High performance computing and numerical modeling

Volker Springel

Plan for my lectures

Lecture 1:	Collisional and collisionless N-body dynamics
Lecture 2:	Gravitational force calculation
Lecture 3:	Basic gas dynamics
Lecture 4:	Smoothed particle hydrodynamics
Lecture 5:	Eulerian hydrodynamics
Lecture 6:	Moving-mesh techniques
Lecture 7:	Towards high dynamic range
Lecture 8:	Parallelization techniques and current computing trends

Heidelberg Institute for Theoretical Studies
HITS
Universität Heidelberg

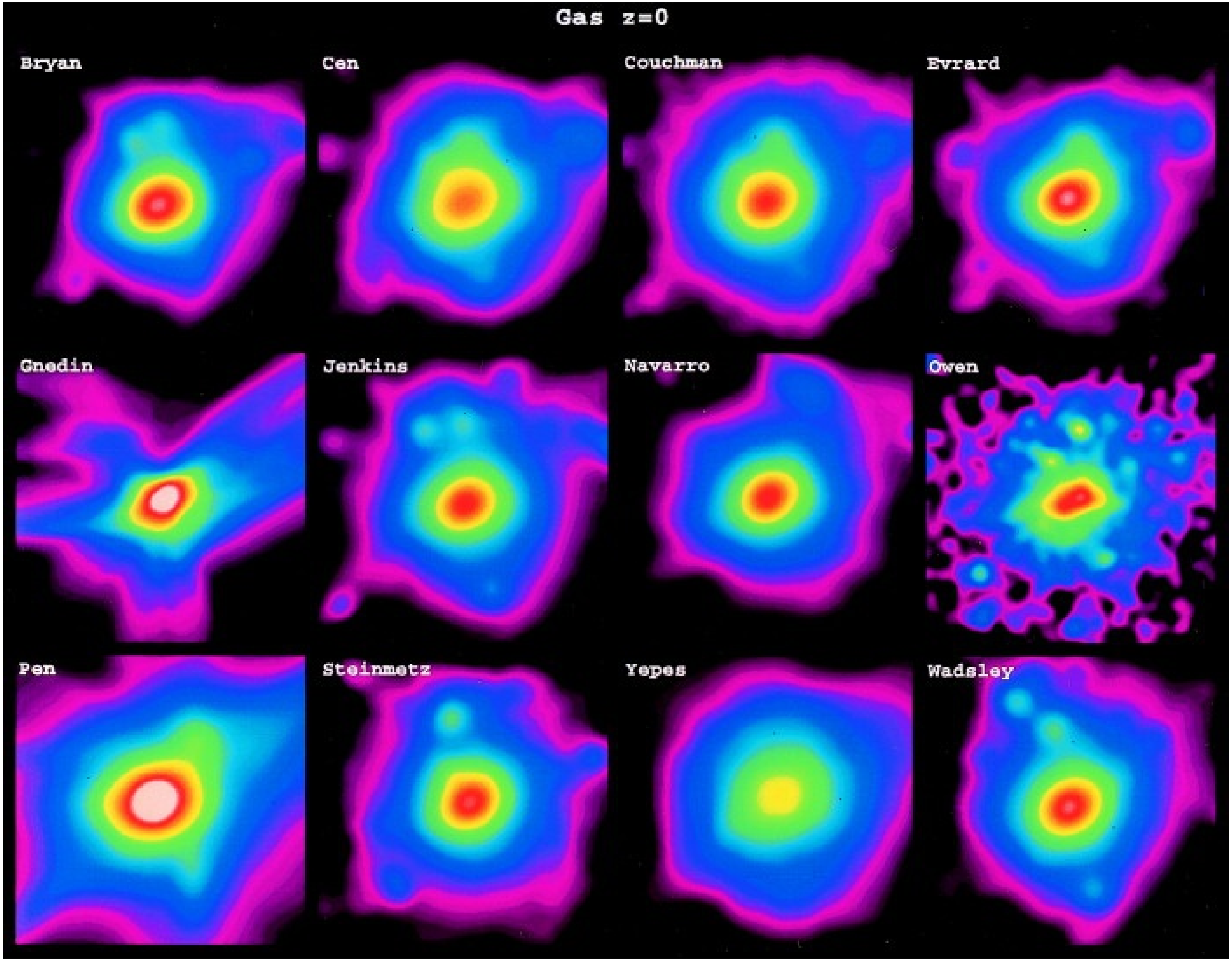
43rd Saas Fee Course
Villars-Sur-Ollon, March 2013

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

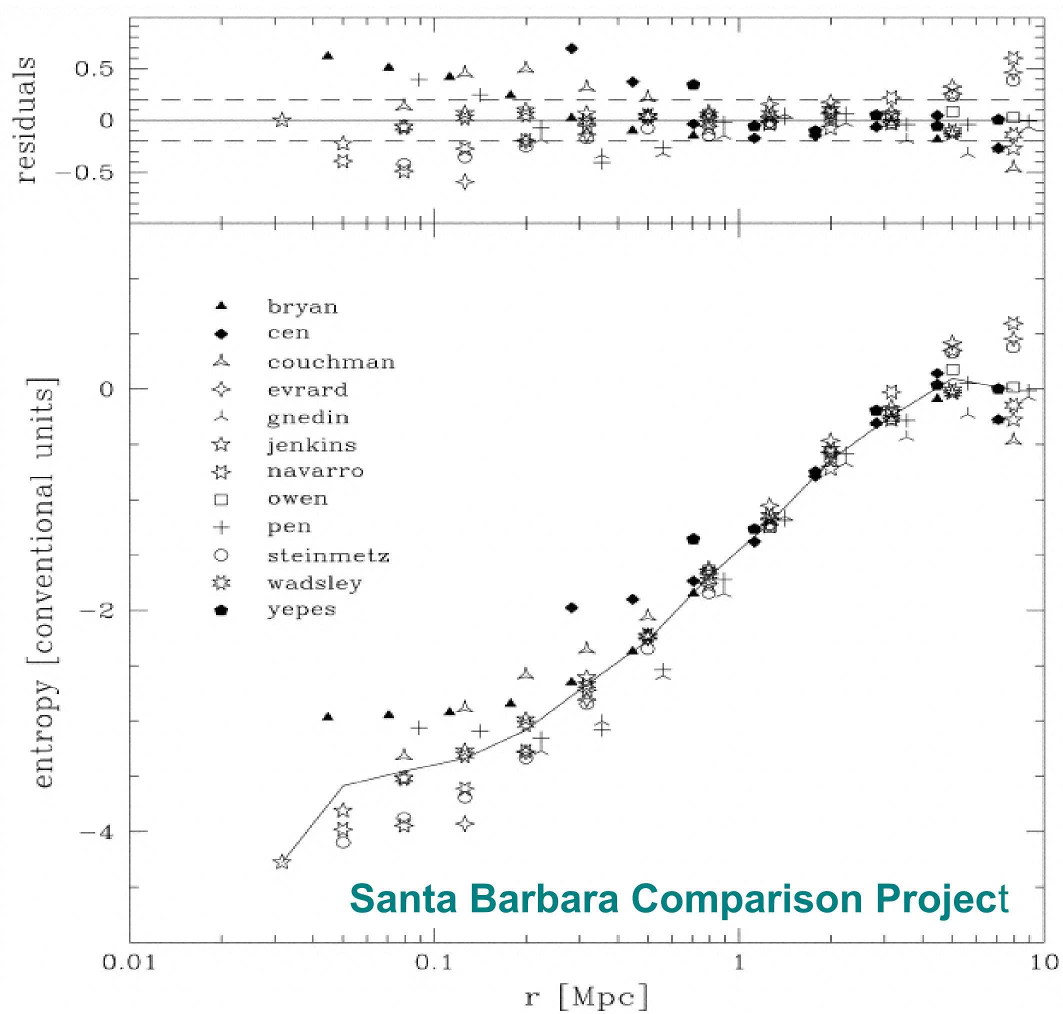
Frenk, White & 23 co-authors (1999)

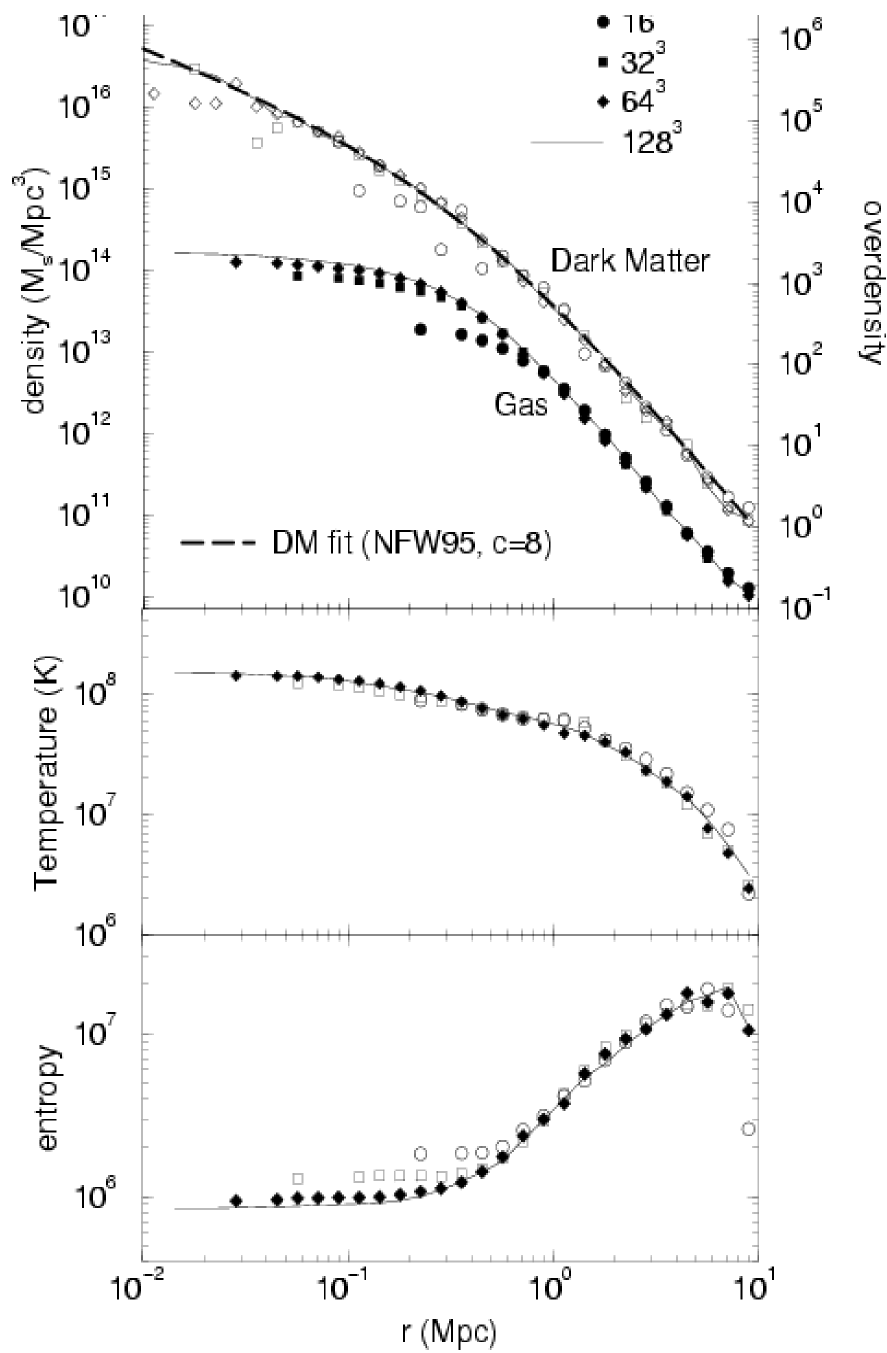


Mesh codes appear to produce higher entropy in the cores of clusters

RADIAL ENTROPY PROFILE

Bryan & Norman 1997

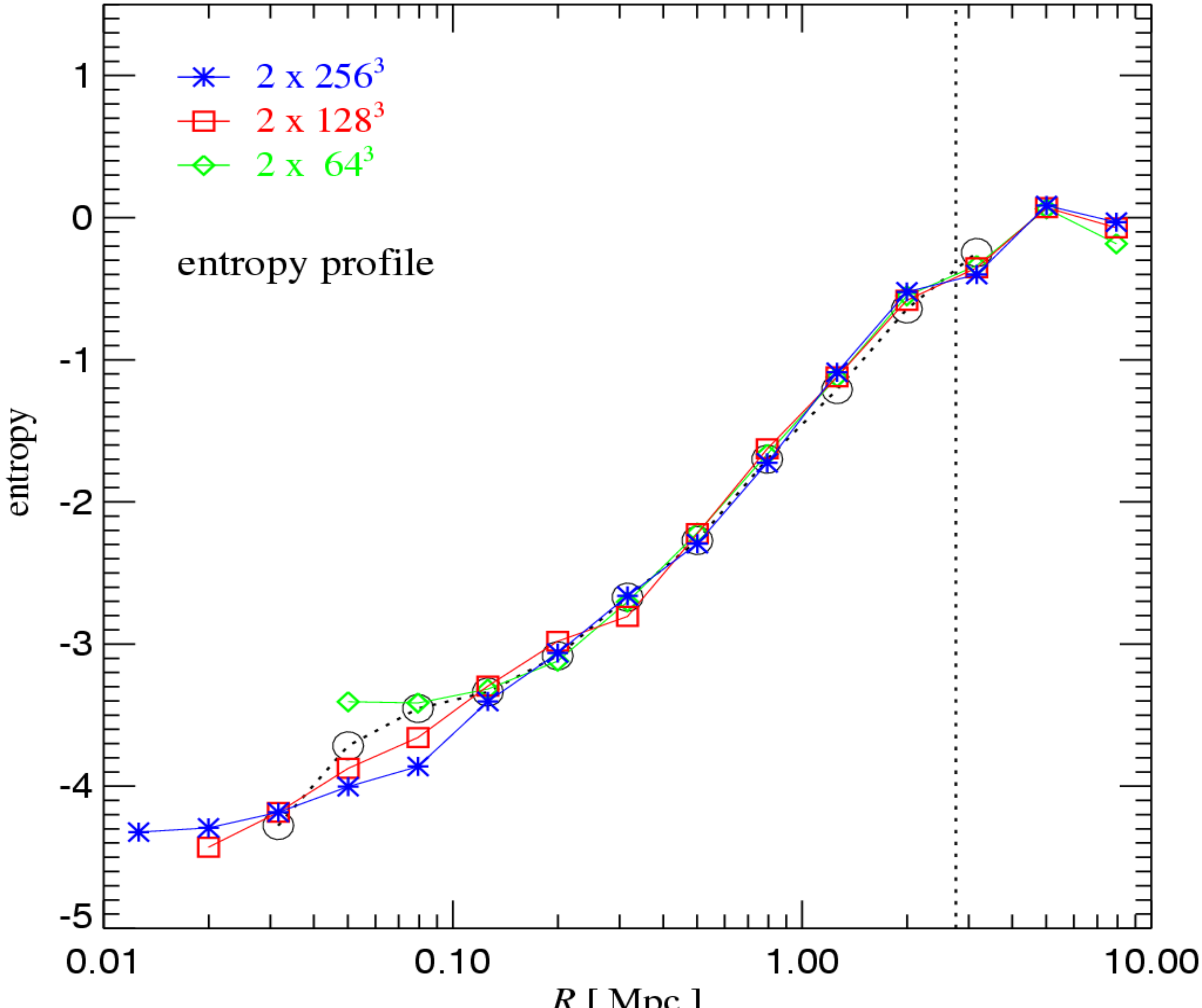




Ascasibar, Yepes, Müller & Gottlöber (2003):
Entropy formulation of SPH also gives somewhat higher core entropy

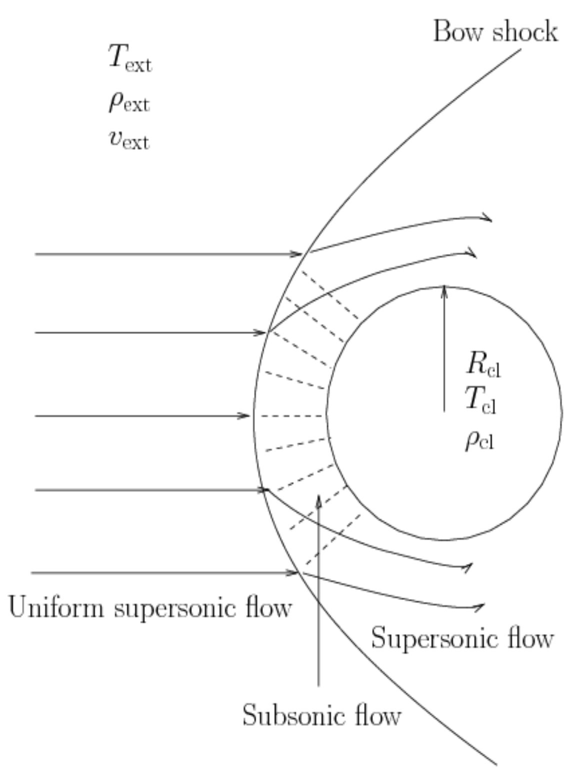
The entropy profile of the Santa Barbara cluster appears to converge well with SPH, yielding a lower level in the center than found with mesh codes

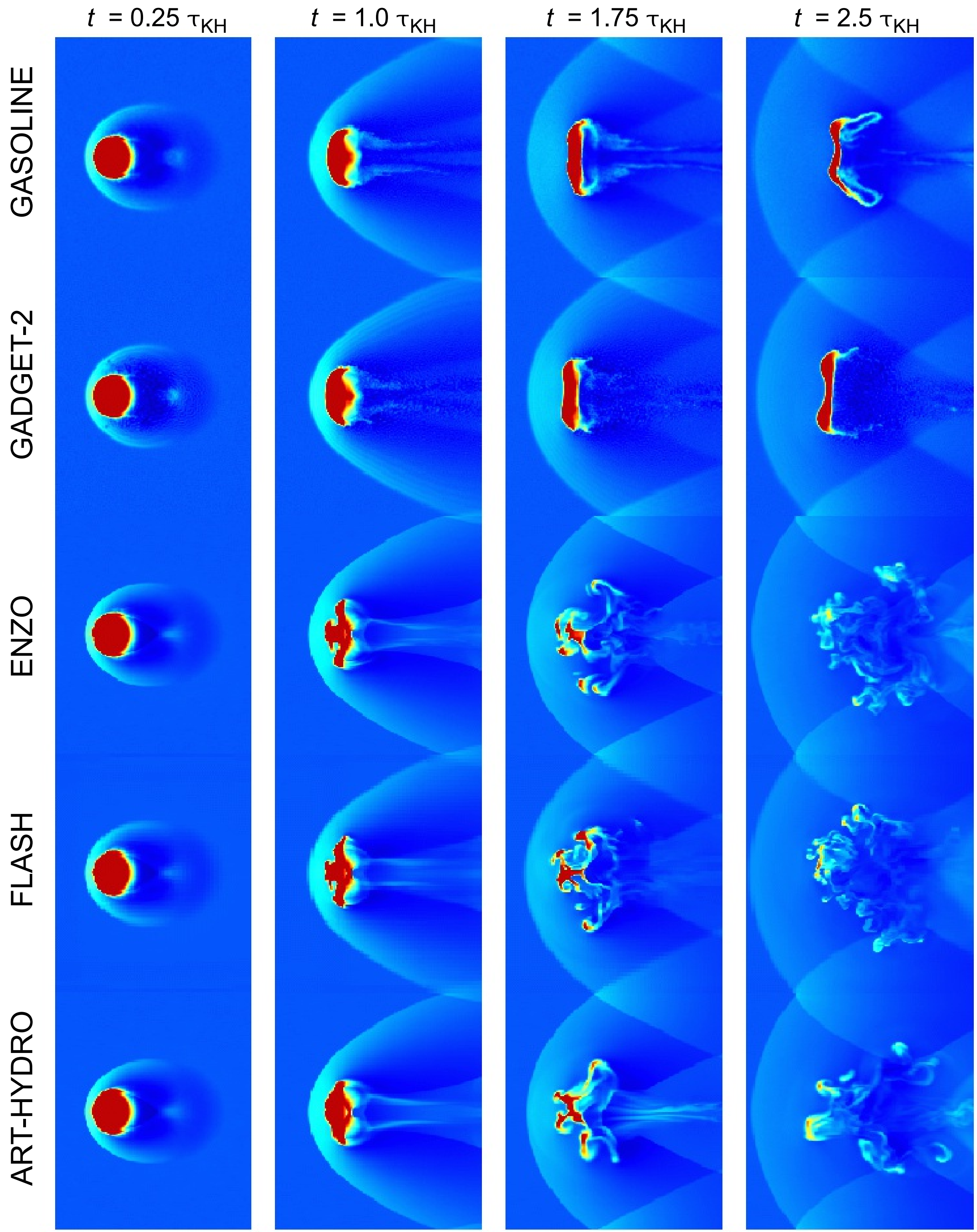
ENTROPY PROFILES OBTAINED WITH GADGET2 AT DIFFERENT RESOLUTION



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

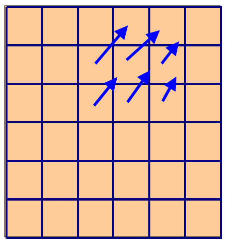


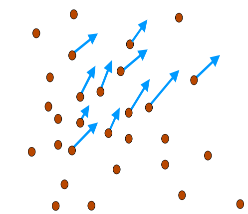


Agertz et al. (2007)

There are principal differences between SPH and Eulerian schemes

SOME FUNDAMENTAL DIFFERENCE BETWEENS SPH AND MESH-HYDRODYNAMICS





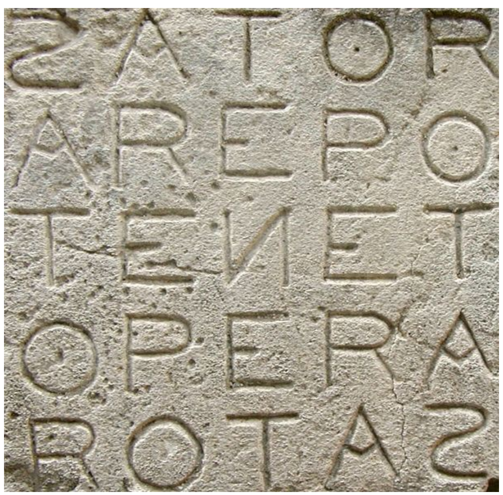
Eulerian	Lagrangian
sharp shocks, somewhat less sharp contact discontinuities (best schemes resolve fluid discontinuities it in one cell)	**shocks broadened over roughly 2-3 smoothing lengths** (post-shock properties are correct though)
mixing happens implicitly at the cell level (but advection adds 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)

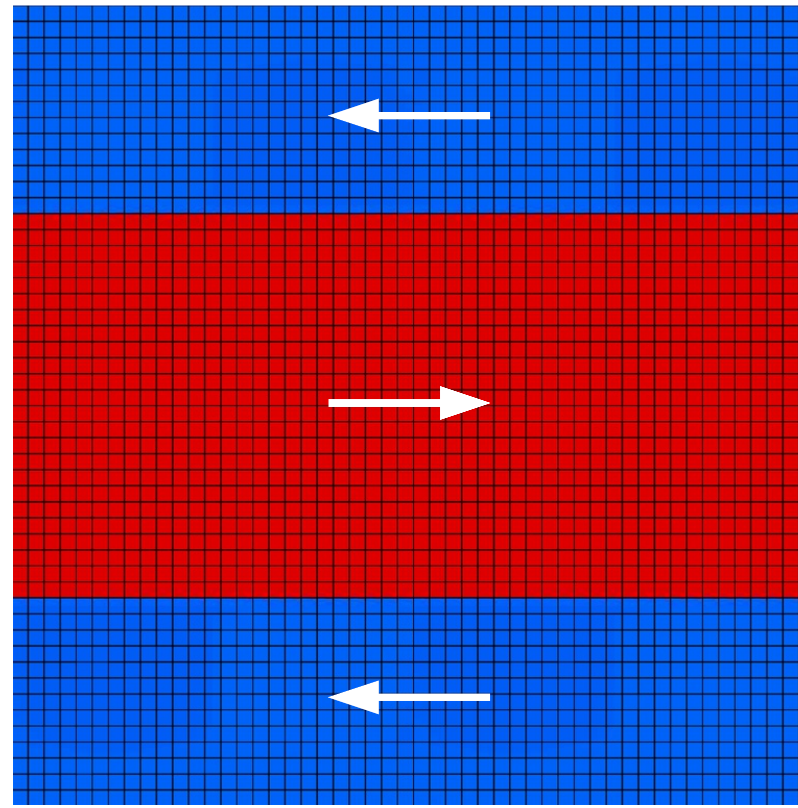


ρ = 1
v_x = -0.5
P = 2.5

ρ = 2
v_x = 0.5
P = 2.5

ρ = 1
v_x = -0.5
P = 2.5

periodic boundaries

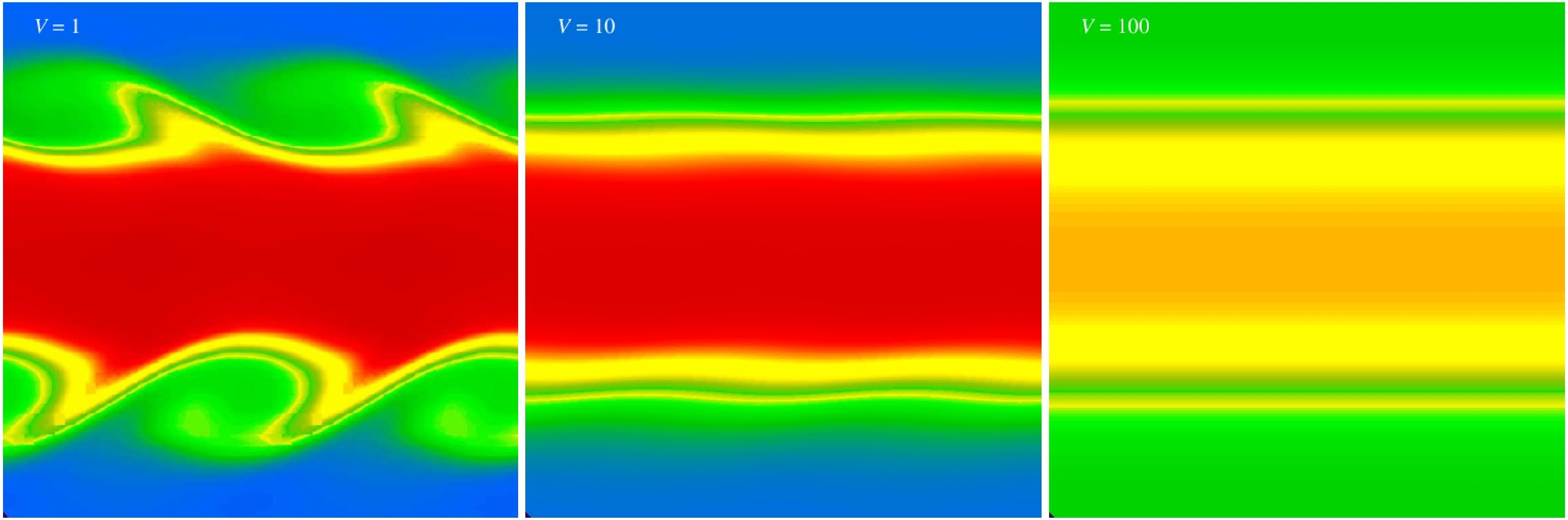


50x50 resolution

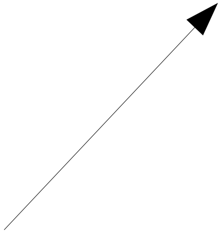
When the mesh is fixed, the results may change if a bulk velocity is imposed

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.

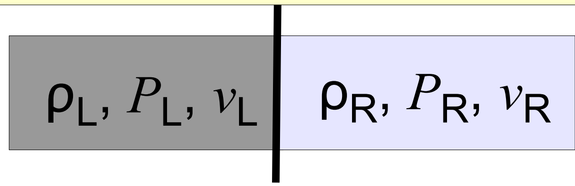
With enough cells, the truncation error can always be reduced, so that for properly resolved initial conditions, effective Galilean invariance is reached.

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.

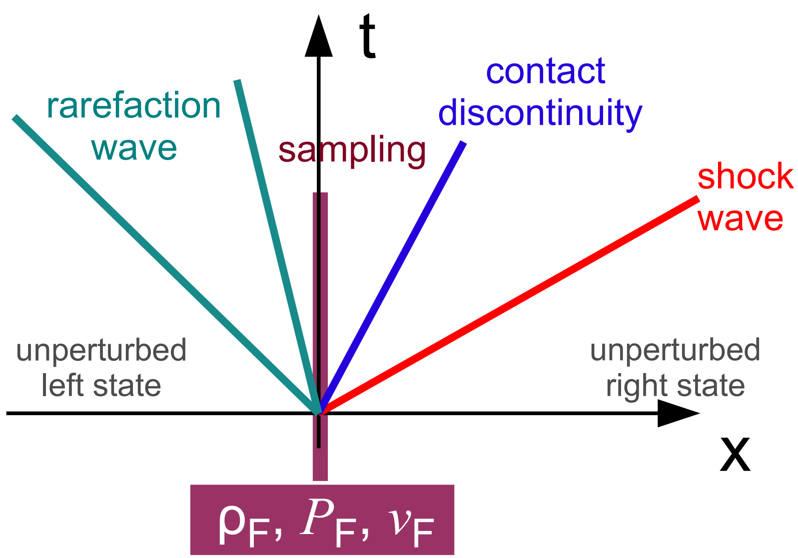
The Riemann problem as basis for high-accuracy Godunov schemes

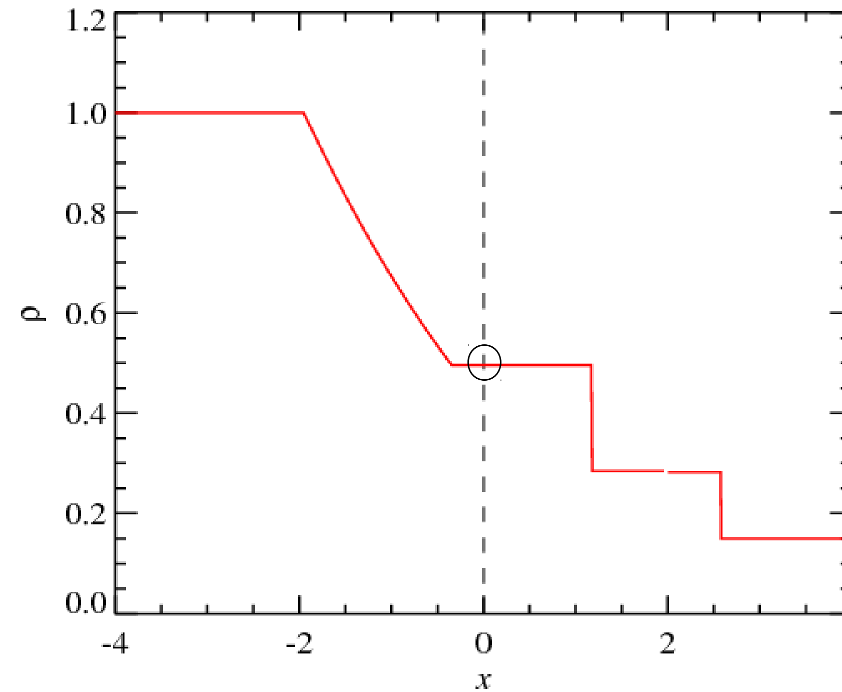
CALCULATION OF THE GODUNOV FLUX

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



Calculate the self-similar time evolution (Riemann problem)

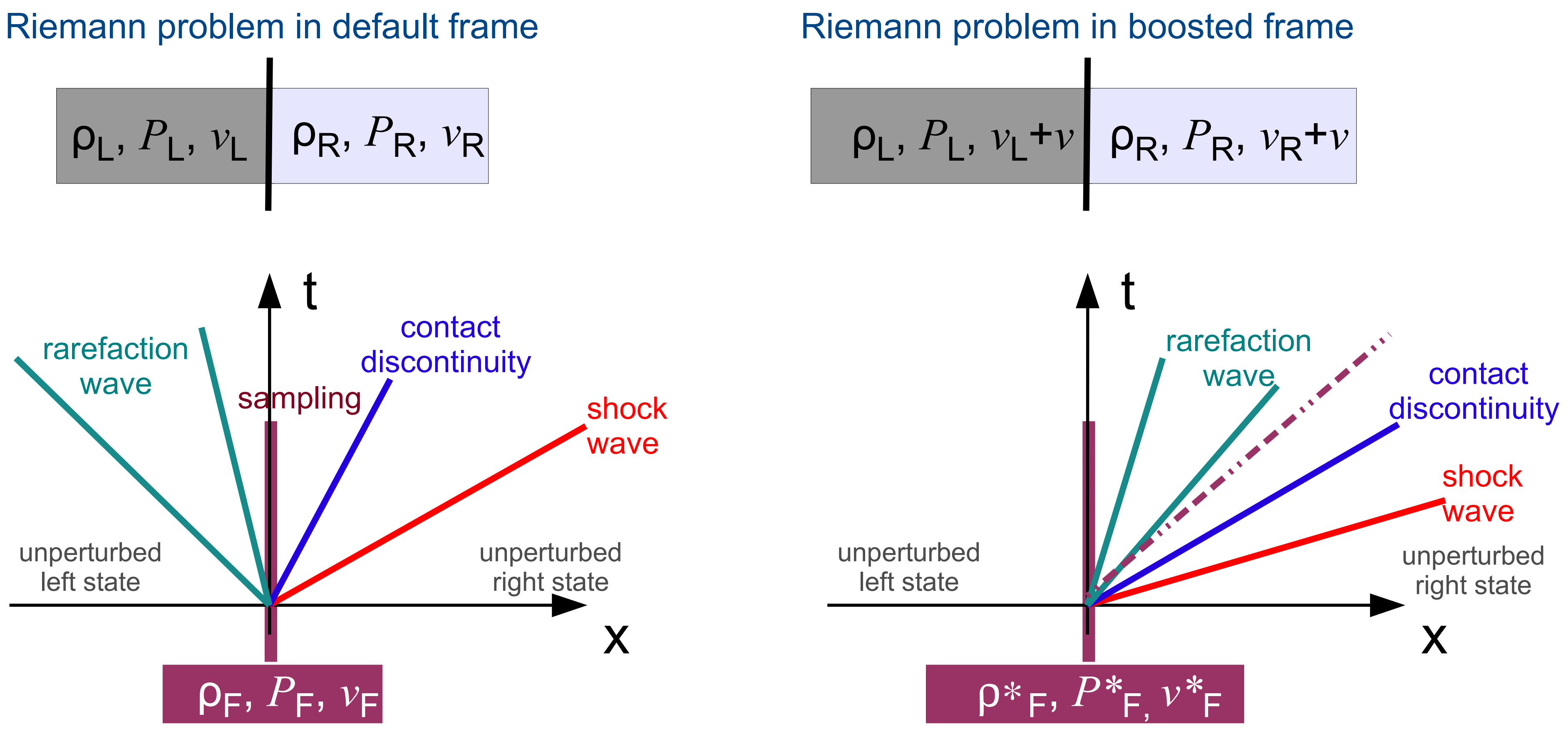




Sample the solution along x/ t=0, which yields the Godunov flux

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

THE GODUNOV FLUX IN DIFFERENT REFERENCE FRAMES



expected mass flux in boosted frame:

$\rho_F\,(v_F + v)$

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

→ **Numerical scheme not manifestly Galilean invariant**

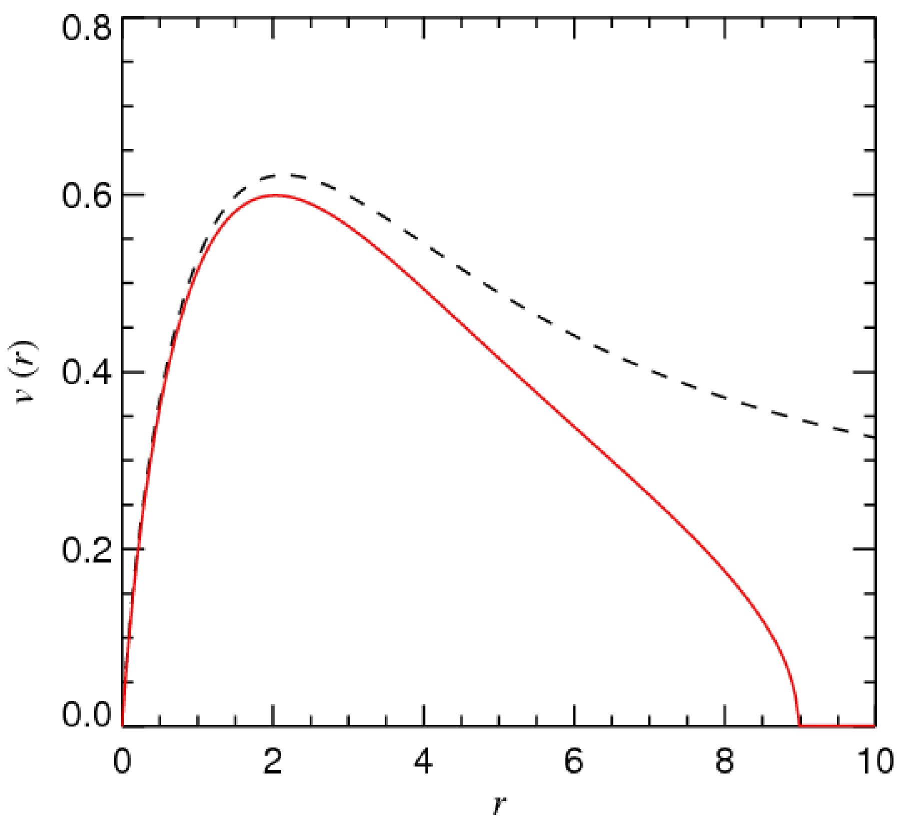
How well does this work?

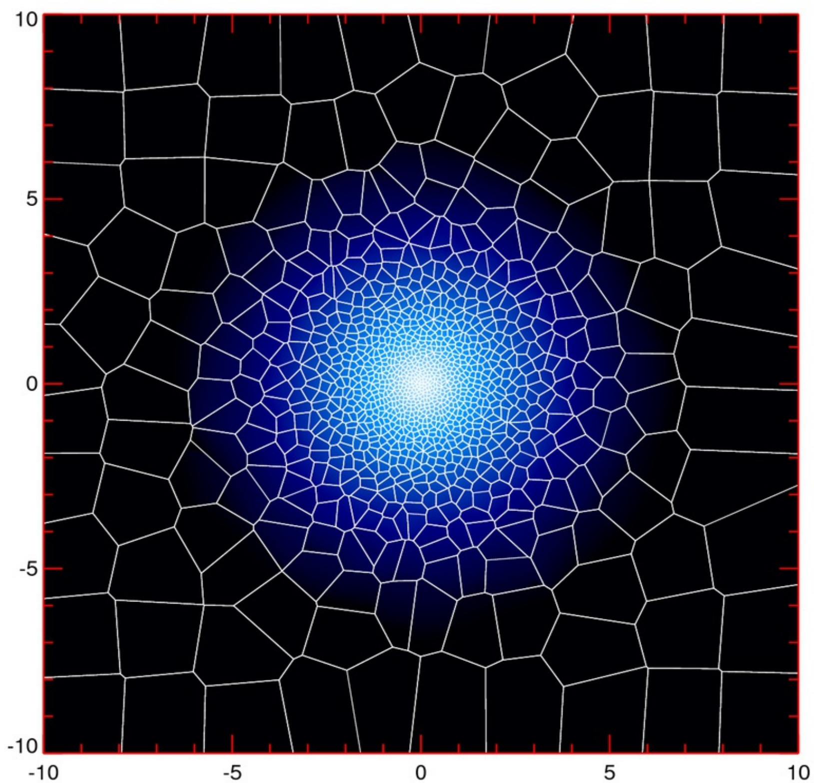
A differentially rotating gaseous disk with strong shear can be simulated well with the moving mesh code

MODEL FOR A CENTRIFUGALLY SUPPORTED, THIN DISK

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

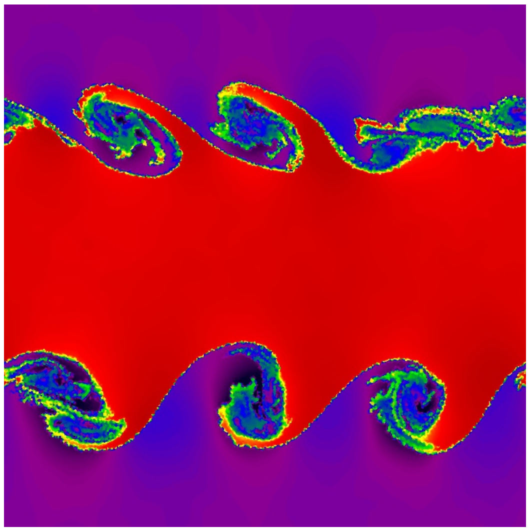
$$v_c^2(r) \equiv r\frac{\partial \Phi}{\partial r} = 2\frac{Gm}{h} y^2 \left[I_0(y)K_0(y) - I_1(y)K_1(y)\right]$$



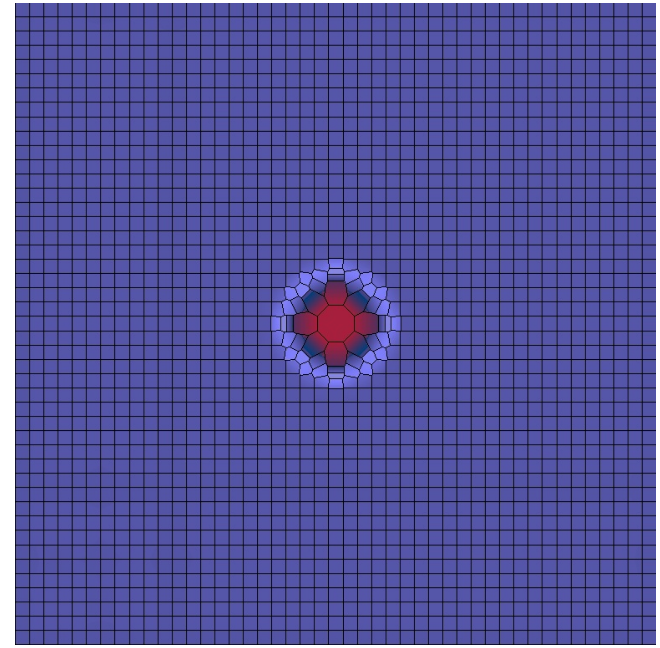


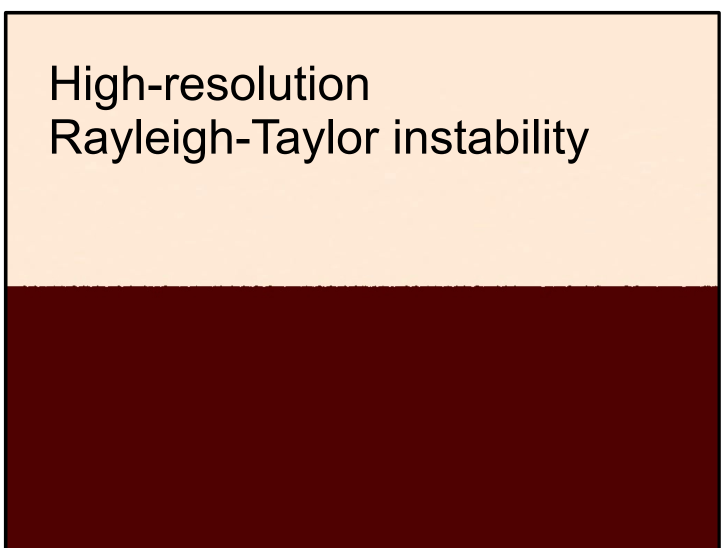
Different examples of test problems with the moving-mesh code

High-resolution
Kelvin-Helmholtz instability



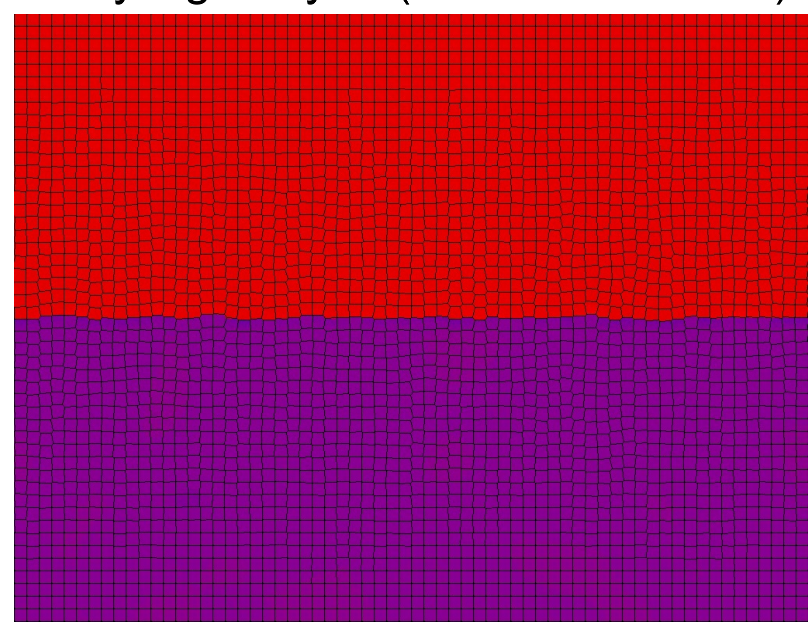
Sedov-Taylor Exposion





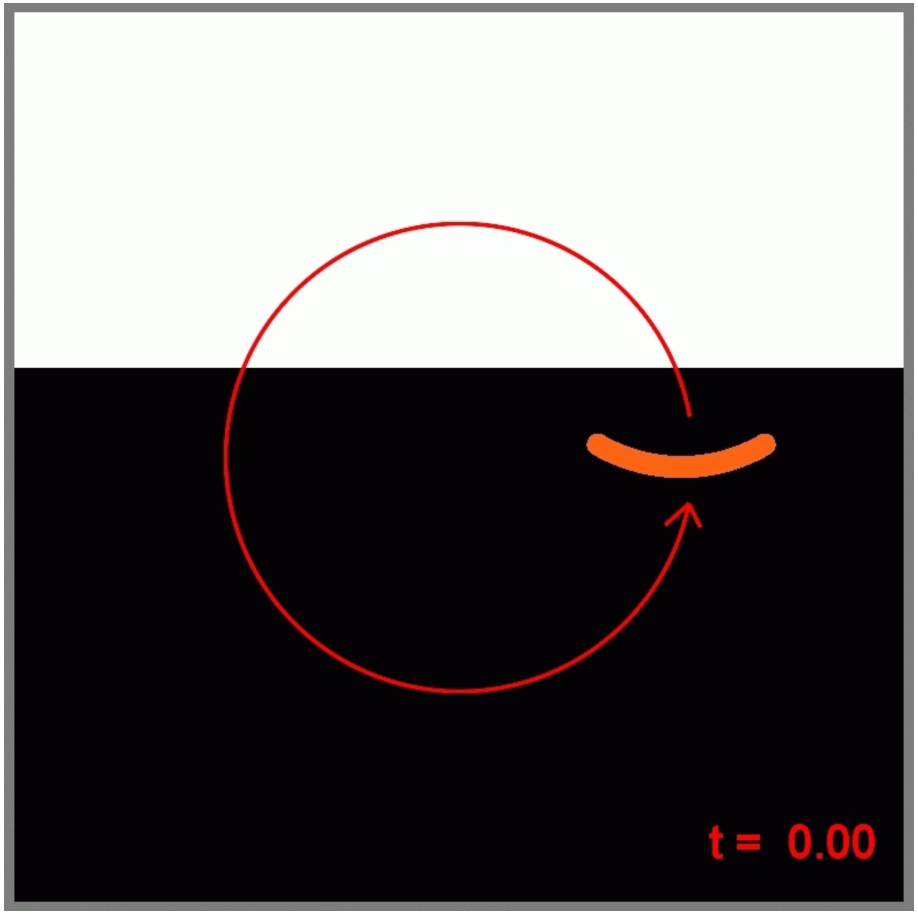
High-resolution
Rayleigh-Taylor instability

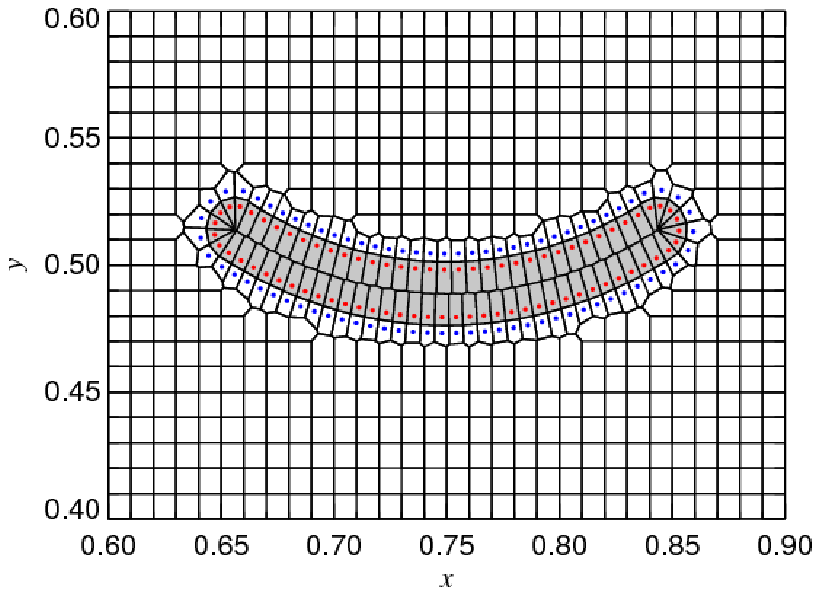
Rayleigh-Taylor (with visible mesh)

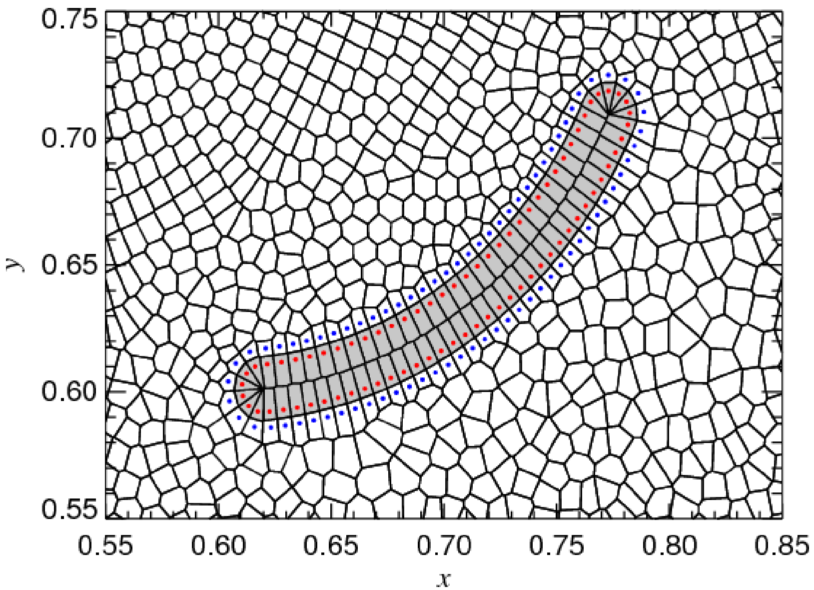


The moving-mesh approach can also be used to realize arbitrarily shaped, moving boundaries

STIRRING A COFFEE MUG

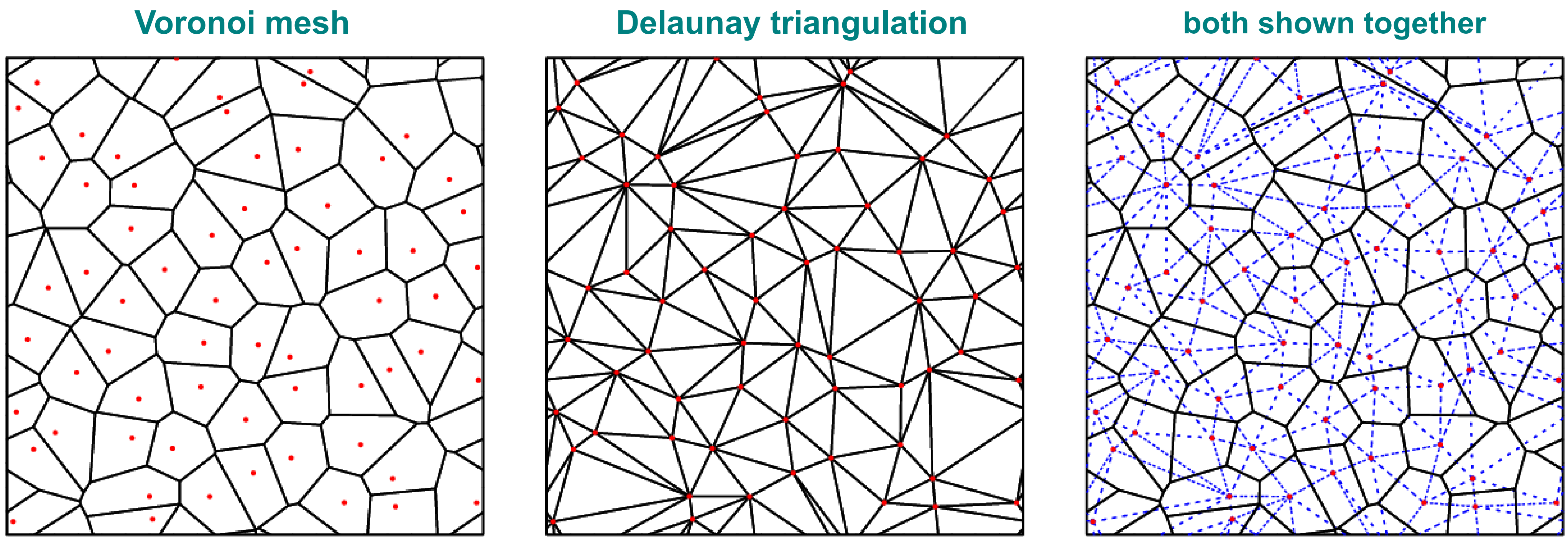






Voronoi and Delaunay tessellations provide unique partitions of space based on a given sample of mesh-generating points

BASIC PROPERTIES OF VORONOI AND DELAUNAY MESHES



- 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

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

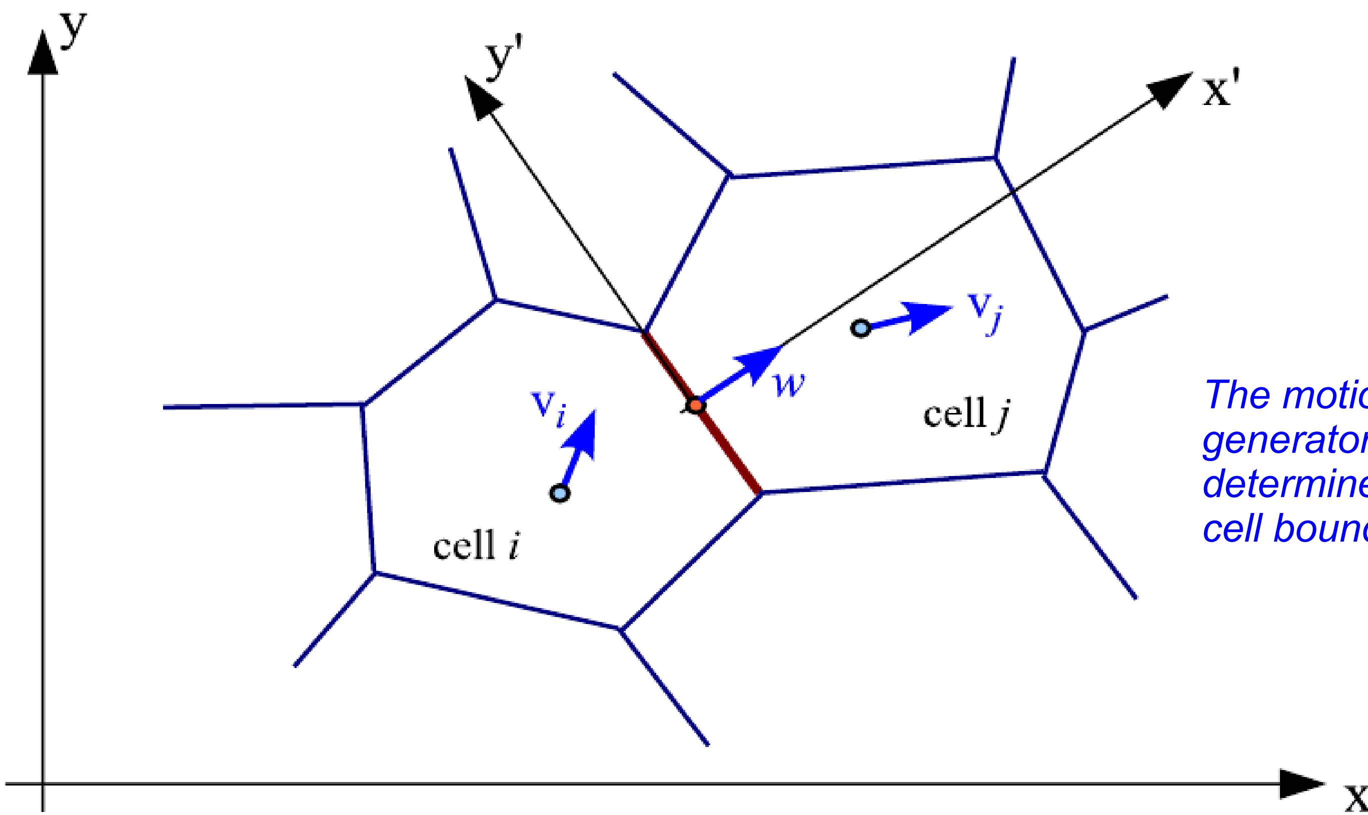
Evolution equation

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

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**

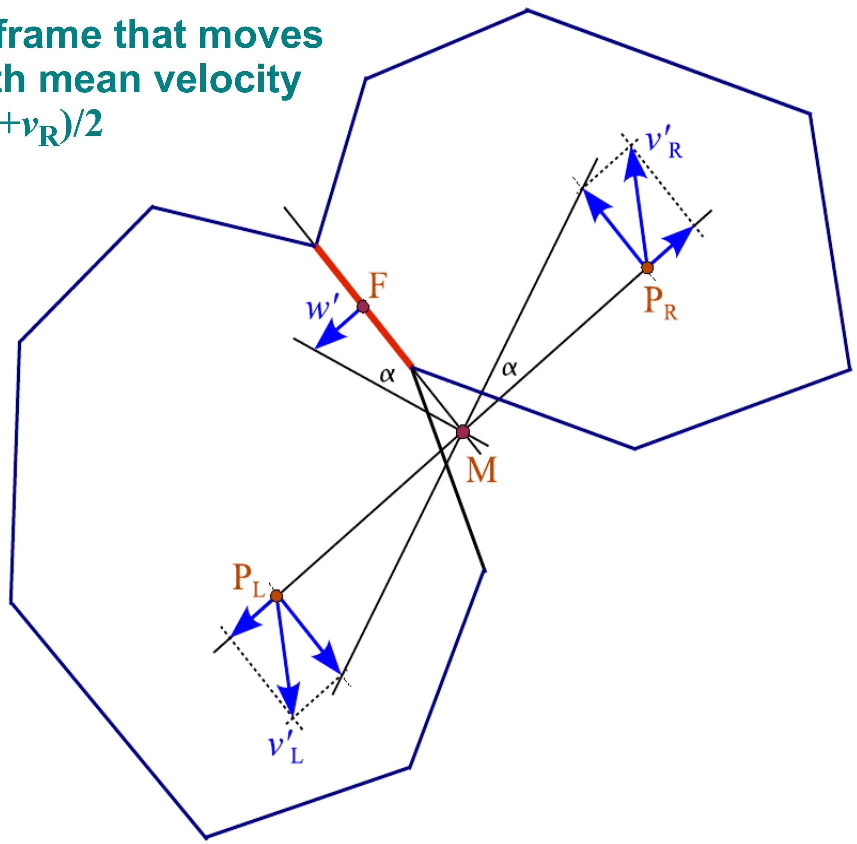
$$\begin{pmatrix} \rho_L \\ \mathbf{v}_L \\ P_L \end{pmatrix} \quad \begin{pmatrix} \rho_R \\ \mathbf{v}_R \\ P_R \end{pmatrix} \longrightarrow \text{Riemann solver (in frame of cell face)} \longrightarrow \begin{pmatrix} \rho \\ \mathbf{v} \\ P \end{pmatrix} \longrightarrow \mathbf{F}(\mathbf{U})$$

Riemann solver
(in frame of cell face)

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 $(v_L+v_R)/2$



$$\boldsymbol{w}' = \frac{(\boldsymbol{v}_L - \boldsymbol{v}_R)\cdot[\boldsymbol{f} - (\boldsymbol{x}_R + \boldsymbol{x}_L)/2]}{|\boldsymbol{x}_R - \boldsymbol{x}_L|}\,\frac{(\boldsymbol{x}_R - \boldsymbol{x}_L)}{|\boldsymbol{x}_R - \boldsymbol{x}_L|}$$

$$\boldsymbol{w} = \frac{\boldsymbol{v}_R + \boldsymbol{v}_L}{2} + \boldsymbol{w}'$$

rate of change of volume of a cell

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

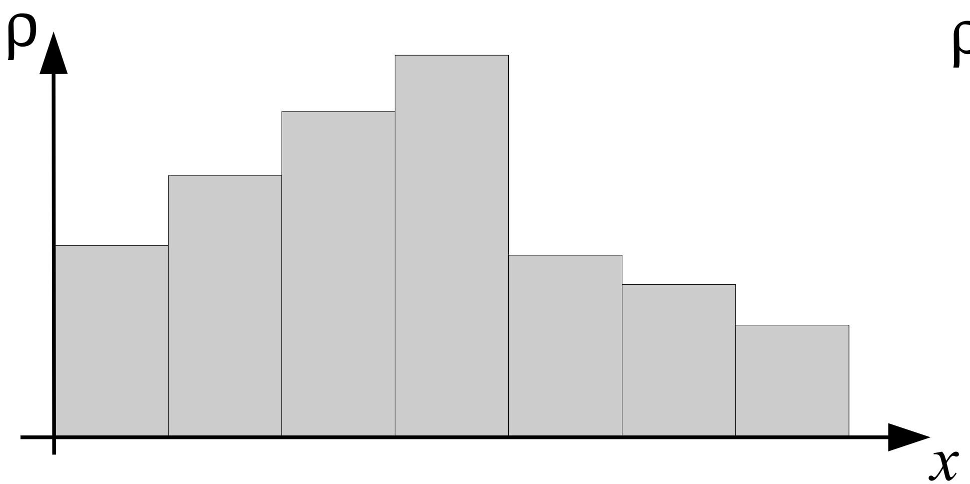
$$\boldsymbol{r}_{ij} = \boldsymbol{x}_i - \boldsymbol{x}_j$$

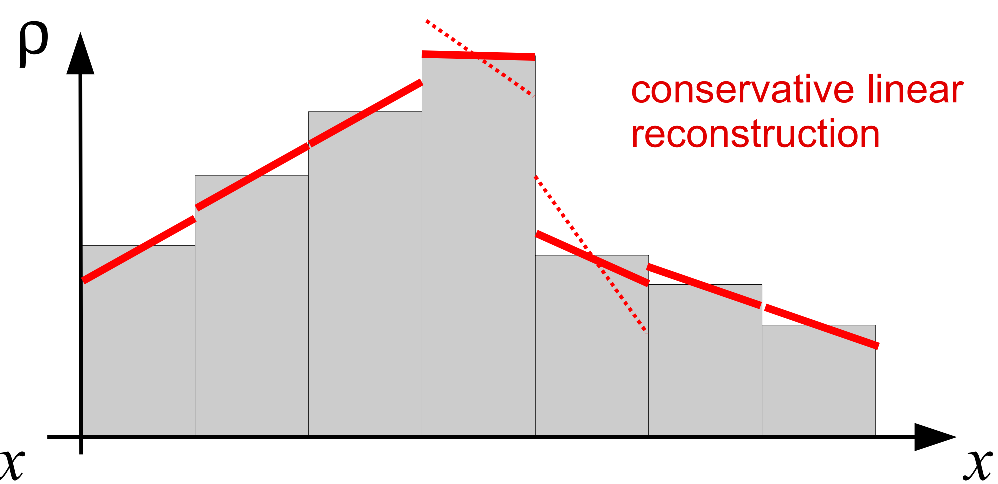
$$\boldsymbol{c}_{ij} = \boldsymbol{f}_{ij} - (\boldsymbol{x}_i + \boldsymbol{x}_j)/2$$

(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:

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

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

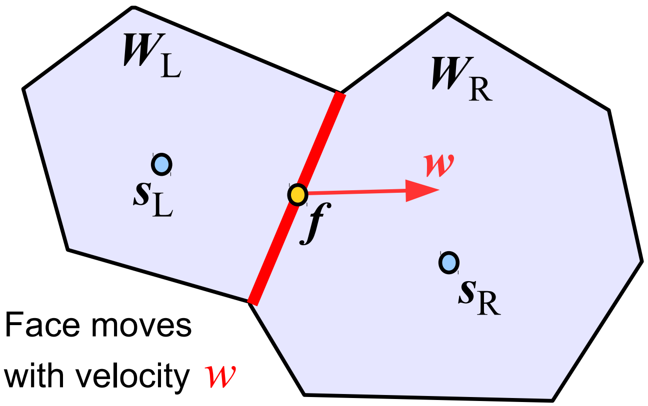
$$\psi_{ij} = \begin{cases} (\phi_i^{\max} - \phi_i)/\Delta\phi_{ij} & \text{for} \quad \Delta\phi_{ij} > 0 \\ (\phi_i^{\min} - \phi_i)/\Delta\phi_{ij} & \text{for} \quad \Delta\phi_{ij} < 0 \\ 1 & \text{for} \quad \Delta\phi_{ij} = 0 \end{cases}$$

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

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

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

A MUSCL-LIKE SCHEME



Face moves with velocity w

And finally...

Update the conserved variables of each cell:

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

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

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

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

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

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

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

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

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

Solve the Riemann problem

$$W = R_{\text{iemann}}(W'''_L, W'''_R)$$

Transform the solution back to the calculational frame

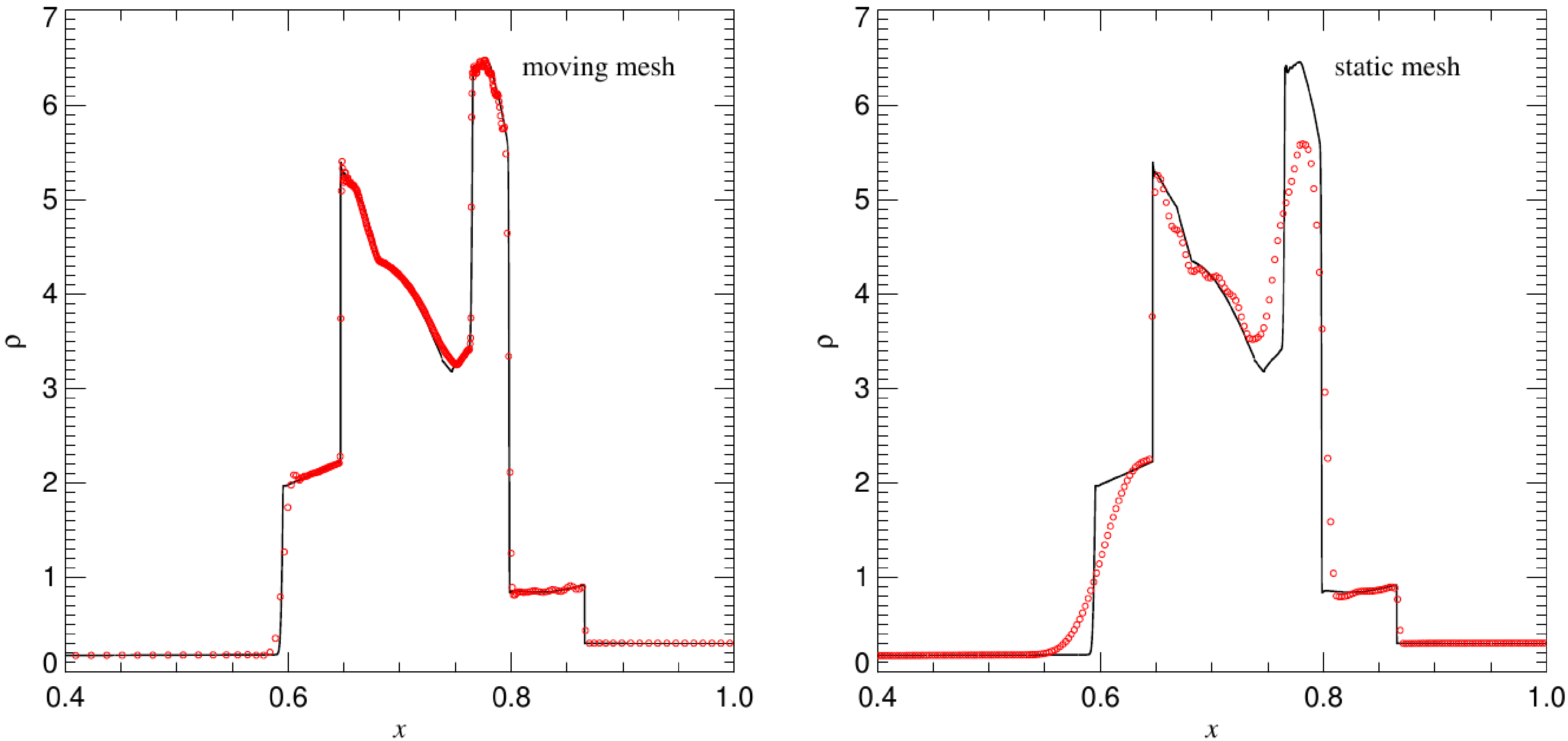
$$W_{\text{lab}} = \begin{pmatrix} \rho \\ v_{\text{lab}} \\ P \end{pmatrix} = \Lambda^{-1} W + \begin{pmatrix} 0 \\ w \\ 0 \end{pmatrix}$$

Calculate the net flux in the calculational frame

$$\hat{F} = F(U) - U w^T = \begin{pmatrix} \rho(v_{\text{lab}} - w) \\ \rho v_{\text{lab}}(v_{\text{lab}} - w)^T + P \\ \rho e_{\text{lab}}(v_{\text{lab}} - w) + P v_{\text{lab}} \end{pmatrix}$$

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

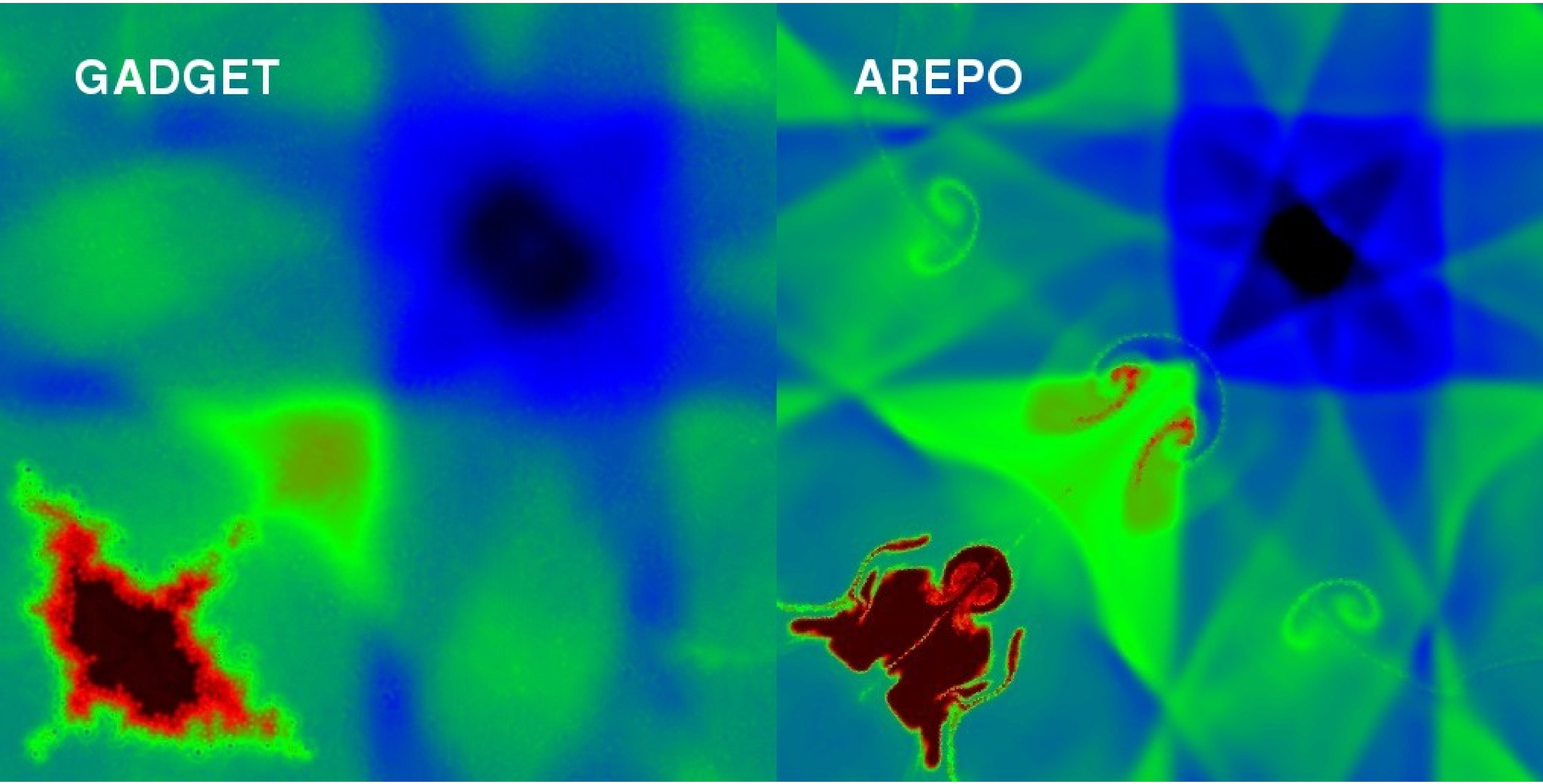
WOODWARD & COLELLA'S INTERACTING DOUBLE BLAST PROBLEM



Interacting shock waves reveal significant differences in vorticity production

TWO-DIMENSIONAL IMPLOSION PROBLEM

Sijacki et al. (2011)



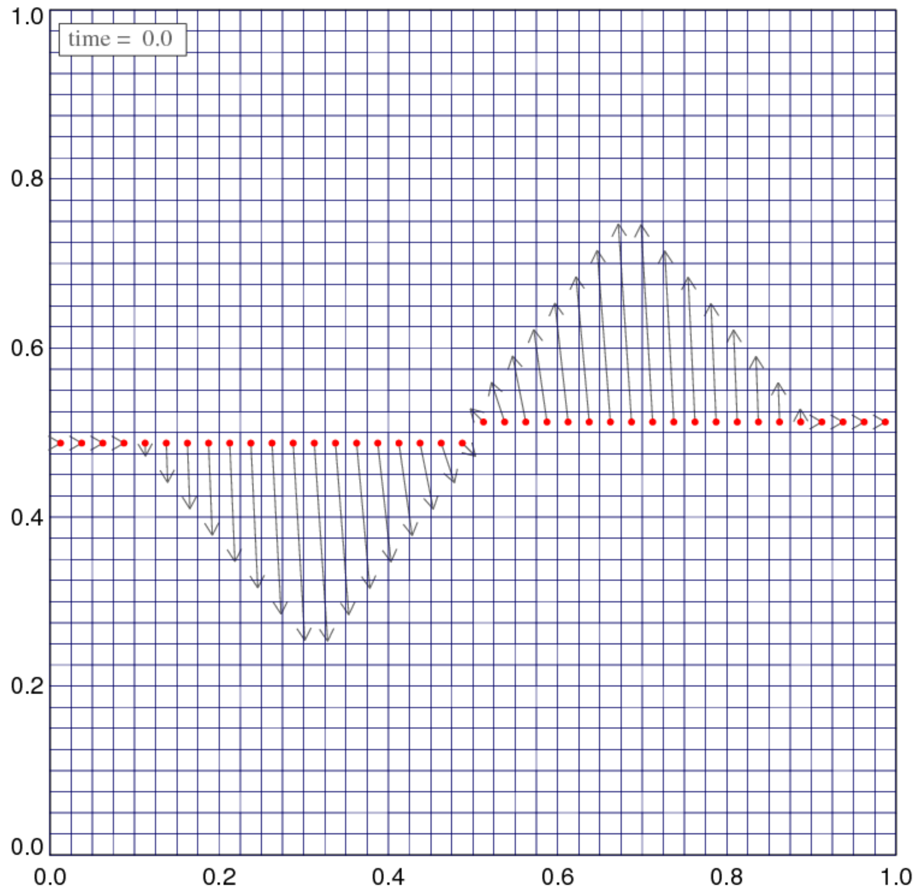
The Gresho vortex test in two dimensions

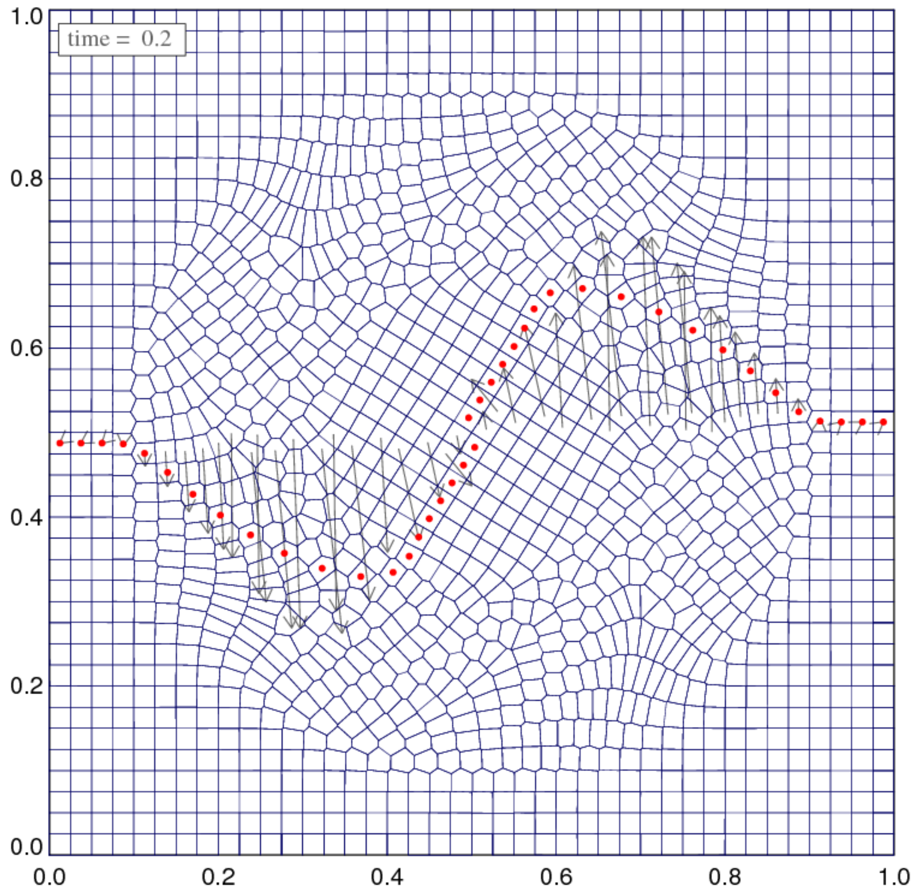
EVOLUTION OF A STATIONARY VORTEX FLOW

Initial conditions:

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

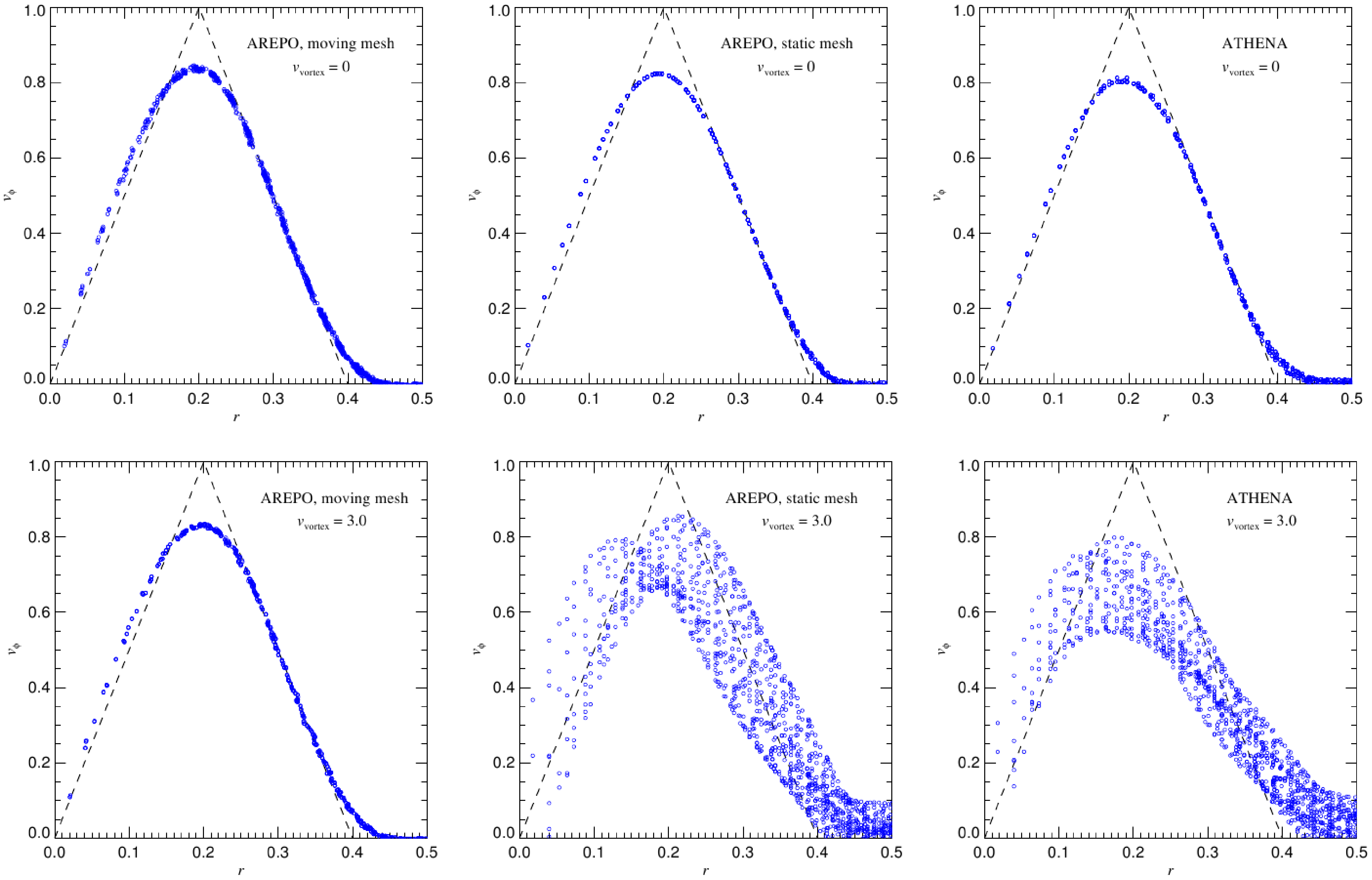
$$P(r) = \begin{cases} 5 + 25/2r^2 & \text{for} \quad 0 \leq r < 0.2 \\ 9 + 25/2r^2 - \\ \qquad 20r + 4\ln(r/0.2) & \text{for} \quad 0.2 \leq r < 0.4 \\ 3 + 4\ln 2 & \text{for} \quad r \geq 0.4 \end{cases}$$





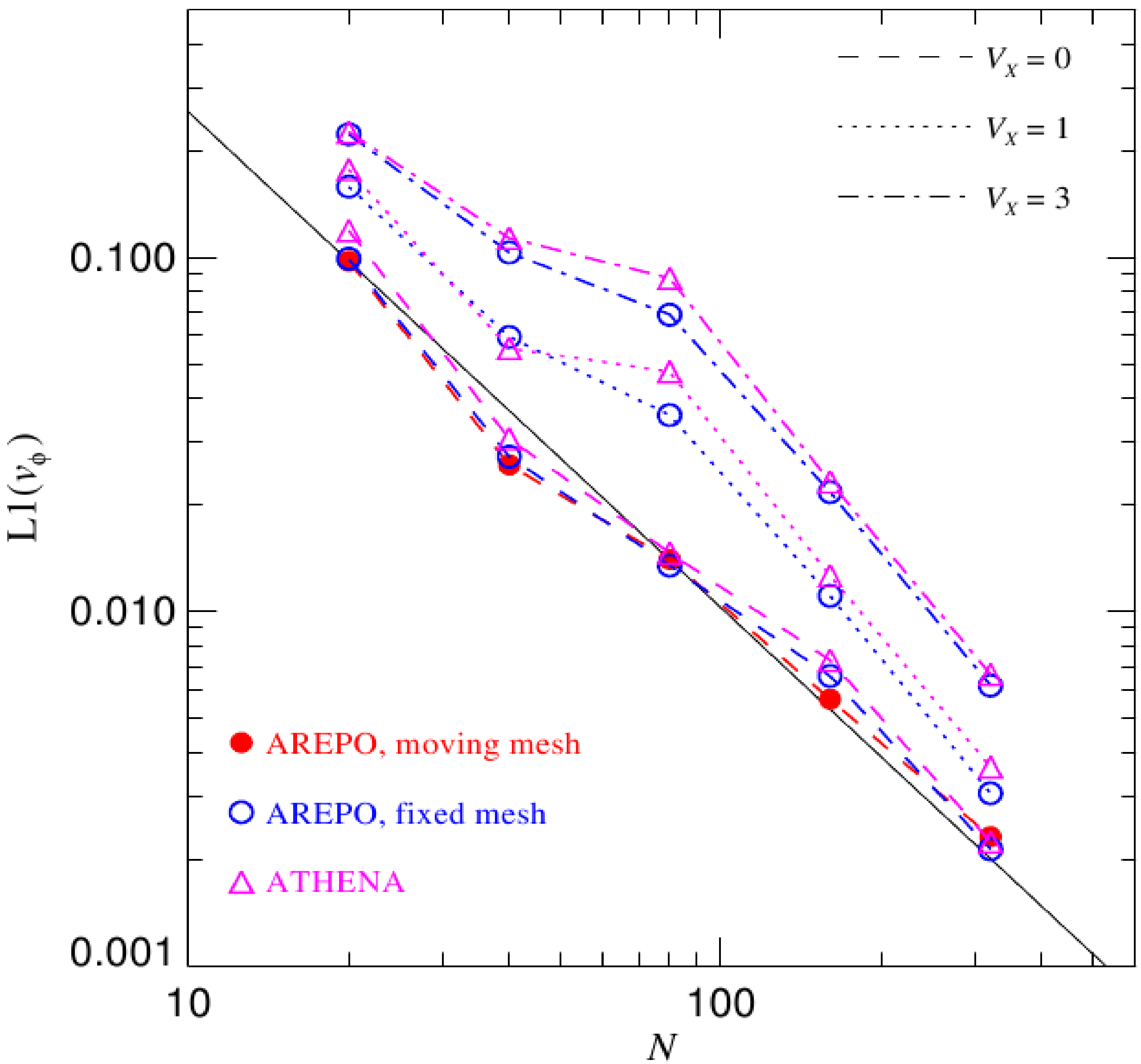
The Gresho vortex test in two dimensions

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:

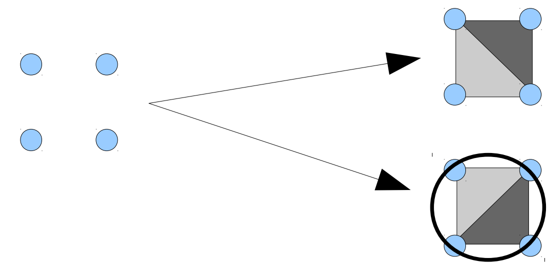
(1) **Point location:** Find triangle/tetrahedron that contains point

(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

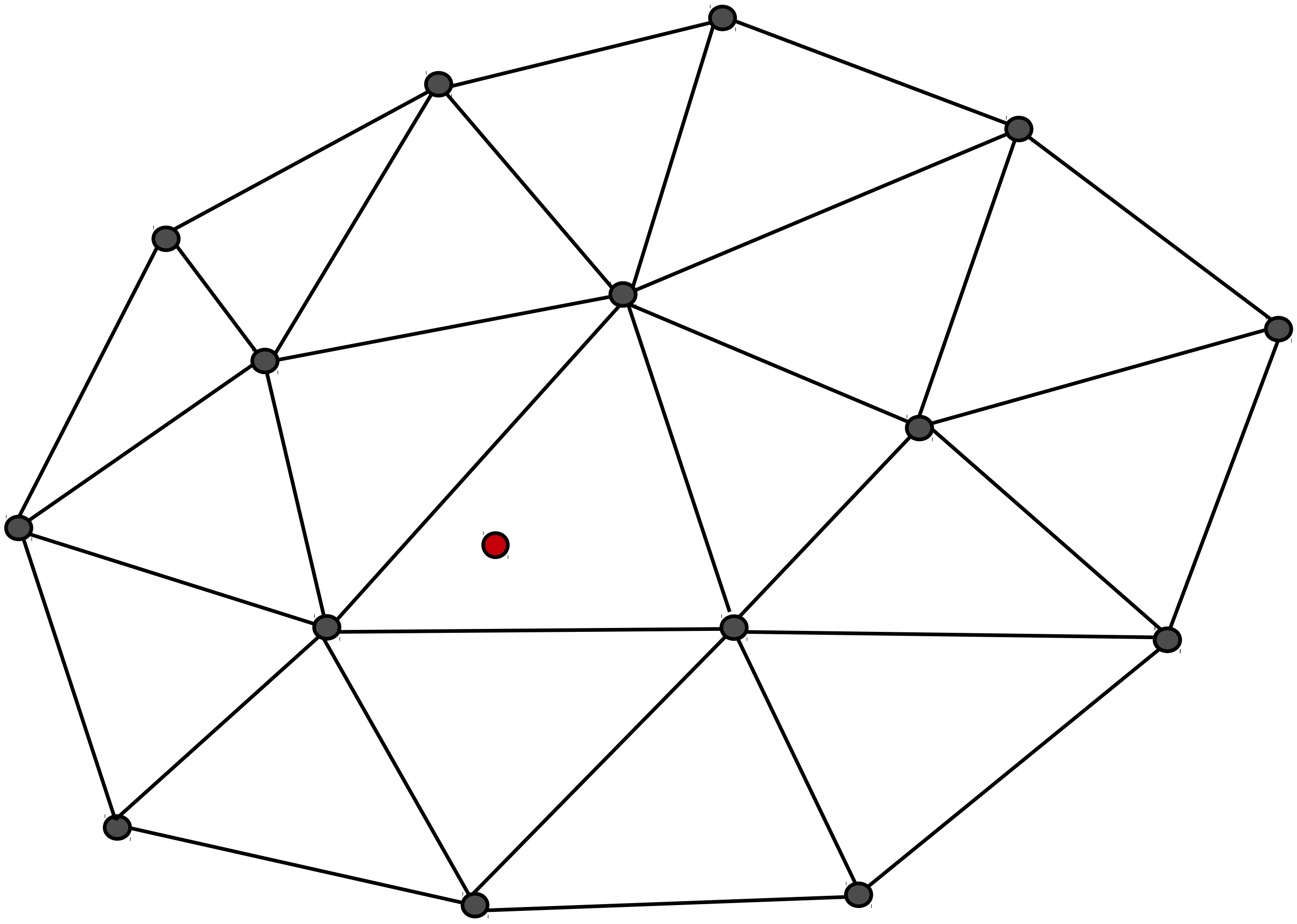


4 cocircular points !

How can we consistently break ties?

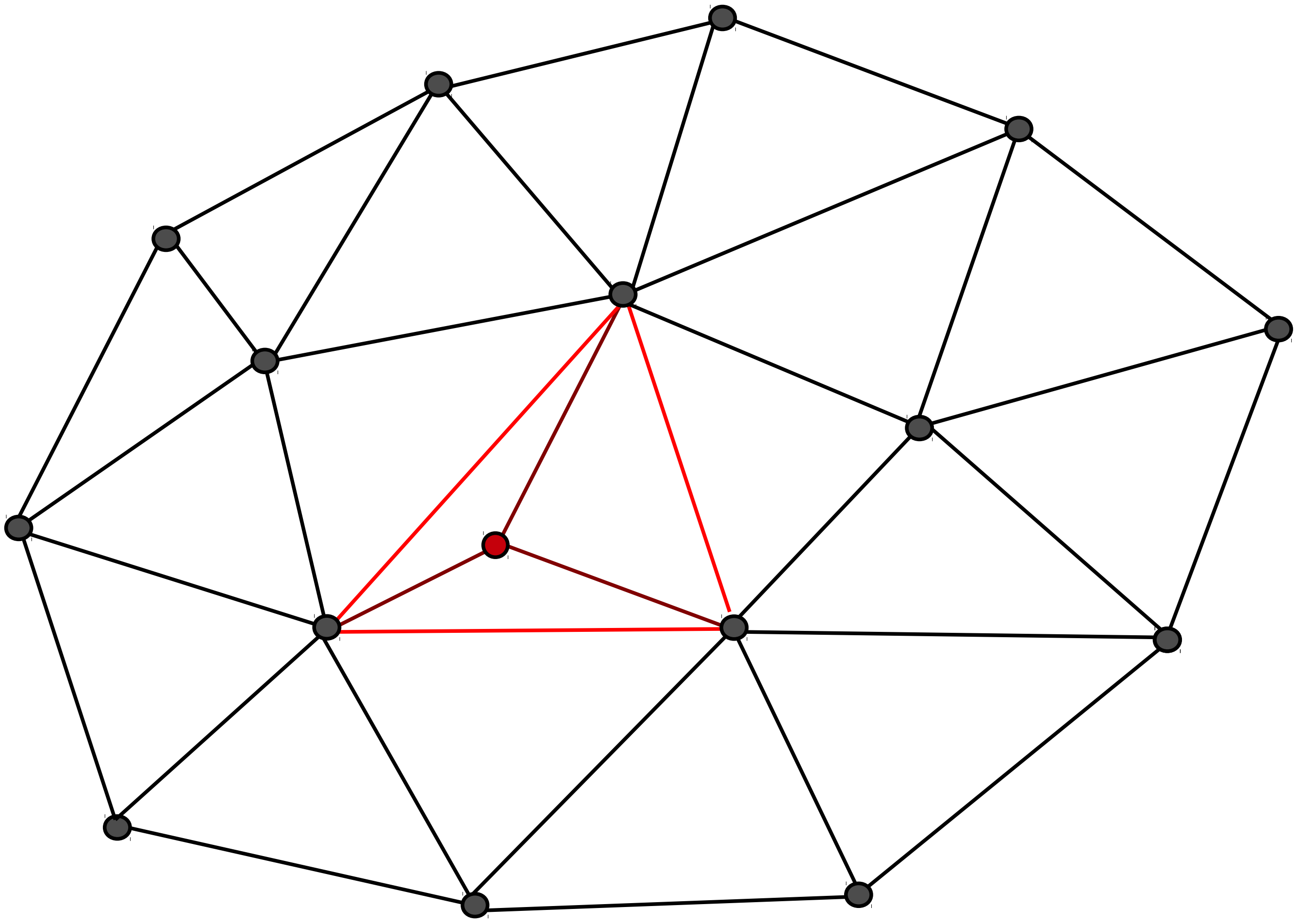
Adding a point by sequential insertion

1. Step: Locate the triangle that contains the point



Adding a point by sequential insertion

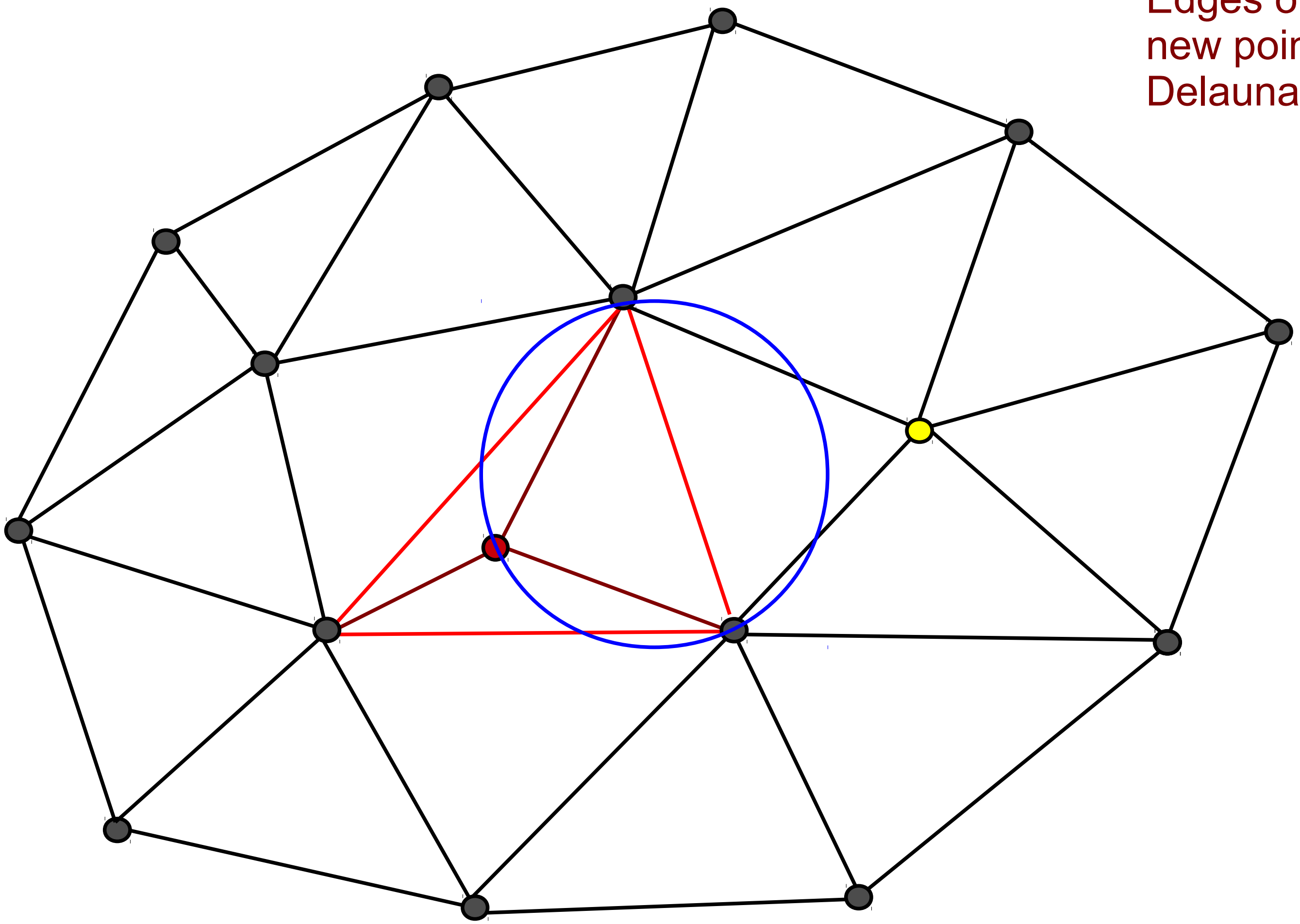
2. Step: Split the triangle into three triangles



Adding a point by sequential insertion

3. Step: Legalize the new triangles

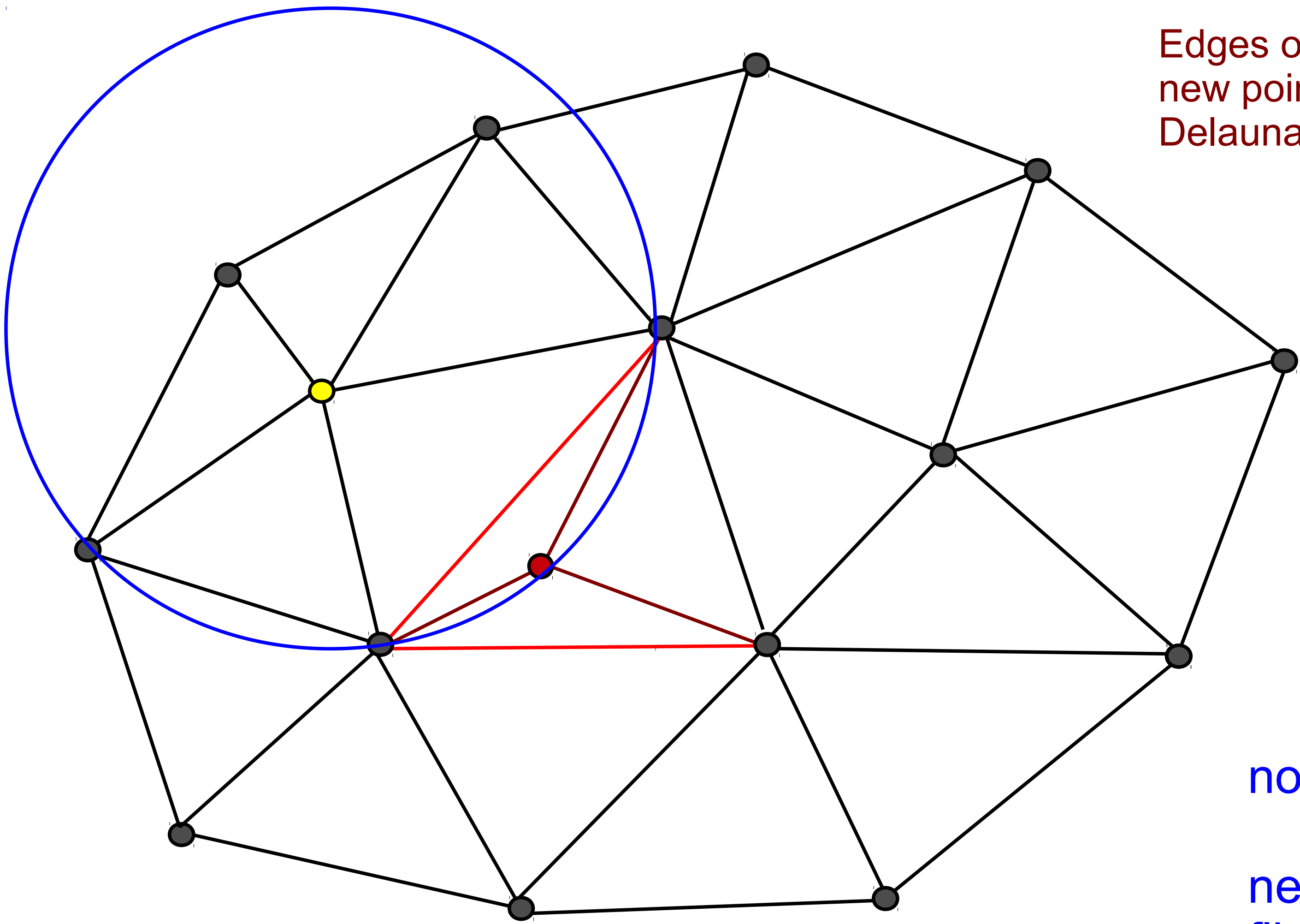
Edges opposite of new point may violate Delaunayhood



Ok!

Adding a point by sequential insertion

3. Step: Legalize the new triangles



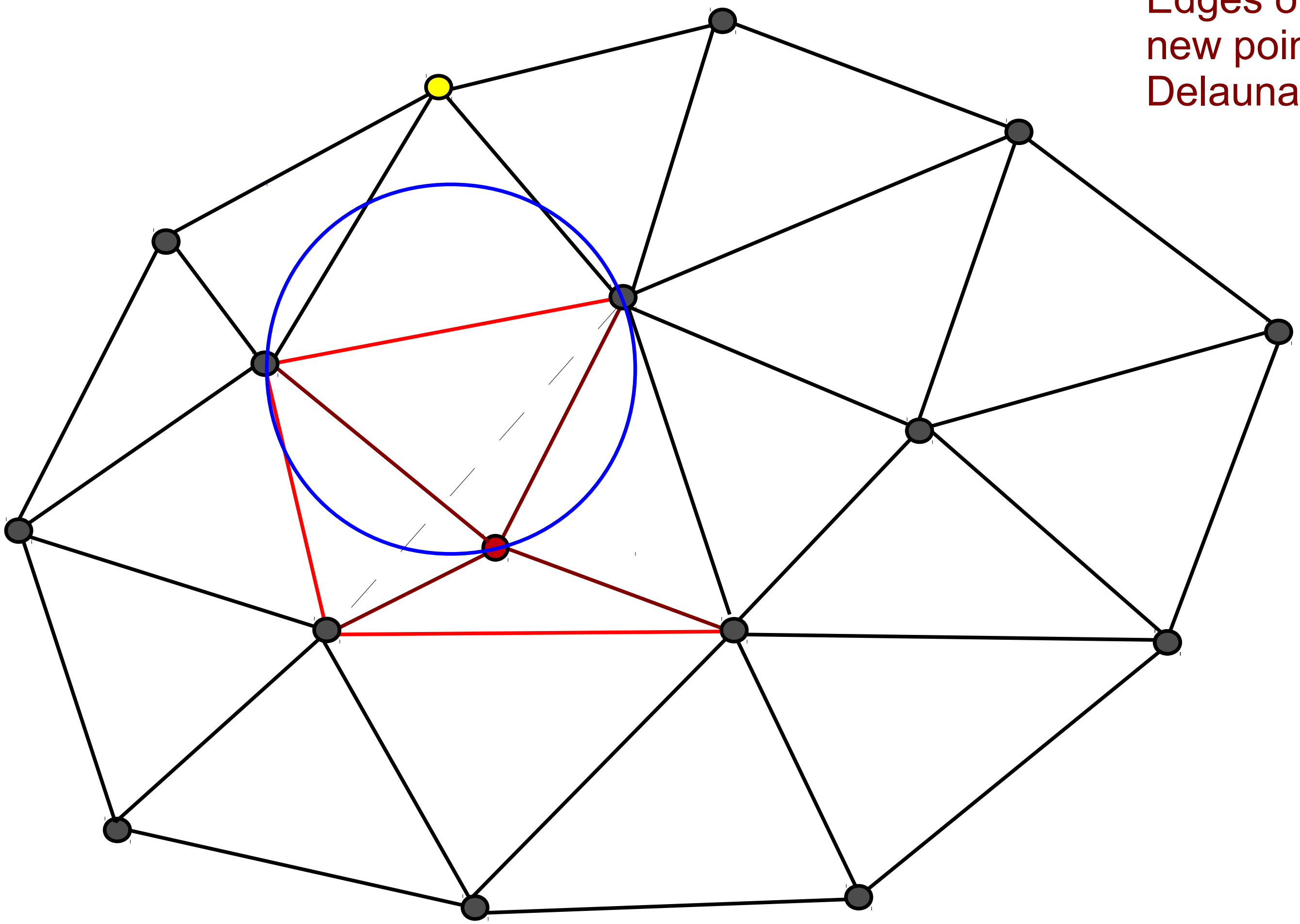
Edges opposite of new point may violate Delaunayhood

not ok...!

need to flip edge

Adding a point by sequential insertion

3. Step: Legalize the new triangles

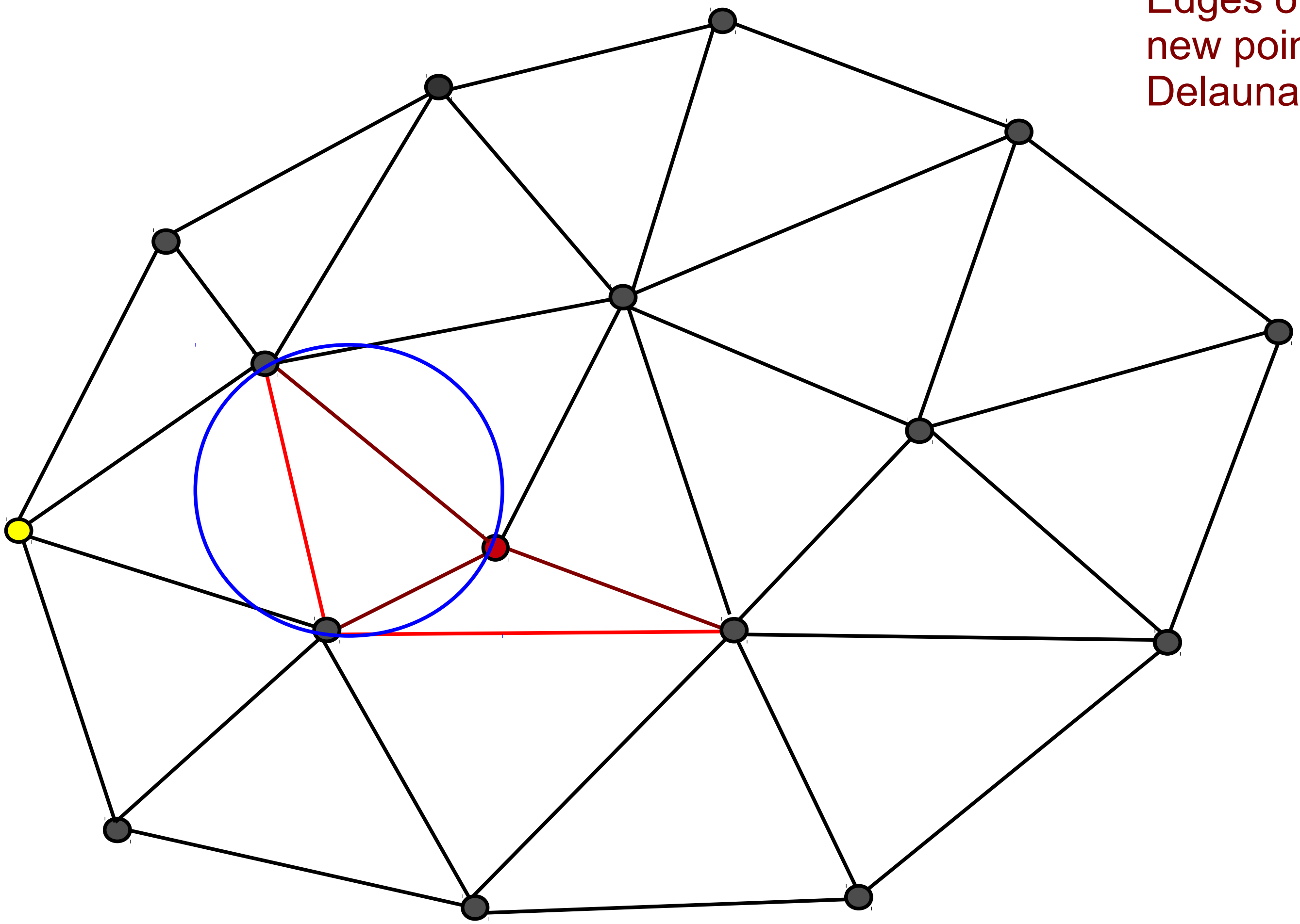


Edges opposite of new point may violate Delaunayhood

Ok!

Adding a point by sequential insertion

3. Step: Legalize the new triangles

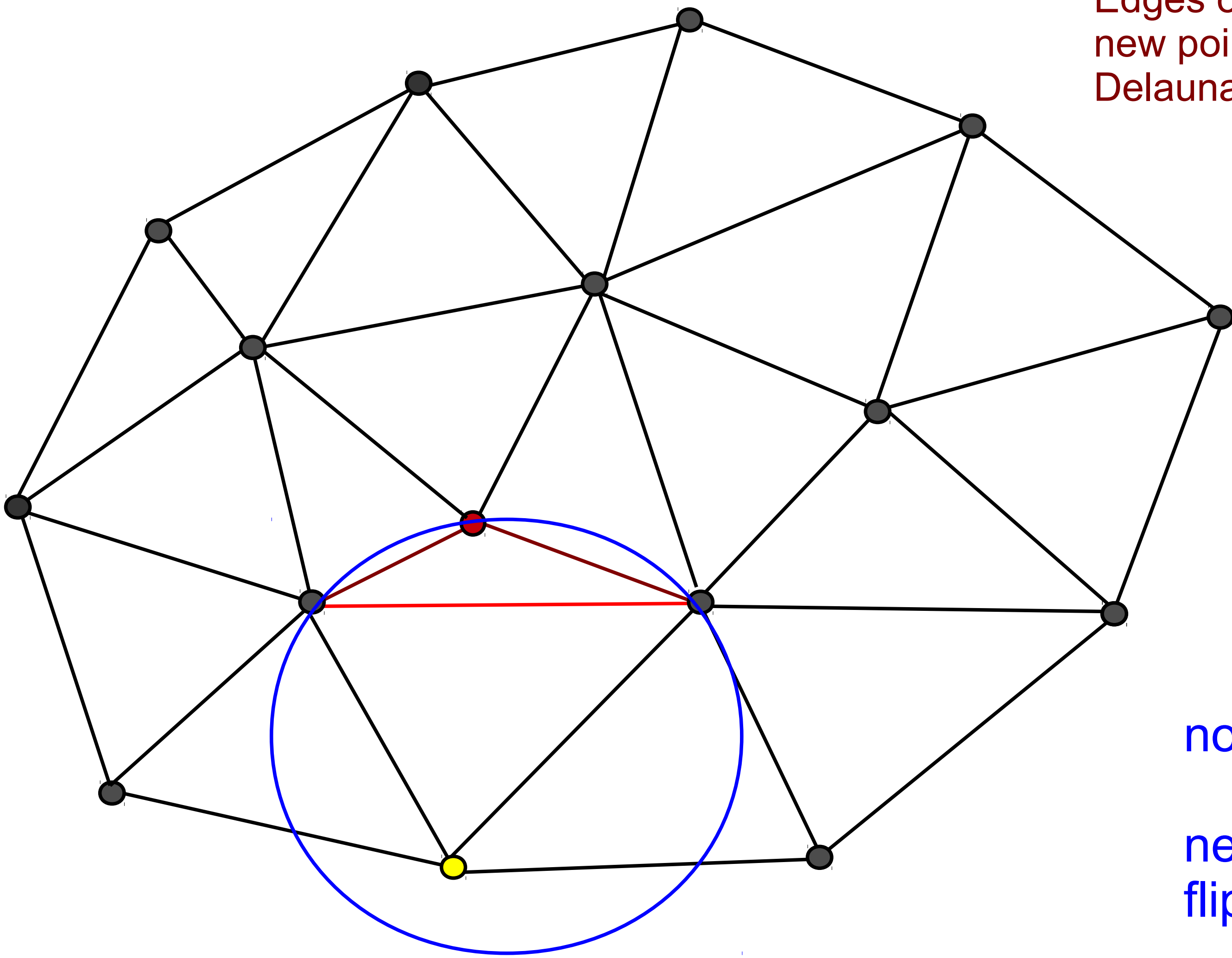


Edges opposite of
new point may violate
Delaunayhood

Ok!

Adding a point by sequential insertion

3. Step: Legalize the new triangles



Edges opposite of new point may violate Delaunayhood

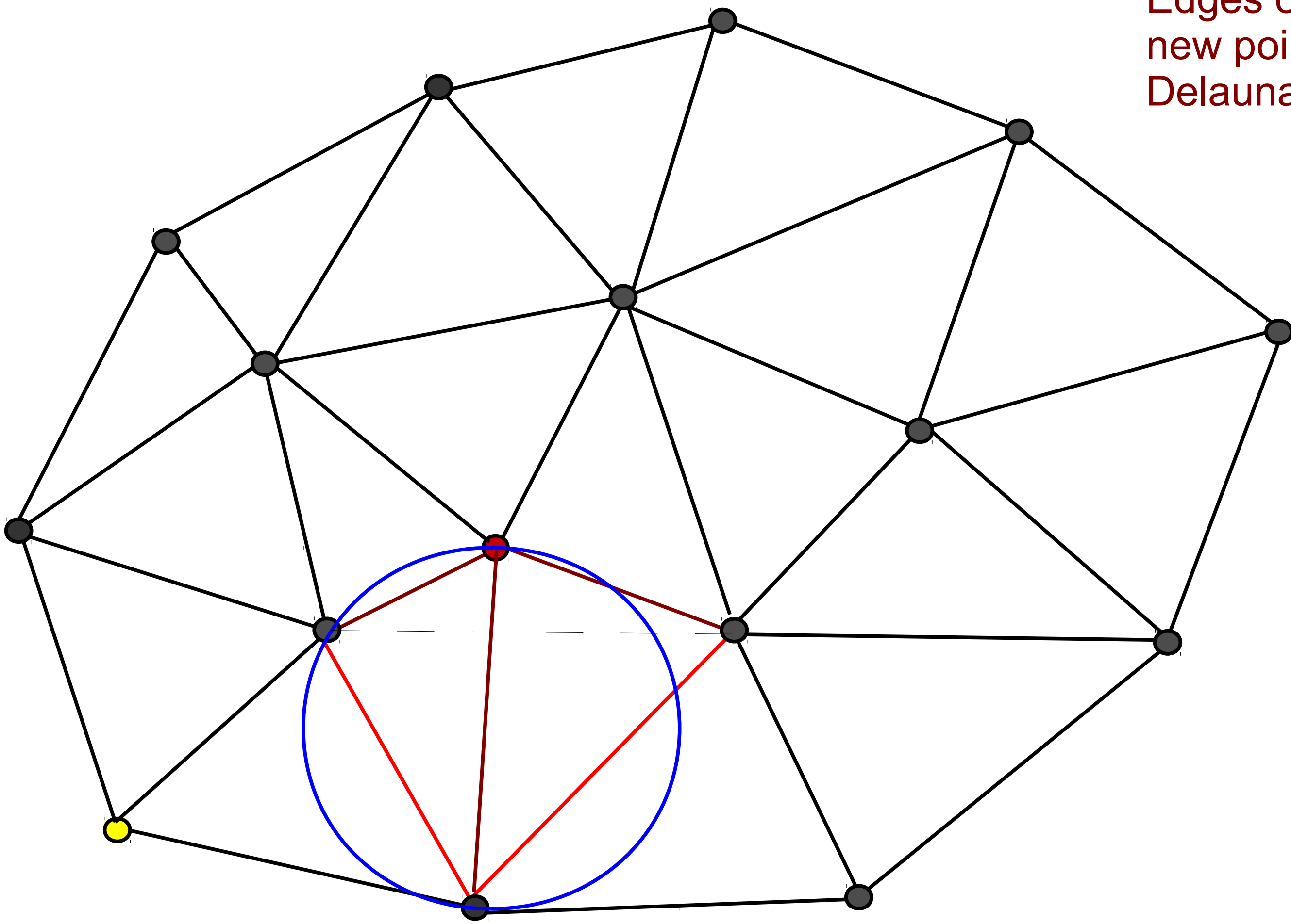
not ok...!

need to flip edge

Adding a point by sequential insertion

3. Step: Legalize the new triangles

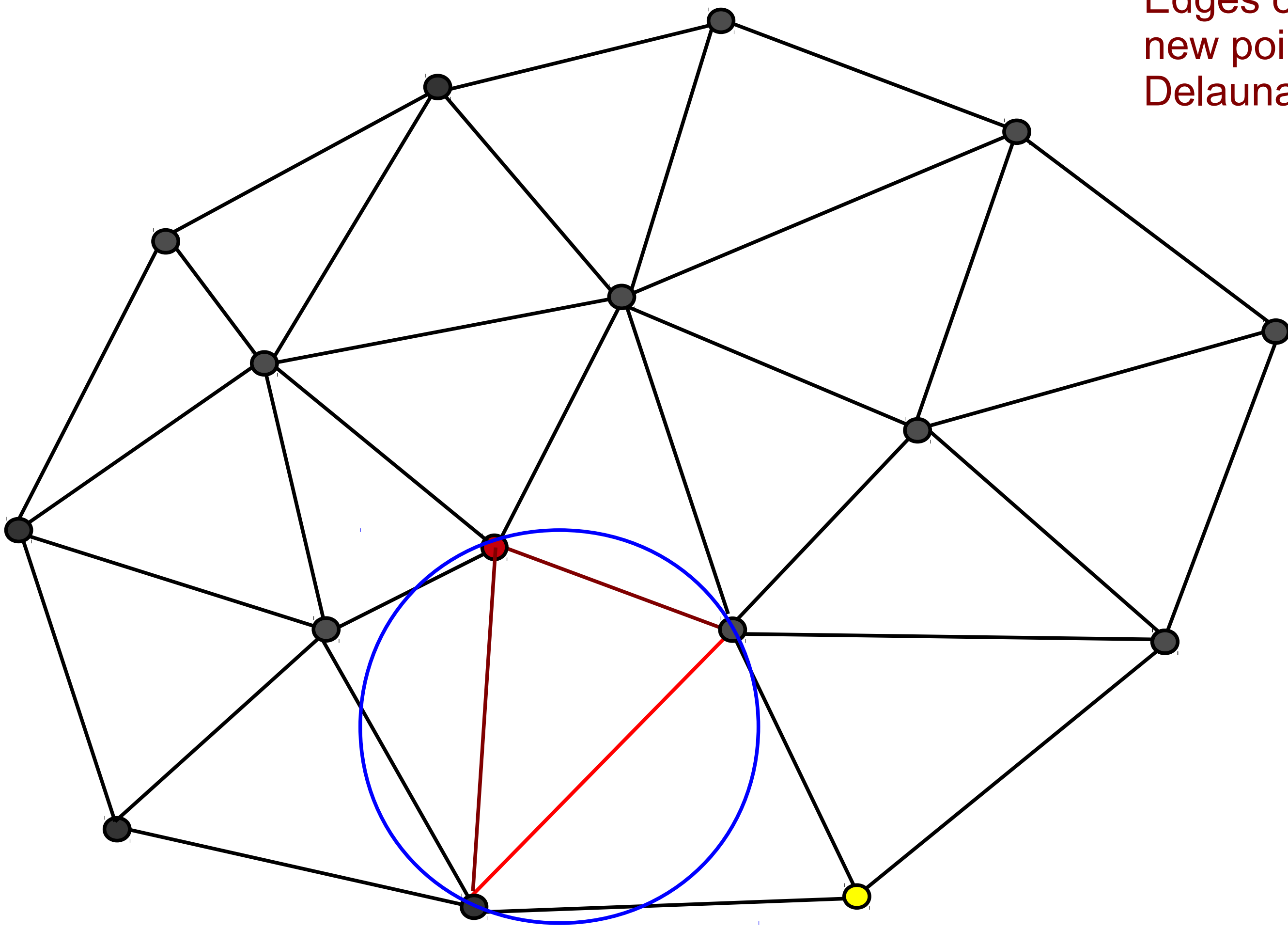
Edges opposite of new point may violate Delaunayhood



Ok!

Adding a point by sequential insertion

3. Step: Legalize the new triangles

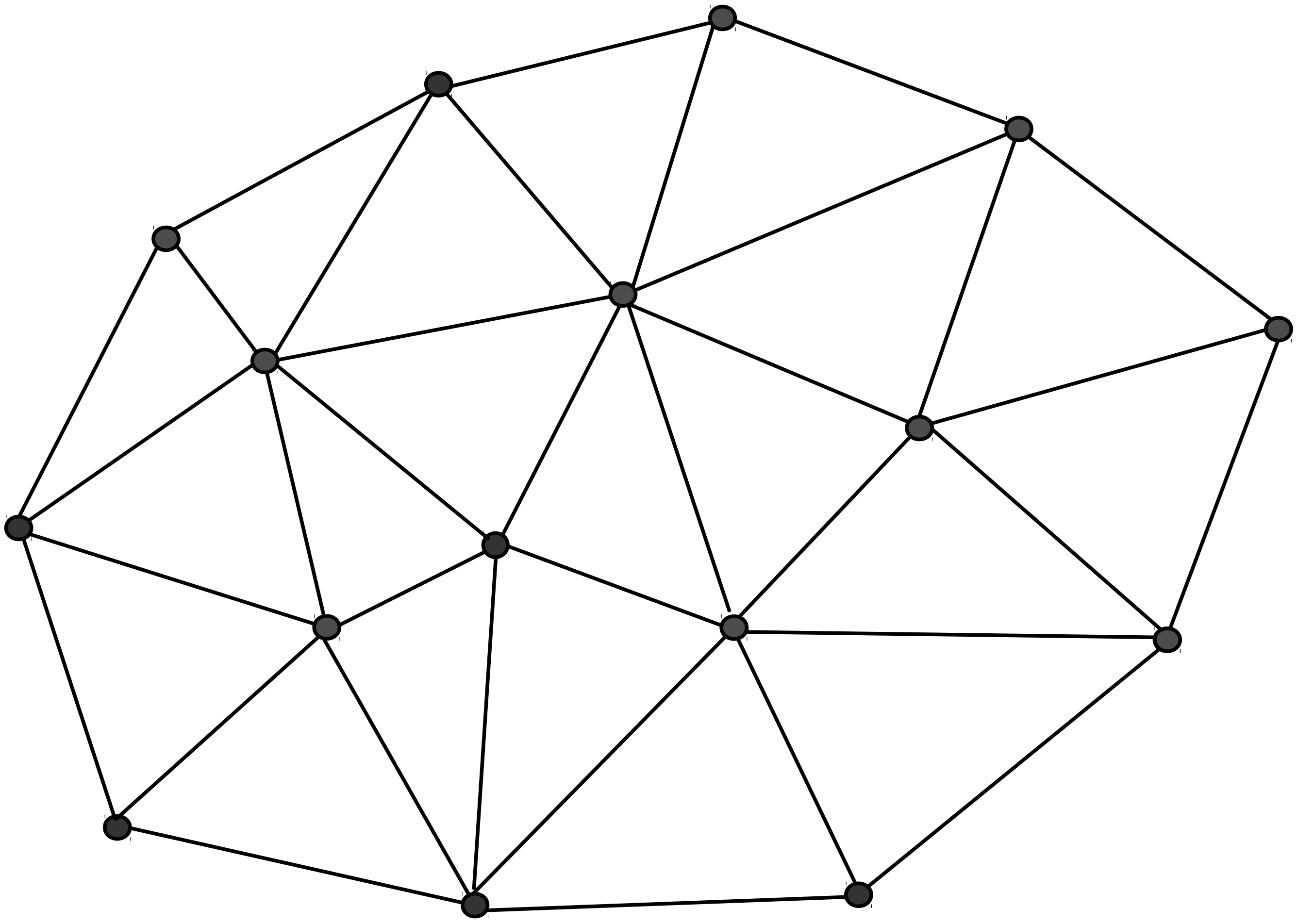


Edges opposite of new point may violate Delaunayhood

Ok!

Adding a point by sequential insertion

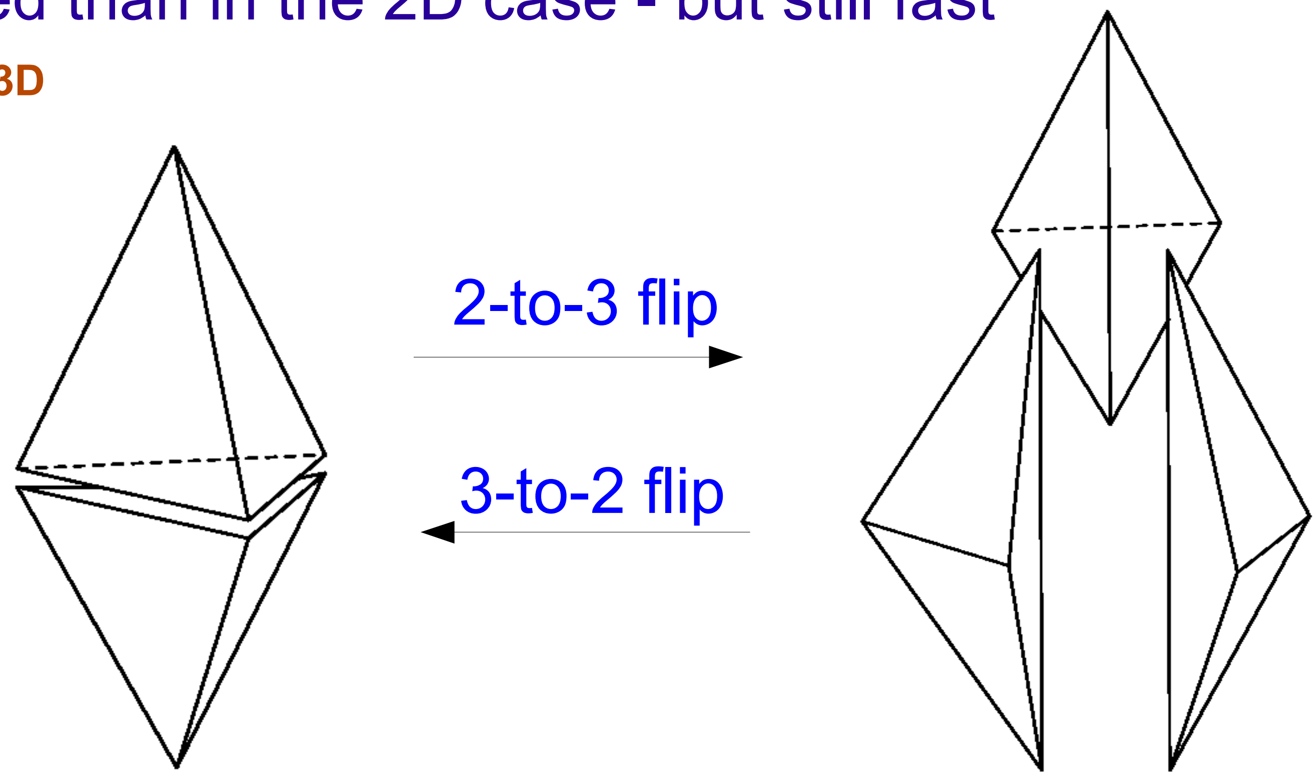
4. Step: Finished! (Or insert next point)



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

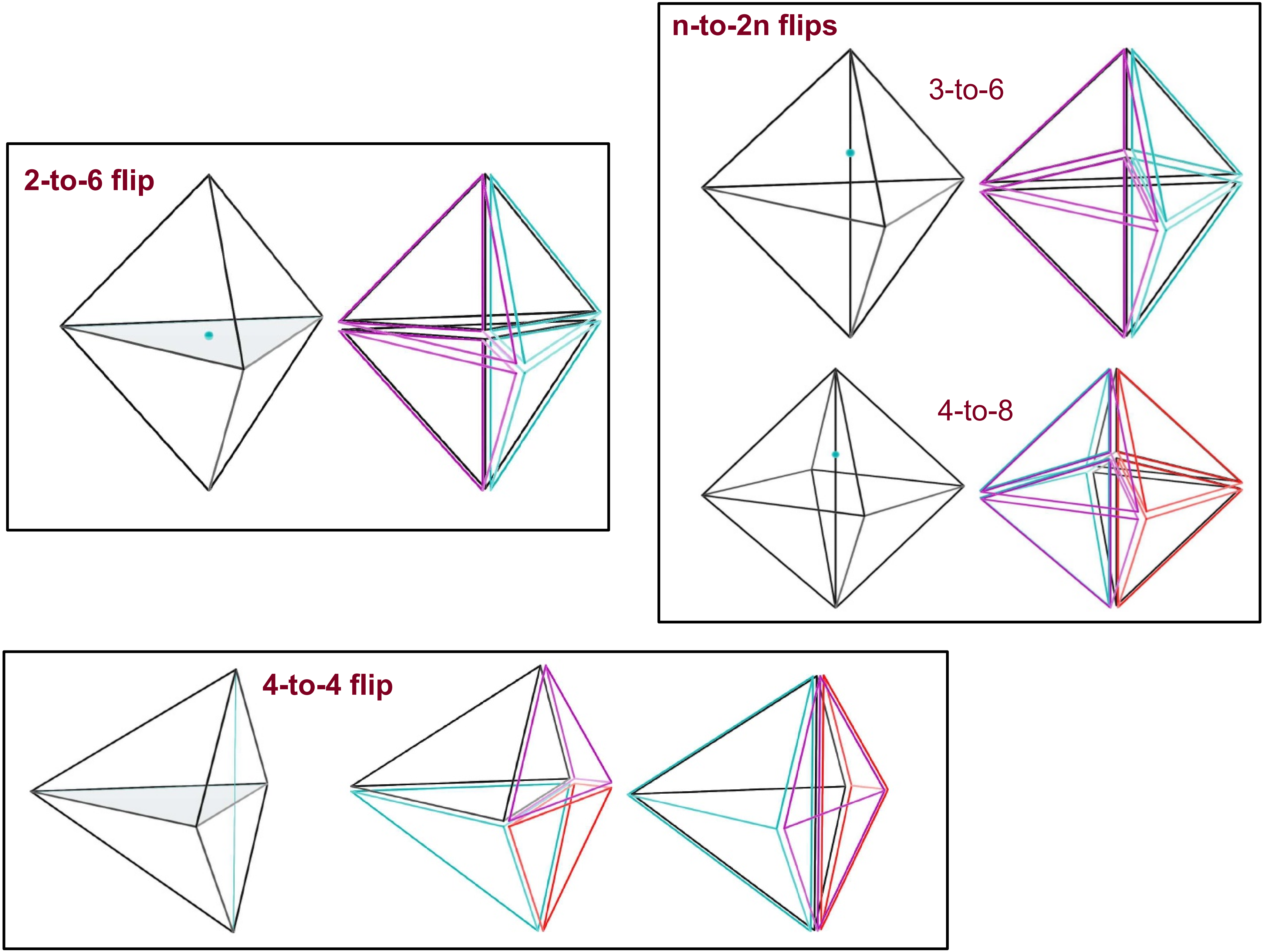
FLIP OPERATIONS IN 3D

1-to-4 flip
(point insertion)



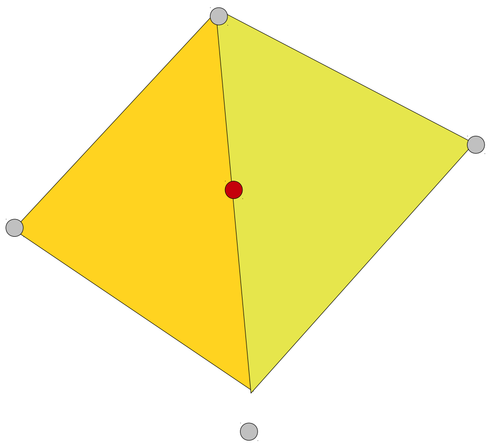
If the **general position assumption** is not fulfilled, degenerate cases can occur. This makes thinks a lot more complicated. One then needs:

- **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)


2-to-6 flip
n-to-2n flips
3-to-6
4-to-8
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)

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

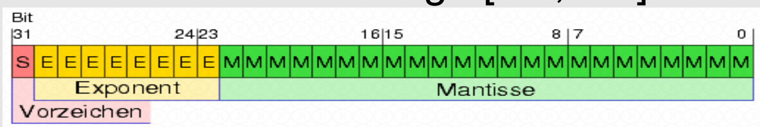
Delaunay algorithms tend to crash if wrong decisions are made!

Solution

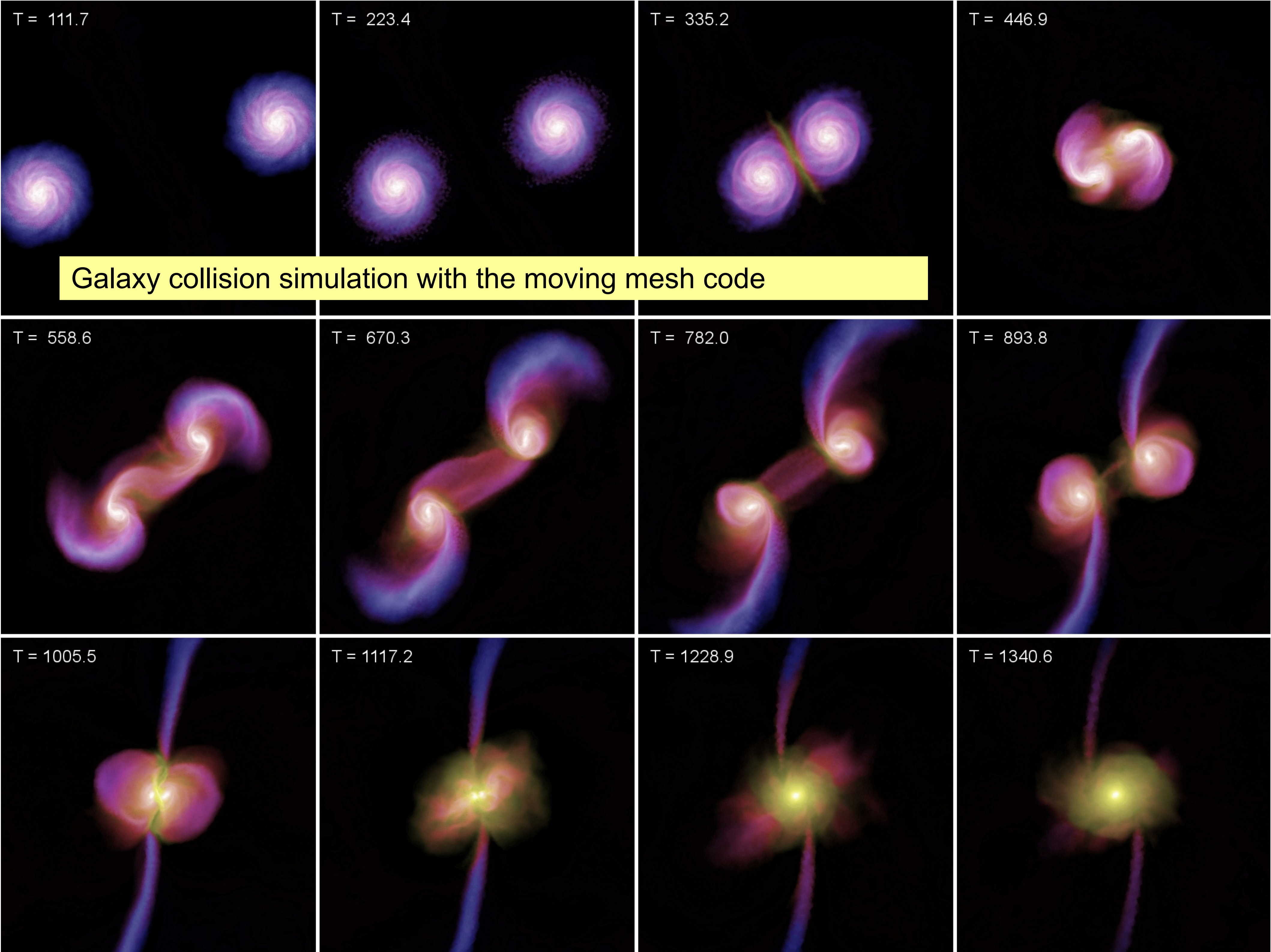
- **Calculate maximum round-off error** in geometric tests, and check whether result could be incorrect
- If the decision is ambiguous due to floating point round-off, **use exact arithmetic** instead

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

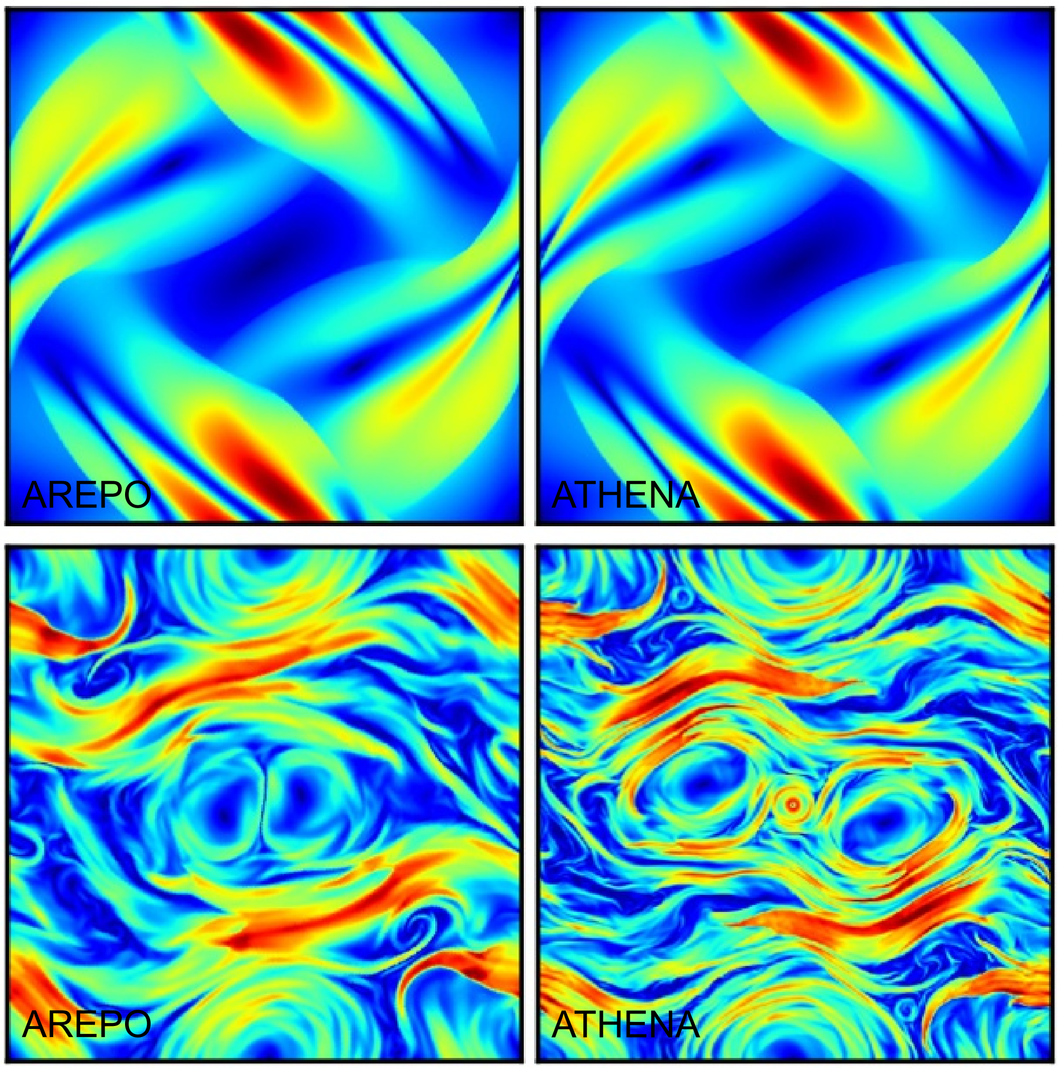

Galaxy collision simulation with the moving mesh code
T = 111.7
T = 223.4
T = 335.2
T = 446.9
T = 558.6
T = 670.3
T = 782.0
T = 893.8
T = 1005.5
T = 1117.2
T = 1228.9
T = 1340.6

There is an MHD implementation in AREPO that works reasonably well

EQUATIONS AND SOME TESTS

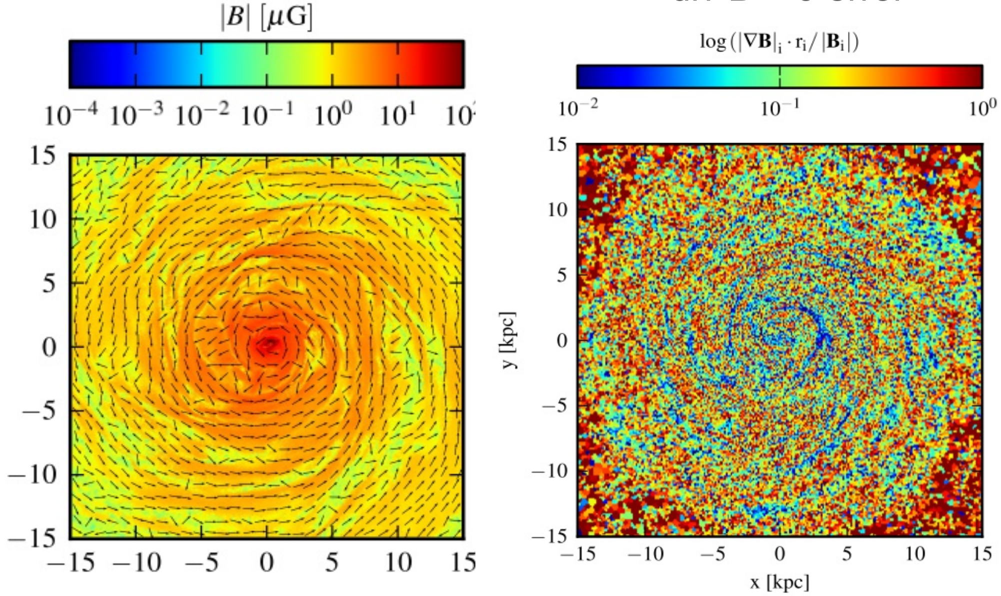
$$\mathbf{U} = \begin{pmatrix} \rho \\ \rho\mathbf{v} \\ \rho e \\ \mathbf{B} \\ \psi \end{pmatrix} \qquad \mathbf{F}(\mathbf{U}) = \begin{pmatrix} \rho\mathbf{v} \\ \rho\mathbf{v}\mathbf{v}^T + p - \mathbf{B}\mathbf{B}^T \\ \rho e\mathbf{v} + p\mathbf{v} - \mathbf{B}\,(\mathbf{v}\cdot\mathbf{B}) \\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

Orszag-Tang vortex test



Magnetic field in a disk galaxy

div B = 0 error

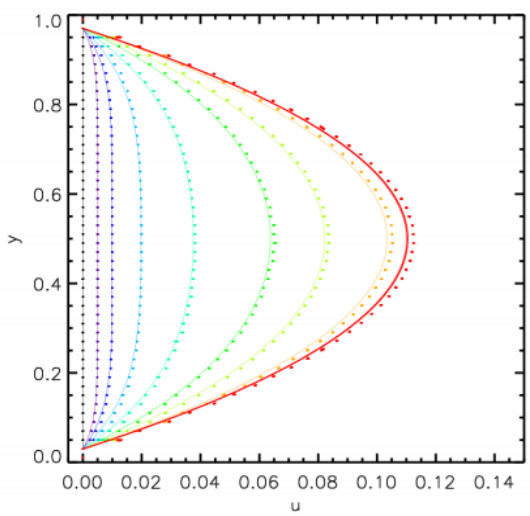


Pakmor, Bauer & Springel (2011)

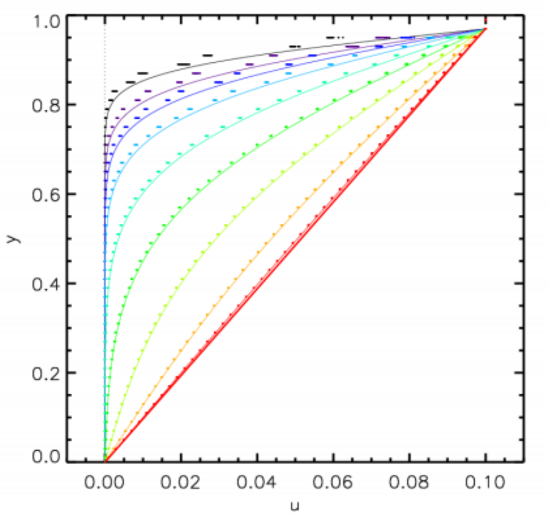
Explizit physical viscosity has been added to AREPO to obtain a **Navier-Stokes solver** on a moving mesh

SOME BASIC EXAMPLES

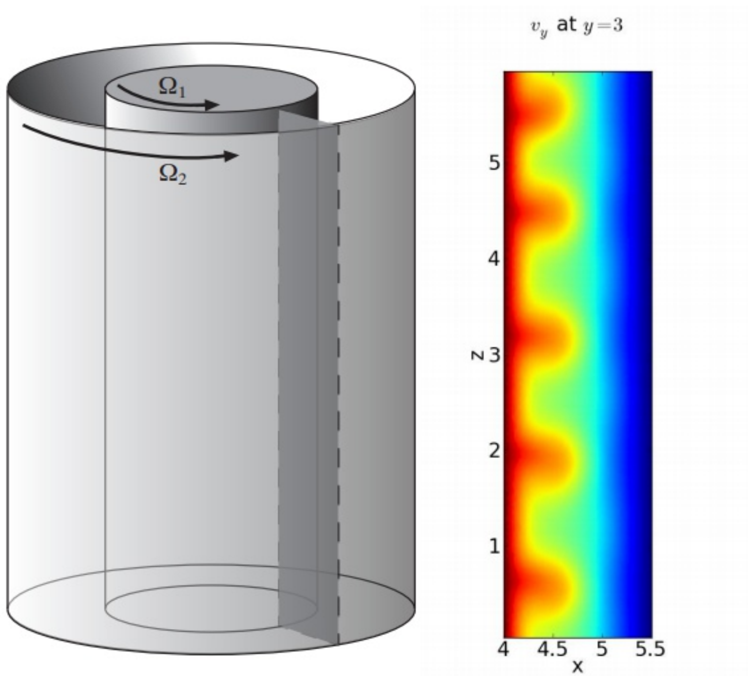
Munoz, VS et al. (2012)



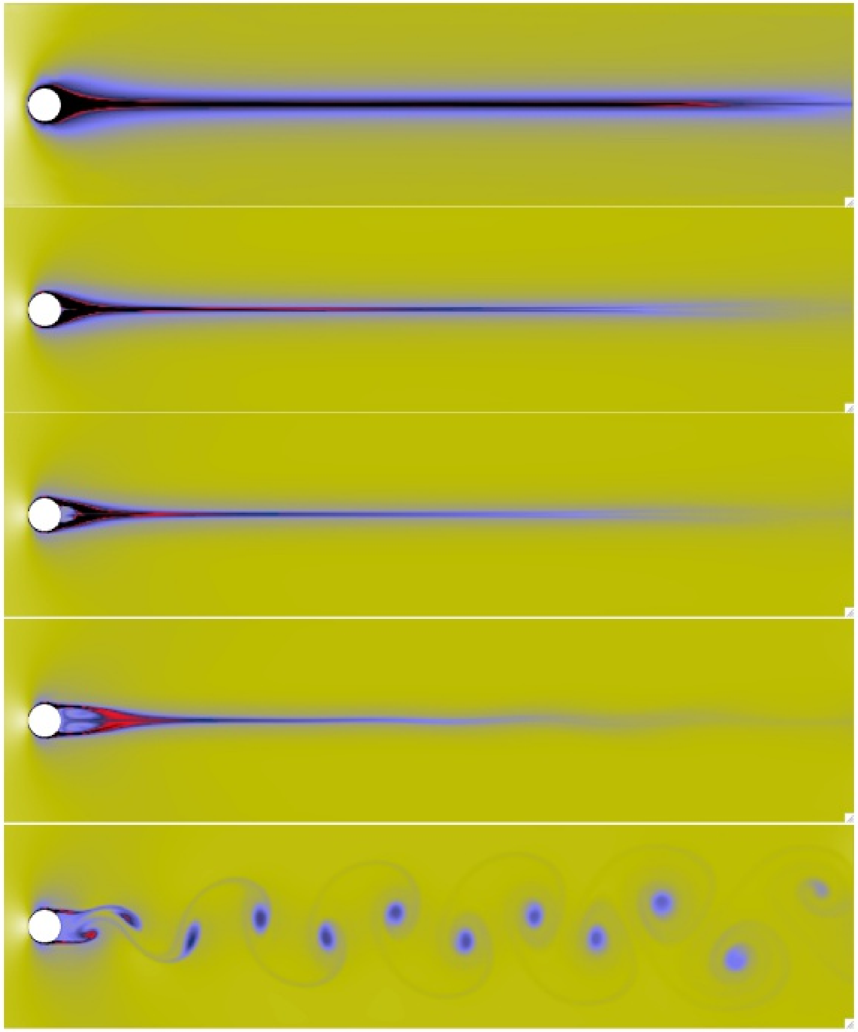
(a) Plane Poiseuille flow



(b) Plane Couette flow



Taylor vortex flow



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)

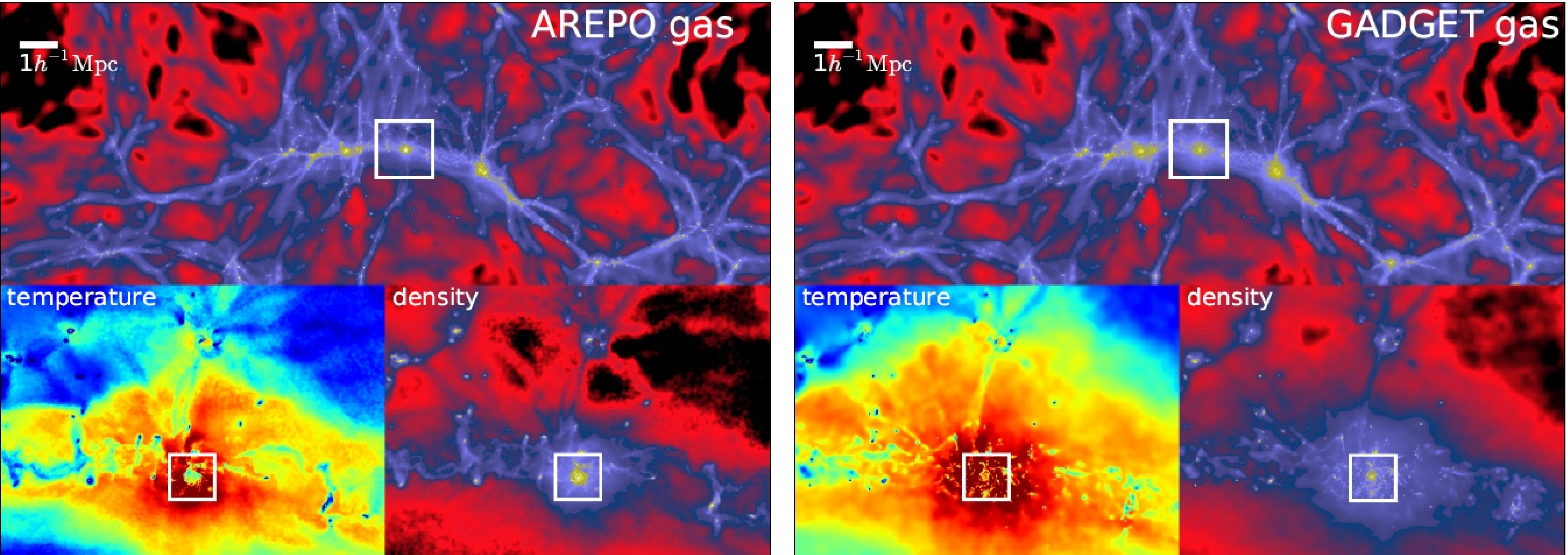
Subsonic turbulence in moving-mesh and SPH

Thomas Greif, VS, et al. (2011)

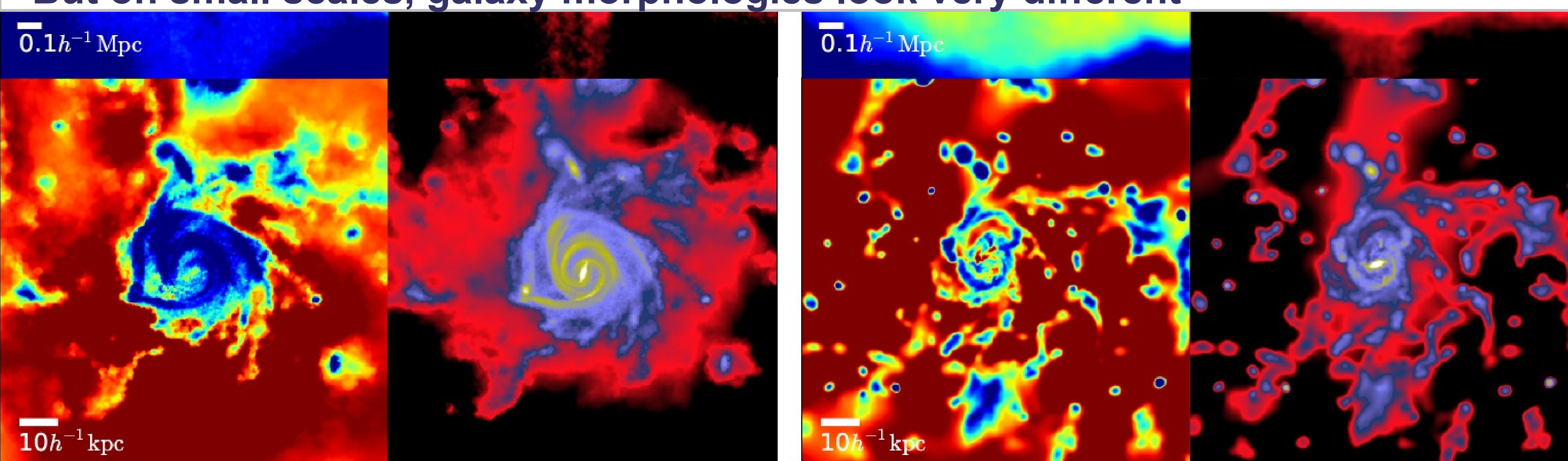
Population III star formation

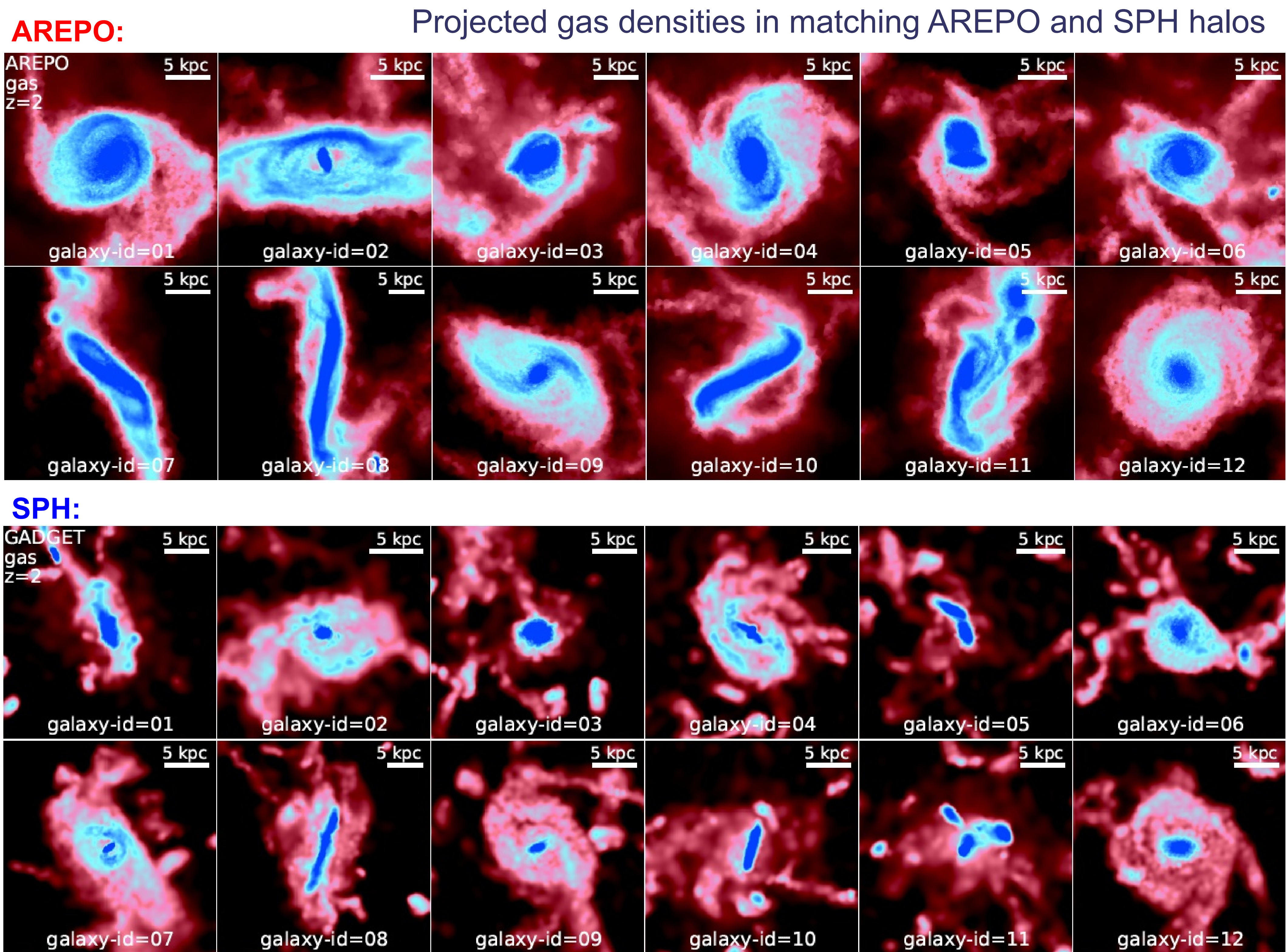
On large scales, the code produces similar results as standard SPH techniques

GAS AND TEMPERATURE FIELDS IN A COSMOLOGICAL HYDRODYNAMIC SIMULATION



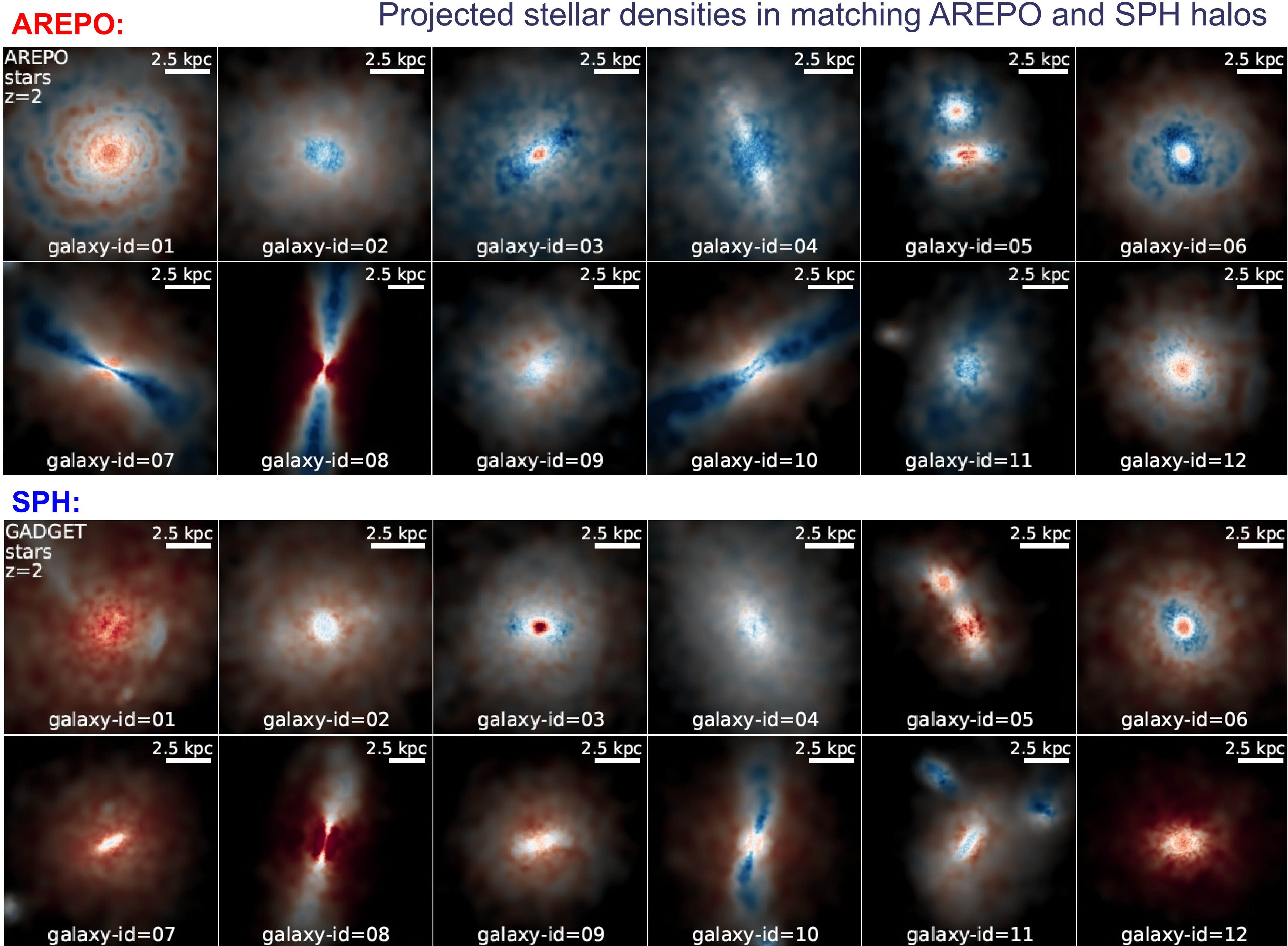
But on small scales, galaxy morphologies look very different





AREPO:
Projected gas densities in matching AREPO and SPH halos
AREPO
gas
z=2
5 kpc
galaxy-id=01
galaxy-id=02
galaxy-id=03
galaxy-id=04
galaxy-id=05
galaxy-id=06
galaxy-id=07
galaxy-id=08
galaxy-id=09
galaxy-id=10
galaxy-id=11
galaxy-id=12
SPH:
GADGET
gas
z=2
5 kpc
galaxy-id=01
galaxy-id=02
galaxy-id=03
galaxy-id=04
galaxy-id=05
galaxy-id=06
galaxy-id=07
galaxy-id=08
galaxy-id=09
galaxy-id=10
galaxy-id=11
galaxy-id=12

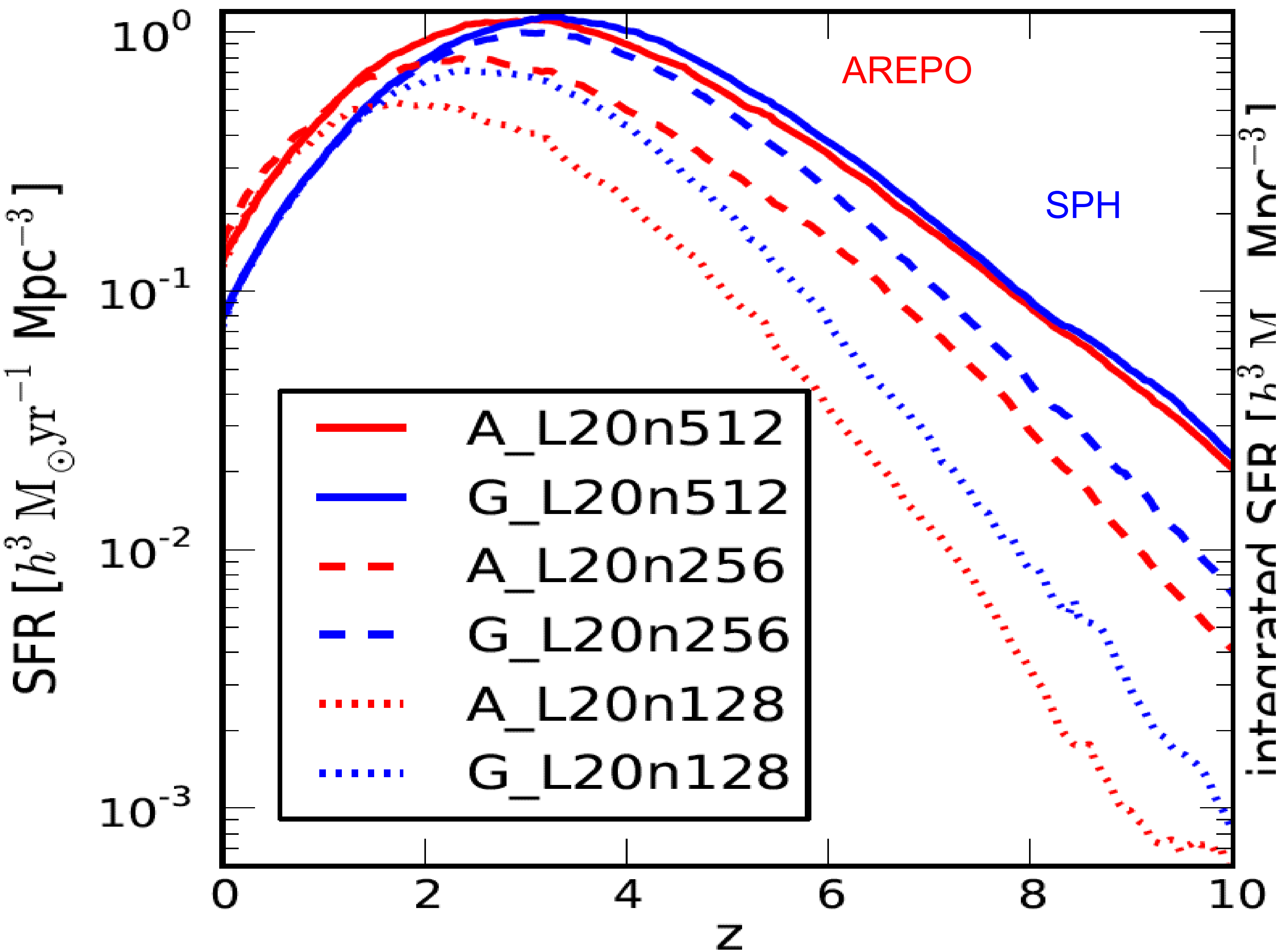
Projected stellar densities in matching AREPO and SPH halos



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

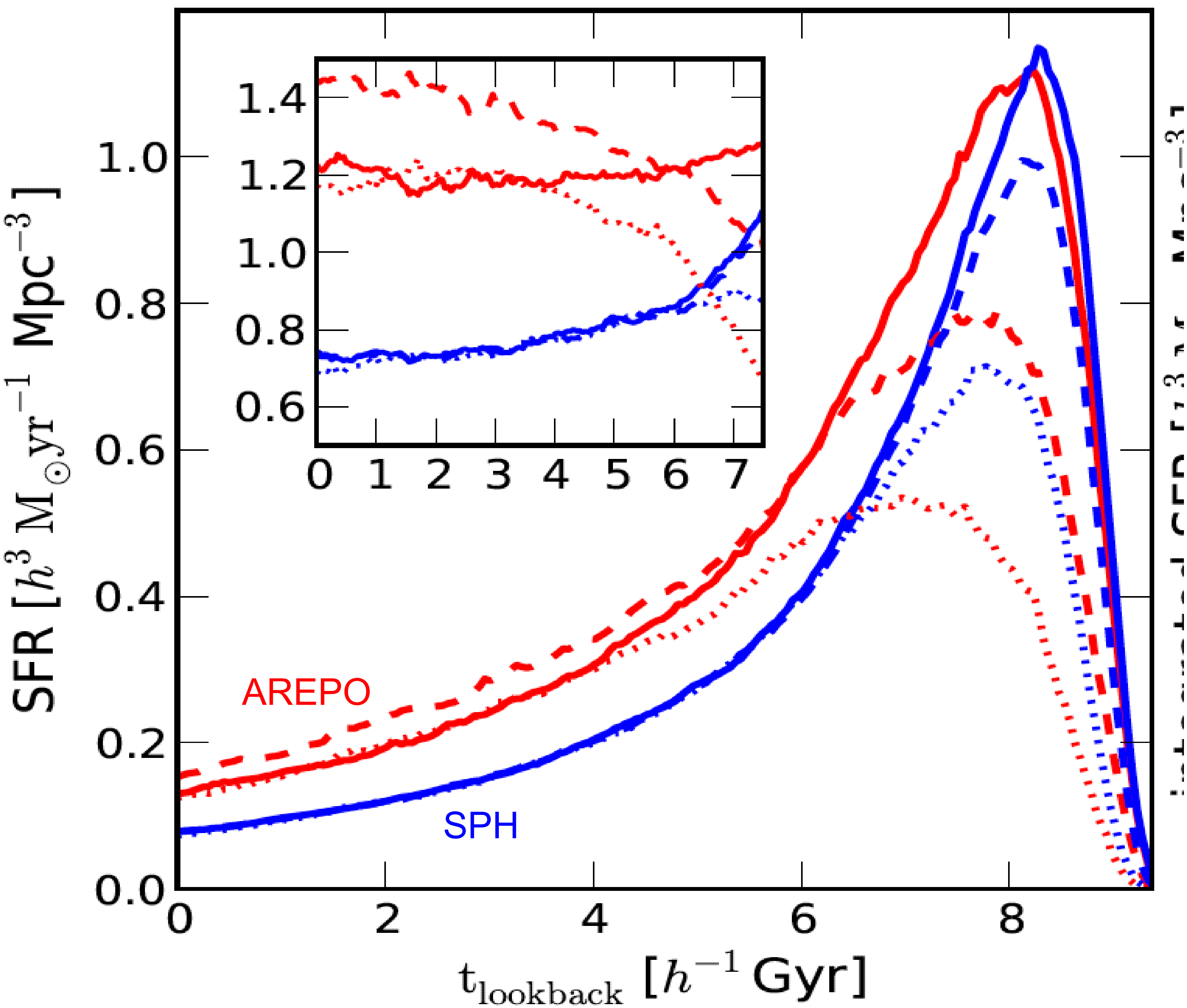
SFR-DENSITY AS A FUNCTION OF REDSHIFT FOR DIFFERENT RESOLUTIONS AND CODES

Vogelsberger et al. (2011)



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

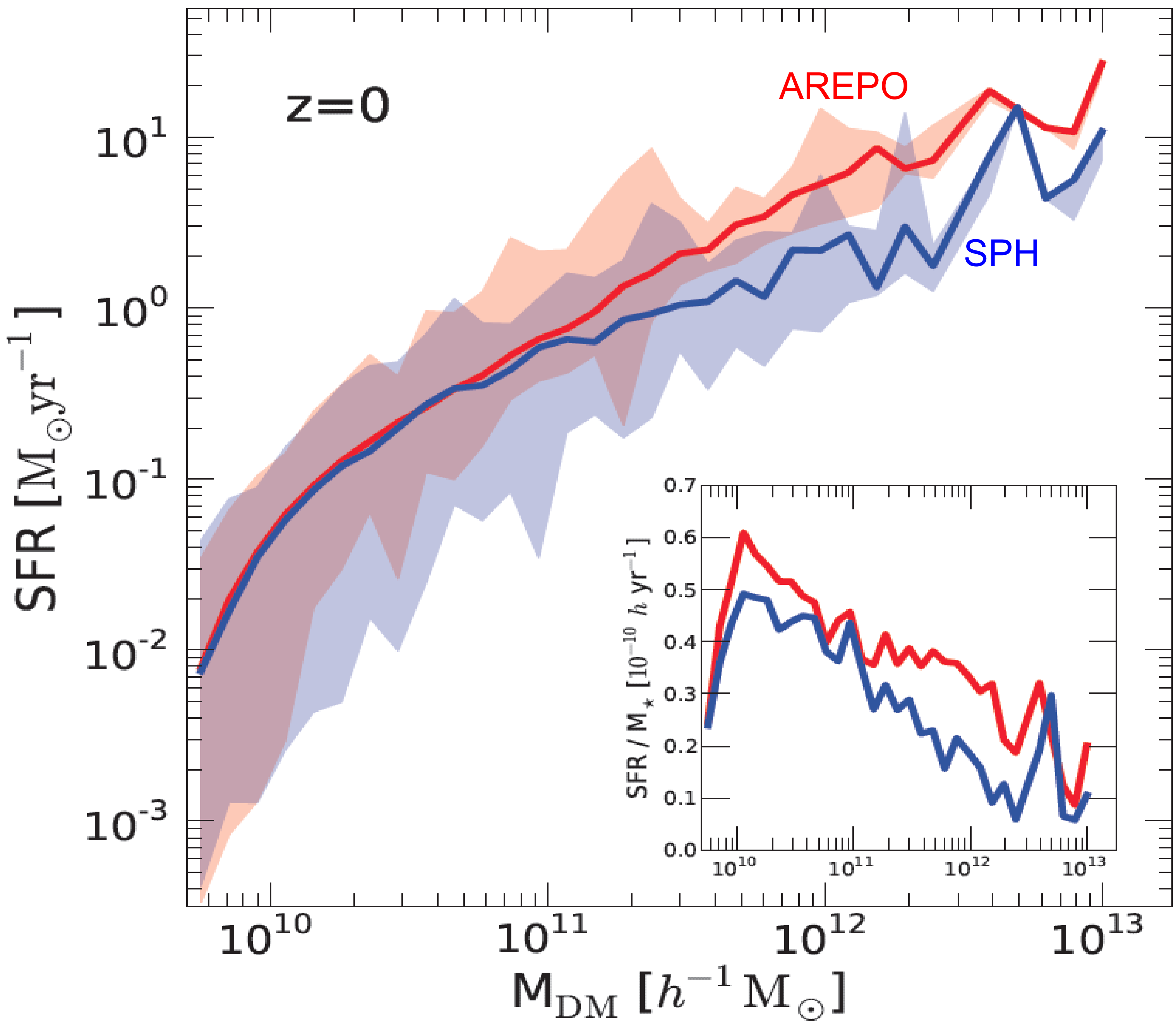
SFR-DENSITY AS A FUNCTION OF TIME FOR DIFFERENT RESOLUTIONS AND CODES



The difference in star formation originates in massive halos

STAR FORMATION RATE AS A FUNCTION OF HALO MASS

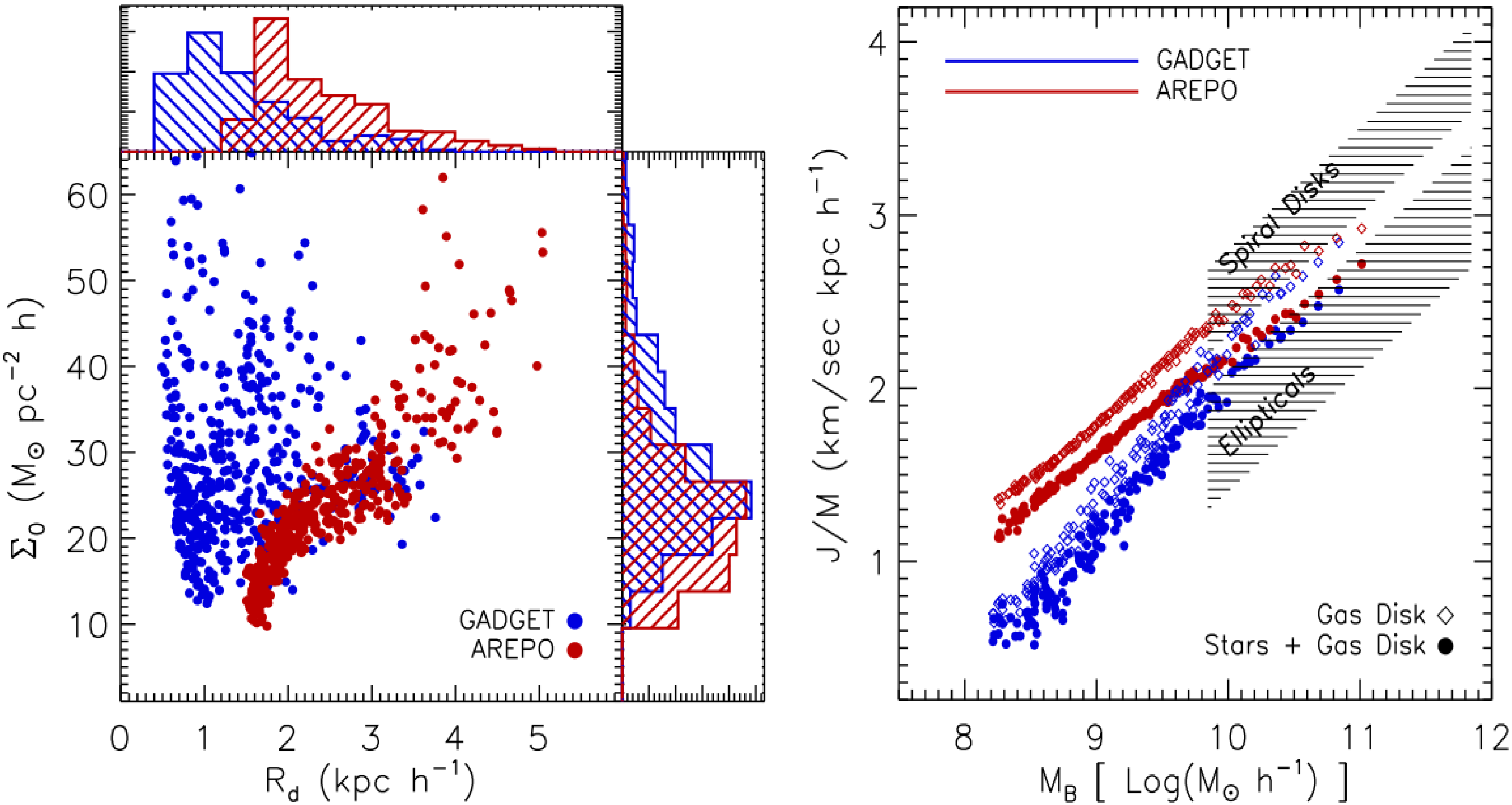
Vogelsberger et al. (2011)



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

DISK SCALE LENGTHS AND ANGULAR MOMENTUM IN GADGET AND AREPO

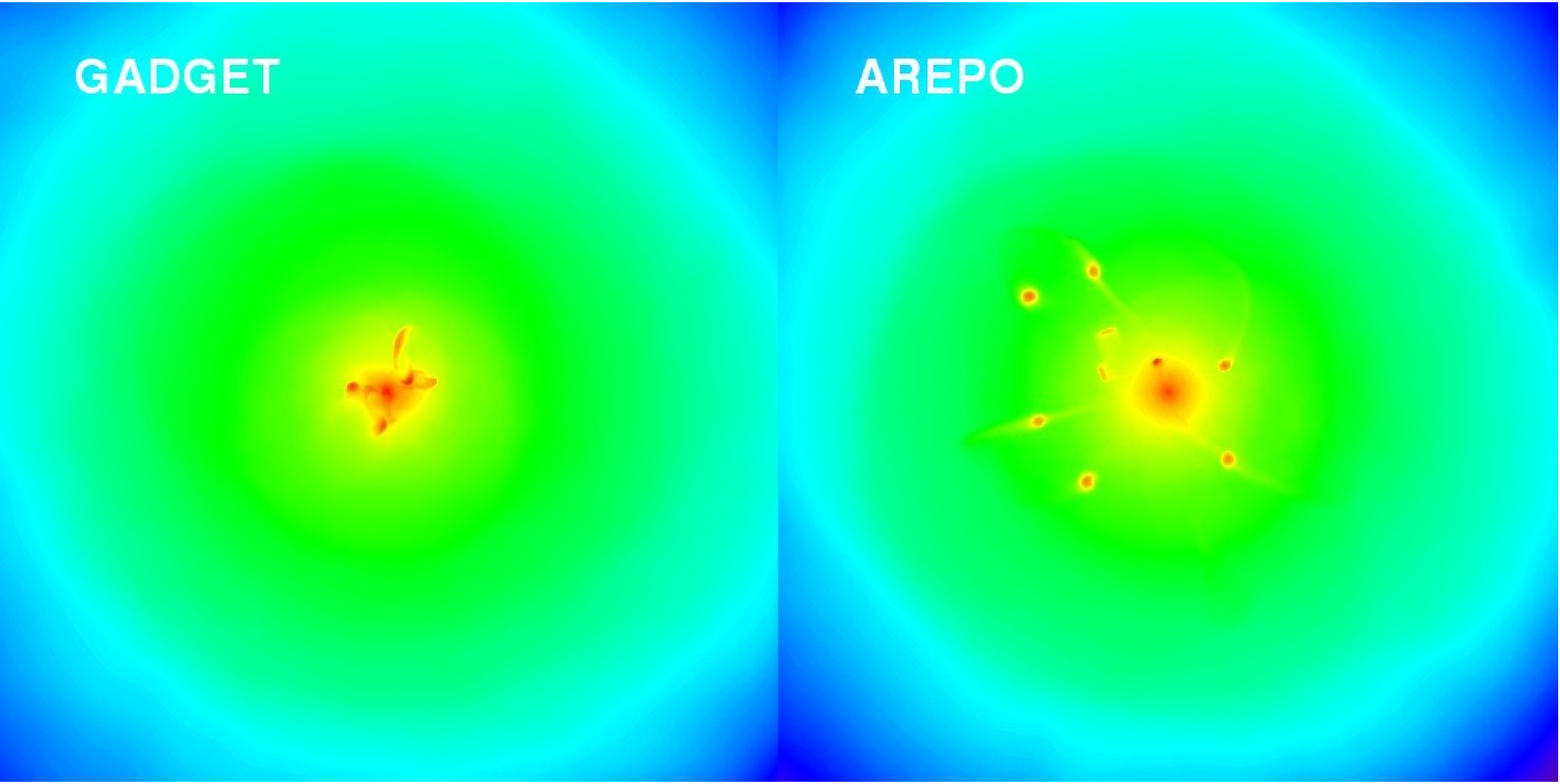
Torrey et al. (2011)



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

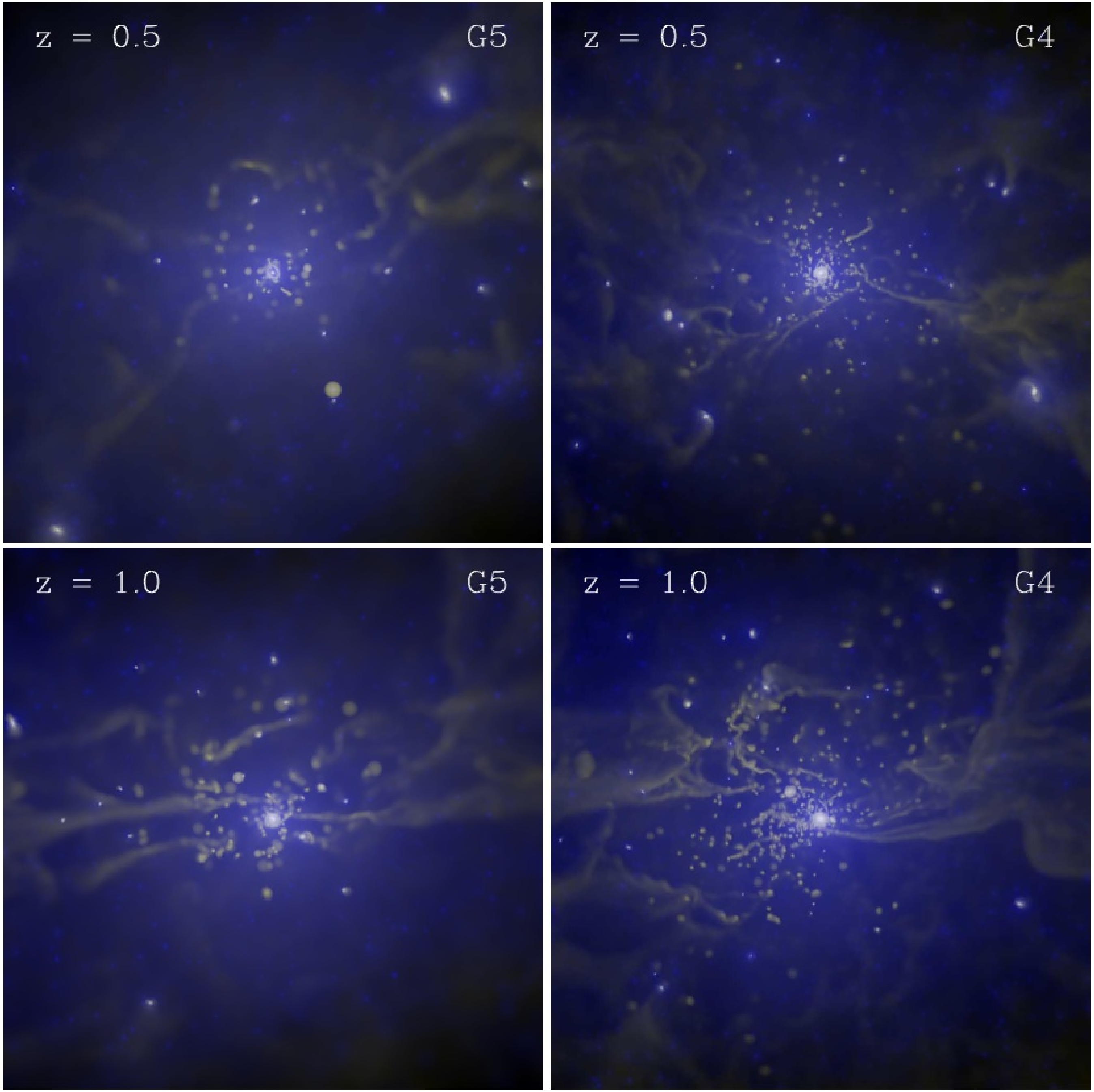
FIDUCIAL GAS BLOBS IN ORBIT IN A CLUSTER

Sijacki et al. (2011)



Clumpy gas distribution around Aquila galaxy in GADGET

GAS BLOBS IN ORBIT AROUND AQUILA AT DIFFERENT TIMES AND RESOLUTIONS

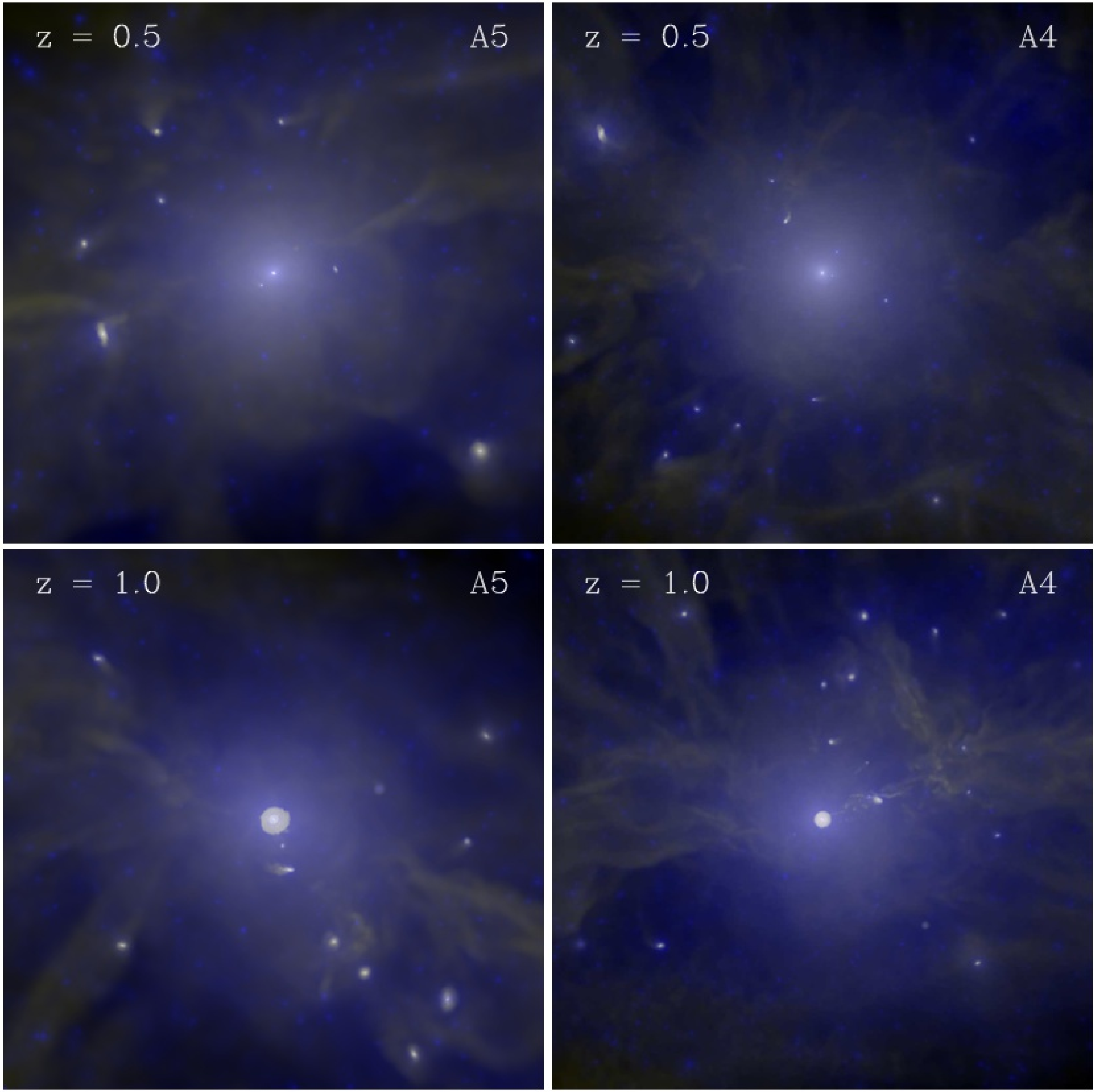


(Wadepuhl & Springel, 2011)

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

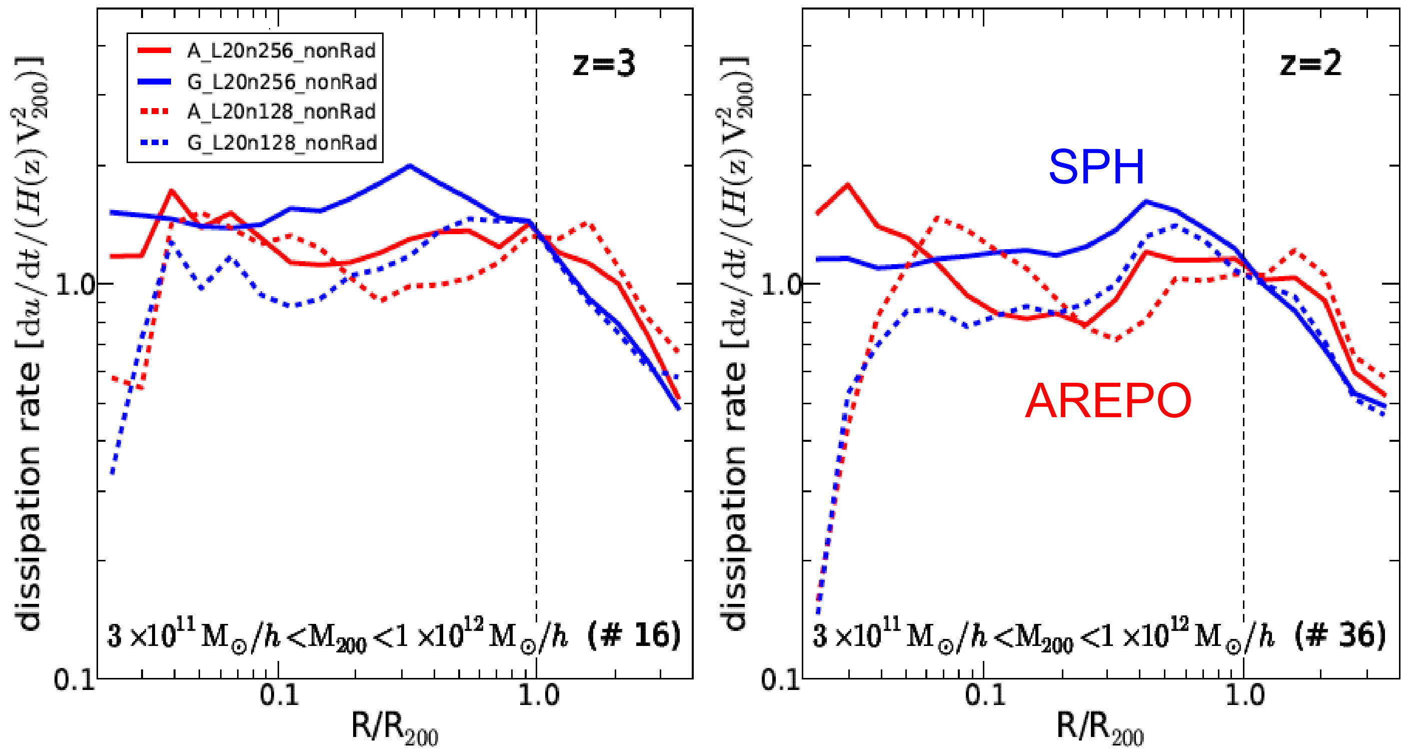


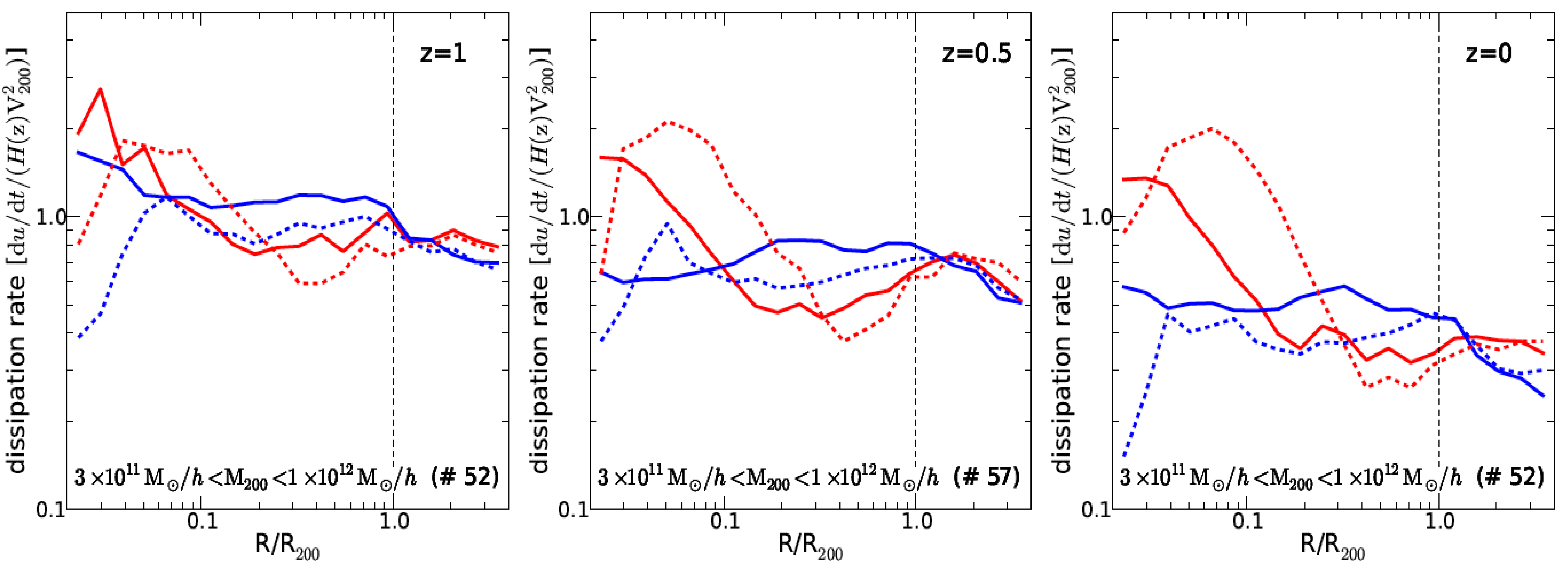
(Wadepuhl & Springel, 2011)

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)

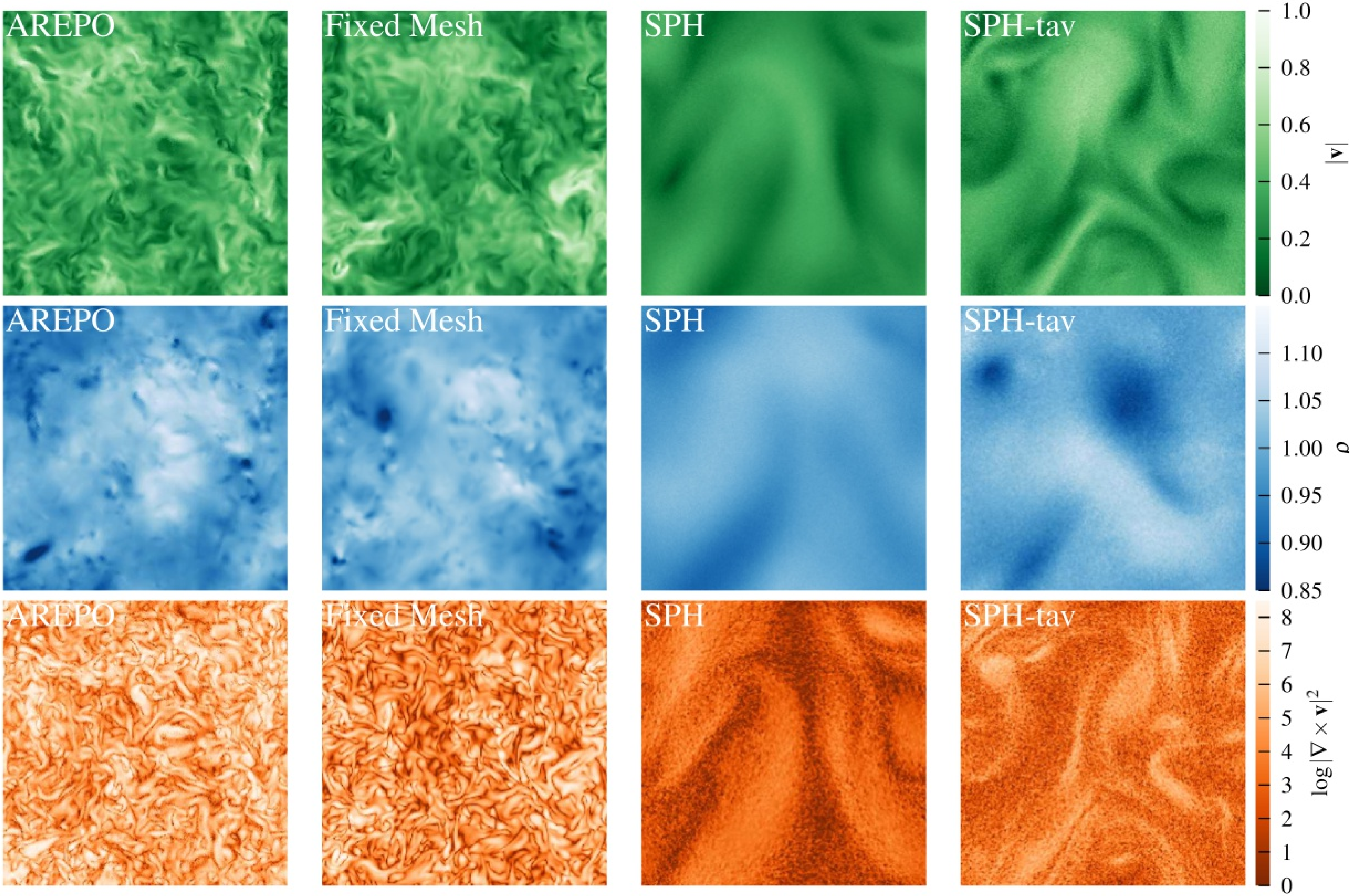




Enstrophy fields in **subsonic** turbulence are different in SPH and mesh-codes

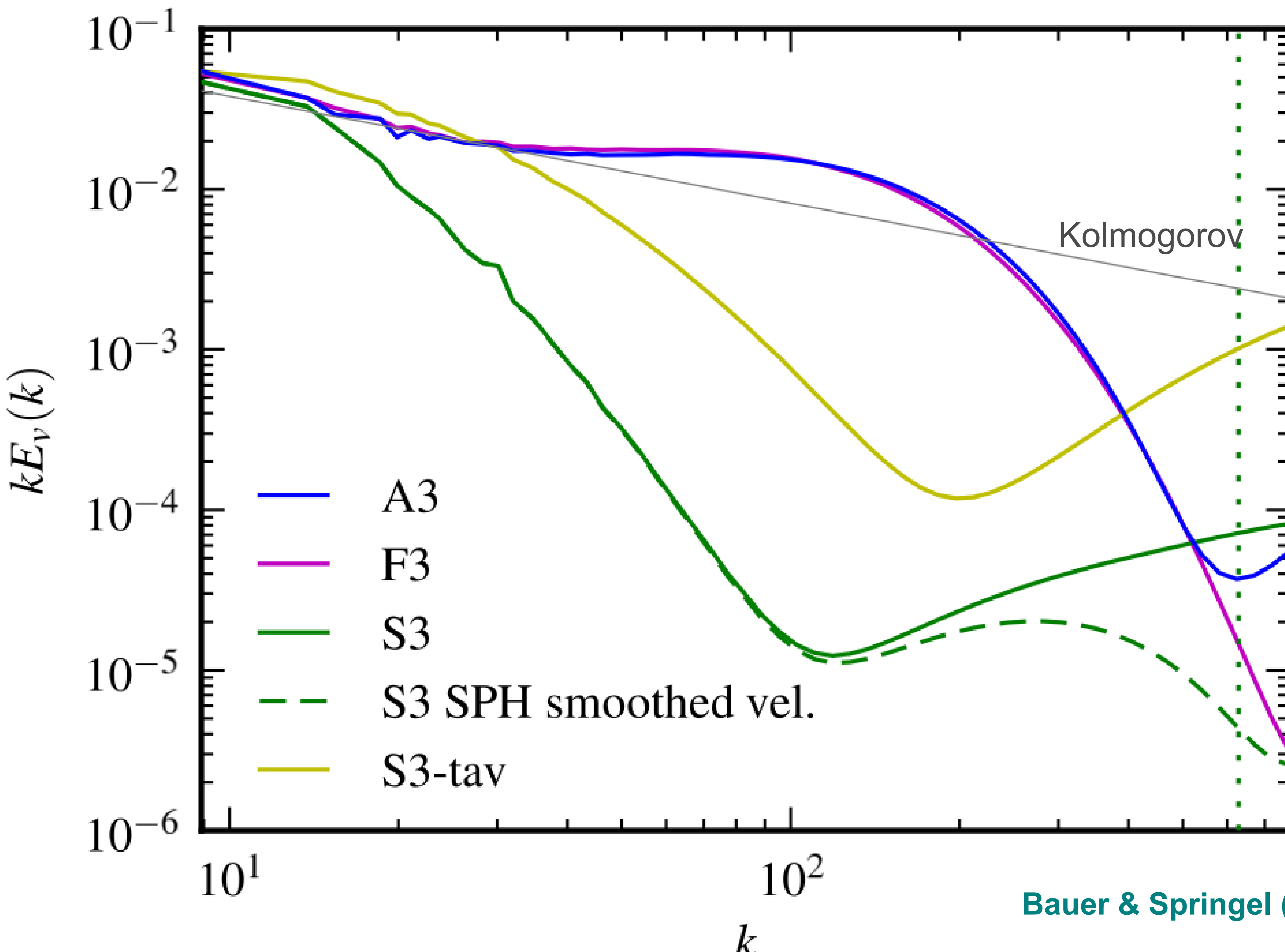
TURBULENT FIELDS FOR EQUAL DRIVING IN DIFFERENT SIMULATION CODES

Bauer & Springel (2012)



Driven subsonic turbulence in AREPO yields a Kolmogorov cascade

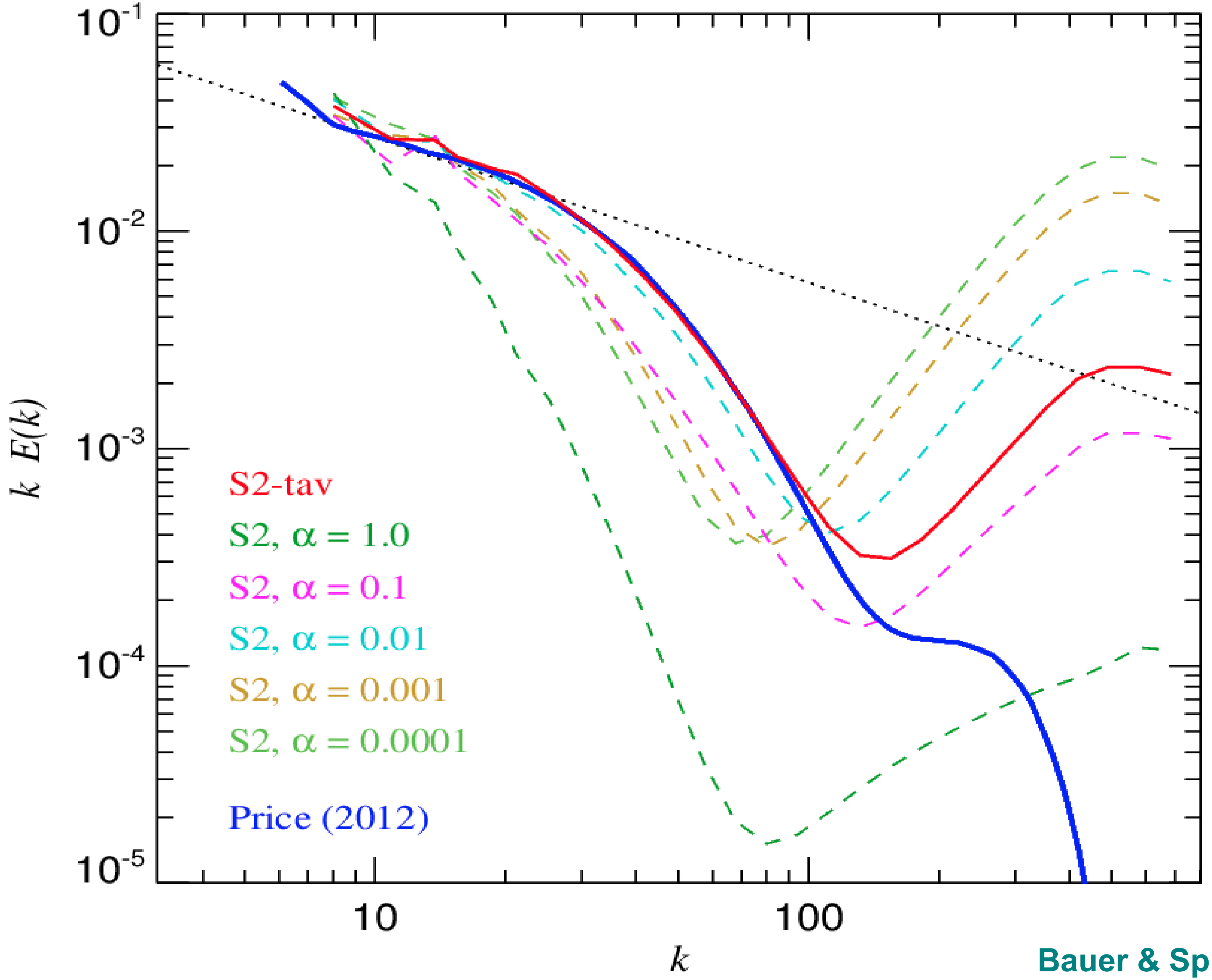
VELOCITY POWER SPECTRUM AT DIFFERENT RESOLUTIONS



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)

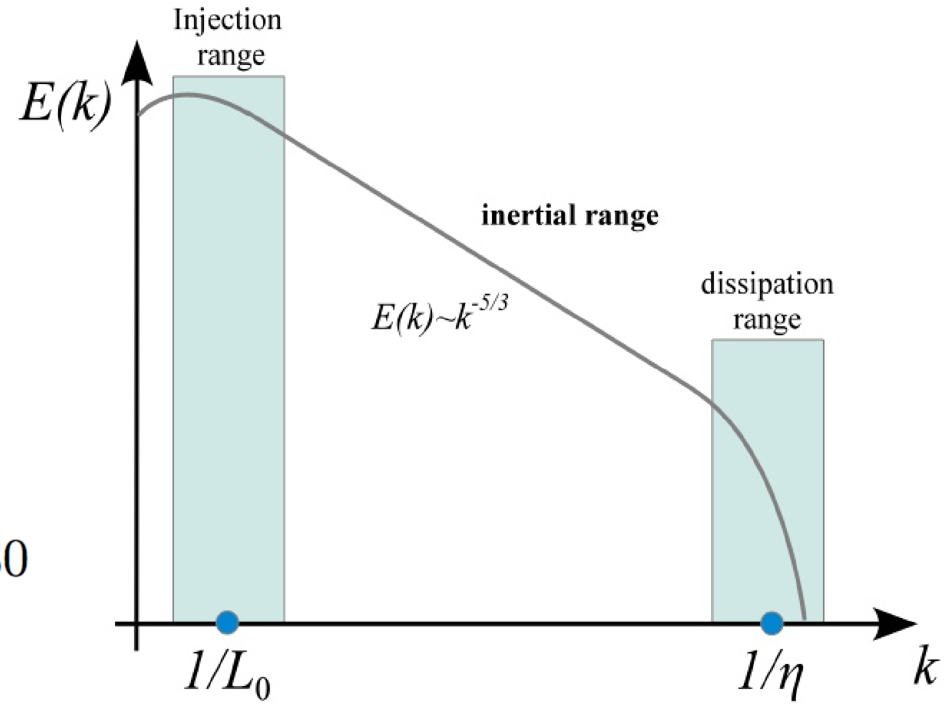
The shape of the dissipation range for Kolmogorv turbulence is universal

REYNOLDS NUMBERS AND THE KOLMOGOROV SCALE

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

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

For $\mathrm{Re} = 6000$ 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:

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

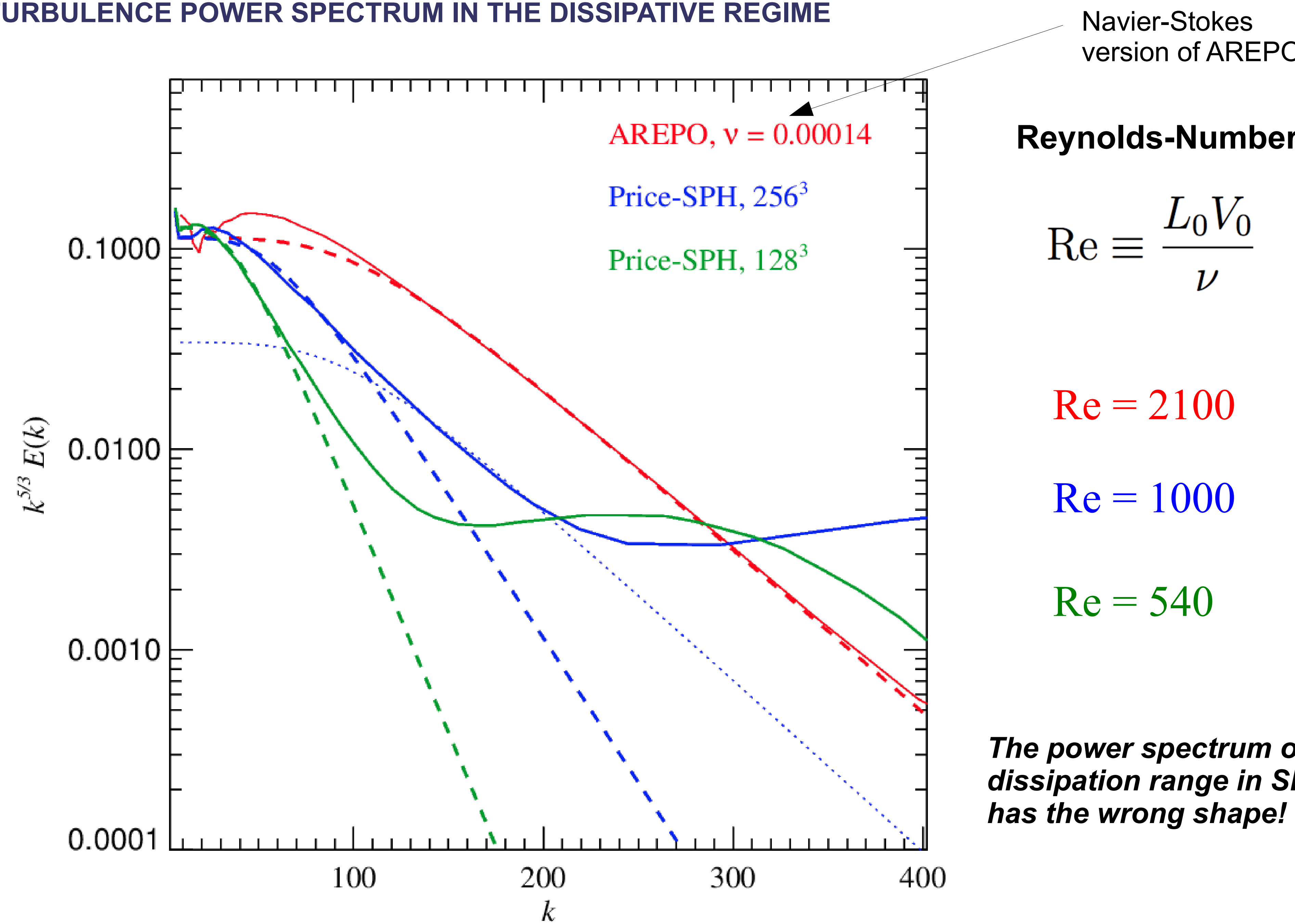
Experiments (and simulations) give a universal function for f_η

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

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

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

$$\mathcal{R}_e \equiv \frac{VL}{\nu} \qquad \frac{\eta}{L_0} \sim \mathrm{Re}^{-\frac{3}{4}}$$

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

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

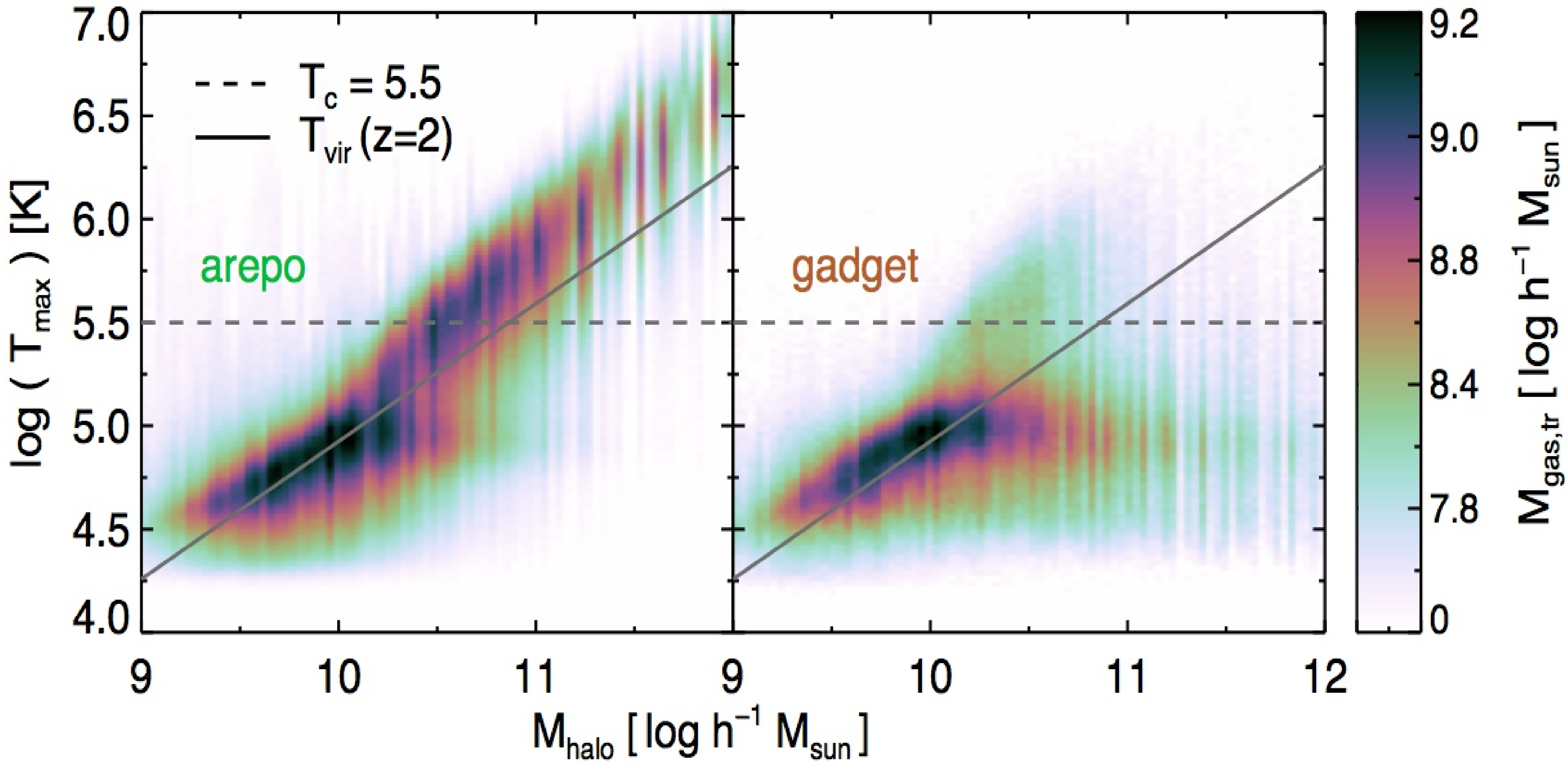
CPU ~ Re^4

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

CPU ~ Re^3

There are marked differences in cold vs. hot accretion for massive galaxies

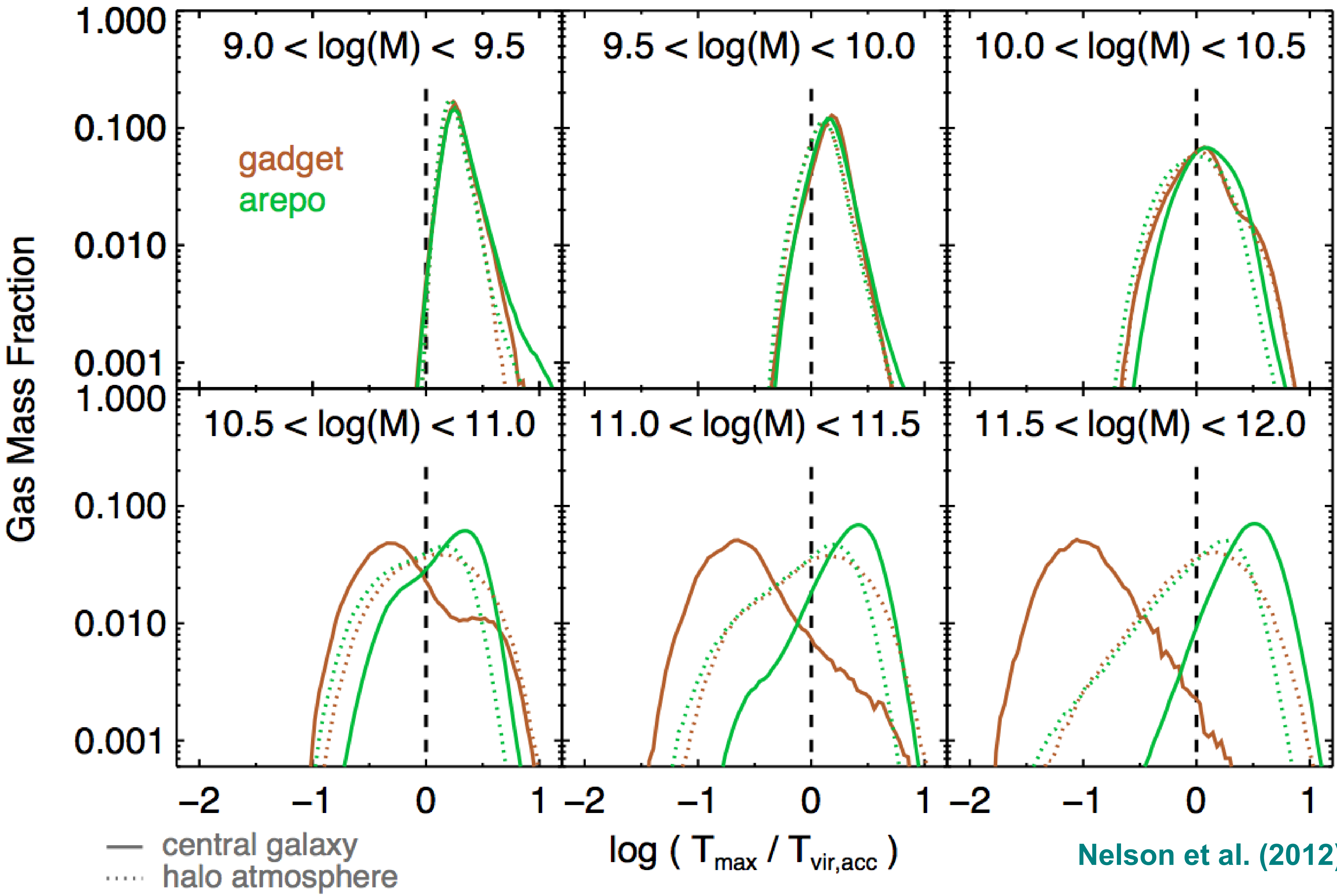
PAST MAXIMUM TEMPERATURE OF GAS ACCRETED ONTO CENTRAL GALAXIES



Nelson et al. (2012)

There are marked differences in cold vs. hot accretion for massive galaxies

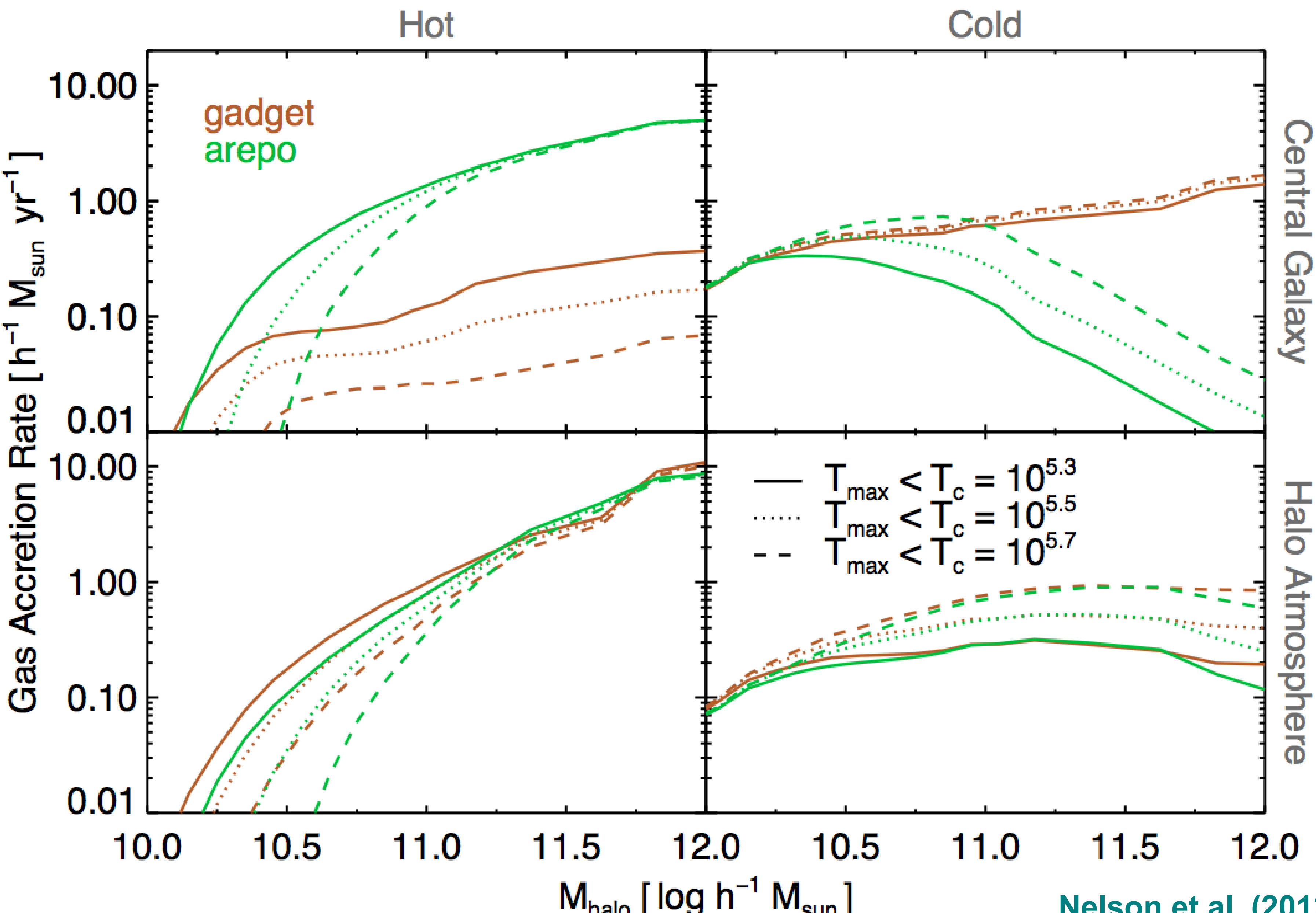
DISTRIBUTION OF PAST MAXIMUM TEMPERATURE OF ACCRETED GAS AT Z = 2



Nelson et al. (2012)

The relative importance of "hot" and "cold" modes of accretion are different for massive halos

ACCRETION RATES OF HOT AND COLD GAS AS A FUNCTION OF HALO MASS AT Z = 2

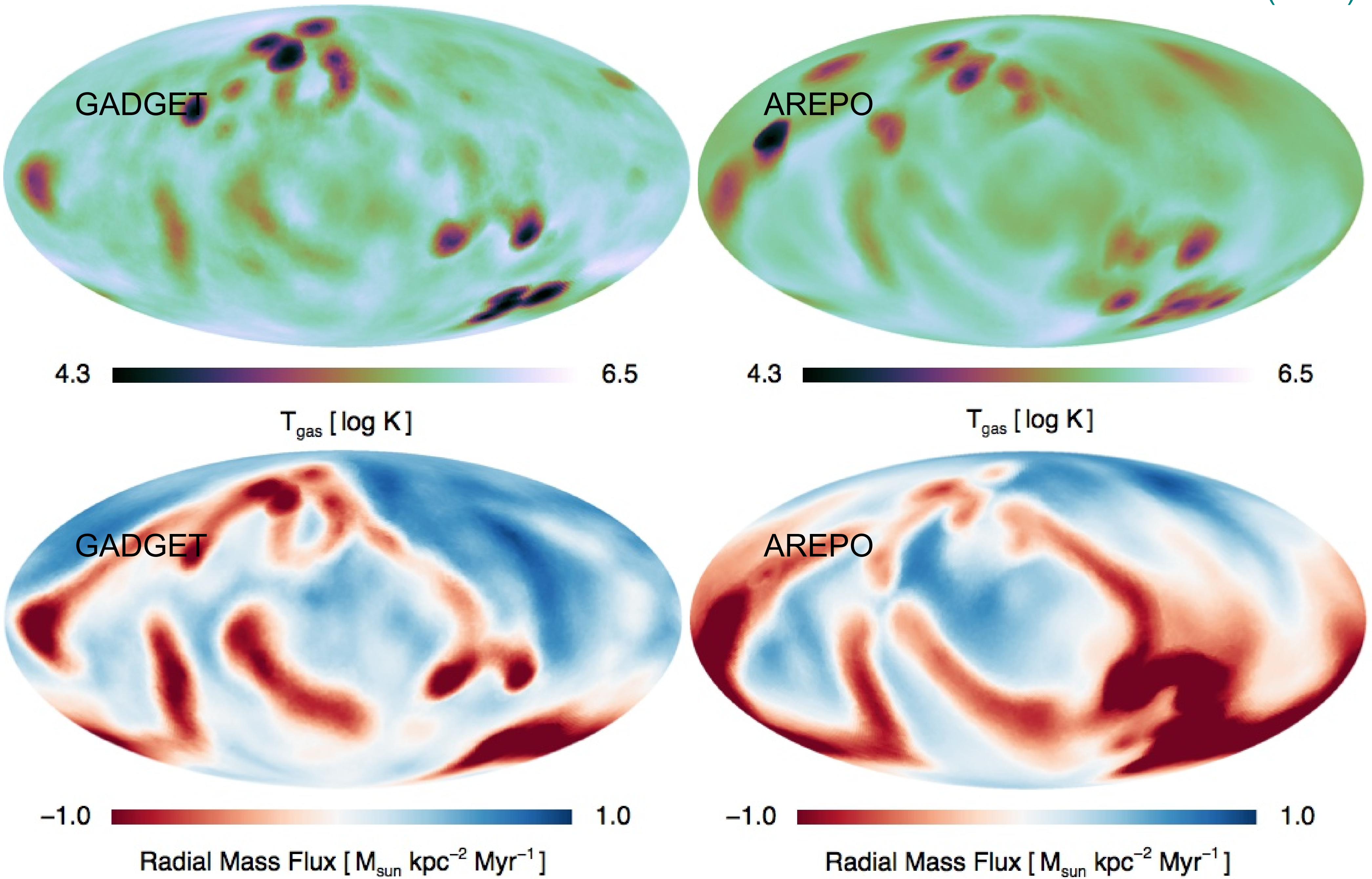


Nelson et al. (2012)

At **the virial radius**, only moderate differences in the gas flow are seen

ALL-SKY MAPS OF GAS PROPERTIES AROUND A TYPICAL log(M)=11.5 HALO AT Z=2

Nelson et al. (2012)



At **half the virial radius**, pronounced differences in the gas flow are apparent

ALL-SKY MAPS OF GAS PROPERTIES AROUND A TYPICAL HALO

Nelson et al. (2012)