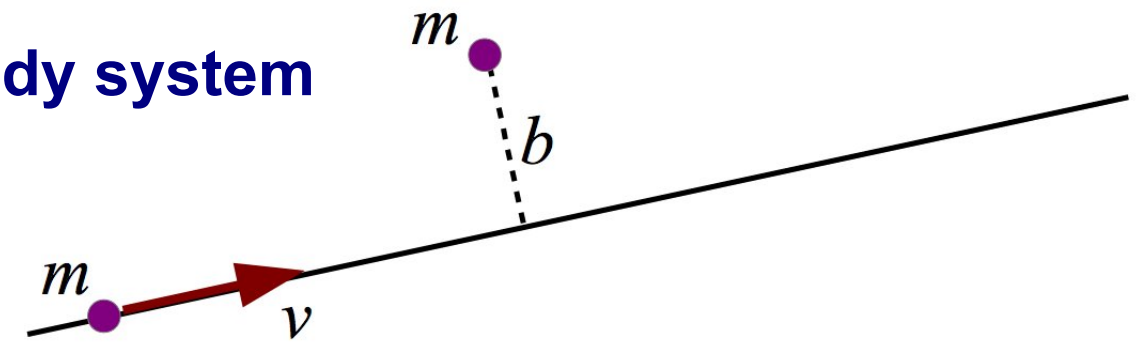
Numerical methods for cosmological simulations

Prof. Dr. Volker Springel

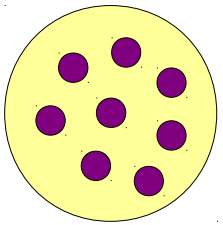
Basics of collisionless simulations

Relaxation time of an N-body system



Transverse momentum change:

$$\Delta p_\perp = m\Delta v_\perp = \int F_\perp \mathrm{d}t = \int \frac{Gm^2}{x^2+b^2}\frac{b}{\sqrt{x^2+b^2}}\frac{\mathrm{d}x}{v} = \frac{2Gm^2}{bv}$$



N points
size R

Particles encountered in one crossing in a ring

$$\mathrm{d}n = N\frac{2\pi b\,\mathrm{d}b}{\pi R^2}$$

Different encounters add incoherently:

$$(\Delta v)^2 = \int \left(\frac{2Gm}{bv}\right)^2 \mathrm{d}n = 8N\frac{Gm^2}{Rv}\int\frac{\mathrm{d}b}{b} = 8N\frac{Gm^2}{Rv}\ln\Lambda$$

Coloumb logarithm: $\Lambda = \ln\dfrac{b_{\max}}{b_{\min}}$

Typical specific energy of a particle: $$v^2 \simeq v_c^2 = \frac{GNm}{R}$$

Crossing time through the system: $$t_{\rm cross} = \frac{R}{v}$$

Relaxation time: $$t_{\rm relax} \equiv \frac{v^2}{(\Delta v)^2 / t_{\rm cross}} = \frac{N}{8 \ln \Lambda} t_{\rm cross}$$

But what about the Coloumb logarithm?

Maximal scattering: $\Delta v_\perp \simeq v$

$$\frac{2Gm}{b_{\rm min} v} = v \qquad b_{\rm min} \simeq \frac{2R}{N}$$

Minimal scattering: $b_{\rm max} \simeq R$

$\rightarrow \quad \Lambda \simeq \ln N/2.$

Relaxation time of N-body system:

$$t_{\rm relax} \simeq \frac{N}{8 \ln N} t_{\rm cross}$$

Small globular star cluster:

$$N \sim 10^5 \qquad t_{\rm cross} \sim \frac{3\,{\rm pc}}{6\,{\rm km/s}} \simeq \frac{1}{2}\,{\rm Myr}$$

This is a collisional system, and stellar encounters are important for the evolution.

Stars in a galaxy:

$$N \sim 10^{11} \qquad t_{\rm cross} \sim \frac{1}{100}\frac{1}{H_0}$$

Behaves as a collisionless system.

Dark matter particles in a galaxy (100 GeV WIMP):

$$N \sim 10^{77} \qquad t_{\rm cross} \sim \frac{1}{10 H_0}$$

The mother of all collisionless systems!

In an N-body model of a collisionless system, we must ensure that the simulated time is smaller than the relaxation time

$$t_{\rm sim} \ll t_{\rm relax}$$

We assume that the only appreciable interaction of dark matter particles is **gravity**

COLLISIONLESS DYNAMICS

Because there are **so many** dark matter particles, it's best to describe the system in terms of the **single particle distribution function**

$$f = f(\mathbf{x}, \mathbf{v}, t)$$

There are so many dark matter particles that they do not scatter locally on each other, they just respond to their collective gravitational field

Collisionless Boltzmann equation

Poisson-Vlasov System

$$\frac{\mathrm{d}f}{\mathrm{d}t} = \frac{\partial f}{\partial t} + \frac{\partial f}{\partial \mathbf{x}} \cdot \mathbf{v} + \frac{\partial f}{\partial \mathbf{v}} \cdot \left(-\frac{\partial \Phi}{\partial \mathbf{x}} \right) = 0$$

$$\nabla^2 \Phi(\mathbf{x}, t) = 4\pi G \int f(\mathbf{x}, \mathbf{v}, t)\, \mathrm{d}\mathbf{v}$$

Phase-space is conserved along each characteristic (i.e. particle orbit).

The number of stars in galaxies is so large that the two-body relaxation time by far exceeds the Hubble time. Stars in galaxies are therefore also described by the above system.

This system of partial differential equations is very difficult (impossible) to solve directly in non-trivial cases.

The N-body method uses a finite set of particles to sample the underlying distribution function

"MONTE-CARLO" APPROACH TO COLLISIONLESS DYNAMICS

We discretize in terms of N particles, which approximately move along characteristics of the underlying system.

$$\ddot{\mathbf{x}}_i = -\nabla_i \Phi(\mathbf{x}_i)$$

$$\Phi(\mathbf{x}) = -G \sum_{j=1}^{N} \frac{m_j}{\left[(\mathbf{x} - \mathbf{x}_j)^2 + \epsilon^2\right]^{1/2}}$$

The need for **gravitational softening**:

- Prevent large-angle particle scatterings and the formation of bound particle pairs.
- Helps to ensure that the two-body relaxation time is sufficiently large.
- Allows the system to be integrated with low-order integration schemes.

} Needed for faithful collisionless behavior

But how should we pick the gravitational softening length?

Let's first look at typical cosmological halos

$$M_{200} = 200 \times \rho_{\rm crit} \times \frac{4\pi}{3} R_{200}^3$$

$$v_{200}^2 \equiv \frac{GM_{200}}{R_{200}}$$

$$\rho_{\rm crit} = \frac{3H^2}{8\pi G}$$

Relations between halo virial quantities:

$$M_{200} = \frac{v_{200}^3}{10GH}$$

$$R_{200} = \frac{v_{200}}{10H}$$

Specific binding energy of a softened particle pair at vanishing distance

$$v_{\rm bind}^2 \simeq \frac{Gm}{\epsilon}$$

Specific energy of particles in the halo

$$v_{200}^2 \simeq (10GHM_{200})^{2/3}$$

For collisionless behavior, we must at least have:

$$v_{200}^2 > v_{\rm bind}^2$$

Let's introduce the mean particle distance: (for simplicity in a EdS universe)

$$\rho_{\rm crit} d^3 = m \qquad\qquad M_{200} = N \cdot m$$

Hence we get the condition:

$$\epsilon \geq \left(\frac{3}{800\pi N^2} \right)^{1/3} d$$

For: $N = 5$ $\epsilon \geq \frac{1}{30} d$

But if we also recall the relaxation time of dark matter in halos:

$$t_{\rm relax} = \frac{N}{8 \ln \Lambda} t_{\rm cross} \qquad\qquad t_{\rm cross} \simeq \frac{R_{200}}{v_{200}} = \frac{1}{10H}$$

We see that halos with well below 100 particles are typically always affected by relaxation over a Hubble time.

Compromise in practice:

$$\epsilon \sim \left(\frac{1}{40} - \frac{1}{20} \right) d$$

Derivation of the collisionless cosmological equation of motion

$$\mathbf{u} = \dot{\mathbf{r}}$$

Newtonian equation of motion

$$\frac{\mathrm{D}\mathbf{u}}{\mathrm{D}t} = -\nabla_r \Phi + \frac{\Lambda c^2}{3}\mathbf{r}$$

$$\frac{\mathrm{D}}{\mathrm{D}t} = \frac{\partial}{\partial t} + \mathbf{u}\cdot\nabla_r$$

$$\frac{\Lambda c^2}{3} \equiv \Omega_\Lambda H_0^2$$

Introduction of comoving coordinates

$$\mathbf{r} = a\mathbf{x}$$

Hubble flow

peculiar velocity

$$\mathbf{u} = \dot{\mathbf{r}} = \dot{a}\mathbf{x} + a\dot{\mathbf{x}} = H(t)\mathbf{r} + \mathbf{v}$$

Carry out a variable transformation of the equations...

$$(\mathbf{r}, t) \rightarrow (\mathbf{x}, t)$$

$$\nabla_r \rightarrow \frac{1}{a}\nabla_x$$

$$\left(\frac{\partial}{\partial t}\right)_r \rightarrow \left(\frac{\partial}{\partial t}\right)_x + \left(\frac{\partial \mathbf{x}}{\partial t}\right)_r \nabla_x$$

Rewriting yields...

$$\left[\left(\frac{\partial}{\partial t}\right)_x + \frac{1}{a}\left(\mathbf{v}\cdot\nabla_x\right)\right](\dot{a}\mathbf{x}+\mathbf{v}) = -\frac{1}{a}\nabla_x\Phi + \frac{\Lambda c^2}{3}a\mathbf{x}$$

And then...

$$\frac{\partial \mathbf{v}}{\partial t} + H(t)\mathbf{v} + \frac{1}{a}(\mathbf{v}\cdot\nabla_x)\mathbf{v} = -\frac{1}{a}\nabla_x\Phi - \ddot{a}\mathbf{x} + \frac{\Lambda c^2}{3}a\mathbf{x}$$

Recalling the Friedmann equation (in the matter dominated era)

$$\frac{\ddot{a}}{a} = -\frac{4\pi G}{3}\bar{\rho} + \frac{\Lambda c^2}{3}$$

yields the equation of motion in the form:

$$\frac{\mathrm{D}\mathbf{v}}{\mathrm{D}t} + H(t)\mathbf{v} = -\frac{1}{a}\nabla_x\Phi + \frac{4\pi G}{3}\bar{\rho}a\mathbf{x}$$

We define the peculiar gravitational potential as

$$\phi = \Phi - \frac{2\pi G}{3}\bar{\rho}a^2\mathbf{x}^2$$

This implies:

$$-\frac{1}{a}\nabla\phi = -\frac{1}{a}\nabla\Phi + \frac{4\pi G}{3}\bar{\rho}a\mathbf{x}$$

$$\nabla^2\phi = \nabla^2\Phi - 4\pi G\bar{\rho}a^2 = 4\pi G(\rho - \bar{\rho})a^2$$

So that we finally get the equations of motion as:

$$\frac{\mathrm{D}\mathbf{v}}{\mathrm{D}t} + H(t)\mathbf{v} = -\frac{1}{a}\nabla_x\phi$$

$$\nabla^2\phi = 4\pi G\bar{\rho}a^2\delta$$

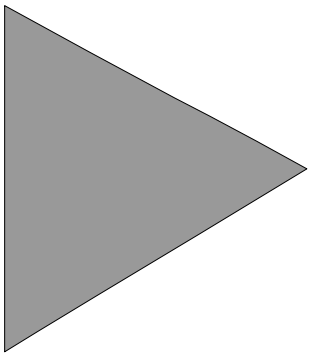
- Motion is created by density fluctuations around the background
- Infinite space is no problem any more

Two conflicting requirements complicate the study of **hierarchical** structure formation

DYNAMIC RANGE PROBLEM FACED BY COSMOLOGICAL SIMULATIONS

Want **small particle mass** to resolve internal structure of halos

Want **large volume** to obtain respresentative sample of universe



need large N

where N is the particle number

Problems due to a small box size:

- Fundamental mode goes non-linear soon after the first halos form. ⇨ Simulation cannot be meaningfully continued beyond this point.
- No rare objects (the first halo, **rich** galaxy clusters, etc.)

Problems due to a large particle mass:

- Physics cannot be resolved.
- Small galaxies are missed.

At any given time, halos exist on a large range of mass-scales !

Several questions come up when we try to use the N-body approach for collisionless simulations

- How do we compute the gravitational forces efficiently and accurately?
- How do we integrate the orbital equations in time?
- How do we generate appropriate initial conditions?
- How do we parallelize the simulation?

$$\ddot{\mathbf{x}}_i = -\nabla_i \Phi(\mathbf{x}_i)$$

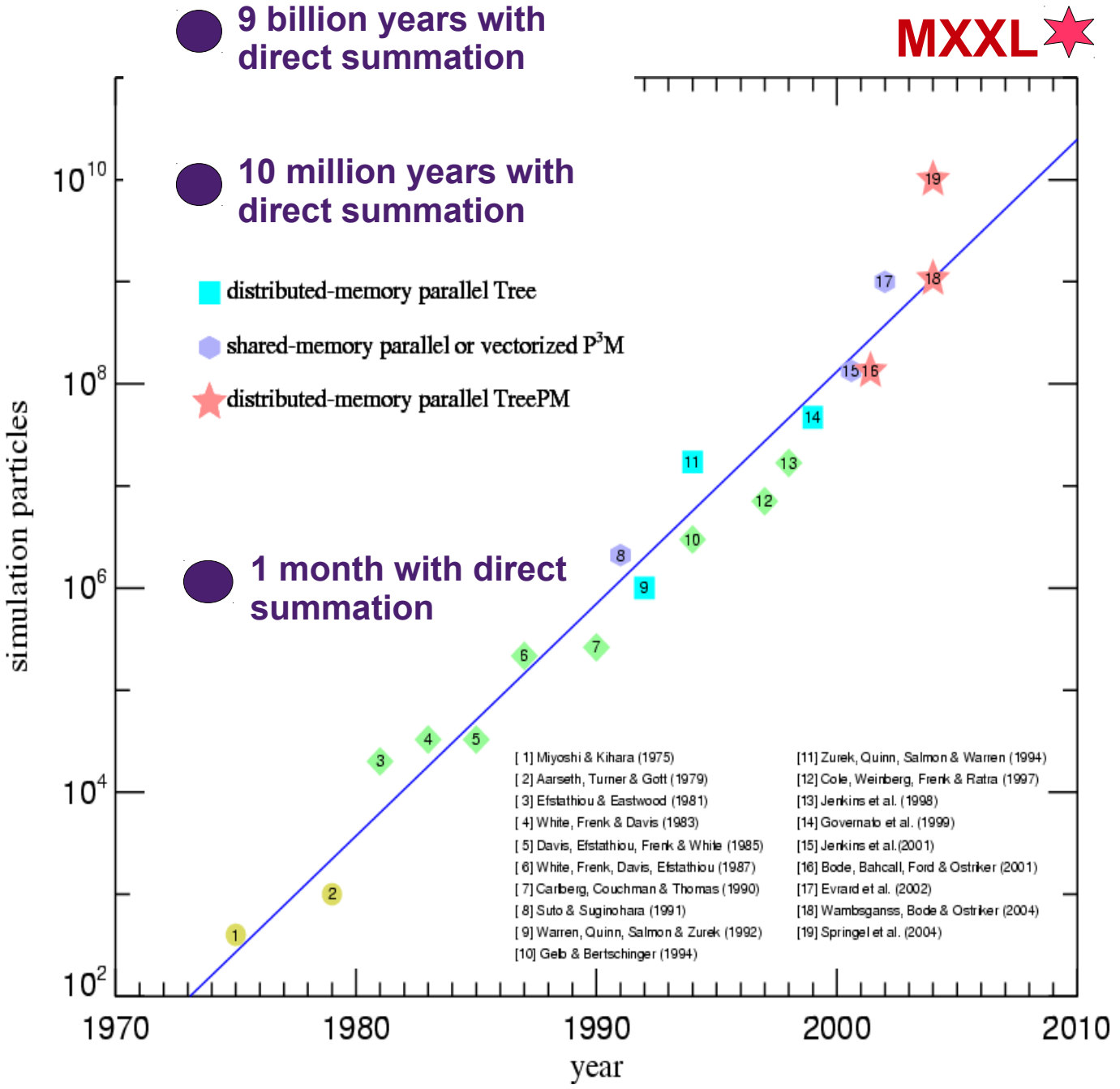
$$\Phi(\mathbf{x}) = -G \sum_{j=1}^{N} \frac{m_j}{[(\mathbf{x} - \mathbf{x}_j)^2 + \epsilon^2]^{1/2}}$$

Note: The naïve computation of the forces is an N^2 - task.

Cosmological N-body simulations have grown rapidly in size over the last three decades

"N" AS A FUNCTION OF TIME

- Computers double their speed every 18 months (Moore's law)
- N-body simulations have doubled their size every 16-17 months
- Recently, growth has accelerated further.



Time integration issues

Time integration methods

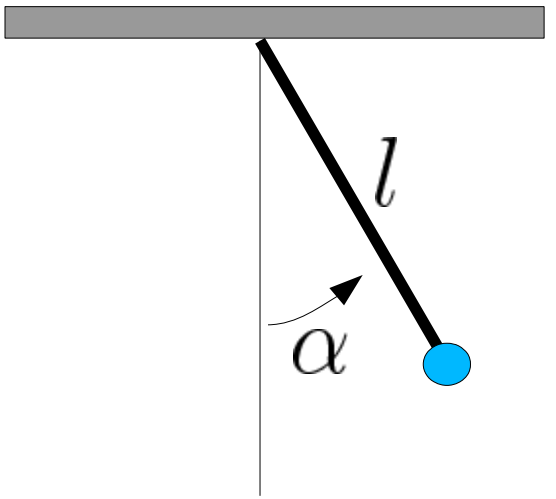
Want to numerically integrate an **ordinary differential equation** (ODE)

$$\dot{\mathbf{y}} = f(\mathbf{y})$$

Note: $\mathbf{y}$ can be a vector

Example: Simple pendulum

$$\ddot{\alpha} = -\frac{g}{l}\sin\alpha$$



$$y_0 \equiv \alpha \quad y_1 \equiv \dot{\alpha}$$

$$\dot{\mathbf{y}} = f(y) = \begin{pmatrix} y_1 \\ -\frac{g}{l}\sin y_0 \end{pmatrix}$$

A numerical approximation to the ODE is a set of values $\{\mathbf{y}_0, \mathbf{y}_1, \mathbf{y}_2, \ldots\}$ at times $\{t_0, t_1, t_2, \ldots\}$

There are many different ways for obtaining this.

Explicit Euler method

$$y_{n+1} = y_n + f(y_n)\Delta t$$

- Simplest of all
- Right hand-side depends only on things already known, **explicit method**
- The error in a single step is $O(\Delta t^2)$, but for the N steps needed for a finite time interval, the total error scales as $O(\Delta t)$!
- Never use this method, it's only **first order accurate**.

Implicit Euler method

$$y_{n+1} = y_n + f(y_{n+1})\Delta t$$

- **Excellent** stability properties
- Suitable for very stiff ODE
- Requires implicit solver for y_{n+1}
- But still low order

Implicit mid-point rule

$$y_{n+1} = y_n + f\left(\frac{y_n + y_{n+1}}{2}\right)\Delta t$$

- **2nd order accurate**
- Time-symmetric, in fact **symplectic**
- But still implicit...

Runge-Kutta methods

whole class of integration methods

2nd order accurate

$$\begin{aligned} k_1 &= f(y_n) \\ k_2 &= f(y_n + k_1\Delta t) \\ y_{n+1} &= y_n + \left(\frac{k_1 + k_2}{2}\right)\Delta t \end{aligned}$$

4th order accurate.

$$\begin{aligned} k_1 &= f(y_n, t_n) \\ k_2 &= f(y_n + k_1\Delta t/2, t_n + \Delta t/2) \\ k_3 &= f(y_n + k_2\Delta t/2, t_n + \Delta t/2) \\ k_4 &= f(y_n + k_3\Delta t/2, t_n + \Delta t) \\ y_{n+1} &= y_n + \left(\frac{k_1}{6} + \frac{k_2}{3} + \frac{k_3}{3} + \frac{k_4}{6}\right)\Delta t \end{aligned}$$

The Leapfrog

For a second order ODE: $\ddot{\mathbf{x}} = f(\mathbf{x})$

"Drift-Kick-Drift" version

$$
\begin{aligned}
x_{n+\frac{1}{2}} &= x_n + v_n \frac{\Delta t}{2} \\
v_{n+1} &= v_n + f(x_{n+\frac{1}{2}}) \Delta t \\
x_{n+1} &= x_{n+\frac{1}{2}} + v_{n+1} \frac{\Delta t}{2}
\end{aligned}
$$

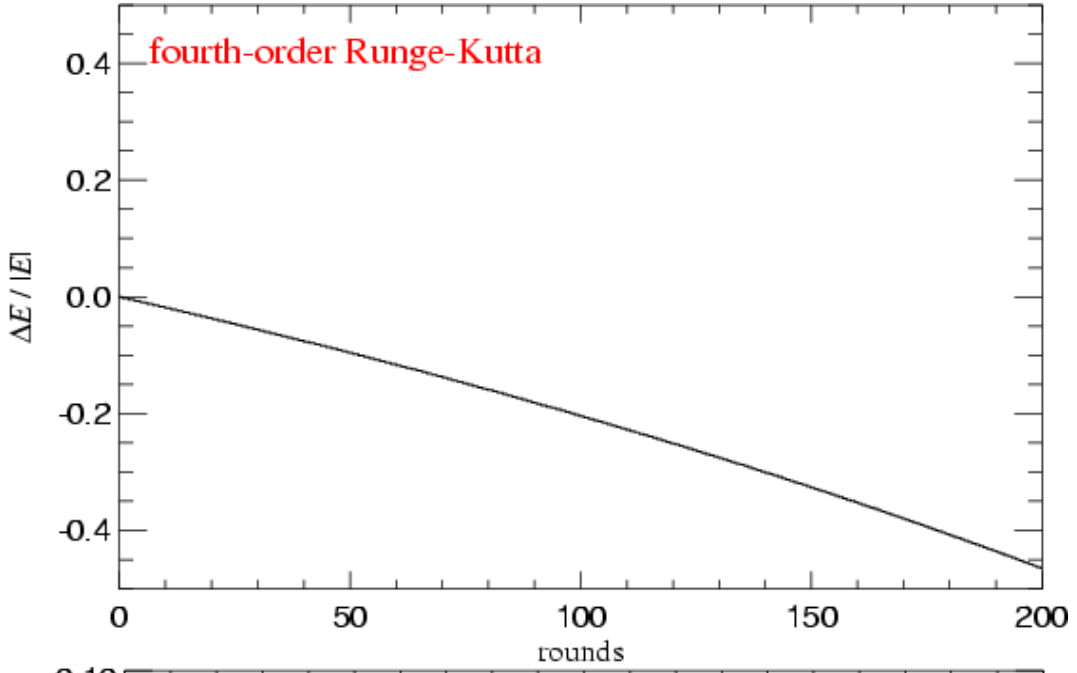
"Kick-Drift-Kick" version

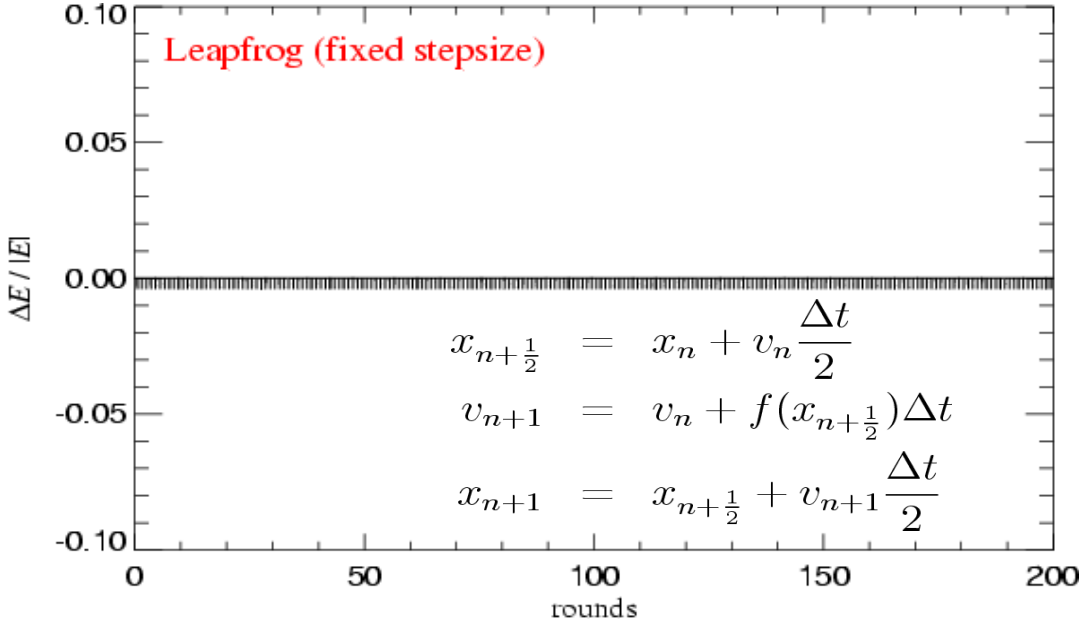
$$
\begin{aligned}
v_{n+\frac{1}{2}} &= v_n + f(x_n) \frac{\Delta t}{2} \\
x_{n+1} &= x_n + v_{n+\frac{1}{2}} \frac{\Delta t}{2} \\
v_{n+1} &= v_{n+\frac{1}{2}} + f(x_{n+1}) \frac{\Delta t}{2}
\end{aligned}
$$

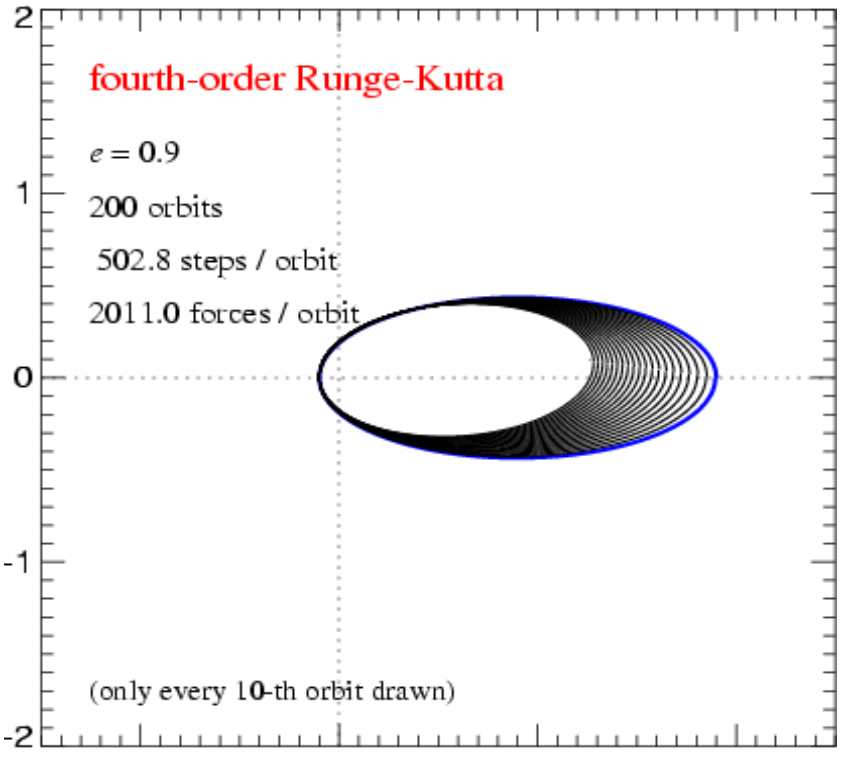
- **2nd order accurate**
- **symplectic**
- can be rewritten into time-centered formulation

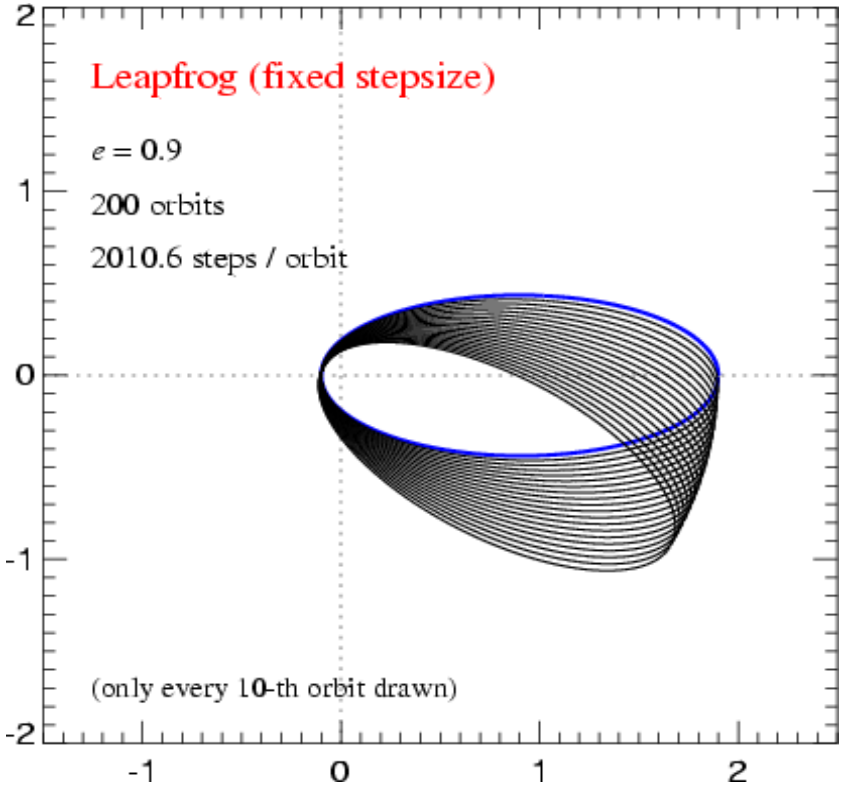
The leapfrog is behaving much better than one might expect...

INTEGRATING THE KEPLER PROBLEM



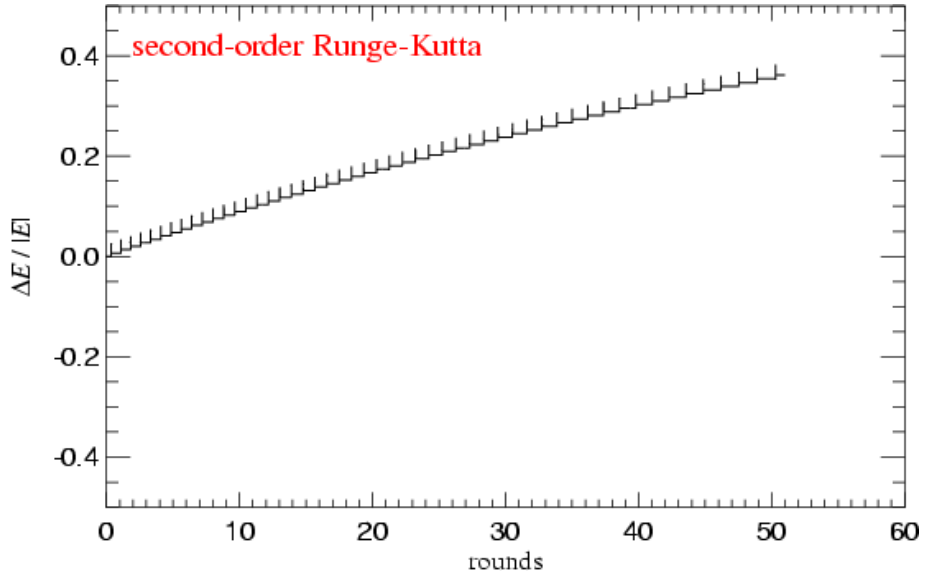


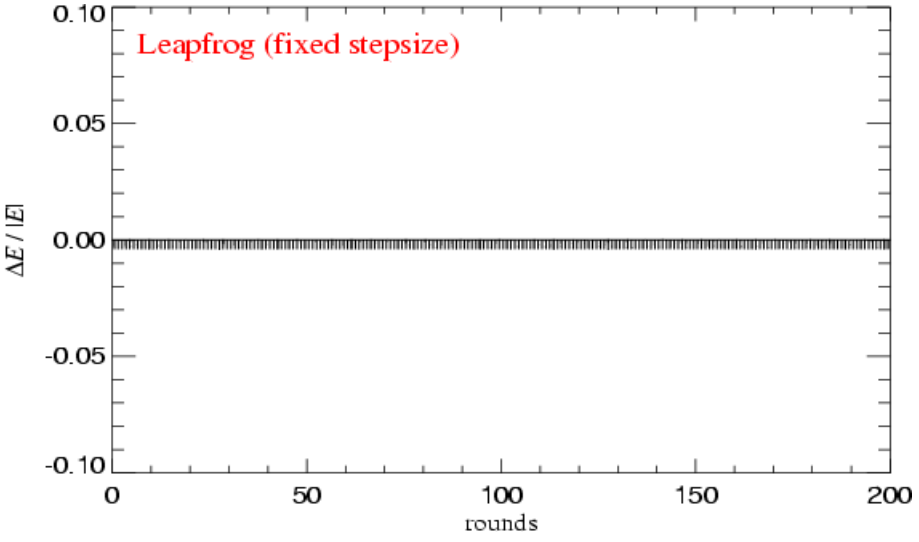


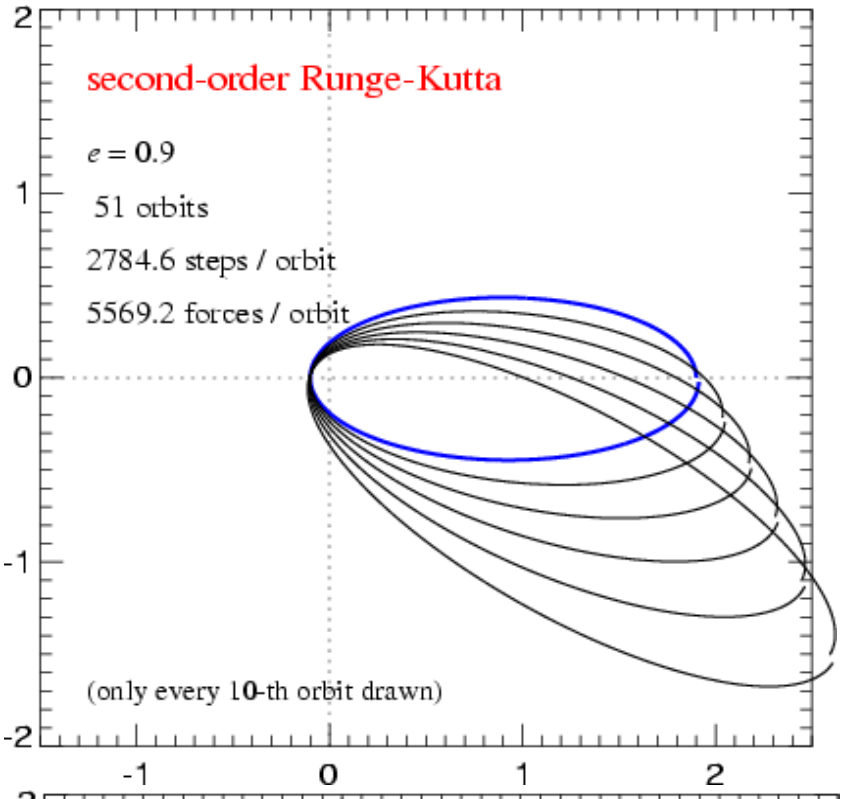


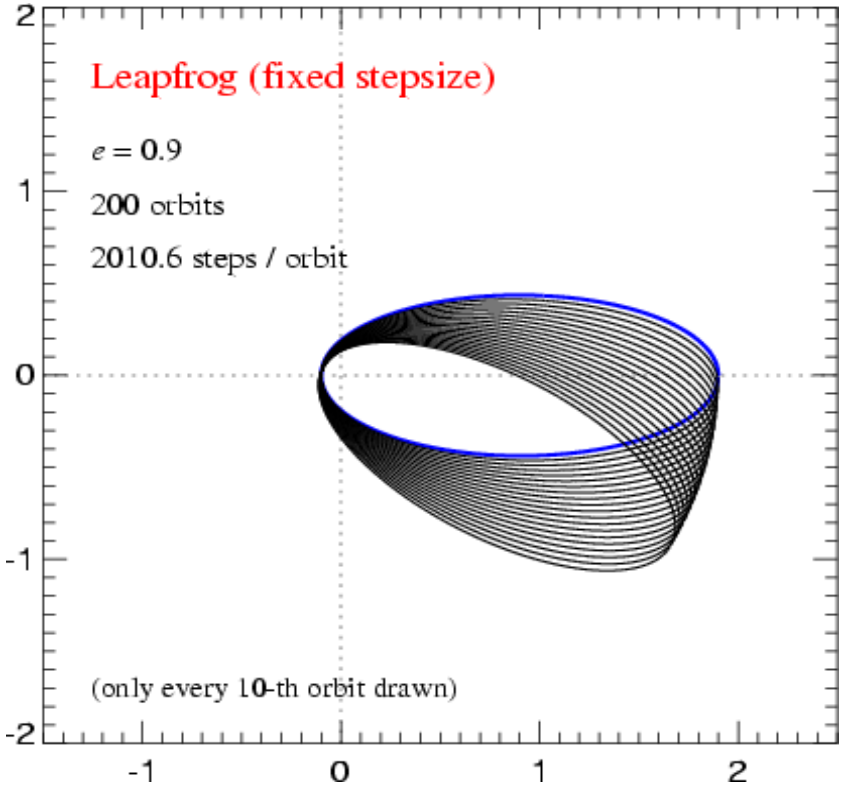
When compared with an integrator of the same order, the leapfrog is highly superior

INTEGRATING THE KEPLER PROBLEM



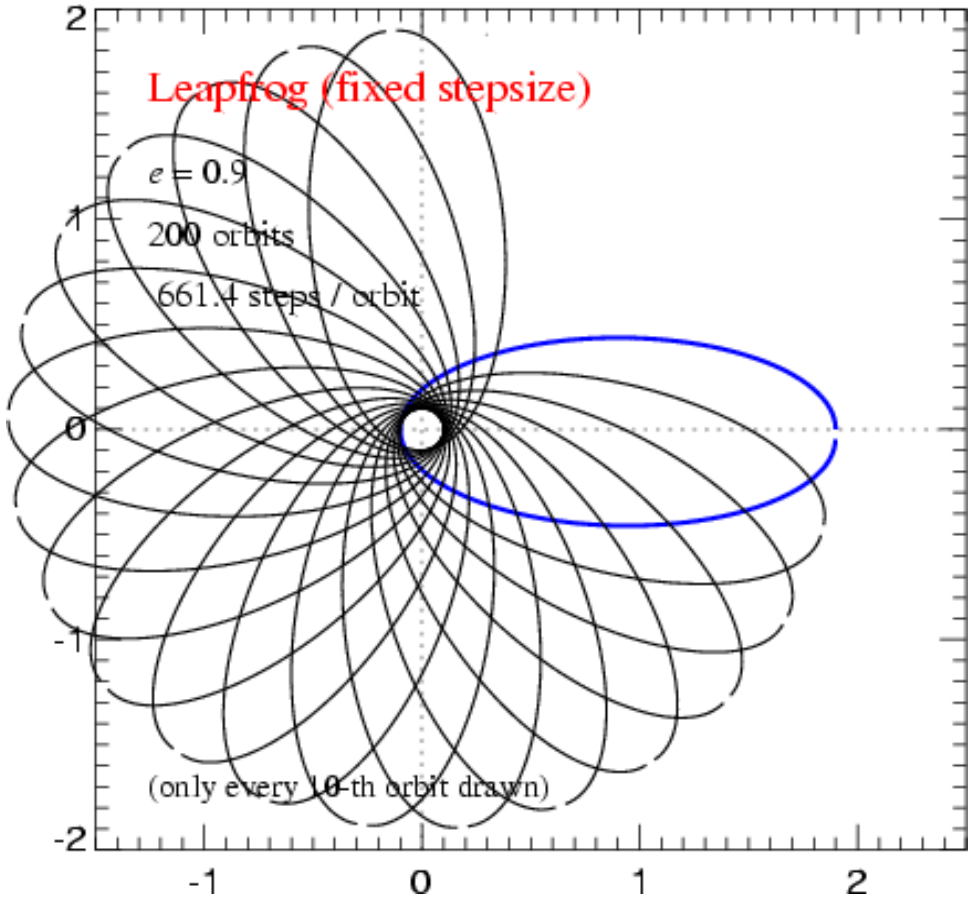


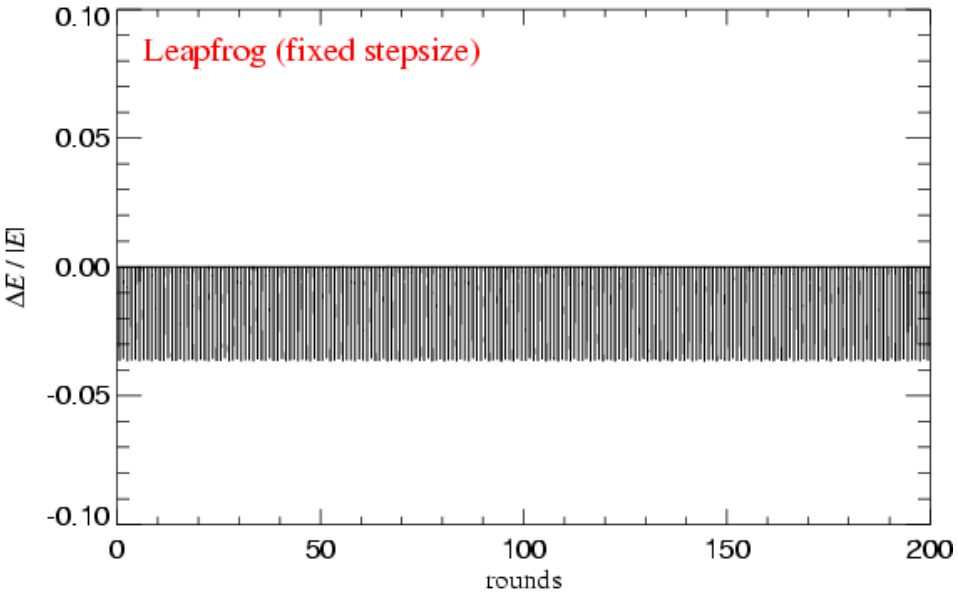




Even for rather large timesteps, the leapfrog maintains qualitatively correct behaviour without long-term secular trends

INTEGRATING THE KEPLER PROBLEM





What is the underlying mathematical reason for the very good long-term behaviour of the leapfrog ?

HAMILTONIAN SYSTEMS AND SYMPLECTIC INTEGRATION

$$H(\mathbf{p}_1,\ldots,\mathbf{p}_n,\mathbf{x}_1,\ldots,\mathbf{x}_n) = \sum_i \frac{\mathbf{p}_i^2}{2m_i} + \frac{1}{2}\sum_{ij} m_i m_j \phi(\mathbf{x}_i - \mathbf{x}_j)$$

The Hamiltonian structure of the system can be preserved in the integration if each step is formulated as a *canonical transformation*. Such integration schemes are called *symplectic*.

Poisson bracket:

$$\{A,B\} \equiv \sum_i \left(\frac{\partial A}{\partial \mathbf{x}_i}\frac{\partial B}{\partial \mathbf{p}_i} - \frac{\partial A}{\partial \mathbf{p}_i}\frac{\partial B}{\partial \mathbf{x}_i} \right)$$

Hamilton's equations

$$\frac{\mathrm{d}\mathbf{x}_i}{\mathrm{d}t} = \{\mathbf{x}_i, H\}$$

$$\frac{\mathrm{d}\mathbf{p}_i}{\mathrm{d}t} = \{\mathbf{p}_i, H\}$$

Hamilton operator

$$\mathbf{H}f \equiv \{f, H\}$$

System state vector

$$|t\rangle \equiv |\mathbf{x}_1(t),\ldots,\mathbf{x}_n(t),\mathbf{p}_1(t),\ldots,\mathbf{p}_n(t),t\rangle$$

Time evolution operator

$$|t_1\rangle = \mathbf{U}(t_1,t_0)\ |t_0\rangle$$

$$\mathbf{U}(t+\Delta t,t) = \exp\left(\int_t^{t+\Delta t} \mathbf{H}\,\mathrm{d}t\right)$$

The time evolution of the system is a continuous canonical transformation generated by the Hamiltonian.

Symplectic integration schemes can be generated by applying the idea of operating splitting to the Hamiltonian

THE LEAPFROG AS A SYMPLECTIC INTEGRATOR

Separable Hamiltonian

$$H = H_{\rm kin} + H_{\rm pot}$$

Drift- and Kick-Operators

$$\mathbf{D}(\Delta t) \equiv \exp\left(\int_t^{t+\Delta t} \mathrm{d}t\, \mathbf{H}_{\rm kin}\right) = \left\{ \begin{array}{lcl} \mathbf{p}_i & \mapsto & \mathbf{p}_i \\ \mathbf{x}_i & \mapsto & \mathbf{x}_i + \frac{\mathbf{p}_i}{m_i}\Delta t \end{array} \right.$$

$$\mathbf{K}(\Delta t) = \exp\left(\int_t^{t+\Delta t} \mathrm{d}t\, \mathbf{H}_{\rm pot}\right) = \left\{ \begin{array}{lcl} \mathbf{x}_i & \mapsto & \mathbf{x}_i \\ \mathbf{p}_i & \mapsto & \mathbf{p}_i - \sum_j m_i m_j \frac{\partial \phi(\mathbf{x}_{ij})}{\partial \mathbf{x}_i}\Delta t \end{array} \right.$$

The drift and kick operators are symplectic transformations of phase-space !

The Leapfrog

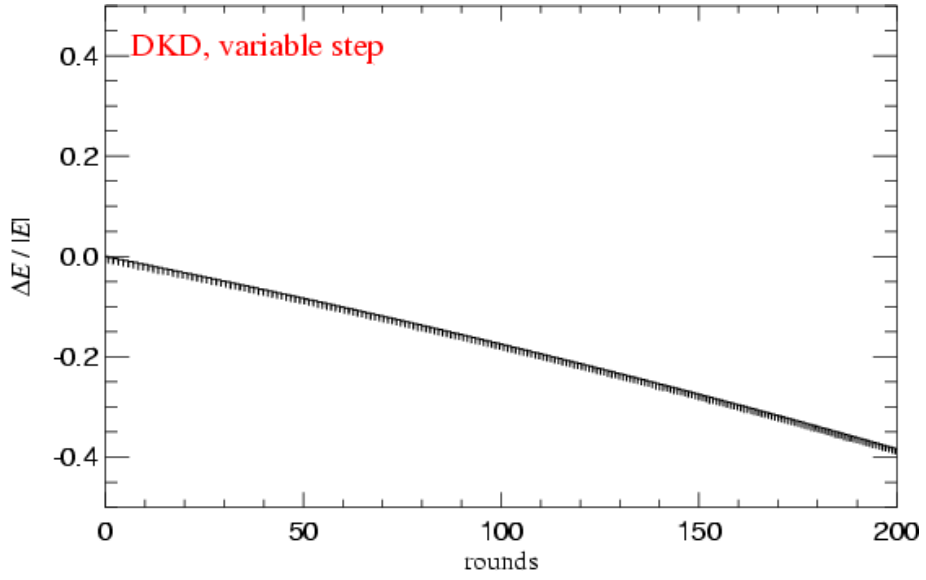
Drift-Kick-Drift: $$\tilde{\mathbf{U}}(\Delta t) = \mathbf{D}\left(\frac{\Delta t}{2}\right)\, \mathbf{K}(\Delta t)\, \mathbf{D}\left(\frac{\Delta t}{2}\right)$$

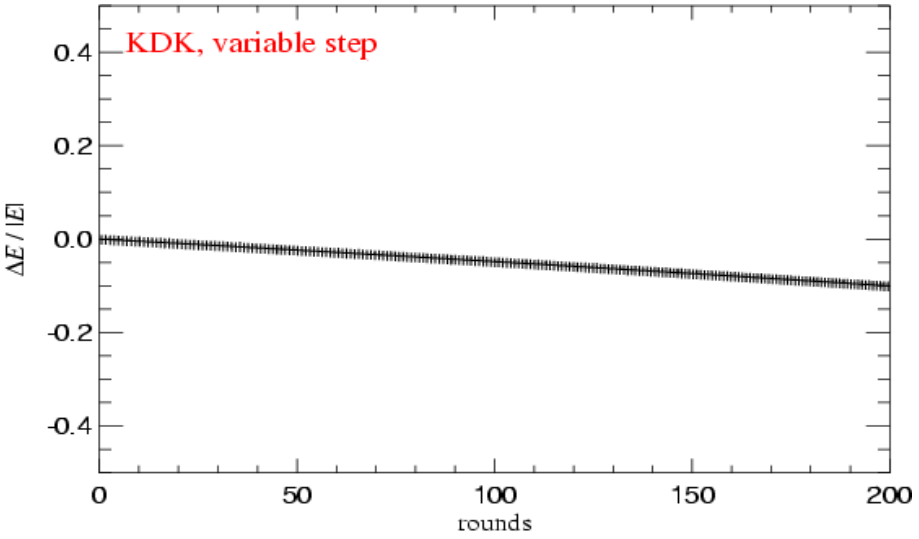
Kick-Drift-Kick: $$\tilde{\mathbf{U}}(\Delta t) = \mathbf{K}\left(\frac{\Delta t}{2}\right)\, \mathbf{D}(\Delta t)\, \mathbf{K}\left(\frac{\Delta t}{2}\right)$$

Hamiltonian of the numerical system: $$\tilde{H} = H + H_{\rm err} \qquad H_{\rm err} = \frac{\Delta t^2}{12}\left\{\{H_{\rm kin}, H_{\rm pot}\}, H_{\rm kin} + \frac{1}{2}H_{\rm pot}\right\} + \mathcal{O}(\Delta t^3)$$

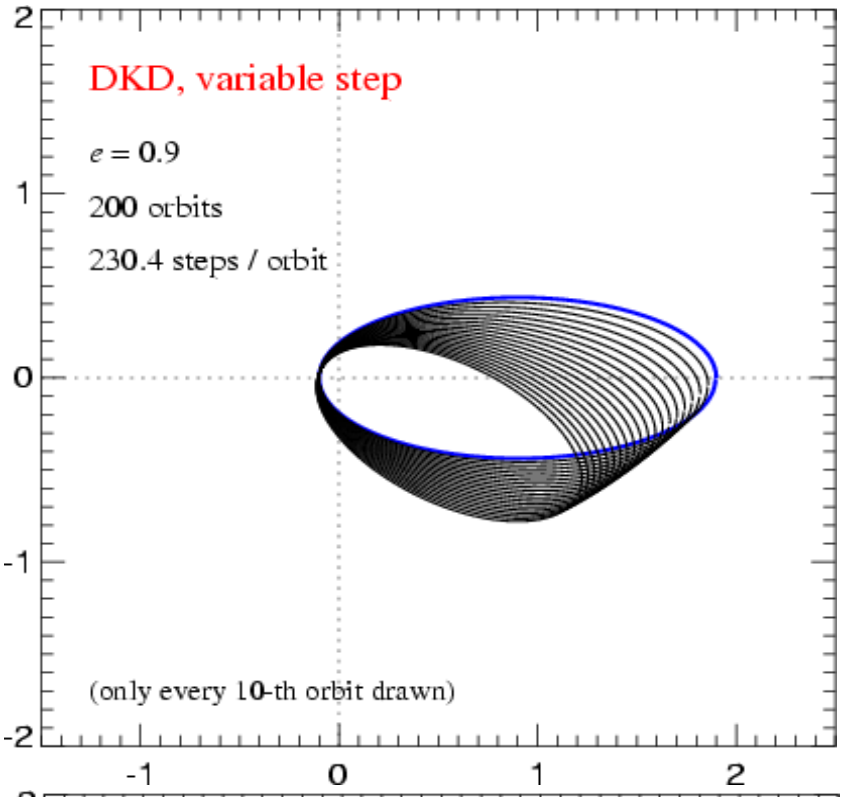
When an adaptive timestep is used, much of the symplectic advantage is lost

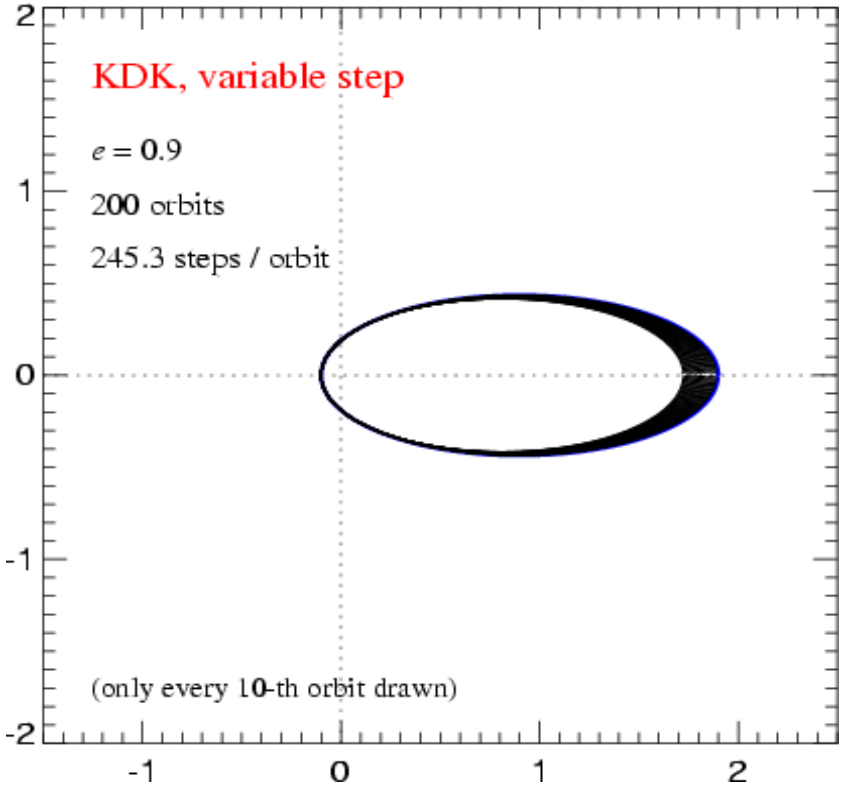
INTEGRATING THE KEPLER PROBLEM





→ Going to KDK reduces the error by a factor 4, at the same cost !

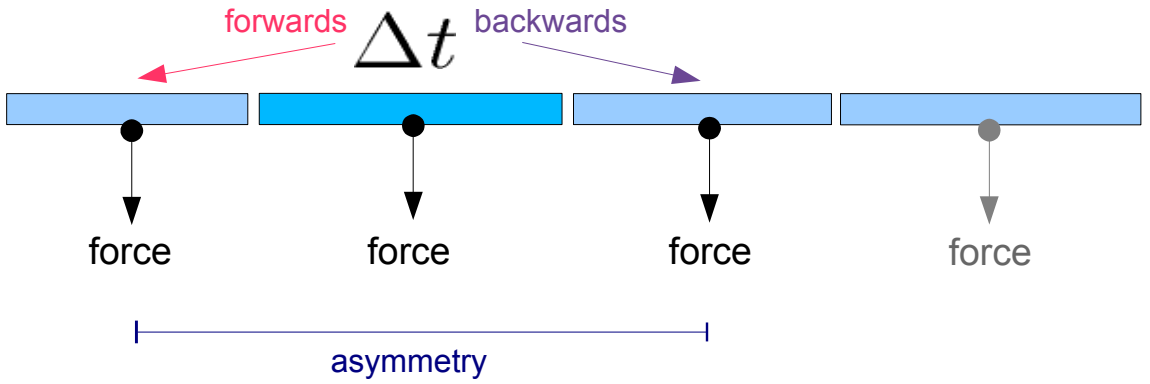




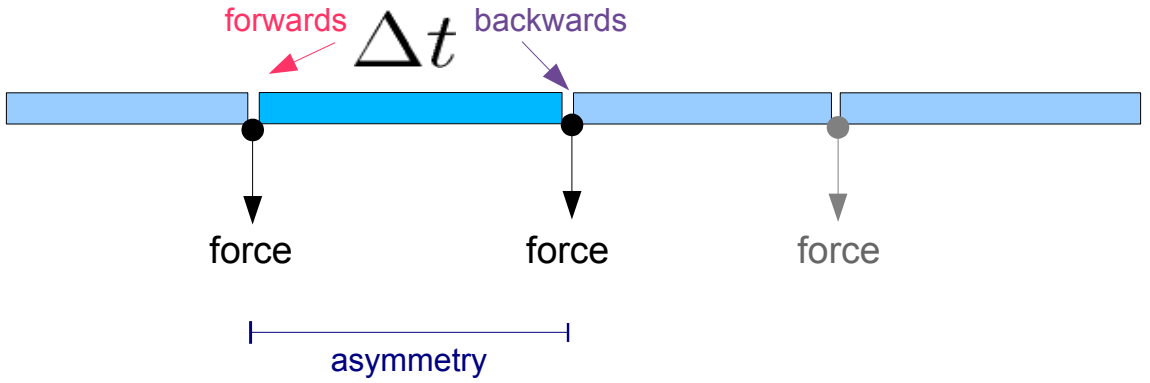
For periodic motion with adaptive timesteps, the DKD leapfrog shows more time-asymmetry than the KDK variant

LEAPFROG WITH ADAPTIVE TIMESTEP

DKD



KDK



Collisionless dynamics in an expanding universe is described by a Hamiltonian system

THE HAMILTONIAN IN COMOVING COORDINATES

Conjugate momentum $\mathbf{p} = a^2 \dot{\mathbf{x}}$

$$H(\mathbf{p}_1, \ldots, \mathbf{p}_n, \mathbf{x}_1, \ldots, \mathbf{x}_n, t) = \sum_i \frac{\mathbf{p}_i^2}{2 m_i a(t)^2} + \frac{1}{2} \sum_{ij} \frac{m_i m_j \phi(\mathbf{x}_i - \mathbf{x}_j)}{a(t)}$$

Drift- and Kick operators

$$\mathbf{D}(t + \Delta t, t) = \exp\left(\int_t^{t+\Delta t} \mathrm{d}t \, \mathbf{H}_{\mathrm{kin}} \right) = \left\{ \begin{array}{lcl} \mathbf{p}_i & \mapsto & \mathbf{p}_i \\ \mathbf{x}_i & \mapsto & \mathbf{x}_i + \frac{\mathbf{p}_i}{m_i} \int_t^{t+\Delta t} \frac{\mathrm{d}t}{a^2} \end{array} \right.$$

$$\mathbf{K}(t + \Delta t, t) = \exp\left(\int_t^{t+\Delta t} \mathrm{d}t \, \mathbf{H}_{\mathrm{pot}} \right) = \left\{ \begin{array}{lcl} \mathbf{x}_i & \mapsto & \mathbf{x}_i \\ \mathbf{p}_i & \mapsto & \mathbf{p}_i - \sum_j m_i m_j \frac{\partial \phi(\mathbf{x}_{ij})}{\partial \mathbf{x}_i} \int_t^{t+\Delta t} \frac{\mathrm{d}t}{a} \end{array} \right.$$

Choice of timestep

For linear growth, fixed step in log(a) appears most appropriate... → timestep is then a constant fraction of the Hubble time

$$\Delta t = \frac{\Delta \log a}{H(a)}$$

Calculating gravitational forces

Direct summation calculates the gravitational field **exactly**

FORCE ACCURACY IN COLLISIONLESS SIMULATIONS

Direct summation approch:

$$\ddot{\mathbf{x}}_i = -\nabla_i \Phi(\mathbf{x}_i)$$

$$\Phi(\mathbf{x}) = -G \sum_{j=1}^{N} \frac{m_j}{\left[(\mathbf{x}-\mathbf{x}_j)^2 + \epsilon^2\right]^{1/2}}$$

N^2 complexity

Are ***approximate*** force calculations sufficient?

Yes, provided the force errors are random and small enough.

Since the N-body force field is noisy anyway, small random errors will only insignificantly reduce the relaxation time.

Systematic errors in the force, or error correlations are however very problematic.

The particle mesh (PM) force calculation

The particle-mesh method

Poisson's equation can be solved in real-space by a convolution of the density field with a Green's function.

$$\Phi(\mathbf{x}) = \int g(\mathbf{x}-\mathbf{x}')\,\rho(\mathbf{x})\,\mathrm{d}\mathbf{x}'$$

Example for vacuum boundaries:

$$\Phi(\mathbf{x}) = -G\int \frac{\rho(\mathbf{x})}{|\mathbf{x}-\mathbf{x}'|}\,\mathrm{d}\mathbf{x}' \qquad g(\mathbf{x}) = -\frac{G}{|\mathbf{x}|}$$

In Fourier-space, the convolution becomes a simple multiplication!

$$\hat{\Phi}(\mathbf{k}) = \hat{g}(\mathbf{k})\cdot\hat{\rho}(\mathbf{k})$$

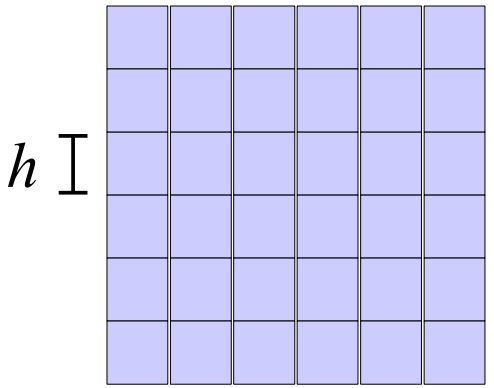
→ **Solve the potential in these steps:**

(1) FFT forward of the density field
(2) Multiplication with the Green's function
(3) FFT backwards to obtain potential

The four steps of the PM algorithm

(a) Density assignment
(b) Computation of the potential
(c) Determination of the force field
(d) Assignment of forces to particles

Density assignment



$\{\mathbf{x}_m\}$ set of discrete mesh centres

Give particles a “shape” S(x). Then to each mesh cell, we assign the fraction of mass that falls into this cell. The overlap for a cell is given by:

$$W(\mathbf{x_m} - \mathbf{x}_i) = \int_{\mathbf{x_m}-\frac{h}{2}}^{\mathbf{x_m}+\frac{h}{2}} S(\mathbf{x}' - \mathbf{x}_i)\,\mathrm{d}\mathbf{x}' = \int \Pi\left(\frac{\mathbf{x}' - \mathbf{x_m}}{h}\right) S(\mathbf{x}' - \mathbf{x}_i)\,\mathrm{d}\mathbf{x}'$$

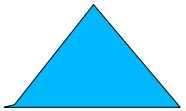
The assignment function is hence the convolution:

$$W(\mathbf{x}) = \Pi\left(\frac{\mathbf{x}}{h}\right) \star S(\mathbf{x}) \qquad \text{where} \qquad \Pi(x) = \begin{cases} 1 & \text{for } |x| \leq \frac{1}{2} \\ 0 & \text{otherwise} \end{cases}$$

The density on the mesh is then a sum over the contributions of each particle as given by the assignment function:

$$\rho(\mathbf{x_m}) = \frac{1}{h^3} \sum_{i=1}^{N} m_i\, W(\mathbf{x}_i - \mathbf{x_m})$$

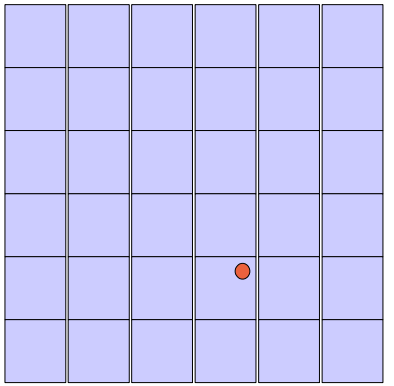
Commenly used particle shape functions and assignment schemes

Name	Shape function S(x)	# of cells involved	Properties of force
NGP Nearest grid point	 $\delta(\mathbf{x})$	$1^3 = 1$	piecewise constant in cells
CIC Clouds in cells	 $\frac{1}{h^3}\Pi\left(\frac{\mathbf{x}}{h}\right) \star \delta(\mathbf{x})$	$2^3 = 8$	piecewise linear, continuous
TSC Triangular shaped clouds	 $\frac{1}{h^3}\Pi\left(\frac{\mathbf{x}}{h}\right) \star \frac{1}{h^3}\Pi\left(\frac{\mathbf{x}}{h}\right)$	$3^3 = 27$	continuous first derivative

Note: For interpolation of the grid to obtain the forces, the same assignment function needs to be used to ensure momentum conservation. (In the CIC case, this is identical to tri-linear interpolation.)

Finite differencing of the potential to get the force field

Approximate the force field $\mathbf{f} = -\nabla\Phi$ by finite differencing



2nd order accurate scheme:

$$f^{(x)}_{i,j,k} = -\frac{\Phi_{i+1,j,k} - \Phi_{i-1,j,k}}{2h}$$

4th order accurate scheme:

$$f^{(x)}_{i,j,k} = -\frac{4}{3}\frac{\Phi_{i+1,j,k} - \Phi_{i-1,j,k}}{2h} + \frac{1}{3}\frac{\Phi_{i+2,j,k} - \Phi_{i-2,j,k}}{4h}$$

Interpolating the mesh-forces to the particle locations

$$F(\mathbf{x}_i) = \sum_{\mathbf{m}} W(\mathbf{x}_i - \mathbf{x}_\mathbf{m})\, f_\mathbf{m}$$

The interpolation kernel needs to be the same one used for mass-assignment to ensure force anti-symmetry.

Advantages and disadvantages of the PM-scheme

Pros: **SPEED** and simplicity

Cons:

- Spatial force resolution limited to mesh size.
- Force errors somewhat anisotropic on the scale of the cell size

→ *serious problem:*

cosmological simulations cluster strongly and have a very large dynamic range

cannot make the PM-mesh fine enough and resolve internal structure of halos as well as large cosmological scales

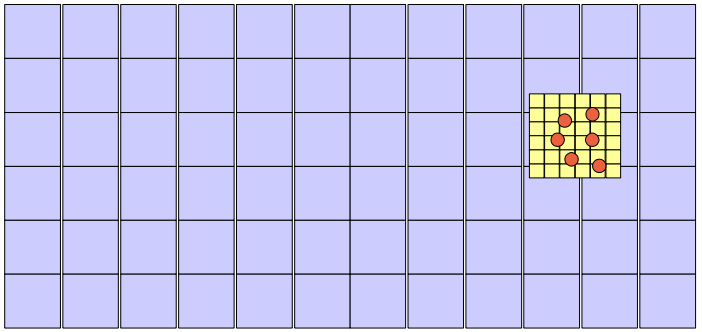
→ we need a method to increase the **dynamic range** available in the force calculation

Particle-Particle PM schemes (P^3M)

Idea: Supplement the PM force with a direct summation short-range force at the scale of the mesh cells. The particles in cells are linked together by a chaining list.

Offers much higher dynamic range, but becomes slow when clustering sets in.

In AP^3M, mesh-refinements are placed on clustered regions



Can avoid clustering slow-down, but has higher complexity and ambiguities in mesh placement

Codes that use AP^3M: **HYDRA** (Couchman)

Iterative Poisson solvers can determine the potential directly on a (hierarchical) grid

Idea: Start with a trial potential and then iteratively relax the solution by updating with a finite difference approximation to the Laplacian.

$$\Phi'_{i,j,k} = \frac{1}{6}\left(\Phi_{i+1,j,k} + \Phi_{i-1,j,k} + \Phi_{i,j+1,k} + \Phi_{i,j-1,k} + \Phi_{i,j,k+1} + \Phi_{i,j,k-1} - 4\pi G h^2 \rho_{i,j,k}\right)$$

This updating eliminates errors on the scale of a few grid cells rapidly, but longer-range fluctuations die out much more slowly.

In **multigrid methods**, a hierarchy of meshes is used to speed up convergence, resulting in a fast method that allows for locally varying resolution.

Examples for codes that use a real-space Poisson solver:

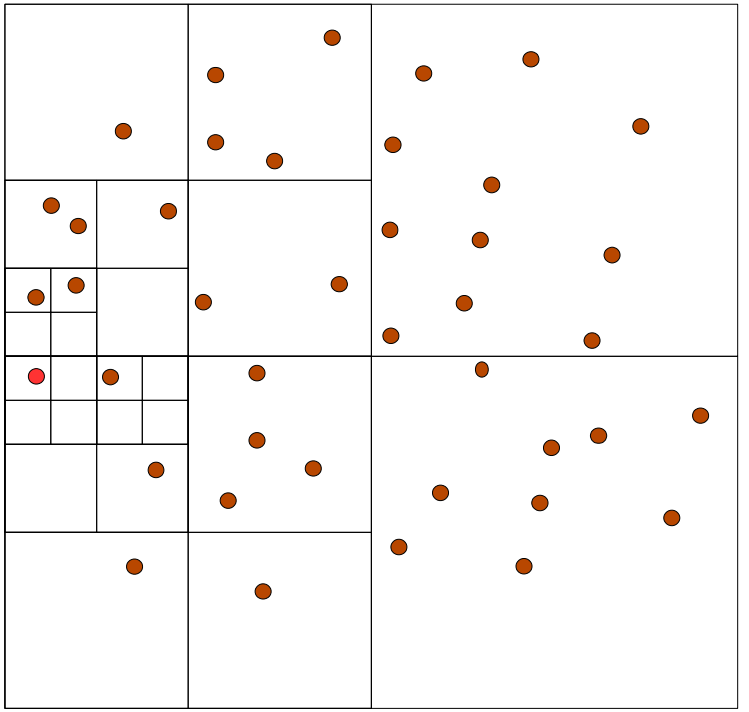
RAMSES (Teyssier)
ART (Kravtsov)
MLAPM (Knebe)

On adaptive meshes, sometimes a combination of Fourier techniques and real-space solvers is used.

TREE algorithms

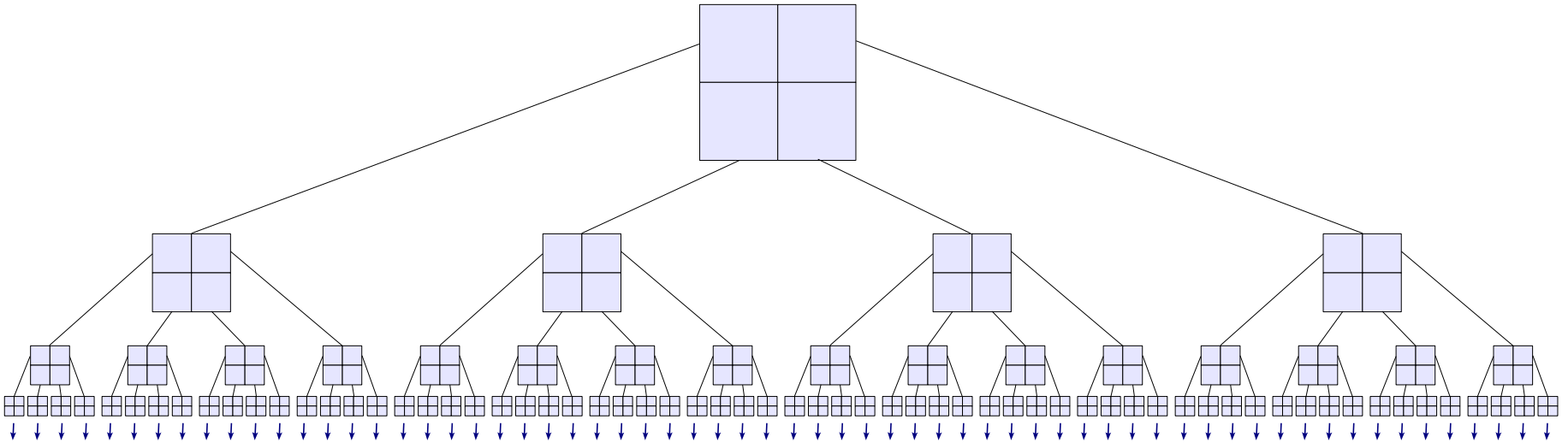
Tree algorithms approximate the force on a point with a multipole expansion

HIERARCHICAL TREE ALGORITHMS

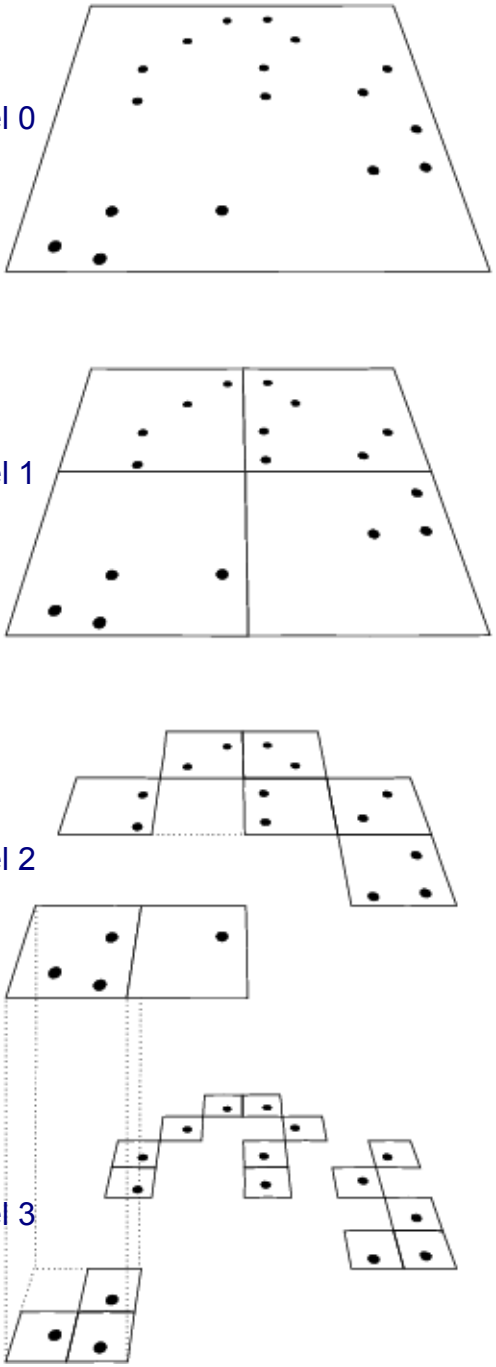


Idea: Group distant particles together, and use their multipole expansion.

→ Only ~ log(N) force terms per particle.



Oct-tree in two dimensions



Tree algorithms

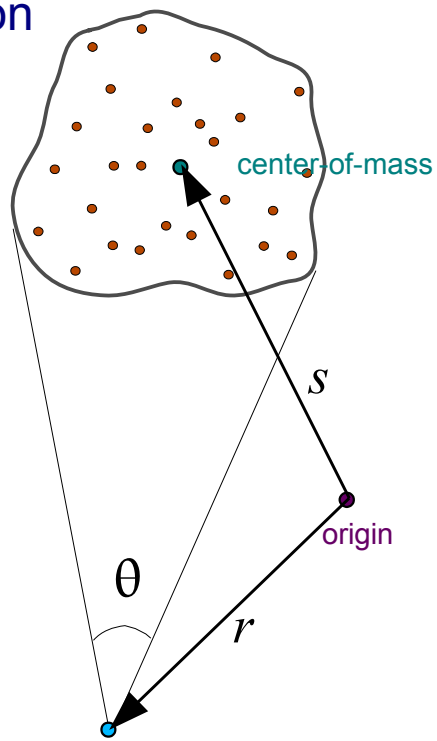
Idea: Use hierarchical multipole expansion to account for distant particle groups

$$\Phi(\mathbf{r}) = -G\sum_i \frac{m_i}{|\mathbf{r}-\mathbf{x}_i|}$$

We expand:

$$\frac{1}{|\mathbf{r}-\mathbf{x}_i|} = \frac{1}{|(\mathbf{r}-\mathbf{s})-(\mathbf{x}_i-\mathbf{s})|}$$

for $|\mathbf{x}_i-\mathbf{s}| \ll |\mathbf{r}-\mathbf{s}|$ $\mathbf{y} \equiv \mathbf{r}-\mathbf{s}$



and obtain:

$$\frac{1}{|\mathbf{y}+\mathbf{s}-\mathbf{x}_i|} = \frac{1}{|\mathbf{y}|} - \frac{\mathbf{y}\cdot(\mathbf{s}-\mathbf{x}_i)}{|\mathbf{y}|^3} + \frac{1}{2}\frac{\mathbf{y}^T\left[3(\mathbf{s}-\mathbf{x}_i)(\mathbf{s}-\mathbf{x}_i)^T - \mathbf{I}(\mathbf{s}-\mathbf{x}_i)^2\right]\mathbf{y}}{|\mathbf{y}|^5} + \ldots$$

the dipole term vanishes when summed over all particles in the group

The multipole moments are computed for each node of the tree

Monpole moment: Mass and center-of-mass

$$M = \sum_i m_i$$

Quadrupole tensor:

$$Q_{ij} = \sum_k m_k \left[3(\mathbf{x}_k - \mathbf{s})_i (\mathbf{x}_k - \mathbf{s})_j - \delta_{ij} (\mathbf{x}_k - \mathbf{s})^2 \right]$$

Resulting potential/force approximation:

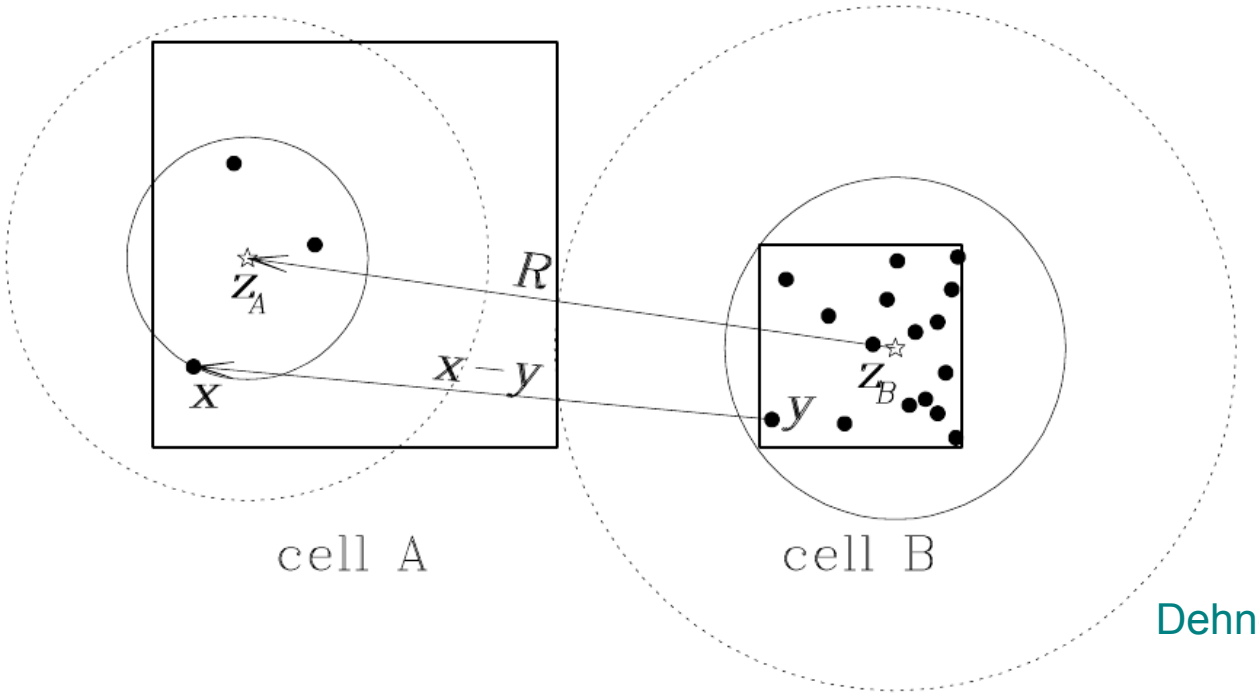
$$\Phi(\mathbf{r}) = -G \left[\frac{M}{|\mathbf{y}|} + \frac{1}{2} \frac{\mathbf{y}^T \mathbf{Q}\, \mathbf{y}}{|\mathbf{y}|^5} \right]$$

For a single force evaluation, not *N* single-particle forces need to be computed, but **only of order *log(N)* multipoles**, depending on the opening angle.

- The tree algorithm has no intrinsic restrictions for its dynamic range
- force accuracy can be conveniently adjusted to desired level
- the speed does depend only very weakly on clustering state
- geometrically flexible, allowing arbitrary geometries

The fast multipole method (FFM) generalizes the tree algorithm and expands the field symmetrically for each pair of interacting cells

Two interacting cells:



Dehnen (2002)

- Very fast
- Manifest momentum conservation

But:

- Doesn't work well with individual timesteps
- Difficult to parallelize for distributed memory machines

TreePM force calculation algorithm

Particularly at high redshift, it is expensive to obtain accurate forces with the tree-algorithm

THE TREE-PM FORCE SPLIT

Periodic peculiar potential

$$\nabla^2\phi(\mathbf{x}) = 4\pi G[\rho(\mathbf{x}) - \overline{\rho}] = 4\pi G \sum_{\mathbf{n}} \sum_i m_i \left[\tilde{\delta}(\mathbf{x} - \mathbf{x}_i - \mathbf{n}L) - \frac{1}{L^3}\right]$$

Idea: Split the potential (of a single particle) in Fourier space into a long-range and a short-range part, and compute them separately with PM and TREE algorithms, respectively.

Poisson equation in Fourier space:

$$\phi_{\mathbf{k}} = -\frac{4\pi G}{\mathbf{k}^2}\rho_{\mathbf{k}} \qquad (\mathbf{k} \neq 0)$$

$$\phi_{\mathbf{k}}^{\mathrm{long}} = \phi_{\mathbf{k}} \exp(-\mathbf{k}^2 r_s^2)$$

Solve with PM-method

- CIC mass assignment
- FFT
- multiply with kernel
- FFT backwards
- Compute force with 4-point finite difference operator
- Interpolate forces to particle positions

$$\phi_{\mathbf{k}}^{\mathrm{short}} = \phi_{\mathbf{k}} \left[1 - \exp(-\mathbf{k}^2 r_s^2)\right]$$

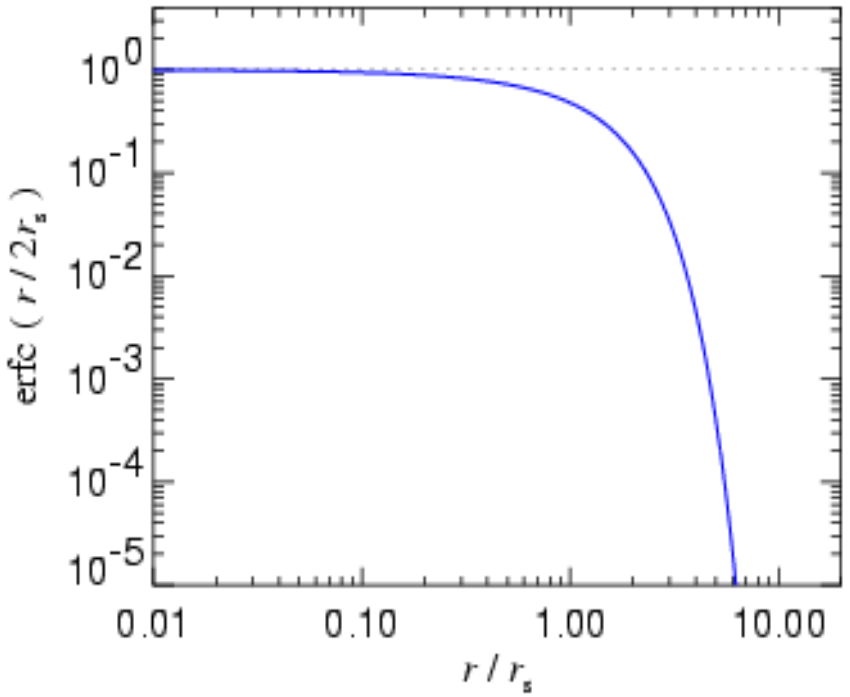
FFT to real space $\quad \phi(r) = -\frac{Gm}{r}\,\mathrm{erfc}\left(\frac{r}{2r_s}\right)$

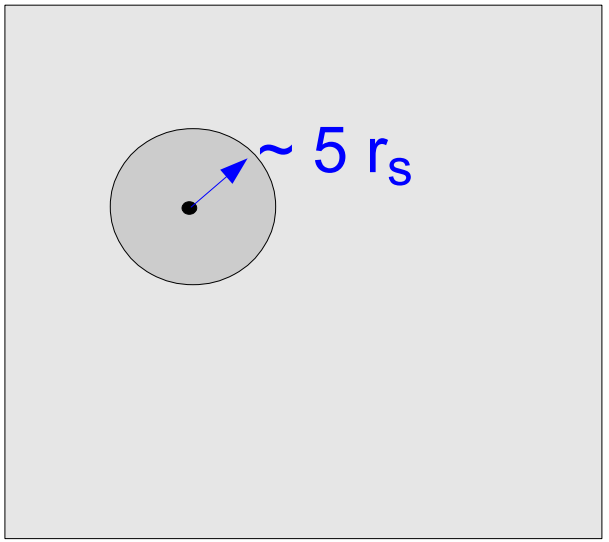
Solve in real space with TREE

In the TreePM algorithm, the tree has to be walked only locally

PERFORMANCE GAIN DUE TO LOCAL TREE WALK

$$\phi(r) = -\frac{Gm}{r}\,\mathrm{erfc}\left(\frac{r}{2r_s}\right)$$





Advantages of TreePM include:

- Accurate and fast long-range force
- No force anisotropy
- Speed is largely insensitive to clustering (as for tree algorithm)
- No Ewald correction necessary for periodic boundary conditions

Using zero-padding and a different Greens-Function, the long-range force can also be computed for vaccuum boundaries using the FFT.
(Implemented in Gadget-2)

Brief digression back to time integration

The force-split can be used to construct a symplectic integrator where long- and short-range forces are treated independently

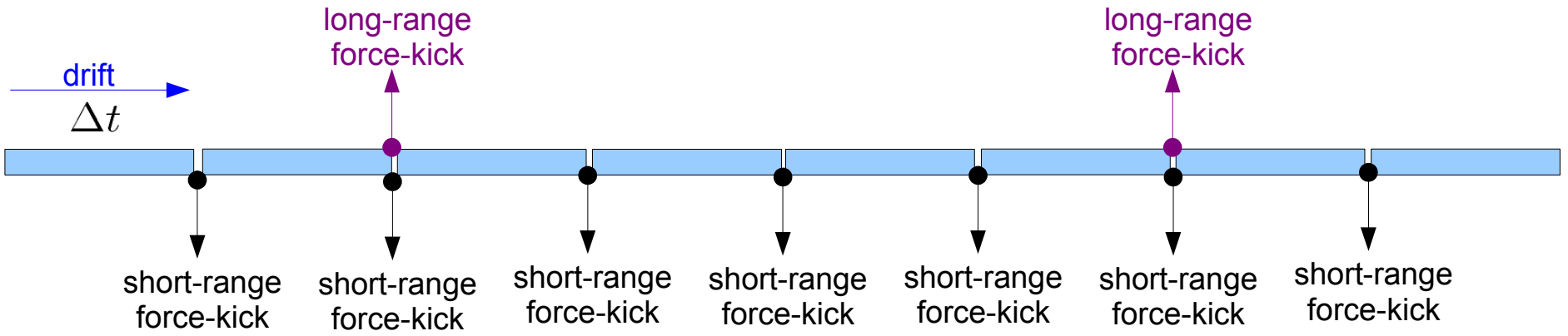
TIME INTEGRATION FOR LONG AND SHORT-RANGE FORCES

Separate the potential into a long-range and a short-range part:

$$H = \sum_i \frac{\mathbf{p}_i^2}{2m_i a(t)^2} + \frac{1}{2}\sum_{ij} \frac{m_i m_j\, \varphi_{\rm sr}(\mathbf{x}_i - \mathbf{x}_j)}{a(t)} + \frac{1}{2}\sum_{ij} \frac{m_i m_j\, \varphi_{\rm lr}(\mathbf{x}_j - \mathbf{x}_j)}{a(t)}$$

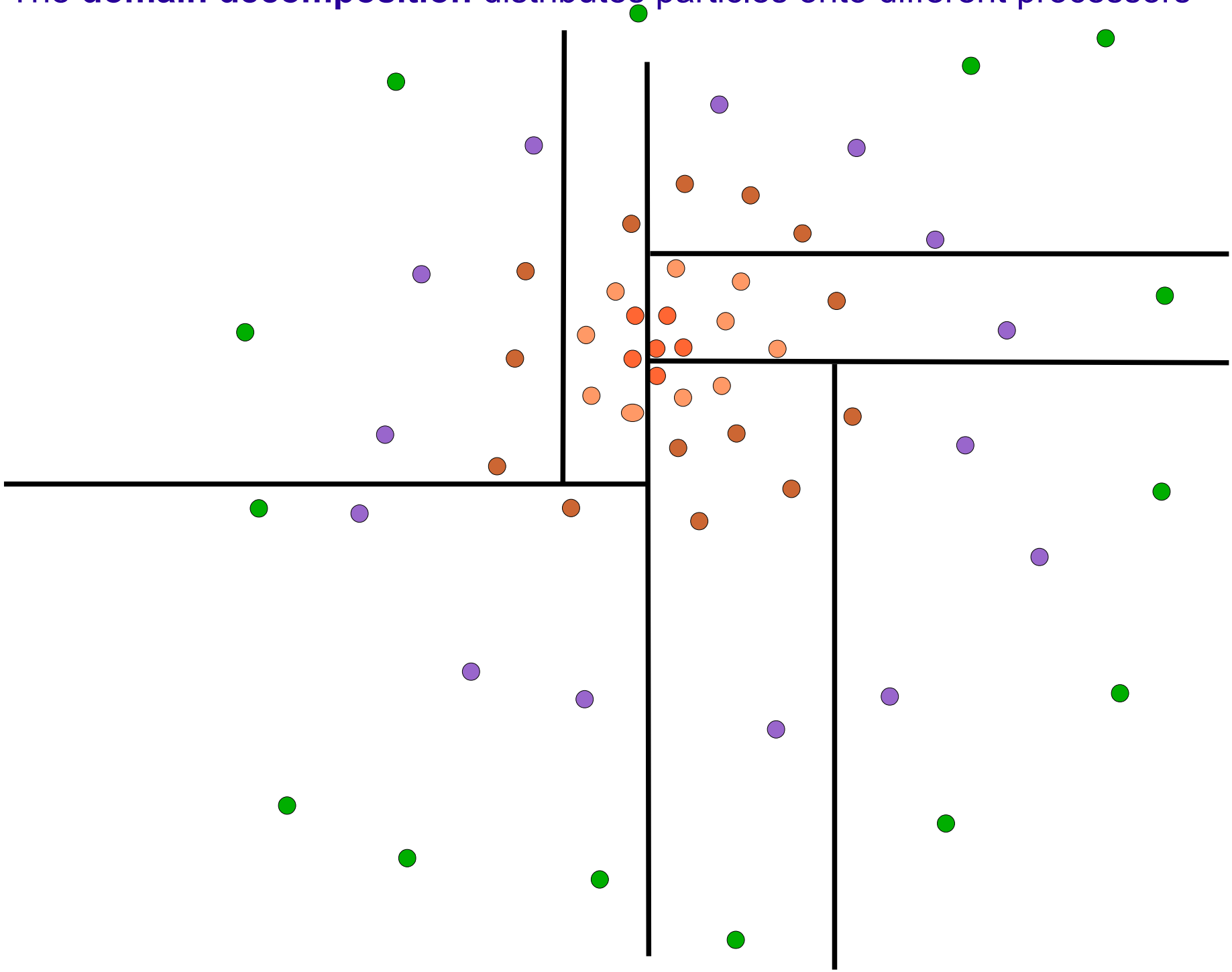
The short-range force can then be evolved in a symplectic way on a smaller timestep than the long range force:

$$\tilde{\mathbf{U}}(\Delta t) = \mathbf{K}_{\rm lr}\left(\frac{\Delta t}{2}\right)\left[\mathbf{K}_{\rm sr}\left(\frac{\Delta t}{2m}\right)\mathbf{D}\left(\frac{\Delta t}{m}\right)\mathbf{K}_{\rm sr}\left(\frac{\Delta t}{2m}\right)\right]^m \mathbf{K}_{\rm lr}\left(\frac{\Delta t}{2}\right)$$



Parallelization:
Domain decomposition

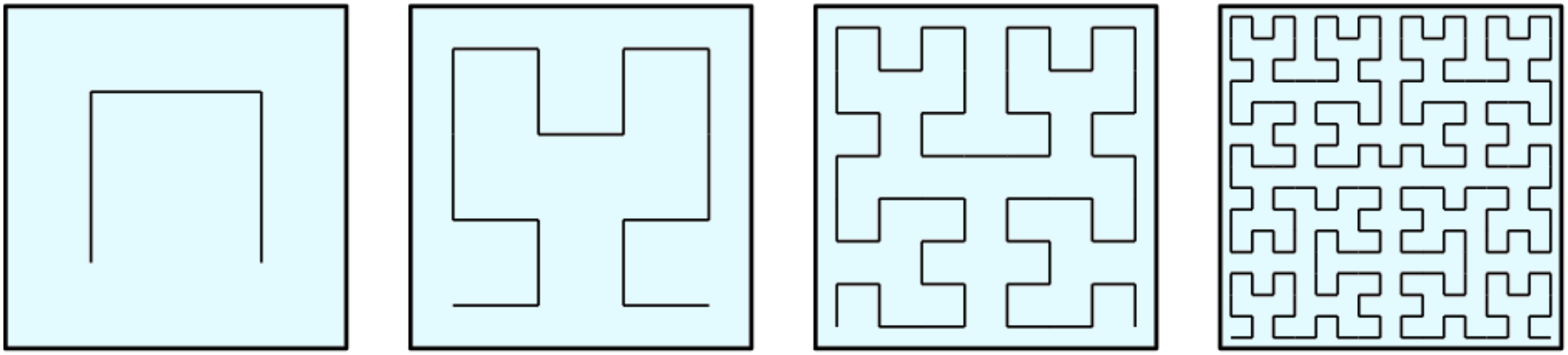
The **domain decomposition** distributes particles onto different processors

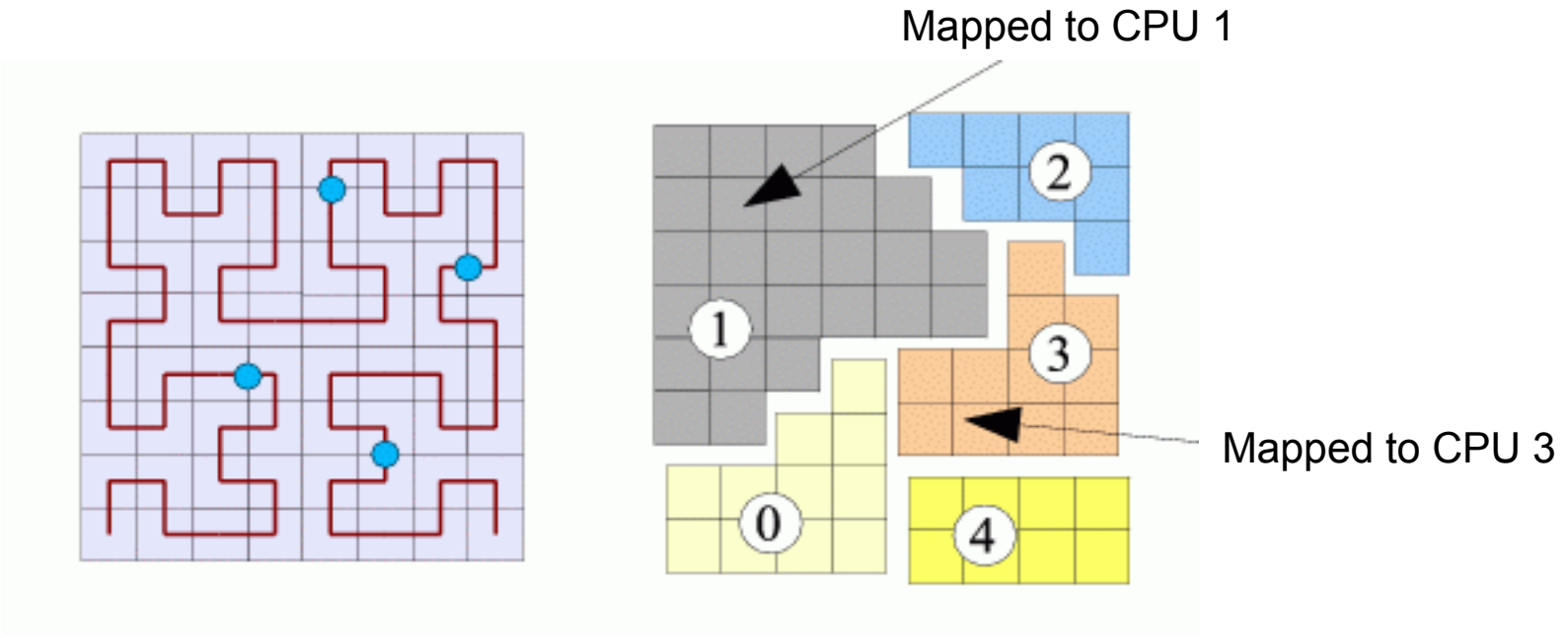


The space-filling Hilbert curve is a fractal that fills the square

CONSTRUCTION OF A FLEXIBLE DOMAIN DECOMPOSITION WITH CACHE BENEFITS

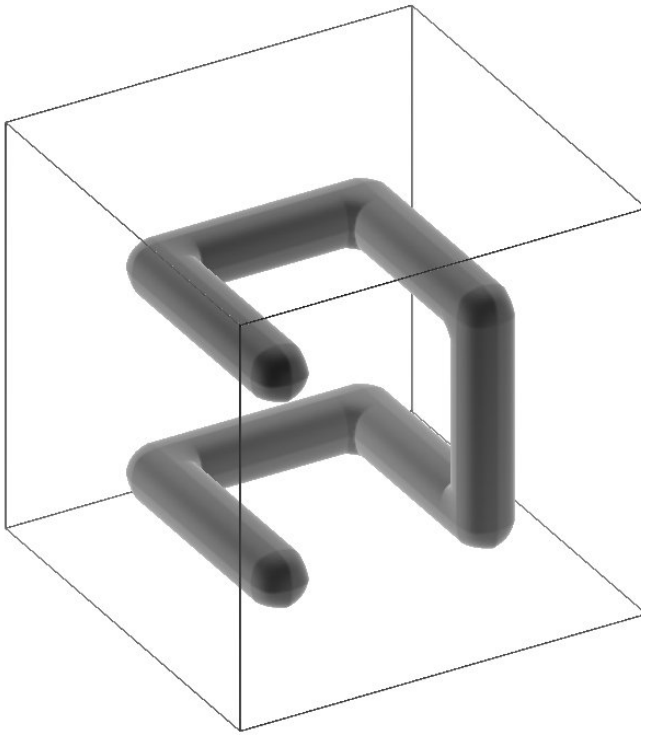
Idea: Order the particles along a space-filling curve

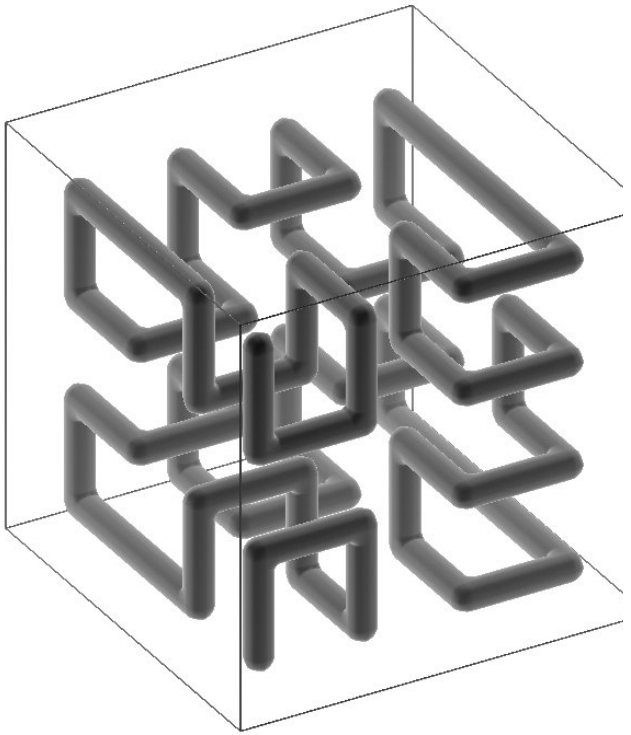


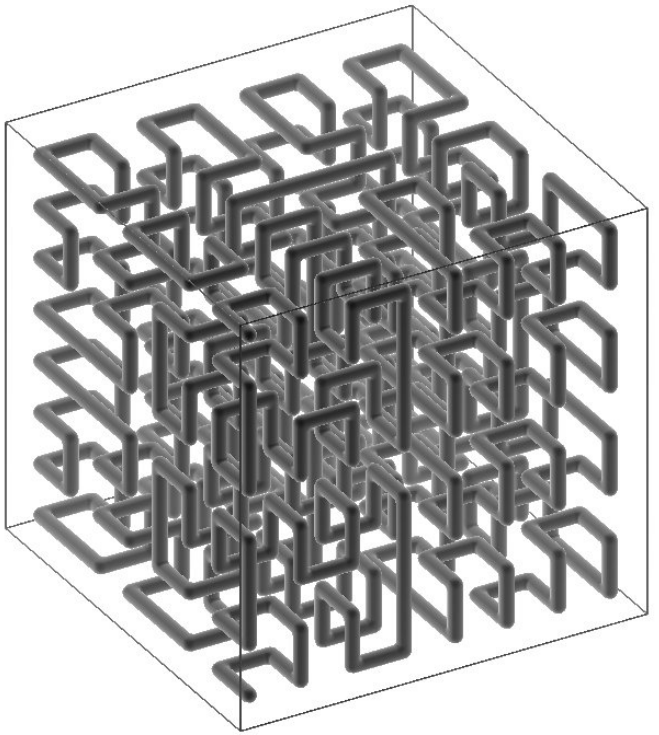


The space-filling Hilbert curve can be readily generalized to 3D

THE PEANO-HILBERT CURVE

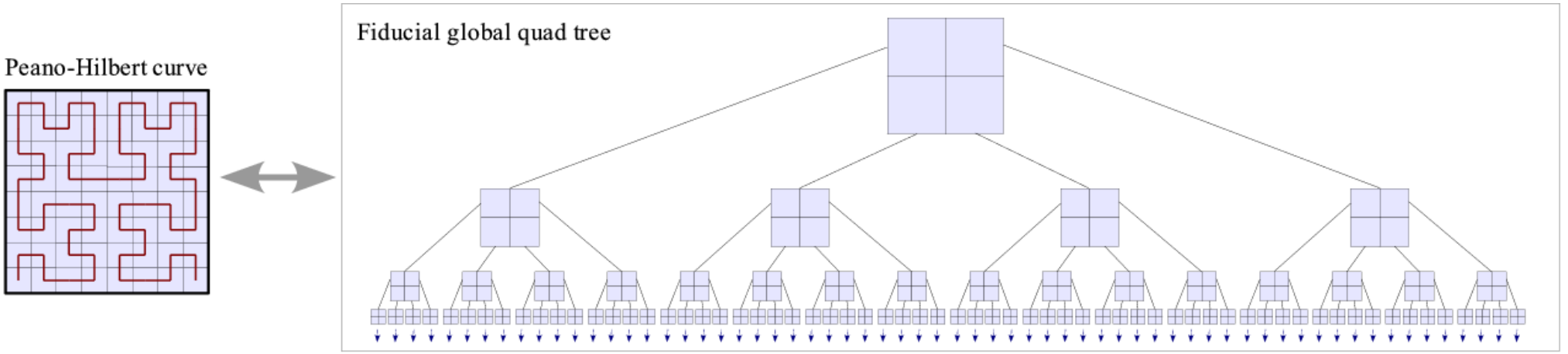


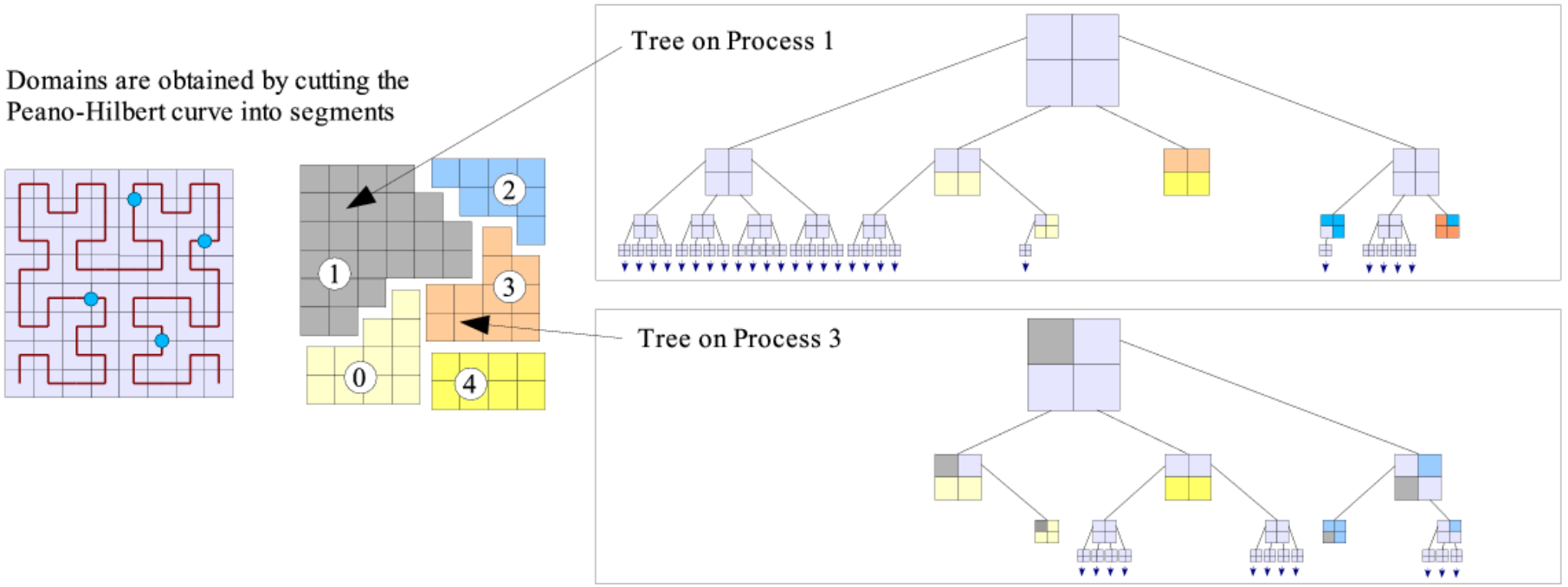




The space-filling Peano-Hilbert is used in GADGET and other codes for the domain-decomposition

SPLITTING UP THE TREE FOR DIFFERENT PROCESSORS





Initial conditions generation

- Prefabricated galaxies / halos / disks
- Cosmological initial conditions

In special cases, the distribution function for static solutions of the CBE can be constructed analytically

An integral of motion $I = I(\mathbf{x}(t), \mathbf{v}(t))$ is constant along orbits, i.e.: $\frac{\mathrm{d}I}{\mathrm{d}t} = 0$

→ Then I is a solution of the CBE.

Jeans theorem: Steady-state solutions of the CBE only depend on integrals of motion.

For a spherical mass distribution, a DF that only depends on energy can be constructed with **Eddington's formula**.

Example:

Hernquist halo:

$$\rho(r) = \frac{M}{2\pi} \frac{a}{r(r+a)^3}$$

$$f(E) = \frac{1}{\sqrt{2}(2\pi)^3 (GMa)^{3/2}} \frac{\sqrt{e}}{(1-e)^2} \left[(1-2e)(8e^2 - 8e - 3) + \frac{3\sin^{-1}(\sqrt{e})}{\sqrt{e(1-e)}} \right]$$

where: $e = -\frac{aE}{GM}$ $\qquad E = \frac{\mathbf{v}^2}{2} + \Phi$

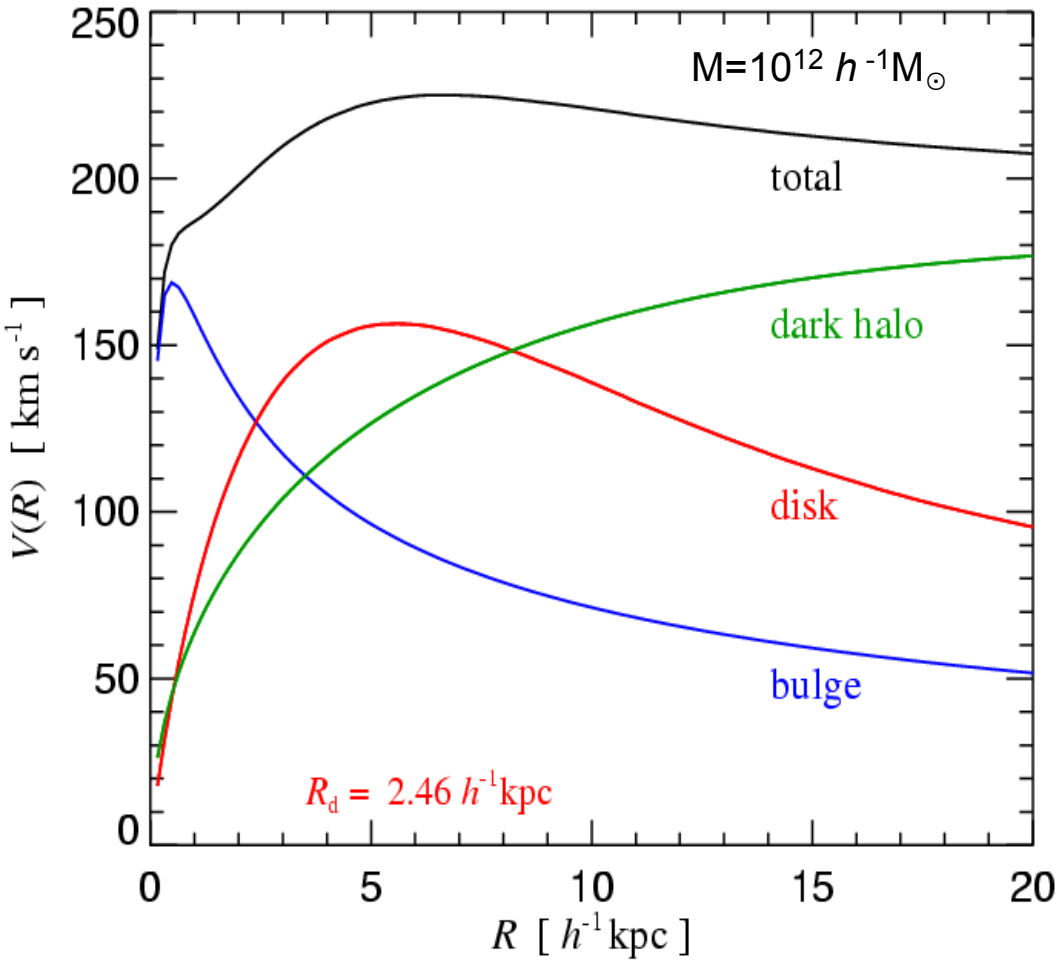
Construction of compound disk galaxies that are in dynamical equilibrium

STRUCTURAL PROPERTIES OF MODEL GALAXIES

Components:

- Dark halo (Hernquist profile matched to NFW halo)
- Stellar disk (exponential)
- Stellar bulge
- Gaseous disk (exponential)
- Central supermassive black hole

One approach: Compute the exact gravitational potential for the axisymmetric mass distribution and solve the **Jeans equations**



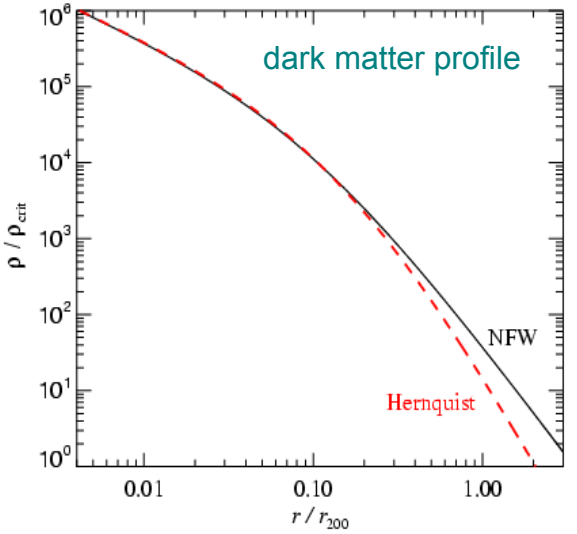
The first step in constructing an isolated galaxy model is the specification of the density structure of all mass components

DENSITY DISTRIBUTIONS OF DARK MATTER AND STARS IN BULGE AND DISK

Dark matter:

$$\rho_{\rm dm}(r) = \frac{M_{\rm dm}}{2\pi}\,\frac{a}{r(r+a)^3}$$

Hernquist or NFW profile



Stars in the disk:

$$\Sigma_\star(r) = \frac{M_\star}{2\pi h^2}\,\exp(-r/h)$$

"Isothermal sheet" with exponential profile

$$\rho_\star(R,z) = \frac{M_\star}{4\pi z_0\, h^2}\,{\rm sech}^2\left(\frac{z}{2\,z_0}\right)\exp\left(-\frac{R}{h}\right)$$

Disk scale length ***h*** determined by spin parameter of halo.

Stars in the bulge:

$$\rho_{\rm b}(r) = \frac{M_{\rm b}}{2\pi}\,\frac{b}{r(r+b)^3}$$

Bulge scale length ***b*** can be set to a fraction of the disk scale-length ***h.***

Gas in the disk:

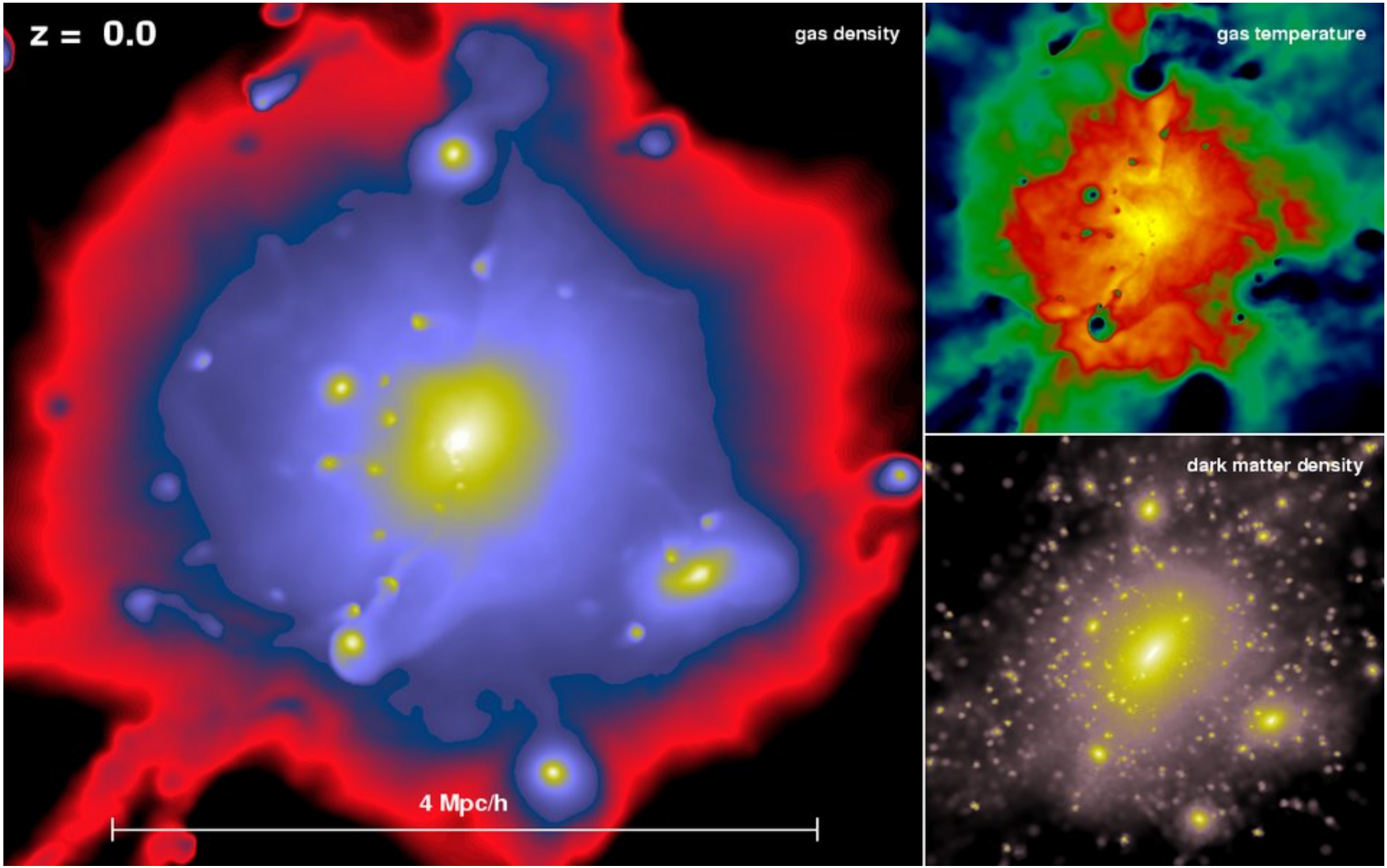
$$\Sigma_{\rm gas}(r) = \frac{M_{\rm gas}}{2\pi h^2}\,\exp(-r/h)$$

Vertical structure given by hydrostatic equilibrium.
Depends on the equation of state of the gas.

$$-\frac{1}{\rho_{\rm g}}\frac{\partial P}{\partial z} - \frac{\partial \Phi}{\partial z} = 0$$

Non-radiative gas shows markedly different behavior from dark matter once pressure forces become important

A SIMULATED CLUSTER WITH GAS



The Euler equations for inviscid gas dynamics

LAGRANGIAN FORM

Equation of motion:

$$\frac{\mathrm{d}\mathbf{v}}{\mathrm{d}t} = -\frac{\nabla P}{\rho}$$

Convective derivative

$$\frac{\mathrm{d}}{\mathrm{d}t} = \frac{\partial}{\partial t} + \mathbf{v}\cdot\nabla$$

Continuity equation:

$$\frac{\mathrm{d}\rho}{\mathrm{d}t} + \rho\,\nabla\cdot\mathbf{v} = 0$$

Thermal energy equation:

$$\frac{\mathrm{d}u}{\mathrm{d}t} = -\frac{P}{\rho}\nabla\cdot\mathbf{v}$$

Equation of state:

$$P = (\gamma - 1)\rho u \qquad \gamma = \frac{c_P}{c_v}$$

For mono-atomic gas: $\gamma = \frac{5}{3}$ $\mu u = \frac{3}{2}k_B T$

Entropy equation:

$$\frac{\mathrm{d}A}{\mathrm{d}t} = 0 \qquad A \equiv \frac{P}{\rho^\gamma}$$

The Euler equations for inviscid gas dynamics

EULERIAN FORM

Mass conservation:

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

Momentum conservation:

$$\frac{\partial}{\partial t}(\rho \mathbf{v}) + \nabla(\rho \mathbf{v}\mathbf{v}^T + P) = 0$$

Energy conservation:

$$\frac{\partial}{\partial t}(\rho e) + \nabla[(\rho e + P)\mathbf{v}] = 0$$

Total specific energy:

$$e = \frac{1}{2}\mathbf{v}^2 + u$$

The Navier-Stokes equations for viscous fluids

FLOW WITH FINITE VISCOSITY

mass conservation: $$\frac{\partial \rho}{\partial t} + \nabla(\rho \mathbf{v}) = 0$$

momentum conservation: $$\frac{\partial}{\partial t}(\rho \mathbf{v}) + \nabla(\rho \mathbf{v}\mathbf{v}^T + P) = \nabla \mathbf{\Pi}$$

energy conservation: $$\frac{\partial}{\partial t}(\rho e) + \nabla[(\rho e + P)\mathbf{v}] = \nabla(\mathbf{\Pi}\mathbf{v})$$

Viscous stress tensor:

$$\mathbf{\Pi} = \eta \left[\nabla \mathbf{v} + (\nabla \mathbf{v})^T - \frac{2}{3}(\nabla \cdot \mathbf{v})\mathbf{1} \right] + \xi(\nabla \cdot \mathbf{v})\mathbf{1}$$

Special case: incompressible flow

Kinematic viscosity: $$\nu \equiv \frac{\eta}{\rho}$$

$$\frac{D\mathbf{v}}{Dt} = -\frac{\nabla P}{\rho} + \nu \nabla^2 \mathbf{v}$$

The equations of ideal magneto-hydrodynamics (MHD)

FLOW OF PERFECTLY CONDUCTING MEDIA

$\sigma = \infty$ from Ohm's law: $\mathbf{j} = \sigma\left(\mathbf{E} + \frac{\mathbf{v}\times\mathbf{B}}{c}\right) \longrightarrow \mathbf{E} = -\frac{\mathbf{v}\times\mathbf{B}}{c}$

only magnetic field left in Maxwell's equations:

Induction equation: $$\frac{\partial \mathbf{B}}{\partial t} + \nabla \times (\mathbf{B} \times \mathbf{v}) = 0$$

Divergence constraint: $$\nabla \cdot \mathbf{B} = 0$$

changes in the Euler equations:

$$P_{\text{tot}} = P_{\text{gas}} + \frac{1}{2}\mathbf{B}^2 \qquad \frac{\partial}{\partial t}(\rho\mathbf{v}) + \nabla(\rho\mathbf{v}\mathbf{v}^T - \mathbf{B}\mathbf{B}^T + P) = 0$$

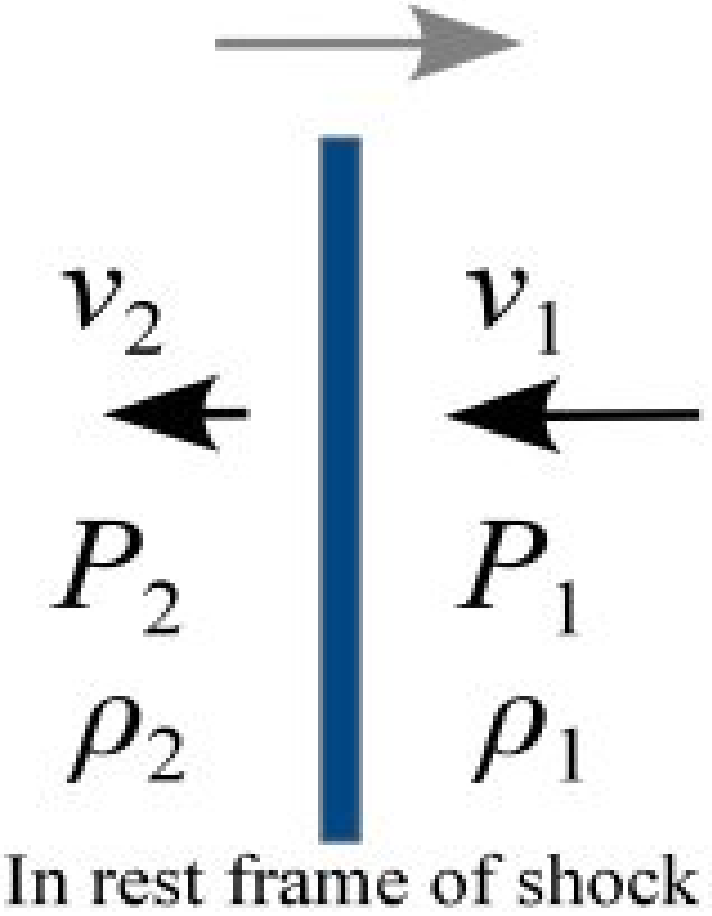
$$e = u + \frac{1}{2}\mathbf{v}^2 + \frac{1}{2}\frac{\mathbf{B}^2}{\rho} \qquad \frac{\partial}{\partial t}(\rho e) + \nabla[(\rho e + P)\mathbf{v} - \mathbf{B}(\mathbf{B}\cdot\mathbf{v}] = 0$$

Shocks

The flow of an ideal gas can develop discontinuities, e.g. by sound wave steepening or by supersonically converging flows.

At such discontinuities, the entropy is no longer preserved.

The generation of the entropy happens in a thin shock layer where the viscosity can not be neglected any more.



Mach number: $\mathcal{M} = \frac{v_1}{c_1}$

The continuity of mass, energy and momentum flux across a shock relates the upstream and downstream flows.

Rankine-Huigonot jump conditions

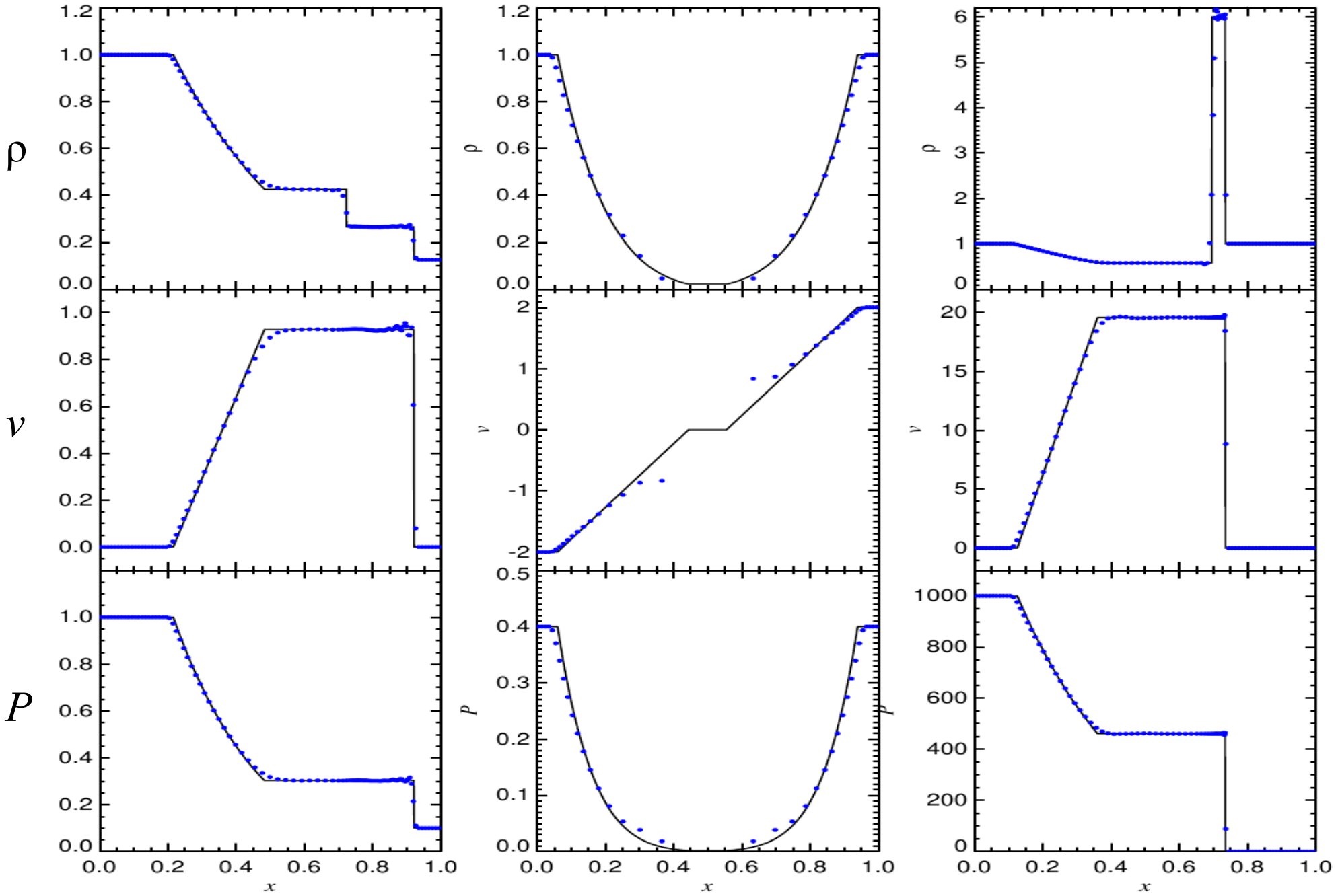
$$\frac{\rho_1}{\rho_2} = \frac{v_2}{v_1} = \frac{\gamma - 1}{\gamma + 1} + \frac{2}{(\gamma + 1)\mathcal{M}^2}$$

$$\frac{P_2}{P_1} = \frac{2\gamma\mathcal{M}^2}{\gamma + 1} - \frac{\gamma - 1}{\gamma + 1}$$

Note: density and velocity jumps are bound,
while the pressure and temperature jumps can be arbitrarily large.

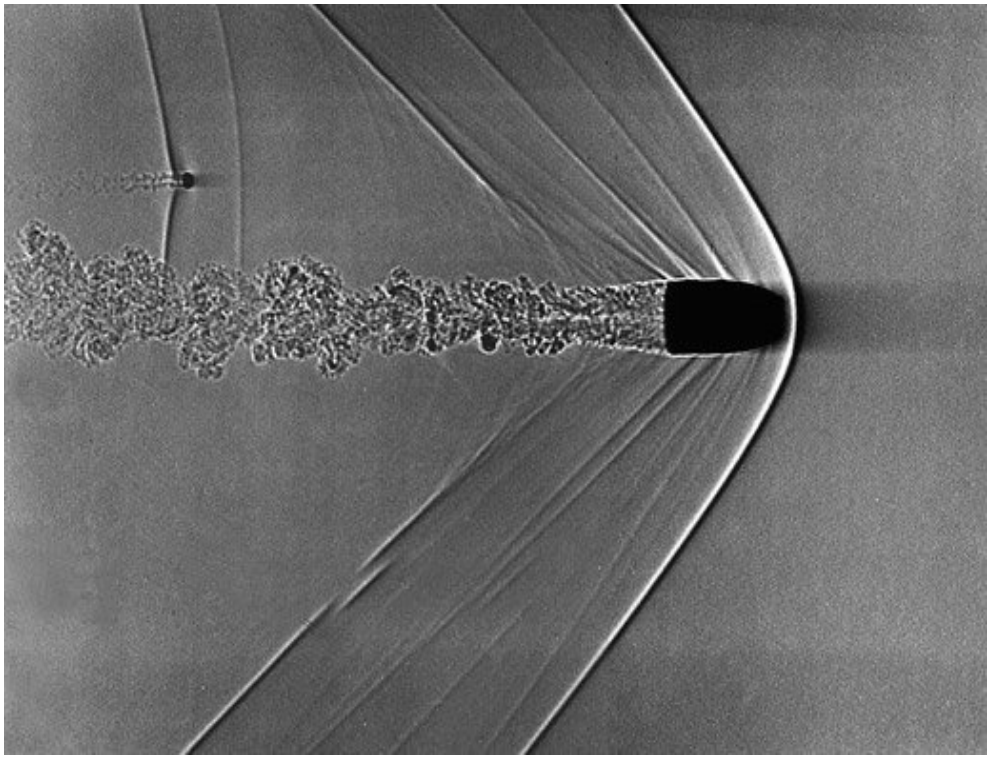
Sod-Shock tubes are simple test problems for shocks and rarefaction waves

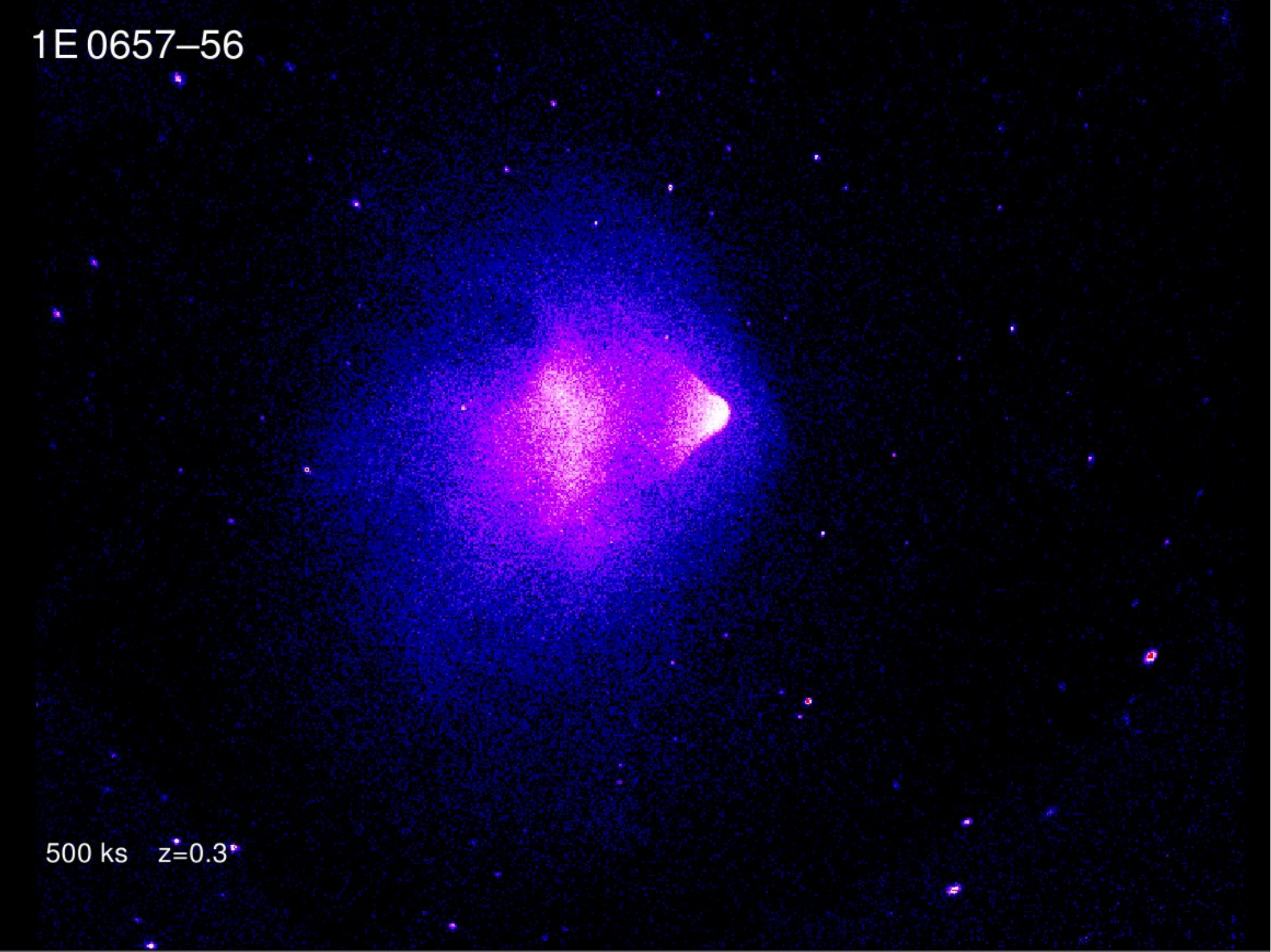
TWO SHOCK PROBLEMS AND A STRONG RAREFACTION



Supersonic motion creates shock waves

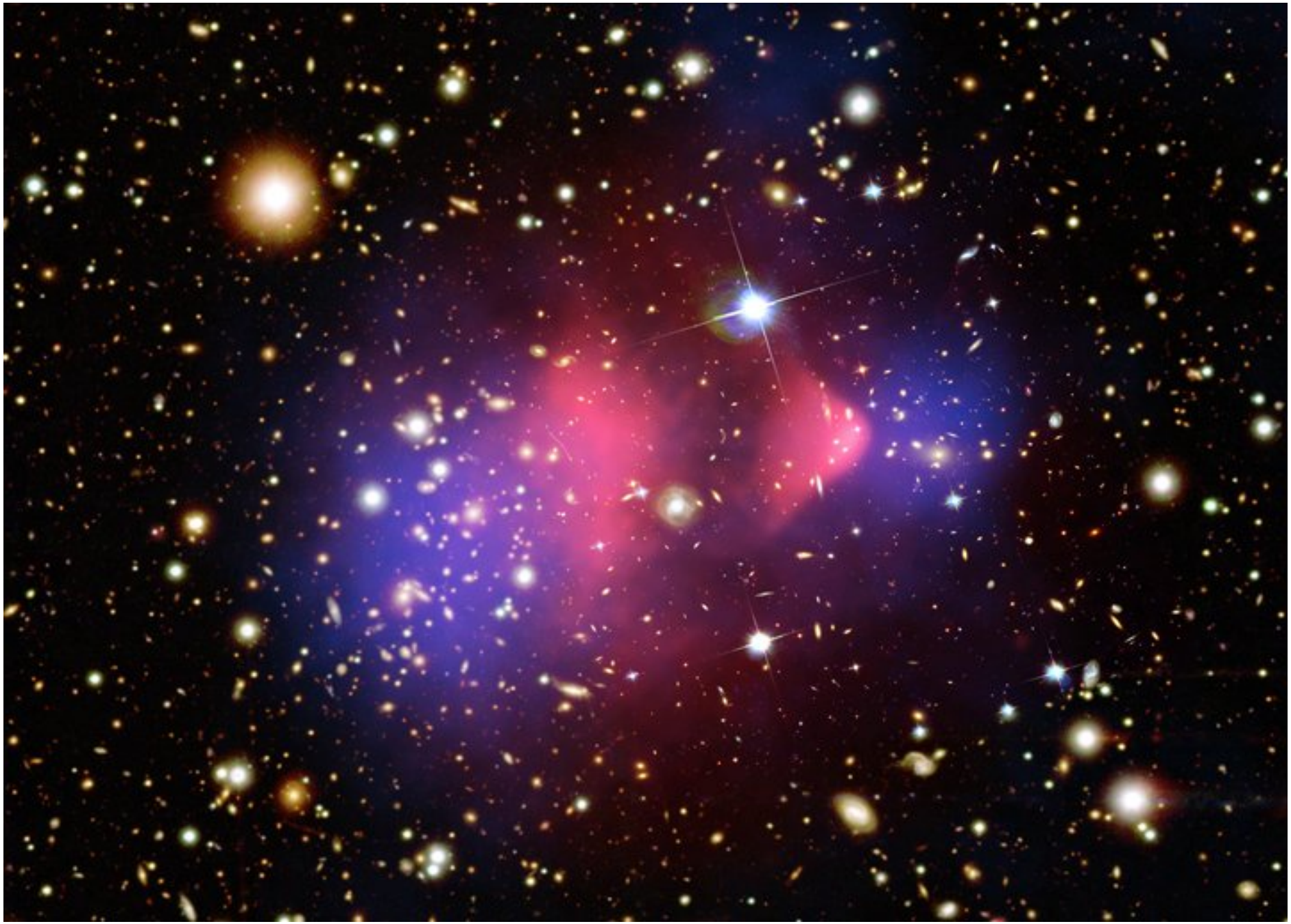
SHOCK WAVES OF A BULLET TRAVELLING IN AIR




1E 0657–56
500 ks
z=0.3

NASA Press Release Aug 21, 2006:

1E 0657-56: NASA Finds Direct Proof of Dark Matter

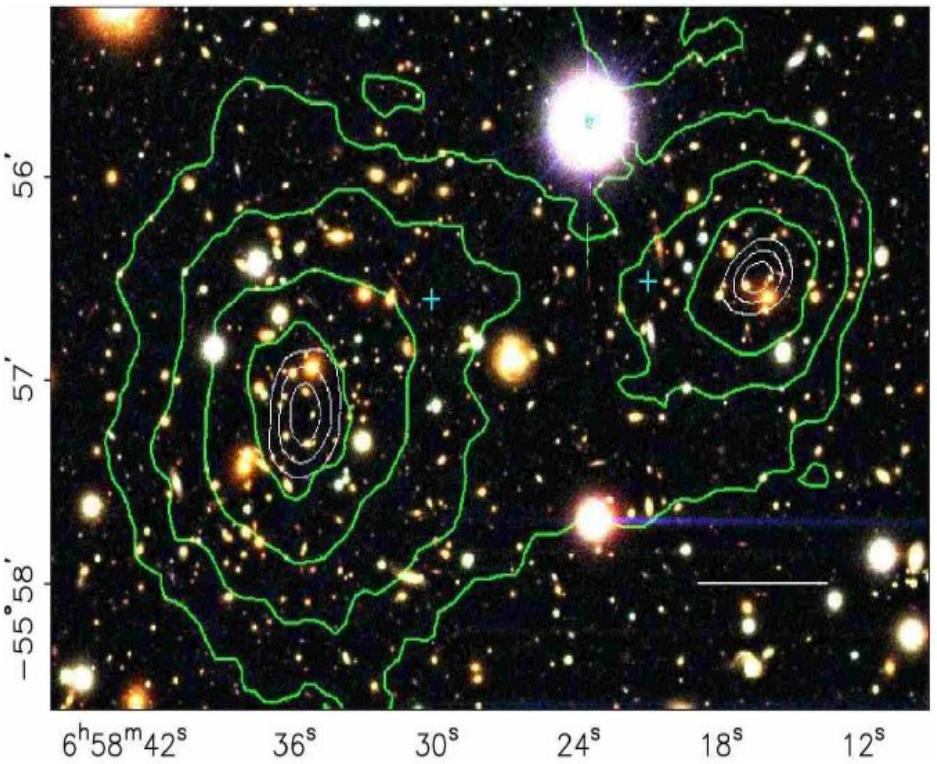


New weak lensing mass reconstructions have confirmed an offset between mass peaks and X-ray emission

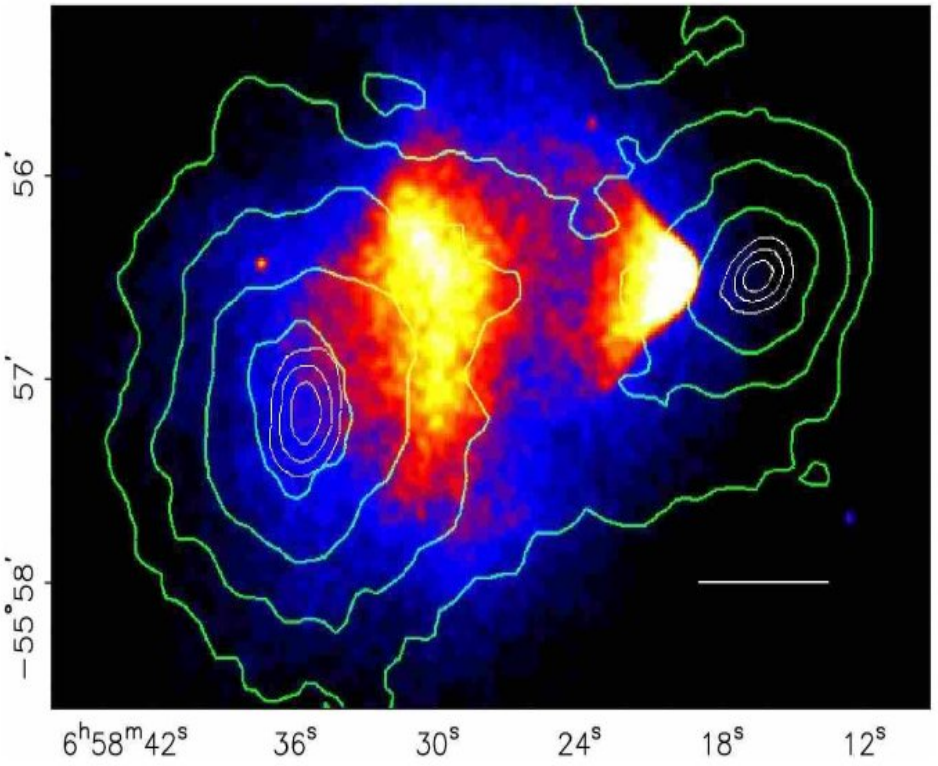
MASS CONTOURS FROM LENSING COMPARD TO X-RAY EMISSION

Clowe et al. (2006)

Magellan Optical Image



500 ksec Chandra exposure

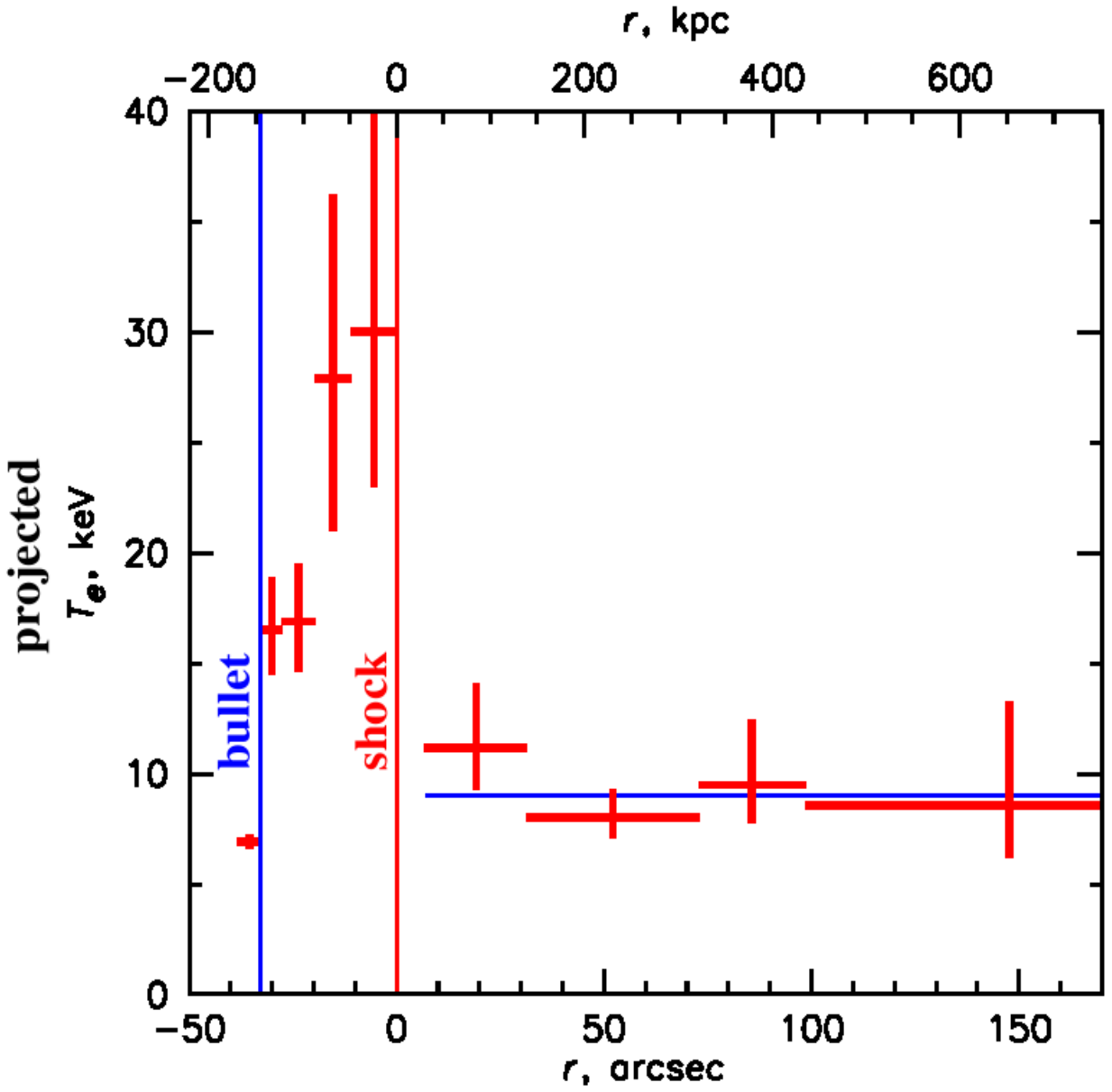


weak lensing mass contours overlaid

The temperature profile through the nose of the shock shows a strong shock and a cold front

X-RAY TEMPERATURE PROFILE FROM CHANDRA OBSERVATIONS

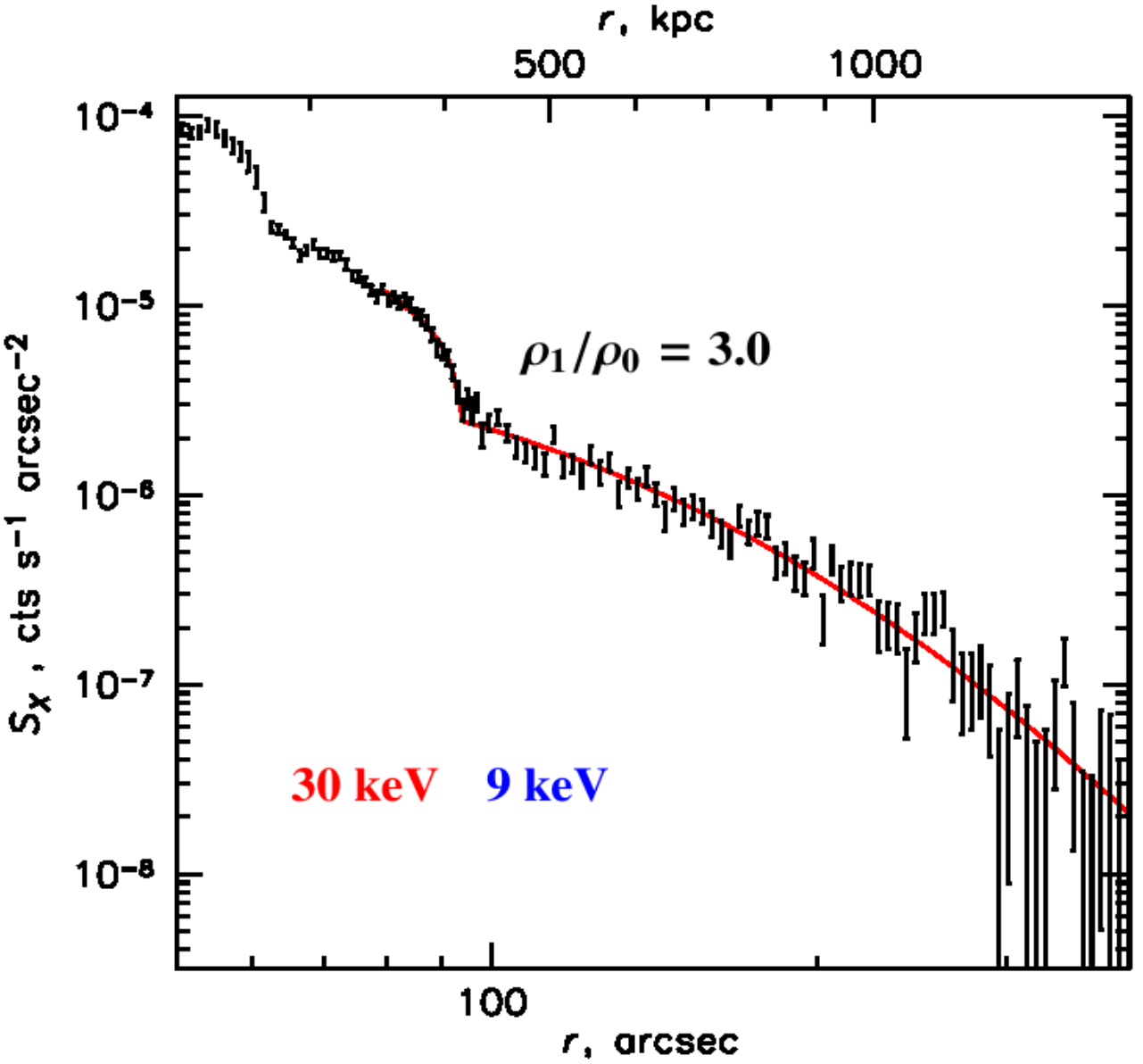
Markevitch et al. (2006)



Fitting the density jump in the X-ray surface brightness profile allows a measurement of the shock's Mach number

X-RAY SURFACE BRIGHTNESS PROFILE

Markevitch et al. (2006)



shock strength:

M = 3.0 ± 0.4

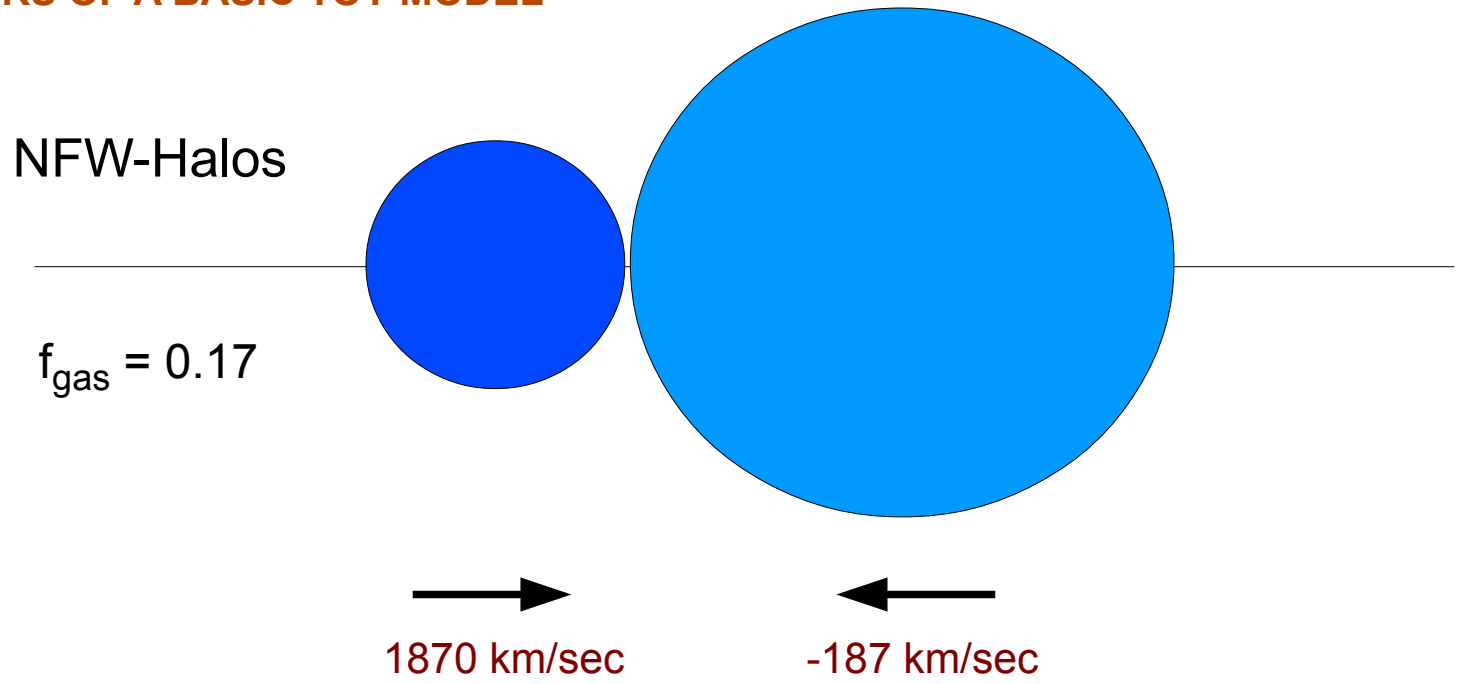
shock velocity:

v_s = 4700 km/s

Usually, shock velocity has been identified with velocity of the bullet.

A simple toy merger model of two NFW halos on a zero-energy collision orbit

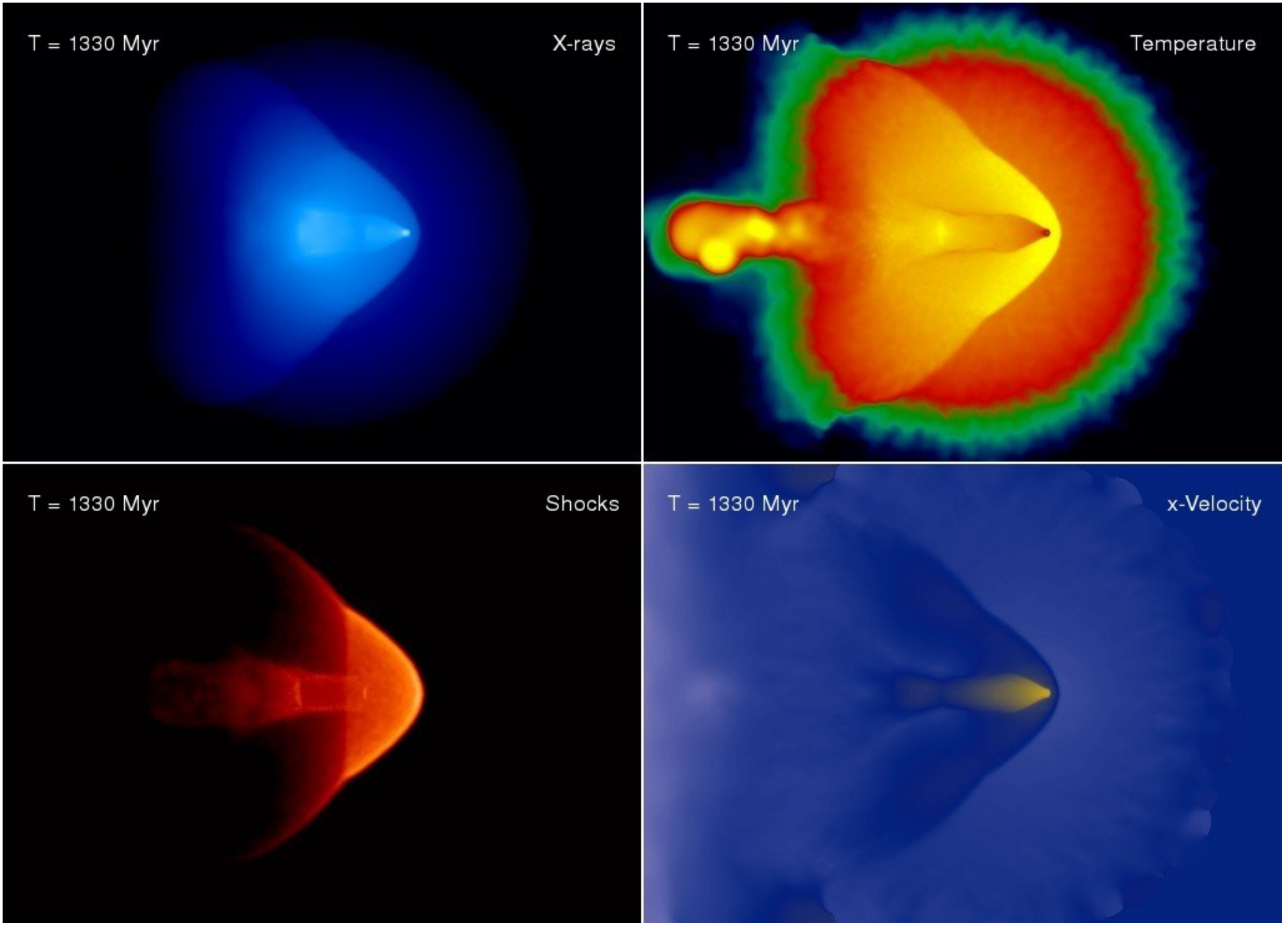
PARAMETERS OF A BASIC TOY MODEL



Mass model from **Clowe et al. (2006):**

M_{200} = 1.5 x 10^{14} $M_\odot$	**M_{200} = 1.5 x 10^{15} $M_\odot$**
R_{200} = 1.1 Mpc	R_{200} = 2.3 Mpc
c = 7.2	c = 2.0
V_{200} = 780 km/sec	V_{200} = 1680 km/sec

VIDEO OF THE TIME EVOLUTION OF A SIMPLE BULLET CLUSTER MODEL

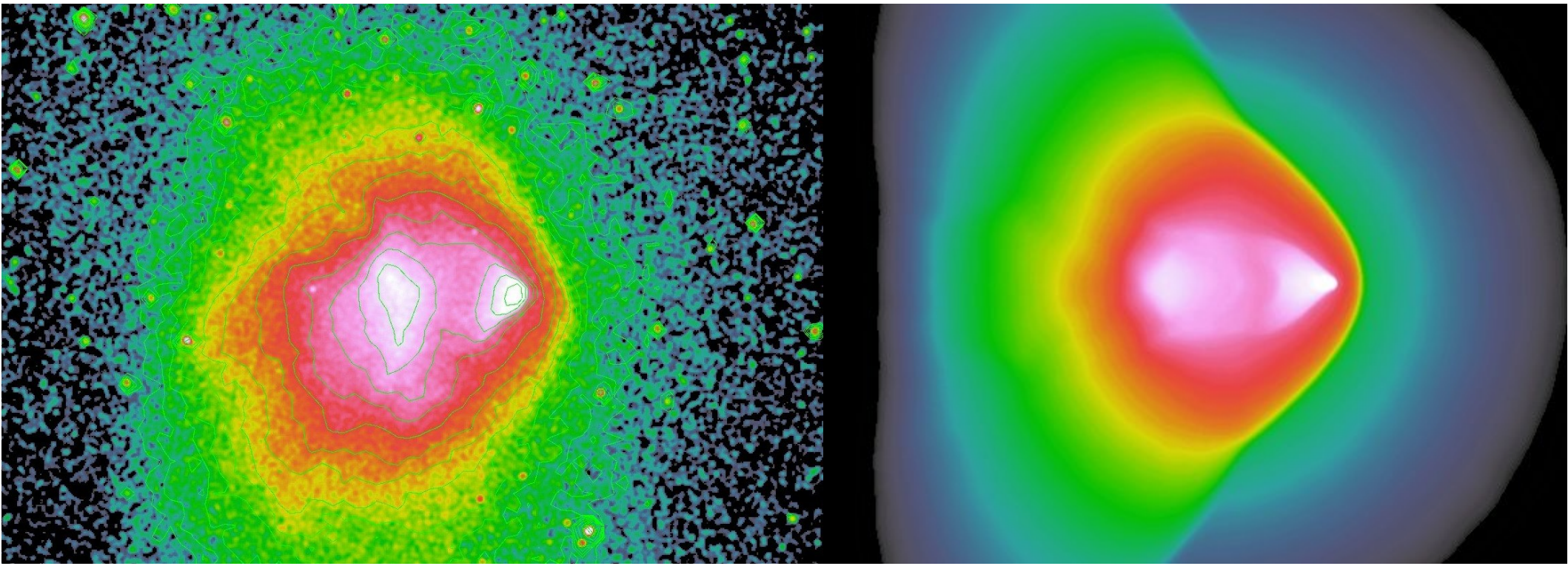


Drawing the observed X-ray map and the simulation images with the same color-scale simplifies the comparison

SIMULATED X-RAY MAP COMPARED TO OBSERVATION

Candra 500 ks image

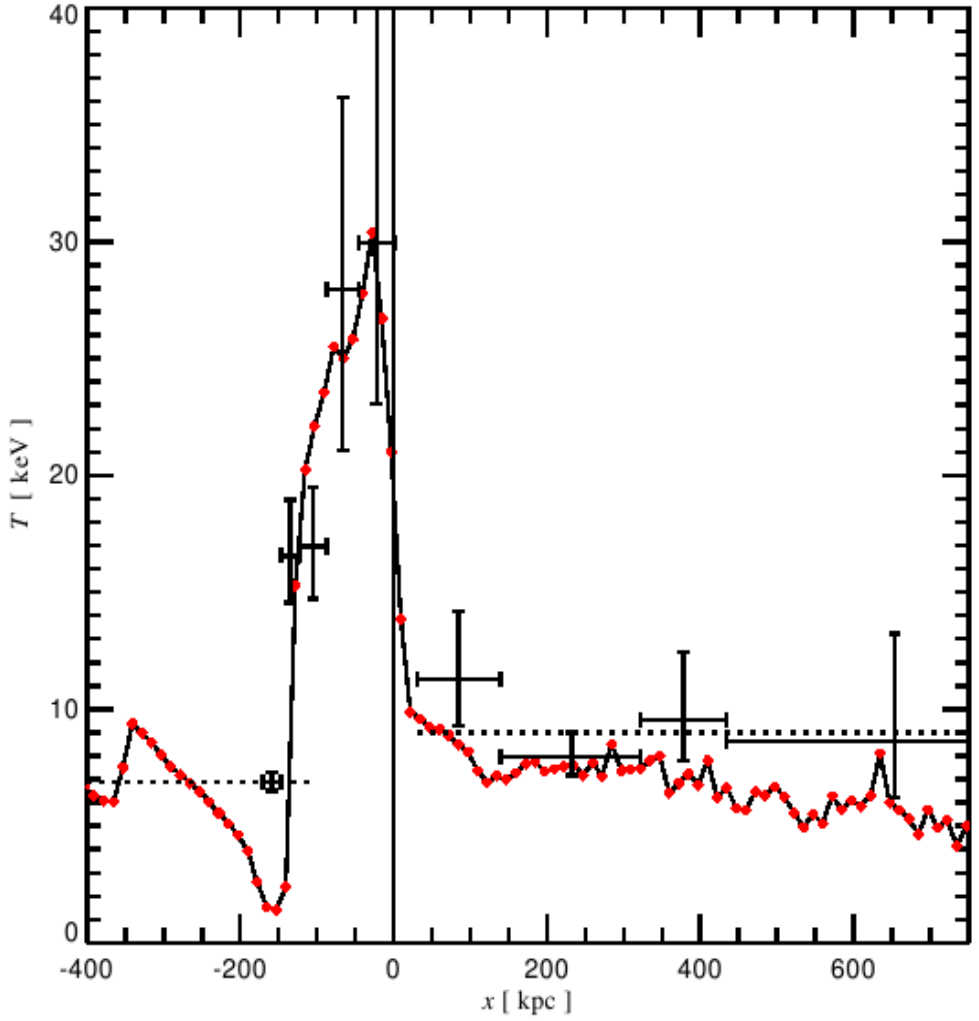
bullet cluster simulation



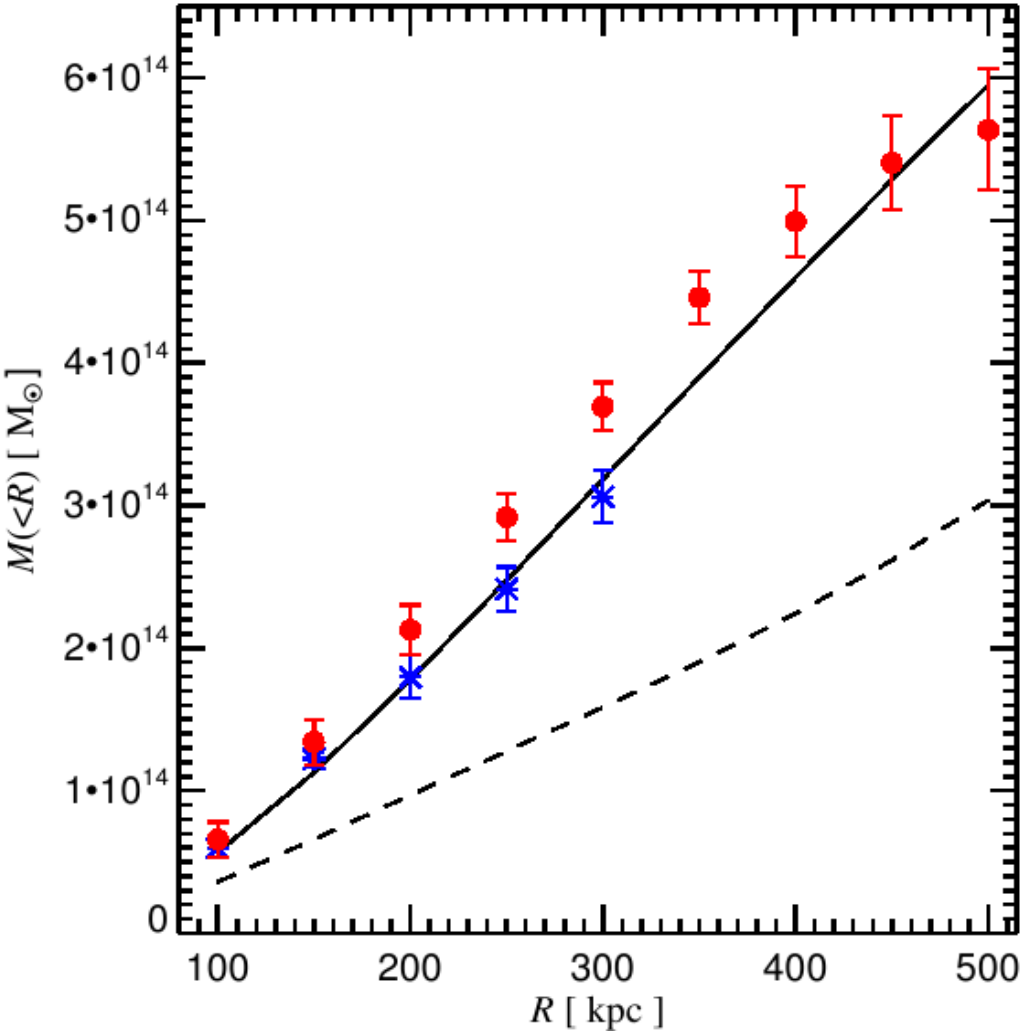
Springel & Farrar (2007)

The model also matches the observed temperature and mass profiles

COMPARISON OF SIMULATED TEMPERATURE AND MASS PROFILE WITH OBSERVATIONS



Data from **Markevitch et al. (2006)**



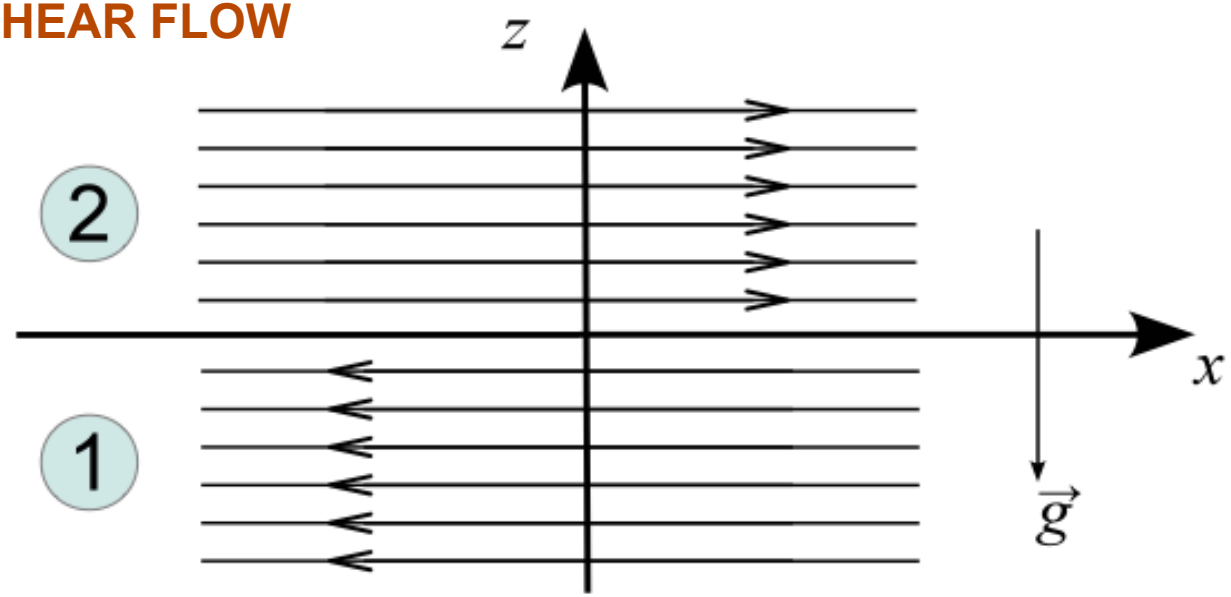
Data from **Bradac et al. (2006)**

In multidimensional hydrodynamics, interesting fluid instabilities can arise which can make a flow prone to develop turbulence

INSTABILITIES IN PLANAR SHEAR FLOW

$\rho_2, \quad U_2$

$\rho_1, \quad U_1$



Put in a wave-like perturbation at the interface and examine linear growth (eigemode analysis)

$$\phi_1 = \phi_1(z)\exp[i(kx - \omega t)]$$

Dispersion relation:

$$\omega^2(\rho_1 + \rho 2) - 2\omega k(\rho_1 U_1 + \rho_2 U_2) + k^2(\rho_1 U_1^2 + \rho_2 U_2^2) + (\rho_2 - \rho 1)kg = 0$$

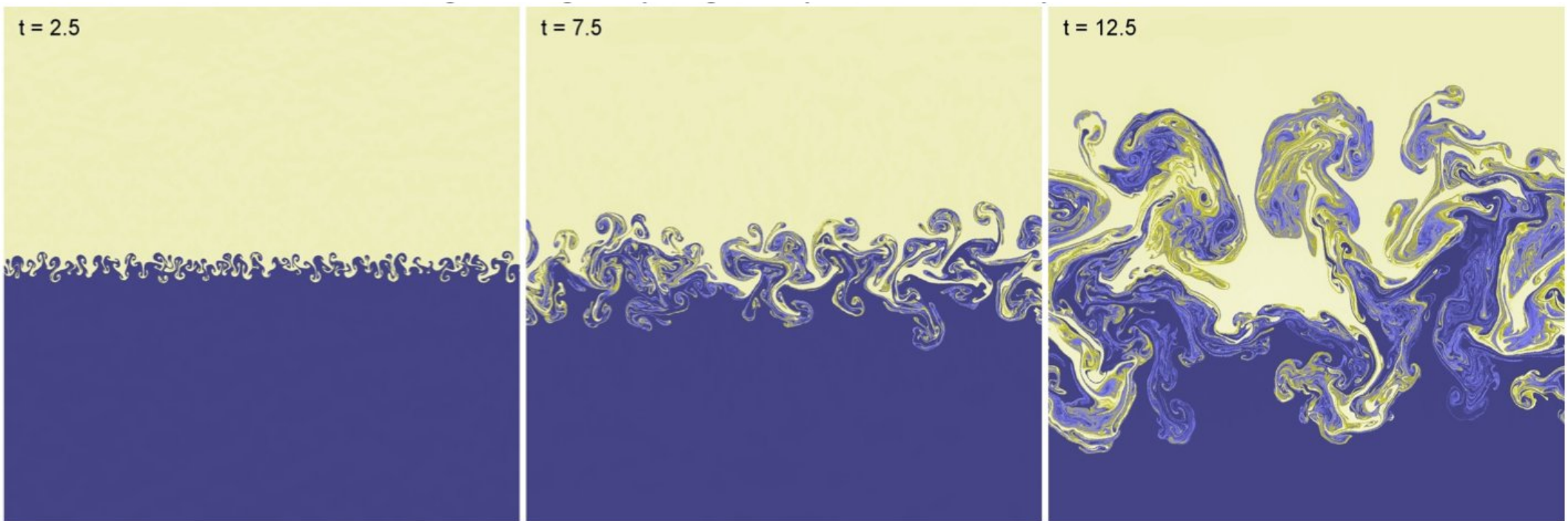
Rayleigh-Taylor instability

If there is no shear flow initially and only buoyancy, we get the RT instability

$$U_1 = U_2 = 0.$$

$$\omega^2 = \frac{(\rho_1 - \rho_2)kg}{\rho_1 + \rho_2}$$

Flow is stable if the lighter fluids is on top.



Kelvin-Helmholtz Instability

If there is no gravitational field, the shear flow is always unstable

for $g = 0$

$$\omega_{1/2} = \frac{k(\rho_1 U_1 + \rho_2 U_2)}{\rho_1 + \rho_2} \pm i \frac{\sqrt{\rho_1 \rho_2}}{\rho_1 + \rho_2} |U_1 - U_2|$$

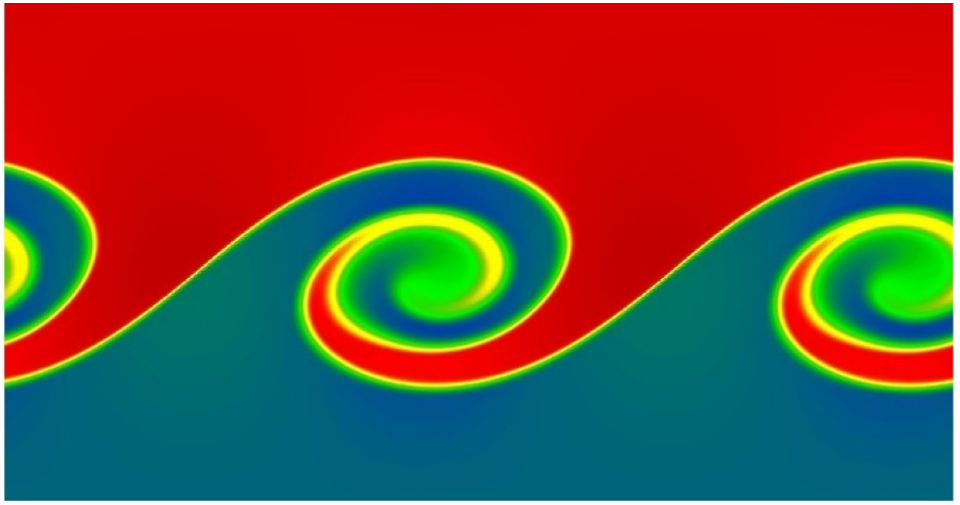
for $g > 0$

$$\omega = \frac{k(\rho_1 U_1 + \rho_2 U_2)}{\rho_1 + \rho_2} \pm \frac{\sqrt{-k^2 \rho_1 \rho_2 (U_1 - U_2)^2 - (\rho_1 + \rho_2)(\rho_2 - \rho_1) k g}}{\rho_1 + \rho_2}$$

Flow is stable if:

- Lighter fluid is on top
- Velocity difference is small enough

$$(U_1 - U_2)^2 < \frac{(\rho_1 + \rho_2)(\rho_1 + \rho_2) g}{k \rho_1 \rho_2}$$



Kelvin-Helmholtz instabilities seen in Nature

Solving the Jeans equations allows the construction of dynamically stable disk galaxy models

MOMENT EQUATIONS FOR THE VELOCITY STRUCTURE

We assume that the **velocity distribution function** of dark matter and stars can be approximated everywhere by a **triaxial Gaussian.**

Further, we assume axisymmetry, and that the distribution function depends only on E and L_z

Then cross-moments vanish:
$$\langle v_R v_z \rangle = \langle v_z v_\phi \rangle = \langle v_R v_\phi \rangle = 0$$
$$\langle v_R \rangle = \langle v_z \rangle = 0$$

The radial and vertical moments are given by:

$$\left\langle v_z^2 \right\rangle = \left\langle v_R^2 \right\rangle = \frac{1}{\rho} \int_z^\infty \rho(z', R) \frac{\partial \Phi}{\partial z'} \, \mathrm{d}z'$$

The azimuthal dispersion fulfills a separate equation:

$$\left\langle v_\phi^2 \right\rangle = \left\langle v_R^2 \right\rangle + \frac{R}{\rho} \frac{\partial \left(\rho \left\langle v_R^2 \right\rangle \right)}{\partial R} + v_c^2$$

Circular velocity: $v_c^2 \equiv R \frac{\partial \Phi}{\partial R}$

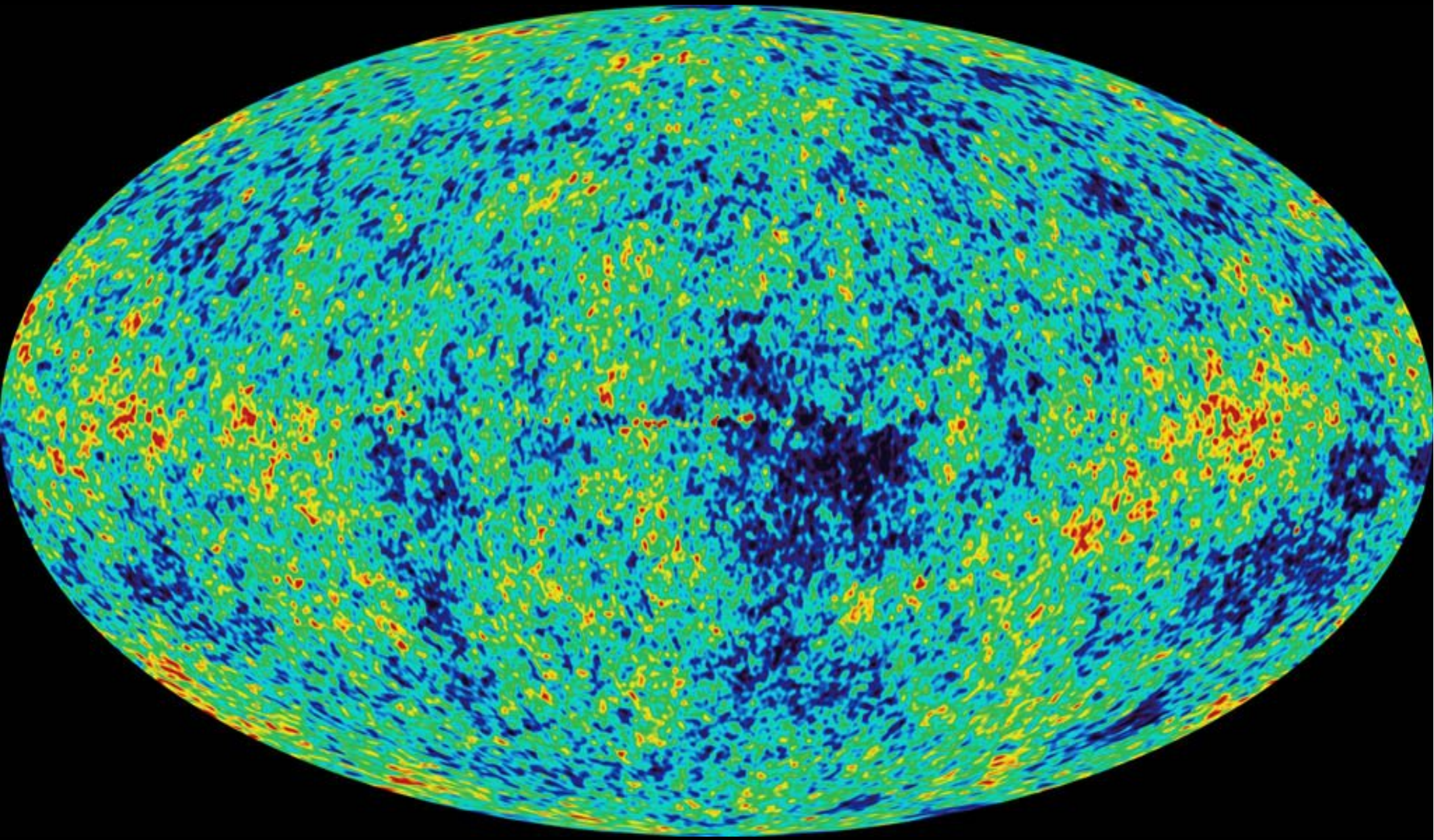
A remaining freedom lies in the azimuthal streaming $\langle v_\phi \rangle$, which is not determined by the above assumptions. For the dark matter, it can be set to zero, or to a value corresponding to a prescribed spin.

$$\sigma_\phi^2 = \left\langle v_\phi^2 \right\rangle - \langle v_\phi \rangle^2$$

Note: For the stellar disk, we instead use the epicycle theory to relate radial and vertical dispersions.

The initial conditions for cosmic structure formation are directly observable

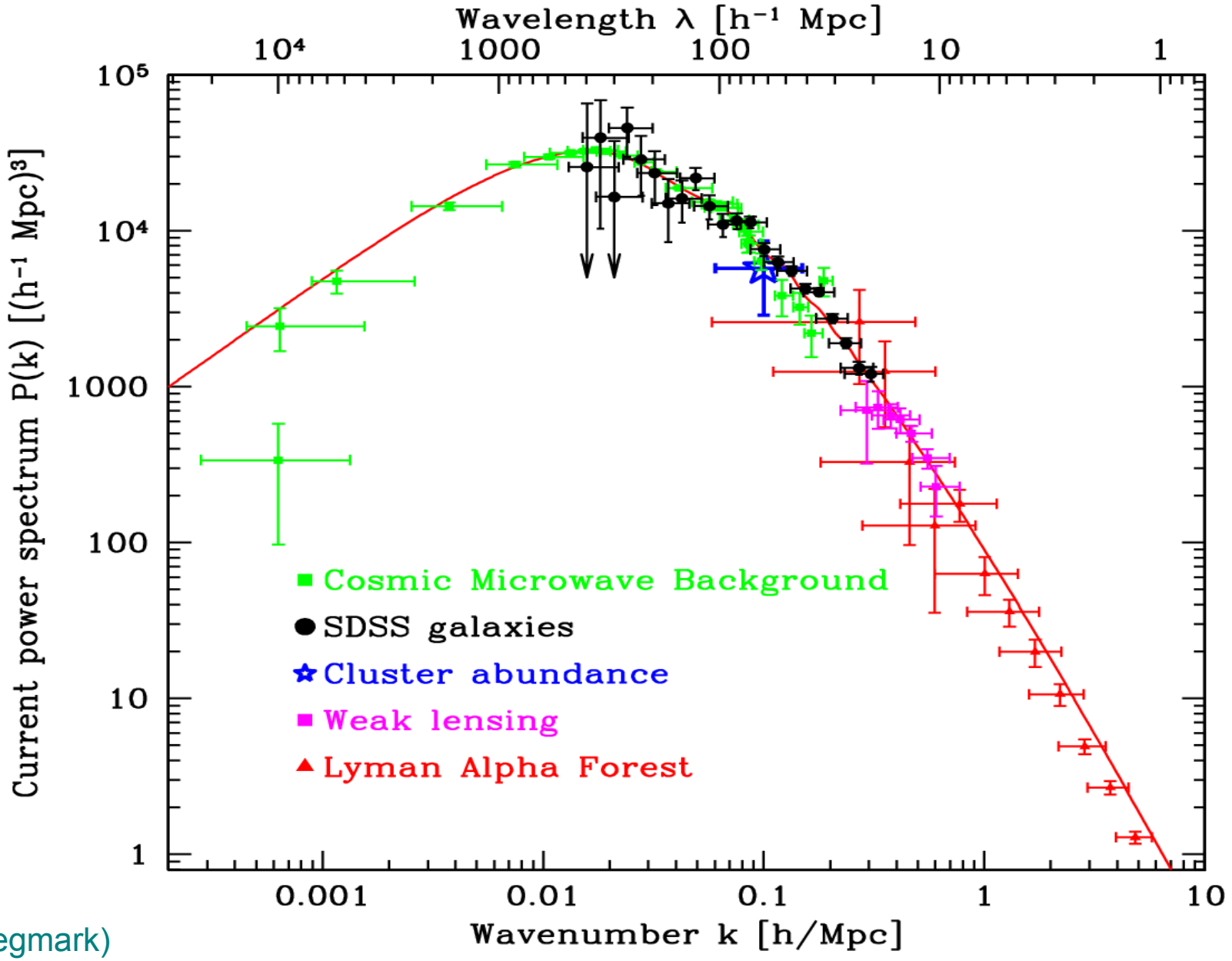
THE MICROWAVE SKY



WMAP Science Team (2003, 2006, 2008)

If the initial fluctuations are a Gaussian random field, we only need to know the power spectrum and the cosmological parameters to describe the ICs

DIFFERENT PROBES OF THE MASS POWER SPECTRUM



(figure from Max Tegmark)

To determine the power spectrum amplitude, we normalize the spectrum to observations of clustering (usually galaxy clusters)

FILTERED DENSITY FIELD AND THE NORMALIZATION OF THE POWER SPECTRUM

The filtered density field:

$$\sigma^2(M,z) = D^2(z)\int_0^\infty \frac{\mathrm{d}k}{2\pi^2}k^2P(k)\left[\frac{3j_1(kR)}{kR}\right]^2$$

Observational input:

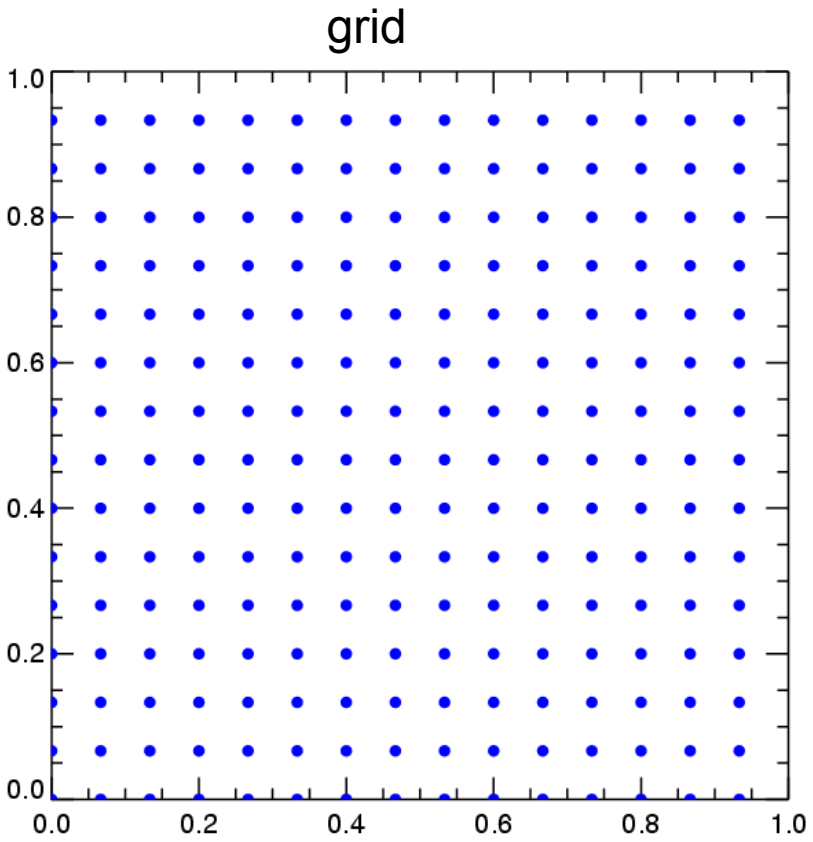
$$\sigma_8 = 0.74 - 0.9 \qquad R = 8\,h^{-1}\mathrm{Mpc}$$

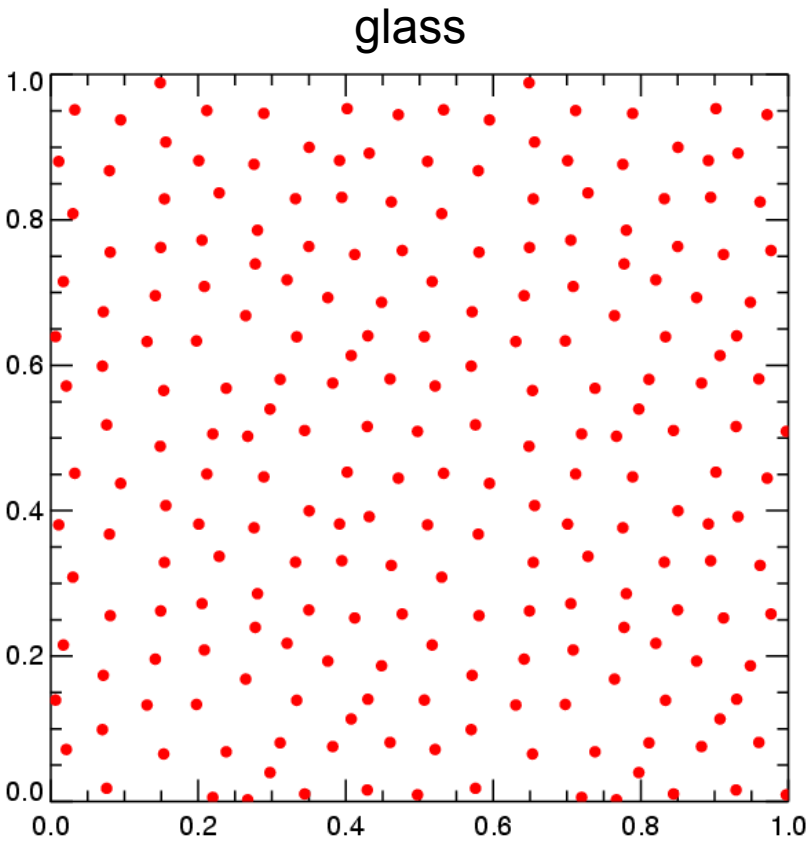
Extrapolate back to the starting redshift with the growth factor D(z)
This depends on cosmology.

→ fluctuation spectrum of initial conditions fully specified.

To create a realization of the perturbation spectrum, a model for an unperturbed density field is needed

GLASS OR CARTESIAN GRID





For CDM, the initial velocity dispersion is negligibly small.

But there is a mean streaming velocity, which we need to imprint in initial conditions.

Using the Zeldovich approximation, density fluctuations are converted to displacements of the unperturbed particle load

SETTING INITIAL DISPLACEMENTS AND VELOCITIES

Particle displacements: $\mathbf{d}_i(t) = \mathbf{x}_i(t) - \mathbf{q}_i$

Density change due to displacements: $\rho(\mathbf{x}) = \dfrac{\rho_0}{\left|\frac{\partial \mathbf{x}}{\partial \mathbf{q}}\right|} = \dfrac{\rho_0}{\left|\delta_{ij} + \frac{\partial \mathbf{d}}{\partial \mathbf{q}}\right|}$

For small displacements: $\left|\delta_{ij} + \dfrac{\partial \mathbf{d}}{\partial \mathbf{q}}\right| \simeq 1 + \nabla_{\mathbf{q}} \cdot \mathbf{d}$

Resulting density contrast: $\delta(\mathbf{x}) = \dfrac{\rho(\mathbf{x}) - \rho_0}{\rho} = -\nabla_{\mathbf{q}} \cdot \mathbf{d}$

During linear growth: $\delta(t) = D(t)\delta_0$, $\mathbf{d}(t) = D(t)\mathbf{d}_0$ ⇒ $\dot{\mathbf{x}} = \dot{\mathbf{d}} = \dot{a}\dfrac{\mathrm{d}D}{\mathrm{d}a}\mathbf{d}_0 = \dfrac{\dot{a}}{a}\dfrac{a}{D}\dfrac{\mathrm{d}D}{\mathrm{d}a}\mathbf{d}$

Particle velocities: $\dot{\mathbf{x}} = H(a)f(\Omega)\mathbf{d}$ $\qquad f(\Omega) = \dfrac{\mathrm{d}\ln D}{\mathrm{d}\ln a} \simeq \Omega^{0.6}$

Note: Particles move on straight lines in the Zeldovich approximation.

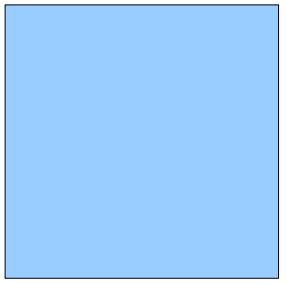
Displacement field: $\nabla^2\phi = \delta$ $\qquad \mathbf{d} = -\nabla\phi$

Fourier realization: $\phi_{\mathbf{k}} = -\dfrac{1}{k^2}\delta_{\mathbf{k}}$ $\qquad \mathbf{d}_{\mathbf{k}} = -i\mathbf{k}\phi_{\mathbf{k}} = \dfrac{i\mathbf{k}}{k^2}\delta_{\mathbf{k}}$ $\qquad \mathbf{d}_{\mathbf{k}} = -\nabla\phi = \sum_{\mathbf{k}} \dfrac{i\mathbf{k}\delta_{\mathbf{k}}}{k^2}\exp(i\mathbf{k}\mathbf{x})$

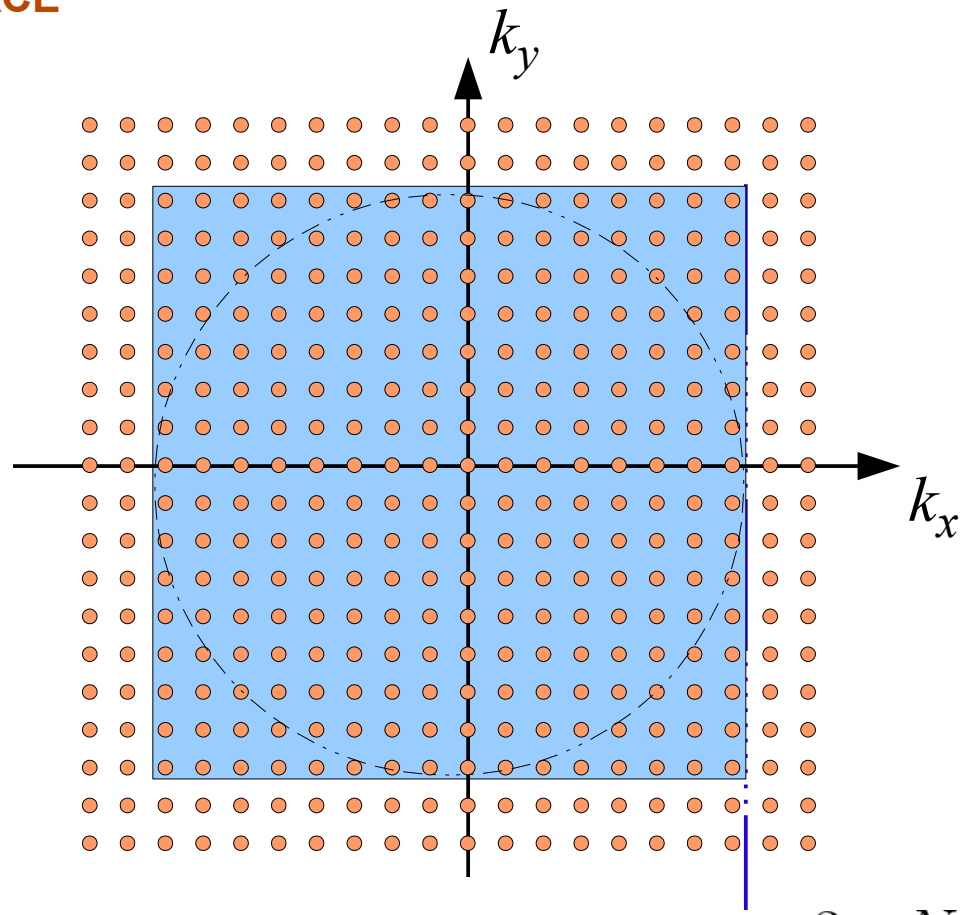
One can assign random amplitudes and phases for individual modes in Fourier space

GENERATING THE FLUCTUATIONS IN K-SPACE

Simulation box



sampled with N^2 points



$$\delta_{\mathbf{k}} = B_{\mathbf{k}} \exp^{\mathbf{i}\,\phi_{\mathbf{k}}}$$

For each mode, draw a random phase, and an amplitude from a Rayleigh distribution.

$$\left\langle \delta_{\mathbf{k}}^2 \right\rangle = P(k)$$

$$k_{\mathrm{Nyquist}} = \frac{2\pi}{L}\frac{N}{2}$$

Basics of SPH

The governing equations of an *ideal* gas can also be written in **Lagrangian form**

BASIC HYDRODYNAMICAL EQUATIONS

Euler equation:

$$\frac{\mathrm{d}\mathbf{v}}{\mathrm{d}t} = -\frac{\nabla P}{\rho} - \nabla\Phi$$

Continuity equation:

$$\frac{\mathrm{d}\rho}{\mathrm{d}t} + \rho\nabla\cdot\mathbf{v} = 0$$

First law of thermodynamics:

$$\frac{\mathrm{d}u}{\mathrm{d}t} = -\frac{P}{\rho}\nabla\cdot\mathbf{v} - \frac{\Lambda(u,\rho)}{\rho}$$

Equation of state of an ideal monoatomic gas:

$$P = (\gamma - 1)\rho u\ , \qquad \gamma = 5/3$$

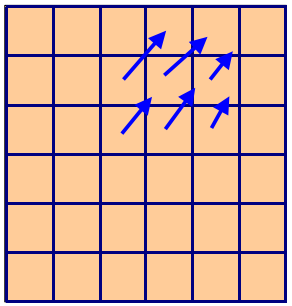
What is smoothed particle hydrodynamics?

DIFFERENT METHODS TO DISCRETIZE A FLUID

Eulerian

discretize space

representation on a mesh
(volume elements)



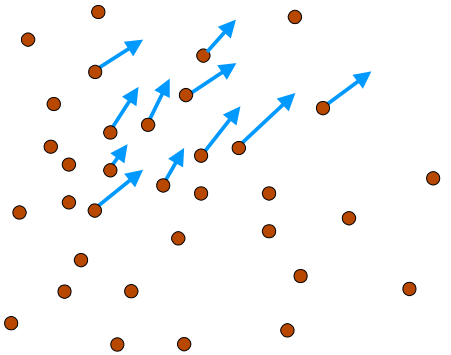
principle advantage:

high accuracy (shock capturing), low numerical viscosity

Lagrangian

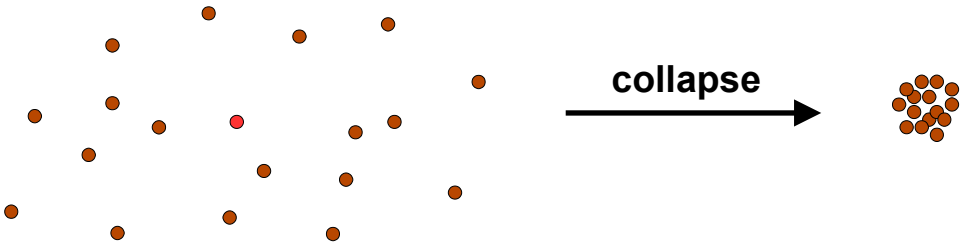
discretize mass

representation by fluid elements
(particles)



principle advantage:

resolutions adjusts automatically to the flow

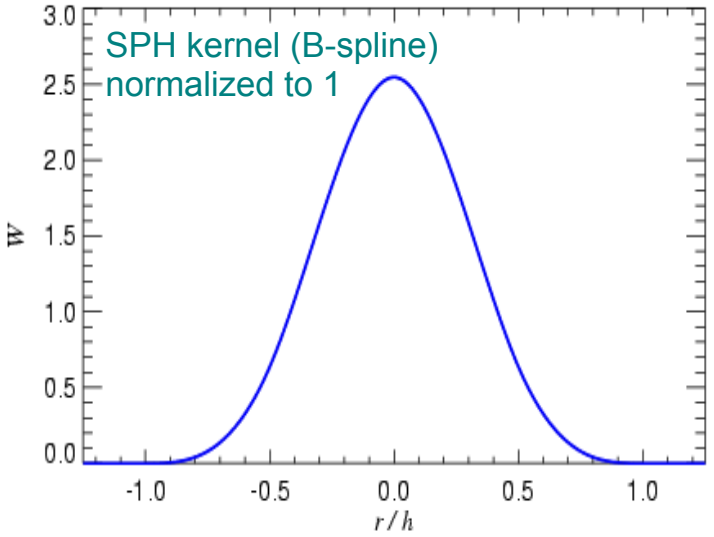


Kernel interpolation is used in smoothed particle hydrodynamics to build continuous fluid quantities from discrete tracer particles

DENSITY ESTIMATION IN SPH BY MEANS OF ADAPTIVE KERNEL ESTIMATION

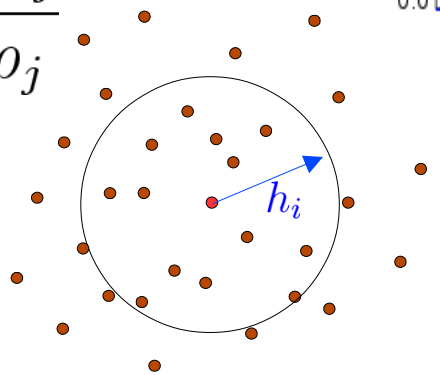
Kernel interpolant of an arbitrary function:

$$\langle A(\mathbf{r})\rangle = \int W(\mathbf{r}-\mathbf{r}', h)\, A(\mathbf{r}')\, \mathrm{d}^3 r'$$



If the function is only known at a set of discrete points, we approximate the integral as a sum, using the replacement:

$$\mathrm{d}^3 r' \mapsto \frac{m_j}{\rho_j}$$



$$\langle A_i\rangle = \sum_{j=1}^{N} \frac{m_j}{\rho_j}\, A_j\, W(\mathbf{r}_{ij}; h_i)$$

This leads to the SPH density estimate, for $A_i = \rho_i$

$$\rho_i = \sum_{j=1}^{N} m_j W(|\mathbf{r}_{ij}|, h_i)$$

→ **This can be differentiated !**

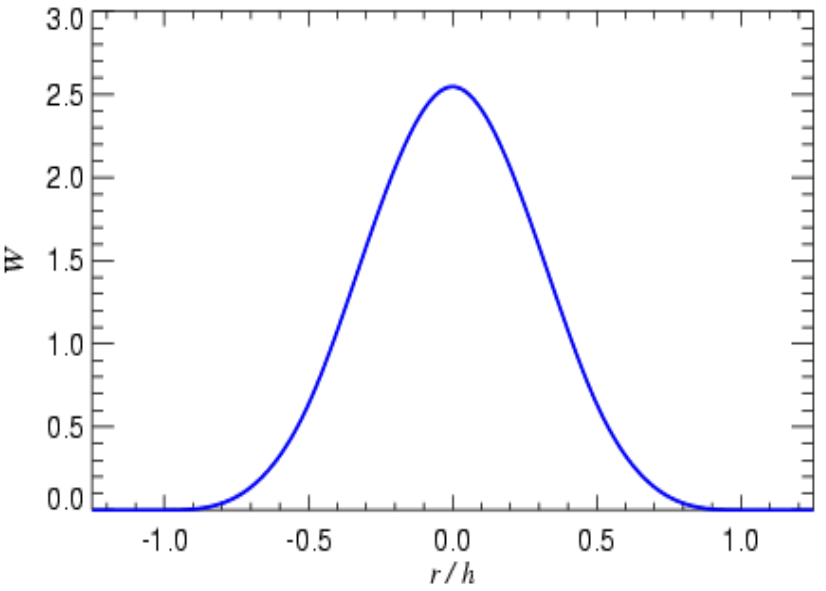
Good kernel shapes need to fulfill a number of constraints

CONDITIONS ON KERNELS

- Must be normalized to unity
- Compact support (otherwise N^2 bottleneck)
- High order of interpolation
- Spherical symmetry (for angular momentum conservation)

Nowadays, almost exclusively the cubic spline is used:

$$W(u) = \frac{8}{\pi} \begin{cases} 1 - 6u^2 + 6u^3, & 0 \le u \le \frac{1}{2}, \\ 2(1-u)^3, & \frac{1}{2} < u \le 1, \\ 0, & u > 1. \end{cases}$$



Kernel interpolants allow the construction of derivatives from a set of discrete tracer points

EXAMPLES FOR ESTIMATING THE VELOCITY DIVERGENCE

Smoothed estimate for the velocity field:

$$\langle \mathbf{v}_i \rangle = \sum_j \frac{m_j}{\rho_j} \mathbf{v}_j W(\mathbf{r}_i - \mathbf{r}_j)$$

Velocity divergence can now be readily estimated:

$$\nabla \cdot \mathbf{v} = \nabla \cdot \langle \mathbf{v}_i \rangle = \sum_j \frac{m_j}{\rho_j} \mathbf{v}_j \nabla_i W(\mathbf{r}_i - \mathbf{r}_j)$$

But alternative (and better) estimates are possible also:

Invoking the identity

$$\rho \nabla \cdot \mathbf{v} = \nabla \cdot (\rho \mathbf{v}) - \mathbf{v} \cdot \nabla \rho$$

one gets a “pair-wise” formula:

$$\rho_i (\nabla \cdot \mathbf{v})_i = \sum_j m_j (\mathbf{v}_j - \mathbf{v}_i) \nabla_i W(\mathbf{r}_i - \mathbf{r}_j)$$

Smoothed particle hydrodynamics is governed by a set of ordinary differential equations

BASIC EQUATIONS OF SMOOTHED PARTICLE HYDRODYNAMICS

Each particle carries either the energy or the entropy per unit mass as independent variable

Density estimate

$$\rho_i = \sum_{j=1}^{N} m_j W(|\mathbf{r}_{ij}|, h_i)$$

→ **Continuity equation automatically fulfilled.**

$$\rightarrow \quad P_i = (\gamma - 1)\rho_i u_i$$

Euler equation

$$\frac{\mathrm{d}\mathbf{v}_i}{\mathrm{d}t} = -\sum_{j=1}^{N} m_j \left(\frac{P_i}{\rho_i^2} + \frac{P_j}{\rho_j^2} + \Pi_{ij} \right) \nabla_i \overline{W}_{ij}$$

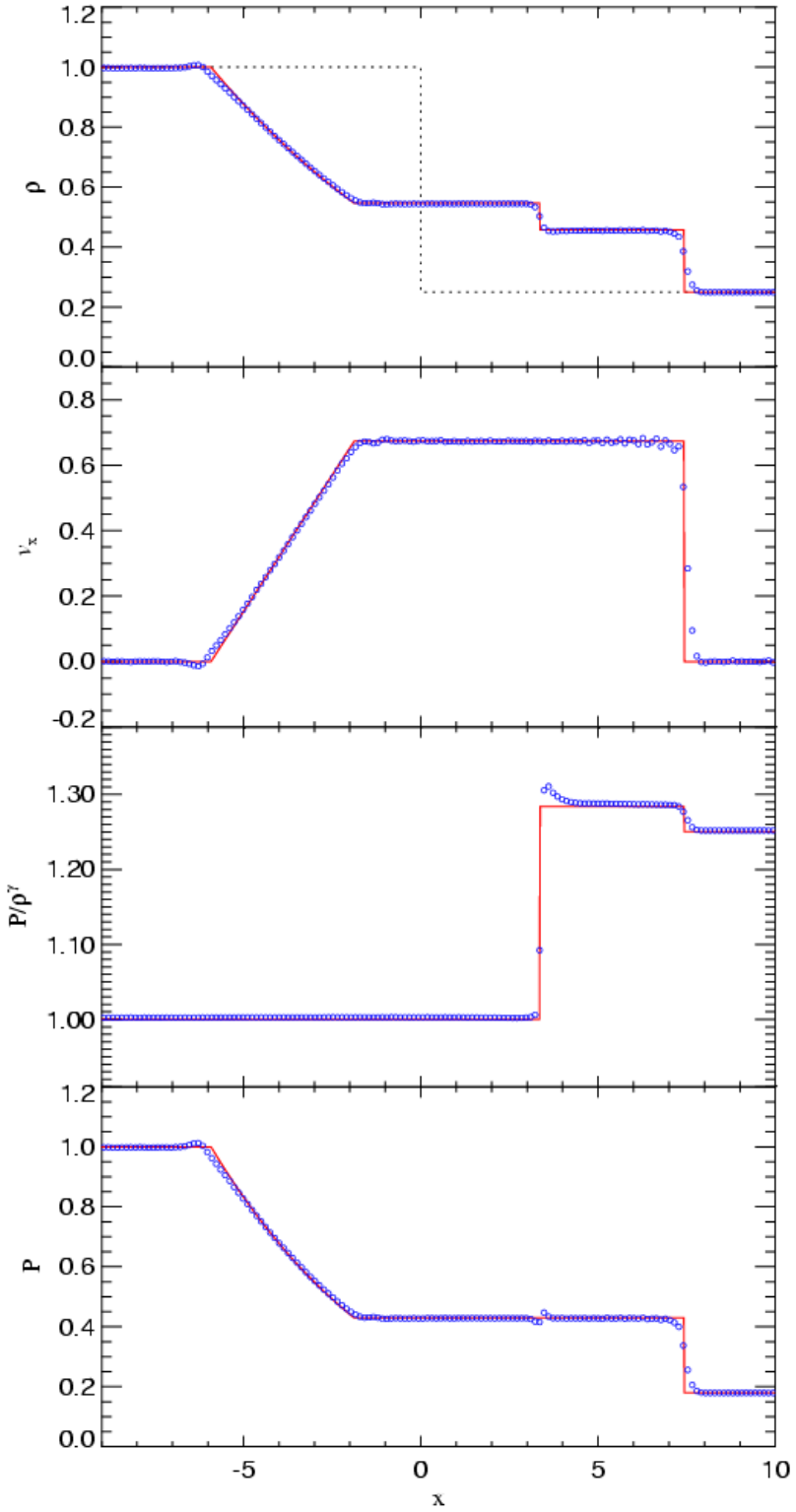
$+\Pi_{ij}$ **Artificial viscosity**

First law of thermodynamics

$$\frac{\mathrm{d}u_i}{\mathrm{d}t} = \frac{1}{2} \sum_{j=1}^{N} m_j \left(\frac{P_i}{\rho_i^2} + \frac{P_j}{\rho_j^2} + \Pi_{ij} \right) \mathbf{v}_{ij} \cdot \nabla_i \overline{W}_{ij}$$

$+\Pi_{ij}$

Viscosity and shock capturing



An artificial viscosity needs to be introduced to capture shocks

SHOCK TUBE PROBLEM AND VISCOSITY

viscous force:

$$\left.\frac{\mathrm{d}\mathbf{v}_i}{\mathrm{d}t}\right|_{\mathrm{visc}} = -\sum_{j=1}^{N} m_j \Pi_{ij} \nabla_i \overline{W}_{ij}$$

parameterization of the artificial viscosity:

$$\Pi_{ij} = \begin{cases} -\frac{\alpha}{2}\frac{[c_i + c_j - 3w_{ij}]w_{ij}}{\rho_{ij}} & \text{if } \mathbf{v}_{ij}\cdot\mathbf{r}_{ij} < 0 \\ 0 & \text{otherwise} \end{cases}$$

$$v_{ij}^{\mathrm{sig}} = c_i + c_j - 3w_{ij},$$

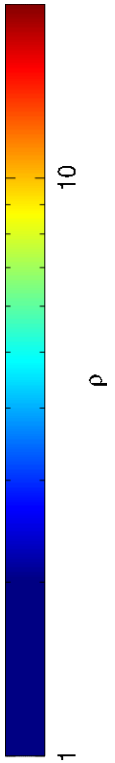
$$w_{ij} = \mathbf{v}_{ij}\cdot\mathbf{r}_{ij}/|\mathbf{r}_{ij}|$$

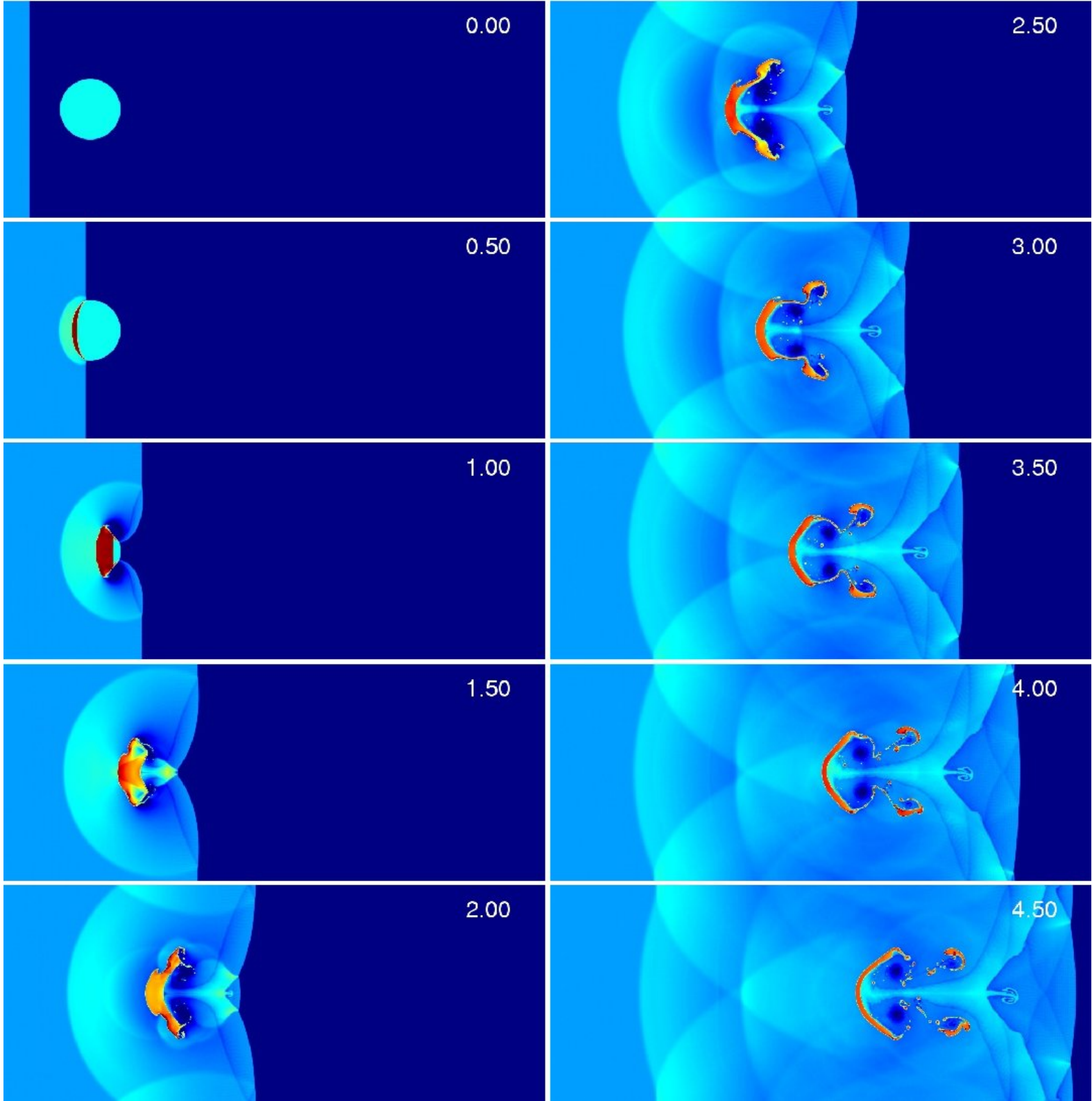
heat production rate:

$$\frac{\mathrm{d}u_i}{\mathrm{d}t} = \frac{1}{2}\sum_{j=1}^{N} m_j \Pi_{ij}\mathbf{v}_{ij}\cdot\nabla_i \overline{W}_{ij}$$

SPH can handle strong shocks and vorticity generation

A MACH NUMBER 10 SHOCK THAT STRIKES AN OVERDENSE CLOUD





Variational derivation of SPH

The traditional way to derive the SPH equations leaves room for many different formulations

SYMMETRIZATION CHOICES

Symmetrized kernel:

$$\overline{W}_{ij} = W(|\mathbf{r}_{ij}|, [h_i + h_j]/2)$$

$$\overline{W}_{ij} = \frac{1}{2}\left[W(|\mathbf{r}_{ij}|, h_i) + W(|\mathbf{r}_{ij}|, h_j)\right]$$

Symmetrization of pressure terms:

Using $\nabla P = 2\sqrt{P}\nabla\sqrt{P}$

$$\frac{1}{2}\left(\frac{P_i}{\rho_i^2} + \frac{P_j}{\rho_j^2}\right) \iff \sqrt{\frac{P_i\, P_j}{\rho_i^2\, \rho_j^2}}$$

Is there a best choice?

For an adiabatic flow, temperature can be derived from the specific entropy

ENTROPY FORMALISM

Definition of an **entropic function**:

$$P_i = A_i \, \rho_i^{\gamma}$$

for an adiabtic flow:

$$A_i = A_i(s_i) = \text{const.}$$

don't integrate the temperature, but infer it from:

$$u_i = \frac{A_i}{\gamma - 1} \rho^{\gamma - 1}$$

Use an artificial viscosity to generate entropy in shocks:

$$\frac{\mathrm{d}A_i}{\mathrm{d}t} = \frac{1}{2} \frac{\gamma - 1}{\rho_i^{\gamma - 1}} \sum_{j=1}^{N} m_j \Pi_{ij} \mathbf{v}_{ij} \cdot \nabla_i \overline{W}_{ij}$$

None of the adaptive classic SPH schemes conserves energy and entropy simultaneously

CONSERVATION LAW TROUBLES

Hernquist (1993):

> If the **thermal energy** is **integrated**, **entropy** conservation can be **violated**...
>
> If the **entropy** is **integrated**, total **energy** is **not** necessarily **conserved**...

The trouble is caused by varying smoothing lengths... ∇h -terms

Do we have to worry about this? **YES**

Can we do better? **YES**

A fully conservative formulation of SPH

Springel & Hernquist (2002)

DERIVATION

Lagrangian:

$$L(\mathbf{q},\dot{\mathbf{q}}) = \frac{1}{2}\sum_{i=1}^{N} m_i \dot{\mathbf{r}}_i^2 - \frac{1}{\gamma-1}\sum_{i=1}^{N} m_i A_i \rho_i^{\gamma-1}$$

$$\mathbf{q} = (\mathbf{r}_1, \ldots, \mathbf{r}_N, h_1, \ldots, h_N)$$

Constraints:

$$\phi_i(\mathbf{q}) \equiv \frac{4\pi}{3} h_i^3 \rho_i - M_{\rm sph} = 0$$

Equations of motion:

$$\frac{\mathrm{d}}{\mathrm{d}t}\frac{\partial L}{\partial \dot{q}_i} - \frac{\partial L}{\partial q_i} = \sum_{j=1}^{N} \lambda_j \frac{\partial \phi_j}{\partial q_i}$$

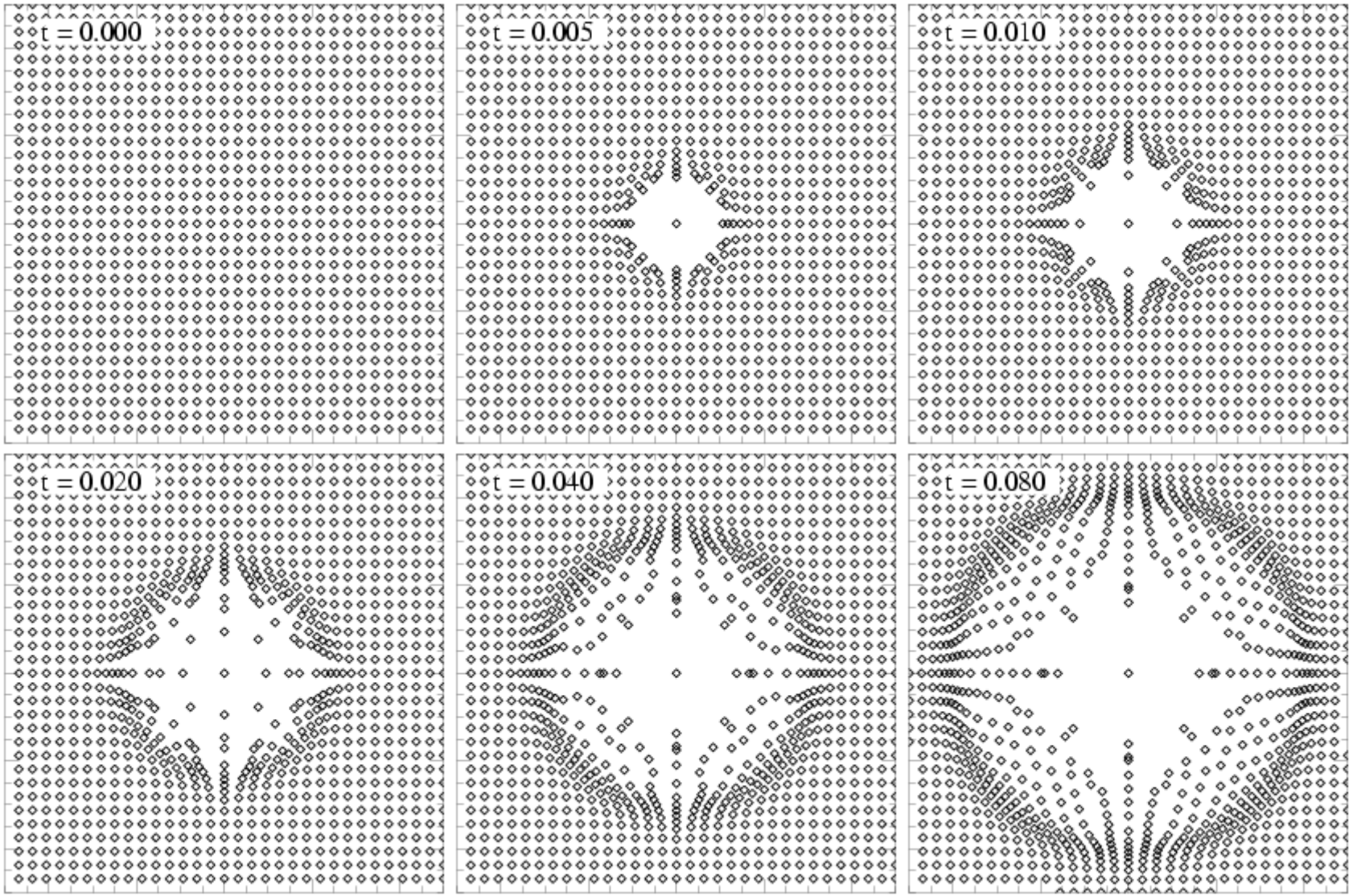
$$\frac{\mathrm{d}\mathbf{v}_i}{\mathrm{d}t} = -\sum_{j=1}^{N} m_j \left[f_i \frac{P_i}{\rho_i^2} \nabla_i W_{ij}(h_i) + f_j \frac{P_j}{\rho_j^2} \nabla_i W_{ij}(h_j) \right]$$

$$f_i = \left[1 + \frac{h_i}{3\rho_i} \frac{\partial \rho_i}{\partial h_i} \right]^{-1}$$

Does the entropy formulation give better results?

A point-explosion in three-dimensional SPH

TAYLOR-SEDOV BLAST

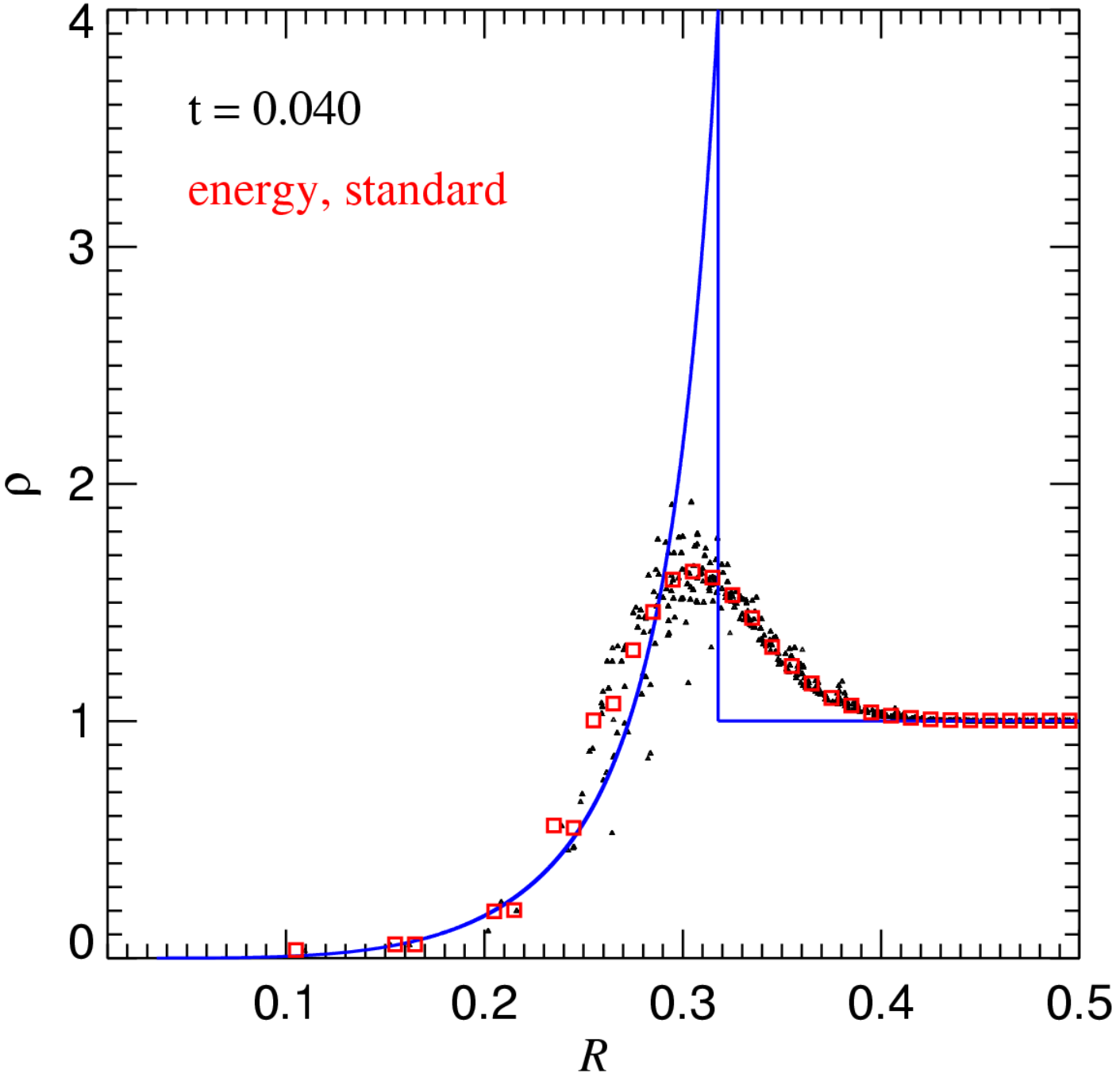


- Geometric formulation gives completely unphysical result (no explosion at all)
- Standard energy formulation produces severe error in total energy, but asymmetric form ok
- Standard entropy formulation ok, but energy fluctuates by several percent

There is a well-known similarity solution for strong point-like explosions

SEDOV-TAYLOR SOLUTIONS FOR **SMOOTHED** EXPLOSION ENERGY

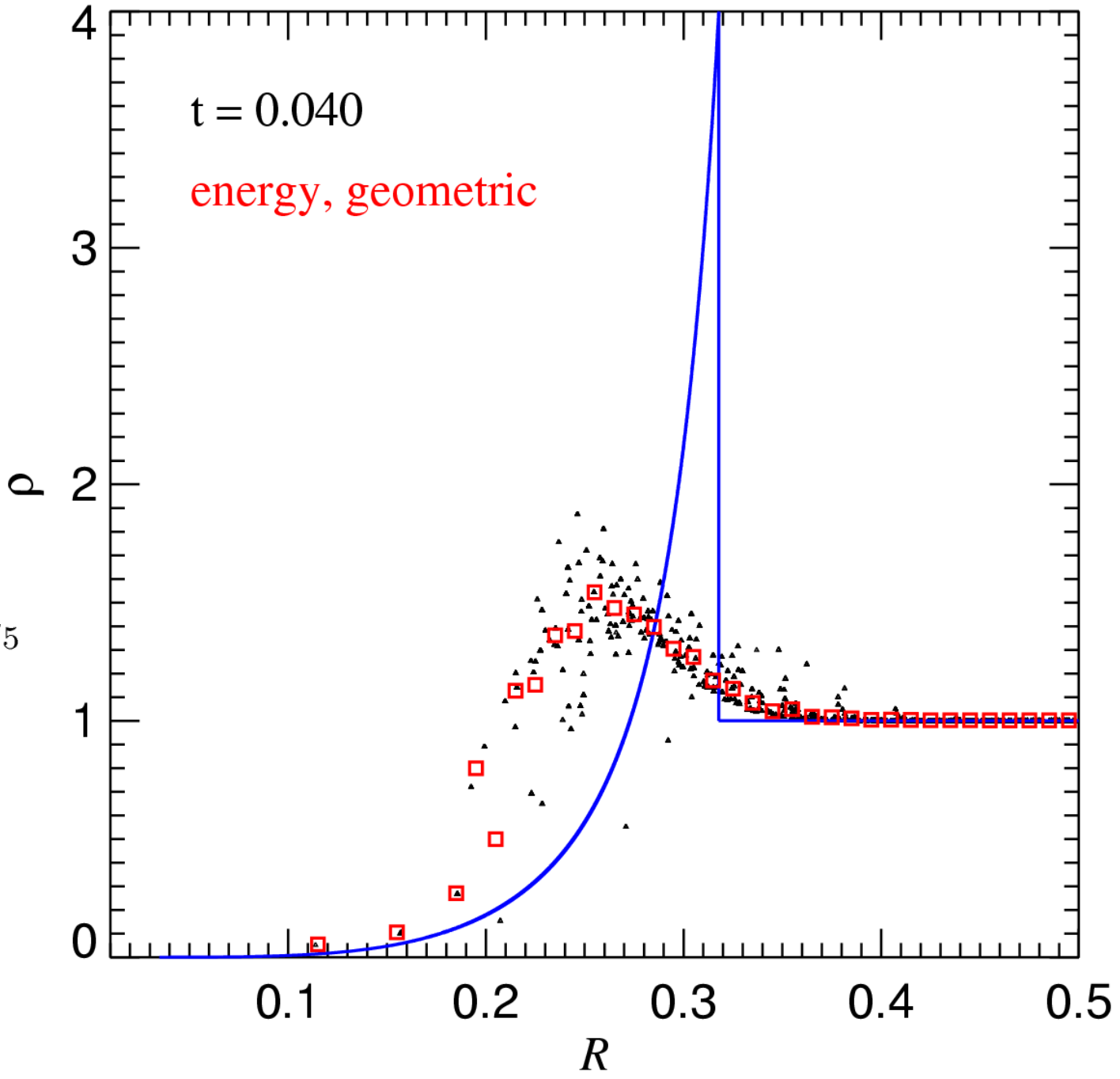
$$R(t) = \beta \left(\frac{Et^2}{\rho} \right)^{1/5}$$



There is a well-known similarity solution for strong point-like explosions

SEDOV-TAYLOR SOLUTIONS FOR **SMOOTHED** EXPLOSION ENERGY

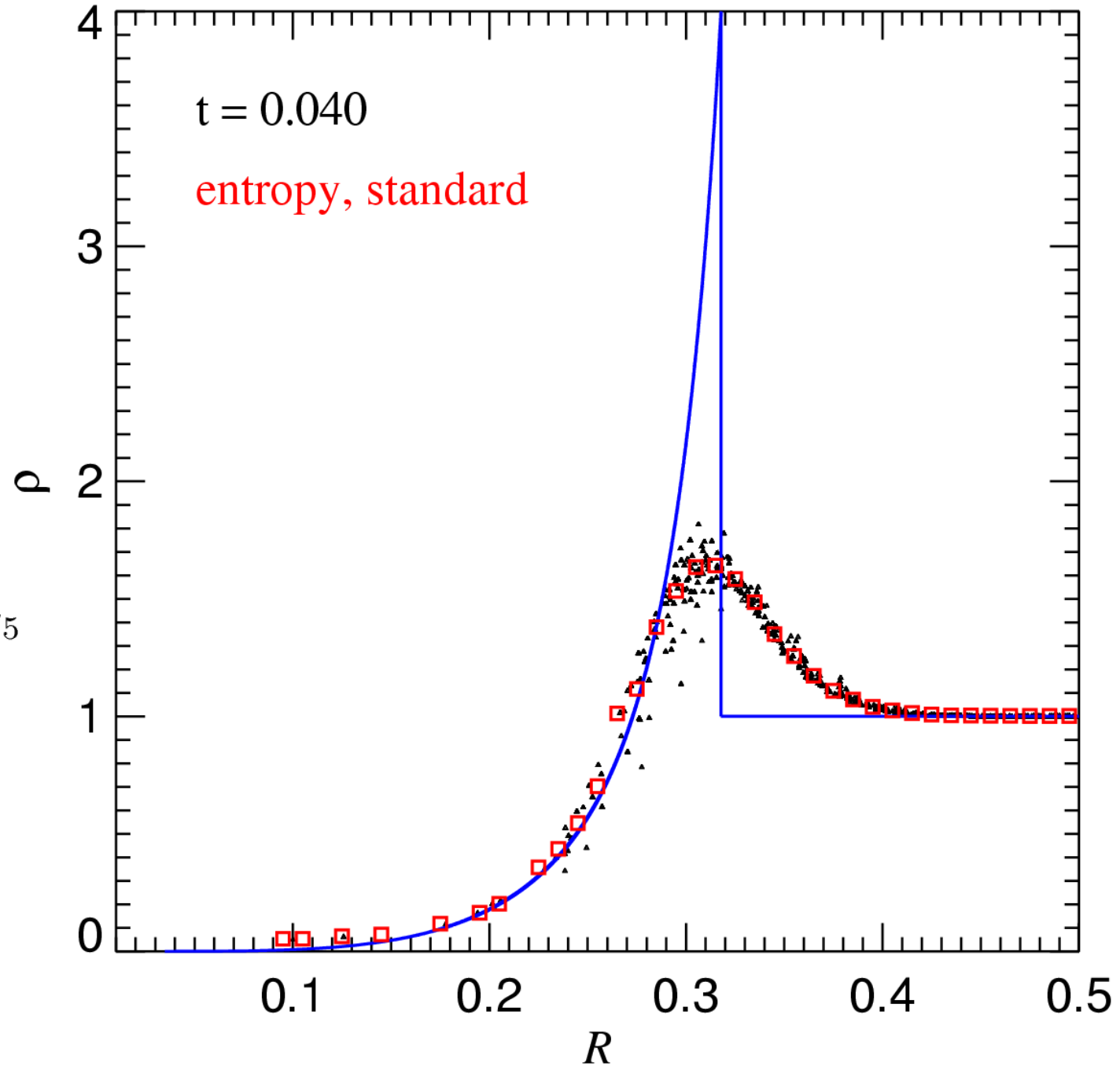
$$R(t) = \beta \left(\frac{Et^2}{\rho} \right)^{1/5}$$



There is a well-known similarity solution for strong point-like explosions

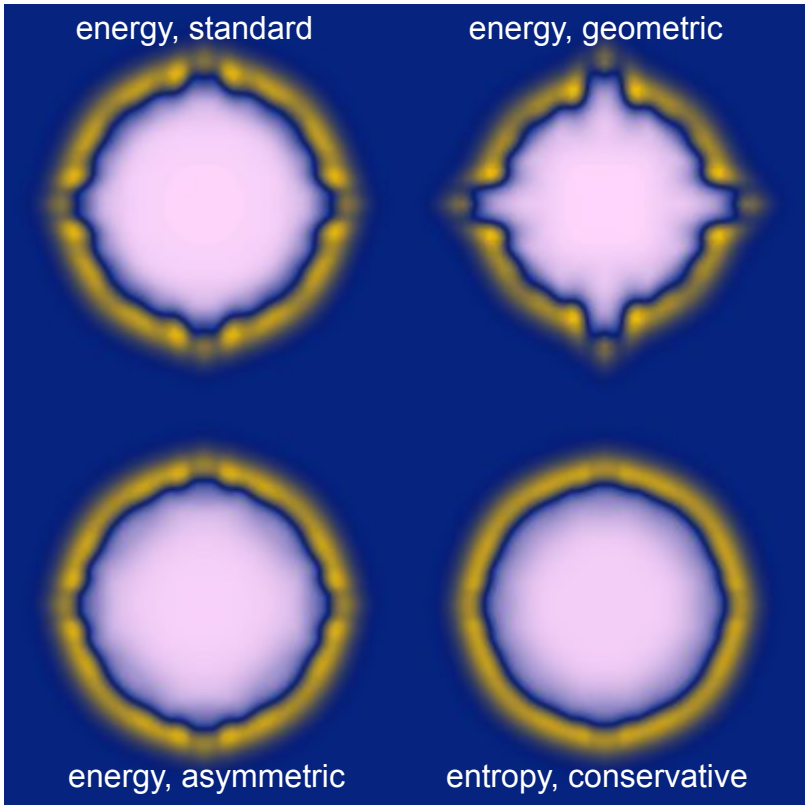
SEDOV-TAYLOR SOLUTIONS FOR **SMOOTHED** EXPLOSION ENERGY

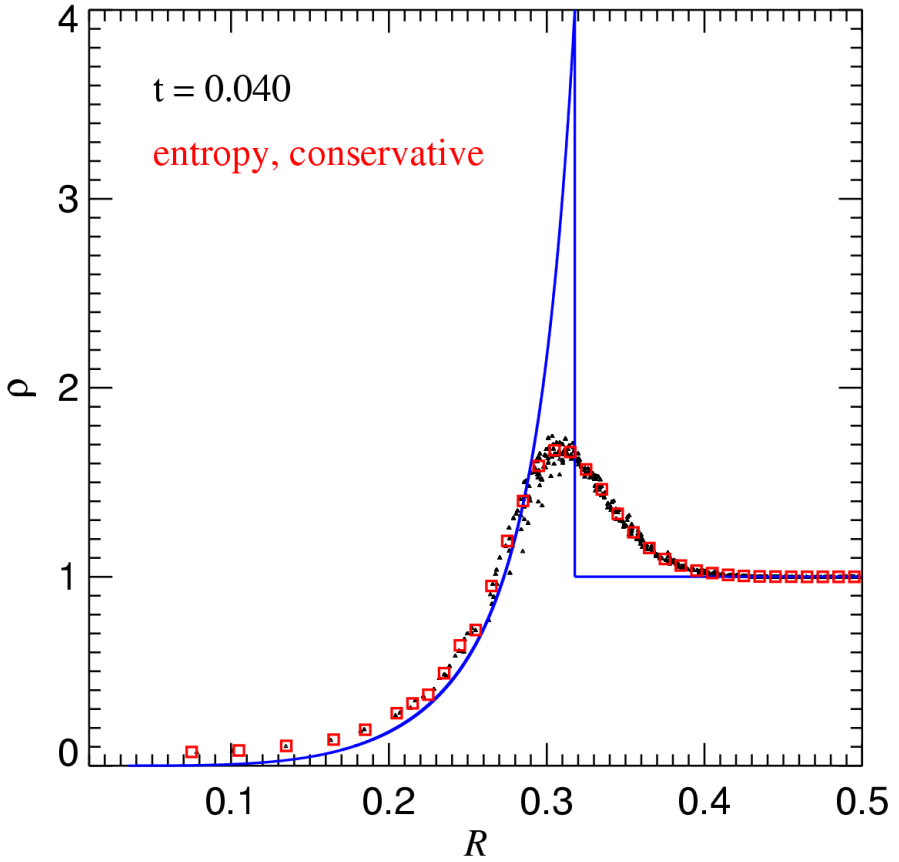
$$R(t) = \beta \left(\frac{Et^2}{\rho} \right)^{1/5}$$



The new conservative formulation gives better results for adiabtic flows

EXPLOSION PROBLEM

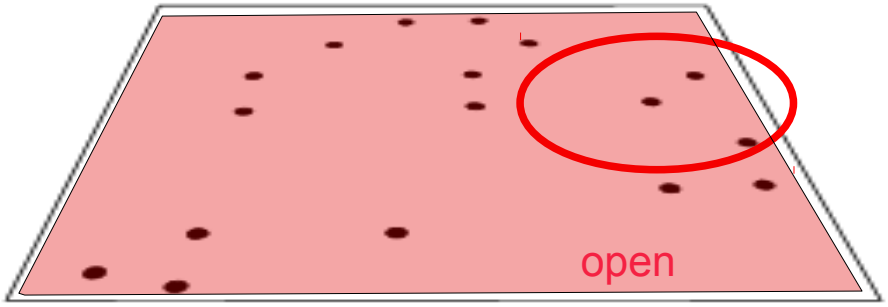


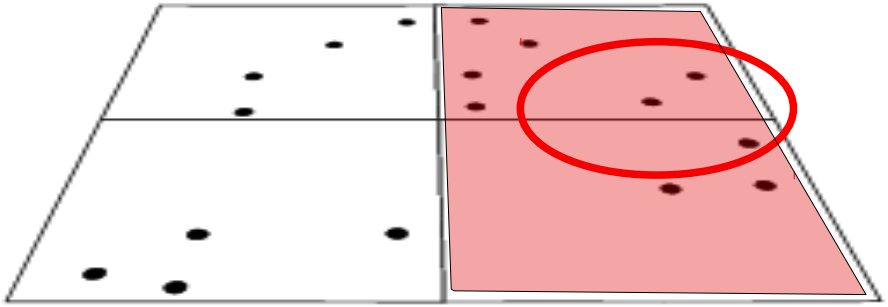


Neighbor search in SPH

RANGE SEARCHING WITH THE TREE

An efficient neighbor search is the most important factor that determines the speed of an SPH code

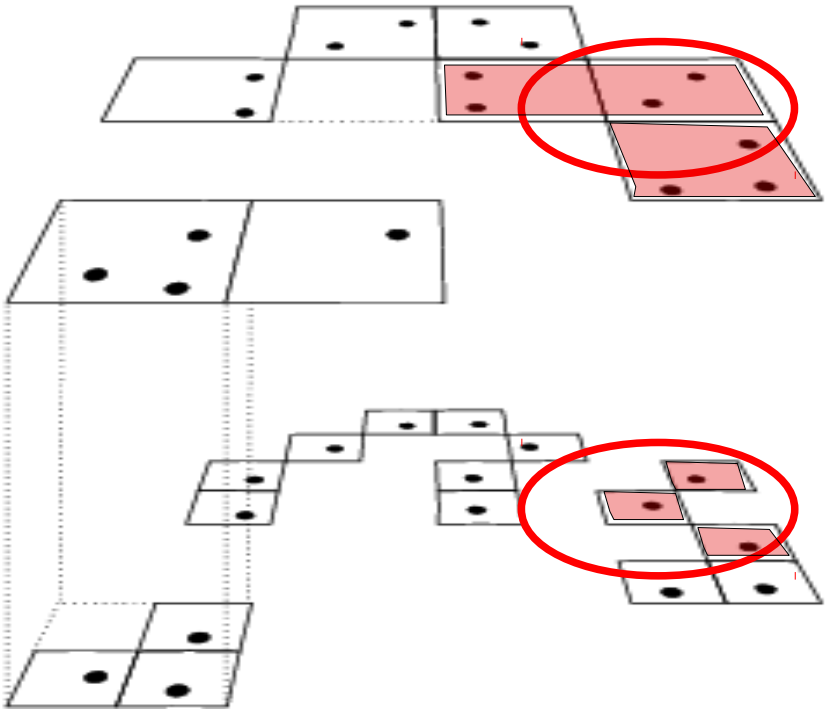




<u>But:</u> A simple search radius is not always sufficient, since for the hydro force we need to find all particles with

$$|\mathbf{r}_i - \mathbf{r}_j| < \max(h_i, h_j)$$

Solution: Store in each tree node the maximum h of all particles in the node.



SPH accurately conserves all relevant conserved quantities in self-gravitating flows

SOME NICE PROPERTIES OF SPH

- **Mass is conserved**
- **Momentum is conserved**
- **Total energy is conserved – also in the presence of self-gravity !**
- **Angular momentum is conserved**
- **Entropy is conserved – only produced by artificial viscosity, no entropy production due to mixing or advection**

Furthermore:

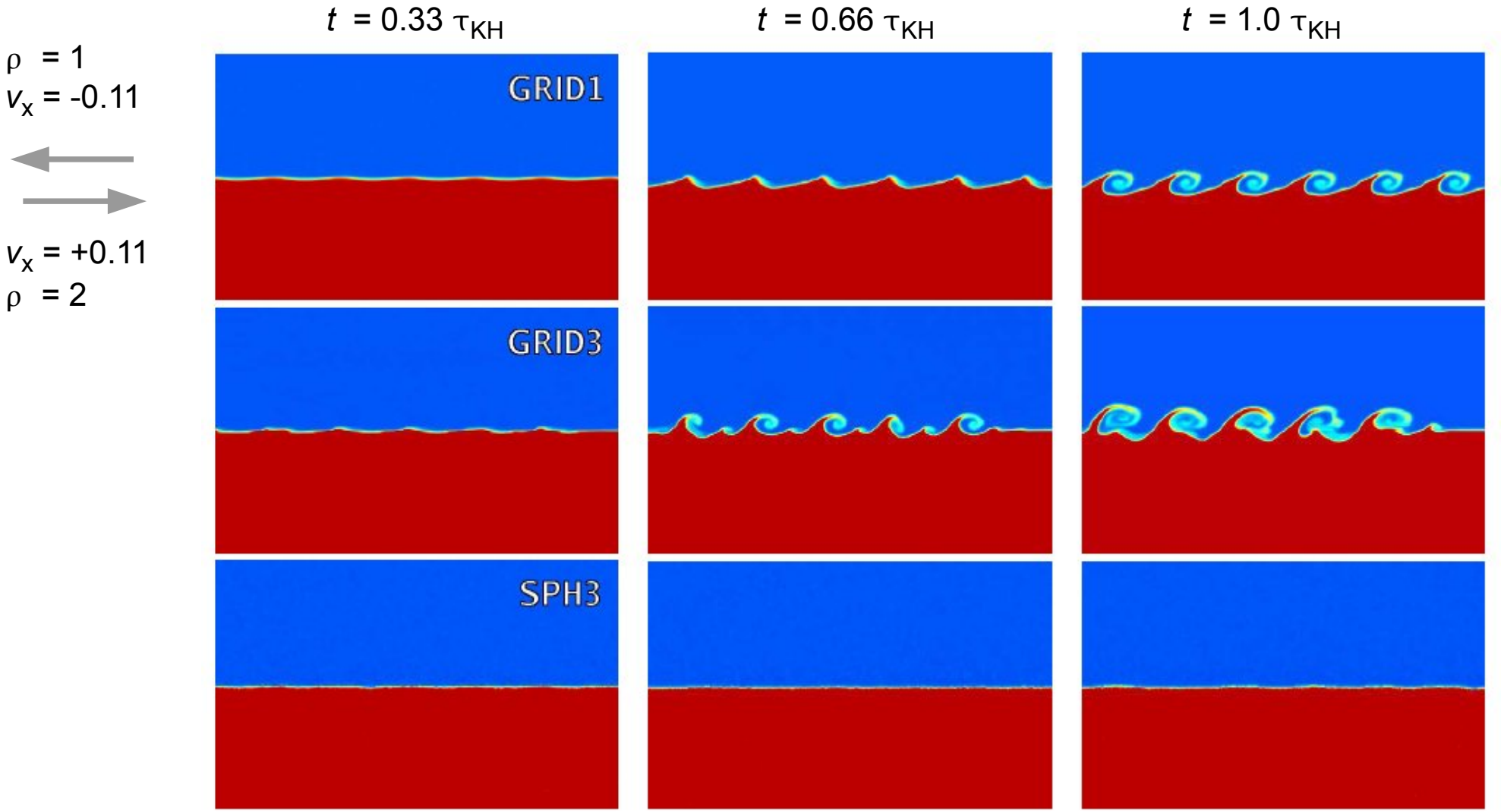
- **High geometric flexibility**
- **Easy incorporation of vacuum boundary conditions**
- **No high Mach number problem**

Fluid instabilities and mixing in SPH

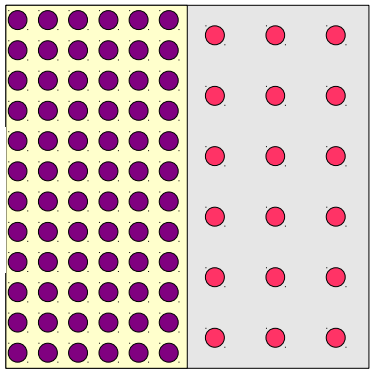
In SPH, fluid instabilities at contact discontinuities with large density jumps tend to be suppressed by a spurious numerical surface tension

KELVIN-HELMHOLTZ INSTABILITIES IN SPH

Agertz et al. (2007)



A simple *Gedankenexperiment* about mixing in SPH



$\rho_1 = 8$ $\quad u_1 = \frac{1}{8}$ $\quad V_1 = \frac{1}{2}$

$\rho_2 = 1$ $\quad u_2 = 1$ $\quad V_2 = \frac{1}{2}$

The pressure is constant:

$$P_1 = (\gamma - 1)\rho_1 u_1 = \frac{2}{3} \qquad P_2 = P_1$$

The specific entropies are:

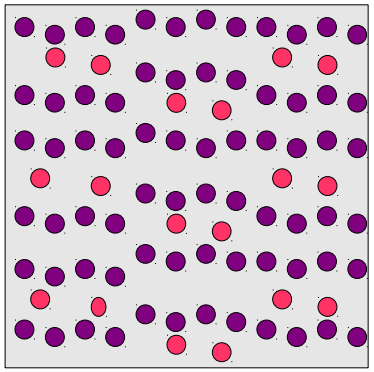
$$A_i = \frac{P_i}{\rho_i^\gamma} \qquad A_1 = \frac{1}{48} \qquad A_2 = \frac{2}{3}$$

Let's calculate the total thermal energy of the system:

$$E_{\rm therm} = \int \frac{A\rho^{\gamma-1}}{\gamma - 1}\,{\rm d}m$$

$$E_{\rm therm} = 1$$

We now mix the particles, keeping their specific entropies fixed:



All particles estimate the same mean density: $M_{\rm tot} = \frac{9}{2}$ $\quad \overline{\rho} = \frac{9}{2}$

The thermal energy thus becomes:

$$E_{\rm therm} = \frac{M_1 A_1 \overline{\rho}^{2/3}}{2/3} + \frac{M_2 A_2 \overline{\rho}^{2/3}}{2/3}$$

$$E_{\rm therm} = \frac{5}{8}\left(\frac{9}{2}\right)^{2/3} \simeq 1.7$$

⇒ **This mixing process is energetically forbidden!**

What happened to the entropy in our *Gedankenexperiment* ?

In slowly mixing the two phases, we preserve the total thermal energy:

Expect: $\bar{u} = \frac{2}{9}$ $\bar{A} = \frac{2}{3}\frac{\bar{u}}{\bar{\rho}^{2/3}}$ $\bar{A} = \frac{2^{8/3}}{3^{13/3}} \simeq 0.054$

The Sackur-Tetrode equation for the entropy of an ideal gas can be written as:

$$S = \frac{3}{2}\frac{k_B}{\mu} M \left[\ln\left(\frac{P}{\rho^\gamma}\right) + \ln\left(\frac{2\pi\mu^{8/3}}{h^2}\right) + \frac{5}{3} \right]$$

If the mass in a system is conserved, it is sufficient to consider the simplified entropy:

$$\tilde{S} = M \ln A$$

When the system is mixed, the change of the entropy is:

$$\Delta\tilde{S} = M_{\rm tot} \ln \bar{A} - (M_1 \ln A_1 + M_2 \ln A_2)$$

$$\Delta\tilde{S} \simeq 2.55 \quad \geq \quad 0$$

Unless this entropy is generated somehow, SPH will have problems to mix different phases of a flow.

(Aside: Mesh codes can generate entropy outside of shocks – this allows them to treat mixing.)

New developments in SPH that try to address mixing

Artificial heat conduction at contact discontinuities has been proposed as a solution for the suppressed fluid instabilities

ARTIFICIAL HEAT MIXING TERMS

Price (2008) **Wadsley, Veeravalli & Couchman (2008)**

Price argues that in SPH every conservation law requires dissipative terms to capture discontinuities.

The normal artificial viscosity applies to the momentum equation, but discontinuities in the (thermal) energy equation should also be treated with a dissipative term.

For every conserved quantity *A*

$$\sum_j m_j \mathrm{d}A_j/\mathrm{d}t = 0$$

a dissipative term is postulated

$$\left(\frac{\mathrm{d}A_i}{\mathrm{d}t}\right)_{\mathrm{diss}} = \sum_j m_j \frac{\alpha_A v_{\mathrm{sig}}}{\bar{\rho}_{ij}} (A_i - A_j)\hat{\mathbf{r}}_{ij} \cdot \nabla W_{ij}$$

that is designed to capture discontinuities.

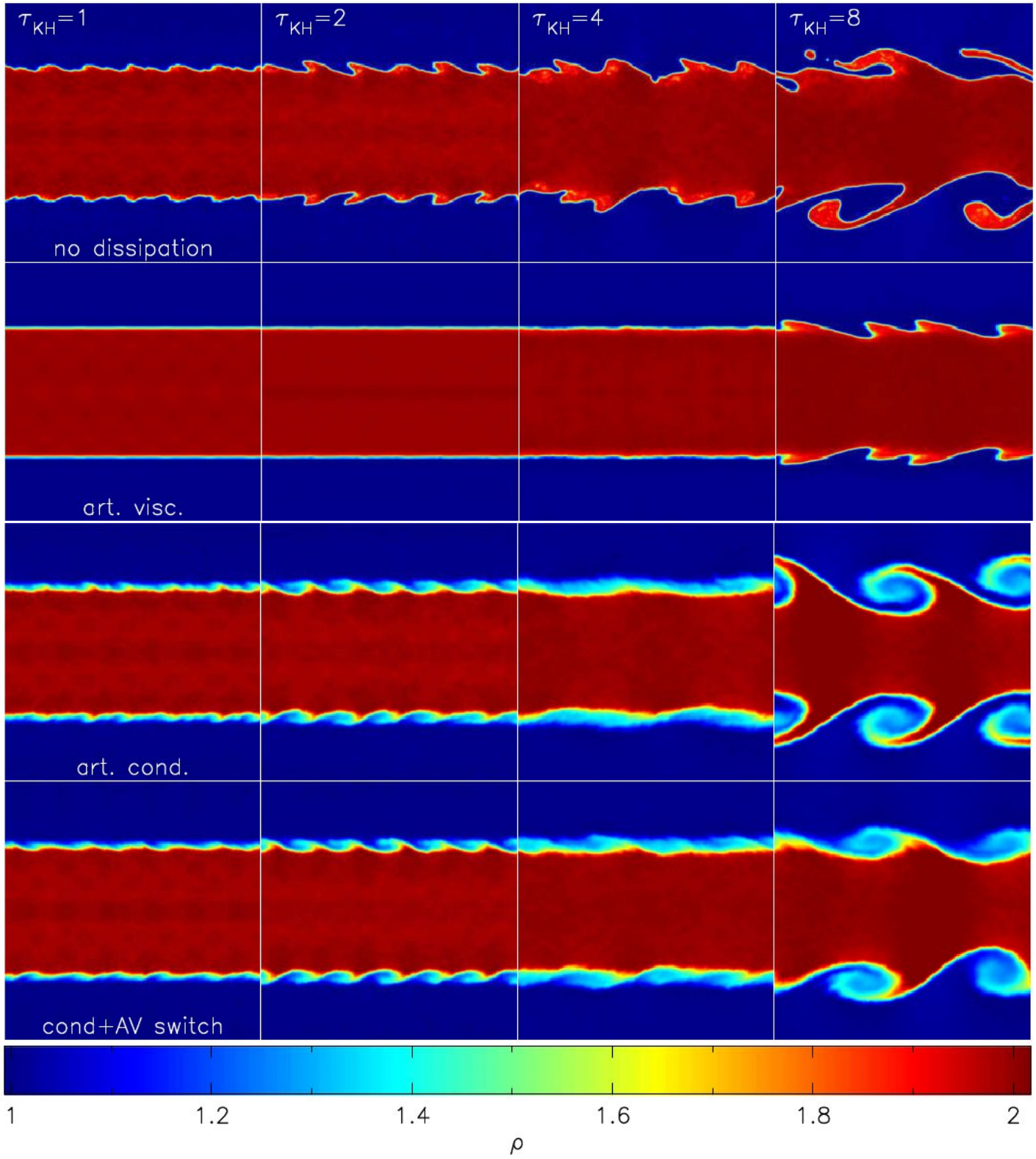
This is the discretized form of a diffusion problem:

$$\left(\frac{\mathrm{d}A}{\mathrm{d}t}\right)_{\mathrm{diss}} \approx \eta \nabla^2 A$$

$$\eta \propto \alpha v_{\mathrm{sig}} |r_{ij}|$$

Artificial heat conduction drastically improves SPH's ability to account for fluid instabilities and mixing

COMPARISON OF KH TESTS FOR DIFFERENT TREATMENTS OF THE DISSIPATIVE TERMS



Price (2008)

Another route to better SPH may lie in different ways to estimate the density

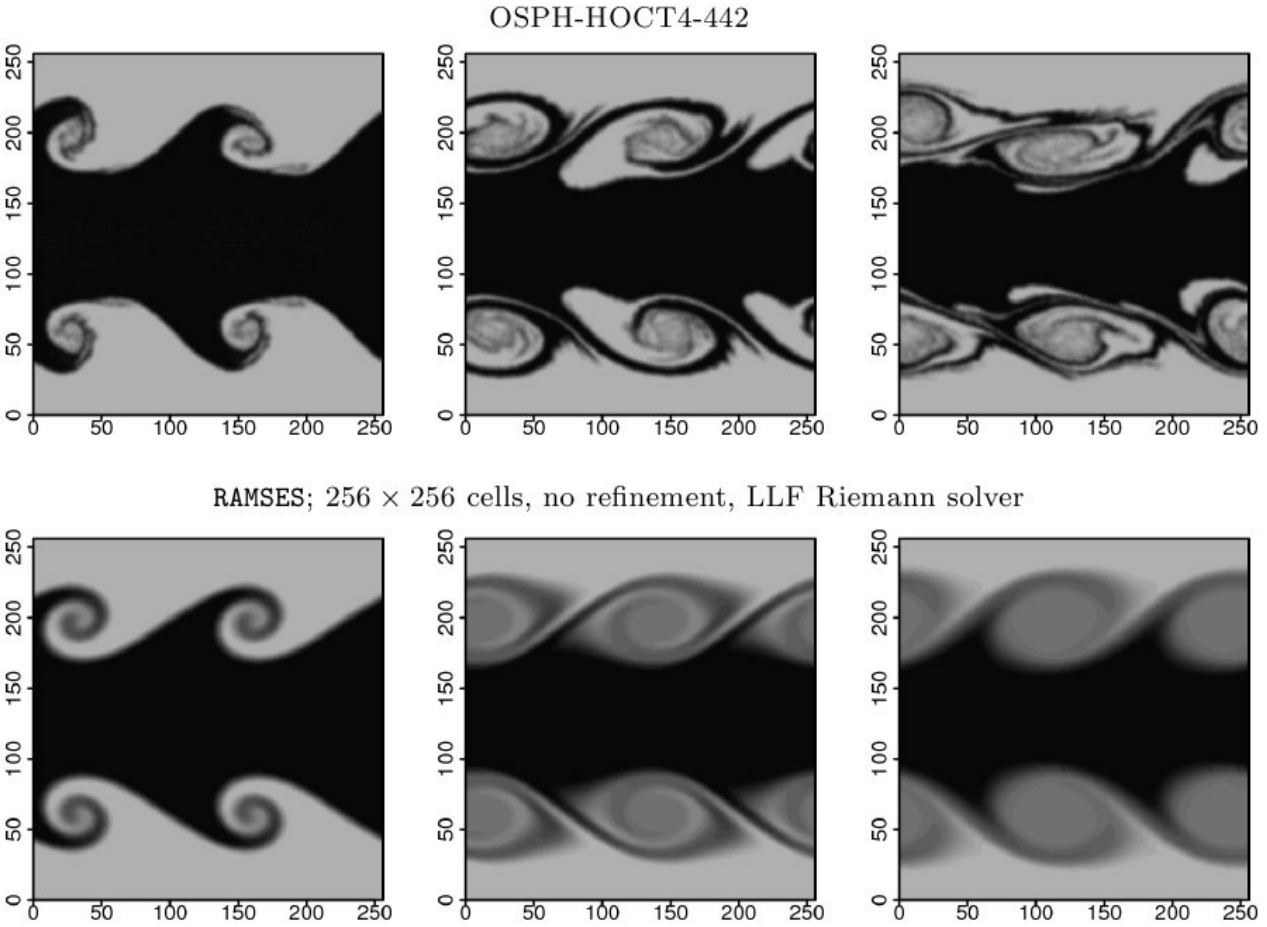
AN ALTERNATIVE SPH FORMULATION

"Optimized SPH" (OSPH) of **Read, Hayfield, Agertz (2009)**

- Density estimate like Ritchie & Thomas (2001):

$$\rho_i = \sum_j^N \left(\frac{A_j}{A_i}\right)^{\frac{1}{\gamma}} m_j \overline{W}_{ij}$$

- Very large number of neighbors (442 !) to beat down noise
- Needs peaked kernel to suppress clumping instability
- This in turn reduces the order of the density estimate, so that a large number of neighbors is required.



Signal propagation with hyperbolic equations

THE ADVECTION EQUATION

Let's take the continuity equation and simplify it a bit by assuming constant velocity and 1 dimension:

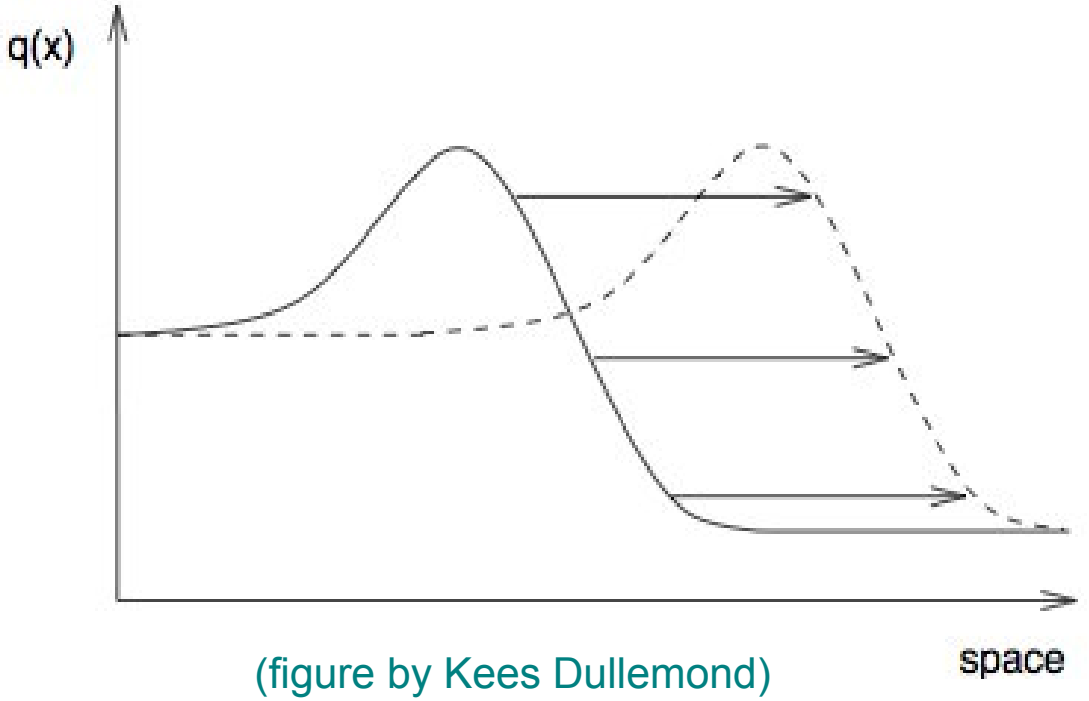
$$\frac{\partial \rho}{\partial t} + \nabla(\rho \mathbf{v}) = 0$$

We get a simple **advection equation**:

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

The solution is:

$$q(x,t) = q(x - ut, 0)$$

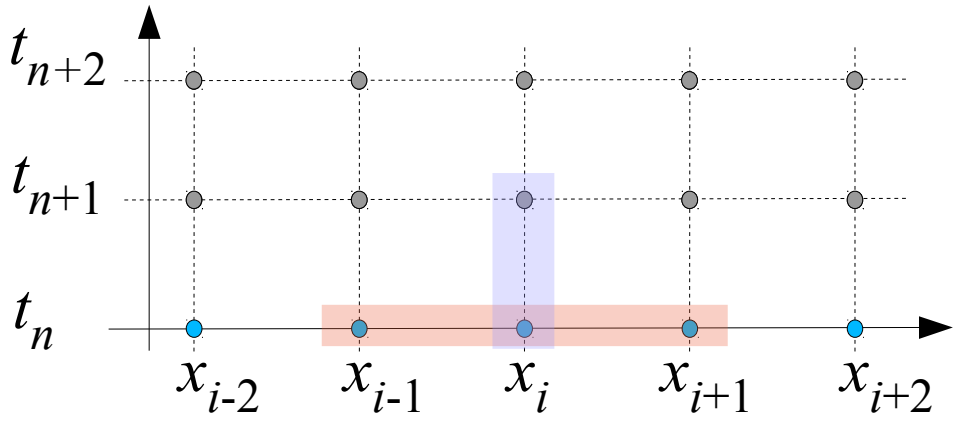


(figure by Kees Dullemond)

Straightforward finite difference discretization schemes for the advection problem can be tried....

CENTERED DIFFERENCE APPROCH FOR THE ADVECTION EQUATION

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



Centered difference discretization:

$$\frac{q_i^{n+1} - q_i^n}{t_{n+1} - t_n} + u \frac{q_{i+1}^n - q_{i-1}^n}{x_{i+1} - x_{i-1}} = 0$$

This yields the update formula:

$$q_i^{n+1} = q_i^n - \frac{\Delta t}{2\Delta x} u (q_{i+1}^n - q_{i-1}^n)$$

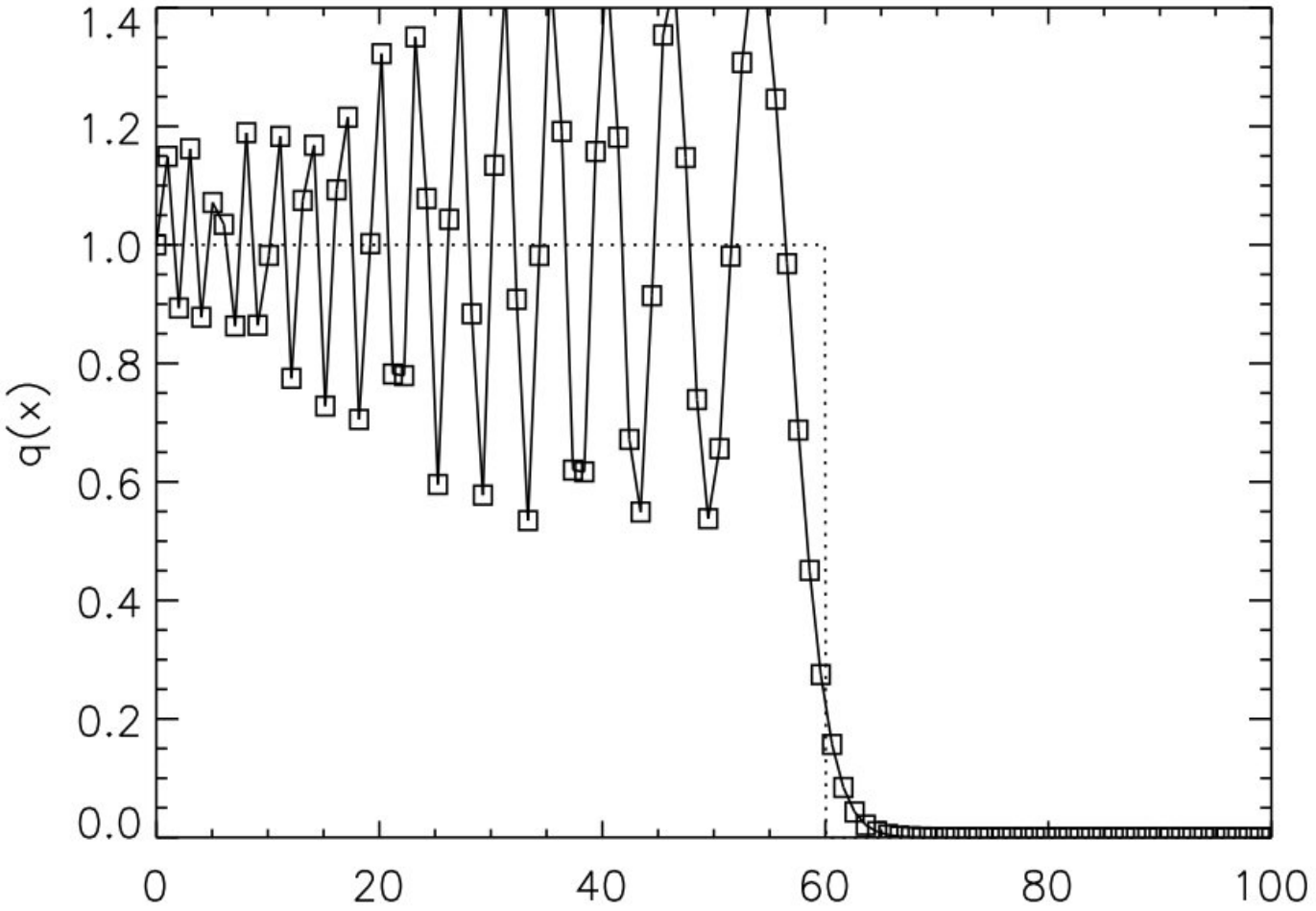
Such attempts tend to fail, often being spectacularly unstable

CENTERED DIFFERENCE ADVECTION OF A STEP FUNCTION

Initial conditions:

$$q(x, t = t_0) = \begin{cases} 1 & \text{for} \quad x \leq 30 \\ 0 & \text{for} \quad x > 30 \end{cases}$$

Evolved state:



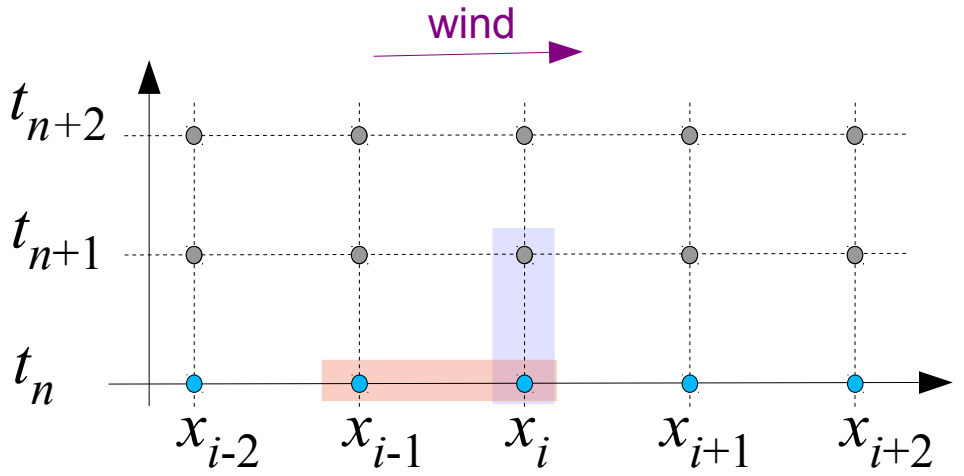
(figure by Kees Dullemond)

Upwinding can render the simple advection algorithm stable

UPWIND DIFFERENCING APPROACH APPLIED TO THE ADVECTION EQUATION

Let's assume:

$$u > 0$$



Now modify the finite difference approximation to be skewed to the upwind side:

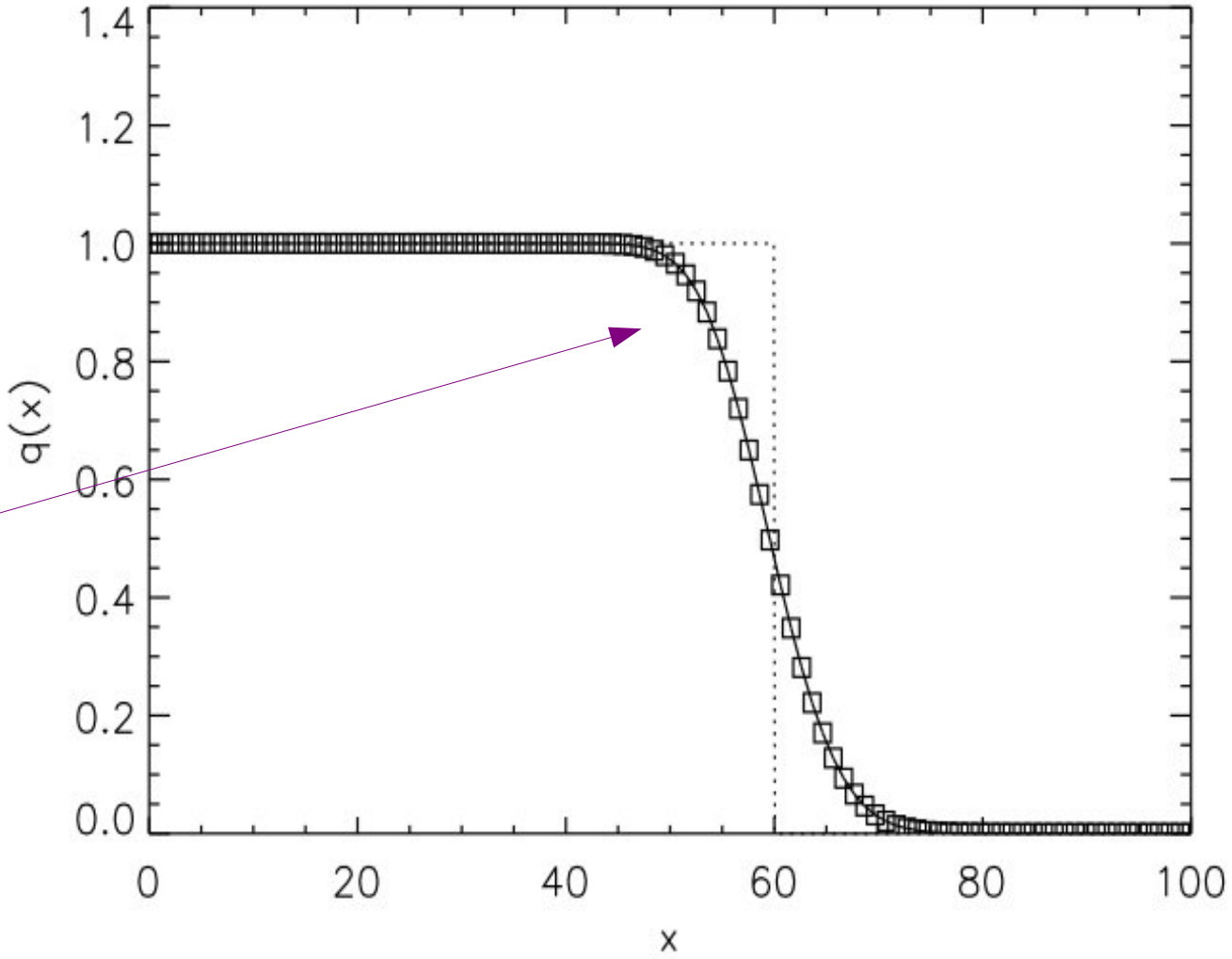
$$\frac{q_i^{n+1} - q_i^n}{t_{n+1} - t_n} + u\frac{q_i^n - q_{i-1}^n}{x_i - x_{i-1}} = 0$$

This yields the update formula:

$$q_i^{n+1} = q_i^n - \frac{\Delta t}{\Delta x} u(q_i^n - q_{i-1}^n)$$

Using upwind differencing, the simple update formula yields stable and robust results

UPWIND ADVECTION OF A STEP FUNCTION



However,
there is considerable
“numerical diffusion”

(figure by Kees Dullemond)

Effectively, upwind differencing adds some diffusion to the central differencing scheme

ANALYSING UPWIND DIFFERENCING

$$\frac{q_i - q_{i-1}}{\Delta x} = \frac{q_{i+1} - q_{i-1}}{2\Delta x} - \Delta x \frac{q_{i+1} - 2q_i + q_{i-1}}{2\Delta x^2}$$

upwind difference

centered difference

diffusion term with diffusion constant

$$D = \frac{\Delta x u}{2}$$

- Once a bit of diffusion is added to the centered scheme, it becomes stable!
- The diffusion vanishes in the limit of $\Delta x \to 0$. It is purely **numerical diffusion**.

The Euler equations as a set of hyperbolic conservation laws

FLUX FORMULATION OF THE EULER EQUATIONS

State vector

$$\mathbf{U} = \begin{pmatrix} \rho \\ \rho\mathbf{v} \\ \rho e \end{pmatrix}$$

$$e = u + \mathbf{v}^2/2$$

Euler equations

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

Flux vector

$$\mathbf{F}(\mathbf{U}) = \begin{pmatrix} \rho\mathbf{v} \\ \rho\mathbf{v}\mathbf{v}^T + P \\ (\rho e + P)\mathbf{v} \end{pmatrix}$$

$$P = (\gamma - 1)\rho u$$

Finite volume discretization:

Cell averages

$$\mathbf{Q}_i = \begin{pmatrix} M_i \\ \mathbf{p}_i \\ E_i \end{pmatrix} = \int_{V_i} \mathbf{U}\, dV$$

Evolution equation

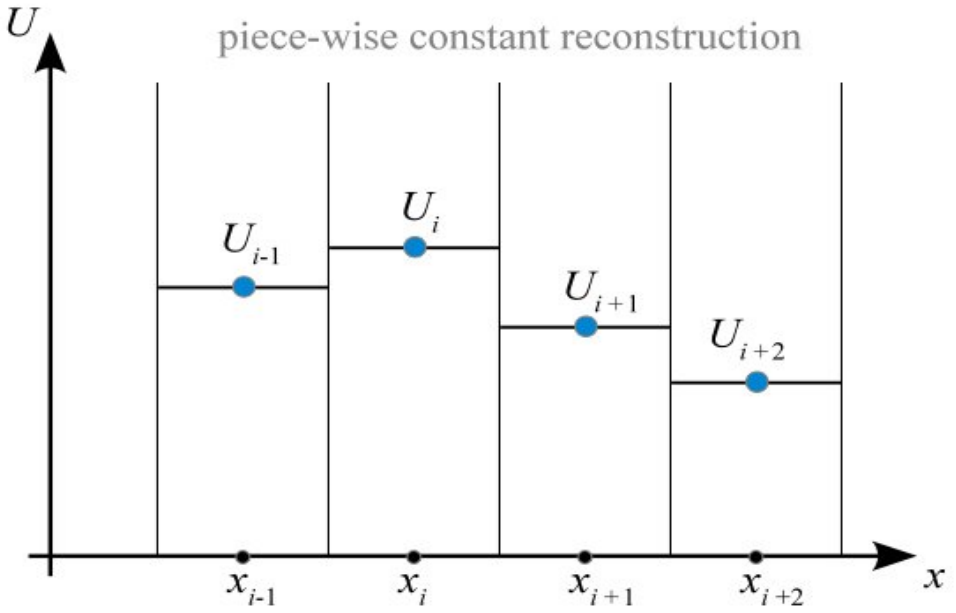
$$\frac{d\mathbf{Q}_i}{dt} = -\int_{\partial V_i} \mathbf{F}(\mathbf{U})\ d\mathbf{n}$$

But how to compute the fluxes through cell surfaces?

The timesteps in many finite volume scheme can be viewed as a sequence of Reconstruct-Evolve-Average (REA) steps

REA SCHEMES

Reconstruct: Using the cell-averaged quantities, determine the run of these quantities everywhere in the cell.

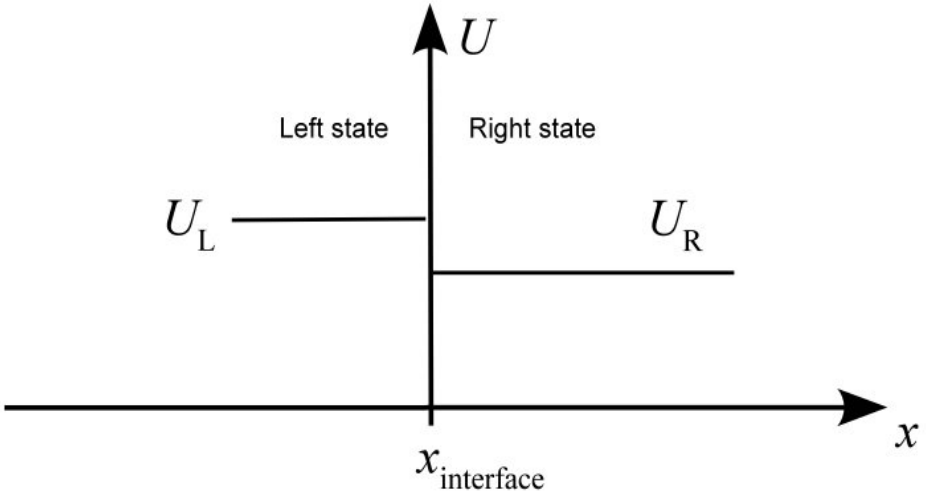


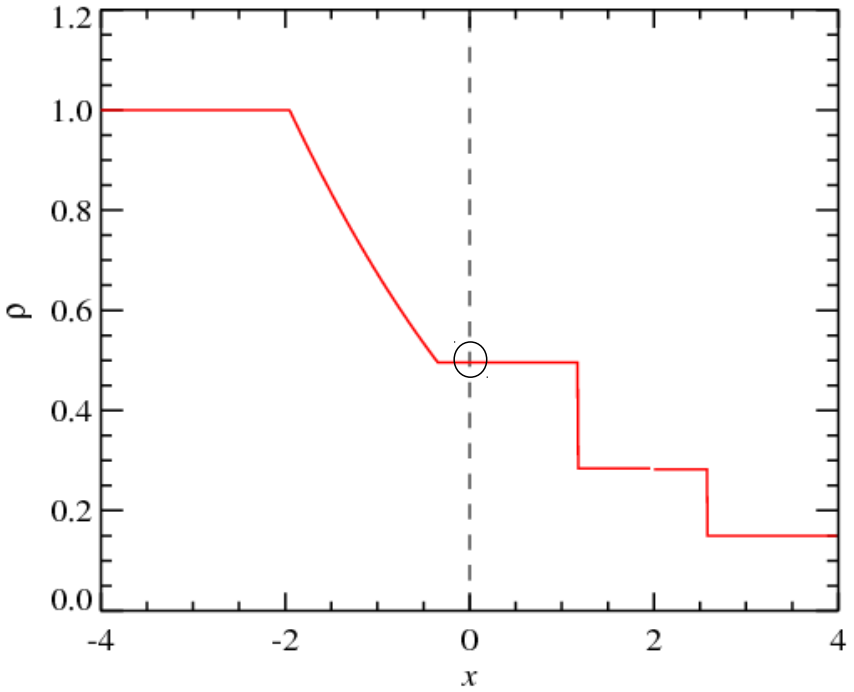
Evolve: The reconstructed state is then evolved forward in time by Δt. In **Godunov's approach**, this is done by treating each cell interface as a piece-wise constant initial value problem which is solved with a **Riemann solver**.

Average: The evolved solution of the previous step is averaged at time Δt to compute new average states U^{n+1} in each cell in a conservative fashion. Then the cycle repeats.

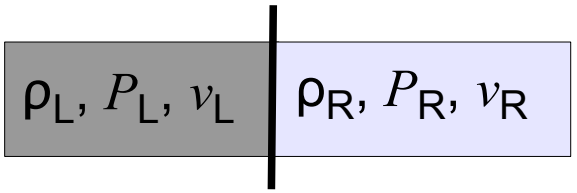
The Riemann problem as basis for Godunov schemes

CALCULATION OF THE GODUNOV FLUX

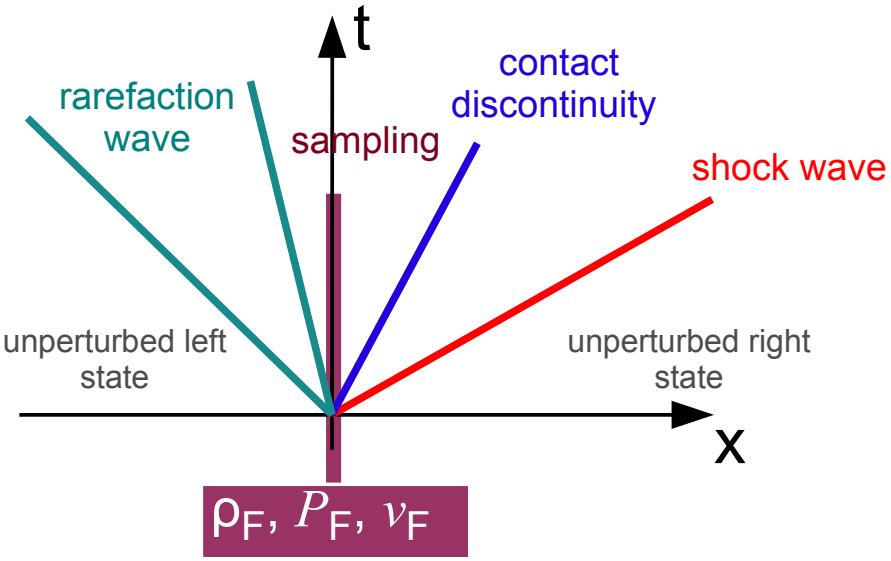




Assume piece-wise constant left and right states for the fluid



Calculate the self-similar time evolution **(Riemann problem)**



Sample the solution along x/ t=0, which yields the **Godunov flux**

Fortunately, the **averaging step** doesn't have to be done explicitly

OBTAINING THE AVERAGED STATE

Let's integrate the Euler equation over a cell and in time:

$$\int_{x_{i-\frac{1}{2}}}^{x_{i+\frac{1}{2}}} \mathrm{d}x \int_{t_n}^{t_{n+1}} \mathrm{d}t \left(\partial_t \mathbf{U} + \partial_x \mathbf{F}\right) = 0$$

This yields:

$$\int_{x_{i-\frac{1}{2}}}^{x_{i+\frac{1}{2}}} \mathbf{U}(x, t_{n+1})\mathrm{d}x - \int_{x_{i-\frac{1}{2}}}^{x_{i+\frac{1}{2}}} \mathbf{U}(x, t_n)\mathrm{d}x + \int_{t_n}^{t_{n+1}} \mathbf{F}(x_{i+\frac{1}{2}}, t)\mathrm{d}t - \int_{t_n}^{t_{n+1}} \mathbf{F}(x_{i-\frac{1}{2}}, t)\mathrm{d}t = 0$$

But the Riemann solution is self-similar, with the Godunov flux being independent of time:

$$\mathbf{F}(x_{i+\frac{1}{2}}, t) = \mathbf{F}^{\star}_{i+\frac{1}{2}} \qquad \mathbf{F}^{\star} = \mathbf{F}_{\text{Riemann}}(\mathbf{U}_L, \mathbf{U}_R)$$

Hence the **new spatially average state** is simply given by:

$$\mathbf{U}_{n+1} = \mathbf{U}_n + \frac{\Delta t}{\Delta x}\left(\mathbf{F}^{\star}_{i-\frac{1}{2}} - \mathbf{F}^{\star}_{i+\frac{1}{2}}\right)$$

Two issues are left open:

- How can this be extended to multiple dimensions?
- How can we reach an accuracy higher than 1^{st} order in space and time?

Operator splitting can be used to extend the scheme to multiple dimensions

FROM 1D TO 3D WITH LITTLE EFFORT

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

Let's write out the full Euler for Cartesian coordinates:

$$\partial_t \begin{pmatrix} \rho \\ \rho u \\ \rho v \\ \rho w \\ \rho e \end{pmatrix} + \partial_x \begin{pmatrix} \rho u \\ \rho u^2 + P \\ \rho uv \\ \rho uw \\ \rho u(\rho e + P) \end{pmatrix} + \partial_y \begin{pmatrix} \rho v \\ \rho uv \\ \rho v^2 + P \\ \rho vw \\ \rho v(\rho e + P) \end{pmatrix} + \partial_z \begin{pmatrix} \rho w \\ \rho uw \\ \rho vw \\ \rho w^2 + P \\ \rho w(\rho e + P) \end{pmatrix} = 0$$

This can be written also in the following form:

$$\mathbf{v} = (u, v, w)$$

$$\partial_t \mathbf{U} + \partial_x \mathbf{F} + \partial_y \mathbf{G} + \partial_z \mathbf{H} = 0$$

Now let's split up the equation into three separate equations:

$\partial_t \mathbf{U} + \partial_x \mathbf{F} = 0$ $\quad \mathcal{X}(\Delta t)$ generates evolution under F

$\partial_t \mathbf{U} + \partial_y \mathbf{G} = 0$ $\quad \mathcal{Y}(\Delta t)$

$\partial_t \mathbf{U} + \partial_z \mathbf{H} = 0$ $\quad \mathcal{Z}(\Delta t)$

Approximate solution can be obtained as:

$$\mathbf{U}^{n+1} \simeq \mathcal{Z}(\Delta t)\mathcal{Y}(\Delta t)\mathcal{X}(\Delta t)\mathbf{U}^n$$

The sequence of dimensionally split sweeps must be alternated to reach higher-order accuracy in time

TIME-ADVANCE IN DIMENSIONALLY SPLIT SCHEMES

Simple operator split in two dimensions:

$$\mathbf{U}^{n+1} \simeq \mathcal{Y}(\Delta t)\mathcal{X}(\Delta t)\mathbf{U}^n$$

For second-order accuracy in time, symmetrize the action of the operators:

$$\mathbf{U}^{n+1} = \frac{1}{2}[\mathcal{X}(\Delta t)\mathcal{Y}(\Delta t) + \mathcal{Y}(\Delta t)\mathcal{X}(\Delta t)]\mathbf{U}^n$$

One may also, e.g., use:

$$\mathbf{U}^{n+1} = \mathcal{X}(\Delta t/2)\mathcal{Y}(\Delta t)\mathcal{X}(\Delta t/2)\mathbf{U}^n$$

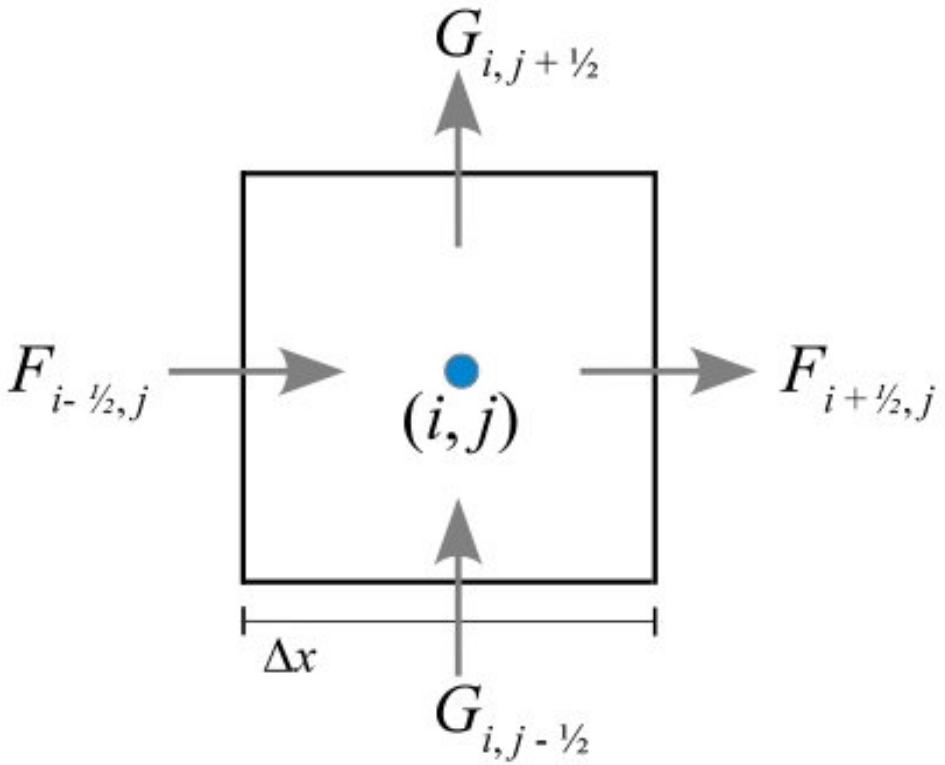
This readily generalizes to three dimensions:

$$\mathbf{U}^{n+2} = \mathcal{X}(\Delta t)\mathcal{Y}(\Delta t)\mathcal{Z}(\Delta t)\mathcal{Z}(\Delta t)\mathcal{Y}(\Delta t)\mathcal{X}(\Delta t)\mathbf{U}^n$$

$$\mathbf{U}^{n+1} = \mathcal{X}(\Delta t/2)\mathcal{Y}(\Delta t/2)\mathcal{Z}(\Delta t)\mathcal{Y}(\Delta t/2)\mathcal{X}(\Delta t/2)\mathbf{U}^n$$

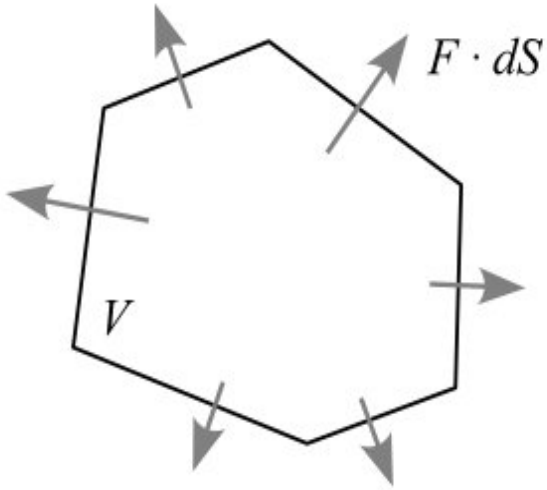
One can also calculate the fluxes directly in multi-D and arrive at an *unsplit* scheme

UNSPLIT FINITE-VOLUME UPDATES



Unstructured mesh case

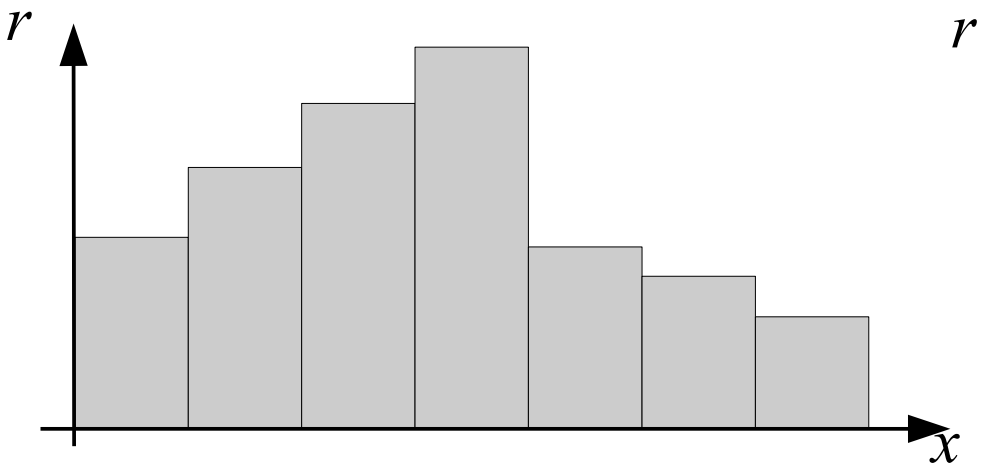
$$\mathbf{U}^{n+1} = \mathbf{U}^{n} - \frac{\Delta t}{V}\int \mathbf{F}\cdot\mathbf{dS}$$

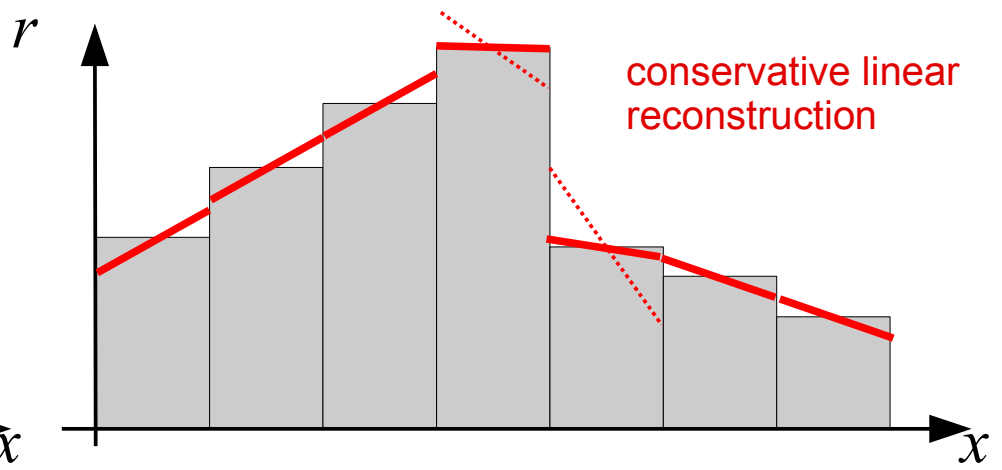


$$U_{i,j}^{n+1} = U_{i,j}^{n} + \frac{\Delta t}{\Delta x}\left(\mathbf{F}_{i-\frac{1}{2},j} - \mathbf{F}_{i+\frac{1}{2},j}\right) + \frac{\Delta t}{\Delta y}\left(\mathbf{G}_{i,j-\frac{1}{2}} - \mathbf{G}_{i,j+\frac{1}{2}}\right)$$

To achieve second-order accuracy, one can use a **piece-wise linear** reconstruction

LINEAR RECONSTRUCTION AND GRADIENT LIMITATION





Slope limiting procedure:

$$\langle \nabla \phi \rangle_i' = \alpha_i \langle \nabla \phi \rangle_i$$

- Needed to prevent creation of new extrema, preventing spurious oscillations
- Reduces the order of the scheme at discontinuities

Example slope limiter:

$$\alpha_i = \min(1, \psi_{ij})$$

$$\psi_{ij} = \begin{cases} (\phi_i^{\max} - \phi_i)/\Delta\phi_{ij} & \text{for} \quad \Delta\phi_{ij} > 0 \\ (\phi_i^{\min} - \phi_i)/\Delta\phi_{ij} & \text{for} \quad \Delta\phi_{ij} < 0 \\ 1 & \text{for} \quad \Delta\phi_{ij} = 0 \end{cases}$$

$$\phi_i^{\max} = \max(\phi_j) \qquad \phi_i^{\min} = \max(\phi_j)$$

$$\Delta\phi_{ij} = \langle \nabla \phi \rangle_i \cdot (\boldsymbol{f}_{ij} - \boldsymbol{s}_i)$$

The gradients can be used to predict the fluid state directly at the interfaces

LINEAR EXTRAPOLATION TO CELL BOUNDARIES

Spatial extrapolation:

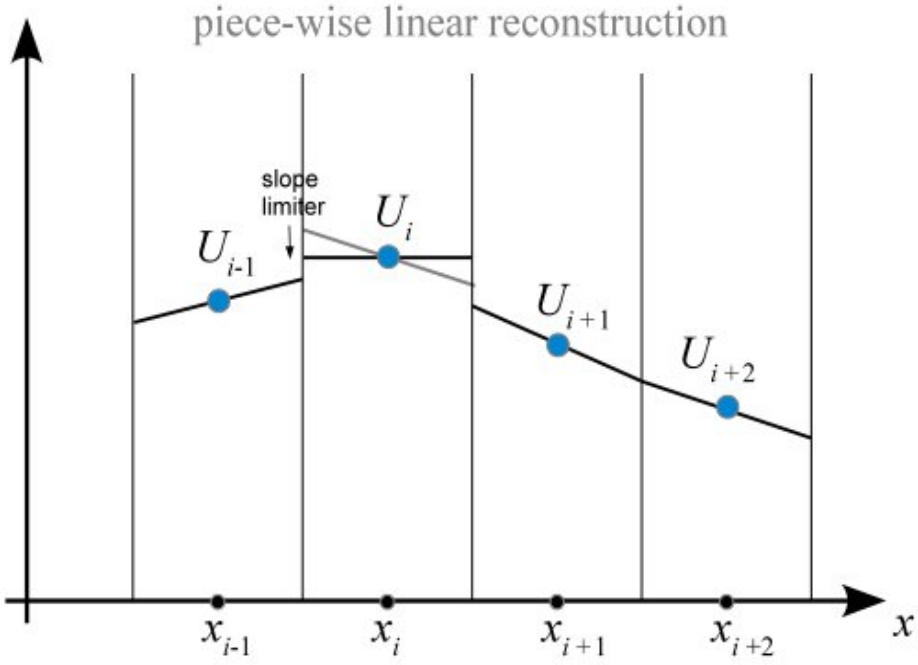
$$\rho^L_{i+\frac{1}{2}} = \rho_i + (\nabla\rho)_i \frac{\Delta x}{2},$$

$$\rho^R_{i+\frac{1}{2}} = \rho_{i+1} - (\nabla\rho)_{i+1} \frac{\Delta x}{2}$$

Temporal extrapolation to mid-step:

$$\rho^L_{i+\frac{1}{2}} = \rho_i + (\nabla\rho)_i \frac{\Delta x}{2} + \left(\frac{\partial\rho}{\partial t}\right)_i \frac{\Delta t}{2}$$

$$\rho^R_{i+\frac{1}{2}} = \rho_{i+1} - (\nabla\rho)_{i+1} \frac{\Delta x}{2} + \left(\frac{\partial\rho}{\partial t}\right)_{i+1} \frac{\Delta t}{2}$$



This combined extrapolation needs to be done for the full fluid state for second-order accuracy:

$$\mathbf{U}^L_{i+\frac{1}{2}} = \mathbf{U}_i + (\partial_x \mathbf{U})_i \frac{\Delta x}{2} + (\partial_t \mathbf{U})_i \frac{\Delta t}{2}$$

$$\mathbf{U}^R_{i+\frac{1}{2}} = \mathbf{U}_{i+1} - (\partial_x \mathbf{U})_{i+1} \frac{\Delta x}{2} + (\partial_t \mathbf{U})_{i+1} \frac{\Delta t}{2}$$

But how do we get the time derivative?

The MUSCL-Hancock scheme is a particularly simple second-order extension of Godunov's method

GETTING THE TIME EXTRAPOLATION FROM THE SPATIAL GRADIENTS

Euler equation:

$$\partial_t \mathbf{U} = -\partial_x \mathbf{F}(\mathbf{U}) = -\frac{\partial \mathbf{F}}{\partial \mathbf{U}} \partial_x \mathbf{U} = -\mathbf{A}(\mathbf{U})\, \partial_x \mathbf{U}$$

This means we can estimate the time derivate based on the current state of a cell and the spatial gradient estimates:

$$(\partial_t \mathbf{U})_i = -\mathbf{A}(\mathbf{U}_i)\, (\partial_x \mathbf{U})_i$$

gradient estimate

MUSCL-Hancock prediction:

$$\begin{aligned}
\mathbf{U}^L_{i+\frac{1}{2}} &= \mathbf{U}_i + \left[\frac{\Delta x}{2} - \frac{\Delta t}{2}\mathbf{A}(\mathbf{U}_i)\right](\partial_x \mathbf{U})_i, \\
\mathbf{U}^R_{i+\frac{1}{2}} &= \mathbf{U}_{i+1} + \left[-\frac{\Delta x}{2} - \frac{\Delta t}{2}\mathbf{A}(\mathbf{U}_{i+1})\right](\partial_x \mathbf{U})_{i+1}
\end{aligned}$$

- At discontinuities, the slope limiter will suppress the derivative, making the scheme automatically first order
- We recall also Godunov's theorem, according to which *any linear algorithm that does not produce new extrema (aka TVD) can be at most first order*.

Thermal energy, temperature and entropy are treated in a fundamentally different way in Eulerian codes compared to SPH

QUESTIONS ABOUT ENTROPY AND FRIENDS IN EULERIAN CODES

Why is there no artificial viscosity needed in the MUSCL scheme?

How is entropy produced?

Some notes may help to clarify this:

- Temperature is defined as difference between total energy and kinetic energy (note: *cold flow problem*)
- Entropy is not followed explicitly – one simply assigns the entropy according to what the conservation laws dictate
- The break-down of the differential form of the equations at shocks is circumvented by using the integrated flux-form of the equations

Approximate Riemann solvers are often used for computational efficiency

EXAMPLE: ROE SOLVER FOR ISOTHERMAL GAS IN TWO DIMENSIONS

2d state vector for isothermal gas:

$$\mathbf{U} = \begin{pmatrix} \rho \\ \rho u \\ \rho v \end{pmatrix}$$

The x-sweep part of the equations is:

$$\partial_t \mathbf{U} + \partial_x \mathbf{F} = 0 \qquad \mathbf{F} = \begin{pmatrix} \rho u \\ \rho u^2 + \rho c_s^2 \\ \rho u v \end{pmatrix}$$

We want to solve for the x-sweep:

$$\mathbf{U}_i^{n+1} = \mathbf{U}_i^n + \frac{\Delta t}{\Delta x}\left[\mathbf{F}_{i-\frac{1}{2}} - \mathbf{F}_{i+\frac{1}{2}}\right]$$

To derive an approximate Riemann solver, we want to linearize the equation:

$$\partial_t \mathbf{U} + \mathbf{A}(\mathbf{U})\,\partial_x \mathbf{U} = 0 \qquad \mathbf{A} = \frac{\partial \mathbf{F}}{\partial \mathbf{U}}$$

Roe's suggestion:

$$\tilde{\mathbf{A}} = \tilde{\mathbf{A}}(\mathbf{U}_L, \mathbf{U}_R)$$

$$\tilde{\mathbf{A}} = \begin{pmatrix} 0 & 1 & 0 \\ c_s^2 - \tilde{u}^2 & 2\tilde{u} & 0 \\ -\tilde{u}\tilde{v} & \tilde{v} & \tilde{u} \end{pmatrix}$$

The Jacobian is approximated as being constant and only a function of left and right states.

$$\tilde{u} = (\sqrt{\rho_L} u_L + \sqrt{\rho_R} u_R)/(\sqrt{\rho_L} + \sqrt{\rho_R})$$

Eigenstructure of the linearized isothermal equations:

Eigenvalues:

$$\lambda_1 = \tilde{u} - c_s, \qquad \lambda_2 = \tilde{u} + c_s \qquad \lambda_3 = \tilde{u}$$

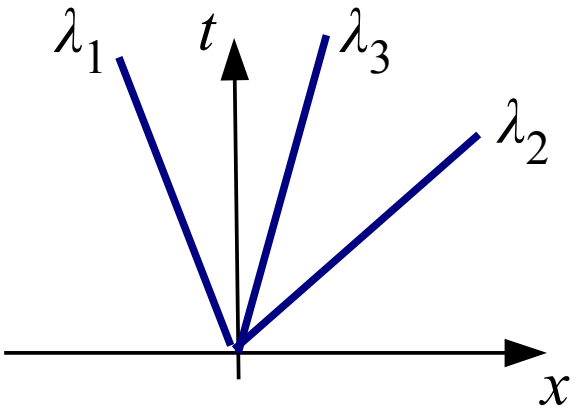
Eigenvectors:

$$\mathbf{K}_1 = \begin{pmatrix} 1 \\ \tilde{u} - c_s \\ \tilde{v} \end{pmatrix} \qquad \mathbf{K}_2 = \begin{pmatrix} 1 \\ \tilde{u} + c_s \\ \tilde{v} \end{pmatrix} \qquad \mathbf{K}_3 = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$$

Expand the jump in states in the eigenbasis:

$$\Delta \mathbf{U} = (u_1, u_2, u_3) = \mathbf{U}_R - \mathbf{U}_L$$

$$\Delta \mathbf{U} = \sum_i \alpha_i \mathbf{K}_i$$



Linearized solution is easily obtained.

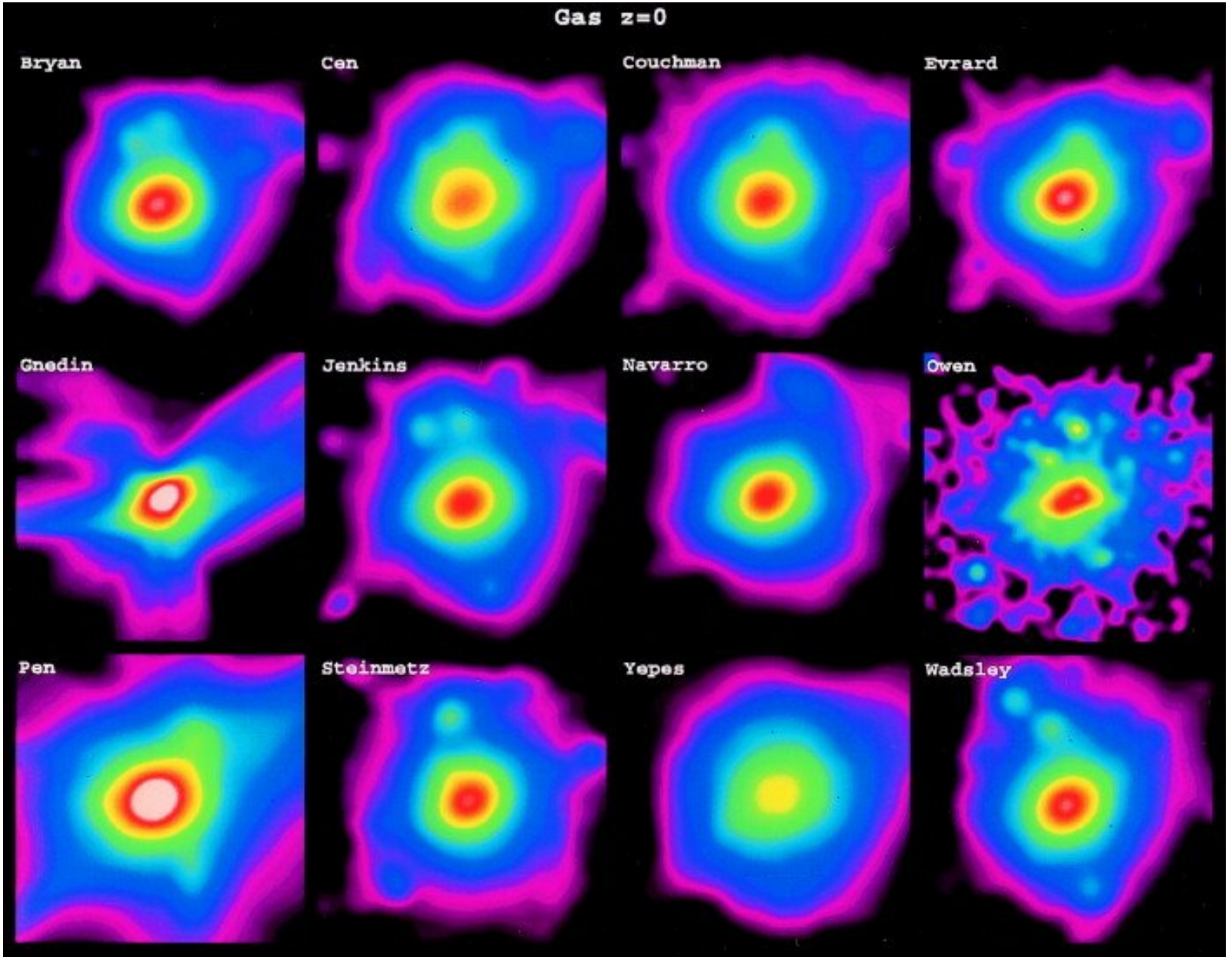
Roe's flux: $$\mathbf{F}^\star = \frac{1}{2}(\mathbf{F}_L + \mathbf{F}_R) - \frac{1}{2}\sum_i \alpha_i \left|\lambda_i\right| \mathbf{K}_i$$

Accuracy issues in cosmological simulations

Different hydrodynamical simulation codes are broadly in agreement, but show substantial scatter and differences in detail

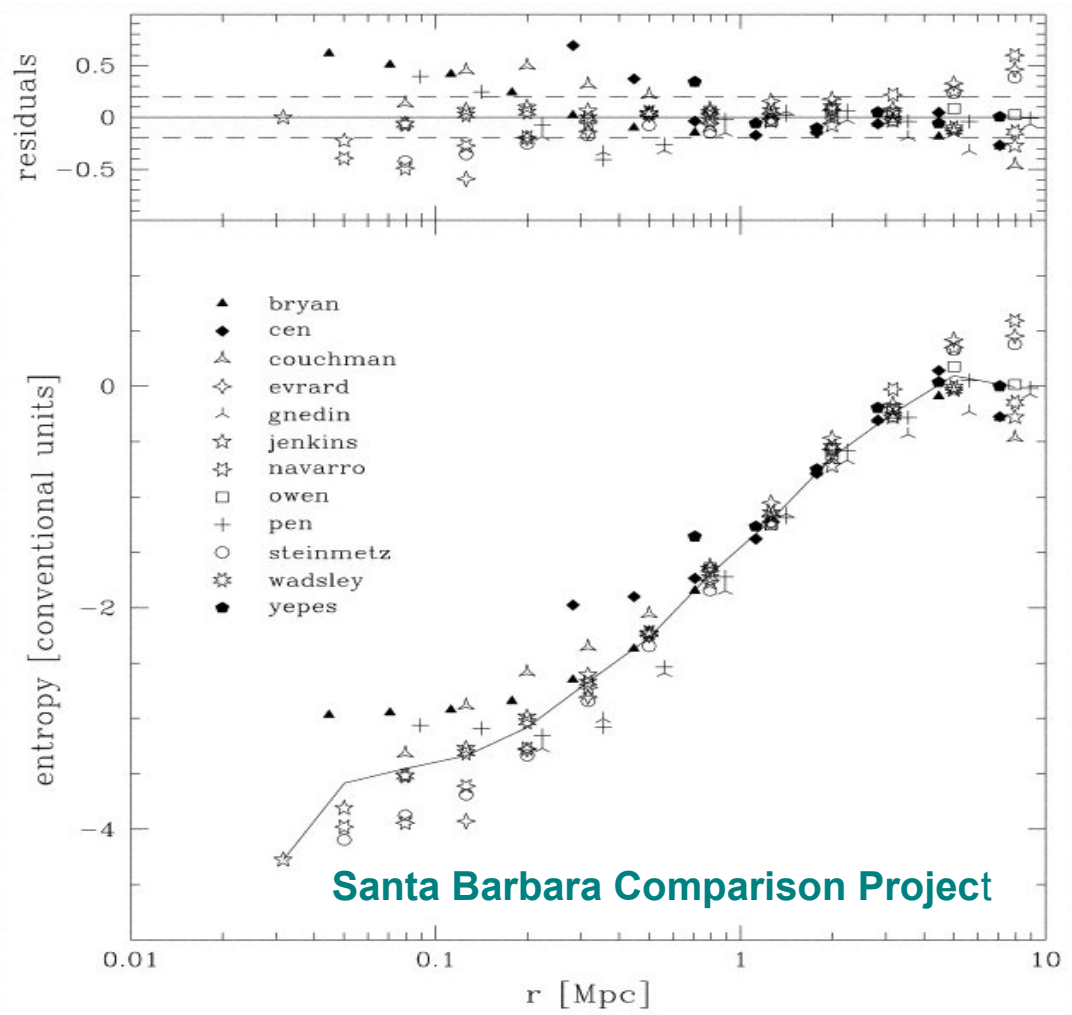
THE SANTA BARBARA CLUSTER COMPARISON PROJECT

Frenk, White & 23 co-authors (1999)



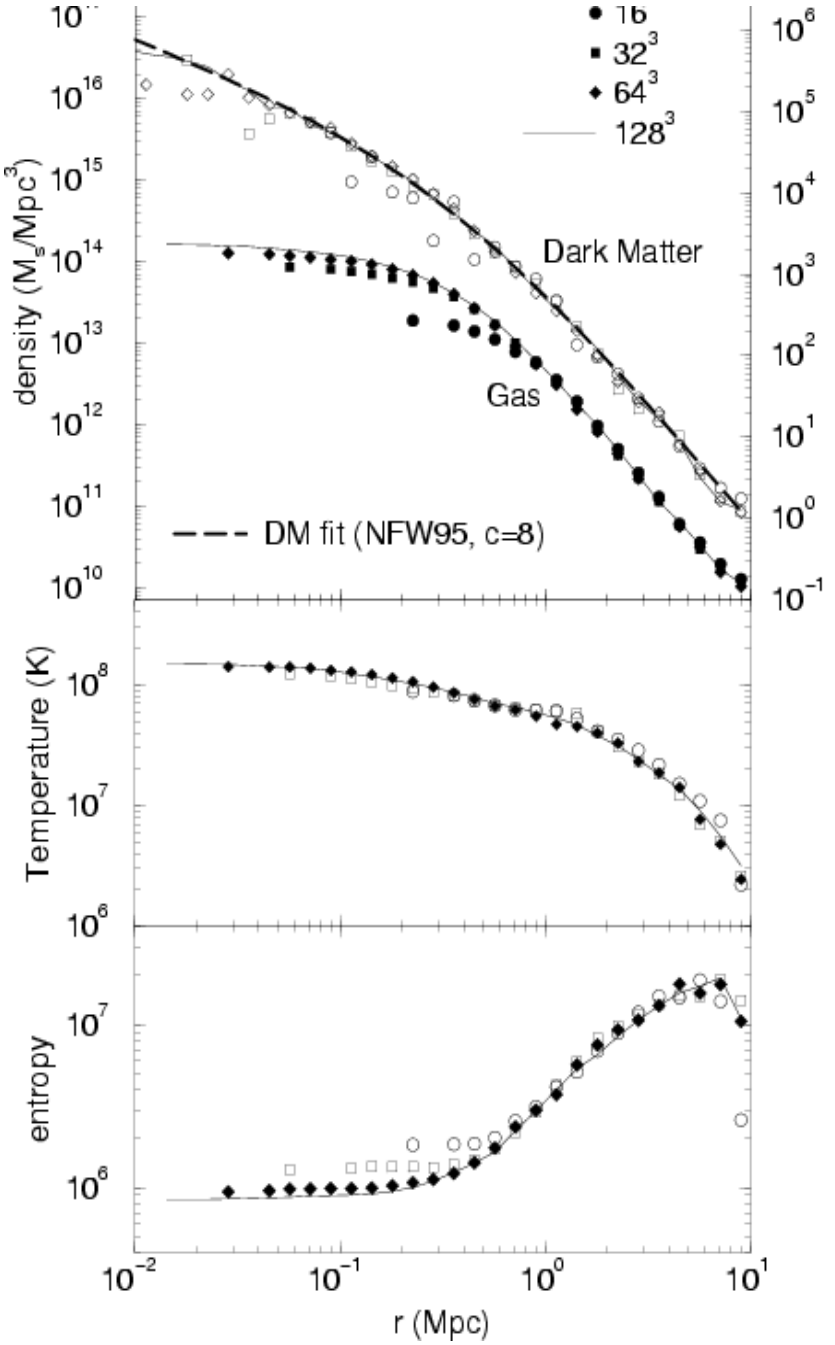
Mesh codes appear to produce higher entropy in the cores of clusters

RADIAL ENTROPY PROFILE



Santa Barbara Comparison Project

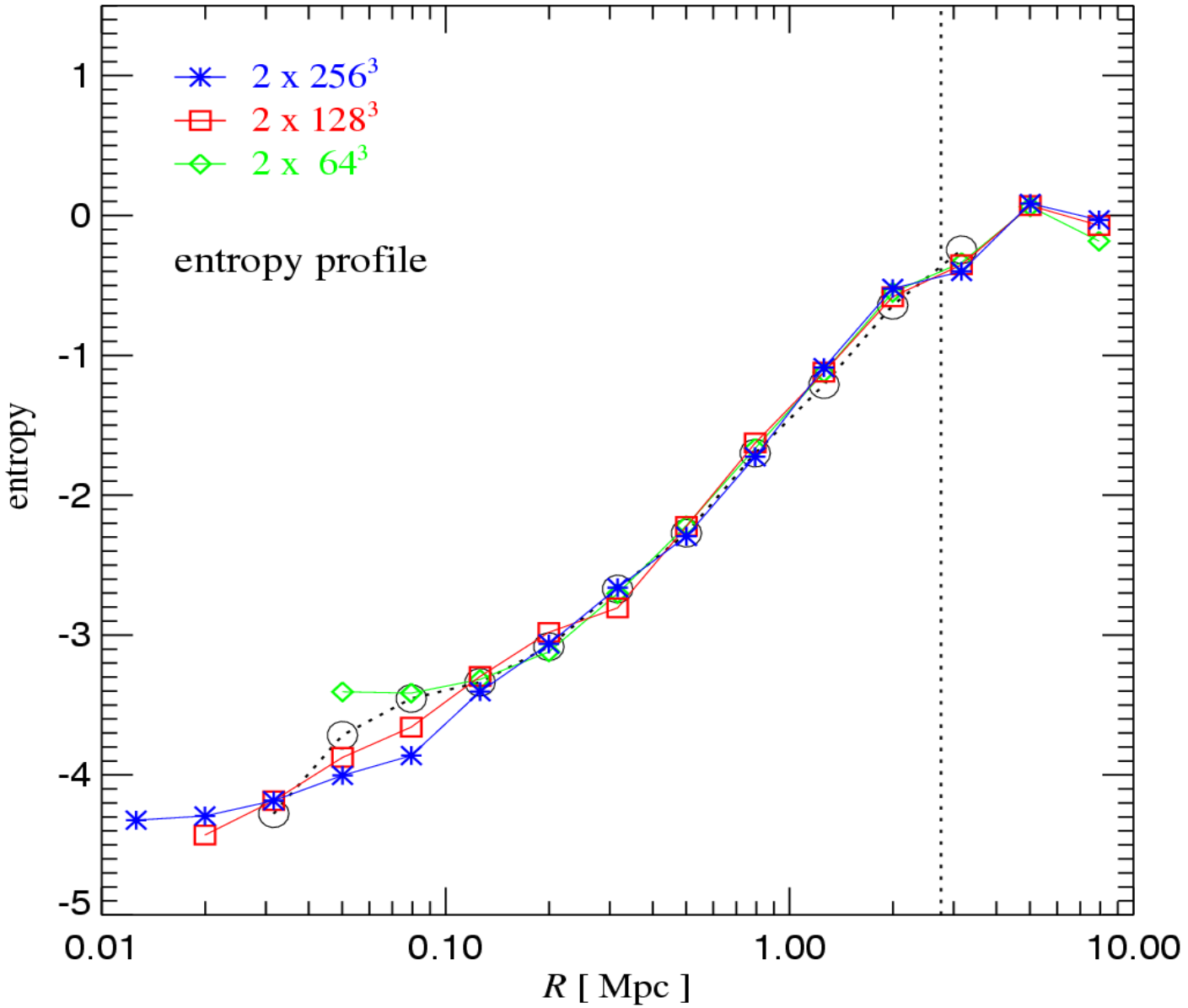
Bryan & Norman 1997



Ascasibar, Yepes, Müller & Gottlöber (2003):
Entropy formulation of SPH also gives somewhat higher core entropy

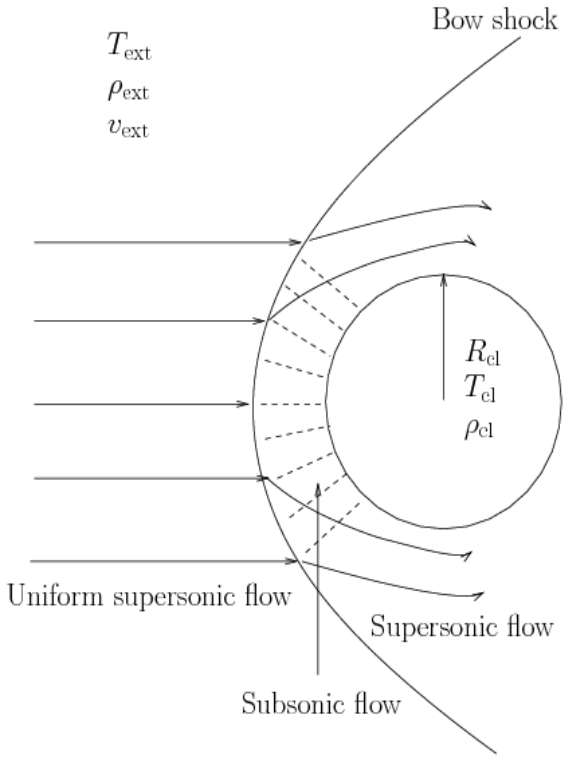
The entropy profile of the Santa Barbara cluster appears to converge well with SPH, yielding a lower level in the center than found with mesh codes

ENTROPY PROFILES OBTAINED WITH GADGET2 AT DIFFERENT RESOLUTION

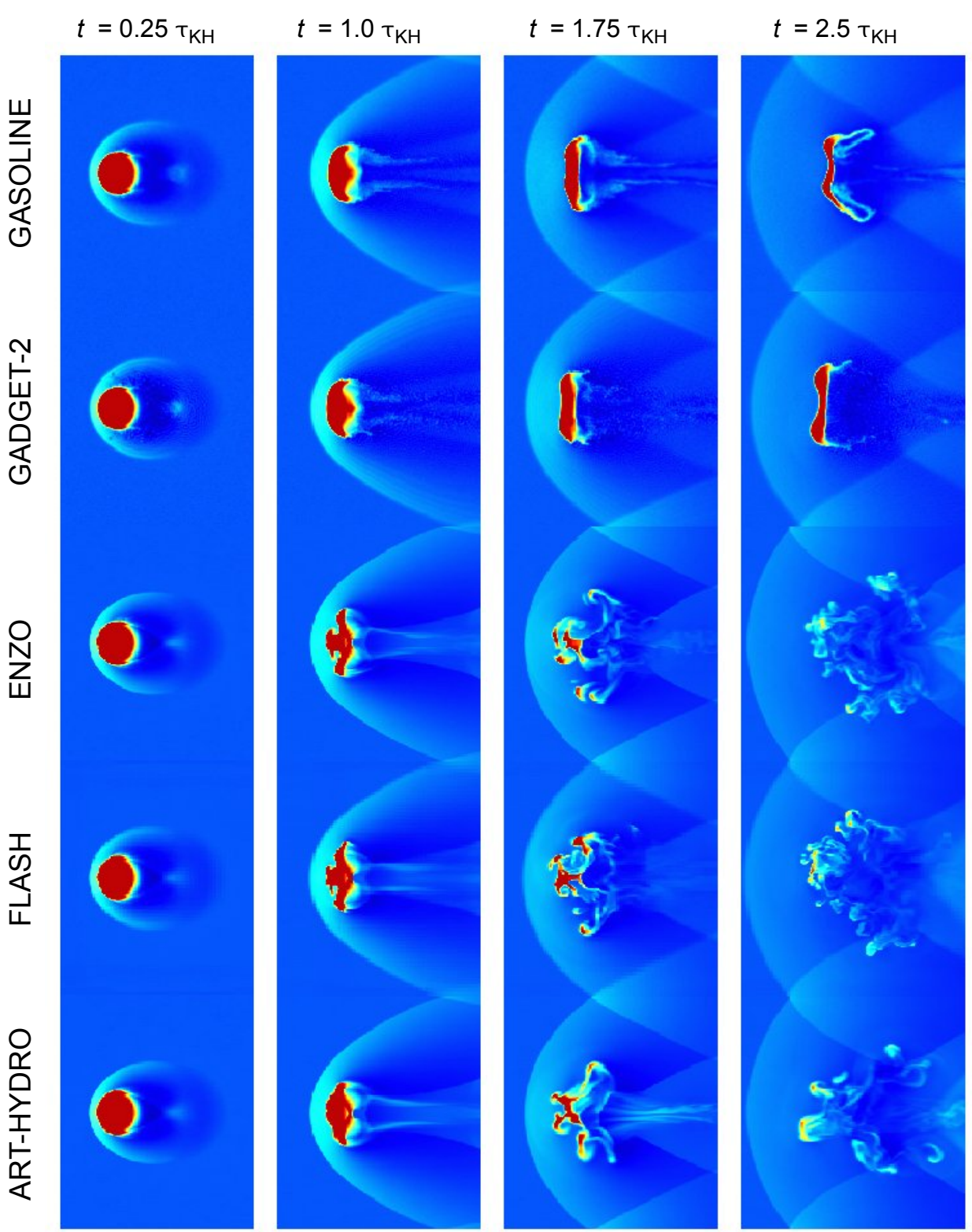


A cloud moving through ambient gas shows markedly different long-term behavior in SPH and Eulerian mesh codes

DISRUPTION OF A CLOUD BY KELVIN-HELMHOLTZ INSTABILITIES

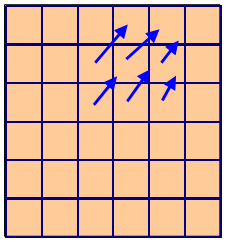


Agertz et al. (2007)

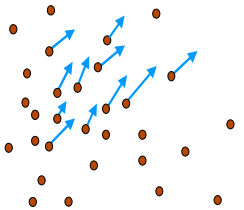


There are principal differences between SPH and Eulerian schemes

SOME FUNDAMENTAL DIFFERENCE BETWEENS SPH AND MESH-HYDRODYNAMICS



Eulerian	Lagrangian
sharp shocks, somewhat less sharp contact discontinuities (best schemes resolve fluid discontinuities it in one cell)	**shocks broadened over roughly 2-3 smoothing lengths** (post-shock properties are correct though)
mixing happens implicitly at the cell level (but advection adds numerical diffusivity and may provide a source of spurious entropy)	**mixing entirely suppressed at the particle-level** (no spurious entropy production, but fluid instabilities may be suppressed)
no need for artificial viscosity (in Godunov schemes)	**requires artificial viscosity**
Truncation error not Galilean invariant ***("high Mach number problem")***	**Galilean invariant**
self-gravity of the gas done on a mesh (but dark matter must still be represented by particles) **no explicit conservation of total energy when self-gravity is included**	**self-gravity of the gas naturally treated with the same accuracy as the dark matter, total energy conserved**



A moving-mesh Lagrangian finite volume code can combine the advantages of SPH and Eulerian methods

KELVIN-HELMHOLTZ INSTABILITY WITH A MOVING MESH CODE

AREPO Code

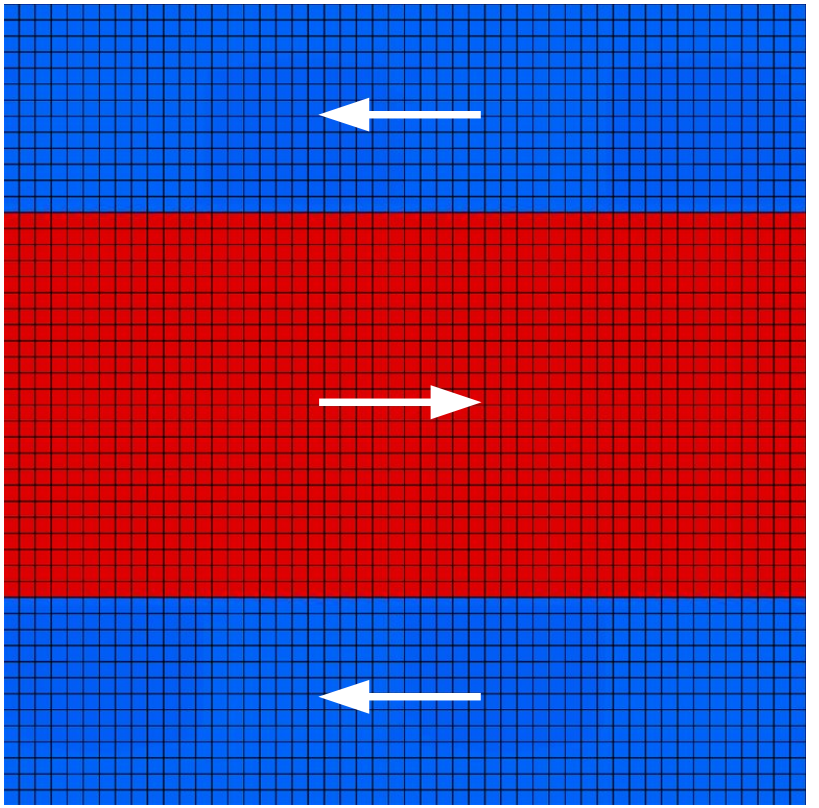
Springel (2010)

$\rho = 1$
$v_x = -0.5$
$P = 2.5$

$\rho = 2$
$v_x = 0.5$
$P = 2.5$

$\rho = 1$
$v_x = -0.5$
$P = 2.5$

periodic boundaries

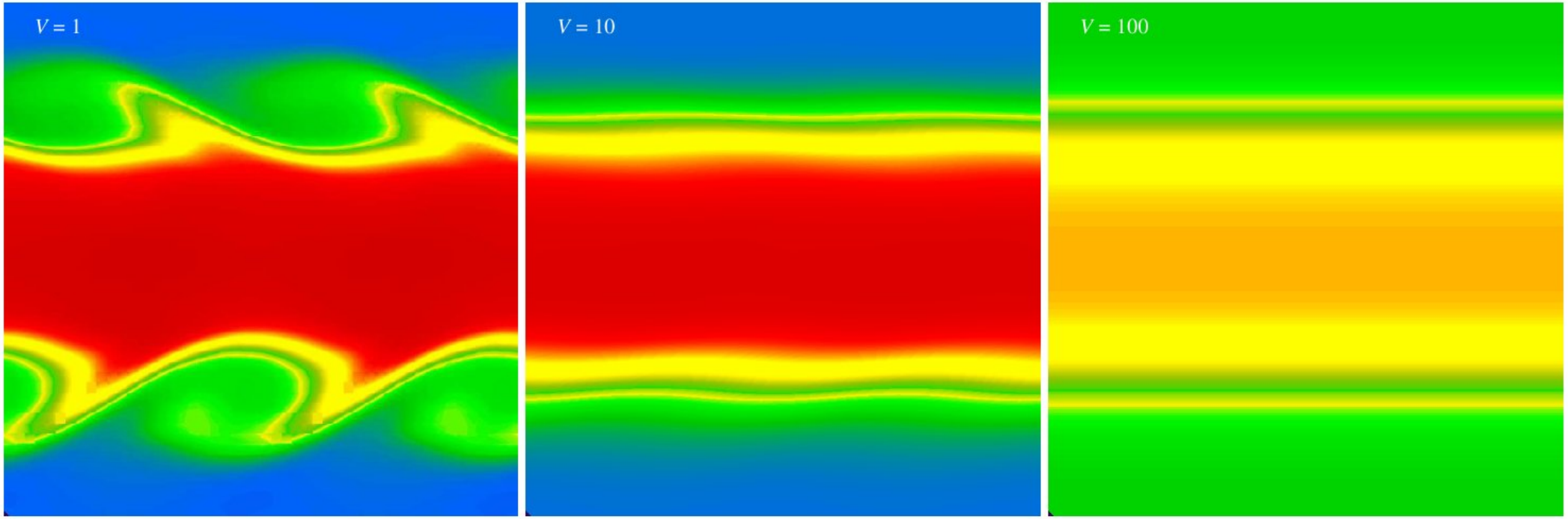


50x50 resolution

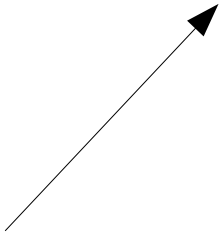
When the mesh is fixed, the results may change if a bulk velocity is imposed

KELVIN-HELMHOLTZH INSTABILITY AT 50 x 50 RESOLUTION WITH A FIXED MESH FOR DIFFERENT GALILEI BOOSTS

This was started from a sharp initial contact discontinuity.



Boost both in x- and y- directions



The truncation error in Eulerian codes is not Galilean invariant.

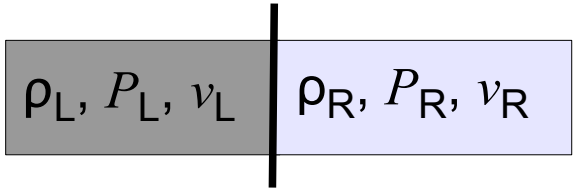
With enough cells, the truncation error can always be reduced, so that for properly resolved initial conditions, effective Galilean invariance is reached.

Nevertheless, this is an unwanted feature that is problematic for simulations of cosmological structure formation. Here the accuracy with which individual galaxies are modeled depends on their velocity magnitude.

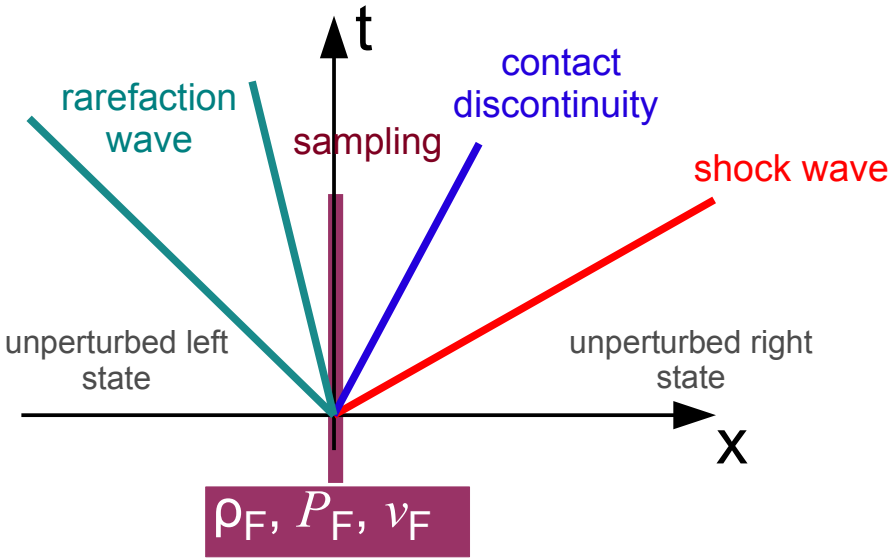
The Riemann problem as basis for high-accuracy Godunov schemes

CALCULATION OF THE GODUNOV FLUX

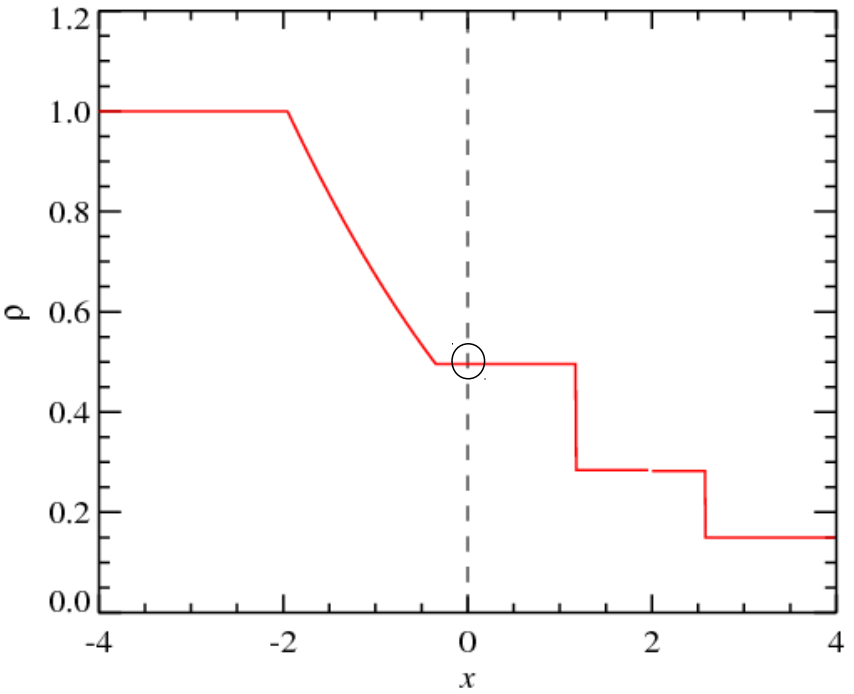
Assume piece-wise constant left and right states for the fluid



Calculate the self-similar time evolution (Riemann problem)



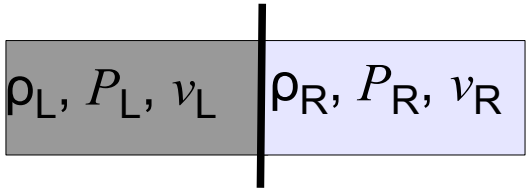
Sample the solution along x/ t=0, which yields the Godunov flux

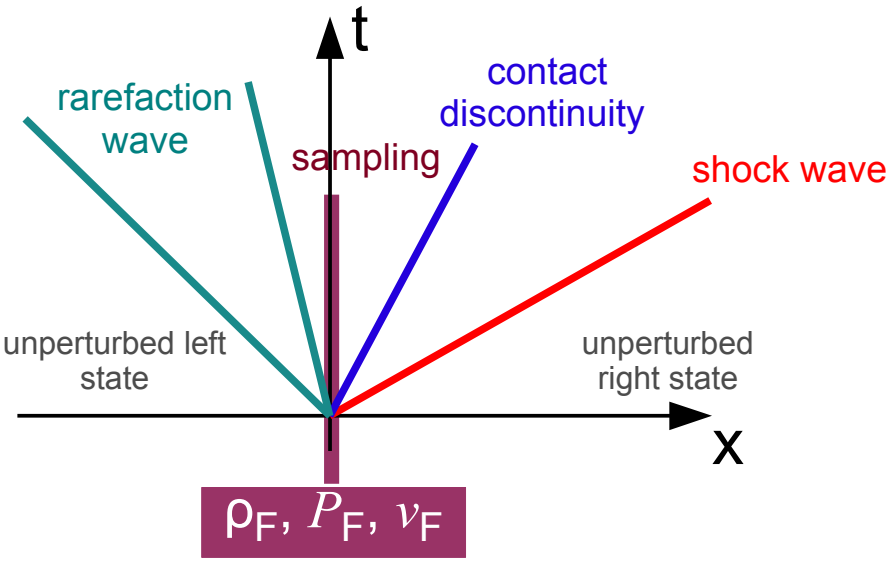


The “upwind side” of the flow depends on the frame of reference

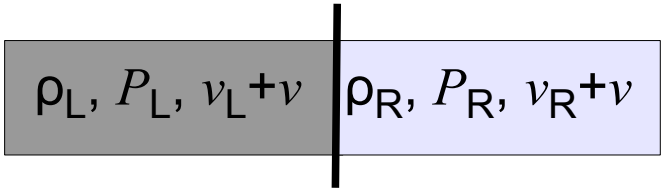
THE GODUNOV FLUX IN DIFFERENT REFERENCE FRAMES

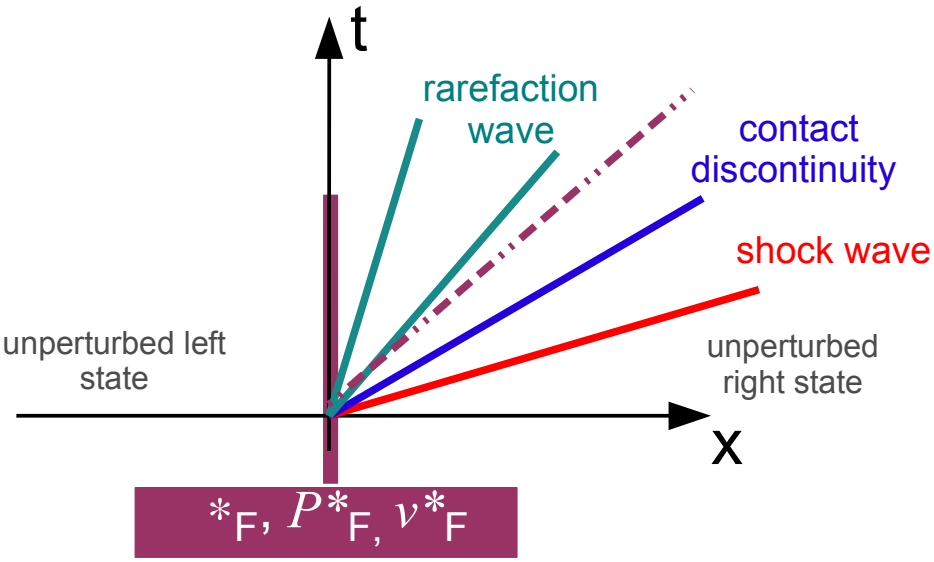
Riemann problem in default frame





Riemann problem in boosted frame





expected mass flux in boosted frame:

$\rho_F\,(v_F+v)$

BUT, in general: $\rho_F\,(v_F+v) \neq {}^*_F\,v^*_F$

→ **Numerical scheme not manifestly Galilean invariant**

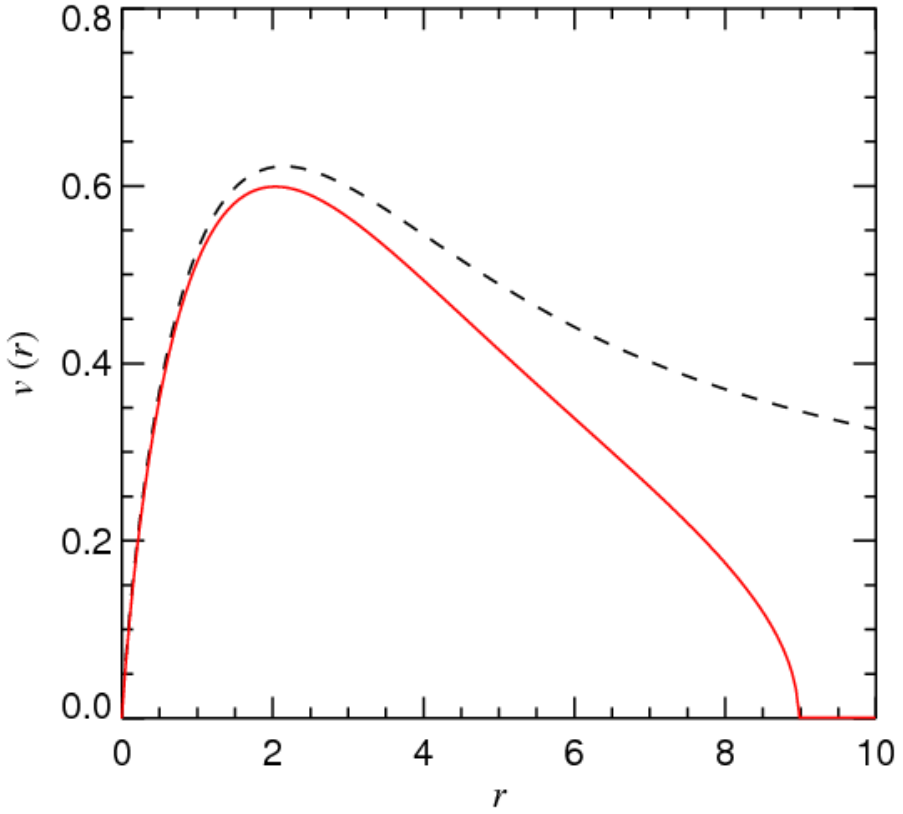
How well does this work?

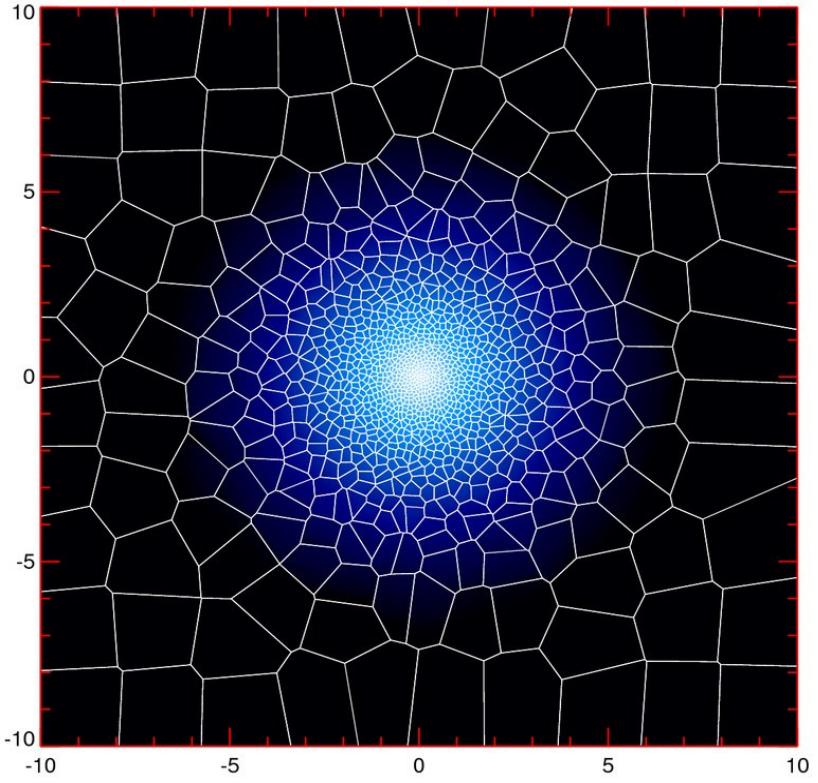
A differentially rotating gaseous disk with strong shear can be simulated well with the moving mesh code

MODEL FOR A CENTRIFUGALLY SUPPORTED, THIN DISK

$$\Sigma(r) = \Sigma_0 \exp(-r/h)$$

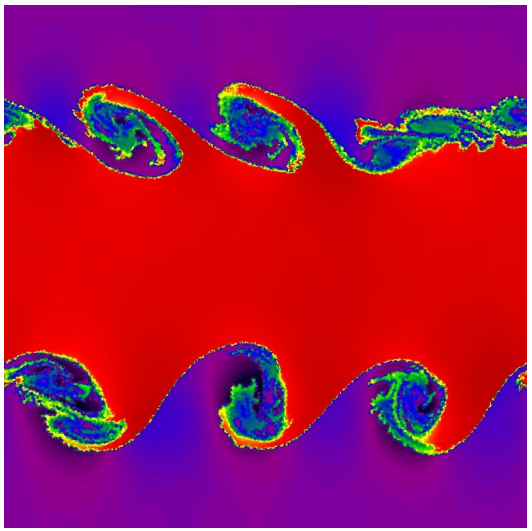
$$v_c^2(r) \equiv r\frac{\partial \Phi}{\partial r} = 2\frac{Gm}{h}y^2\left[I_0(y)K_0(y) - I_1(y)K_1(y)\right]$$



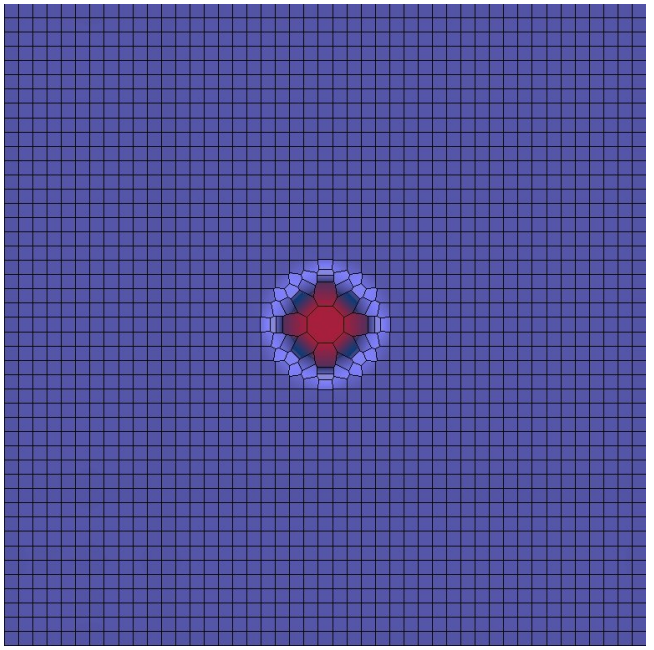


Different examples of test problems with the moving-mesh code

High-resolution
Kelvin-Helmholtz instability

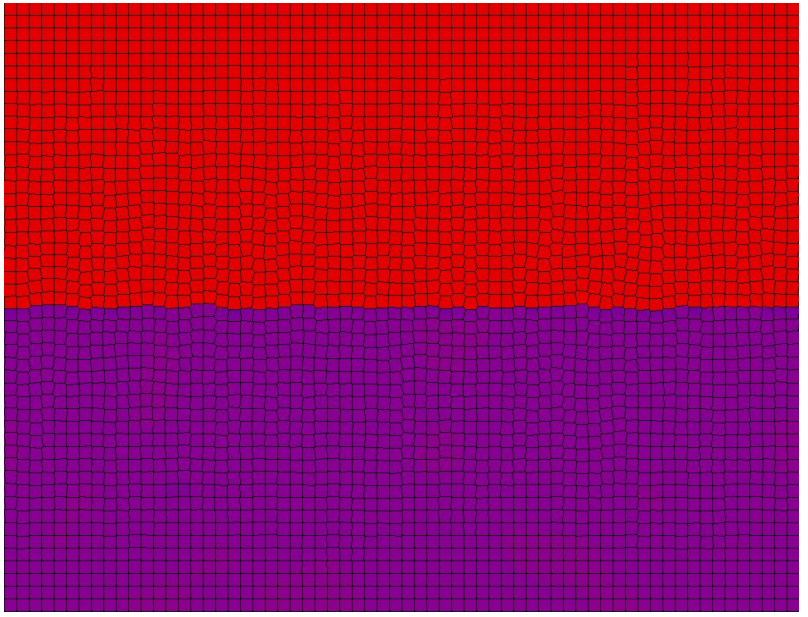


Sedov-Taylor Exposion



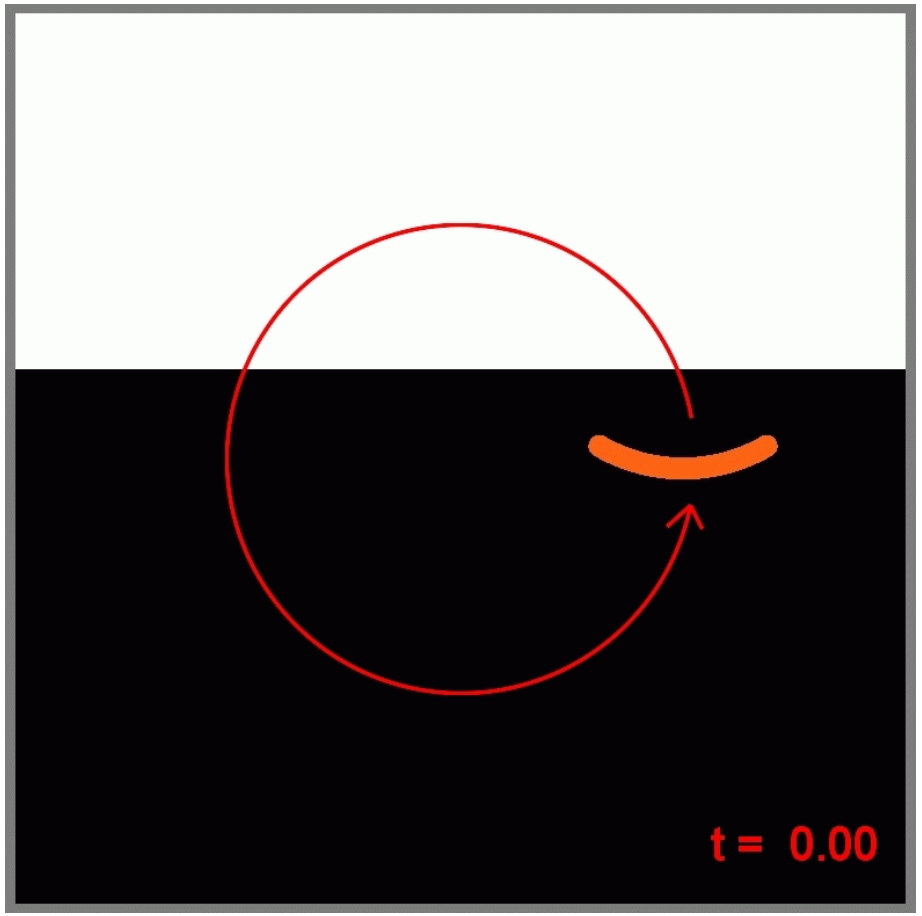
High-resolution
Rayleigh-Taylor instability

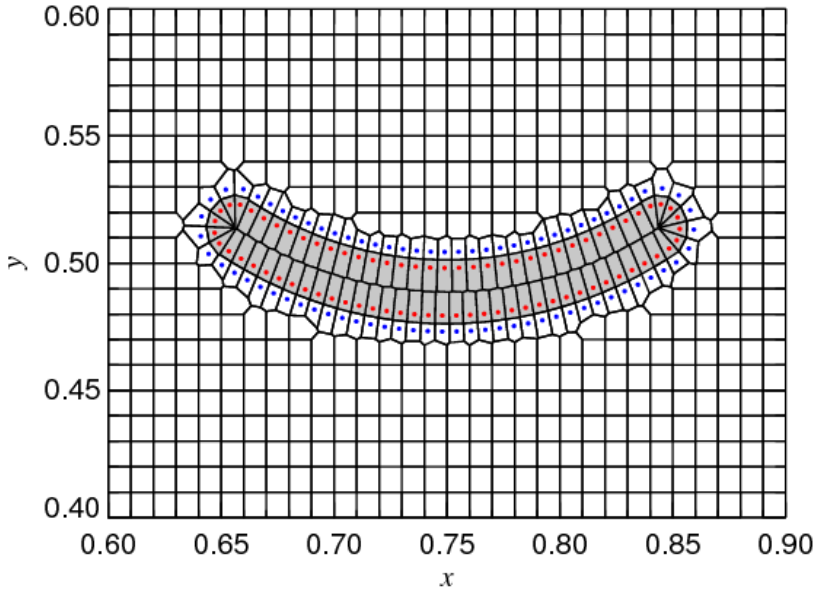
Rayleigh-Taylor (with visible mesh)

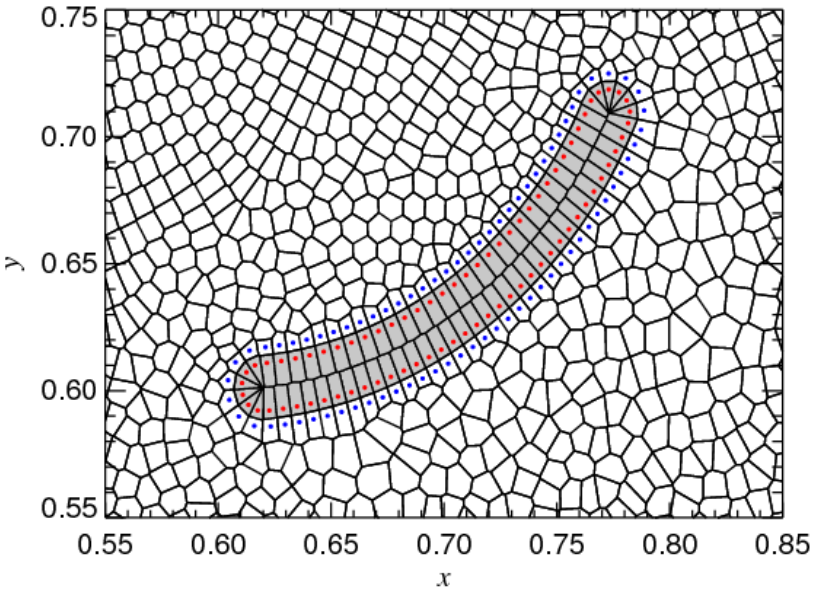


The moving-mesh approach can also be used to realize arbitrarily shaped, moving boundaries

STIRRING A COFFEE MUG



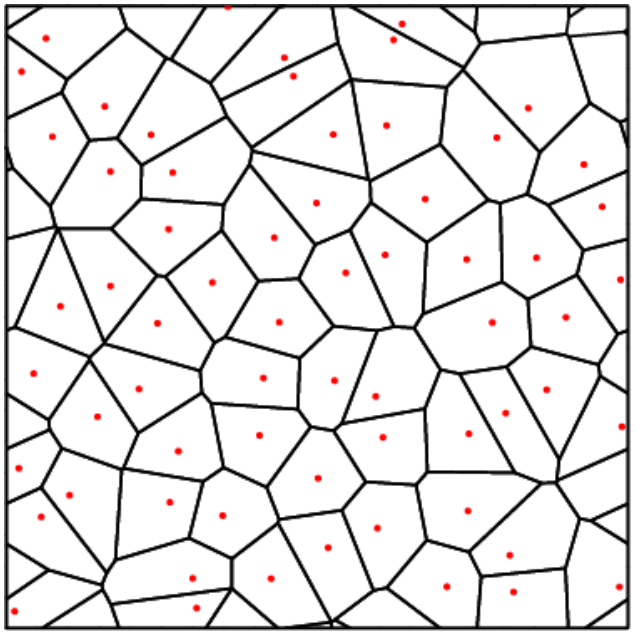




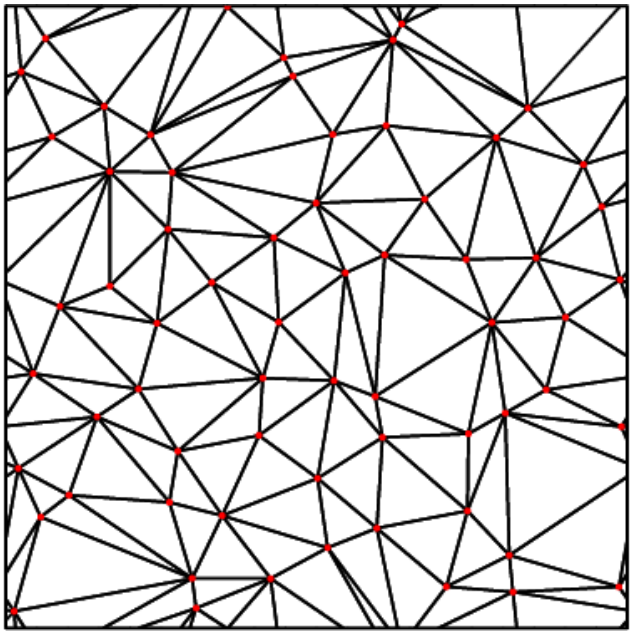
Voronoi and Delaunay tessellations provide unique partitions of space based on a given sample of mesh-generating points

BASIC PROPERTIES OF VORONOI AND DELAUNAY MESHES

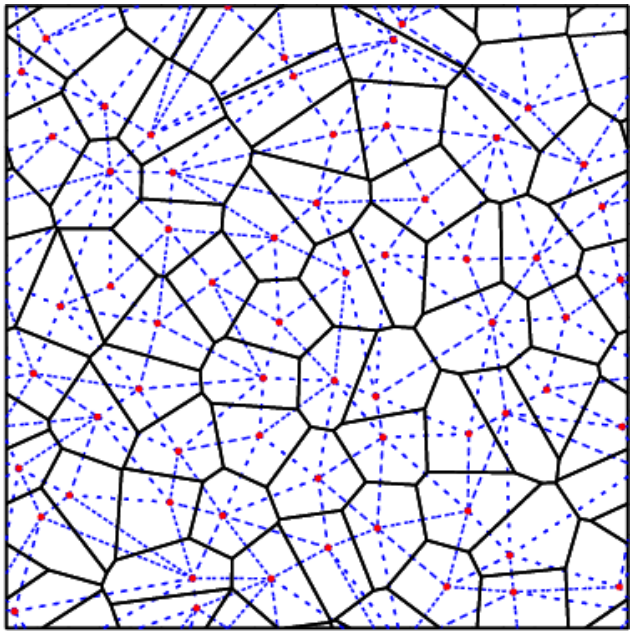
Voronoi mesh



Delaunay triangulation



both shown together



- Each Voronoi cell contains the **space closest** to its generating point
- The Delaunay triangulation contains only triangles with an **empty circumcircle**. The Delaunay tiangulation maximizes the minimum angle occurring among all triangles.
- The centres of the circumcircles of the Delaunay triangles are the vertices of the Voronoi mesh. In fact, the two tessellations are the topological **dual graph** to each other.

A finite volume discretization of the Euler equations on a moving mesh can be readily defined

THE EULER EQUATIONS AS HYPERBOLIC SYSTEM OF CONSERVATION LAWS

Euler equations

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

State vector

$$\mathbf{U} = \begin{pmatrix} \rho \\ \rho \mathbf{v} \\ \rho e \end{pmatrix}$$

Flux vector

$$\mathbf{F}(\mathbf{U}) = \begin{pmatrix} \rho \mathbf{v} \\ \rho \mathbf{v}\mathbf{v}^T + P \\ (\rho e + P)\mathbf{v} \end{pmatrix}$$

$$e = u + \mathbf{v}^2/2$$

Equation of state: $P = (\gamma - 1)\rho u$

Discretization in terms of a number of finite volume cells:

Cell averages

$$\mathbf{Q}_i = \begin{pmatrix} M_i \\ \mathbf{p}_i \\ E_i \end{pmatrix} = \int_{V_i} \mathbf{U}\, \mathrm{d}V$$

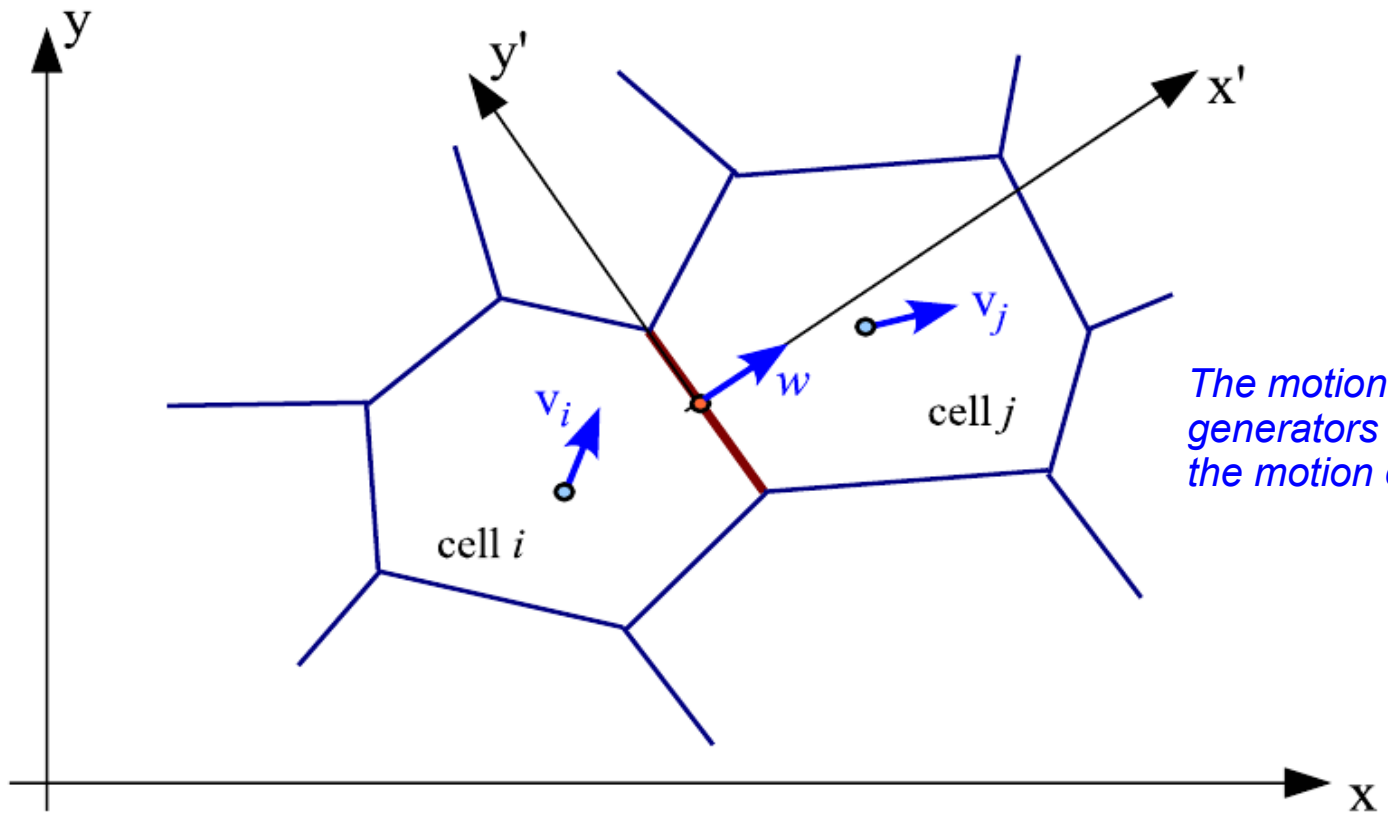
Evolution equation

$$\frac{\mathrm{d}\mathbf{Q}_i}{\mathrm{d}t} = -\int_{\partial V_i} \left[\mathbf{F}(\mathbf{U}) - \mathbf{U}\mathbf{w}^T\right] \mathrm{d}\mathbf{n}$$

Additional term for a moving mesh:
w is the velocity of the cell boundary

The fluxes are calculated with an exact Riemann solver in the frame of the moving cell boundary

SKETCH OF THE FLUX CALCULATION



The motion of the mesh generators uniquely determines the motion of all cell boundaries

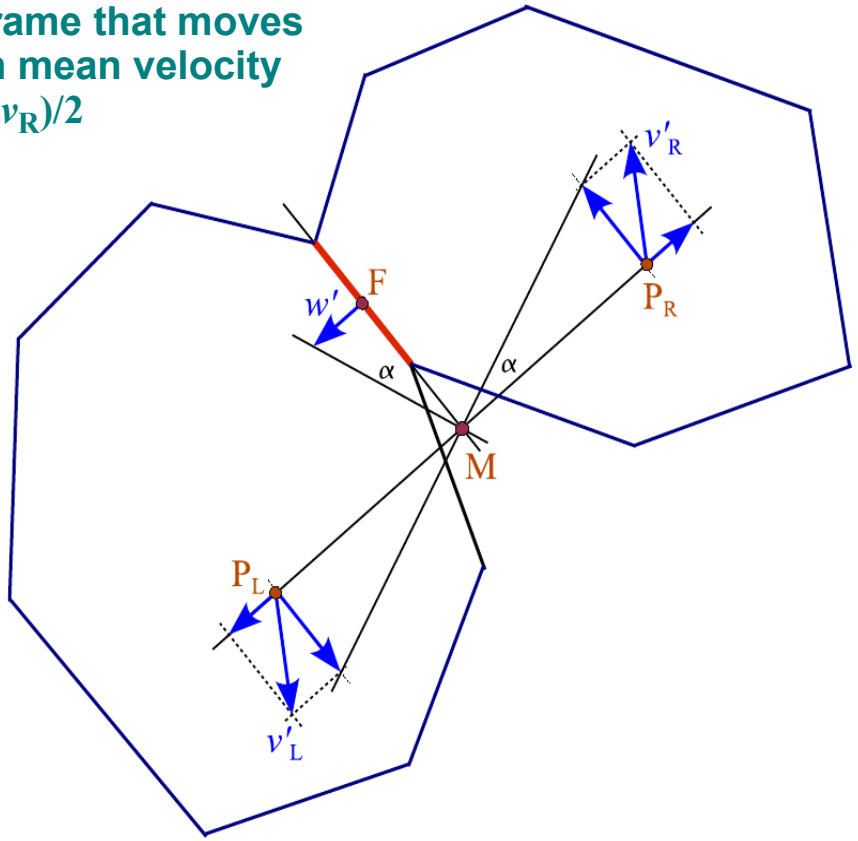
State left of cell face **State right of cell face**

$$\begin{pmatrix} \rho_L \\ \mathbf{v}_L \\ P_L \end{pmatrix} \quad \begin{pmatrix} \rho_R \\ \mathbf{v}_R \\ P_R \end{pmatrix} \longrightarrow \textbf{Riemann solver} \text{ (in frame of cell face)} \longrightarrow \begin{pmatrix} \rho \\ \mathbf{v} \\ P \end{pmatrix} \longrightarrow \mathbf{F}(\mathbf{U})$$

The velocities of the mesh-generating points uniquely determine the motion of all Voronoi faces

CHANGE OF VORONOI CELLS AS A FUNCTION OF TIME

in frame that moves with mean velocity $(v_L+v_R)/2$



rate of change of volume of a cell

$$\frac{\mathrm{d}V_i}{\mathrm{d}t} = -\sum_{j\neq i} A_{ij}\left[\frac{\boldsymbol{c}_{ij}}{r_{ij}}(\boldsymbol{v}_j-\boldsymbol{v}_i)+\frac{\boldsymbol{r}_{ij}}{2r_{ij}}(\boldsymbol{v}_j+\boldsymbol{v}_i)\right]$$

$$\boldsymbol{r}_{ij} = \boldsymbol{x}_i - \boldsymbol{x}_j$$

$$\boldsymbol{c}_{ij} = \boldsymbol{f}_{ij} - (\boldsymbol{x}_i+\boldsymbol{x}_j)/2$$

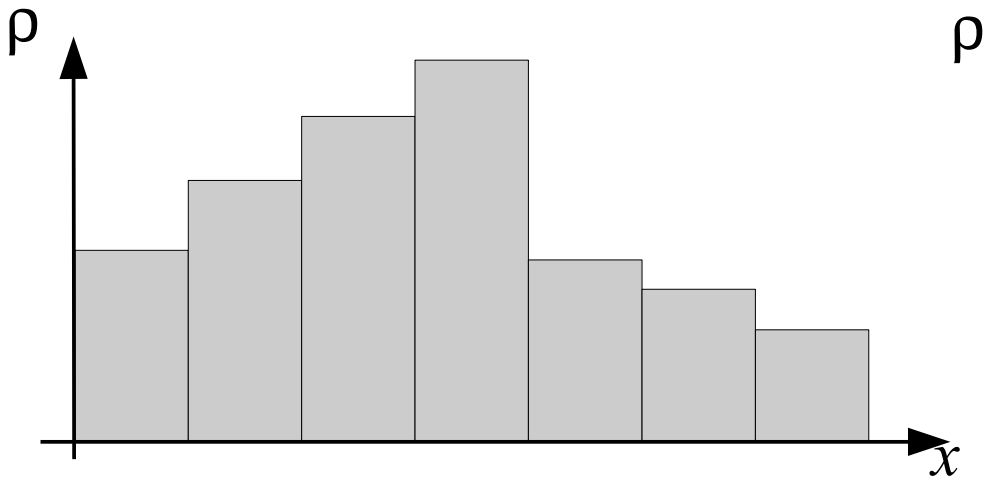
$$w' = \frac{(\boldsymbol{v}_\mathrm{L}-\boldsymbol{v}_\mathrm{R})\cdot[\boldsymbol{f}-(\boldsymbol{x}_\mathrm{R}+\boldsymbol{x}_\mathrm{L})/2]}{|\boldsymbol{x}_\mathrm{R}-\boldsymbol{x}_\mathrm{L}|}\,\frac{(\boldsymbol{x}_\mathrm{R}-\boldsymbol{x}_\mathrm{L})}{|\boldsymbol{x}_\mathrm{R}-\boldsymbol{x}_\mathrm{L}|}$$

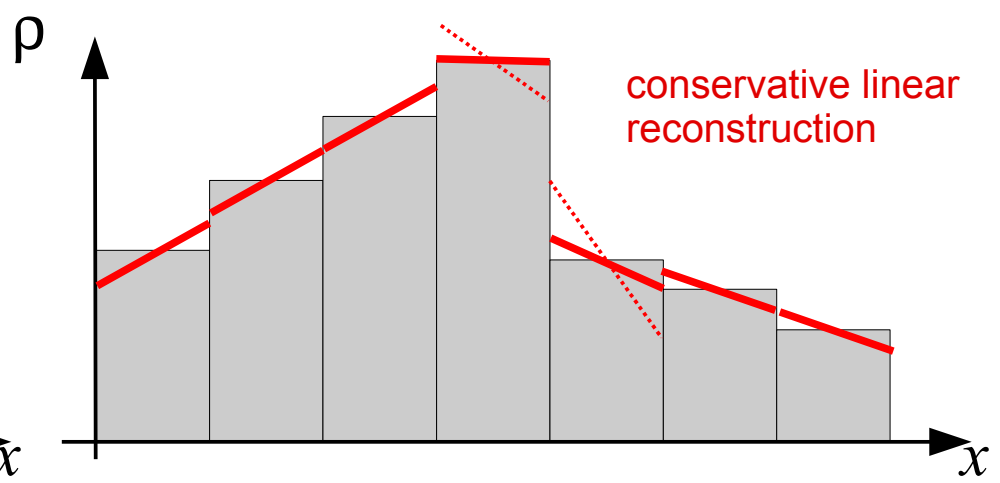
$$w = \frac{v_\mathrm{R}+v_\mathrm{L}}{2}+w'$$

(see also Serrano & Espanol 2001)

To achieve second-order accuracy, we use a **piece-wise linear** reconstruction

GRADIENT ESTIMATION AND LINEAR RECONSTRUCTION





Green-Gauss gradient estimation:

$$\int_{\partial V} \phi \, \mathrm{d}\boldsymbol{n} = \int_{V} \boldsymbol{\nabla} \phi \, \mathrm{d}V.$$

Leads to:

$$\langle \boldsymbol{\nabla} \phi \rangle_i = \frac{1}{V_i} \sum_{j \neq i} A_{ij} \left([\phi_j - \phi_i] \frac{\boldsymbol{c}_{ij}}{r_{ij}} - \frac{\phi_i + \phi_j}{2} \frac{\boldsymbol{r}_{ij}}{r_{ij}} \right)$$

Slope limiting procedure:

$$\langle \boldsymbol{\nabla} \phi \rangle_i' = \alpha_i \langle \boldsymbol{\nabla} \phi \rangle_i$$

$$\alpha_i = \min(1, \psi_{ij})$$

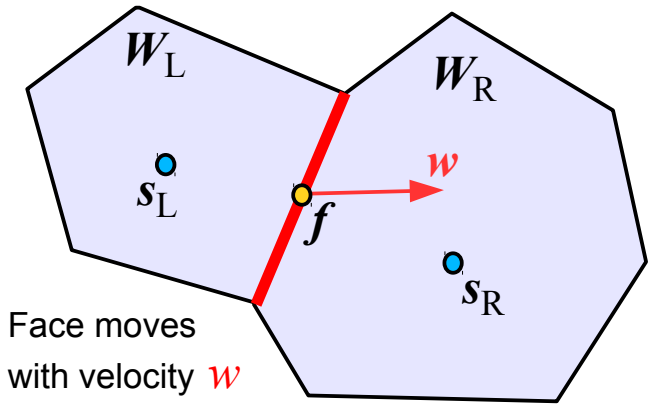
$$\psi_{ij} = \begin{cases} (\phi_i^{\max} - \phi_i)/\Delta\phi_{ij} & \text{for} \quad \Delta\phi_{ij} > 0 \\ (\phi_i^{\min} - \phi_i)/\Delta\phi_{ij} & \text{for} \quad \Delta\phi_{ij} < 0 \\ 1 & \text{for} \quad \Delta\phi_{ij} = 0 \end{cases}$$

$$\Delta\phi_{ij} = \langle \boldsymbol{\nabla} \phi \rangle_i \cdot (\boldsymbol{f}_{ij} - \boldsymbol{s}_i)$$

$$\phi_i^{\max} = \max(\phi_j) \qquad \phi_i^{\min} = \max(\phi_j)$$

Our second-order time integration scheme uses a half-step prediction in primitive variable formulation

A MUSCL-LIKE SCHEME



And finally...

Update the conserved variables of each cell:

$$\boldsymbol{Q}_i^{(n+1)} = \boldsymbol{Q}_i^{(n)} - \Delta t \sum_j A_{ij} \hat{\boldsymbol{F}}_{ij}^{(n+1/2)}$$

This scheme is **Galilean invariant** if $\boldsymbol{w}$ is tied to the fluid velocity.

Transform left and right fluid states into rest frame of face

$$\boldsymbol{W}'_{\mathrm{L,R}} = \boldsymbol{W}_{\mathrm{L,R}} - \begin{pmatrix} 0 \\ \boldsymbol{w} \\ 0 \end{pmatrix}$$

Linearly predict the states to the midpoint of the face, and evolve them forward in time by half a timestep:

$$\boldsymbol{W}''_{\mathrm{L,R}} = \boldsymbol{W}'_{\mathrm{L,R}} + \left.\frac{\partial \boldsymbol{W}}{\partial \boldsymbol{r}}\right|_{\mathrm{L,R}} (\boldsymbol{f} - \boldsymbol{s}_{\mathrm{L,R}}) + \left.\frac{\partial \boldsymbol{W}}{\partial t}\right|_{\mathrm{L,R}} \frac{\Delta t}{2}$$

The prediction in time can be done with the Euler equations:

$$\frac{\partial \boldsymbol{W}}{\partial t} + \boldsymbol{A}(\boldsymbol{W}) \frac{\partial \boldsymbol{W}}{\partial \boldsymbol{r}} = 0 \qquad \boldsymbol{A}(\boldsymbol{W}) = \begin{pmatrix} \boldsymbol{v} & \rho & 0 \\ 0 & \boldsymbol{v} & 1/\rho \\ 0 & \gamma P & \boldsymbol{v} \end{pmatrix}$$

Rotate the states such that one coordinate is normal to the face

$$\boldsymbol{W}'''_{\mathrm{L,R}} = \boldsymbol{\Lambda}\, \boldsymbol{W}''_{\mathrm{L,R}} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \boldsymbol{\Lambda}_{3\mathrm{D}} & 0 \\ 0 & 0 & 1 \end{pmatrix} \boldsymbol{W}''_{\mathrm{L,R}}$$

Solve the Riemann problem

$$\boldsymbol{W} = R_{\mathrm{iemann}}(\boldsymbol{W}'''_{\mathrm{L}}, \boldsymbol{W}'''_{R})$$

Transform the solution back to the calculational frame

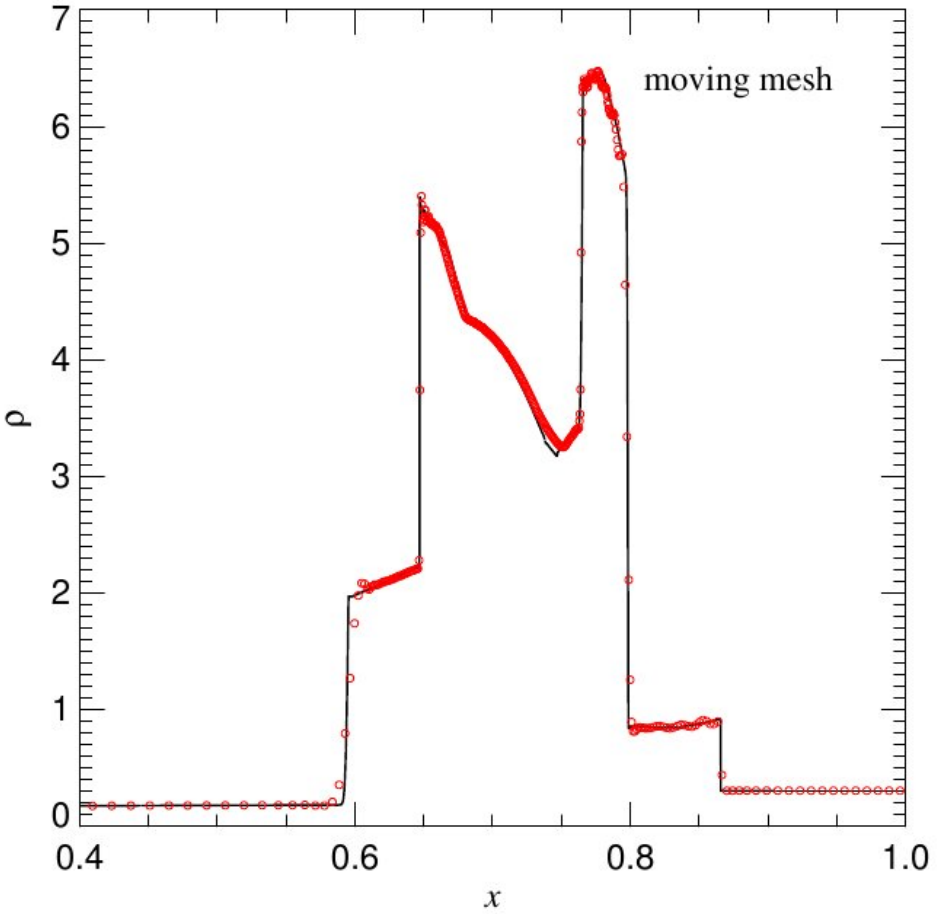
$$\boldsymbol{W}_{\mathrm{lab}} = \begin{pmatrix} \rho \\ \boldsymbol{v}_{\mathrm{lab}} \\ P \end{pmatrix} = \Lambda^{-1} \boldsymbol{W} + \begin{pmatrix} 0 \\ \boldsymbol{w} \\ 0 \end{pmatrix}$$

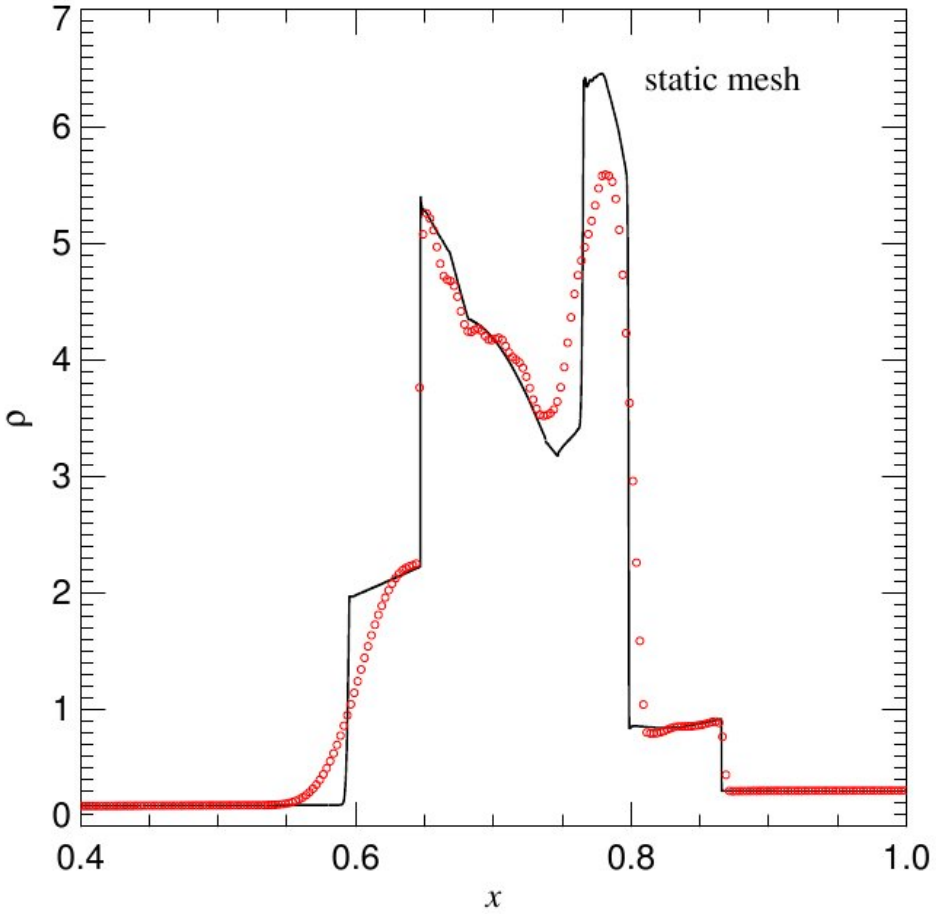
Calculate the net flux in the calculational frame

$$\hat{\boldsymbol{F}} = \boldsymbol{F}(\boldsymbol{U}) - \boldsymbol{U}\boldsymbol{w}^{\mathrm{T}} = \begin{pmatrix} \rho(\boldsymbol{v}_{\mathrm{lab}} - \boldsymbol{w}) \\ \rho \boldsymbol{v}_{\mathrm{lab}} (\boldsymbol{v}_{\mathrm{lab}} - \boldsymbol{w})^{\mathrm{T}} + P \\ \rho e_{\mathrm{lab}} (\boldsymbol{v}_{\mathrm{lab}} - \boldsymbol{w}) + P \boldsymbol{v}_{\mathrm{lab}} \end{pmatrix}$$

The moving-mesh code deals well will problems that involve complicated shock interactions

WOODWARD & COLELLA'S INTERACTING DOUBLE BLAST PROBLEM

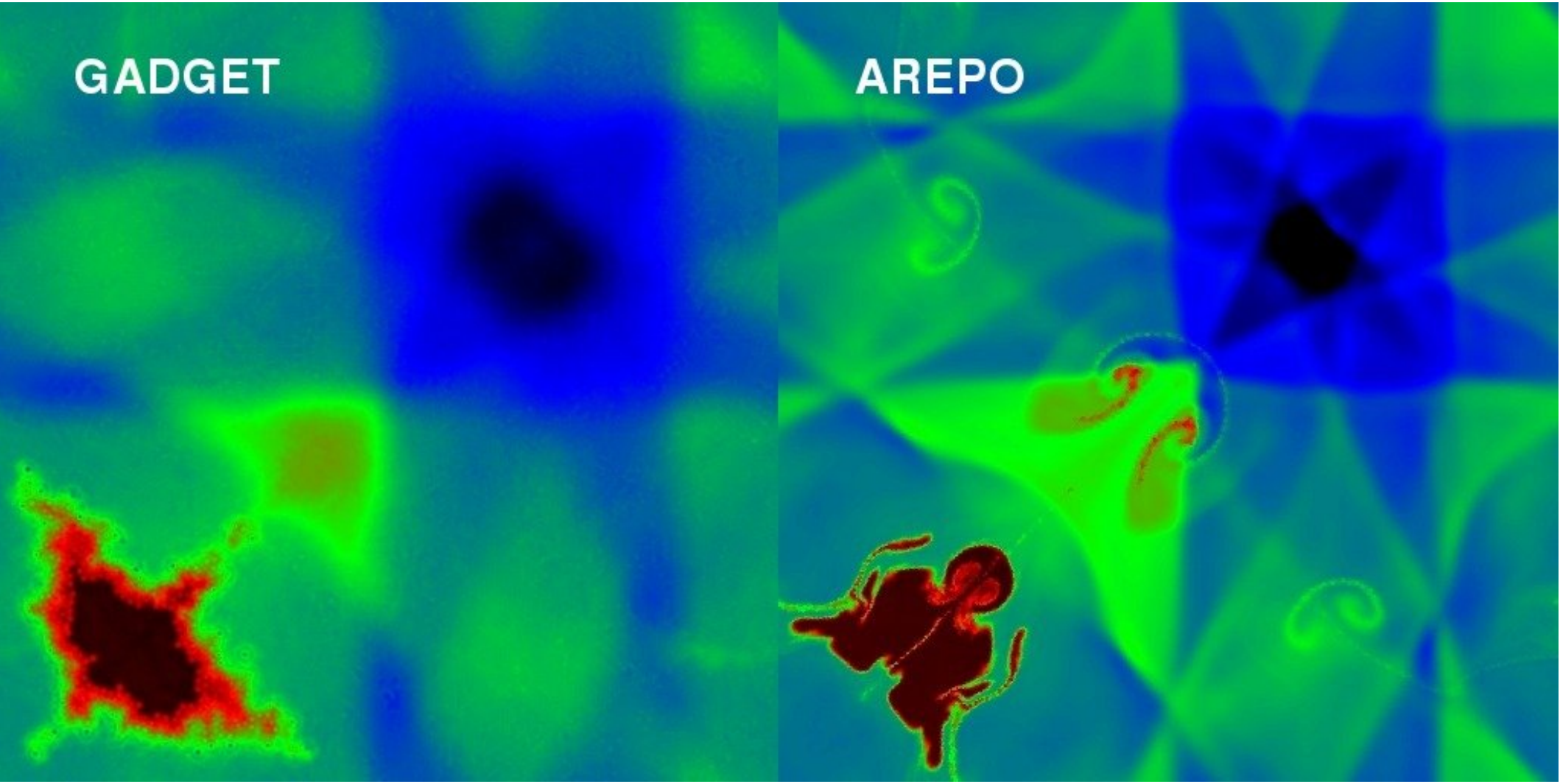




Interacting shock waves reveal significant differences in vorticity production

TWO-DIMENSIONAL IMPLOSION PROBLEM

Sijacki et al. (2011)



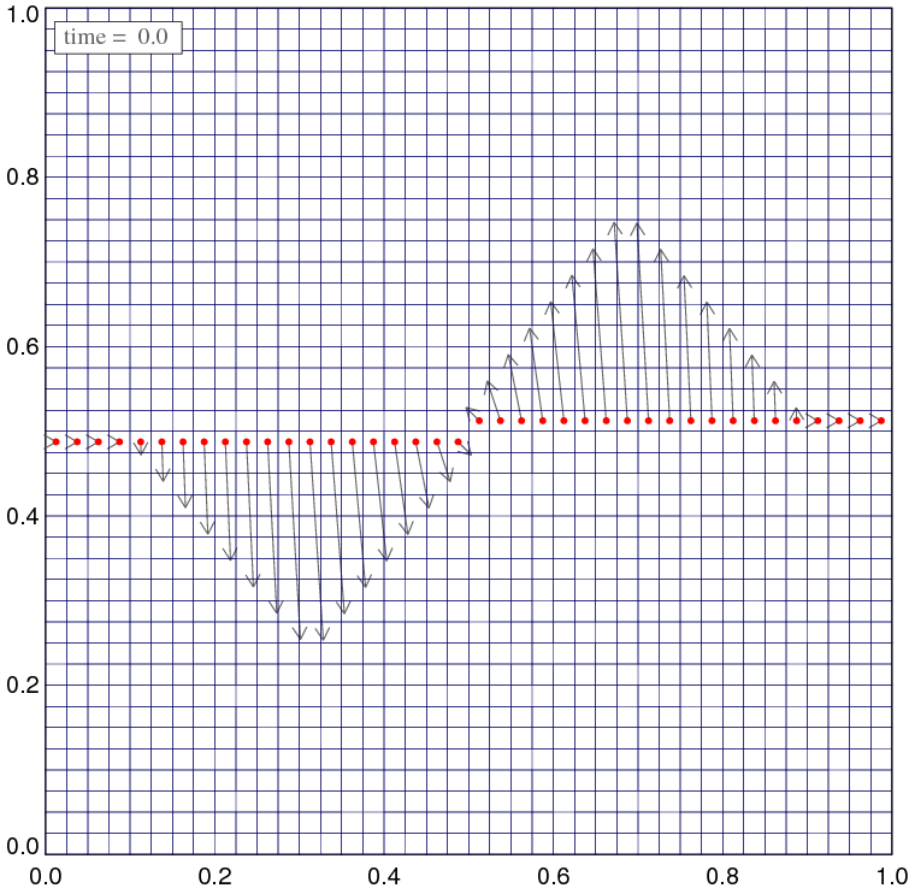
The Gresho vortex test in two dimensions

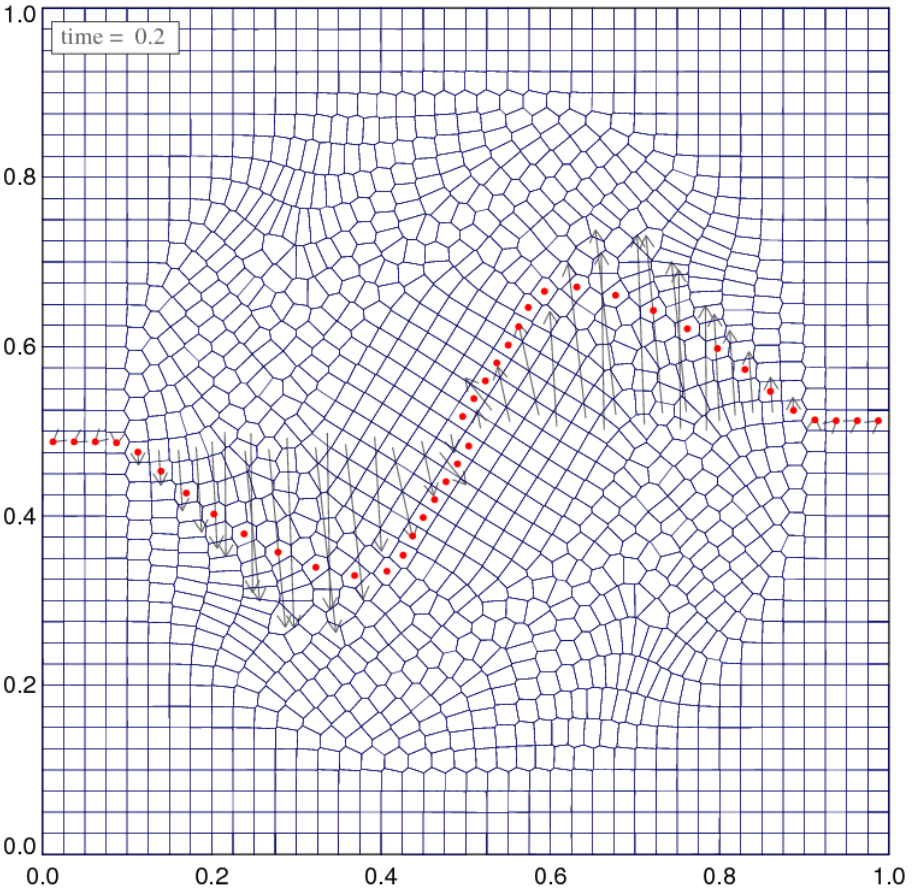
EVOLUTION OF A STATIONARY VORTEX FLOW

Initial conditions:

$$v_\phi(r) = \begin{cases} 5r & \text{for} \quad 0 \le r < 0.2 \\ 2 - 5r & \text{for} \quad 0.2 \le r < 0.4 \\ 0 & \text{for} \quad r \ge 0.4 \end{cases}$$

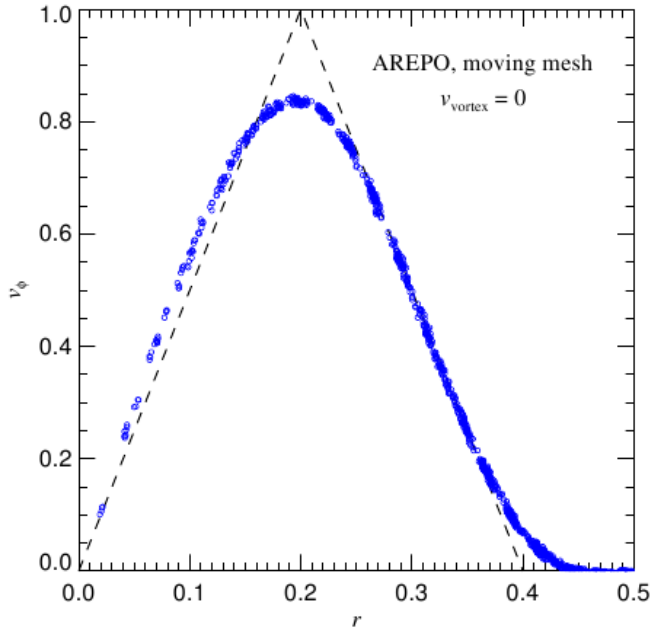
$$P(r) = \begin{cases} 5 + 25/2r^2 & \text{for} \quad 0 \le r < 0.2 \\ 9 + 25/2r^2 - \\ \quad 20r + 4\ln(r/0.2) & \text{for} \quad 0.2 \le r < 0.4 \\ 3 + 4\ln 2 & \text{for} \quad r \ge 0.4 \end{cases}$$

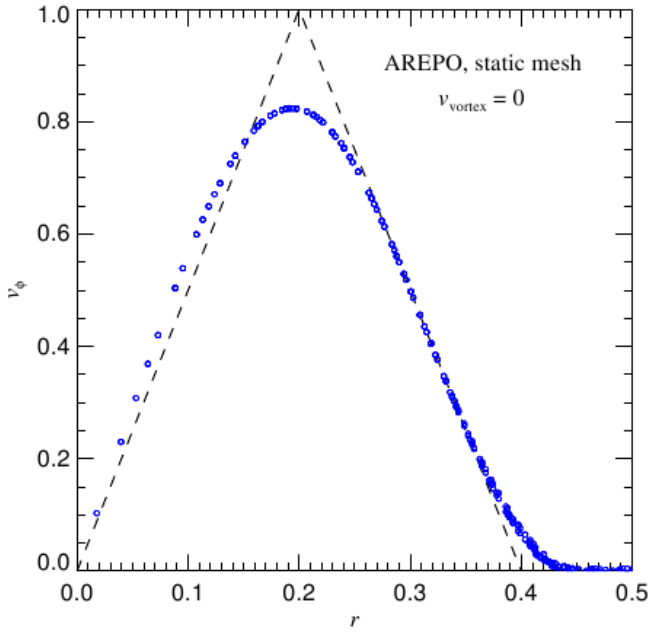


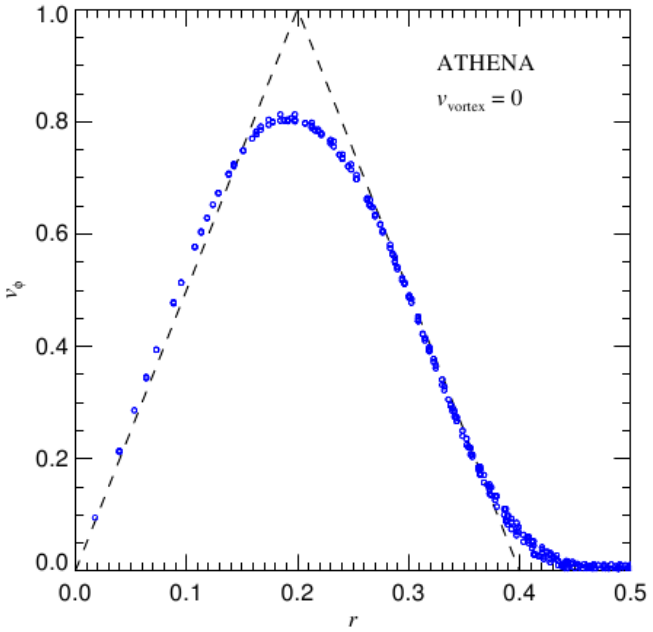


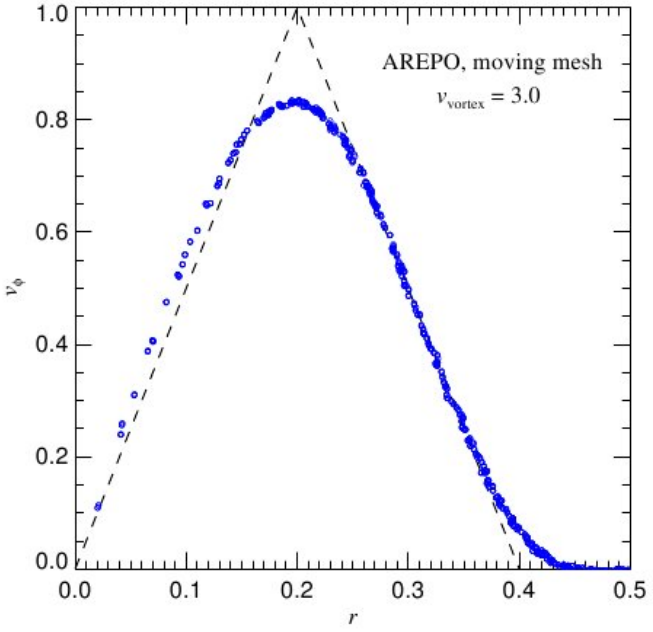
The Gresho vortex test in two dimensions

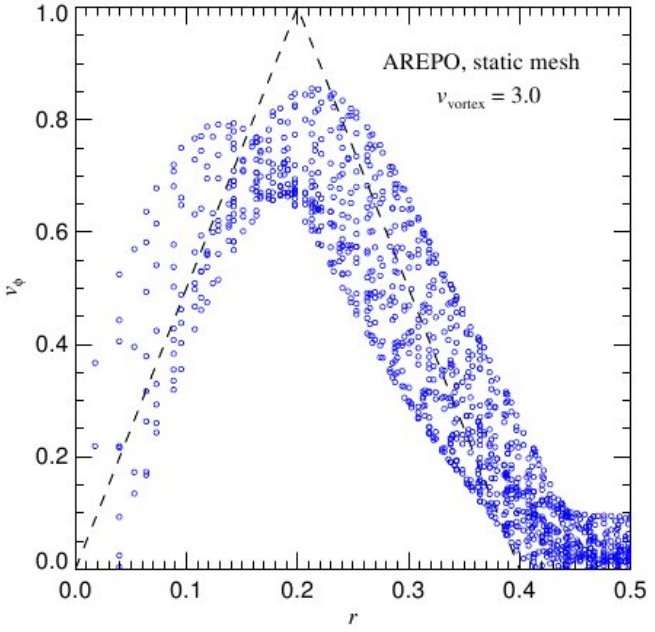
EVOLVED AZIMUTHAL VELOCITY PROFILE FOR DIFFERENT CODES AND BOOSTS

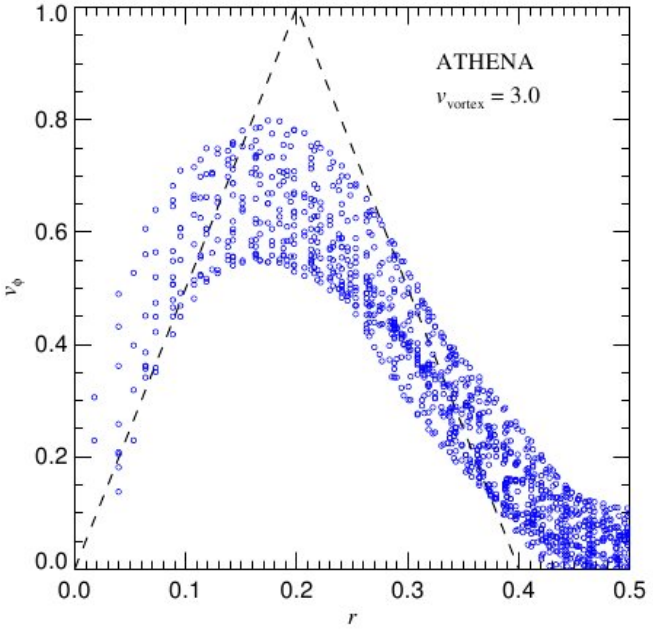






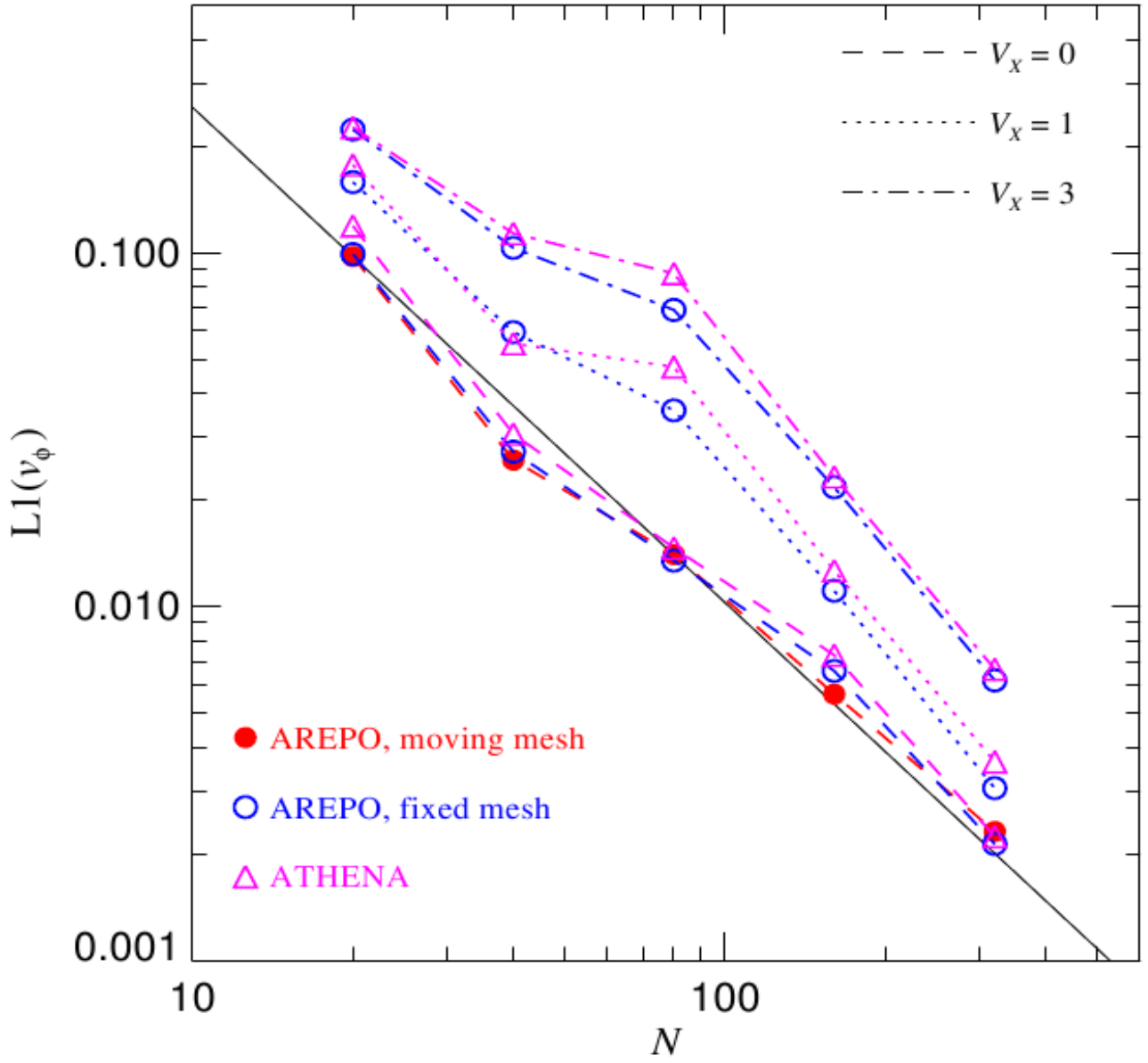






The Gresho vortex test in two dimensions

CONVERGENCE RATE AGAINST ANALYTIC SOLUTION



How to construct the Voronoi mesh

Construction of the Voronoi diagram is most efficiently done by constructing it as dual of the Delaunay tessellation

A FEW ALGORITHMS FOR DELAUNAY TRIANGULATIONS

2D

- Divide & Conquer (fastest)
- **Sequential insertion**
- Sweepline algorithm
- Projection of 3D convex hull to 3D

3D

- **Sequential insertion**
- Projection of 4D convex hull to 3D
- Incremental construction

Sequential insertion:

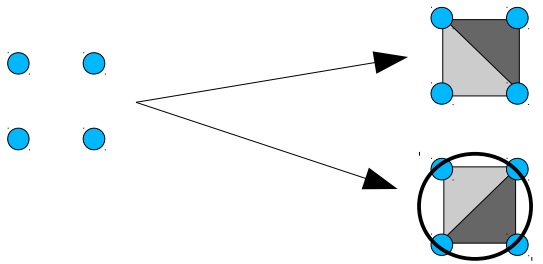
(1) **Point location:** Find triangle/tetrahedron that contains point

(2) **Point insertion**: Split enclosing triangle/tetrahedron into several simplices

(3) **Flips to restore Delaunayhood:** Replace edges/facets around the inserted point if they violate the Delaunay condition (empty circumcircle)

Most algorithms assume the **general position assumption**

Unfortunately, **degenerate cases** do occur in practice, and induce numerical difficulties due to numerical round-off

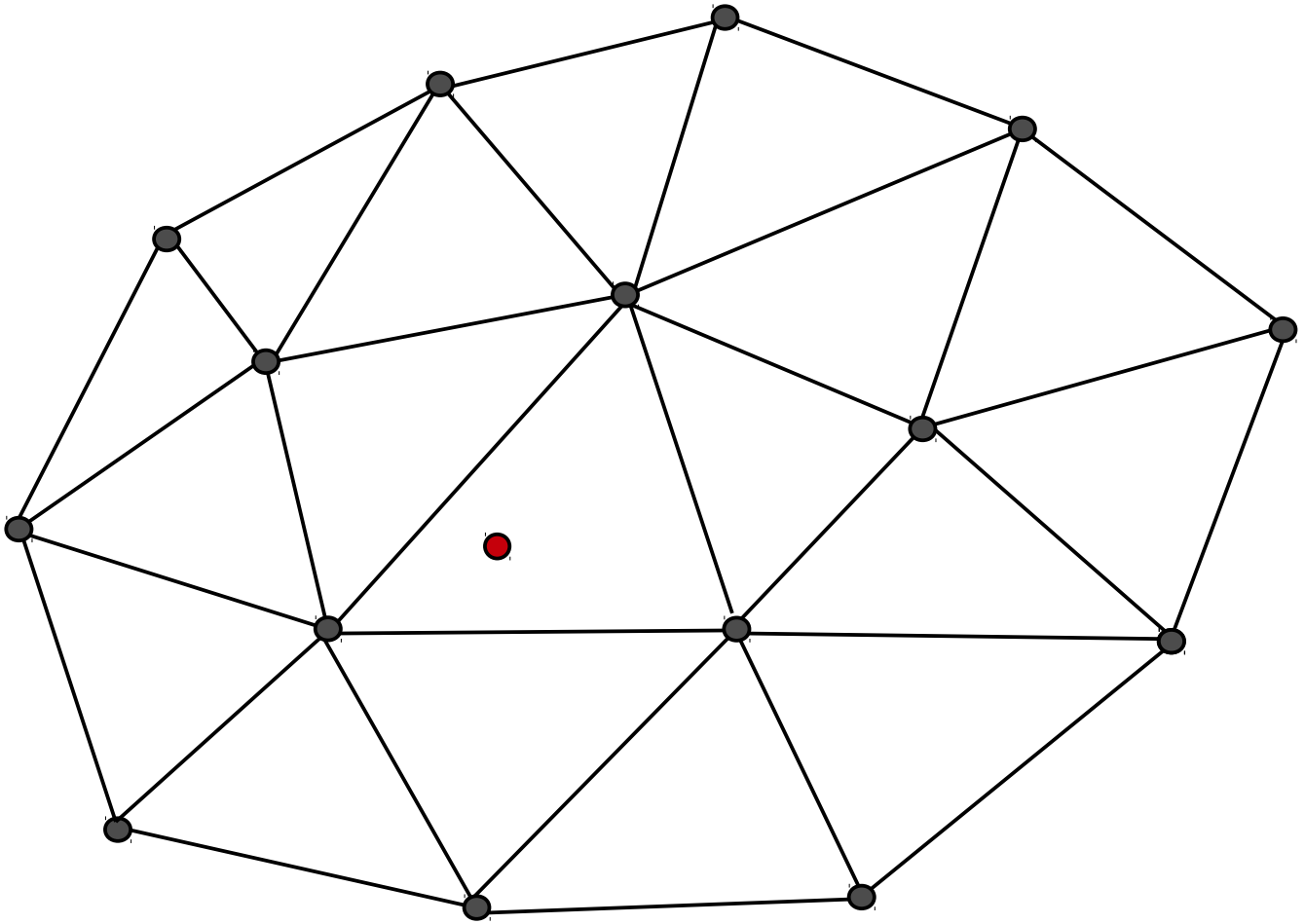


4 cocircular points !

How can we consistently break ties?

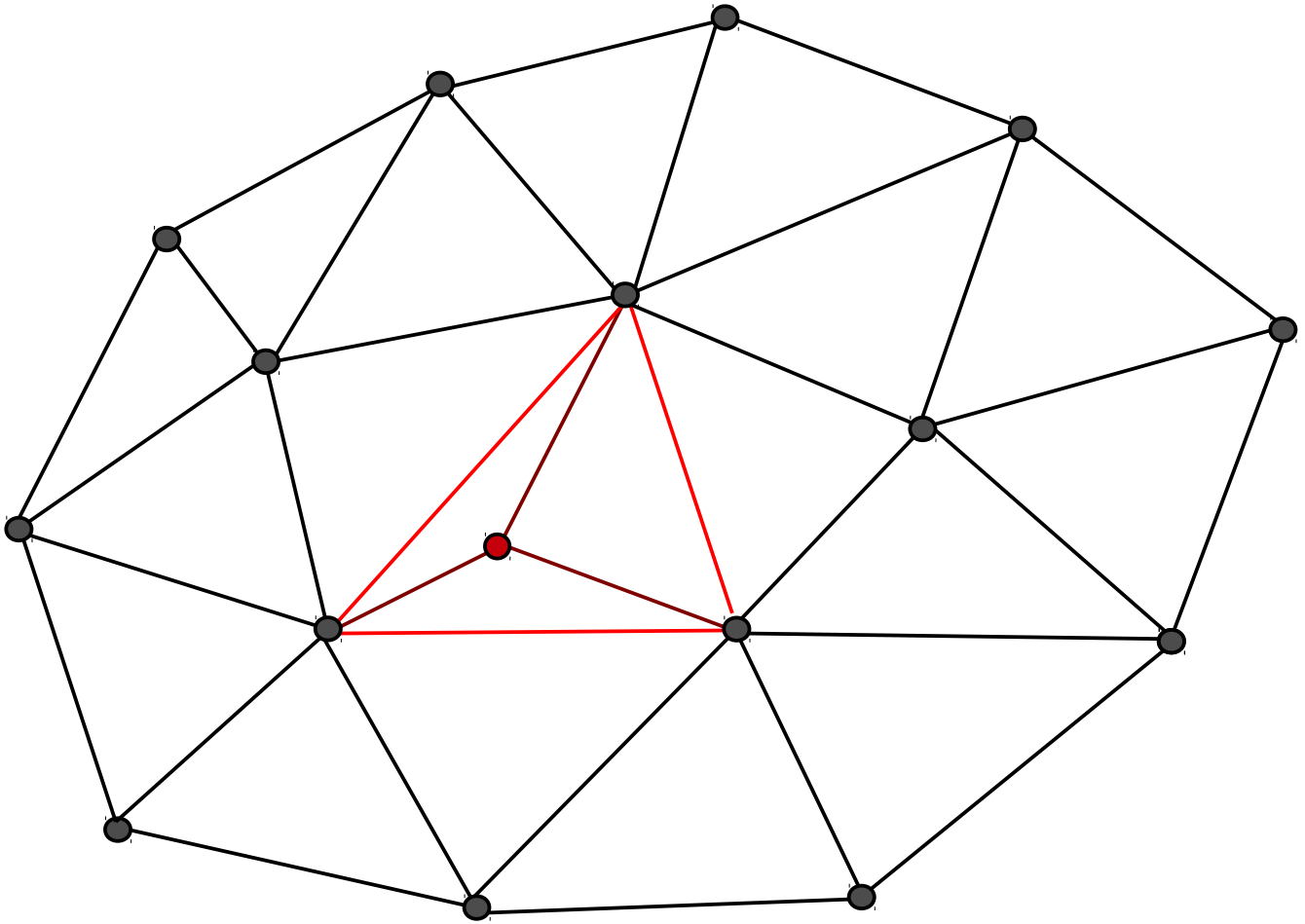
Adding a point by sequential insertion

1. Step: Locate the triangle that contains the point



Adding a point by sequential insertion

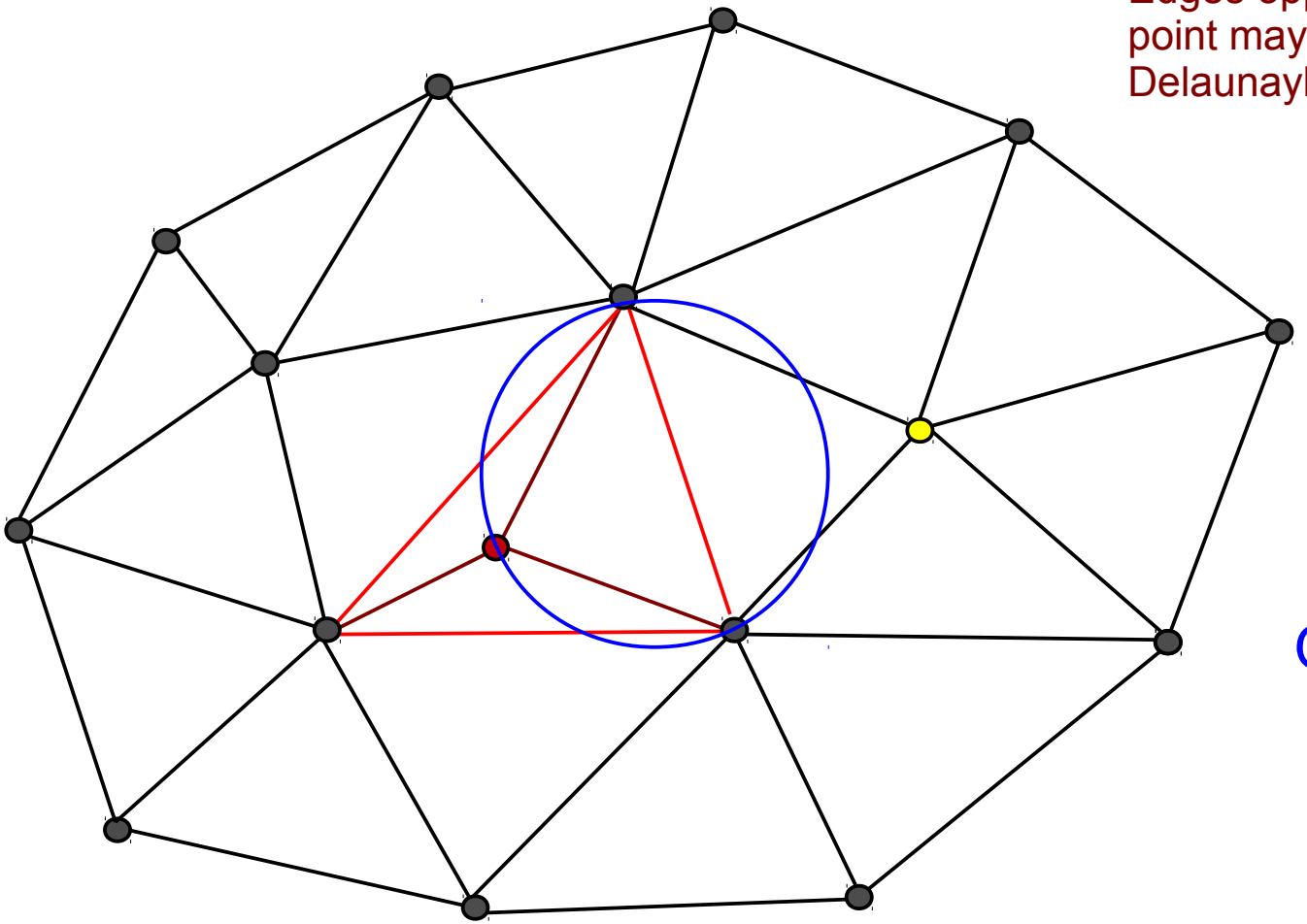
2. Step: Split the triangle into three triangles



Adding a point by sequential insertion

3. Step: Legalize the new triangles

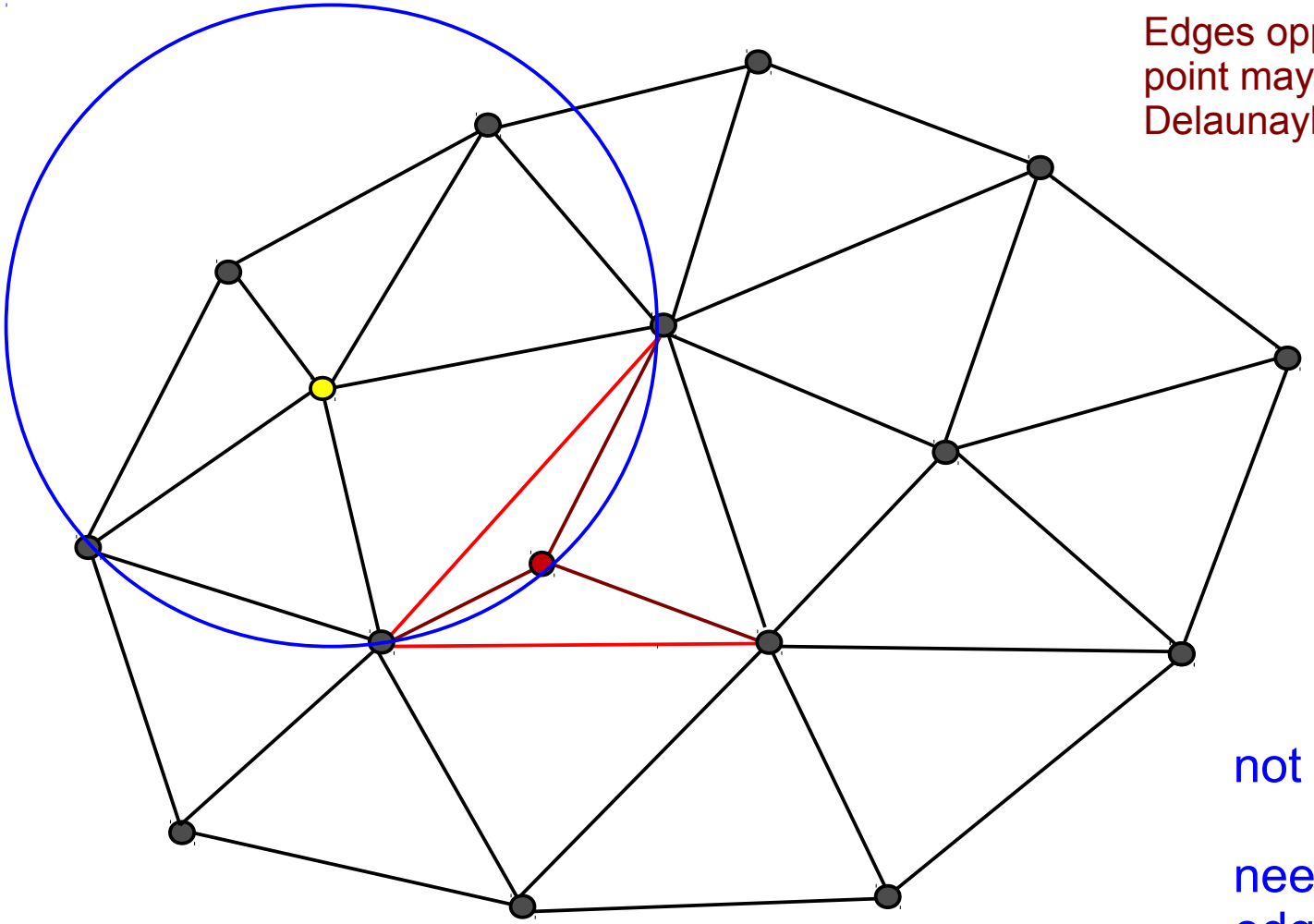
Edges opposite of new point may violate Delaunayhood



Ok!

Adding a point by sequential insertion

3. Step: Legalize the new triangles



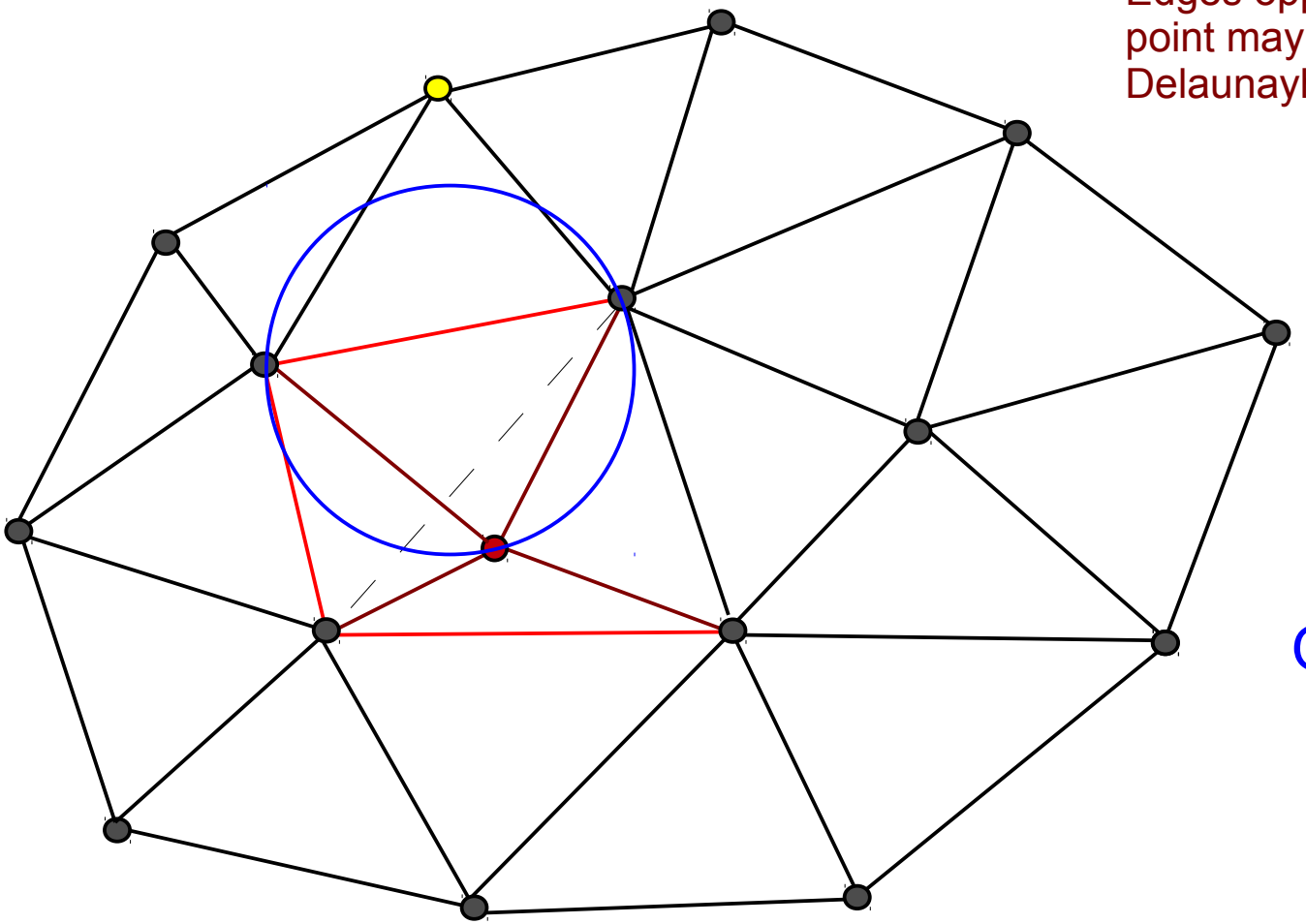
Edges opposite of new point may violate Delaunayhood

not ok...!

need to flip edge

Adding a point by sequential insertion

3. Step: Legalize the new triangles



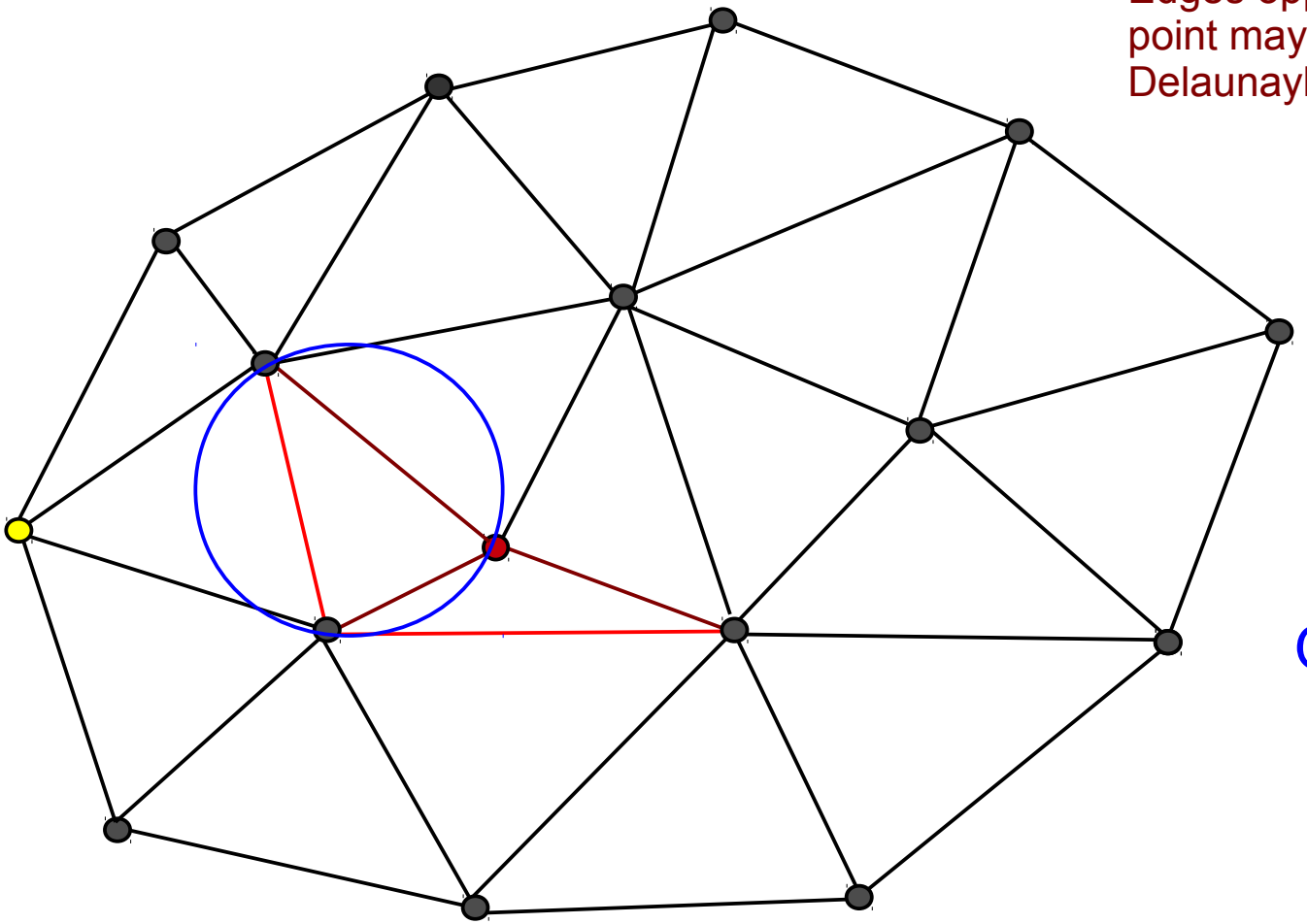
Edges opposite of new point may violate Delaunayhood

Ok!

Adding a point by sequential insertion

3. Step: Legalize the new triangles

Edges opposite of new point may violate Delaunayhood

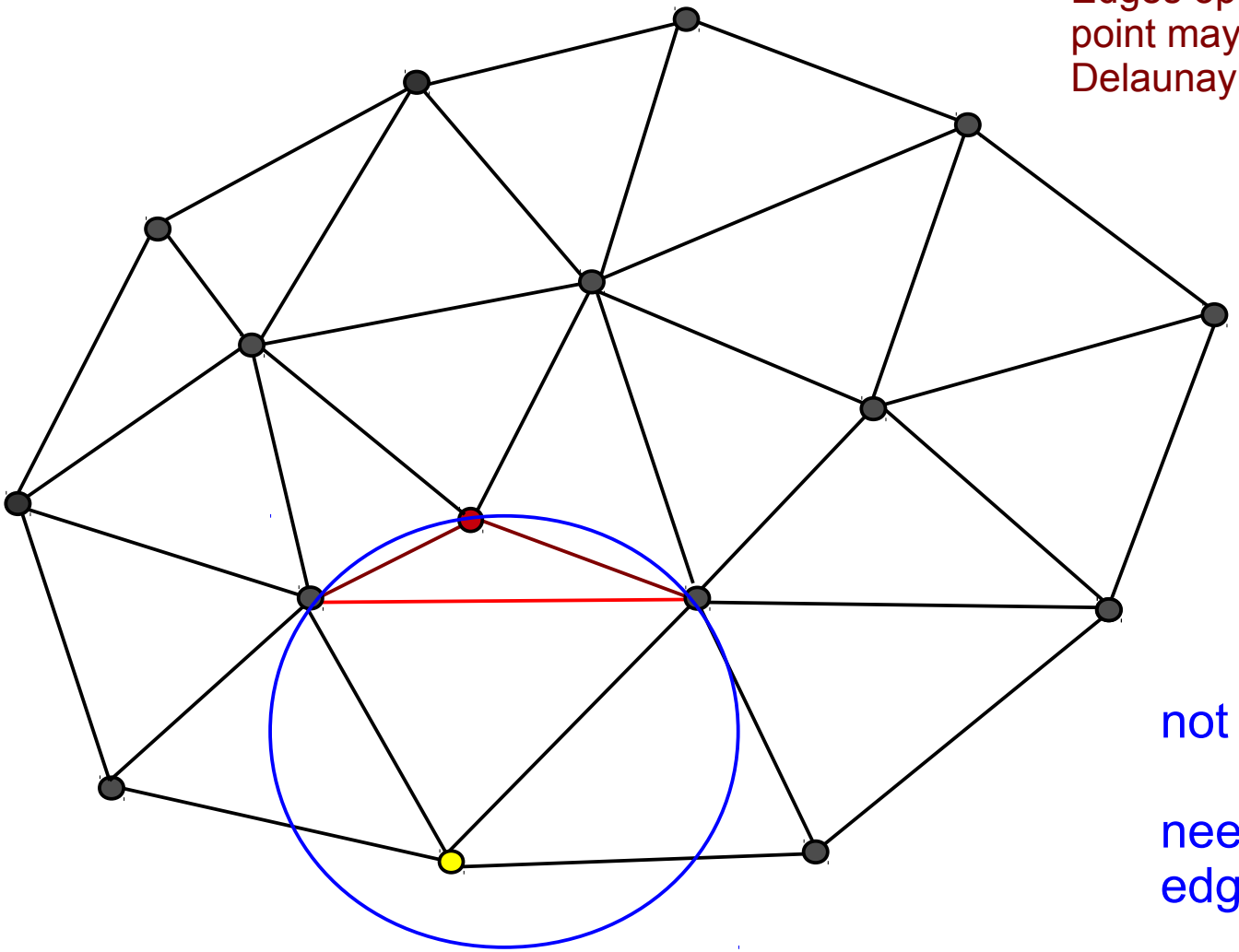


Ok!

Adding a point by sequential insertion

3. Step: Legalize the new triangles

Edges opposite of new point may violate Delaunayhood



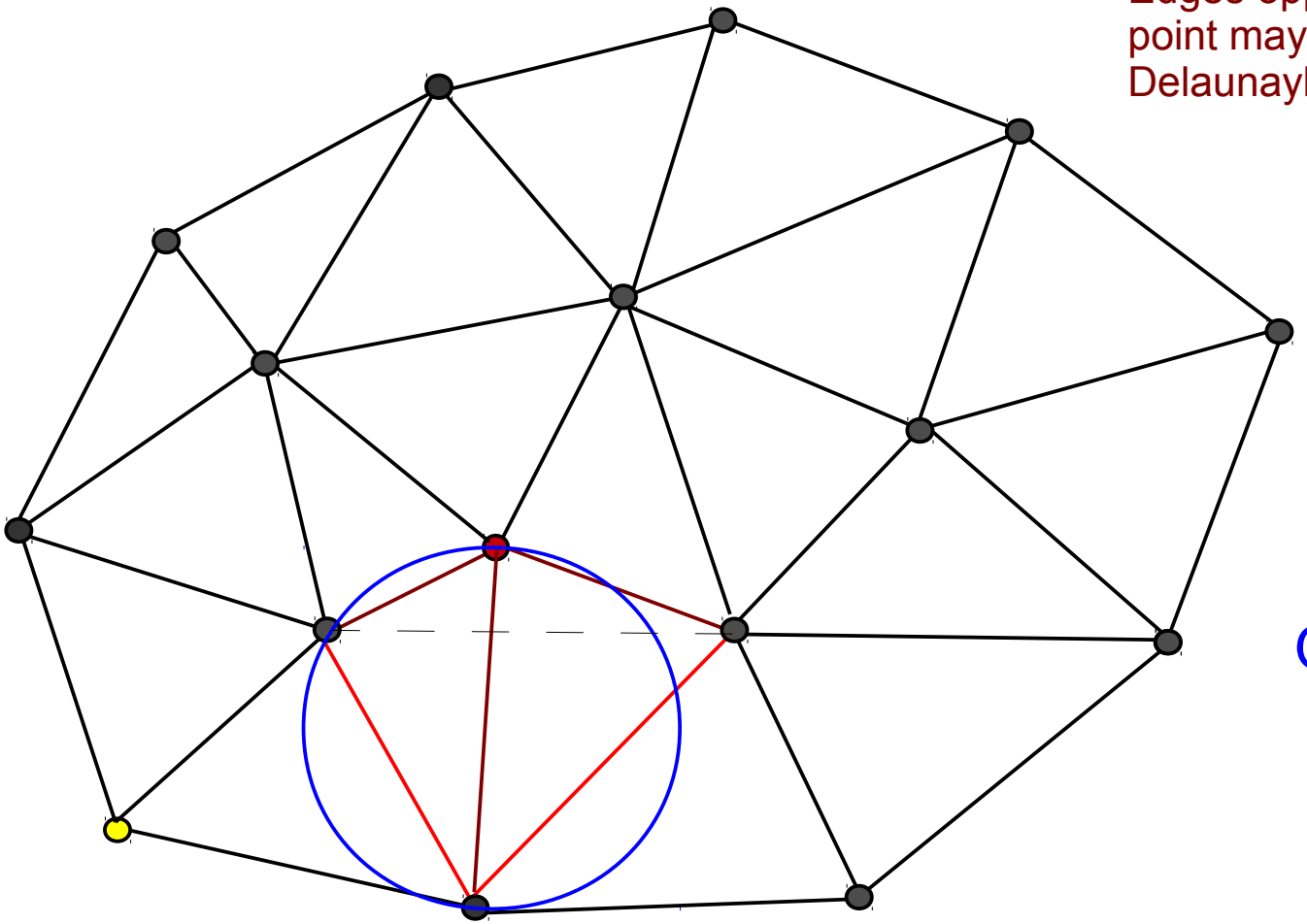
not ok...!

need to flip edge

Adding a point by sequential insertion

3. Step: Legalize the new triangles

Edges opposite of new point may violate Delaunayhood

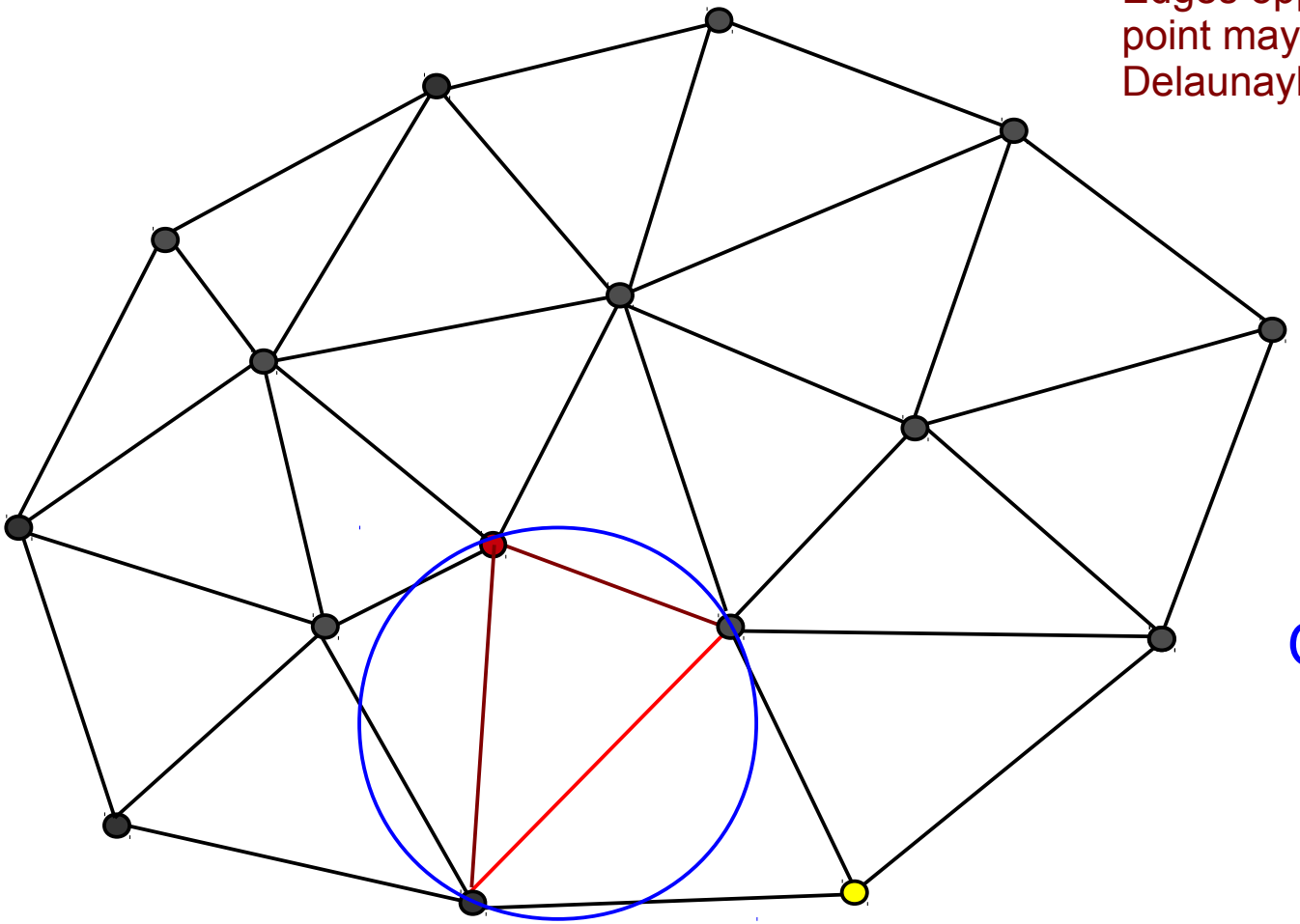


Ok!

Adding a point by sequential insertion

3. Step: Legalize the new triangles

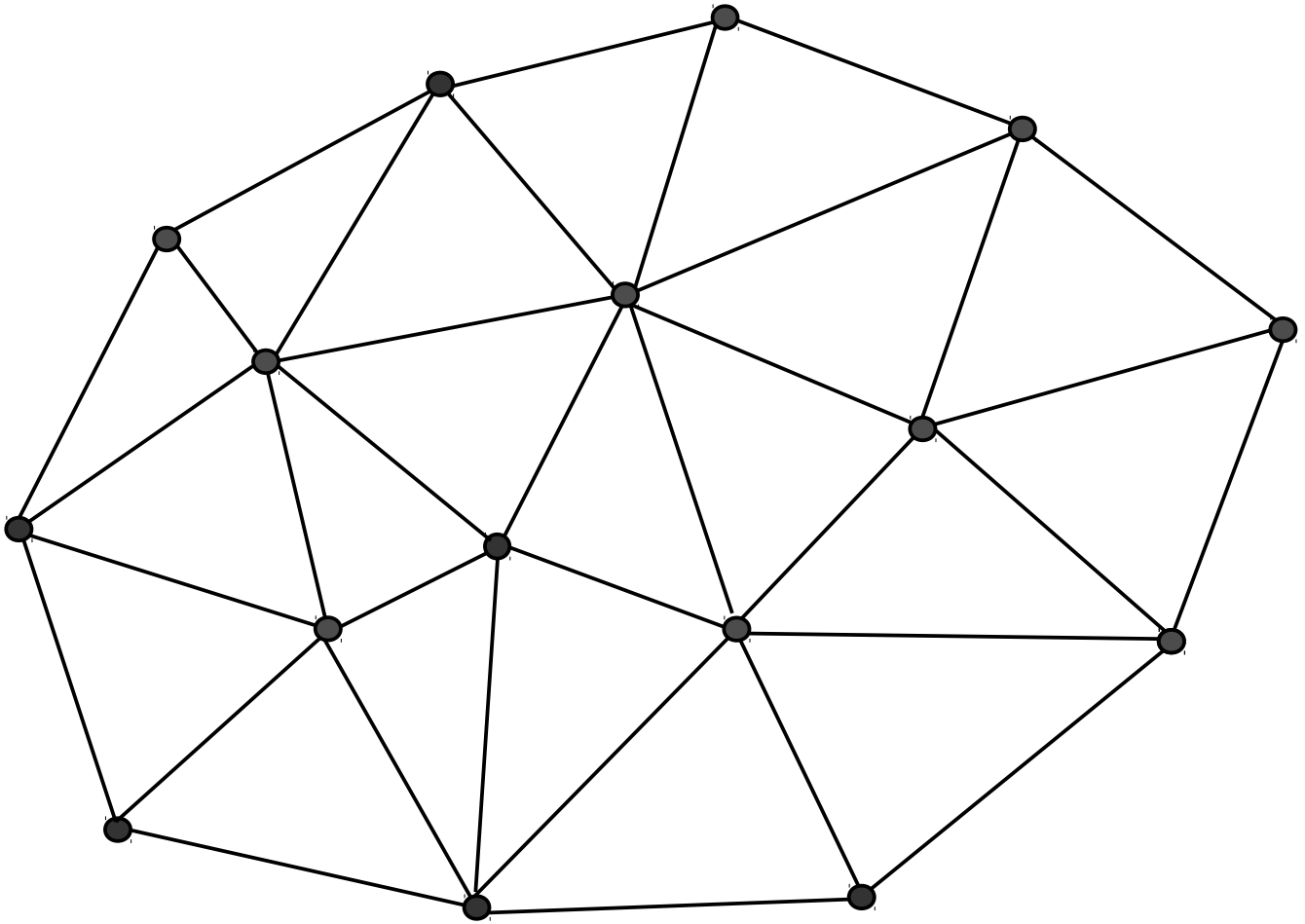
Edges opposite of new point may violate Delaunayhood



Ok!

Adding a point by sequential insertion

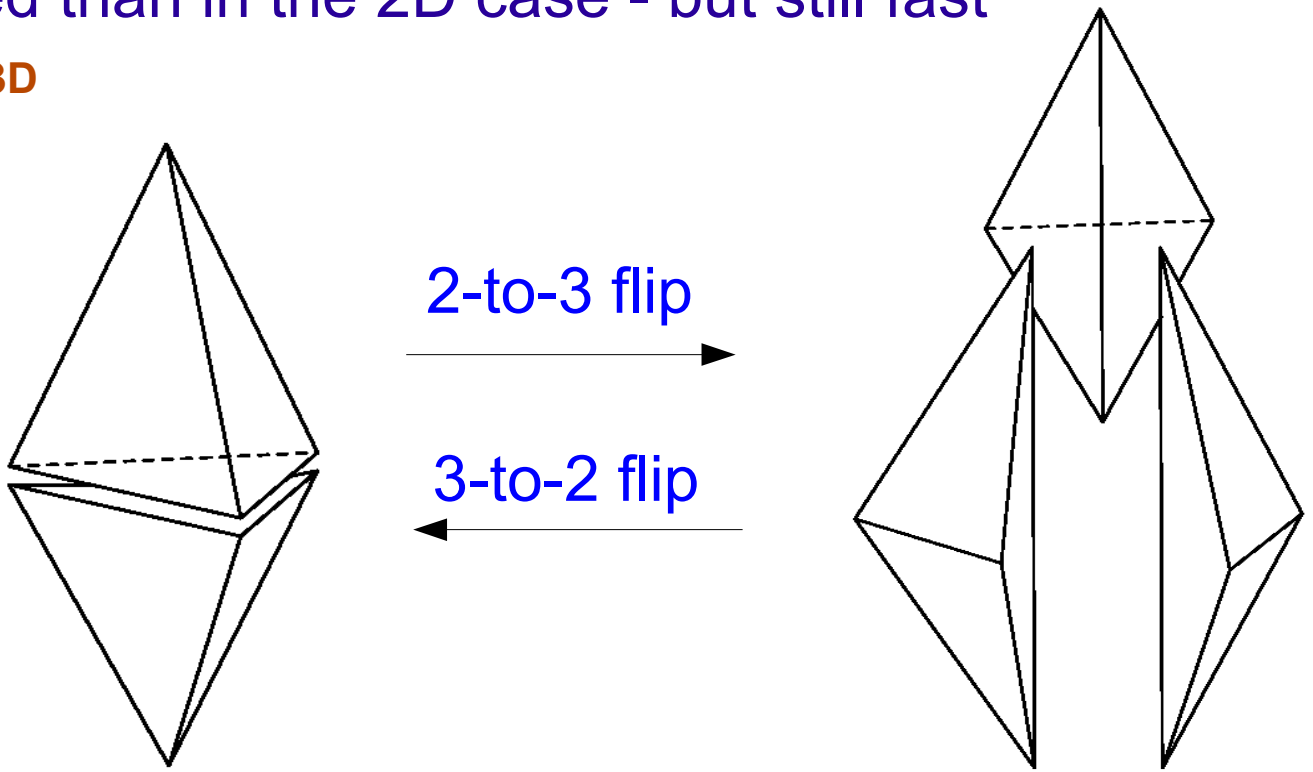
4. Step: Finished! (Or insert next point)



The construction of the 3D Delaunay tessellation is significantly more complicated than in the 2D case - but still fast

FLIP OPERATIONS IN 3D

1-to-4 flip
(point insertion)

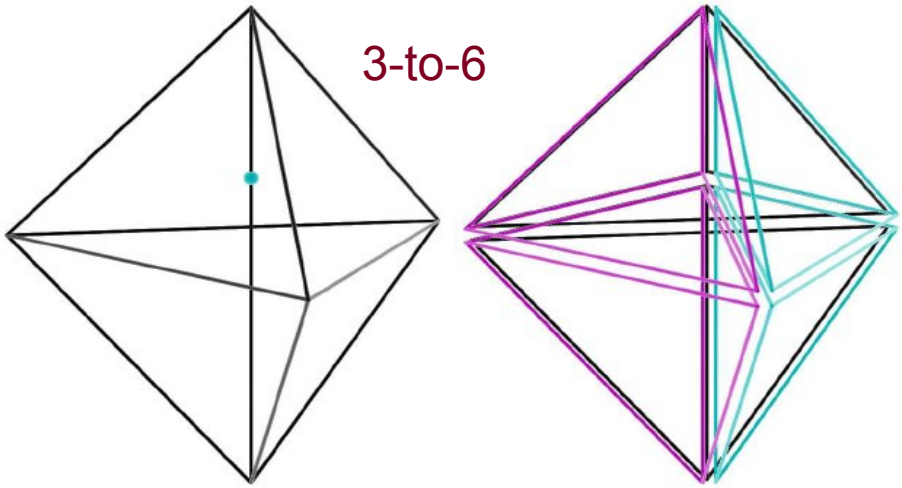


If the **general position assumption** is not fulfilled, degenerate cases can occur. This makes thinks a lot more complicated. One then needs:

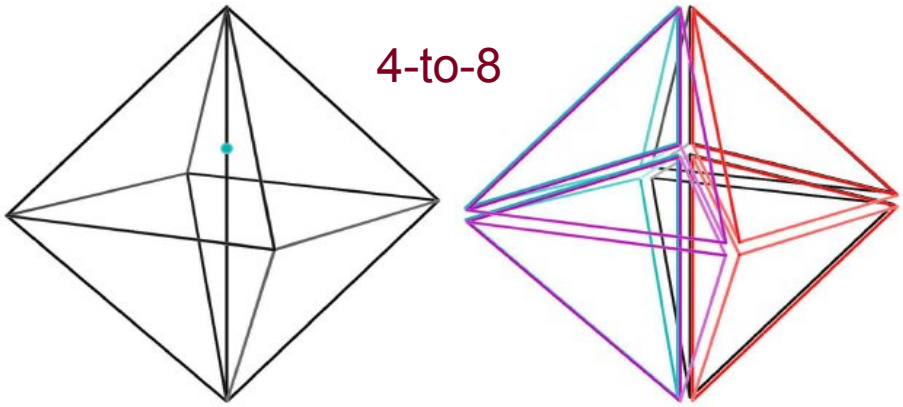
- **1-to-N flips** for point insertion when the point lies on an edge
- **2-to-6 flips** if the point lies on a face
- **4-to-4 flips** for reestablishing Delaunayhood
- Accurate geometric predicates required (difficult! Occasionally requires *exact* arithmetic)

n-to-2n flips

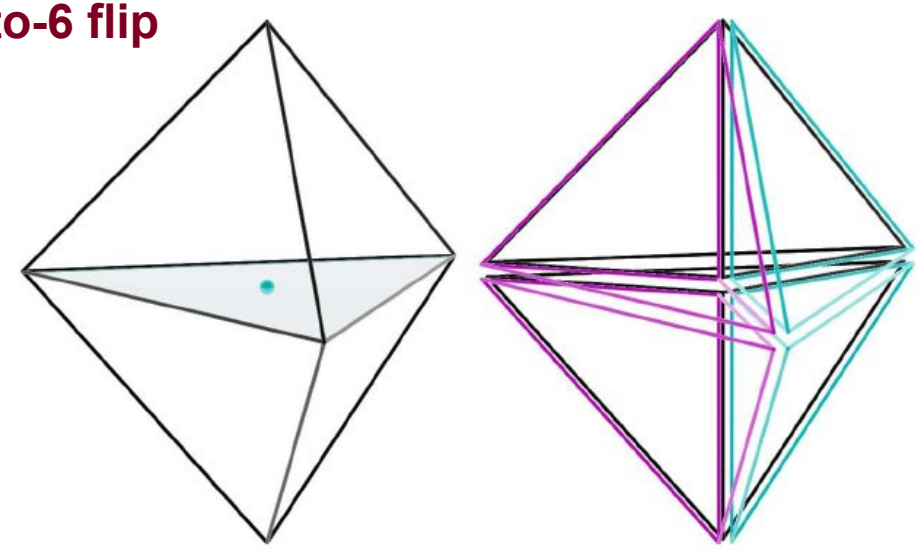
3-to-6



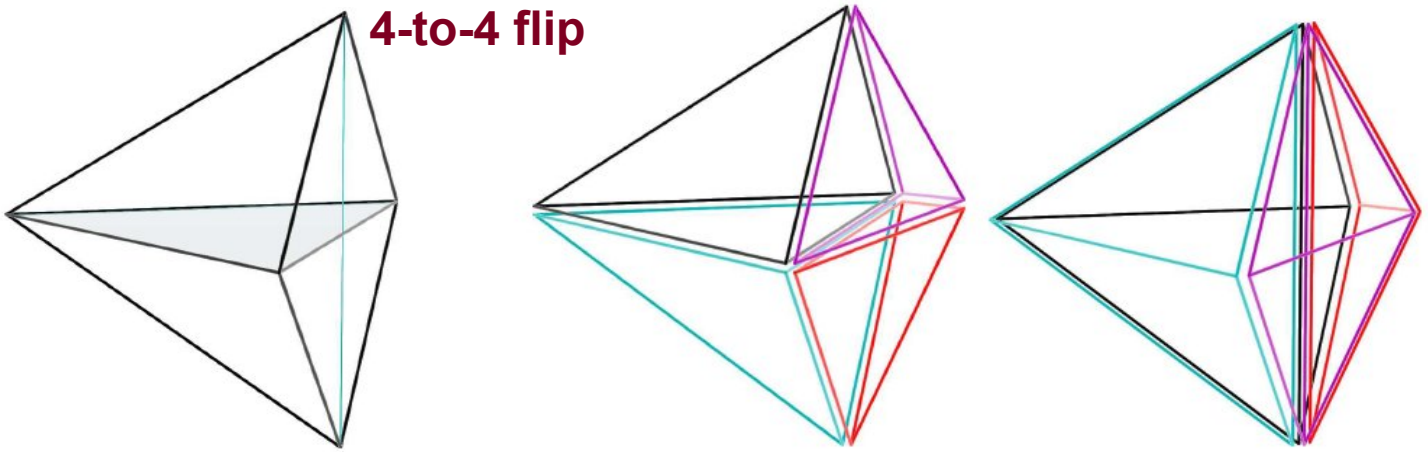
4-to-8



2-to-6 flip

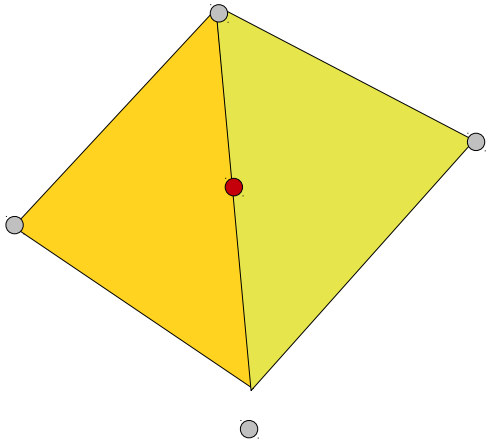


4-to-4 flip



Degenerate point configurations cause trouble – exact arithmetic is required to guarantee robustness

USE OF EXACT ARITHMETIC TO DEAL WITH POINTS IN NON-GENERAL POSITION



Is the point in the left or right triangle?

Or is it *exactly* on the line?

(boils down to evaluating the sign of geometric tests)

$$T_{\mathrm{InCircle}}(a,b,c,d) = \begin{vmatrix} 1 & a_x & a_y & a_x^2+a_y^2 \\ 1 & b_x & b_y & b_x^2+b_y^2 \\ 1 & c_x & c_y & c_x^2+c_y^2 \\ 1 & d_x & d_y & d_x^2+d_y^2 \end{vmatrix}$$

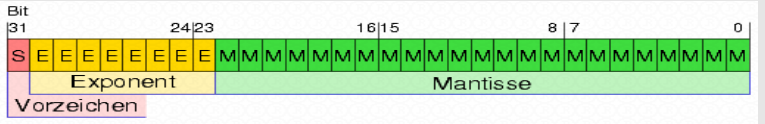
Delaunay algorithms tend to crash if wrong decisions are made!

Solution

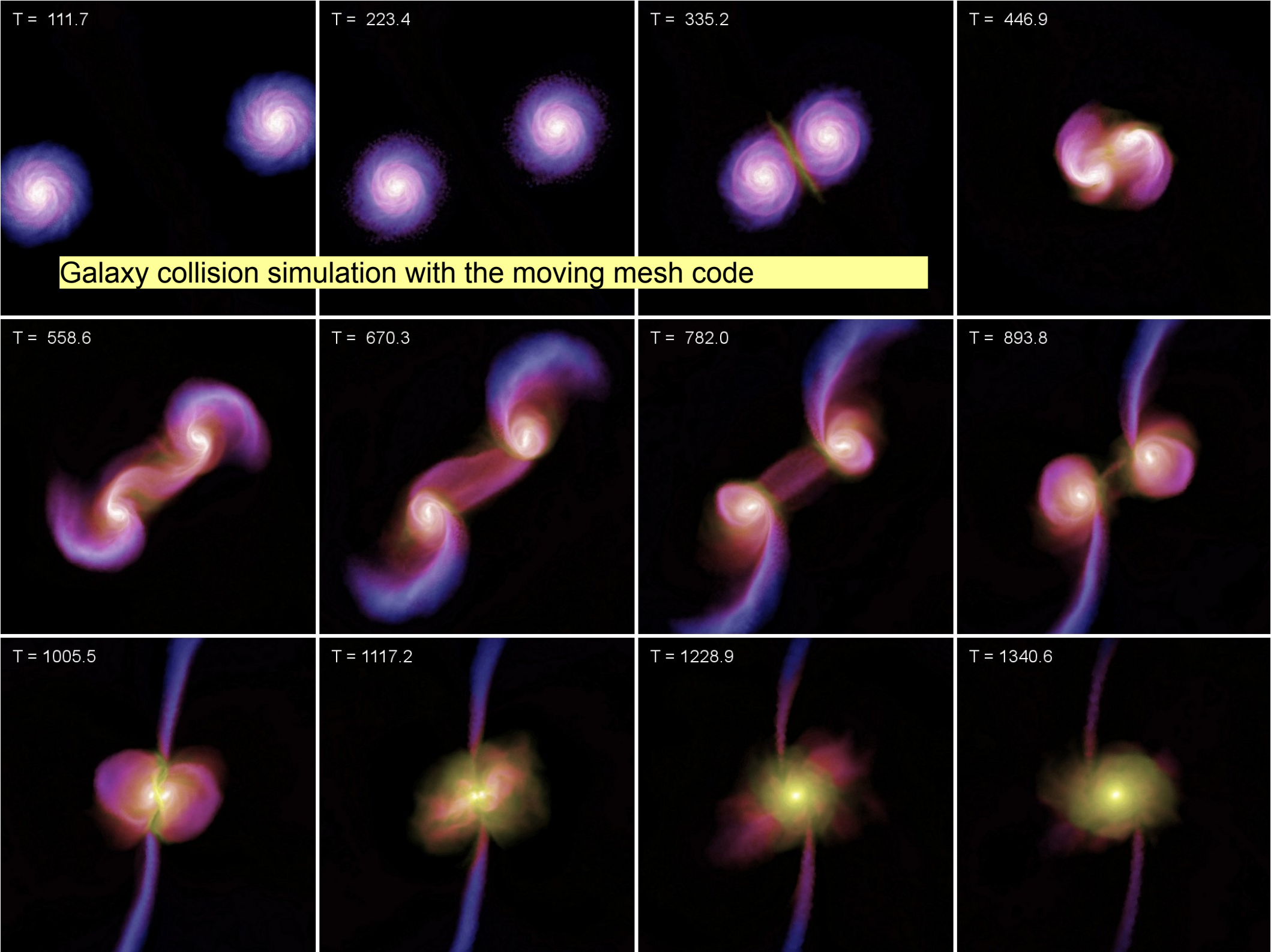
- **Calculate maximum round-off error** in geometric tests, and check whether result could be incorrect
- If the decision is ambiguous due to floating point round-off, **use exact arithmetic** instead

We use **exact integer arithmetic** if needed:

- Domain is mapped to floating point numbers in the range [1.0, 2.0]



- Mantissa provides a 53-bit integer with a unique one-to-one mapping to the floating point numbers
- Carry out the geometric test with the GMP-library using long integers

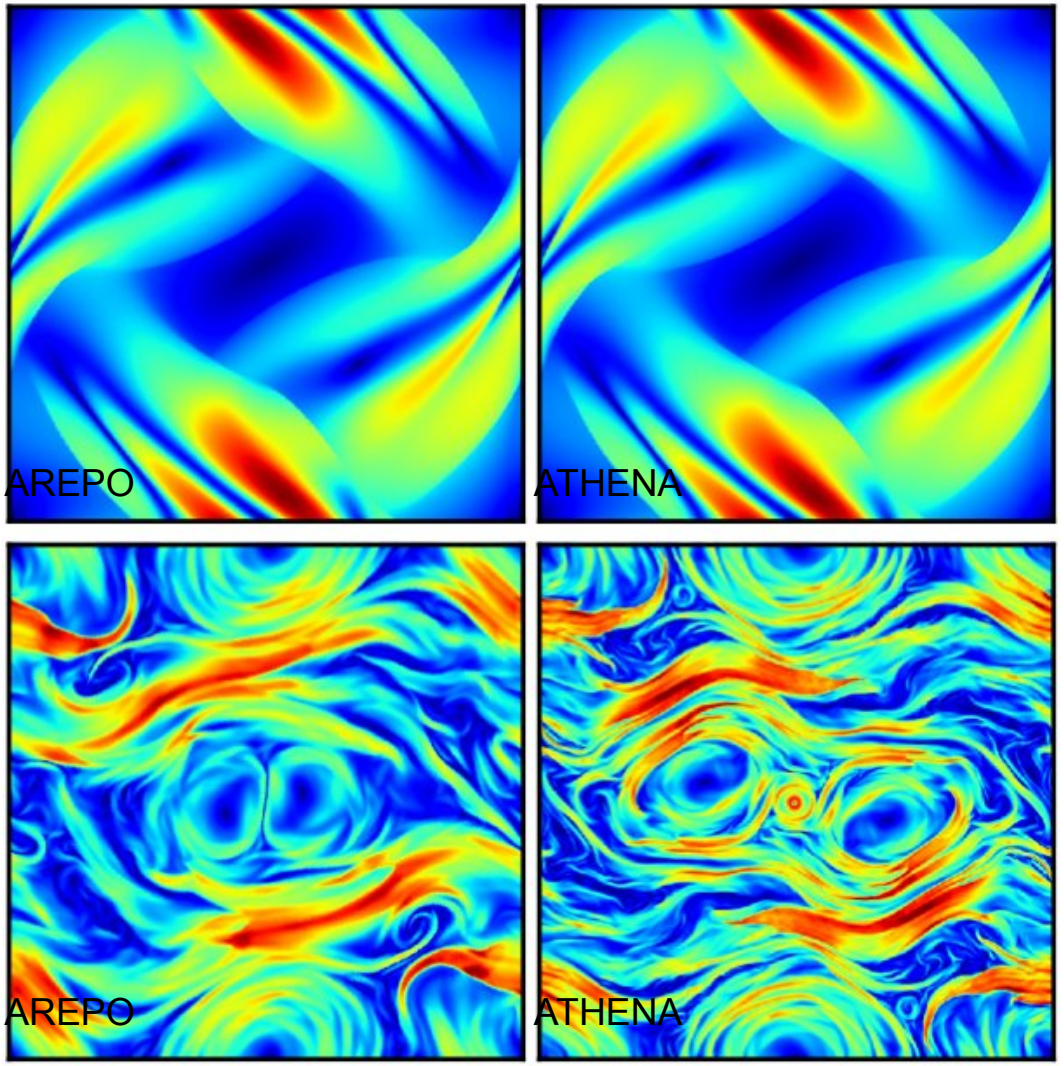

T = 111.7
T = 223.4
T = 335.2
T = 446.9
Galaxy collision simulation with the moving mesh code
T = 558.6
T = 670.3
T = 782.0
T = 893.8
T = 1005.5
T = 1117.2
T = 1228.9
T = 1340.6

There is an MHD implementation in AREPO that works reasonably well

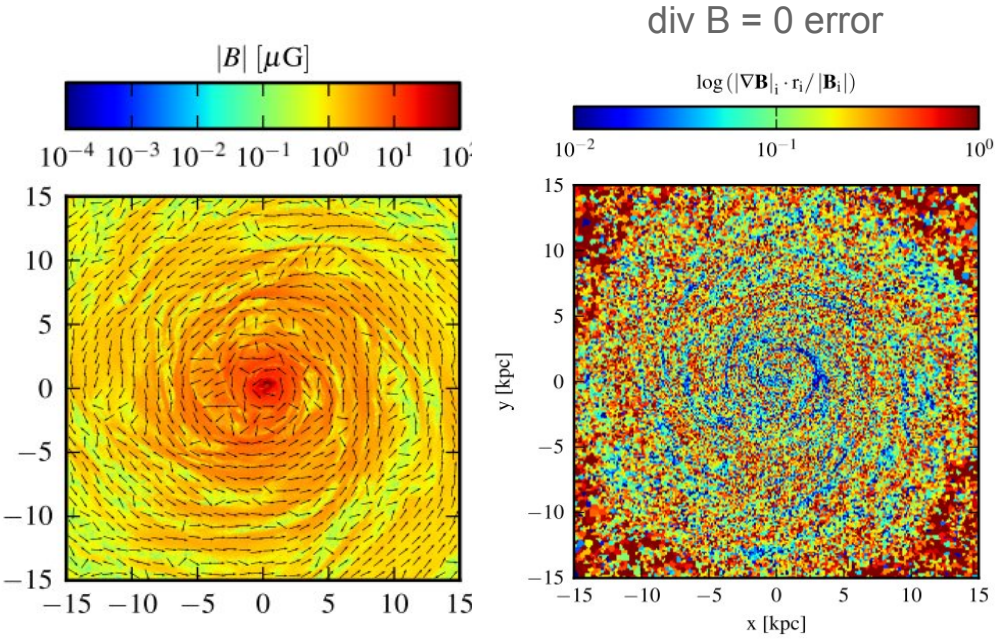
EQUATIONS AND SOME TESTS

$$\mathbf{U} = \begin{pmatrix} \rho \\ \rho\mathbf{v} \\ \rho e \\ \mathbf{B} \\ \psi \end{pmatrix} \qquad \mathbf{F}(\mathbf{U}) = \begin{pmatrix} \rho\mathbf{v} \\ \rho\mathbf{v}\mathbf{v}^T + p - \mathbf{B}\mathbf{B}^T \\ \rho e\mathbf{v} + p\mathbf{v} - \mathbf{B}\,(\mathbf{v}\cdot\mathbf{B}) \\ \mathbf{B}\mathbf{v}^T - \mathbf{v}\mathbf{B}^T + \psi I \\ c_h^2\mathbf{B} \end{pmatrix}$$

Orszag-Tang vortex test



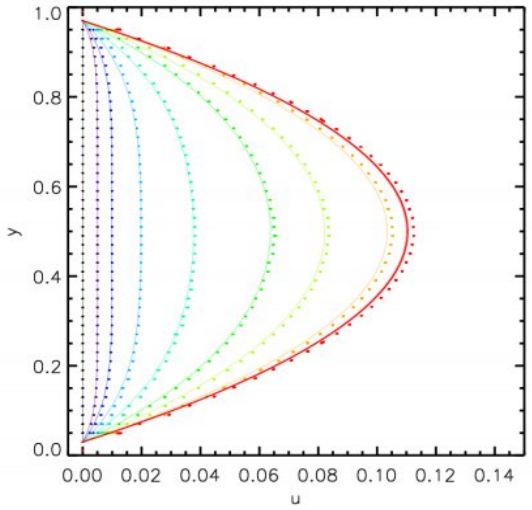
Magnetic field in a disk galaxy



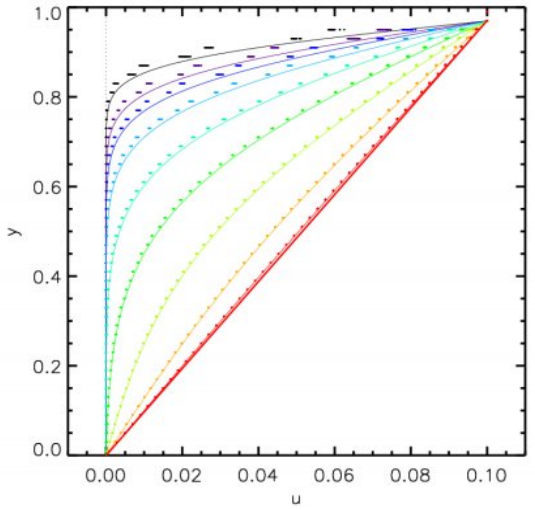
Pakmor, Bauer & Springel (2011)

Explizit physical viscosity has been added to AREPO to obtain a **Navier-Stokes solver** on a moving mesh

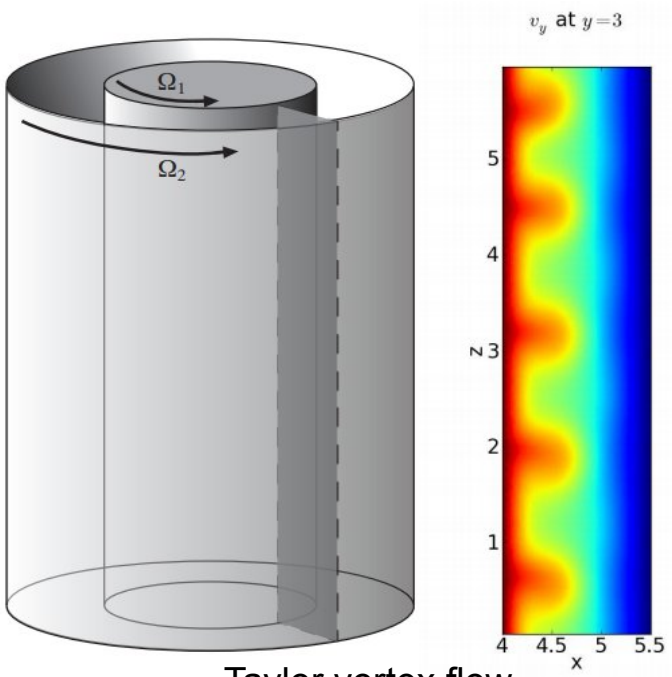
SOME BASIC EXAMPLES



(a) Plane Poiseuille flow

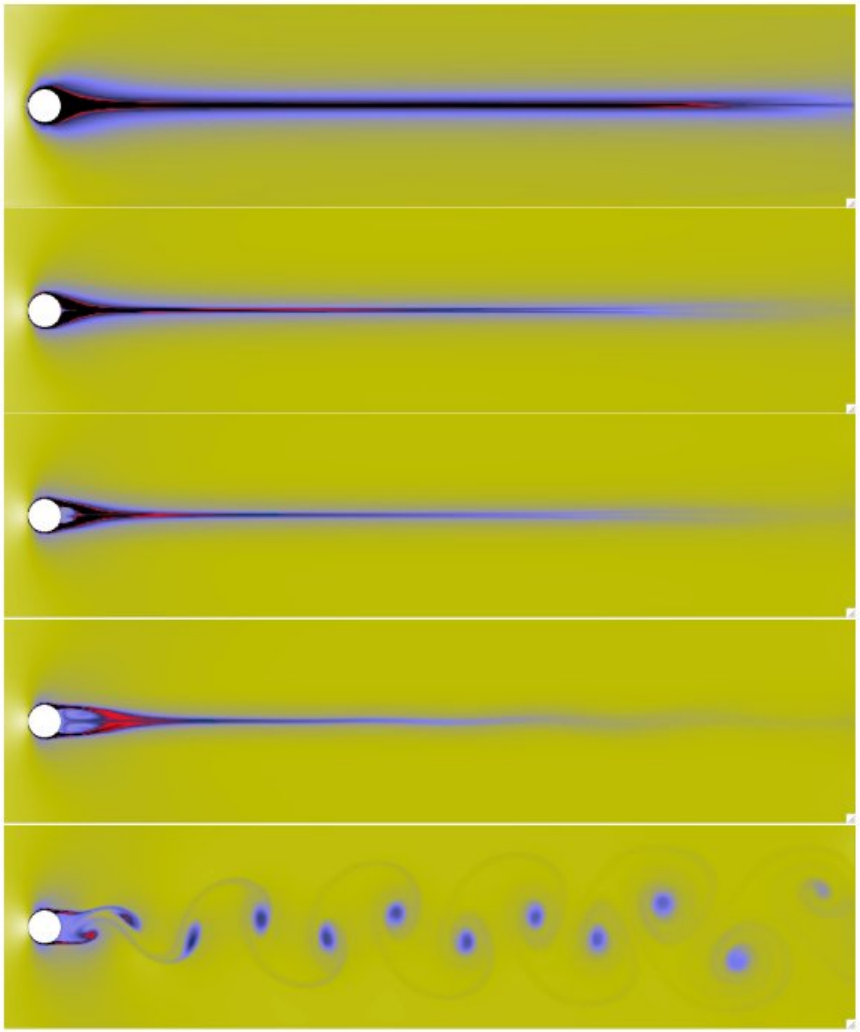


(b) Plane Couette flow



Taylor vortex flow

Munoz, VS et al. (2012)



But in the end: **Does it matter for galaxy formation?**

Moving-mesh cosmology: First applications of AREPO

Mark Vogelsberger
Debora Sijacki
Dusan Keres
Paul Torrey
Lars Hernquist
Volker Springel

4 new papers, astro-ph (2011)

20 Mpc/h box, WMAP7 cosmology

Resolutions: 2 x 128^3, 2 x 256^3, 2 x 512^3

AREPO and GADGET runs

equal physics, equal gravity solver

Andreas Bauer & VS (2011)

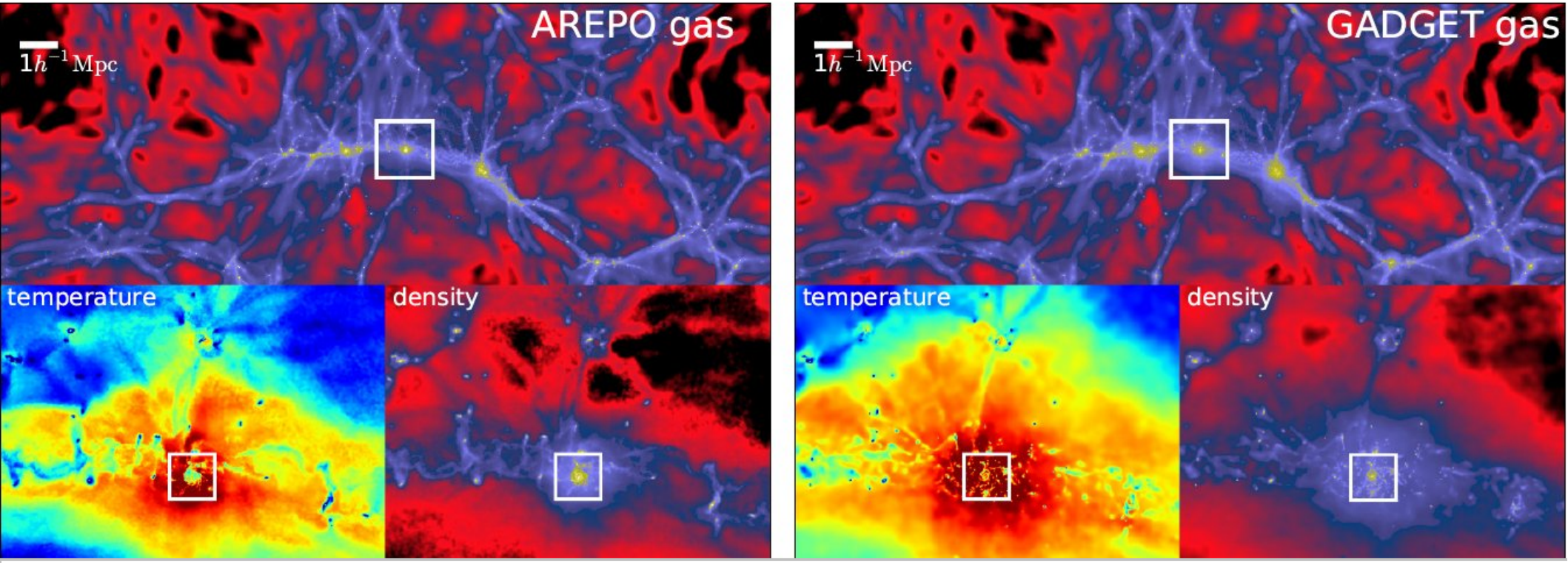
Subsonic turbulence in moving-mesh and SPH

Thomas Greif, VS, et al. (2011)

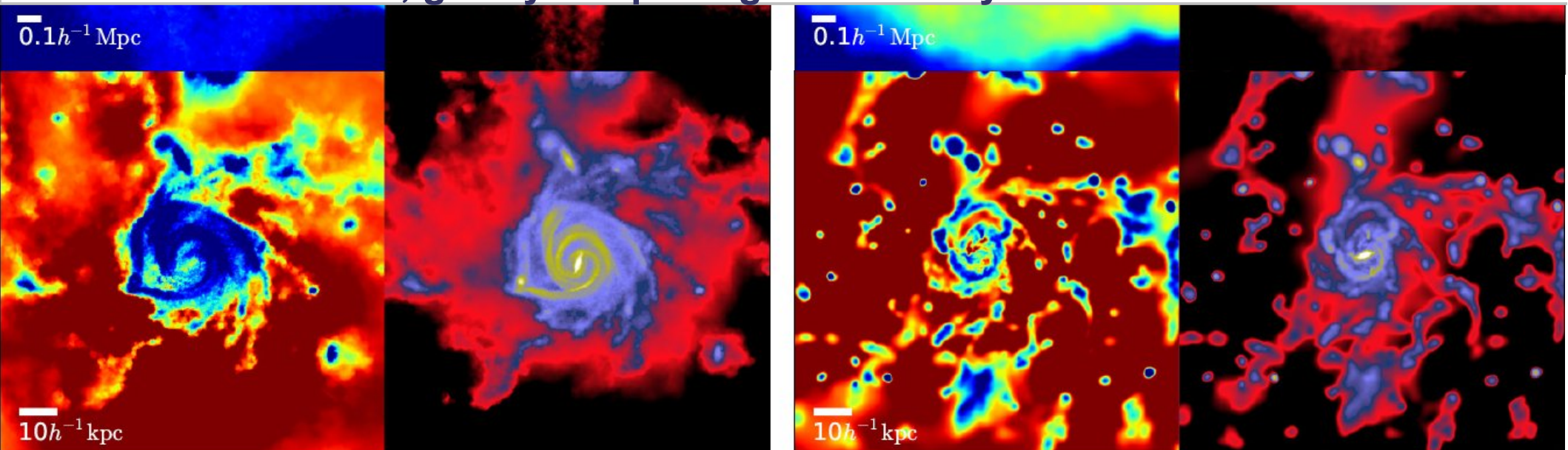
Population III star formation

On large scales, the code produces similar results as standard SPH techniques

GAS AND TEMPERATURE FIELDS IN A COSMOLOGICAL HYDRODYNAMIC SIMULATION

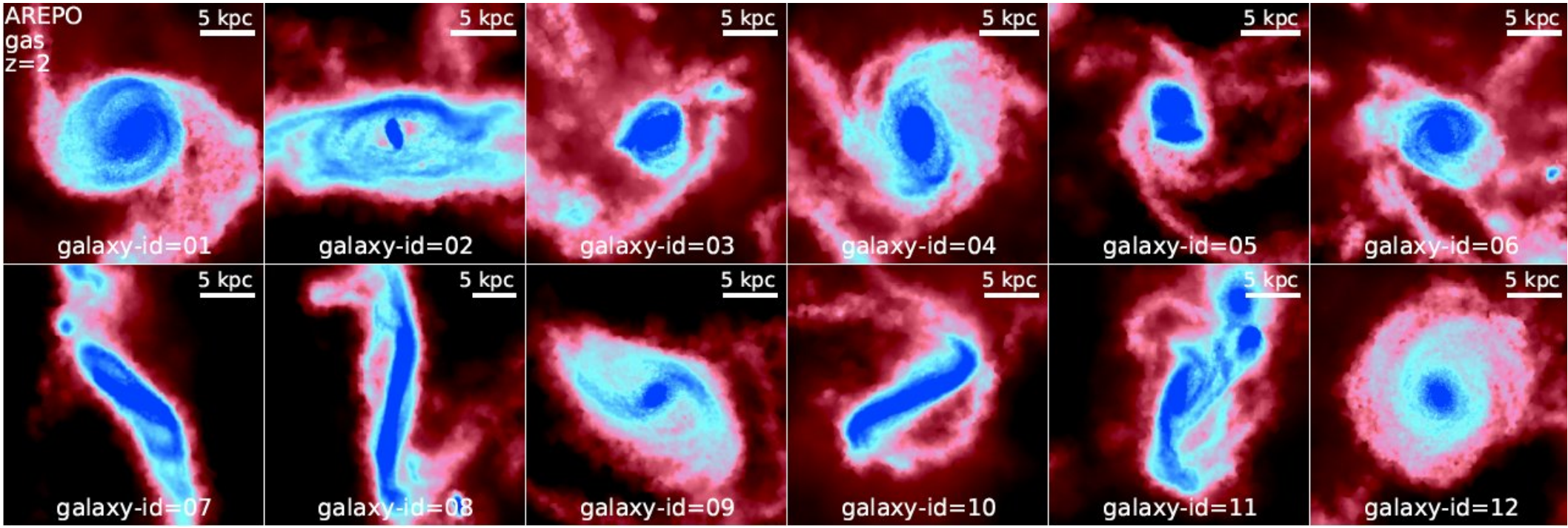


But on small scales, galaxy morphologies look very different

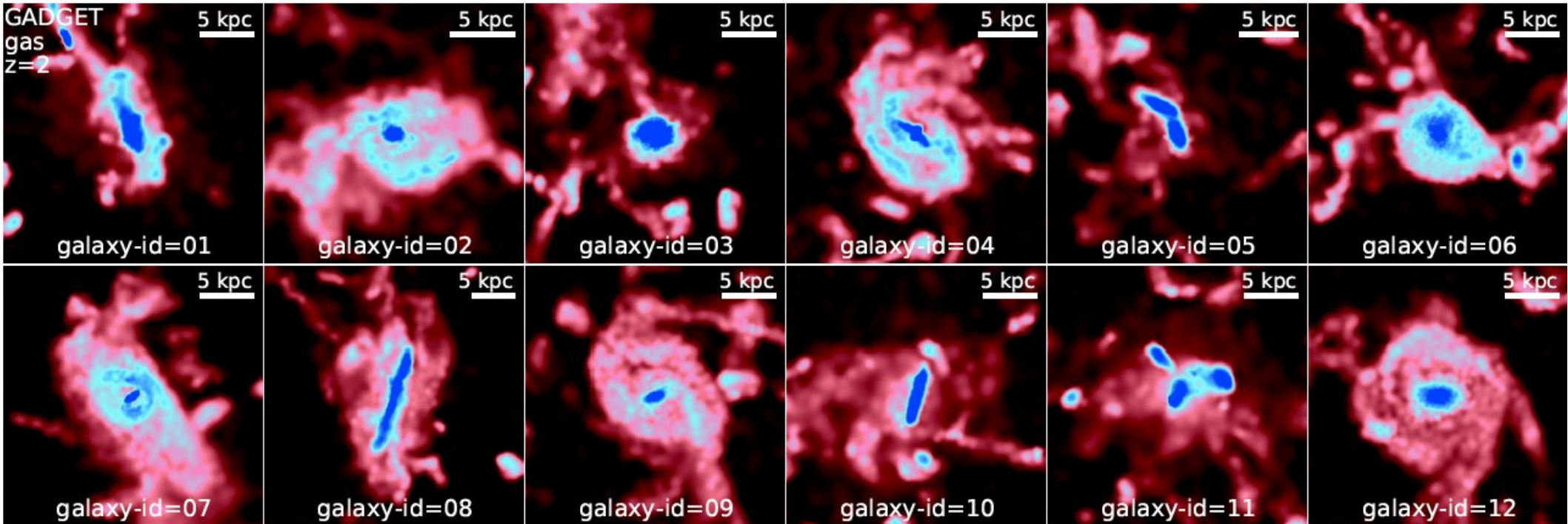


AREPO:

Projected gas densities in matching AREPO and SPH halos

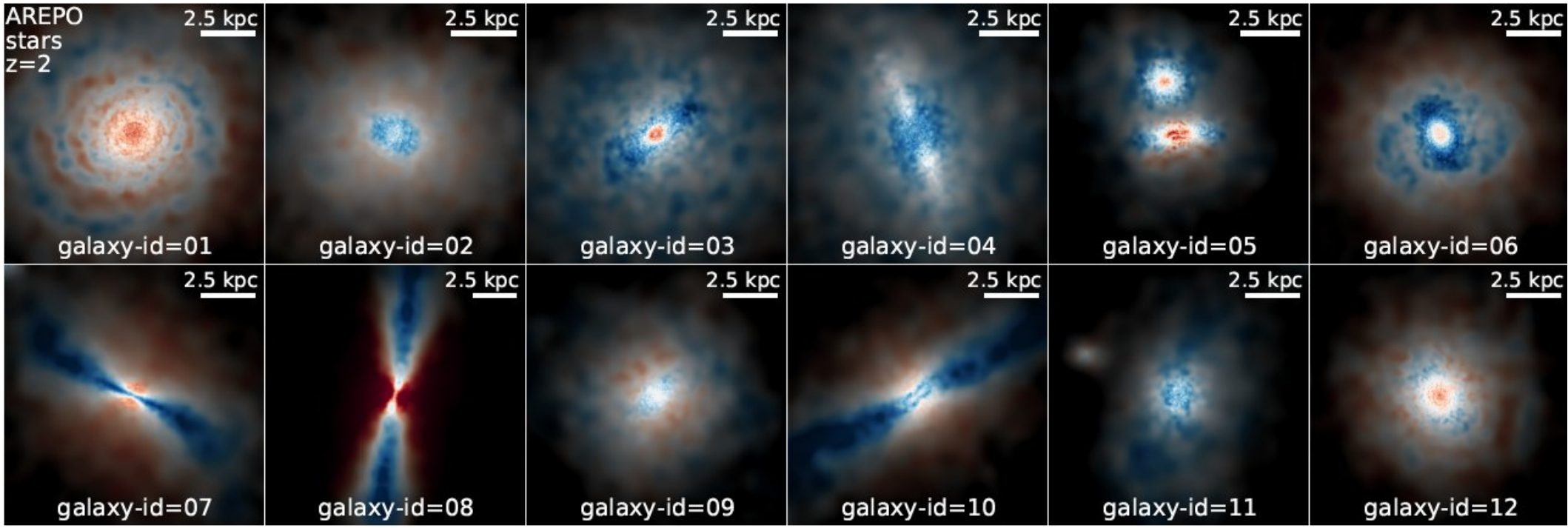


SPH:

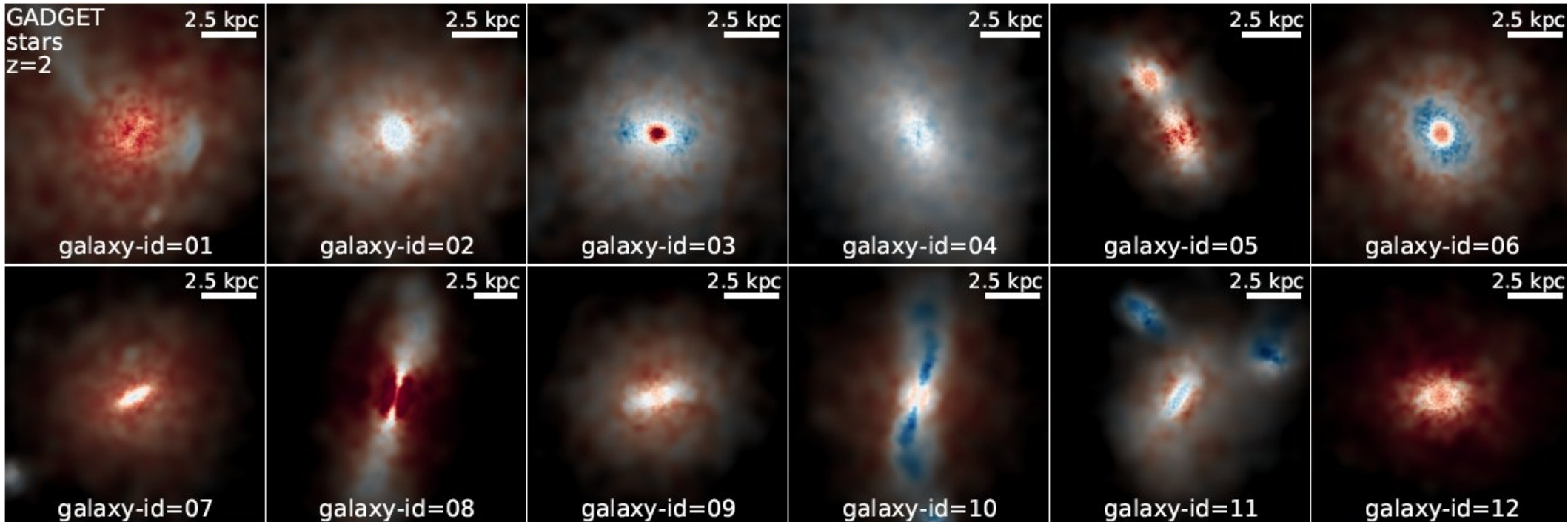


AREPO:

Projected stellar densities in matching AREPO and SPH halos



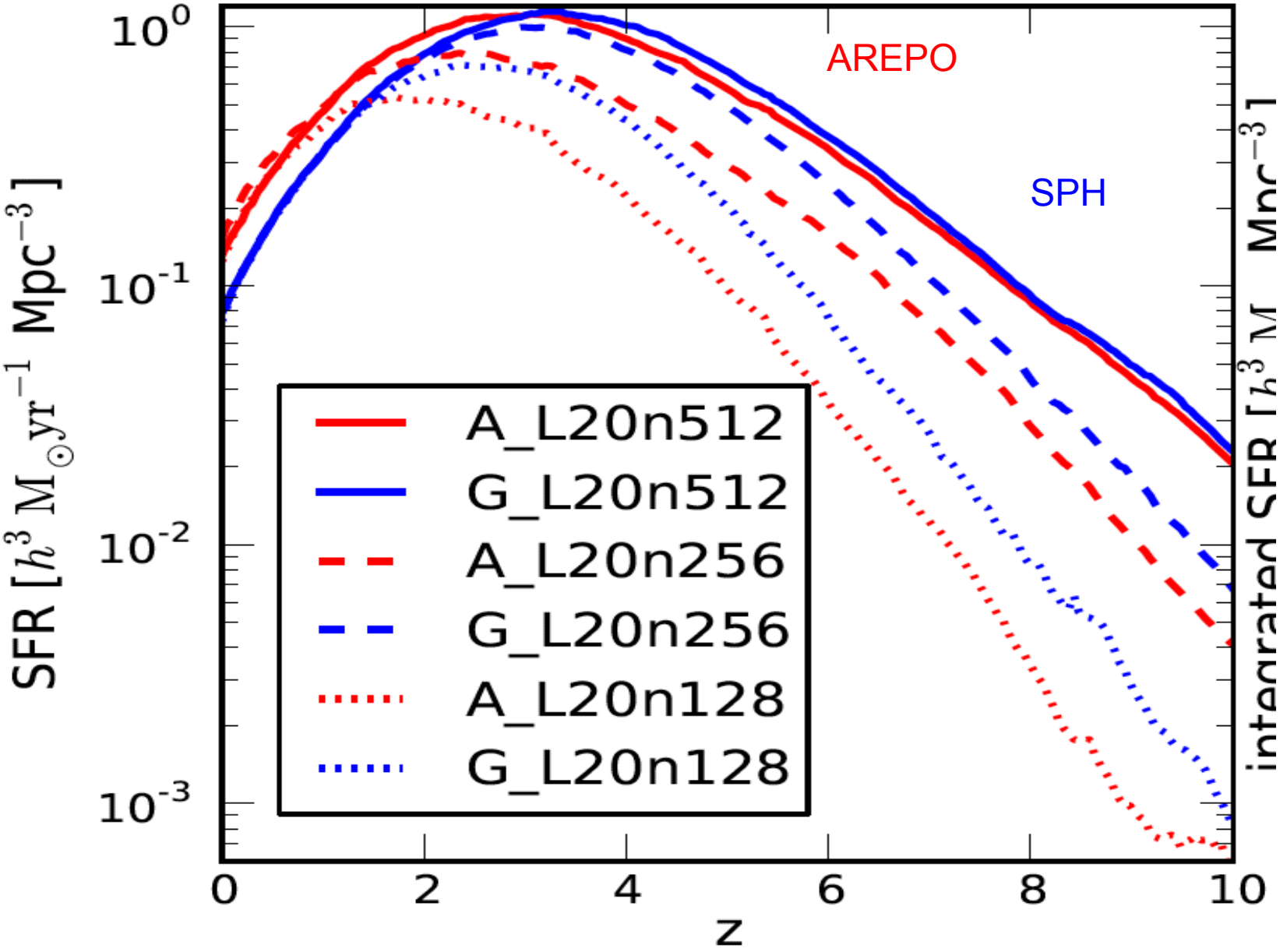
SPH:



Compared with SPH, the cosmic star formation rate density is higher in AREPO at low redshift

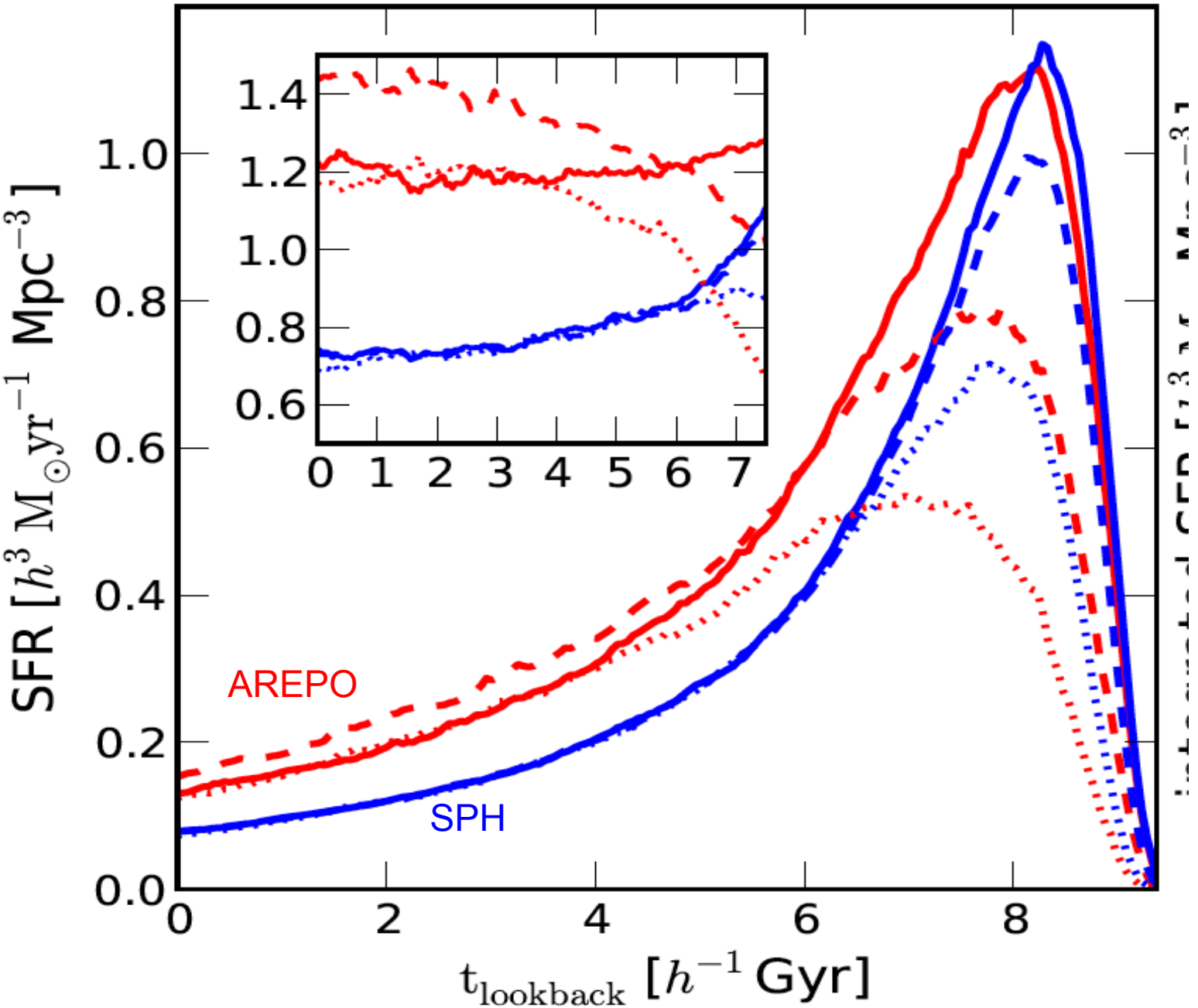
SFR-DENSITY AS A FUNCTION OF REDSHIFT FOR DIFFERENT RESOLUTIONS AND CODES

Vogelsberger et al. (2011)



Compared with SPH, the cosmic star formation rate density is higher in AREPO at low redshift

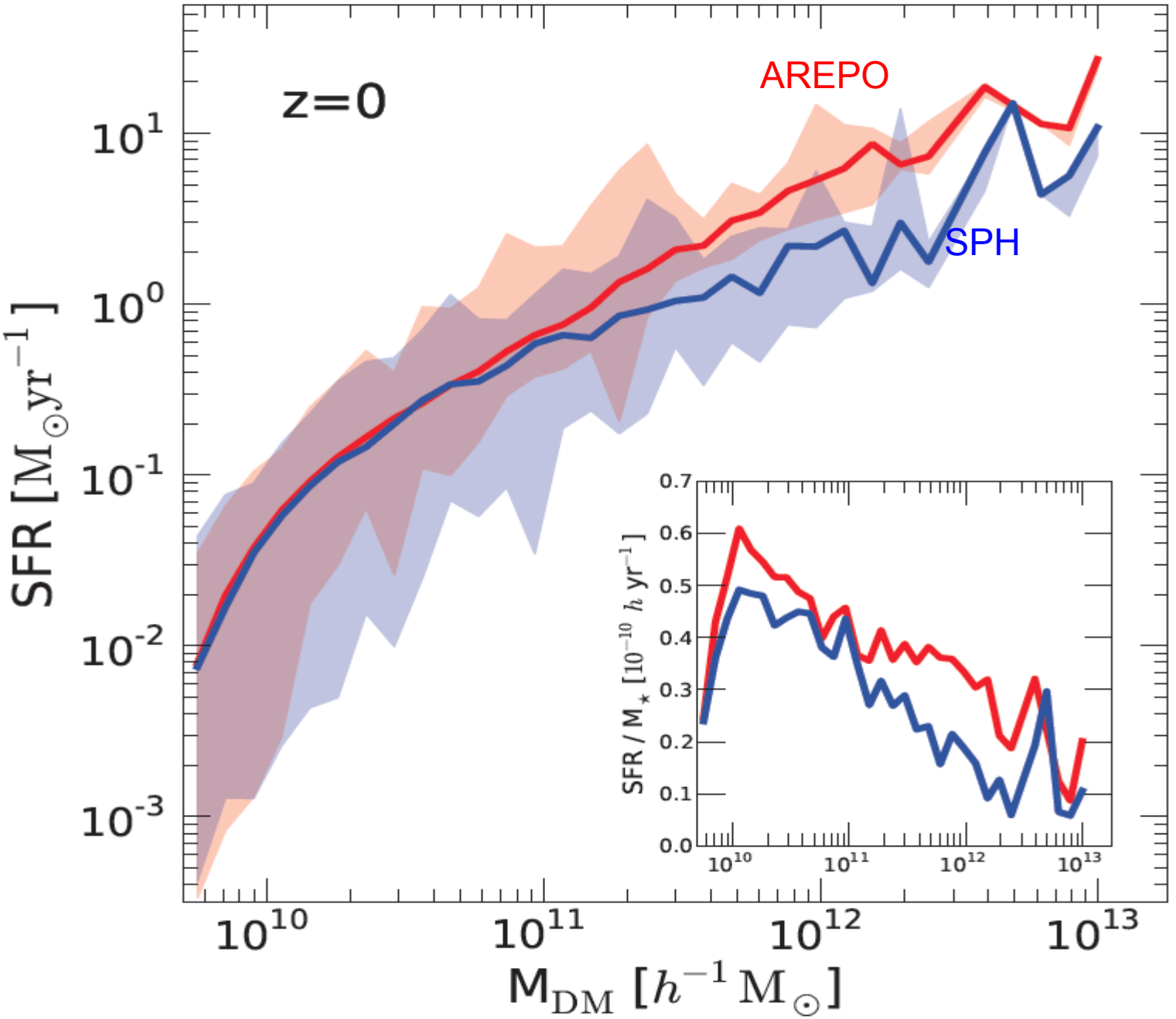
SFR-DENSITY AS A FUNCTION OF TIME FOR DIFFERENT RESOLUTIONS AND CODES



The difference in star formation originates in massive halos

STAR FORMATION RATE AS A FUNCTION OF HALO MASS

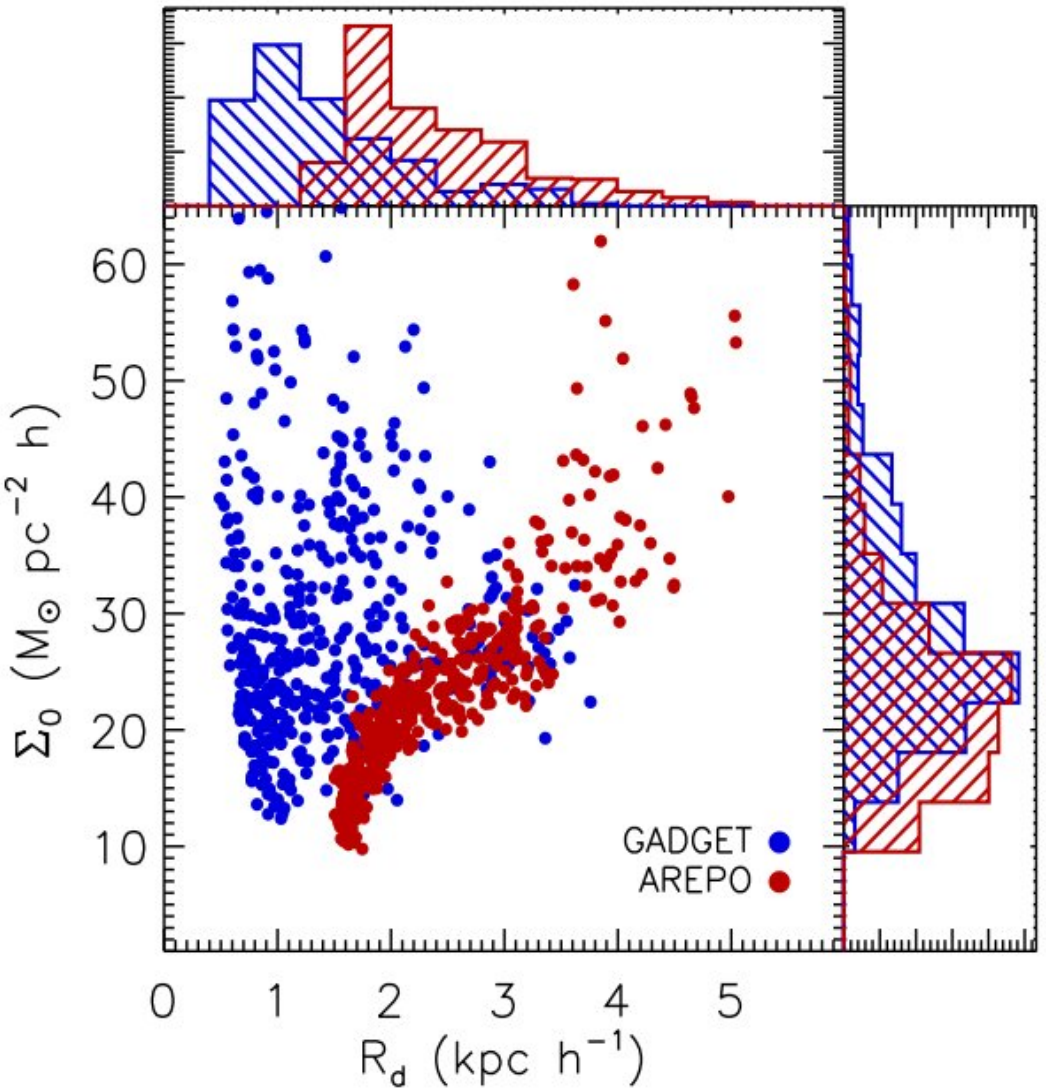
Vogelsberger et al. (2011)

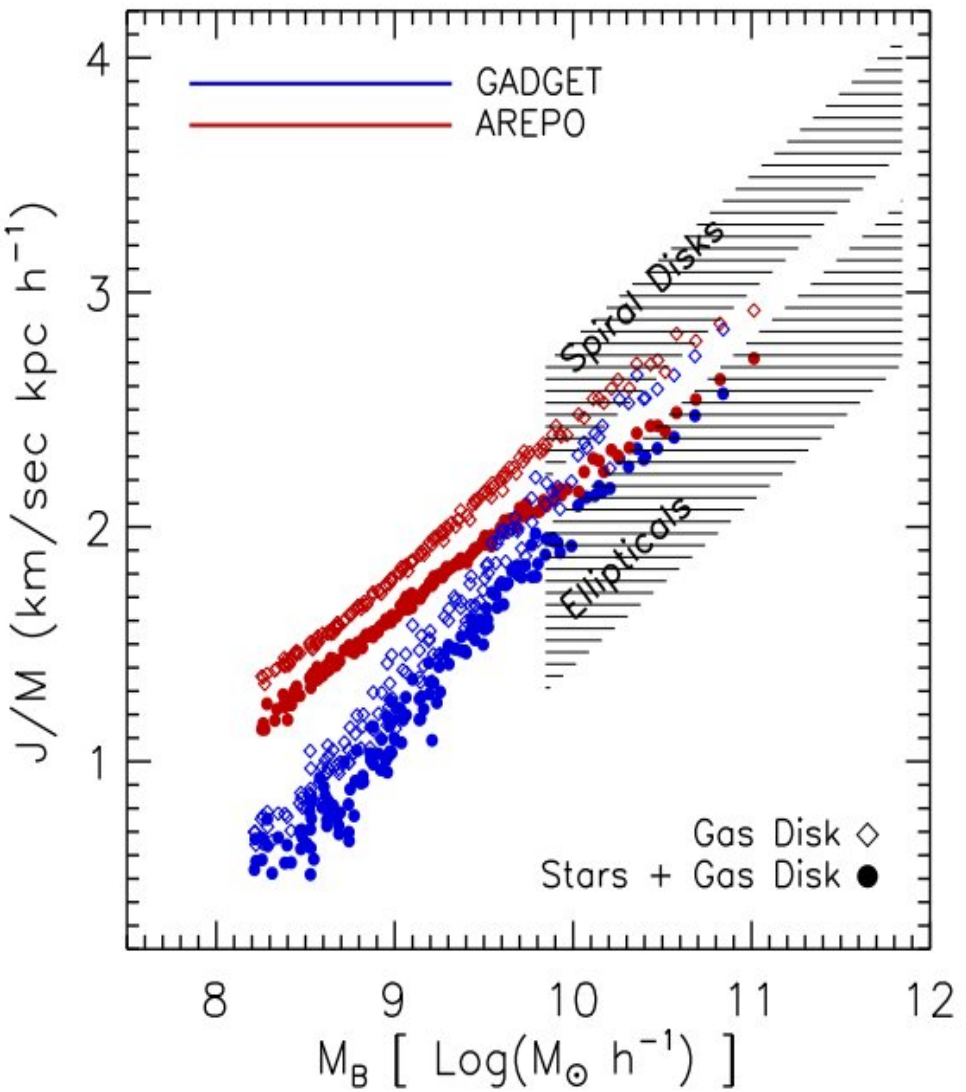


Gasous disk scale lengths are much larger in the moving-mesh code

DISK SCALE LENGTHS AND ANGULAR MOMENTUM IN GADGET AND AREPO

Torrey et al. (2011)

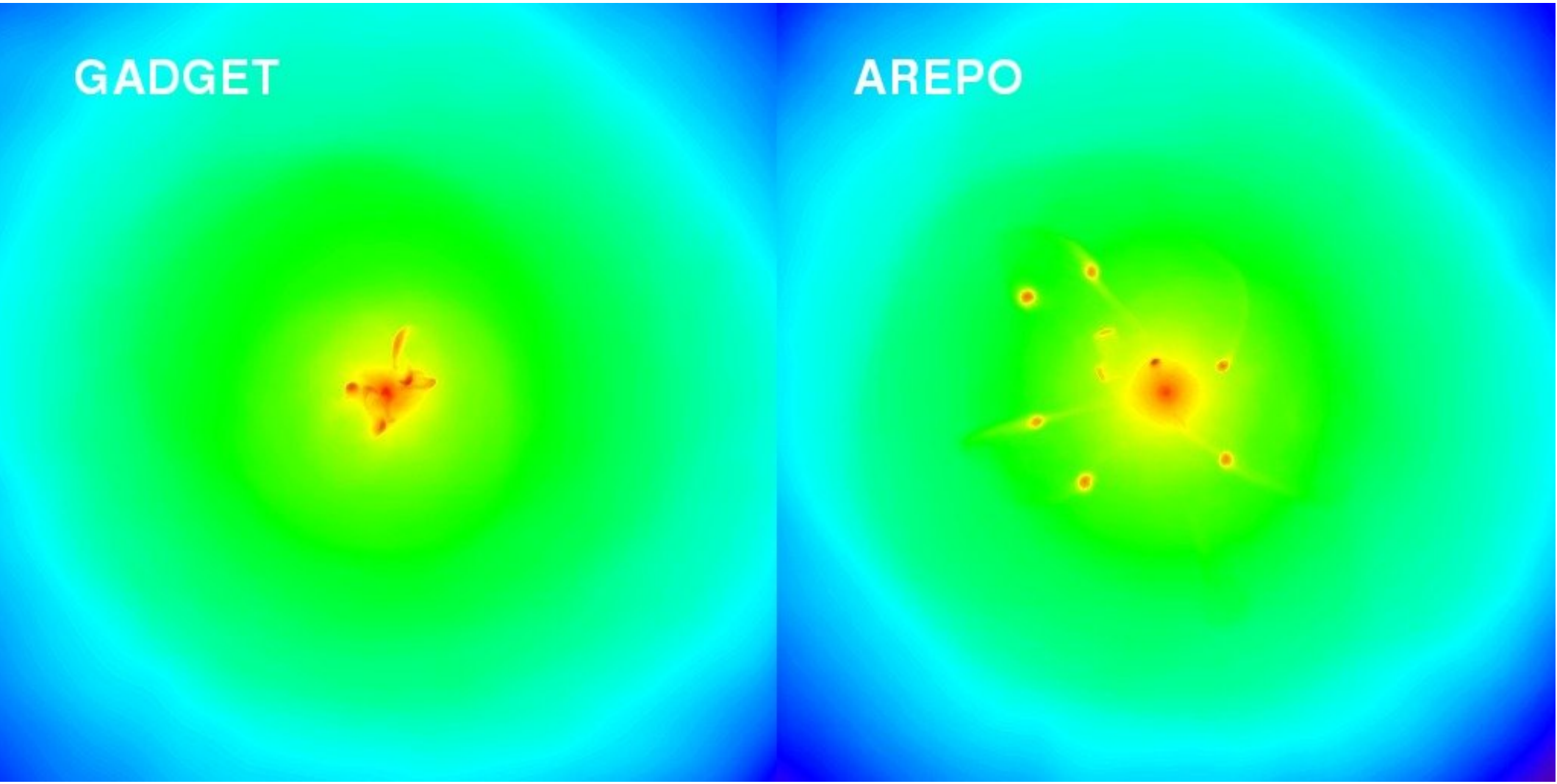




Satellite mass loss and orbitial decay is different in SPH and AREPO

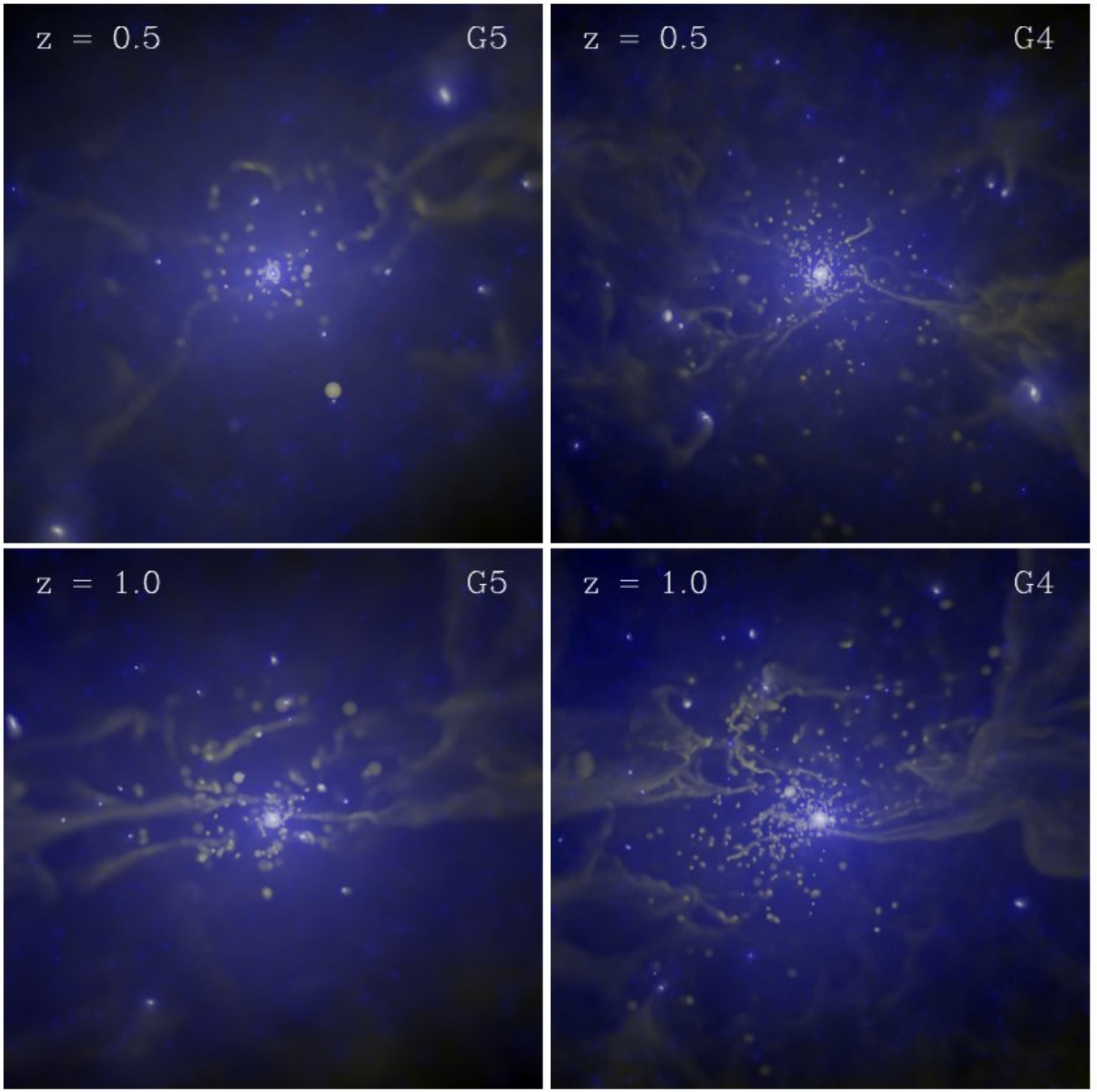
FIDUCIAL GAS BLOBS IN ORBIT IN A CLUSTER

Sijacki et al. (2011)



Clumpy gas distribution around Aquila galaxy in GADGET

GAS BLOBS IN ORBIT AROUND AQUILA AT DIFFERENT TIMES AND RESOLUTIONS

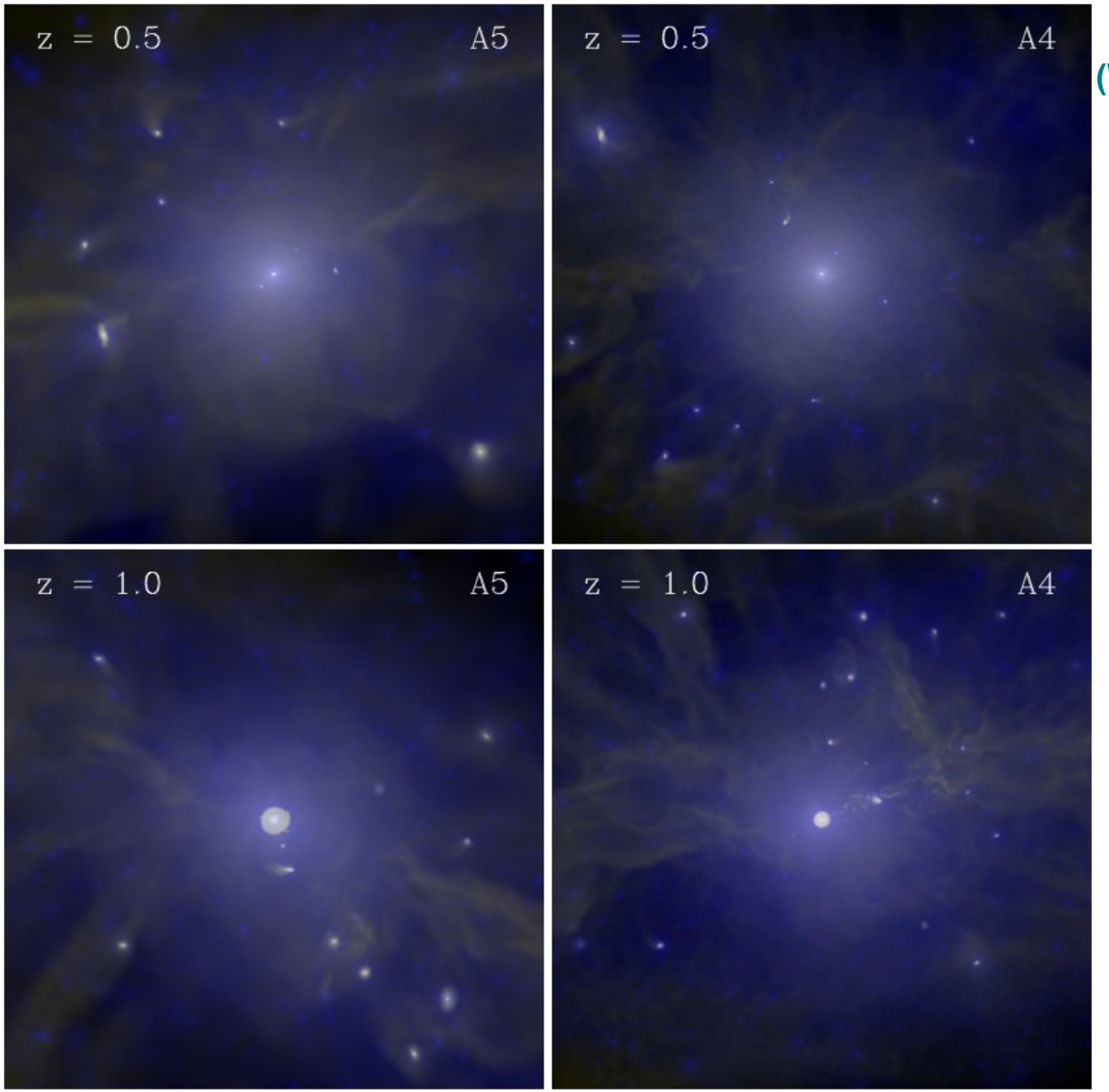


(Wadepuhl & Springel, 2011)

Also seen, e.g, in ERIS (Guedes et al., 2011)

Smooth gas distribution around Aquila galaxy in AREPO

GAS IN THE HALO AT DIFFERENT TIMES AND RESOLUTIONS

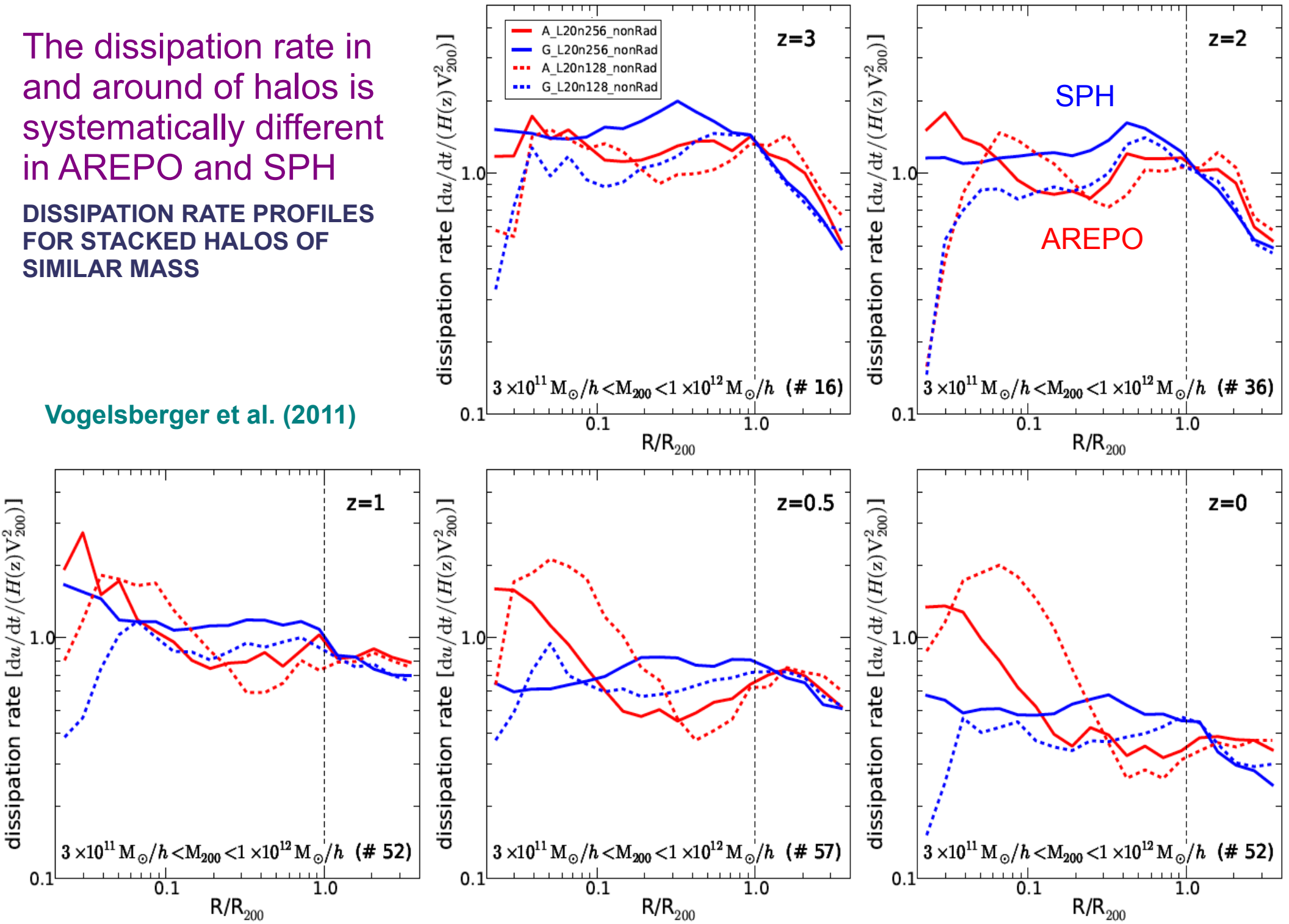


(Wadepuhl & Springel, 2011)

The dissipation rate in and around of halos is systematically different in AREPO and SPH

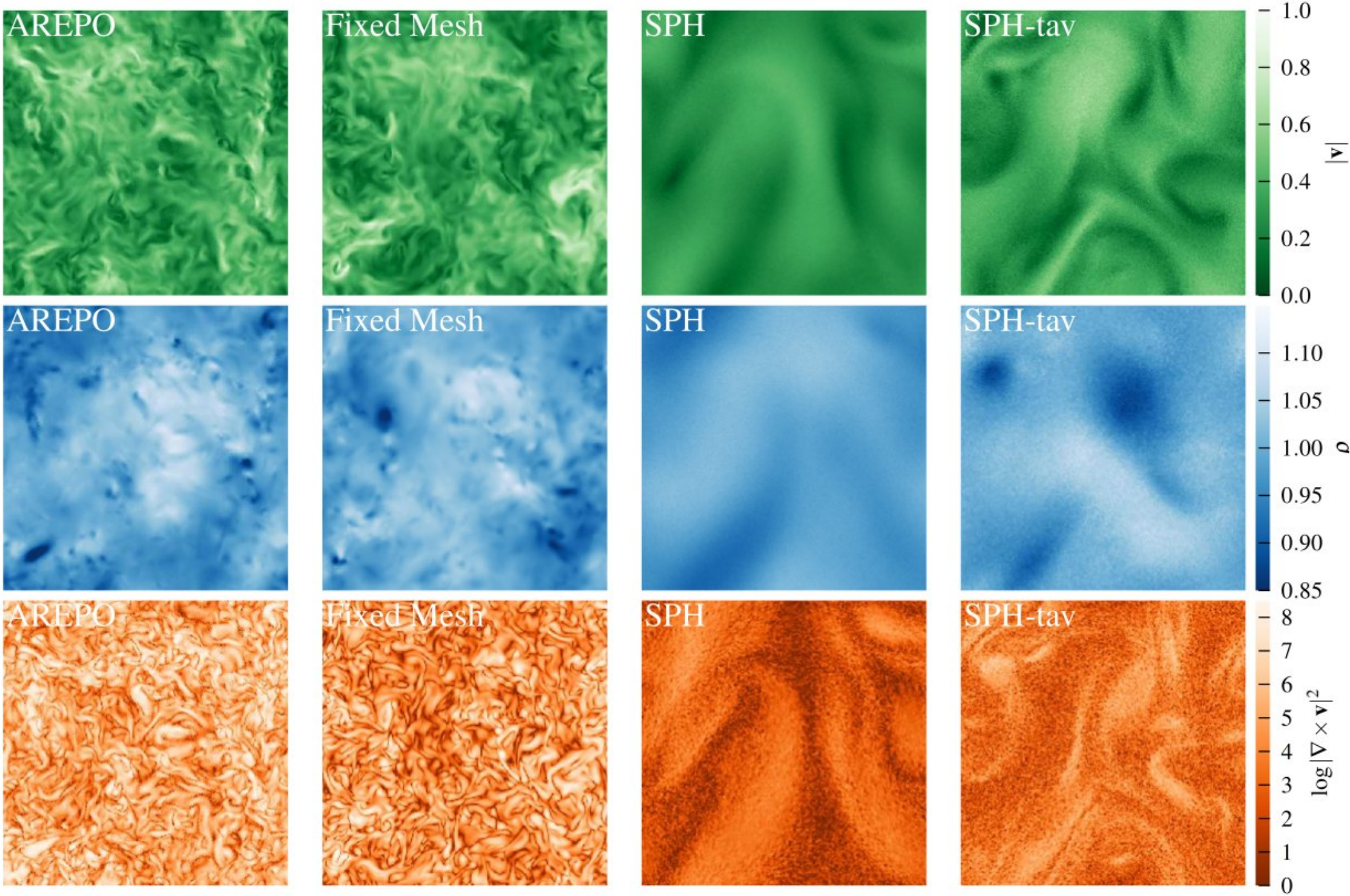
DISSIPATION RATE PROFILES FOR STACKED HALOS OF SIMILAR MASS

Vogelsberger et al. (2011)



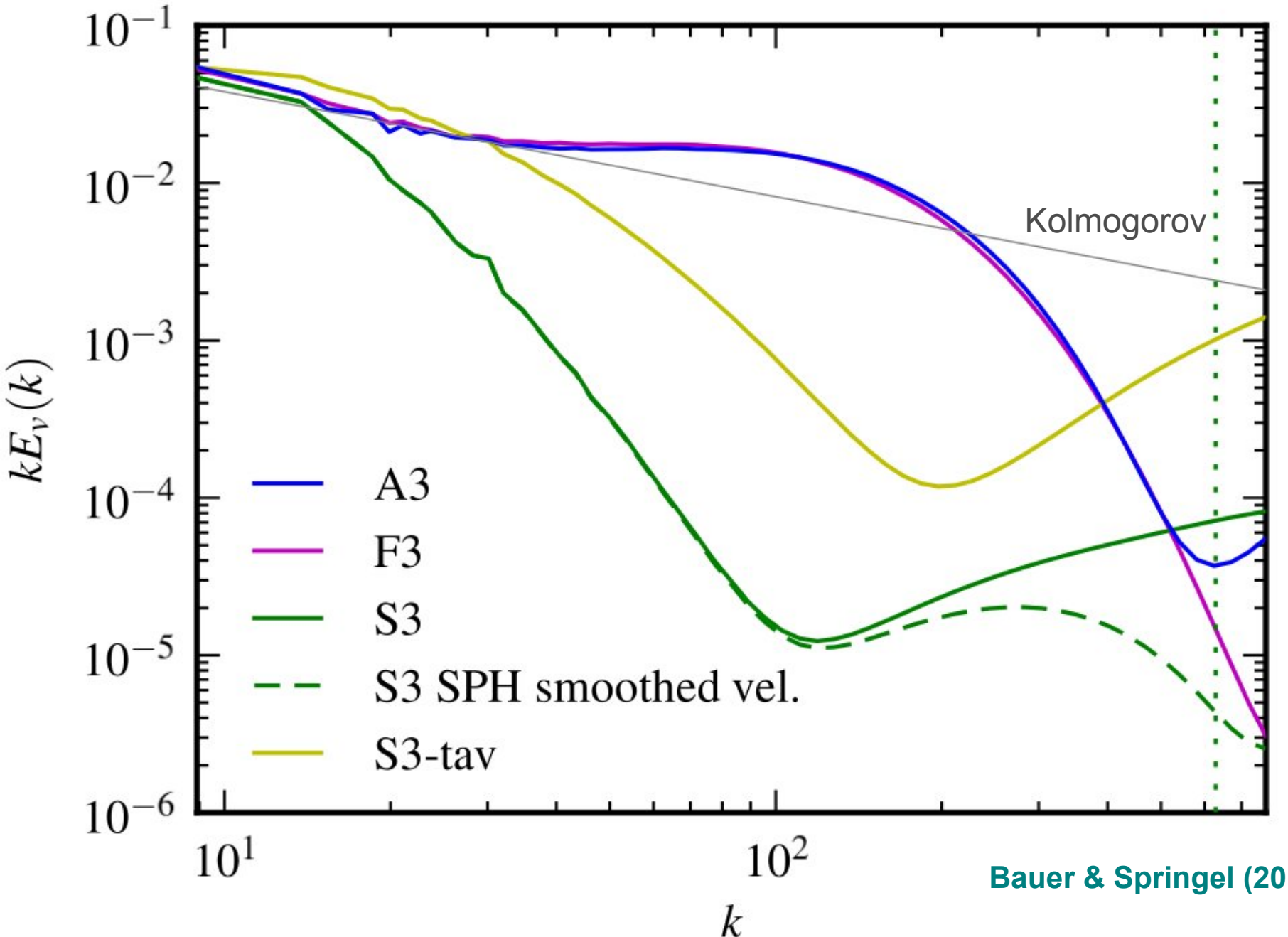
Enstrophy fields in **subsonic** turbulence are different in SPH and mesh-codes

TURBULENT FIELDS FOR EQUAL DRIVING IN DIFFERENT SIMULATION CODES **Bauer & Springel (2012)**



Driven subsonic turbulence in AREPO yields a Kolmogorov cascade

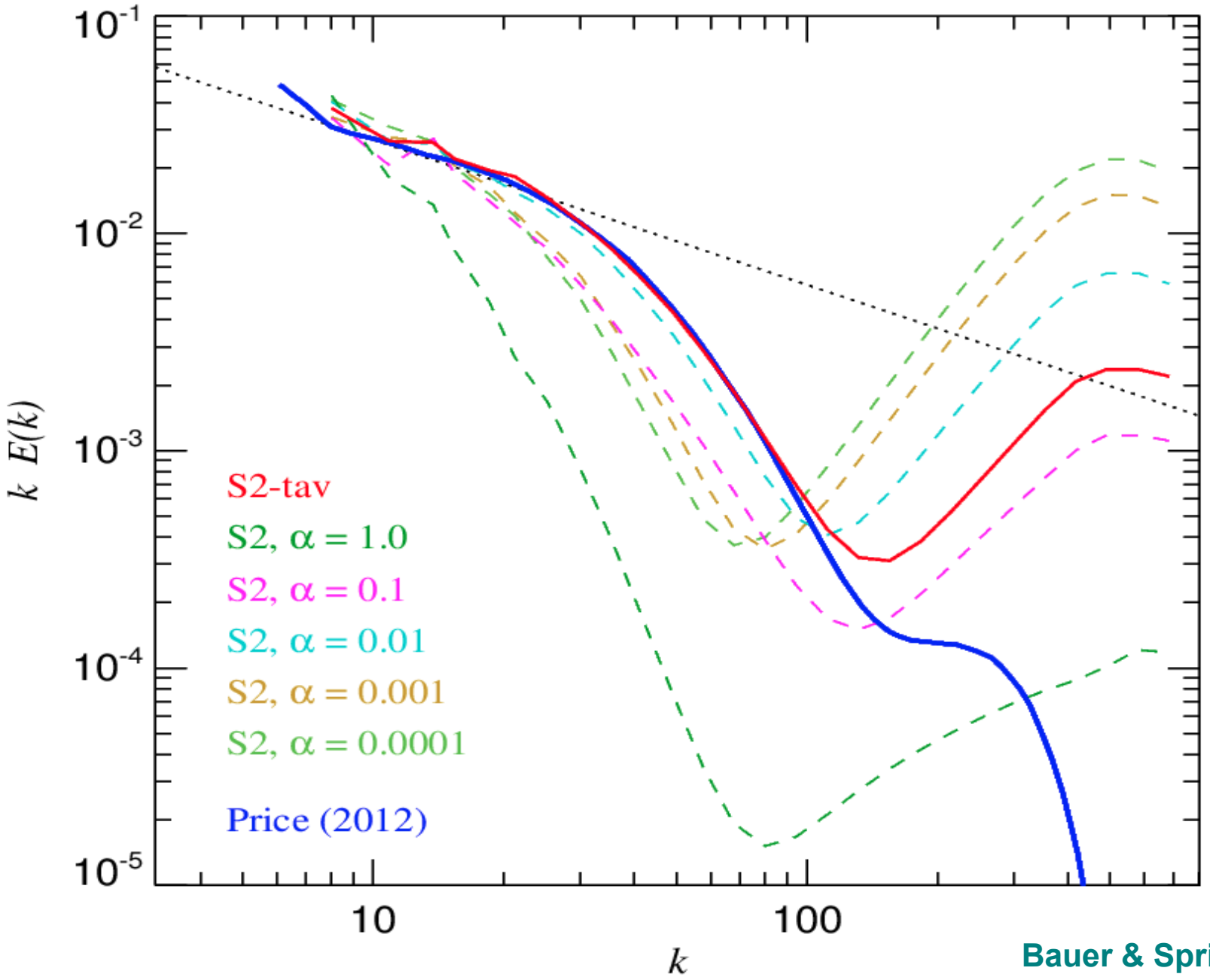
VELOCITY POWER SPECTRUM AT DIFFERENT RESOLUTIONS



Bauer & Springel (2012)

The results of Price are consistent with our own low-viscosity SPH results

VELOCITY POWER SPECTRA FOR DIFFERENT VISCOSITY SETTINGS



Bauer & Springel (2012)

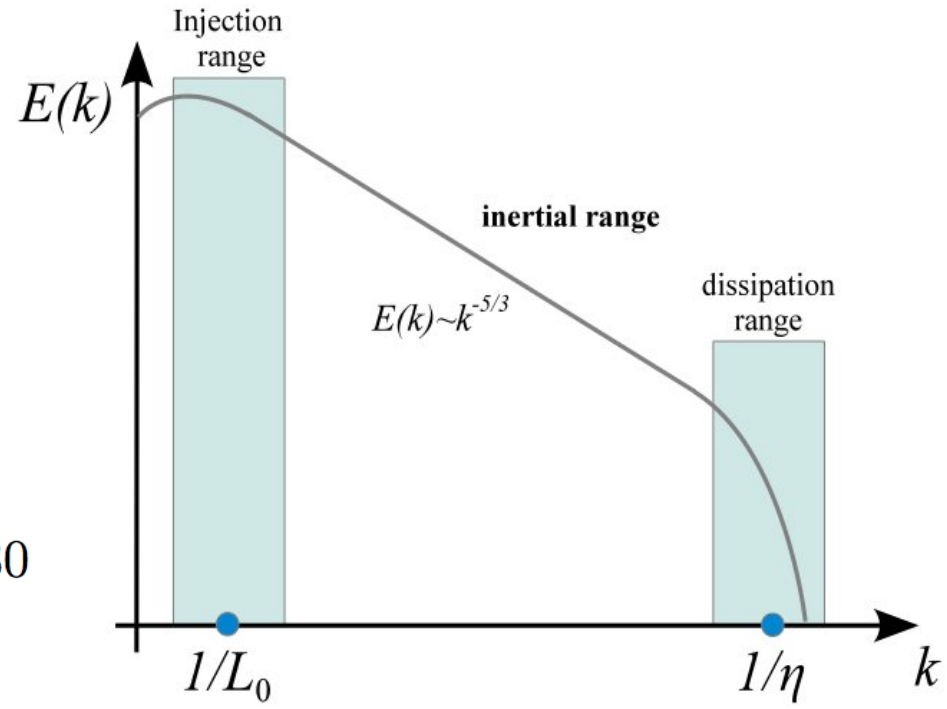
The shape of the dissipation range for Kolmogorv turbulence is universal

REYNOLDS NUMBERS AND THE KOLMOGOROV SCALE

Kolmogorov scale: $$\eta \equiv \left(\frac{\nu^3}{\epsilon}\right)^{1/4}$$

Dynamic range of inertial range: $$\frac{\eta}{L_0} \sim \mathrm{Re}^{-\frac{3}{4}}$$

$$\text{For } \mathrm{Re} = 6000 \text{ expect } \frac{L_0}{\eta} \sim 680$$



Universality of Kolmogorov turbulence also applies to the dissipation range!

For a Navier-Stokes flow with kinematic viscosity ν:

$$E(k) = C\,\epsilon^{2/3}\,k^{-5/3}\,f_\eta(k\eta)$$

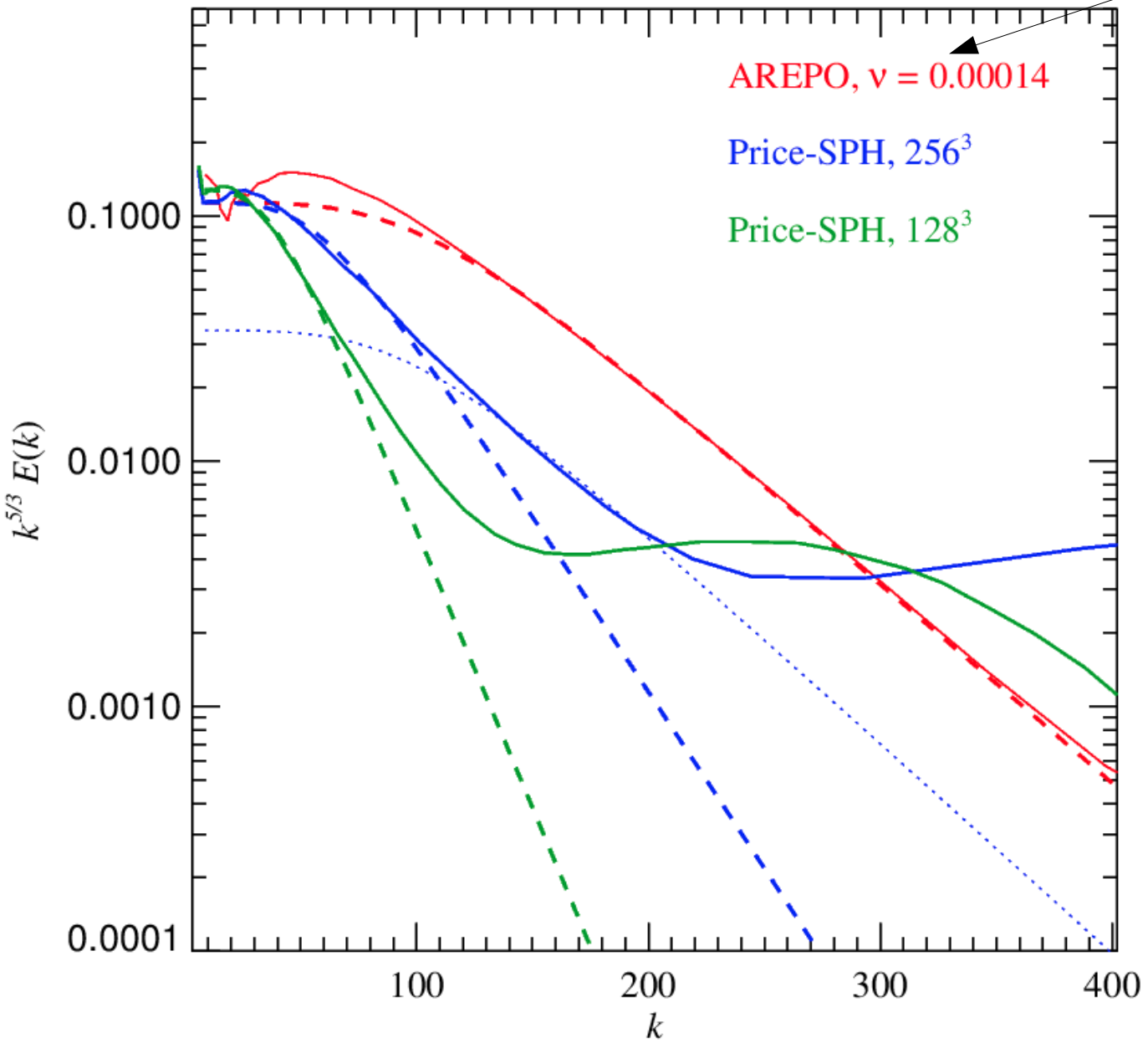
Experiments (and simulations) give a univers f_η function for

$$f_\eta(x) = \exp\left(-\beta[(x^4 + c^4)^{1/4} - c]\right)$$

$$\beta \sim 5.2 \qquad c \sim 0.4$$

The shape of the subsonic dissipation range is problematic in SPH

TURBULENCE POWER SPECTRUM IN THE DISSIPATIVE REGIME



Reynolds-Numbers

$$\mathrm{Re} \equiv \frac{L_0 V_0}{\nu}$$

Re = 2100

Re = 1000

Re = 540

The power spectrum of the dissipation range in SPH has the wrong shape!

The computational cost to reach a desired Reynolds number in subsonic turbulence grows more quickly in SPH than in a mesh code

REYNOLDS NUMBER AND COMPUTATIONAL COST

$$\mathcal{R}_{\rm e} \equiv \frac{VL}{\nu} \qquad\qquad \frac{\eta}{L_0} \sim {\rm Re}^{-\frac{3}{4}}$$

Computational cost: CPU ~ d^{-4}, where d = mean cell/particle spacing

Assume that we indeed could describe SPH by: $\nu \sim \frac{1}{10}\alpha v_{\rm sig} h$

CPU ~ Re^4

In the (moving) mesh code we however $\frac{\eta}{L_0} \sim d$

CPU ~ Re^3

Thank you for your attention!

More information: www.sppexa.de

This work is supported by the German Research Foundation (DFG), TODO include ANR/JST for bi/trilateral projects, as part of the priority programme 1648 *Software for Exascale Computing*.