Computational Astrophysics:

Differential Equations

Adaptive Mesh Refinement

Troels Haugbølle

Niels Bohr Institute and STARPLAN
University of Copenhagen

Topics

- Numerical solutions: finite difference / finite volume
- The integral equations for finite volume
- Godunov method and higher order space and time updates
- Adaptive Mesh Refinement

Numerical Solutions to Differential Equations

- Partial differential equations come in three different types

  - Hyperbolic: Solution depends on the initial value

$$\frac{\partial q(x,t)}{\partial t} + A\frac{\partial q(x,t)}{\partial x} = 0$$

  - Elliptic: Solution depends on the boundary values

$$\nabla^2\phi = 4\pi G\rho$$

  - Parabolic equations: A mixture of the two

- Today we will be concerned with the first type. In a general physics problem, the system of equations will contain all types

Numerical Solutions to Differential Equations

- To solve differential equations; for example the advection equation

$$\frac{\partial q(x,t)}{\partial t} + A\frac{\partial q(x,t)}{\partial x} = 0$$

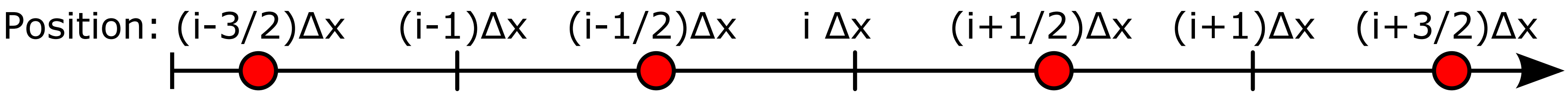
there are two popular approaches:

Finite Difference and ***Finite Volume*** methods

- While related, the mathematical theories behind the two techniques are very different

Finite Difference Method

- Assume the solution is known ("sampled") at a distinct set of points:



- ***$q(x_i,t)$*** is the value at each point ***$x_i = (i+1/2)\Delta x$*** at time ***t***

- Derivatives in time and space are approximated by differences:

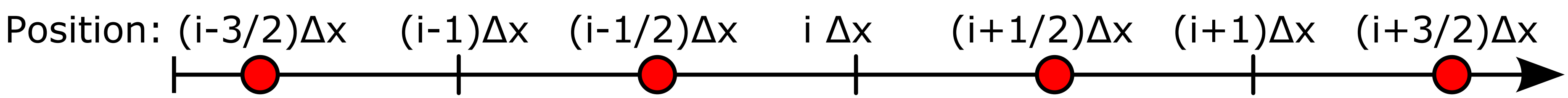
$$\left.\frac{\partial q(x,t)}{\partial x}\right|_{x=x_i} \rightarrow \frac{q(x_i+\Delta x,t)-q(x_i,t)}{\Delta x}$$

- Example: *a first order in time, second order in space* approximation

$$\frac{\rho^{i,j,k}(t+\Delta t)-\rho^{i,j,k}(t)}{\Delta t} = -\frac{\rho u_x^{i+1,j,k}(t)-\rho u_x^{i-1,j,k}(t)}{2\Delta x} - \ldots$$

Finite Difference Method

- Assume the solution is known ("sampled") at a distinct set of points:



- $q(x_i,t)$ is the value at each point $x_i = (i+1/2)\Delta x$ at time t

- Derivatives in time and space are approximated by differences:

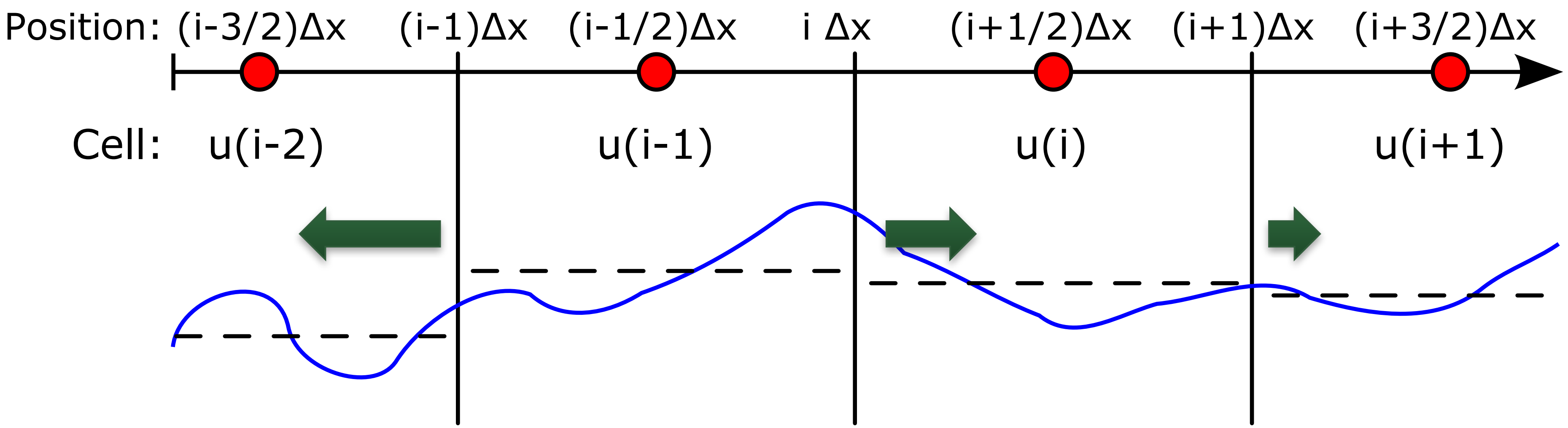
$$\left.\frac{\partial q(x,t)}{\partial x}\right|_{x=x_i} \rightarrow \frac{q(x_i+\Delta x,t)-q(x_i,t)}{\Delta x}$$

- The advantage of finite difference methods is that they are conceptually simple, and very fast. For smooth flows, high order methods can be extremely precise. For non-smooth flows, viscosity has to added by hand

- The disadvantage is that they do not always respect the properties of the equations, because they consider point values

Finite Volume Method

- In the finite volume method the **fundamental variable is the *volume average*** of the function inside a cell:



- ***u(x$_i$,t)*** is the average value in the interval [$\boldsymbol{x_{i-1/2}}$, $\boldsymbol{x_{i+1/2}}$] at time ***t***

$$u(x_i,t) = \frac{1}{\Delta x}\int_{x_i-\frac{1}{2}}^{x_i+\frac{1}{2}} q(x,t)\,dx$$

- To find the solution to the volume average we have to consider the ***flux through the surface*** of each cell

Finite Volume Method – Evolution on Integral Form

- To find the evolution of the *volume average* we integrate the differential equation:

$$\int_t^{t+\Delta t} dt \int_{x_i-\frac{1}{2}}^{x_i+\frac{1}{2}} dx \frac{\partial q}{\partial t} + \frac{\partial F}{\partial x} = 0, \qquad F(x,t) = Aq(x,t)$$

- $\boldsymbol{F_{i(+1)}=F\,(x_{i\pm1/2},t)}$ is called the flux
- We can do the spatial integral to find

$$\int_t^{t+\Delta t} dt \Delta x \frac{\partial u}{\partial t} + (F_{i+1} - F_i) = 0$$

- Finally doing the time-integral we find

$$u(t+\Delta t) - u(t) = -\frac{\Delta t}{\Delta x}(\tilde{F}_{i+1} - \tilde{F}_i)$$

where $\tilde{F}_i(t+\Delta t/2) = \frac{1}{\Delta t}\int_t^{t+\Delta t} dt\; F_i$ is the time-averaged flux

General Finite Volume Method – an excursion

- In general we can imagine a problem that is written as:

$$\frac{\partial q(x,t)}{\partial t} + \frac{\partial F(q,t)}{\partial x} = S(q,x,t)$$

- The solution to the evolution will be the result of **fluxes *F*** moving things around, while **sources *S*** are changing the values inside the cells:

$$u(x,t+\Delta t) - u(x,t) = \Delta t\left[\tilde{S}_i(t+\Delta t/2) - \frac{\tilde{F}_{i+1}(t+\Delta t/2) - \tilde{F}_i(t+\Delta t/2)}{\Delta x}\right]$$

where the ***time averaged flux*** and **time and space averaged source** are:

$$\tilde{F}_i(t+\Delta t/2) = \frac{1}{\Delta t}\int_t^{t+\Delta t} dt\ F_i\ , \quad \tilde{S}_i(t+\Delta t/2) = \frac{1}{\Delta x \Delta t}\int_t^{t+\Delta t}\int_x^{x+\Delta x} dt\,dx\ S(x,t)$$

General Finite Volume Method – an excursion

- In general we can imagine a problem that is written as:

$$\frac{\partial q(x,t)}{\partial t} + \frac{\partial F(q,t)}{\partial x} = S(q,x,t)$$

- The solution to the evolution will be the result of **fluxes *F*** moving things around, while **sources *S*** are changing the values inside the cells:

$$u(x,t+\Delta t) - u(x,t) = \Delta t \left[\tilde{S}_i(t+\Delta t/2) - \frac{\tilde{F}_{i+1}(t+\Delta t/2) - \tilde{F}_i(t+\Delta t/2)}{\Delta x} \right]$$

- **Fluxes *F*** are related to conserved quantities, while **sources *S*** corresponds to the creation, destruction or transfer of a quantity. Examples are
  - Mass, momentum and total energy of a system (fluxes)
  - Energy cooling and heating (sources); Gravitation (source or flux!)
  - Geometric source terms (e.g. in a spherical coordinate system)

Finite Volume Method – Evolution equation

- The integral evolution equation of the *volume average*

$$u(t+\Delta t)-u(t)=\frac{\Delta t}{\Delta x}(\tilde{F}_{i+1}-\tilde{F}_{i})$$

is exact.

- ***Derivatives*** are converted into ***differences***
  - This is well suited for numerical evaluation
  - The absence of partial derivatives means the equations are well defined even for discontinuous functions

Finite Volume Method

- The problem is to find an expression for the time averaged fluxes

$$\tilde{F}_i(t+\Delta t/2) = \frac{1}{\Delta t}\int_t^{t+\Delta t} dt\ F_i(x_{i-1/2},t)$$

- Problems:
  - The flux is calculated from the actual point values ***q*** at the interface, not the cell-averaged values ***u***.
  - We need to approximate the time integral.

- Solutions:
  - We need to reconstruct the value at the interface based on the cell average. This is called ***slope reconstruction***.
  - For the time evolution we can use either ***implicit methods*** (difficult) or some kind of ***predictor-corrector*** scheme.

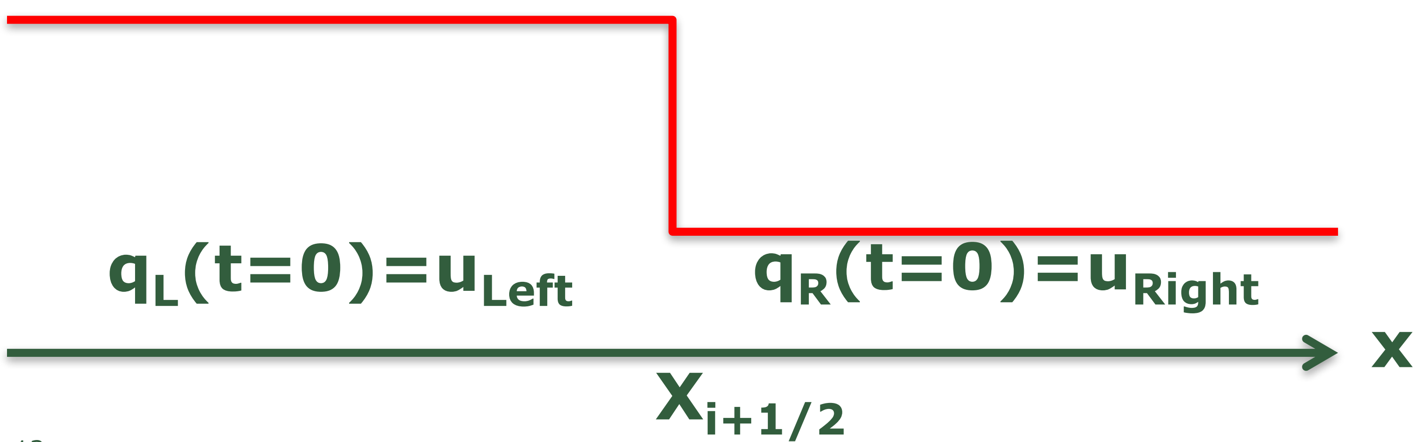
The Riemann Problem

- The problem is to find an expression for the time averaged fluxes

$$\tilde{F}_i(t+\Delta t/2)=\frac{1}{\Delta t}\int_{t}^{t+\Delta t} dt\; F_i(x_{i-1/2},t)$$

- To the very lowest order we could approximate the solution inside the cell to be constant

- What is $\mathbf{q(x_{i+1/2},t)}$ then ???



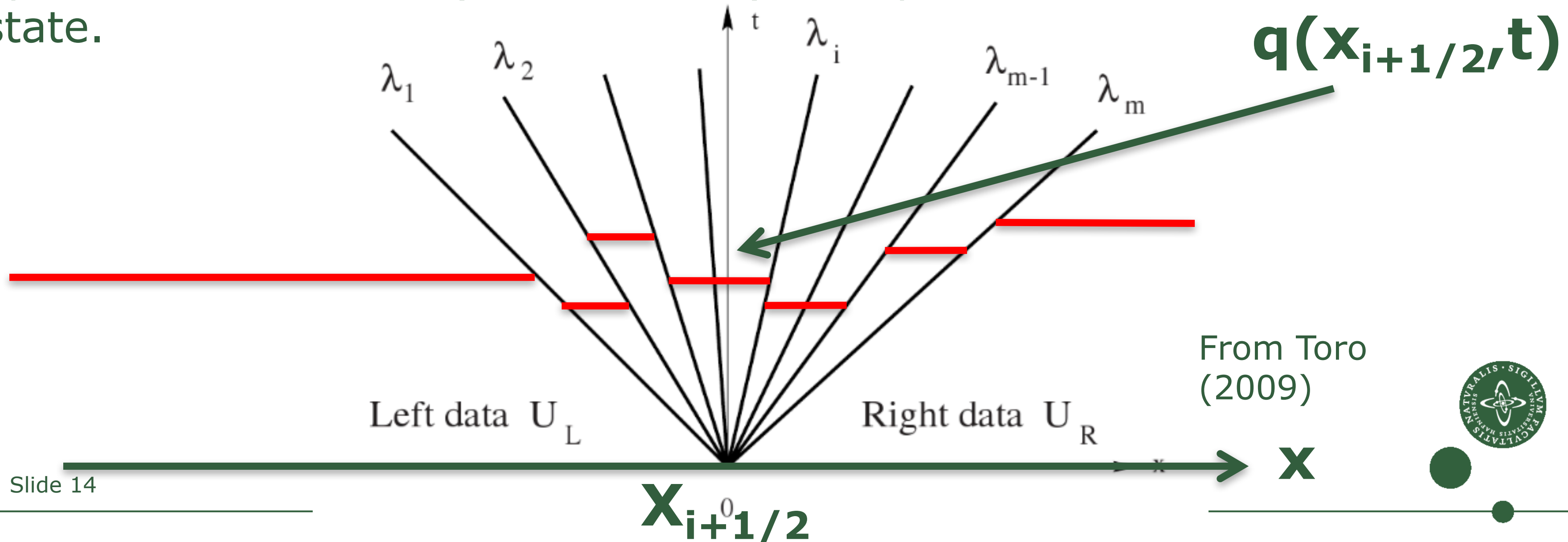
The Riemann Problem

- The problem is to find an expression for the time averaged fluxes

$$\tilde{F}_i(t+\Delta t/2) = \frac{1}{\Delta t}\int_t^{t+\Delta t} dt\ F_i(x_{i-1/2}, t)$$

- To the very lowest order we could approximate the solution inside the cell to be constant

- For a general system of equations there will be several wave speeds apart from advection (HD 3, MHD 7). Compute to find interface state.



From Toro (2009)

Problems with basic FV Godunov Method

- The problem is to find an expression for the time averaged fluxes

$$\tilde{F}_i(t+\Delta t/2) = \frac{1}{\Delta t}\int_t^{t+\Delta t} dt\ F_i(x_{i-1/2}, t)$$

- In general one can solve the ***Riemann problem.***

- Problem: The solution is extremely diffusive,
  - We need to reconstruct the value at the interface based on the cell average. This is called ***slope reconstruction***.
  - For the time evolution we can use either ***implicit methods*** (difficult) or some kind of ***predictor-corrector*** scheme.

Higher Order Godunov Solvers – Time:

- Make a better prediction for the flux integral by for example

$$\tilde{F}_i(t+\Delta t/2)=\frac{1}{2}\left[F_i(x_{i-1/2},t)+F_i(x_{i-1/2},t+\Delta t)\right]$$

- The problem is that we do not know the value of ***q(x,t+Δt)***

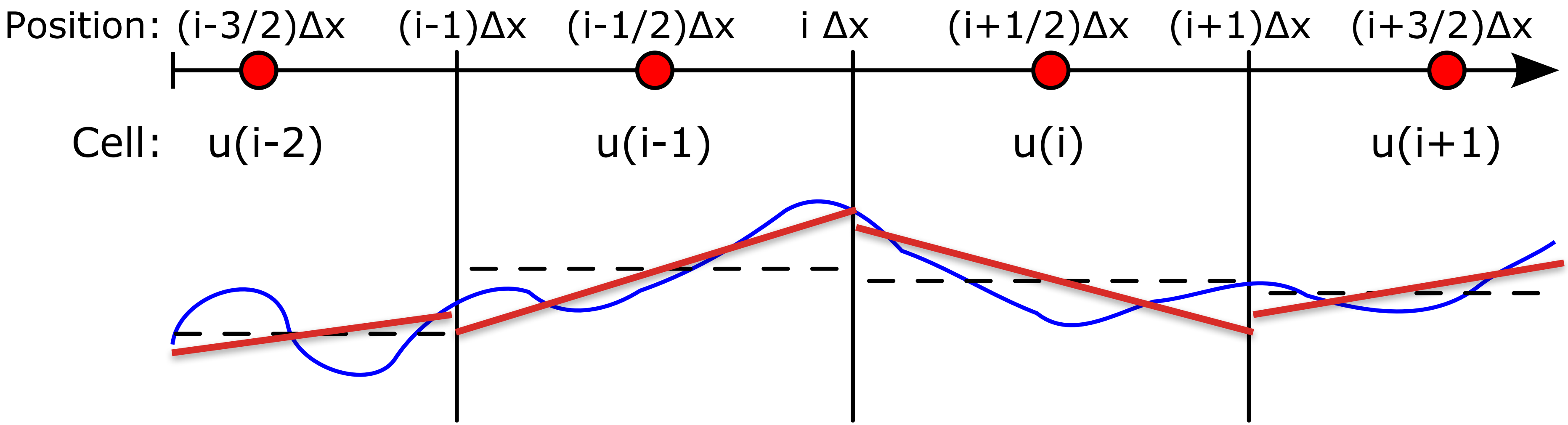
- Use a ***predictor scheme:***

q* = q(x,t)+Δt/2 Centered Difference

Calculate F from q*

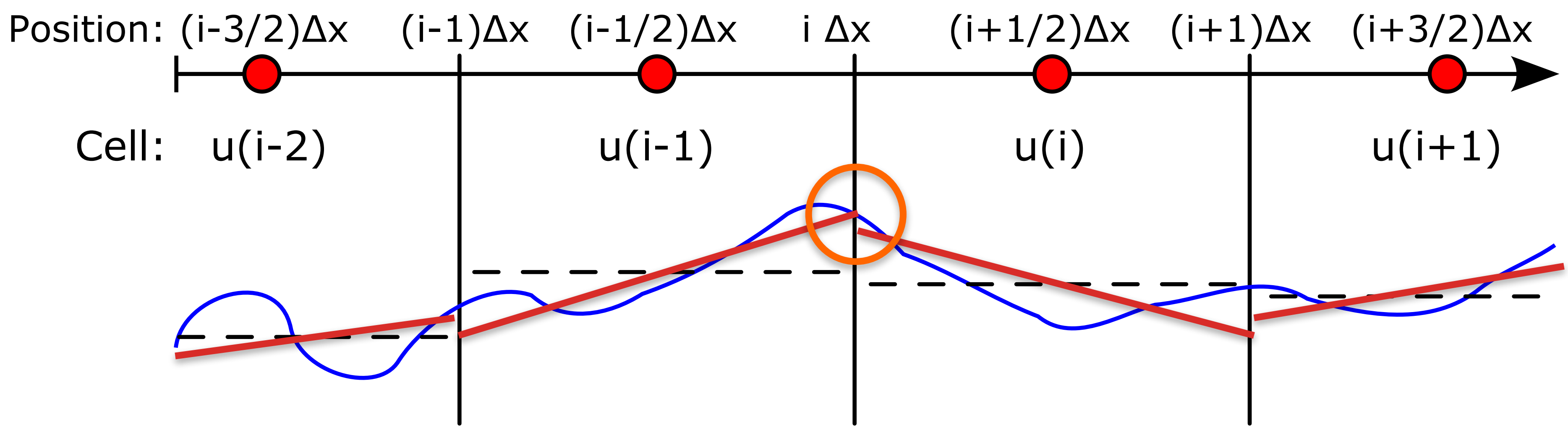
Higher Order in Space – Slope reconstruction

- The Godunov method is very diffusive. Van Leer got the idea (1979) to also use spatial reconstruction for the Flux



Higher Order in Space – Slope reconstruction

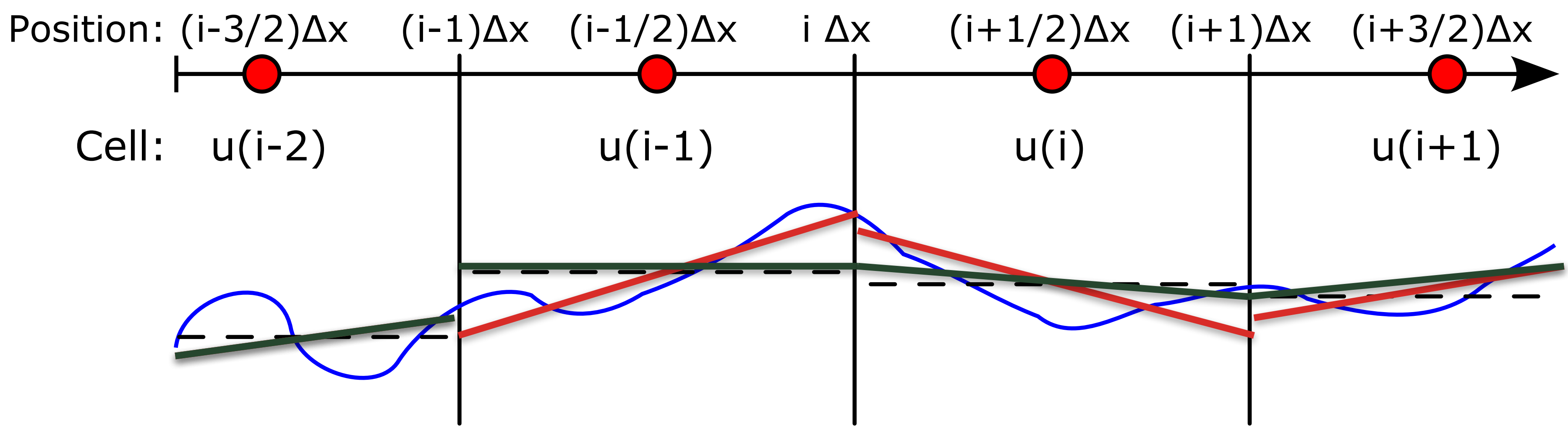
- The Godunov method is very diffusive. Van Leer got the idea (1979) to also use spatial reconstruction for the Flux



- A slope reconstruction has to be ***Total-Variation-Diminishing (TVD) [Harten 1983]***. It cannot introduce new maxima, at the interface. This would lead to oscillations in the solution.

Higher Order in Space – Slope reconstruction

- The Godunov method is very diffusive. Van Leer got the idea (1979) to also use spatial reconstruction for the Flux



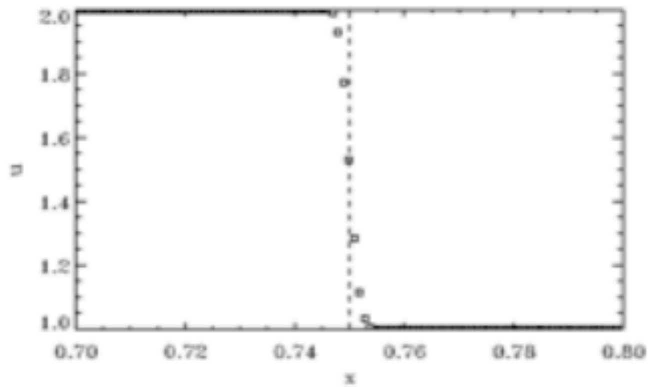
- A slope reconstruction has to be ***Total-Variation-Diminishing (TVD) [Harten 1983]***. It cannot introduce new maxima, at the interface. This would lead to oscillations in the solution.

- Different slope limiters are more or less aggressive in limiting the state at the interface.

Higher Order in Space – Slope reconstruction

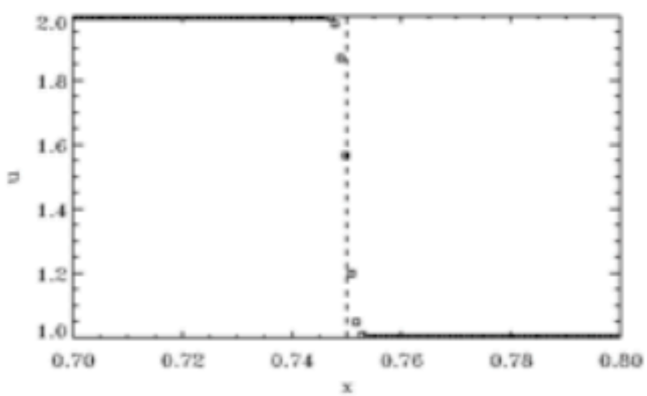
first order

slope_type=0



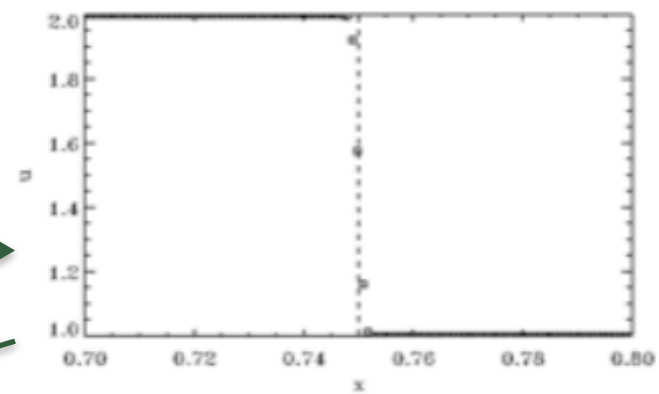
minmod

slope_type=1

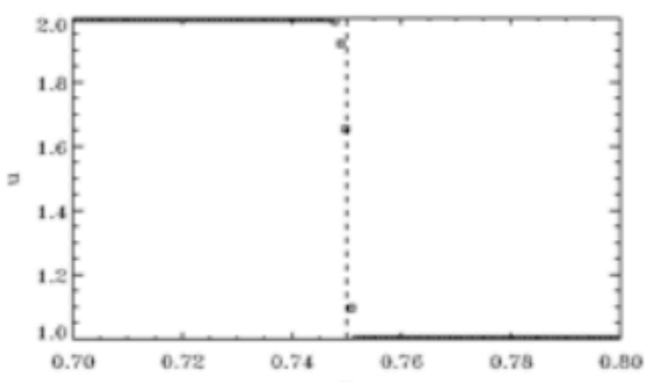


moncen

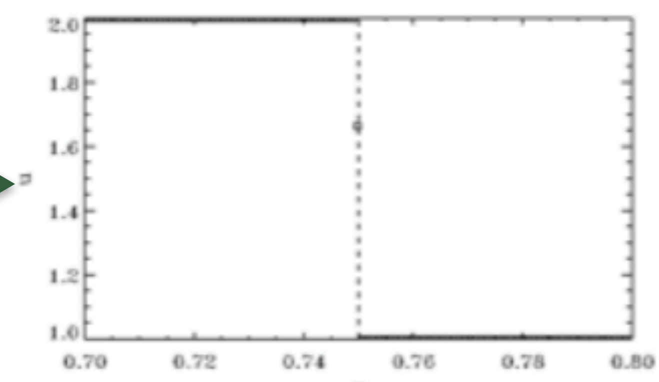
slope_type=2



superbee



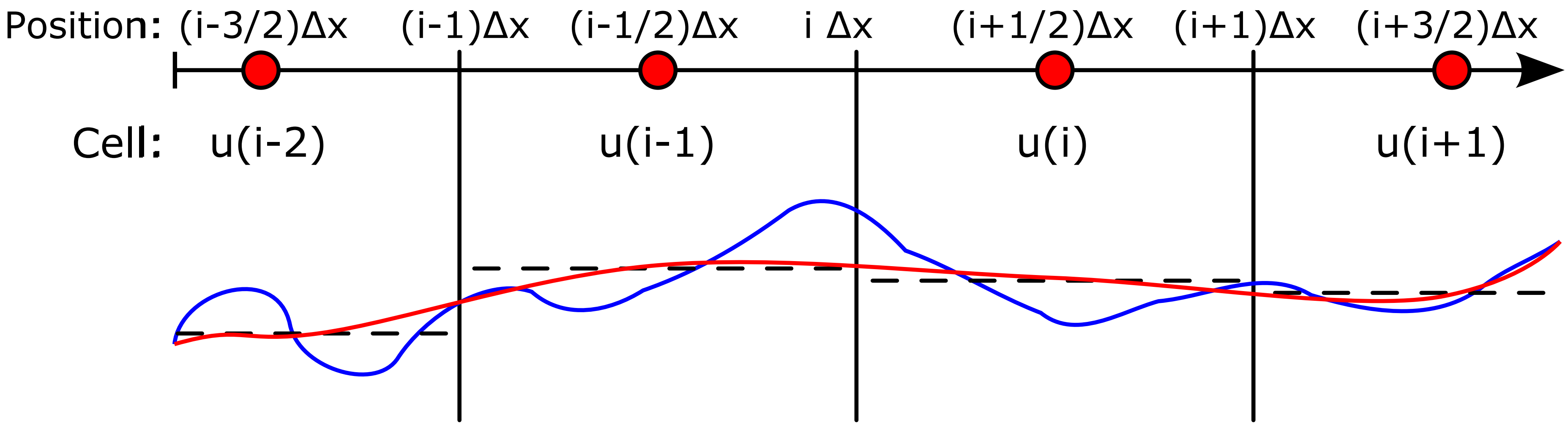
ultrabee



Made with RAMSES by Romain Teyssier

Higher Order in Space – Slope reconstruction

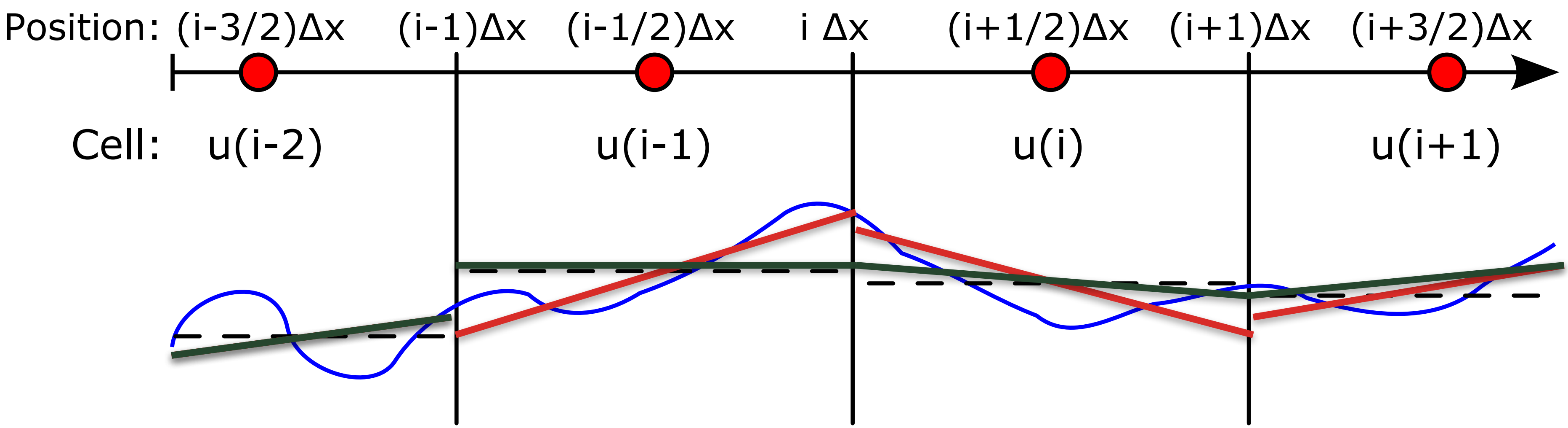
- The Godunov method is very diffusive. Van Leer got the idea (1979) to also use spatial reconstruction for the Flux



- Even higher order methods uses piece-wise parabolic reconstruction (PPM) or higher order polynomials (WENO).

Summary Finite Volume Methods for PDE's

1. Start with the average values in a cell u(x,t).
2. Find the fastest signal speed and adjust the timestep size Δt
3. Reconstruct the interface values through slope reconstruction



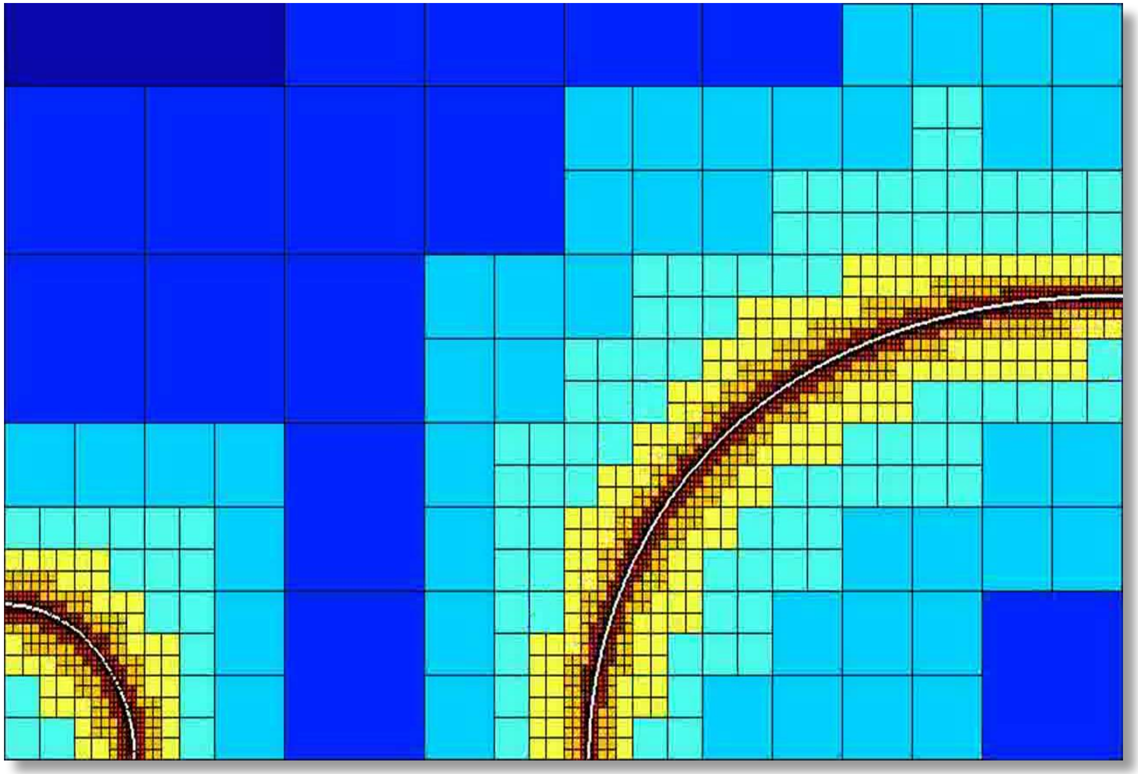
4. Calculate the time averaged flux. Either directly using the equation or indirectly by solving the Riemann problem.
5. Evolve the equation: $u(t+\Delta t)-u(t)=-\frac{\Delta t}{\Delta x}(\tilde{F}_{i+1}-\tilde{F}_i)$

Adaptive Mesh Refinement

What is Adaptive Mesh Refinement (AMR) ?

Most of you probably already know about Adaptive Mesh Refinement, or have heard about the concept. But here is a quick summary:

- AMR-codes are able to (recursively) resolve small details, by using patches (small or large) with increasingly large resolutions
- It is *adaptive*, because the cell placement can change with time



Motivations for Adaptive Mesh Refinement

- Fluid dynamics in three dimensions is costly:
  - Cost of a uniform grid scales as the resolution to the fourth power
  - Even today only ~1024^3 is routine, and the largest unigrid run to date is ~16384^3

- Many problems in astrophysics contain relevant, coupled processes at very different scales
  - Use a sub-grid model description
  - Use different resolution at different places → AMR in space

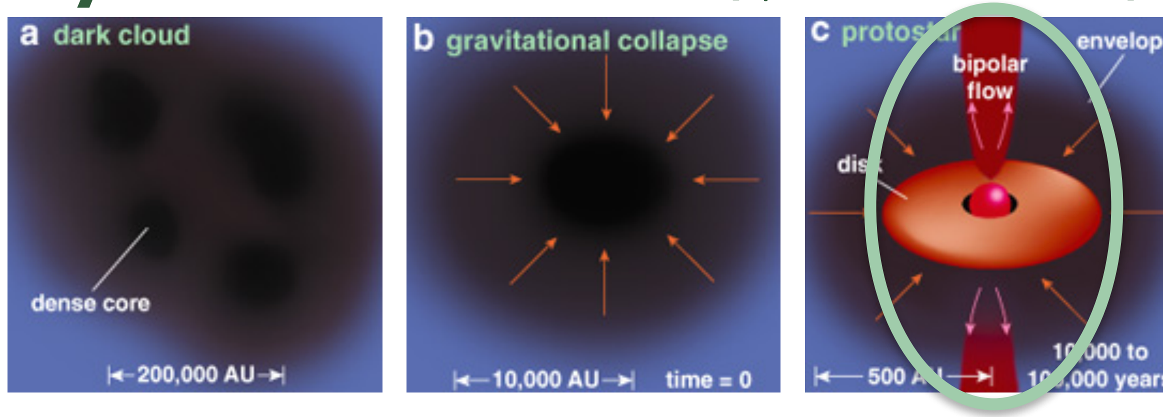
- If velocities are approximately (order of magnitude!) constant the dynamical time-scale scales with the physical scale
  - Large scales evolve slower than small scales → AMR in time

- Using adaptive meshes we can "easily" supply realistic boundaries to a local problem; the ladder of astrophysical AMR is:
  - Cosmology → Galaxy formation → Star formation → Planet Formation

Multiscale Astrophysics

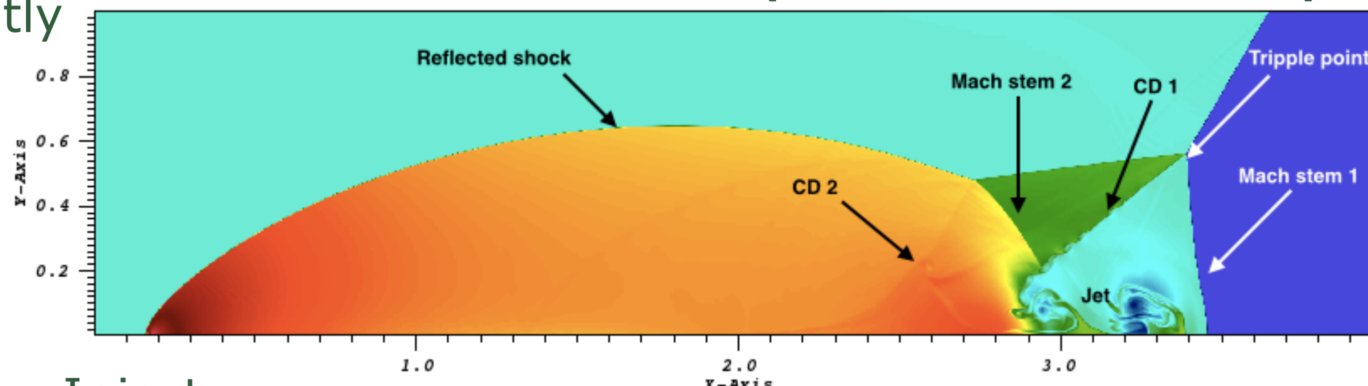
- Selfgravity: induces collapse, with a rapid decrease in scales

[Spitzer Science Center]

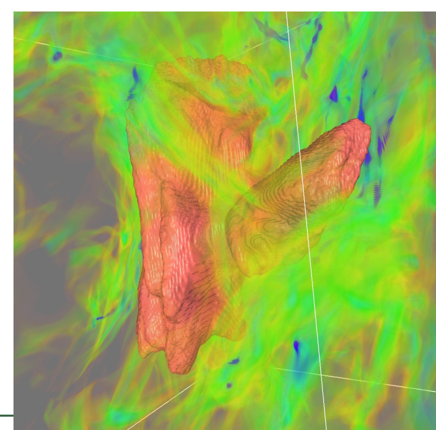


- Shocks: inherently localized; often part of complex flows

[Double Mach shock with Flash]



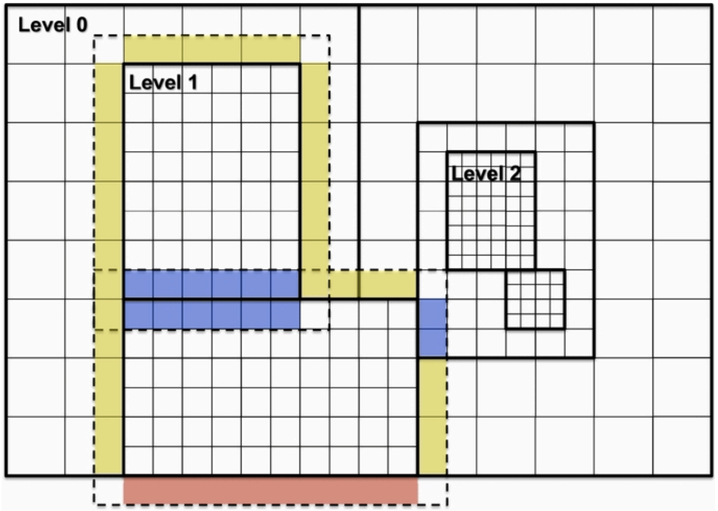
- Compact sources: Injects energy from the smallest scales to the largest



[Ionization front in a molecular cloud launched by central source]

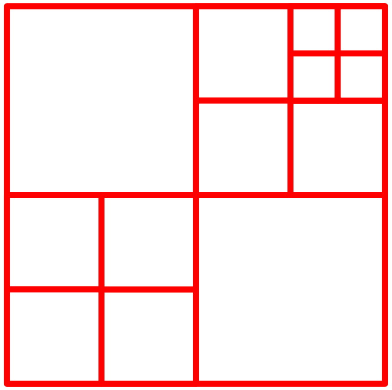
Flavors of Adaptive Mesh Refinement

- Block based AMR (original Collela, Berger, Oliger ***'84 & '89***) [FLASH, **Enzo, Nirvana, Pluto, AZeus**]
  - Use patches of higher resolution completely contained inside lower resolution patches

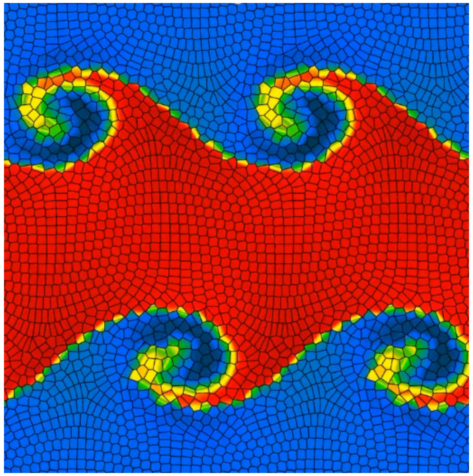


[PLUTO AMR paper]

- Oct based Fully-Threaded-Tree (Khokhlov '98) [**RAMSES**, AMRVAC, ART]
  - Refine on a cell-by-cell basis, with one cell being split in to 8 (in 3D)

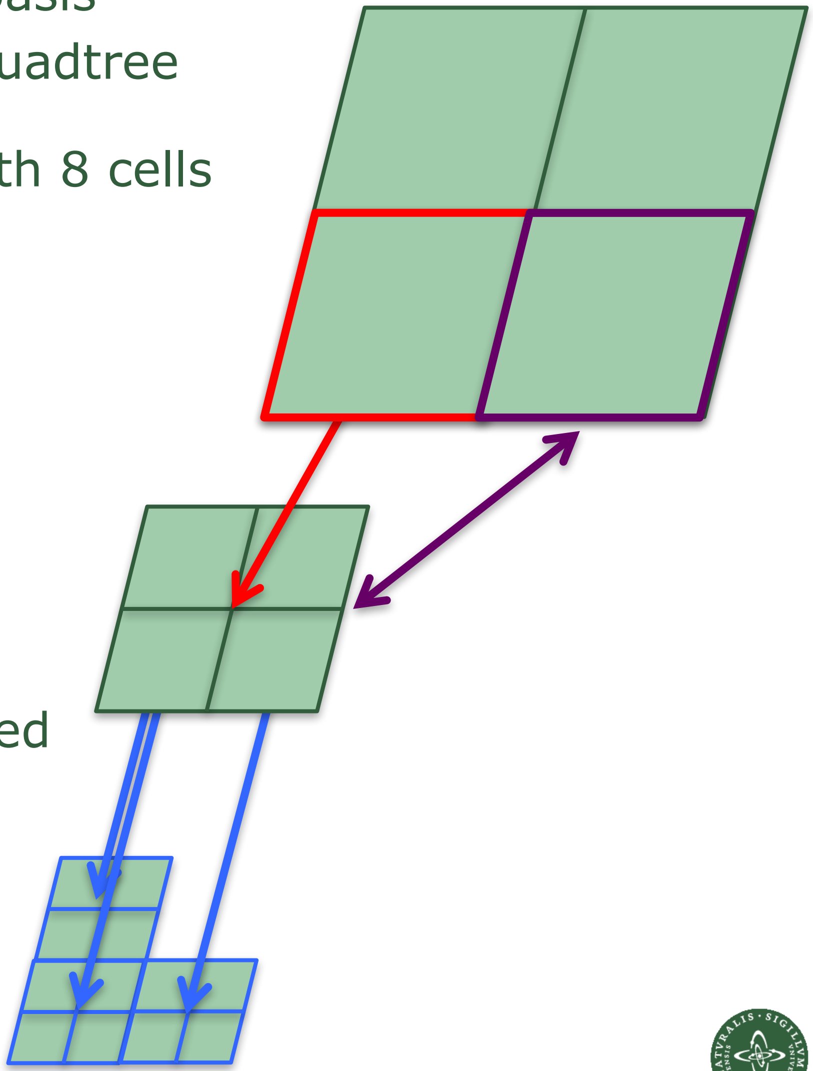


- Unstructured Meshes [AREPO, GIZMO, finite elements]
  - Partition space using one volume per tracer particle; f.ex. using a Voronoi tessellation.



Fully-Threaded-Tree Adaptive Mesh Refinment

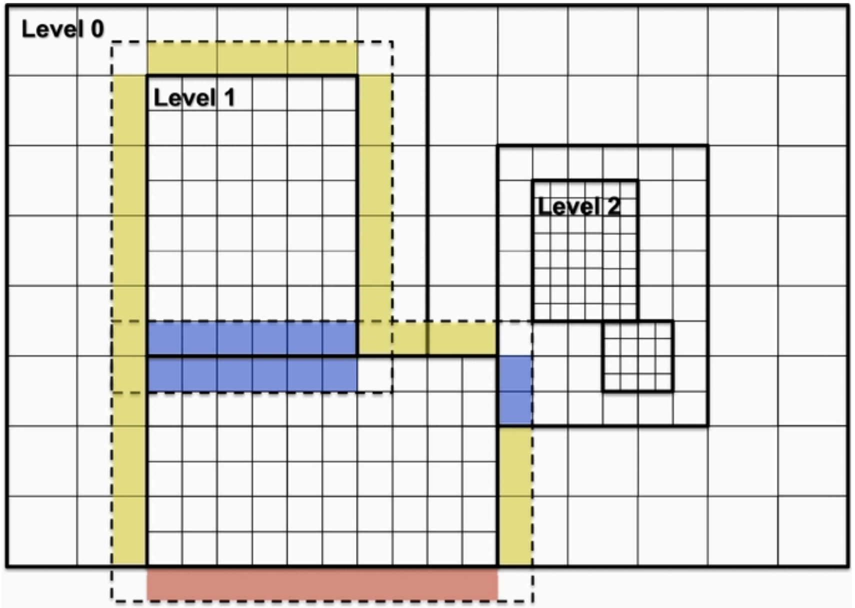
- Refinement is done on a cell-by-cell basis
  - 3D: octree (8 cells per oct), 2D: quadtree
- Each cell can be refined to one oct with 8 cells
  - Very adaptive grid
- Very simple relationship structure
  - 1 parent cell
  - 6 neighbor parent cells
  - Potentially 8 children octs
- Everything is constructed recursively
  - Position and relationships are picked up from parent cell at creation
- All cells in the tree are kept
  - Leaf cells have no children octs
  - Refined cells are inactive



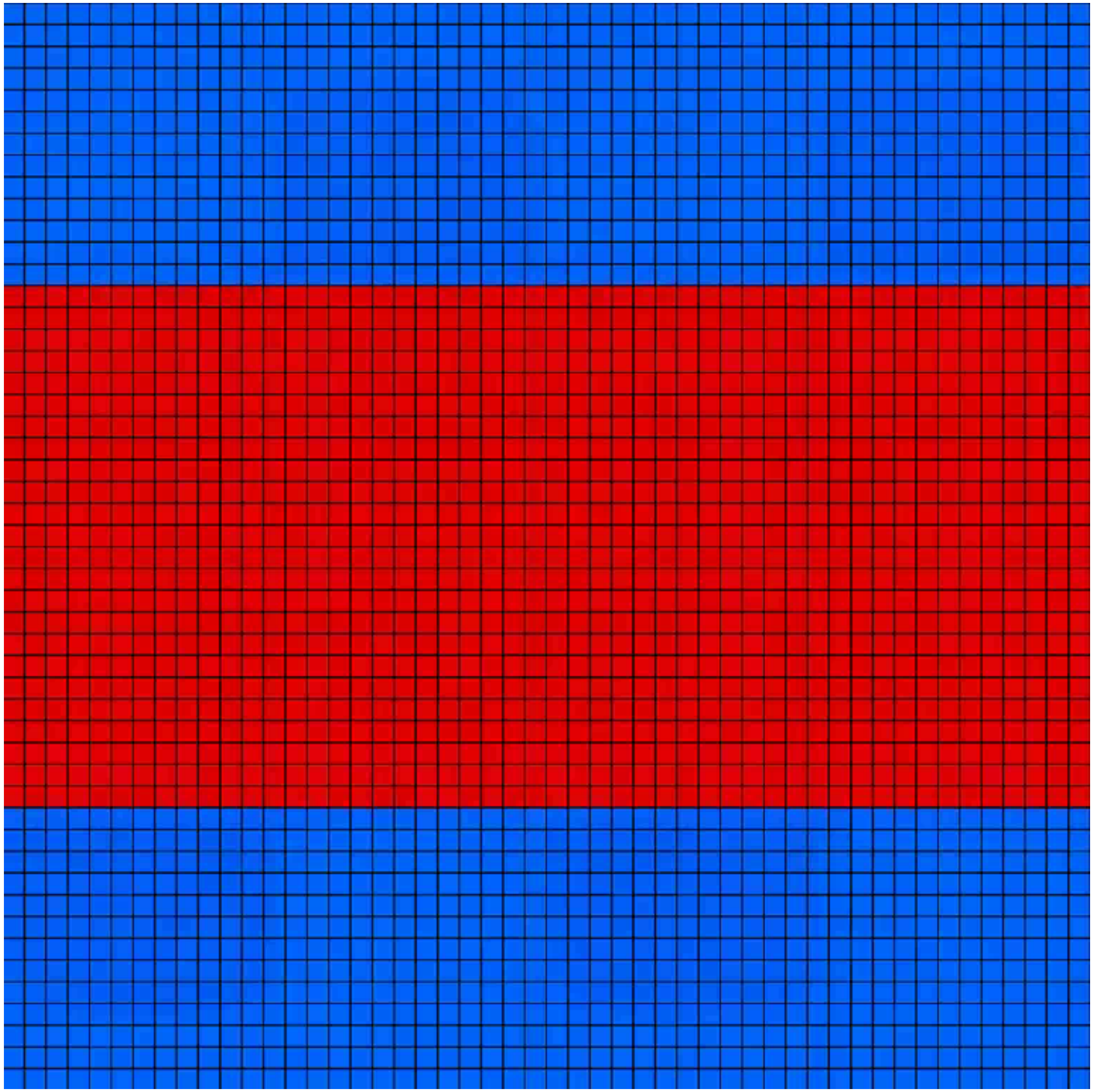
Block based Adaptive Mesh Refinement

- Refinement is done in a number of cells inside a patch to create a new patch
- Typical size of a patch is $(12\text{-}16)^{Ndim}$
  - Reasonable grid size. Efficient to manage
- Complex but flexible relationship structure
  - 1 parent patch (in simplest version)
  - N_{bor} neighbor patches (at different levels)
  - N_{child} children patches
- Everything is constructed recursively
  - Position and relationships are picked up from parent patches at creation
- Normally all cells in a patch are kept
  - Leaf cells have no patches on top
  - Refined cells are inactive

[PLUTO AMR paper]



Unstructured (moving) meshes



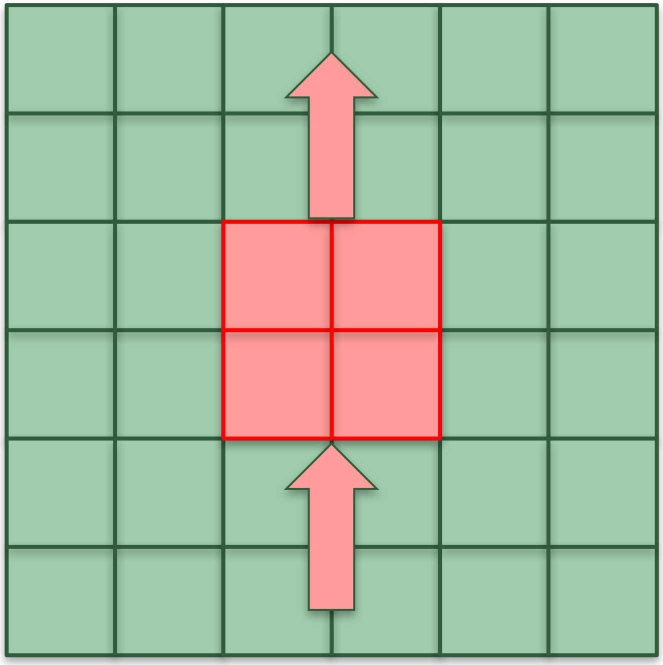
- Unstructured codes are using tracer points for their geometry.
- F.x. the AREPO and GIZMO codes use unstructured and meshless representations, respectively.
- Their representation has the important advantage (over fixed-grid codes), to respect Galilean invariance; i.e., their results are the same, independent of any bulk motion of the system under study.
- The Courant condition is only due to *relative motion*.
- But cost is high!

Ingredients of an AMR method

Fluid Dynamics on an AMR patch / in an oct

- The fluid dynamics in on a adaptive mesh is solved exactly as on a normal mesh

1. Extract patch or oct
2. Pick up "guard cells":
   1. Check if neighbors are refined
   2. Else create new cells on-the-fly

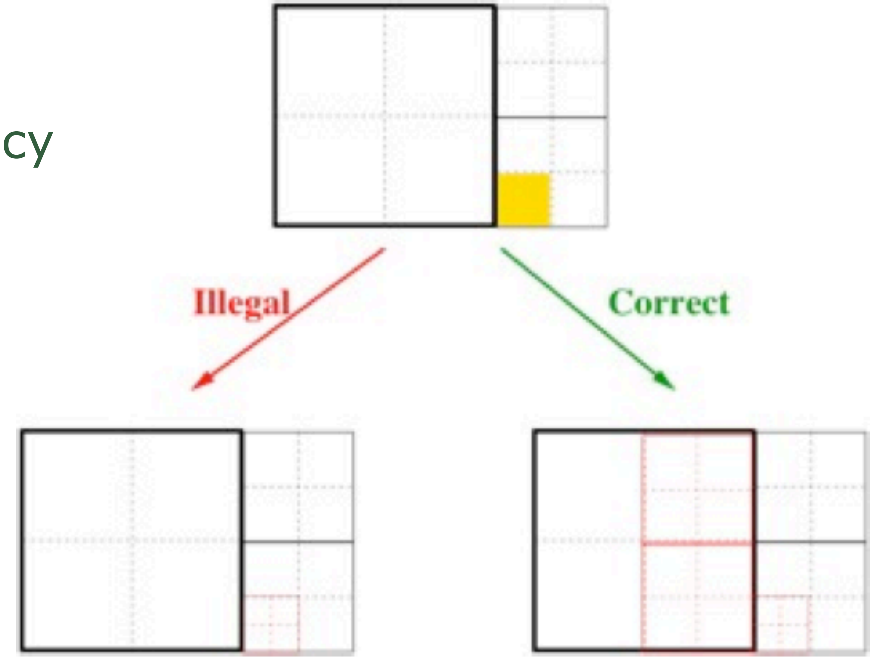


- Use MHD solver from e.g. the uni-grid code
- Hard to use higher order methods; we typically only use 2 guard cells to maintain a reasonable surface-to-volume ratio
  - Motivates the use of Godunov methods or very compact finite differences
- *Parent neighbors always need to exist*, or we cannot get boundaries on-the-fly

AMR Refinement Rules

- Most codes enforce mesh consistency and grading of patches / octs
  - All neighbors have to be at the same level or one level below or above

- Specific Criteria
  - Jeans Criteria
  - Gradients
  - Quasi-Lagranian
  - Geometrical (zoom)



[from talk by Romain Teyssier]

Refinement Criteria

- Jeans / Truelove Criteria
  - If the Jeans length is not resolved, collapse becomes unphysical

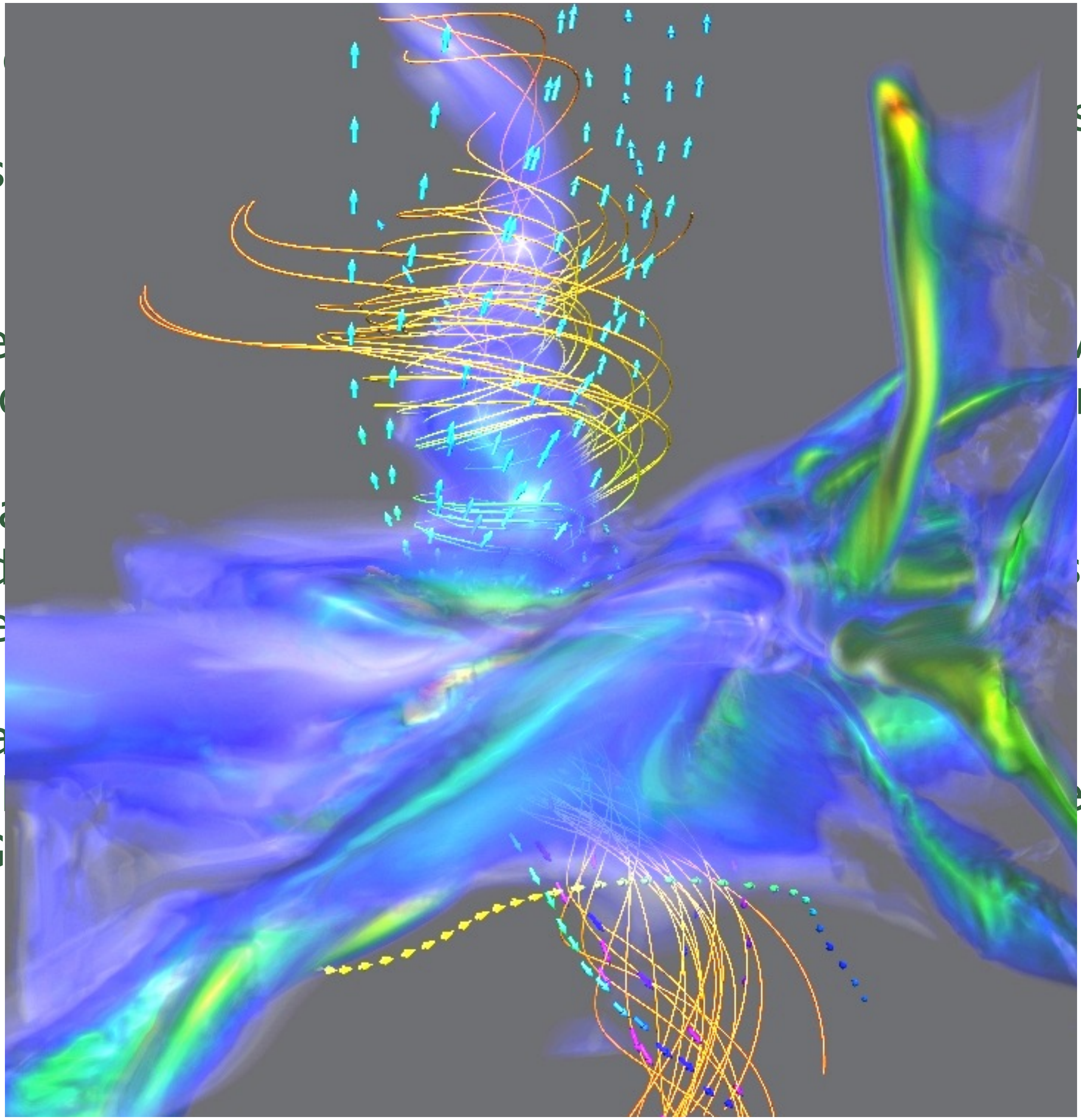
- Gradients
  - Beware of shocks; if encounter a real jump you get AMR catastrophe -> individual max level for gradient-refinement

- Quasi-Lagranian
  - Related to Jeans. F.x. refine every time density goes up with 4 to have same Jeans resolution on all levels

- Geometrical (zoom)
  - Only allow code to refine in specific region. Adapt center to flow; Gallilean transform; follow star

Refinement Criteria

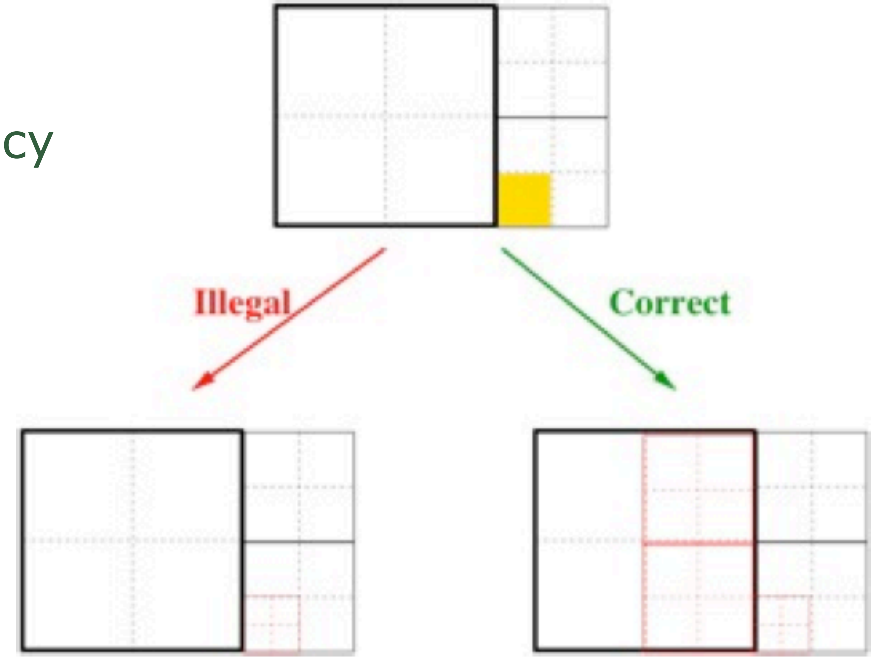
- Jeans / Tru
  - If the J ... s unphys
- Gradients
  - Beware ... AMR catastr ... nement
- Quasi-Lagr
  - Related ... up with 4 to have
- Geometrica
  - Only al ... nter to flow; G



[Refinement to pick up Jet; VAPOR viz by M. Küffmeier]

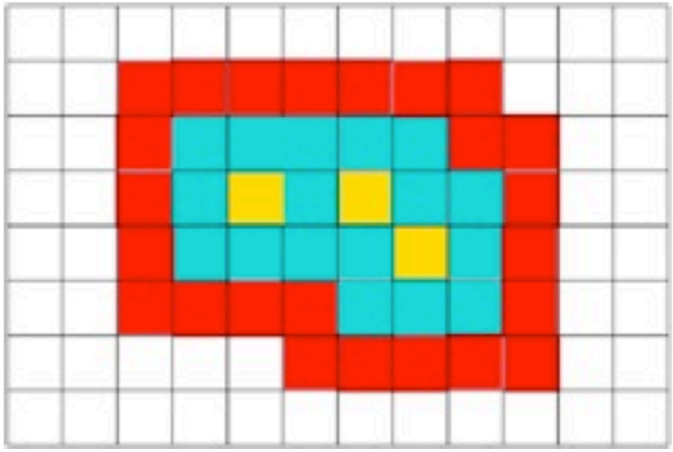
AMR Refinement Rules

- Most codes enforce mesh consistency and grading of patches / octs
  - All neighbors have to be at the same level or one level below or above



- Specific Criteria
  - Jeans Criteria
  - Gradients
  - Quasi-Lagranian
  - Geometrical (zoom)

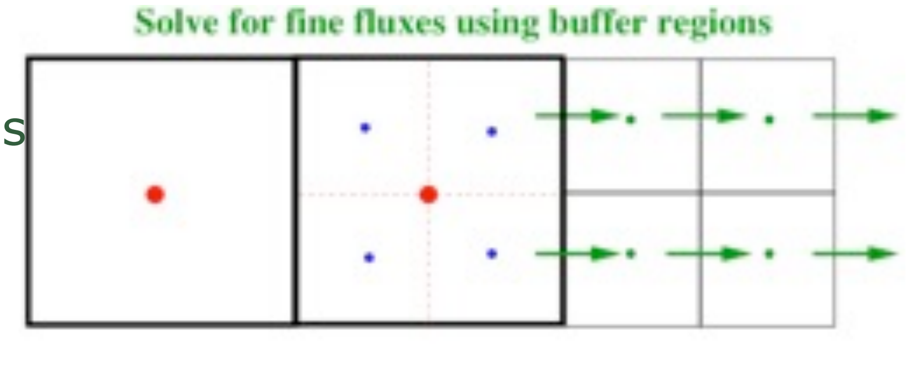
- Smoothing is often done
  - Patch has to be convex
  - Expand patch sizes with *nexpand* layers (ex. 2 layers)



[from lectures by Romain Teyssier]

Interpolation and Flux consistency

- *Prolongation:* Interpolation to finer meshes
  - Creation of new cells
  - On-the-fly boundary cells
- *Restriction:* Averaging to coarser levels
  - Destruction / derefinement
  - Filling of the threaded tree
- Flux correction at boundaries
  - Relatively easy for volume fluxes
  - Tricky for EMF at edges
- Different interpolators
  - Conservative / internal energy
  - Apply different slope limiters
  - ***When changing the resolution the method looses one order***



[from lectures by Romain Teyssier]

Time-adaptivity: Fine→coarse level time Evolution

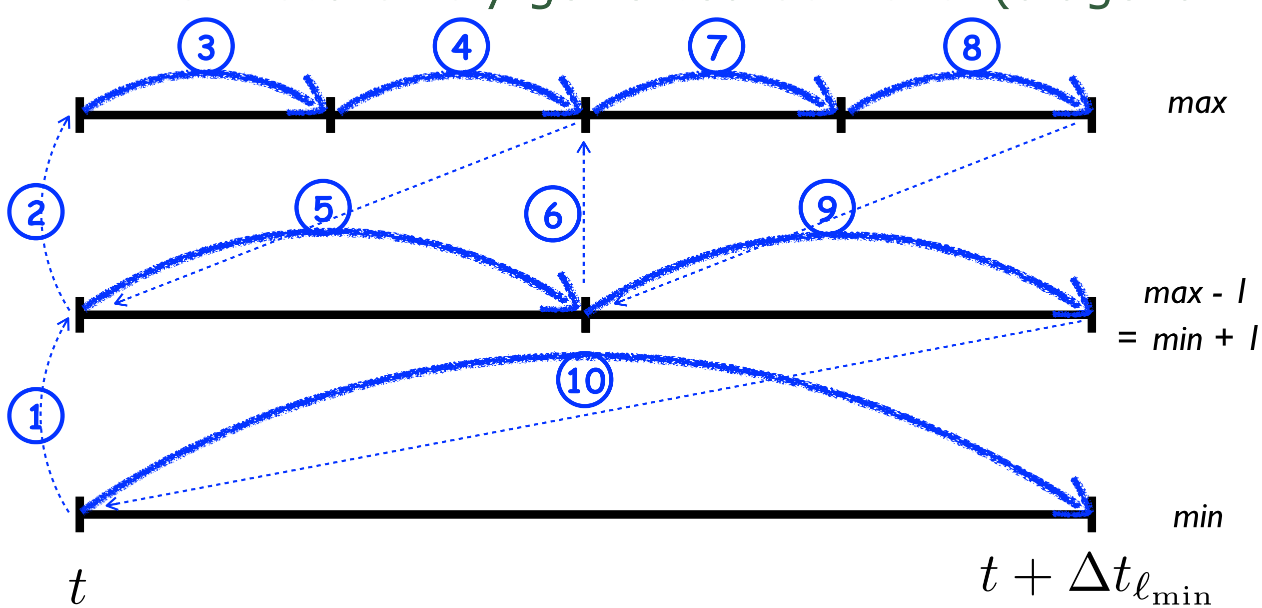
- Evolves in a W-cycle recursively from coarser to finer mesh and back

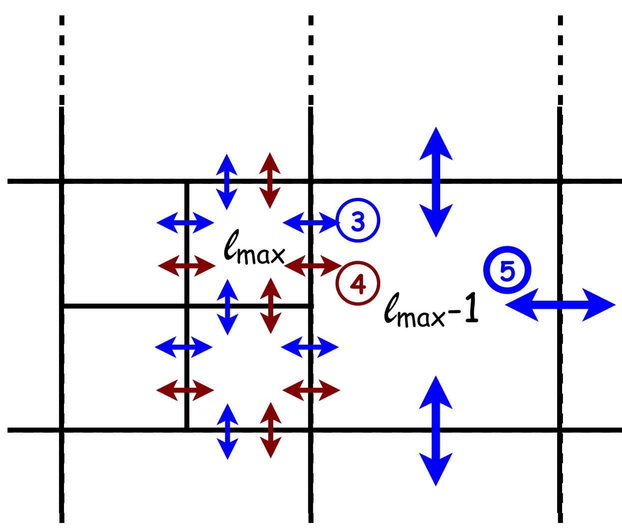
  1. Prepare: Check on coarse level if we need to create new cells, prepare boundary conditions for finer levels. Then go to finer level.

  2+6: Repeat step 1 and recursively progress to finer levels.

  3-5,7-10. Evolve: Solve MHD on finest level; update Courant, update flux for coarser cells via neighbor pointer; flag any cells on this level that have to be refined or destroyed.

  Then recursively go to coarser level (diagonal lines) or repeat timestep





[Figs from Rosdahl 2013]