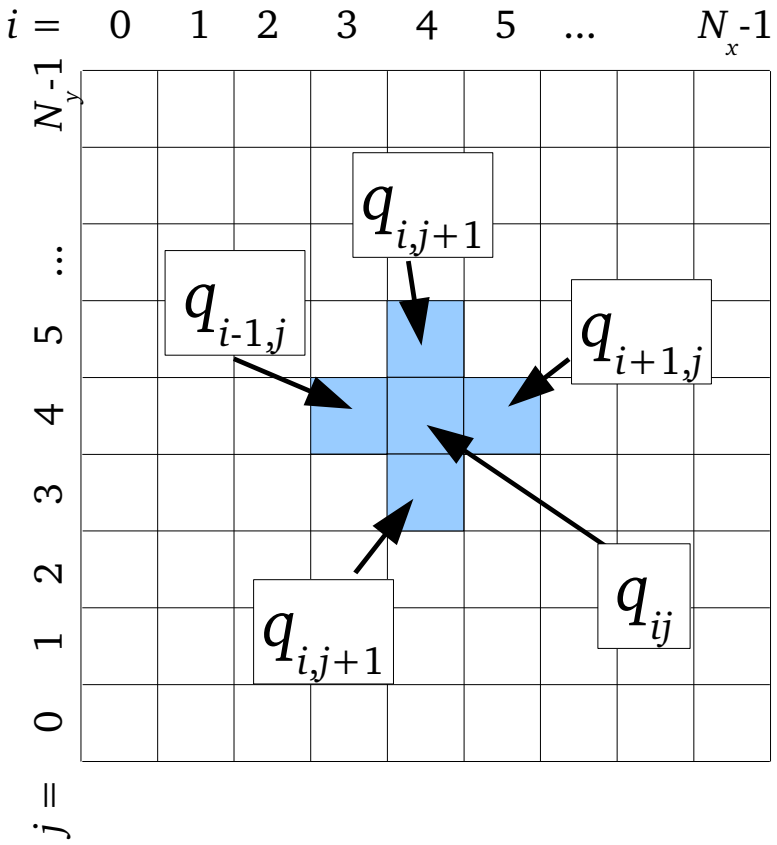
Finite-volume methods

- This method is widely used ***grid-based*** hydro codes
- Mesh points -> ***grid cells***
- $q_i^n = q(t = t_n, x = x_i)$ now represents ***cell-averaged*** values instead of cell-centered values:

$$q_{ij} \equiv \frac{1}{\Delta x \Delta y} \int_{x_{i-1/2}}^{x_{i+1/2}} \int_{y_{j-1/2}}^{y_{j+1/2}} q(x, y)\, dx\, dy$$

- We can write down a general ***conservation*** law:

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

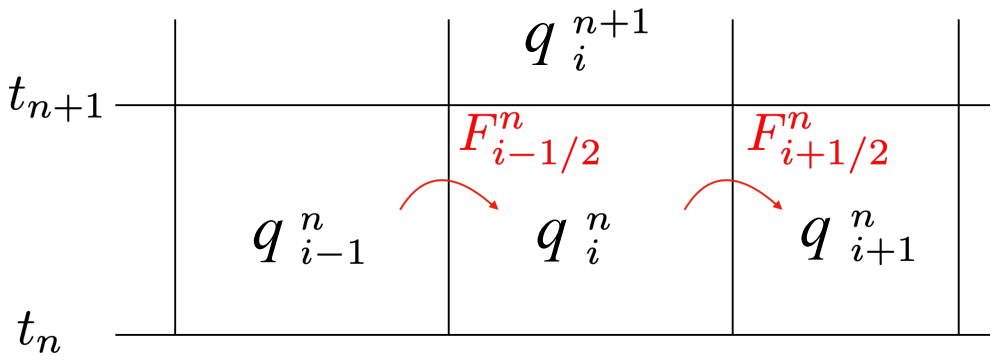


Finite-volume methods

- Cell values can be updated by evaluating ***fluxes*** at cell edges (in 1D):

★ *Step 2*

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



- The above equation is exact. So the problem becomes how to ★ ***approximate the fluxes at cell edges!!***

Step 1

Recall

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

***Note that the fluxes here represent *time-averaged* fluxes across the cell edges between t_n and t_{n+1}!!

Example: advection equation $u_t = -cu_x$

- Rewrite the advection equation in the conservative form $\frac{\partial q}{\partial t} + \nabla \cdot \boldsymbol{F}(q,t) = 0$

  then $\boldsymbol{q = u, F(q,t) = cu}$

- Now use the ***finite-volume*** method: $q_i^{n+1} = q_i^n - \frac{\Delta t}{\Delta x}\left(F_{i+1/2}^n - F_{i-1/2}^n\right)$

- The simplest approach to approximate fluxes at cell edges as simple ***arithmetic averages***:

$$F_{i-1/2}^n = \frac{1}{2}\left[F(q_{i-1}^n) + F(q_i^n)\right]$$

-> This method is unstable!

Algorithms for evaluating the fluxes

Lax-Friedrichs method:

$$F^n_{i-1/2} = \frac{1}{2}\left[F(q^n_{i-1}) + F(q^n_i)\right] - \frac{\Delta x}{2\Delta t}\left(q^n_i - q^n_{i-1}\right)$$

Upwind method

$$F^n_{i-1/2} = F(q^n_{i-1})$$

Lax-Wendroff method:

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

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

evolution.f90 - where the update subroutine & fluxes are evaluated

In order to compute fluxes using solutions at previous time step

Should use "uold" as defined in Simulation_data for computing fluxes

```
subroutine update(time, dt)
    use Simulation_data
    implicit none
    real, intent(in) :: time ,dt
    integer :: i
    real    :: FL, FR

    uold  = u
    do i = istart, iend
        call flux(i,dt,FL,FR)
        u(i) = uold(i) - dt/dx*(FR-FL)
    enddo

end subroutine update

!
! Routine to evaluate flux cell edges
!
subroutine flux(i,dt,FL, FR)
    use Simulation_data
    implicit none
    integer, intent(in) :: i
    real, intent(in)    :: dt
    real, intent(out)   :: FL, FR

    ! Arithmetic average method
    FL = 0.5*c*(uold(i-1)+uold(i))
    FR = 0.5*c*(uold(i+1)+uold(i))

    ! The Lax-Friedrichs Method
    FL = 0.5*c*(uold(i-1)+uold(i)) - 0.5*dx/dt*(uold(i)-uold(i-1))
    FR = 0.5*c*(uold(i+1)+uold(i)) - 0.5*dx/dt*(uold(i+1)-uold(i))

    ! The upwind method
    !FL = ! TODO
    !FR = ! TODO

    ! The Lax-Wendroff Method
    !FL = ! TODO
    !FR = ! TODO

end subroutine flux
```

Finite-volume method for hydrodynamic equations

- The finite-volume method is good for solving conservation laws in the form:

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

- For example, we can rewrite the Euler equations in conservation forms:

$$q = \begin{Bmatrix} \rho \\ \rho u_x \\ \rho u_y \\ \rho u_z \\ \rho E \end{Bmatrix} \qquad \boldsymbol{F}(q) = \begin{Bmatrix} \rho u_x & \rho u_y & \rho u_z \\ \rho u_x^2 + P & \rho u_x u_y & \rho u_x u_z \\ \rho u_x u_y & \rho u_y^2 + P & \rho u_y u_z \\ \rho u_x u_z & \rho u_y u_z & \rho u_z^2 + P \\ (\rho E + P) u_x & (\rho E + P) u_y & (\rho E + P) u_z \end{Bmatrix}$$

Finite-volume method for hydrodynamic equations

- We can then easily update the solution for the cell-averaged quantities at cell (i,j,k) by:

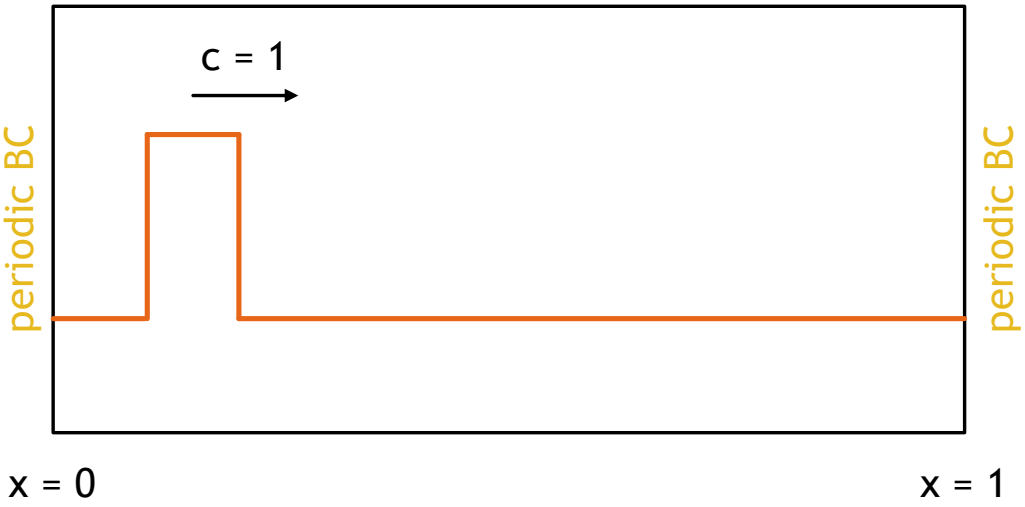
$$q_{ijk}^{n+1} = q_{ijk}^{n} - \Delta t \left\{ \frac{1}{\Delta x}\left[\boldsymbol{F}_{i+1/2,jk}^{n+1/2} - \boldsymbol{F}_{i-1/2,jk}^{n+1/2} \right] + \frac{1}{\Delta y}\left[\boldsymbol{F}_{i,j+1/2,k}^{n+1/2} - \boldsymbol{F}_{i,j-1/2,k}^{n+1/2} \right] + \frac{1}{\Delta z}\left[\boldsymbol{F}_{i,j,k+1/2}^{n+1/2} - \boldsymbol{F}_{i,j,k-1/2}^{n+1/2} \right] \right\}$$

- The above update is done to all 5 fluid quantities, i.e., $\boldsymbol{q} = [\rho, \rho u_x, \rho u_y, \rho u_z, \rho E]$
- Fluxes with superscript n+1/2 represent time-average fluxes
- This equation is exact, so if we had an exact expression for the time-averaged fluxes then we would be done
- In practice we have to approximate the fluxes (e.g., Lax-Friedrichs, Lax-Wendroff, etc), which would introduce truncation errors

Recall last exercise – solving advection equation using finite-volume methods

$$u_t = -cu_x$$

- Let's propagate the top-hat function using the ***finite-volume*** methods

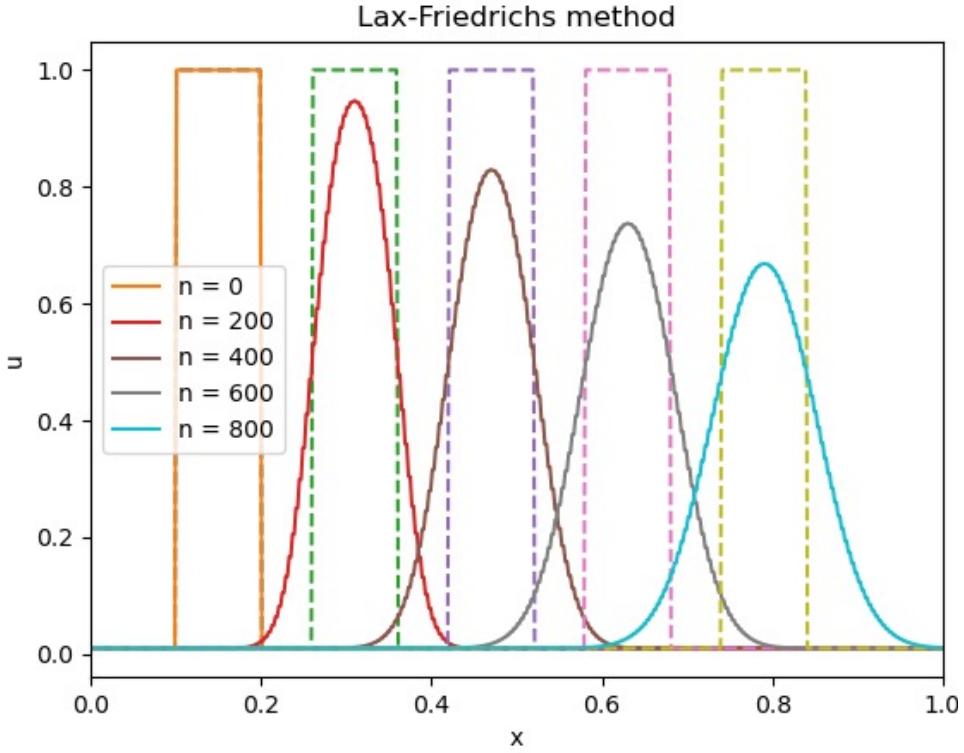


$$\frac{\partial q}{\partial t} + \nabla \cdot \boldsymbol{F}(q,t) = 0$$

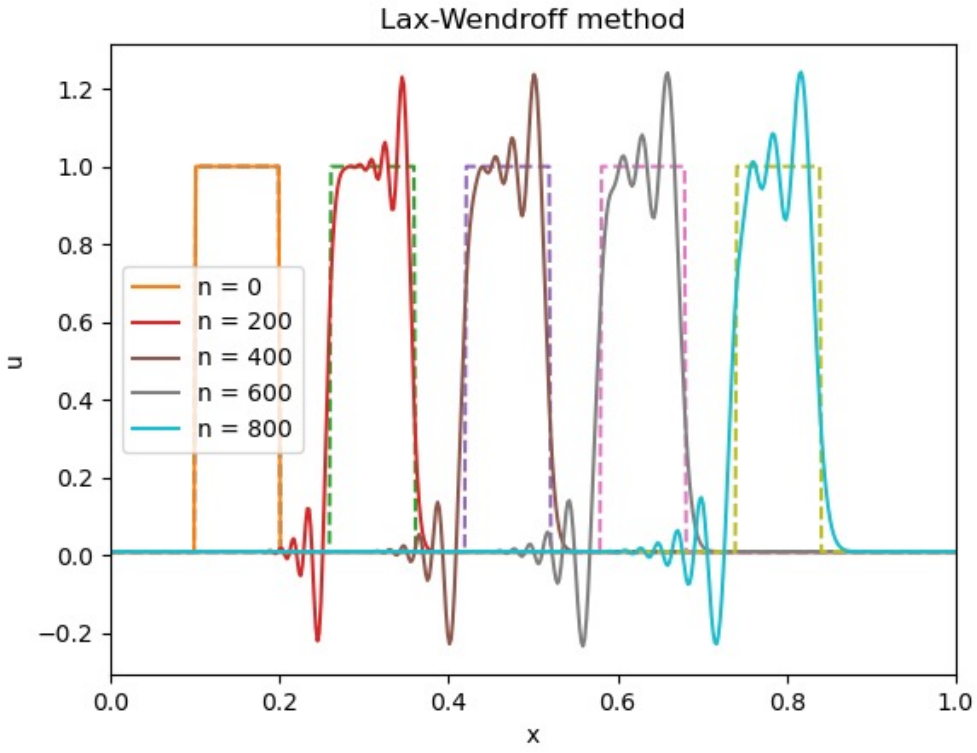
Rewrite the advection equation in the conservative form then

$$\boldsymbol{q = u, F(q,t) = cu}$$

Results of the advection test for a top-hat function



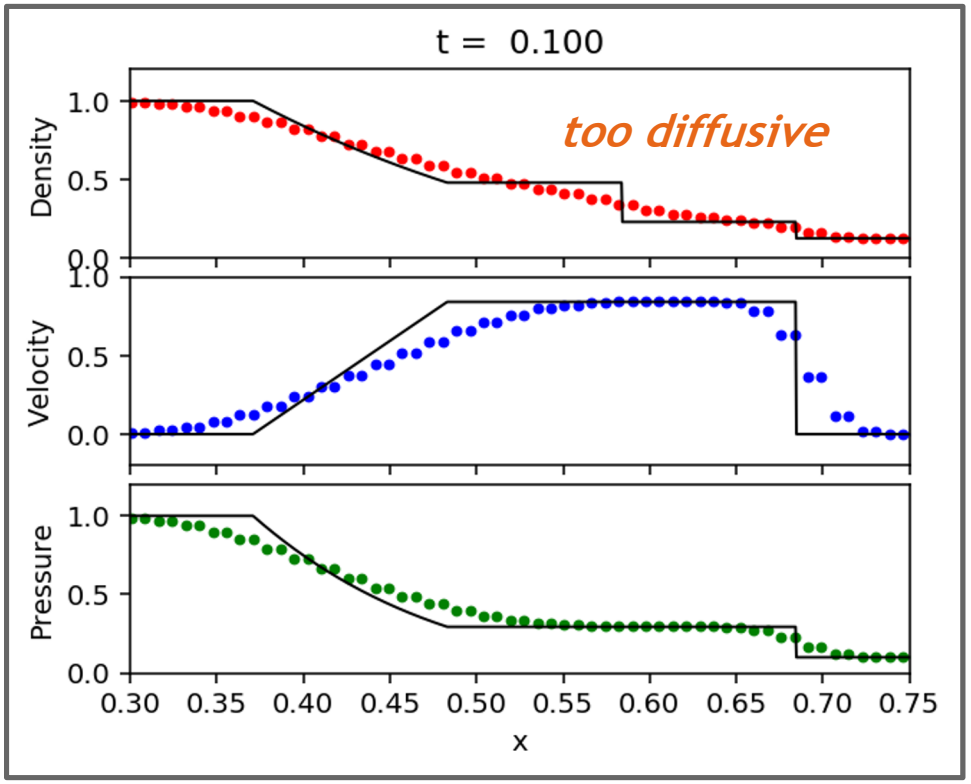
too diffusive



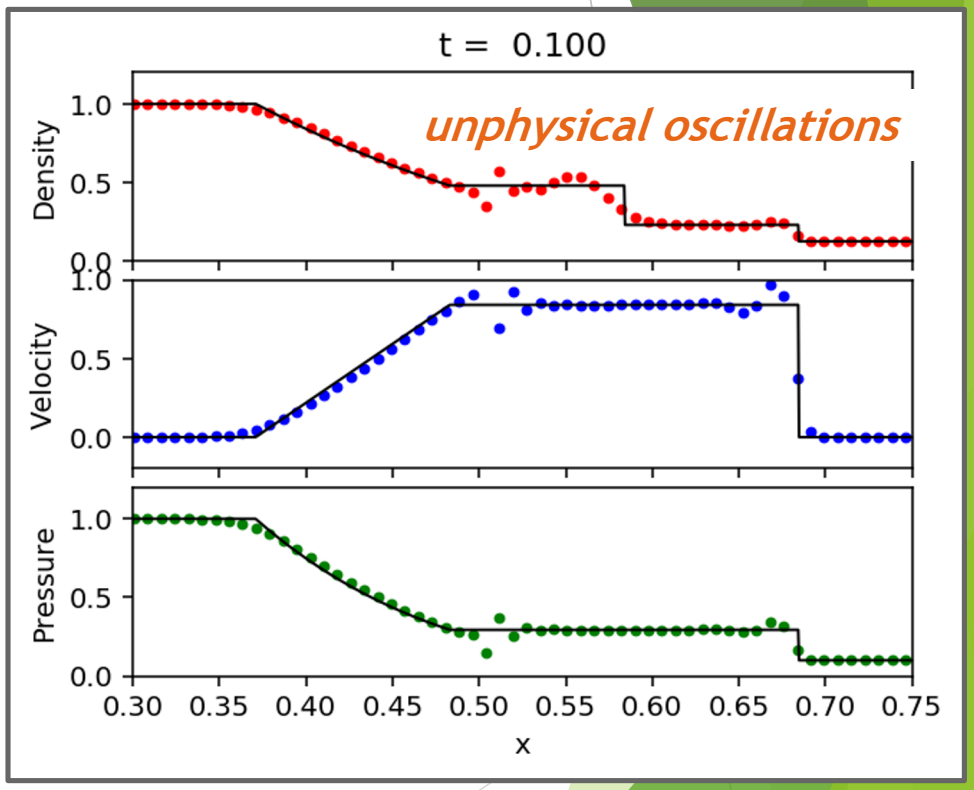
unphysical oscillations

Results of the Sod shocktube test

Lax-Friedrichs method



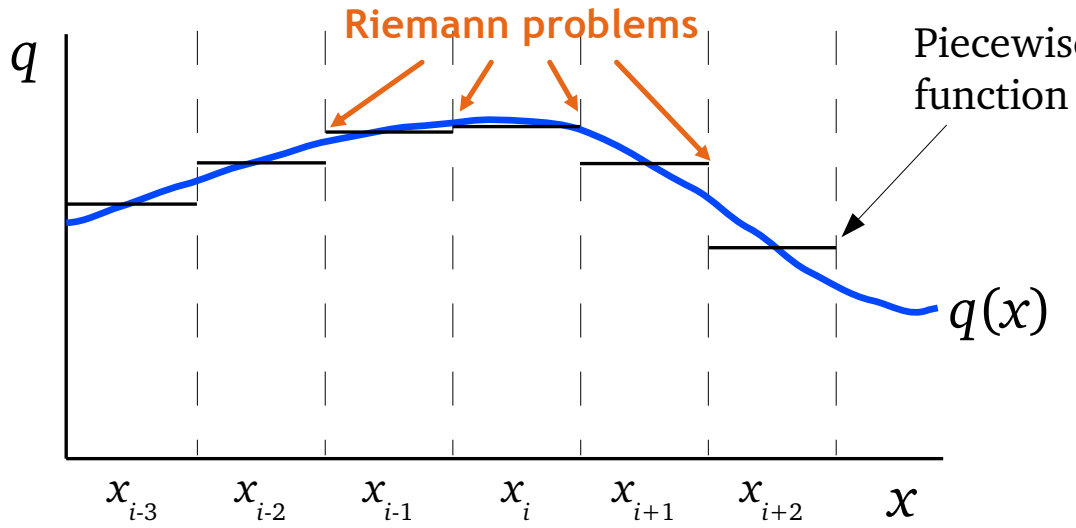
Lax-Wendroff method



=> Need high-resolution shock-capturing schemes!!

Godunov method

- Godunov method is a high-resolution ***shock-capturing*** method for obtaining the fluxes at cell edges
- Procedure:
  - **Step 1**: *approximate* the solution function at t_n as a ***piecewise constant*** function
  - **Step 2**: solve the 1D ***Riemann problem*** at each cell edge => *exact* solution between t_n and t_{n+1} at cell edge
  - **Step 3**: find the time-averaged flux at the cell edges

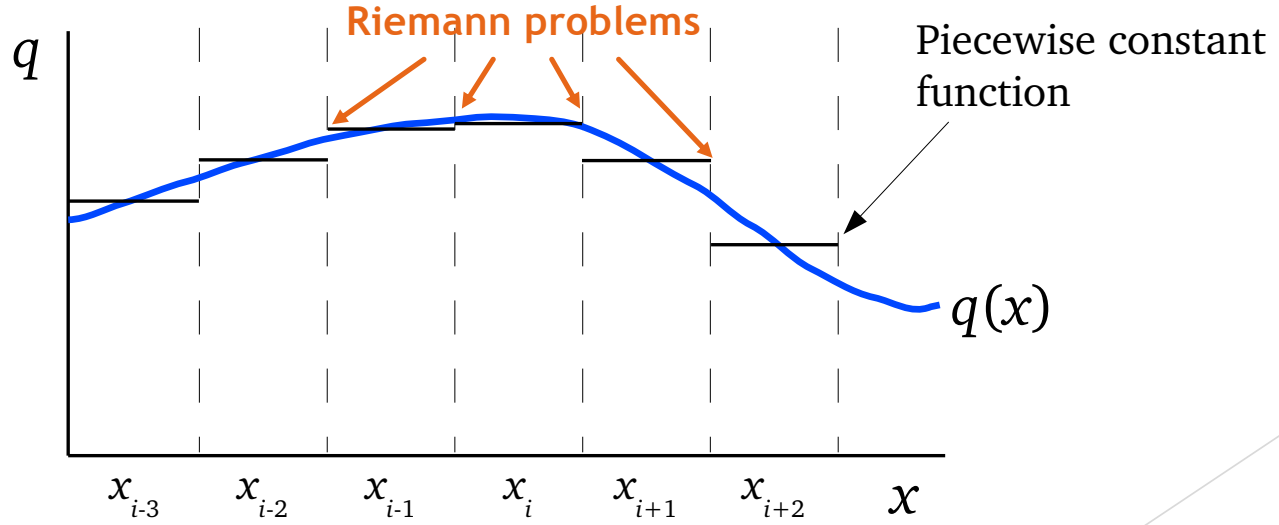


Godunov method

- Godunov method is a high-resolution ***shock-capturing*** method for obtaining the fluxes at cell edges
- Procedure:
  - **Step 1**: *approximate* the solution function at t_n as a ***piecewise constant*** function
  - **Step 2**: solve the 1D ***Riemann problem*** at each cell edge => *exact* solution between t_n and t_{n+1} at cell edge
  - **Step 3**: find the time-averaged flux at the cell edges

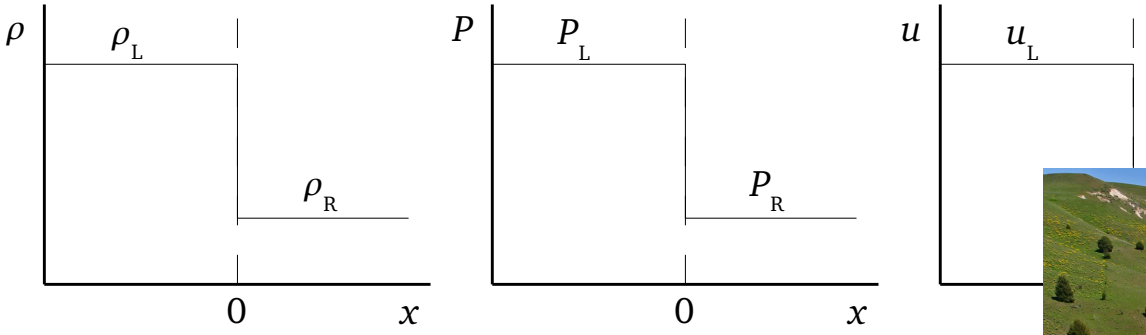
Key Step

This is in fact similar to solving the shocktube problem, but for arbitrary initial conditions



The Riemann problem

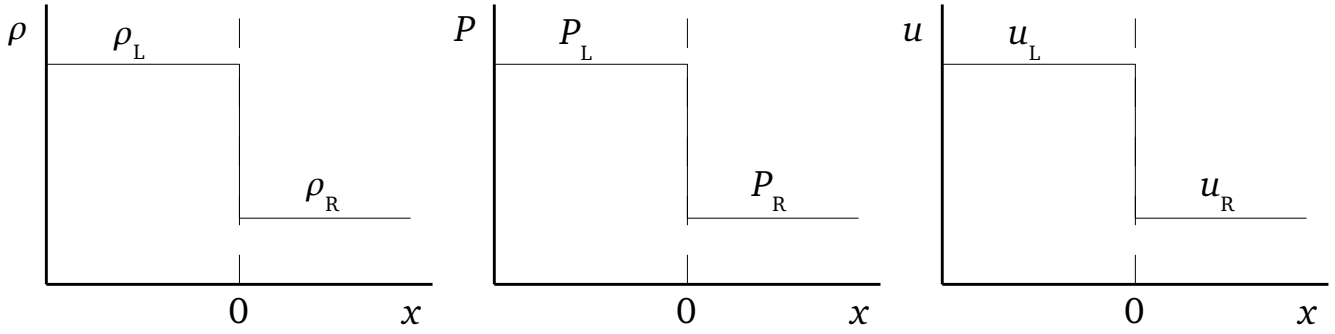
- Goal: find the fluid solution between t_n and t_{n+1} given an ***arbitrary 1D discontinuity*** between a left state and a right state



Analogy: flood discharge of a dam

The Riemann problem

- Goal: find the fluid solution between t_n and t_{n+1} given an ***arbitrary 1D discontinuity*** between a left state and a right state



Q: How to find the solutions?

- Similar to the shocktube problem, we could find the ***characteristics***

Q: How to find the characteristics?

- By solving the ***eigenvalue problem*** for the hydrodynamic equations

The Riemann problem

- Let's find the characteristics for the Euler's equations in 1D:

$$\frac{\partial}{\partial t}\begin{bmatrix}\rho\\ u\\ P\end{bmatrix}+\begin{bmatrix}\frac{\partial \rho u}{\partial x}\\ u\frac{\partial u}{\partial x}+\frac{1}{\rho}\frac{\partial P}{\partial x}\\ u\frac{\partial P}{\partial x}+\gamma P\frac{\partial u}{\partial x}\end{bmatrix}=0$$

- We can write these in the matrix form: $\frac{\partial \boldsymbol{Q}}{\partial t}+\boldsymbol{A}(\boldsymbol{Q})\frac{\partial \boldsymbol{Q}}{\partial x}=0$

where $\boldsymbol{Q}=\begin{bmatrix}\rho\\ u\\ P\end{bmatrix}$ $\boldsymbol{A}=\begin{bmatrix}u & \rho & 0\\ 0 & u & 1/\rho\\ 0 & \rho a^2 & u\end{bmatrix}$

The Riemann problem

- The ***eigenvectors*** and associated ***eigenvalues*** of ***A*** are

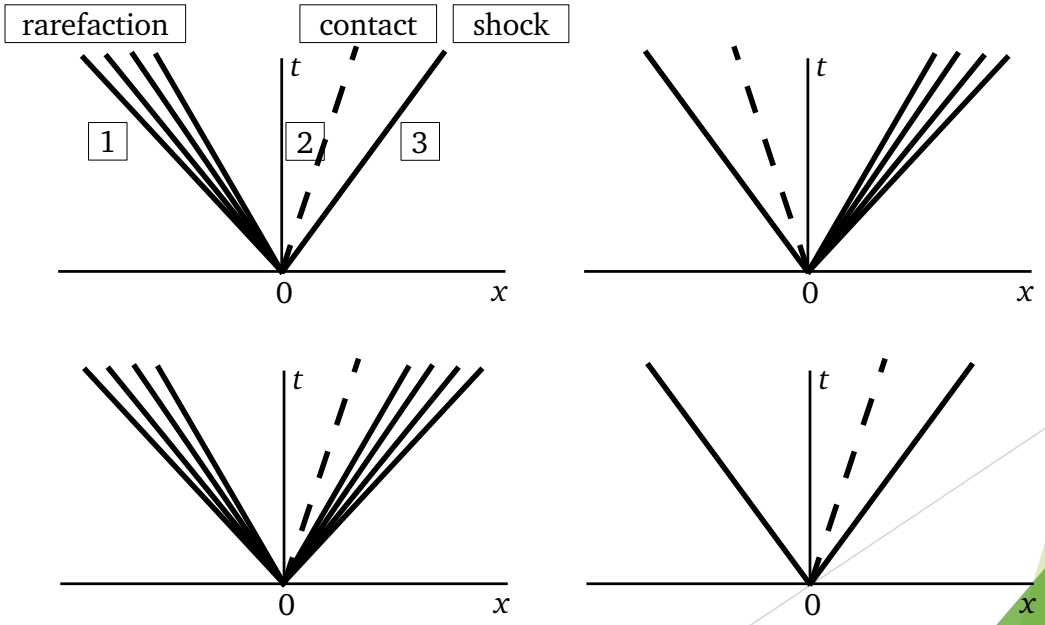
$$\boldsymbol{K}^{(1)}=\begin{bmatrix}1\\-a/\rho\\a^2\end{bmatrix} \qquad \boldsymbol{K}^{(2)}=\begin{bmatrix}1\\0\\0\end{bmatrix} \qquad \boldsymbol{K}^{(3)}=\begin{bmatrix}1\\a/\rho\\a^2\end{bmatrix}$$

$$\lambda_1=u-a \qquad \lambda_2=u \qquad \lambda_3=u+a$$

- Each eigenvalue corresponds to the ***characteristic*** speeds of a nonlinear wave
- Given any initial left state (ρ_L, u_L, P_L) and right state (ρ_R, u_R, P_R), we can decompose the these two states into a superposition of these three nonlinear waves and find their subsequent evolution

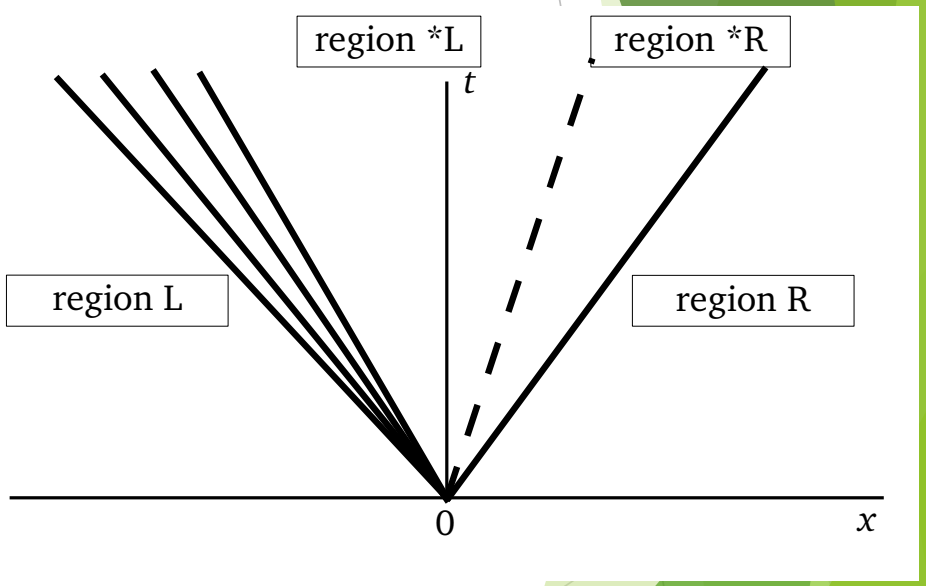
The Riemann problem – possible solutions

- The solution to the Riemann problem always contains three nonlinear waves (one or two amplitudes may be zero). One nonlinear wave is associated with one characteristic
- The center wave is always a contact discontinuity; the outer wave can be either a rarefaction or a shock
- Example solutions:



The Riemann problem – analytical solutions

- Given the left/right states, one can then obtain the analytical solution to the Riemann problem in the "star" regions
- This can be done by, e.g., using the Rankine-Hugoniot jump conditions for shocks, and other conserved quantities for contact discontinuities/rarefactions
- Once the star solutions are found, one could obtain the time-averaged fluxes at the cell edge (x=0)
- If the above is done, it is called "***exact Riemann solvers***"
- However, this is often computationally expensive to find the solutions exactly

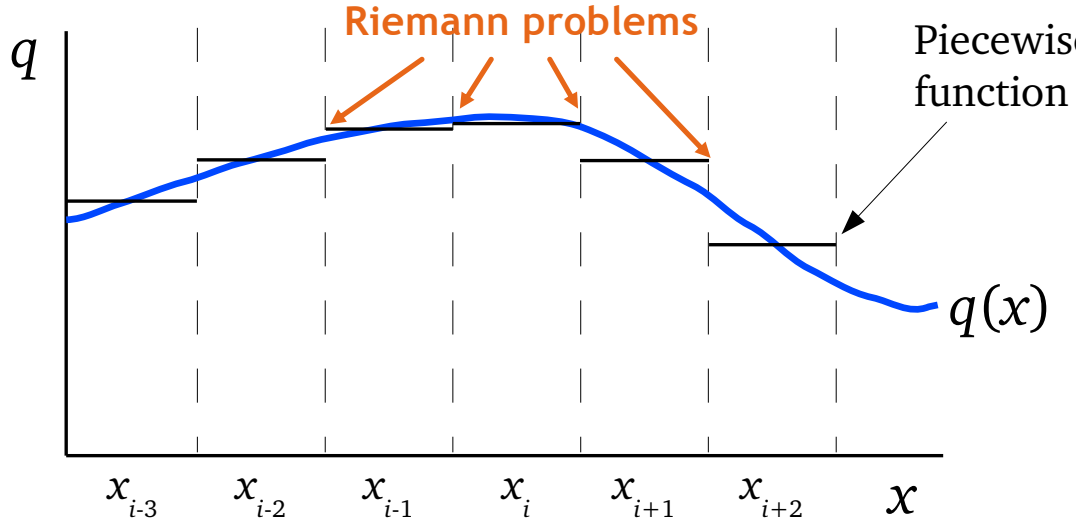


Approximate Riemann solvers

- Exact Riemann solvers are very computationally expensive, so ***approximate Riemann solvers*** are often used
- Approximate Riemann solvers make some simplified assumptions, but are usually accurate enough for computing the fluxes
- Examples of approximate Riemann solvers:
  - ***Roe*** Riemann solver: solve Riemann problem for linearized Euler equations; easy to adapt to new systems of equations; problems in entropy and rarefactions
  - Harten-Lax-van Leer (***HLL***) Riemann solver: treat solution as consisting of two waves separated by three constant states; easy to find direct solutions; poor treatment of contact discontinuities

Now we can use the Godunov method to find time-averaged fluxes and update the solutions

- Recall procedure of the Godunov method:
  - **Step 1**: *approximate* the solution function at t_n as a ***piecewise constant*** function
  - **Step 2**: solve the 1D ***Riemann problem*** at each cell edge => *exact* solution between t_n and t_{n+1} at cell edge
  - **Step 3**: find the time-averaged flux at the cell edges



Finite-volume methods -- comparisons

$$q_i^{n+1} = q_i^n - \frac{\Delta t}{\Delta x}\left[F_{i+1/2}^{n+1/2} - F_{i-1/2}^{n+1/2}\right]$$

Godunov method ($\mathcal{O}(\Delta x, \Delta t)$)

$$F_{i+1/2}^{n+1/2} = F\left[q_{i+1/2}^{n+1/2}\right]$$

$$q_{i+1/2}^{n+1/2} = Riemann\left(q_i^n, q_{i+1}^n\right)(x/t = 0)$$

Lax-Friedrichs ($\mathcal{O}(\Delta x^2, \Delta t)$)

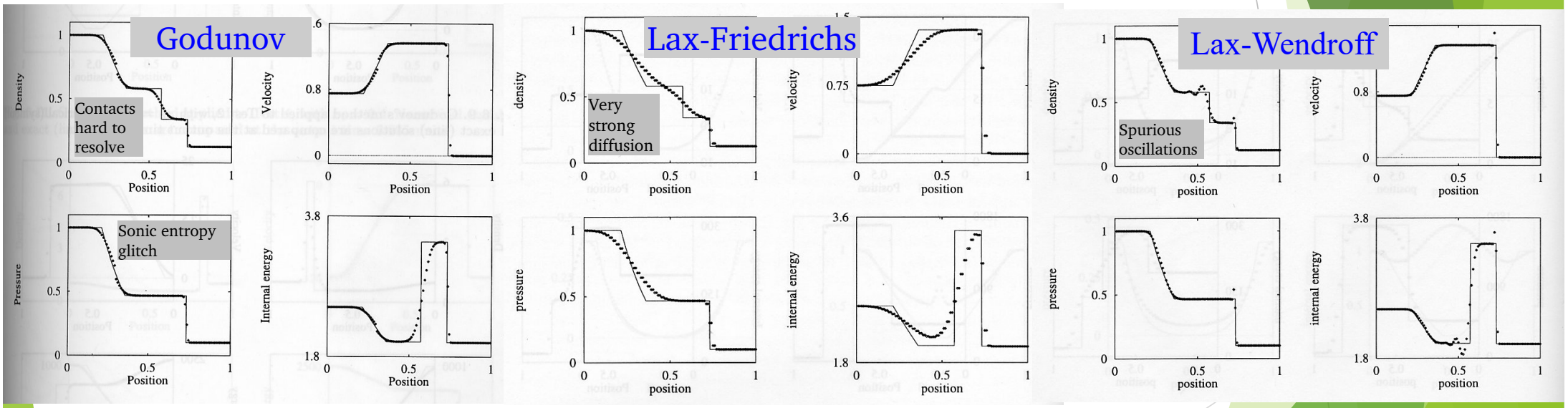
$$F_{i+1/2}^{n+1/2} = \frac{1}{2}\left[F\left(q_i^n\right) + F\left(q_{i+1}^n\right)\right] - \frac{\Delta x}{2\Delta t}\left(q_{i+1}^n - q_i^n\right)$$

Lax-Wendroff ($\mathcal{O}(\Delta x^2, \Delta t^2)$)

$$q_{i+1/2}^{n+1/2} = \frac{1}{2}\left(q_i^n + q_{i+1}^n\right) - \frac{\Delta x}{2\Delta t}\left[F\left(q_{i+1}^n\right) - F\left(q_i^n\right)\right]$$

$$F_{i+1/2}^{n+1/2} = F\left(q_{i+1/2}^{n+1/2}\right)$$

Finite-volume methods – comparisons of the shocktube test



Higher-order method – MUSCL-Hancock scheme

- ***MUSCL*** = Monotone Upwind Scheme for Conservation Laws
- Rather than using piecewise constant L/R states as in Godunov method, use ***extrapolated and evolved*** L/R states to solve the Riemann problem at cell edges
- **Step 1:** ***reconstruction***

$$\boldsymbol{q}_i^L = \boldsymbol{q}_i^n - \frac{1}{2}\boldsymbol{\sigma}_i \Delta x$$

$$\boldsymbol{q}_i^R = \boldsymbol{q}_i^n + \frac{1}{2}\boldsymbol{\sigma}_i \Delta x$$

- **Step 2:** ***evolution***

$$\bar{\boldsymbol{q}}_i^L = \boldsymbol{q}_i^L + \frac{\Delta t}{2\Delta x}[\boldsymbol{F}(\boldsymbol{q}_i^L) - \boldsymbol{F}(\boldsymbol{q}_i^R)]$$

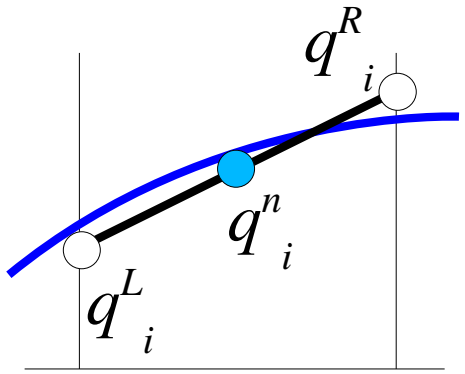
$$\bar{\boldsymbol{q}}_i^R = \boldsymbol{q}_i^R + \frac{\Delta t}{2\Delta x}[\boldsymbol{F}(\boldsymbol{q}_i^L) - \boldsymbol{F}(\boldsymbol{q}_i^R)]$$

- **Step 3:** ***Riemann problem***

$$\boldsymbol{q}_{i+1/2}^{n+1/2} = Riemann(\bar{\boldsymbol{q}}_i^R, \bar{\boldsymbol{q}}_{i+1}^L)(x/t = 0)$$

$$\boldsymbol{F}_{i+1/2}^{n+1/2} = \boldsymbol{F}[\boldsymbol{q}_{i+1/2}^{n+1/2}]$$

- Order: $\mathcal{O}(\Delta x^2, \Delta t^2)$



zone i

Piecewise linear method

Piecewise linear method (PLM) for data reconstruction

Choices for computing the slopes

- **Centered slope:**

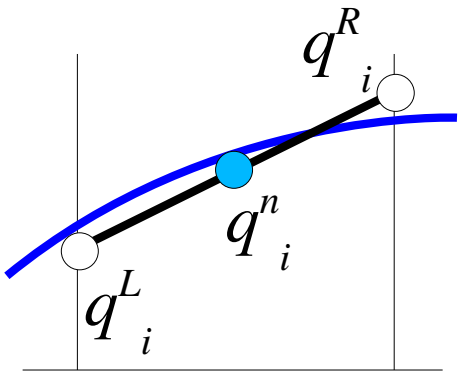
$$\sigma_i^n = \frac{q_{i+1}^n - q_{i-1}^n}{2\Delta x}$$

- **Upwind slope:**

$$\sigma_i^n = \frac{q_i^n - q_{i-1}^n}{\Delta x}$$

- **Downwind slope:**

$$\sigma_i^n = \frac{q_{i+1}^n - q_i^n}{\Delta x}$$



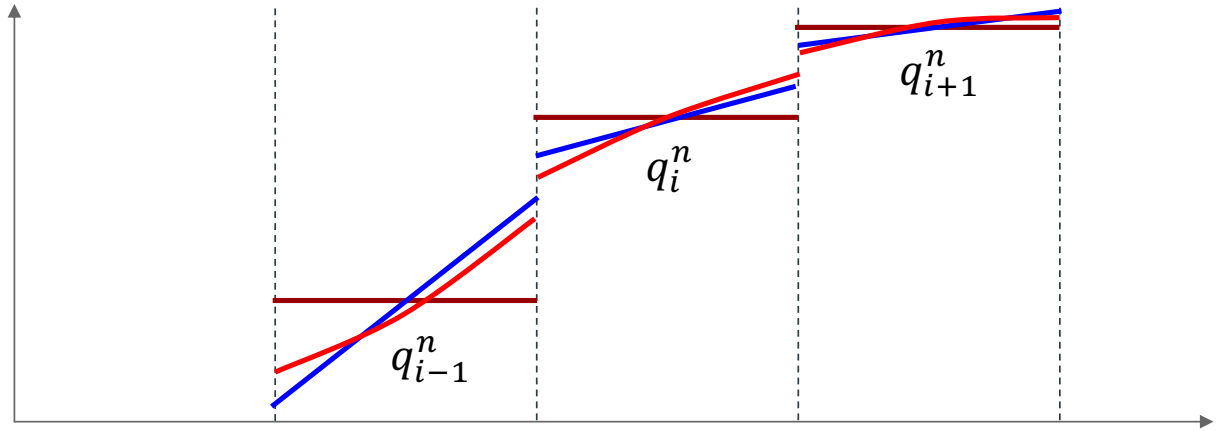
zone i

$$\boldsymbol{q}_i^L = \boldsymbol{q}_i^n - \frac{1}{2}\boldsymbol{\sigma}_i \Delta x$$

$$\boldsymbol{q}_i^R = \boldsymbol{q}_i^n + \frac{1}{2}\boldsymbol{\sigma}_i \Delta x$$

More about data reconstruction

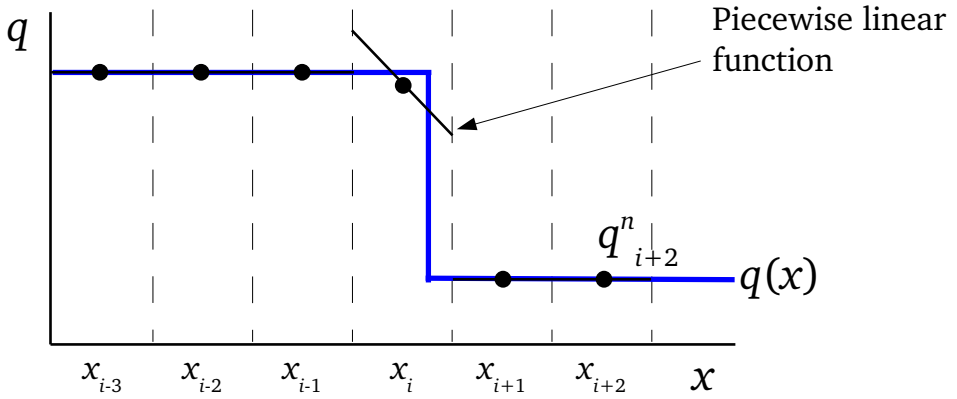
- There are different ways to do the ***data reconstruction*** (step 1 in MUSCL)
  - Godunov scheme – piecewise constant method (**PCM**)
  - Piecewise linear method (**PLM**)
  - Piecewise parabolic method (**PPM;** formally could yield $\mathcal{O}(\Delta x^3)$ accuracy)



Data reconstruction – the need to limit slopes

- During data reconstruction, we require $\frac{1}{\Delta x}\int_{x_{i-1/2}}^{x_{i+1/2}} q(x,t_n)\,dx = q_i^n$

- However, at discontinuities, we may obtain meaningless slopes



- We would introduce unphysical oscillations unless we ***limit the slopes***

Slope limiters – some examples

Godunov method (setting slope to zero) $\sigma_i^n = 0$

Minmod limiter

$$\sigma_i^n = \text{minmod}\left[\frac{q_i^n - q_{i-1}^n}{\Delta x}, \frac{q_{i+1}^n - q_i^n}{\Delta x}\right]$$

where

$$\text{minmod}(a,b) = \begin{cases} a & \text{if } |a| < |b| \text{ and } ab > 0 \\ b & \text{if } |b| < |a| \text{ and } ab > 0 \\ 0 & \text{if } ab \le 0 \end{cases}$$

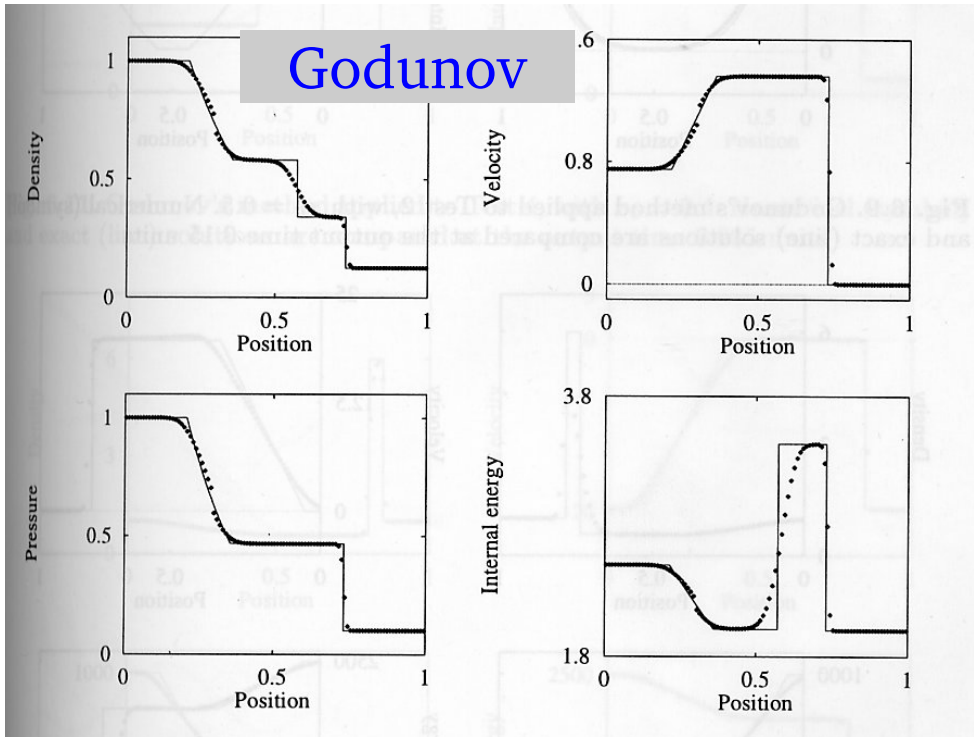
Monotonized central-difference (MC) limiter

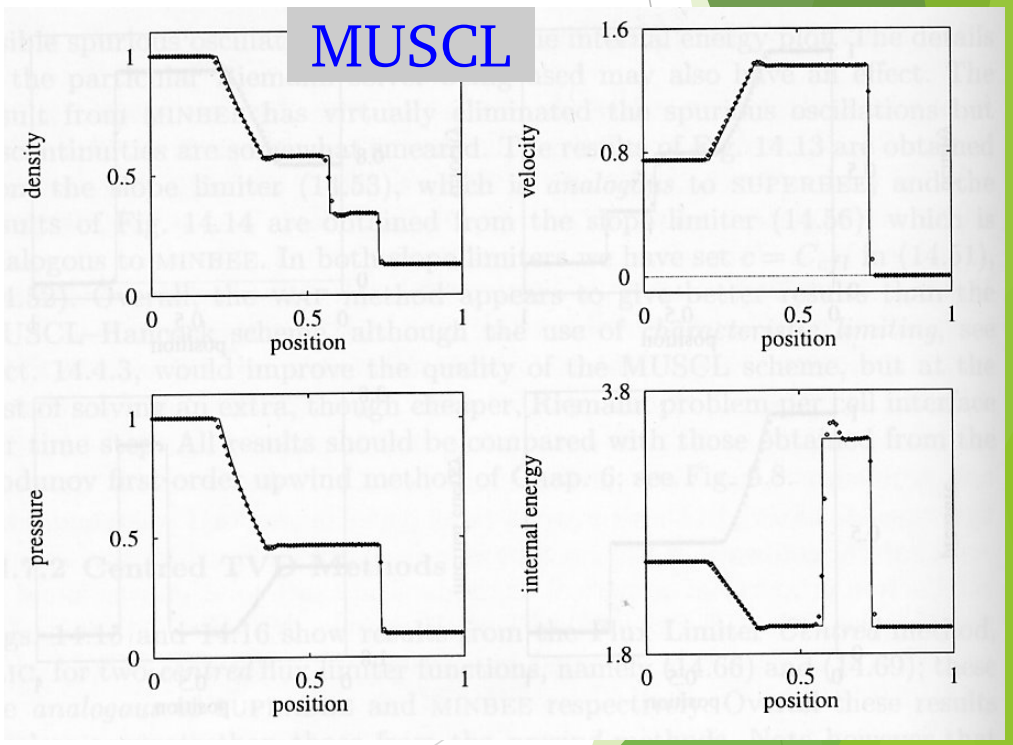
$$\sigma_i^n = \text{minmod}\left[\frac{q_{i+1}^n - q_{i-1}^n}{2\Delta x}, \text{minmod}\left[2\left(\frac{q_i^n - q_{i-1}^n}{\Delta x}\right), 2\left(\frac{q_{i+1}^n - q_i^n}{\Delta x}\right)\right]\right]$$

Total variation diminishing (TVD) methods

- Define the ***total variation***: $TV(q) \equiv \sum_{i=1}^{N} |q_i - q_{i-1}|$
- The more wiggles in q, the larger the total variation
- For simple advection, *TV* is constant in time. Therefore we should expect that our method must not increase *TV* -- it should preserve ***monotonicity*** in the solution
- If $TV(q^{n+1}) \leq TV(q^n)$, the method is ***total variation diminishing (TVD)***

Comparisons between the Godunov & MUSCL schemes





Finite-volume methods for hydrodynamic simulations -- summary

$$\boldsymbol{q}_i^{n+1} = \boldsymbol{q}_i^n - \frac{\Delta t}{\Delta x}\left[\boldsymbol{F}_{i+1/2}^{n+1/2} - \boldsymbol{F}_{i-1/2}^{n+1/2}\right]$$

- One could write the hydro equations in conservative forms and solve them using the finite-volume method. Key is to approximate the ***time-averaged fluxes at cell edges*** accurately
- Lax-Friedrichs/upwind/Lax-Wendroff methods do not work well for discontinuities
- ***Godunov method*** – a high-resolution, shock-capturing method
  - Step 1: approximate solution using ***piecewise constant method (PCM)***
  - Step 2: solve the ***Riemann problem*** at each cell edge
  - Step 3: obtain the time-averaged fluxes; update the solutions

Finite-volume methods for hydrodynamic simulations -- summary

- ***Riemann problem*** -- find the fluid solution between t_n and t_{n+1} given an ***arbitrary 1D discontinuity*** between a left state and a right state
  - Exact Riemann solvers calculate the solutions exactly by solving the eigenvalue problem of hydro equations and finding the characteristics
  - Approximate Riemann solvers are often used to save computational cost
- ***MUSCL-Hancock scheme*** – high-order method
  - Step 1: ***data reconstruction using PLM/PPM***
  - Step 2: evolve the solution for half a timestep
  - Step 3: solve the Riemann problem at cell edges to obtain fluxes
- For data reconstruction using PLM/PPM, ***slope limiters*** and ***TVD methods*** are often used for reducing unphysical oscillations

Summary of finite-volume methods – when to use what

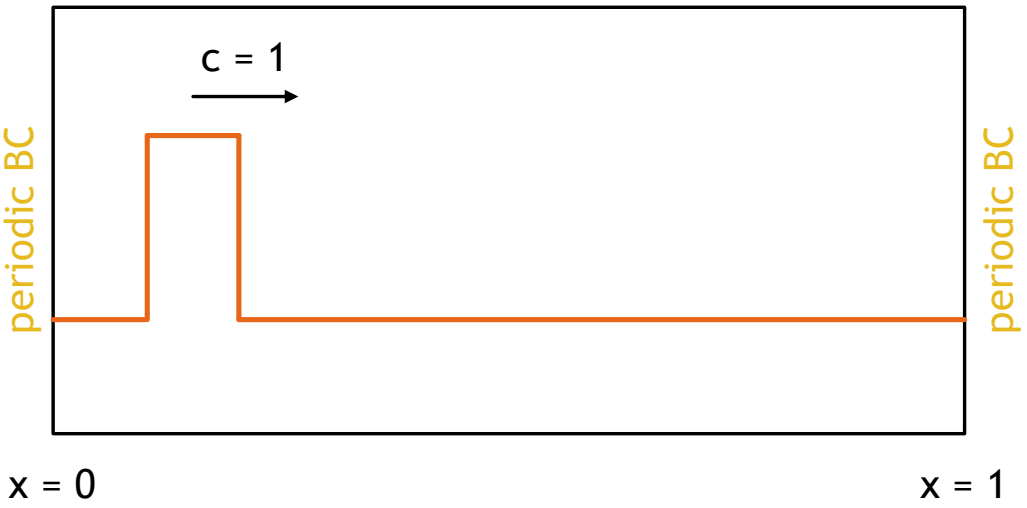
- What not to use
  - 1st-order methods (upwind, Godunov, Lax-Friedrichs): too diffusive
  - 2nd-order non-TVD methods (Lax-Wendroff): too dispersive
- Compressible flows with discontinuities (most astrophysical hydrodynamic simulations)
  - ***2nd-order MUSCL schemes (PLM, PPM)***
- Compressible flows with extra physics (e.g., MHD, relativity)
  - ***2nd-order semidiscrete TVD methods*** (not covered in the lecture)
  - 2nd-order MUSCL schemes (more complex)

In-class exercises

Exercise #1 – solving advection equation using shock-capturing methods (PLM)

$$u_t = -cu_x$$

- Let's propagate the top-hat function using the ***piecewise-linear method (PLM)*** which is more suitable for shock capturing



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

Rewrite the advection equation in the conservative form then

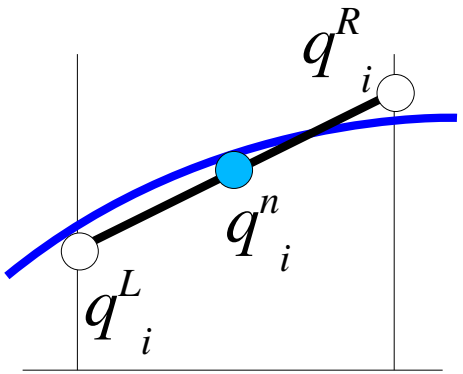
$$\boldsymbol{q = u, F(q,t) = cu}$$

Exercise #1 – solving advection equation using shock-capturing methods (PLM)

- The piecewise-linear method (PLM) approximates the solution function using a linear function of the form:

$$\boldsymbol{q}_i^L = \boldsymbol{q}_i^n - \frac{1}{2}\boldsymbol{\sigma}_i \Delta x$$

$$\boldsymbol{q}_i^R = \boldsymbol{q}_i^n + \frac{1}{2}\boldsymbol{\sigma}_i \Delta x$$



- We could also apply the slope limiters to avoid unphysical oscillations, e.g., the minmod limiter:

$$\sigma_i^n = \text{minmod}\left[\frac{q_i^n - q_{i-1}^n}{\Delta x}, \frac{q_{i+1}^n - q_i^n}{\Delta x}\right]$$

where

$$\text{minmod}(a,b) = \begin{cases} a & \text{if } |a|<|b| \text{ and } ab>0 \\ b & \text{if } |b|<|a| \text{ and } ab>0 \\ 0 & \text{if } ab \le 0 \end{cases}$$

Exercise #1 – solving advection equation using shock-capturing methods (PLM)

- For 1D advection, the flux can be computed analytically (no Riemann solver or MUSCL scheme needed):

$$
\begin{aligned}
F^n_{i-1/2} &= \frac{1}{\Delta t}\int_{t_n}^{t_{n+1}} c q^n(x_{i-1/2}, t)dt \\
&= \frac{1}{\Delta t}\int_{t_n}^{t_{n+1}} c q^n(x_{i-1/2} - c(t-t_n), t_n)dt \\
&= \frac{1}{\Delta t}\int_{t_n}^{t_{n+1}} c\left[q^n_{i-1} + (x_{i-1/2} - c(t-t_n) - x_{i-1})\sigma^n_{i-1}\right]dt \\
&= c\,q^n_{i-1} + \frac{1}{2}c(\Delta x - c\Delta t)\sigma^n_{i-1}
\end{aligned}
$$

Exercise #1 – solving advection equation using shock-capturing methods (PLM)

- Connect to CICA and load the Fortran compiler:

  ```
  module load pgi
  ```

- Go to your exercise directory on CICA:

  ```
  cd astr660/exercise
  ```

- Copy the template Fortran files to the current directory:

  ```
  cp -r /data/hyang/shared/astro660CompAstro/L12exercise ./
  ```

- Enter the fvm directory and see what files are there:

  ```
  cd L12exercise/fvm
  ls
  ```

evolution.f90 - where the flux subroutine is evaluated

Use the PLM for updating the fluxes

A new subroutine for computing the slope σ

Please implement the minmod slope limiter

```
subroutine flux(i,dt,FL, FR)
    use Simulation_data
    implicit none
    integer, intent(in) :: i
    real, intent(in)    :: dt
    real, intent(out)   :: FL, FR

    !! ...

    !! Use piecewise linear and slope limiter

    !! left state
    call get_slope(dx,uold(i-2),uold(i-1),uold(i),sig) ! compute sig(i-1)
    FL = ! TODO

    !! right state
    call get_slope(dx,uold(i-1),uold(i),uold(i+1),sig) ! compute sig(i)
    FR = ! TODO
end subroutine flux

subroutine get_slope(dx,l,m,r,sig)
    implicit none
    real, intent(in)  :: dx
    real, intent(in)  :: l   ! left
    real, intent(in)  :: m   ! middle
    real, intent(in)  :: r   ! right
    real, intent(out) :: sig ! the slope
    real :: a, b

    ! compute a and b as the left/right slopes
    a = (m-l)/dx
    b = (r-m)/dx

    ! TODO: implement the minmod limiter

    return
end subroutine get_slope
```

Exercise #1 – solving advection equation using shock-capturing methods (PLM)

- Take a look at evolution.f90 and make sure you understand how it works
- Modify the flux and get_slope subroutines in evolution.f90 in order to implement the PLM method
- Compile and run the code by

```
make
./advection
```

- Use your favorite plotting routine to plot the results from output files "advection_*.d"
- To get the bonus credit, please your code and the plot to the TAs by end of today (12/01/2022)

$$F^n_{i-1/2} = cq^n_{i-1} + \frac{1}{2}c(\Delta x - c\Delta t)\sigma^n_{i-1}$$

$$F^n_{i+1/2} = cq^n_{i} + \frac{1}{2}c(\Delta x - c\Delta t)\sigma^n_{i}$$

The Ulula code

- *Ulula* (= owl in Latin) is a Python-based, ultra-lightweight 2D hydrodynamic code for teaching and experimentation
- Ulula solves the fluid equations on a grid using the Godunov method
- Easy to modify switches in the hydro solvers (e.g., reconstruction, slope limiters, Riemann solvers, time-stepping algorithms)
- Not designed for optimal performance and scientific production runs
- Sample problems include advection test, Kelvin-Helmholtz instability, Sedov-Taylor explosion, Sod shocktube test
- Website: https://bdiemer.bitbucket.io/ulula/index.html

Exercise #2 – trying out the Sedov-Taylor test using Ulula

- Connect to CICA and activate your Python environment:

  ```
  module load python
  source activate compAstro
  ```

- Install Ulula:

  ```
  pip install ulula
  ```

- Go to the following directory:

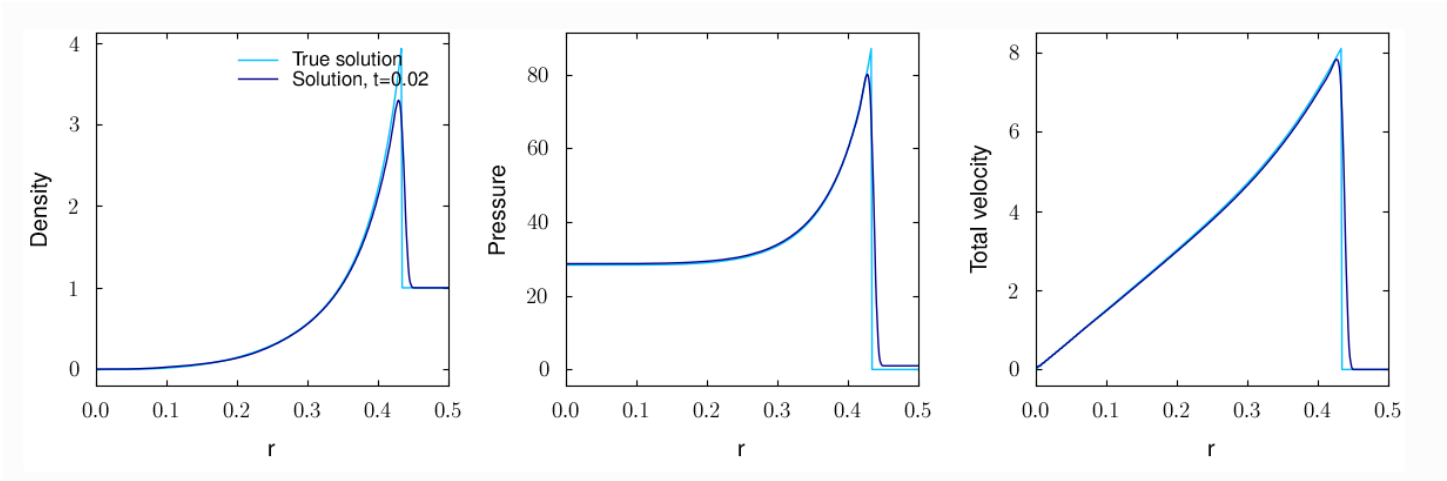
  ```
  cd astr660/exercise/L12exercise/ulula
  ```

Exercise #2 – trying out the Sedov-Taylor test using Ulula

- Take a look at sedov.py and make sure you understand how it works (please read the documentation on the Ulula website)
- Try running the Python script to produce a plot for the Sedov test:

```
python sedov.py
```

- Please read the Ulula documentation and try plotting radial profiles for gas density, pressure, and total velocity



References & acknowledgements

- Course materials of Computational Astrophysics from Prof. Kuo-Chuan Pan (NTHU)
- Course materials of Computational Astrophysics from Prof. Hsi-Yu Schive (NTU)
- Course materials of Computational Astrophysics and Cosmology from Prof. Paul Ricker (UIUC)
- “Computational Physics” by Rubin H. Landau, Manuel Jose Paez and Cristian C. Bordeianu
- “Scientific Computing – An Introductory Survey” by Michael T. Heath