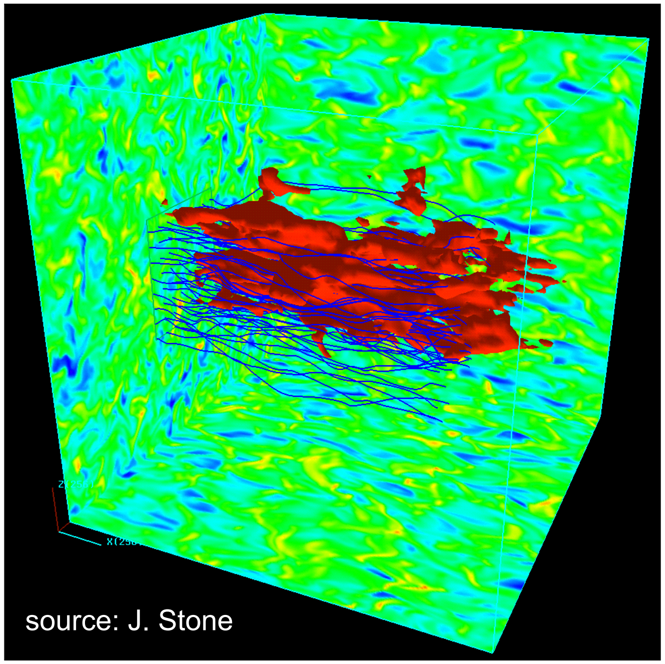
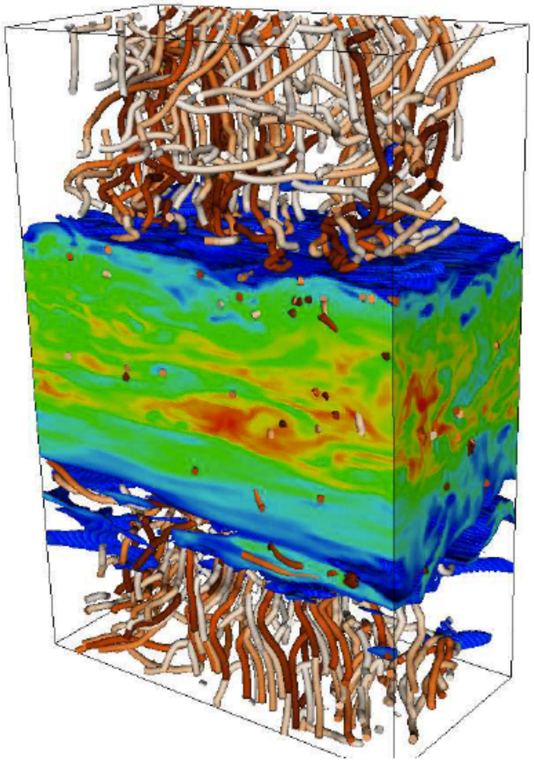
Astronomy 253, Plasma Astrophysics, Harvard University

Computational Magnetohydrodynamics


source: J. Stone



Instructor: Xuening Bai

April 11, 2016

Why computational MHD?

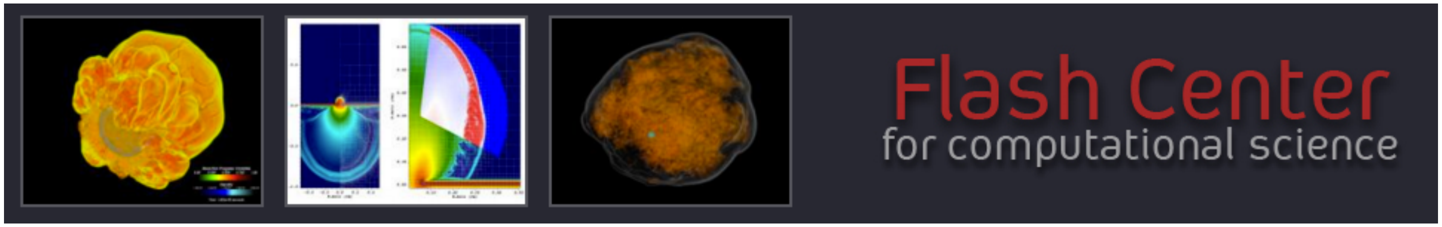
- Astrophysical plasma phenomena are typically highly-nonlinear, and in multi-D: need reliable numerical algorithms to solve MHD equations.

  Computational science has constantly been growing with progress in computer hardware, numerical analysis, software engineering.

  High-performance computing has become standard practice in science.

- MHD provides accurate description of collisional plasmas.
- MHD is often reasonably OK to describe the large-scale phenomenon (even) for collisionless plasmas.
- Computational MHD has been playing a major role in nearly all subfields of astrophysics from star/planet formation to cosmology.

Codes, and Jargons



PLUTO

A modular code for computational astrophysics

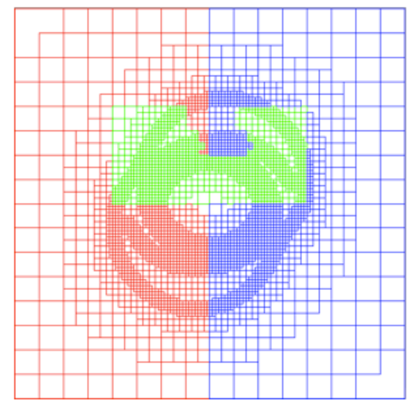
UNIVERSITÀ DEGLI STUDI DI TORINO
ALMA UNIVERSITAS TAURINENSIS

The Pencil Code

a high-order finite-difference code for compressible MHD

The RAMSES code

Adaptive Mesh Refinement for self-gravitating magnetized fluid flows



NIRVANA code

Godunov Code
Finite volume/difference
Riemann solvers
PPM reconstruction
CT algorithm
…

Outline

- Hyperbolic PDEs and conservation laws
- Solving the linear advection equation
- Finite volume (Godunov) methods
- Preserving divergence-free B field
- Code tests
- Adding source terms
- Other methods for solving MHD equations

Useful reference: *Finite Volume Methods for Hyperbolic Problems*, LeVeque, 2002, Cambridge University Press.

Types of PDEs

Hydro and MHD equations are a system of partial differential equations (PDEs).

There are in general 3 types of PDEs. For a 2nd order PDE of the form

$$a\partial_{xx}^2 u + b\partial_{xy}^2 u + c\partial_{yy}^2 u + d\partial_x u + e\partial_y u + fu = g \; ,$$

it can be categorized based on the discriminant:

$$b^2 - 4ac \begin{cases} < 0 & \rightarrow \quad \text{elliptic,} \\ = 0 & \rightarrow \quad \text{parabolic,} \\ > 0 & \rightarrow \quad \text{hyperbolic,} \end{cases}$$

Hydro/ideal MHD equations are hyperbolic PDEs, but source terms (resistivity/viscosity/self-gravity) can be of other types.

Types of PDEs

Prototype of elliptic PDE:

Poisson equation: $$\nabla^2 u = f$$ (self-gravity)

Prototype of parabolic PDE:

Diffusion equation: $$\partial_t u = D\partial^2_{xx} u$$ (viscosity, resistivity, heat conduction)

Prototype of hyperbolic PDE:

Wave equation: $$\partial^2_{tt} u - c^2 \partial^2_{xx} u = 0$$

Linear advection equation: $$\partial_t u + A\partial_x u = 0$$

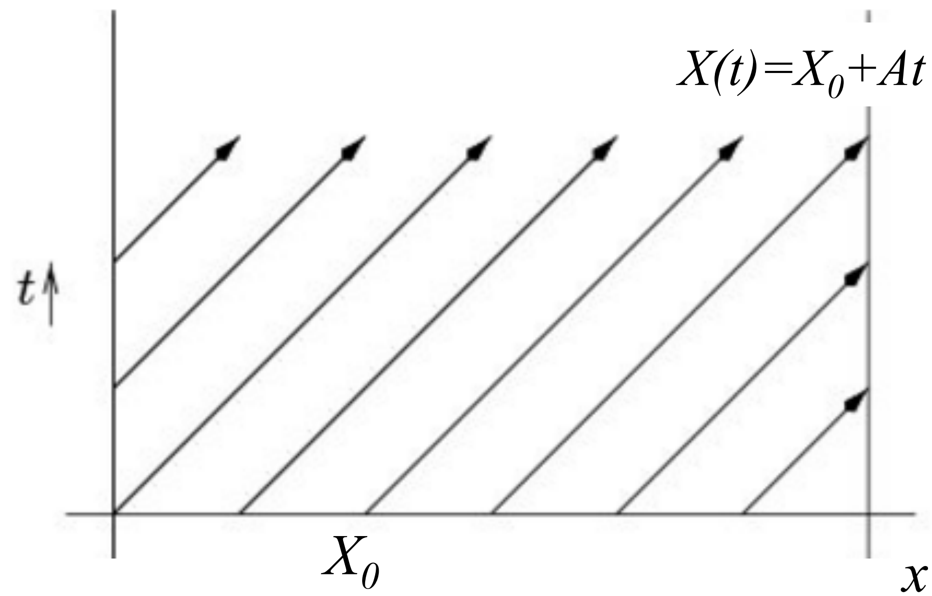
The linear advection equation

Consider linear advection eqs with constant A: $\partial_t u + A\partial_x u = 0$

Solution: $u(x,t) = u_0(x - At)$

The solution is constant along the ray (called the characteristic curve):

$$X(t) = X_0 + At$$



Proof:

$$\frac{d}{dt}u(X(t),t) = \partial_t u(X(t),t) + X'(t)\partial_x u(X(t),t)$$

$$= \partial_t u(X,t) + A\partial_x u(X,t) = 0$$

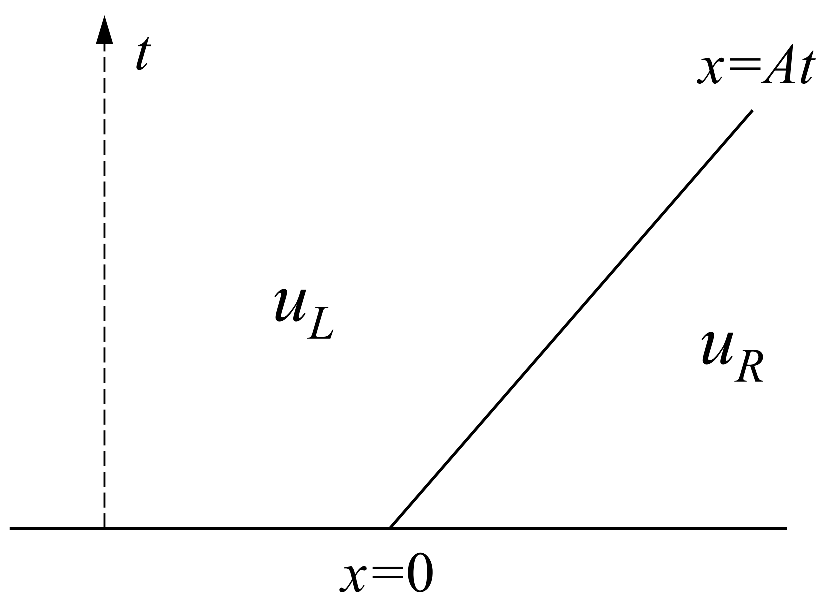
The Riemann problem (for linear advection eq)

$$\partial_t u + A\partial_x u = 0$$

Initial condition: $u = u_L, \ (x<0)$

$u = u_R, \ (x \geq 0)$

Result: discontinuity propagates along the characteristic curve.



Hyperbolicity of linear systems

A linear system of the form

$$\partial_t \boldsymbol{u} + \mathbf{A} \cdot \partial_x \boldsymbol{u} = 0$$

is hyperbolic if matrix **A** is diagonalizable with real eigenvalues.

Let us denote the eigenvalues by $\lambda^1 \leq \lambda^2 \leq ... \leq \lambda^m$

The matrix is diagonalizable if there is a complete set of eigenvectors such that

$$A\boldsymbol{r}^p = \lambda^p \boldsymbol{r}^p$$

The right-eigenvectors jointly form a matrix R, so that

$$R^{-1}AR = \Lambda = \mathrm{diag}(\lambda^1, \lambda^2, ..., \lambda^m)$$

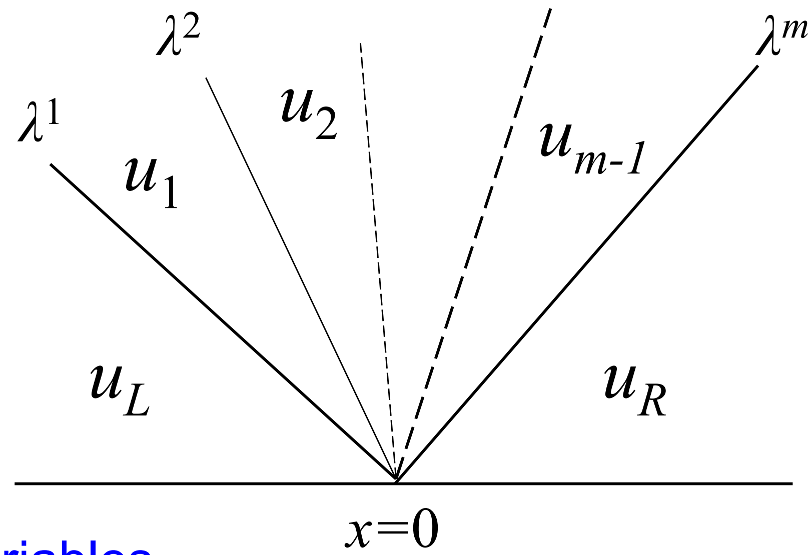
We can then define characteristic variable $\boldsymbol{w} = R^{-1}\boldsymbol{u}$, and the equation becomes

$$\partial_t w^p + \lambda^p \partial_x w^p = 0$$

=> a set of decoupled linear advection equations, with λ^p being wave speeds.

The Riemann problem (for a linear system)

$$\partial_t \boldsymbol{u} + \mathbf{A} \cdot \partial_x \boldsymbol{u} = 0$$



Initial condition: $\boldsymbol{u}$=$\boldsymbol{u}_L$, (x<0)
$\boldsymbol{u}$=$\boldsymbol{u}_R$, (x≥0)

Solution:

1). Decompose u_L, u_R into characteristic variables.

$$\boldsymbol{u}_{L,R} = \sum_p w^p_{L,R} \boldsymbol{r}^p$$

2). Each characteristic variable evolves according to its own characteristics.

$$w^p(x,t) = w^p_L \text{ if } x - \lambda^p t < 0 \text{, otherwise, } w^p(x,t) = w^p_R$$

3). Convert back to original variables.

$$\boldsymbol{u}(x,t) = \sum_{p:\lambda^p < x/t} w^p_R \boldsymbol{r}^p + \sum_{p:\lambda^p > x/t} w^p_L \boldsymbol{r}^p$$

Hyperbolic PDEs and conservation laws

- An important class of hyperbolic PDEs is conservation laws:

$$\partial_t \boldsymbol{u} + \partial_x \boldsymbol{F}(\boldsymbol{u}) = 0$$

where $\boldsymbol{F}(\boldsymbol{u})$ is the flux function. It can be rewritten in quasi-linear form

$$\partial_t \boldsymbol{u} + \boldsymbol{F}'(\boldsymbol{u}) \cdot \partial_x \boldsymbol{u} = 0$$

It is hyperbolic if $\boldsymbol{F}'(\boldsymbol{u})$ is diagonalizable with real eigenvalues for all $\boldsymbol{u}$.

- Ideal MHD equations are (non-linear, multi-D) conservation laws:

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

$$\frac{\partial \rho \boldsymbol{v}}{\partial t} + \nabla \cdot (\rho \boldsymbol{v}\boldsymbol{v} - \boldsymbol{B}\boldsymbol{B} + \mathbf{P}^*) = 0,$$

$$\frac{\partial E}{\partial t} + \nabla \cdot [(E + P^*)\boldsymbol{v} - \boldsymbol{B}(\boldsymbol{B} \cdot \boldsymbol{v})] = 0,$$

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

Ideal MHD equations are hyperbolic because all wave speeds are real.

Non-linear equations

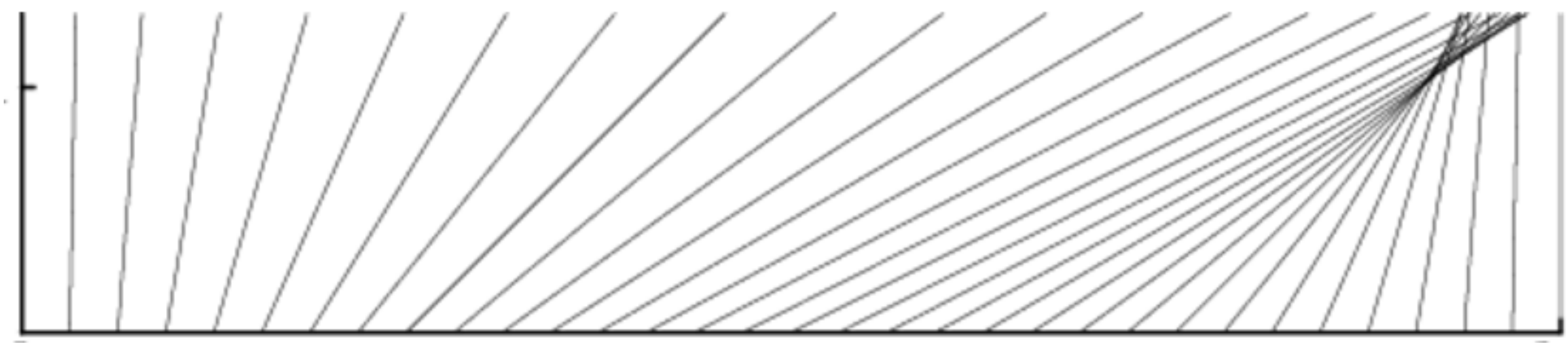
- Hydro/MHD equations are non-linear => further complications
- Simplest example: Burger's Equation

$$\partial_t u + u\partial_x u = 0$$

Characteristic curve: $X(t) = X_0 + u_0(X_0)t$

Characteristic curves converge when $\partial_x u_0 > 0$

Characteristic curves diverge when $\partial_x u_0 < 0$



When converging, characteristic curves cross!

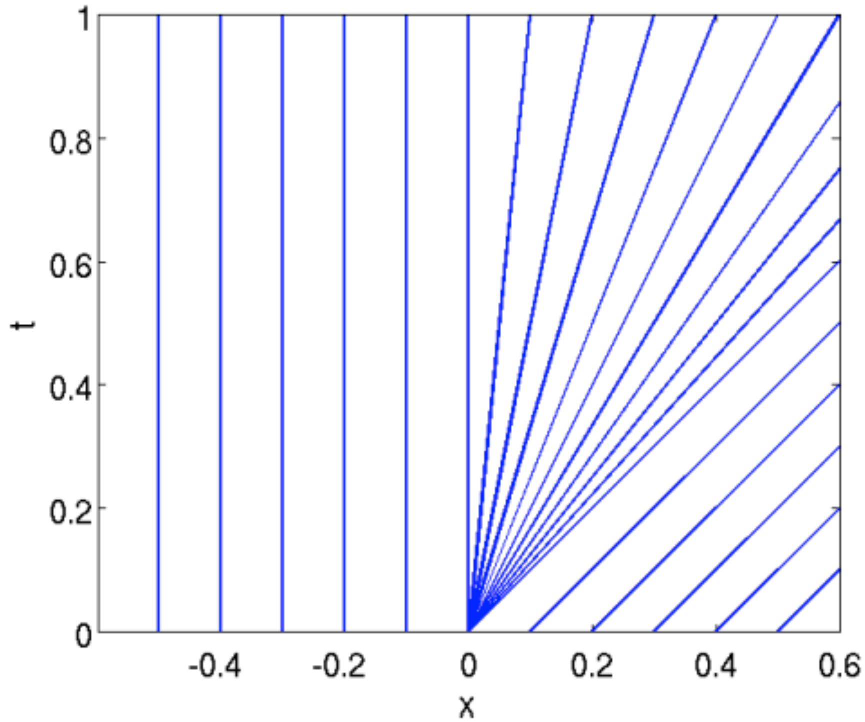
This should not happen physically: non-linear steepening into shocks.

Riemann problem for non-linear equations

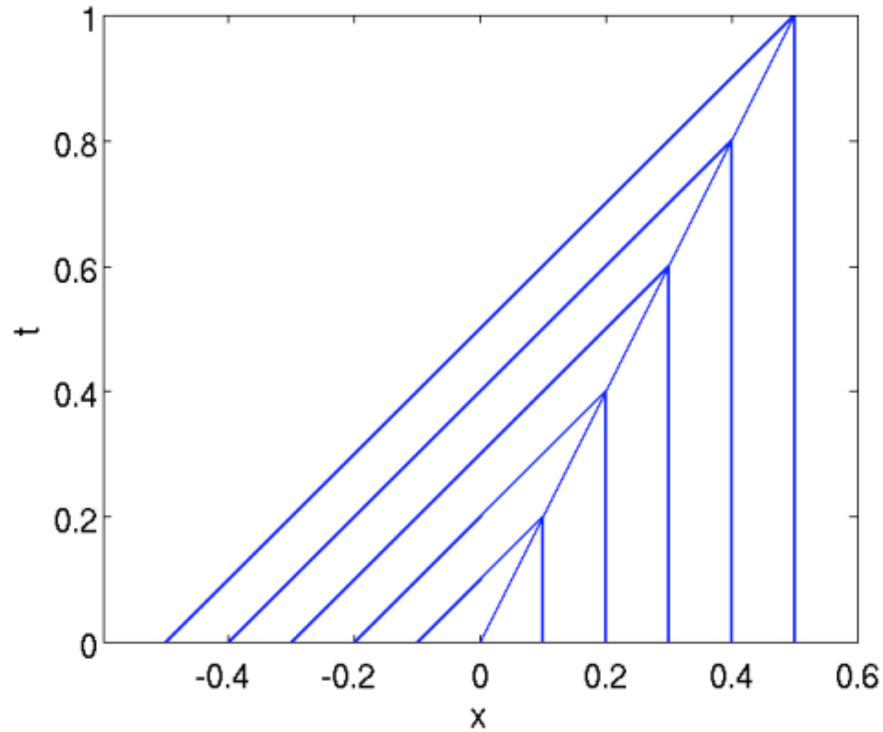
$$\partial_t u + u\partial_x u = 0$$

Initial condition: $u=u_L$, $(x<0)$

$u=u_R$, $(x\geq 0)$



$u_L<u_R$: rarefaction



$u_L>u_R$: shock

These behaviors can also be obtained by adding an infinitely small diffusion term.

Discontinuities

- Remarkably, hyperbolic PDEs admit discontinuous solutions. In MHD, these are contact discontinuities and shocks!
- Mathematically, these are exact non-linear solutions to the PDEs
- At discontinuities, the classical form of the PDEs fail. These solutions are captured in the integral form (more fundamental):

$$\int_{x_1}^{x_2} [u(x,t_2) - u(x,t_1)]dx = \int_{t_1}^{t_2} F[u(x_1,t)] - F[u(x_2,t)]dt$$

for any t_1, t_2, x_1, x_2.

Solutions to this are called weak solutions.

- MHD algorithms must be able to handle both smooth flows and discontinuous.

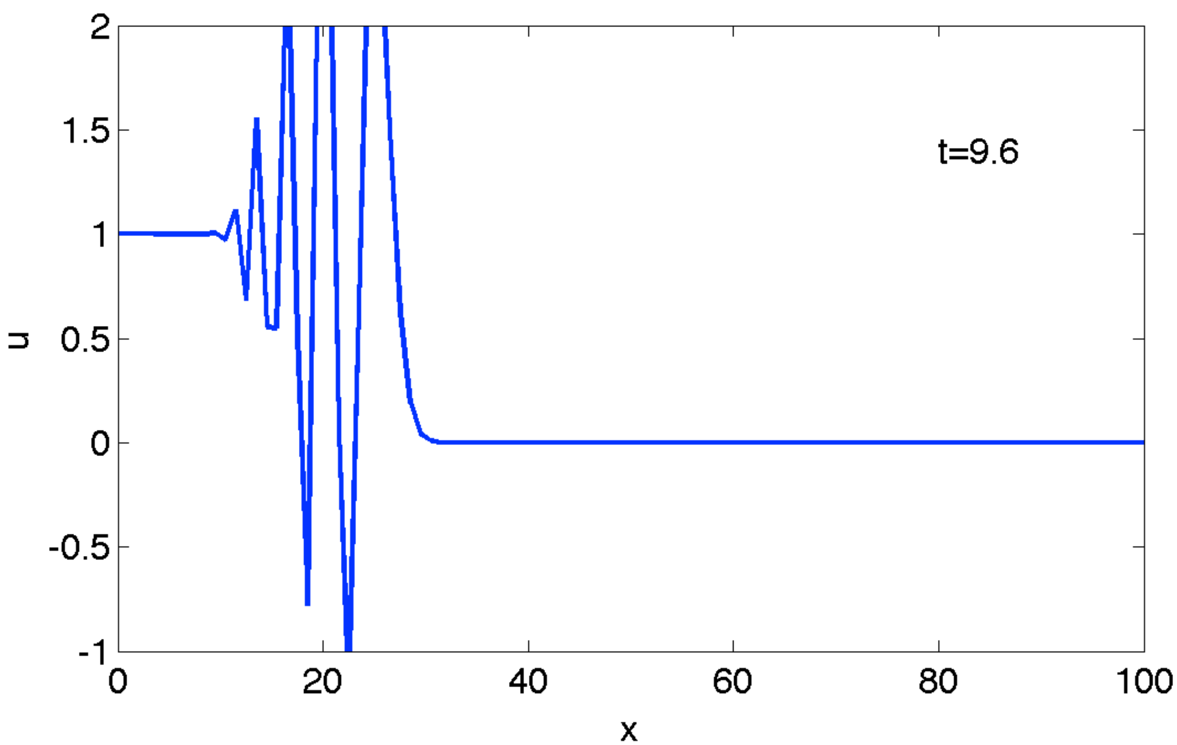
Solve the linear advection equation on a grid

$$\partial_t u + A\partial_x u = 0$$

Initial condition: u=1 (x<50), u=0 (x>50), A=1.

1. Forward-time central-space (FTCS):

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



The method is unconditionally unstable!

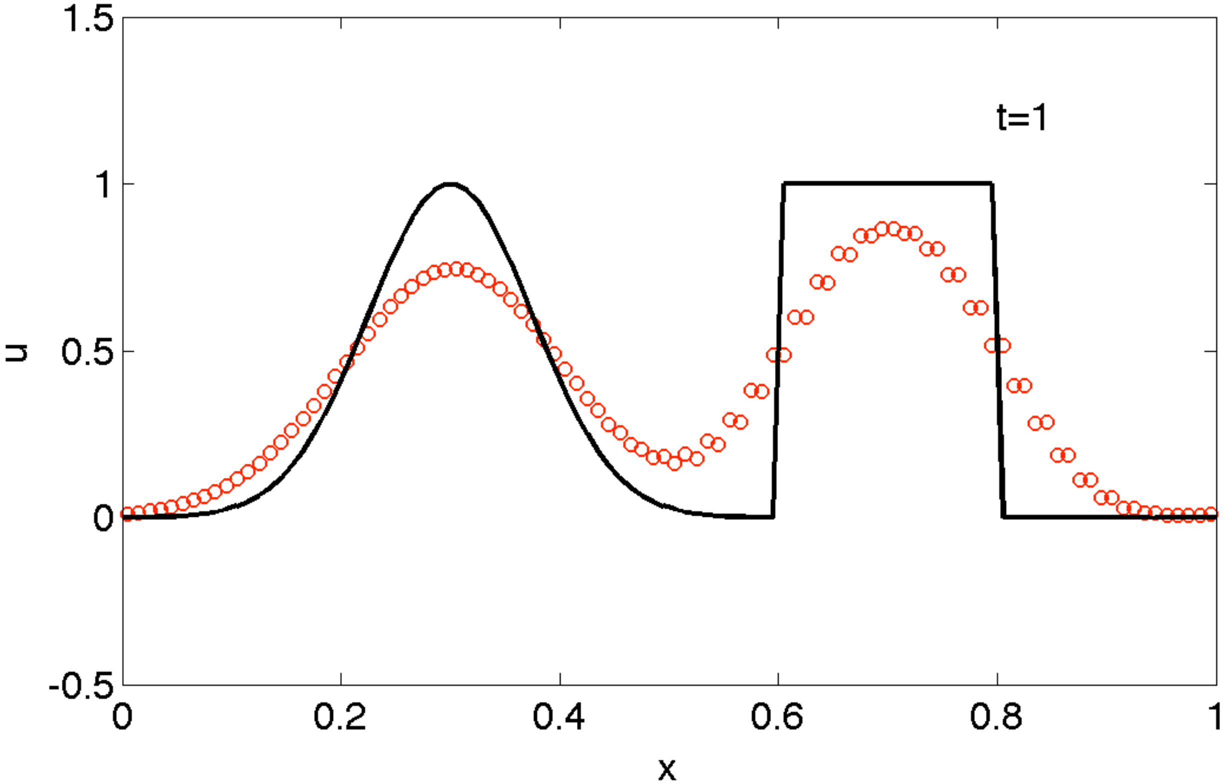
Solve the linear advection equation on a grid

$$\partial_t u + A\partial_x u = 0$$

IC: one Gaussian, one square waves, $A=1$, periodic BC.

2. Lax-Friedrichs (LF) method:

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



Effectively, added extra numerical diffusion to stabilize FTCS.

The method is stable, but VERY diffusive!

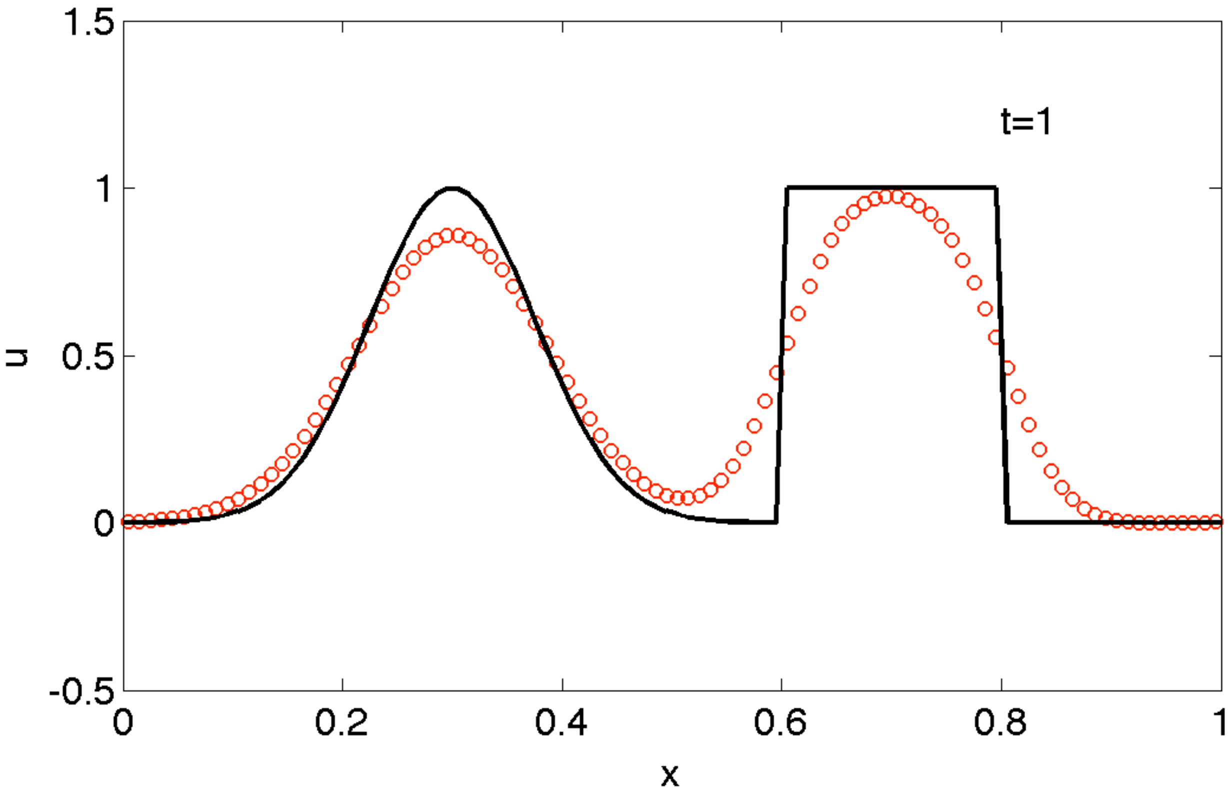
Solve the linear advection equation on a grid

$$\partial_t u + A\partial_x u = 0$$

IC: one Gaussian, one square waves, A=1, periodic BC.

3. Upwind method:

$$\frac{u_i^{n+1} - u_i^n}{\Delta t} = \begin{cases} -A(u_i^n - u_{i-1}^n)/\Delta x & (A \geq 0) \\ -A(u_{i+1}^n - u_i^n)/\Delta x & (A < 0) \end{cases}$$



The method is stable and less diffusive, though still only first order accurate.

Can be improved using higher-order spatial interpolation schemes.

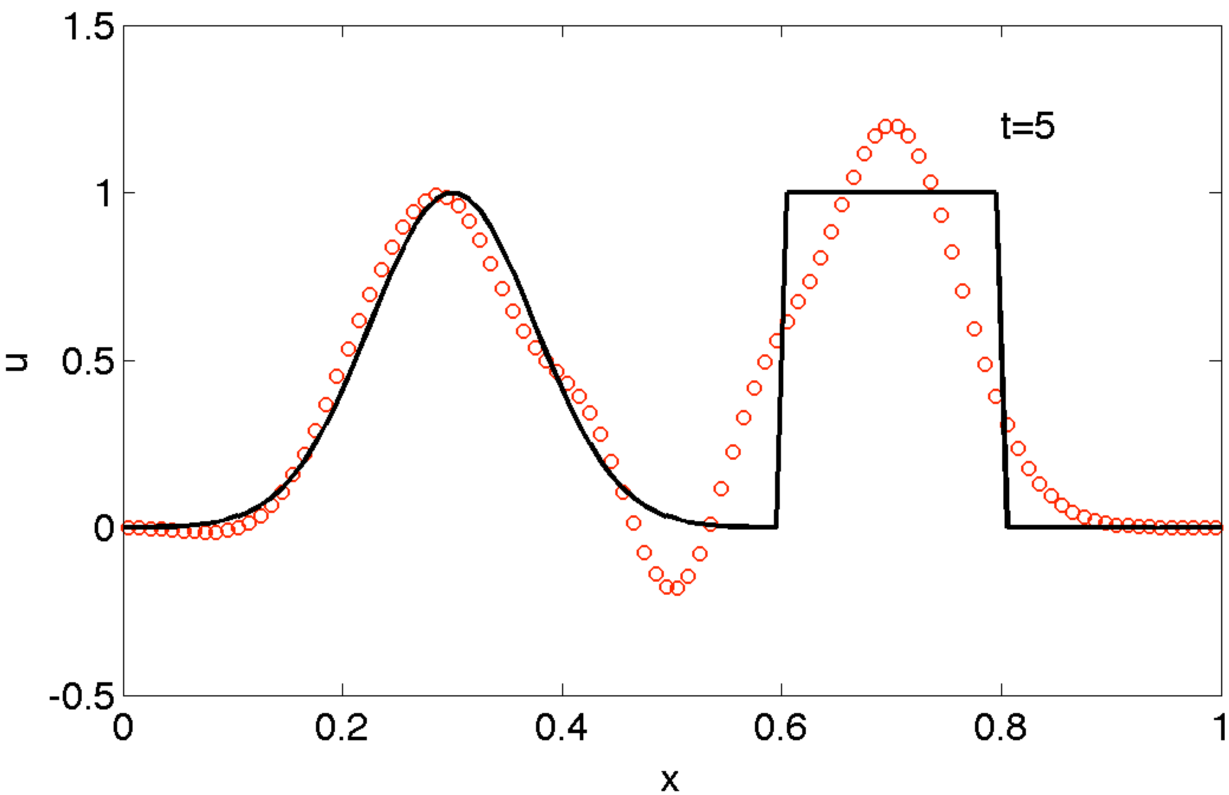
Towards higher-order accuracy

$$\partial_t u + A\partial_x u = 0$$

IC: one Gaussian, one square waves, $A=1$, periodic BC.

4. Lax-Wendroff method:

$$\frac{u_i^{n+1} - u_i^n}{\Delta t} = -A\left(\frac{u_{i+1}^n - u_{i-1}^n}{2\Delta x}\right) + \frac{A^2\Delta t}{2}\frac{(u_{i_1}^n - 2u_i^n + u_{i-1}^n)}{\Delta x^2}$$



Method is stable but:

1). Oscillatory solution at discontinuities.

2). It is dispersive.

The Courant-Friedrichs-Lewy (CFL) condition

- Numerical timestep Δt must be sufficiently small so that information propagates no more than one grid point per timestep.
- Fundamental criterion for numerical stability (can be derived rigorously via von Neumann analysis).

For the linear advection problem $\partial_t u + A\partial_x u = 0$,

$$\Delta t \equiv C\frac{\Delta x}{A}$$, where the **CFL number** $C \leq 1$.

I took C=0.8 for the previous two examples. In fact, these methods are more accurate as C approaches 1.

For MHD equations, take A to be:

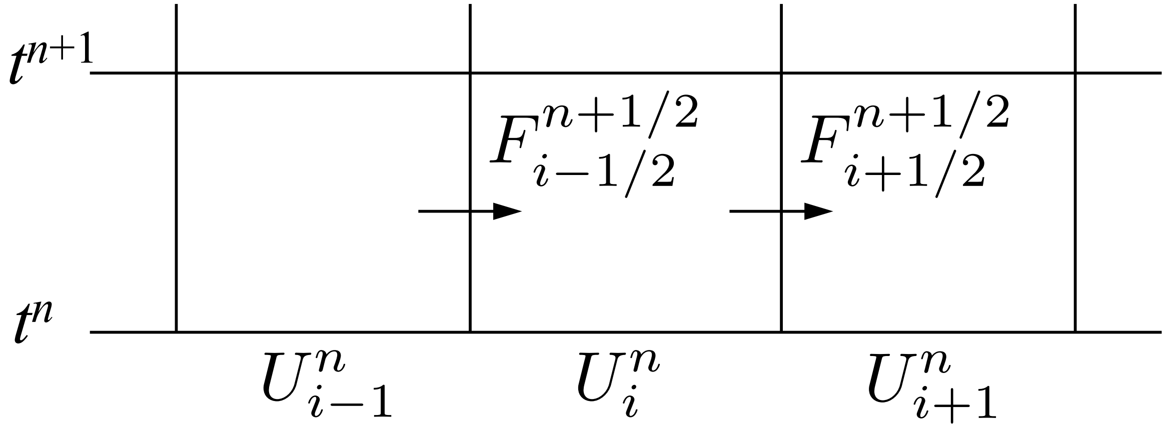
$$A = \max[\mathrm{abs}(v + v_f),\ \mathrm{abs}(v - v_f)]$$

where v is flow speed, v_f is the fast magnetosonic speed.

Timestep is determined by the minimum Δt across the entire mesh.

Finite volume method

- FVM works with the integral form of the conservation laws.



Conserved variables are volume-averaged:

$$U_i^n = \frac{1}{\Delta x}\int_{x_{i-1/2}}^{x_{i+1/2}} u(x, t_n)dx$$

Interface fluxes are time-averaged:

$$F^{n+1/2}_{i-1/2} = \frac{1}{\Delta t}\int_{t_n}^{t_{n+1}} f(u(x_{i-1/2}, t))dt$$

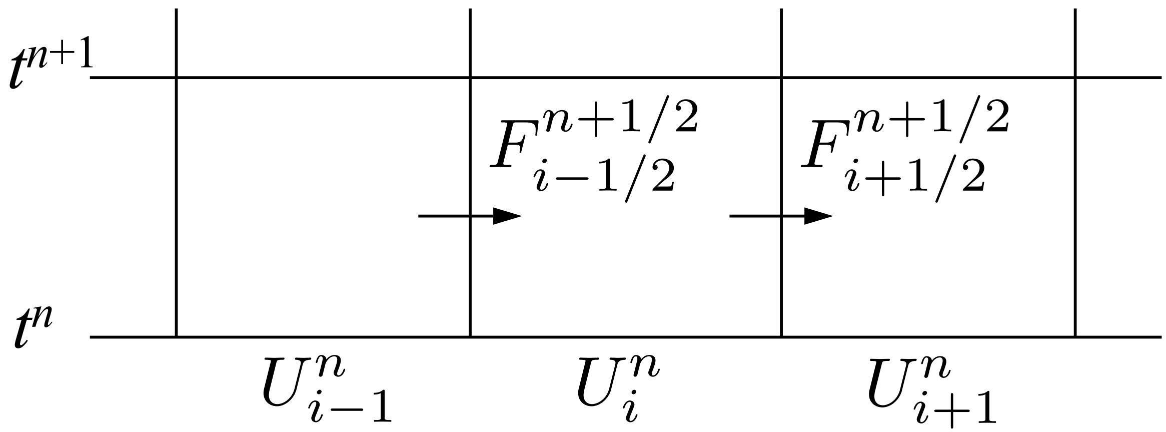
Finite volume update:

$$U_i^{n+1} = U_i^n + \frac{\Delta t}{\Delta x}(F^{n+1/2}_{i+1/2} - F^{n+1/2}_{i-1/2})$$

Conserved variables are conserved to machine accuracy.

How to compute the fluxes?

Consider again the linear advection equation: $\partial_t u + A\partial_x u = 0$



We only know the volume-averaged values U, but to get the flux at cell interfaces, we need to know the value of u at $x_{i+1/2}$. This has to be done approximately.

Upwind flux:

$$F_{i-1/2} = \begin{cases} AU_{i-1} & (A \geq 0) \\ AU_i & (A < 0) \end{cases}$$

Lax-Friedrichs:

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

Lax-Wendroff:

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

Godunov method (Basic idea) (Godunov, 1959)

- 1. Given volume averaged values U_i^n (defined at each cell), reconstruct piecewise polynomial function $\tilde{u}^n(x)$ (defined at all x).

  Simplest scenario (piecewise constant/donor cell):

$$\tilde{u}^n(x) = U_i^n \quad \text{for } x_{i-1/2} \le x < x_{i_1/2}$$

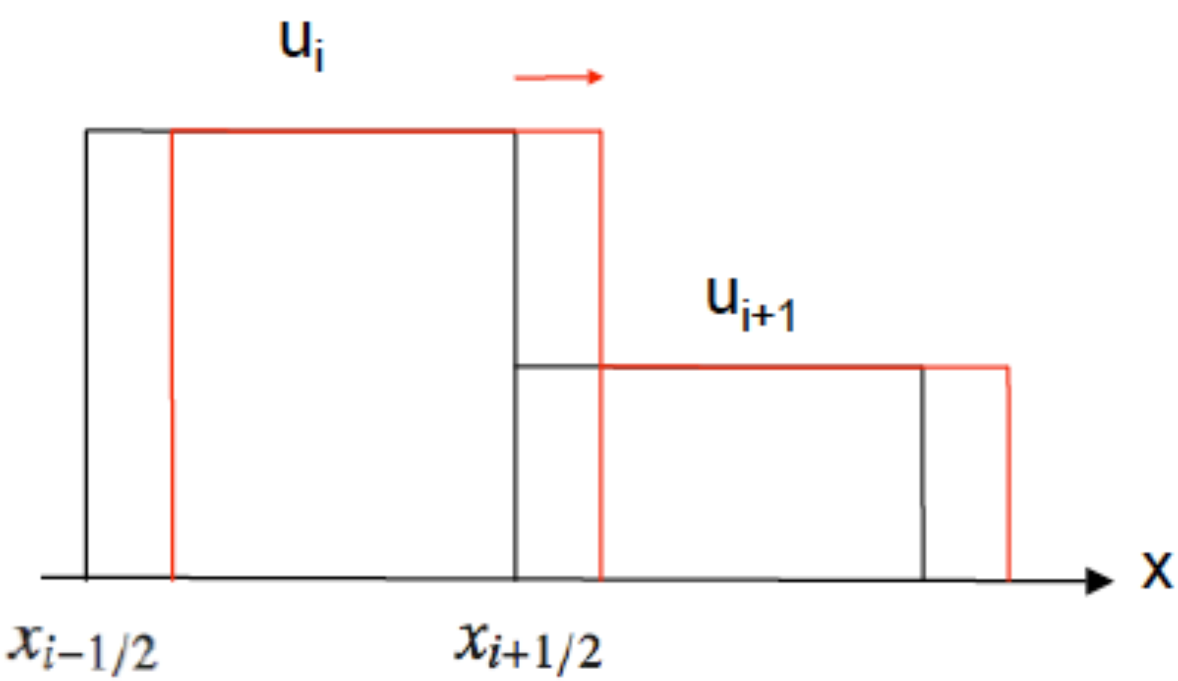
- 2. Using $\tilde{u}^n(x)$ as initial condition, evolve the hyperbolic equation exactly (or approximately) for Δt to obtain $\tilde{u}^{n+1}(x)$.
- 3. Average $\tilde{u}^{n+1}(x)$ over each cell to obtain new cell averages:

$$U_i^{n+1} = \frac{1}{\Delta x} \int_{x_{i-1/2}}^{x_{i+1/2}} \tilde{u}^{n+1}(x) dx$$

Godunov method (Basic idea)

A finite volume method originally proposed by Godunov (1959) for solving (non-linear) equations of gas dynamics.

Reconstruct ->
Evolve ->
Average



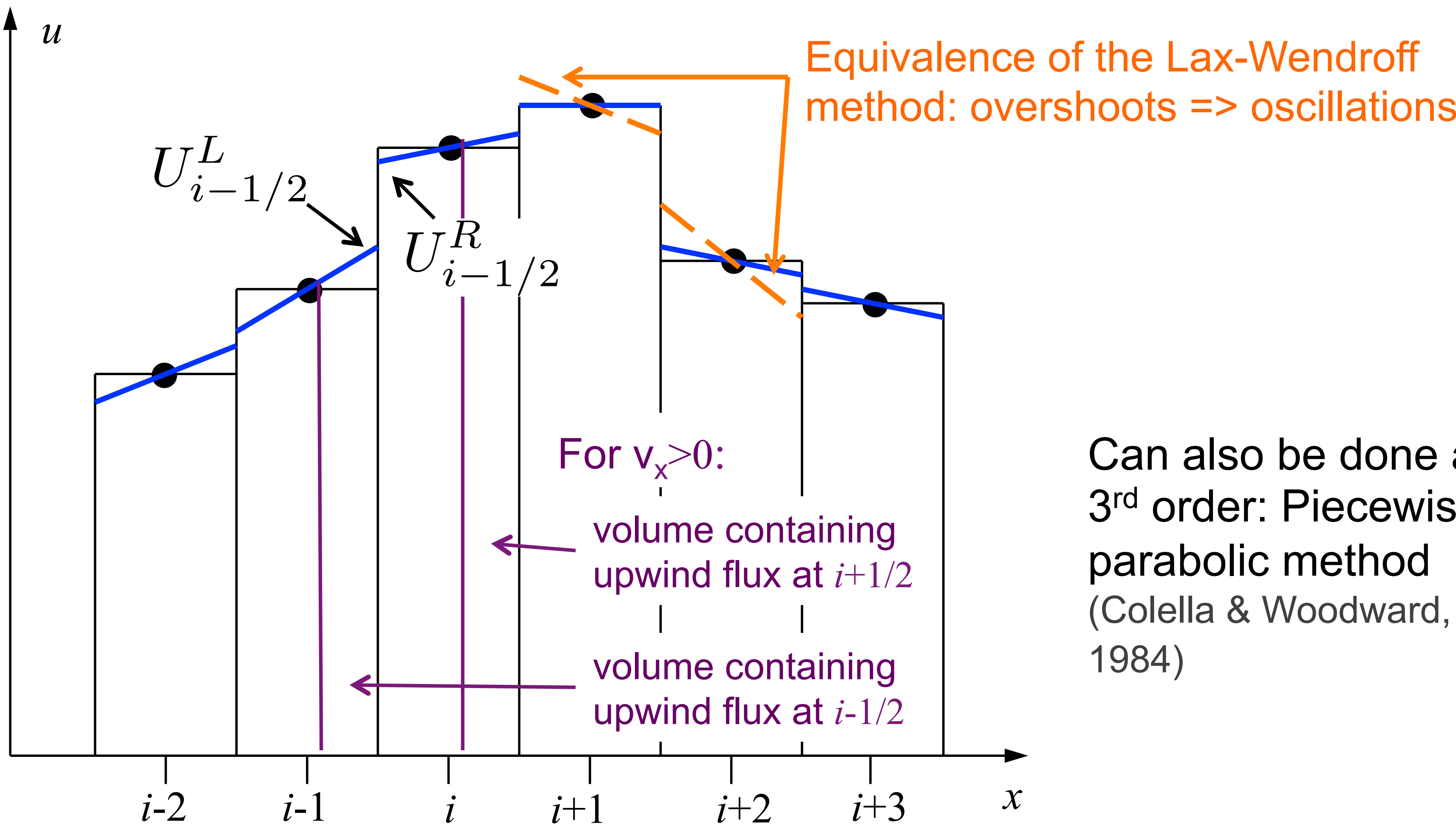
For linear advection equations, Godunov method with piecewise constant reconstruction = upwind method.

Key property: flux is properly upwinded to avoid spurious oscillations.

Toward higher order accuracy

Piecewise linear reconstruction (MUSCL scheme, van Leer, 1979)

Need slope limiting to avoid oscillations near discontinuities.



Can also be done at 3rd order: Piecewise-parabolic method (Colella & Woodward, 1984)

Slope limiters

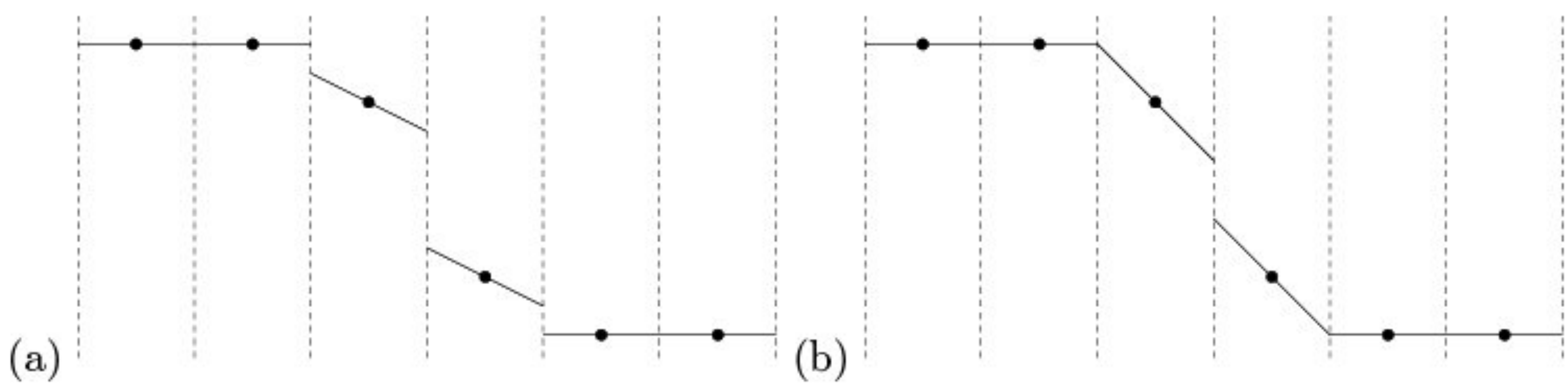
A slope limiter needs to be "monotonicity preserving": not to produce any local extrema.

It can be shown that "total variation diminishing" (TVD) limiters are monotonicity preserving, where "total variation" is defined as

$$\mathrm{TV}(Q) = \sum_{i=-\infty}^{\infty} |Q_i - Q_{i-1}|$$

TVD means $\mathrm{TV}(Q^{n+1}) \leq \mathrm{TV}(Q^n)$.

Popular choices of slope limiters include minmod, MC, van Leer, etc.



minmod monotonized central-difference (MC)

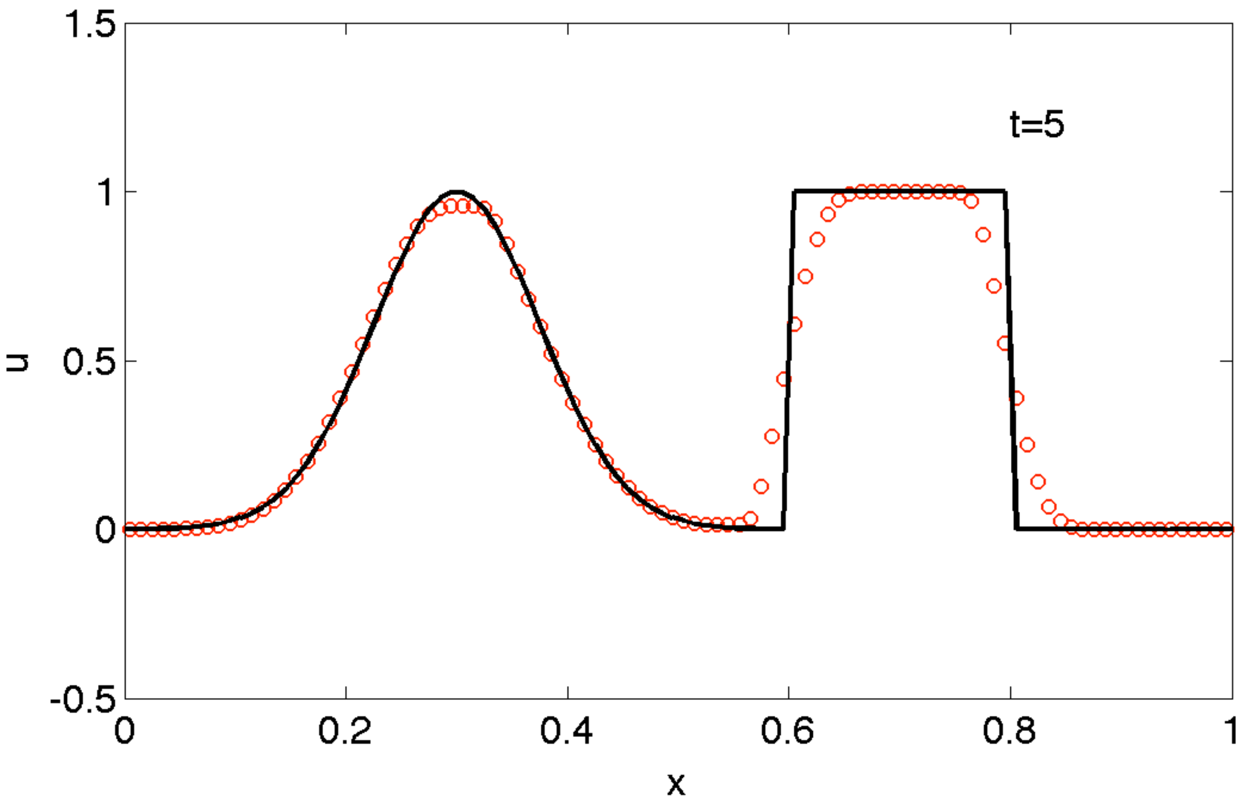
Solve the linear advection equation

Higher-order Godunov method with piecewise linear reconstruction

+the MC slope limiter.

$$\partial_t u + A\partial_x u = 0$$

Initial condition: one Gaussian, one square waves, $A=1$, periodic BC.



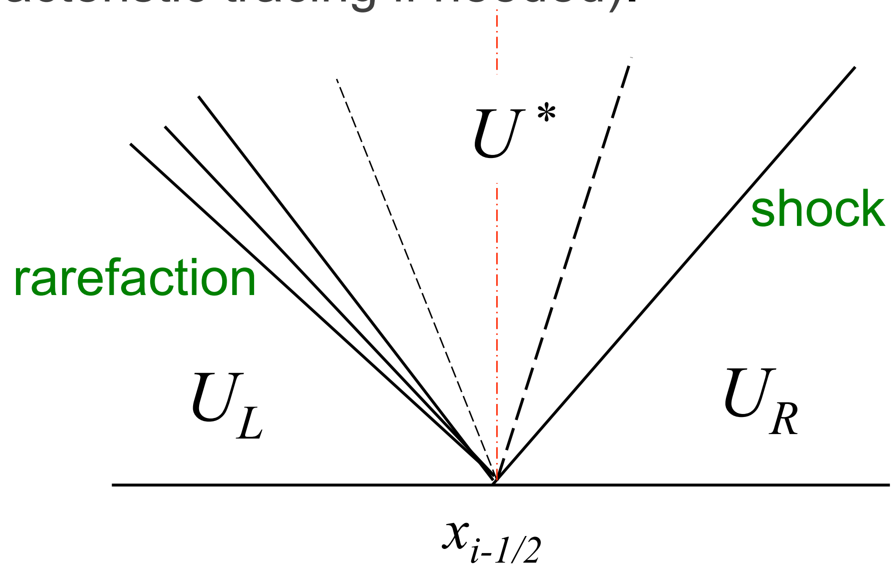
The method is stable and much more accurate:

2nd order accurate for smooth flow
1st order accurate at discontinuities.

Godunov method (Generalization to non-linear problems)

- 1. Given volume averaged values U_i^n (defined at each cell), compute the left/right states $U_{L,i\text{-}1/2}/U_{R,i\text{-}1/2}$ at cell interfaces based on a reconstruction method (+characteristic tracing if needed).

- 2. Solve the Riemann problem across cell interfaces to obtain intermediate state U^*.



- 3. Define interface flux

$$F_{i-1/2} = F(U^*)$$

where F(U) is the flux function.

- 4. Apply the flux-differencing formula:

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

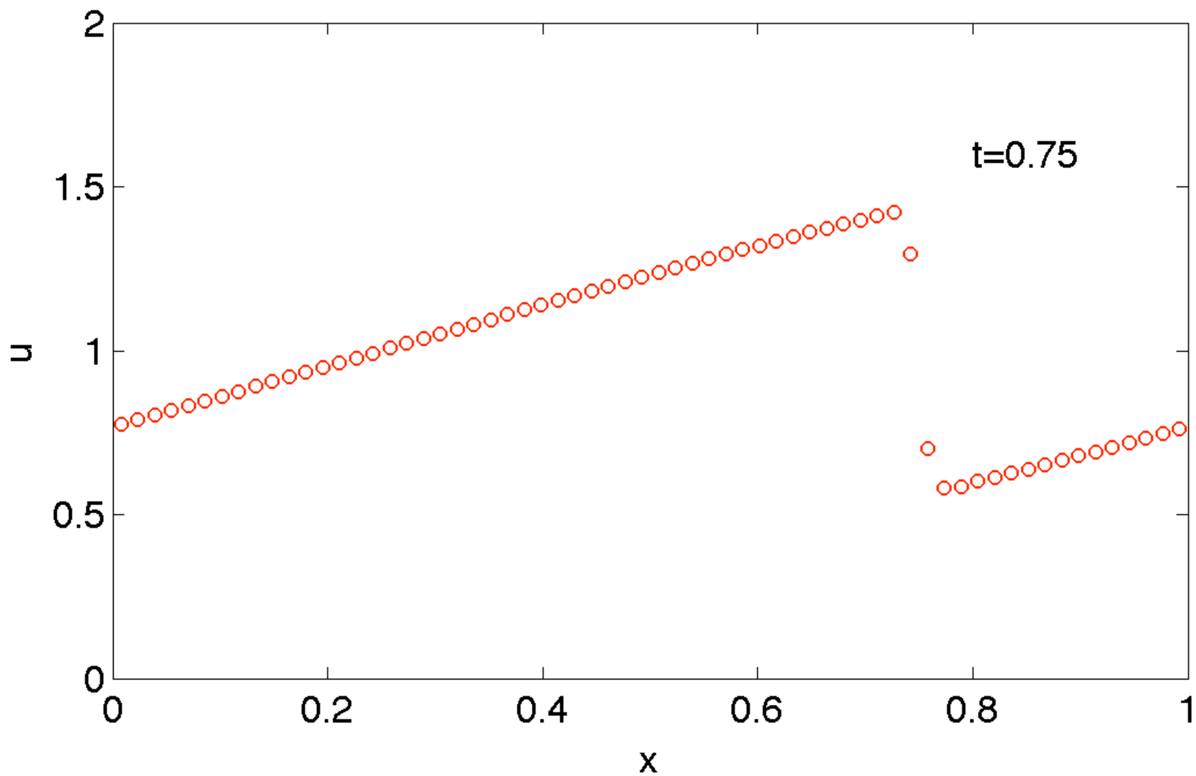
Solving non-linear equations

- Simplest example: Burger's Eqs $\partial_t u + u\partial_x u = 0$

In conservative form: $\partial_t u + \partial_x \left(\frac{u^2}{2}\right) = 0$

Initial condition: u=1-sin(2πx)/2 in [0, 1], periodic BC.

Solved with Godunov method + 2nd order reconstruction



1D MHD Equations

1D equations are plane-symmetric: $\nabla \cdot \boldsymbol{B} = 0$ => $B_x = \mathrm{const}$

1D adiabatic MHD equations in conservative form: $\dfrac{\partial \boldsymbol{U}}{\partial t} + \dfrac{\partial \boldsymbol{F}}{\partial x} = 0$

$$\boldsymbol{U} = \begin{bmatrix} \rho \\ M_x \\ M_y \\ M_z \\ E \\ B_y \\ B_z \end{bmatrix} \qquad \boldsymbol{F} = \begin{bmatrix} \rho v_x \\ \rho v_x^2 + P + B^2/2 - B_x^2 \\ \rho v_x v_y - B_x B_y \\ \rho v_x v_z - B_x B_z \\ (E + P^*) v_x - (\boldsymbol{B} \cdot \boldsymbol{v}) B_x \\ B_y v_x - B_x v_y \\ B_z v_x - B_x v_z \end{bmatrix}$$

7 variables, 7 waves

The Eigensystem

The Jacobian matrix $\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{q}}$ has 7 real eigenvalues, one for each wave:

2 fast magnetosonic waves: $\lambda_1 = v_x - c_f \qquad \lambda_7 = v_x + c_f$

2 Alfven waves: $\lambda_2 = v_x - v_{A,x} \qquad \lambda_6 = v_x + v_{A,x}$

2 slow magnetosonic waves: $\lambda_3 = v_x - c_s \qquad \lambda_5 = v_x + c_s$

1 entropy wave: $\lambda_4 = v_x$

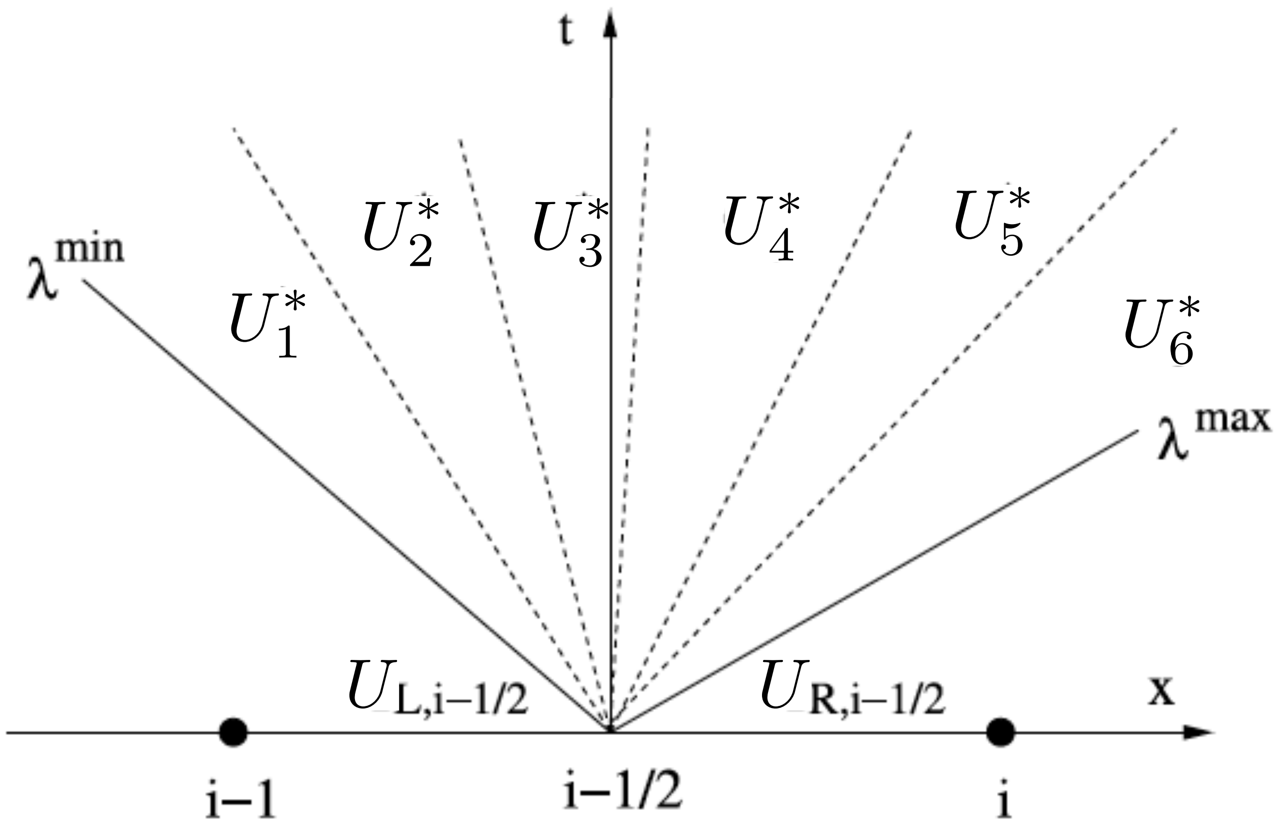
where $$c_{f,s}^2 = \frac{1}{2}\left[(a^2 + v_A^2) \pm \sqrt{(a^2 + v_A^2)^2 - 4a^2 v_{A,x}^2}\right]$$

$$v_{A,x} = \frac{B_x}{\sqrt{4\pi\rho}} \qquad v_A = \frac{B}{\sqrt{4\pi\rho}} \qquad a^2 = \gamma P/\rho$$

An "entropy wave" is a contact discontinuity.

For isothermal MHD, the number reduces to 6 (no entropy wave).

The MHD Riemann problem



Due to highly non-linear nature, there is no exact MHD Riemann solver.

Exact hydro Riemann solver is possible, but numerically very expensive.

Approximate Riemann solvers are widely used in computational MHD.

MHD Riemann solvers

The HLLE solver: (Harten, Lax & van Leer, 1983, Einfeldt et al. 1991)

Keeps the fastest/slowest waves, averages all intermediate states in between.

Advantage: very simple and efficient; intermediate state is positive definite.

However, it is very diffusive, especially at contact discontinuities.

The HLLD solver: (Miyoshi & Kasano, 2005)

5-wave Riemann solver with 4 intermediate states: resolves fast, Alfven waves and the contact discontinuity.

Reasonably simple and efficient, guarantees positivity in 1D, better resolution at contact discontinuities.

There is also the HLLC solver for hydrodynamics (which includes a contact wave).

The Roe solver: (Roe, 1981, Cargo & Gallice, 1997)

Exact Riemann solver derived from linearized (approximate) MHD equations.

Captures all 7 waves, generally less diffusive and more accurate, but requires characteristic decomposition (expensive), and can fail for certain L/R states.

Primitive vs. conserved variables

It is necessary to convert conserved variables $\boldsymbol{U}$ to primitive variables $\boldsymbol{W}$ in various stages of the computation.

$$\boldsymbol{U} = \begin{bmatrix} \rho \\ M_x \\ M_y \\ M_z \\ E \\ B_x \\ B_y \\ B_z \end{bmatrix}, \quad \mathbf{W} = \begin{bmatrix} \rho \\ v_x \\ v_y \\ v_z \\ P \\ B_x \\ B_y \\ B_z \end{bmatrix},$$

Caveat: Due to the approximate nature of the Riemann solver, one might get negative density after one step of integration.

Similarly, with

$$E = \frac{P}{\gamma - 1} + \frac{1}{2}\rho v^2 + \frac{B^2}{8\pi}$$

one might obtain negative pressure following conversion from conserved to primitive variables.

These issues can be more severe in relativistic MHD.

Solution:

1). Add density/pressure floors.

2). Use a more diffusive solver.

MHD integrator

- Godunov's original method (1st order)

Step 1: Donor-cell reconstruction to obtain interface L/R states.

Step 2: Use an MHD Riemann solver to compute 1st order fluxes.

Step 3: Update the system for a full time step using 1st order fluxes.

Robust, but very diffusive.

MHD integrator

- Second-order accuracy can be achieved using predictor-corrector type method (with a number of varieties).

Step 1: Donor-cell reconstruction to obtain interface L/R states.

Step 2: Use a Riemann solver to compute 1st order fluxes F^n.

Step 3: Advance the system for ½ time step (predict step).

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

Step 4: Use the second-order (piecewise-linear) reconstruction to compute the L/R states from $U^{n+1/2}$.

Step 5: Use a Riemann solver to compute 2nd order fluxes $F^{n+1/2}$.

Step 6: Update the system for a full time step.

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

This is one algorithm adopted in Athena, following Falle (1991), modified from the MUSCL-Hancock schemes.

Multi-dimension MHD

- MHD equations in conservative form in 3D:

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

- Traditionally, multi-D methods are constructed using directional spliting:

1. Solve $U_t = F_x$ as in 1D MHD.
2. Solve $U_t = G_y$, with G constructed from result of the x-update.
3. Solve $U_t = H_z$, with H constructed from result of the y-update.

Pros: easy to implement.

Cons: symmetry is not preserved, incompatible with constrained transport, extension to AMR (adaptive mesh refinement) is not straightforward.

- Need unsplit methods (which most current MHD codes adopt): 3 directions updated at the same time.

Importance to preserve divergence of B

- For multi-dimensions numerical schemes, there is no guarantee that divergence of B is kept zero, due to truncation error.

Consequence:

spurious parallel acceleration

$$\boldsymbol{J} \times \boldsymbol{B} = -\nabla \cdot \left(\frac{B^2}{8\pi} \mathbf{I} - \frac{\boldsymbol{B}B}{4\pi} \right) - \frac{(\nabla \cdot \boldsymbol{B})\boldsymbol{B}}{4\pi}$$

Divergence error can accumulate, leading to inconsistent results over long term.

In some cases, it can lead to numerical instabilities and make the code crash…

Techniques to preserve divergence of B

- Divergence cleaning:

Powell's 8-wave scheme (Powell, 1999):

Add source terms to momentum/induction equations to advect magnetic monopoles away. But: can give the wrong shock jump conditions.

Projection method (Brackbil & Bams, 1980):

Solve a Poisson equation for the "magnetic charge": $\Delta\Phi = \nabla \cdot \boldsymbol{B}$

Then clean the divergence field: $\boldsymbol{B} \rightarrow \boldsymbol{B} - \nabla\Phi$

But: very expensive to solve elliptic PDE, and may smooth discontinuities in B.

Dedner's scheme (Dedner et al. 2002): introducing a general Lagrangian multiplier, transporting div(B) errors away. Reasonably robust in most cases.

- Use vector potential (usually used in finite-difference codes, e.g., Pencil)

$$\frac{\partial \boldsymbol{A}}{\partial t} = \boldsymbol{v} \times \boldsymbol{B} \ , \quad \boldsymbol{B} = \nabla \times \boldsymbol{A}$$

Div(B)=0 by construction, but need hyper-resistivity for stabilization.

Constrained transport (CT)

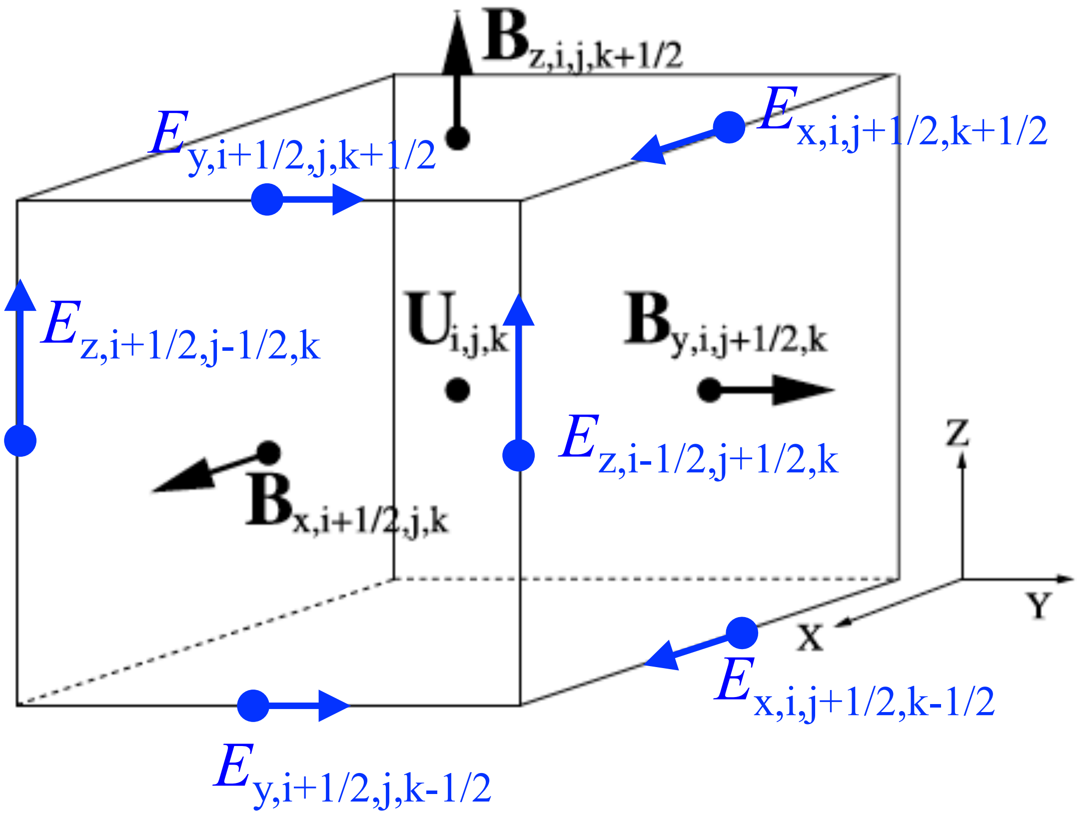
(Evans & Hawley, 1988)

Magnetic fields defined at face-center, area-averaged:

$$(B_x)_{i+1/2,j,k} = \frac{1}{\Delta y \Delta z} \int_S B_x(y,z)\,dydz$$

Electromotive forces (vxB) defined at edges, line-averaged:

$$(E_x)_{i,j+1/2,k-1/2} = \frac{1}{\Delta x \Delta t} \int E_x(x)\,dxdt$$



Evolve magnetic field via Stoke's law:

$$\frac{\partial}{\partial t}\int_S \boldsymbol{B}\cdot d\boldsymbol{S} = -\int_L \boldsymbol{E}\cdot d\boldsymbol{l}$$

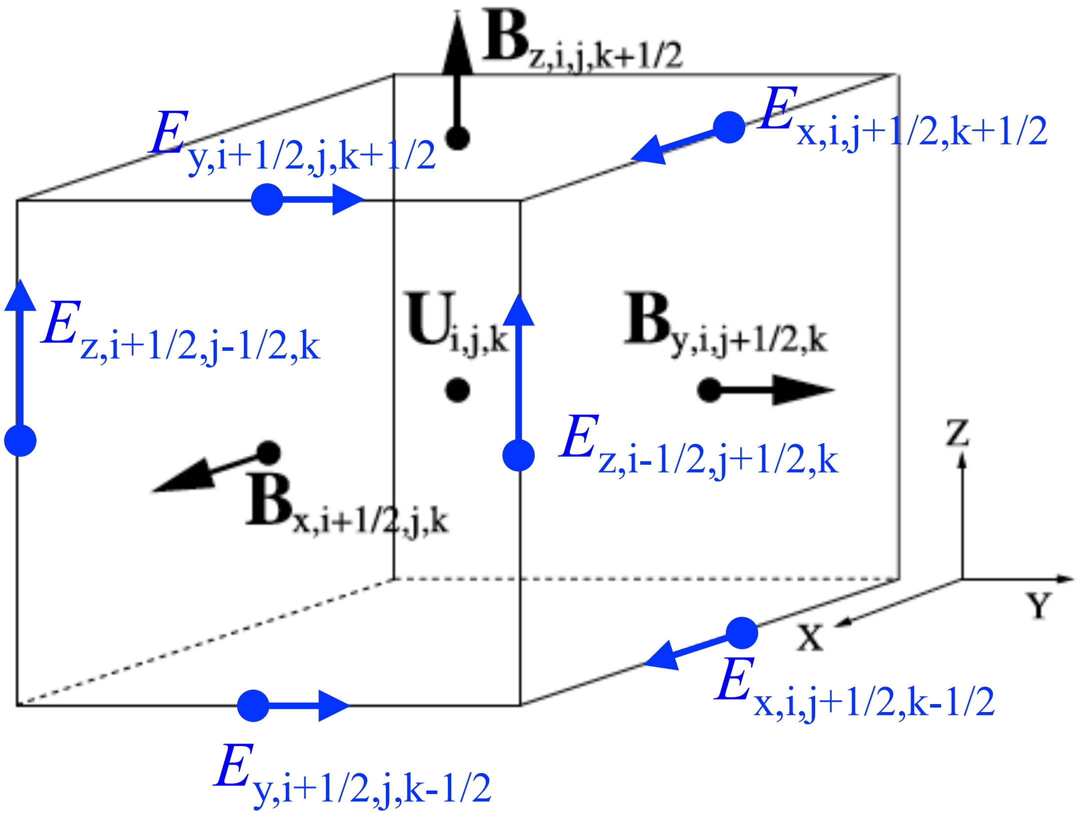
$$B^{n+1}_{x,i+1/2,j,k} = B^{n}_{x,i+1/2,j,k} - \frac{\Delta t}{\Delta y}\left(E^{n+1/2}_{z,i-1/2,j+1/2,k} - E^{n+1/2}_{z,i-1/2,j-1/2,k}\right) + \frac{\Delta t}{\Delta z}\left(E^{n+1/2}_{y,i-1/2,j,k+1/2} - E^{n+1/2}_{z,i-1/2,j,k-1/2}\right)$$

These equations are exact: no approximations.

Constrained transport (CT)

Div (B)=0 is preserved to machine accuracy:

$$\nabla \cdot \boldsymbol{B} = \frac{B_{x,i+1/2,j,k} - B_{x,i-1/2,j,k}}{\Delta x} + \frac{B_{y,i,j+1/2,k} - B_{x,i,j-1/2,k}}{\Delta x} + \frac{B_{z,i,j,k+1/2} - B_{z,i,j,k-1/2}}{\Delta x}$$



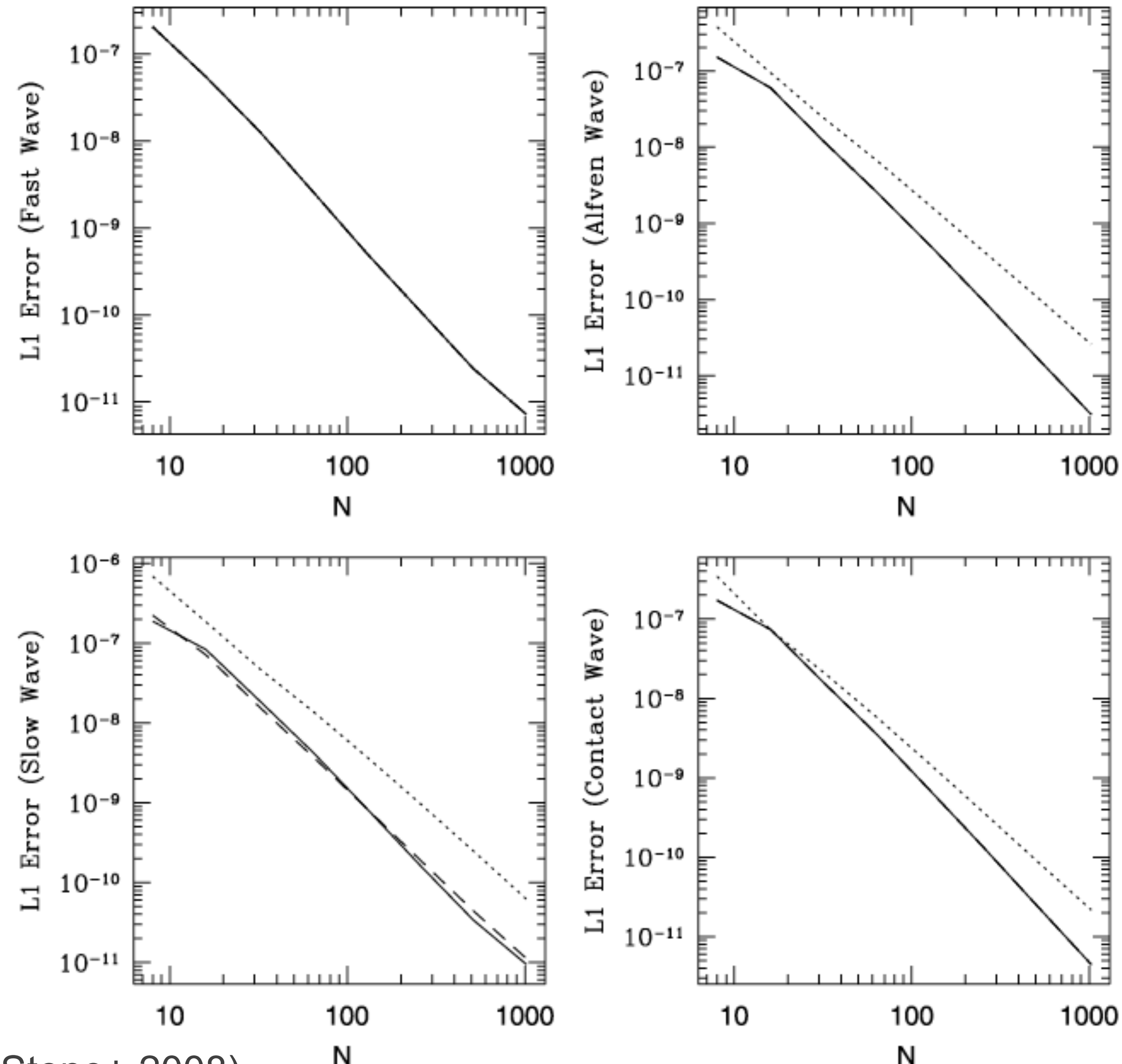
Updates in Div(B) corresponds to differences in the EMFs that cancel exactly.

Main challenge: construct electric fields at cell edges (3D) or corners (2D).

By arithmetic averaging the EMFs returned from the Riemann solvers (at face centers), the EMFs are not properly upwinded.

Need to reconstruct the EMF at the corners (Gardiner & Stone, 2005).

Code test: linear wave convergence



Initialize a pure eigenmode, and measure the L1 error after one wave period:

Quantitative test of the accuracy of the scheme.

With the ATHENA code:

Solid: Roe solver

Dashed: HLLD solver

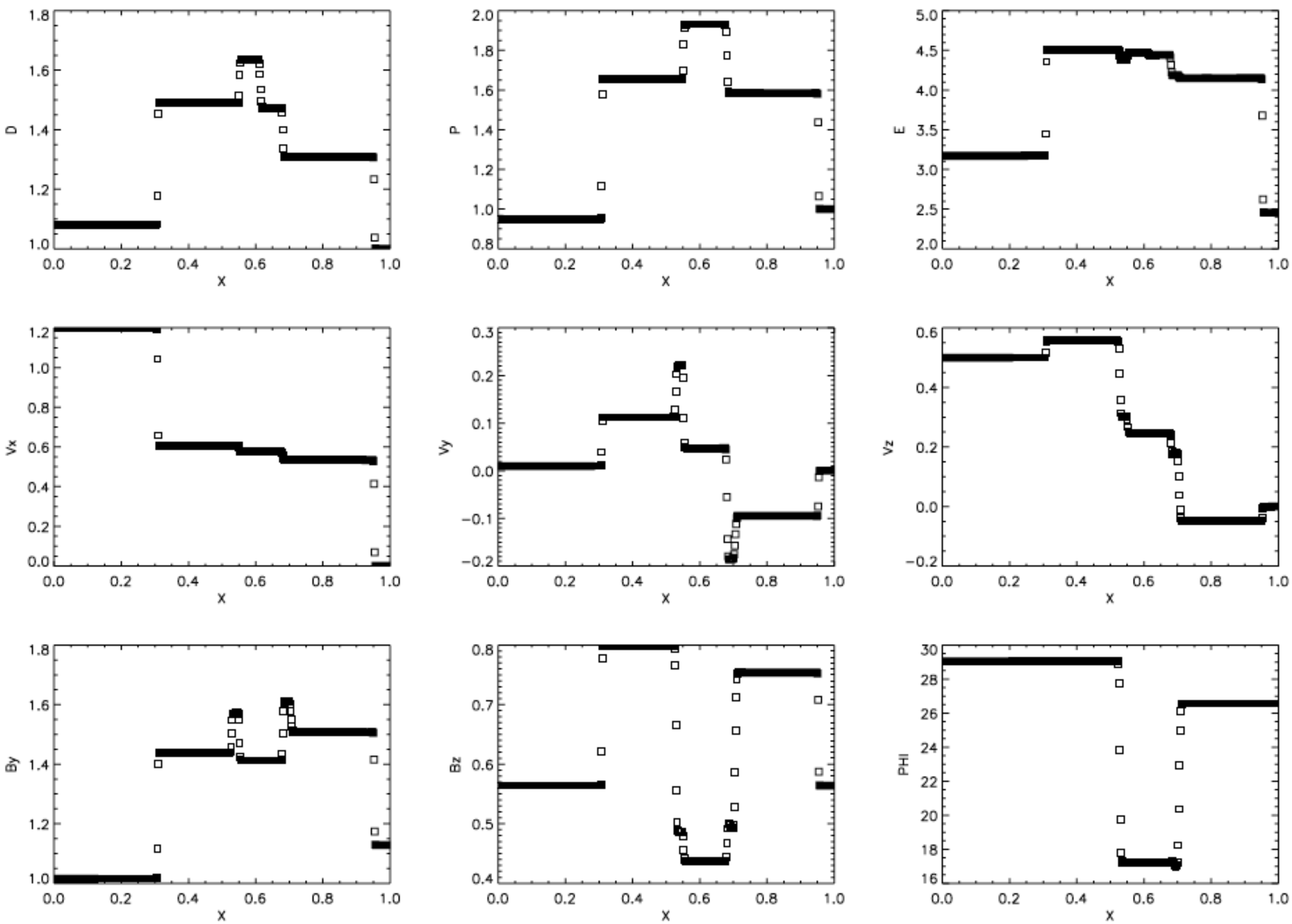
Dotted: HLLE solver

+3rd order reconstruction

(Stone+ 2008)

Code test: shock tube

Test	ρ_L	$v_{x,L}$	$v_{y,L}$	$v_{z,L}$	P_L	$B_{y,L}$	$B_{z,L}$	ρ_R	$v_{x,R}$	$v_{y,R}$	$v_{z,R}$	P_R	$B_{y,R}$	$B_{z,R}$
RJ2a..............................	1.08	1.2	0.01	0.5	0.95	3.6	2	1	0	0	0	1	4	2

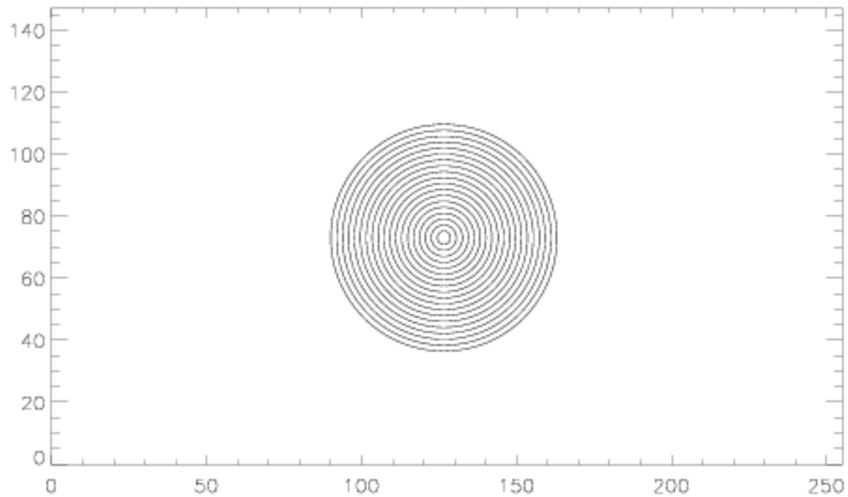


RJ2a shocktube (Ryu & Jones, 95) test: all 7 MHD waves are present.

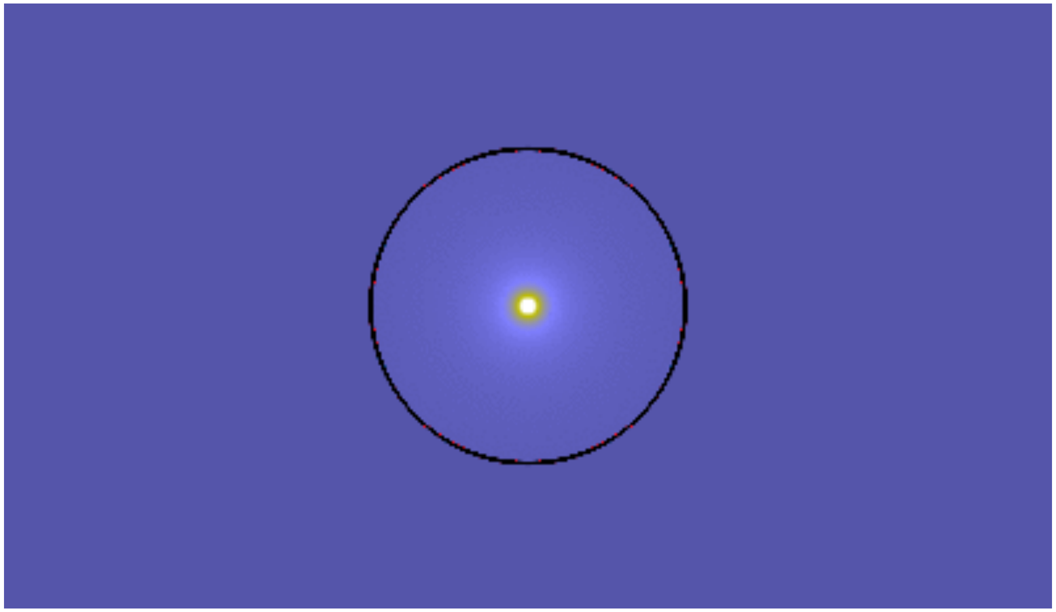
HLLD solver with 3rd order reconstruction: all 7 waves are captured well.

Code test: field loop advection

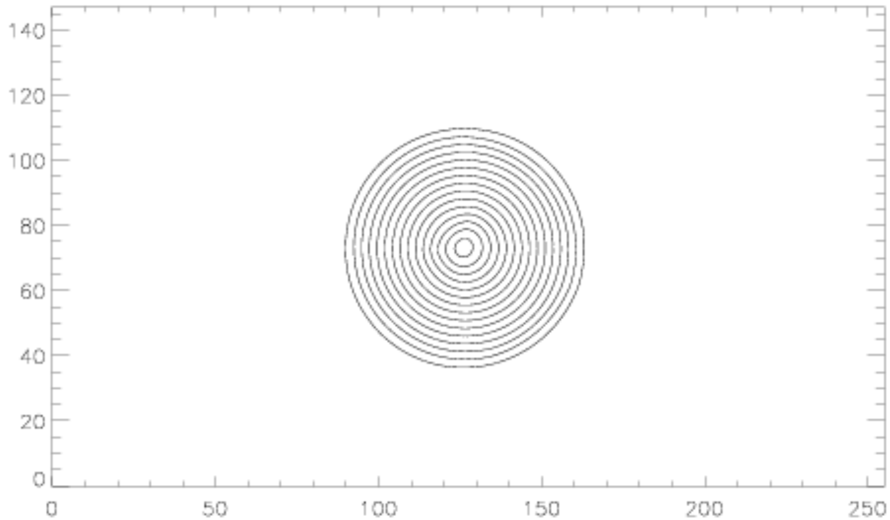
Initial:



Current density:



After two periods:



Very sensitive test of the constrained transport algorithm.

http://www.astro.princeton.edu/~jstone/Athena/tests/index.html

Code test: Orszag-Tang vortex

Standard test problem for transition from subsonic to supersonic 2D MHD turbulnce, after Orszag & Tang, 1998.

Shock-shock interaction, effect of div B error on evolution.

Add more physics

- Depending on the problem, adding more physics can require just small changes, or a complete rewrite of the algorithm.

1). Simple changes:

Adding local source terms (e.g., cooling, thermal relaxation).

2). Modest changes:

Adding flux-divergence terms (e.g., viscosity, resistivity).

Add terms requiring elliptic solvers (e.g., self-gravity).

3). Complete re-write:

Adding new dynamical equations (e.g., special/general relativity, cosmic-ray particles, radiation).

Add more physics

- Simple source terms (cases 1, 2) are usually added via operator splitting:

  For equation: $$\frac{\partial U}{\partial t} + \nabla \cdot \boldsymbol{F} = S$$

  Solve it by sequentially solving two separate equations:

  $$\frac{\partial U}{\partial t} + \nabla \cdot \boldsymbol{F} = 0 \quad \text{and} \quad \frac{\partial U}{\partial t} = S$$

  Formally, operate splitting makes the scheme first order in time.

  Higher-order accuracy can be achieved using multi-step methods (e.g., RK)

  Stability issues can be addressed using implicit methods.

- New dynamical equations (case 3) are solved separately, which then supply source terms to the MHD equations. They can be handled either by operator splitting or multi-step methods.

Example: resistivity

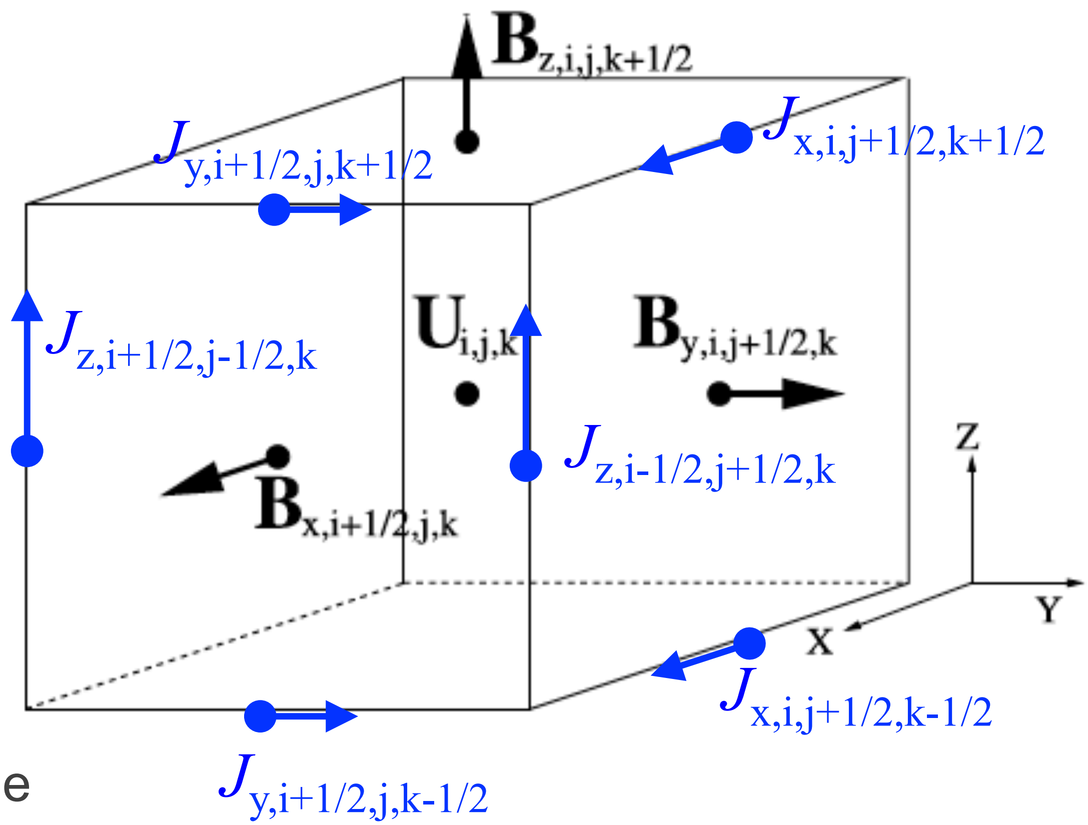
Operator split for resistivity:

$$\frac{\partial \boldsymbol{B}}{\partial t} = -\nabla \times (\eta \boldsymbol{J})$$

The overriding concern to keep div(B)=0 suggests CT differencing:

Define $\boldsymbol{J}$ at cell edges => resistive EMF = $\eta \boldsymbol{J}$.

J can be easily obtained by taking the curl face-centered B field.



CFL condition for a diffusion equation can be much more restrictive:

$$\Delta t_\eta \leq \frac{\Delta x^2}{4D\eta}$$

(D: dimension)

Timestepping issue can be alleviated using sub-cycling or super-timestepping

Other methods for computational MHD

- Finite-difference method

  Can achieve very high order accuracy for smooth flows, but requires artificial and hyper viscosity/resistivity to stabilize the code. Poor performance at strong shocks.

  Example: Pencil code

- Spectral method

  Usually for incompressible/anelastic flow (filter out sound waves). Convergence is exponential. Main application: atmosphere, stellar interior, (occasionally) accretion disks.

  Example: Snoopy, Dedalus

- Smoothed particle (magneto-)hydrodynamics

  Mesh-free Lagrangian method, mostly for hydrodynamic applications, but recent development include magnetic fields with divergence cleaning. Very flexible to handle flows with large dynamical range, but has issues dealing with shocks and turbulence.

  Example: Phantom code

- Moving-mesh/meshless MHD

  Computation based on unstructured Lagrangian points. Partition the volume and use Riemann solvers (fully conservative). Implementing CT is possible but very difficult, mostly use divergence cleaning. Reduced advection error but enhanced grid noise.

  Example: Arepo, Gizmo

Summary

- Computational MHD is an important tool to study a wide range of astrophysical plasma phenomena.
- MHD equations are hyperbolic conservation laws
- Godunov method: fully conservative

  Main idea: reconstruct-evolve-average, with proper upwinding.

  Shock capturing using Riemann solvers.

- Preserving div(B)=0 is crucial:

  Use divergence cleaning or constrained transport.

- A wide variety of MHD codes are developed, with pros and cons, and with more capabilities to handle more complex problems!