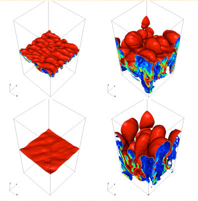
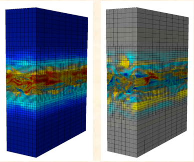
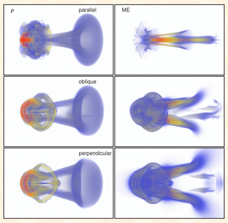
Grid-based methods for hydrodynamics, MHD, and radiation hydrodynamics.

(Four lectures)







Jim Stone
Department of Astrophysical Sciences
Princeton University

http://www.astro.princeton.edu/~jstone/downloads/papers/Lecture3.pdf

Outline of lectures

Lecture 1. Introduction to physics and numerics

Lecture 2. Operator split (ZEUS-like) methods

Lecture 3. Godunov (PPM-like) methods

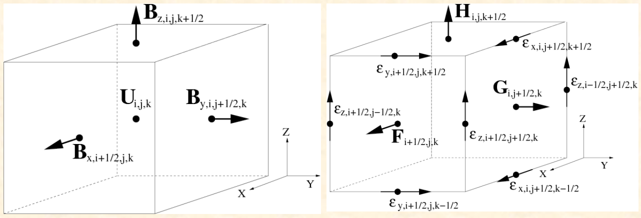
Lecture 4. Radiation Hydrodynamics

Lecture 3:
Godunov (PPM-like) methods.

1. The Godunov algorithm
   - Riemann solvers
   - Reconstruction
   - Unsplit Integrators
2. Implementation issues: the Athena code
3. Tests
4. Comparison of grid and SPH methods.
5. Galilean invariance of grid codes.

Summary of the discretization.

Uses cell-centered mass, momentum, energy; face-centered field:

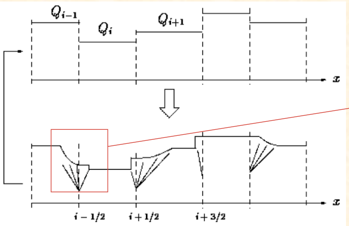

$\mathbf{B}_{z,i,j,k+1/2}$
$\mathbf{U}_{i,j,k}$
$\mathbf{B}_{y,i,j+1/2,k}$
$\mathbf{B}_{x,i+1/2,j,k}$
Z
Y
X
$\mathbf{H}_{i,j,k+1/2}$
$\varepsilon_{x,i,j+1/2,k+1/2}$
$\varepsilon_{y,i+1/2,j,k+1/2}$
$\mathbf{G}_{i,j+1/2,k}$
$\varepsilon_{z,i-1/2,j+1/2,k}$
$\varepsilon_{z,i+1/2,j-1/2,k}$
$\mathbf{F}_{i+1/2,j,k}$
$\varepsilon_{z,i+1/2,j+1/2,k}$
Z
Y
X
$\varepsilon_{x,i,j+1/2,k-1/2}$
$\varepsilon_{y,i+1/2,j,k-1/2}$

Uses face-centered fluxes, and edge-centered EMFs.

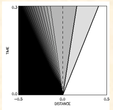
The key is how to compute these fluxes and EMFs all at once!

Godunov's orginal (first-order) method

- Difference in cell-averaged values at each grid interface define set of Riemann problems (evolution of initially discontinuous states).
- Solution of Riemann problems averaged over cell give time-evolution of cell-averaged values, until waves from one interface crosses the grid and interacts with the other, that is for $\Delta t \leq \Delta x/(v+C)$
- Due to conservation, don't actually need to solve Riemann problem exactly. Just need to compute state *at location of interface* to compute fluxes.



Flux given by solution along x=0



Then, solution evolved according to

$$\frac{\mathbf{U}_j^{n+1} - \mathbf{U}_j^n}{\Delta t} = -\frac{\mathbf{F}_{j+1/2}^{n+1/2} - \mathbf{F}_{j-1/2}^{n+1/2}}{\Delta x}$$

Riemann solvers

For pure hydrodynamics of ideal gases, exact/efficient nonlinear Riemann solvers are possible.

In MHD, nonlinear Riemann solvers are complex because:

1. There are 3 wave families in MHD – 7 characteristics
2. In some circumstances, 2 of the 3 waves can be degenerate (e.g. $V_{Alfven} = V_{slow}$)

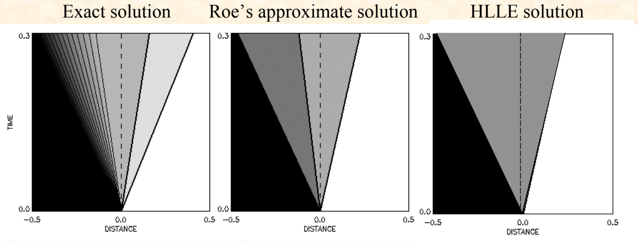
Equations of MHD are not *strictly hyperbolic*
(Brio & Wu, Zachary & Colella)

Thus, in practice, MHD Godunov schemes use approximate and/or linearized Riemann solvers.

Many different approximations are possible:

1. Roe's method – keeps all 7 characteristics, but treats each as a simple wave.
   Good resolution of all waves
   Requires characteristic decomposition in conserved variables
   Expensive and difficult to add new physics
   Fails for strong rarefactions
2. Harten-Lax-van Leer-Einfeldt (HLLE) method – keeps only largest and smallest characteristics, averages intermediate states in-between.
   Very simple and efficient
   Guarantees positivity in 1D
   Very diffusive for contact discontinuities
3. HLLC(HLLD) methods – Adds entropy (and Alfven) wave back into HLLE method, giving two (four) intermediate states.
   Reasonably simple and efficient
   Guarantees positivity in 1D
   Better resolution of contact discontinuities

Effect of various approximations on the solution to the Riemann problem in hydrodynamics

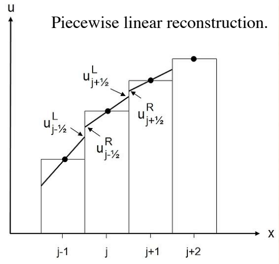


So which approximation is "best"? Must explore the use of each.

Use of a Riemann solvers is a benefit, not a weakness, of a Godunov method: makes shock capturing more accurate.

Higher-order reconstruction

- Using cell-centered values for left- and right-states to define Riemann problems at cell interfaces is first-order and very diffusive.
- Higher-order methods use piecewise linear (MUSCL) or piecewise-parabolic (PPM) reconstruction within cells.
- Difference between L/R states is small for smooth flow, large near shocks. Riemann solver automatically gives correct dissipation for shocks. *No artificial viscosity is needed.*



Time advance of L/R-states required.

(see Gardiner & Stone, JCP, 2005)

For second- (or higher-) order *single-step methods*, the L/R states must be evolved for Δt/2 so that the Riemann solver computes the correct time-averaged fluxes.

Characteristic tracing in the primitive variables can be used for this step

$$\mathbf{w}_{L,i+1/2} = \hat{\mathbf{w}}_{L,i+1/2} + \frac{\delta t}{2\delta x} \sum_{\lambda^\alpha > 0} \left((\lambda_i^M - \lambda_i^\alpha) \mathsf{L}^\alpha \cdot \delta \mathbf{w}_i^m \right) \mathsf{R}^\alpha$$

$$\mathbf{w}_{R,i-1/2} = \hat{\mathbf{w}}_{R,i-1/2} + \frac{\delta t}{2\delta x} \sum_{\lambda^\alpha < 0} \left((\lambda_i^0 - \lambda_i^\alpha) \mathsf{L}^\alpha \cdot \delta \mathbf{w}_i^m \right) \mathsf{R}^\alpha$$

λ=eigenvalues
L,R=eigenvectors of linearized system
δ**w**^{m}=monotonized differences

This is essentially a directionally-split time-advance. In multi-dimensions, source terms must be added to account for evolution of the longitudinal component of B.

e.g., in 2D source terms are

$$\delta B_{y,L,i-1/2,j} = \frac{\delta t}{2\delta x} v_{y,i-1,j} \left(B_{x,i-1/2,j} - B_{x,i-3/2,j} \right)$$

$$\delta B_{x,L,i,j-1/2} = \frac{\delta t}{2\delta y} v_{x,i,j-1} \left(B_{y,i,j-1/2} - B_{y,i,j-3/2} \right)$$

Alternatively, second-order temporal evolution can be achieved by *multi-step* methods (e.g. RK integration)

van Leer unsplit integrator.

Stone & Gardiner, NewA, 2009

- For multidimensional hydrodynamics, directional splitting can be used.
- For MHD, unsplit integrators are necessary if the conservative form is adopted.
- Simplest integrator: modified MUSCL-Hancock ("van Leer") method due to Falle (1991).

Steps in algorithm

1. Compute first-order fluxes at every interface
2. Use these fluxes to advance solution for Δt/2 (predict step)
3. Compute L/R states using time-advanced state, and compute fluxes
4. Advance solution over full time step (correct step) using new fluxes

Since this is a multi-step method, time-advance of L/R states (characteristic tracing) is NOT needed in reconstruction step.

This greatly simplifies algorithm, and makes it much easier to extend to multi-physics, since characteristic decomposition of linearized equations not needed.

Corner transport upwind (CTU) integrator

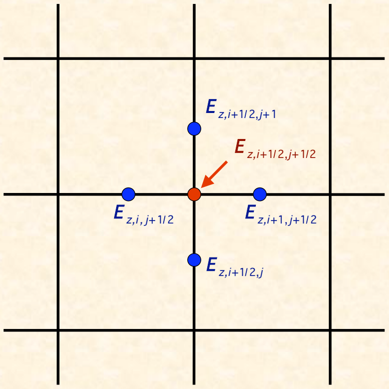
A more accurate unsplit integrator is due to Colella (1990), extended to MHD by Gardiner & Stone (2005; 2008)

Steps in algorithm:

1. Compute L/R states including time advance using characteristic tracing and source terms for multi-dimensional MHD
2. Compute fluxes from Riemann solver
3. Correct L/R states with transverse flux gradients for Δt/2 including source terms for MHD, e.g. in 2D x-face states corrected via:
$$\mathbf{q}^{n+1/2}_{L,i-1/2,j} = \mathbf{q}_{L,i-1/2,j} + \frac{\delta t}{2\delta y} \left(\mathbf{g}^*_{i-1,j+1/2} - \mathbf{g}^*_{i-1,j-1/2} \right) + \frac{\delta t}{2} \mathbf{s}_{x,i-1,j}$$
$$\mathbf{q}^{n+1/2}_{R,i-1/2,j} = \mathbf{q}_{R,i-1/2,j} + \frac{\delta t}{2\delta y} \left(\mathbf{g}^*_{i,j+1/2} - \mathbf{g}^*_{i,j-1/2} \right) + \frac{\delta t}{2} \mathbf{s}_{x,i,j}$$
4. Compute multi-dimensional fluxes from corrected L/R states
5. Advance solution full time step using multi-dimensional fluxes

see Stone et al, APJS, 2008

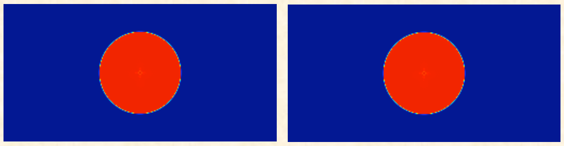
The CT Algorithm in 2D



- Finite Volume / Godunov algorithm gives E-field at face centers.
- "CT Algorithm" needs E-field at grid cell corners.
- Arithmetic averaging: 2D plane-parallel flow does not reduce to equivalent 1D problem
- Algorithms which reconstruct E-field at corner are superior Gardiner & Stone 2005

Advection of a field loop (2N x N grid)

Field Loop Advection ($\beta = 10^6$): MUSCL - Hancock integrator
Movies of B^2



Arithmetic average (Balsara & Spicer 1999) Gardiner & Stone 2005

Good test of stability of CT algorithm (obviously trivial for vector potential approaches)

Good test of whether codes preserves div(B) on appropriate stencil: Run in 3D with non-zero V_z. Does method keep B_z zero?

Other ways to keep div(B)=0 in Godunov codes

1. Do nothing. Assume errors remain small and bound.
   You know what happens when you assume.
2. Evolve B using vector potential A, where $\mathbf{B} = \nabla \times \mathbf{A}$
   Requires taking second difference numerically to compute Lorentz force
3. Remove solenoidal part of B using "flux-cleaning". That is, set $\mathbf{B} \rightarrow \mathbf{B} - \nabla\phi$ where $\nabla^2\phi + \nabla\cdot\mathbf{B} = 0$
   Requires solving elliptic PDE every timestep – expensive
   May smooth discontinuities in B
4. Use Powell's "8-wave solver"
   Gives wrong jump conditions for some shock problems
5. Evolve integral form of induction equation so as to conserve magnetic flux (constrained transport).
   Requires staggered grid for B (although see Toth 2000)

Carbuncle instability

Quirk 1994, Sutherland et al 2003

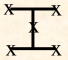
Small perturbations in upstream flow produce large perturbations in postshock gas.

For grid aligned shocks, transverse dissipation is too small to damp perturbations. Transverse pressure gradient produces flow which amplifies perturbations in shocks: *carbuncle instability*

Solution, increase dissipation in transverse direction for grid-aligned shocks, e.g. using **H-correction** in Roe solver:

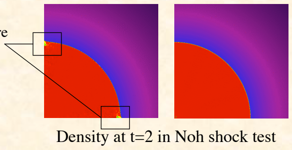
Replace eigenvalues (wave speeds) in Roe's linearization with $|\bar{\lambda}^\alpha| = \max(|\lambda^\alpha|, \bar{\eta}_{i-1/2,j})$.
where $\bar{\eta}_{i-1/2,j} = \max(\eta_{i-1,j+1/2}, \eta_{i-1,j-1/2}, \eta_{i-1/2,j}, \eta_{i,j+1/2}, \eta_{i,j-1/2})$

$$\eta_{i-1/2,j} = \frac{1}{2}|(u_{i,j} + C_{f,i,j}) - (u_{i-1,j} - C_{f,i-1,j})|$$



Test with Noh shocktube (M=10^6 converging flow)

Carbuncle in regions where shock aligned with grid

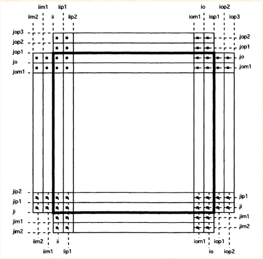


With H-correction, carbuncle is fixed.

Density at t=2 in Noh shock test

Boundary conditions

As in ZEUS, BCs are applied by specifying solution in "ghost zones"



Several default choices for BCs provided, e.g.

1. Reflecting
2. Inflow
3. Outflow
4. Periodic

- Unsplit integrator with PPM requires 4 rows of ghost zones.
- BCs applied only once per time step -- more efficient parallelization.
- New user-defined BCs easily added through use of function pointers.

Athena: one implementation of a MHD Godunov scheme.

- Two versions: C (most capable) and F90 (cleanest)
  http://www.astro.princeton.edu/~jstone/athena.html for C version
  http://www.astro.virginia.edu/VITA/athena.php for F90 version
- Modularity: makes extensions to code easier
  - Riemann solvers, reconstruction algorithms, unsplit integrators all separate functions with common interface.
- Ease-of-use:
  - configure in C, modules in F90
  - flexible variety of output files (that don't depend on external libraries!)
  - Input files have intuitive format enabled by special-purpose parser.
- Portability ensured by:
  - Strict adherence to ANSI standards (don't use language extensions!)
  - No reliance on external libraries (except when absolutely necessary, e.g. parallelization with MPI)
- Performance: unsplit integrators require large number (~100) of 3D scratch arrays, however method is so expensive (~10^4 flops per cell) the overall method is cpu, not memory, bound.

Configure provides a very useful way to control physics and algorithm options before compiling. Usage: `configure [--with-package=choice] [--enable-feature]`

The -c command-line option enables output of configuration details from executable:

```
ophir> athena -c

Configuration details:

 Problem:                    linear_wave1d
 Gas properties:             MHD
 Equation of State:          ADIABATIC
 Passive scalars:            0
 Self-gravity:               OFF
 Ohmic resistivity:          OFF
 Viscosity:                  OFF
 Thermal conduction:         OFF
 Particles:                  OFF
 Coordinate System:          Cartesian
 Special Relativity:         OFF
 Ionizing radiation:         OFF
   Ionizing point sources:   OFF
   Ionizing plane sources:   OFF
 Order of Accuracy:          2 (SECOND_ORDER_CHAR)
 Flux:                       roe
 Unsplit integrator:         ctu
 Precision:                  DOUBLE_PREC
 Ghost cell Output:          OFF
 Parallel Modes: MPI:        OFF
 H-correction:               OFF
 FFT:                        OFF
 Shearing Box:               OFF
 FARGO:                      OFF
 All-wave integration:       OFF
```

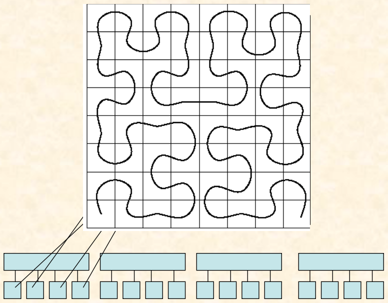
Parallelization

1. Parallelization with MPI via domain decomposition.
   - Any arbitrary decomposition in X, Y, or Z possible (blocks are best for large N_p)
   - Can compute optimum decomposition to minimize data communicated automatically for given N_p
   - No diagonal communication required if data swapped sequentially in each direction.
   - Ideal MPI blocksize seems to be 64^3 on current processors.
2. Balancing workload is easy since flops/zone fixed.
3. Can overlap work and communication by updating outer zones in MPI block first.
4. Tried OpenMP on multi-core, and find it does not perform any better then pure MPI (but saves some memory).
5. FFTs parallelized using block (not just slab) decomposition using Steve Plimpton's interface to FFTW.

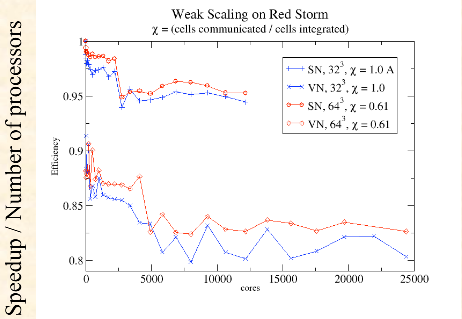
Domain decomposition on multi-core processors to ensure locality

Best to map blocks of MPI domains to cores on processors, rather than using linear ordering along dimensions.

Equivalent to using Peano-Hilbert ordering for space-filling curve.



Weak scaling of Athena is very good, since it is all just explicit MHD (nearest-neighbor communications).



2700^3 grid; 2×10^9 grid cell updates per second

Bug tracking.

We are now using Trac+SVN to manage software development, the Trac website will go live soon (for now access is by permission only).

See https://trac.princeton.edu/Athena

Site contains documentation, milestones, bug tickets, and ability to browse SVN repository.

Some Tests

Five test problems we have found very useful (all drawn from basic physics of fluids studied in Lecture 1):

1. Linear wave convergence
2. Nonlinear circularly polarized Alfven waves
3. Brio & Wu, and Ryu-Jones shocktubes
4. Field loop advection
5. MHD instabilities (KH, RT, MRI, etc.)

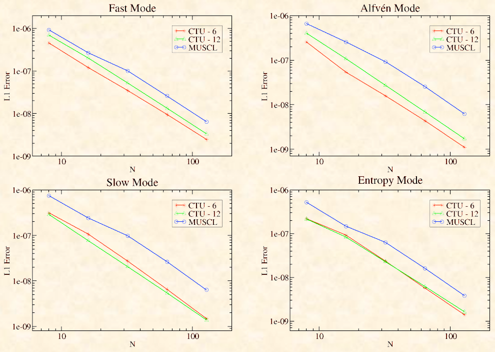
For MHD, ***must*** focus on multidimensional tests.

Convergence rate and ability to capture shocks are equally important.

See http://www.astro.princeton.edu/~jstone/athena.html

Linear Wave Convergence: 3D (2N x N x N) grid

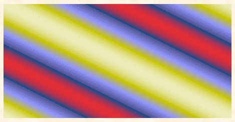
Initialize pure eigenmode for each wave family
Measure RMS error in U after propagating one wavelength
quantitative test of accuracy of scheme

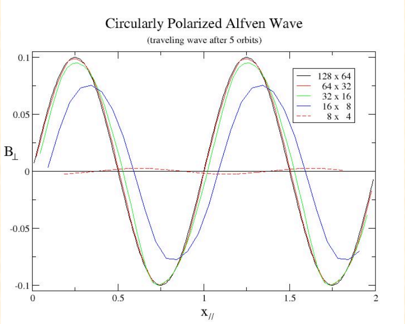


Circularly Polarized Alfvén Wave (2N x N grid)

$\beta \approx 0.2$, wave amp. = 0.1 (Toth 2000)

$L_x = 2L_y$, $\delta x = \delta y$, wave propagates at $\tan^{-1}\Theta = 1/2$

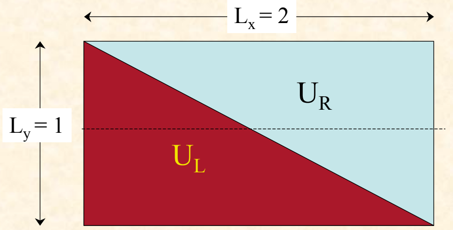




Scatter plot showing *all* grid points - no parametric instability present

RJ2a Riemann problem rotated to grid

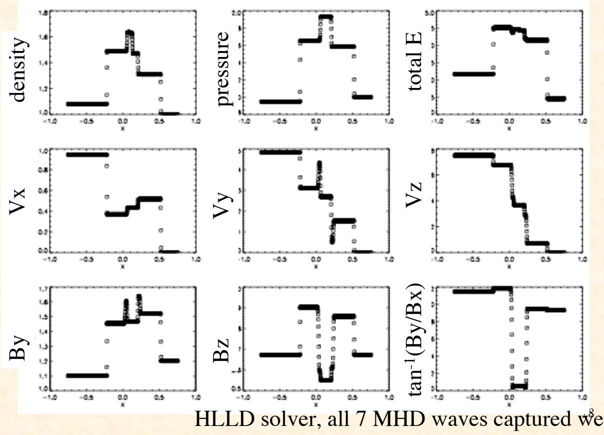
Initial discontinuity inclined to grid at $\tan^{-1}\theta = 1/2$
Magnetic field initialized from vector potential to ensure div(B)=0
$\Delta x = \Delta y$, 512 x 256 grid



Problem is Fig. 2a from Ryu & Jones 1995

Final result plotted along horizontal line at center of grid

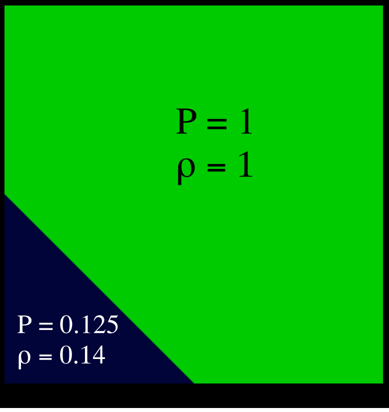
RJ2a shocktube in 3D (2N x N x N grid)



HLLD solver, all 7 MHD waves captured well.

Hydrodynamical Implosion

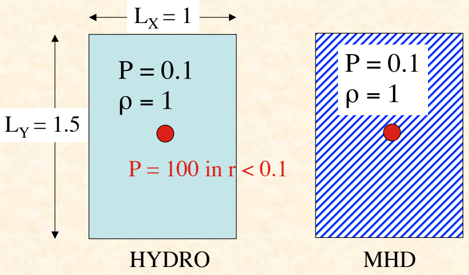
From Liska & Wendroff; 400 x 400 grid,



Additional benefit of using unsplit integration scheme: Code maintains symmetry

RM instability in Spherical Blast Waves

Impulsive acceleration of a dense fluid by a less-dense fluid (e.g. by a shock propagating across a CD) is subject to RT-like instability. Algebraic rather than exponential growth.



B at 45 degrees, $\beta = 0.1$

$\Delta x = \Delta y$, 400 x 600 grid, *periodic* boundary conditions

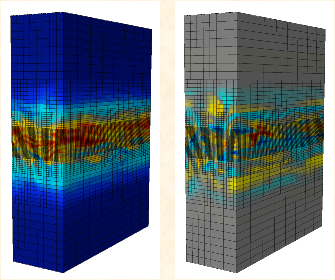
Hydrodynamic Blast Wave
400 x 600 grid

Compare to Fig. 23 in Springel (2009)

MHD Blast Wave
400 x 600 grid

Static mesh refinement.

Uses divergence-free prolongation and restriction operators of Toth & Roe (2002). Requires embedding flux-correction algorithm at fine/coarse boundaries in CTU+CT integrator to preserve $\nabla\cdot\mathbf{B} = 0$.



Density

Angular momentum fluctuations

Static nested-grids are ideal to refine midplane in MHD studies of thin disks.

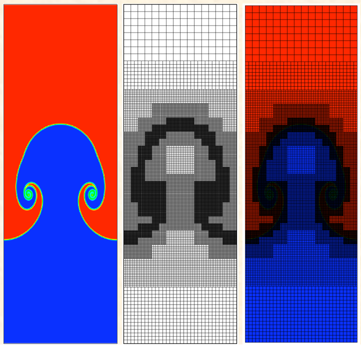
Extensions to Athena: AMR

Uses same prolongation and restriction operators (Toth & Roe 2002) and flux-correction algorithm at fine/coarse boundaries as SMR. Grid generation and destruction handled by in-house functions (Gardiner).

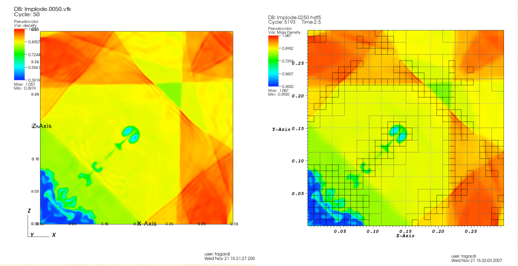
Results from growth of MHD RT instability in 2D, single-mode perturbation.

Five levels of refinement.

Code runs 30 faster than than equivalent single-mesh calculation.



Another example of AMR in Athena:
hydrodynamic implosion test of Liska & Wendroff.

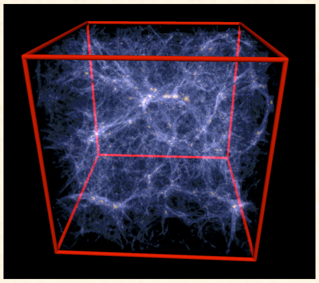


Fixed grid *(left)*, AMR with same effective resolution *(right)*

Full disclosure: neither SMR nor AMR in Athena is parallelized (yet).

Comparison of grid codes to SPH.

SPH is an extremely useful tool for studying hydrodynamical flows with gravity (e.g. cosmological structure formation)

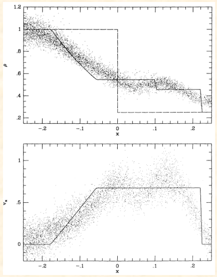


Springel & Hernquist (2003)

Achieving the same dynamic range in grid codes is very challenging.

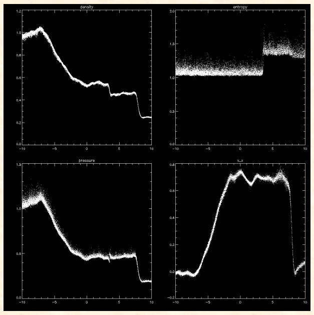
But, for studying basic fluid dynamical processes *without gravity*, grid codes are often better.

e.g., Sod shock test (run in 3D) with SPH using random particle positions initially



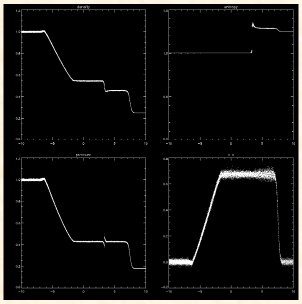
Rasio & Shapiro 1991

N=10^5, showing all particles



State-of-the-art code (Gadget-2) gives similar result (image courtesy V. Springel).

Results considerably improved by using *glass* as initial condition.

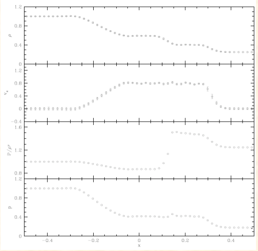


N=10^5, showing all particles

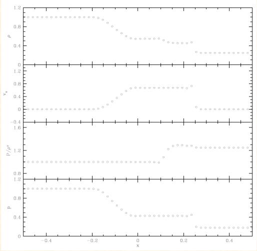
3D Sod shock test using Gadget-2 and glass for initial particle positions (image courtesy V. Springel).

But glass is not a good description of how particles will be distributed in a real application.

Grid codes can capture shocks in 3D in less time.



Gadget-2, 10^6 particles starting as glass, solution binned into cells



Athena, 50^3 grid, shock run at oblique angle to grid, showing *all* grid points.

Gadget-2 with 10^6 particles takes several cpu hours for this test.
Athena with 50^3 grid takes 1 cpu minute.

Decay of MHD turbulence.

A recent code comparison test of the decay rate of supersonic MHD turbulence (Kritsuk et al. 2009) shows SPH is very dissipative for this problem.

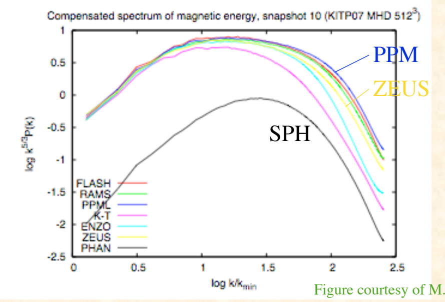


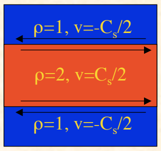
Figure courtesy of M. Norman

Taken together, these results suggest:

- SPH is best for problems where gravity dominates,
- grid-based codes are more cost-effective for basic fluid dynamic problems in most (but not necessarily all!) cases.

Galilean invariance of grid codes

Recently, Springel (2009) has shown grid codes produce different results when the calculations are run in different frames of reference (by adding constant background velocity).



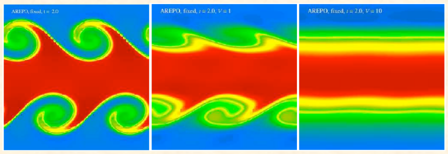
ρ=1, v=-C_s/2

ρ=2, v=C_s/2

ρ=1, v=-C_s/2

e.g. Kelvin-Helmholtz instability between different density shear layers

Add long wavelength perturbation V_y=sin(4πx/L_x)



V_0/C_s = 0 V_0/C_s = 1 V_0/C_s =10

Why aren't grid codes Galilean invariant?

Recall 1D x-flux of conserved variables is $\mathbf{F} = \begin{bmatrix} \rho v_x \\ \rho v_x^2 + P \\ \rho v_x v_y \\ \rho v_x v_z \\ (E+P)v_x \end{bmatrix}$,

So fluxes are not Galilean invariant.

Numerical algorithm approximates fluxes to some order. Truncation error of this approximation cannot be Galilean invariant.

By studying slip-surfaces with contact discontinuities and no viscosity or surface tension, there is no minimum scale in the problem. Solutions are unresolved (strongly affected by truncation error at grid scale).

Boosting solution to different velocities by adding constant velocity affects truncation error, and so changes solution.

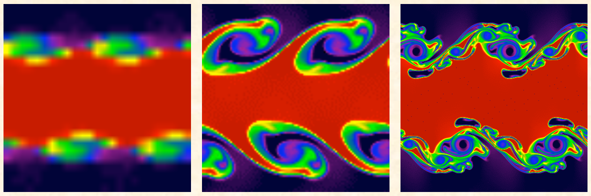
Because solution is unresolved, can get same effect just by varying numerical resolution.

Changing numerical resolution affects solution in the same way.

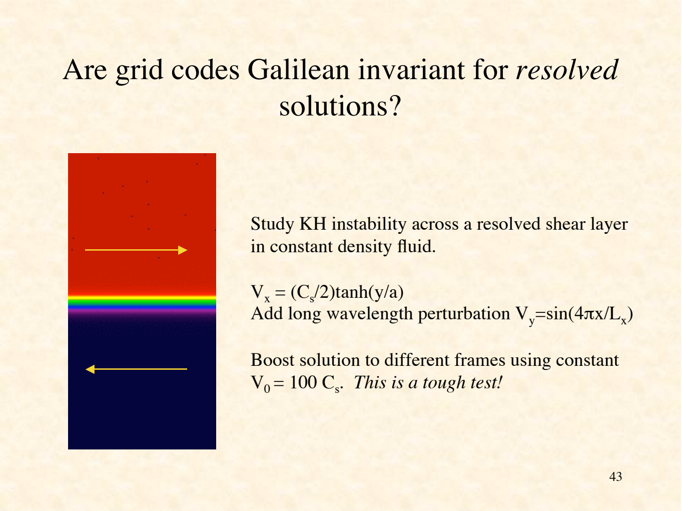
Repeat exactly same test with V_0=0 and different numerical resolutions. Solutions are vastly different.

Which one is correct?

In other words, which solution should the calculation in the V_0=100C_s frame agree with?



25^2 100^2 400^2

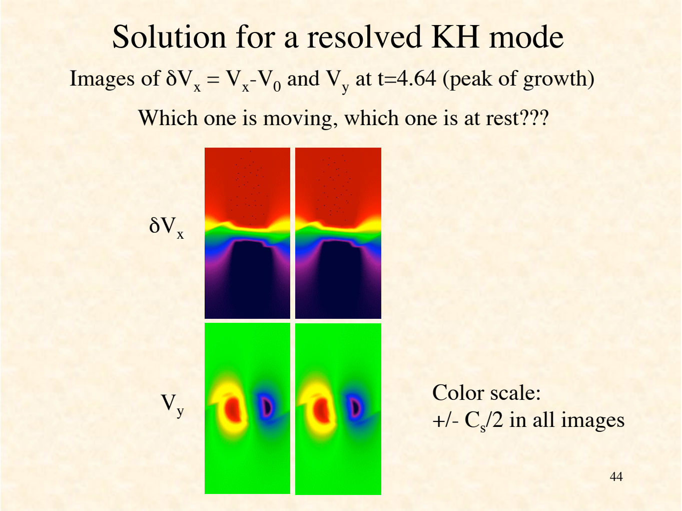


Are grid codes Galilean invariant for *resolved* solutions?

Study KH instability across a resolved shear layer in constant density fluid.

$V_x = (C_s/2)\tanh(y/a)$
Add long wavelength perturbation $V_y=\sin(4\pi x/L_x)$

Boost solution to different frames using constant $V_0 = 100\ C_s$. *This is a tough test!*

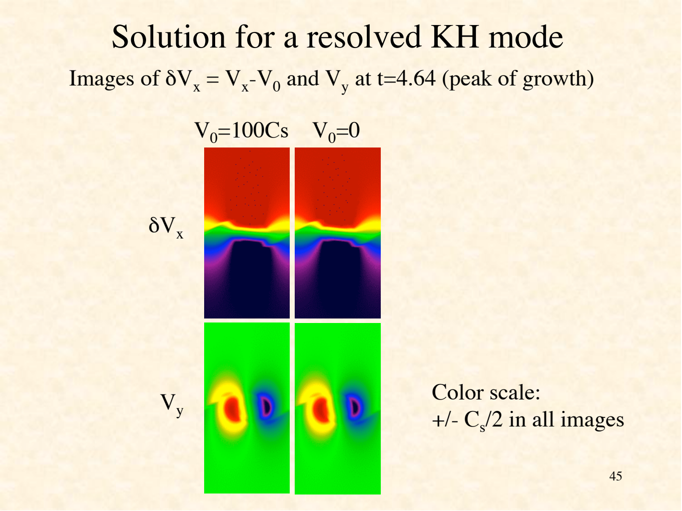


Solution for a resolved KH mode

Images of $\delta V_x = V_x\text{-}V_0$ and V_y at t=4.64 (peak of growth)

Which one is moving, which one is at rest???

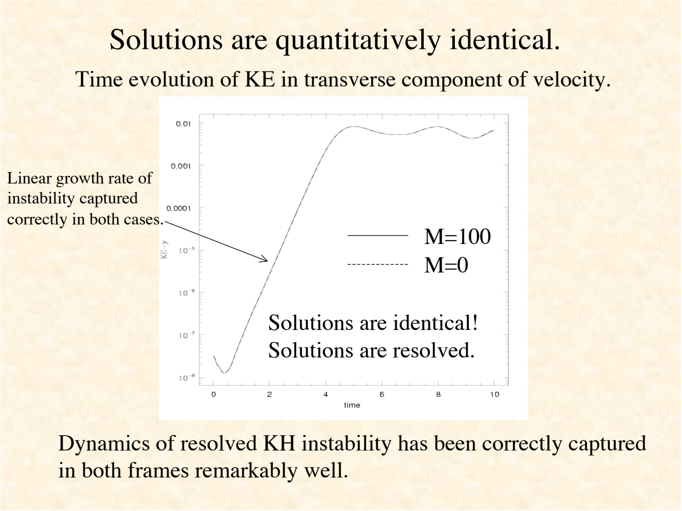
Color scale:
+/- $C_s/2$ in all images



Solution for a resolved KH mode

Images of $\delta V_x = V_x\text{-}V_0$ and V_y at t=4.64 (peak of growth)

Color scale:
+/- $C_s/2$ in all images



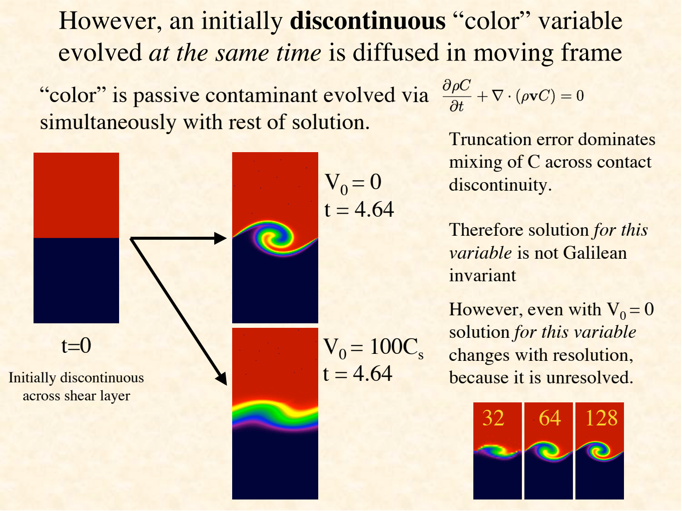
Solutions are quantitatively identical.

Time evolution of KE in transverse component of velocity.

Linear growth rate of instability captured correctly in both cases.

Solutions are identical!
Solutions are resolved.

Dynamics of resolved KH instability has been correctly captured in both frames remarkably well.



However, an initially **discontinuous** "color" variable evolved *at the same time* is diffused in moving frame

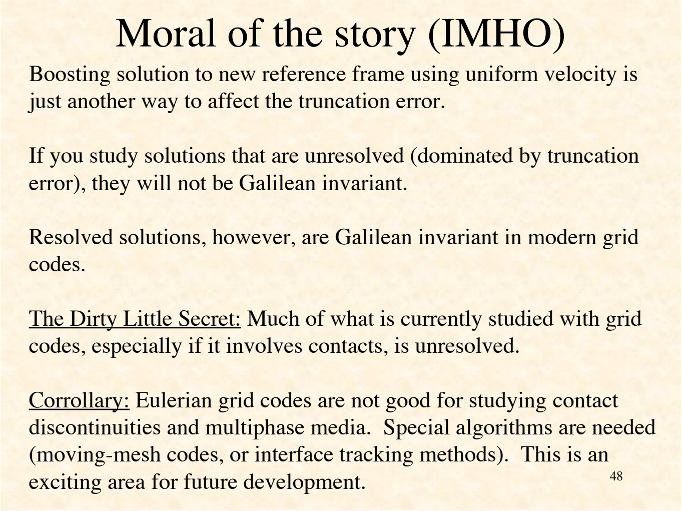
"color" is passive contaminant evolved via $\frac{\partial \rho C}{\partial t} + \nabla \cdot (\rho \mathbf{v} C) = 0$ simultaneously with rest of solution.

Initially discontinuous across shear layer

Truncation error dominates mixing of C across contact discontinuity.

Therefore solution *for this variable* is not Galilean invariant

However, even with $V_0 = 0$ solution *for this variable* changes with resolution, because it is unresolved.



Moral of the story (IMHO)

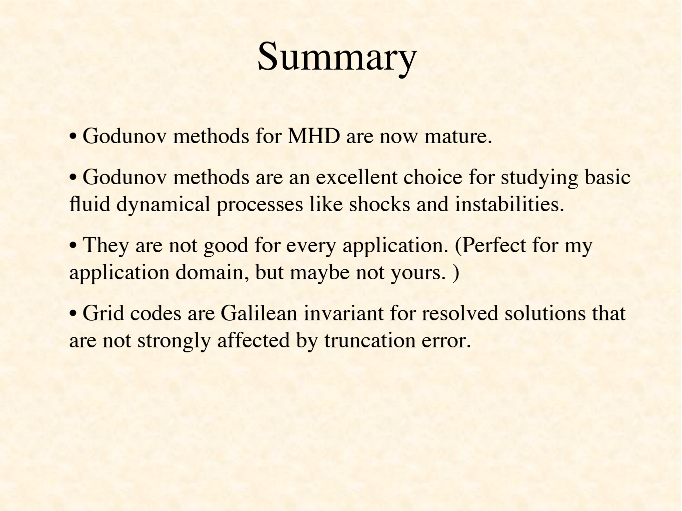
Boosting solution to new reference frame using uniform velocity is just another way to affect the truncation error.

If you study solutions that are unresolved (dominated by truncation error), they will not be Galilean invariant.

Resolved solutions, however, are Galilean invariant in modern grid codes.

The Dirty Little Secret: Much of what is currently studied with grid codes, especially if it involves contacts, is unresolved.

Corrollary: Eulerian grid codes are not good for studying contact discontinuities and multiphase media. Special algorithms are needed (moving-mesh codes, or interface tracking methods). This is an exciting area for future development.



Summary

- Godunov methods for MHD are now mature.
- Godunov methods are an excellent choice for studying basic fluid dynamical processes like shocks and instabilities.
- They are not good for every application. (Perfect for my application domain, but maybe not yours.)
- Grid codes are Galilean invariant for resolved solutions that are not strongly affected by truncation error.