RAMSES

A short introduction

Benoît Commerçon, Ugo Lebreuilly

Action Spécifique Numérique

- Inscrivez-vous sur la liste de diffusion!

https://listes.services.cnrs.fr/wws/subscribe/asn-insu?previous_action=info2

Plan

- **Introduction and overview of RAMSES methods**

B. Commerçon

- **Example of numerical developments and applications**

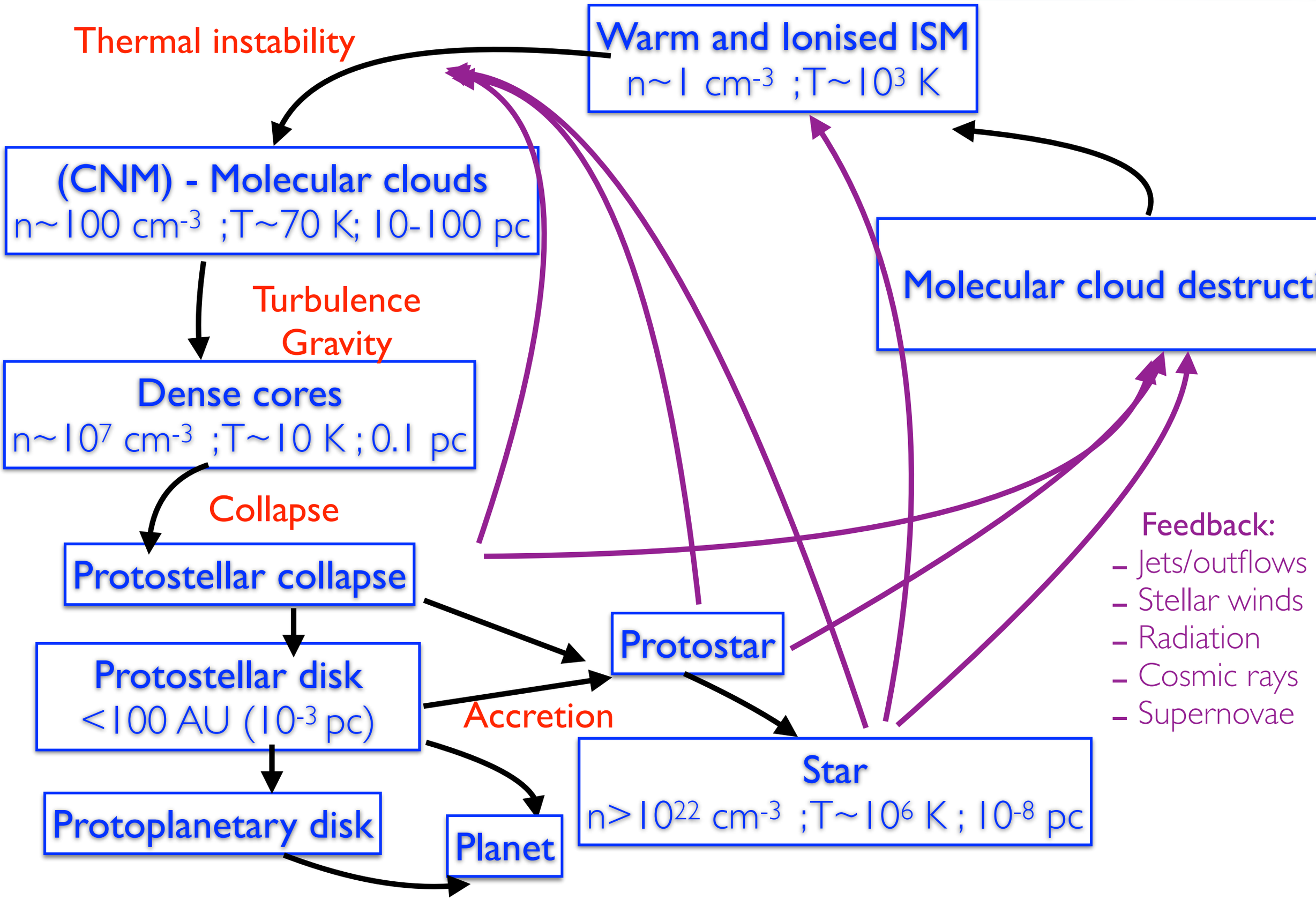
U. Lebreuilly

- **Practical session: download, compile & run RAMSES on laptops and mesopsl**

Outline

- Introduction
- Finite Volume & Godunov solver
- MHD solver
- AMR
- Parallel computing
- Implicit scheme example

Interstellar matter cycle



What do we find in the interstellar medium?

- photons at all wavelengths
- gas (mainly H, 10% He and 10^{-4} heavy elements), turbulent
- magnetic fields (from galactic dynamo?)
- dust (solid phase, 1% mass compared to the gas), but (thermo)dynamically important...
- cosmic rays (high energy particles)

$$E_{th}=E_{grav}=E_{kin}=E_{mag}=E_{rad}=E_{cr} \sim 1 \text{ ev/cm}^3$$

➡ multifold research field, all processes couple together…

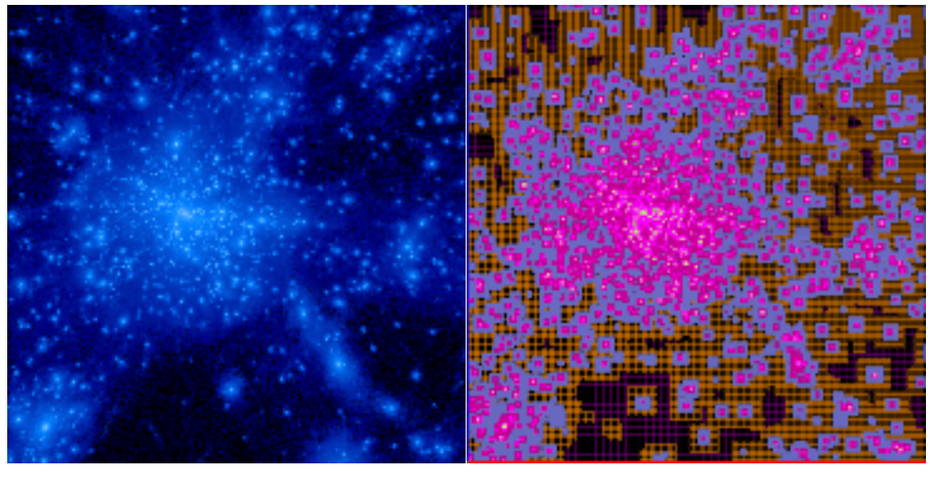
➡ slow progress

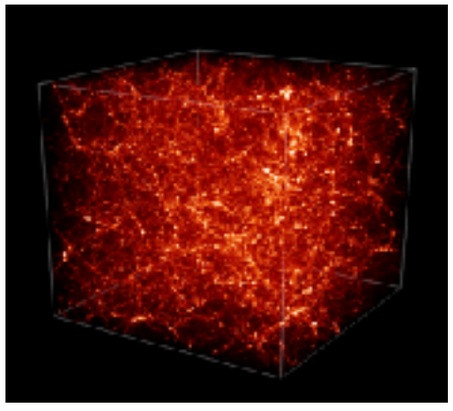
RAMSES

Principal developer: Romain Teyssier (Princeton University, aup. Zurich and CEA Saclay**)**

Mean features

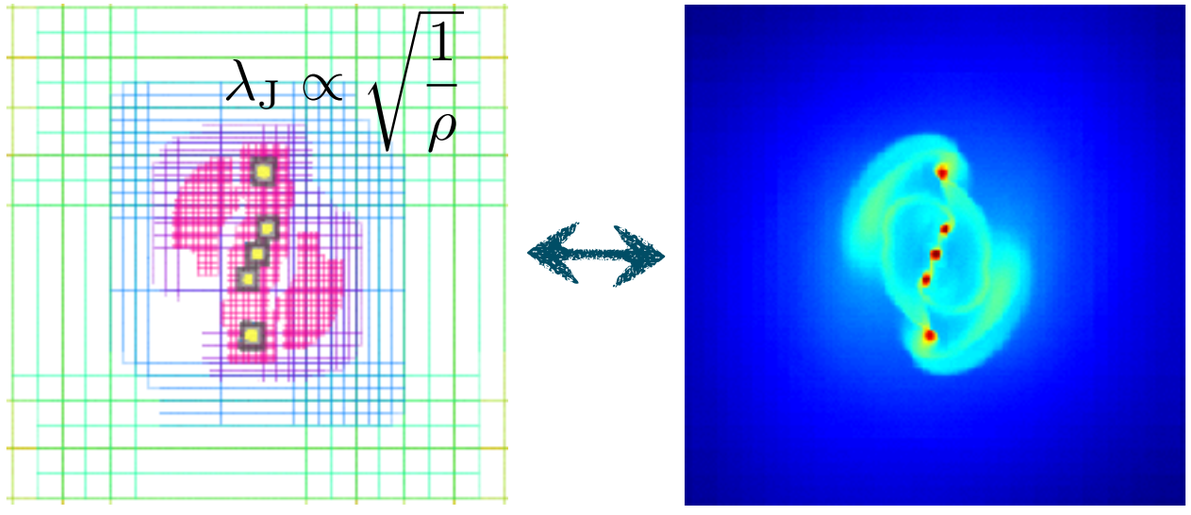
- Adaptive-mesh refinement
- Oct-tree structure
- versatile
- “low” memory footprint
- **Stand alone**
- Adaptive time-stepping
- Parallelised (MPI)



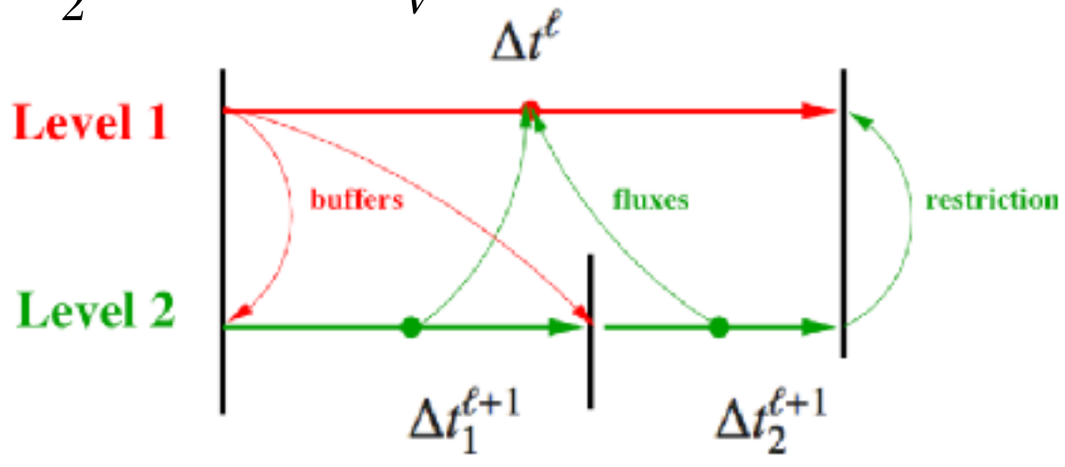


Numerous astrophysical applications (PCMI, PNCG, PNPS, PNP, PNHE)

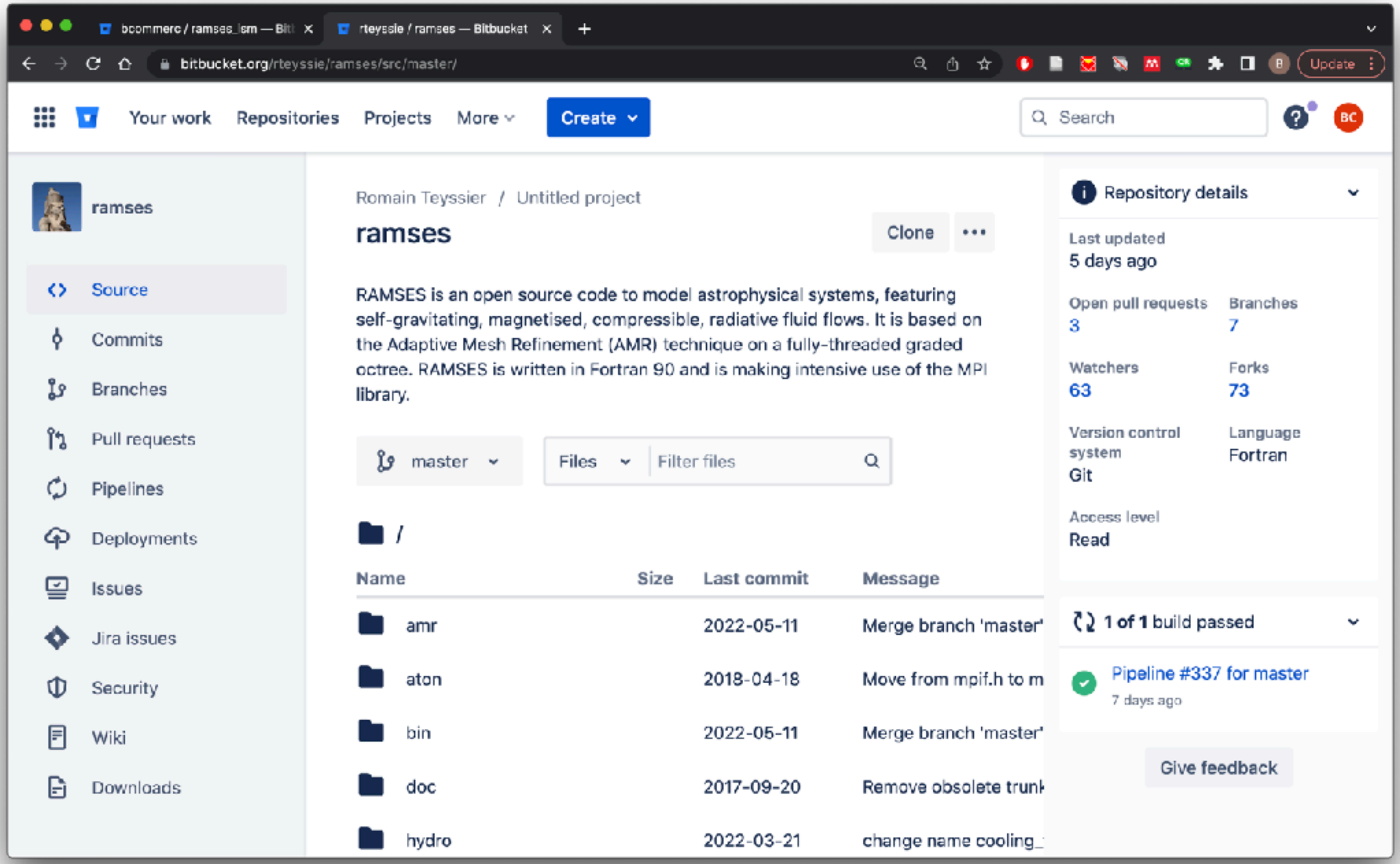
- Cosmology
- Galaxy formation and evolution
- Star and planet formation
- Compressible turbulence



$$\Delta x^{\ell+1} = \frac{\Delta x^{\ell}}{2} \qquad \Delta t < C\frac{\Delta x}{V}$$

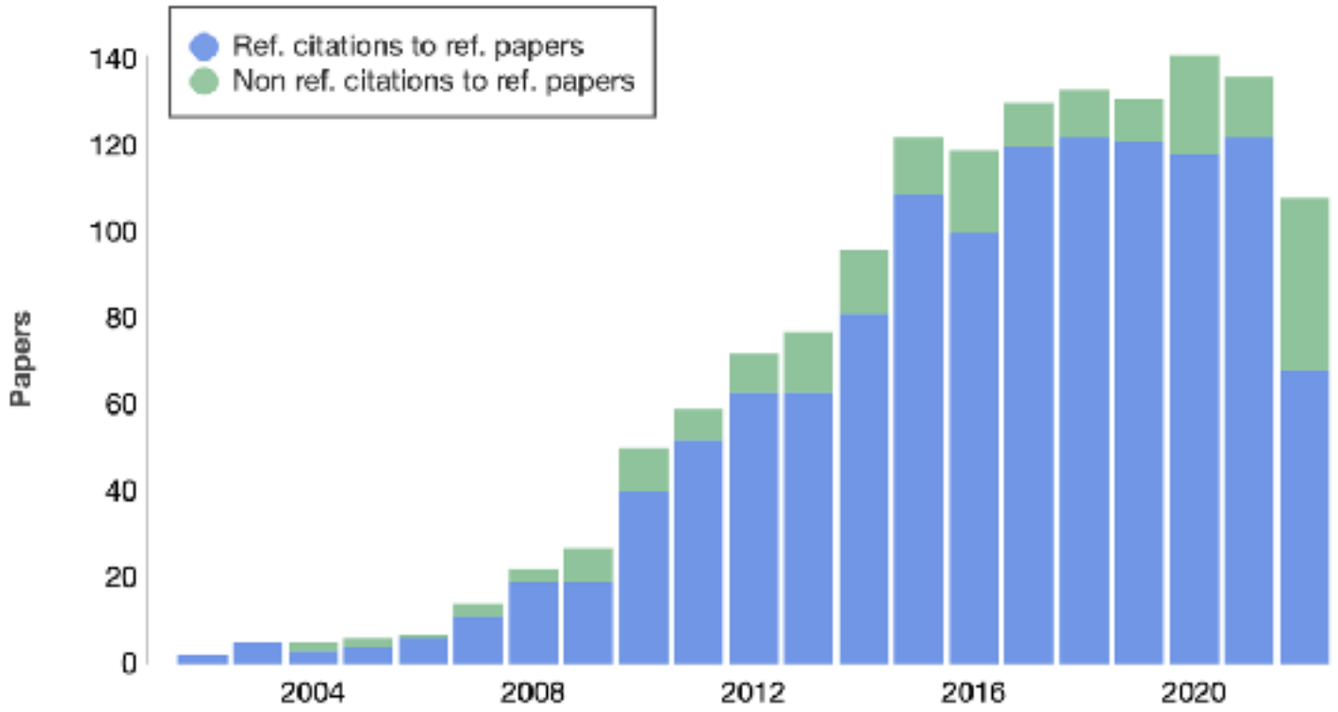


RAMSES


bcommerc / ramses_lsm — Bitl
rteyssie / ramses — Bitbucket
bitbucket.org/rteyssie/ramses/src/master/
Update
Your work
Repositories
Projects
More
Create
Search
BC
ramses
Source
Commits
Branches
Pull requests
Pipelines
Deployments
Issues
Jira issues
Security
Wiki
Downloads
Romain Teyssier / Untitled project
ramses
Clone
RAMSES is an open source code to model astrophysical systems, featuring self-gravitating, magnetised, compressible, radiative fluid flows. It is based on the Adaptive Mesh Refinement (AMR) technique on a fully-threaded graded octree. RAMSES is written in Fortran 90 and is making intensive use of the MPI library.
master
Files
Filter files
/
Name
Size
Last commit
Message
amr
2022-05-11
Merge branch 'master'
aton
2018-04-18
Move from mpif.h to m
bin
2022-05-11
Merge branch 'master'
doc
2017-09-20
Remove obsolete trunk
hydro
2022-03-21
change name cooling_
Repository details
Last updated
5 days ago
Open pull requests
3
Branches
7
Watchers
63
Forks
73
Version control system
Git
Language
Fortran
Access level
Read
1 of 1 build passed
Pipeline #337 for master
7 days ago
Give feedback

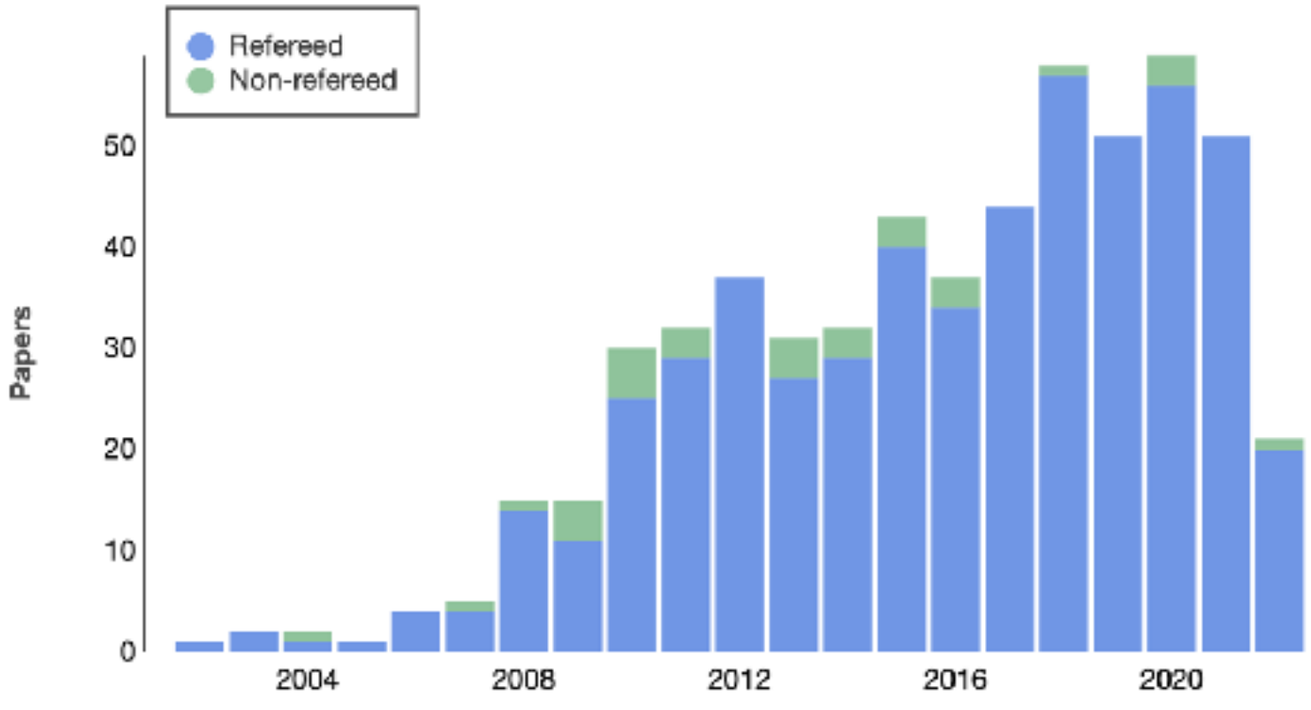
RAMSES community

International community



Evolution of the number of citations of the RAMSES first paper (Teyssier 2002). (Source: ADS.)

Evolution of the number of papers citing the RAMSES first paper (Teyssier 2002) with a French coauthor (Source: ADS.)



RAMSES community

Increasing international user community

High proportion of European and french among the developers and the users

~40-50 papers/year from the French community

One the most used code by the astro French community at GENCI and at PRACE (~ 100 Mh CPUS/year <=> ~1 M€/year)

RAMSES User Meeting every year with 40-60 participants (100 in 2021, remote)

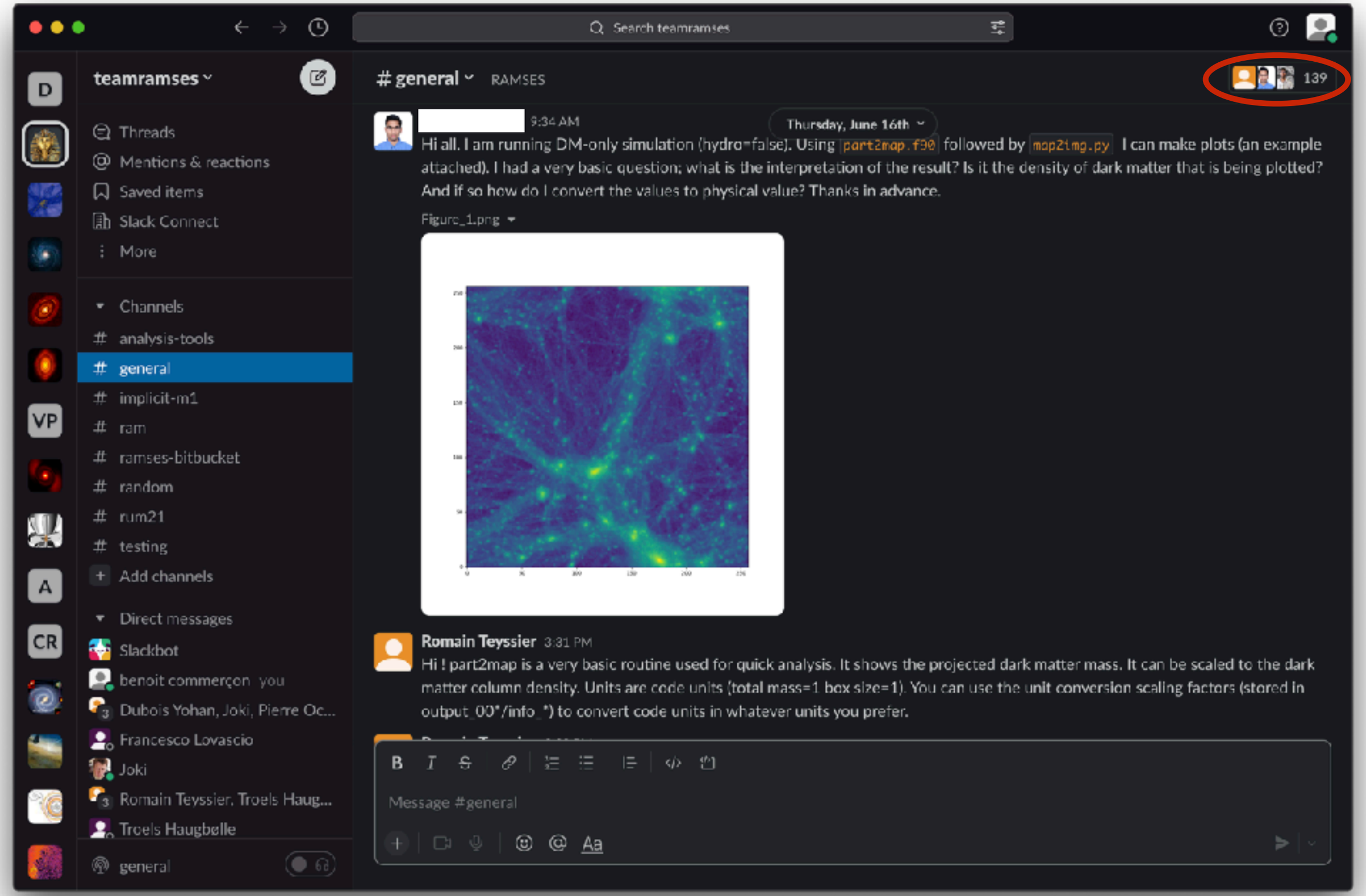
- Release stable version
- Hackathon to implement the last developments

Formation within the ASTROSIM schools (2017-2020)

- Practical session
- Formation on other astro codes (PHANTOM, PLUTO, MAGIC, etc…)

User forum, Slack workspace, online documentation (https://bitbucket.org/rteyssie/ramses/wiki/Content)

RAMSES community



RAMSES

Physics included

In the public version

https://bitbucket.org/rteyssie/ramses/src/master/

Gravity

- self-gravity
- Dark matter
- User-defined

Collision-less particles

- Dark matter
- Stars
- Sink particles
- Tracer particles
- Subgrid modules (star formation, feedback, AGN, etc…)
- Dust

Hydrodynamics

- Radiative transfer (M1)
- Magnetic fields (ideal MHD)
- Relativity
- Non-thermal fluids
- Chemistry (e.g. KROME)
- low-Mach

(2002-2022)

In "forked" version ramses_ism

(non-exhaustive list, merging in progress)

Ask to get access

Non-ideal MHD

- Ohmic diffusion
- Ambipolar diffusion
- Hall effect

Hydrodynamics

- Comics rays
- Electronic conduction
- Dust dynamics
- Multi-frequency RT
- Non-ideal gas EOS

Particles

- Monte Carlo tracers
- Dust (bifluid)

Hydrodynamics

Euler equations in conservative form:

$$\frac{\partial \rho}{\partial t} + \nabla \cdot [\rho \mathbf{u}] = 0$$

$$\frac{\partial \rho \mathbf{u}}{\partial t} + \nabla \cdot [\rho \mathbf{u} \otimes \mathbf{u} + P\mathbb{I}] = -\rho \nabla \phi$$

$$\frac{\partial E_{\mathrm{T}}}{\partial t} + \nabla \cdot [\mathbf{u}\,(E_{\mathrm{T}} + P_{\mathrm{T}})] = -\rho \mathbf{u} \cdot \nabla \phi$$

$$\longrightarrow \quad \frac{\partial \mathbb{U}}{\partial t} + \nabla \cdot \mathbb{F}(\mathbb{U}) = \mathbb{S}(\mathbb{U})$$

$$E_{\mathrm{T}} = e + 1/2\rho \mathbf{u}^2 \qquad P = (\gamma - 1)e$$

Conservatives

$$\mathbb{U} = \begin{bmatrix} \rho \\ \rho \mathbf{u} \\ E_{\mathrm{T}} \end{bmatrix}$$

Primitives

$$\mathbb{V} = \begin{bmatrix} \rho \\ \mathbf{u} \\ P \end{bmatrix}$$

Hydrodynamics

Euler equations in conservative form:

$$\frac{\partial \rho}{\partial t} + \nabla \cdot [\rho \mathbf{u}] = 0$$

$$\frac{\partial \rho \mathbf{u}}{\partial t} + \nabla \cdot [\rho \mathbf{u} \otimes \mathbf{u} + P \mathbb{I}] = -\rho \nabla \phi$$

$$\frac{\partial E_{\mathrm{T}}}{\partial t} + \nabla \cdot [\mathbf{u} (E_{\mathrm{T}} + P_{\mathrm{T}})] = -\rho \mathbf{u} \cdot \nabla \phi$$

$$\longrightarrow \quad \frac{\partial \mathbb{U}}{\partial t} + \nabla \cdot \mathbb{F}(\mathbb{U}) = \mathbb{S}(\mathbb{U})$$

Operator splitting: conservative update + source terms

Hydrodynamics

Euler equations in conservative form:

$$\frac{\partial \rho}{\partial t} + \nabla \cdot [\rho \mathbf{u}] = 0$$

$$\frac{\partial \rho \mathbf{u}}{\partial t} + \nabla \cdot [\rho \mathbf{u} \otimes \mathbf{u} + P \mathbb{I}] = -\rho \nabla \phi$$

$$\frac{\partial E_{\mathrm{T}}}{\partial t} + \nabla \cdot [\mathbf{u} (E_{\mathrm{T}} + P_{\mathrm{T}})] = -\rho \mathbf{u} \cdot \nabla \phi$$

$$\longrightarrow \quad \frac{\partial \mathbb{U}}{\partial t} + \nabla \cdot \mathbb{F}(\mathbb{U}) = \mathbb{S}(\mathbb{U})$$

Operator splitting: conservative update + source terms

Finite volume:

$$\int_V \left(\frac{\partial \mathbb{U}}{\partial t} + \nabla \cdot \mathbb{F}(\mathbb{U}) \right) dV = \int_V \frac{\partial \mathbb{U}}{\partial t} dV + \oint_S \mathbb{F}(\mathbb{U}) \cdot dS$$

Hydrodynamics

Euler equations in conservative form:

$$\frac{\partial \rho}{\partial t} + \nabla \cdot [\rho \mathbf{u}] = 0$$

$$\frac{\partial \rho \mathbf{u}}{\partial t} + \nabla \cdot [\rho \mathbf{u} \otimes \mathbf{u} + P\mathbb{I}] = -\rho \nabla \phi$$

$$\frac{\partial E_{\mathrm{T}}}{\partial t} + \nabla \cdot [\mathbf{u}\,(E_{\mathrm{T}} + P_{\mathrm{T}})] = -\rho \mathbf{u} \cdot \nabla \phi$$

$$\longrightarrow \quad \frac{\partial \mathbb{U}}{\partial t} + \nabla \cdot \mathbb{F}(\mathbb{U}) = \mathbb{S}(\mathbb{U})$$

Operator splitting: conservative update + source terms

Finite volume:

$$\frac{\mathbb{U}^{n+1} - \mathbb{U}^{n}}{\Delta t} V = -\sum \mathbb{F} \times S$$

Hydrodynamics

Euler equations in conservative form:

$$\frac{\partial \rho}{\partial t} + \nabla \cdot [\rho \mathbf{u}] = 0$$

$$\frac{\partial \rho \mathbf{u}}{\partial t} + \nabla \cdot [\rho \mathbf{u} \otimes \mathbf{u} + P \mathbb{I}] = -\rho \nabla \phi$$

$$\frac{\partial E_{\mathrm{T}}}{\partial t} + \nabla \cdot [\mathbf{u} (E_{\mathrm{T}} + P_{\mathrm{T}})] = -\rho \mathbf{u} \cdot \nabla \phi$$

$$\longrightarrow \quad \frac{\partial \mathbb{U}}{\partial t} + \nabla \cdot \mathbb{F}(\mathbb{U}) = \mathbb{S}(\mathbb{U})$$

Operator splitting: conservative update + source terms

Finite volume

Same applies for magnetohydronamics, radiation hydrodynamics, cosmic rays, and all non-thermal energy

Godunov scheme for general hyperbolic systems

The system of conservation laws

$$\partial_t \mathbf{U} + \partial_x \mathbf{F} = 0$$

is discretized using the following integral form:

$$\frac{\mathbf{U}_i^{n+1} - \mathbf{U}_i^n}{\Delta t} + \frac{\mathbf{F}_{i+1/2}^{n+1/2} - \mathbf{F}_{i-1/2}^{n+1/2}}{\Delta x} = 0$$

The time average flux function is computed using the self-similar solution of the inter-cell Riemann problem:

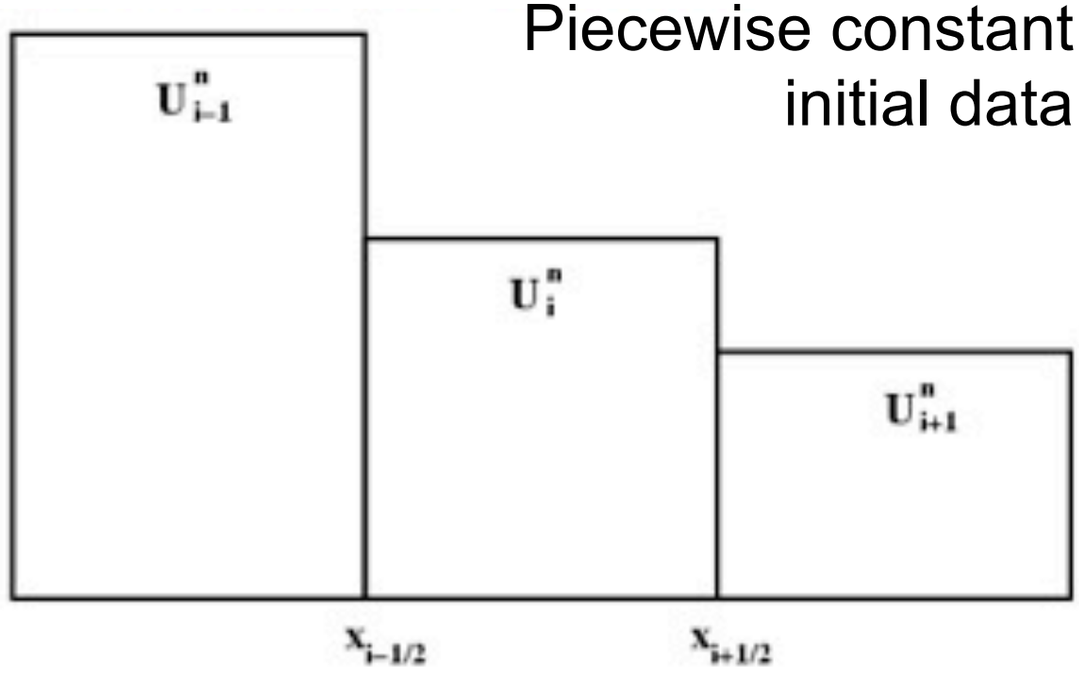
$$\mathbf{U}_{i+1/2}^*(x/t) = \mathcal{RP}\left[\mathbf{U}_i^n, \mathbf{U}_{i+1}^n\right]$$

$$\mathbf{F}_{i+1/2}^{n+1/2} = \mathbf{F}(\mathbf{U}_{i+1/2}^*(0))$$

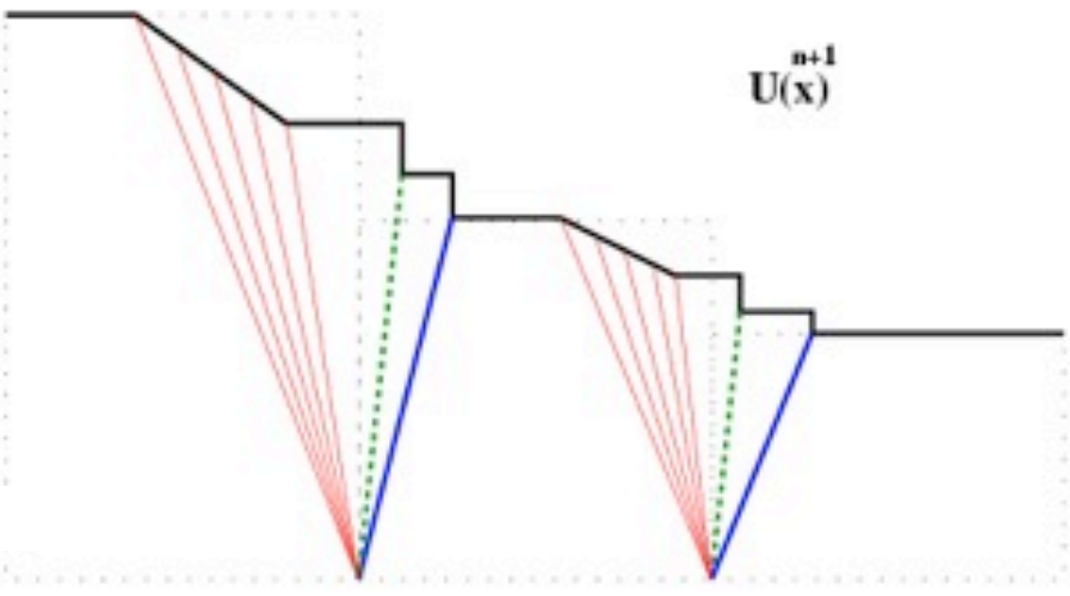
This defines the Godunov flux:

$$\mathbf{F}_{i+1/2}^{n+1/2} = \mathbf{F}^*(\mathbf{U}_i^n, \mathbf{U}_{i+1}^n)$$

Piecewise constant initial data



- **Godunov, S. K.** (1959), A Difference Scheme for Numerical Solution of Discontinuos Solution of Hydrodynamic Equations, *Math. Sbornik*, ***47***, *271-306*, translated US Joint Publ. Res. Service, JPRS 7226, 1969.

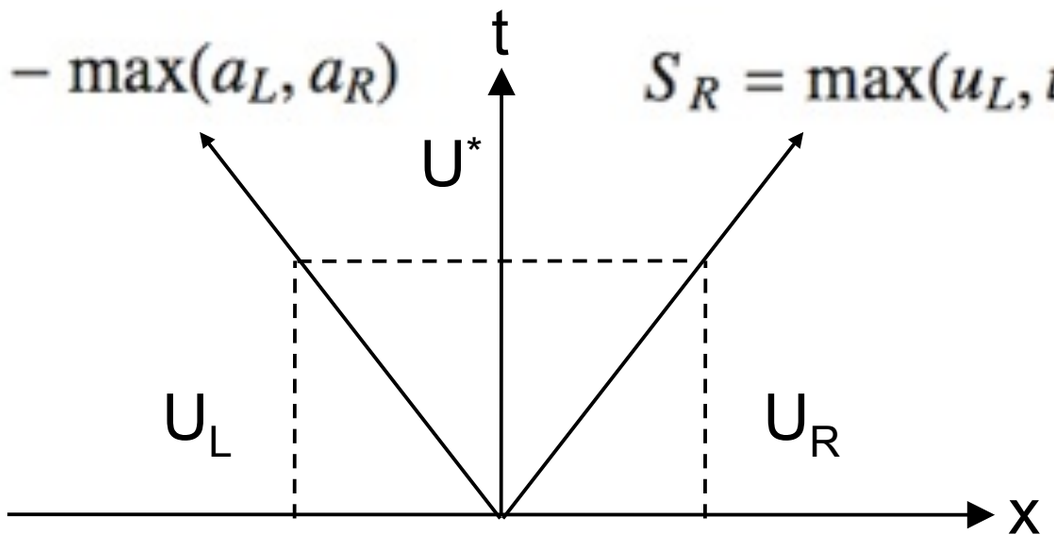


Advection: 1 wave, Euler: 3 waves, MHD: 7 waves

HLL Riemann solver

Approximate the true Riemann fan by 2 waves and 1 intermediate state:

$$S_L = \min(u_L, u_R) - \max(a_L, a_R) \qquad S_R = \max(u_L, u_R) + \max(a_L, a_R)$$



Compute U* using the integral form between $S_L t$ and $S_R t$

$$\mathbf{U}^*(\mathbf{U}_L, \mathbf{U}_R) = \frac{S_R \mathbf{U}_R - S_L \mathbf{U}_L - (\mathbf{F}_R - \mathbf{F}_L)}{S_R - S_L}$$

Compute F* using the integral form between $S_L t$ and 0.

$$S_L > 0 \quad \mathbf{F}^*(U_L, U_R) = \mathbf{F}_L$$

$$S_R < 0 \quad \mathbf{F}^*(U_L, U_R) = \mathbf{F}_R$$

$$S_L < 0 \text{ and } S_R > 0 \quad \mathbf{F}^*(U_L, U_R) = \frac{S_R \mathbf{F}_L - S_L \mathbf{F}_R + S_L S_R (\mathbf{U}_R - \mathbf{U}_L)}{S_R - S_L}$$

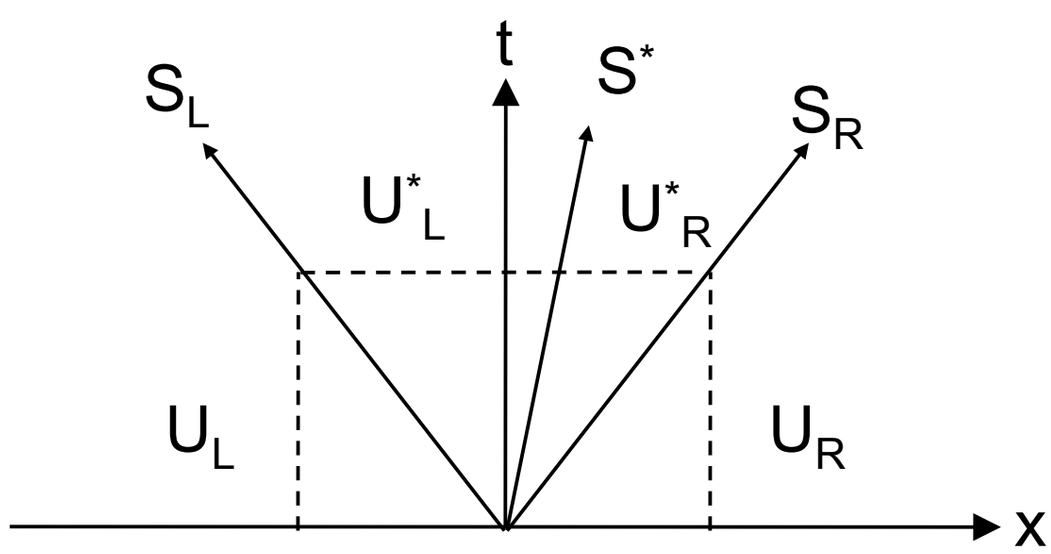
Other HLL-type Riemann solvers

Lax-Friedrich Riemann solver: $S_* = S_R = -S_L = \max(|u_L| + a_L, |u_R| + a_R)$

$$\mathbf{F}^*(U_L, U_R) = \frac{\mathbf{F}_L + \mathbf{F}_R}{2} - S_* \frac{\mathbf{U}_R - \mathbf{U}_L}{2}$$

HLLC Riemann solver: add a third wave for the contact (entropy) wave.

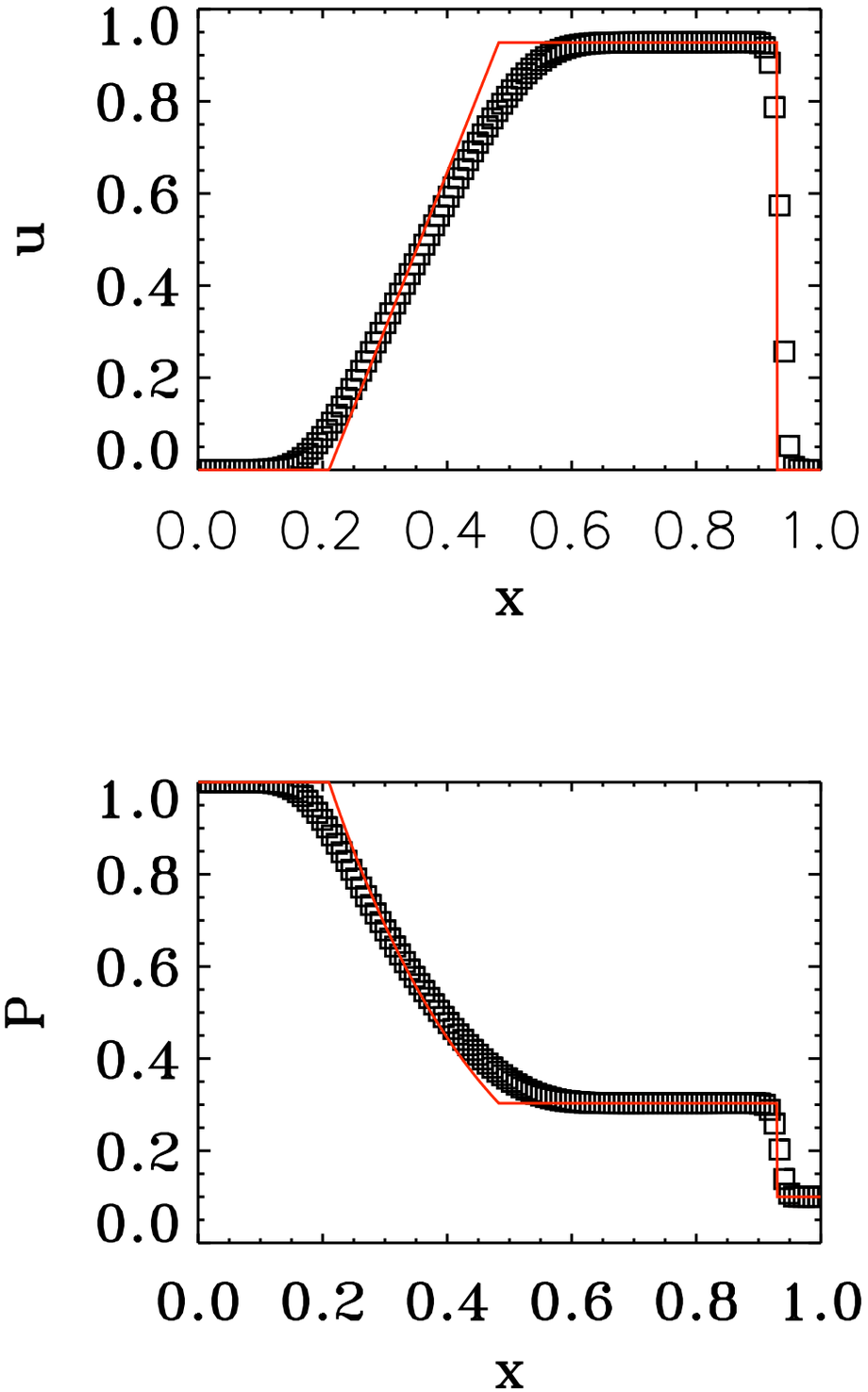
$$S_* = \frac{(\rho u)^*_{HLL}}{\rho^*_{HLL}}$$

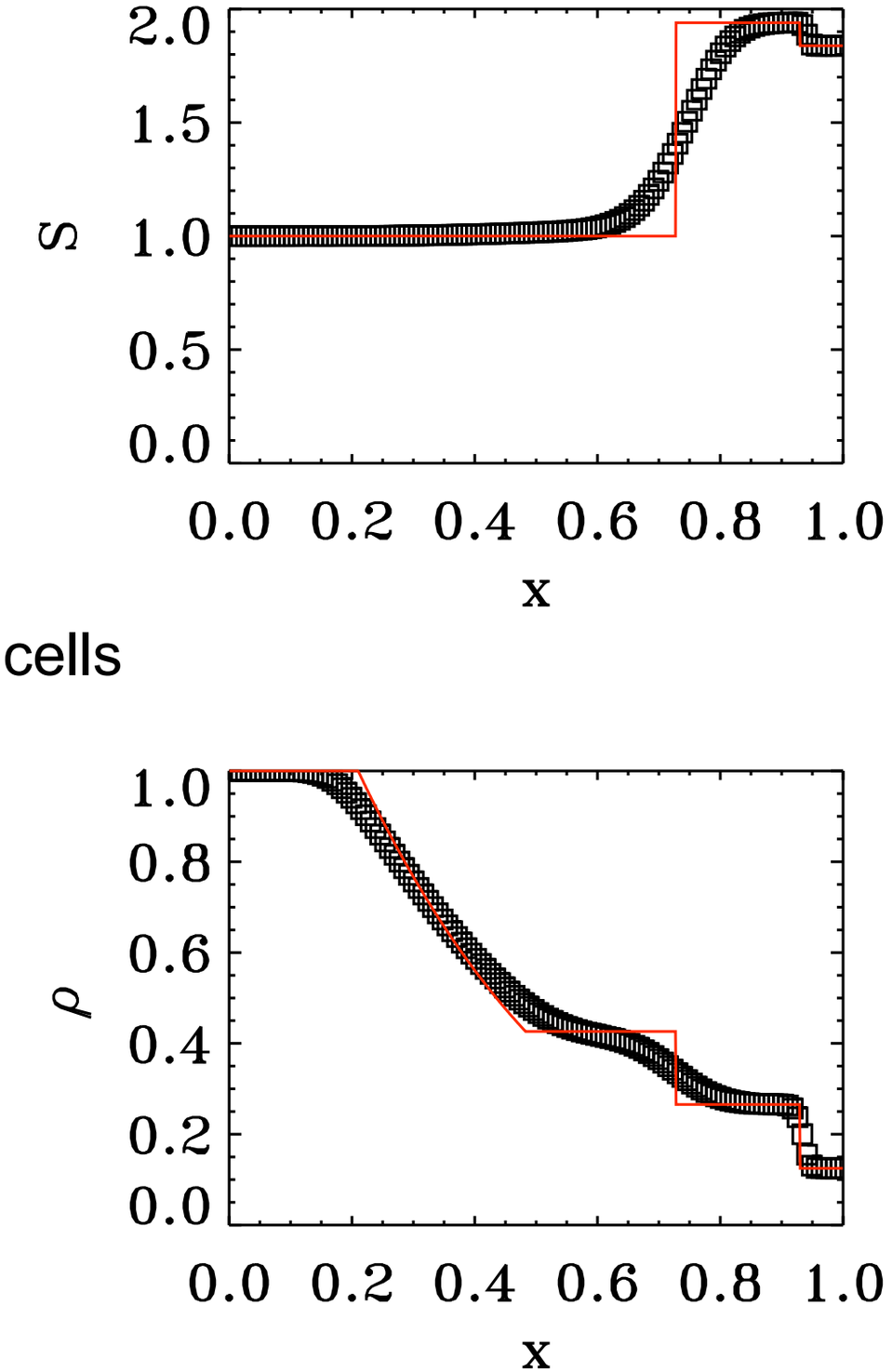


See Toro (1997) for details.

Sod test with the Godunov scheme

Lax-Friedrich Riemann solver

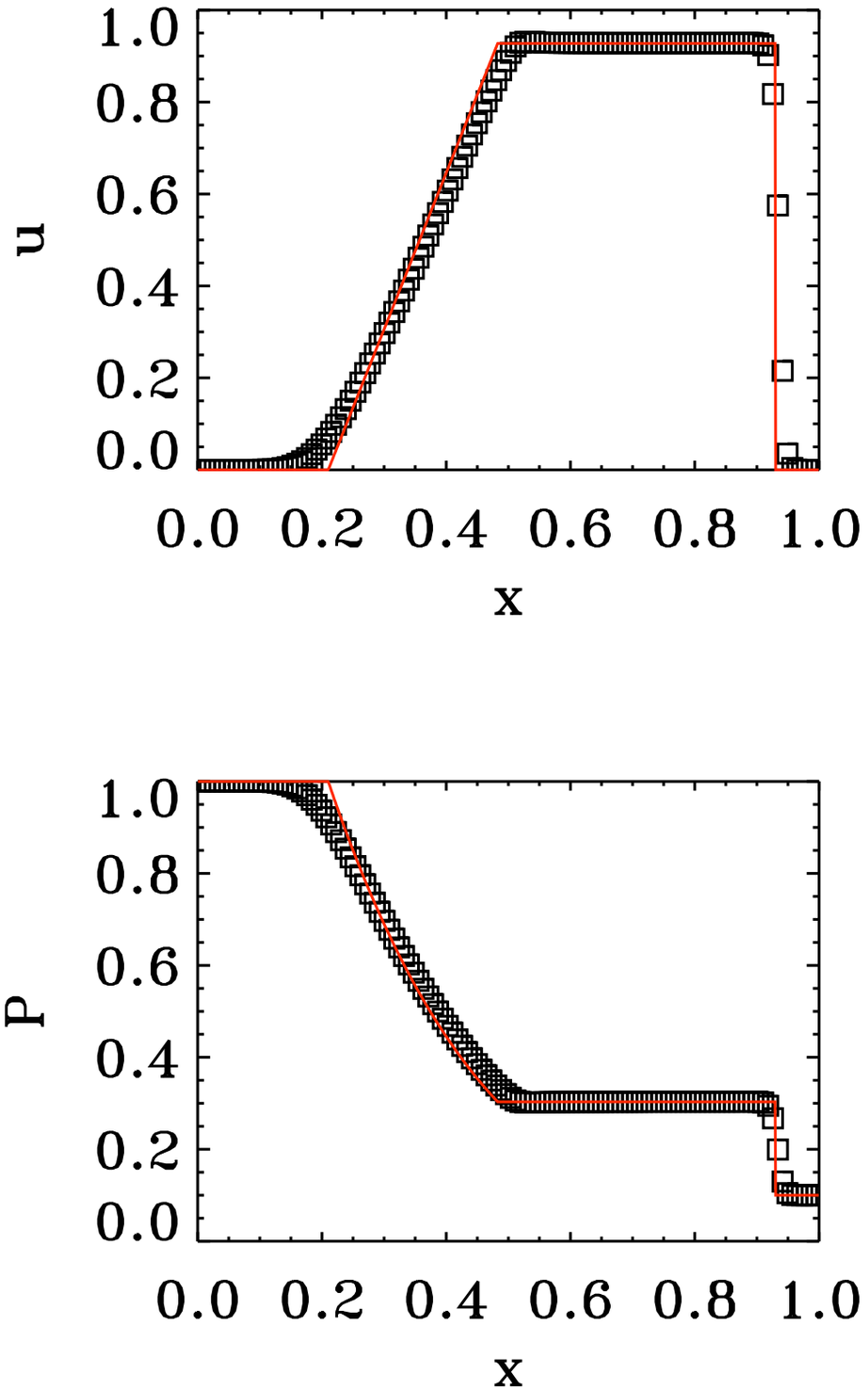


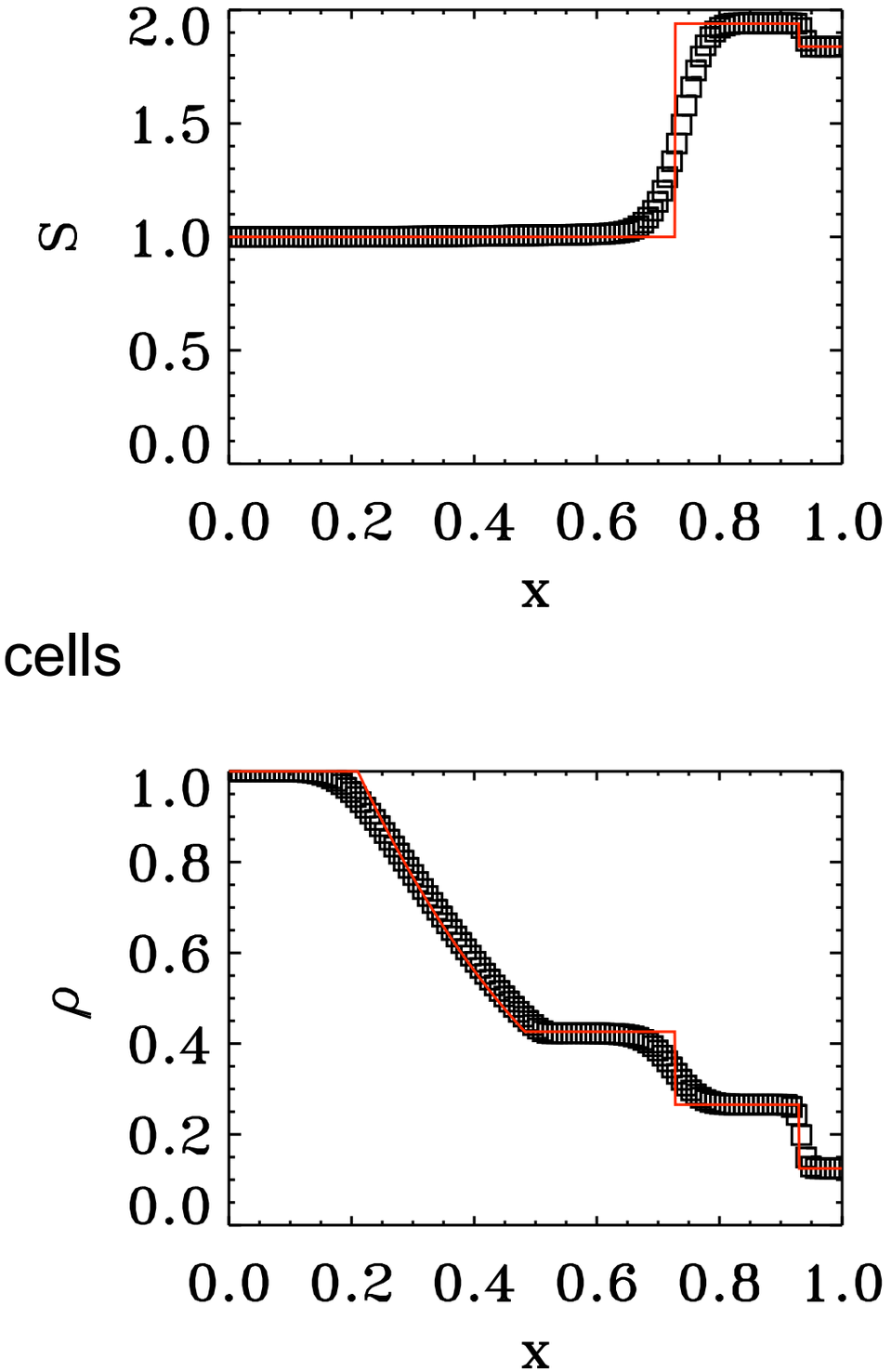


128 cells

Sod test with the Godunov scheme

HLLC Riemann solver

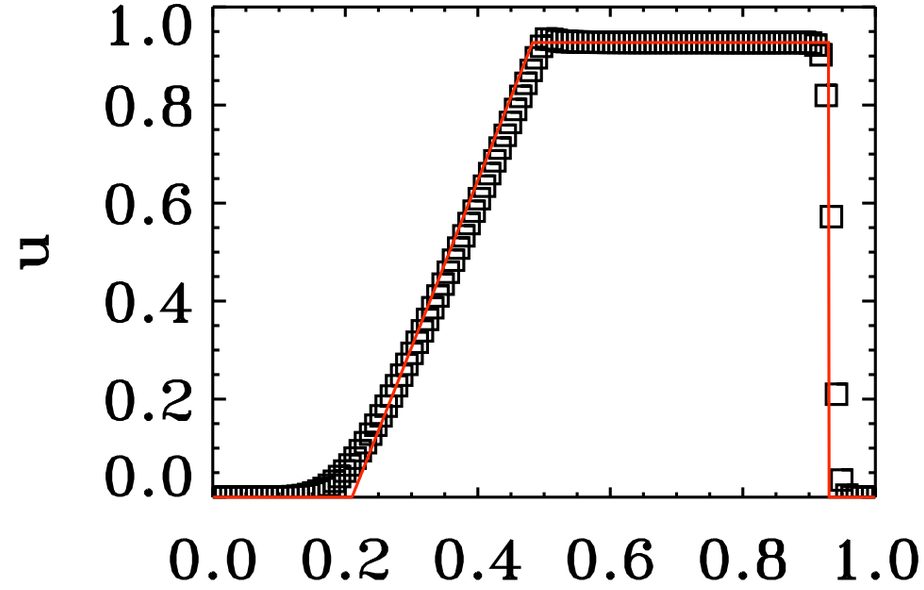


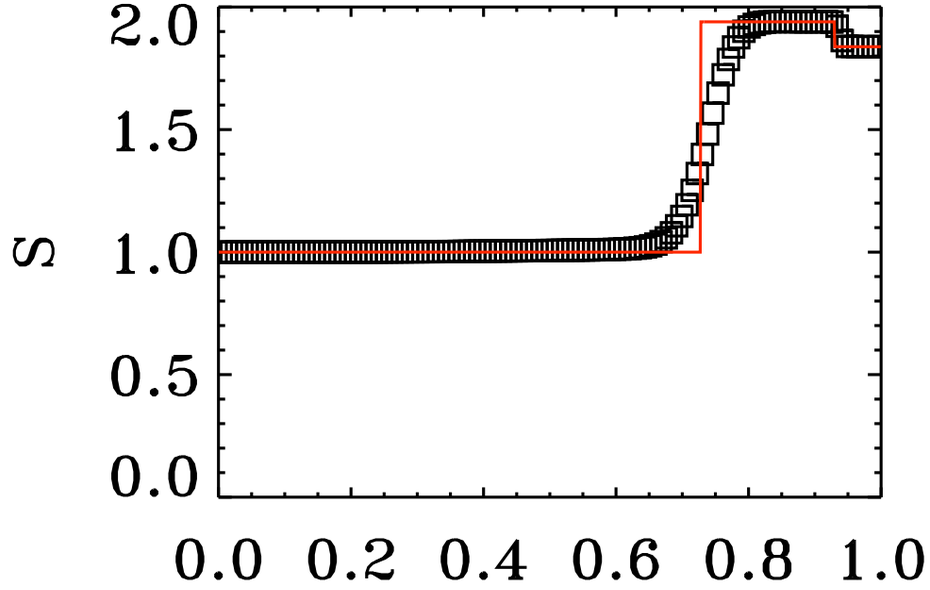


128 cells

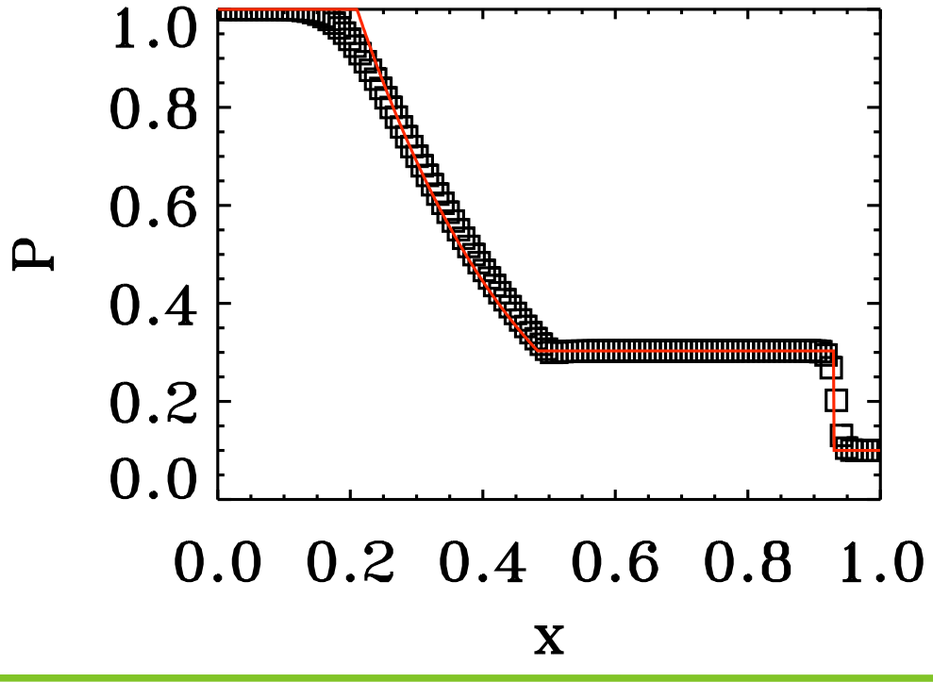
Sod test with the Godunov scheme

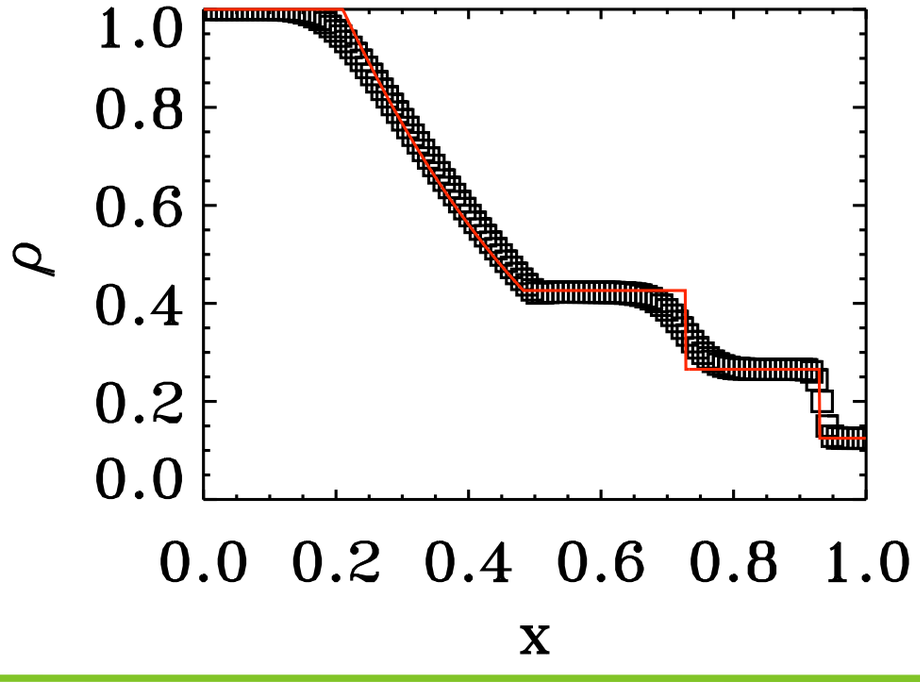
Exact Riemann solver





128 cells





Higher Order Godunov schemes

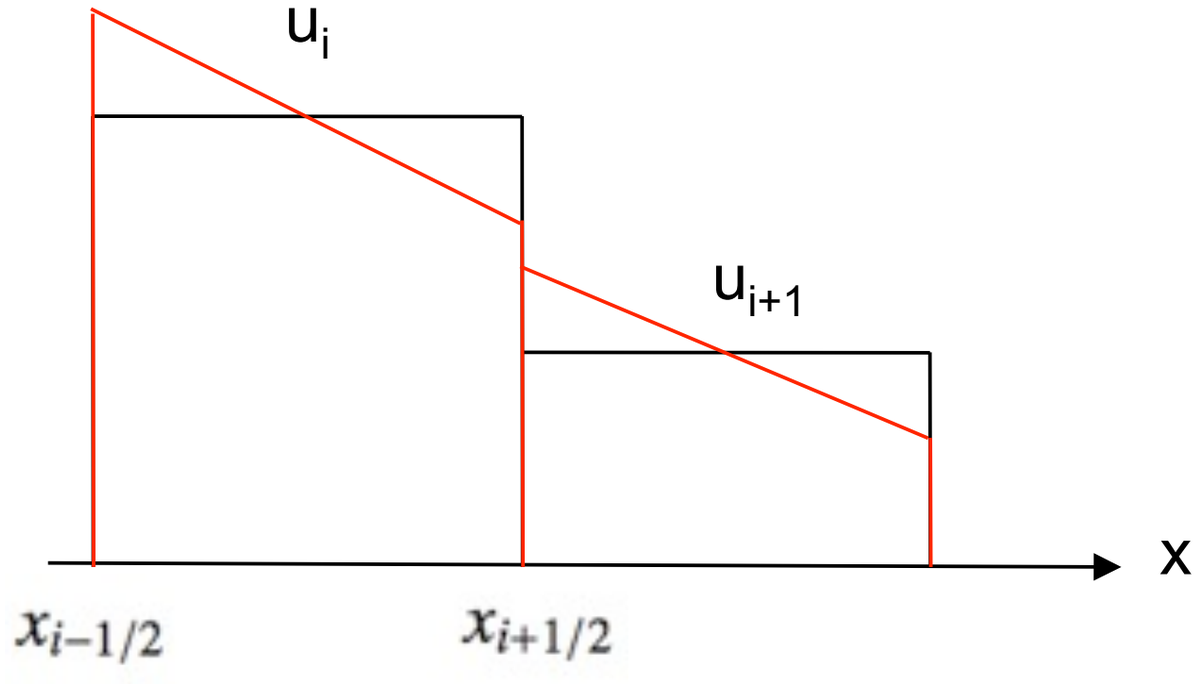
Godunov method is stable but very diffusive. It was abandoned for two decades, until…

Bram Van Leer

- **van Leer, B.** (1979), Towards the Ultimate Conservative Difference Scheme, V. A Second Order Sequel to Godunov's Method, *J. Com. Phys..*, *32*, 101–136.

Second Order Godunov scheme

Piecewise linear approximation of the solution:



The linear profile introduces a length scale: the Riemann solution is not self-similar anymore:

$$\mathbf{F}_{i+1/2}^{n+1/2} \neq \mathbf{F}(\mathbf{U}_{i+1/2}^{*}(0))$$

The flux function is approximated using a *predictor-corrector* scheme:

$$\mathbf{F}_{i+1/2}^{n+1/2} = \frac{1}{\Delta t}\int_{t^n}^{t^{n+1}} \mathbf{F}(x_{i+1/2}, t)\,\mathrm{d}t \quad \Longrightarrow \quad \mathbf{F}_{i+1/2}^{n+1/2} \simeq \mathbf{F}(\mathbf{U}_{i+1/2}^{*}(\frac{\Delta t}{2}))$$

The *corrected* Riemann solver has now *predicted* states as initial data:

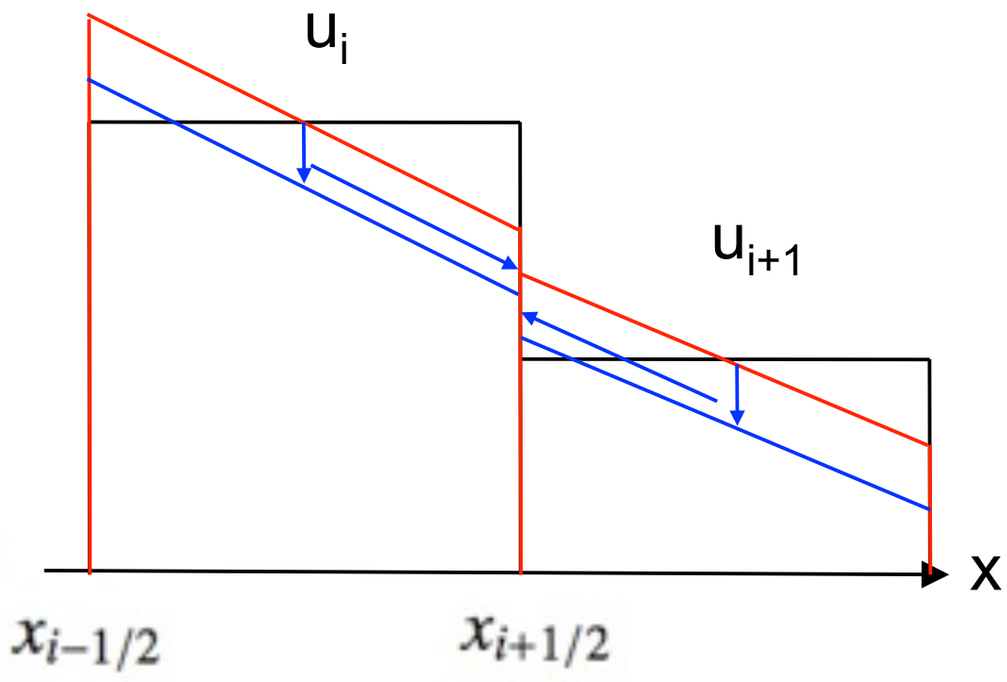
$$\mathbf{U}_{i+1/2}^{*}(x/t) = \mathcal{RP}\left[\mathbf{U}_{i+1/2,L}^{n+1/2}, \mathbf{U}_{i+1,R}^{n+1/2}\right]$$

Predictor Step for the advection equation

The predicted states are computed using a Taylor expansion in space and time:

$$u_{i+1/2,L}^{n+1/2} = u_i^n + \frac{\Delta t}{2}\left(\frac{\partial u}{\partial t}\right)_i + \frac{\Delta x}{2}\left(\frac{\partial u}{\partial x}\right)_i$$

$$u_{i+1/2,R}^{n+1/2} = u_{i+1}^n + \frac{\Delta t}{2}\left(\frac{\partial u}{\partial t}\right)_{i+1} - \frac{\Delta x}{2}\left(\frac{\partial u}{\partial x}\right)_{i+1}$$



Second order predicted states are the new initial conditions for the Riemann solver:

$$u_{i+1/2,L}^{n+1/2} = u_i^n + (1-C)\frac{\Delta x}{2}\left(\frac{\partial u}{\partial x}\right)_i \qquad u_{i+1/2,R}^{n+1/2} = u_{i+1}^n - (1+C)\frac{\Delta x}{2}\left(\frac{\partial u}{\partial x}\right)_{i+1}$$

The *corrected* flux function is the *upwind* predicted state:

$$f_{i+1/2}^{n+1/2} = a u_{i+1/2,L}^{n+1/2} \quad \text{if} \quad a > 0 \qquad f_{i+1/2}^{n+1/2} = a u_{i+1/2,R}^{n+1/2} \quad \text{if} \quad a < 0$$

Summary: the MUSCL scheme for systems

Compute second order predicted states using a Taylor expansion:

$$\begin{cases} \mathbf{W}_{i+1/2,L}^{n+1/2} = \mathbf{W}_i^n + \dfrac{\Delta t}{2}\left(\dfrac{\partial \mathbf{W}}{\partial t}\right)_i + \dfrac{\Delta x}{2}\left(\dfrac{\partial \mathbf{W}}{\partial x}\right)_i \\ \mathbf{W}_{i+1/2,R}^{n+1/2} = \mathbf{W}_{i+1}^n + \dfrac{\Delta t}{2}\left(\dfrac{\partial \mathbf{W}}{\partial t}\right)_{i+1} - \dfrac{\Delta x}{2}\left(\dfrac{\partial \mathbf{W}}{\partial x}\right)_{i+1} \end{cases}$$

$$\begin{cases} \mathbf{W}_{i+1/2,L}^{n+1/2} = \mathbf{W}_i^n + (\mathbf{I} - \mathbf{A}\dfrac{\Delta t}{\Delta x})\dfrac{\Delta x}{2}\left(\dfrac{\partial \mathbf{W}}{\partial x}\right)_i \\ \mathbf{W}_{i+1/2,R}^{n+1/2} = \mathbf{W}_{i+1}^n - (\mathbf{I} + \mathbf{A}\dfrac{\Delta t}{\Delta x})\dfrac{\Delta x}{2}\left(\dfrac{\partial \mathbf{W}}{\partial x}\right)_{i+1} \end{cases}$$

Update conservative variables using corrected Godunov fluxes

$$\mathbf{F}_{i+1/2}^{n+1/2} = \mathbf{F}^*(\mathbf{W}_{i+1/2,L}^{n+1/2}, \mathbf{W}_{i+1/2,R}^{n+1/2}) \qquad \frac{\mathbf{U}_i^{n+1} - \mathbf{U}_i^n}{\Delta t} + \frac{\mathbf{F}_{i+1/2}^{n+1/2} - \mathbf{F}_{i-1/2}^{n+1/2}}{\Delta x} = 0$$

Monotonicity preserving schemes

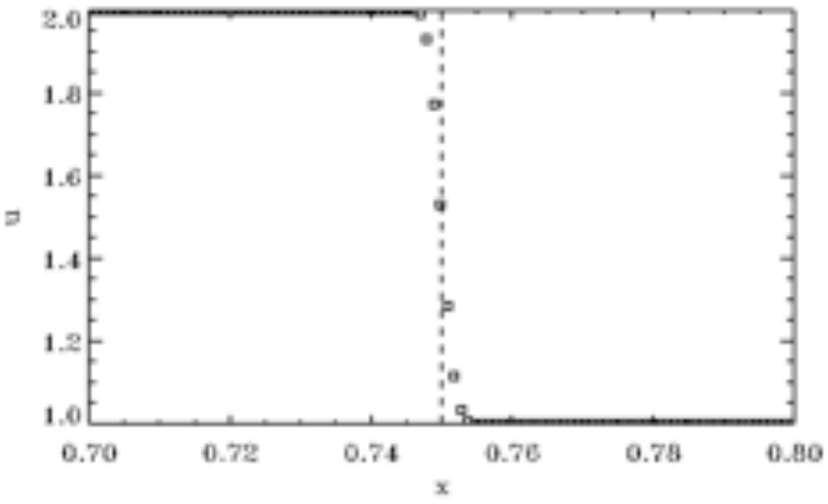
We use the central finite difference approximation for the slope:

$$\left(\frac{\partial u}{\partial x}\right)_i = \frac{\Delta u_i}{\Delta x} = \frac{u_{i+1} - u_{i-1}}{2}$$

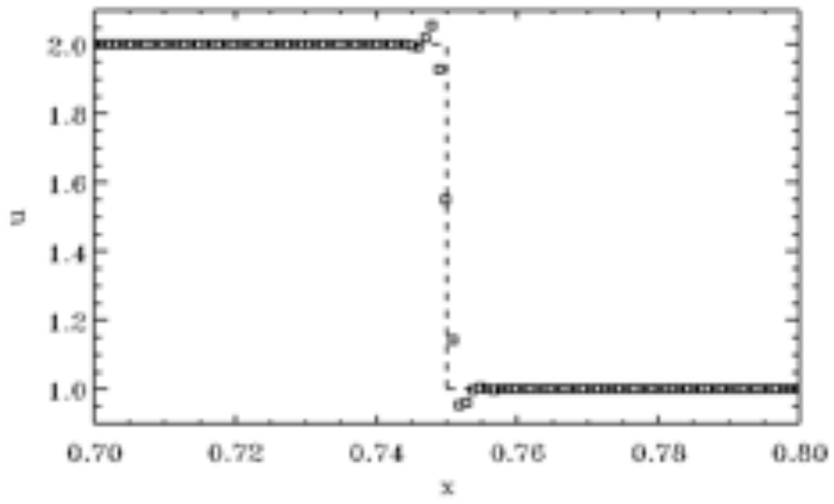
Second order linear scheme.

In this case, the solution is oscillatory, and therefore non physical.

first order



second order



Oscillations are due to the *non monotonicity* of the numerical scheme.

A scheme is monotonicity preserving if:

- No new local extrema are created in the solution
- Local minimum (maximum) non decreasing (increasing) function of time.

Godunov theorem: only first order linear schemes are monotonicity preserving !

Slope limiters

Harten introduced the Total Variation of the numerical solution:

$$TV^n = \sum_i^n |u_{i+1} - u_i|$$

Harten's theorem: a Total Variation Diminishing (TVD) scheme is monotonicity preserving.

$$TV^{n+1} \leq TV^n$$

Design non-linear TVD second order scheme using slope limiters:

$$\left(\frac{\partial u}{\partial x}\right)_i = \frac{\Delta u_i}{\Delta x} = \lim(u_{i-1}, u_i, u_{i+1}) \left(\frac{u_{i+1} - u_{i-1}}{2}\right)$$

where the slope limiter is a non-linear function satisfying:

$$0 \leq \lim(u_{i-1}, u_i, u_{i+1}) \leq 1$$

- Harten, Ami (1983), "High resolution schemes for hyperbolic conservation laws", *J. Comput. Phys* **49**: 357-393, doi:10.1006/jcph.1997.5713

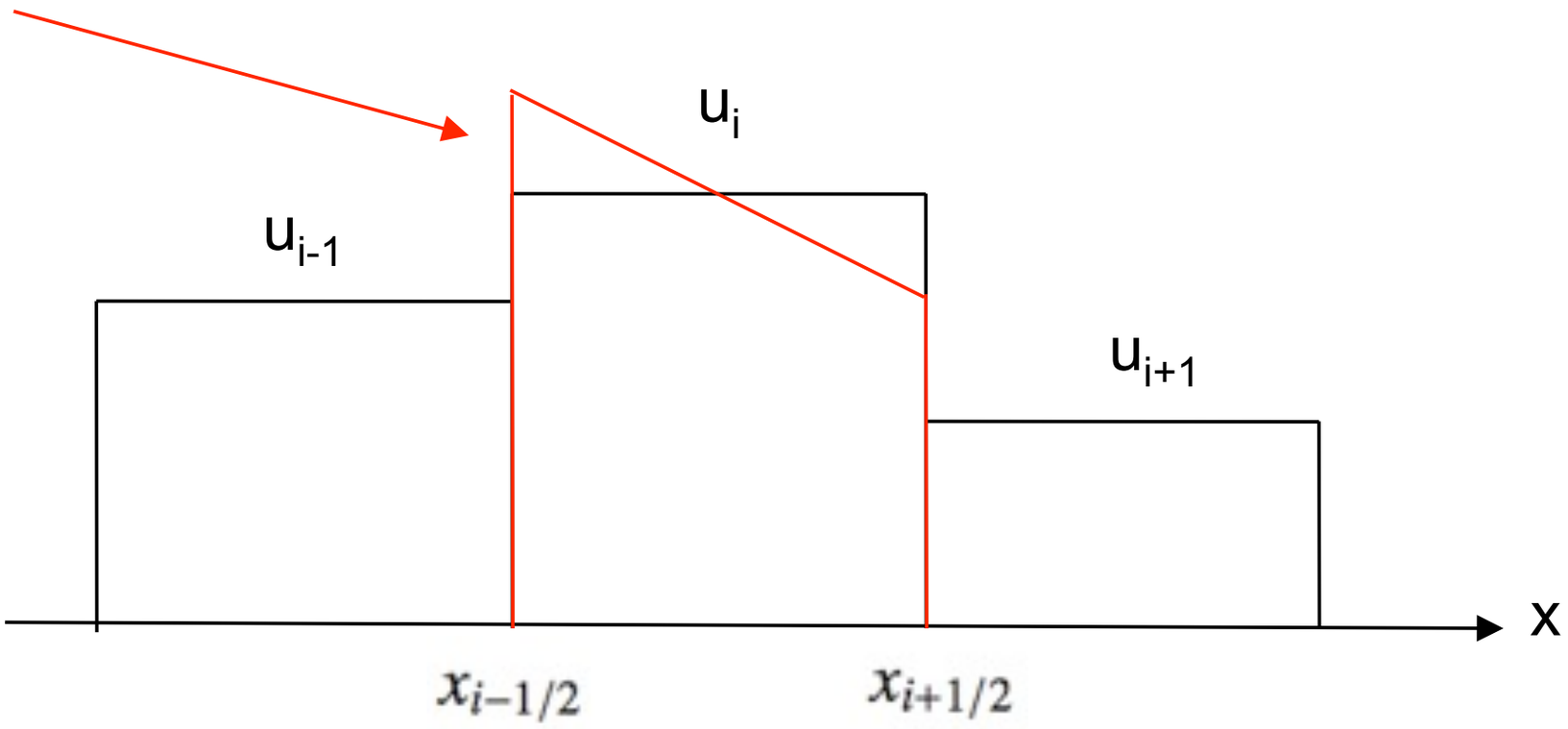
No local extrema

$$\left(\frac{\partial u}{\partial x}\right)_i = \frac{\Delta u_i}{\Delta x} = \lim(u_{i-1}, u_i, u_{i+1})\left(\frac{u_{i+1} - u_{i-1}}{2}\right)$$

We define 3 local slopes: left, right and central slopes

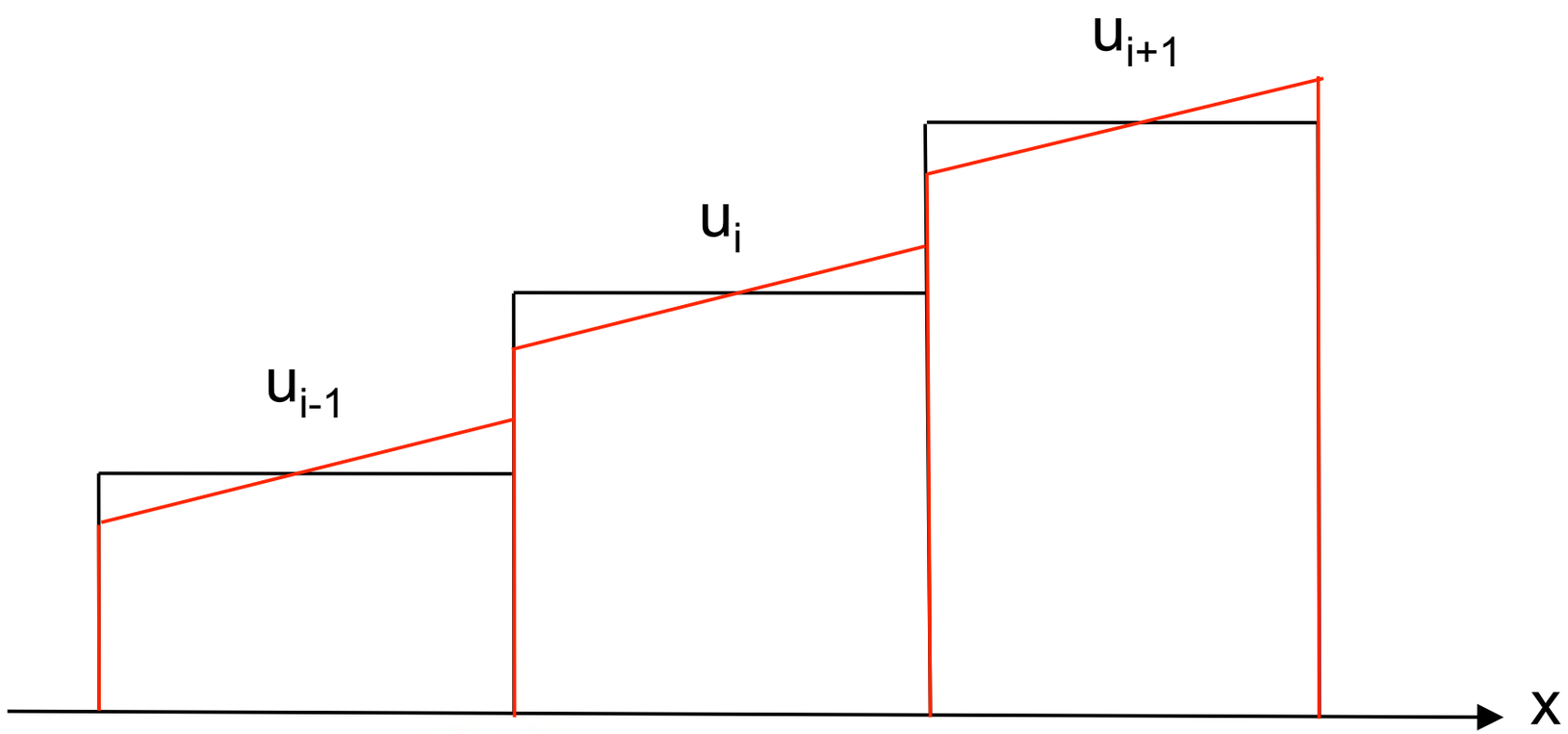
$$\Delta u_L = u_i - u_{i-1} \quad \Delta u_R = u_{i+1} - u_i \quad \text{and} \quad \Delta u_C = \frac{u_{i+1} - u_{i-1}}{2}$$

New maximum !



For all slope limiters: $\Delta u_i = 0 \quad \text{if} \quad \Delta u_L \Delta u_R < 0$

The *minmod* slope



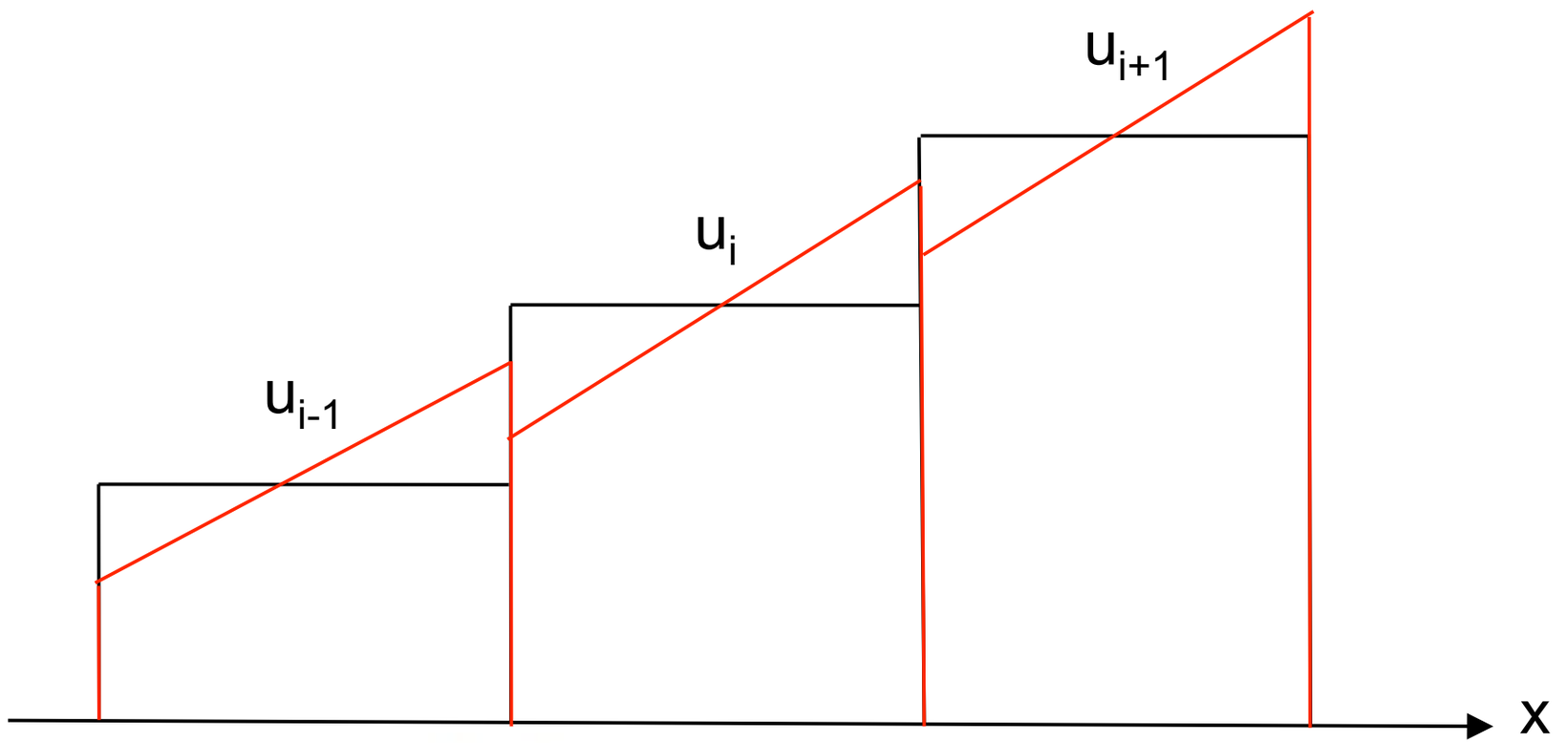
Linear reconstruction is monotone at time t^n

$$u^n_{i+1/2,L} = u^n_i + \frac{\Delta u_i}{2} \qquad u^n_{i-1/2,R} = u^n_i - \frac{\Delta u_i}{2}$$

Minmod slope limiting is never truly second order !

$$u^n_{i+1/2,L} \leq u^n_{i+1/2,R} \qquad \Delta u_i = \min(\Delta u_L, \Delta u_R)$$

The *moncen* slope



$x_{i-1/2}$ $x_{i+1/2}$

Extreme values must be bounded by the *initial average* states.

$$u^n_{i-1/2,R} = u^n_i - \frac{\Delta u_i}{2} \qquad u^n_{i+1/2,L} = u^n_i + \frac{\Delta u_i}{2}$$

$$u^n_{i-1} \le u^n_{i-1/2,R} \le u^n_i \qquad u^n_i \le u^n_{i+1/2,L} \le u^n_{i+1}$$

$$\Delta u_i = \min(2\Delta u_L, \Delta u_C, 2\Delta u_R)$$

The *superbee* slope

Predicted states must be bounded by the initial average states.

$$u_{i+1/2,L}^{n+1/2} = u_i^n + (1 - C)\frac{\Delta u_i}{2}$$

$$u_{i+1/2,R}^{n+1/2} = u_{i+1}^n - (1 + C)\frac{\Delta u_{i+1}}{2}$$

TVD constraint is preserved by the Riemann solver.

$$u_i^n \le u_{i+1/2,L}^{n+1/2} \le u_{i+1}^n$$

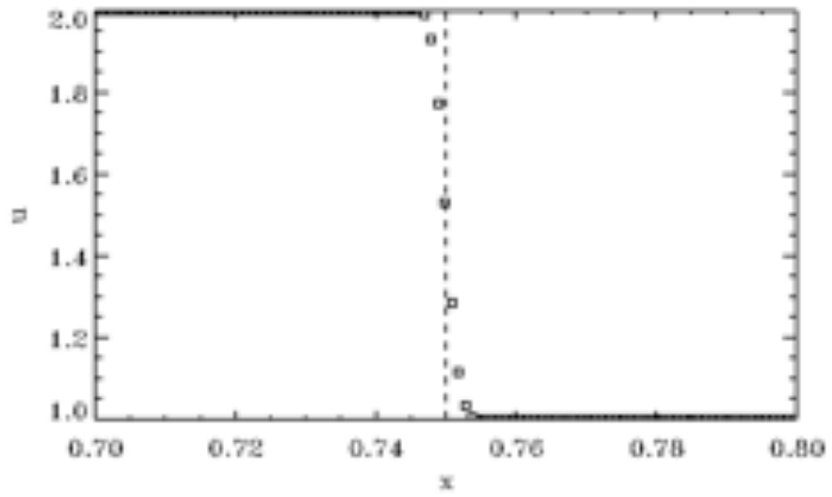
$$u_{i-1}^n \le u_{i-1/2,R}^{n+1/2} \le u_i^n$$

The Courant factor now enters the slope definition.

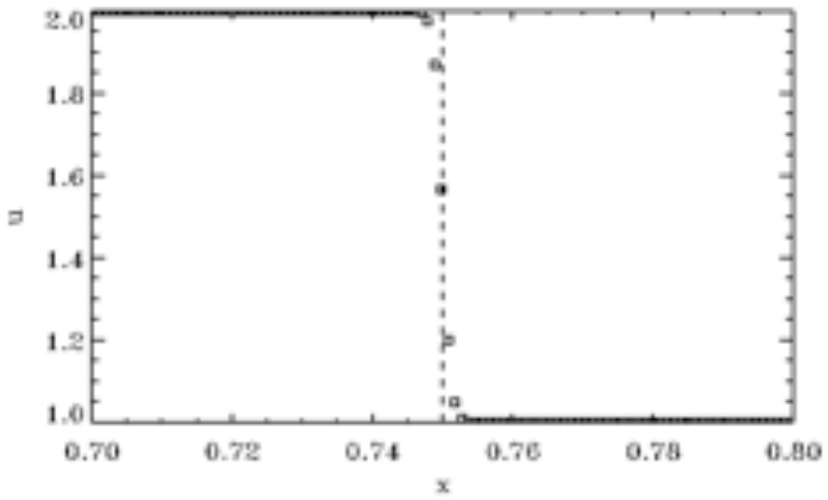
$$\Delta u_i = \min\left(\frac{2}{1+C}\Delta u_L, \frac{2}{1-C}\Delta u_R\right)$$

Summary: slope limiters

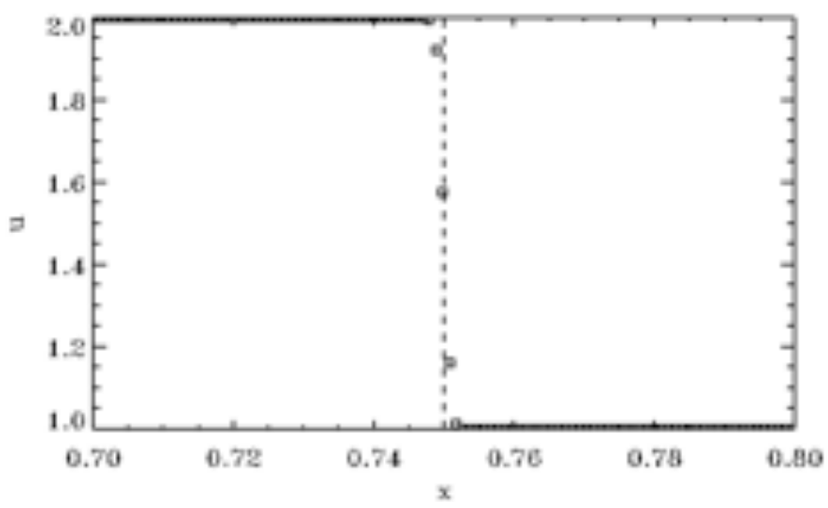
first order



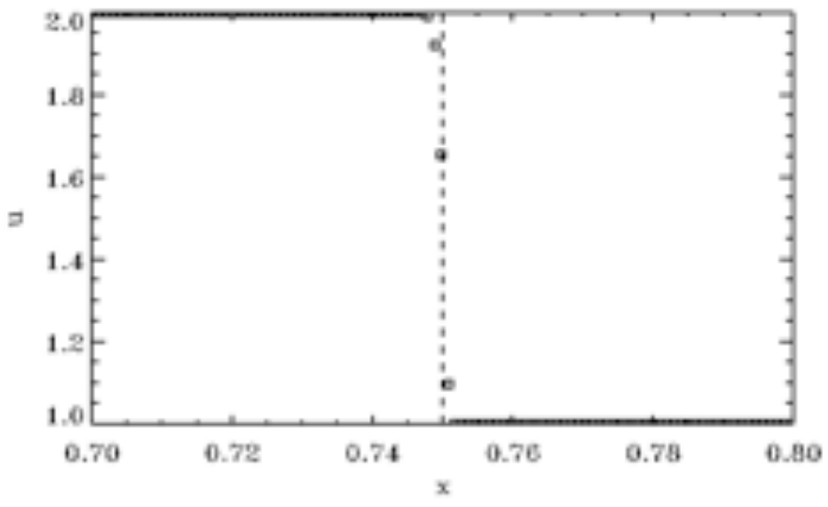
minmod



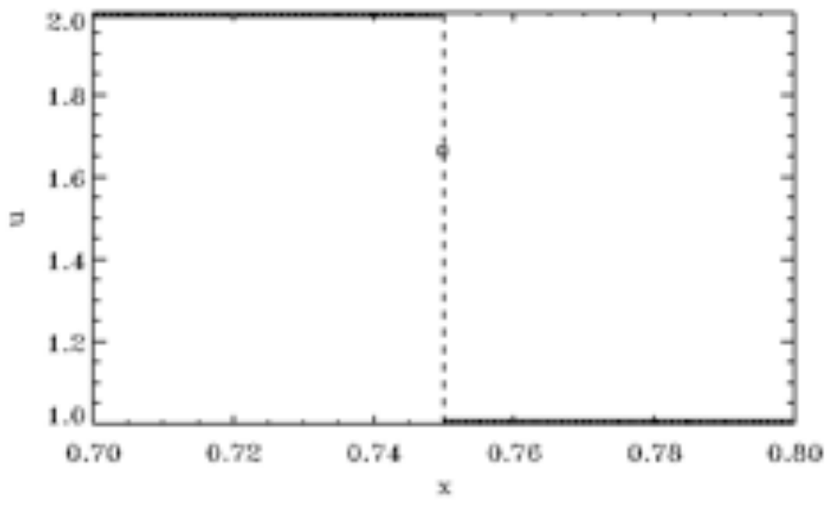
moncen



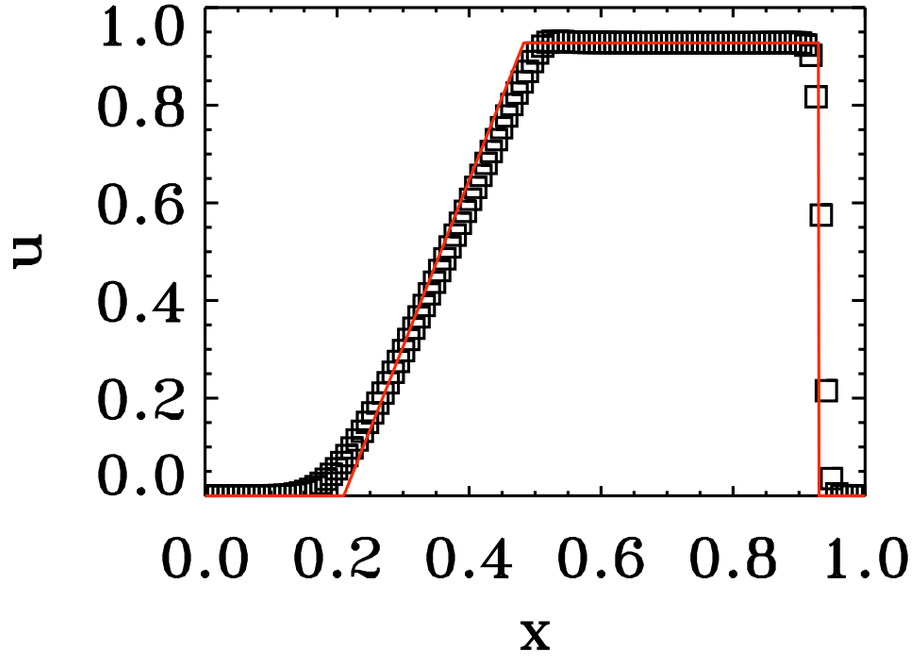
superbee

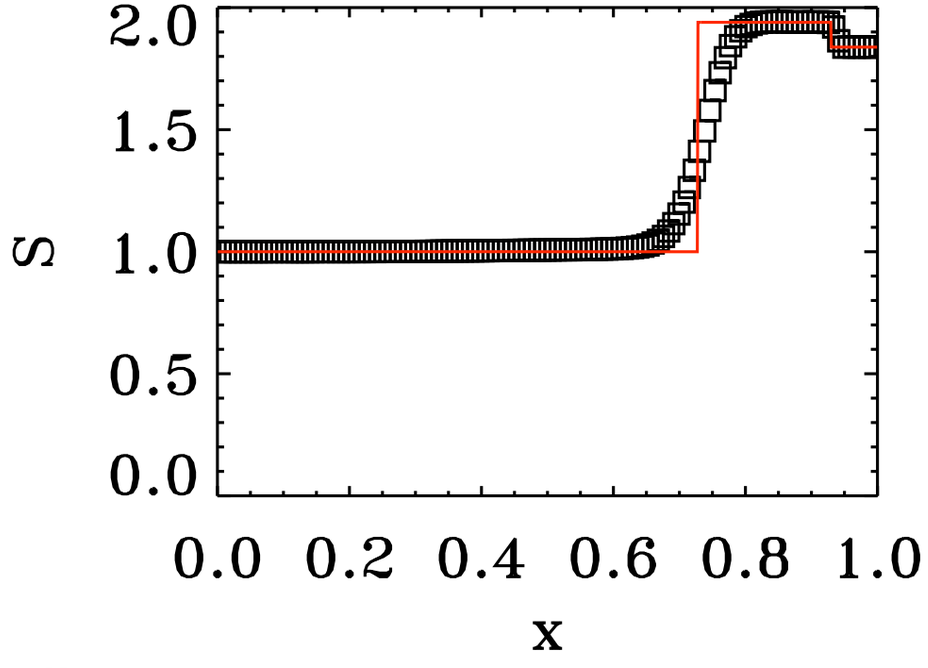


ultrabee

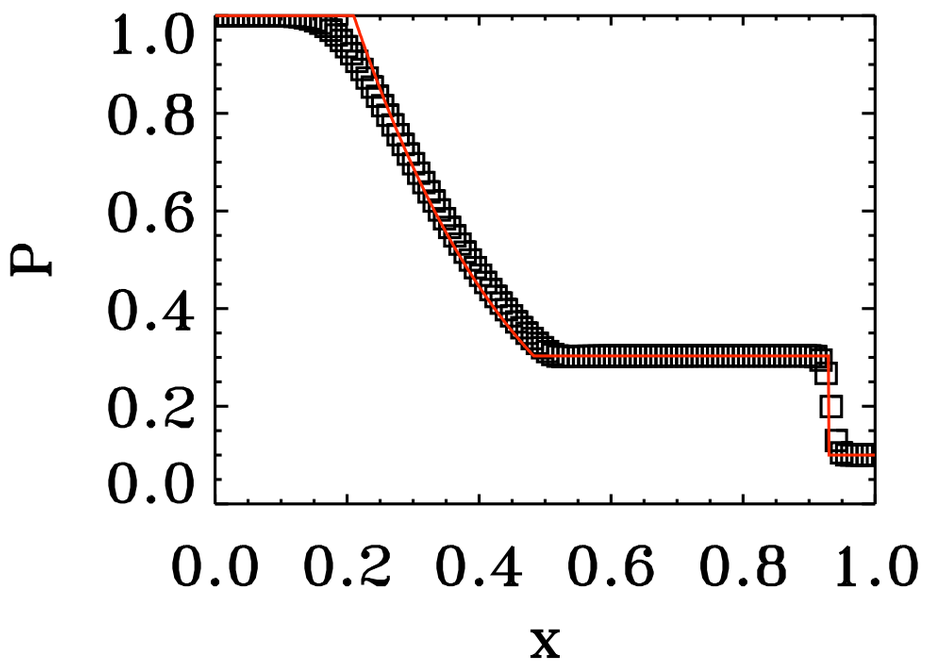


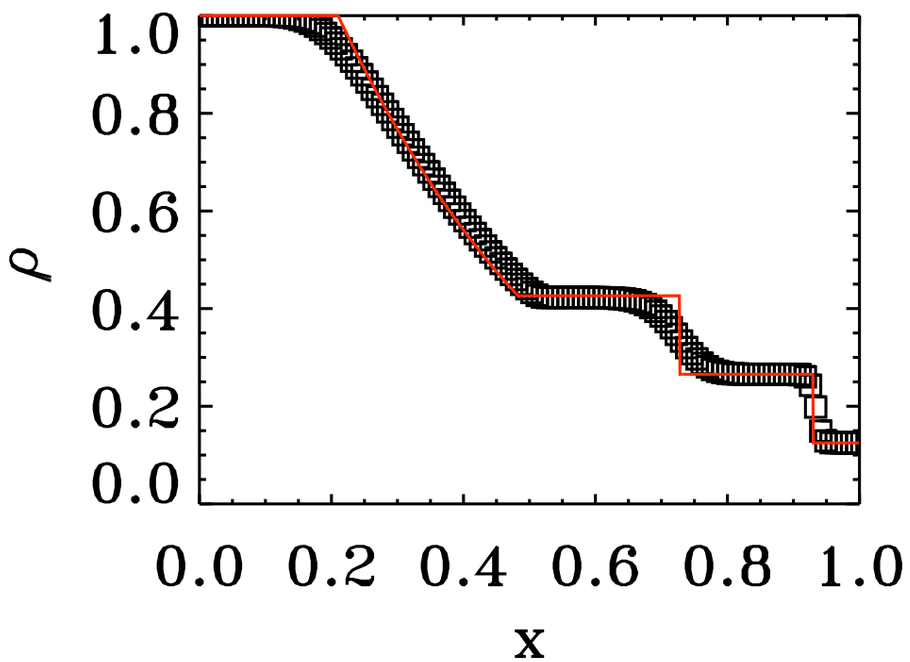
Sod test with HLLC first order



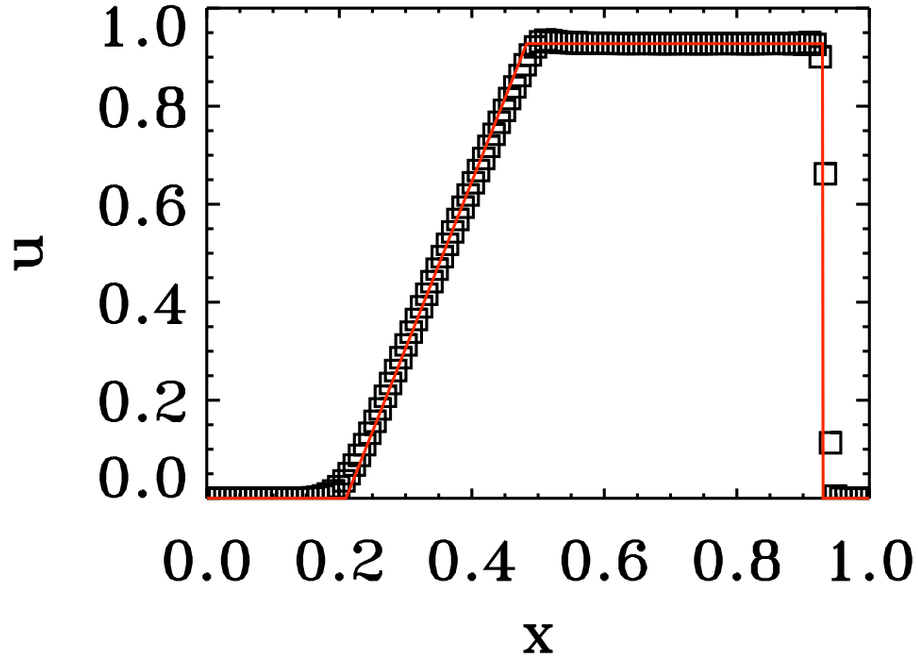


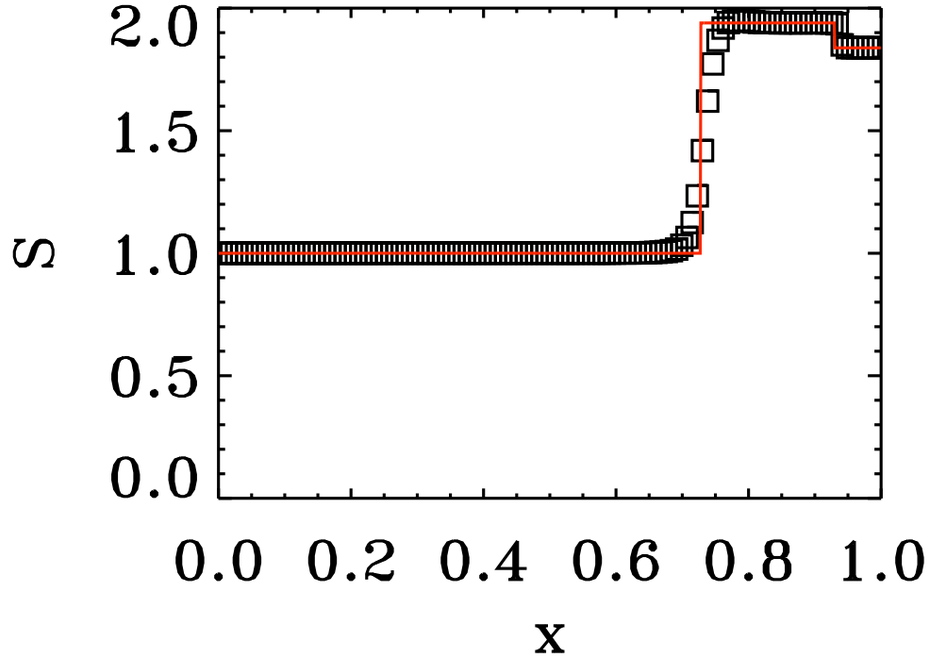
128 cells



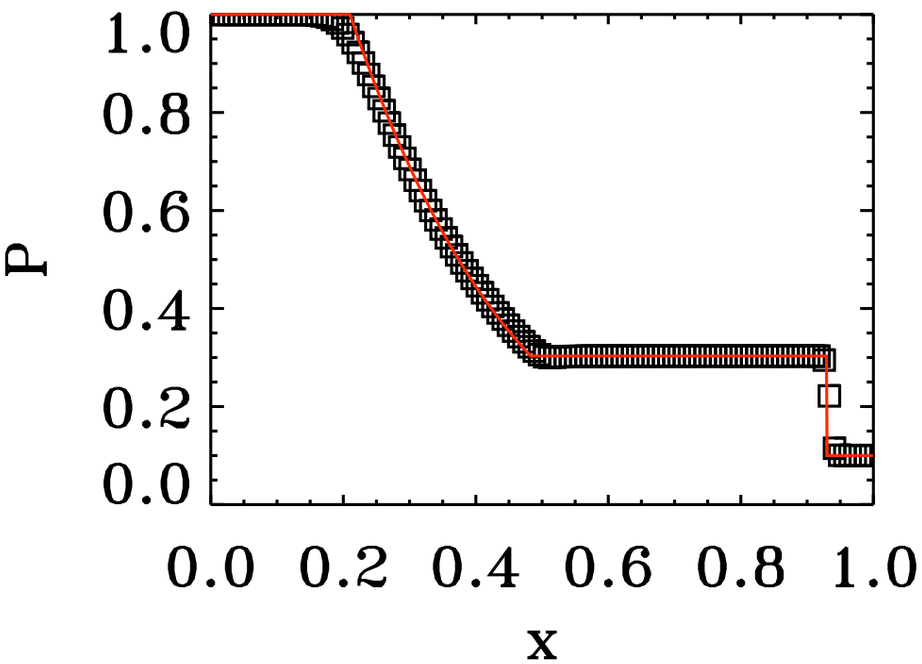


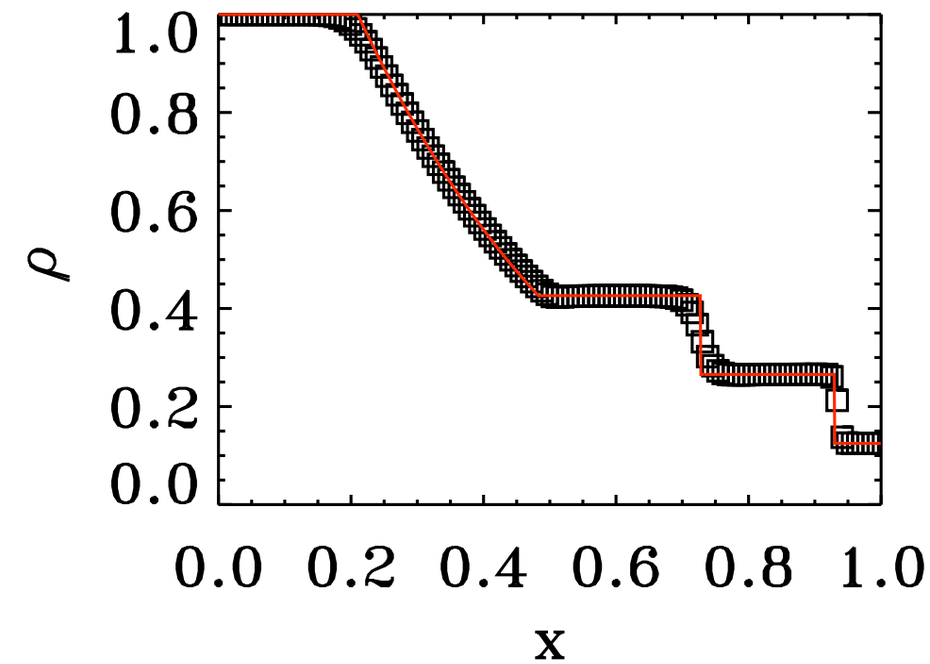
Sod test with HLLC and MinMod



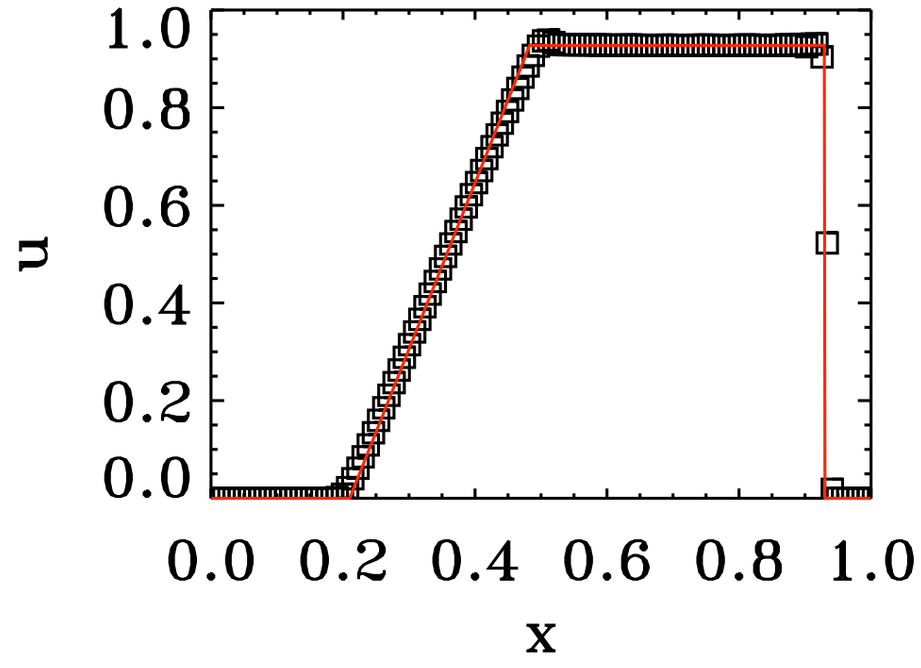


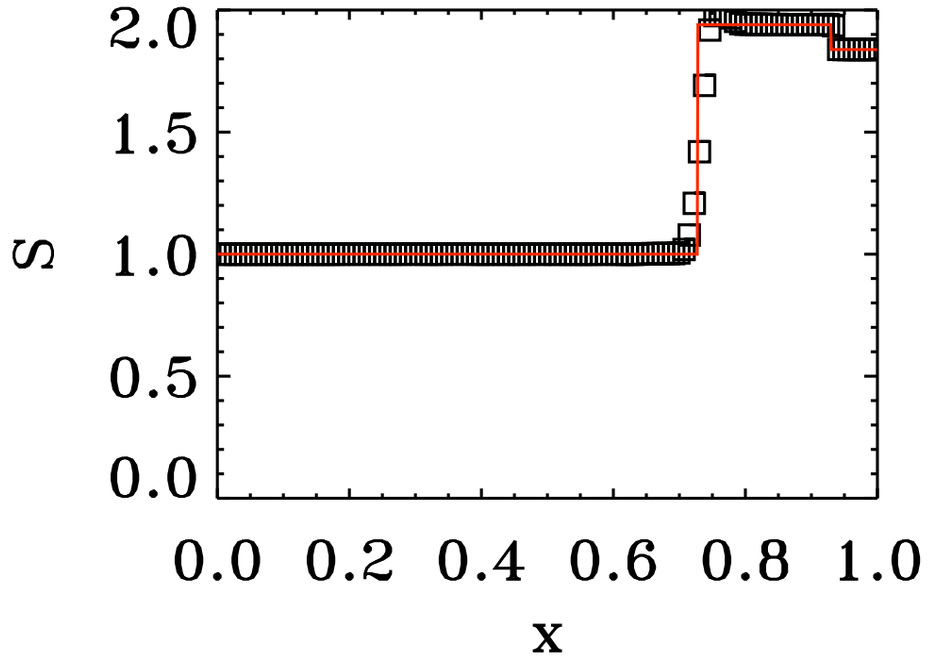
128 cells



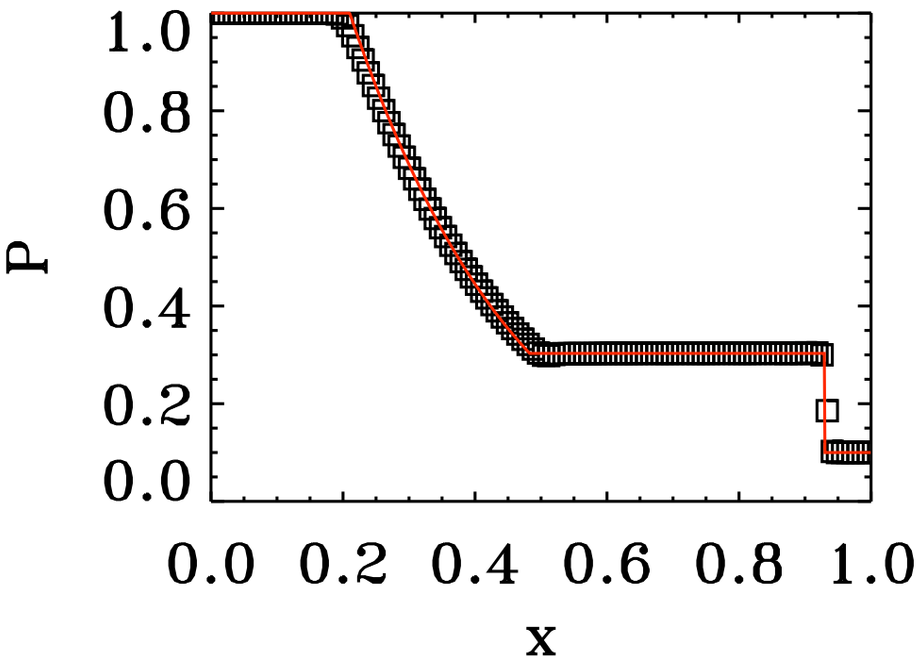


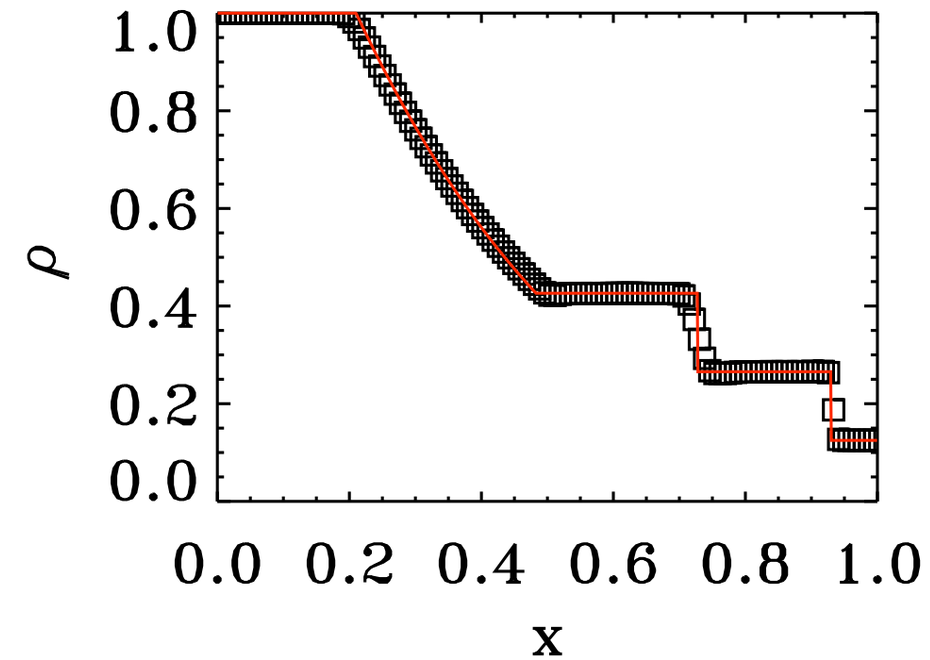
Sod test with HLLC and MonCen





128 cells





Multidimensional Godunov schemes

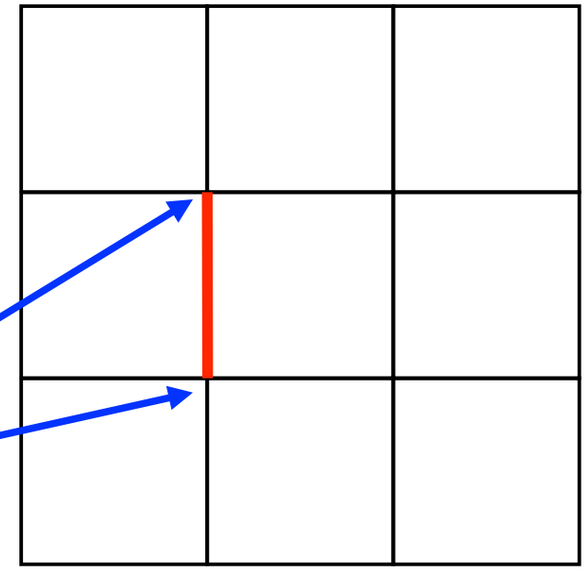
2D Euler equations in integral (conservative) form

$$\mathbf{U}_{i,j}^{n+1} - \mathbf{U}_{i,j}^{n} + \frac{\Delta t}{\Delta x}\left(\mathbf{F}_{i+1/2,j}^{n+1/2} - \mathbf{F}_{i-1/2,j}^{n+1/2}\right) + \frac{\Delta t}{\Delta y}\left(\mathbf{G}_{i,j+1/2}^{n+1/2} - \mathbf{G}_{i,j-1/2}^{n+1/2}\right) = 0$$

Flux functions are now time and space average.

$$\mathbf{F}_{i+1/2,j}^{n+1/2} = \frac{1}{\Delta t}\frac{1}{\Delta y}\int_{t^n}^{t^{n+1}}\int_{y_{j-1/2}}^{y_{j+1/2}} \mathbf{F}(x_{i+1/2}, y, t)\,\mathrm{d}t\,\mathrm{d}y$$

$$\mathbf{G}_{i,j+1/2}^{n+1/2} = \frac{1}{\Delta t}\frac{1}{\Delta x}\int_{t^n}^{t^{n+1}}\int_{x_{i-1/2}}^{x_{i+1/2}} \mathbf{G}(x, y_{j+1/2}, t)\,\mathrm{d}t\,\mathrm{d}x$$



2D Riemann problems interact along cell edges:

$$\mathbf{U}^{*}_{i+1/2,j+1/2}(x/t, y/t) = \mathcal{RP}\left[\langle\mathbf{U}\rangle^{n}_{i,j}, \langle\mathbf{U}\rangle^{n}_{i+1,j}\langle\mathbf{U}\rangle^{n}_{i,j+1}\langle\mathbf{U}\rangle^{n}_{i+1,j+1}\right]$$

Even at first order, self-similarity does not apply to the flux functions anymore.

Predictor-corrector schemes ?

The ideal MHD equations in conservative forms

Mass conservation $$\partial_t \rho + \nabla \cdot (\rho \mathbf{u}) = 0$$

Momentum conservation $$\partial_t (\rho \mathbf{u}) + \nabla \cdot (\rho \mathbf{u}\mathbf{u} - \frac{1}{4\pi}\mathbf{B}\mathbf{B}) + \nabla P_{tot} = 0$$

Total energy conservation $$\partial_t E + \nabla \cdot \left[(E + P_{tot})\mathbf{u} - \frac{1}{4\pi}\mathbf{B}(\mathbf{B} \cdot \mathbf{u}) \right] = 0$$

Magnetic flux conservation $$\partial_t \mathbf{B} + \nabla \times (\mathbf{B} \times \mathbf{u}) = 0$$

Total energy $$E = \rho\epsilon + \frac{1}{2}\rho \mathbf{u}^2 + \frac{1}{8\pi}\mathbf{B}^2$$

Total pressure $$P_{tot} = P + \frac{1}{8\pi}\mathbf{B}^2$$

No magnetic monopoles $$\nabla \cdot \mathbf{B} = 0$$

MHD waves

Compute the Jacobian matrix $\mathbf{J} = \dfrac{\partial \mathbf{F}}{\partial \mathbf{U}}$

It has 7 real eigenvalues (ideal MHD equations are hyperbolic), one for each wave:

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

2 Alfven waves: $\lambda_2 = u - c_a$ $\lambda_6 = u + c_a$

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

1 entropy waves: $\lambda_4 = u$

Fast magnetosonic waves are longitudinal waves with variations in pressure and density (correlated with magnetic field)

Slow magnetosonic waves are longitudinal waves with variations in pressure and density (anti-correlated with magnetic field)

Alfven waves are transverse waves with no variation in pressure and density.

Entropy wave is a contact discontinuity with no variation in pressure and velocity.

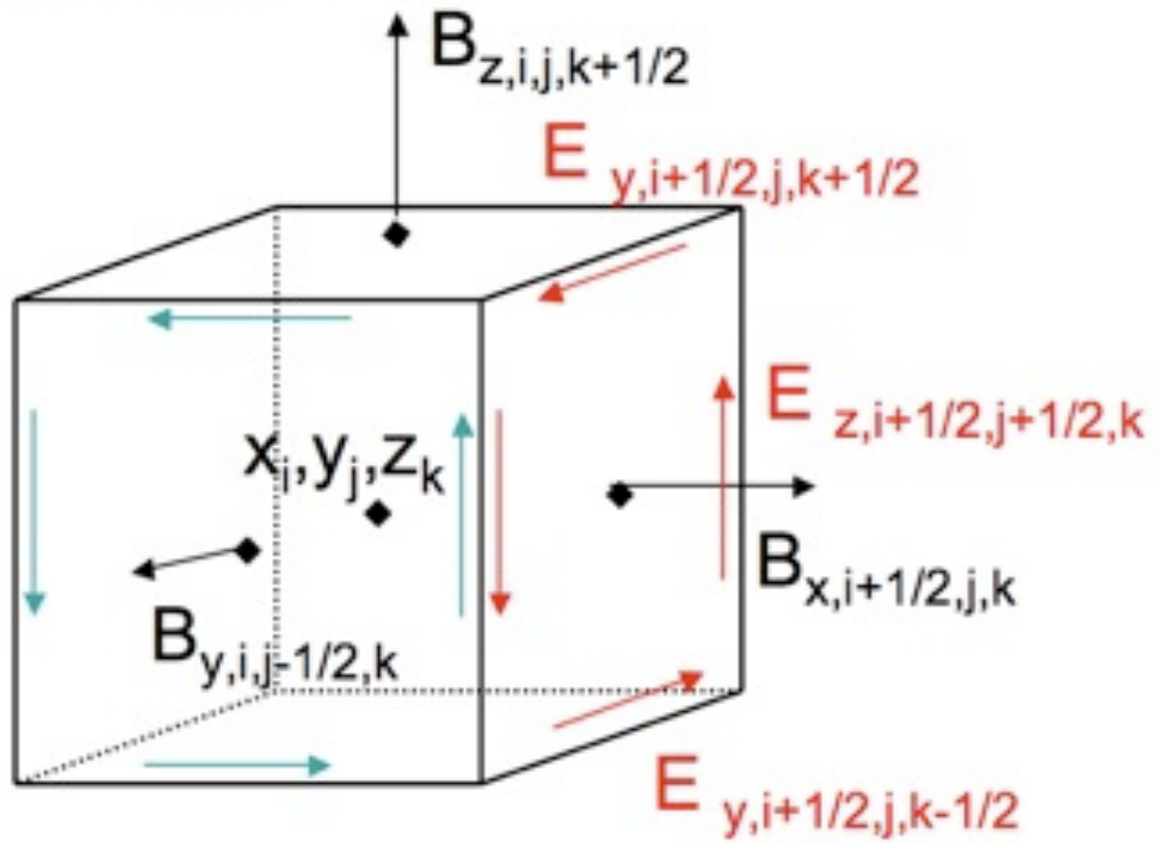
Godunov method with Constrained Transport

The induction equation in integral form suggests a surface-average form:

$$\partial_t \mathbf{B} + \nabla \times (\mathbf{B} \times \mathbf{u}) = 0 \qquad \text{(Stokes theorem)} \quad \partial_t \int_S \mathbf{B} \cdot \mathrm{d}\mathbf{s} + \int_L (\mathbf{B} \times \mathbf{u}) \cdot \mathrm{d}\mathbf{l} = 0$$

The magnetic field is face-centered while Euler-type variables are cell-centered (staggered mesh approach).

$$(B_x)_{i+1/2,jk} = \frac{1}{S} \int_S B_x(y,z)\mathrm{d}y\mathrm{d}z \qquad S = [y_{i-1/2}, y_{i+1/2}] \times [z_{i-1/2}, z_{i+1/2}]$$



Similar to potential vector methods (Yee 1966; Dorfi 1986; Evans & Hawley 1988).

CT: exact div B preserving scheme

Surface-averaged magnetic fields are updated conservatively:

$$B^{n+1}_{z,i,j,k-1/2} = B^{n}_{x,i,j,k-1/2} + \frac{\Delta t}{\Delta x}\left(E^{n+1/2}_{y,i+1/2,j,k-1/2} - E^{n+1/2}_{y,i-1/2,j,k-1/2}\right) - \frac{\Delta t}{\Delta y}\left(E^{n+1/2}_{x,i,j+1.2,k-1/2} - E^{n+1/2}_{x,i,j-1/2,k-1/2}\right)$$

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

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

using time-averaged electric fields defined at cell edge center:

$$E^{n+1/2}_{x,i,j-1/2,k-1/2} = \frac{1}{\Delta t \Delta x}\int_{t^n}^{t^{n+1}}\int_{x_{i-1/2}}^{x_{i+1/2}} E_x(x, y_{j-1/2}, z_{k-1/2})\mathrm{d}t\mathrm{d}x$$

$$E^{n+1/2}_{y,i-1/2,j,k-1/2} = \frac{1}{\Delta t \Delta y}\int_{t^n}^{t^{n+1}}\int_{y_{j-1/2}}^{y_{j+1/2}} E_y(x_{i-1/2}, y, z_{k-1/2})\mathrm{d}t\mathrm{d}y$$

$$E^{n+1/2}_{z,i-1/2,j-1/2,k} = \frac{1}{\Delta t \Delta z}\int_{t^n}^{t^{n+1}}\int_{z_{k-1/2}}^{z_{k+1/2}} E_z(x_{i-1/2}, y_{j-1/2}, z)\mathrm{d}t\mathrm{d}z$$

The total flux (div B) across each cell bounding surface vanishes exactly !

The induction equation in 2D

We write Faraday's law $\partial_t \mathbf{B} = \nabla \times \mathbf{E}$ using now the EMF vector $\mathbf{E} = \mathbf{u} \times \mathbf{B}$

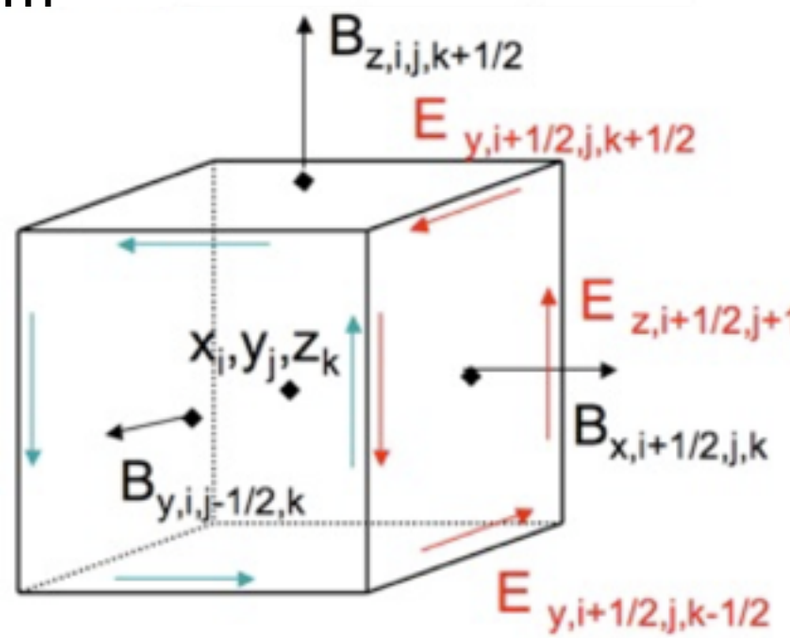
We use a finite-surface approximation for the magnetic field

$$B^n_{x,i+1/2,j} = \frac{1}{\Delta y}\int_{y_{j-1/2}}^{y_{j+1/2}} B_x(x_{i+1/2}, y)\mathrm{d}y \qquad B^n_{y,i,j+1/2} = \frac{1}{\Delta x}\int_{x_{i-1/2}}^{x_{i+1/2}} B_y(x, y_{j+1/2})\mathrm{d}x$$

Integral form of the induction equation using Stoke's theorem

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

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



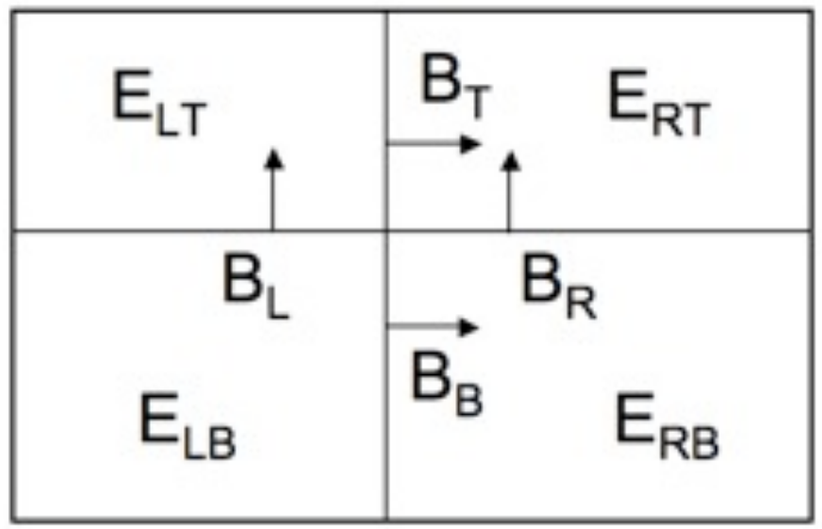
By construction, div B vanishes exactly:

$$\frac{B^{n+1}_{x,i+1/2,j,k} - B^{n+1}_{x,i-1/2,j,k}}{\Delta x} + \frac{B^{n+1}_{y,i,j+1/2,k} - B^{n+1}_{y,i,j-1/2,k}}{\Delta y} + \frac{B^{n+1}_{z,i,j,k+1/2} - B^{n+1}_{z,i,j,k-1/2}}{\Delta z}$$

$$\text{if } \nabla \cdot \mathbf{B}^n = 0$$

2D Riemann solvers for MHD

Londrillo & Del Zana 2004, Gardiner & Stone 2005, Teyssier et al. 2006; Fromang et al. 2006



1- Linear 2D Riemann solvers:

For a 1D Riemann solver (e.g. Roe):

$$F(U(0)) = \frac{1}{2}(F_L + F_R) + \frac{1}{2}\sum_{i=1,m}\left|\tilde{\lambda}_i\right|(\tilde{\beta}_i - \tilde{\alpha}_i)\tilde{K}^i$$

The 2D flux is given by:

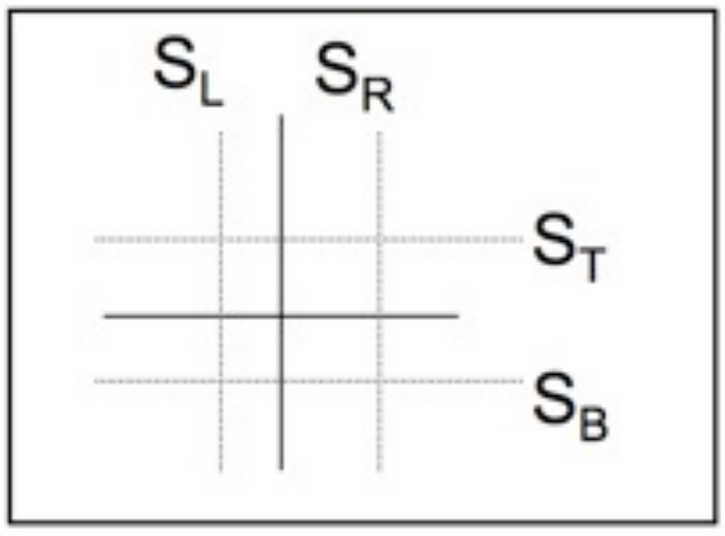
$$F(U(0)) = \frac{1}{4}(F_{LL} + F_{RL} + F_{LR} + F_{RR}) +$$

$$\frac{1}{2}\sum_{i=1,m}\left|\tilde{\tilde{\lambda}}_{x,i}\right|(\tilde{\tilde{\beta}}_{x,i} - \tilde{\tilde{\alpha}}_{x,i})\tilde{\tilde{K}}^i_x - \frac{1}{2}\sum_{i=1,m}\left|\tilde{\tilde{\lambda}}_{y,i}\right|(\tilde{\tilde{\beta}}_{y,i} - \tilde{\tilde{\alpha}}_{y,i})\tilde{\tilde{K}}^i_y$$

2- The HLL solver in 2D:

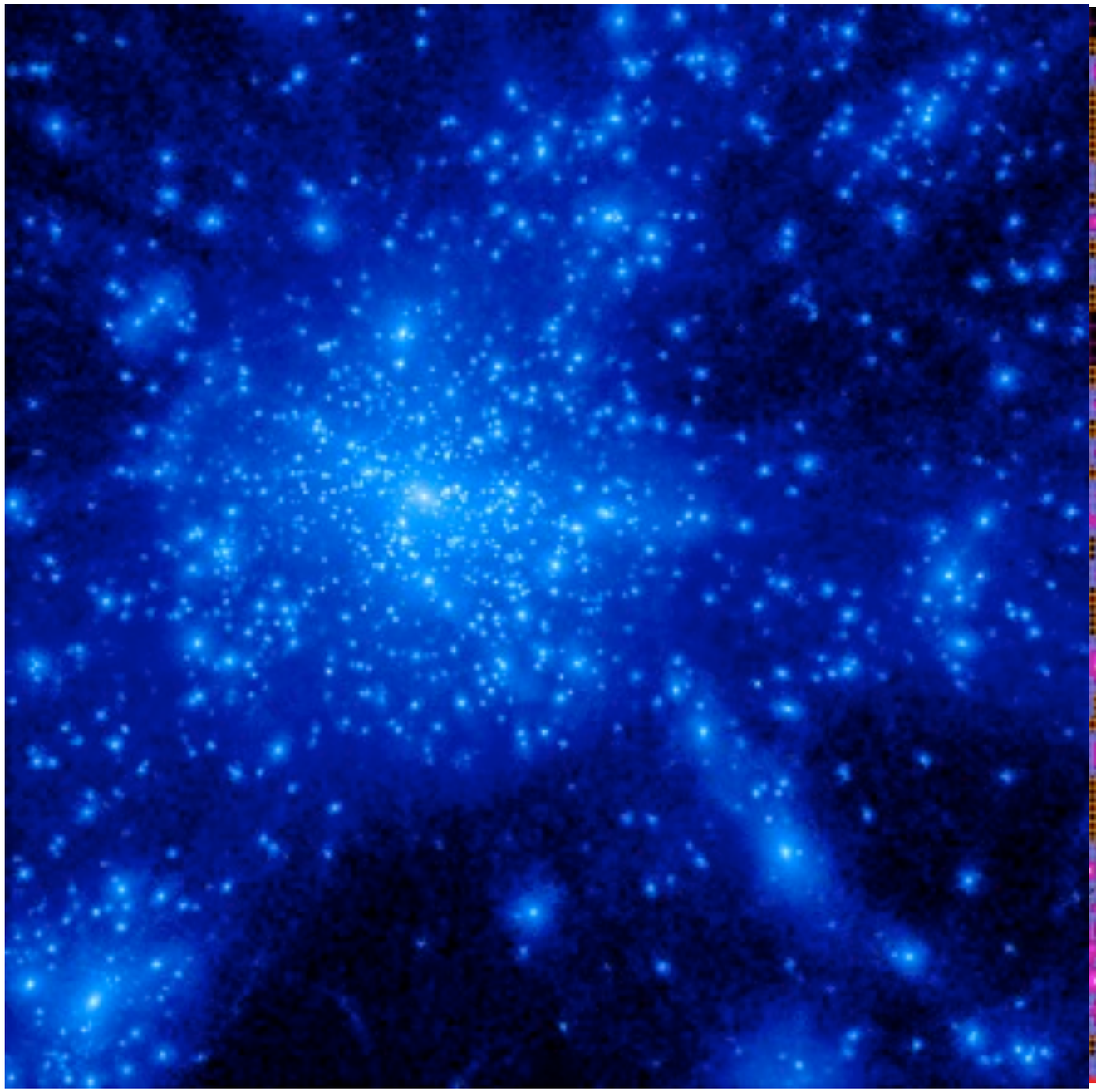
$$E^* = \frac{S_R S_T E_{LB} + S_L S_B E_{RT} - S_L S_T E_{RB} - S_R S_B E_{LT}}{(S_R - S_L)(S_T - S_B)}$$

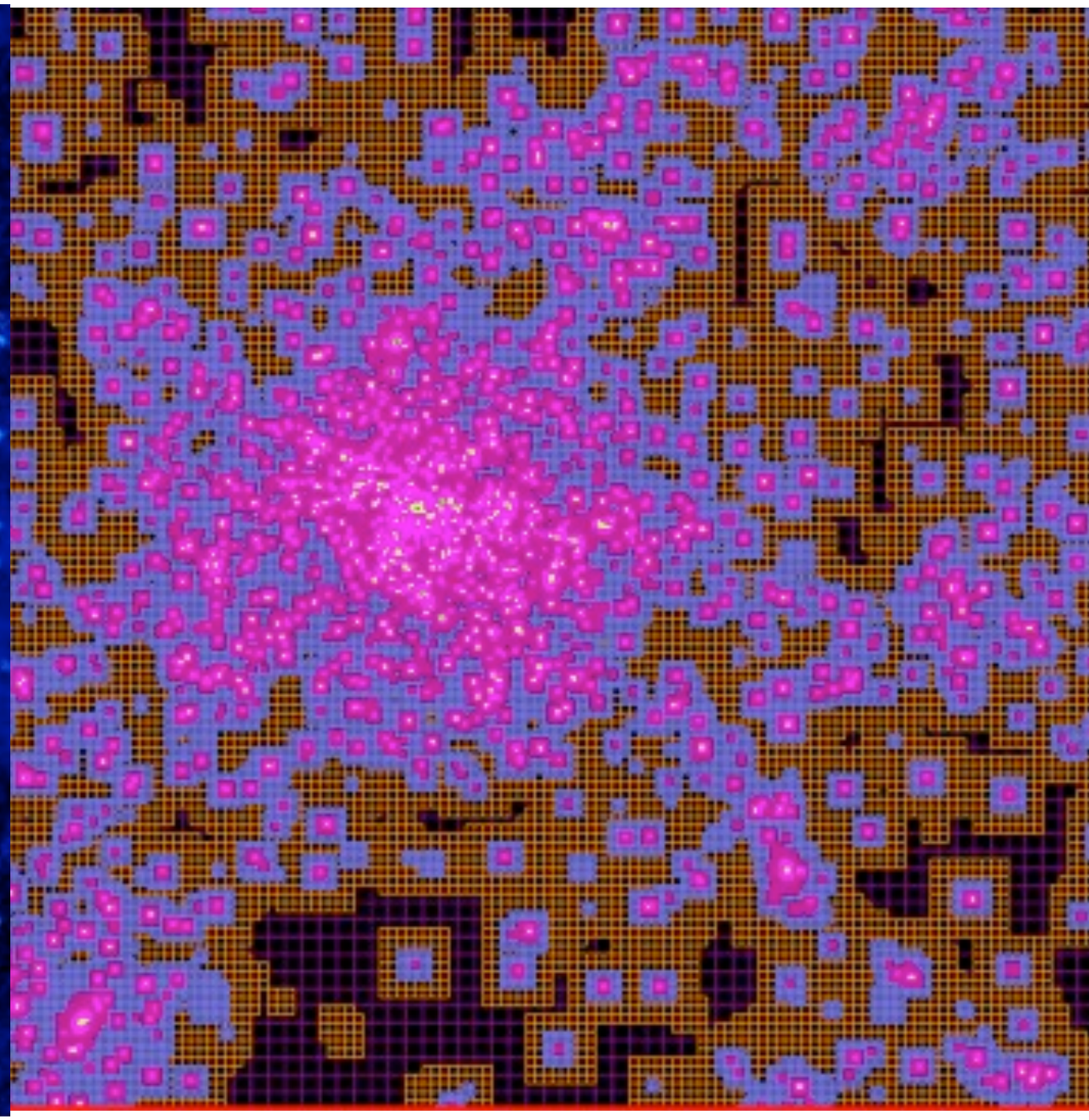
$$-\frac{S_B S_T}{S_T - S_B}(B_R - B_L) - \frac{S_L S_R}{S_R - S_L}(B_T - B_B)$$



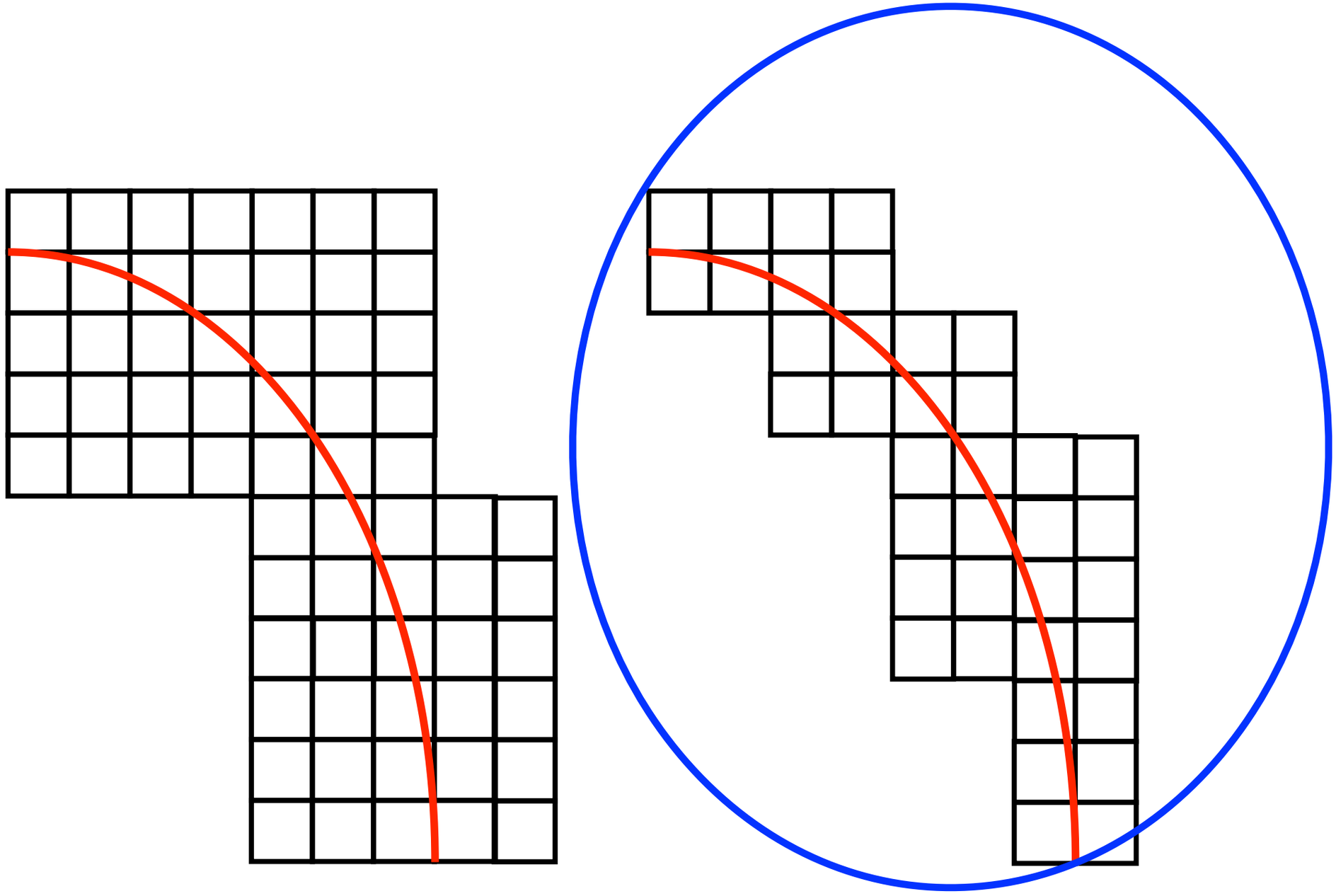
3- 2D version of HLLD in RAMSES.

Adaptive Mesh Refinement





Patch-based versus tree-based



Graded Octree structure

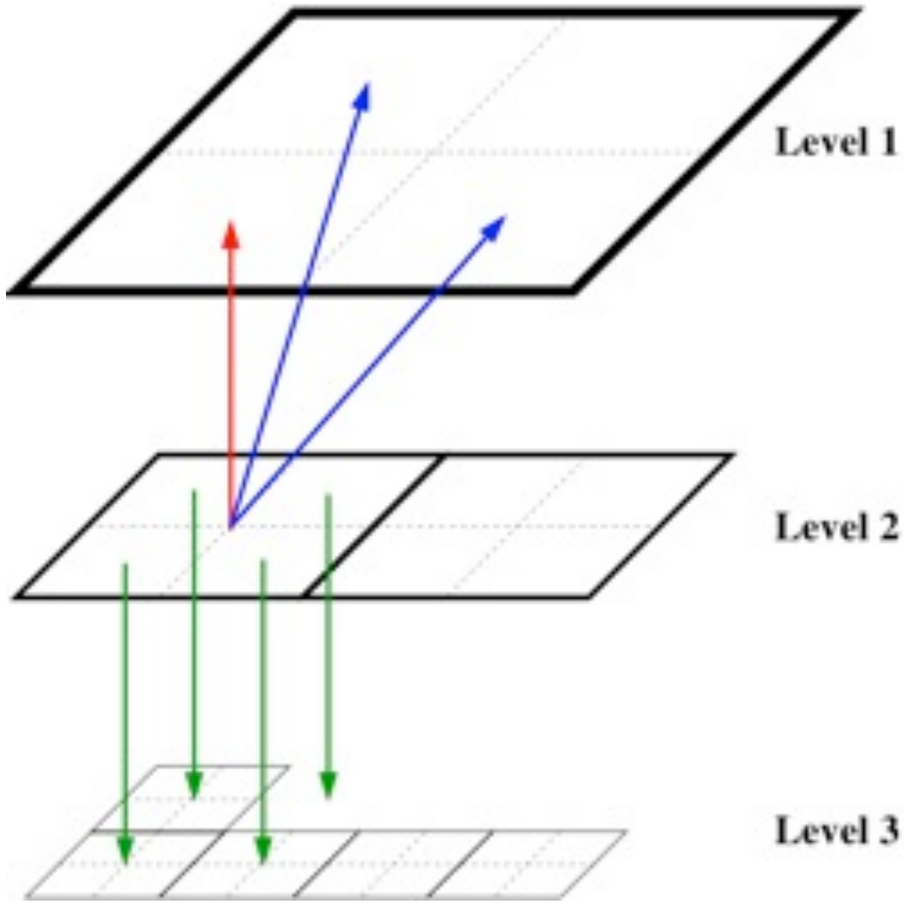
Fully Threaded Tree (Khokhlov 98).

Cartesian mesh refined on a *cell by cell basis.*

`octs`: small grid of 8 cells

Pointers (arrays of index)

- 1 parent cell
- 6 neighboring parent cells
- 8 children octs
- 2 linked list indices



Cell-centered variables are updated level by level using linked lists.

Cost = 2 integer per cell.

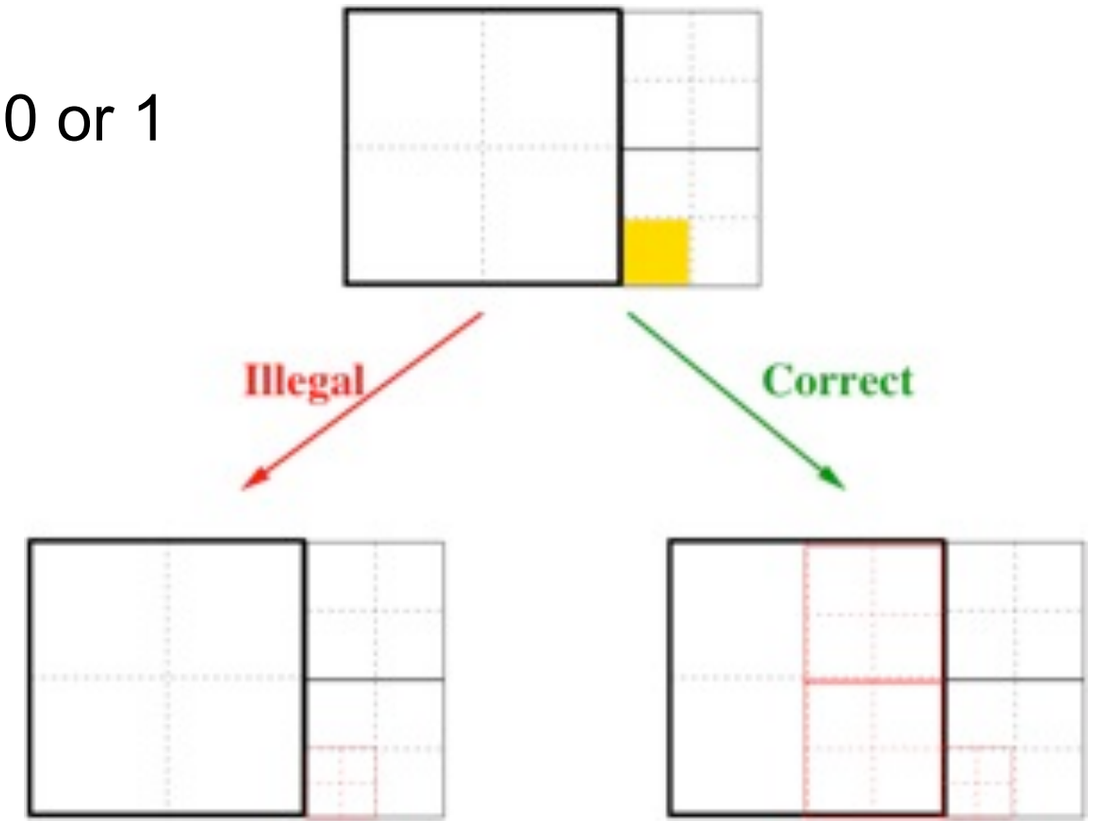
Optimize mesh adaptation to complex flow geometries, but CPU overhead compared to unigrid can be as large as 50%.

2 type of cell: - “leaf” or active cell

- “split” or inactive cell

Refinement rules for graded octree

Compute the refinement map: flag = 0 or 1



Step 1: mesh consistency

if a split cell contains at least one split or marked cell, then mark the cell with flag = 1 and mark its 26 neighbors

Step 2: physical criteria

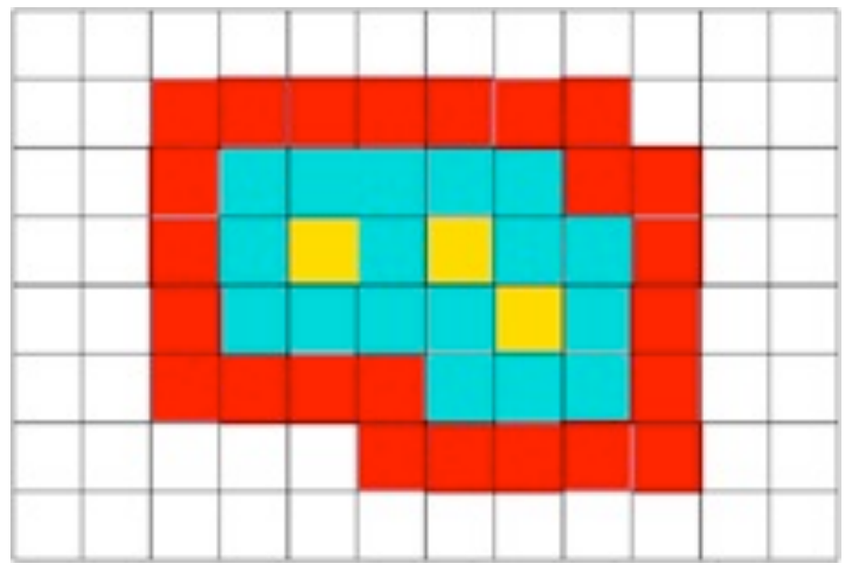
quasi-Lagrangian evolution, Jeans mass

geometrical constraints (zoom)

Truncation errors, density gradients…

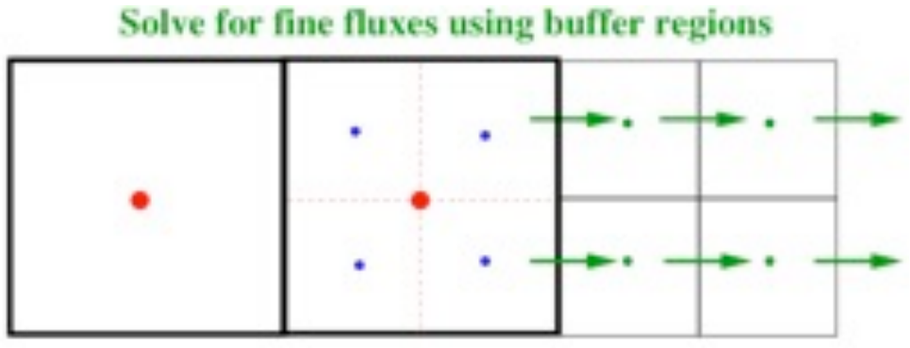
Step 3: mesh smoothing

apply a dilatation operator (mathematical morphology) to regions marked for refinement → convex hull



Godunov schemes and AMR

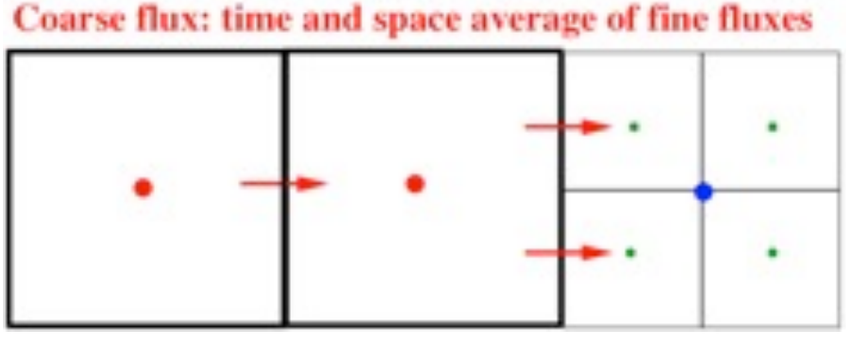
Berger & Oliger (84), Berger & Collela (89)



Prolongation (interpolation) to finer levels

- fill buffer cells (boundary conditions)

- create new cells (refinements)



Restriction (averaging) to coarser levels

- destroy old cells (de-refinements)

Flux correction at level boundary

$$(\mathbf{F}^{n+1/2,\ell}_{i+1/2,j}) = \frac{(\mathbf{F}^{n+1/2,\ell+1}_{i+1/2,j-1/4}) + (\mathbf{F}^{n+1/2,\ell+1}_{i+1/2,j+1/4})}{2}$$

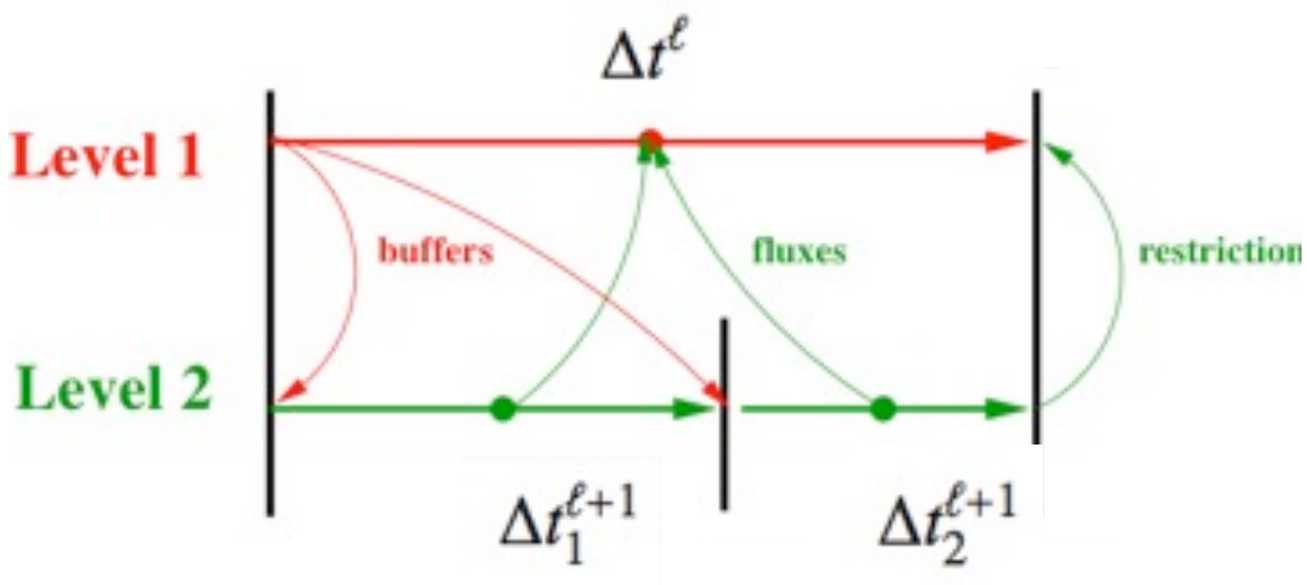
Careful choice of interpolation variables (conservative or not ?)

Several interpolation strategies (with $R^T P = I$) :

- straight injection

- tri-linear, tri-parabolic reconstruction

Adaptive Time Stepping



Time integration: single time step or recursive sub-cycling

- froze coarse level during fine level solves (one order of accuracy down !)
- average fluxes in time at coarse fine boundaries

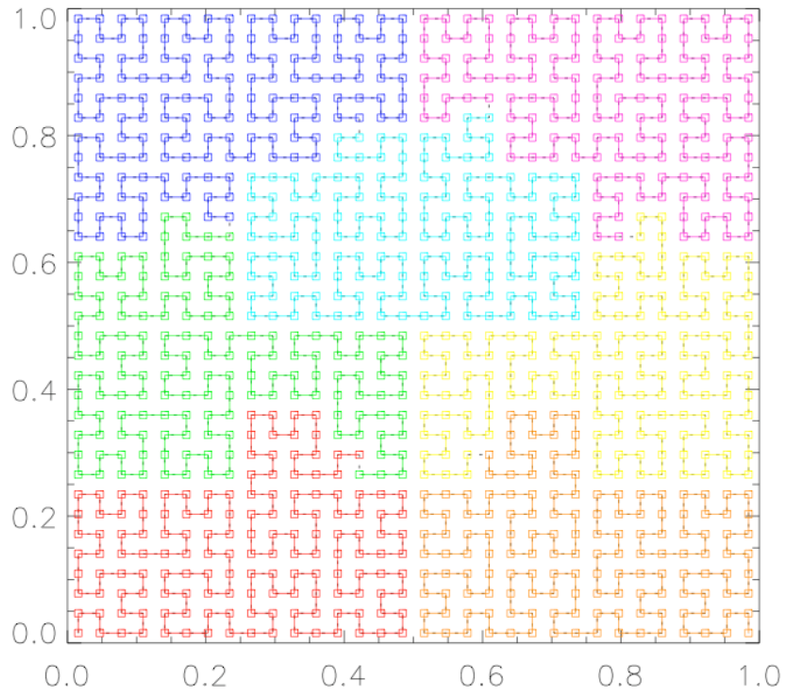
$$(\mathbf{F}_{i+1/2,j}^{n+1/2,\ell}) = \frac{1}{\Delta t_1^{\ell+1} + \Delta t_2^{\ell+1}} \left(\Delta t_1^{\ell+1} \frac{(\mathbf{F}_{i+1/2,j-1/4}^{n+1/4,\ell+1}) + (\mathbf{F}_{i+1/2,j+1/4}^{n+1/4,\ell+1})}{2} + \Delta t_2^{\ell+1} \frac{(\mathbf{F}_{i+1/2,j-1/4}^{n+3/4,\ell+1}) + (\mathbf{F}_{i+1/2,j+1/4}^{n+3/4,\ell+1})}{2} \right)$$

dapnia

cea

saclay

Domain decomposition for parallel computing



Parallel computing using the MPI library with a domain decomposition based on the *Peano-Hilbert curve*.

Algorithm inspired by gravity solvers (tree codes).

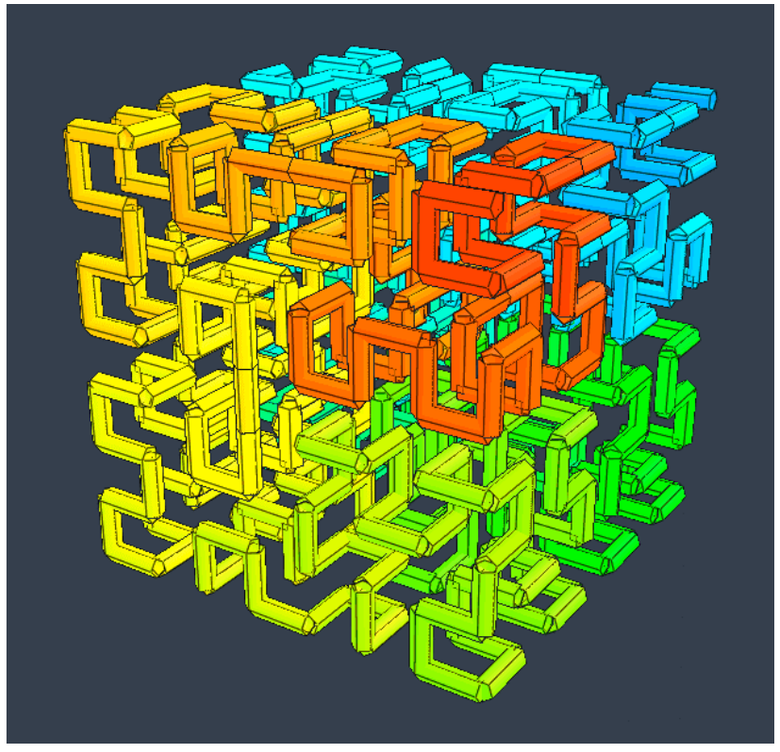
Use locally essential tree.

Tested and operational up to 6144 core.

Scaling depends on problem size and complexity.

Salmon, J.K. and Warren, M.S., "Parallel out-of-core methods for N-body simulation", In Eighth SIAM Conference on Parallel Processing for Scientific Computing, SIAM,1997.

Peter MacNeice, Kevin M. Olsonb, Clark Mobarryc, Rosalinda de Fainchteind and Charles Packer, « PARAMESH: A parallel adaptive mesh refinement community toolkit, », 2000, Computer Physics Communications, 126, 3.



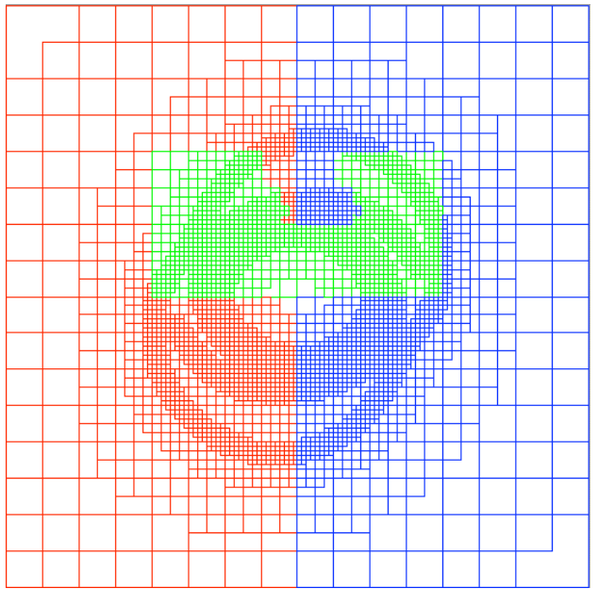
dapnia

cea

saclay

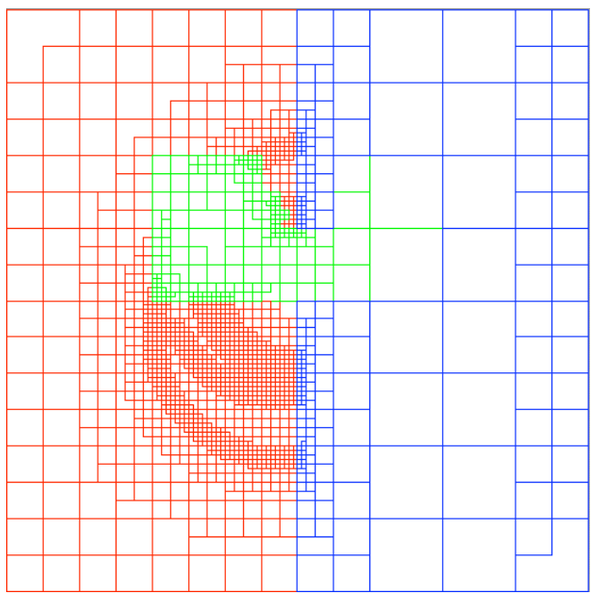
Locally essential trees

Salmon 90,
Warren 92,
Dubinsky 96

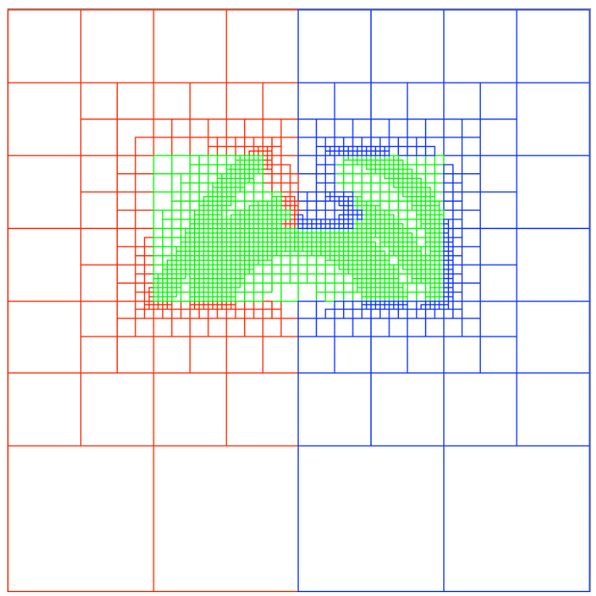


Domain decomposition over 3 processors

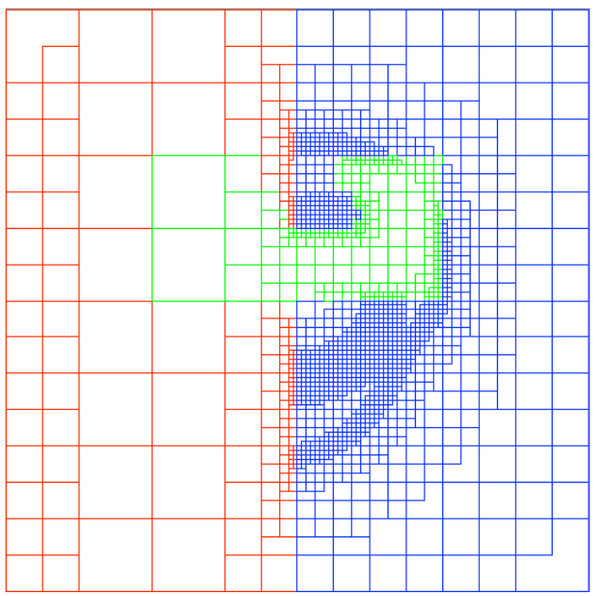
Each processor octree is surrounded by ghost cells (local copy of distant processor octrees) so that the resulting local octree contains all the necessary information.



Locally essential tree in processor #1



Locally essential tree in processor #2



Locally essential tree in processor #3

dapnia

cea

saclay

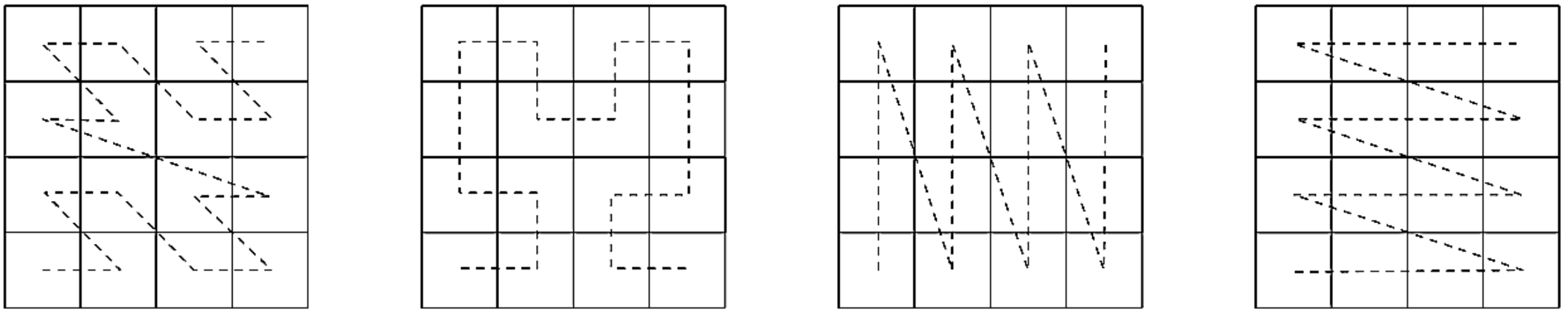
Dynamic partitioning in RAMSES

Several cell ordering methods:

1- column, row or plane major

2- Hilbert or Morton

3- User defined (angular, column+Hilbert…)



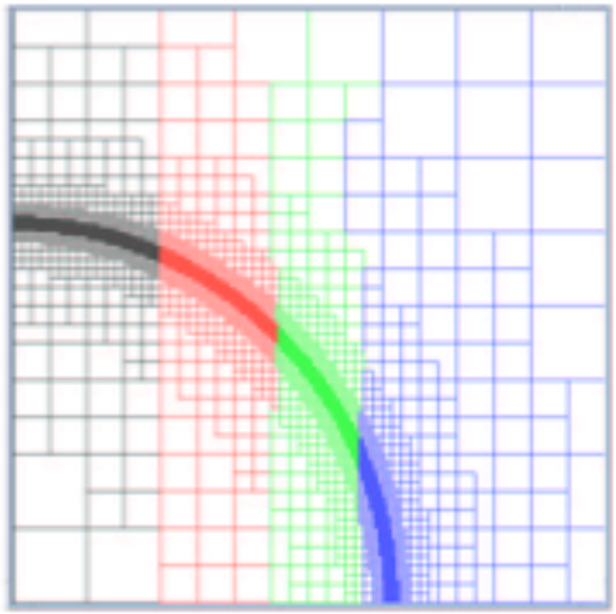
Dynamic partioning is performed every N steps by sorting each cell along chosen ordering and redistributing the mesh across processors. Usually, a good choice is N=10 (10% overhead).

dapnia

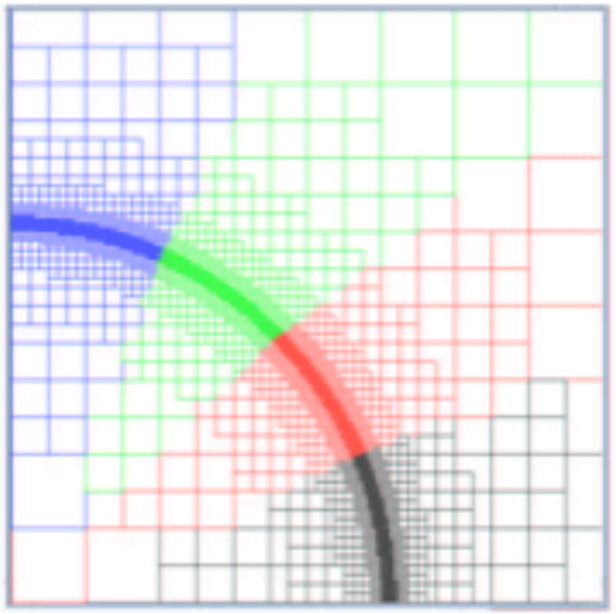
cea

saclay

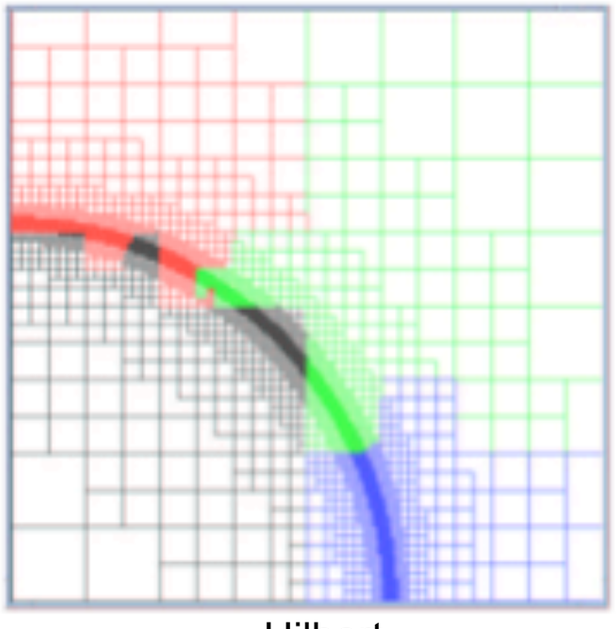
Are there any optimal load balancing strategy ?



Column major



Angular



Hilbert

RHD with Flux Limited Diffusion in RAMSES

✓ **RHD** solver in the *comoving* frame using the grey Flux Limited Diffusion approximation (*Commerçon et al. 2011a, 2014*)

$$
\begin{cases}
\partial_t \rho & + & \nabla\left[\rho \mathbf{u}\right] & = & 0 \\
\partial_t \rho \mathbf{u} & + & \nabla\left[\rho \mathbf{u} \otimes \mathbf{u} + (P + 1/3 E_\mathrm{r})\mathbb{I}\right] & = & -(\lambda - 1/3)\nabla E_\mathrm{r} \\
\partial_t E_\mathrm{T} & + & \nabla\left[\mathbf{u}\left(E_\mathrm{T} + P + 1/3 E_\mathrm{r}\right)\right] & = & -(\lambda - 1/3)\nabla(\mathbf{u} E_\mathrm{r}) + \nabla \cdot \left(\frac{c\lambda}{\rho \kappa_\mathrm{R}} \nabla E_\mathrm{r}\right) \\
\partial_t E_\mathrm{r} & + & \nabla\left[\mathbf{u} E_\mathrm{r}\right] & = & -\mathbb{P}_\mathrm{r} : \nabla \mathbf{u} + \kappa_\mathrm{P} \rho c (a_\mathrm{R} T^4 - E_\mathrm{r}) + \nabla \cdot \left(\frac{c\lambda}{\rho \kappa_\mathrm{R}} \nabla E_\mathrm{r}\right)
\end{cases}
$$

Riemann solver - explicit Corrective terms - explicit Coupling + Diffusion - implicit

$$\partial t \mathbb{U} + \nabla \mathbb{F}(\mathbb{U}) = \mathbb{S}(\mathbb{U})$$

$$
\mathbb{P} = \begin{bmatrix} \rho \\ \mathbf{u} \\ P \\ E_\mathrm{r} \end{bmatrix} \qquad \mathbb{U} = \begin{bmatrix} \rho \\ \rho \mathbf{u} \\ E_\mathrm{T} \\ E_\mathrm{r} \end{bmatrix}
$$

Primitive Conservative

$$
\Delta t \leq C_\mathrm{CFL} \frac{\Delta x}{u + \sqrt{\frac{\gamma P}{\rho} + \frac{4 E_\mathrm{r}}{9\rho}}}
$$

Godunov part

$$\partial t\mathbb{U} + \nabla\mathbb{F}(\mathbb{U}) = 0 \quad \rightarrow \quad \partial t\mathbb{P} + \mathbb{B}(\mathbb{P})\nabla\mathbb{P} = 0$$

- Jacobian matrix

$$\mathbb{B}(\mathbb{V}) = \begin{pmatrix} u & \rho & 0 & 0 \\ 0 & u & \frac{1}{\rho} & \frac{1}{3\rho} \\ 0 & \gamma P & u & 0 \\ 0 & \frac{4E_{\mathrm{r}}}{3} & 0 & u \end{pmatrix}$$

- 3 eigenvalues

$$\lambda_i = \begin{cases} u - \sqrt{\frac{\gamma P}{\rho} + \frac{4E_{\mathrm{r}}}{9\rho}} \\ u \\ u + \sqrt{\frac{\gamma P}{\rho} + \frac{4E_{\mathrm{r}}}{9\rho}} \end{cases}$$

✓ Largest fan of solution

Godunov part

$$\partial t\mathbb{U} + \nabla\mathbb{F}(\mathbb{U}) = 0 \quad \rightarrow \quad \partial t\mathbb{P} + \mathbb{B}(\mathbb{P})\nabla\mathbb{P} = 0$$

- Jacobian matrix

$$\mathbb{B}(\mathbb{V}) = \begin{pmatrix} u & \rho & 0 & 0 \\ 0 & u & \frac{1}{\rho} & \frac{1}{3\rho} \\ 0 & \gamma P & u & 0 \\ 0 & \frac{4E_{\mathrm{r}}}{3} & 0 & u \end{pmatrix}$$

- update the flux using

approximate Riemann solvers (LF, HLL, HLLD)

$$\mathbb{F}(\mathbb{U}) = \begin{bmatrix} \rho\mathbf{u} \\ \rho\mathbf{u}\otimes\mathbf{u} + (P + 1/3E_{\mathrm{r}})\mathbb{I} \\ \mathbf{u}\left(E_{\mathrm{T}} + P + 1/3E_{\mathrm{r}}\right) \\ \mathbf{u}E_{\mathrm{r}} \end{bmatrix}$$

Same can be done for any other non-thermal energy (e.g. cosmic rays)

Source terms

- Correct for too large radiative pressure in radiative force (momentum) and radiative pressure work (total energy)

+ update radiative energy

$$\mathbb{S}(\mathbb{U}) = \begin{pmatrix} 0 \\ -(\lambda - 1/3)\nabla E_{\mathrm{r}} \\ -(\lambda - 1/3)(\mathbf{u} \cdot \nabla E_{\mathrm{r}} + E_{\mathrm{r}} \nabla : \mathbf{u}) \\ \mathbb{P}_{\mathrm{r}} \nabla : \mathbf{u} \end{pmatrix}.$$

- Terms are estimated using finite differences, accounting for AMR grid effects
- Accounts for anisotropy in the radiative pressure tensor

Implicit update of diffusion/coupling

✓ Finite volume framework

$$\frac{\Delta E_{\mathrm{r}}}{\Delta t} V = F \times S$$

$$\begin{cases} \dfrac{C_v T^{n+1} - C_v T^n}{\Delta t} = & -\kappa_{\mathrm{P}}^n \rho^n \mathrm{c}(a_{\mathrm{R}}(T^{n+1})^4 - E_{\mathrm{r}}^{n+1}) \\ \dfrac{E_{\mathrm{r}}^{n+1} - E_{\mathrm{r}}^n}{\Delta t} - \nabla \dfrac{\mathrm{c}\lambda^n}{\kappa_{\mathrm{R}}^n \rho^n} \nabla E_{\mathrm{r}}^{n+1} = & +\kappa_{\mathrm{P}}^n \rho^n \mathrm{c}(a_{\mathrm{R}}(T^{n+1})^4 - E_{\mathrm{r}}^{n+1}), \end{cases}$$

✓ Implicit discretization

$$\begin{aligned} (E_{r,i}^{n+1} - E_{r,i}^n)V_i \quad - \quad & c\Delta t \left(\frac{\lambda}{\kappa_{\mathrm{R}}}\right)_{i+1/2} S_{i+1/2} \frac{E_{r,i+1}^{n+1} - E_{r,i}^{n+1}}{\Delta x_{i+1/2}} \\ + \quad & c\Delta t \left(\frac{\lambda}{\kappa_{\mathrm{R}}}\right)_{i-1/2} S_{i-1/2} \frac{E_{r,i}^{n+1} - E_{r,i-1}^{n+1}}{\Delta x_{i-1/2}} \\ = \quad & c\Delta t \kappa_{\mathrm{P},i}^n \left(4a_{\mathrm{R}}(T_i^n)^3 T_i^{n+1} - 3a_{\mathrm{R}}(T_i^n)^4 - E_{r,i}^{n+1}\right) V_i \end{aligned}$$

✓ Linearize emission term

$$(T^{n+1})^4 = (T^n)^4 \left(1 + \frac{(T^{n+1} - T^n)}{T^n}\right)^4 \approx 4(T^n)^3 T^{n+1} - 3(T^n)^4$$

✓ Matrix system to invert which symmetric and positive definite

Implicit update of diffusion/coupling

✓ **Implicit solved with an iterative conjugate gradient algorithm**

$$-C_{i-1/2}E^{n+1}_{\mathrm{r},i-1} + (1 + C_{i-1/2} + C_{i+1/2})E^{n+1}_{\mathrm{r},i} - C_{i+1/2}E^{n+1}_{\mathrm{r},i+1} = f(E^{n}_{\mathrm{r},i}, T^{n+1}_{i})$$

- ✓ matrix elements are stored during iterations
- ✓ diagonal preconditionning
- ✓ scales in N log(N)

✓ Update gas temperature

$$T_i^{n+1} = \frac{3a_{\mathrm{R}}\kappa^n_{\mathrm{P},i}c\Delta t\left(T_i^n\right)^4 + C_v T_i^n + \kappa^n_{\mathrm{P},i}c\Delta t E^{n+1}_{r,i}}{C_v + 4a_{\mathrm{R}}\kappa^n_{\mathrm{P},i}c\Delta t\left(T_i^n\right)^3}$$

Linearisation only work if temperature changes are small

Implicit integration of a diffusion equation

- **Simple heat equation:** $$\frac{\partial E_{\rm r}}{\partial t} = \nabla . K \nabla E_{\rm r}$$

✓ Implicit scheme is unconditionally stable

➡ How to speed-up implicit schemes on AMR grids?

✓ use a unique time step for all the levels and couple all the levels (*Commerçon et al. 2011*)

✓ each level evolves "independently" form the others: needs to specify boundary conditions at level interfaces (e.g., nested grids, *Tomida et al.)*

✓ use adaptive time stepping (*ORION, CASTRO*)

Implicit integration of a diffusion equation

- **Simple heat equation:** $$\frac{\partial E_{\mathrm{r}}}{\partial t} = \nabla . K \nabla E_{\mathrm{r}}$$

✓ Implicit scheme is unconditionally stable

➡ How to speed-up implicit schemes on AMR grids?

✓ use a unique time step for all the levels and couple all the levels (*Commerçon et al. 2011*)

✓ each level evolves "independently" form the others: needs to specify boundary conditions at level interfaces (e.g., nested grids, *Tomida et al.)*

✓ use **adaptive time stepping** (*ORION, CASTRO*)

Implicit integration of a diffusion equation

- **Simple heat equation:** $$\frac{\partial E_{\mathrm{r}}}{\partial t} = \nabla . K \nabla E_{\mathrm{r}}$$

✓ Implicit scheme is unconditionally stable

➡ How to speed-up implicit schemes on AMR grids?

✓ use a unique time step for all the levels and couple all the levels (*Commerçon et al. 2011*)

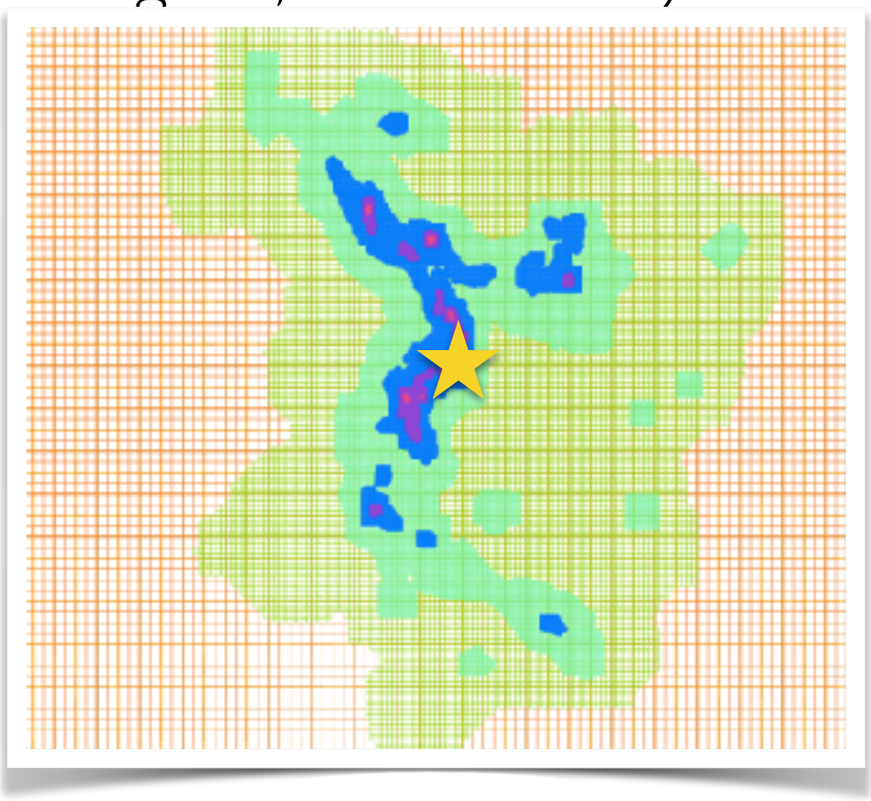
✓ each level evolves "independently" form the others: needs to specify boundary conditions at level interfaces (e.g., nested grids, *Tomida et al.)*

✓ use **adaptive time stepping** (*ORION, CASTRO*)

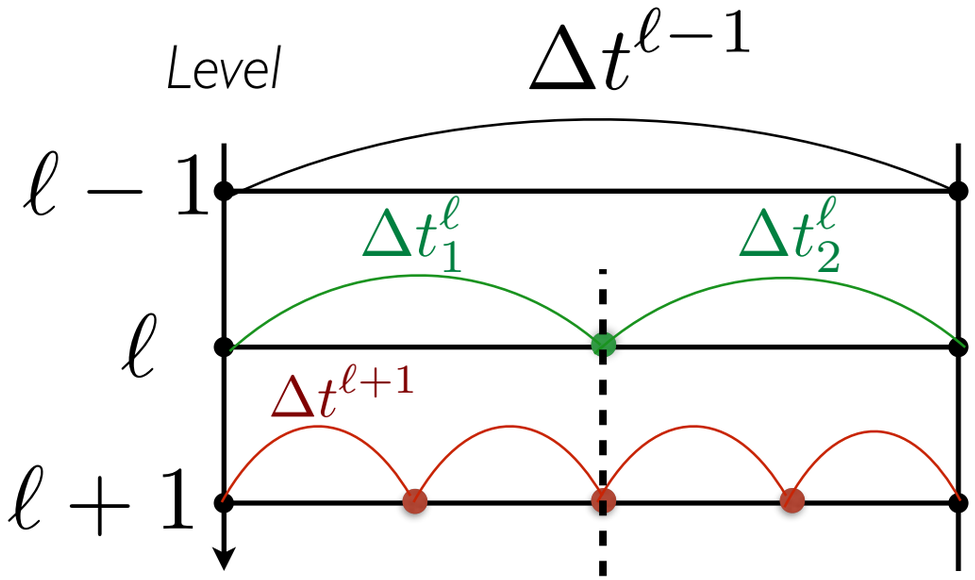
✓ Recurrent problems:

✓ energy propagation

✓ energy conservation



Adaptive time-stepping on AMR grid



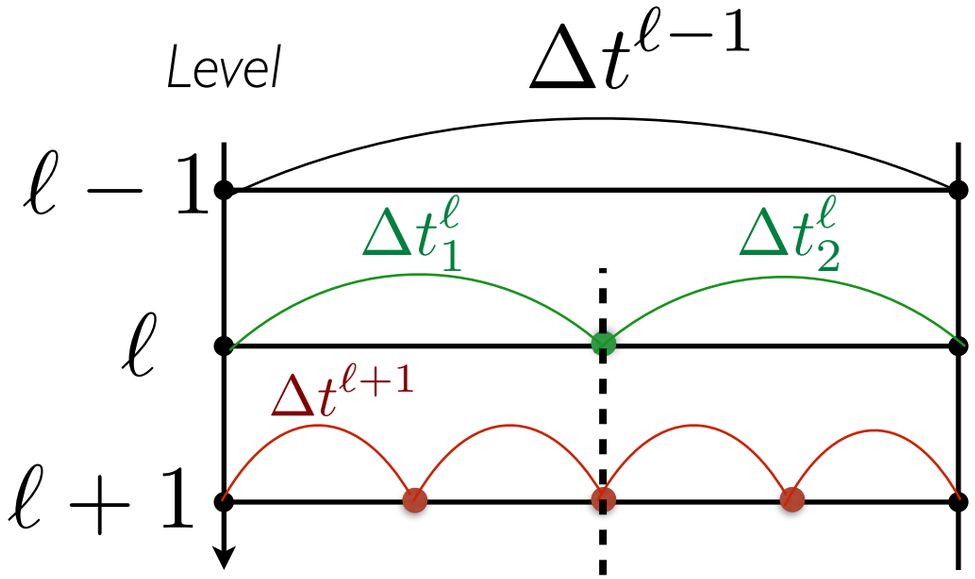
Straightforward for explicit scheme at coarse-to-fine interface:

$$F_{i+1/2}^{n+\Delta t^{\ell-1}} = \frac{1}{\Delta t_1^\ell + \Delta t_2^\ell}\left(\Delta t_1^\ell F_{i+1/2}^{n+\Delta t_1^\ell} + \Delta t_2^\ell F_{i+1/2}^{n+\Delta t_1^\ell+\Delta t_2^\ell}\right)$$

+ energy is conserved
+ highly efficient for hydrodynamics

Time-stepping

Adaptive time-stepping on AMR grid



Time-stepping

Straightforward for explicit scheme at coarse-to-fine interface:

$$F_{i+1/2}^{n+\Delta t^{\ell-1}} = \frac{1}{\Delta t_1^\ell + \Delta t_2^\ell}\left(\Delta t_1^\ell F_{i+1/2}^{n+\Delta t_1^\ell} + \Delta t_2^\ell F_{i+1/2}^{n+\Delta t_1^\ell+\Delta t_2^\ell}\right)$$

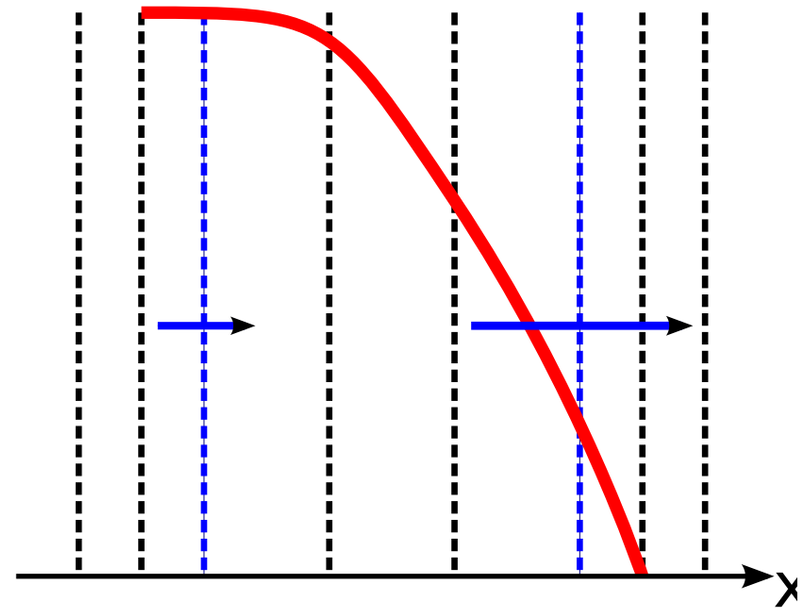
+ energy is conserved
+ highly efficient for hydrodynamics

but….

energy does not propagate more than one cell (CFL condition)

=> What happens for implicit schemes when flux are stored?

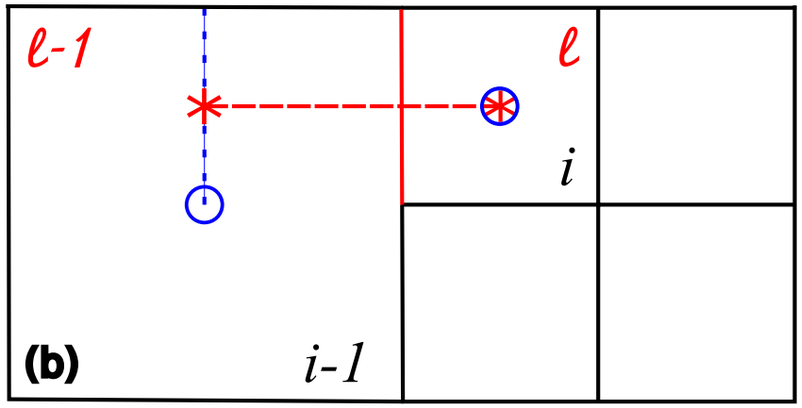
NEGATIVE ENERGY!!!!



Grid configuration and boundary conditions

- **Dirichlet:** imposed boundary value ($E_r = E_{r,b}$)
  - ➡ robust , but energy is not conserved
- **Neumann:** imposed flux condition ($\mathbf{F}_r = \mathbf{F}_{r,b}$)
  - ➡ energy is conserved (e.g. *Howell & Greenhough 2003*)
- **Robin:** mix between Dirichlet and Neumann, weighted by a parameter α

Commerçon et al. (2014)

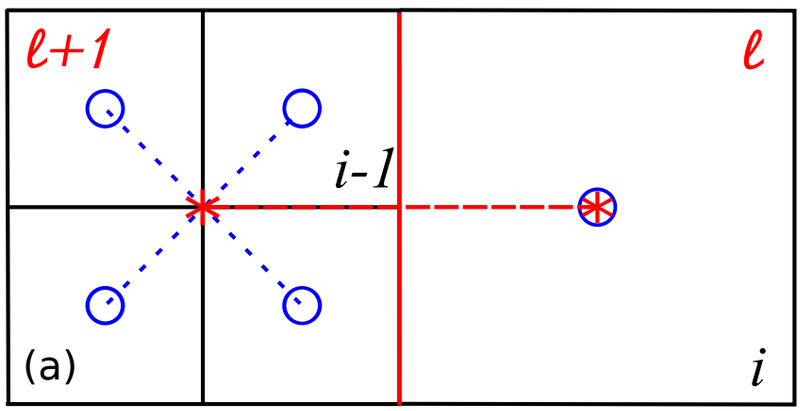


Neighbor is less refined

- **Fine-to-coarse interface:** Dirichlet BC

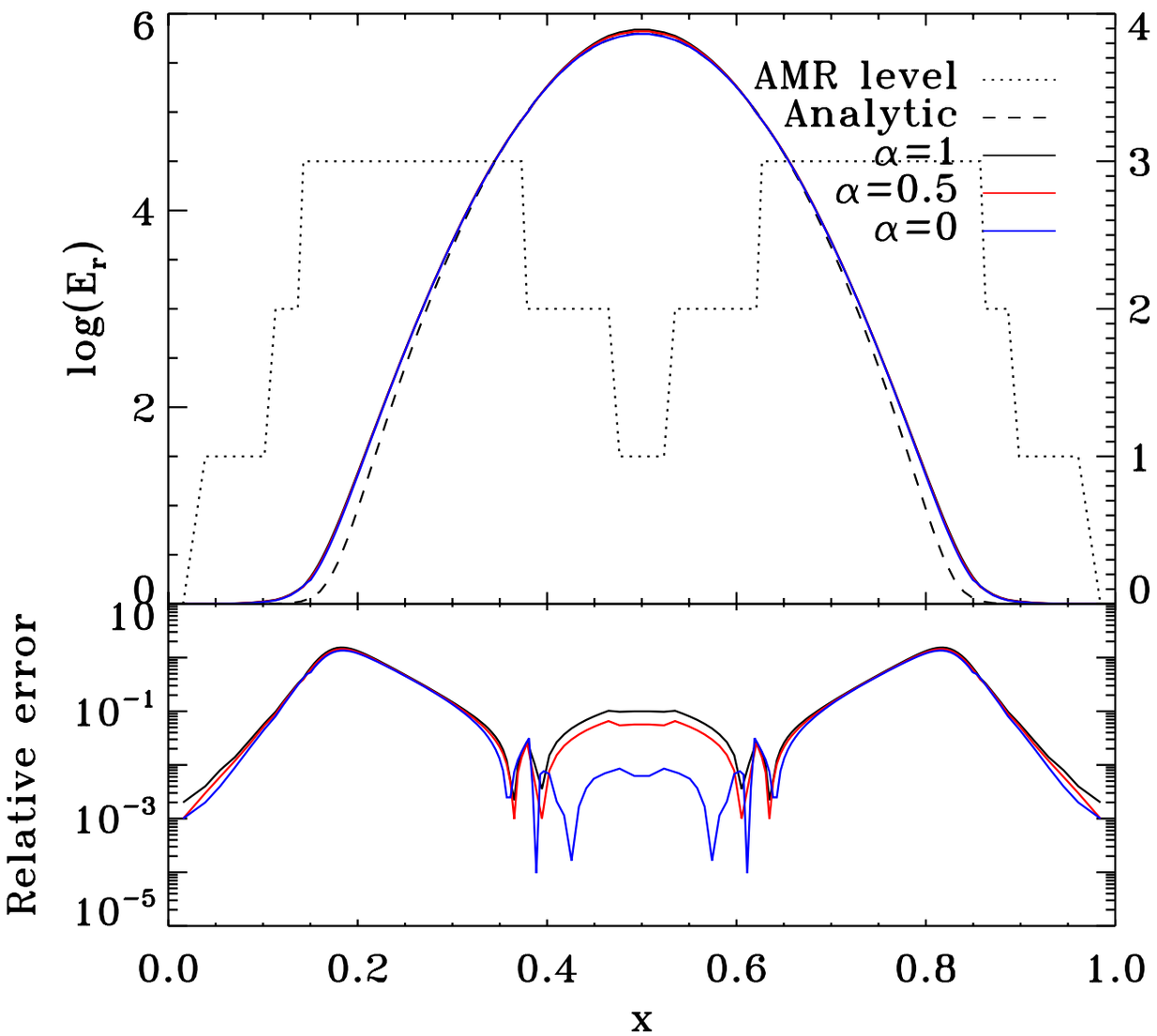
$$\tilde{F}_{i-1/2} = -K_{i-1/2}\frac{E_{\mathrm{r},i}^{n+1} - E_{\mathrm{r},i-1}^{n}}{\frac{3}{2}\Delta x}$$

- **Coarse-to-fine:** 3 possibilities
  …but energy mismatch in Dirichlet and Robin case



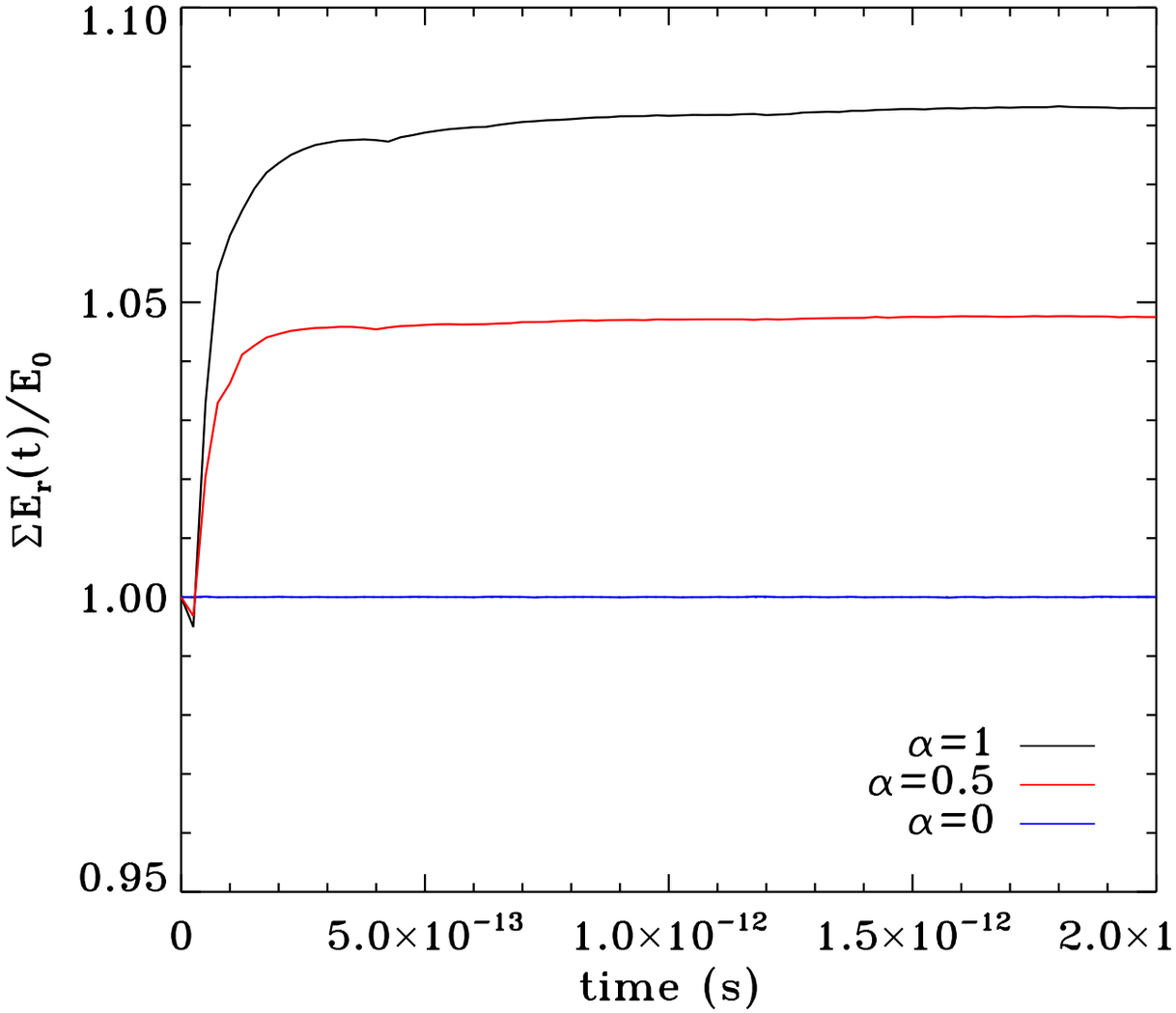
Neighbor is more refined

Test: 1D Dirac diffusion



Dirichlet: $\alpha = 1$

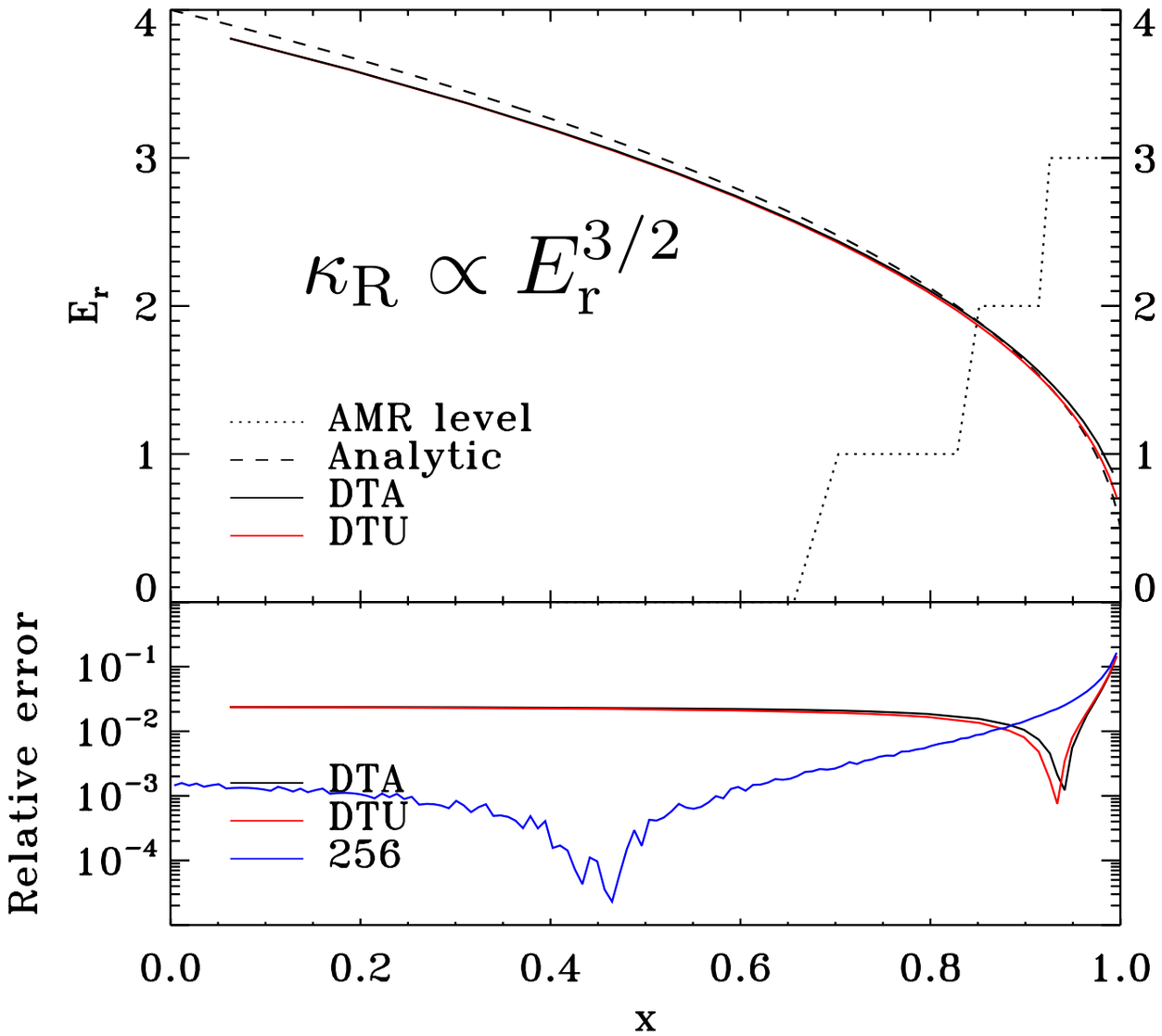
Commerçon et al. (2014)



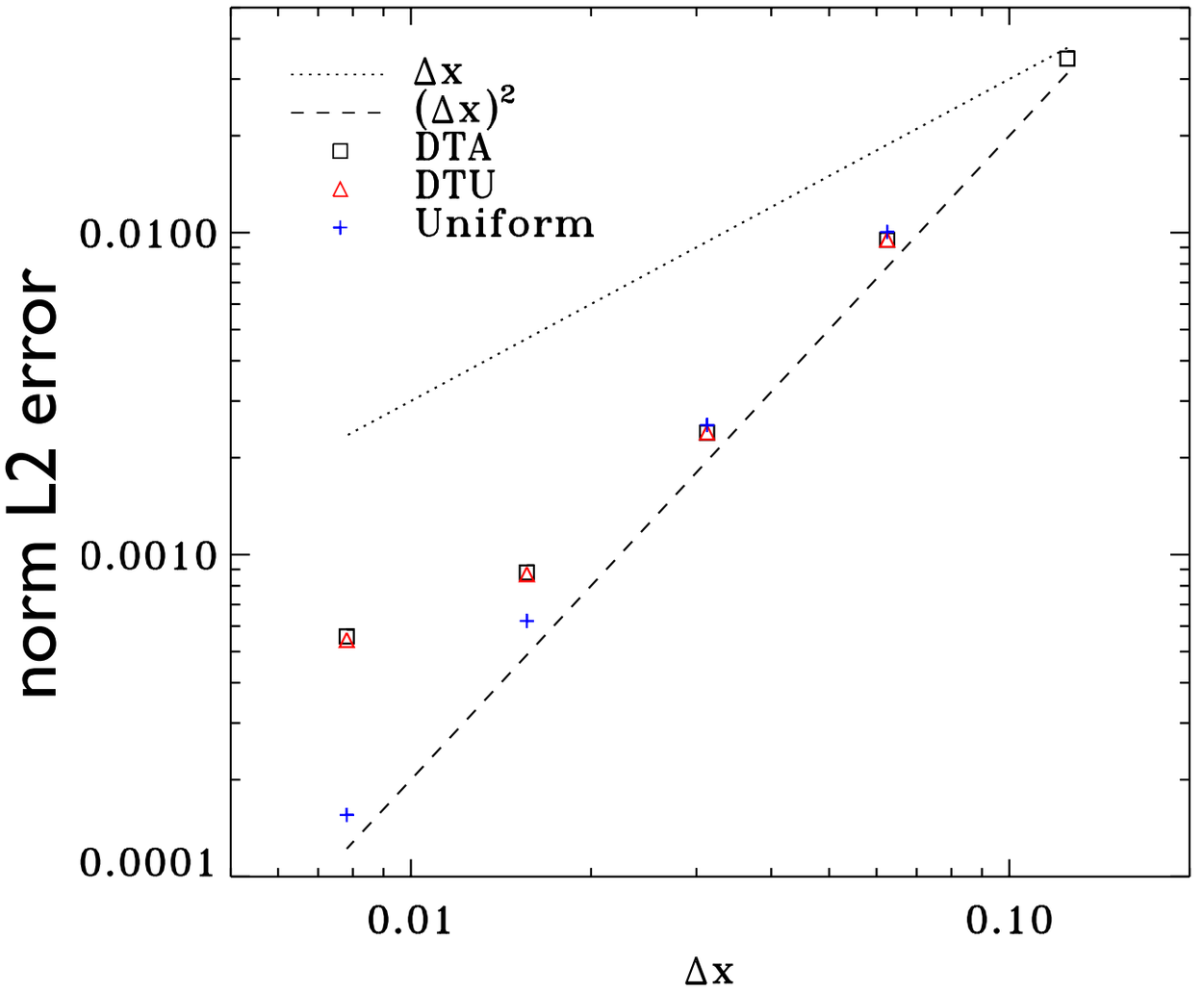
Energy conservation

good results with Dirichlet even if energy is not conserved

Test: stationary non-linear diffusion



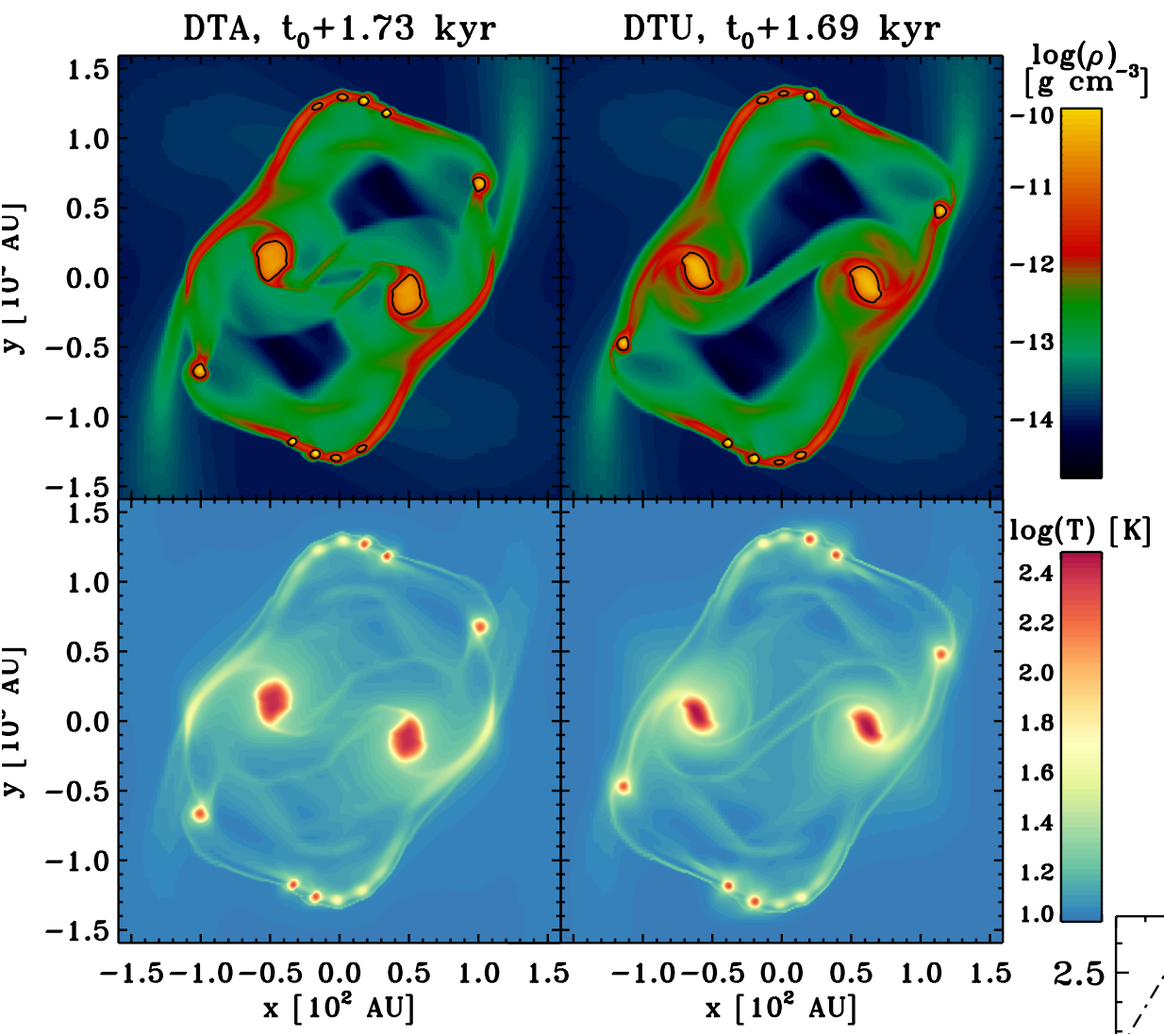
Commerçon et al. (2014)

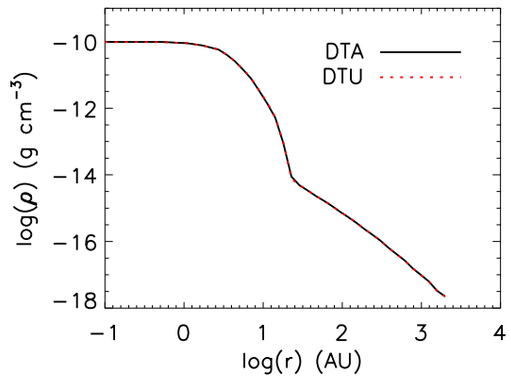


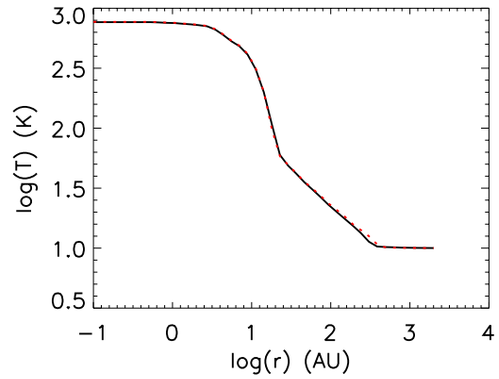
2nd order in space

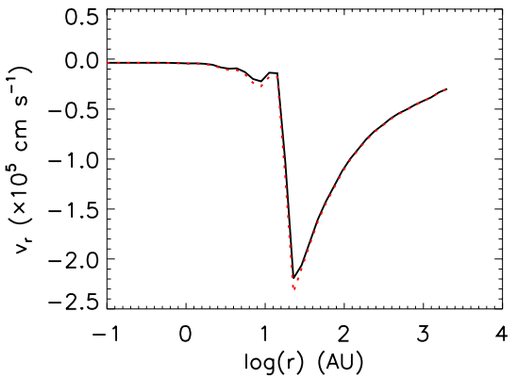
- similar results than using using a unique time step
- Neumann and Robin BC do not pass this test because of negative energy (initial gradient)…

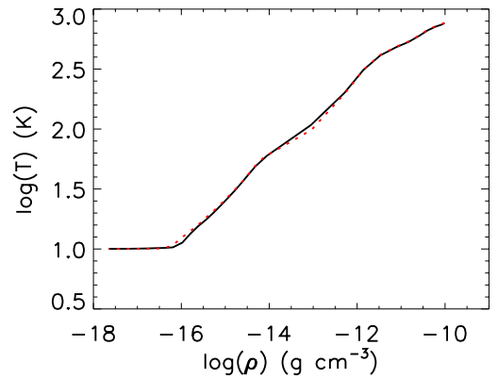
Test 3D: 1 solar mass dense core collapse





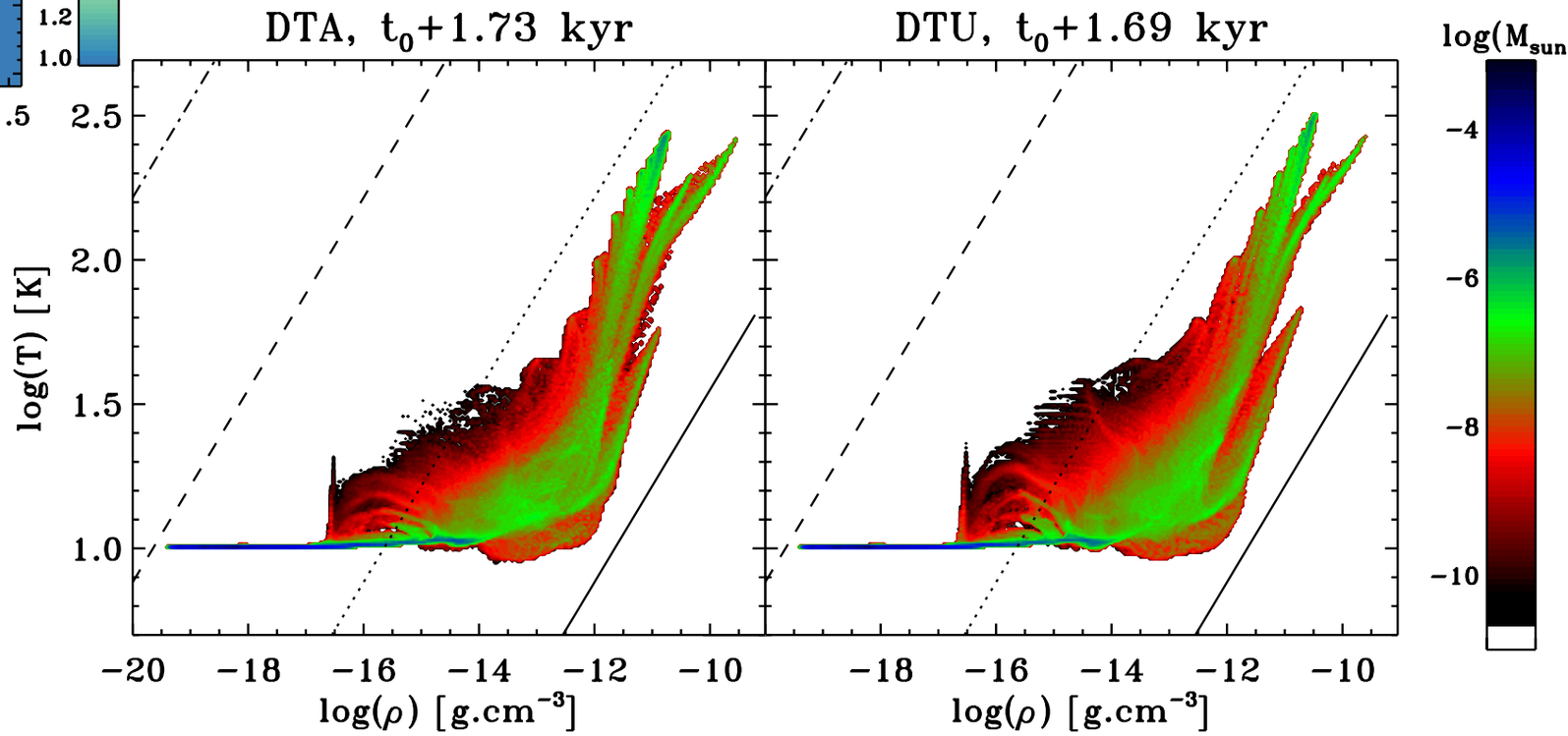






Gain in CPU time compared to DTU: factor 10

Commerçon et al. (2014)



What’s next?

Technologies

Supercomputers: TOP500

Rank	System	Cores	Rmax (TFlop/s)	Rpeak (TFlop/s)	Power (kW)
1	**Supercomputer Fugaku** - Supercomputer Fugaku, A64FX 48C 2.2GHz, Tofu interconnect D, **Fujitsu** RIKEN Center for Computational Science Japan	7,630,848	442,010.0	537,212.0	29,899
2	**Summit** - IBM Power System AC922, IBM POWER9 22C 3.07GHz, NVIDIA Volta GV100, Dual-rail Mellanox EDR Infiniband, **IBM** DOE/SC/Oak Ridge National Laboratory United States	2,414,592	148,600.0	200,794.9	10,096
3	**Sierra** - IBM Power System AC922, IBM POWER9 22C 3.1GHz, NVIDIA Volta GV100, Dual-rail Mellanox EDR Infiniband, **IBM / NVIDIA / Mellanox** DOE/NNSA/LLNL United States	1,572,480	94,640.0	125,712.0	7,438
4	**Sunway TaihuLight** - Sunway MPP, Sunway SW26010 260C 1.45GHz, Sunway, **NRCPC** National Supercomputing Center in Wuxi China	10,649,600	93,014.6	125,435.9	15,371
5	**Perlmutter** - HPE Cray EX235n, AMD EPYC 7763 64C 2.45GHz, NVIDIA A100 SXM4 40 GB, Slingshot-10, **HPE** DOE/SC/LBNL/NERSC United States	706,304	64,590.0	89,794.5	2,528

ARM

GPU

GPU

GPU

1 CPUh = 4g of CO_2 emission

Road to HPC exascale

Exascale = ExaFLOP=**10^{18}** floating operations/sec
Expected for 2023**!**

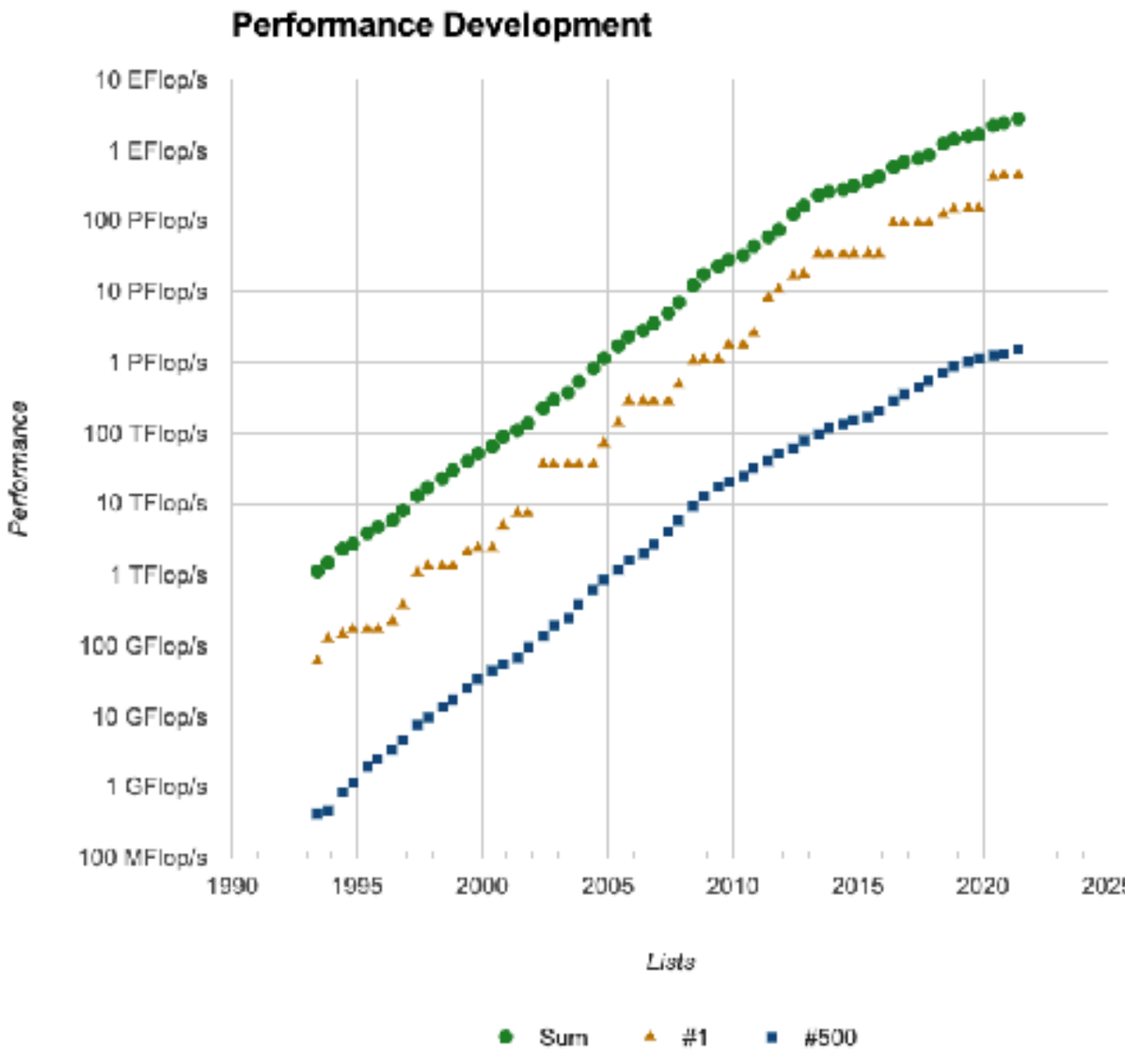
Challenge:
Increasing number of computing units: load balancing
Decreasing memory/thread
Huge stress on I/Os

New architectures: GPUs, ARM, FPGA

Hybrid architectures, but continuously evolving

New needs: large data analysis (HPDA), artificial intelligence (AI)

New jobs: co-design with specialists of informatics, numerical schemes, and computational science



Goal:
New tools built for **exascale AND hybrid architectures, a**ble to adapt to different technologies
Multi-scales, multi-physics, heterogeneous data (grid and particles)
Massive data analysis (e.g. Coda III = 20 Po!)
Prepare large observational campaign

Towards next generation codes

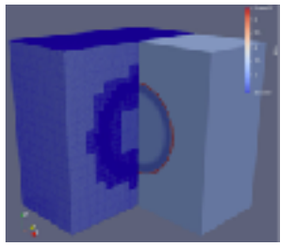
The GINEA group

(Groupe d'Investigation Numérique pour l'Exascale en Astrophysique)

Arnaud Durocher (CEA), Benoît Commerçon (CRAL), Bruno Thooris (CEA), Corentin Cadiou (UCL), Damien Chapon(CEA), **Dominique Aubert** (OAS), Yohan Dubois (IAP), Bournaud Frédéric (CEA), Lesur Geoffroy (IPAG), Jeremy Blaizot (CRAL), Jérémy Fensch (CRAL), Joki Rosdahl (CRAL), Leo Michel-Dansac (CRAL) , Strafella Loic (CEA), Maxime Trebitsch (KAI), Matthias González (AIM), Maxime Delorme (CEA), Olga Abramkina (IDRIS), Pascal Tremblin (CEA), Patrick Hennebelle (CEA) , Pierre Kestener (CEA), Ocvirk Pierre (OAS), Timothée David--Cléris (CRAL), Vincent Reverdy (ENS), Yann Rasera (LUTH)

DYABLO

(Durocher, Delorme)

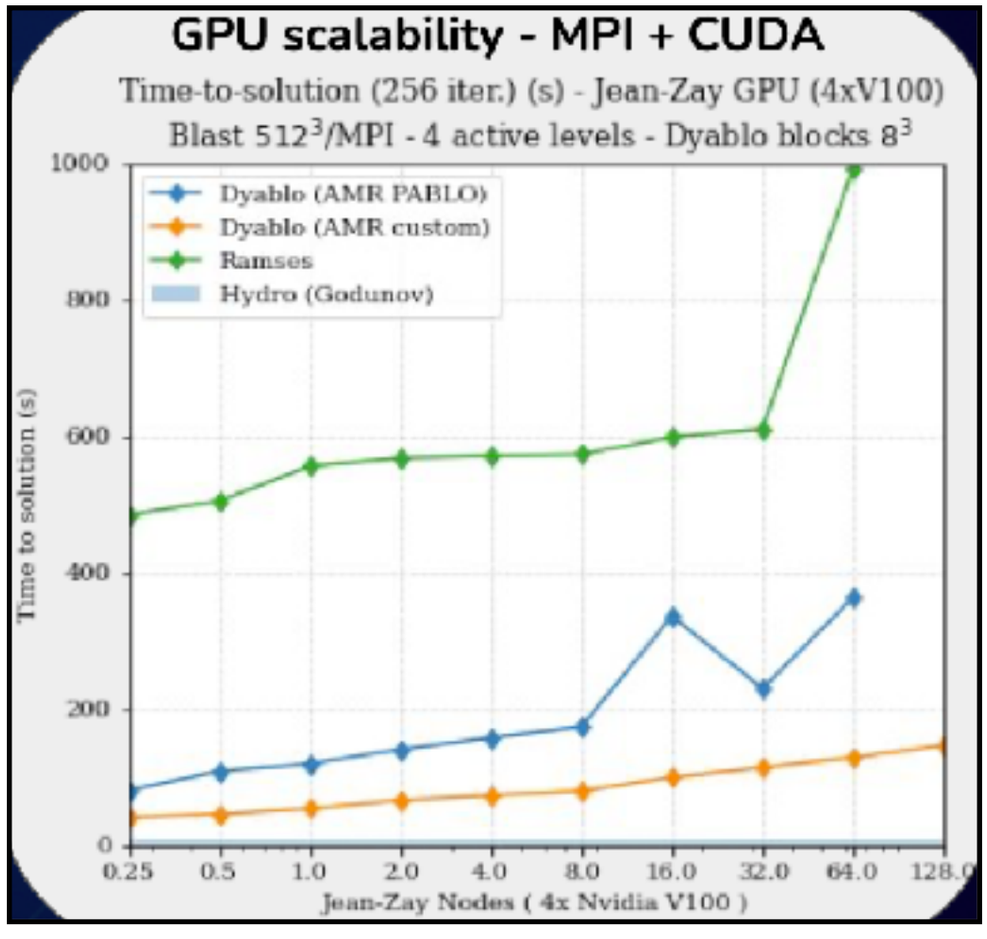


Modern and modular: C++ (>C++14), Applicative layer (plugins) to add physics modules
Hardware-agnostic: Kokkos library that supports OpenMP, GPU (CUDA) and future architectures

Adaptive Mesh Refinement: own new AMR algorithms

For the astro community: Aimed at the RAMSES community (but not exclusively)
Exascale project: Dyablo-Ginea / Dyablo-Wholesun

Other initiative in the French community: SPH (Laibe), Idefix (Lesur), etc…

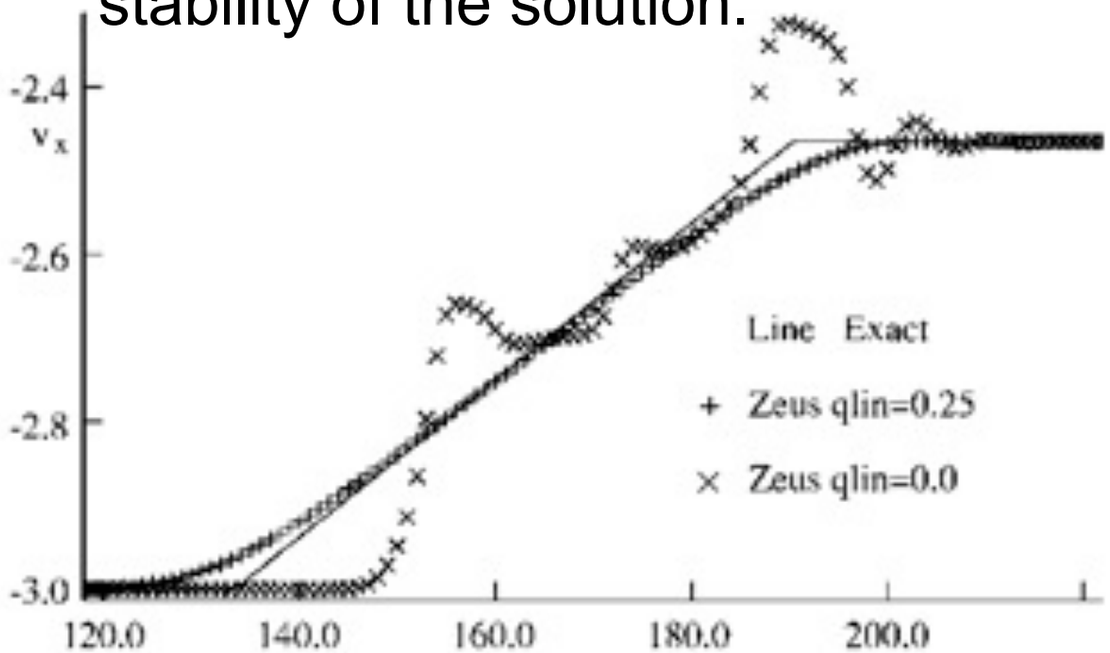


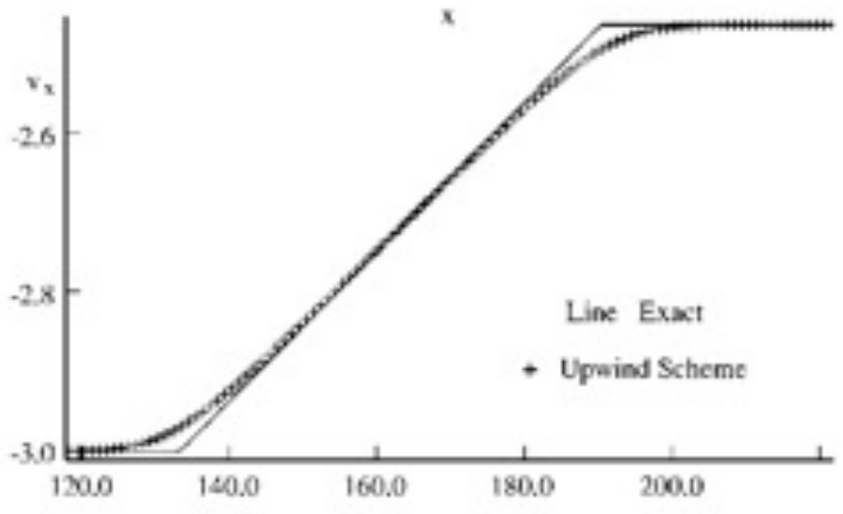
Database


Not Secure
galactica-simulations.eu/db/
Home
Topics
About
Search project/simulation
Log in
Turbulent flow in the interstellar medium
Galactica
The Galactica simulation database
The Galactica database results of heavy numerical simulations computed in various fields of computational astrophysics (solar magnetohydrodynamics, star-planet interactions, star formation, galaxy formation, galaxy mergers). The Galactica project gives observers and computational astrophysicists access to the results of these numerical simulations, which could be useful to help prepare or analyze observations or compare with other numerical studies.
The contributors of this database will provide a wide range of reduced data but will also give authenticated users the possibility to run online post-processing requests on the raw simulation data to fulfill one's specific needs.
Star-planet interactions
Project
Description
Fragdisk
Fragmentation of self-gravitating disks
Star formation
Project
Description
ORION
This project aims at providing a self-consistently description of molecular cloud fragmentation and evolution including the prestellar cores and up to the formation of dense fragments inside them (sink particles representing the stars)
LS
This project aims at modelling self-consistently a 1kpc piece of galaxy including star formation and stellar feedback
FRIG
This project aims at providing a self-consistently description from a 1kpc piece of galaxy to the prestellar cores
DustyCollapses
Dusty collapse calculations
Go to top

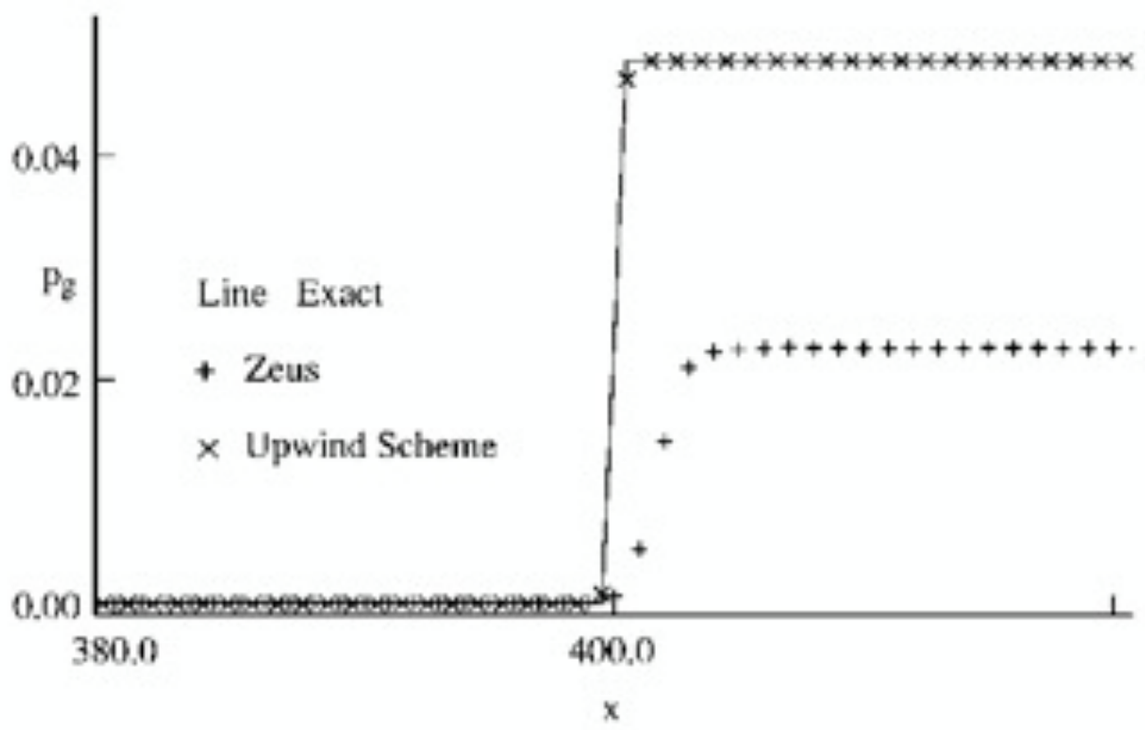
Why bother with a Godunov scheme ?

Proper upwinding of the numerical flux with respect to ALL waves ensures stability of the solution.





Using a strictly conservative update ensures proper jump conditions and therefore correct shock velocities.



Falle (2000)