Magnetohydrodynamic (MHD) simulations

Seiji ZENITANI
Space Research Institute (IWF)
Austrian Academy of Sciences

August 2024, Munich

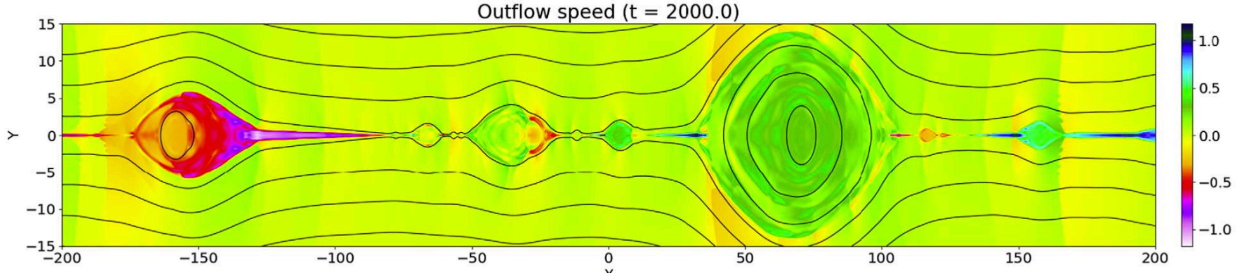
Outline

1. MHD at a glance
2. Basic theory: Advection problem
3. Basic theory: Finite-volume method and Riemann solver
4. MHD simulation with Riemann solver
5. MHD simulation in multi-dimensions
6. Hands on

1. MHD at a glance

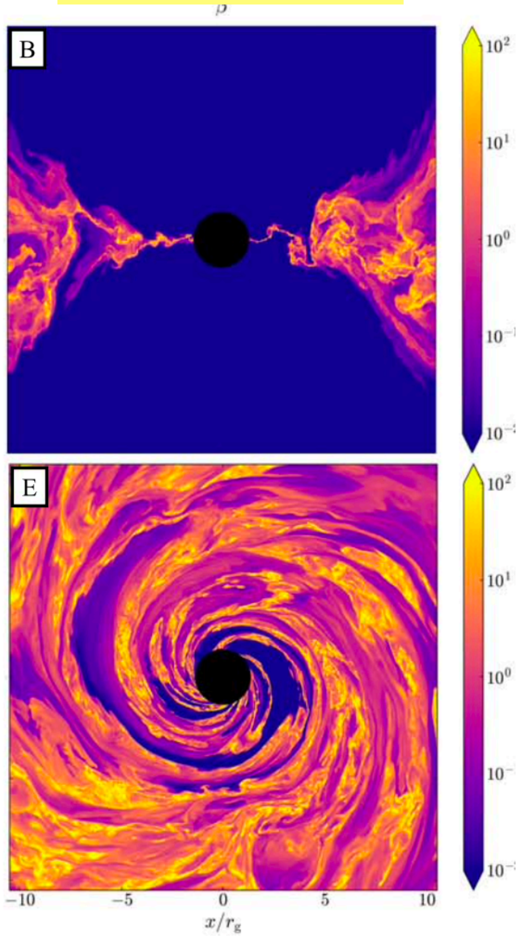
MHD gallery

Basic process (magnetic reconnection)

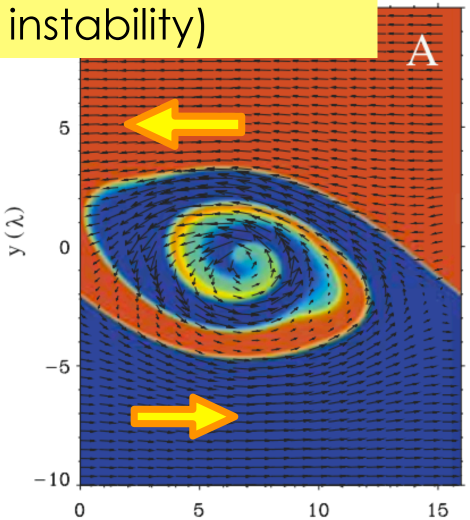


Zenitani+ 2020 ApJ

Black hole accretion disk

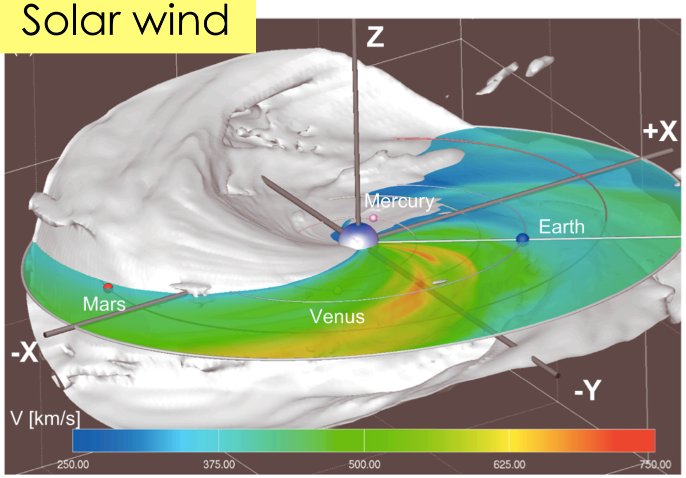


Basic process (Kelvin-Helmholtz instability)



Matsumoto+ 2004 GRL

Solar wind



Shiota+ 2014 Space Weather

Ripperda+ 2022 ApJ

MHD equations = fluid + mag. field + ?

- Continuity $$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \boldsymbol{v}) = 0$$
- Momentum $$\rho \frac{d\boldsymbol{v}}{dt} = -\nabla p + \frac{\boldsymbol{j} \times \boldsymbol{B}}{c}$$ $$= -\nabla p - \nabla\left(\frac{B^2}{8\pi}\right) + \frac{1}{4\pi}(\boldsymbol{B} \cdot \nabla)\boldsymbol{B}$$
- Energy* $$\frac{d}{dt}\left(\frac{p}{\rho^\gamma}\right) = 0$$
- Magnetic field $$\frac{\partial \boldsymbol{B}}{\partial t} = -c(\nabla \times \boldsymbol{E})$$

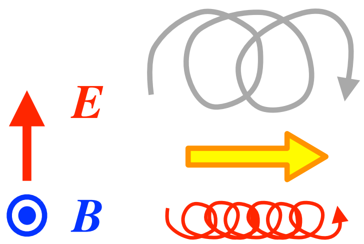
Ampere's law

$$\nabla \times \boldsymbol{B} = \frac{4\pi}{c}\boldsymbol{j} + \cancel{\frac{1}{c}\frac{\partial \boldsymbol{E}}{\partial t}}$$

We have 8 equations, but 11 unknowns...

MHD equations = fluid + mag. field + Ohm's law

- Continuity $$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \boldsymbol{v}) = 0$$
- Momentum $$\rho \frac{d\boldsymbol{v}}{dt} = -\nabla p + \frac{\boldsymbol{j} \times \boldsymbol{B}}{c}$$ $$= -\nabla p - \nabla \left(\frac{B^2}{8\pi} \right) + \frac{1}{4\pi} (\boldsymbol{B} \cdot \nabla) \boldsymbol{B}$$
- Energy* $$\frac{d}{dt} \left(\frac{p}{\rho^\gamma} \right) = 0$$
- Magnetic field $$\frac{\partial \boldsymbol{B}}{\partial t} = -c (\nabla \times \boldsymbol{E})$$
- Ohm's law $$\boldsymbol{E} + \frac{\boldsymbol{v}}{c} \times \boldsymbol{B} = 0$$



$$\boldsymbol{v}_{\mathrm{E \times B}} = \frac{c\boldsymbol{E} \times \boldsymbol{B}}{B^2}$$

MHD equations - conservative form (1/2)

- Continuity

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

- Momentum

$$\frac{\partial \rho \boldsymbol{v}}{\partial t} + \nabla \cdot \left(\rho \boldsymbol{v}\boldsymbol{v} + (p + \frac{B^2}{8\pi})\boldsymbol{I} - \frac{1}{4\pi}\boldsymbol{B}\boldsymbol{B} \right) = 0$$

- Energy

$$\frac{\partial \mathcal{E}}{\partial t} + \nabla \cdot \left((\mathcal{E} + p)\boldsymbol{v} \right) = \boldsymbol{j} \cdot \boldsymbol{E} \qquad \mathcal{E} = \frac{1}{2}\rho v^2 + \frac{1}{\gamma - 1}p$$

$$\frac{\partial}{\partial t}\left(\frac{B^2}{8\pi} + \frac{E^2}{8\pi} \right) + \nabla \cdot \left(\frac{c}{4\pi}\boldsymbol{E} \times \boldsymbol{B} \right) = -\boldsymbol{j} \cdot \boldsymbol{E}$$

$$\frac{\partial}{\partial t}\left(\mathcal{E} + \frac{B^2}{8\pi} \right) + \nabla \cdot \left((\mathcal{E} + p)\boldsymbol{v} + \frac{c}{4\pi}\boldsymbol{E} \times \boldsymbol{B} \right) = 0$$

- Mag. field

$$\frac{\partial \boldsymbol{B}}{\partial t} = -c(\nabla \times \boldsymbol{E})$$

- Ohm's law

$$\boldsymbol{E} + \frac{\boldsymbol{v}}{c} \times \boldsymbol{B} = 0$$

$$\frac{\partial}{\partial t}\boldsymbol{B} - \nabla \times (\boldsymbol{v} \times \boldsymbol{B}) = 0$$

MHD equations - conservative form (2/2)

Conserved quantities — Numerical flux — Source term

$$\frac{\partial}{\partial t}\begin{pmatrix} \rho \\ \rho\boldsymbol{v} \\ \frac{1}{2}\rho v^2 + \frac{1}{\gamma-1}p + \frac{1}{8\pi}B^2 \\ \boldsymbol{B} \end{pmatrix} + \nabla\cdot\begin{pmatrix} \rho\boldsymbol{v} \\ \rho\boldsymbol{v}\boldsymbol{v} + (p + \frac{B^2}{8\pi})\boldsymbol{I} - \frac{1}{4\pi}\boldsymbol{B}\boldsymbol{B} \\ (\frac{1}{2}\rho v^2 + \frac{\gamma}{\gamma-1}p)\boldsymbol{v} + \frac{c}{4\pi}\boldsymbol{E}\times\boldsymbol{B} \\ \boldsymbol{v}\boldsymbol{B} - \boldsymbol{B}\boldsymbol{v} \end{pmatrix} = 0$$

$$\frac{\partial}{\partial t}\boldsymbol{U} + \nabla\cdot\boldsymbol{F} = 0$$

Conserved quantities

$$\boldsymbol{U} \equiv \left(\rho \quad \rho\boldsymbol{v} \quad \frac{1}{2}\rho v^2 + \frac{p}{\gamma-1} + \frac{B^2}{8\pi} \quad \boldsymbol{B}\right)^T$$

⟺

Primitive variables

$$\boldsymbol{V} \equiv (\rho \quad \boldsymbol{v} \quad p \quad \boldsymbol{B})^T$$

MHD waves - magnetosonic waves

- Alfvén wave

$$\left(\frac{\omega}{k}\right)^2 = c_A^2 \cos^2\theta$$

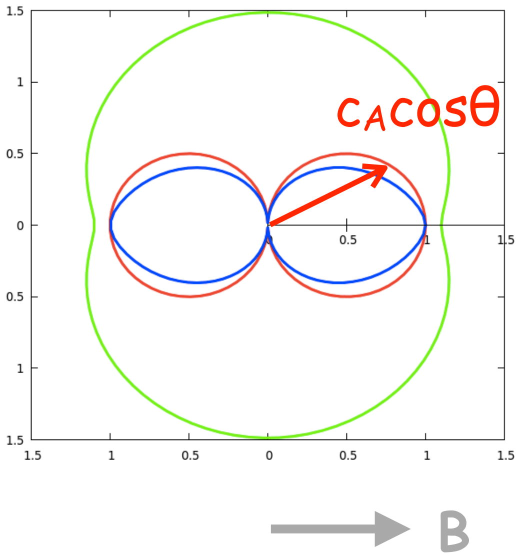
Alfvén speed $c_A^2 = \frac{B^2}{4\pi\rho}$ Sound speed $c_s^2 = \frac{\gamma p}{\rho}$

- Fast and slow magnetosonic waves

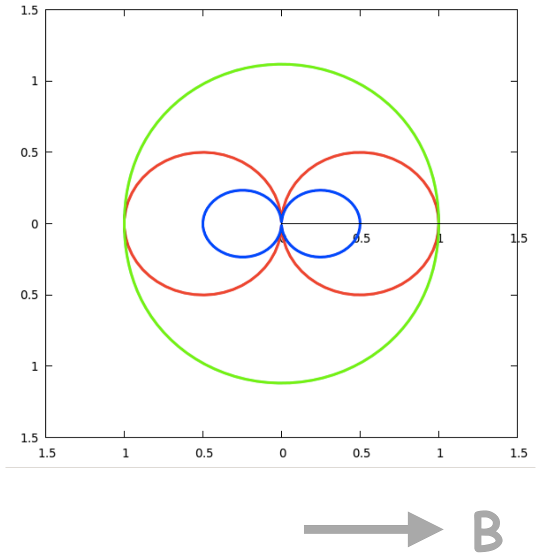
$$\left(\frac{\omega}{k}\right)^2 = \frac{1}{2}\left\{(c_A^2 + c_s^2) \pm \sqrt{(c_A^2 + c_s^2)^2 - 4c_A^2 c_s^2 \cos^2\theta}\right\}$$

- Friedrichs diagram (phase-speed)

  - $c_s > c_A$



  - $c_s < c_A$



2. Advection problem

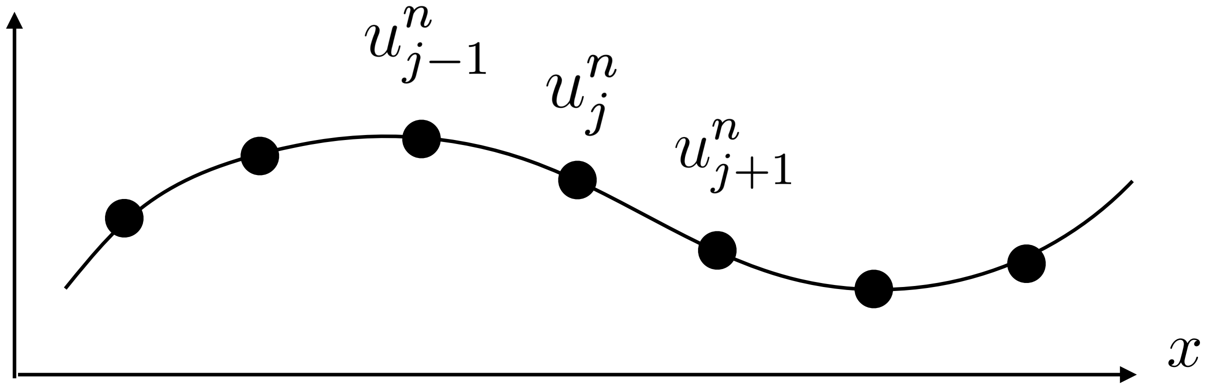
Discretization

- We approximate the real system,

  by using a finite number of discrete grid points

- Grid points in space (x) and time (t) directions

Some quantity: u



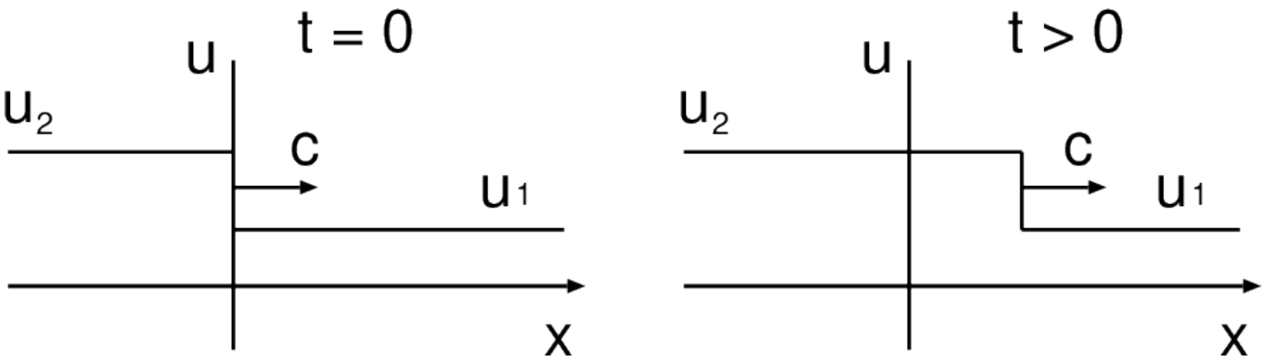
n : time step
j : spatial step

Linear advection equation

- Linear advection equation (Note: c is a positive constant)

$$\frac{\partial u}{\partial t} + c\frac{\partial u}{\partial x} = 0 \qquad\qquad u = f(x - ct)$$

- It allows any profiles traveling at the speed of c



FTCS scheme

(Forward in Time and Centered in Space scheme)

- Equation

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

- Time derivative

$$\frac{\partial u}{\partial t} \approx \frac{u_j^{n+1} - u_j^n}{\Delta t}$$

forward difference

explicit method

- Spatial derivative

$$\frac{\partial u}{\partial x} \approx \frac{u_{j+1}^n - u_{j-1}^n}{2\Delta x}$$

central difference

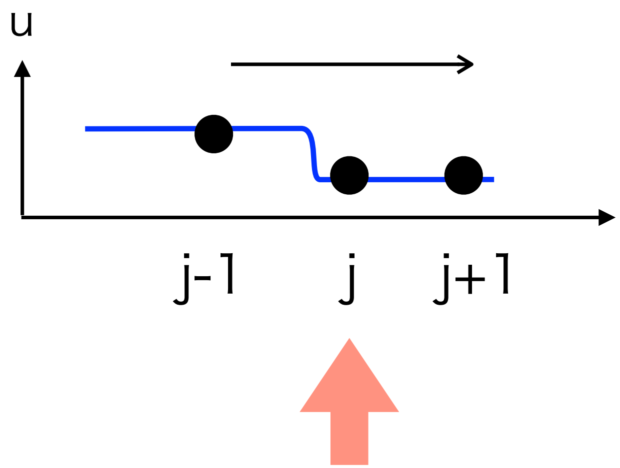
- Discretized equation

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

n : time step

j : spatial step

Upwind scheme (1/2)



- Information comes from the left

  $\rightarrow$ Left information should be used

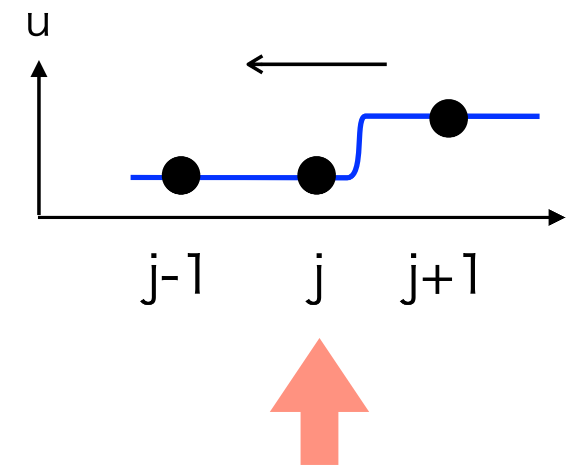
- Equation
$$\frac{\partial u}{\partial t} + c\frac{\partial u}{\partial x} = 0$$

- Time derivative
$$\frac{\partial u}{\partial t} \approx \frac{u_j^{n+1} - u_j^n}{\Delta t}$$

- Spatial derivative
$$\frac{\partial u}{\partial x} \approx \frac{u_j^n - u_{j-1}^n}{\Delta x}$$

- Discretized eq.
$$u_j^{n+1} = u_j^n - c\frac{\Delta t}{\Delta x}\left(u_j^n - u_{j-1}^n\right)$$

Upwind scheme (2/2)



- Information from the right (c<0)

  → Right information should be used

- Spatial derivative

$$\frac{\partial u}{\partial x} \approx \frac{u_{j+1}^n - u_j^n}{\Delta x}$$

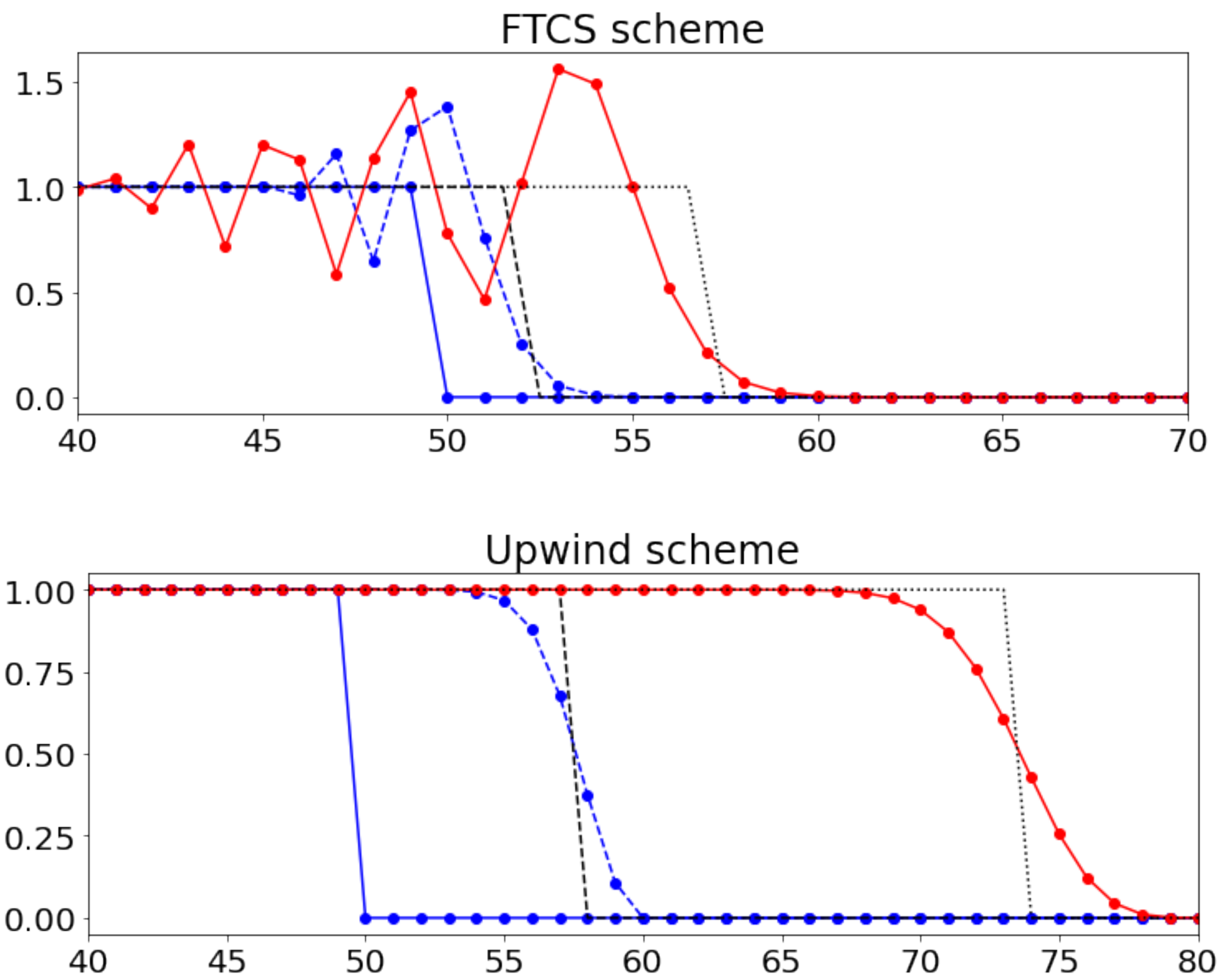
- Discretized eq.

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

- Combing left and right cases:

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

Linear advection problem



von Neumann analysis

- Fourier component

wavenumber

$$u_j^n = g^n \exp\left(i(j\theta)\right) = g^n e^{ij\theta}$$

amplification factor

$\exp(ix) = \cos x + i \sin x$

Courant number

$$\nu = c\frac{\Delta t}{\Delta x}$$

- FTCS scheme

$$g^{n+1}e^{ij\theta} = g^n\left\{e^{ij\theta} - \frac{\nu}{2}(e^{i(j+1)\theta} - e^{i(j-1)\theta})\right\}$$

$$g = g^{n+1}/g^n = 1 - \frac{\nu}{2}(e^{i\theta} - e^{-i\theta})$$

$$= 1 - i\nu\sin\theta$$

$$|g|^2 = 1 + \nu^2\sin^2\theta$$

The code always amplify waves!!

Unconditionally unstable

- Upwind scheme

$$g^{n+1}e^{ij\theta} = g^n\left\{e^{ij\theta} - \nu(e^{ij\theta} - e^{i(j-1)\theta})\right\}$$

$$g = g^{n+1}/g^n = 1 - \nu(1 - e^{-i\theta})$$

$$= (1-\nu) + \nu\cos\theta + i\nu\sin\theta$$

$$|g|^2 = 1 + 2(1-\nu)\nu(1-\cos\theta) \le 1$$

when $0 < \nu \le 1$ Stable

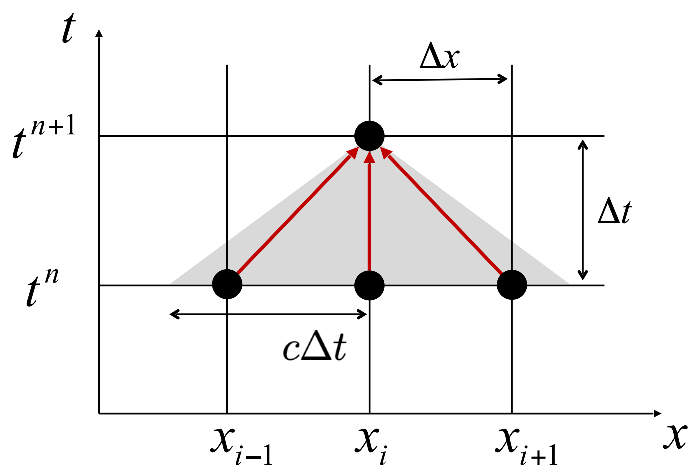
Counrant-Friedrich-Lewy (CFL) condition

- Courant number

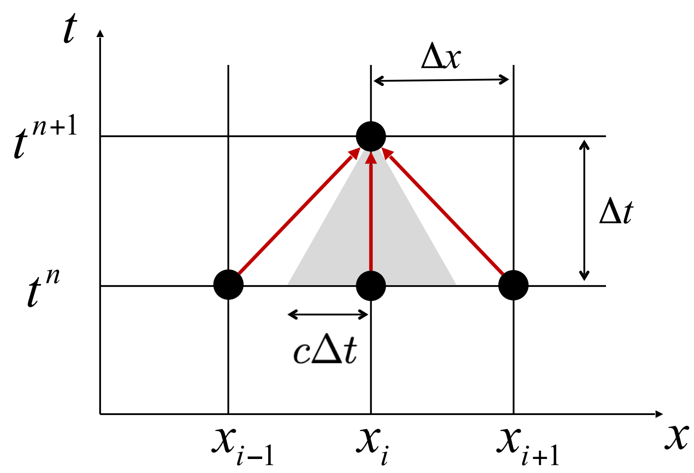
$$\nu = c\frac{\Delta t}{\Delta x} \qquad\qquad \nu \le 1$$

- A "must" condition for explicit schemes

$$\nu > 1 \implies c\Delta t > \Delta x \qquad\qquad \nu \le 1 \implies c\Delta t \le \Delta x$$



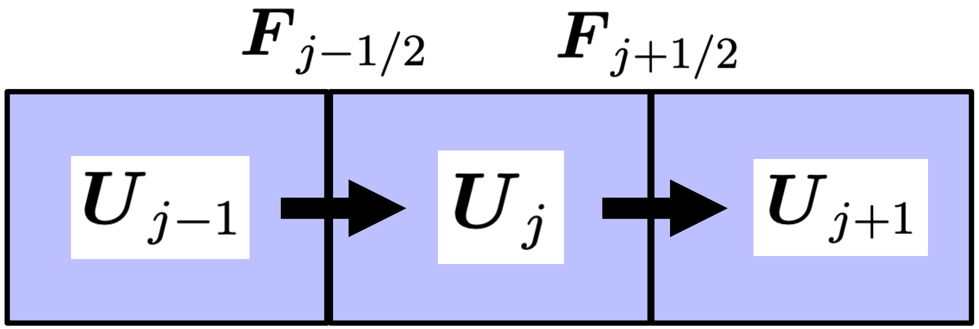
Unstable; breaks causality



Stable

3. Finite-volume method and Riemann solver

Finite volume method

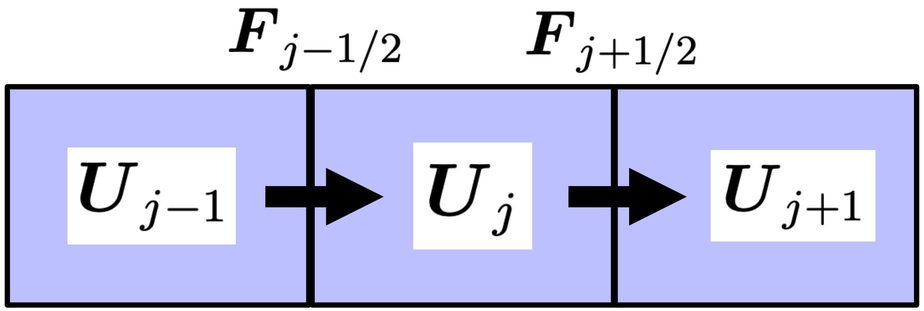


$$\boldsymbol{U}_j(t+\Delta t) = \boldsymbol{U}_j(t) - \frac{\Delta t}{\Delta x}(\boldsymbol{F}_{j+1/2} - \boldsymbol{F}_{j-1/2})$$

$$\frac{\partial}{\partial t}\boldsymbol{U} + \nabla \cdot \boldsymbol{F} = 0$$

- Physical quantities (U) are defined at the cell center
- Numerical fluxes (F) are defined between the cells

Numerical flux in linear advection problem



$$\frac{\partial u}{\partial t} + c\frac{\partial u}{\partial x} = 0 \qquad F_j = cu_j$$

- FTCS scheme

$$F^n_{j+1/2} = \frac{1}{2}\left(cu^n_{j+1} + cu^n_j\right) = \frac{1}{2}\left(F^n_{j+1} + F^n_j\right)$$

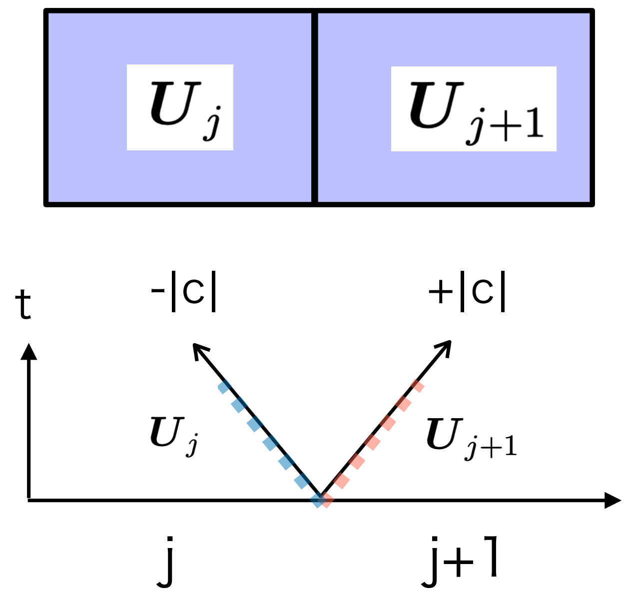
- Upwind scheme

$$F^n_{j+1/2} = \frac{1}{2}\left(F^n_{j+1} + F^n_j\right) - \frac{|c|}{2}\left(u^n_{j+1} - u^n_j\right)$$

$$F^n_{j+1/2} = \begin{cases} F^n_j & \text{(for c} \geq 0) \\ F^n_{j+1} & \text{(for c} < 0) \end{cases}$$

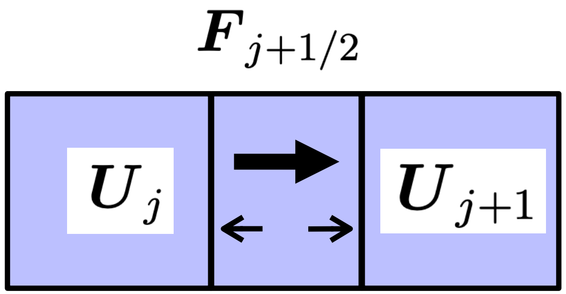
Upwind method - intermediate state

- Let's consider a pair of boundaries, which expand at the speed of ±|c|



Upwind method - intermediate state

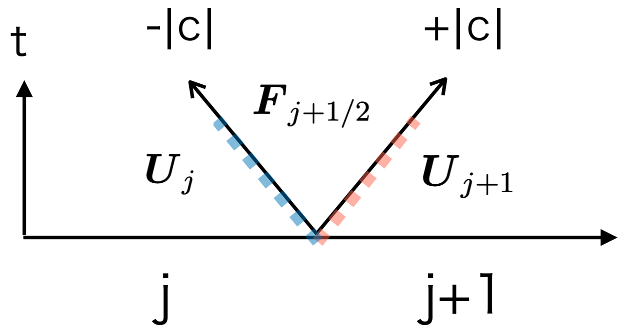
- Let's consider a pair of boundaries, which expand at the speed of ±|c|



- Conservation of physical quantities

$$F_{left} - \lambda U_{left} = F_{right} - \lambda U_{right}$$

λ: speed of the boundary



- Across the two boundaries

$$F_{j+1} - |c|U_{j+1} = F_{j+1/2} - |c|U_{j+1/2}$$

$$F_j + |c|U_j = F_{j+1/2} + |c|U_{j+1/2}$$

$$F_{j+1/2} = \frac{1}{2}\left(F_{j+1} + F_j\right) - \frac{|c|}{2}\left(U_{j+1} - U_j\right)$$

Reminder: MHD equations - conservative form

Conserved quantities | Numerical flux | Source term

$$\frac{\partial}{\partial t}\begin{pmatrix} \rho \\ \rho \boldsymbol{v} \\ \frac{1}{2}\rho v^2 + \frac{1}{\gamma-1}p + \frac{1}{8\pi}B^2 \\ \boldsymbol{B} \end{pmatrix} + \nabla \cdot \begin{pmatrix} \rho \boldsymbol{v} \\ \rho \boldsymbol{v}\boldsymbol{v} + (p + \frac{B^2}{8\pi})\boldsymbol{I} - \frac{1}{4\pi}\boldsymbol{B}\boldsymbol{B} \\ (\frac{1}{2}\rho v^2 + \frac{\gamma}{\gamma-1}p)\boldsymbol{v} + \frac{c}{4\pi}\boldsymbol{E}\times\boldsymbol{B} \\ \boldsymbol{v}\boldsymbol{B} - \boldsymbol{B}\boldsymbol{v} \end{pmatrix} = 0$$

$$\frac{\partial}{\partial t}\boldsymbol{U} + \nabla \cdot \boldsymbol{F} = 0$$

Conserved quantities

$$\boldsymbol{U} \equiv (\rho \quad \rho\boldsymbol{v} \quad \frac{1}{2}\rho v^2 + \frac{p}{\gamma - 1} + \frac{B^2}{8\pi} \quad \boldsymbol{B})^T$$

⟷

Primitive variables

$$\boldsymbol{V} \equiv (\rho \quad \boldsymbol{v} \quad p \quad \boldsymbol{B})^T$$

MHD equations in the X direction (1/3)

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

$$\frac{\partial}{\partial t}\boldsymbol{U} + \nabla \cdot \boldsymbol{F} = 0 \qquad B_x = \text{const.}$$

$$\frac{\partial}{\partial t}\begin{pmatrix} \rho \\ \rho v_x \\ \rho v_y \\ \rho v_z \\ B_y \\ B_z \\ \mathcal{E} \end{pmatrix} + \frac{\partial}{\partial x}\begin{pmatrix} \rho v_x \\ \rho v_x v_x + p \\ \rho v_x v_y \\ \rho v_x v_z \\ B_y v_x - v_y B_x \\ B_z v_x - v_z B_x \\ (\mathcal{E} + p_T)\, v_x - B_x(v_x B_x + v_y B_y + v_z B_z) \end{pmatrix} = 0$$

$$\mathcal{E} = \frac{1}{2}\rho v^2 + \frac{1}{\gamma - 1}p + \frac{B^2}{2}, \qquad p_T = p + \frac{B^2}{2}$$

MHD equations in the X direction (2/3)

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

Jacobean matrix

$$\frac{\partial \boldsymbol{U}}{\partial t} + \boldsymbol{A}\frac{\partial \boldsymbol{U}}{\partial x} = 0, \quad \boldsymbol{A} = \frac{\partial \boldsymbol{F}}{\partial \boldsymbol{U}}$$

$\boldsymbol{\Lambda}$ Diagonal matrix

$\boldsymbol{R}$ Right eigenvectors

$\boldsymbol{R}^{-1}$ Left eigenvectors

$$\boldsymbol{R}^{-1}\frac{\partial \boldsymbol{U}}{\partial t} + \boldsymbol{R}^{-1}\boldsymbol{A}\boldsymbol{R}\boldsymbol{R}^{-1}\frac{\partial \boldsymbol{U}}{\partial x} = 0$$

$$\frac{\partial \boldsymbol{W}}{\partial t} + \boldsymbol{\Lambda}\frac{\partial \boldsymbol{W}}{\partial x} = 0, \quad d\boldsymbol{W} = \boldsymbol{R}^{-1}d\boldsymbol{U}$$

Characteristic variables
(Properties transported by waves)

$$\boldsymbol{R}^{-1}\boldsymbol{A}\boldsymbol{R} = \boldsymbol{\Lambda} = \mathrm{diag}\left(v_x - c_f, v_x - c_a, \ldots, v_x + c_a, v_x + c_f\right)$$

- See Stone et al. 2008 ApJS for further detail

MHD equations in the X direction (3/3)

$$\frac{\partial \boldsymbol{W}}{\partial t} + \boldsymbol{\Lambda} \frac{\partial \boldsymbol{W}}{\partial x} = 0, \; d\boldsymbol{W} = \boldsymbol{R}^{-1} d\boldsymbol{U}$$

$$\frac{\partial}{\partial t}\begin{pmatrix} w_1 \\ w_2 \\ \vdots \\ w_m \end{pmatrix} + \begin{pmatrix} \lambda_1 & 0 & \cdots & 0 \\ 0 & \lambda_2 & & \vdots \\ \vdots & & \ddots & 0 \\ 0 & \cdots & 0 & \lambda_m \end{pmatrix} \frac{\partial}{\partial x}\begin{pmatrix} w_1 \\ w_2 \\ \vdots \\ w_m \end{pmatrix} = 0$$

Characteristic variables
(Properties transported
by waves)

MHD wavespeeds

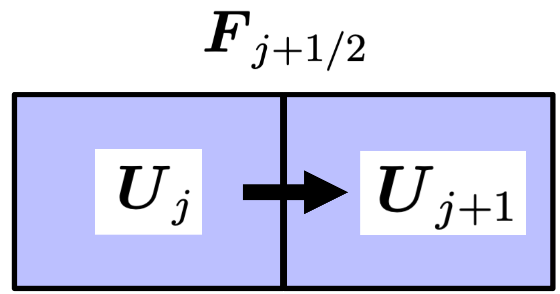
$$\boldsymbol{R}^{-1}\boldsymbol{A}\boldsymbol{R} = \boldsymbol{\Lambda} = \mathrm{diag}\left(v_x - c_f, v_x - c_a, \ldots, v_x + c_a, v_x + c_f\right)$$

fast Alfvén ... Alfvén fast

An advection problem of various MHD waves

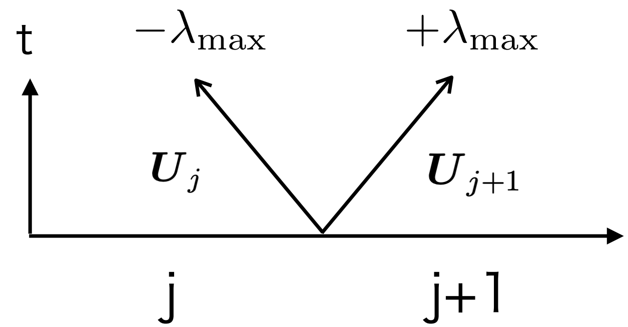
Local Lax-Friedrich (LLF) method

- MHD variant of the upwind scheme
- We employ fastest fast-wave speeds as the signal speeds



$$\lambda_{\max} = \max\Big(|v - c_f|_j, |v - c_f|_{j+1}, |v + c_f|_j, |v + c_f|_{j+1}\Big)$$

c_f : the fast-mode speed

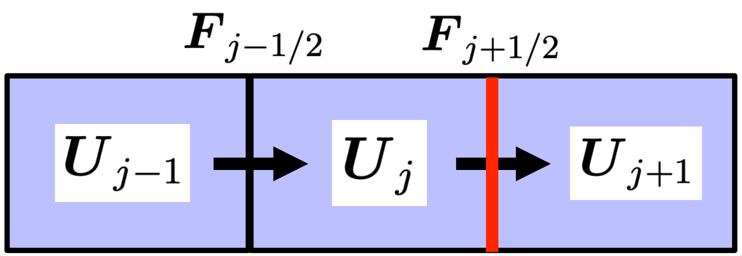


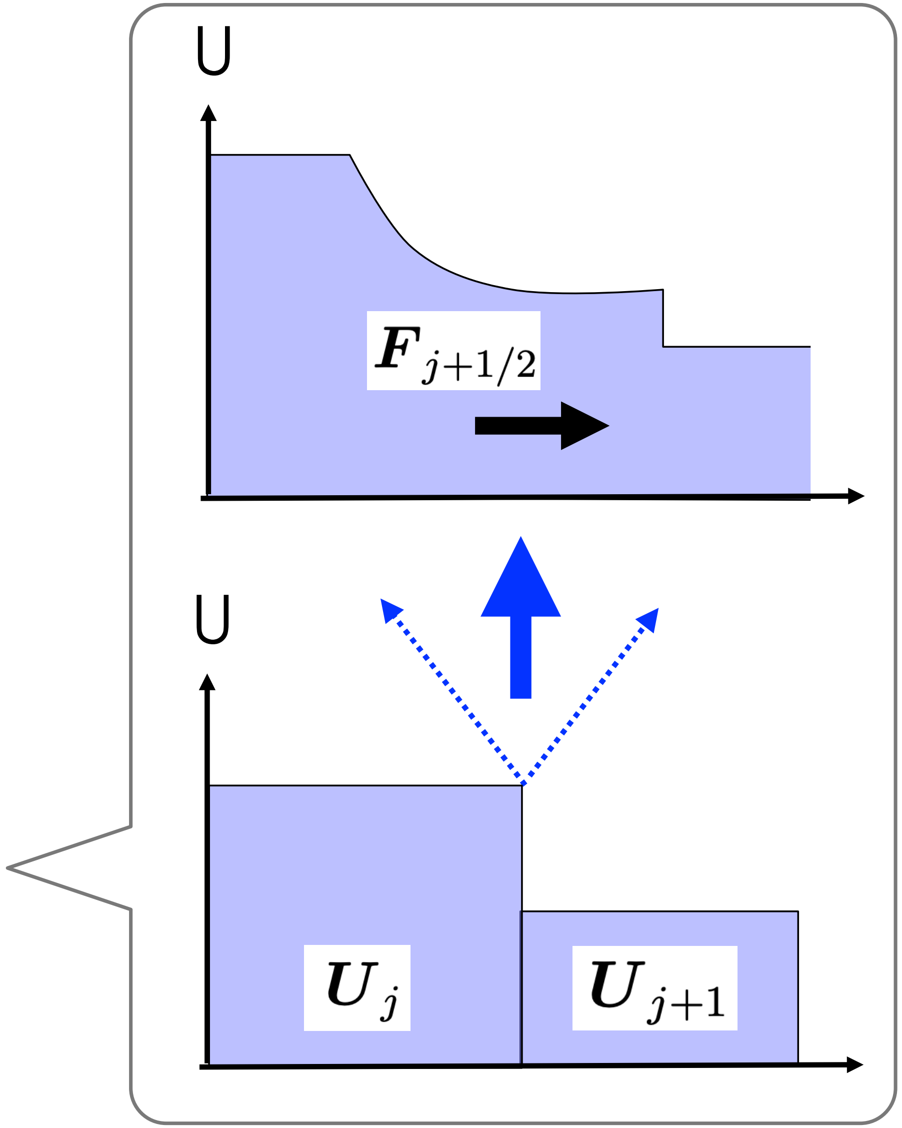
- Numerical flux can be calculated easily

$$\boldsymbol{F}_{j+1/2} = \frac{1}{2}\left(\boldsymbol{F}_{j+1} + \boldsymbol{F}_j\right) - \frac{|\lambda_{\max}|}{2}\left(\boldsymbol{U}_{j+1} - \boldsymbol{U}_j\right)$$

Riemann solver (1/2)

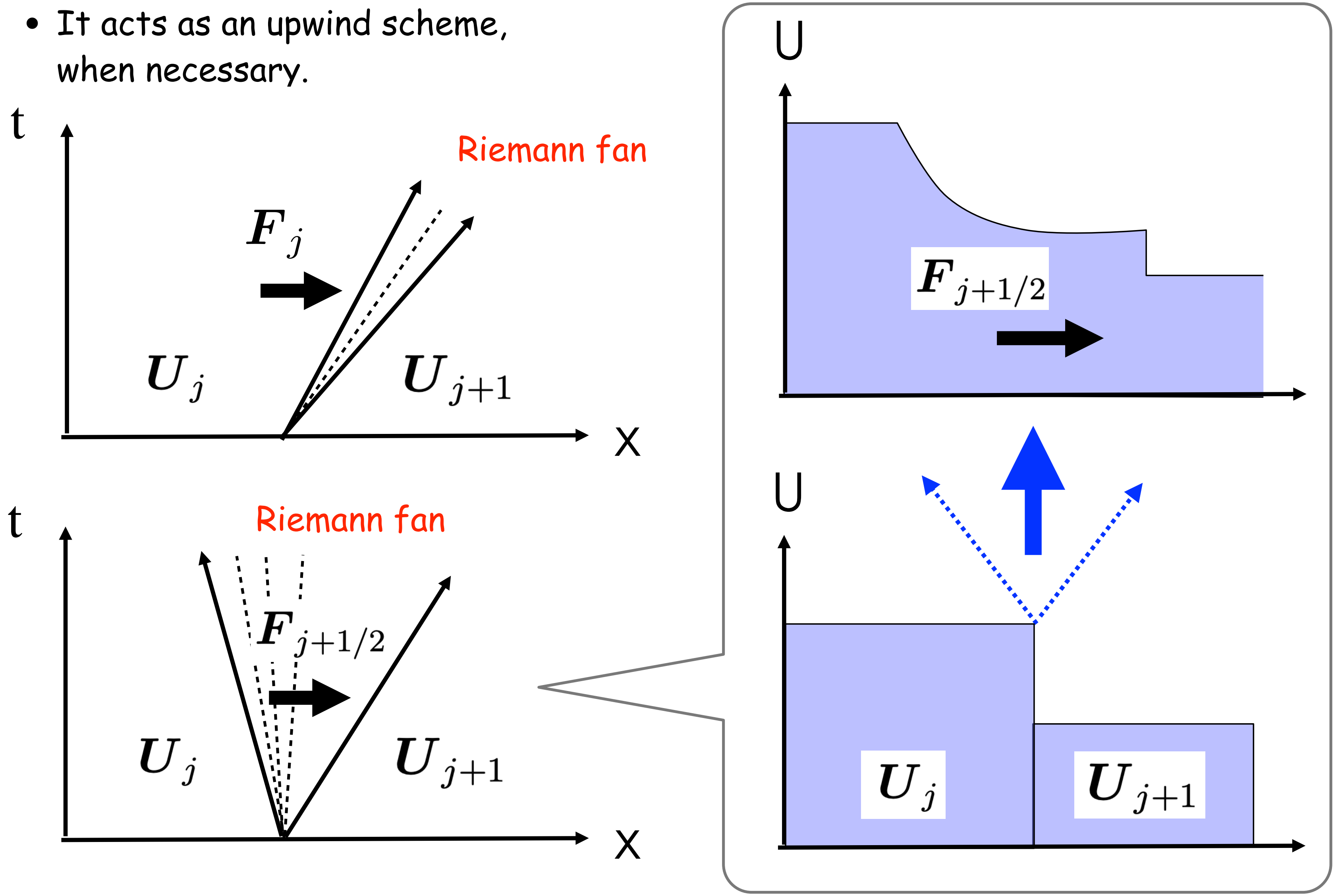
- Riemann problem
  - Time evolution from two flat states
  - Basic problem in hydrodynamics
- Riemann solver
  - Solve Riemann problem at each cell interface



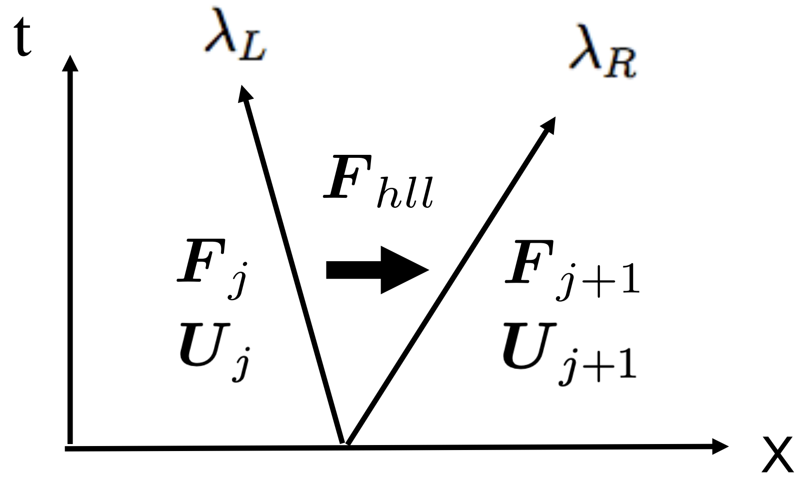


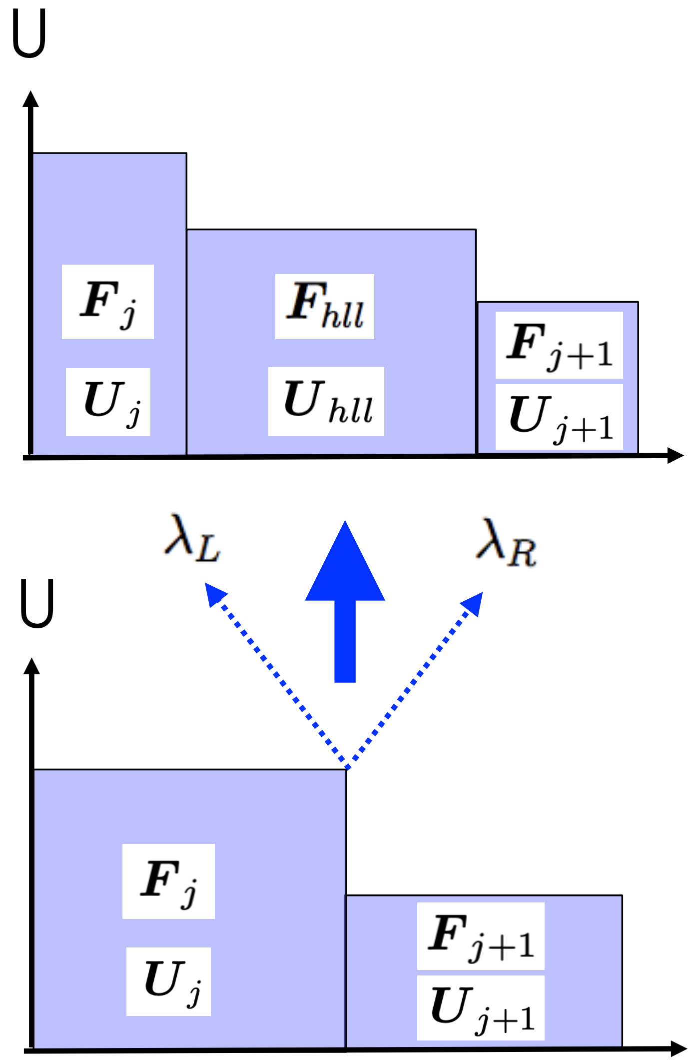
Riemann solver (2/2)

- It acts as an upwind scheme, when necessary.



Approximate Riemann solver - HLL solver (1/2)



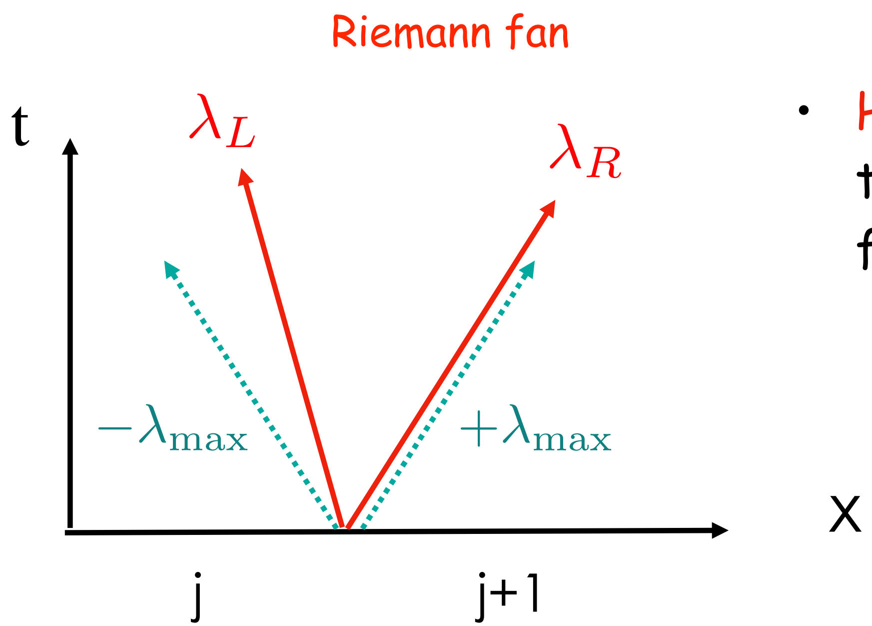


$$\boldsymbol{F} = \begin{cases} \boldsymbol{F}_j & (\lambda_L > 0) \\ \boldsymbol{F}_{hll} & (\lambda_L \leq 0 \leq \lambda_R) \\ \boldsymbol{F}_{j+1} & (\lambda_R < 0) \end{cases}$$

$$\boldsymbol{F}_{hll} = \frac{\lambda_R \boldsymbol{F}_j - \lambda_L \boldsymbol{F}_{j+1} + \lambda_R \lambda_L (\boldsymbol{U}_{j+1} - \boldsymbol{U}_j)}{\lambda_R - \lambda_L}$$

Harten, Lan, & van Leer 1983 *SIAM Rev.*

Approximate Riemann solver - HLL solver (2/2)



- HLL solver (or Riemann solvers) uses the fastest left-going signals and the fastest right-going signals

$$\lambda_L = \min\Big((v - c_f)_j, (v - c_f)_{j+1}\Big)$$

$$\lambda_R = \max\Big((v + c_f)_j, (v + c_f)_{j+1}\Big)$$

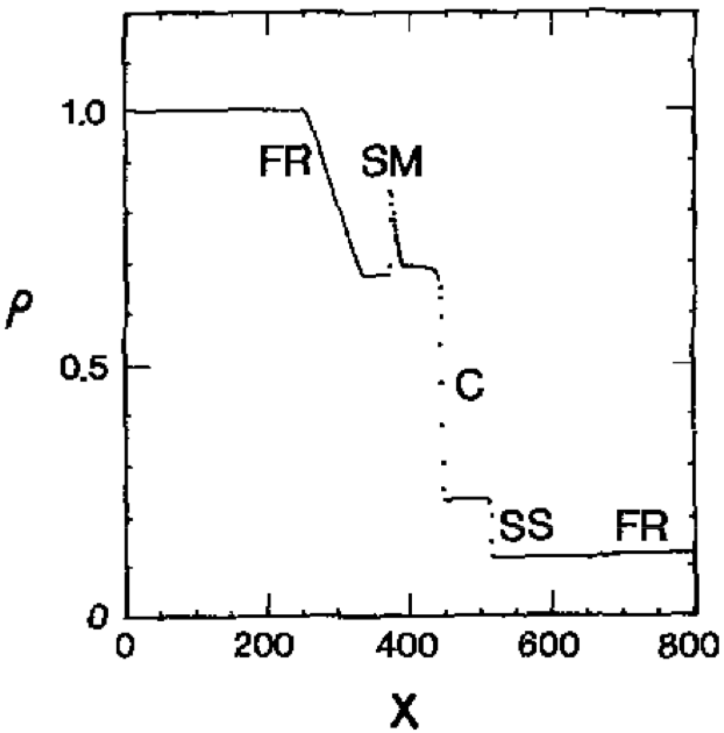
- Local Lax-Friedrich solver is more diffusive than HLL solver, because it spreads physical quantities wider

$$\lambda_{\max} = \max\Big(|v - c_f|_j, |v - c_f|_{j+1}, |v + c_f|_j, |v + c_f|_{j+1}\Big)$$

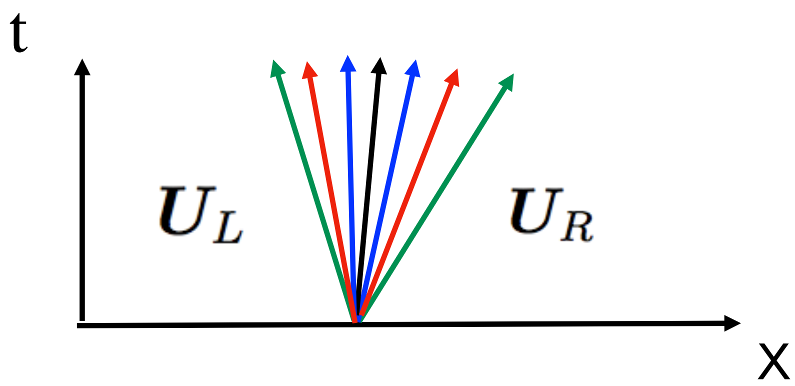
$$\equiv \max(|\lambda_R|, |\lambda_L|)$$

MHD Riemann problem

- There are 6 intermediate states, corresponding to 6 MHD waves
- Ex. Brio=Wu problem



Brio & Wu 1988 JCP

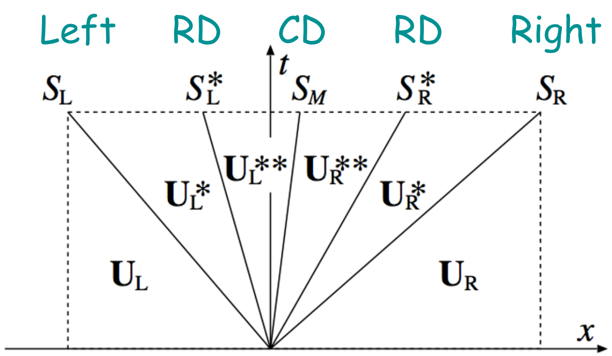


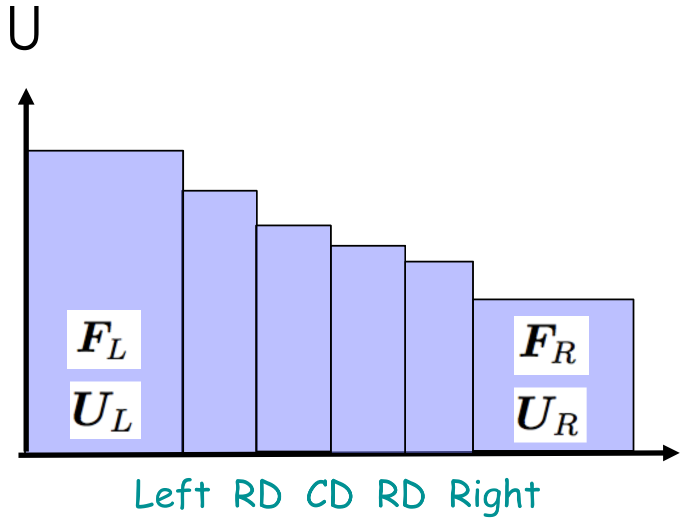
- Alfvén wave
- Fast and slow magnetosonic waves

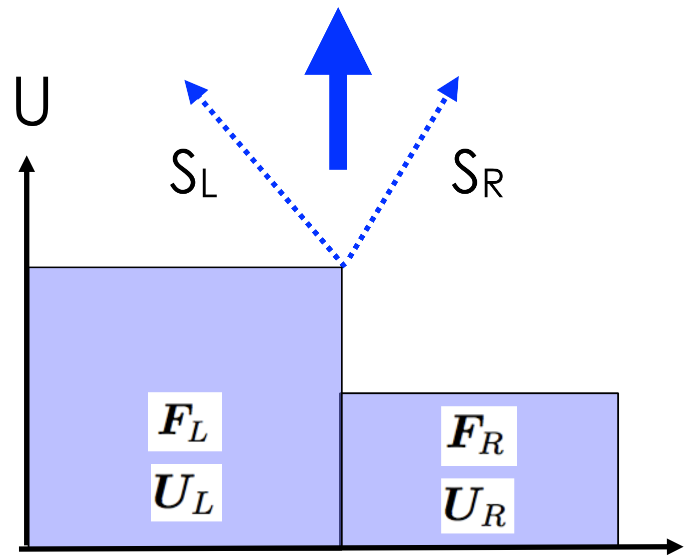
Approximate Riemann solver - HLLD solver (1/3)

- Four intermediate states, separated by MHD discontinuities
- Note: there can be six states in the MHD

$$\mathbf{F}_{\mathrm{HLLD}} = \begin{cases} \mathbf{F}_{\mathrm{L}} & \text{if } S_{\mathrm{L}} > 0, \\ \mathbf{F}^{*}_{\mathrm{L}} & \text{if } S_{\mathrm{L}} \leqslant 0 \leqslant S^{*}_{\mathrm{L}}, \\ \mathbf{F}^{**}_{\mathrm{L}} & \text{if } S^{*}_{\mathrm{L}} \leqslant 0 \leqslant S_{M}, \\ \mathbf{F}^{**}_{\mathrm{R}} & \text{if } S_{M} \leqslant 0 \leqslant S^{*}_{\mathrm{R}}, \\ \mathbf{F}^{*}_{\mathrm{R}} & \text{if } S^{*}_{\mathrm{R}} \leqslant 0 \leqslant S_{\mathrm{R}}, \\ \mathbf{F}_{\mathrm{R}} & \text{if } S_{\mathrm{R}} < 0. \end{cases}$$



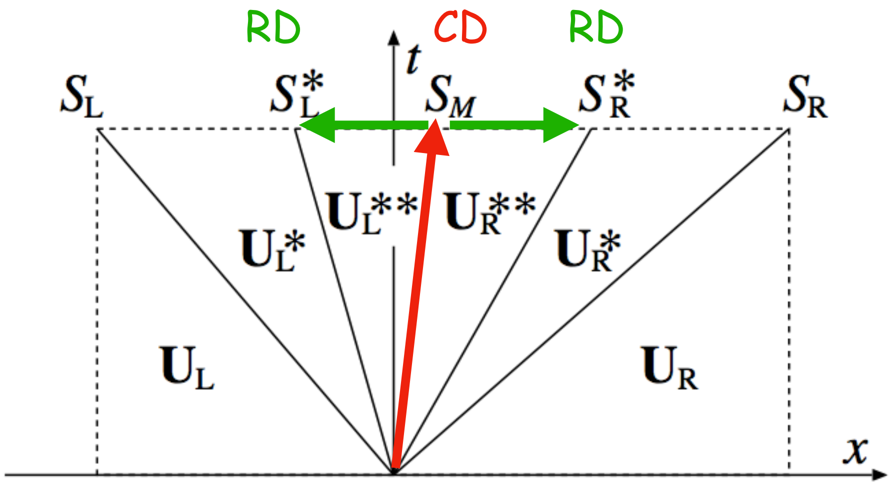




Miyoshi & Kusano 2005 JCP

HLLD solver (2/3) - key points

- 1. We first derive the entropy wave speed (S_M)
- 2. Then we assume two rotational discontinuities (RDs), which propagate outwards at local Alfvén speeds
- 3. We consider the conservation laws across all the boundaries
- 4. We calculate an appropriate numerical flux F_{HLLD}

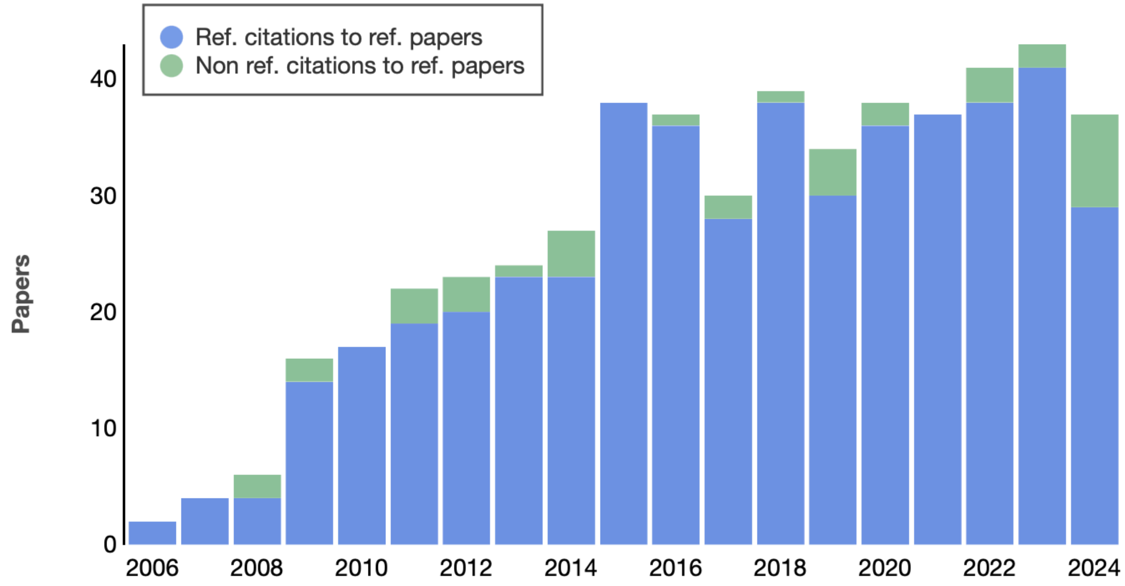


$$\mathbf{F}_{\mathrm{HLLD}} = \begin{cases} \mathbf{F}_{\mathrm{L}} & \text{if } S_{\mathrm{L}} > 0, \\ \mathbf{F}^{*}_{\mathrm{L}} & \text{if } S_{\mathrm{L}} \leqslant 0 \leqslant S^{*}_{\mathrm{L}}, \\ \mathbf{F}^{**}_{\mathrm{L}} & \text{if } S^{*}_{\mathrm{L}} \leqslant 0 \leqslant S_{M}, \\ \mathbf{F}^{**}_{\mathrm{R}} & \text{if } S_{M} \leqslant 0 \leqslant S^{*}_{\mathrm{R}}, \\ \mathbf{F}^{*}_{\mathrm{R}} & \text{if } S^{*}_{\mathrm{R}} \leqslant 0 \leqslant S_{\mathrm{R}}, \\ \mathbf{F}_{\mathrm{R}} & \text{if } S_{\mathrm{R}} < 0. \end{cases}$$

Miyoshi & Kusano 2005 JCP

HLLD solver (3/3) - some more

- HLLD solver is a de-facto standard MHD solver
- If you want to be an MHD expert, I highly recommend you to read the original paper (Miyoshi & Kusano 2005, JCP).
- Fortran/C/Python source files are available.



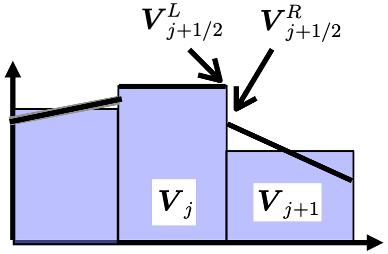
https://ui.adsabs.harvard.edu/abs/2005JCoPh.208..315M

4. MHD simulation with Riemann solver

Simulation cycle

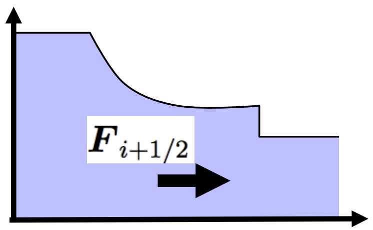
- 1. Spatial interpolation

  Compute left and right states: V_L, V_R



- 2. Calculate numerical flux F,

  considering a Riemann problem



- 3. Update conservative variable U

$$U_i(t+\Delta t) = U_i(t) - \frac{\Delta t}{\Delta x}(F_{i+1/2} - F_{i-1/2})$$

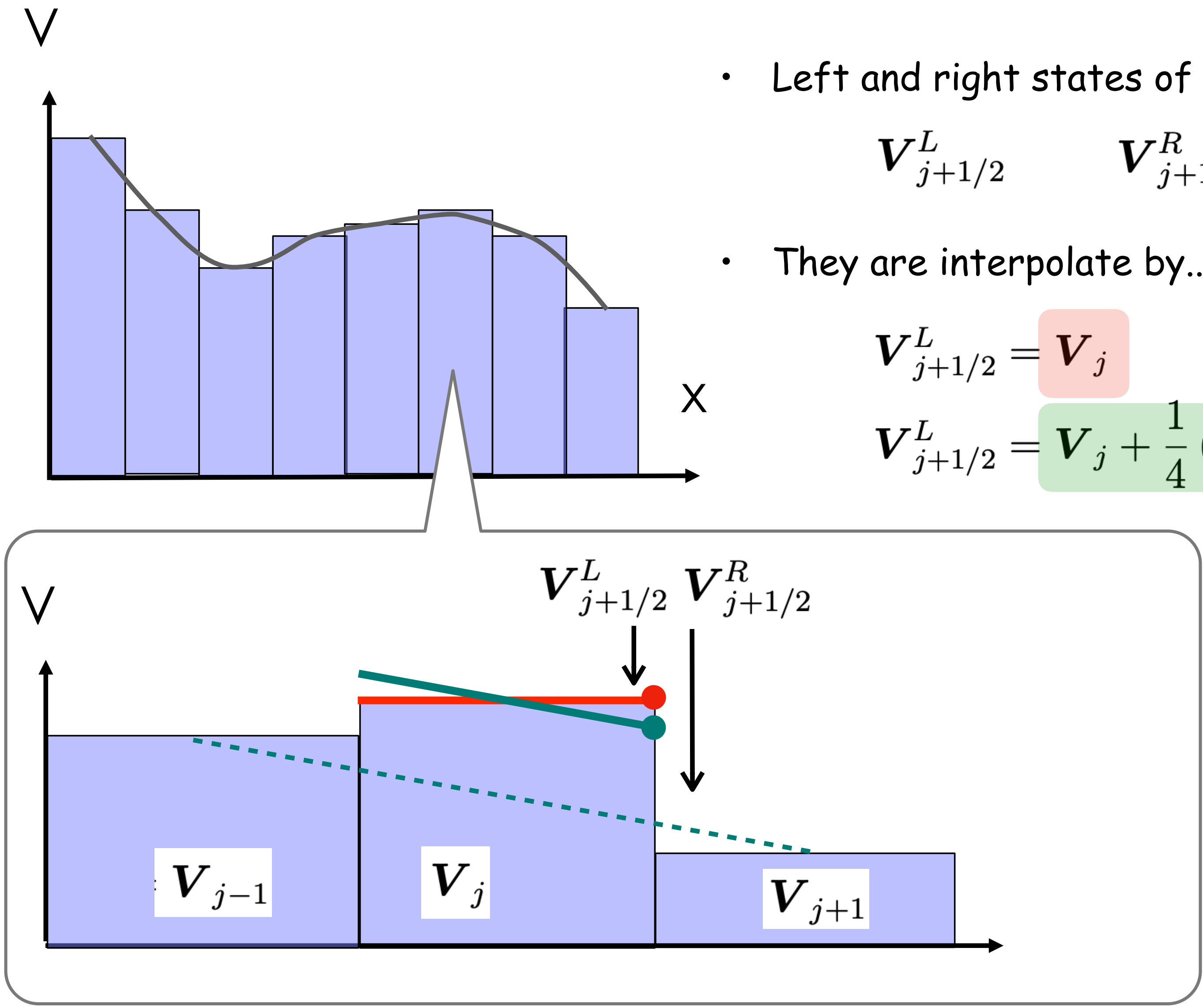
- 4. Recovery of primitive variables

  Convert U to V

$$U \equiv \left(\rho \quad \rho v \quad \frac{1}{2}\rho v^2 + \frac{p}{\gamma - 1} + \frac{B^2}{8\pi} \quad B\right)^T$$

$$\Rightarrow \quad V \equiv (\rho \quad v \quad p \quad B)^T$$

STEP 1/4: Spatial interpolation



- Left and right states of the cell boundary

$$\boldsymbol{V}^L_{j+1/2} \qquad \boldsymbol{V}^R_{j+1/2}$$

- They are interpolate by...

$$\boldsymbol{V}^L_{j+1/2} = \boldsymbol{V}_j$$

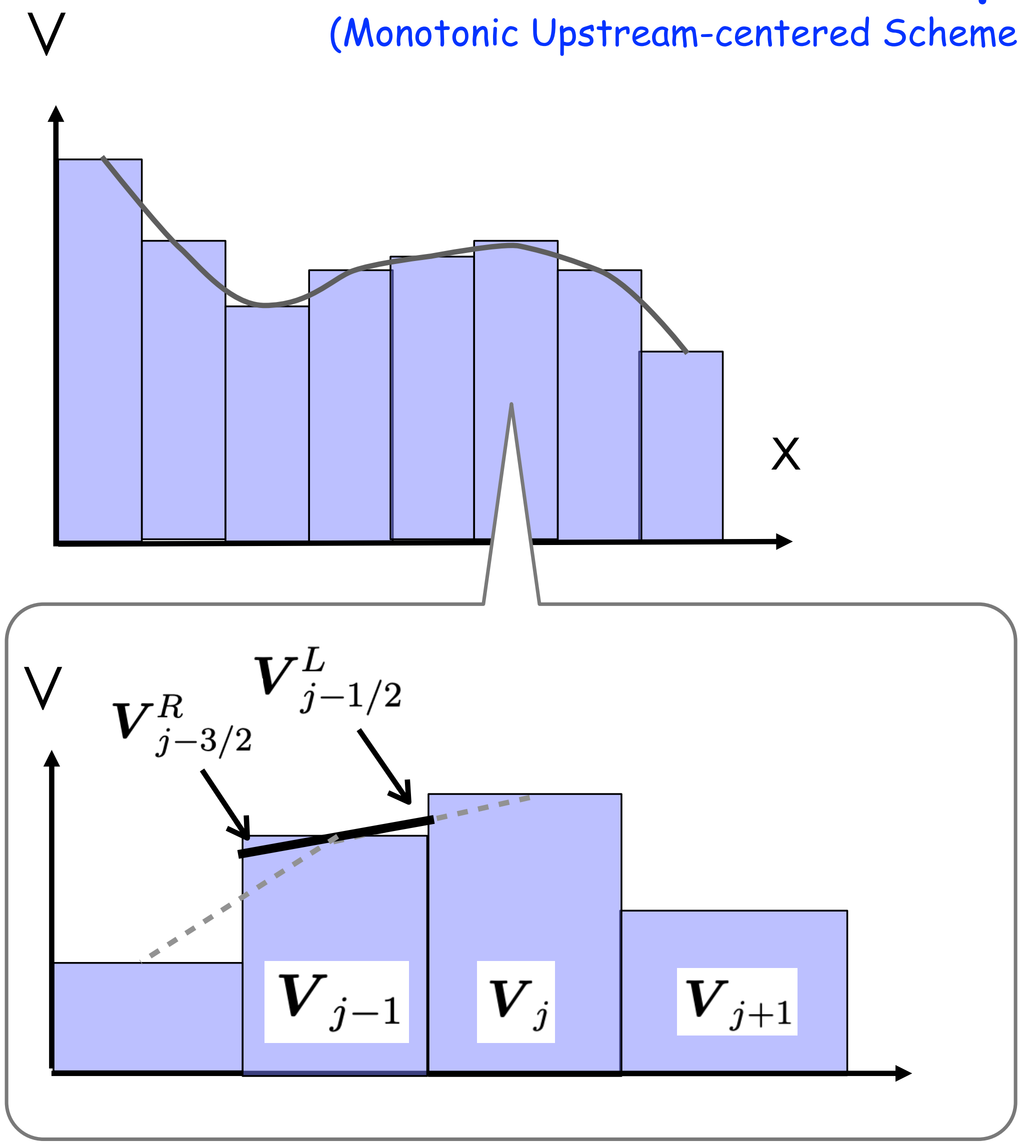
No interpolation

$$\boldsymbol{V}^L_{j+1/2} = \boldsymbol{V}_j + \frac{1}{4}(\boldsymbol{V}_{j+1} - \boldsymbol{V}_{j-1})$$

2nd-order interpolation

MUSCL interpolation

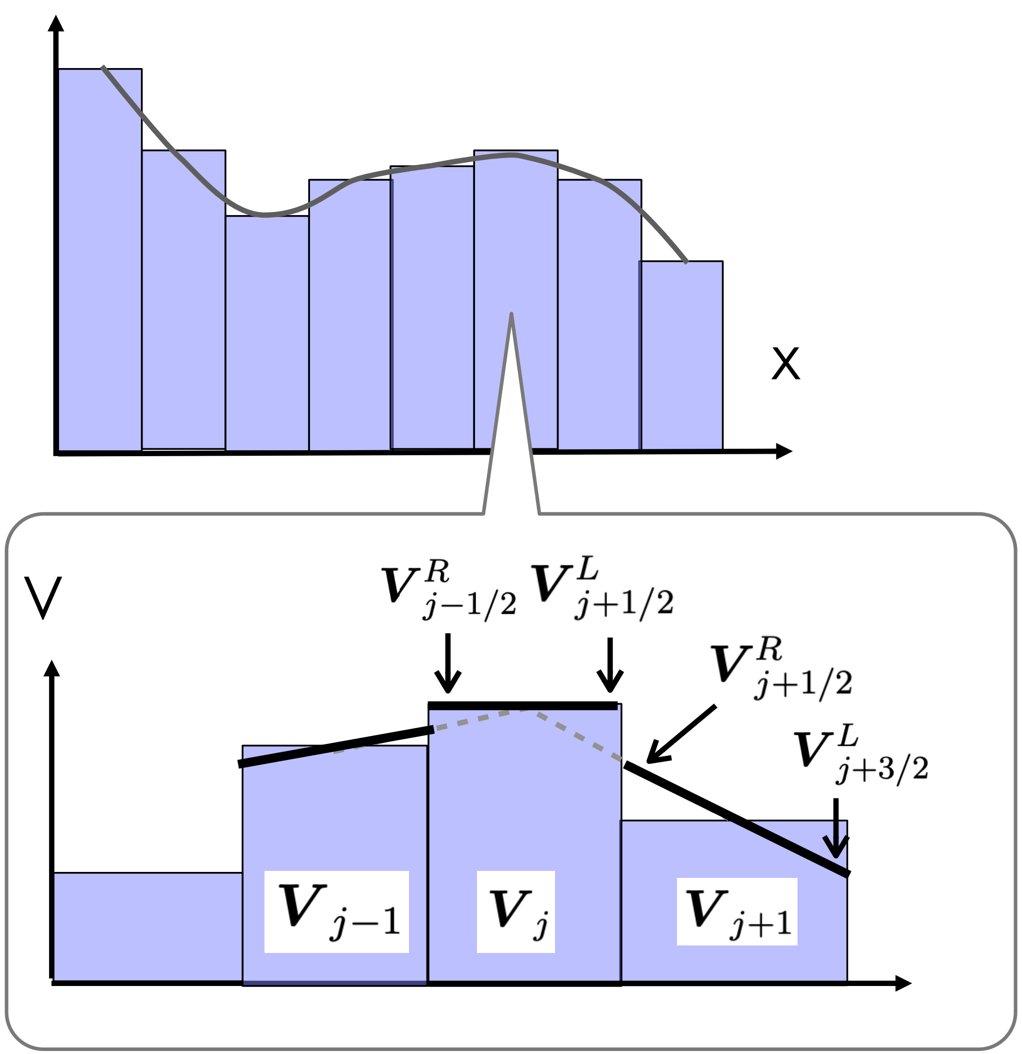
(Monotonic Upstream-centered Scheme for Conservation Laws)



- TVD (total variation diminishing) method
- minmod limiter
  - employ a smaller slope
  - employ a zero slope

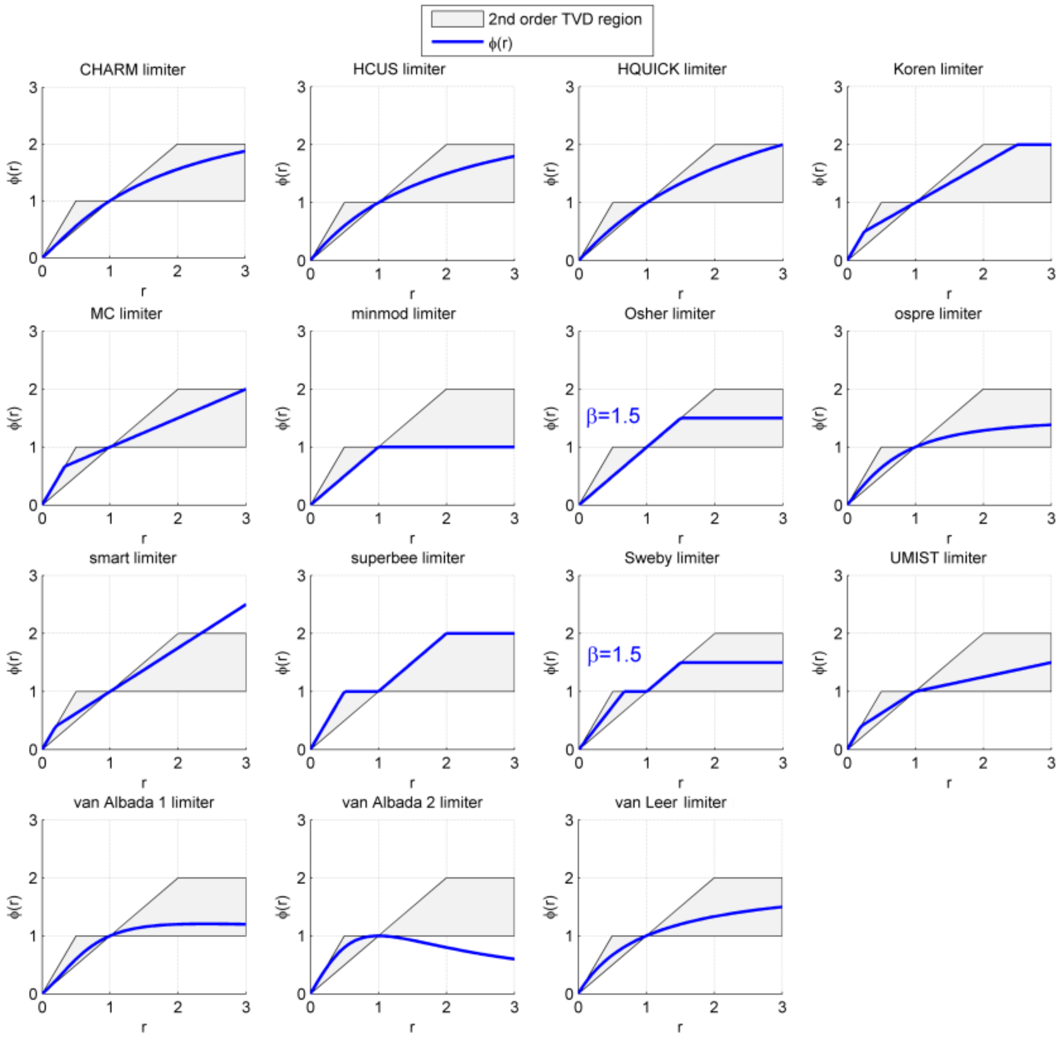
MUSCL interpolation

(Monotonic Upstream-centered Scheme for Conservation Laws)



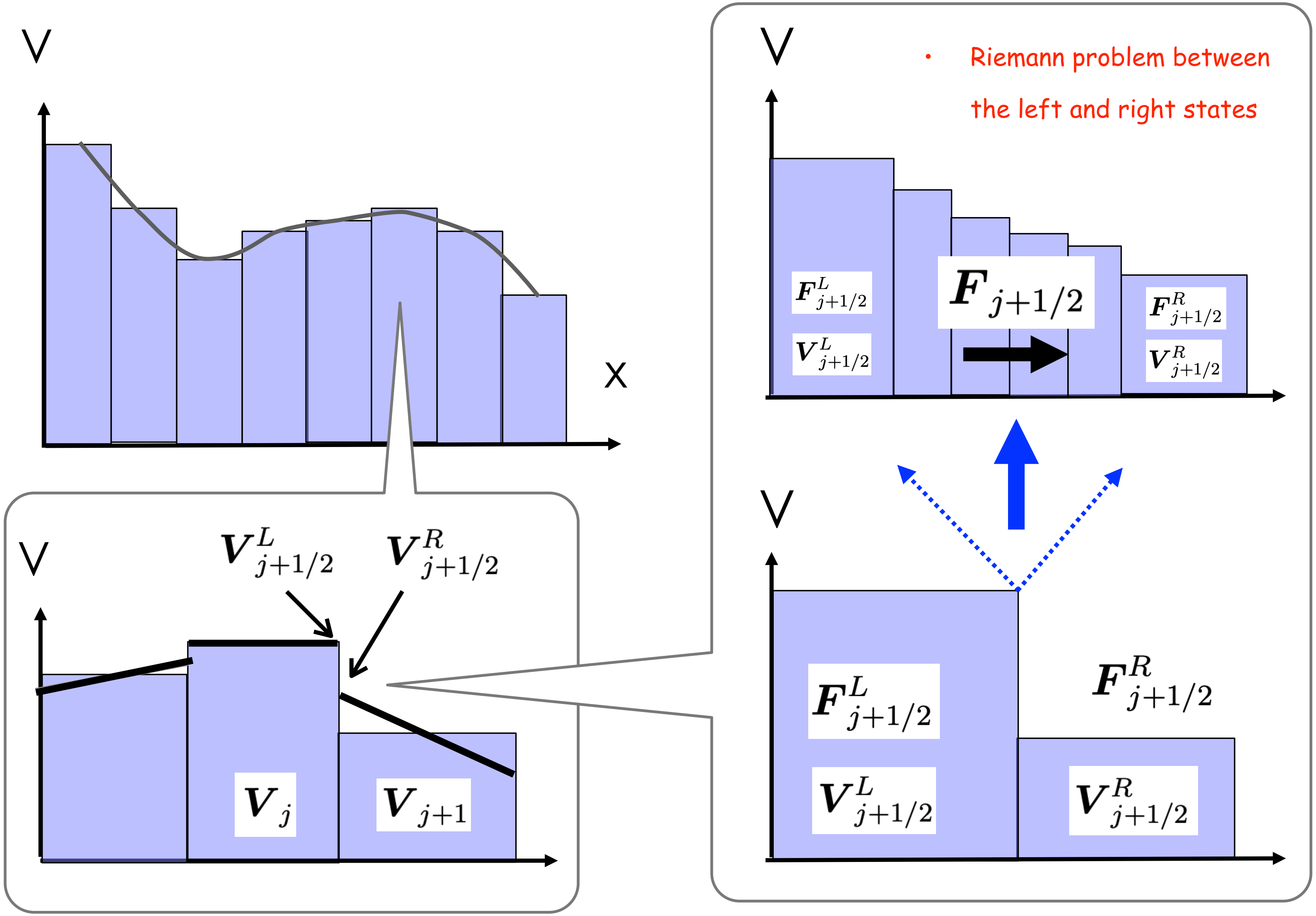
- TVD (total variation diminishing) method
- minmod limiter
  - employ a smaller slope
  - employ a zero slope

Various slope limiters

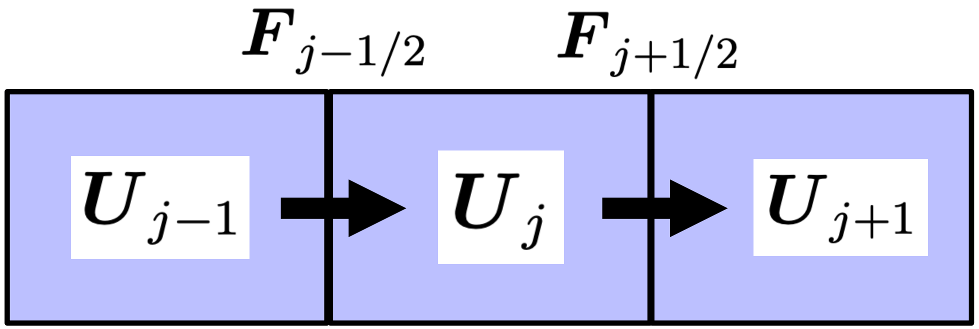


https://en.wikipedia.org/wiki/Flux_limiter

STEP 2/4: Numerical flux



STEP 3/4: Update U (1st order in time)



$$\boldsymbol{U}_j(t+\Delta t) = \boldsymbol{U}_j(t) - \frac{\Delta t}{\Delta x}(\boldsymbol{F}_{j+1/2} - \boldsymbol{F}_{j-1/2})$$

Higher-order evolution in time

- Time evolution (1st order)

$$\boldsymbol{U}_j^{n+1} = \boldsymbol{U}_j^n - \frac{\Delta t}{\Delta x}(\boldsymbol{F}_{j+1/2}^n - \boldsymbol{F}_{j-1/2}^n)$$
$$\equiv \boldsymbol{U}_j^n - \Delta t \mathcal{L}(\boldsymbol{U}_j^n)$$

- 2nd-order SSP Runge-Kutta (also known as TVD Runge-Kutta)

$$\boldsymbol{U}_j^* = \boldsymbol{U}_j^n - \Delta t \mathcal{L}(\boldsymbol{U}_j^n)$$
$$\boldsymbol{U}_j^{n+1} = \frac{1}{2}\left(\boldsymbol{U}_j^n\right) + \frac{1}{2}\left\{\boldsymbol{U}_j^* - \Delta t \mathcal{L}(\boldsymbol{U}_j^*)\right\}$$

- 3rd-order SSP Runge-Kutta is also popular

STEP 4/4: Recovery of primitive variables

Conserved variables

$$\boldsymbol{U} \equiv (\rho \quad \rho\boldsymbol{v} \quad \frac{1}{2}\rho v^2 + \frac{p}{\gamma - 1} + \frac{B^2}{8\pi} \quad \boldsymbol{B})^T$$

→

Primitive variables

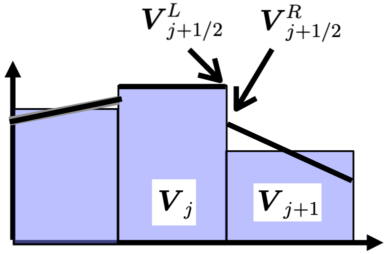
$$\boldsymbol{V} \equiv (\rho \quad \boldsymbol{v} \quad p \quad \boldsymbol{B})^T$$

- When the pressure becomes negative ($p<0$), we have to stop the simulation.
- Serious problem for finite-volume MHD codes in a low-beta plasma ($\beta \ll 0.1$).

Simulation cycle, again

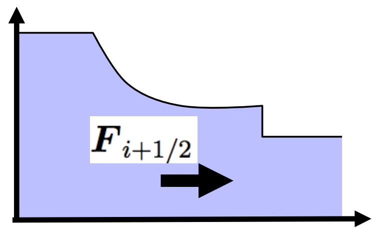
- 1. Spatial interpolation

  Compute left and right states: V_L, V_R



- 2. Calculate numerical flux F,

  considering a Riemann problem



- 3. Update conservative variable U

$$U_i(t+\Delta t) = U_i(t) - \frac{\Delta t}{\Delta x}(F_{i+1/2} - F_{i-1/2})$$

- 4. Recovery of primitive variables

  Convert U to V

$$U \equiv \left(\rho \quad \rho v \quad \frac{1}{2}\rho v^2 + \frac{p}{\gamma - 1} + \frac{B^2}{8\pi} \quad B\right)^T$$

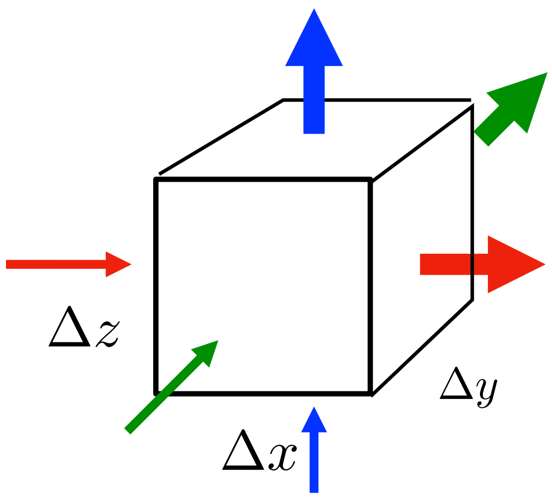
$$V \equiv (\rho \quad v \quad p \quad B)^T$$

5. MHD simulation in multi-dimensions

Finite volume method in multi-dimensions

$$\frac{\partial}{\partial t}\boldsymbol{U} + \frac{\partial}{\partial x}\boldsymbol{F} + \frac{\partial}{\partial y}\boldsymbol{G} + \frac{\partial}{\partial z}\boldsymbol{H} = 0$$

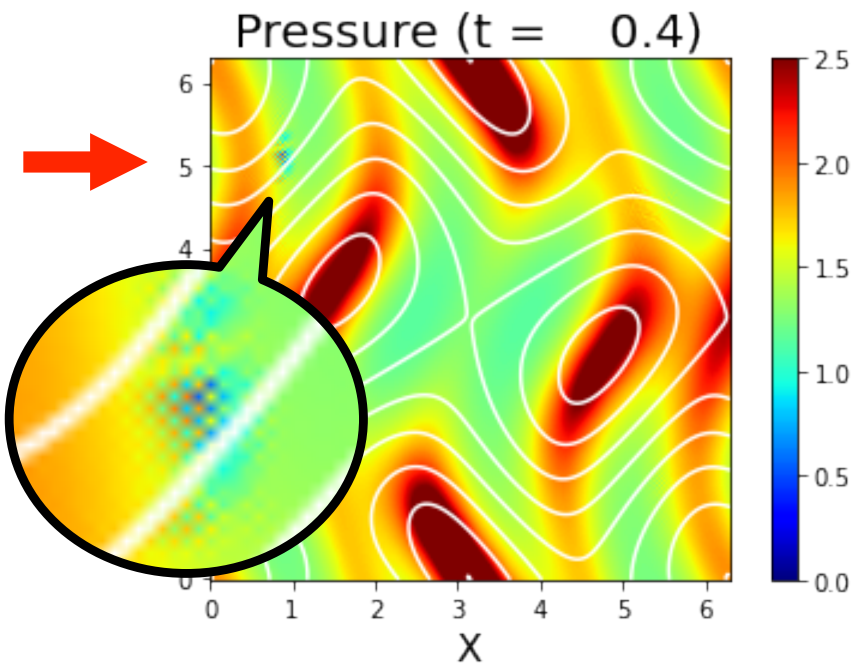
$$\begin{aligned}\boldsymbol{U}_{i,j,k}(t+\Delta t) = \boldsymbol{U}_{i,j,k}(t) &- \frac{\Delta t}{\Delta x}(\boldsymbol{F}_{i+1/2,j,k} - \boldsymbol{F}_{i-1/2,j,k}) \\ &- \frac{\Delta t}{\Delta y}(\boldsymbol{G}_{i,j+1/2,k} - \boldsymbol{G}_{i,j-1/2,k}) \\ &- \frac{\Delta t}{\Delta z}(\boldsymbol{H}_{i,j,k+1/2} - \boldsymbol{H}_{i,j,k-1/2})\end{aligned}$$

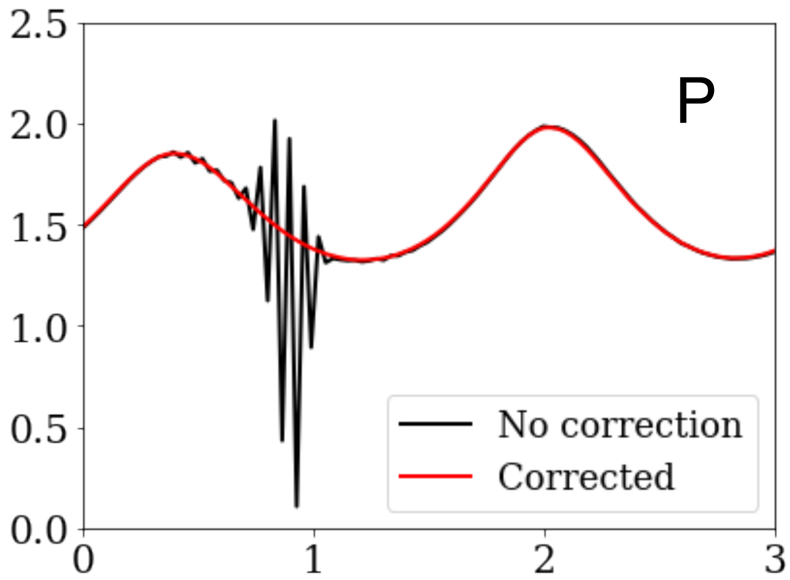


- Is that all? .... NO

Numerical oscillation in 2D/3D MHD

- MHD simulation often suffers from numerical oscillations, caused by divergence B
- Some correction is necessary





Unphysical force by div.B

- Gauss's law $$\nabla \cdot \boldsymbol{E} = 4\pi\rho_c$$
- Newton-Lorentz force $$\rho_c \left(\boldsymbol{E} + \frac{\boldsymbol{v}}{c} \times \boldsymbol{B} \right) = \frac{\nabla \cdot \boldsymbol{E}}{4\pi} \left(\boldsymbol{E} + \frac{\boldsymbol{v}}{c} \times \boldsymbol{B} \right)$$
- Numerical magnetic charge $$\nabla \cdot \boldsymbol{B} \neq 0$$
- Unphysical force starts to work $$\frac{\nabla \cdot \boldsymbol{B}}{4\pi} \left(\boldsymbol{B} - \frac{\boldsymbol{v}}{c} \times \boldsymbol{E} \right)$$
- "Divergence cleaning": we need to keep div B to small or zero

Workaround 1: Projection method

- Helmholtz decomposition for arbitrary vector field

$$\boldsymbol{u} = \nabla\phi + \nabla \times \boldsymbol{A}$$

- We assume $\boldsymbol{B} = \nabla\phi_{\rm err} + \nabla \times \boldsymbol{A}$

- Solve a Poisson equation (SOR method, FFT ...) to obtain $\phi_{\rm err}$

$$\nabla \cdot \boldsymbol{B}_{\rm sim} = \nabla \cdot \nabla\phi_{\rm err} + \nabla \cdot (\nabla \times \boldsymbol{A}) = \Delta\phi_{\rm err}$$

- Correct the magnetic field

$$\boldsymbol{B}_{\rm new} = \boldsymbol{B}_{\rm sim} - \nabla\phi_{\rm err}$$

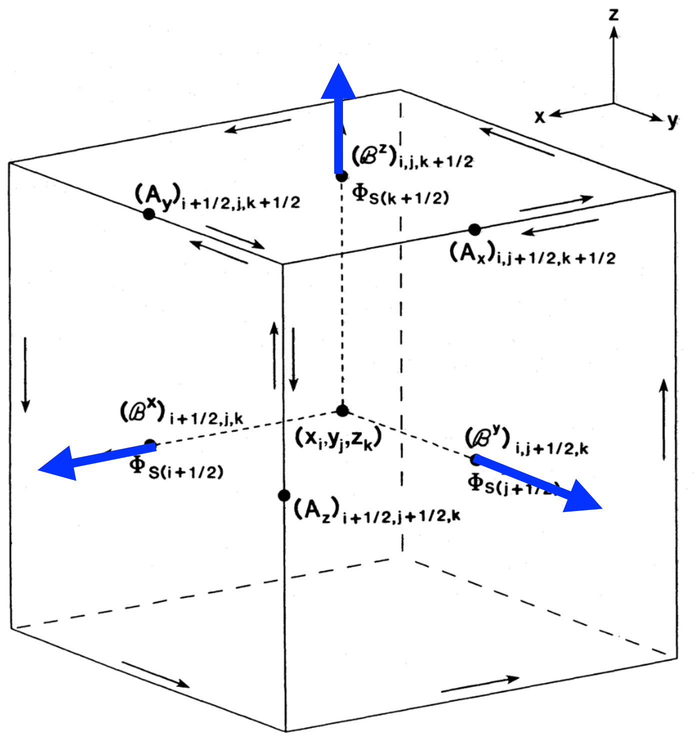
- Then the new magnetic field satisfies

$$\nabla \cdot \boldsymbol{B}_{\rm new} = 0$$

Brackbill & Barnes 1980 JCP

Workaround 2: Constraint Transport (CT)

- Assign the magnetic field on a cell center
- There are several variants



$$
\begin{aligned}
B^{n+1}_{x,i-1/2,j,k} &= B^{n}_{x,i-1/2,j,k} \\
&\quad - \frac{\Delta t}{\Delta y}\left(E^{n+1/2}_{z,i-1/2,j+1/2,k} - E^{n+1/2}_{z,i-1/2,j,k}\right) \\
&\quad + \frac{\Delta t}{\Delta z}\left(E^{n+1/2}_{y,i-1/2,j,k+1/2} - E^{n+1/2}_{y,i-1/2,j,k-1/2}\right)
\end{aligned}
$$

$$
B^{n+1}_{y,i,j-1/2,k} = \dots
$$

$$
B^{n+1}_{z,i,j,k-1/2} = \dots
$$

$$
\begin{aligned}
(\nabla \cdot \boldsymbol{B})^{n+1}_{i,j,k} &= \frac{B^{n+1}_{x,i+1/2,j,k} - B^{n+1}_{x,i-1/2,j,k}}{\Delta x} + \frac{B^{n+1}_{x,i,j+1/2,k} - B^{n+1}_{x,i,j-1/2,k}}{\Delta y} \\
&\quad + \frac{B^{n+1}_{x,i,j,k+1/2} - B^{n+1}_{x,i,j,k-1/2}}{\Delta z} \\
&= (\nabla \cdot \boldsymbol{B})^{n}_{i,j,k}
\end{aligned}
$$

Evans & Hawley 1988 ApJ

Workaround 3: Hyperbolic divergence cleaning

- A virtual potential ψ is introduced

$$
\begin{aligned}
\frac{\partial \rho}{\partial t} &+ \nabla \cdot (\rho \boldsymbol{v}) = 0 \\
\frac{\partial \rho \boldsymbol{v}}{\partial t} &+ \nabla \cdot (\rho \boldsymbol{v}\boldsymbol{v} + p_t \mathbb{I} - \boldsymbol{B}\boldsymbol{B}) = 0 \\
\frac{\partial \mathcal{E}}{\partial t} &+ \nabla \cdot \Big((\mathcal{E} + p_t)\boldsymbol{v} - (\boldsymbol{v} \cdot \boldsymbol{B})\boldsymbol{B} \Big) = 0 \\
\frac{\partial \boldsymbol{B}}{\partial t} &+ \nabla \cdot (\boldsymbol{v}\boldsymbol{B} - \boldsymbol{B}\boldsymbol{v}) + \nabla \psi = 0 \\
\frac{\partial \psi}{\partial t} &+ c_h^2 \nabla \cdot \boldsymbol{B} = -\left(\frac{c_h^2}{c_p^2}\right)\psi
\end{aligned}
$$

- Div B is temporally stored to ψ.
- Then ψ tries to adjust B, diffuse itself, and decay.

Dedner+ 2002 JCP

Almost done!

1. MHD at a glance
2. Basic theory: Advection problem
3. Basic theory: Finite-volume method and Riemann solver
4. MHD simulation with Riemann solver
5. MHD simulation in multi-dimensions
6. Hands on

Further reading

- Plasma Physics for Astrophysics, R. M. Kulsrud (2004)
- Magnetohydrodynamics of the Sun, E. R. Priest (2014)
- A multi-state HLL approximate Riemann solver for ideal magnetohydrodynamics, Miyoshi & Kusano, J. Comput. Phys. (2005)
- The div B = 0 Constraint in Shock-Capturing Magnetohydrodynamics Codes, Toth, J. Comput. Phys. (2000)
- Riemann Solvers and Numerical Methods for Fluid Dynamics: A Practical Introduction, E. F. Toro (2010)

Public MHD codes

- Athena++ (Princeton)
  - https://www.athena-astro.app/
  - C++, MPI+OpenMP, General relativistic MHD
- Pluto (A. Mignone)
  - https://plutocode.ph.unito.it/
  - C, MPI, Relativistic MHD
- MURaM (Max Planck, U. Chicago)
  - https://www2.mps.mpg.de/projects/solar-mhd/muram_site/
  - Fortran 90, MPI+OpenACC, Radiative MHD
- CANS+ (Y. Matsumoto)
  - http://www.astro.phys.s.chiba-u.ac.jp/cans/doc/ (in Japanese)
  - Fortran 90, MPI+OpenMP, Python, IDL visualization
- OpenMHD
  - https://sci.nao.ac.jp/MEMBER/zenitani/openmhd-e.html
  - Fortran 90 and CUDA Fortran, MPI+OpenMP, Python, IDL visualization

- I recommend you to find a well-tested MHD code, written by your favorite language (C/C++ or Fortran).

Backup slides

NOTE: Units

	Ohm's law	Alfvén speed
• cgs units	$\boldsymbol{E} + \frac{\boldsymbol{v}}{c} \times \boldsymbol{B} = 0,$	$c_A = \frac{B}{\sqrt{4\pi\rho}}$
• Simulation units	$\boldsymbol{E} + \boldsymbol{v} \times \boldsymbol{B} = 0,$	$c_A = \frac{B}{\sqrt{\rho}}$
• MKS units	$\boldsymbol{E} + \boldsymbol{v} \times \boldsymbol{B} = 0,$	$c_A = \frac{B}{\sqrt{\mu_0\rho}}$

Lorentz force = magnetic pressure + magnetic tension

Ampere's law

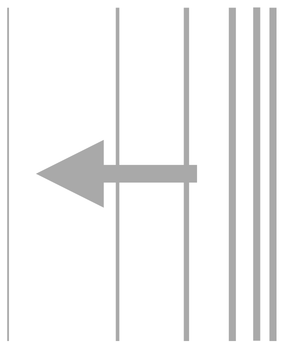
$$\nabla \times \boldsymbol{B} = \frac{4\pi}{c}\boldsymbol{j} + \frac{1}{c}\frac{\partial \boldsymbol{E}}{\partial t}$$

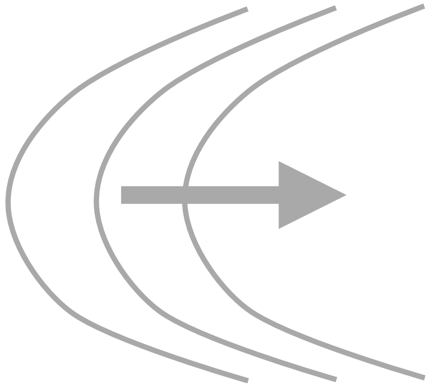
$$\rho \frac{d\boldsymbol{v}}{dt} = -\nabla p + \frac{\boldsymbol{j} \times \boldsymbol{B}}{c}$$

$$= -\nabla p - \nabla\left(\frac{B^2}{8\pi}\right) + \frac{1}{4\pi}(\boldsymbol{B}\cdot\nabla)\boldsymbol{B}$$

"magnetic pressure" "magnetic tension"

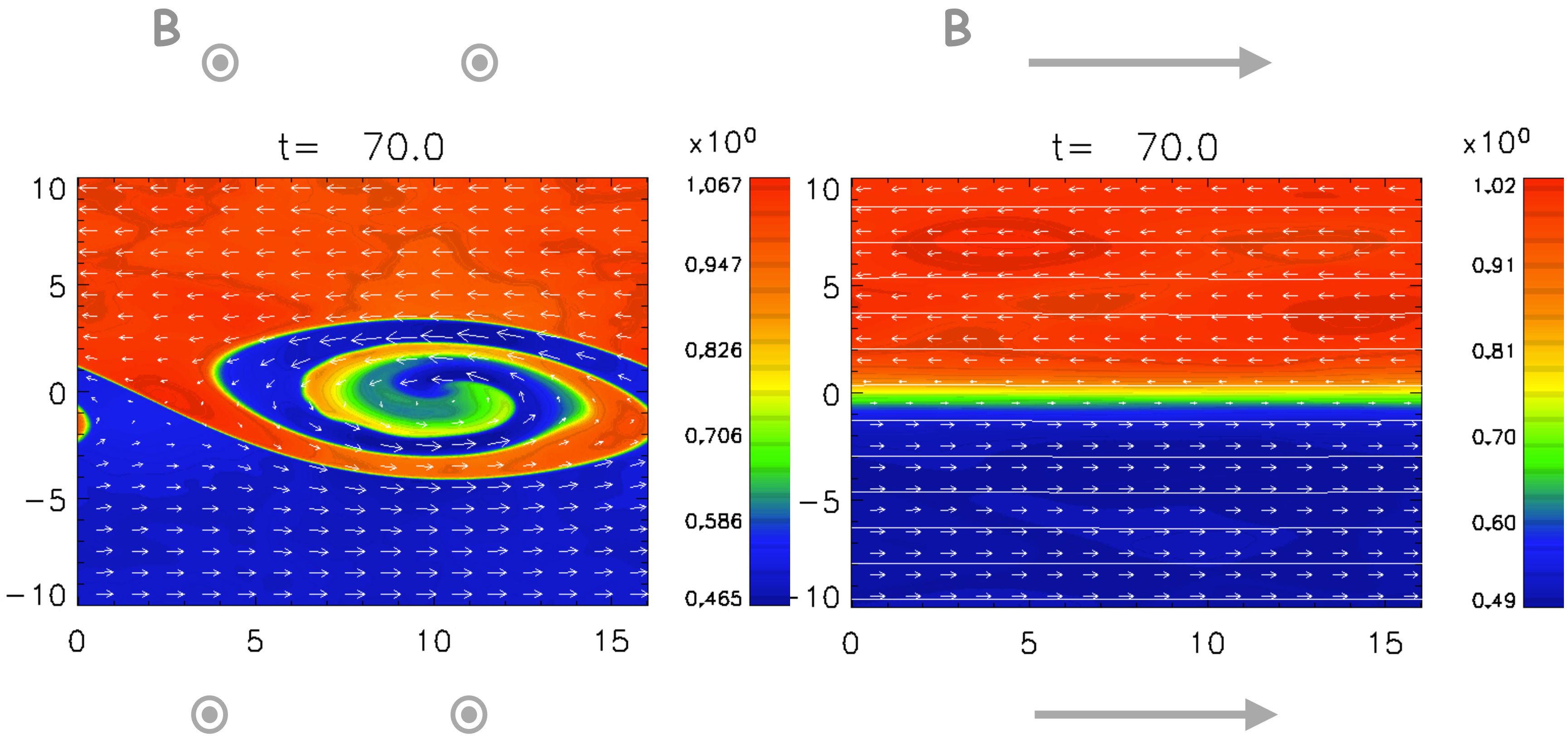
- Examples:
  - B = (0,x,0) B = (y,1,0)



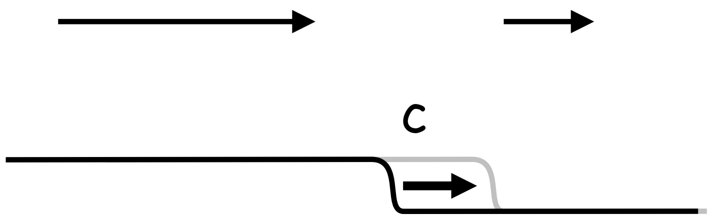


How magnetic tension works ...

- Kelvin-Helmholtz instability in a flow-shear region
- Magnetic field lines tend to be straight



FTCS method vs Upwind method



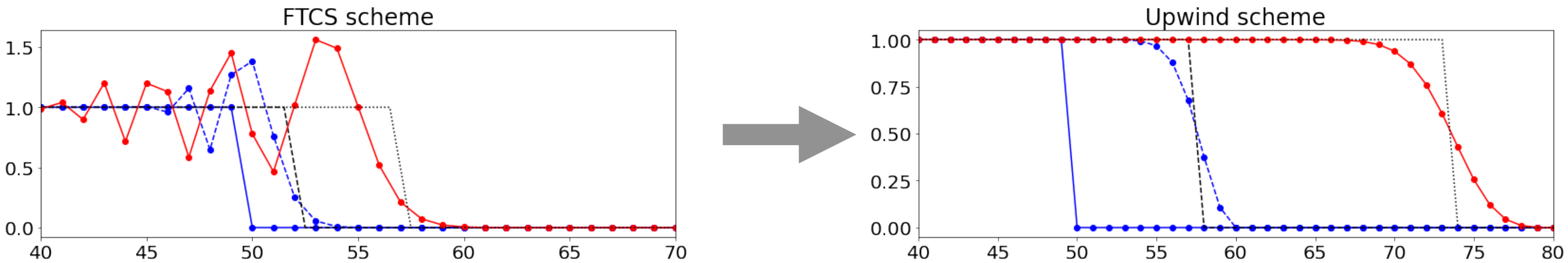
$$
\begin{aligned}
\frac{\partial u}{\partial t} + c\frac{\partial u}{\partial x} &= 0 \\
f &= cu
\end{aligned}
$$

- Equivalent to adding a second-order derivative (diffusion term)

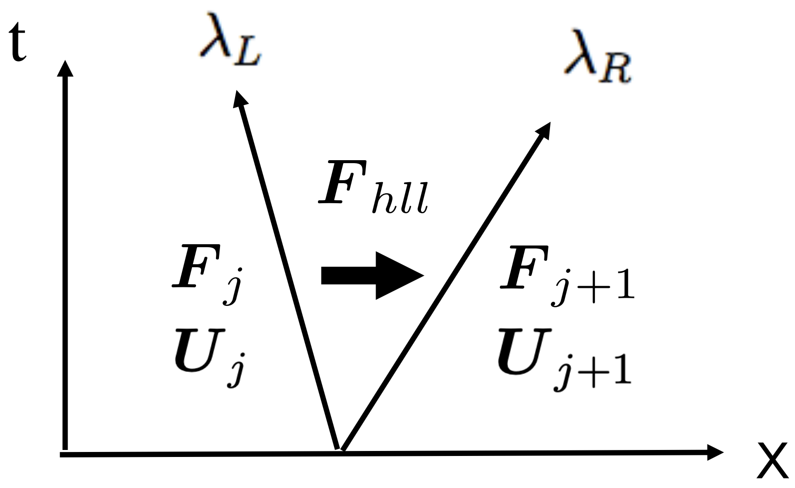
$$u_j^{n+1} = u_j^n - \frac{c\Delta t}{2\Delta x}\left(u_{j+1}^n - u_{j-1}^n\right) + \frac{|c|\Delta t\Delta x}{2}\frac{\left(u_{j+1}^n - 2u_j^n + u_{j-1}^n\right)}{\Delta x^2}$$

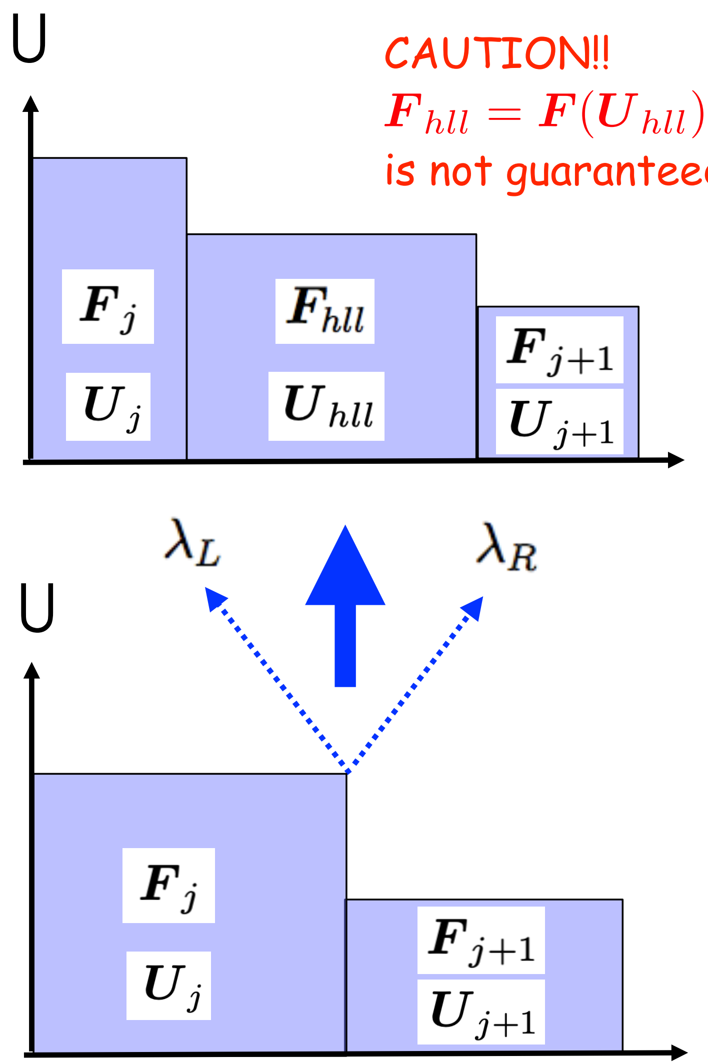
$$\approx \frac{\partial^2 u}{\partial x^2}$$

FTCS solver — Diffusion term



Approximate Riemann solver - HLL solver





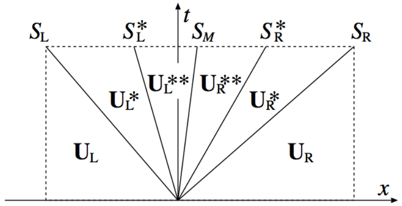
CAUTION!!
$\boldsymbol{F}_{hll} = \boldsymbol{F}(\boldsymbol{U}_{hll})$
is not guaranteed

$$\boldsymbol{F} = \begin{cases} \boldsymbol{F}_j & (\lambda_L > 0) \\ \boldsymbol{F}_{hll} & (\lambda_L \leq 0 \leq \lambda_R) \\ \boldsymbol{F}_{j+1} & (\lambda_R < 0) \end{cases}$$

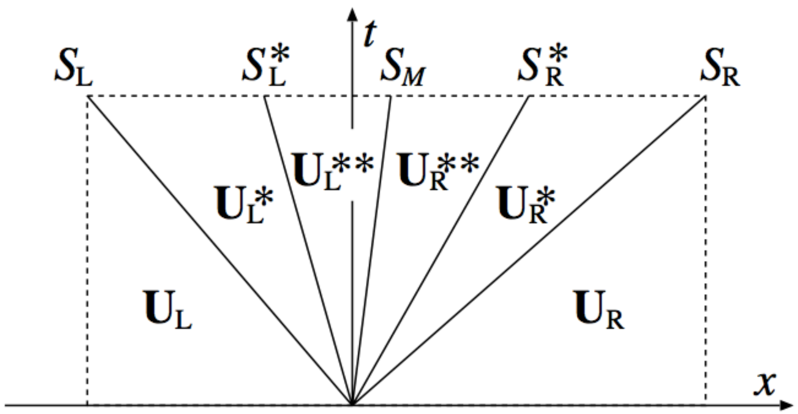
$$\boldsymbol{F}_{hll} = \frac{\lambda_R \boldsymbol{F}_j - \lambda_L \boldsymbol{F}_{j+1} + \lambda_R \lambda_L (\boldsymbol{U}_{j+1} - \boldsymbol{U}_j)}{\lambda_R - \lambda_L}$$

Harten, Lan, & van Leer 1983 *SIAM Rev.*

HLLD solver in detail [1/24]



- In the next 24 slides, I show the derivation of the HLLD solution.
- The presentation materials are provided by Dr. Miyoshi, who developed the HLLD scheme.

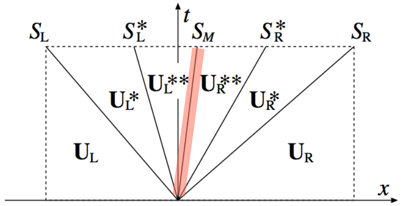


$$\frac{\partial \boldsymbol{U}}{\partial t} + \frac{\partial \boldsymbol{F}}{\partial x} = 0, \quad B_x = \text{const.},$$

$$\boldsymbol{U} = \begin{pmatrix} \rho \\ \rho u \\ \rho v \\ \rho w \\ B_y \\ B_z \\ e \end{pmatrix}, \quad \boldsymbol{F} = \begin{pmatrix} \rho u \\ \rho u u + p_T - B_x^2 \\ \rho v u - B_x B_y \\ \rho w u - B_x B_z \\ B_y u - B_x v \\ B_z u - B_x w \\ (e + p_T) u - B_x (u B_x + v B_y + w B_z) \end{pmatrix}$$

Miyoshi & Kusano 2005 JCP

HLLD solver [2/24]



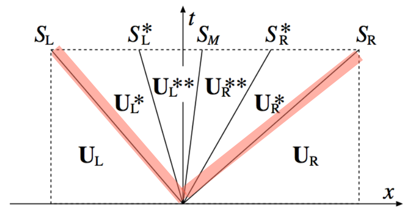
- Evaluating an entropy wave speed (Battern+ 1997)

$$S_M = \frac{(\rho u)^*}{\rho^*} = \frac{(S_R - u_R)\rho_R u_R - (S_L - u_L)\rho_L u_L - p_{TR} + p_{TL}}{(S_R - u_R)\rho_R - (S_L - u_L)\rho_L}$$

- Evaluating the total pressure

$$\begin{aligned} p_T^* &= p_{TL} + \rho_L (S_L - u_L)(S_M - u_L) \\ &= p_{TR} + \rho_R (S_R - u_R)(S_M - u_R) \\ &= \frac{(S_R - u_R)\rho_R p_{TL} - (S_L - u_L)\rho_L p_{TR} + \rho_L \rho_R (S_R - u_R)(u_R - u_L)}{(S_R - u_R)\rho_R - (S_L - u_L)\rho_L} \end{aligned}$$

HLLD solver [3/24]

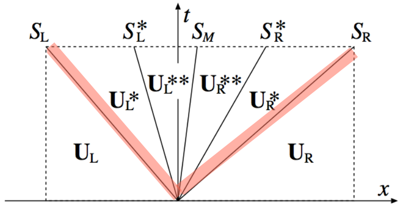


- Jump conditions across the left/right boundaries

$$S_\alpha \begin{pmatrix} \rho_\alpha^* \\ \rho_\alpha^* S_M \\ \rho_\alpha^* v_\alpha^* \\ \rho_\alpha^* w_\alpha^* \\ B_{y\alpha}^* \\ B_{z\alpha}^* \\ e_\alpha^* \end{pmatrix} - \begin{pmatrix} \rho_\alpha^* S_M \\ \rho_\alpha^* S_M^2 + p_T^* - B_x^2 \\ \rho_\alpha^* v_\alpha^* S_M - B_x B_{y\alpha}^* \\ \rho_\alpha^* w_\alpha^* S_M - B_x B_{z\alpha}^* \\ B_{y\alpha}^* S_M - B_x v_\alpha^* \\ B_{z\alpha}^* S_M - B_x w_\alpha^* \\ (e_\alpha^* + p_T^*) S_M - B_x (\boldsymbol{v}_\alpha^* \cdot \boldsymbol{B}_\alpha^*) \end{pmatrix} = S_\alpha \begin{pmatrix} \rho_\alpha \\ \rho_\alpha u_\alpha \\ \rho_\alpha v_\alpha \\ \rho_\alpha w_\alpha \\ B_{y\alpha} \\ B_{z\alpha} \\ e_\alpha \end{pmatrix} - \begin{pmatrix} \rho_\alpha u_\alpha \\ \rho_\alpha u_\alpha + p_{T\alpha} - B_x^2 \\ \rho_\alpha v_\alpha u_\alpha - B_x B_{y\alpha} \\ \rho_\alpha w_\alpha u_\alpha - B_x B_{z\alpha} \\ B_{y\alpha} u_\alpha - B_x v_\alpha \\ B_{z\alpha} u_\alpha - B_x w_\alpha \\ (e_\alpha + p_{T\alpha}) u_\alpha - B_x (\boldsymbol{v}_\alpha \cdot \boldsymbol{B}_\alpha) \end{pmatrix}$$

$$\boldsymbol{v}_\alpha^* = (S_M, v_\alpha^*, w_\alpha^*), \boldsymbol{B}_\alpha^* = (B_x, B_{y\alpha}^*, B_{z\alpha}^*) \alpha = R, L$$

HLLD solver [4/24]

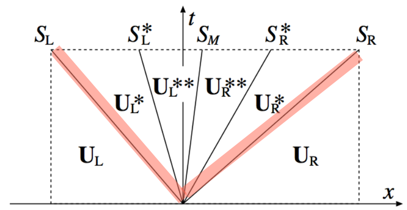


- Solution:

$$\boldsymbol{U}_\alpha^*$$

$$\rho_\alpha^* = \rho_\alpha \frac{S_\alpha - u_\alpha}{S_\alpha - S_M}$$

HLLD solver [5/24]

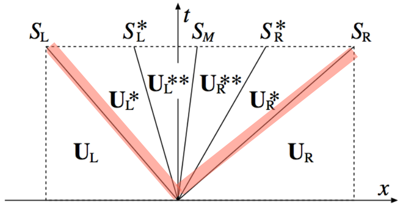


- Jump conditions across the left/right boundaries

$$S_\alpha \begin{pmatrix} \rho_\alpha^* \\ \rho_\alpha^* S_M \\ \rho_\alpha^* v_\alpha^* \\ \rho_\alpha^* w_\alpha^* \\ B_{y\alpha}^* \\ B_{z\alpha}^* \\ e_\alpha^* \end{pmatrix} - \begin{pmatrix} \rho_\alpha^* S_M \\ \rho_\alpha^* S_M^2 + p_T^* - B_x^2 \\ \rho_\alpha^* v_\alpha^* S_M - B_x B_{y\alpha}^* \\ \rho_\alpha^* w_\alpha^* S_M - B_x B_{z\alpha}^* \\ B_{y\alpha}^* S_M - B_x v_\alpha^* \\ B_{z\alpha}^* S_M - B_x w_\alpha^* \\ (e_\alpha^* + p_T^*) S_M - B_x (\boldsymbol{v}_\alpha^* \cdot \boldsymbol{B}_\alpha^*) \end{pmatrix} = S_\alpha \begin{pmatrix} \rho_\alpha \\ \rho_\alpha u_\alpha \\ \rho_\alpha v_\alpha \\ \rho_\alpha w_\alpha \\ B_{y\alpha} \\ B_{z\alpha} \\ e_\alpha \end{pmatrix} - \begin{pmatrix} \rho_\alpha u_\alpha \\ \rho_\alpha u_\alpha + p_{T\alpha} - B_x^2 \\ \rho_\alpha v_\alpha u_\alpha - B_x B_{y\alpha} \\ \rho_\alpha w_\alpha u_\alpha - B_x B_{z\alpha} \\ B_{y\alpha} u_\alpha - B_x v_\alpha \\ B_{z\alpha} u_\alpha - B_x w_\alpha \\ (e_\alpha + p_{T\alpha}) u_\alpha - B_x (\boldsymbol{v}_\alpha \cdot \boldsymbol{B}_\alpha) \end{pmatrix}$$

$$\boldsymbol{v}_\alpha^* = (S_M, v_\alpha^*, w_\alpha^*), \boldsymbol{B}_\alpha^* = (B_x, B_{y\alpha}^*, B_{z\alpha}^*), \alpha = R, L$$

HLLD solver [6/24]



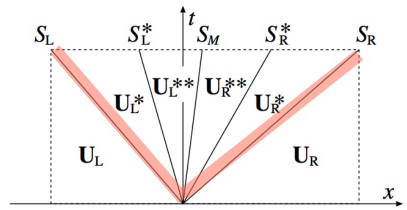
- Solution:

$$\boldsymbol{U}_\alpha^*$$

$$\rho_\alpha^* = \rho_\alpha \frac{S_\alpha - u_\alpha}{S_\alpha - S_M}$$

$$\begin{cases} \boldsymbol{v}_{t\alpha}^* = \boldsymbol{v}_{t\alpha} - B_x \boldsymbol{B}_{t\alpha} \dfrac{S_M - u_\alpha}{\rho_\alpha (S_\alpha - u_\alpha)(S_\alpha - S_M) - B_x^2} \\ \boldsymbol{B}_{t\alpha}^* = \boldsymbol{B}_{t\alpha} \dfrac{\rho_\alpha (S_\alpha - u_\alpha)^2 - B_x^2}{\rho_\alpha (S_\alpha - u_\alpha)(S_\alpha - S_M) - B_x^2} \end{cases} \quad \boldsymbol{v}_t = (0, v, w),\ \boldsymbol{B}_t = (0, B_y, B_z)$$

HLLD solver [7/24]

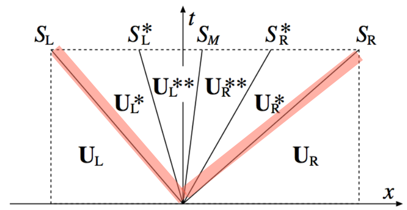


- Jump conditions across the left/right boundaries

$$
S_\alpha \begin{pmatrix} \rho_\alpha^* \\ \rho_\alpha^* S_M \\ \rho_\alpha^* v_\alpha^* \\ \rho_\alpha^* w_\alpha^* \\ B_{y\alpha}^* \\ B_{z\alpha}^* \\ e_\alpha^* \end{pmatrix} - \begin{pmatrix} \rho_\alpha^* S_M \\ \rho_\alpha^* S_M^2 + p_T^* - B_x^2 \\ \rho_\alpha^* v_\alpha^* S_M - B_x B_{y\alpha}^* \\ \rho_\alpha^* w_\alpha^* S_M - B_x B_{z\alpha}^* \\ B_{y\alpha}^* S_M - B_x v_\alpha^* \\ B_{z\alpha}^* S_M - B_x w_\alpha^* \\ (e_\alpha^* + p_T^*) S_M - B_x (\boldsymbol{v}_\alpha^* \cdot \boldsymbol{B}_\alpha^*) \end{pmatrix} = S_\alpha \begin{pmatrix} \rho_\alpha \\ \rho_\alpha u_\alpha \\ \rho_\alpha v_\alpha \\ \rho_\alpha w_\alpha \\ B_{y\alpha} \\ B_{z\alpha} \\ e_\alpha \end{pmatrix} - \begin{pmatrix} \rho_\alpha u_\alpha \\ \rho_\alpha u_\alpha + p_{T\alpha} - B_x^2 \\ \rho_\alpha v_\alpha u_\alpha - B_x B_{y\alpha} \\ \rho_\alpha w_\alpha u_\alpha - B_x B_{z\alpha} \\ B_{y\alpha} u_\alpha - B_x v_\alpha \\ B_{z\alpha} u_\alpha - B_x w_\alpha \\ (e_\alpha + p_{T\alpha}) u_\alpha - B_x (\boldsymbol{v}_\alpha \cdot \boldsymbol{B}_\alpha) \end{pmatrix}
$$

$$
\boldsymbol{v}_\alpha^* = (S_M, v_\alpha^*, w_\alpha^*), \boldsymbol{B}_\alpha^* = (B_x, B_{y\alpha}^*, B_{z\alpha}^*) \alpha = R, L
$$

HLLD solver [8/24]



- Solution:

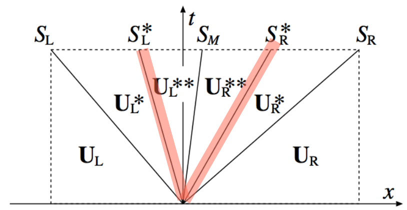
$$\boldsymbol{U}_\alpha^*$$

$$\rho_\alpha^* = \rho_\alpha \frac{S_\alpha - u_\alpha}{S_\alpha - S_M}$$

$$\begin{cases} \boldsymbol{v}_{t\alpha}^* = \boldsymbol{v}_{t\alpha} - B_x \boldsymbol{B}_{t\alpha} \dfrac{S_M - u_\alpha}{\rho_\alpha (S_\alpha - u_\alpha)(S_\alpha - S_M) - B_x^2} \\ \boldsymbol{B}_{t\alpha}^* = \boldsymbol{B}_{t\alpha} \dfrac{\rho_\alpha (S_\alpha - u_\alpha)^2 - B_x^2}{\rho_\alpha (S_\alpha - u_\alpha)(S_\alpha - S_M) - B_x^2} \end{cases} \quad \boldsymbol{v}_t = (0, v, w),\ \boldsymbol{B}_t = (0, B_y, B_z)$$

$$e_\alpha^* = \frac{(S_\alpha - u_\alpha) e_\alpha - p_{T\alpha} + p_T^* + B_x \left(\boldsymbol{v}_\alpha \cdot \boldsymbol{B}_\alpha - \boldsymbol{v}_\alpha^* \cdot \boldsymbol{B}_\alpha^* \right)}{S_\alpha - S_M}$$

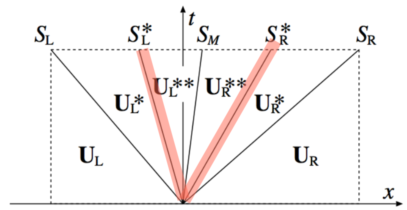
HLLD solver [9/24]



- Jump conditions across the rotational discontinuities

$$S_\alpha^* \begin{pmatrix} \rho_\alpha^{**} \\ \rho_\alpha^{**} S_M \\ \rho_\alpha^{**} v_\alpha^{**} \\ \rho_\alpha^{**} w_\alpha^{**} \\ B_{y\alpha}^{**} \\ B_{z\alpha}^{**} \\ e_\alpha^{**} \end{pmatrix} - \begin{pmatrix} \rho_\alpha^{**} S_M \\ \rho_\alpha^{**} S_M^2 + p_T^* - B_x^2 \\ \rho_\alpha^{**} v_\alpha^{**} S_M - B_x B_{y\alpha}^{**} \\ \rho_\alpha^{**} w_\alpha^{**} S_M - B_x B_{z\alpha}^{**} \\ B_{y\alpha}^{**} S_M - B_x v_\alpha^{**} \\ B_{z\alpha}^{**} S_M - B_x w_\alpha^{**} \\ \left(e_\alpha^{**} + p_T^*\right) S_M - B_x \left(\boldsymbol{v}_\alpha^{**} \cdot \boldsymbol{B}_\alpha^{**}\right) \end{pmatrix} = S_\alpha^* \begin{pmatrix} \rho_\alpha^{*} \\ \rho_\alpha^{*} S_M \\ \rho_\alpha^{*} v_\alpha^{*} \\ \rho_\alpha^{*} w_\alpha^{*} \\ B_{y\alpha}^{*} \\ B_{z\alpha}^{*} \\ e_\alpha^{*} \end{pmatrix} - \begin{pmatrix} \rho_\alpha^{*} S_M \\ \rho_\alpha^{*} S_M^2 + p_T^* - B_x^2 \\ \rho_\alpha^{*} v_\alpha^{*} S_M - B_x B_{y\alpha}^{*} \\ \rho_\alpha^{*} w_\alpha^{*} S_M - B_x B_{z\alpha}^{*} \\ B_{y\alpha}^{*} S_M - B_x v_\alpha^{*} \\ B_{z\alpha}^{*} S_M - B_x w_\alpha^{*} \\ \left(e_\alpha^{*} + p_T^*\right) S_M - B_x \left(\boldsymbol{v}_\alpha^{*} \cdot \boldsymbol{B}_\alpha^{*}\right) \end{pmatrix}$$

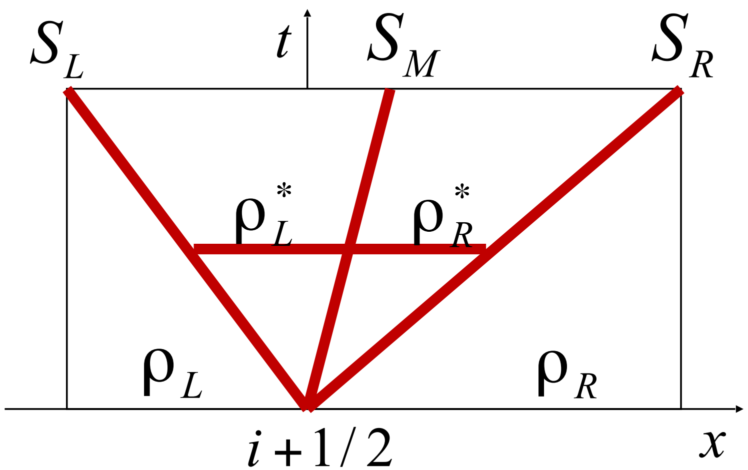
HLLD solver [10/24]



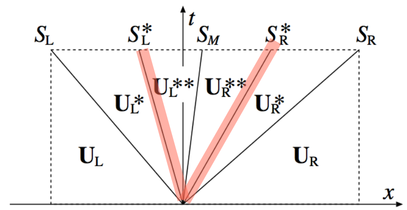
- Solution: $\boldsymbol{U}_\alpha^{**}$

$$\rho_\alpha^{**} = \rho_\alpha^*$$

$$S_R^* = S_M + \frac{|B_x|}{\sqrt{\rho_R^*}}, S_L^* = S_M - \frac{|B_x|}{\sqrt{\rho_L^*}}$$



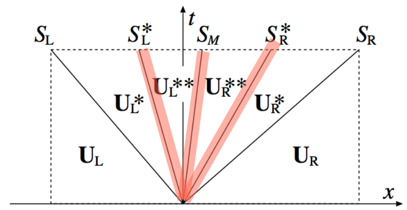
HLLD solver [11/24]



- Jump conditions across the rotational discontinuities

$$S_\alpha^* \begin{pmatrix} \rho_\alpha^{**} \\ \rho_\alpha^{**} S_M \\ \rho_\alpha^{**} v_\alpha^{**} \\ \rho_\alpha^{**} w_\alpha^{**} \\ B_{y\alpha}^{**} \\ B_{z\alpha}^{**} \\ e_\alpha^{**} \end{pmatrix} - \begin{pmatrix} \rho_\alpha^{**} S_M \\ \rho_\alpha^{**} S_M^2 + p_T^* - B_x^2 \\ \rho_\alpha^{**} v_\alpha^{**} S_M - B_x B_{y\alpha}^{**} \\ \rho_\alpha^{**} w_\alpha^{**} S_M - B_x B_{z\alpha}^{**} \\ B_{y\alpha}^{**} S_M - B_x v_\alpha^{**} \\ B_{z\alpha}^{**} S_M - B_x w_\alpha^{**} \\ (e_\alpha^{**} + p_T^*) S_M - B_x (\boldsymbol{v}_\alpha^{**} \cdot \boldsymbol{B}_\alpha^{**}) \end{pmatrix} = S_\alpha^* \begin{pmatrix} \rho_\alpha^* \\ \rho_\alpha^* S_M \\ \rho_\alpha^* v_\alpha^* \\ \rho_\alpha^* w_\alpha^* \\ B_{y\alpha}^* \\ B_{z\alpha}^* \\ e_\alpha^* \end{pmatrix} - \begin{pmatrix} \rho_\alpha^* S_M \\ \rho_\alpha^* S_M^2 + p_T^* - B_x^2 \\ \rho_\alpha^* v_\alpha^* S_M - B_x B_{y\alpha}^* \\ \rho_\alpha^* w_\alpha^* S_M - B_x B_{z\alpha}^* \\ B_{y\alpha}^* S_M - B_x v_\alpha^* \\ B_{z\alpha}^* S_M - B_x w_\alpha^* \\ (e_\alpha^* + p_T^*) S_M - B_x (\boldsymbol{v}_\alpha^* \cdot \boldsymbol{B}_\alpha^*) \end{pmatrix}$$

HLLD solver [12/24]



- Jump conditions across the CD (entropy wave)

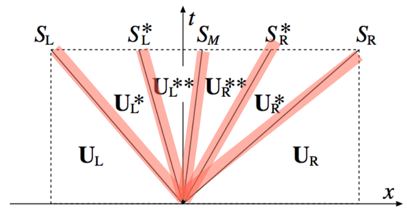
$$\det\left(M\left(\boldsymbol{v}_{t\alpha}^{**}, \boldsymbol{B}_{t\alpha}^{**}\right)\right) = 0$$

- Jump conditions across the RDs (Alfvén wave)

$$S_M \begin{pmatrix} \rho_L^* \boldsymbol{v}_{tL}^{**} \\ \boldsymbol{B}_{tL}^{**} \end{pmatrix} - \begin{pmatrix} \rho_L^* \boldsymbol{v}_{tL}^{**} S_M - B_x \boldsymbol{B}_{tL}^{**} \\ \boldsymbol{B}_{tL}^{**} S_M - B_x \boldsymbol{v}_{tL}^{**} \end{pmatrix} = S_M \begin{pmatrix} \rho_R^* \boldsymbol{v}_{tR}^{**} \\ \boldsymbol{B}_{tR}^{**} \end{pmatrix} - \begin{pmatrix} \rho_R^* \boldsymbol{v}_{tR}^{**} S_M - B_x \boldsymbol{B}_{tR}^{**} \\ \boldsymbol{B}_{tR}^{**} S_M - B_x \boldsymbol{v}_{tR}^{**} \end{pmatrix}$$

$$\boldsymbol{v}_{tL}^{**} = \boldsymbol{v}_{tR}^{**} = \boldsymbol{v}_t^{**}, \boldsymbol{B}_{tL}^{**} = \boldsymbol{B}_{tR}^{**} = \boldsymbol{B}_t^{**} \quad \text{for } B_x \neq 0$$

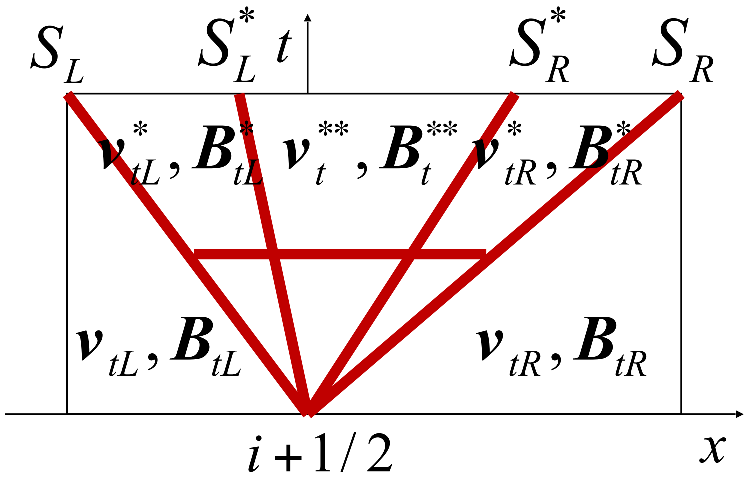
HLLD solver [13/24]



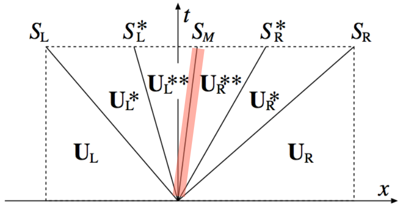
- Jump conditions across the entire intermediate states

$$\left(S_R - S_R^*\right)\begin{pmatrix}\rho_R^* \boldsymbol{v}_{tR}^* \\ \boldsymbol{B}_{tR}^*\end{pmatrix} + \left(S_R^* - S_M\right)\begin{pmatrix}\rho_R^* \boldsymbol{v}_t^{**} \\ \boldsymbol{B}_t^{**}\end{pmatrix} + \left(S_M - S_L^*\right)\begin{pmatrix}\rho_L^* \boldsymbol{v}_t^{**} \\ \boldsymbol{B}_t^{**}\end{pmatrix} + \left(S_L^* - S_{RL}\right)\begin{pmatrix}\rho_L^* \boldsymbol{v}_{tL}^* \\ \boldsymbol{B}_{tL}^*\end{pmatrix}$$

$$+ S_R\begin{pmatrix}\rho_R \boldsymbol{v}_{tR} \\ \boldsymbol{B}_{tR}\end{pmatrix} - S_L\begin{pmatrix}\rho_L \boldsymbol{v}_{tL} \\ \boldsymbol{B}_{tL}\end{pmatrix} + \begin{pmatrix}\rho_R \boldsymbol{v}_{tR} u_R - B_x \boldsymbol{B}_{tR} \\ \boldsymbol{B}_{tR} u_R - B_x \boldsymbol{v}_{tR}\end{pmatrix} - \begin{pmatrix}\rho_L \boldsymbol{v}_{tL} u_L - B_x \boldsymbol{B}_{tL} \\ \boldsymbol{B}_{tL} u_L - B_x \boldsymbol{v}_{tL}\end{pmatrix} = 0$$



HLLD solver [14/24]



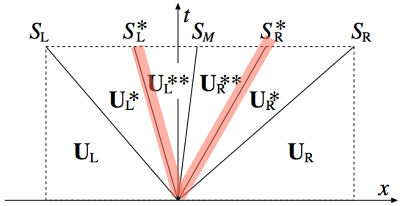
- Solution:

$$\boldsymbol{U}_\alpha^{**}$$

$$\rho_\alpha^{**} = \rho_\alpha^*$$

$$\begin{cases} \boldsymbol{v}_t^{**} = \dfrac{\sqrt{\rho_L^*}\boldsymbol{v}_{tL}^* + \sqrt{\rho_R^*}\boldsymbol{v}_{tR}^* + \left(\boldsymbol{B}_{tR}^* - \boldsymbol{B}_{tL}^*\right)\mathrm{sgn}(B_x)}{\sqrt{\rho_L^*} + \sqrt{\rho_R^*}} \\ \boldsymbol{B}_t^{**} = \dfrac{\sqrt{\rho_L^*}\boldsymbol{B}_{tR}^* + \sqrt{\rho_R^*}\boldsymbol{B}_{tL}^* + \sqrt{\rho_L^*\rho_R^*}\left(\boldsymbol{v}_{tR}^* - \boldsymbol{v}_{tL}^*\right)\mathrm{sgn}(B_x)}{\sqrt{\rho_L^*} + \sqrt{\rho_R^*}} \end{cases}$$

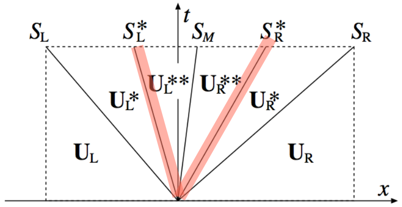
HLLD solver [15/24]



- Jump conditions across the RDs (Alfvén wave)

$$S_\alpha^* \begin{pmatrix} \rho_\alpha^{**} \\ \rho_\alpha^{**} S_M \\ \rho_\alpha^{**} v_\alpha^{**} \\ \rho_\alpha^{**} w_\alpha^{**} \\ B_{y\alpha}^{**} \\ B_{z\alpha}^{**} \\ e_\alpha^{**} \end{pmatrix} - \begin{pmatrix} \rho_\alpha^{**} S_M \\ \rho_\alpha^{**} S_M^2 + p_T^* - B_x^2 \\ \rho_\alpha^{**} v_\alpha^{**} S_M - B_x B_{y\alpha}^{**} \\ \rho_\alpha^{**} w_\alpha^{**} S_M - B_x B_{z\alpha}^{**} \\ B_{y\alpha}^{**} S_M - B_x v_\alpha^{**} \\ B_{z\alpha}^{**} S_M - B_x w_\alpha^{**} \\ (e_\alpha^{**} + p_T^*) S_M - B_x (\boldsymbol{v}_\alpha^{**} \cdot \boldsymbol{B}_\alpha^{**}) \end{pmatrix} = S_\alpha^* \begin{pmatrix} \rho_\alpha^* \\ \rho_\alpha^* S_M \\ \rho_\alpha^* v_\alpha^* \\ \rho_\alpha^* w_\alpha^* \\ B_{y\alpha}^* \\ B_{z\alpha}^* \\ e_\alpha^* \end{pmatrix} - \begin{pmatrix} \rho_\alpha^* S_M \\ \rho_\alpha^* S_M^2 + p_T^* - B_x^2 \\ \rho_\alpha^* v_\alpha^* S_M - B_x B_{y\alpha}^* \\ \rho_\alpha^* w_\alpha^* S_M - B_x B_{z\alpha}^* \\ B_{y\alpha}^* S_M - B_x v_\alpha^* \\ B_{z\alpha}^* S_M - B_x w_\alpha^* \\ (e_\alpha^* + p_T^*) S_M - B_x (\boldsymbol{v}_\alpha^* \cdot \boldsymbol{B}_\alpha^*) \end{pmatrix}$$

HLLD solver [16/24]



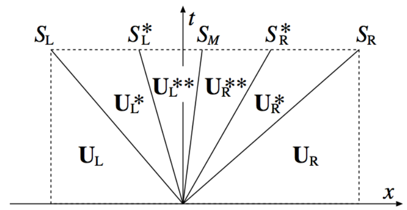
- Solution: U_α^{**}

$$\rho_\alpha^{**} = \rho_\alpha^*$$

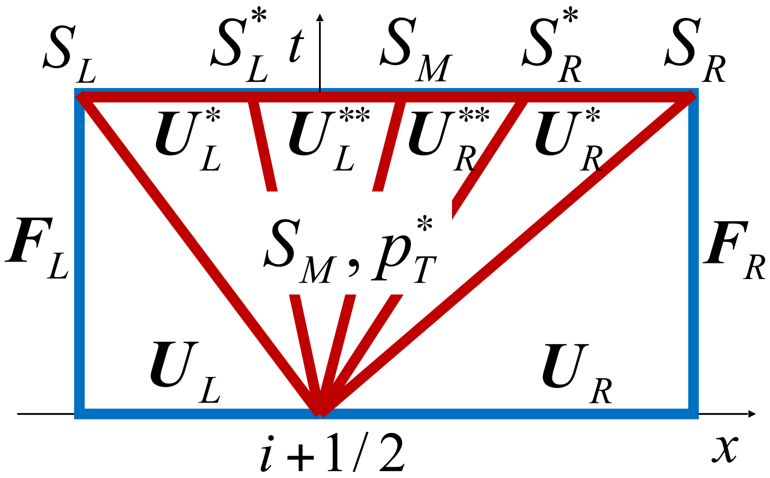
$$\begin{cases} \boldsymbol{v}_t^{**} = \dfrac{\sqrt{\rho_L^*}\boldsymbol{v}_{tL}^* + \sqrt{\rho_R^*}\boldsymbol{v}_{tR}^* + \left(\boldsymbol{B}_{tR}^* - \boldsymbol{B}_{tL}^*\right)\mathrm{sgn}(B_x)}{\sqrt{\rho_L^*} + \sqrt{\rho_R^*}} \\ \boldsymbol{B}_t^{**} = \dfrac{\sqrt{\rho_L^*}\boldsymbol{B}_{tR}^* + \sqrt{\rho_R^*}\boldsymbol{B}_{tL}^* + \sqrt{\rho_L^*\rho_R^*}\left(\boldsymbol{v}_{tR}^* - \boldsymbol{v}_{tL}^*\right)\mathrm{sgn}(B_x)}{\sqrt{\rho_L^*} + \sqrt{\rho_R^*}} \end{cases}$$

$$e_\alpha^{**} = e_\alpha^* \mp \sqrt{\rho_\alpha^*}\left(\boldsymbol{v}_\alpha^* \cdot \boldsymbol{B}_\alpha^* - \boldsymbol{v}^{**} \cdot \boldsymbol{B}^{**}\right)\mathrm{sgn}(B_x) \qquad (- : R, + : L)$$

HLLD solver [17/24]



- 5-wave approximation



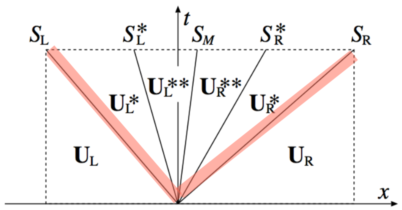
$S_{R,L}$: fast waves

S_M : entropy wave

$S_{R,L}^*$: Alfvén waves

$$S_{R,L}\left(\boldsymbol{U}_{R,L}^* - \boldsymbol{U}_{R,L}\right) = \boldsymbol{F}_{R,L}^* - \boldsymbol{F}_{R,L},\ S_{R,L}^*\left(\boldsymbol{U}_{R,L}^{**} - \boldsymbol{U}_{R,L}^*\right) = \boldsymbol{F}_{R,L}^{**} - \boldsymbol{F}_{R,L}^*,$$

$$S_M\left(\boldsymbol{U}_R^{**} - \boldsymbol{U}_L^{**}\right) = \boldsymbol{F}_R^{**} - \boldsymbol{F}_L^{**},\ \frac{1}{\Delta t}\int_{S_L\Delta t}^{S_R\Delta t} \boldsymbol{U}\left(x, t^{n+1}\right)dx + S_R\boldsymbol{U}_R - S_L\boldsymbol{U}_L + \boldsymbol{F}_R - \boldsymbol{F}_L = 0$$

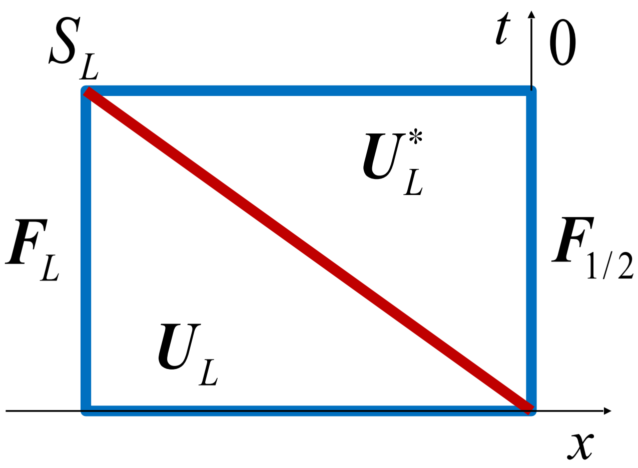


HLLD solver [18/24]

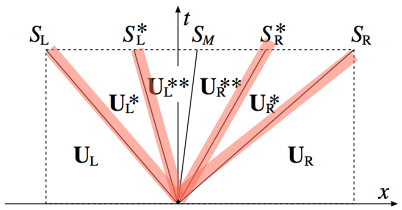
- Numerical flux

$$F_{1/2} = F_L \quad \text{if} \;\; S_L \geq 0$$

$$F_{1/2} = F_L + S_L U_L^* - S_L U_L = F_L^* \quad \text{if} \;\; S_L \leq 0 \leq S_L^*$$



HLLD solver [19/24]



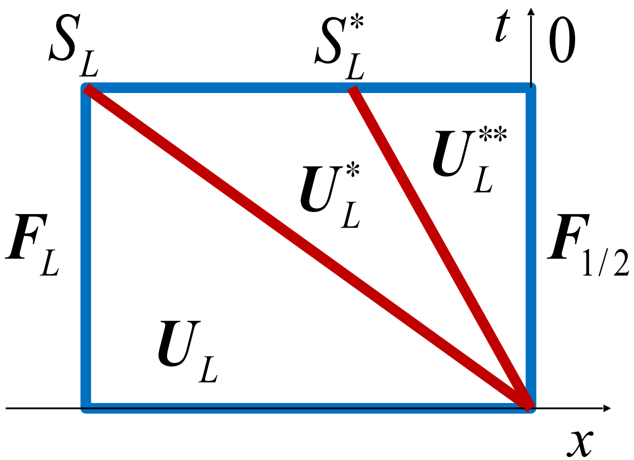
- Numerical flux

$$F_{1/2} = F_L \quad \text{if} \quad S_L \geq 0$$

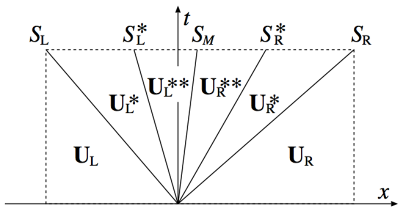
$$F_{1/2} = F_L + S_L U_L^* - S_L U_L = F_L^* \quad \text{if} \quad S_L \leq 0 \leq S_L^*$$

$$F_{1/2} = F_L + S_L^* U_L^{**} - \left(S_L^* - S_L\right) U_L^* - S_L U_L$$

$$= F_L^* + S_L^* U_L^{**} - S_L^* U_L^* = F_L^{**} \quad \text{if} \quad S_L^* \leq 0 \leq S_M$$



HLLD solver [20/24]



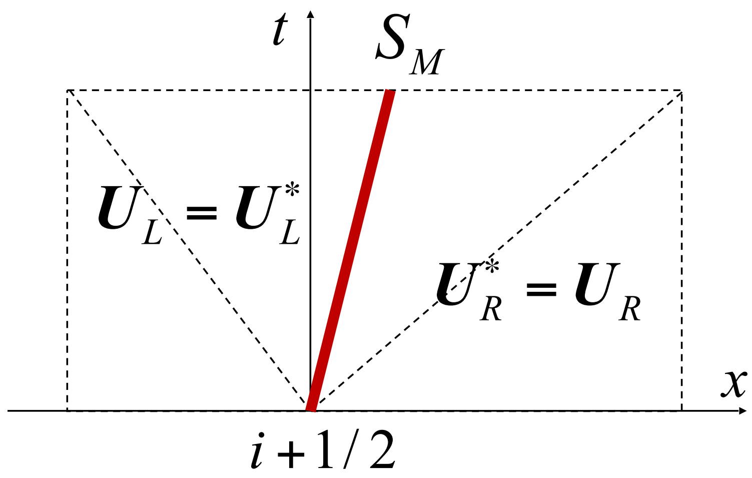
- Numerical flux

$$\boldsymbol{F}_{1/2} = \begin{cases} \boldsymbol{F}_L & \text{if } S_L \ge 0 \\ \boldsymbol{F}_L^* & \text{if } S_L \le 0 \le S_L^* \\ \boldsymbol{F}_L^{**} & \text{if } S_L^* \le 0 \le S_M \\ \boldsymbol{F}_R^{**} & \text{if } S_M \le 0 \le S_R^* \\ \boldsymbol{F}_R^* & \text{if } S_R^* \le 0 \le S_R \\ \boldsymbol{F}_R & \text{if } S_R \le 0 \end{cases}$$

$$\boldsymbol{F}_\alpha^{*/**} = \boldsymbol{F}\left(\rho_\alpha^{*/**}, S_M, \boldsymbol{v}_{t\alpha}^{*/**}, B_x, \boldsymbol{B}_{t\alpha}^{*/**}, e_\alpha^{*/**}, p_T^*\right)$$

HLLD solver [21/24]: HLLD can deal with..

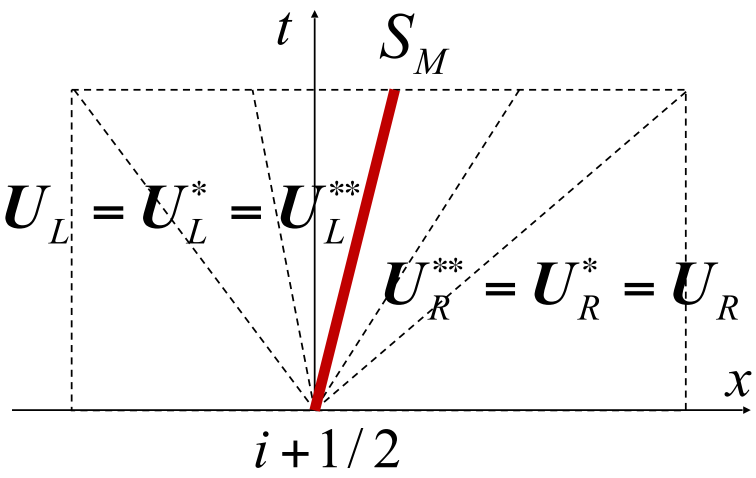
- An isolated tangential discontinuity (TD)



$$S_M = u, B_x = 0$$
$$S_M[\boldsymbol{U}] = [\boldsymbol{F}]$$

HLLD solver [22/24]: HLLD can deal with..

- An isolated tangential discontinuity (TD)
- An isolated contact discontinuity (CD)

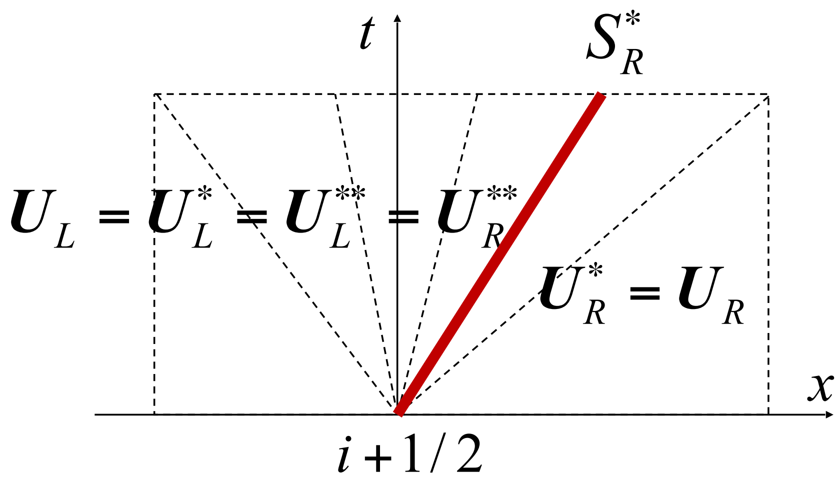


$$S_M = u, B_x \neq 0$$

$$S_M[\boldsymbol{U}] = [\boldsymbol{F}]$$

HLLD solver [23/24]: HLLD can deal with..

- An isolated tangential discontinuity (TD)
- An isolated contact discontinuity (CD)
- An isolated rotational discontinuity (RD)

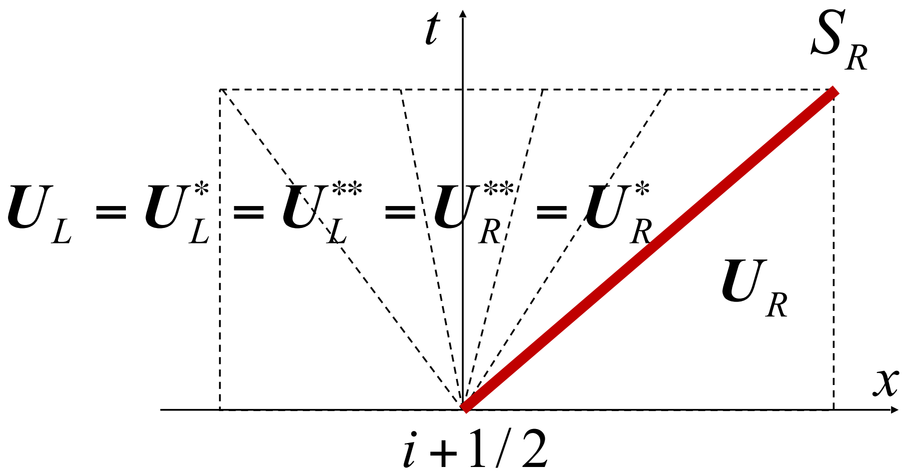


$$S_R^* = u + \frac{B_x}{\sqrt{\rho}}, B_x > 0$$

$$S_R^*[\boldsymbol{U}] = [\boldsymbol{F}]$$

HLLD solver [24/24]: HLLD can deal with..

- An isolated tangential discontinuity (TD)
- An isolated contact discontinuity (CD)
- An isolated rotational discontinuity (RD)
- An isolated fast shock (FS)

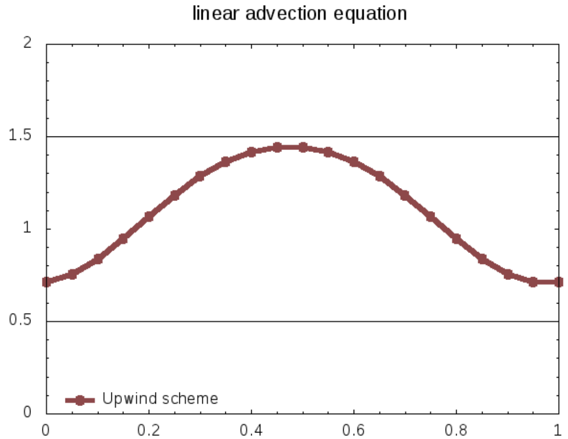


$$S_R[\boldsymbol{U}] = [\boldsymbol{F}]$$

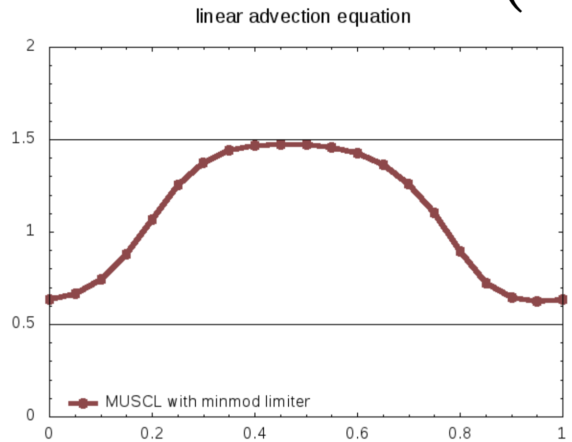
Comparisons of slope limiters (movie)

Courtesy of Dr. Miyoshi (Hiroshima University)

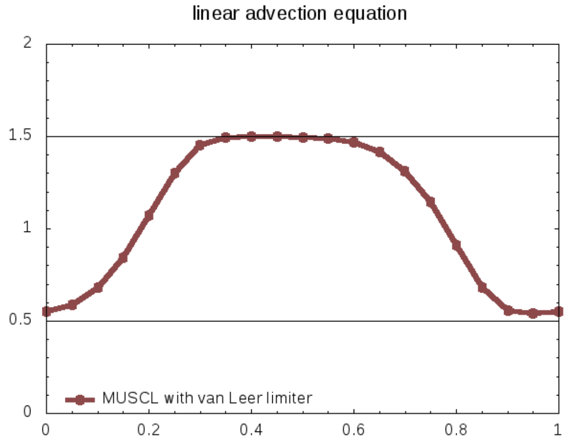
$(\nu = 0.5)$



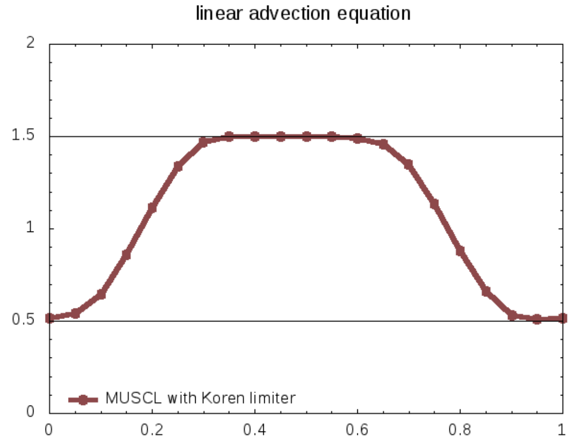
Upwind



MUSCL (minmod)



MUSCL (van Leer)

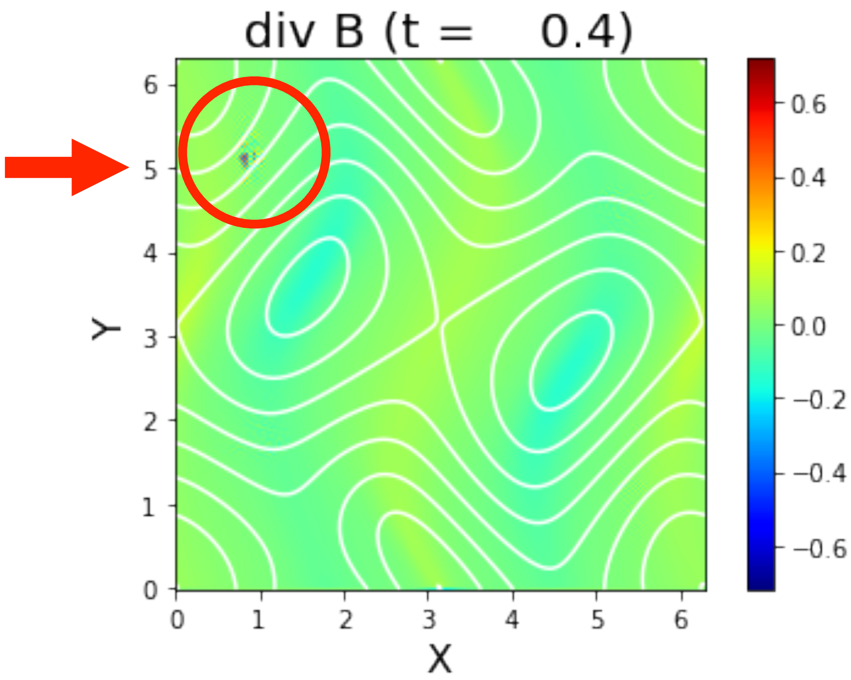


MUSCL (Koren)

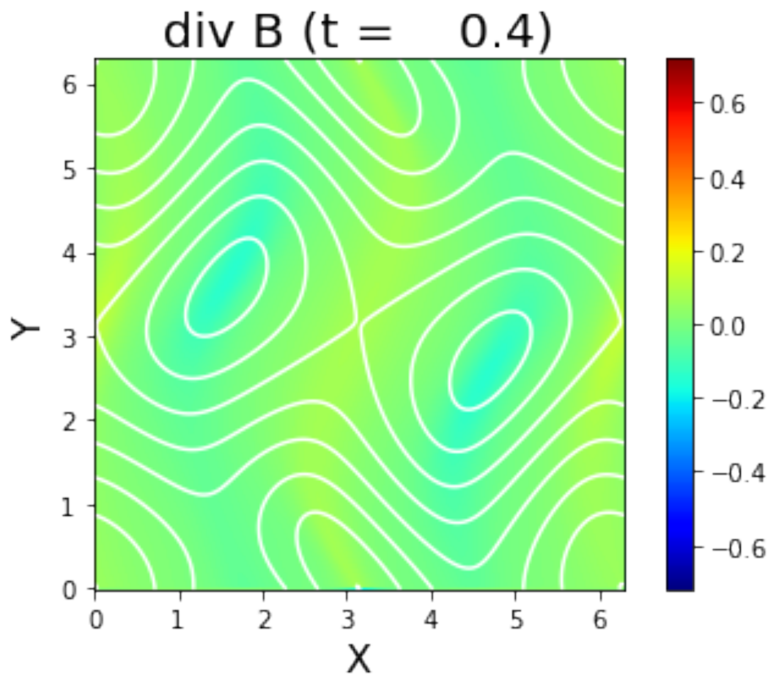
Divergence B

- The oscillation is related to divergence B
- With adequate correction, we can proceed

Without correction



With correction



Hyperbolic divergence cleaning (1/3)

- Additional equation & term for a new variable ψ

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

$$\frac{\partial \rho \boldsymbol{v}}{\partial t} + \nabla \cdot (\rho \boldsymbol{v}\boldsymbol{v} + p_t \mathbb{I} - \boldsymbol{B}\boldsymbol{B}) = 0$$

$$\frac{\partial \mathcal{E}}{\partial t} + \nabla \cdot \Big((\mathcal{E} + p_t)\boldsymbol{v} - (\boldsymbol{v} \cdot \boldsymbol{B})\boldsymbol{B} \Big) = 0$$

$$\frac{\partial \boldsymbol{B}}{\partial t} + \nabla \cdot (\boldsymbol{v}\boldsymbol{B} - \boldsymbol{B}\boldsymbol{v}) + \nabla \psi = 0$$

$$\frac{\partial \psi}{\partial t} + c_h^2 \nabla \cdot \boldsymbol{B} = -\left(\frac{c_h^2}{c_p^2}\right)\psi$$

Many different names:
1. Virtual potential, 2. Divergence-cleaning potential, 3. GLM parameter

- Equation system is sometimes called GLM (Generalized Lagrangian multiplier) MHD

Dedner+ 2002 JCP

Hyperbolic divergence cleaning (2/3)

- GLM terms

$$\frac{\partial \boldsymbol{B}}{\partial t} + \nabla \times (\boldsymbol{v} \times \boldsymbol{B}) + \nabla \psi = 0,$$

$$\frac{\partial \psi}{\partial t} + c_h^2 \nabla \cdot \boldsymbol{B} = -\left(\frac{c_h^2}{c_p^2}\right)\psi,$$

- Some math

$$\frac{\partial (\nabla \cdot \boldsymbol{B})}{\partial t} + \cancel{\nabla \cdot [\nabla \times (\boldsymbol{v} \times \boldsymbol{B})]} + \Delta \psi = 0,$$

$$\frac{\partial^2 \psi}{\partial t^2} + c_h^2 \nabla \cdot \left(\frac{\partial \boldsymbol{B}}{\partial t}\right) = -\left(\frac{c_h^2}{c_p^2}\right)\frac{\partial \psi}{\partial t},$$

- We obtain two telegraph equations

$$\frac{\partial^2 (\nabla \cdot \boldsymbol{B})}{\partial t^2} + \left(\frac{c_h^2}{c_p^2}\right)\frac{\partial (\nabla \cdot \boldsymbol{B})}{\partial t} - c_h^2 \Delta (\nabla \cdot \boldsymbol{B}) = 0,$$

$$\frac{\partial^2 \psi}{\partial t^2} + \left(\frac{c_h^2}{c_p^2}\right)\frac{\partial \psi}{\partial t} - c_h^2 \Delta \psi = 0,$$

- Telegraph equations scatter and dump the variables

Hyperbolic divergence cleaning (3/3)

- GLM terms in 1-D Riemann problem

$$\frac{\partial}{\partial y} = \frac{\partial}{\partial z} = 0$$

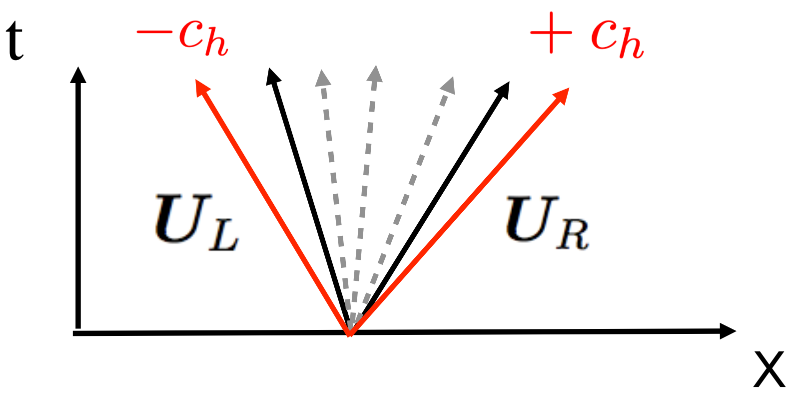
↓ Wave equation

$$\frac{\partial}{\partial t}\begin{pmatrix} B_x \\ \psi \end{pmatrix} + \begin{pmatrix} 0 & 1 \\ c_h^2 & 0 \end{pmatrix} \frac{\partial}{\partial x}\begin{pmatrix} B_x \\ \psi \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

$$\frac{\partial \psi}{\partial t} = -\left(\frac{c_h^2}{c_p^2}\right)\psi$$

← Exponential decay

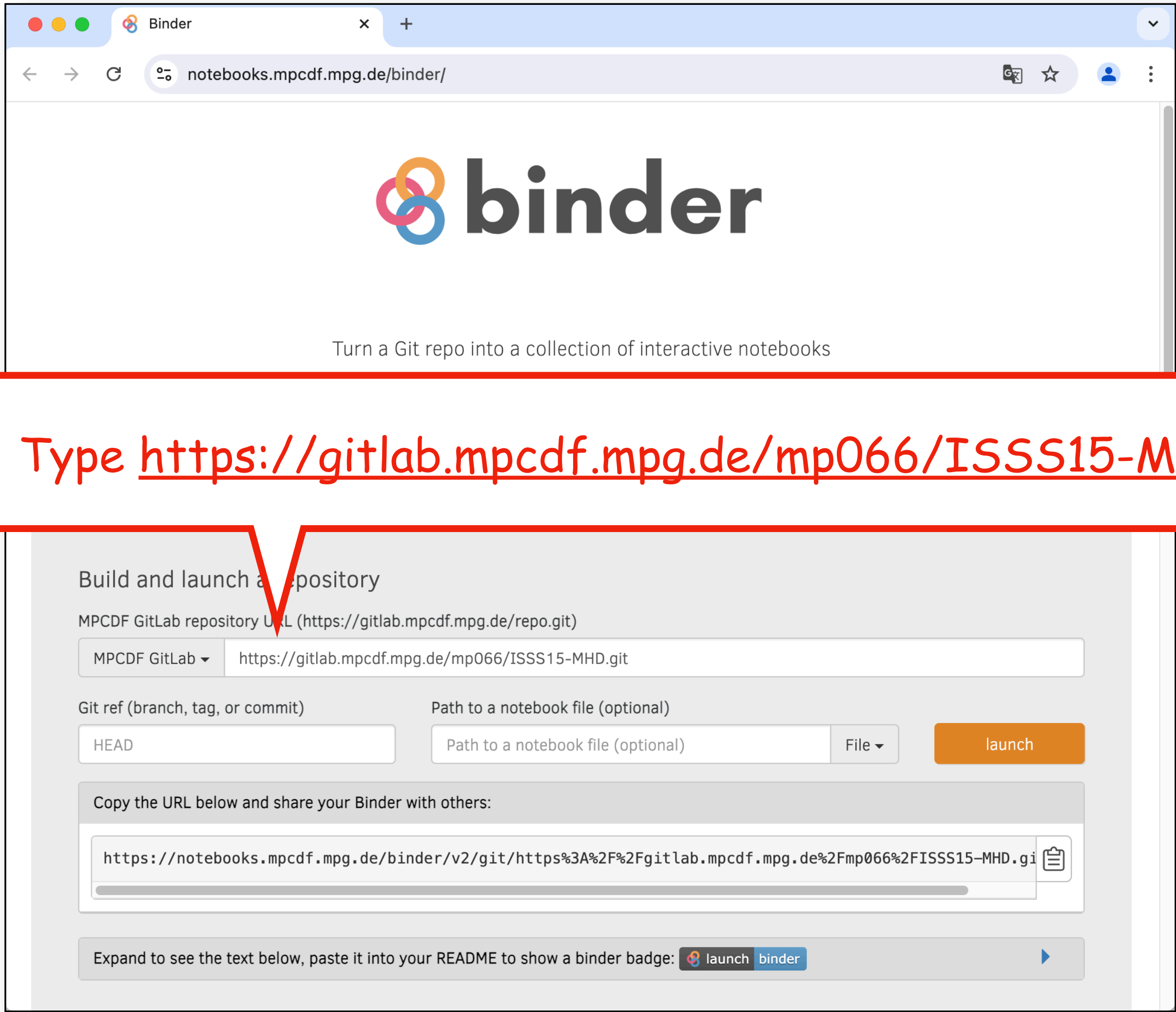
- Riemann problem can be modified to



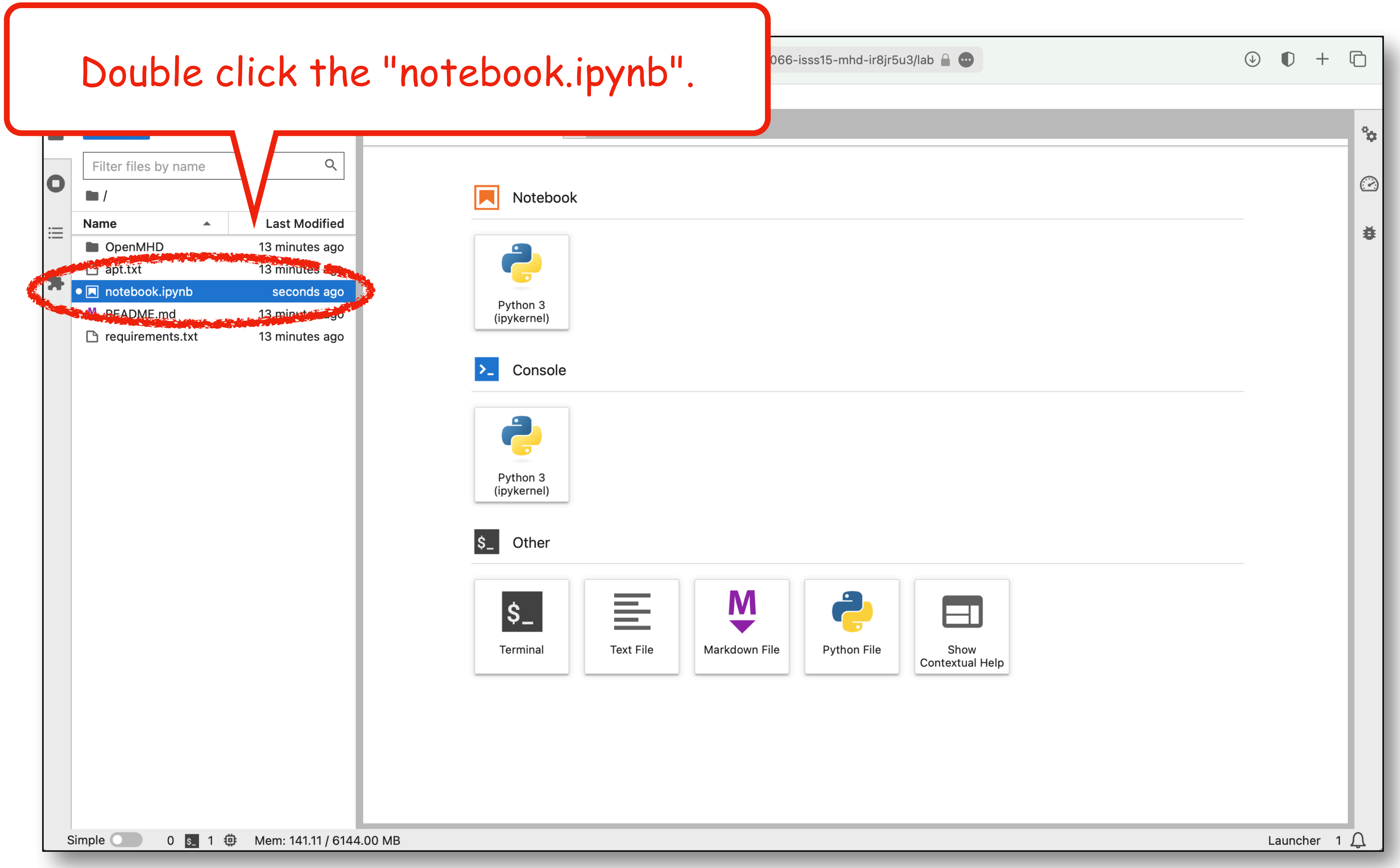
- Typically, c_h is set to the fastest c_f in the simulation domain
- (c_p^2/c_h) = 0.18 is the optimum, according to numerical tests (Dedner+ 2002)

Binder

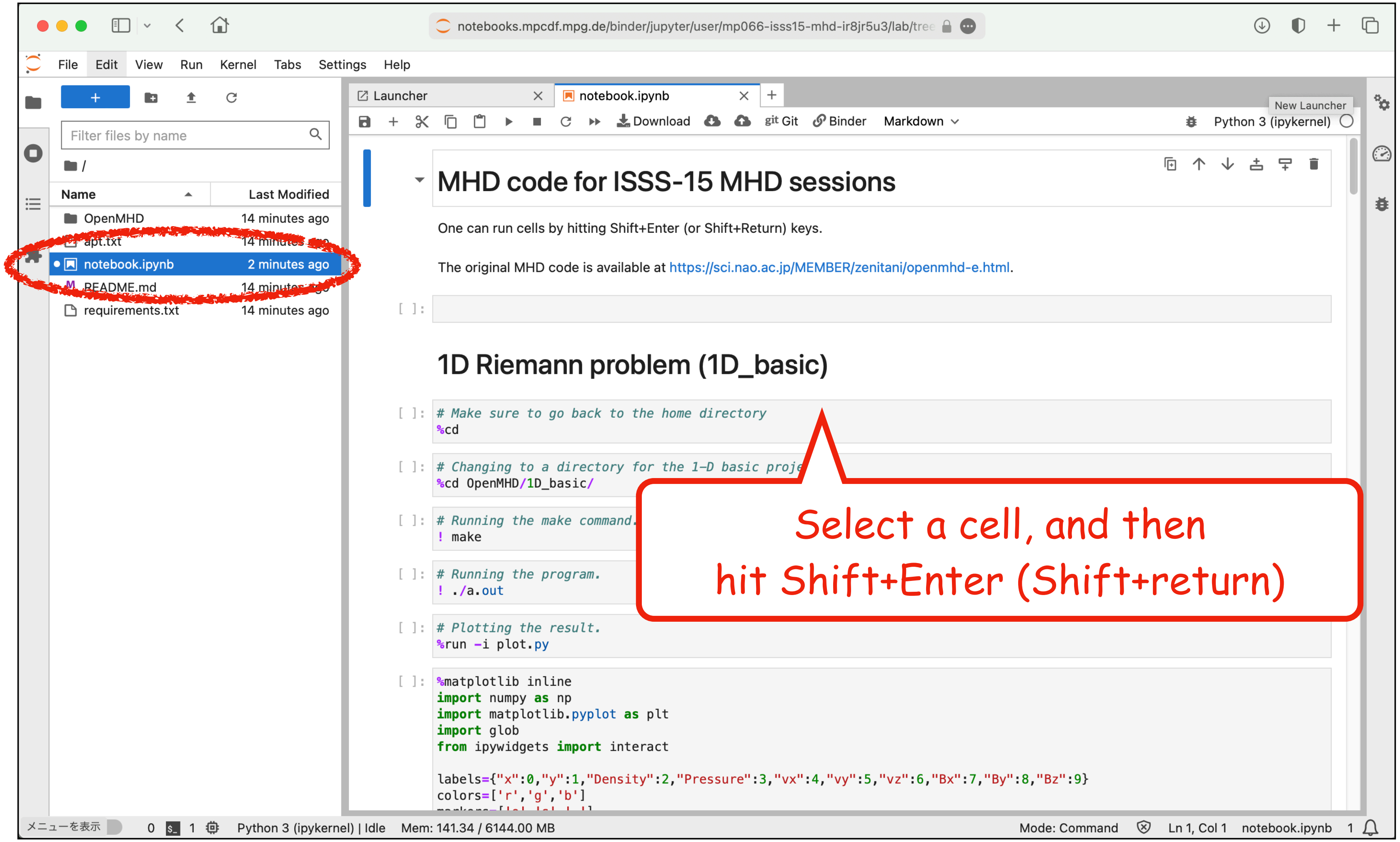
Type https://gitlab.mpcdf.mpg.de/mp066/ISSS15-MHD.git



Using Binder (1/5) - Start page



Using Binder (2/5) - Jupyter notebook

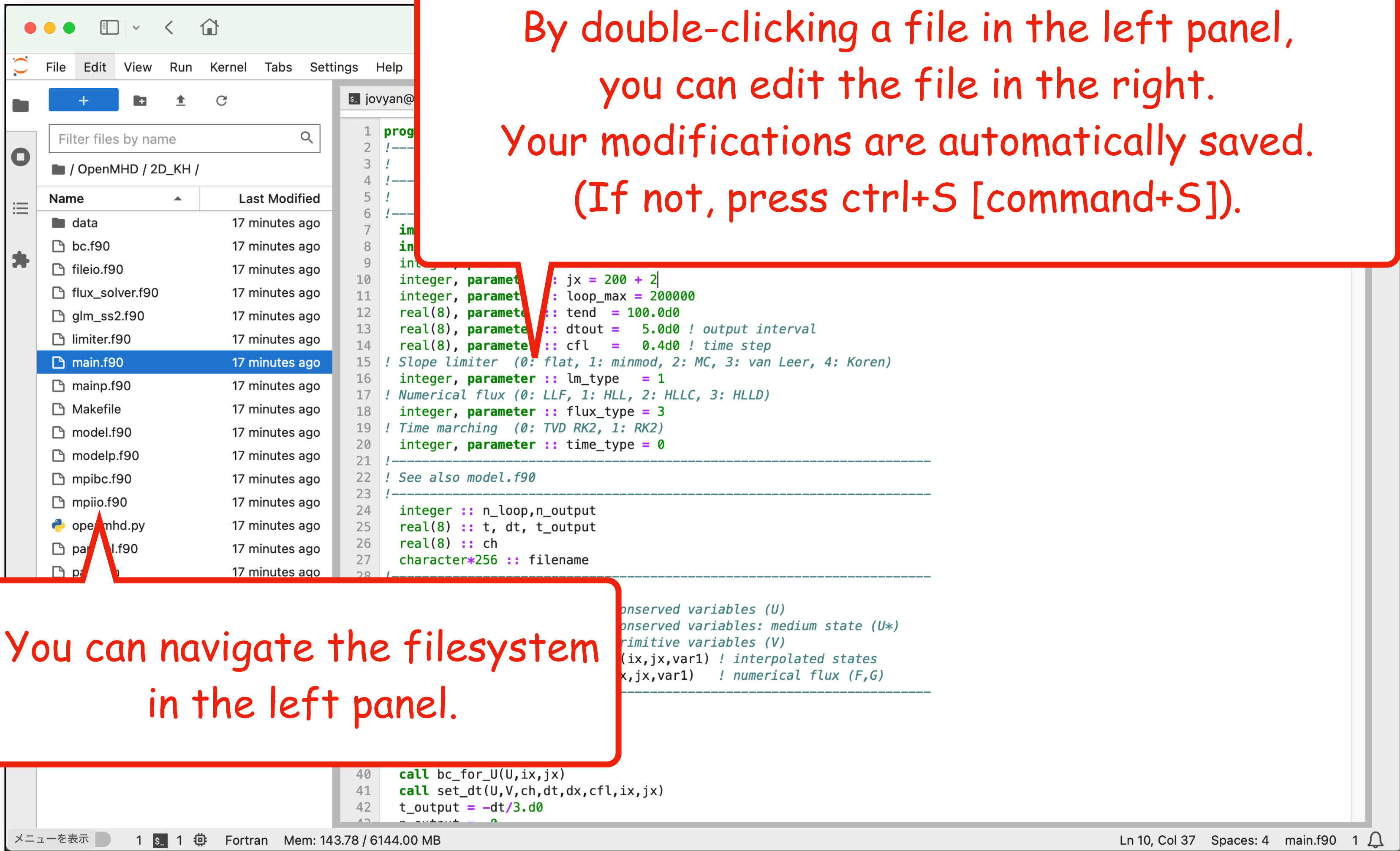


Some more on Jupyter

- Shift + enter (return) will run the command in a cell
- 1) Python codes
- 2) Magic commands
  - %cd # Changing the directory
  - %run -i plot.py # Run the python file
  - %%time # Measure the execution time of the cell
- 3) UNIX shell commands
  - ! pwd
  - ! ./a.out # Use "./" to explicitly specify your program

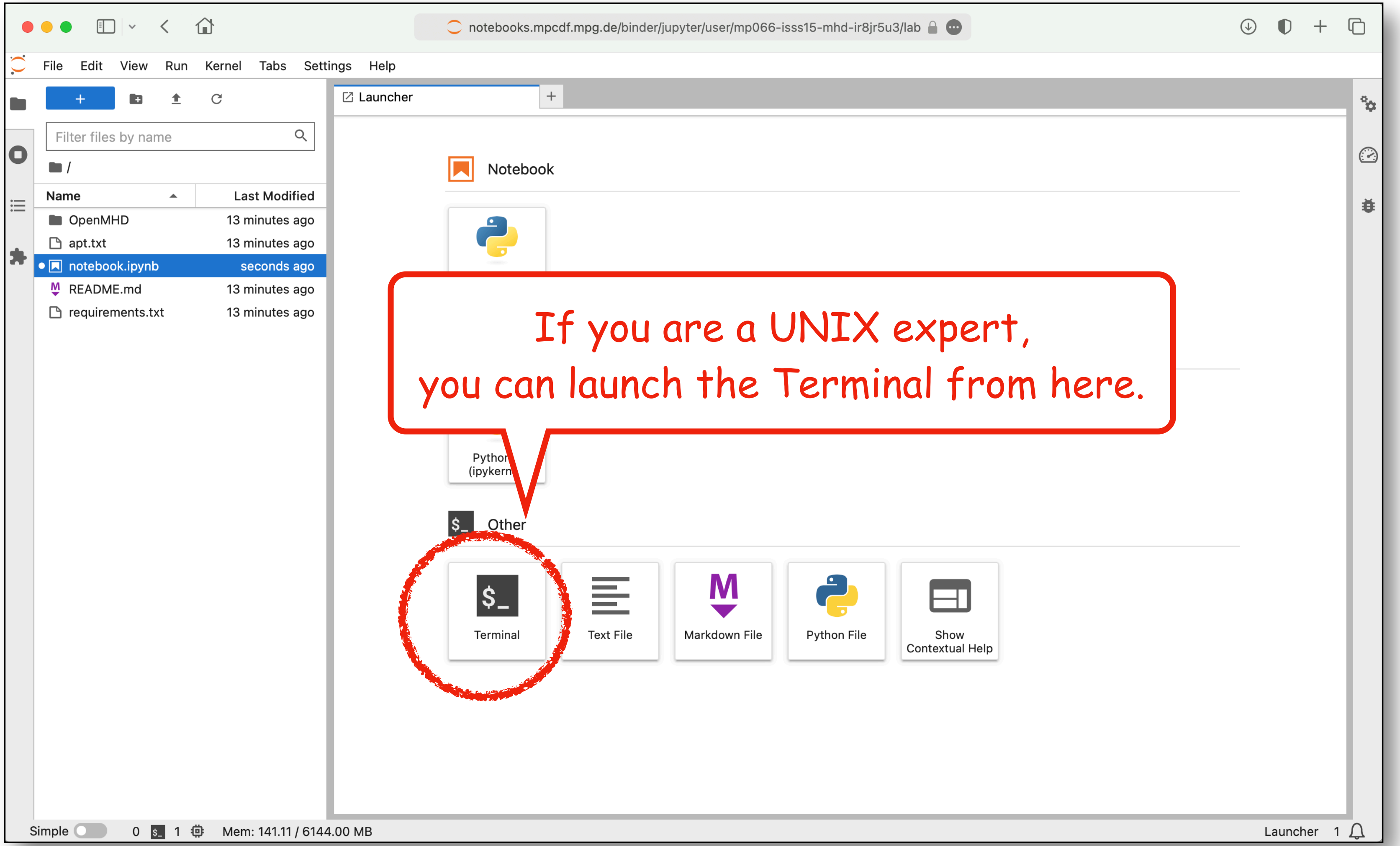
Using Binder (3/5) - File editor

By double-clicking a file in the left panel,
you can edit the file in the right.
Your modifications are automatically saved.
(If not, press ctrl+S [command+S]).

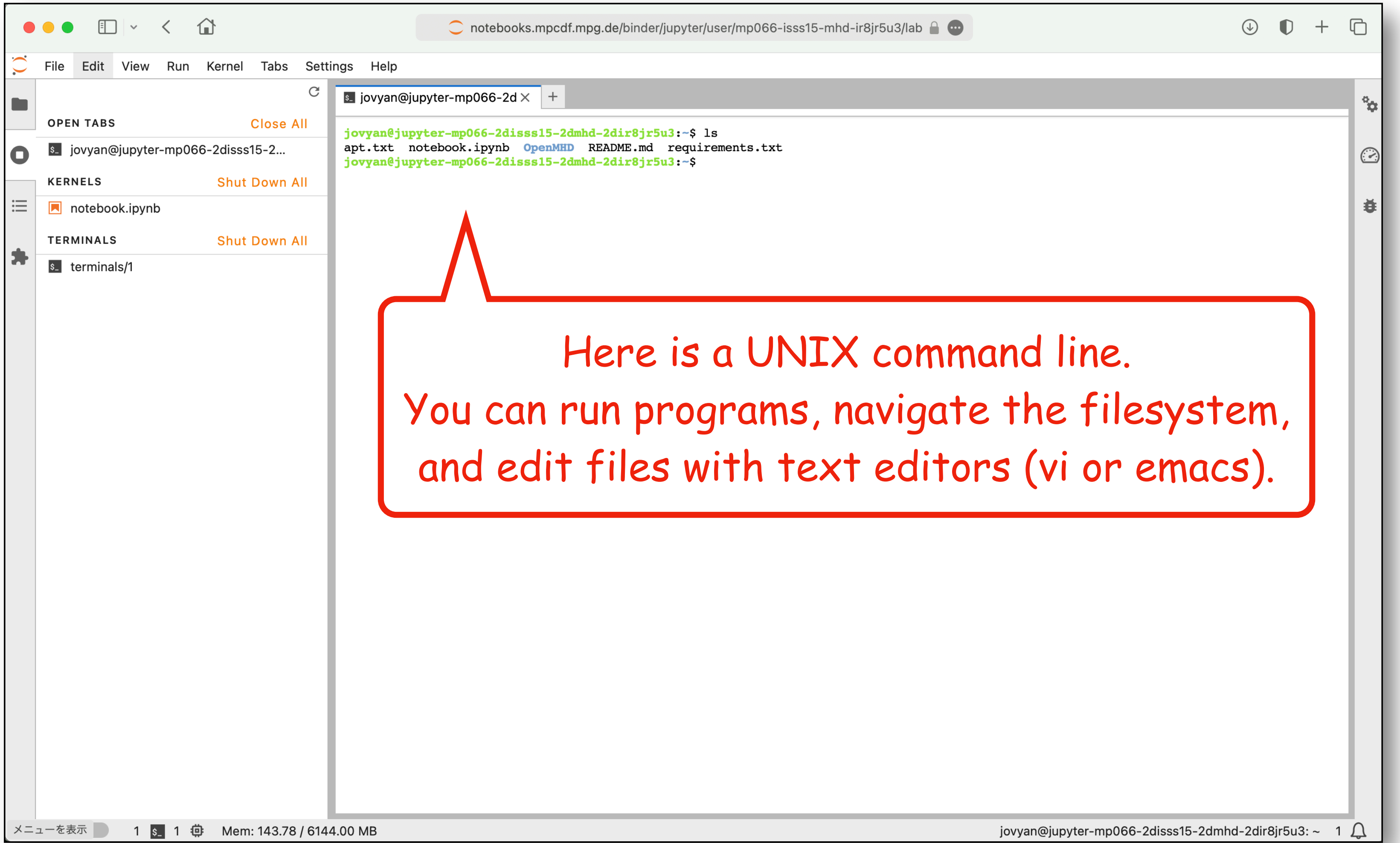


You can navigate the filesystem
in the left panel.

Using Binder (4/5) - Terminal



Using Binder (5/5) - Terminal



Using your own computer (1/3)

- You need:
  - Fortran compiler, (MPI), python, git, make, etc.
- Downloading OpenMHD from the website
  - $ tar zxvf openmhd-20240130.tar.gz.tar.gz
  - $ cd openmhd-20240130/1D_basic/
- Obtaining OpenMHD from GitHub
  - $ git clone https://github.com/zenitani/OpenMHD.git
  - $ cd OpenMHD/1D_basic/

Using your own computer (2/3)

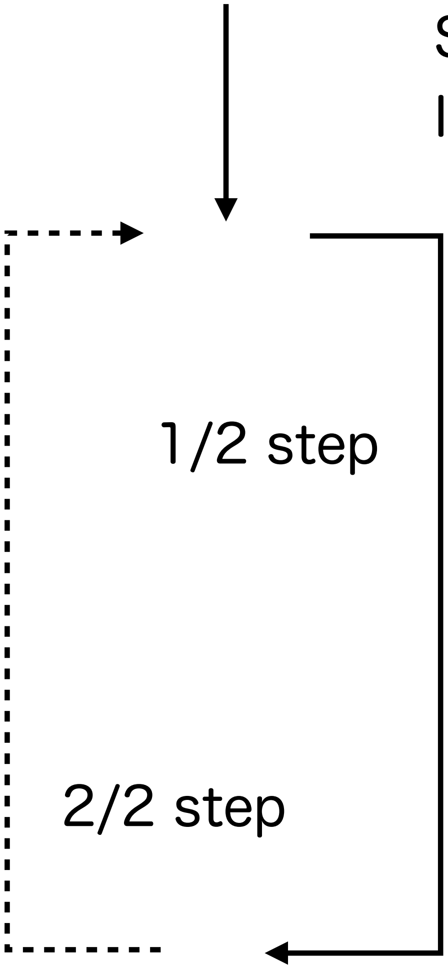
- Edit the Makefile
    - set the F90 variable to your compiler command
- Then compile the source code according to the Makefile
    - $ make # compiling the serial and parallel codes
    - $ make run # compiling the serial code
- Running the program
    - $ ./a.out # Use "./" to explicitly specify your program
- Deleting object files and data files
    - $ make clean

Using your own computer (3/3) - Visualization

- Python (iPython)
  - $ ipython3 --pylab

    In[1]: %run plot.py
- Python (Jupyter notebook)
  - $ jupyter-notebook plot.ipynb
  - You can run the code by hitting Shift + Return (Enter) key
- Installing python libraries
  - $ pip3 install ipython matplotlib
  - $ pip3 install jupyter ipywidgets
  - (If you have a problem with ipywidgets 8, try ipywidgets 7.7.1 instead.)

Appendix: Logical flow of the program



Simulation settings: ix, jx (main.f90)

Initial configuration (model.f90)

- Convert U to V (u2v.f90)
  - Output the data file (output.f90)
  - Calculate the timestep Δt (set_dt.f90)

- Interpolate U and V in the X direction (limiter.f90)
- Boundary condition for interpolated values
- Calculate numerical flux F in X (flux_solver.f90)
- Interpolate U and V in the Y direction (limiter.f90)
- Boundary condition for interpolated values
- Calculate numerical flux G in Y (flux_solver.f90)

- Advance U by using F and G (rk.f90)
- Boundary condition for U

Appendix: Reading data in Python 3

data files

data/fields-00000.dat
fields-00001.dat
...
fields-00100.dat

openmhd.data_read()

3-D array data[ix,jx,9] contains the following nine variables

$$\rho,\ \boldsymbol{v},\ p,\ \boldsymbol{B}\quad \psi$$

data[ix,jx,vx] stands for Vx
data[ix,jx,bx] stands for Bx
...

```
import matplotlib.pyplot as plt
import numpy as np
import openmhd
# dummy index
vx=0;vy=1;vz=2;pr=3;ro=4;bx=5;by=6;bz=7;ps=8

# reading the data ...
x,y,t,data = openmhd.data_read("data/field-00020.dat")
# reading the data (partial domain: [ix1,ix2] x [jx1,jx2])
# x,y,t,data = openmhd.data_read("data/field-00020.dat", ix1=0, ix2=100, jx1=11)

# clearing the current figure, if any
plt.clf()
# extent: [left, right, bottom, top]
extent=[x[0],x[-1],y[0],y[-1]]
# 2D plot (vmin/mymin: minimum value, vmax/mymax: max value)
# Note: ().T is necessary for 2-D plot routines (imshow/pcolormesh...)
tmp = np.ndarray((x.size,y.size),np.double)
tmp[:,:] = data[:,:,pr]
```