Numerical Simulations of Dark Matter

Mark Vogelsberger (Harvard / CfA / M.I.T.)

ISAPP 2013, 29 July - 06 August 2013

Content

(1) **Overview**

(2) **Dark Matter Dynamics**

(3) **Force Calculation**

(4) **Time Integration**

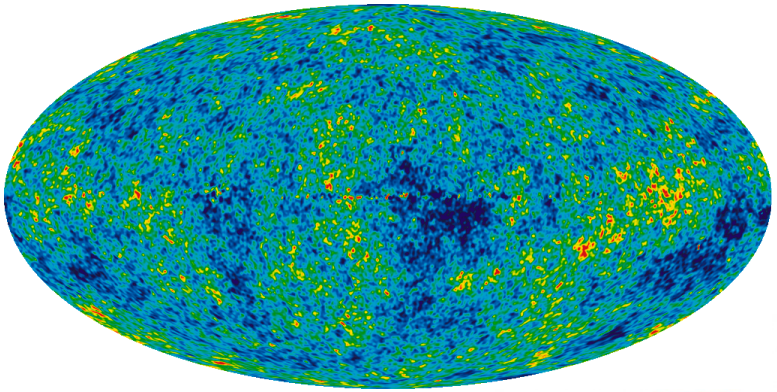
(5) **Recent Developments**

(6) **Hands-on Session**

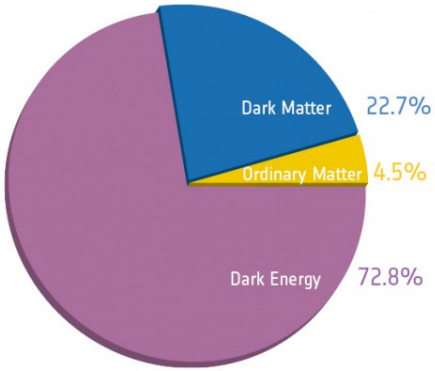
Overview

- Why dark matter simulations?
- What can we learn from them?
- What is the state-of-the-art?

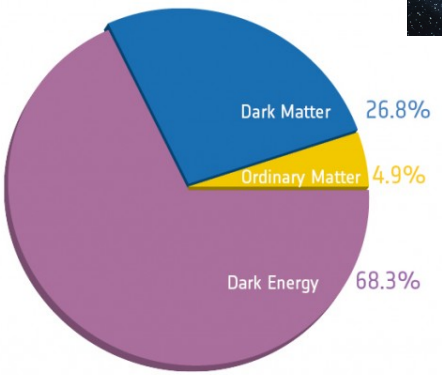
Goal



goal



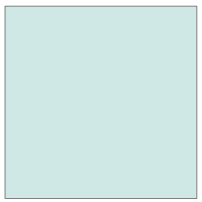
Before Planck



After Planck

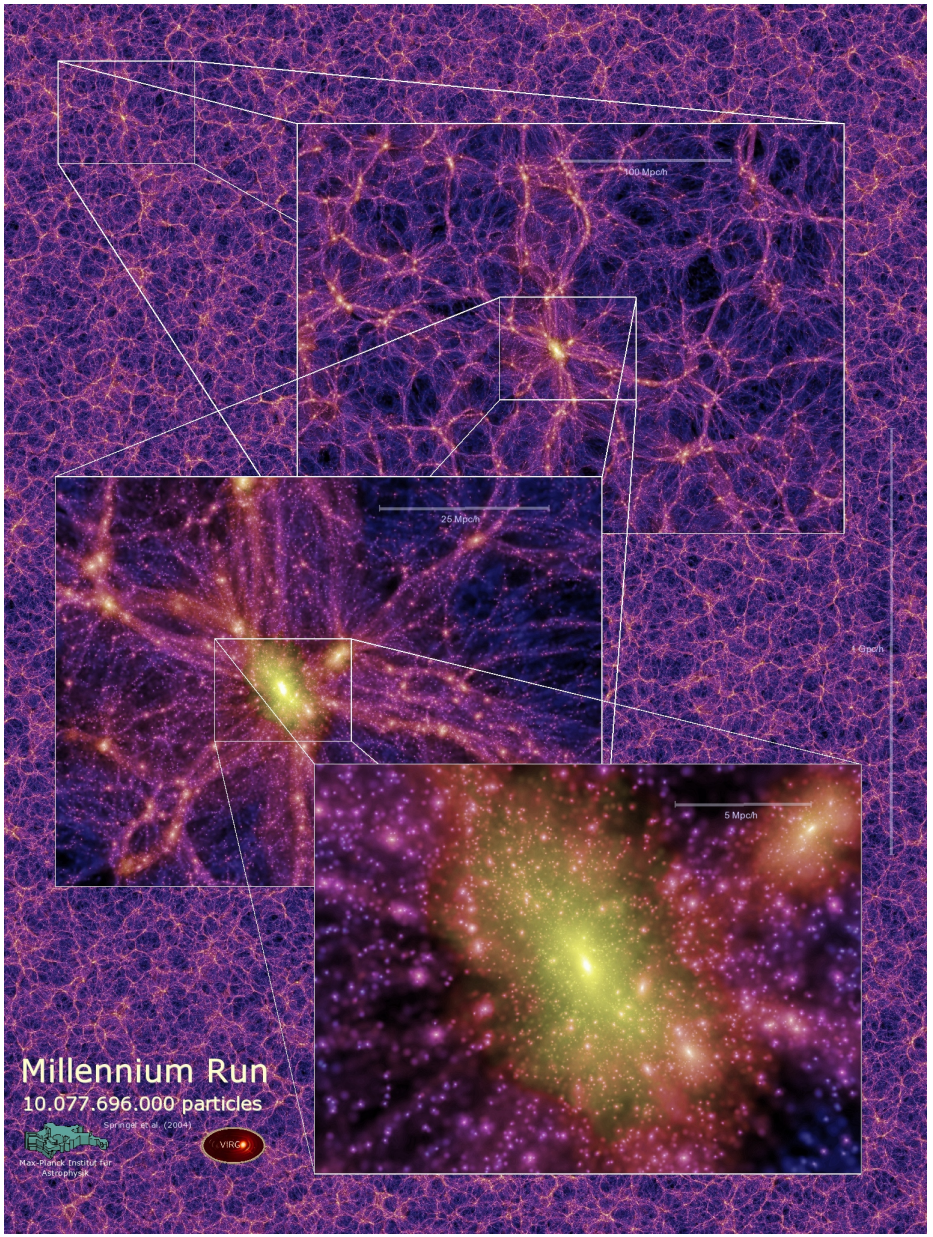
linear theory

e.g. Zel'dovich approximation



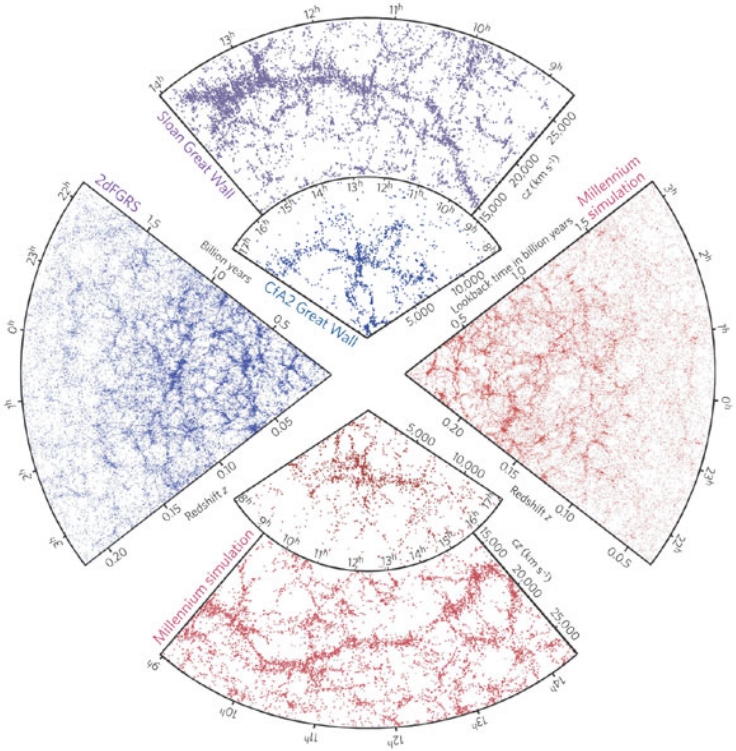
non-linear evolution

Dark Matter Simulations: State-of-the-Art



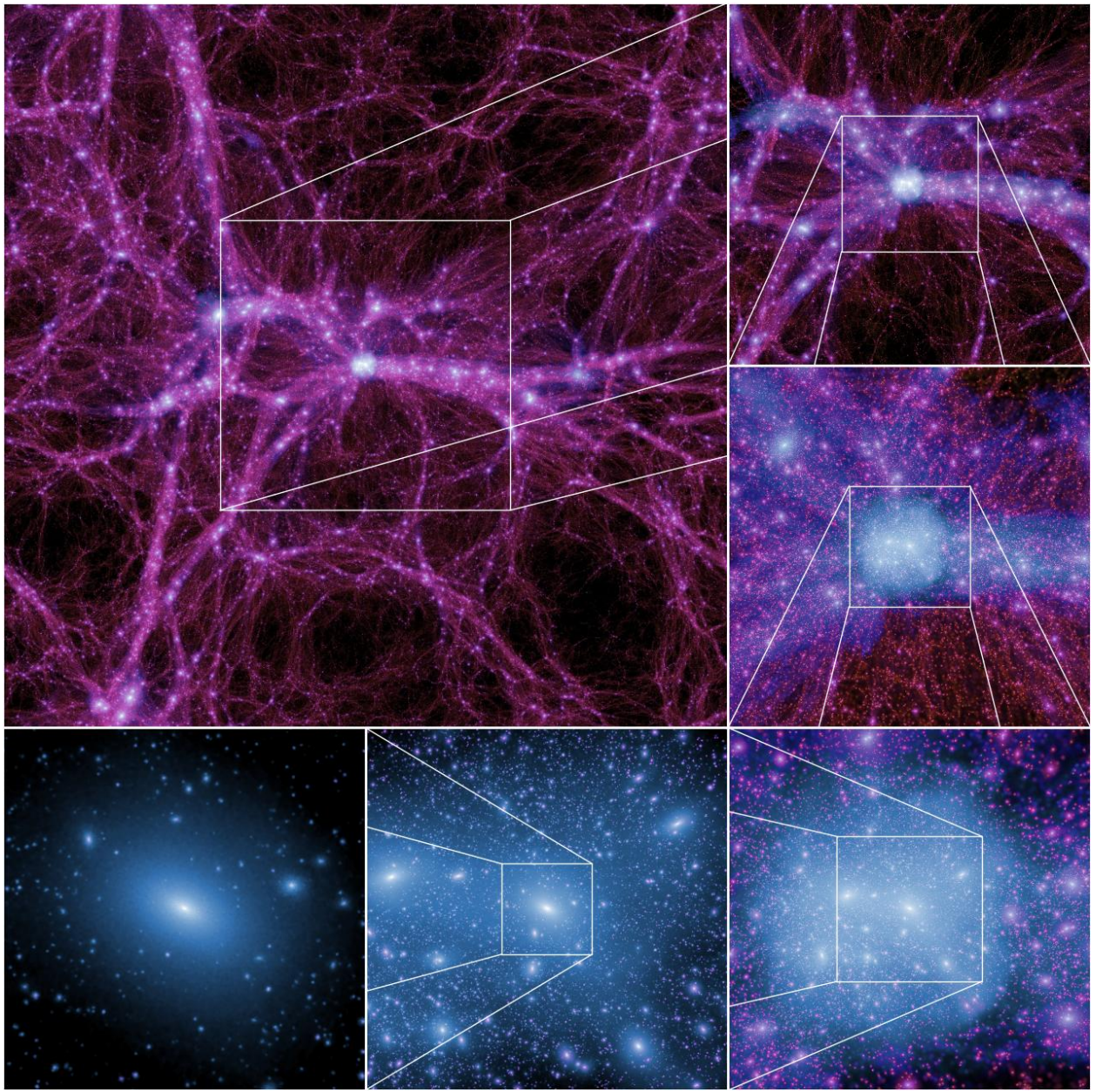
Springel+ (2005)

Millennium I



$$V = (500\,\mathrm{Mpc}/h)^3$$
$$N = 2160^3$$
$$\epsilon = 5\,\mathrm{kpc}/h$$

Dark Matter Simulations: State-of-the-Art

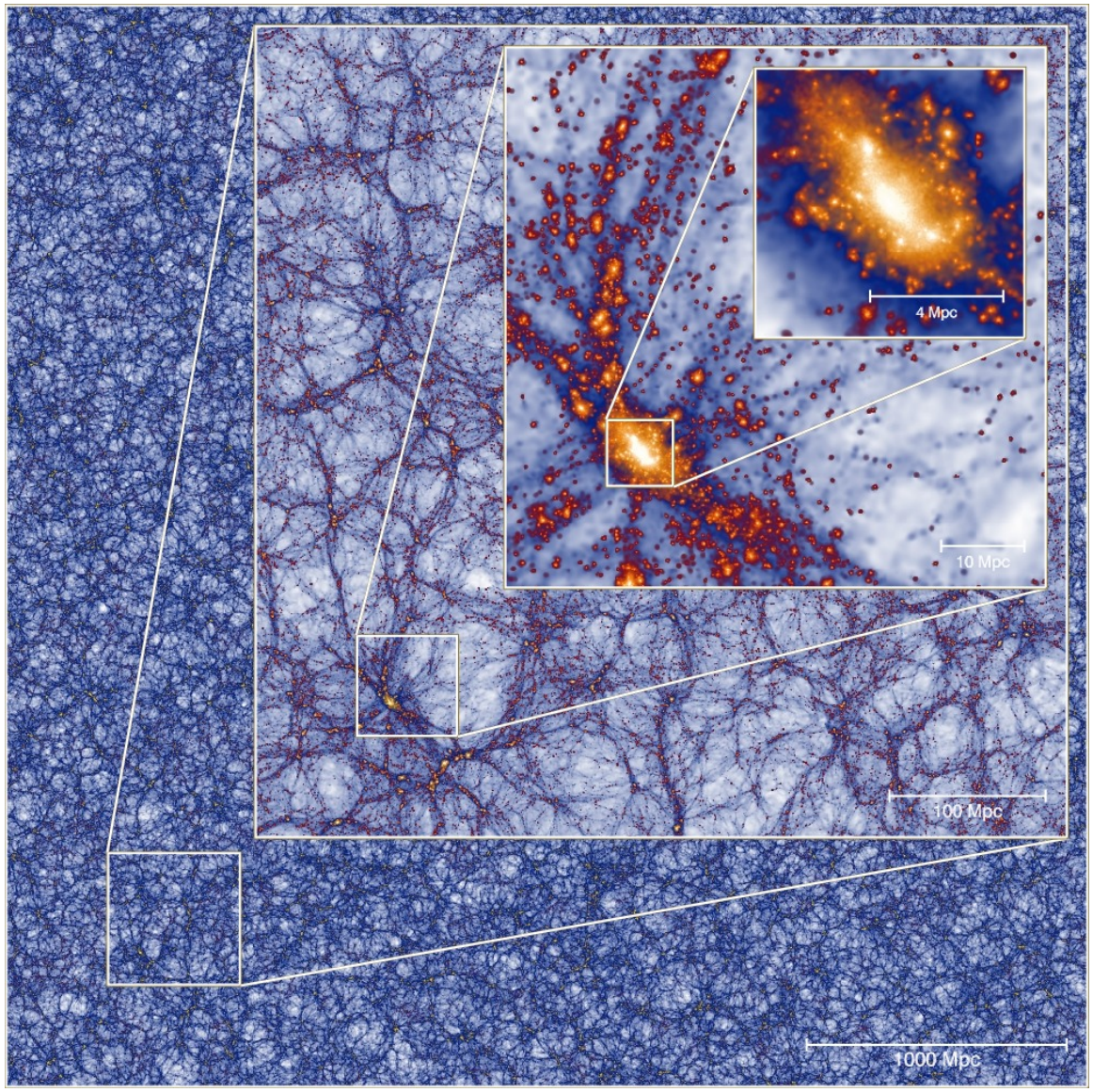


Millennium II

$$V = (100\,\mathrm{Mpc}/h)^3$$
$$N = 2160^3$$
$$\epsilon = 1\,\mathrm{kpc}/h$$

Boylan-Kolchin+ (2009)

Dark Matter Simulations: State-of-the-Art



Millennium XXL

→ 12288 cores at JuRoPa

→ 30 TB RAM

→ 303 billion particles

$$V = (3000\,\mathrm{Mpc}/h)^3$$
$$N = 6720^3$$
$$\epsilon = 10\,\mathrm{kpc}/h$$

Angulo+ (2012)

Dark Matter Simulations: State-of-the-Art

Aquarius

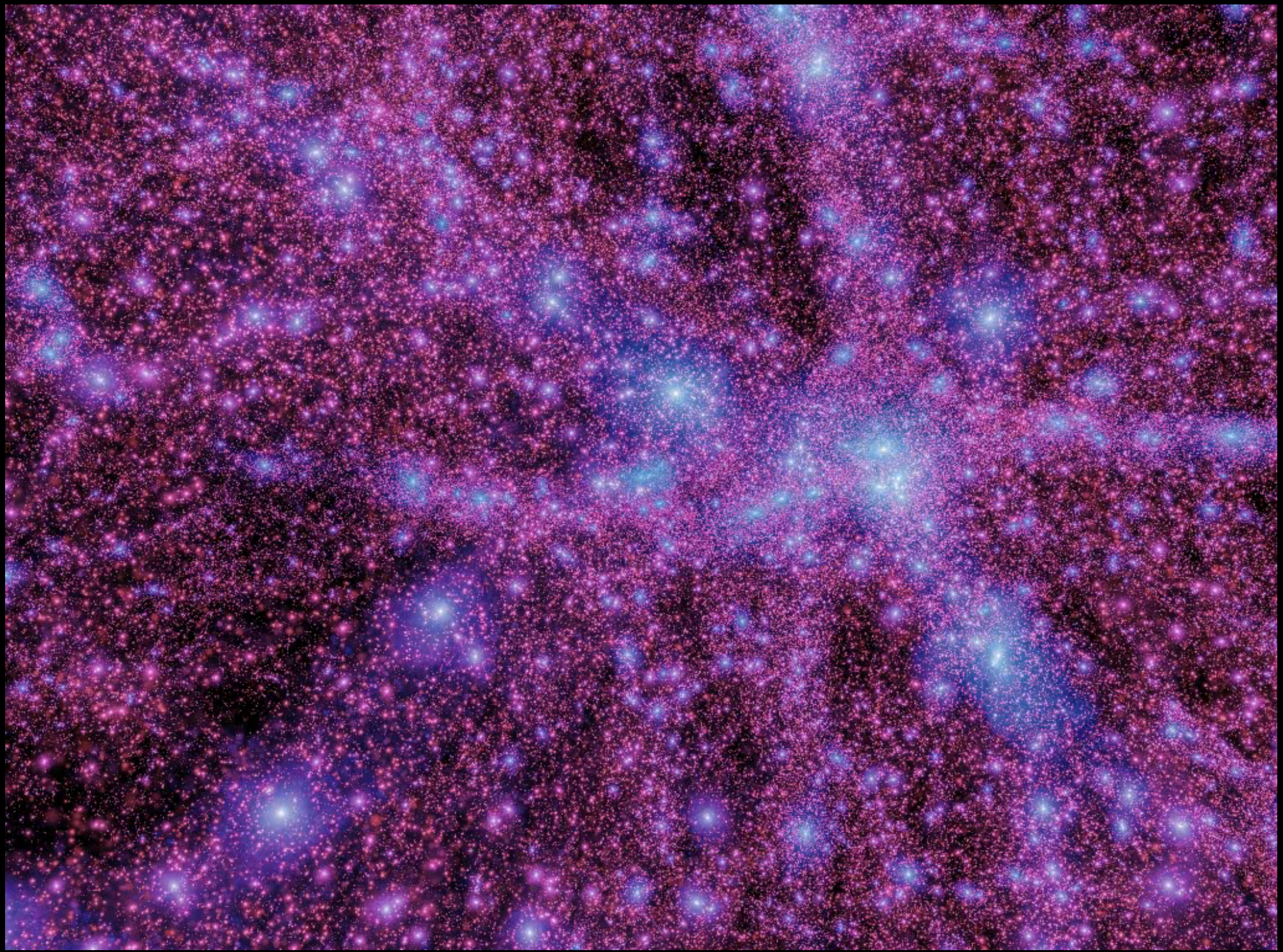
$V \sim (2.5\,\mathrm{Mpc}/h)^3$

$N = 4.5 \times 10^9$

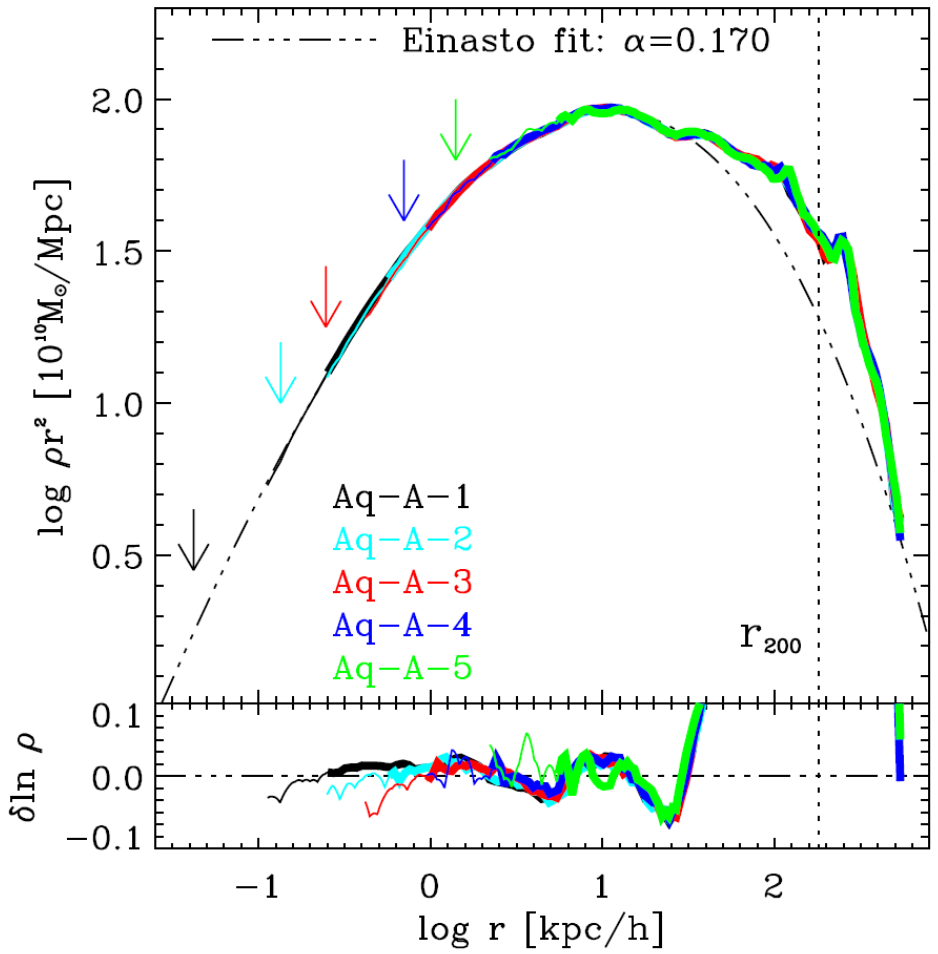
$\epsilon = 15\,\mathrm{pc}/h$

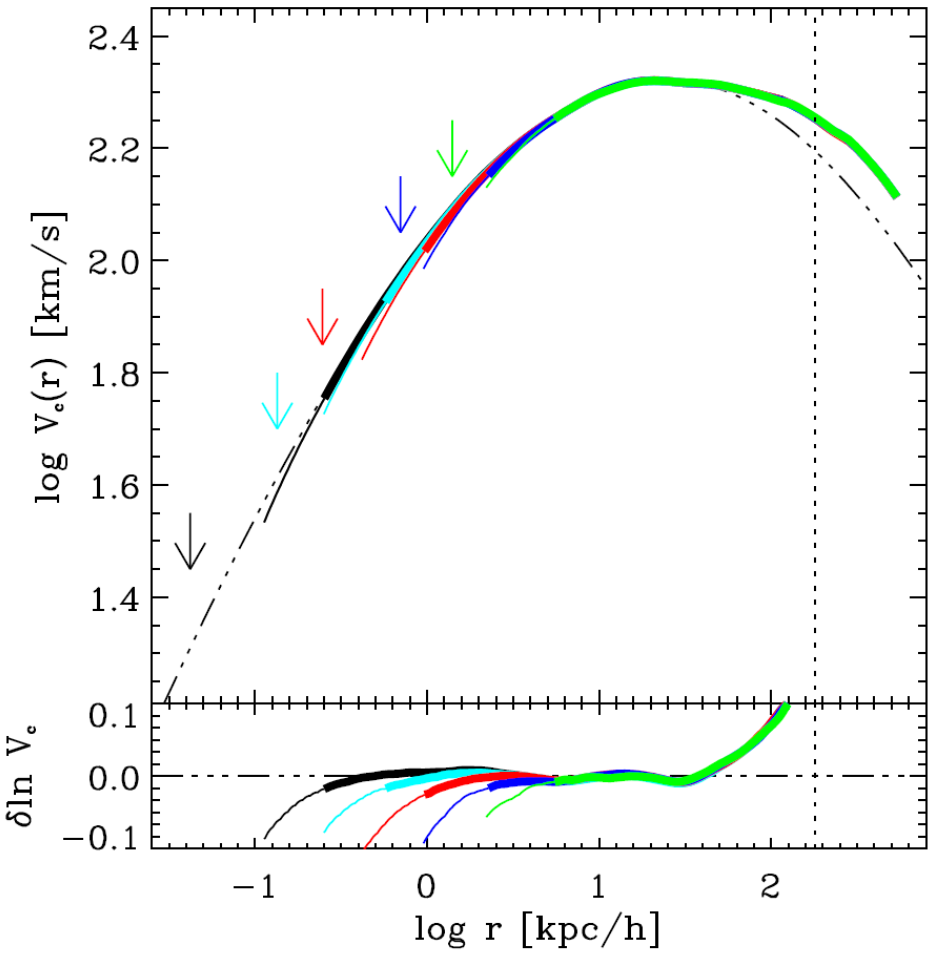
Springel+ (2008)


z = 48.4
T = 0.05 Gyr
500 kpc



Dark Matter Simulations: Some Results

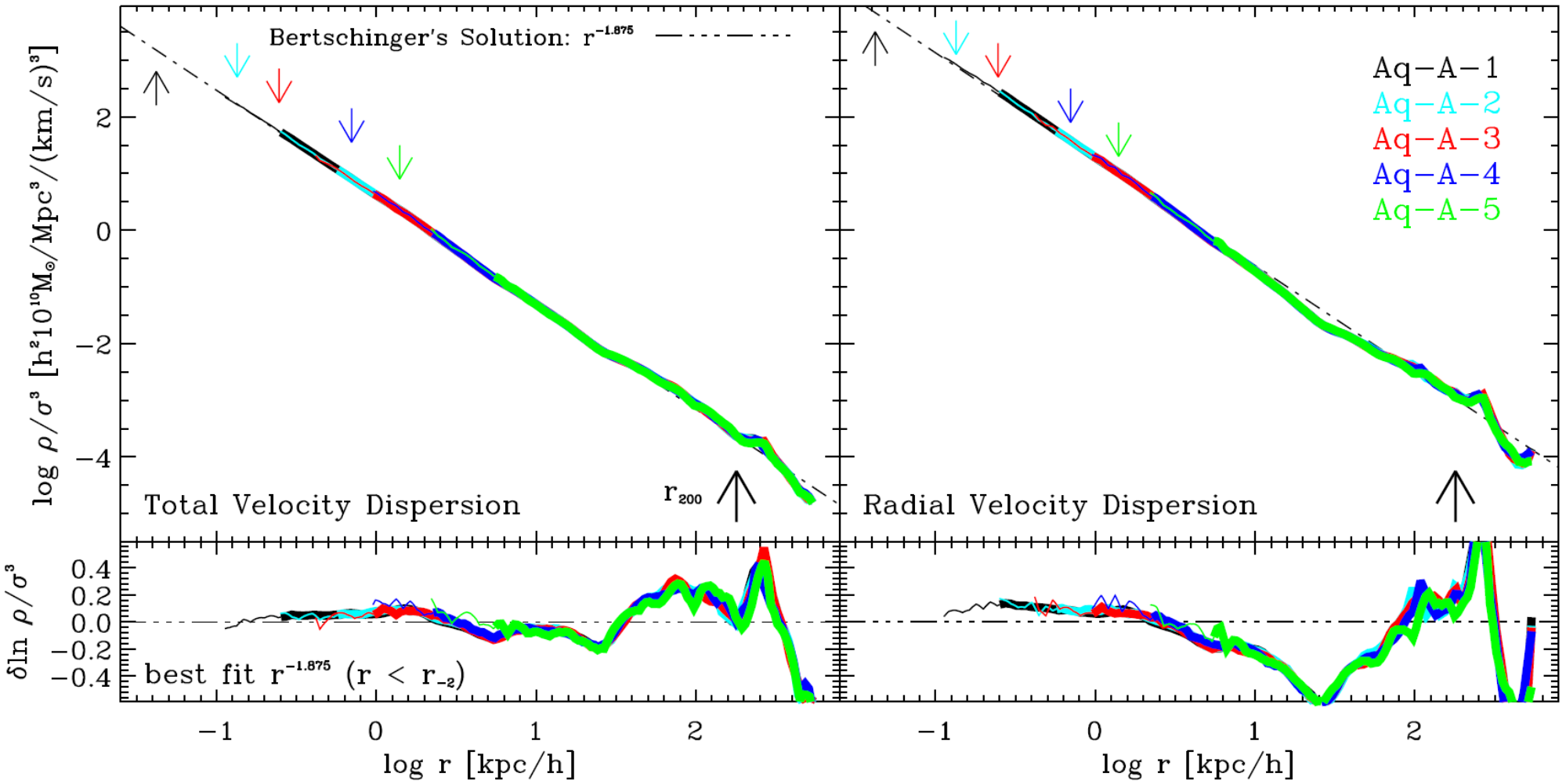




Navarro+ (2010)

DM halo density profiles

Dark Matter Simulations: Some Results



Navarro+ (2010)

pseudo phase-space density

Dark Matter Simulations: DM Detection

Direct Detection:

$$\frac{\mathrm{d}R}{\mathrm{d}E} = \frac{\rho_0}{m_{\mathrm{N}} m_{\mathrm{DM}}} \sigma(E) g(v_{\mathrm{min}})$$

differential scattering rate per unit detector mass

$$g(v_{\mathrm{min}}) = \int_{v_{\mathrm{min}}} \mathrm{d}v \, \frac{f(v)}{v}$$

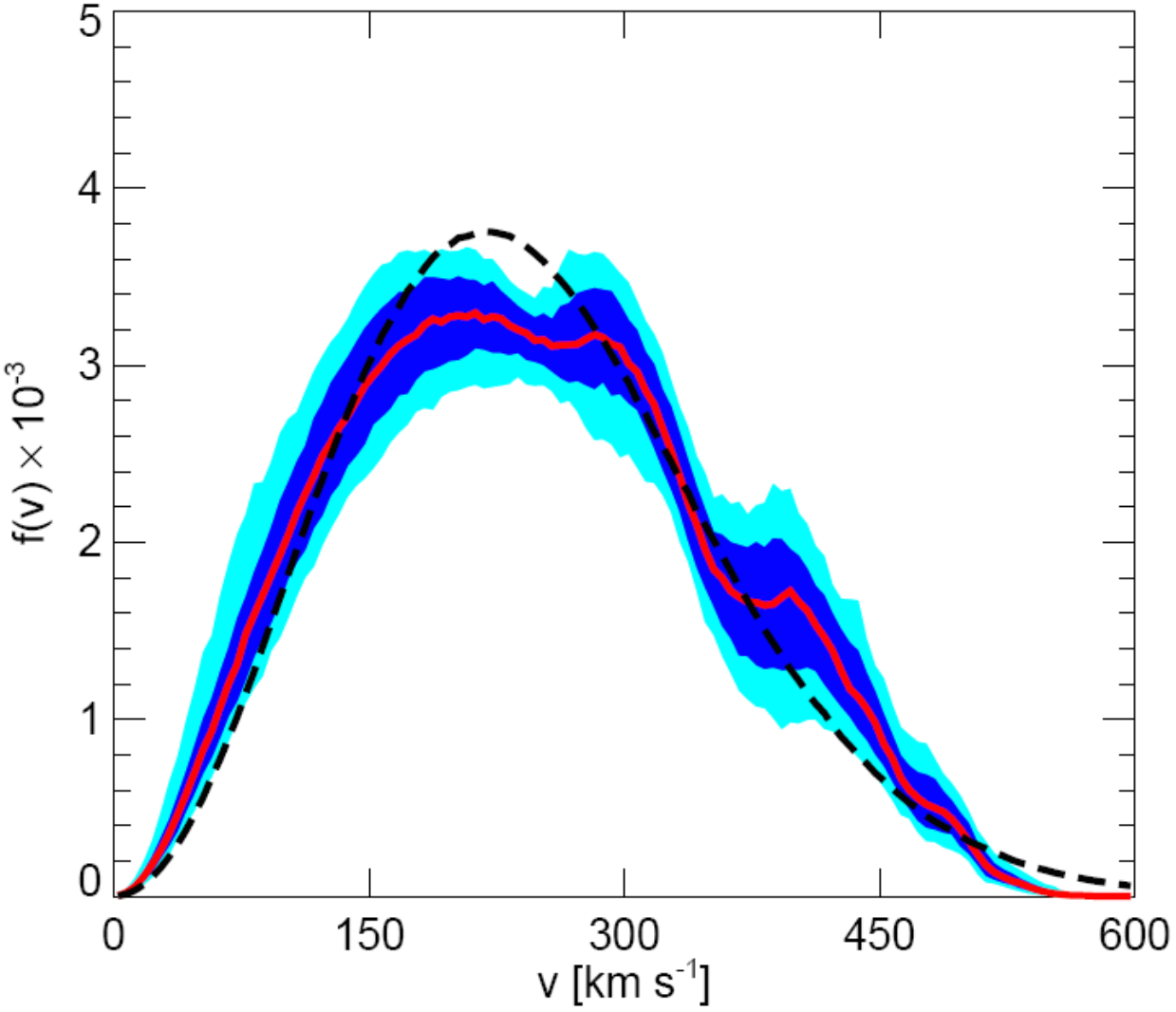
detection signals depend on DM phase-space structure

Indirect Detection:

$$n = \frac{\langle \sigma v \rangle}{2 m_{\mathrm{DM}}^2} \int_V \mathrm{d}^3 x \rho_{\mathrm{DM}}^2(\mathbf{x})$$

Annihilation rate in volume V

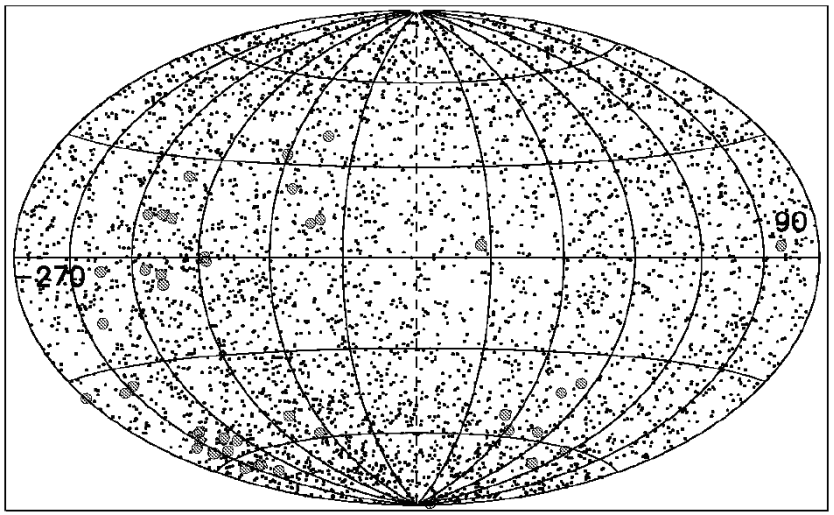
Dark Matter Simulations: DM Detection



Vogelsberger+ (2008)

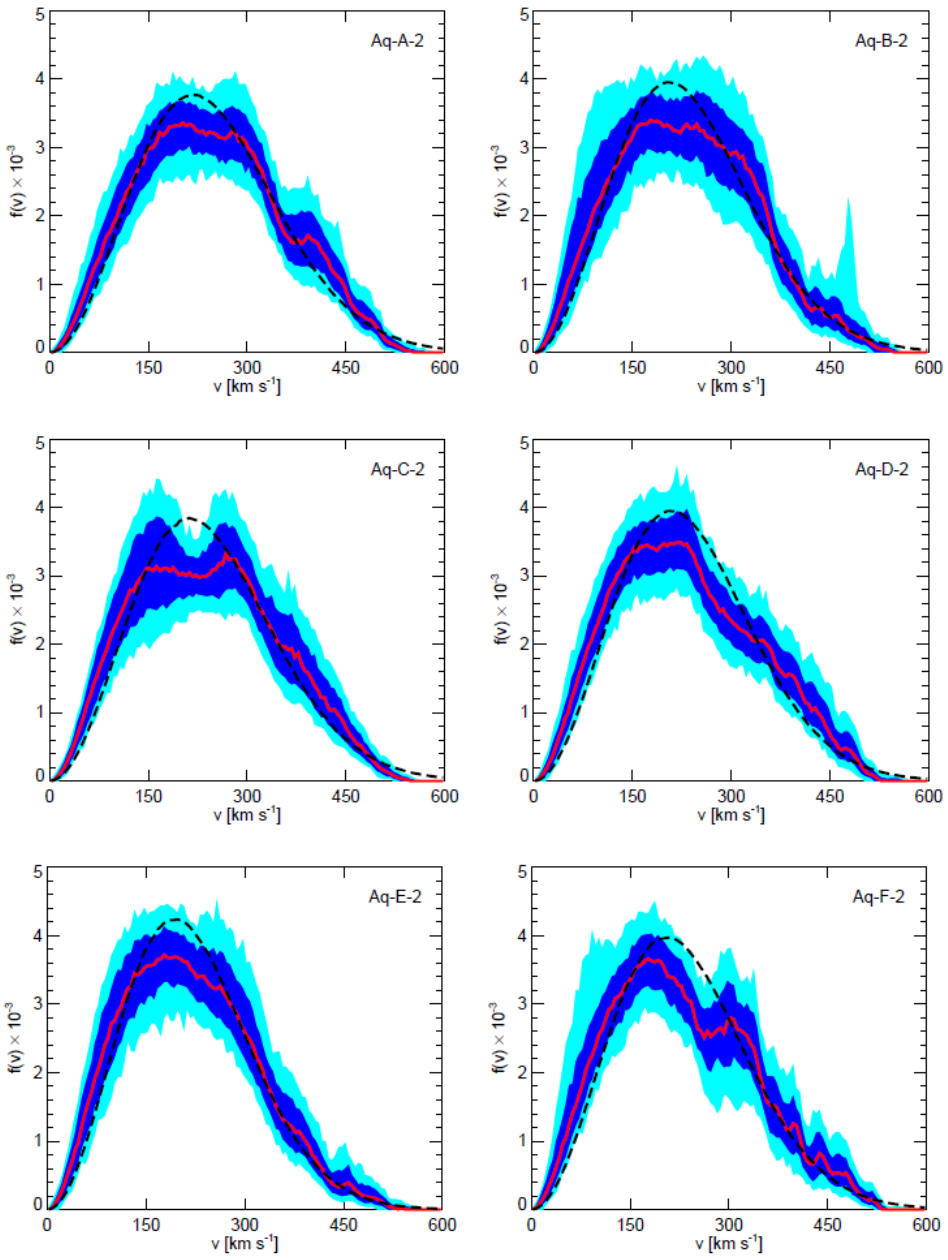
non-Maxwellian
velocity space structures

high velocity particles show
anisotropic distribution
(→ directional detection)



Helmi+ (2002)

Dark Matter Simulations: DM Detection



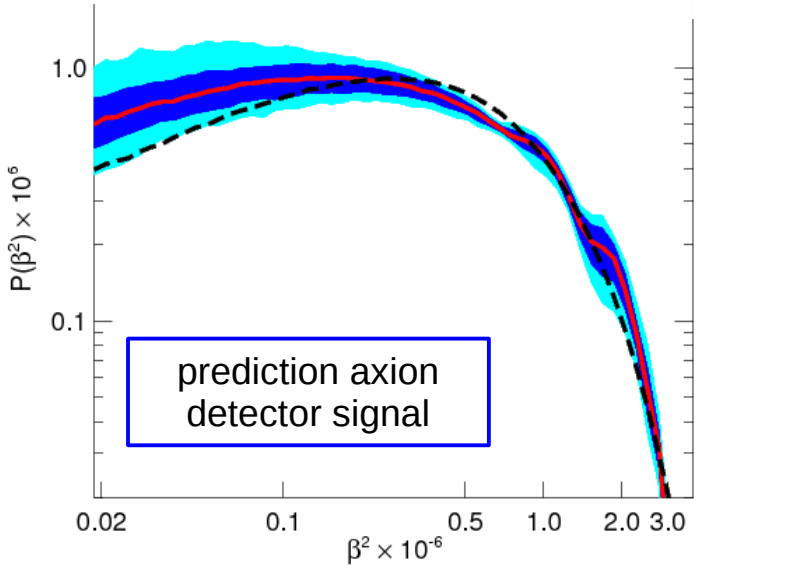
sample of six haloes

all show velocity structure

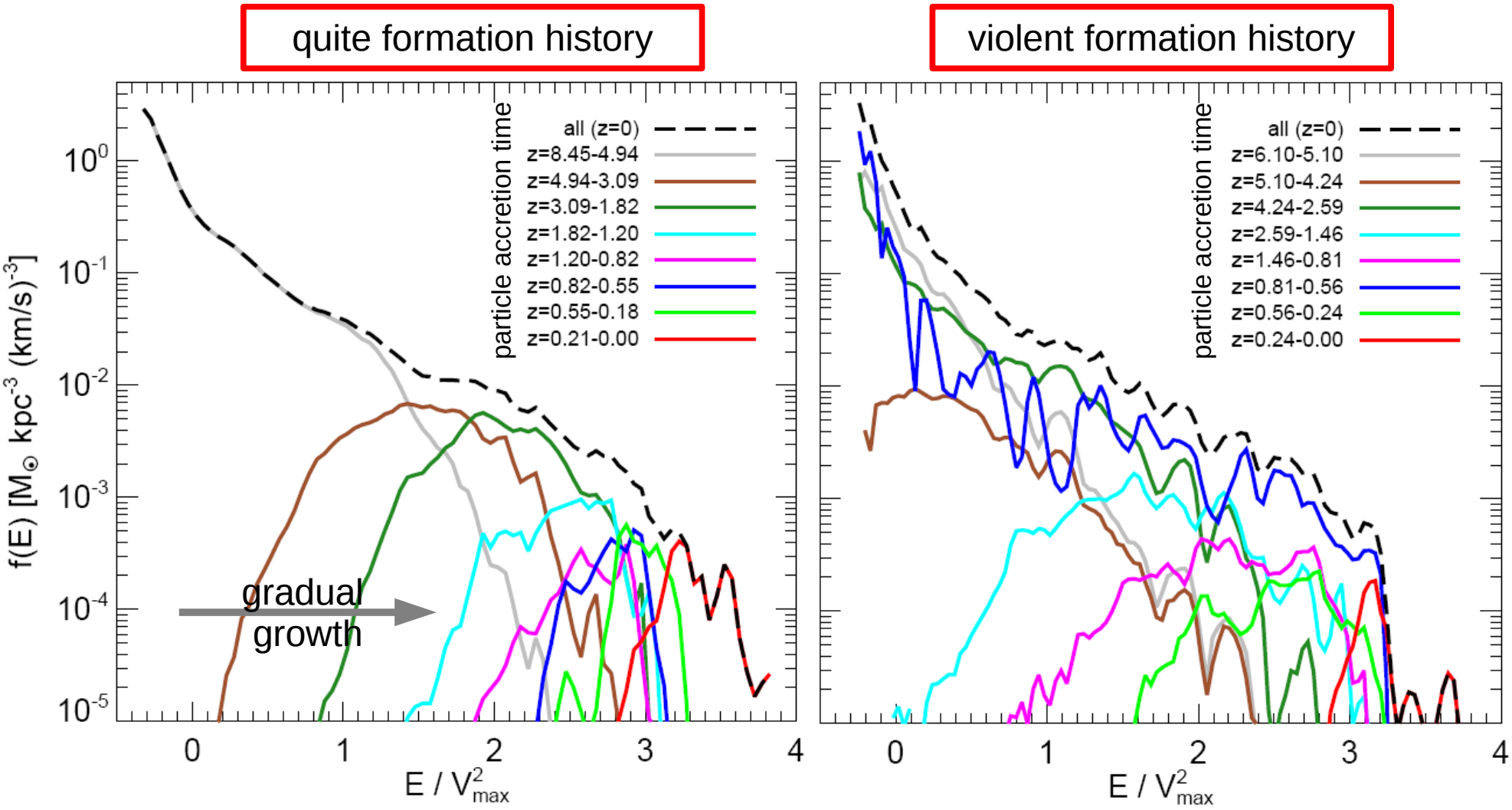
imprints of formation history

dark matter astronomy

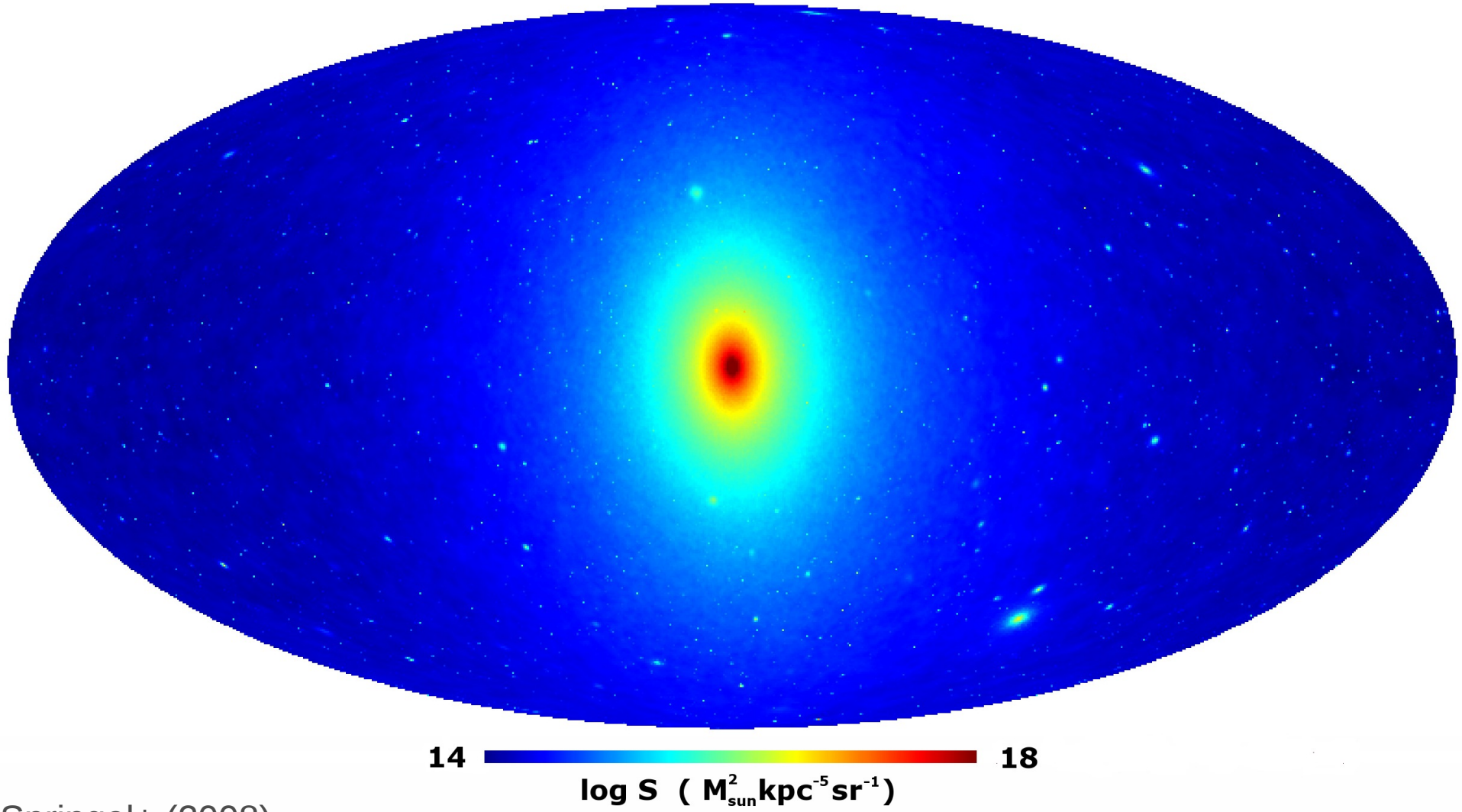
$$\nu_a = 241.8 \left(\frac{m_a}{1\mu\mathrm{eV}/c^2}\right)\left(1+\frac{1}{2}\beta^2\right)\ \mathrm{MHz}$$



Dark Matter Simulations: DM phase-space



Dark Matter Simulations: Annihilation



Springel+ (2008)

Dark Matter Dynamics

- Which equations govern the evolution of DM?
- What is the basic approach to solve them?
- What is special about cold DM (CDM)?

Goal: Predict Distribution of DM

Dark Matter as a Collisionless Fluid

→ a Milky Way like halo has of the order of 10^{67} individual DM particles

→ they do not scatter locally / move smoothly under their collective grav. potential

→ describe the system of DM particles in terms of a distribution function:

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

→ DM dynamics can then be described by the Poisson-Vlasov equations:

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

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

Solving the Poisson-Vlasov Equation

Task: Solve the Poisson-Vlasov equation → DM distribution known

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

$$\rho(\mathbf{x}, t) = \int \mathrm{d}\mathbf{v}\, f(\mathbf{x}, \mathbf{v}, t) \longleftarrow$$

real space density derived from distribution function

Task: Poisson-Vlasov equation is a ordinary PDE → we know how to solve those

Solving the Poisson-Vlasov Equation

→ 7 independent variables (3 coordinates, 3 velocities, time)

→ put a fine grid on top of computational domain

→ **Example:** halo with virial radius ~ 200 kpc/h; velocity dispersion ~ 200 km/s

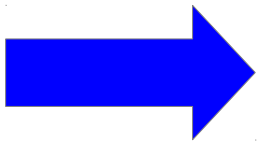
→ resolve small scales in velocity (1 km/s) and space (1 kpc)

→ we need a grid with 200^3 x 200^3 = 64 trillion grid cells

→ assume we store 10 floats = (4 x 10 = 40 Bytes) per grid cell

→ 64 x 10^{12} x 40 Bytes ~ 2.3 PetaByte

→ for a two times finer grid ~ 150 PetaByte



requires a computer that can
hold 150 PetaByte = 150,000 TB in RAM
(15,000,000 x memory of ordinary desktop machine)

HPC where do we stand?

			cores		TFlops/s	
1	National University of Defense Technology China	**Tianhe-2 (MilkyWay-2)** - TH-IVB-FEP Cluster, Intel Xeon E5-2692 12C 2.200GHz, TH Express-2, Intel Xeon Phi 31S1P NUDT	3120000	33862.7	54902.4	17808
2	DOE/SC/Oak Ridge National Laboratory United States	**Titan** - Cray XK7 , Opteron 6274 16C 2.200GHz, Cray Gemini interconnect, NVIDIA K20x Cray Inc.	560640	17590.0	27112.5	8209
3	DOE/NNSA/LLNL United States	**Sequoia** - BlueGene/Q, Power BQC 16C 1.60 GHz, Custom IBM	1572864	17173.2	20132.7	7890
4	RIKEN Advanced Institute for Computational Science (AICS) Japan	K computer, SPARC64 VIIIfx 2.0GHz, Tofu interconnect Fujitsu	705024	10510.0	11280.4	12660
5	DOE/SC/Argonne National Laboratory United States	**Mira** - BlueGene/Q, Power BQC 16C 1.60GHz, Custom IBM	786432	8586.6	10066.3	3945
6	Texas Advanced Computing Center/Univ. of Texas United States	**Stampede** - PowerEdge C8220, Xeon E5-2680 8C 2.700GHz, Infiniband FDR, Intel Xeon Phi SE10P Dell	462462	5168.1	8520.1	4510
7	Forschungszentrum Juelich (FZJ) Germany	**JUQUEEN** - BlueGene/Q, Power BQC 16C 1.600GHz, Custom Interconnect IBM	458752	5008.9	5872.0	2301
8	DOE/NNSA/LLNL United States	**Vulcan** - BlueGene/Q, Power BQC 16C 1.600GHz, Custom Interconnect IBM	393216	4293.3	5033.2	1972
9	Leibniz Rechenzentrum Germany	**SuperMUC** - iDataPlex DX360M4, Xeon E5-2680 8C 2.70GHz, Infiniband FDR IBM	147456	2897.0	3185.1	3423
10	National Supercomputing Center in Tianjin China	**Tianhe-1A** - NUDT YH MPP, Xeon X5670 6C 2.93 GHz, NVIDIA 2050 NUDT	186368	2566.0	4701.0	4040

Solving the Poisson-Vlasov Equation: Brute-Force Approach

DIRECT INTEGRATION OF THE COLLISIONLESS BOLTZMANN EQUATION IN SIX-DIMENSIONAL PHASE SPACE: SELF-GRAVITATING SYSTEMS

KOHJI YOSHIKAWA[1], NAOKI YOSHIDA[2,3], AND MASAYUKI UMEMURA[1]
[1] Center for Computational Sciences, University of Tsukuba, 1-1-1 Tennodai, Tsukuba, Ibaraki 305–8577, Japan; kohji@ccs.tsukuba.ac.jp
[2] Department of Physics, The University of Tokyo, Tokyo 113-0033, Japan
[3] Kavli Institute for the Physics and Mathematics of the Universe, The University of Tokyo, Kashiwa, Chiba 277-8583, Japan
Received 2012 June 18; accepted 2012 November 23; published 2012 December 20

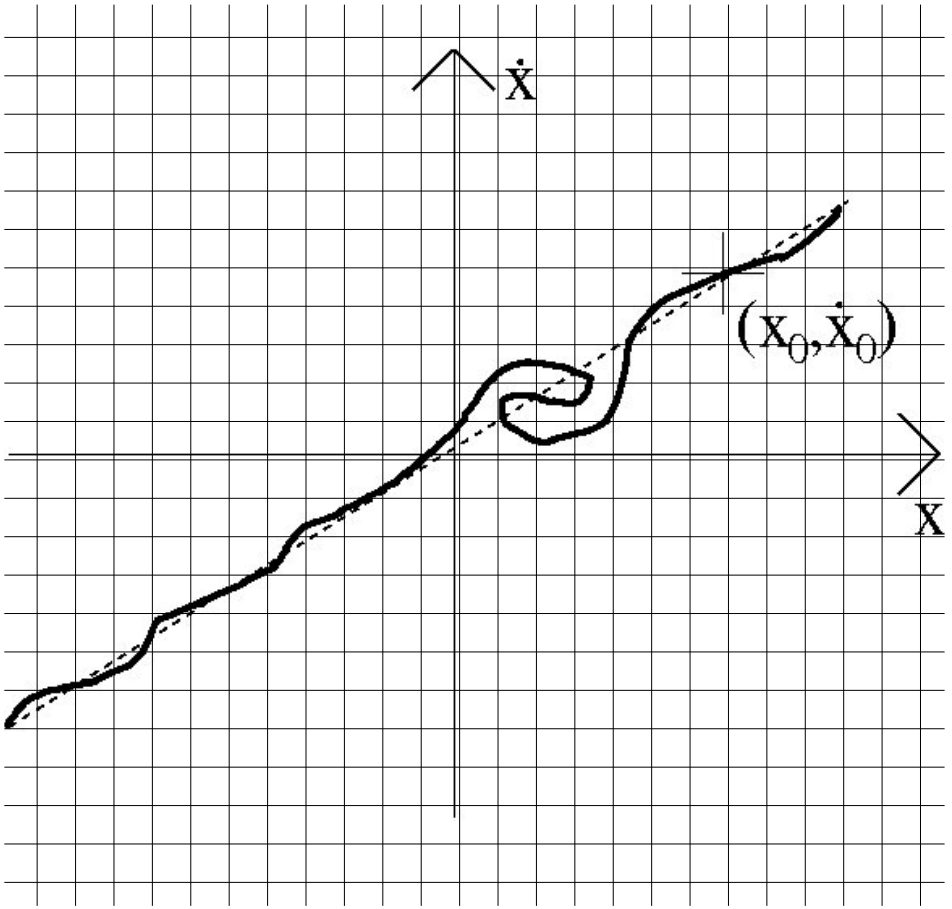
ABSTRACT

We present a scheme for numerical simulations of collisionless self-gravitating systems which directly integrates the Vlasov–Poisson equations in six-dimensional phase space. Using the results from a suite of large-scale numerical simulations, we demonstrate that the present scheme can simulate collisionless self-gravitating systems properly. The integration scheme is based on the positive flux conservation method recently developed in plasma physics. We test the accuracy of our code by performing several test calculations, including the stability of King spheres, the gravitational instability, and the Landau damping. We show that the mass and the energy are accurately conserved for all the test cases we study. The results are in good agreement with linear theory predictions and/or analytic solutions. The distribution function keeps the property of positivity and remains non-oscillatory. The largest simulations are run on 64^6 grids. The computation speed scales well with the number of processors, and thus our code performs efficiently on massively parallel supercomputers.

Solving the Poisson-Vlasov Equation: Beyond Brute-Force

→ use a Monte Carlo approach instead

→ suits the problem very well, because most of phase-space is empty for CDM



CDM occupies
thin 3-dimensional
hyper-surface
in phase-space

a simple grid
covers mostly
empty phase-space

The N-Body Approach

→ discretize in terms of N particles, which sample distribution function

→ follow the equation of motion of these sample particles

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

equation of motion for particle i

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

collective potential

softening

→ N is much smaller than the 'real' N

→ this low N can cause large-angle particle scatterings and formation of bound pairs

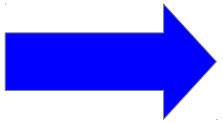
→ add softening to avoid these effects

→ softening to mimic collisionless system evolving according to Poisson-Vlasov

The need for large N: Why?

→ small particle mass to resolve details

→ large volume for representative volume



the more particles
the better

Task: Solve the N-body problem

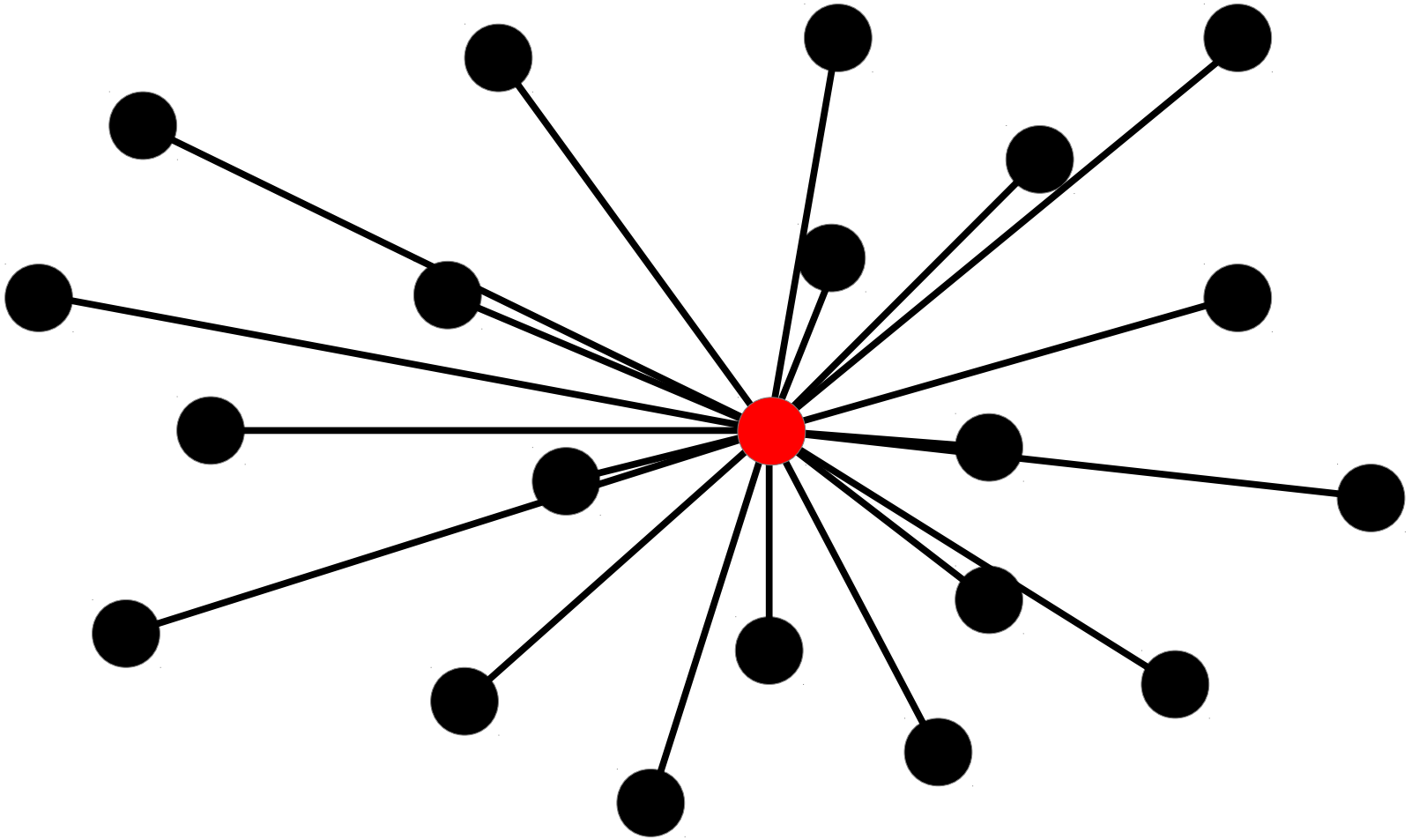
→ how to compute the gravitational forces efficiently and accurately?

→ how to integrate orbital equations?

Force Calculation

- What is the basic problem?
- What are efficient ways to calculate the forces?

Direct Summation



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

N summation terms per particle
N^2 summations per time step

very bad time complexity $O(N^2)$

Particle Mesh Method: Idea

→ Poisson's equation can be solved in real-space by a convolution integral:

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

convolution of density field with Green's function

→ In Fourier-space, the convolution becomes a simple multiplication:

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

→ solve for the potential in Fourier-space and finite differencing to get force field

PM algorithm:
- density assignment (particles to mesh)
- computation of potential field
- computation of force field
- force assignment (mesh to particles)

make use of FFT to efficiently carry out the two Fourier transformations

Particle Mesh Method: Density Assignment

Task: assign particle masses to mesh

→ give particle a shape; i.e. no point mass anymore

→ then to each mesh cell assign the fraction of mass that falls into this cell

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

resulting density on the mesh

Particle Mesh Method: Force Calculation

Task: finite differencing of the potential on the mesh to get forces on the mesh

$$\mathbf{f} = -\nabla\Phi$$ ⟵ replace gradient with finite differencing on mesh

2nd order

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

3rd order

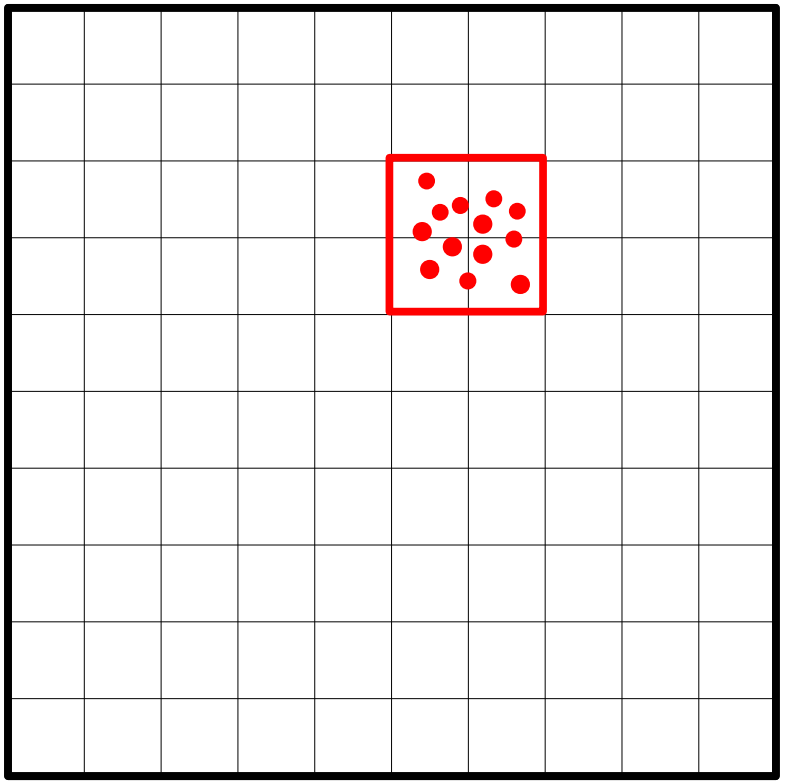
$$f^{(x)}_{i,j,k} = -\frac{4}{3}\frac{\Phi_{i+1,j,k} - \Phi_{i-1,j,k}}{2h} + \frac{1}{3}\frac{\Phi_{i+2,j,k} - \Phi_{i-2,j,k}}{4h}$$

Task: interpolate force field back to particle positions

$$\mathbf{F}(\mathbf{x_i}) = \sum \mathbf{W}(\mathbf{x_i} - \mathbf{x_m})\mathbf{f_m}$$ ⟶ Force calculated for each particle

Particle-Particle PM Method (P^3M): Idea

Task: increase force resolution beyond mesh scale → increase dynamic range



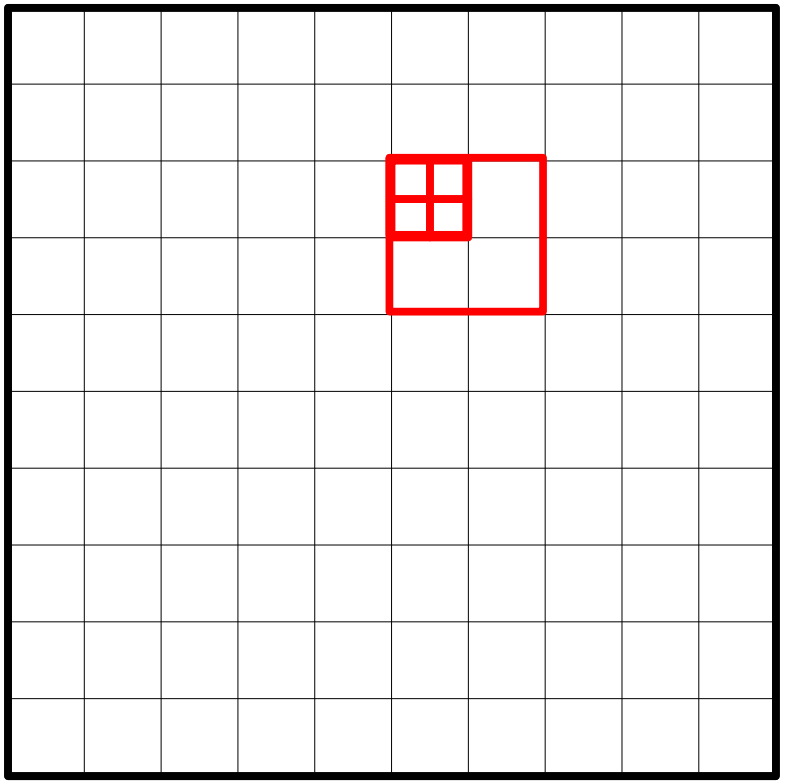
→ do a direct summation at the mesh scale

→ this increases resolution

→ increases dynamic range a lot

→ can get slow if clustering starts ($O(N^2)$)

Mesh-Refinements: Idea



→ additional smaller meshes for small scales

→ use refinement criteria

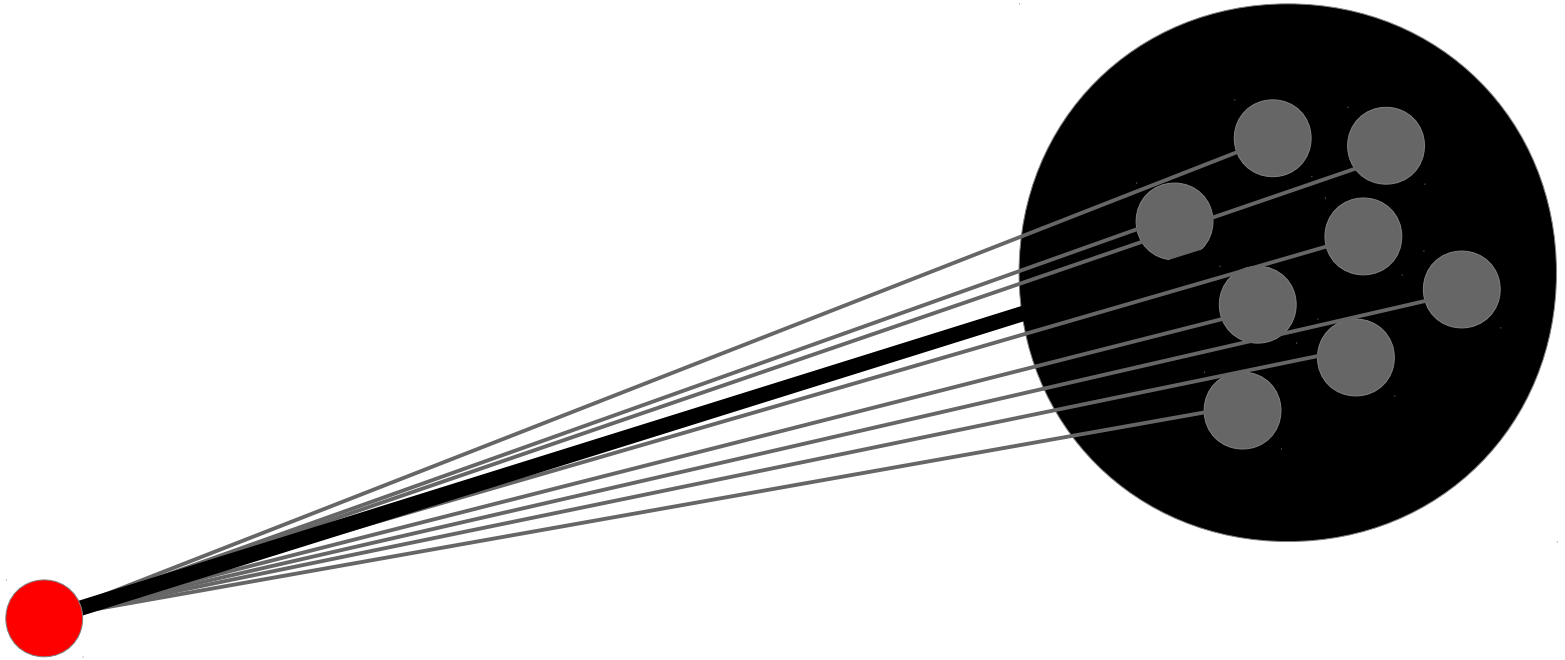
TREE-Algorithm: Idea

Task: efficient force calculation without mesh and FFT

→ pure particle-particle calculation suffers from poor $O(N^2)$ scaling

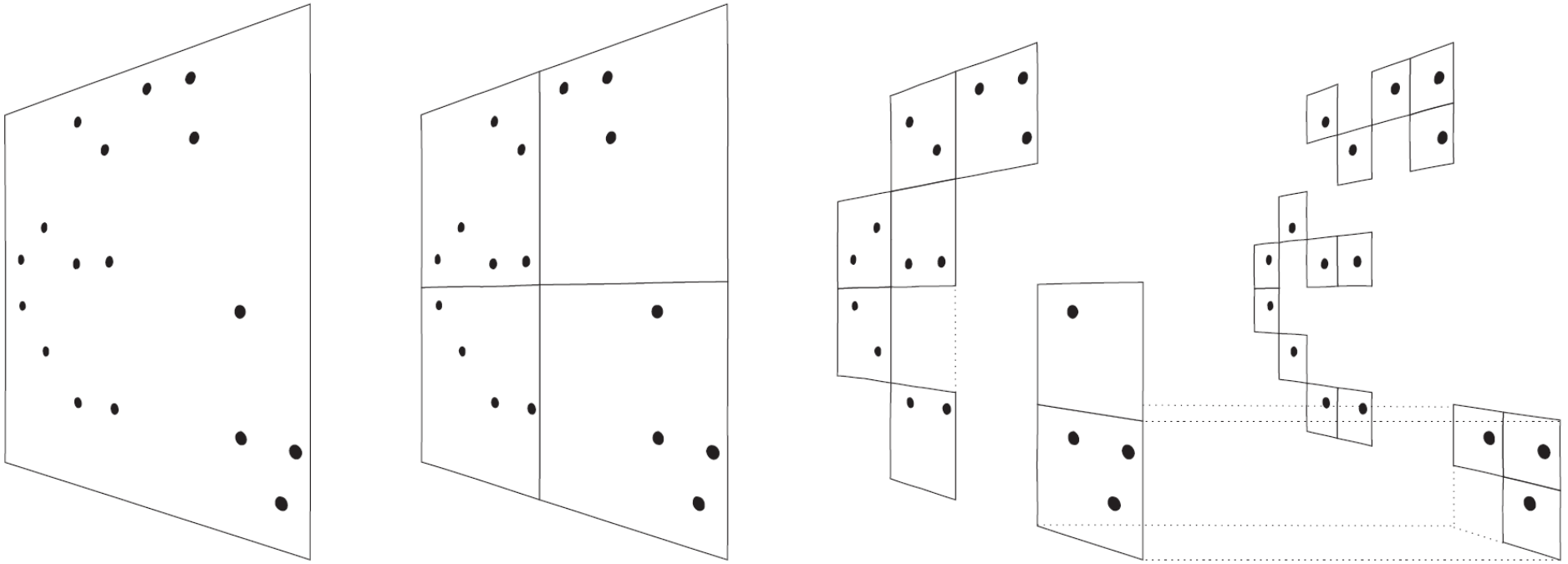
→ i.e. each particle has to calculate a pairwise force with N other particles

→ is there a more efficient way to do this approximately but still accurate enough?



TREE-Algorithm: Oct-Tree

Task: push time complexity to O(N log(N))



[Springel]

→ put particle in an Oct-Tree structure

→ walk the tree for force calculation

→ open nodes based on 'distance' to target particle (BH criterion)

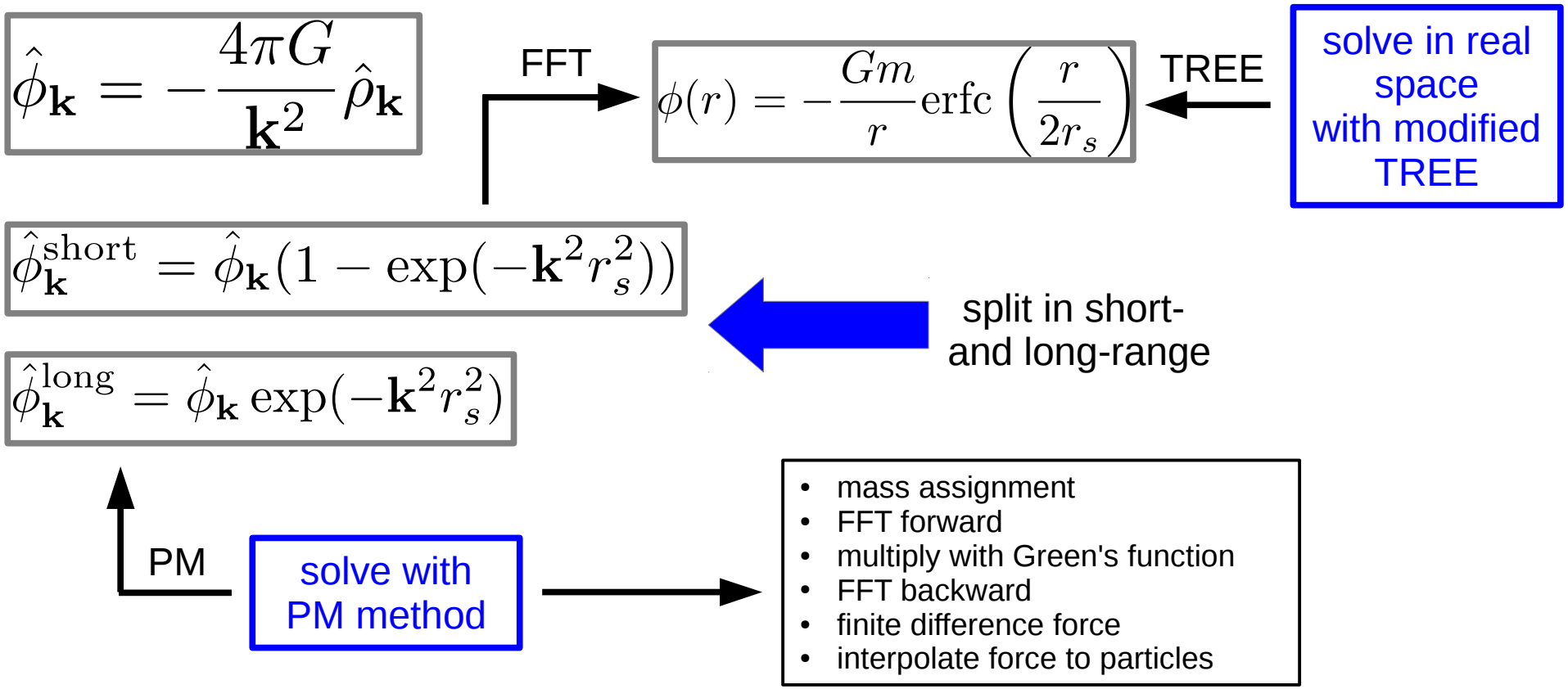
TREE-PM: Idea

Task: combine the advantages of tree algorithms and mesh-based schemes

→ split the potential of single particles in Fourier space into long- and short-range

→ compute both parts separately with the PM and TREE algorithms

→ short-range with TREE and long-range with PM



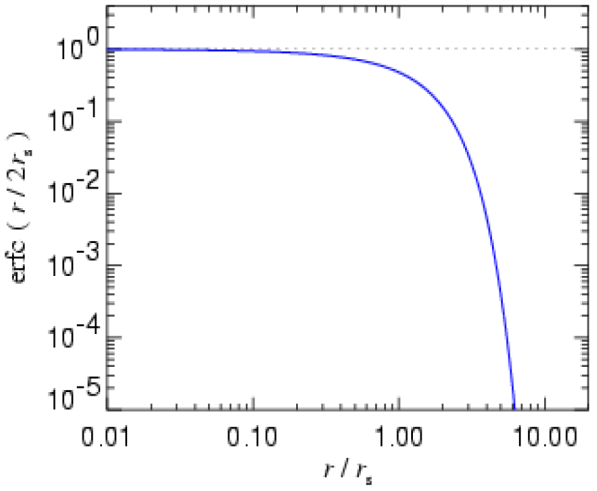
TREE-PM: Walk TREE only locally

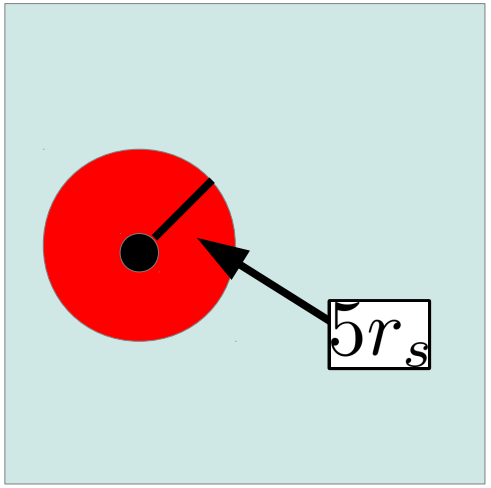
→ erfc(x) drops very quickly: erfc(2.5)~10^{-4}

→ real space only needs to be calculated in vicinity of particle

→ walk tree only in vicinity of particle, no long-range contribution

→ longe-range contribution comes from PM part

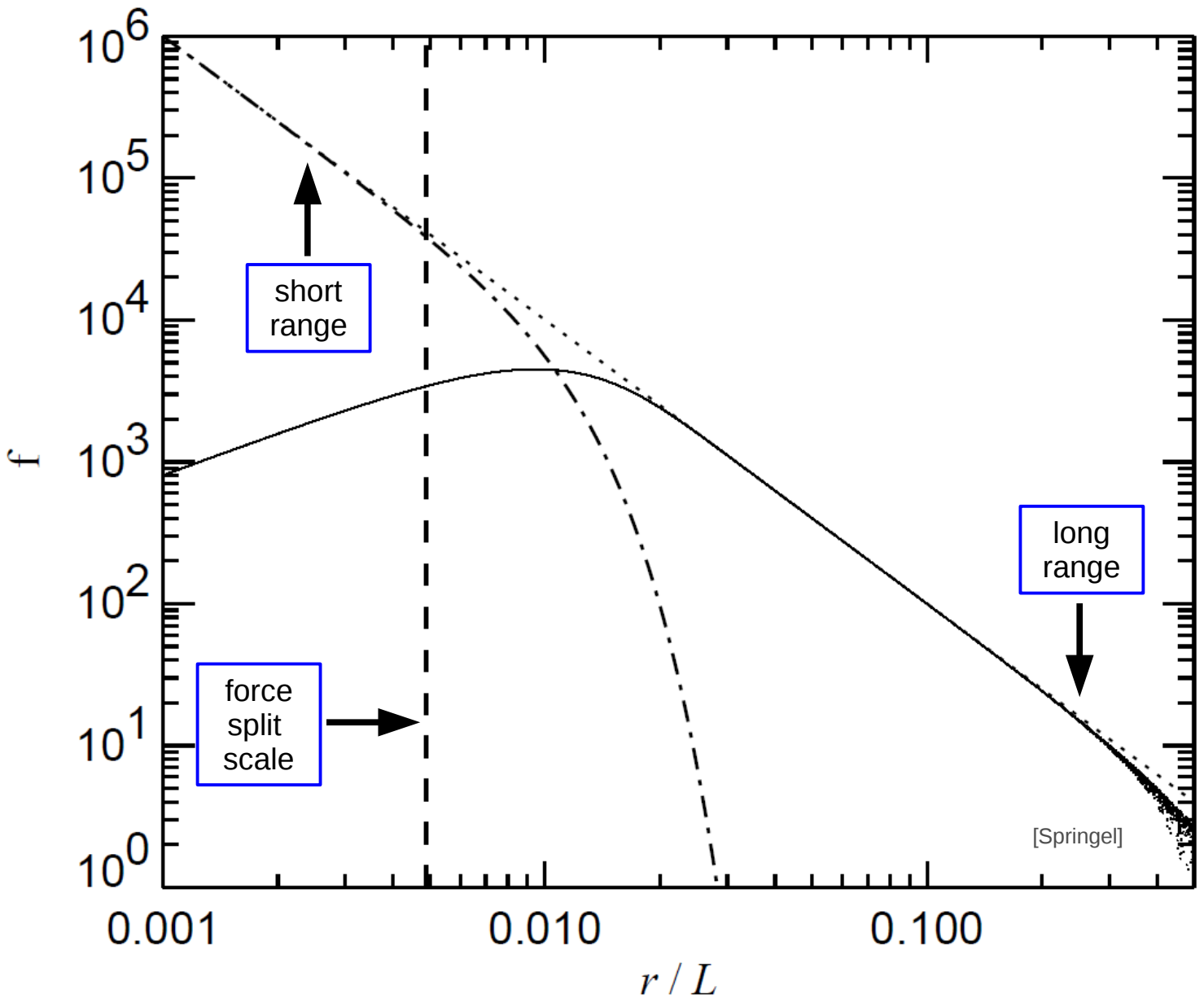




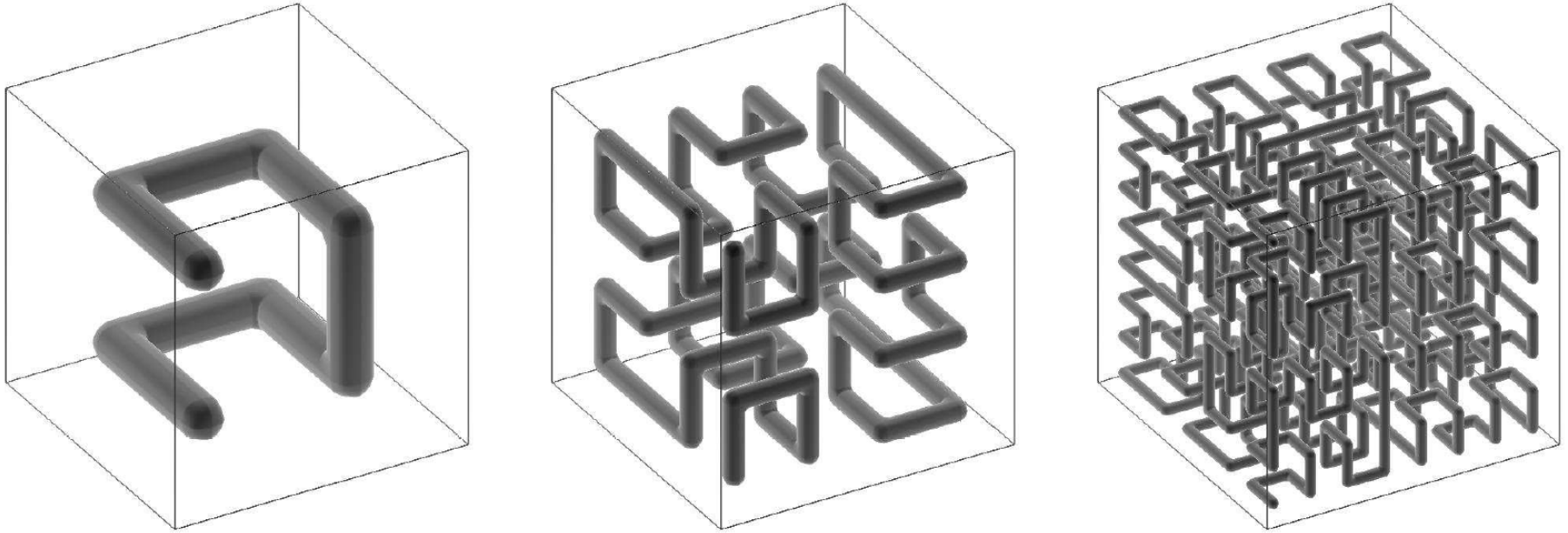
TREE-PM advantages:
- accurate and fast longe-range force
- high resolution on small scales

TREE-PM scheme used for many state-of-the-art simulations:
Millennium (1,2,XXL), Aquarius, Phoenix, Illustris, ….

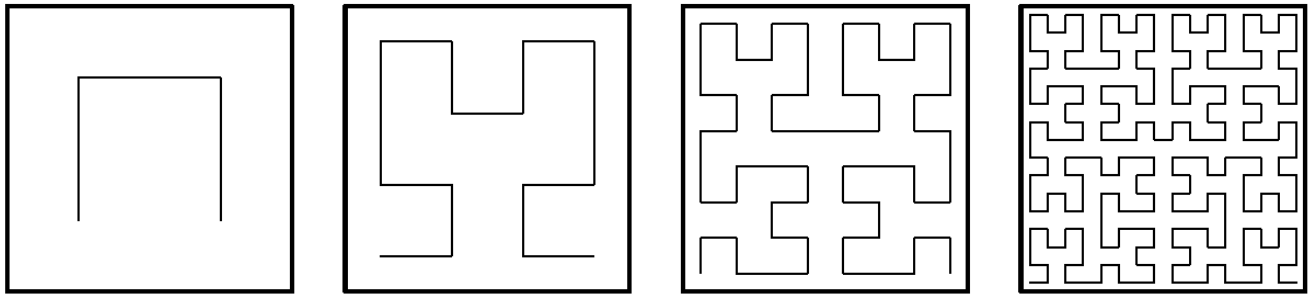
TREE-PM: Patching Forces together



TREE-PM: Parallelization



[Springel]

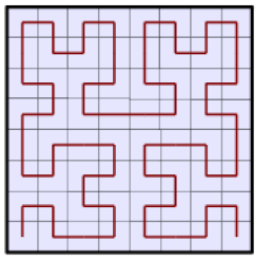


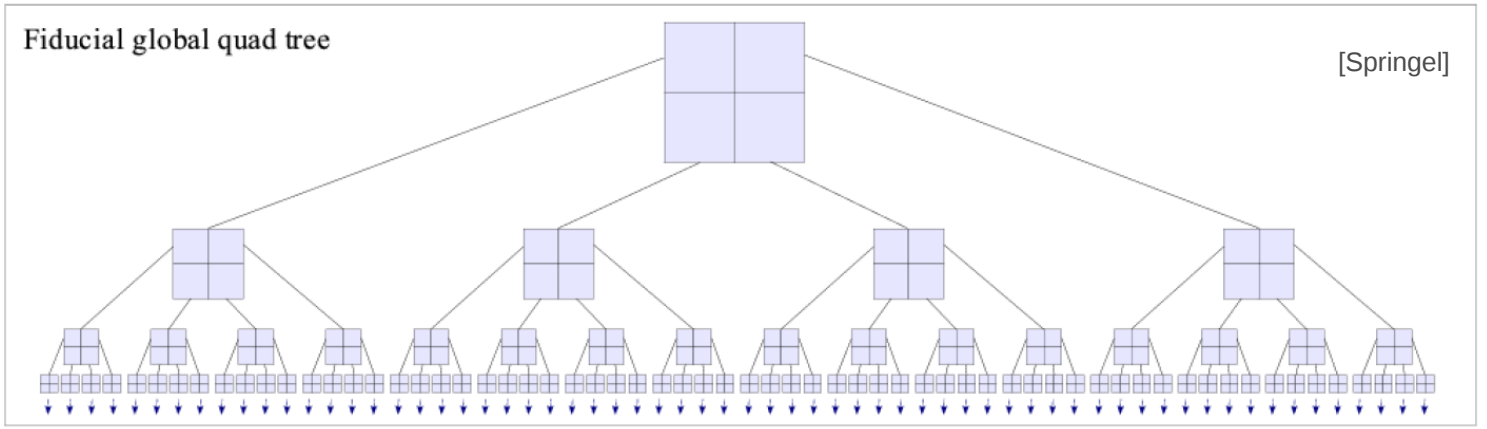
→ sort particles along a fractal Peano-Hilbert curve

→ chop curve for domain decomposition

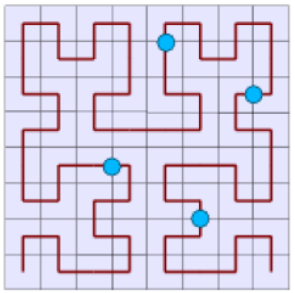
TREE-PM: Parallelization

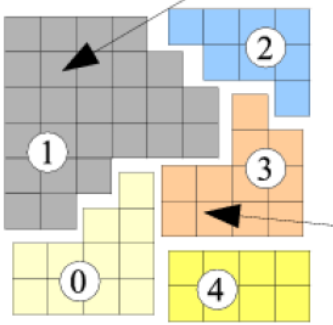
Peano-Hilbert curve

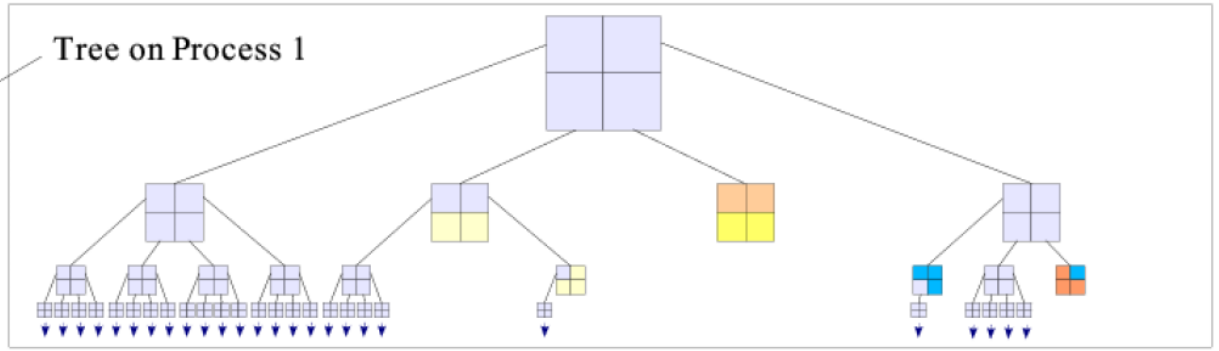


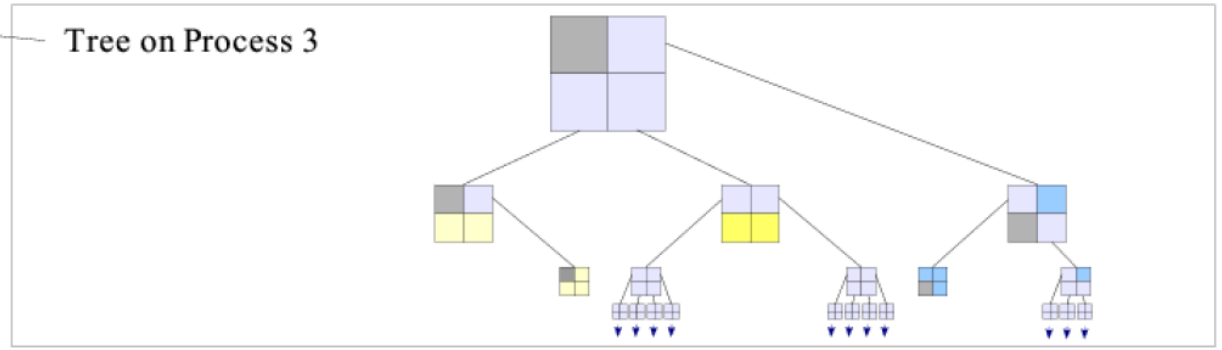


Domains are obtained by cutting the Peano-Hilbert curve into segments

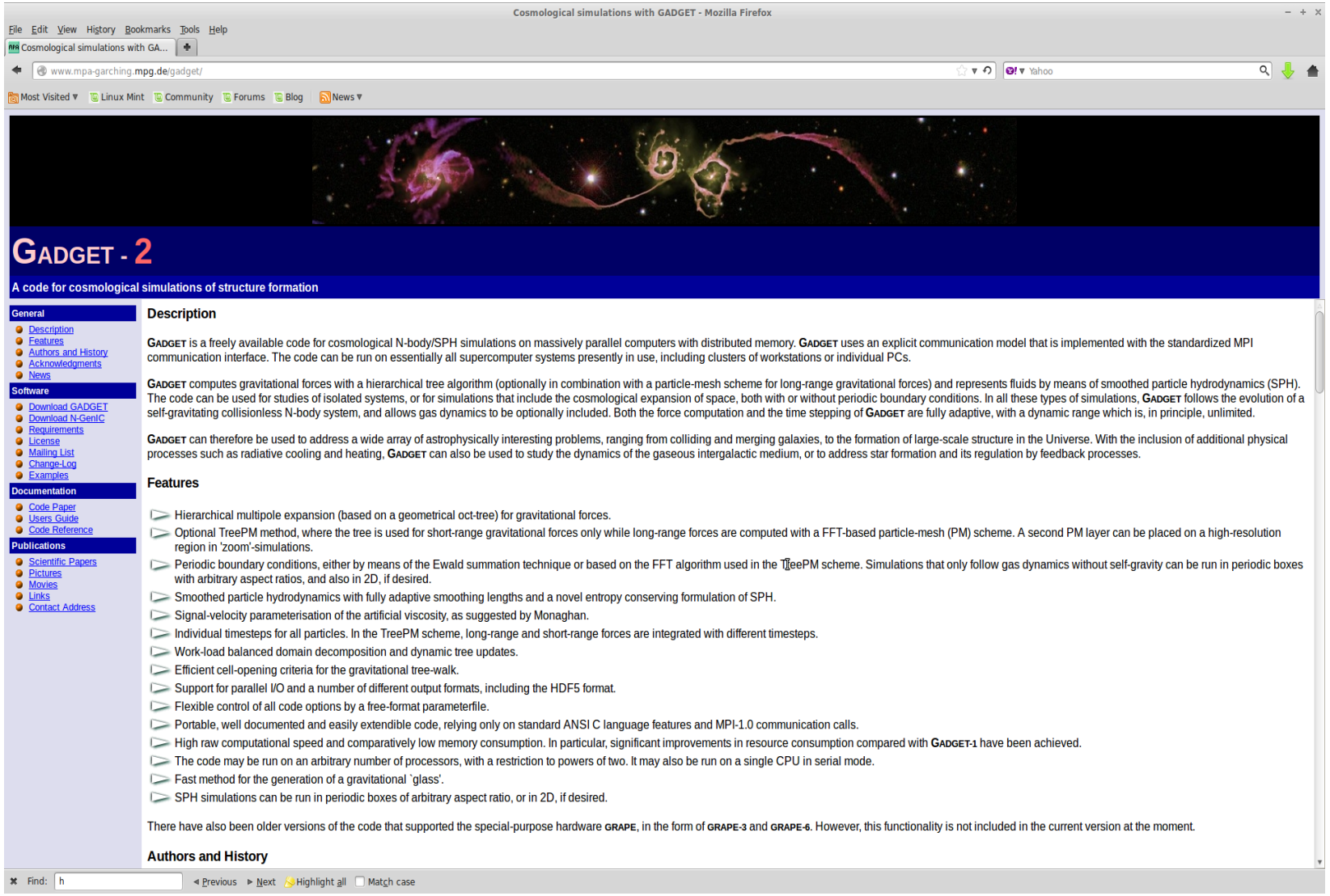








For tomorrow



http://www.mpa-garching.mpg.de/gadget/

Prepacked Setup

http://www.cfa.harvard.edu/~mvogelsb/handson.tar.gz

→ contains: code, initial conditions, I/O reader, simple post-processing example

→ requires: C, make, FFTW, GSL, HDF5, MPI, Python (Numpy, PyTables, ...)

Content:

```
drwxr-xr-x 2 mark mark   4096 Jul 28 13:31 ICs
-rw-r-xr-- 1 mark mark   2451 Jul 28 13:31 param.txt
-rw-r--r-- 1 mark mark  13521 Jul 28 13:31 readsnapGadget2.py
drwxr-xr-x 3 mark mark   4096 Jul 28 13:31 code
-rw-r--r-- 1 mark mark   1085 Jul 28 13:31 plot_map.py
-rw-r--r-- 1 mark mark     43 Jul 28 13:31 scalefactors
-rwxr-xr-x 1 mark mark    185 Jul 28 13:31 job.bsub
-rwxr-xr-x 1 mark mark 606780 Jul 28 15:19 Gadget2
-rw-r--r-- 1 mark mark   2575 Jul 28 15:25 param.txt-usedvalues
drwxr-xr-x 2 mark mark   4096 Jul 28 15:46 output
```

Content

(1) **Overview**

(2) **Dark Matter Dynamics**

(3) **Force Calculation**

(4) **Time Integration**

(5) **Recent Developments**

(6) **Hands-on Session**

Time Integration

- General idea about time integration
- How to solve the second order ODE?
- Which features are desirable for cosmological simulations?

Basic Schemes: Idea

Explicit Euler Method:

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

→ simplest scheme

→ right hand side readily available (i.e. explicit)

→ only first order accurate

Implicit Euler Method:

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

→ good stability properties

→ requires implicit solver (can be expensive, if function not easily invertable)

Basic Schemes: Higher Order

Runga-Kutta methods: standard ODE integration method

$$k_1 = f(y_n, t_n)$$

$$k_2 = f(y_n + k_1 \Delta t, t_n + \Delta t)$$

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

2nd order

$$k_1 = f(y_n, t_n)$$

$$k_2 = f(y_n + k_1 \Delta t/2, t_n + \Delta t/2)$$

$$k_3 = f(y_n + k_2 \Delta t/2, t_n + \Delta t/2)$$

$$k_4 = f(y_n + k_3 \Delta t/2, t_n + \Delta t)$$

$$y_{n+1} = y_n + \left(\frac{k_1}{6} + \frac{k_2}{3} + \frac{k_3}{3} + \frac{k_4}{6} \right) \Delta t$$

4th order

Leapfrog Scheme: Discretization

$$\ddot{\mathbf{x}} = f(\mathbf{x})$$

general type

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

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

Leapfrog Scheme

Drift-Kick-Drift

$$x_{n+1/2} = x_n + v_n \frac{\Delta t}{2}$$

$$v_{n+1} = v_n + f(x_{n+1/2}) \Delta t$$

$$x_{n+1} = x_{n+1/2} + v_{n+1} \frac{\Delta t}{2}$$

Kick-Drift-Kick

$$v_{n+1/2} = v_n + f(x_n) \frac{\Delta t}{2}$$

$$x_{n+1} = x_n + v_{n+1/2} \Delta t$$

$$v_{n+1} = v_{n+1/2} + f(x_{n+1}) \frac{\Delta t}{2}$$

→ 2nd order accurate

→ symplectic, which is important for cosmological simulations

Leapfrog Scheme: Properties

→ Hamiltonian structure preserved if steps formulated as canonical transformations

→ such schemes are called symplectic (symplectic integrators)

→ time evolution is continuous canonical transformation generated by Hamiltonian

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

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

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

Hamiltonian equations of motion

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

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

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

time evolution of state vector through propagator

Leapfrog Scheme: Properties

$$H = H_{\mathrm{kin}} + H_{\mathrm{pot}}$$ ← separable Hamiltonian

→ Drift- and Kick-Operators:

$$D(\Delta t) = \exp\left(\int_t^{t+\Delta t} \mathrm{d}t H_{\mathrm{kin}}\right)$$

$$K(\Delta t) = \exp\left(\int_t^{t+\Delta t} \mathrm{d}t H_{\mathrm{pot}}\right)$$

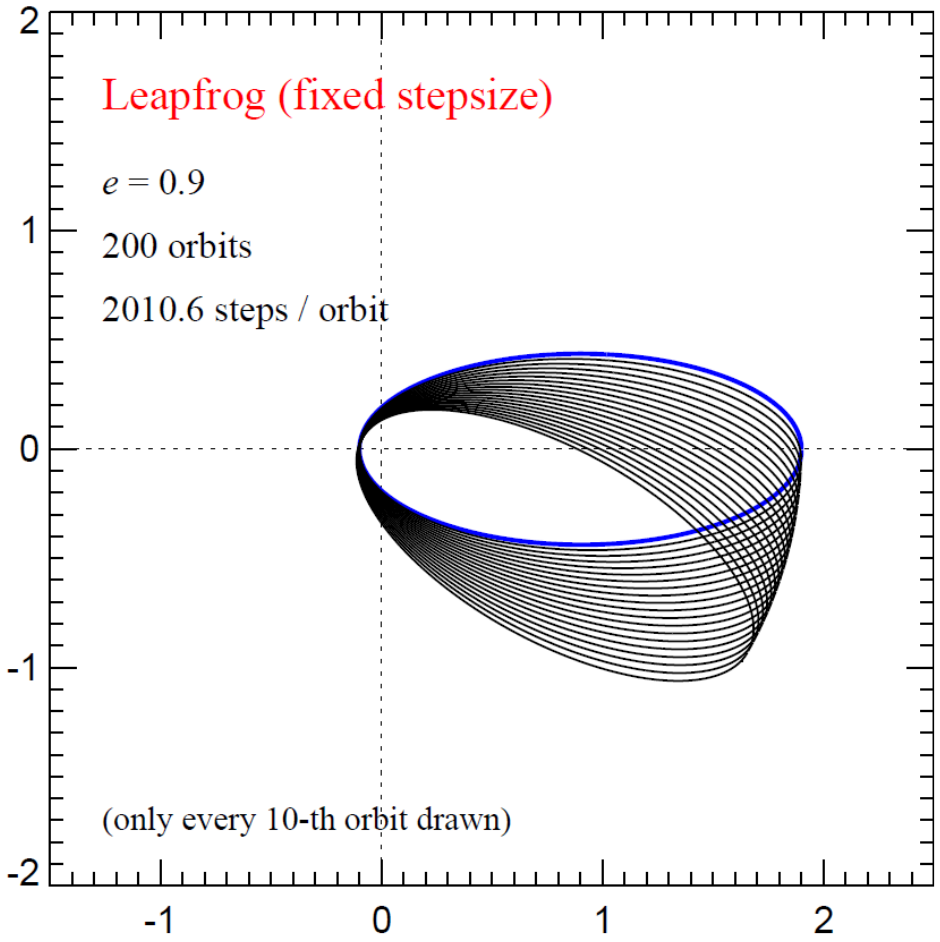
$$\tilde{\mathbf{U}}(\Delta t) = \mathbf{D}\left(\frac{\Delta t}{2}\right)\mathbf{K}(\Delta t)\mathbf{D}\left(\frac{\Delta t}{2}\right)$$

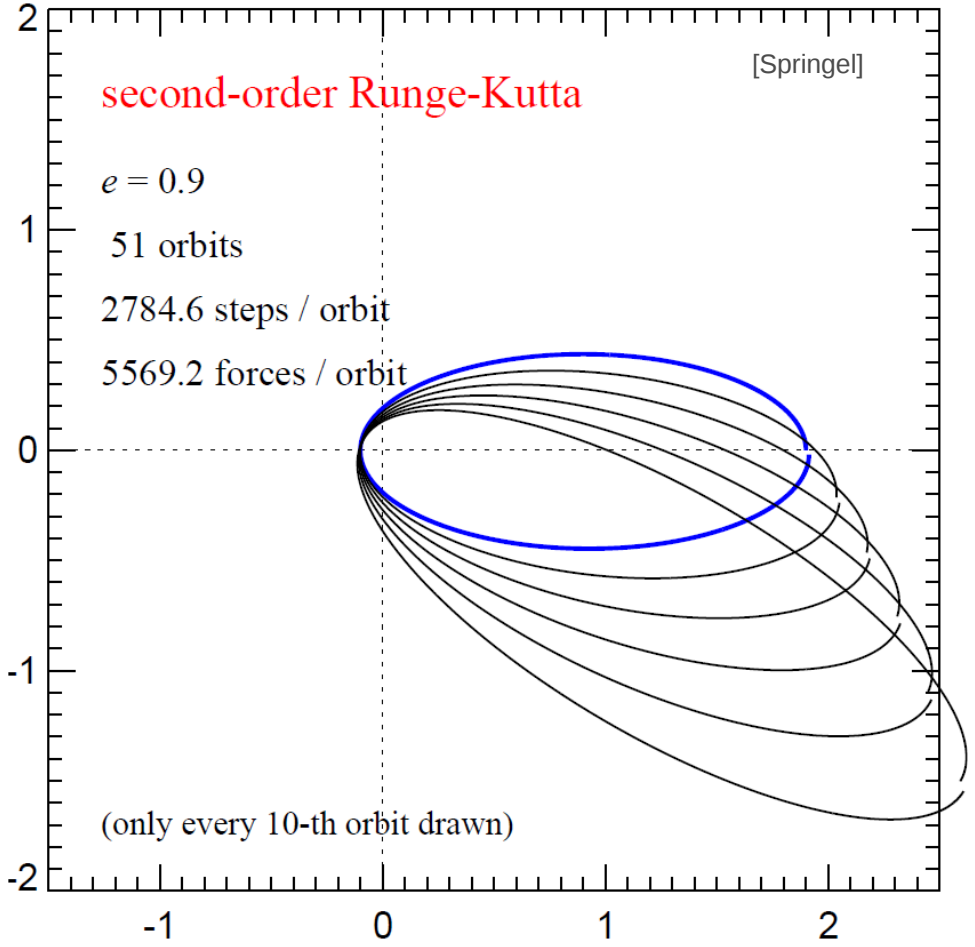
$$\tilde{\mathbf{U}}(\Delta t) = \mathbf{K}\left(\frac{\Delta t}{2}\right)\mathbf{D}(\Delta t)\mathbf{K}\left(\frac{\Delta t}{2}\right)$$

operator spitting ← drift and kick operators are symplectic transformations

→ reason for good conservation properties

Leapfrog Scheme: Properties





→ Leapfrog has no secular evolution in total energy

→ cosmological simulations integrate larger number of particle orbits in halo centers

→ avoid secular evolution

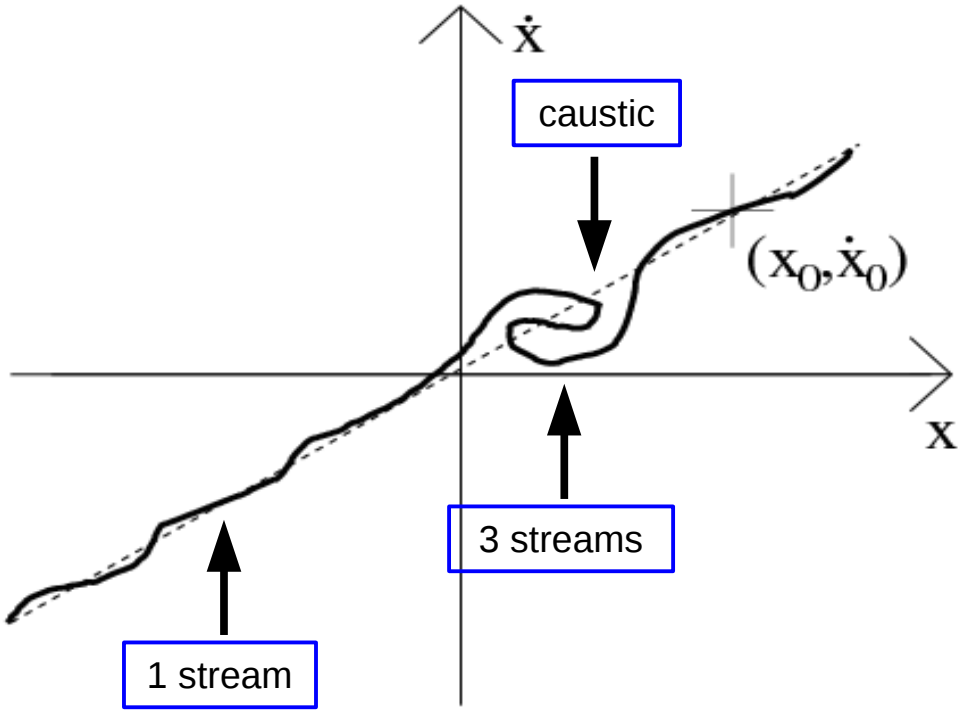
Recent Developments

- Small-scale structure of CDM?
- Self-Interacting DM?

Small-Scale Structure: CDM

→ CDM is cold and collisionless → restricted to 3D hyper-surface

→ thickness of hyper-surface related to primordial velocity dispersion



streams and caustics affect direct and indirect detection

Small-Scale Structure: Estimates

Self-similar halo formation:

Fillmore & Goldreich (1984), Bertschinger (1985),
Mohayaee & Shandarin (2006), Mohayaee & Salati (2008), ...

Caustic ring model:

Sikivie+ (1997),
Onemli & Sikivie (2007),
Natarajan & Sikivie (2007),
Duffy & Sikivie (2008),
Natarajan & Sikivie (2008), ...

General arguments:

Hogan (2001)

Predictions

- ~100 streams at solar position
- significant annihilation boost (5-100)
- strong caustic rings
- discrete velocity distribution
- distinct caustic structures

Strong impact on detection signals?

Small-Scale Structure: Extend N-body

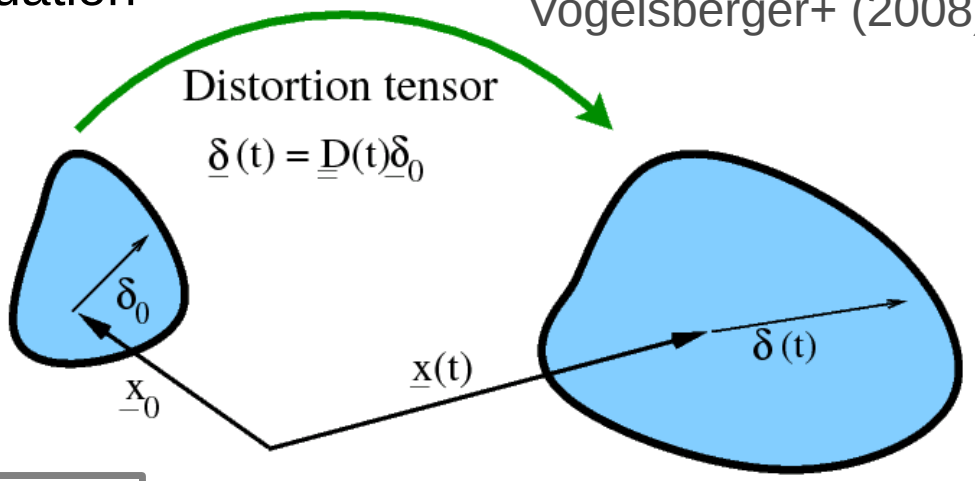
→ use phase-space geodesic deviation equation

→ calculate stream density

→ intra-stream annihilation

→ can be implemented in N-body codes

White & Vogelsberger (2008)
Vogelsberger+ (2008)



Caustic density scales with (vel. disp.)$^{1/2}$

$$\rho_{\max} = \frac{2^{5/4}\Gamma(5/4)\ f_0 m_p}{\sqrt{\pi}\ |\bar{s_1}\bar{s_2}|\ |\sigma\partial^2 A_3/\partial v_3^2|^{1/2}}$$

← scales with (velocity dispersion)$^{-1/2}$

Annihilation radiation through caustic log. divergent:

$$\Delta P = \frac{\langle\sigma_\times v\rangle\ f_0\ \Delta t}{|\bar{s_1}\bar{s_2}|\ |s_3 - s_3'|}\ \ln\left(\frac{2^{9/2}\Gamma(5/4)^2|s_3 s_3'|}{\pi\sigma|\partial^2 A_3/\partial v_3^2|}\right)$$

← log. divergent

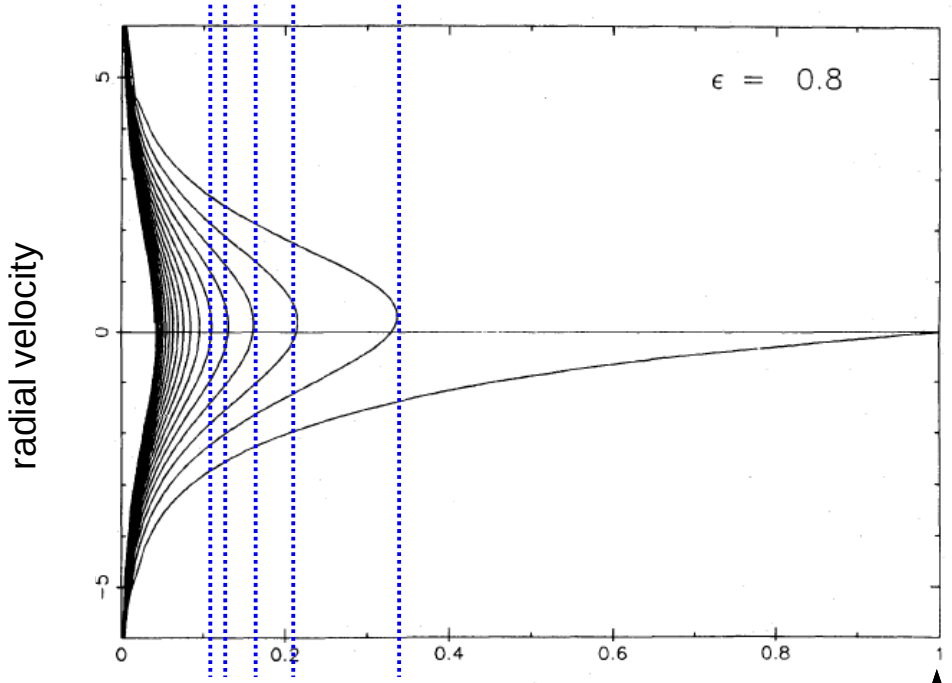
SELF-SIMILAR GRAVITATIONAL COLLAPSE IN AN EXPANDING UNIVERSE[1]

JAMES A. FILLMORE AND PETER GOLDREICH
California Institute of Technology
Received 1983 October 10; accepted 1983 December 5

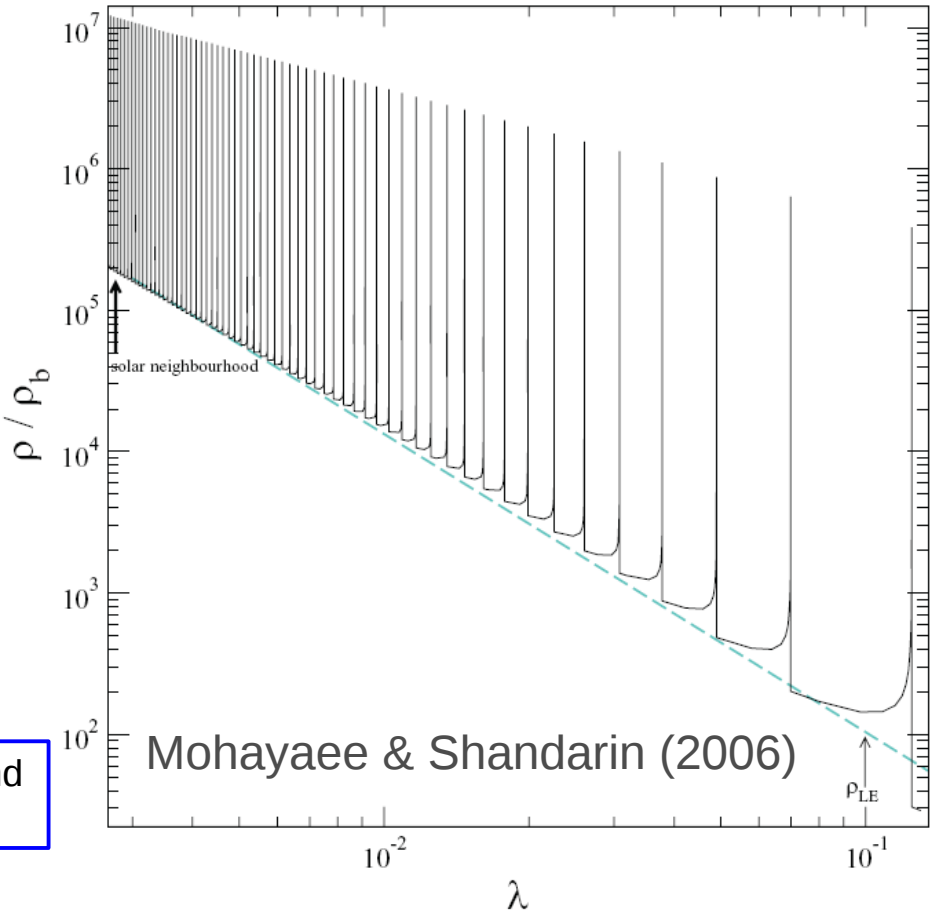
ABSTRACT

We derive similarity solutions which describe the collapse of cold, collisionless matter in a perturbed Einstein–de Sitter universe. We obtain three classes of solutions, one each with planar, cylindrical, and spherical symmetry. Our solutions can be computed to arbitrary accuracy, and they follow the development of structure in both the linear and nonlinear regimes.

Subject headings: cosmology — relativity

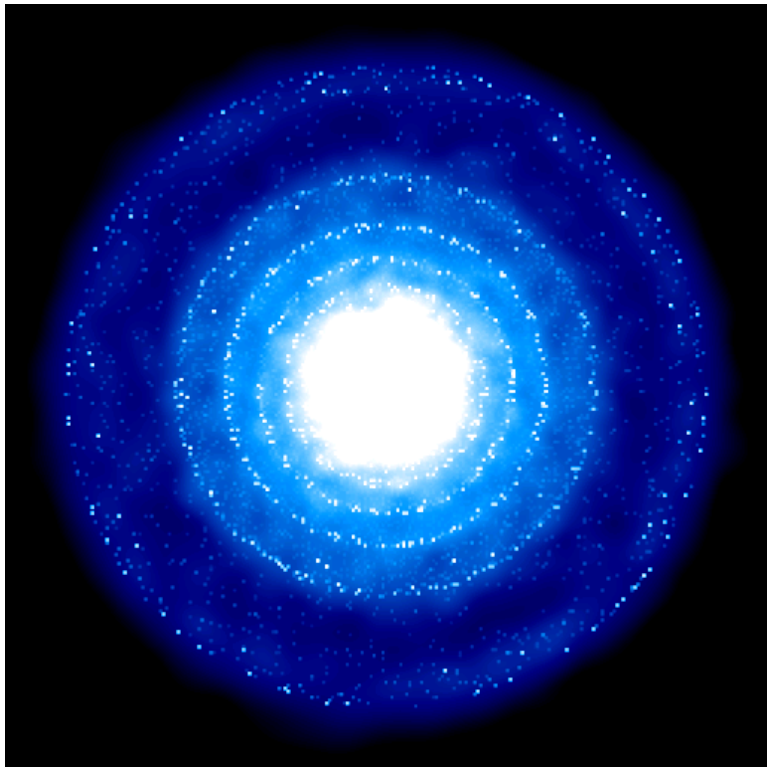


turnaround radius



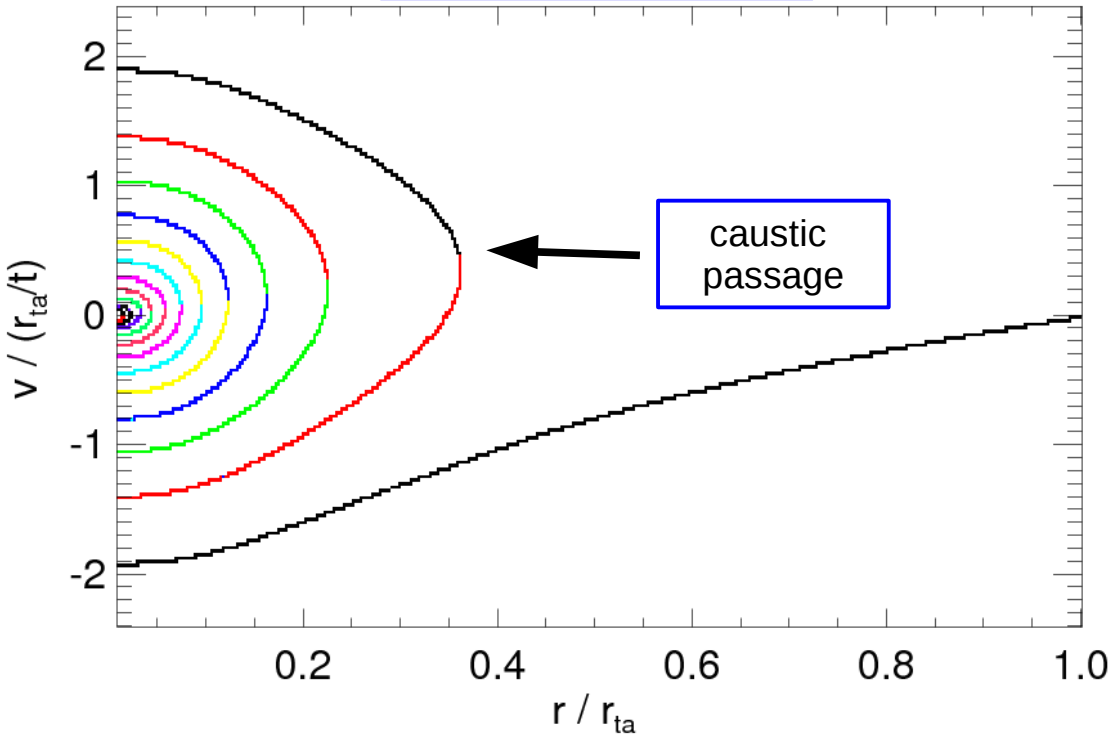
Small-Scale Structure: 1D example

annihilation signal



caustic spheres on top of
smooth annihilation signal

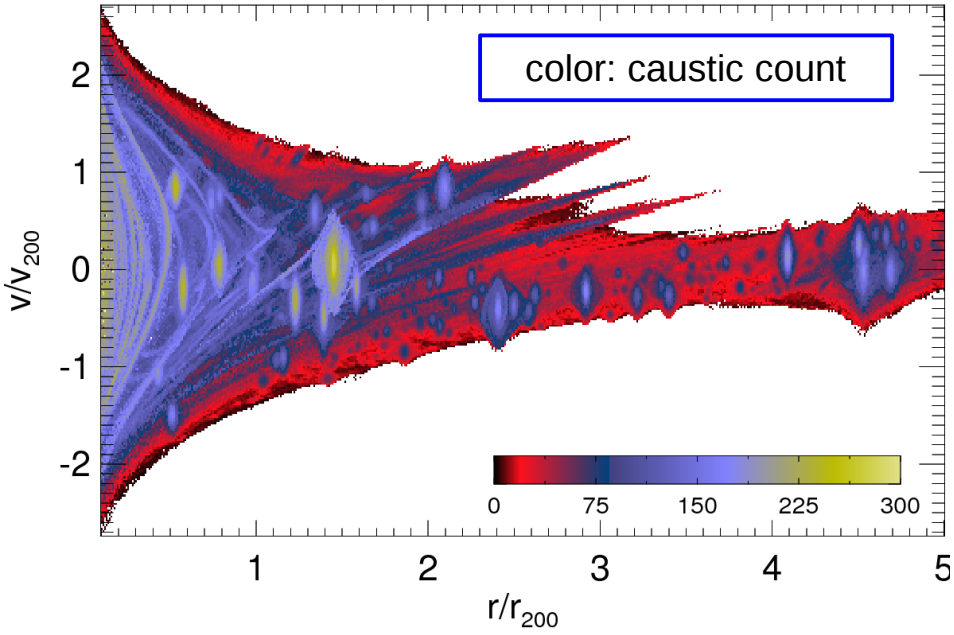
2D phase-space



Vogelsberger+ (2009)

caustic annihilation produces
shells of stronger emission

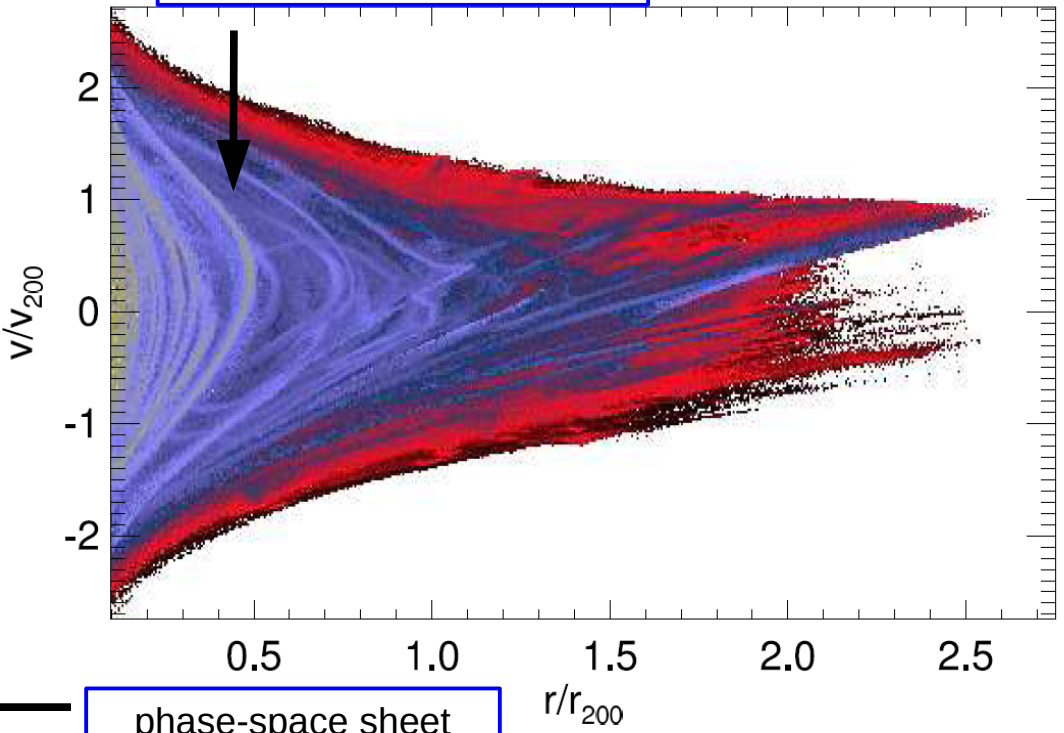
Small-Scale Structure: CDM Halo

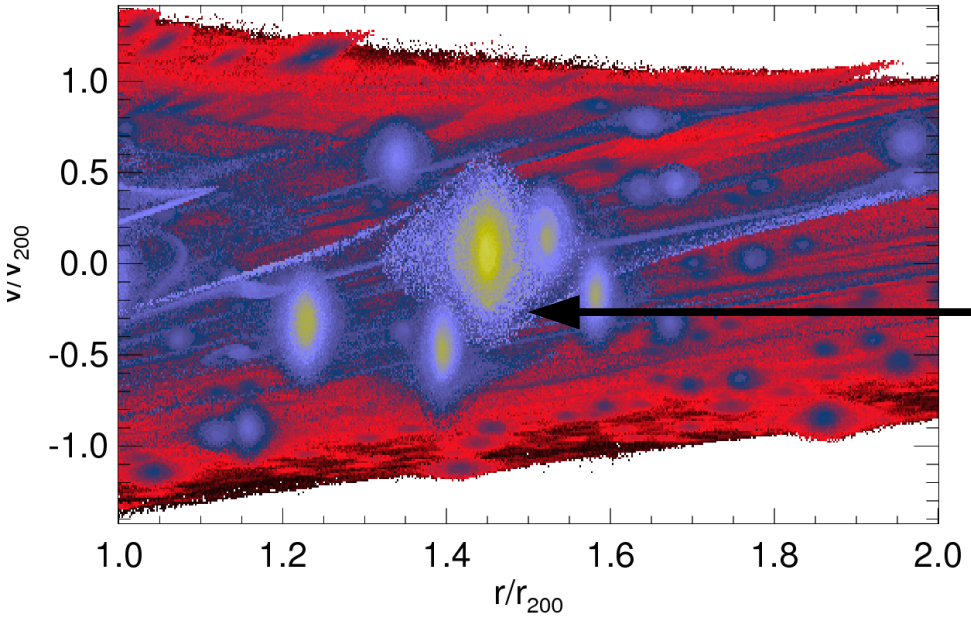


results for Milky Way-like DM halo

Vogelsberger & White (2011)

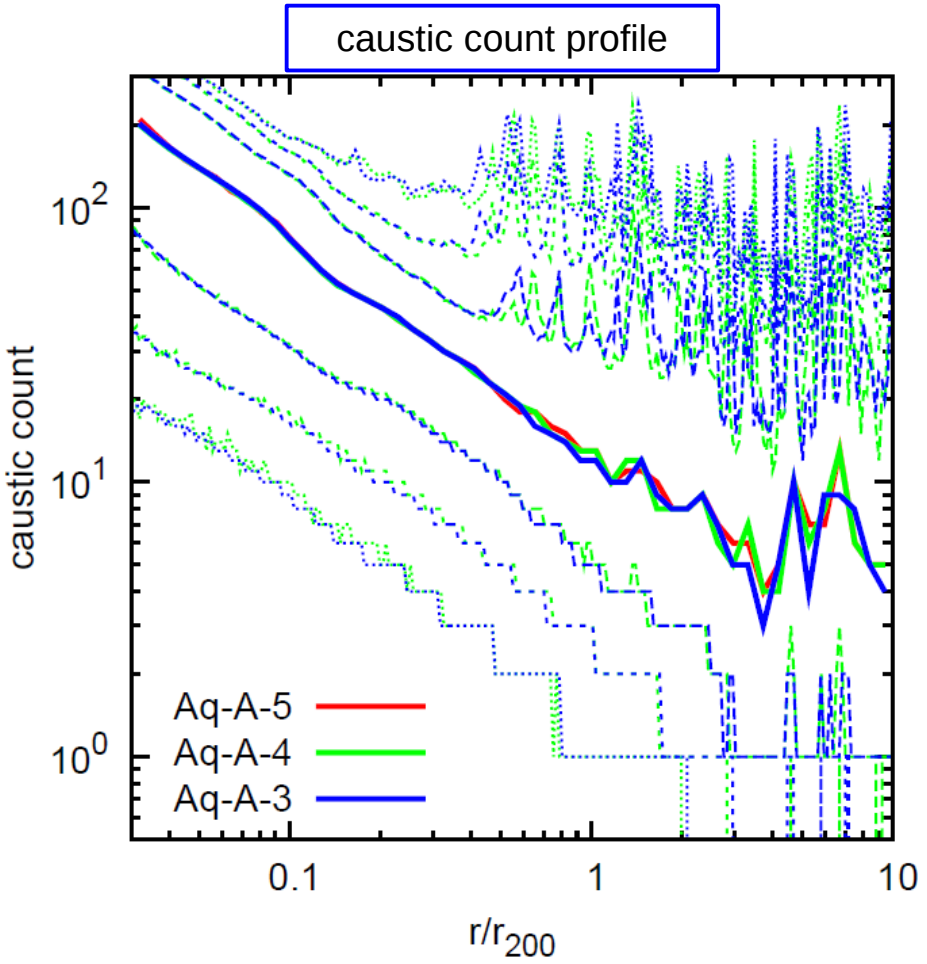
tidal streams (e.g. debris), not fine-grained streams!

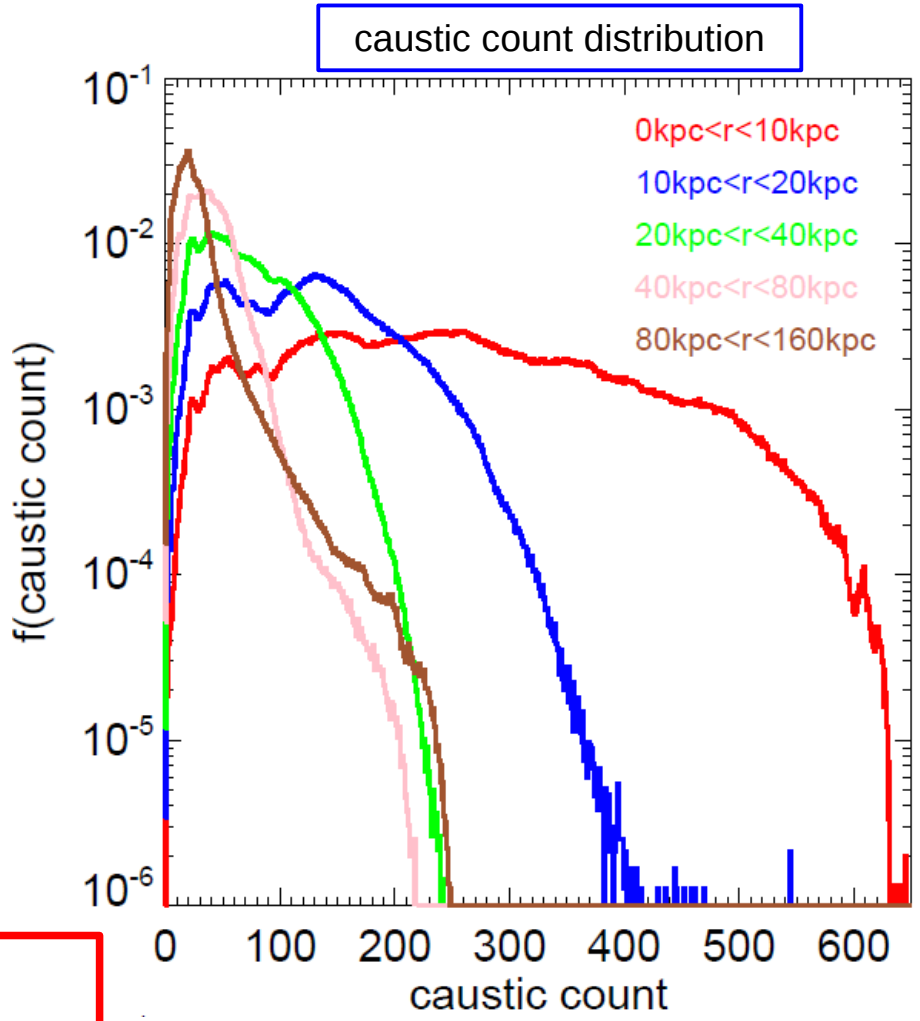




phase-space sheet winding up in subhalo

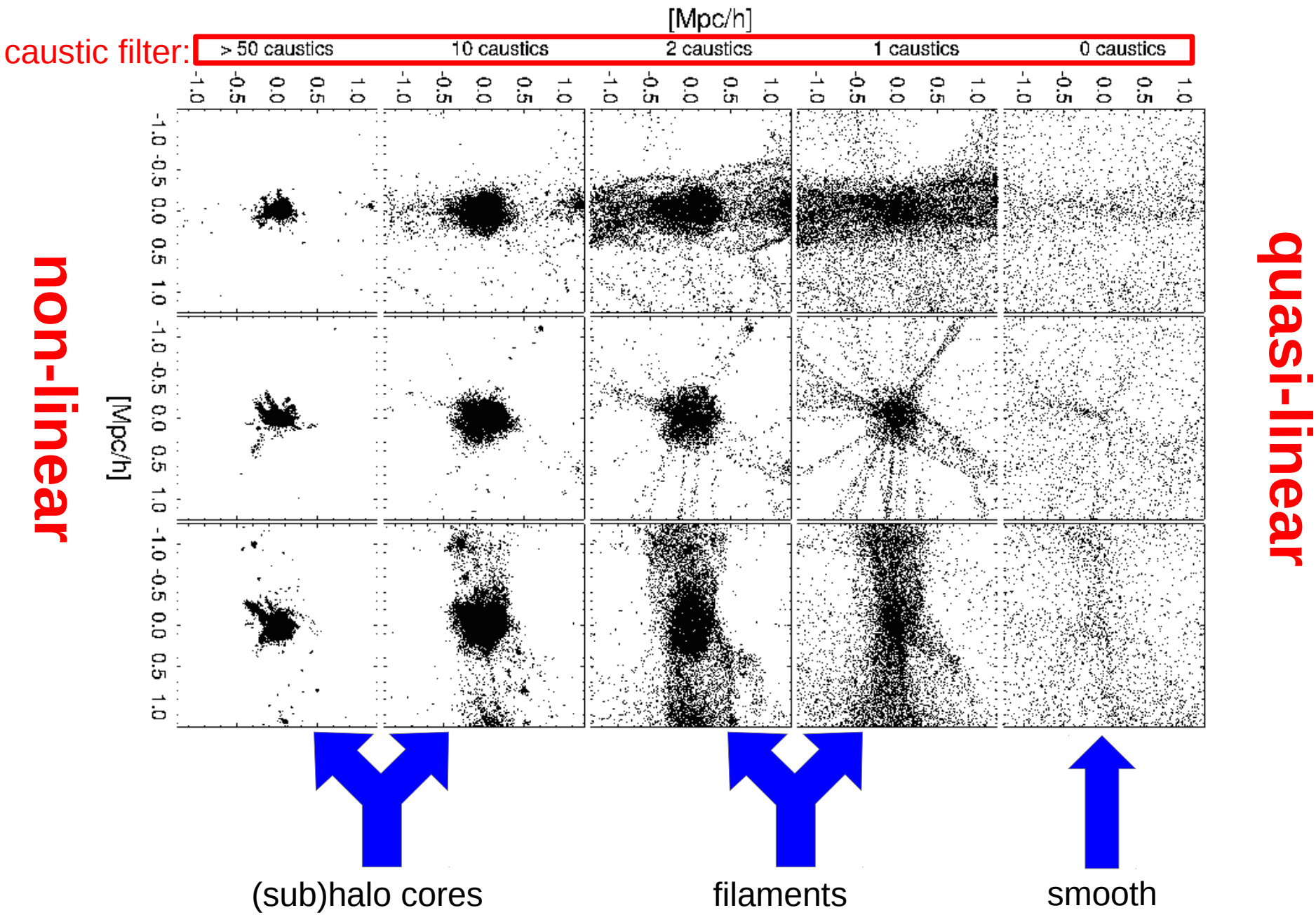
Small-Scale Structure: CDM Caustics



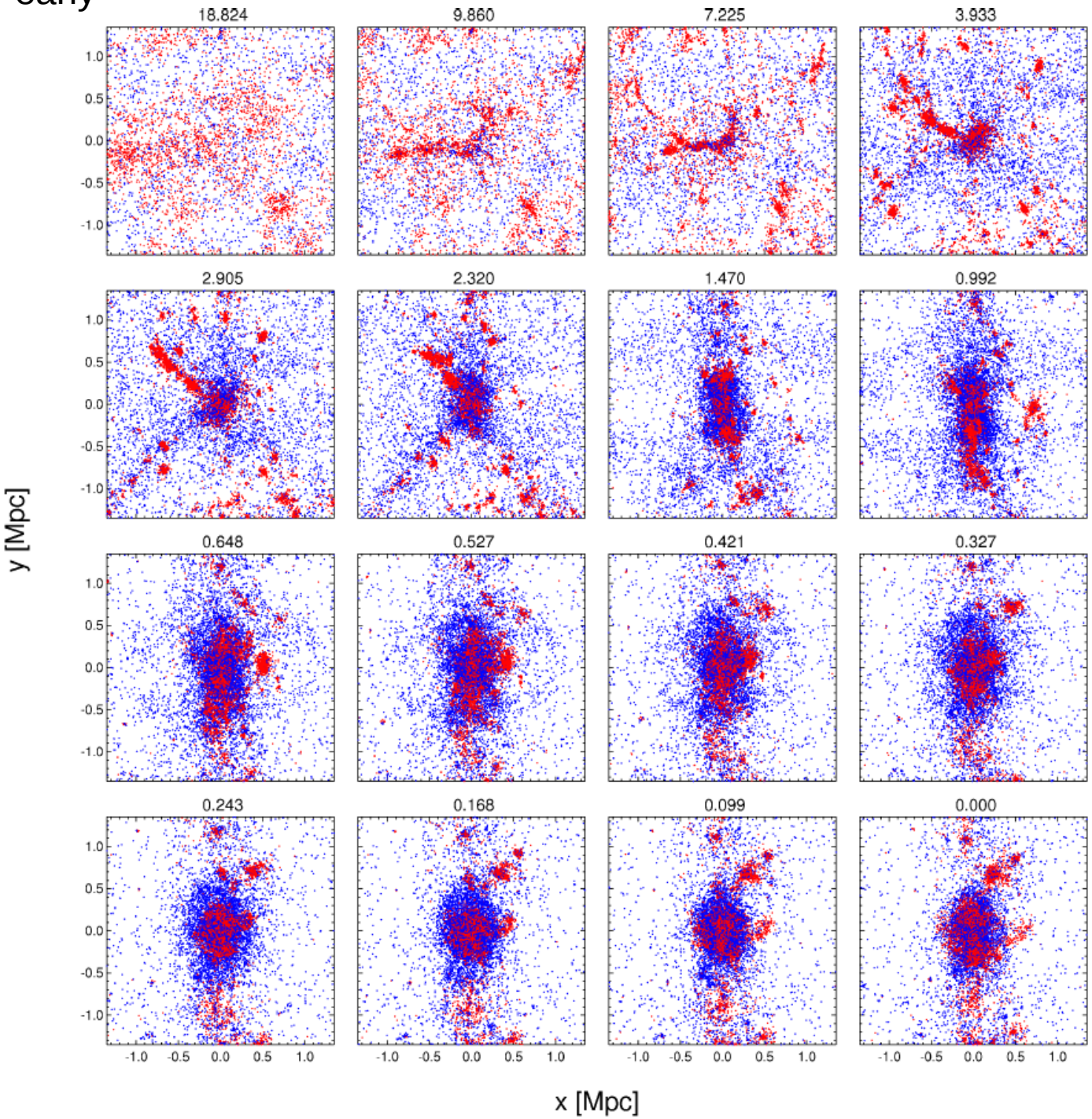


- broad distribution at all radii
- outer part broader due to subhalos
- tracks dynamical age (largest in center)
- converged

Small-Scale Structure: CDM Caustics



early

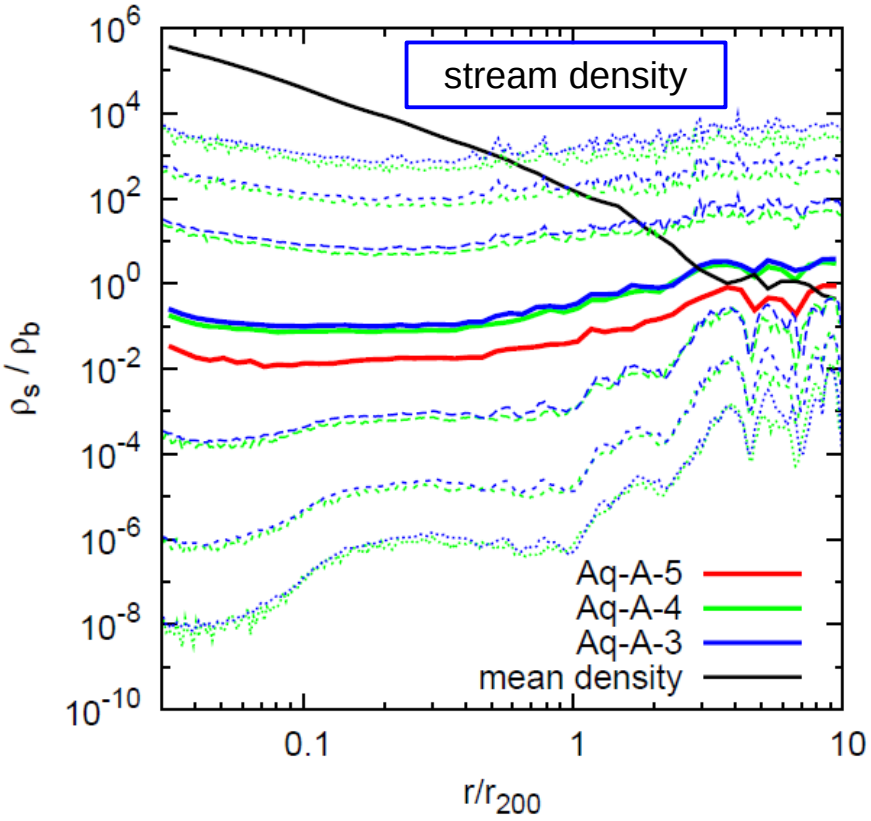


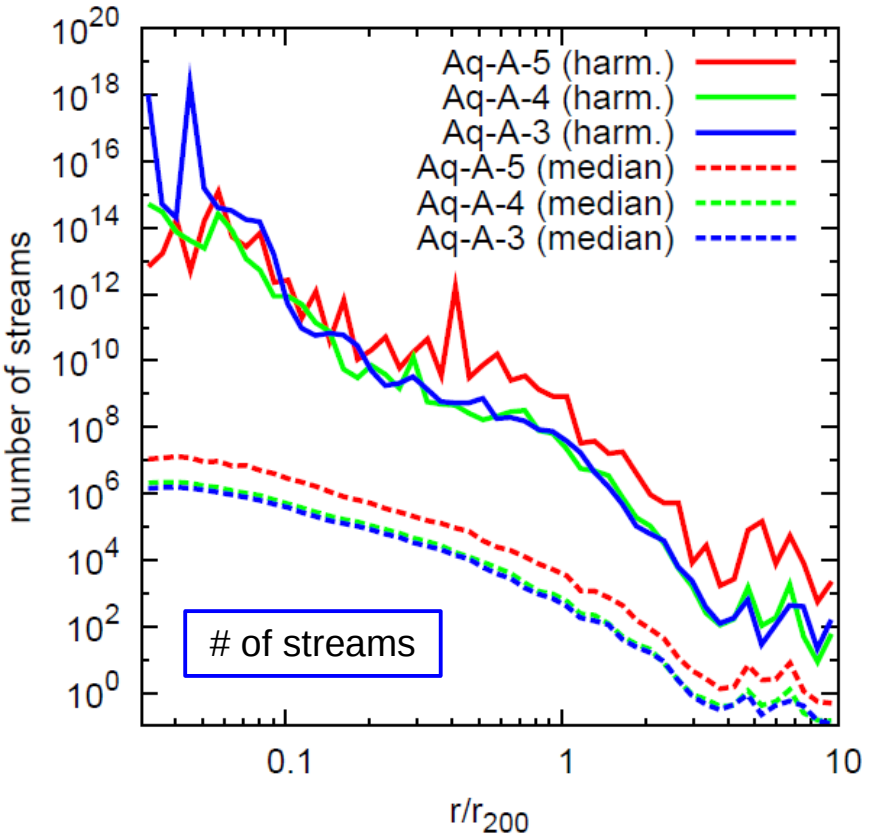
low stream density regions:
mixing in early forming objects at high z
→ **strong** mixing

high stream density regions:
more diffuse, smoothly accreted at low z
→ **weak** mixing

late

Small-Scale Structure: CDM Streams

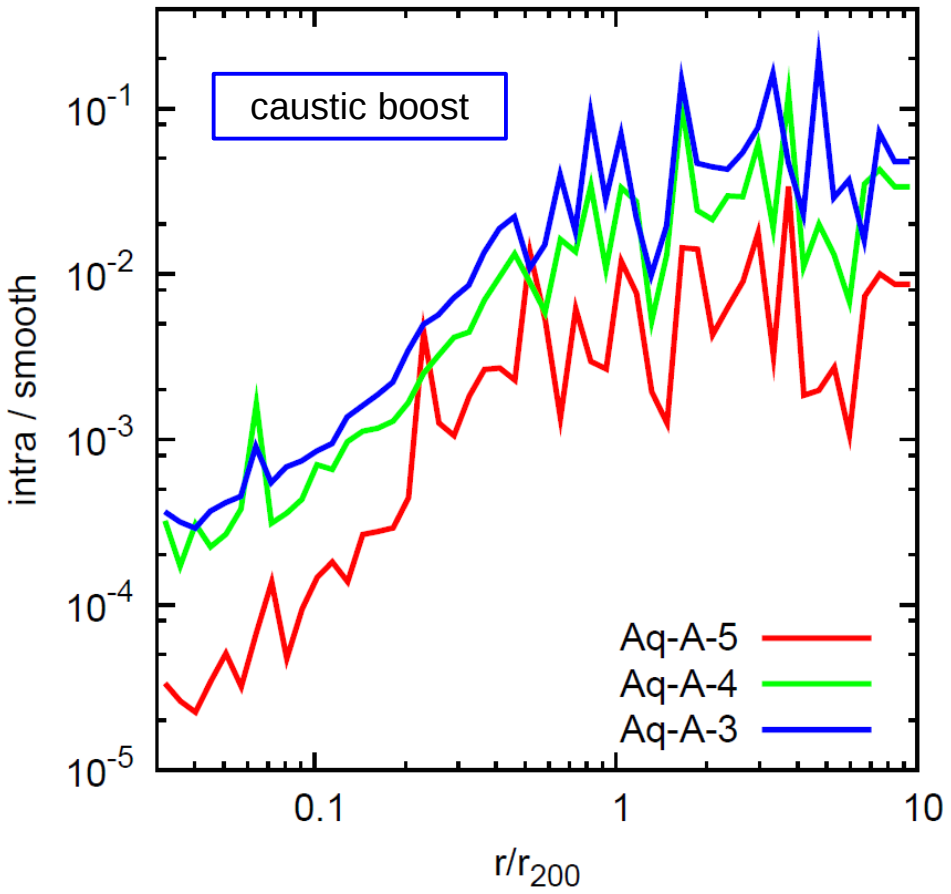




- stream density low
- locally large number of streams

if dark matter made of **axions** resonant detectors should find **energy spectrum** where few tenths of percent of total energy density is concentrated in few **very narrow spectral lines**

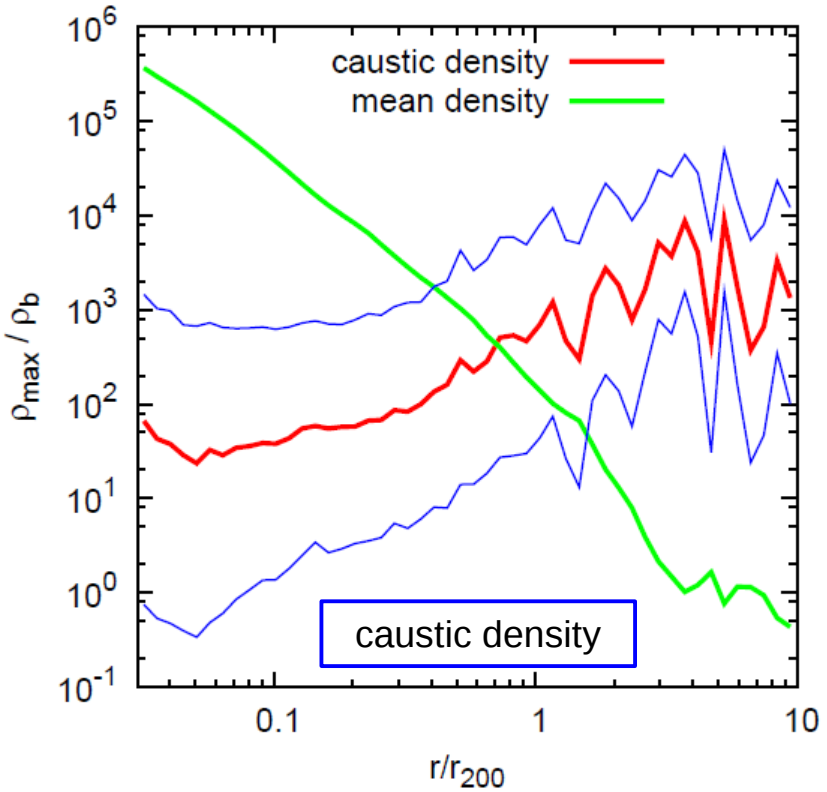
Small-Scale Structure: CDM Boost



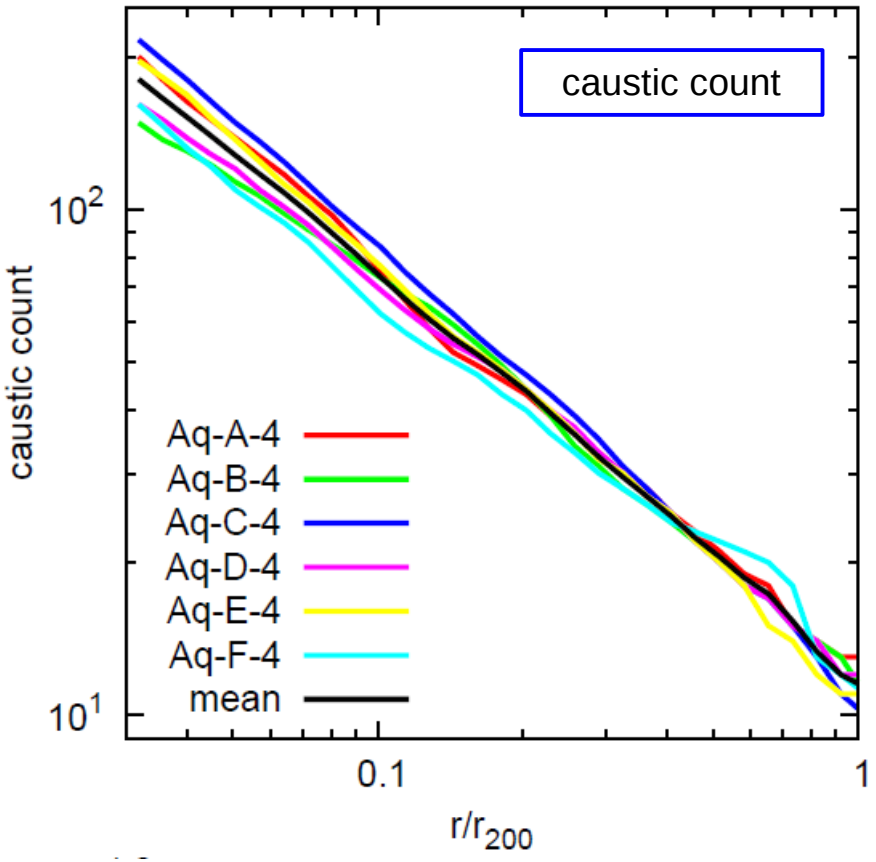
→ essentially no boost at solar circle
→ no distinct / dense caustic structures

'standard' annihilation rate calculations do not miss a large fine-grained contribution

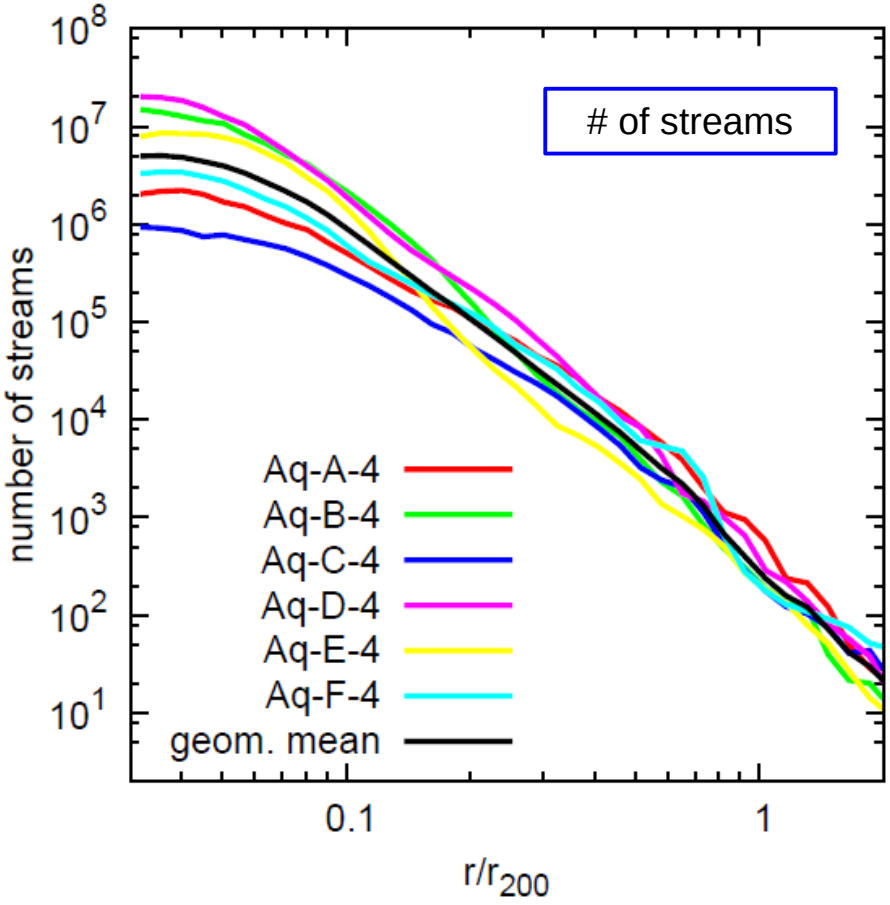
reason:
caustic density small compared to mean density



Small-Scale Structure: CDM Halo Variation

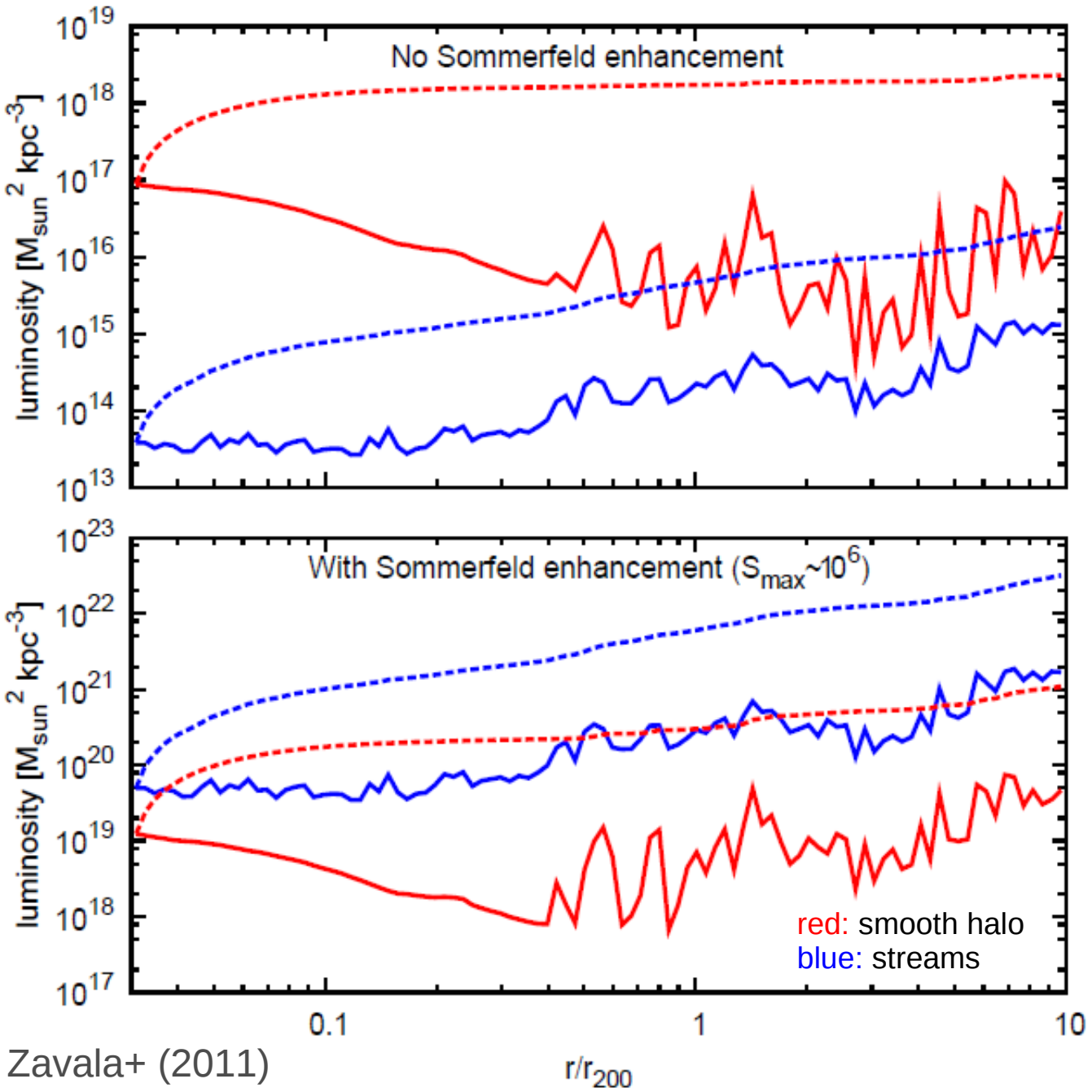


different halos in same mass range



→ there is not much variation
→ results universal for that halo mass

Small-Scale Structure: CDM Halo Variation



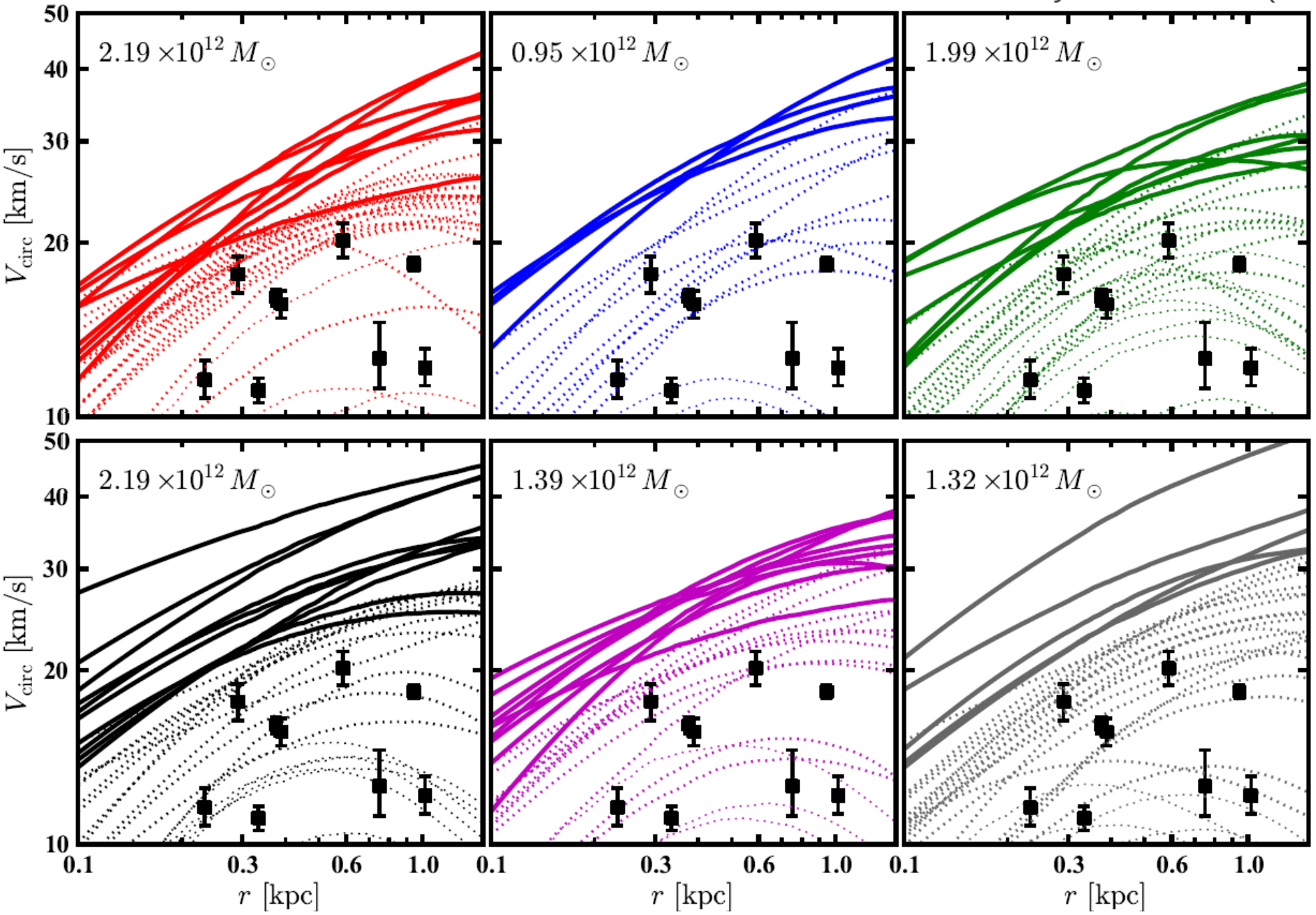
Zavala+ (2011)

→ Liouville's Theorem
→ streams contribute a lot
→ caustics negligible

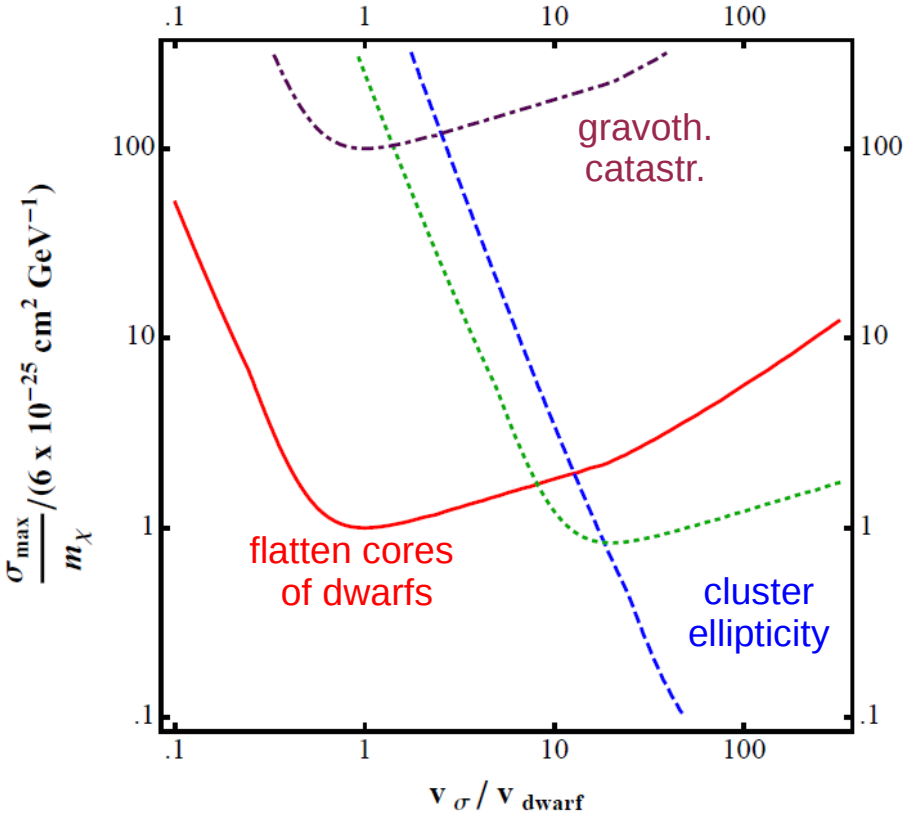
in a Sommerfeld-enhanced scenario the fine-grained structure dominates the smooth annihilation signal

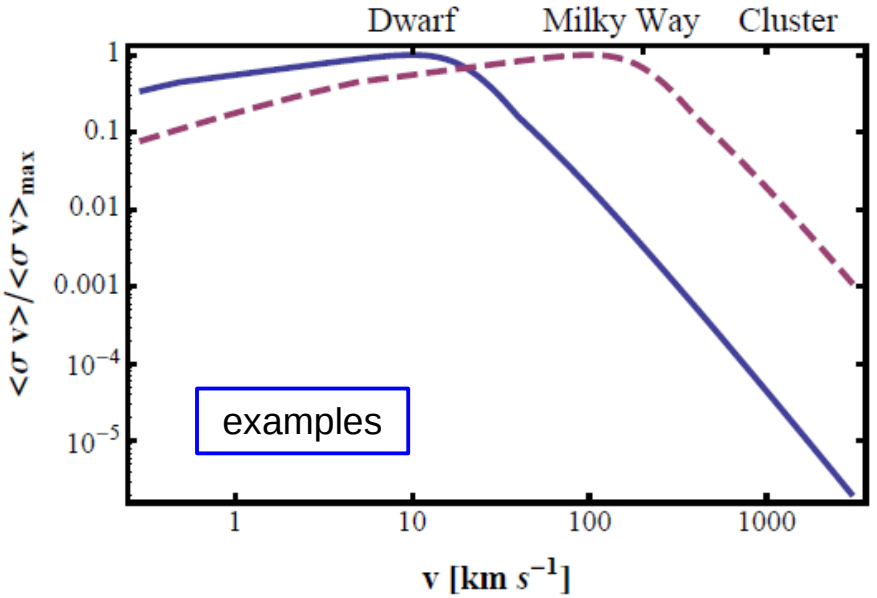
Self-Interacting DM: Motivation

Boylan-Kolchin+ (2012)



Self-Interacting DM: Idea



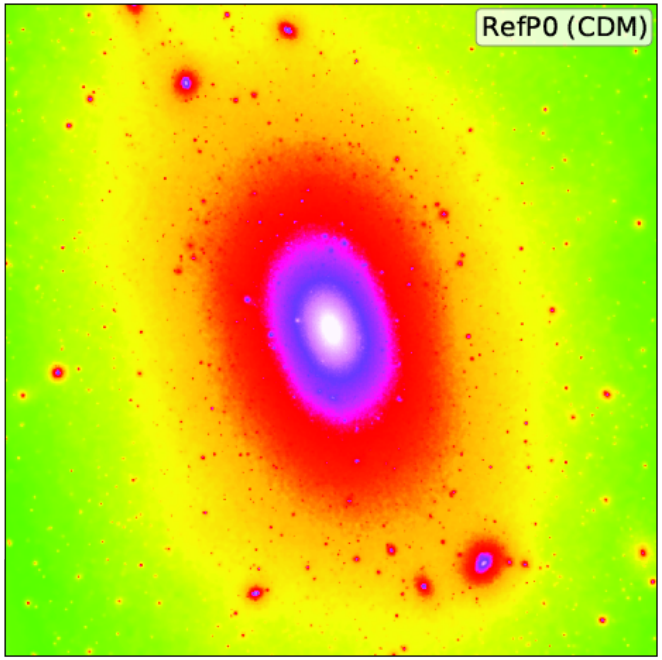


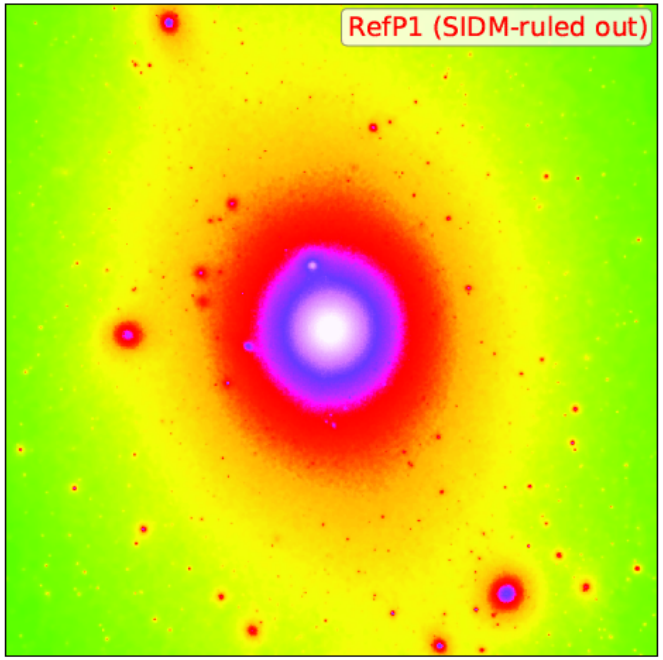
$$\frac{\sigma_T}{\sigma_{\rm max}} \approx \begin{cases} \frac{4\pi}{22.7}\ \beta^2 \ln\left(1+\beta^{-1}\right) & ,\beta < 0.2 \\ \frac{8\pi}{22.7}\ \beta^2\ \left(1+1.5\beta^{1.65}\right)^{-1} & ,0.2 < \beta < 10^3 \\ \frac{\pi}{22.7}\ \left(\ln\beta + 1 - \frac{1}{2}\ln^{-1}\beta\right)^2 & ,\beta > 10^3, \end{cases}$$

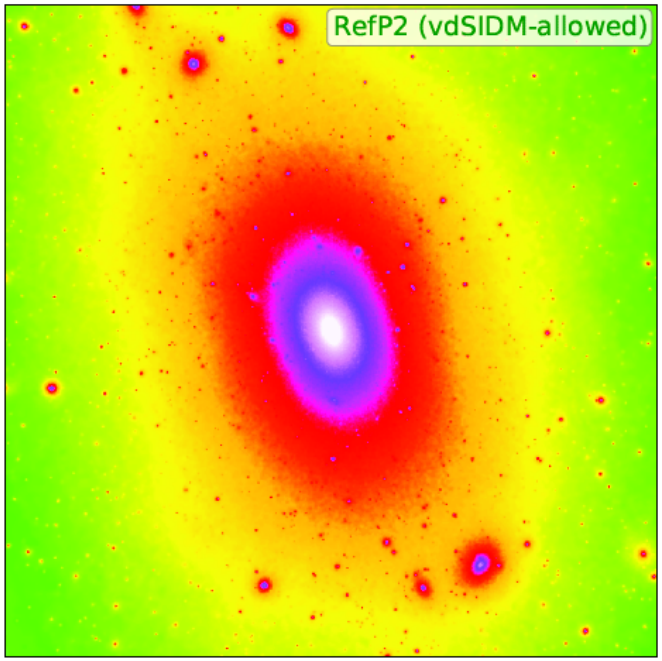
transfer cross-section

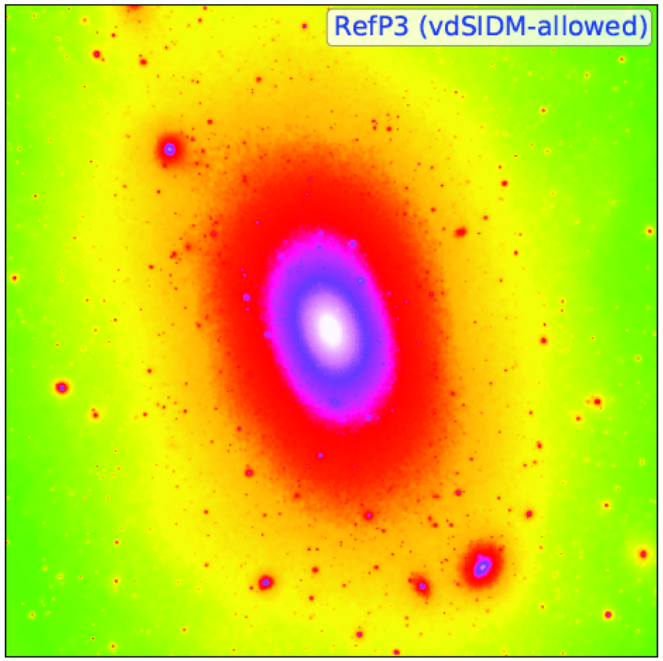
Feng+ (2010) Finkbeiner+ (2010) Loeb & Weiner (2011)

Self-Interacting DM: Halo Structure









→ Monte Carlo implementation

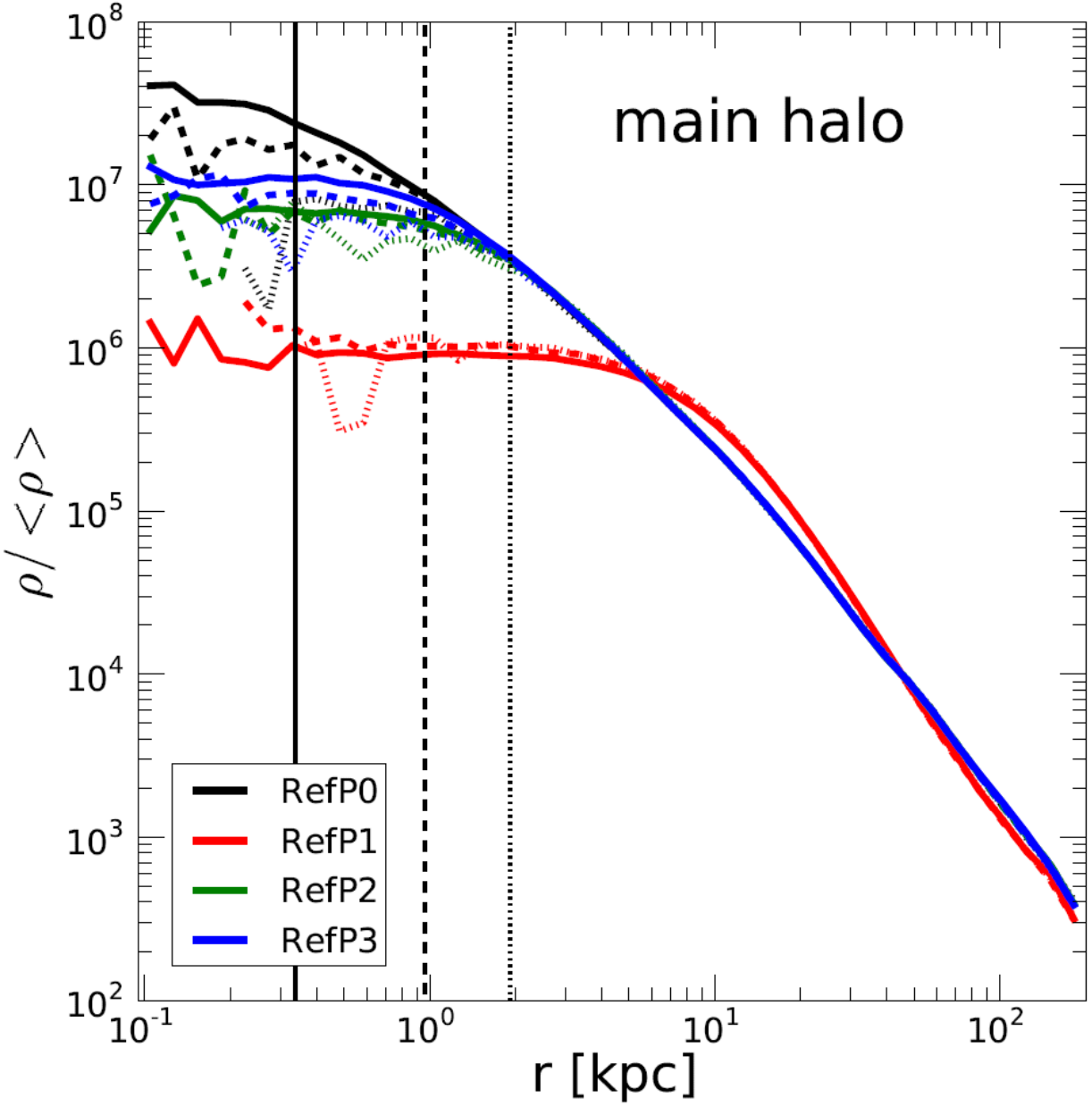
→ 4 different SIDM models

halos rounder

Name	Type	$\sigma_T^{\max}/m_\chi$ [cm^2 g^{-1}]	$v_{\max}$ [km s^{-1}]
RefP0	CDM	/	/
RefP1	SIDM (ruled out)	10	/
RefP2	vdSIDM (allowed)	3.5	30
RefP3	vdSIDM (allowed)	35	10

Vogelsberger+ (2012)

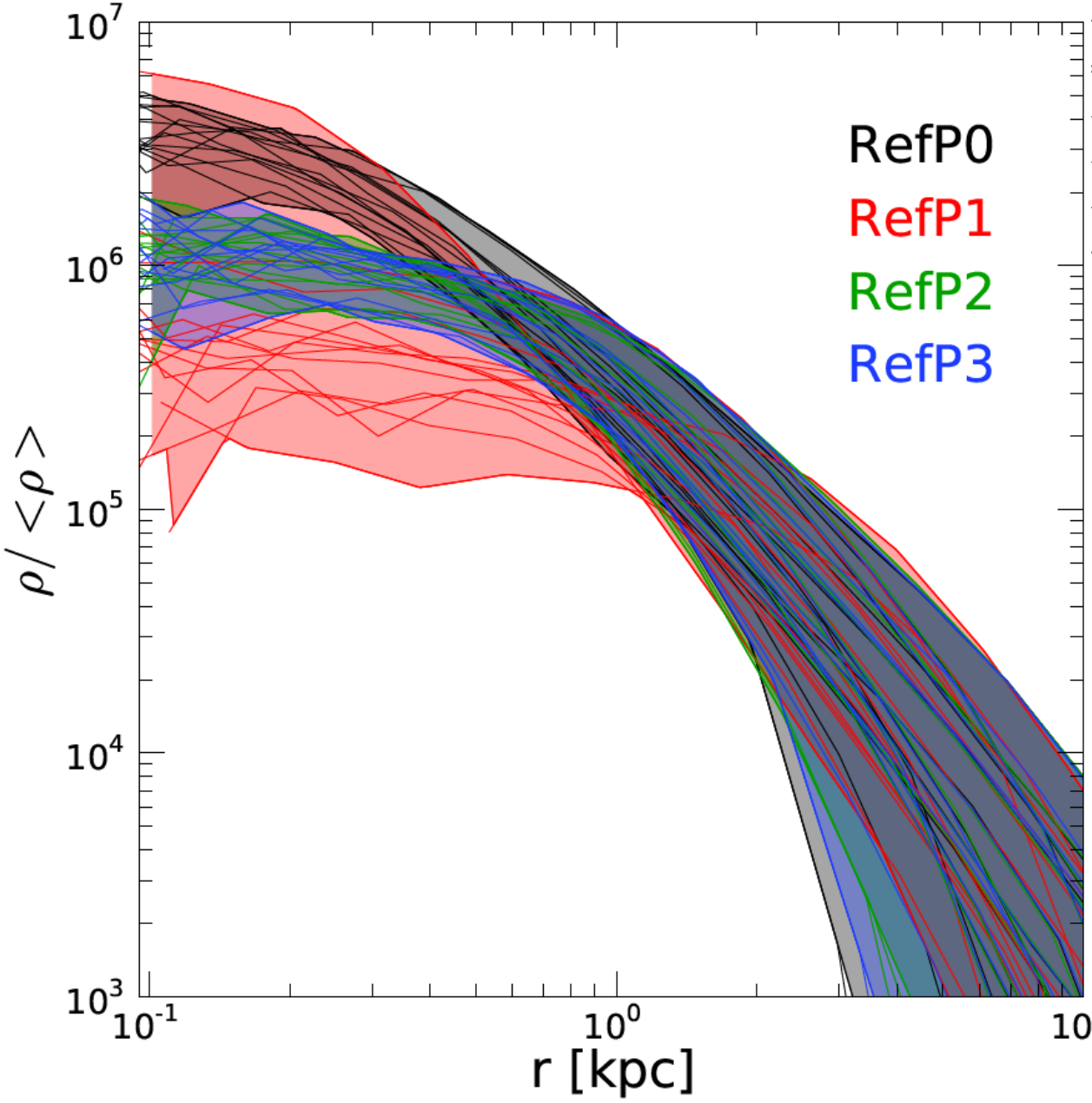
Self-Interacting DM: Halo Structure



Name	Type	$\sigma_T^{\max}/m_\chi$ [$\mathrm{cm^2\,g^{-1}}$]	$v_{\max}$ [$\mathrm{km\,s^{-1}}$]
RefP0	CDM	/	/
RefP1	SIDM (ruled out)	10	/
RefP2	vdSIDM (allowed)	3.5	30
RefP3	vdSIDM (allowed)	35	10

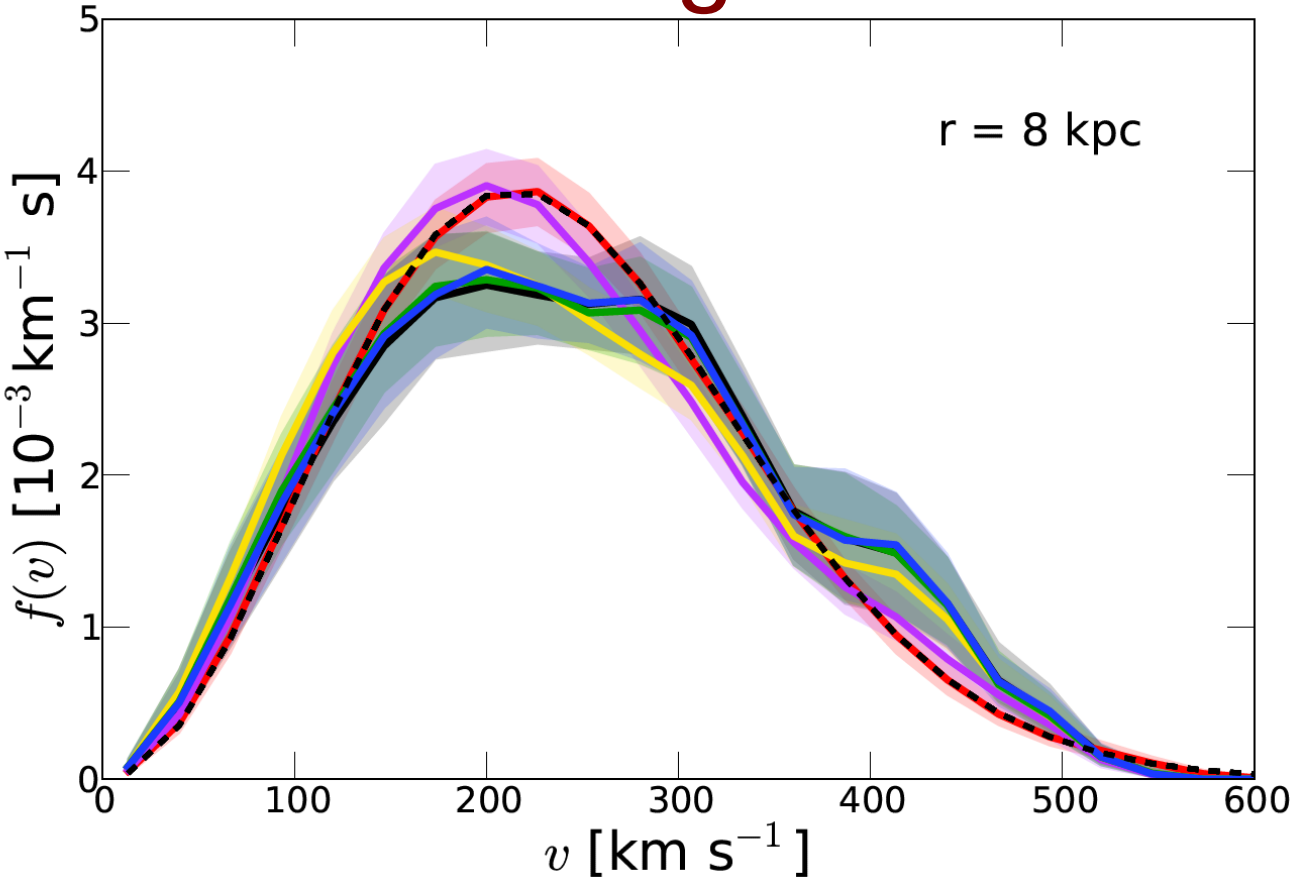
core formation

Self-Interacting DM: Subhalo Structure



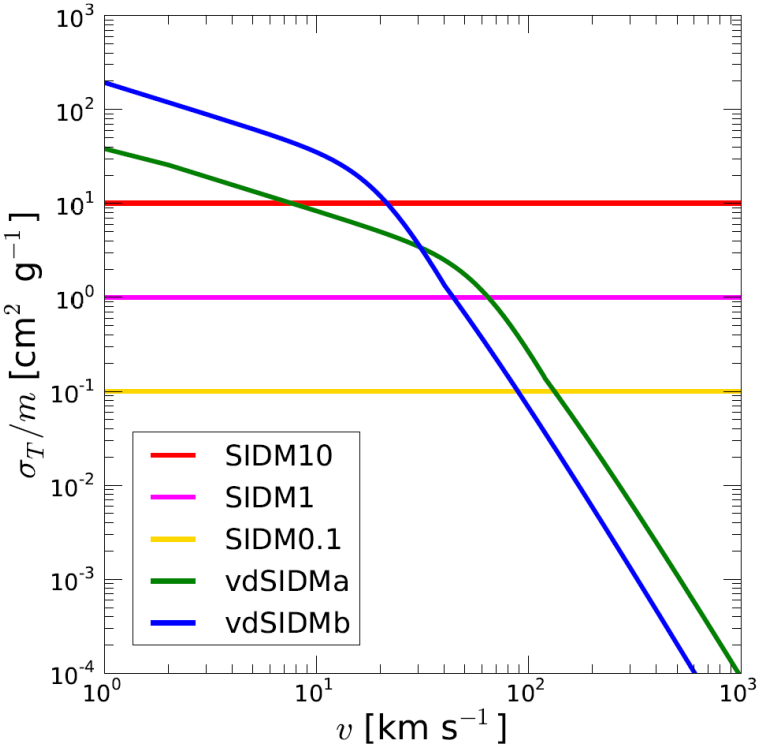
Name	Type	$\sigma_T^{\max}/m_\chi$ [$\mathrm{cm^2\,g^{-1}}$]	$v_{\max}$ [$\mathrm{km\,s^{-1}}$]
RefP0	CDM	/	/
RefP1	SIDM (ruled out)	10	/
RefP2	vdSIDM (allowed)	3.5	30
RefP3	vdSIDM (allowed)	35	10

Self-Interacting DM: Direct Detection



Vogelsberger+ (2013)

Name	$\sigma_T^{\max}/m_\chi[\mathrm{cm^2\,g^{-1}}]$	$v_{\max}[\mathrm{km\,s^{-1}}]$
CDM	/	/
SIDM10	10	/
SIDM1	1	/
SIDM0.1	0.1	/
vdSIDMa	3.5	30
vdSIDMb	35	10



Hands-on

- A cosmological simulation with Gadget-2
- Makefile, Parameter file, Running
- Post-processing

Download the Code: Prepacked Setup

http://www.cfa.harvard.edu/~mvogelsb/handson.tar.gz

→ contains: code, initial conditions, I/O reader, simple post-processing example

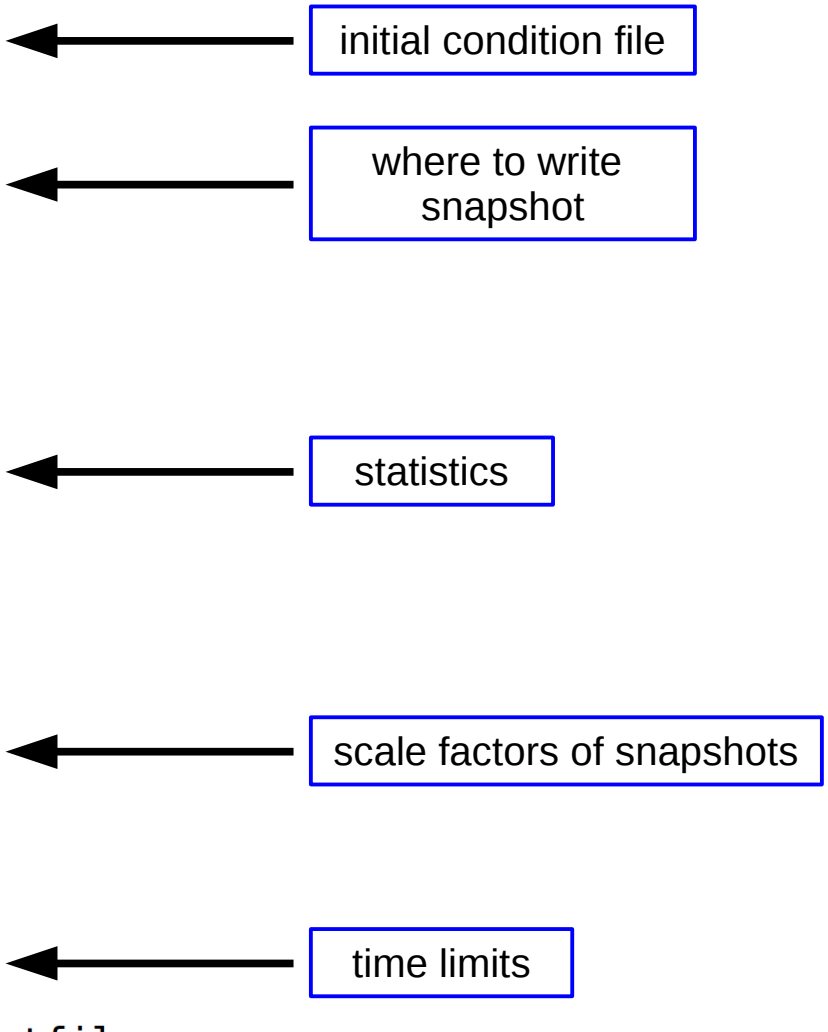
→ requires: C, make, FFTW, GSL, HDF5, MPI, Python (Numpy, PyTables, ...)

Content:

```
drwxr-xr-x 2 mark mark   4096 Jul 28 13:31 ICs
-rw-r-xr-- 1 mark mark   2451 Jul 28 13:31 param.txt
-rw-r--r-- 1 mark mark  13521 Jul 28 13:31 readsnapGadget2.py
drwxr-xr-x 3 mark mark   4096 Jul 28 13:31 code
-rw-r--r-- 1 mark mark   1085 Jul 28 13:31 plot_map.py
-rw-r--r-- 1 mark mark     43 Jul 28 13:31 scalefactors
-rwxr-xr-x 1 mark mark    185 Jul 28 13:31 job.bsub
-rwxr-xr-x 1 mark mark 606780 Jul 28 15:19 Gadget2
-rw-r--r-- 1 mark mark   2575 Jul 28 15:25 param.txt-usedvalues
drwxr-xr-x 2 mark mark   4096 Jul 28 15:46 output
```

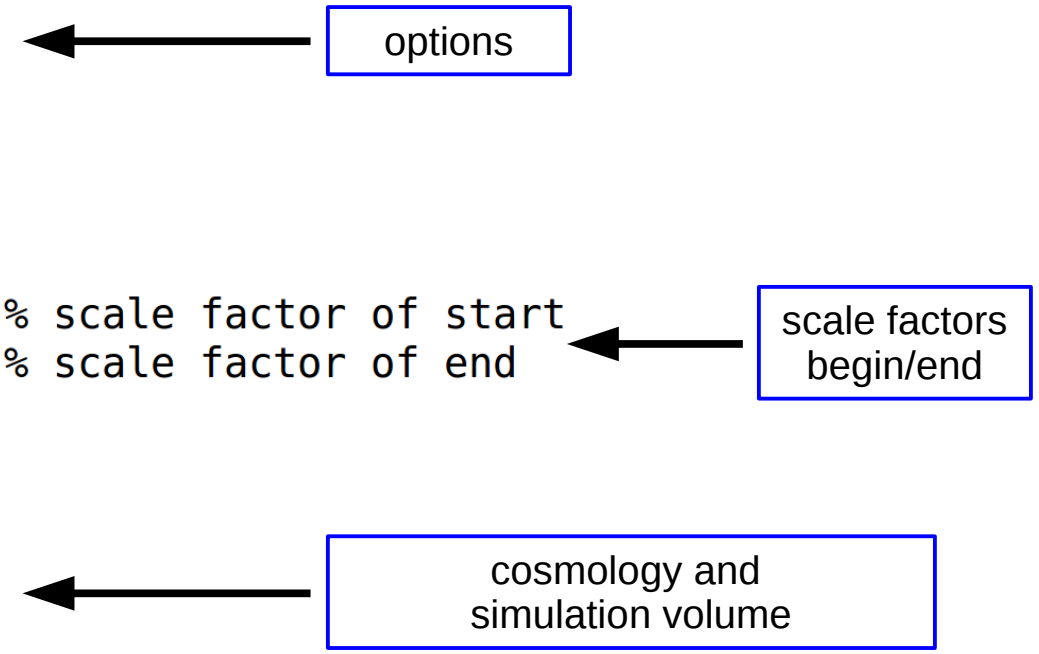
Parameterfile

```
%---initial conditions
InitCondFile       ./ICs/ICs

%---output directory
OutputDir          ./output/

%---simulation statistics
EnergyFile         energy.txt
InfoFile           info.txt
TimingsFile        timings.txt
CpuFile            cpu.txt
RestartFile        restart
SnapshotFileBase   snapshot
TimeBetStatistics  0.05

%---file with output times
OutputListFilename scalefactors

%---CPU time -limit
TimeLimitCPU              86400.0
CpuTimeBetRestartFile     3600.0
ResubmitOn                0
ResubmitCommand           my-scriptfile
```



Parameterfile

```
%---Code options
ICFormat                 1
SnapFormat               3
ComovingIntegrationOn    1
TypeOfTimestepCriterion  0
OutputListOn             1
PeriodicBoundariesOn     1

%---simulation time
TimeBegin           0.01      % scale factor of start
TimeMax             1.0       % scale factor of end

%---cosmology
Omega0                0.27
OmegaLambda           0.73
OmegaBaryon           0
HubbleParam           0.7
BoxSize               50000.0

%---Output frequency
TimeBetSnapshot        1.1
TimeOfFirstSnapshot    1.0
```

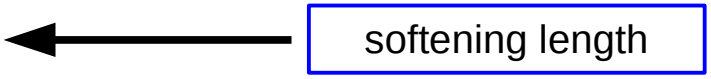


Parameterfile

```
%---output options
NumFilesPerSnapshot       1
NumFilesWrittenInParallel 1

%---Accuracy of time integration
ErrTolIntAccuracy      0.025
MaxRMSDisplacementFac  0.2
CourantFac             0.15
MaxSizeTimestep        0.01
MinSizeTimestep        0

%---tree algorithm, force accuracy, domain update frequency
ErrTolTheta                 0.5
TypeOfOpeningCriterion      1
ErrTolForceAcc              0.005
TreeDomainUpdateFrequency   0.1

%---further parameters of SPH
DesNumNgb              33
MaxNumNgbDeviation     2
ArtBulkViscConst       0.8
InitGasTemp            1000.0
MinGasTemp             50.0
```

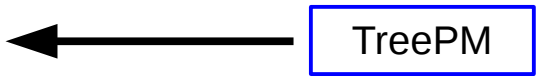
Parameterfile

```
%---Memory allocation
PartAllocFactor       2.0
TreeAllocFactor       0.8
BufferSize            100

%---System of units
UnitLength_in_cm         3.085678e21        ;  1.0 kpc
UnitMass_in_g            1.989e43           ;  1.0e10 solar masses
UnitVelocity_in_cm_per_s 1e5                ;  1 km/sec
GravityConstantInternal  0

% Softening lengths
MinGasHsmlFractional 0.25
SofteningGas       0
SofteningHalo      20.0
SofteningDisk      0
SofteningBulge     0
SofteningStars     0
SofteningBndry     0

SofteningGasMaxPhys       0
SofteningHaloMaxPhys      20.0
SofteningDiskMaxPhys      0
SofteningBulgeMaxPhys     0
SofteningStarsMaxPhys     0
SofteningBndryMaxPhys     0
```



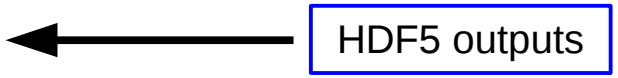
Makefile

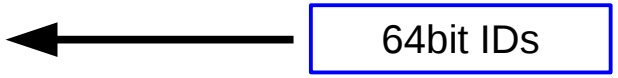
```
#--------------------------------------- Things that are always recommen
OPT   +=  -DPEANOHILBERT
OPT   +=  -DWALLCLOCK


#--------------------------------------- TreePM Options
OPT   +=  -DPMGRID=128
#OPT   +=  -DPLACEHIGHRESREGION=3
#OPT   +=  -DENLARGEREGION=1.2
#OPT   +=  -DASMTH=1.25
#OPT   +=  -DRCUT=4.5


#--------------------------------------- Single/Double Precision
#OPT   +=  -DDOUBLEPRECISION
#OPT   +=  -DDOUBLEPRECISION_FFTW


#--------------------------------------- Time integration options
OPT   +=  -DSYNCHRONIZATION
#OPT   +=  -DFLEXSTEPS
#OPT   +=  -DPSEUDOSYMMETRIC
#OPT   +=  -DNOSTOP_WHEN_BELOW_MINTIMESTEP
#OPT   +=  -DNOPMSTEPADJUSTMENT
```



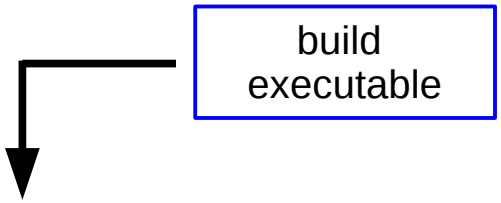
Makefile

```
#--------------------------------------- Output
OPT   +=  -DHAVE_HDF5
#OPT   +=  -DOUTPUTPOTENTIAL
#OPT   +=  -DOUTPUTACCELERATION
#OPT   +=  -DOUTPUTCHANGEOFENTROPY
#OPT   +=  -DOUTPUTTIMESTEP


#--------------------------------------- Things for special behaviour
#OPT   +=  -DNOGRAVITY
#OPT   +=  -DNOTREERND
#OPT   +=  -DNOTYPEPREFIX_FFTW
#OPT   +=  -DLONG_X=60
#OPT   +=  -DLONG_Y=5
#OPT   +=  -DLONG_Z=0.2
#OPT   +=  -DTWODIMS
#OPT   +=  -DSPH_BND_PARTICLES
#OPT   +=  -DNOVISCOSITYLIMITER
#OPT   +=  -DCOMPUTE_POTENTIAL_ENERGY
OPT   +=  -DLONGIDS
#OPT   +=  -DISOTHERM_EQS
#OPT   +=  -DADAPTIVE_GRAVSOFT_FORGAS
#OPT   +=  -DSELECTIVE_NO_GRAVITY=2+4+8+16

#--------------------------------------- Testing and Debugging options
#OPT   +=  -DFORCETEST=0.1


#--------------------------------------- Glass making
#OPT   +=  -DMAKEGLASS=262144
```



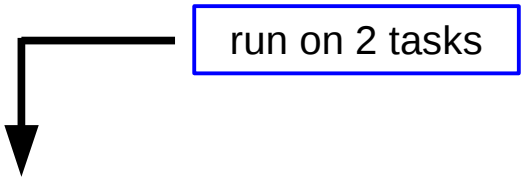


Compiling




```
mark@tux ~/handson/code/Gadget-2.0.7/Gadget2 $ make
mpicc -O2    -g -Wall -Wno-unused-but-set-variable -Wno-uninitialized -Wno-form
at-security -Wno-unused-result -Wno-format -DPERIODIC  -DUNEQUALSOFTENINGS -DPE
ANOHILBERT -DWALLCLOCK    -DPMGRID=128 -DSYNCHRONIZATION -DHAVE_HDF5   -DLONGID
S -DX86FIX   -DH5_USE_16_API   -c -o main.o main.c
mpicc -O2    -g -Wall -Wno-unused-but-set-variable -Wno-uninitialized -Wno-form
at-security -Wno-unused-result -Wno-format -DPERIODIC  -DUNEQUALSOFTENINGS -DPE
ANOHILBERT -DWALLCLOCK    -DPMGRID=128 -DSYNCHRONIZATION -DHAVE_HDF5   -DLONGID
S -DX86FIX   -DH5_USE_16_API   -c -o run.o run.c
mpicc -O2    -g -Wall -Wno-unused-but-set-variable -Wno-uninitialized -Wno-form
at-security -Wno-unused-result -Wno-format -DPERIODIC  -DUNEQUALSOFTENINGS -DPE
ANOHILBERT -DWALLCLOCK    -DPMGRID=128 -DSYNCHRONIZATION -DHAVE_HDF5   -DLONGID
S -DX86FIX   -DH5_USE_16_API   -c -o predict.o predict.c

•••

mpicc -O2    main.o  run.o  predict.o begrun.o endrun.o global.o timestep.o  in
it.o restart.o  io.o accel.o   read_ic.o  ngb.o system.o  allocate.o  density.o
 gravtree.o hydra.o  driftfac.o domain.o  allvars.o potential.o forcetree.o   p
eano.o gravtree_forcetest.o pm_periodic.o pm_nonperiodic.o longrange.o  -lhdf5
-g     -lgsl -lgslcblas -lm  -lsrfftw_mpi -lsfftw_mpi -lsrfftw -lsfftw    -o  G
adget2
mark@tux ~/handson/code/Gadget-2.0.7/Gadget2 $ ls Gadget2
Gadget2
mark@tux ~/handson/code/Gadget-2.0.7/Gadget2 $ cp Gadget2 ../../../
mark@tux ~/handson/code/Gadget-2.0.7/Gadget2 $ cd ../../../
mark@tux ~/handson $ ls Gadget2
Gadget2
mark@tux ~/handson $
```

Running




```
mark@tux ~/handson $ mpirun -np 2 ./Gadget2 param.txt

This is Gadget, version `2.0'.

Running on 2 processors.

found 6 times in output-list.

Allocated 100 MByte communication buffer per processor.

Communication buffer has room for 2016492 particles in gravity computation
Communication buffer has room for 819200 particles in density computation
Communication buffer has room for 655360 particles in hydro computation
Communication buffer has room for 582542 particles in domain decomposition


Hubble (internal units) = 0.1
G (internal units) = 43007.1
UnitMass_in_g = 1.989e+43
UnitTime_in_s = 3.08568e+16
UnitVelocity_in_cm_per_s = 100000
UnitDensity_in_cgs = 6.76991e-22

●●●
```

Running

```
Begin Step 1, Time: 0.0100904, Redshift: 98.1046, Systemstep: 9.03504e-05, Dlog
a: 0.00899447
domain decomposition...
NTopleaves= 64
work-load balance=1.00424   memory-balance=1.00227
domain decomposition done.
begin Peano-Hilbert order...
Peano-Hilbert done.
Start force computation...
Starting periodic PM calculation.
done PM.
Tree construction.
Tree construction done.
Begin tree force.
tree is done.
force computation done.

...
```

output for each Integration times step

creates output files 'snapshots'

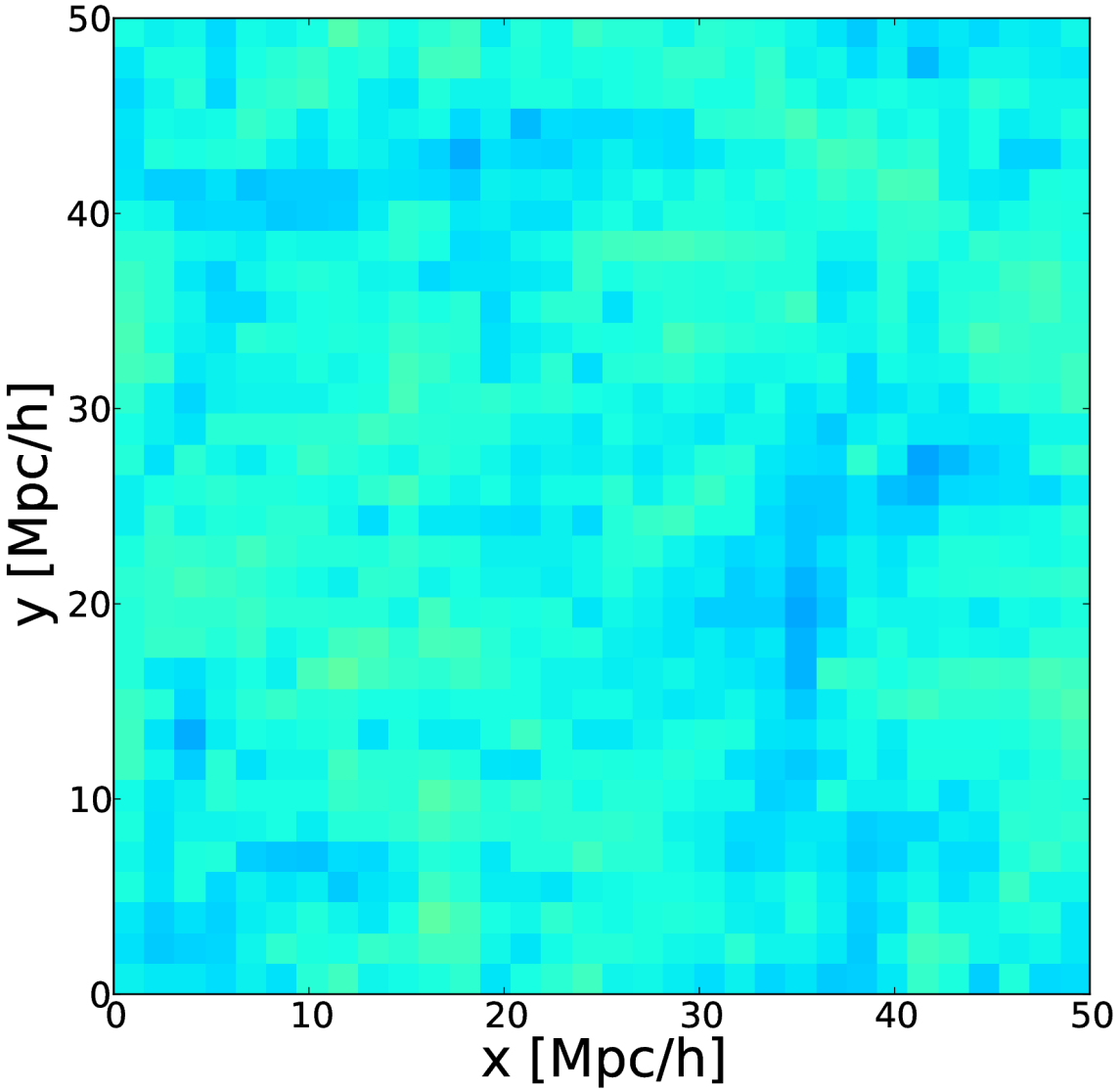
HDF5 snapshots

```
mark@tux ~/handson/output $ ls -ltr
total 112200
-rw-r--r-- 1 mark mark  8395152 Jul 28 15:25 snapshot_000.hdf5
-rw-r--r-- 1 mark mark  8395152 Jul 28 15:30 snapshot_001.hdf5
-rw-r--r-- 1 mark mark  8395152 Jul 28 15:37 snapshot_002.hdf5
-rw-r--r-- 1 mark mark  8395152 Jul 28 15:40 snapshot_003.hdf5
-rw-r--r-- 1 mark mark  8395152 Jul 28 15:42 snapshot_004.hdf5
-rw-r--r-- 1 mark mark  8395152 Jul 28 15:46 snapshot_005.hdf5
-rw-r--r-- 1 mark mark 25865824 Jul 28 15:46 restart.0
-rw-r--r-- 1 mark mark 30225184 Jul 28 15:46 restart.1
-rw-r--r-- 1 mark mark  8395152 Jul 28 15:46 snapshot_006.hdf5
-rw-r--r-- 1 mark mark     2575 Jul 28 16:42 parameters-usedvalues
-rw-r--r-- 1 mark mark       84 Jul 28 16:42 energy.txt
-rw-r--r-- 1 mark mark     1382 Jul 28 16:42 timings.txt
-rw-r--r-- 1 mark mark      441 Jul 28 16:42 info.txt
-rw-r--r-- 1 mark mark     1150 Jul 28 16:42 cpu.txt
```

Simple Post-Processing

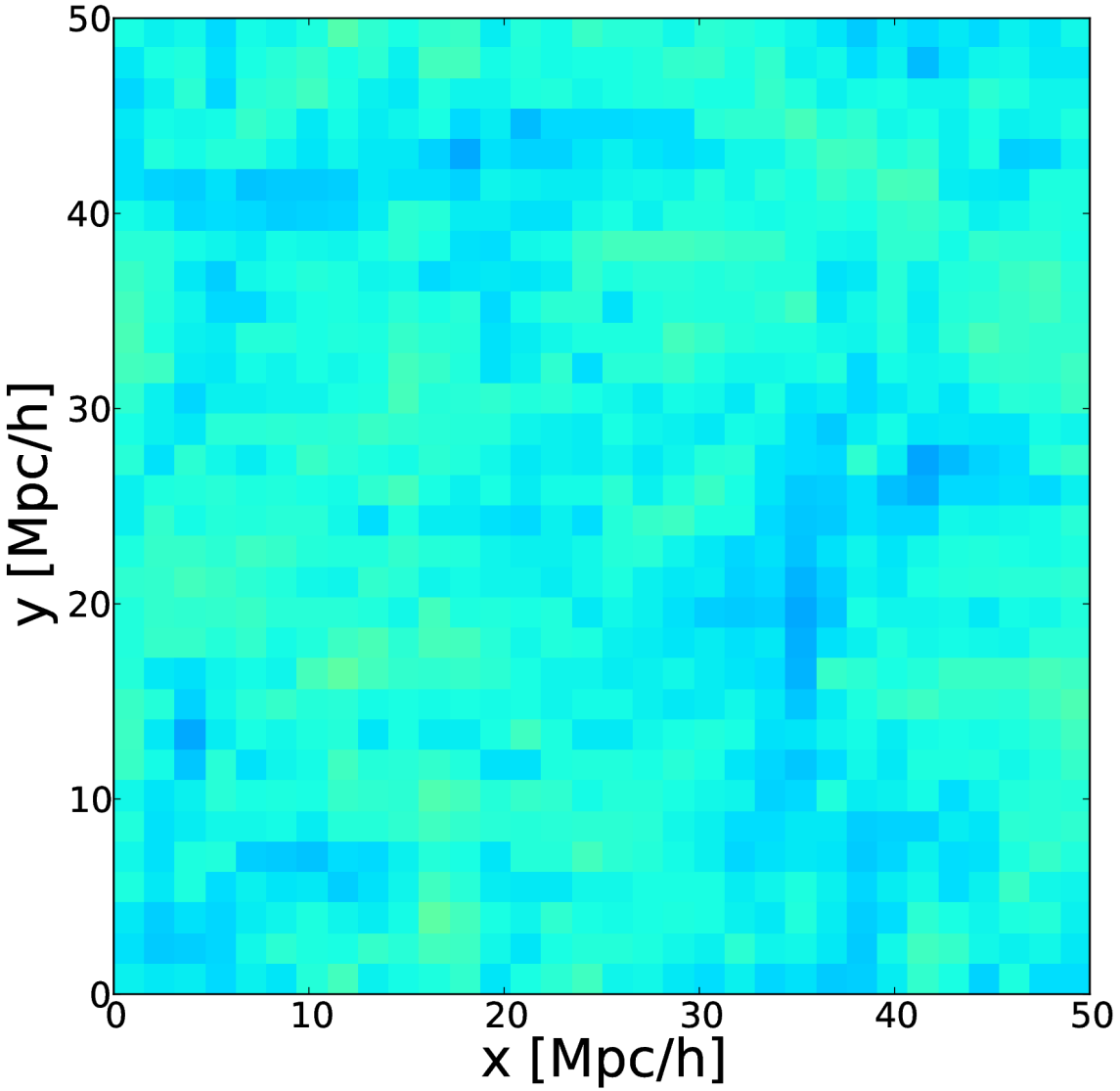
```
import matplotlib.pyplot as plt
import numpy as np
import readsnapGadget2 as snap

#select the snapshot
snapnum = 6

#construct filename
filename="./output/snapshot_"+str(snapnum).zfill(3)

#read header of snapshot
head=snap.snapshot_header(filename)

#print redshift of snapshot
print "z=",head.redshift

#read positions and masses
pos = snap.read_block(filename, "POS ")
mass = snap.read_block(filename, "MASS")

#histogram
h,x,y=np.histogram2d(pos[:,0], pos[:,1], bins=32, normed=False, range=[[0,head.boxsize],[0,head.boxsize]])

#min/max of map
print "min/max histogram", h.min(),h.max()
hmin=60.0
hmax=2900.0

#clip
h[h<hmin]=hmin
h[h>hmax]=hmax
h=np.log10(h)
hmin=np.log10(hmin)
hmax=np.log10(hmax)

#create figure
fig = plt.figure(1, figsize=(10.0,10.0))
ax = fig.add_subplot(1,1,1)
extent = [0, head.boxsize/1000.0, 0, head.boxsize/1000.0]
im = ax.imshow(h.T, origin='lower', vmin=hmin, vmax=hmax, interpolation='nearest', cmap='jet', extent=extent)
ax.set_xlabel("x [Mpc/h]", fontsize=30)
ax.set_ylabel("y [Mpc/h]", fontsize=30)
ax.tick_params(axis='both', labelsize=20)
plt.savefig("map_"+str(snapnum).zfill(3)+".pdf", bbox_inches='tight')
fig.clf()
```

Simple Post-Processing: DM Density Map



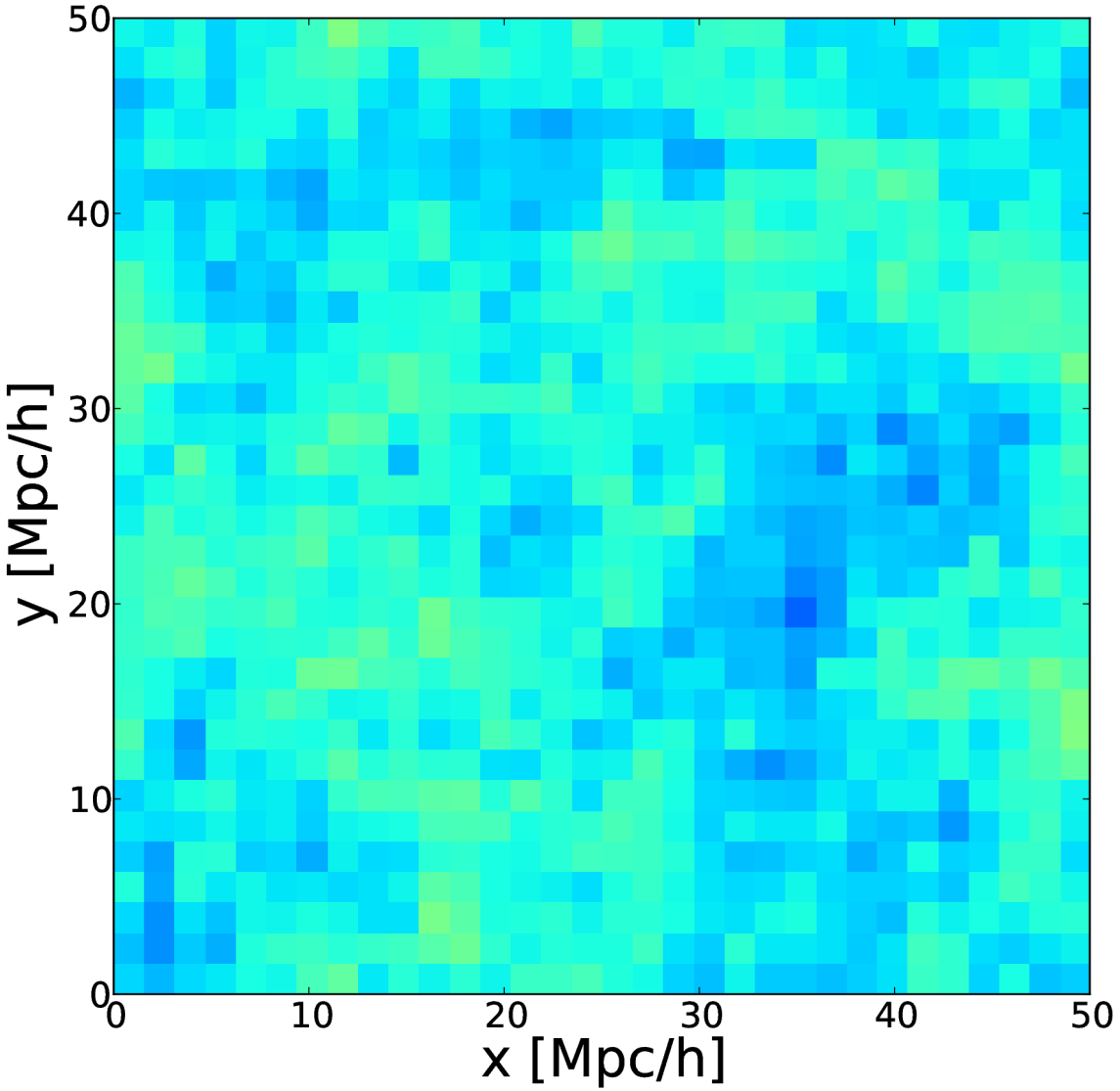
Z = 90

Simple Post-Processing: DM Density Map



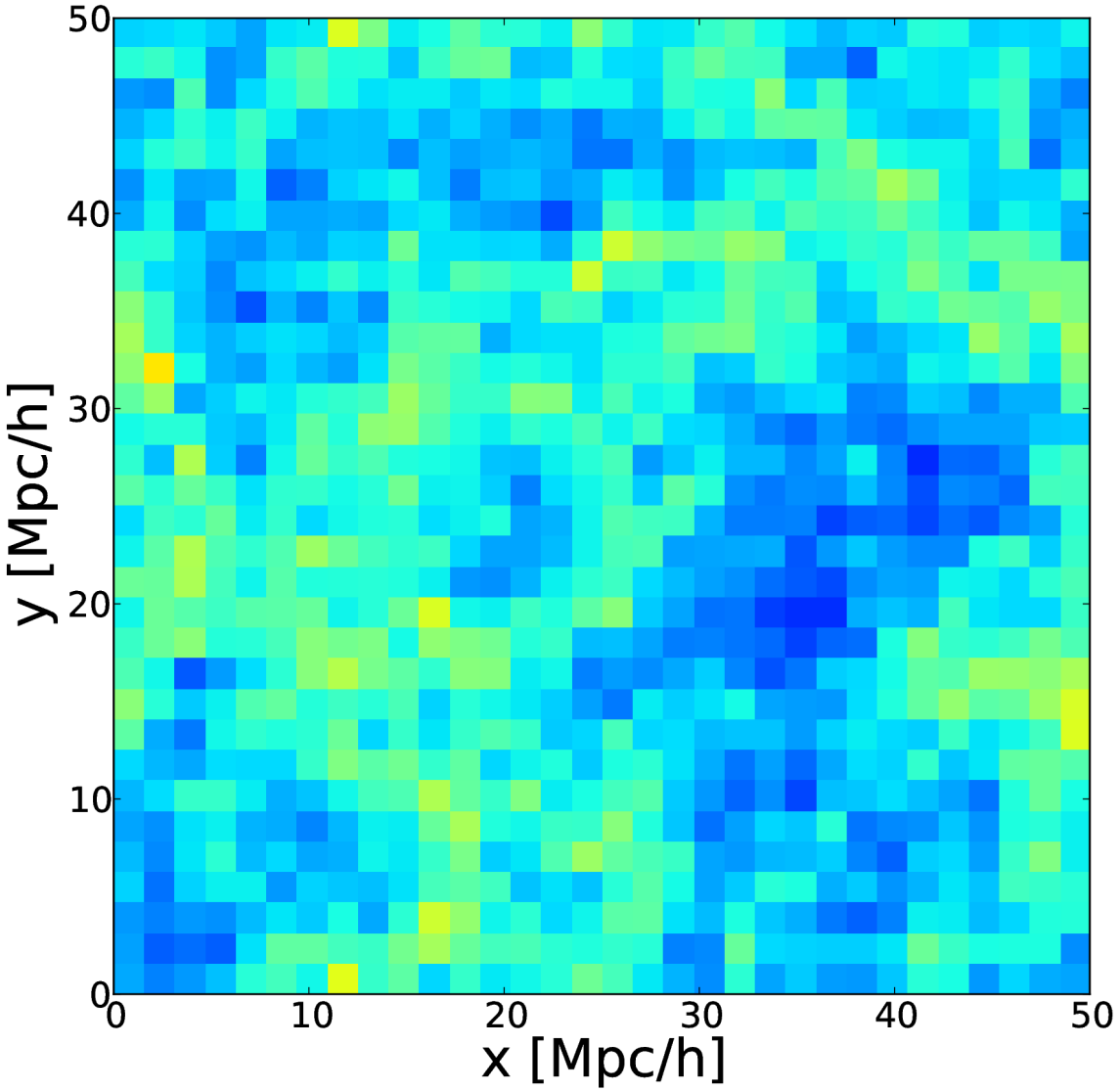
Z = 30

Simple Post-Processing: DM Density Map



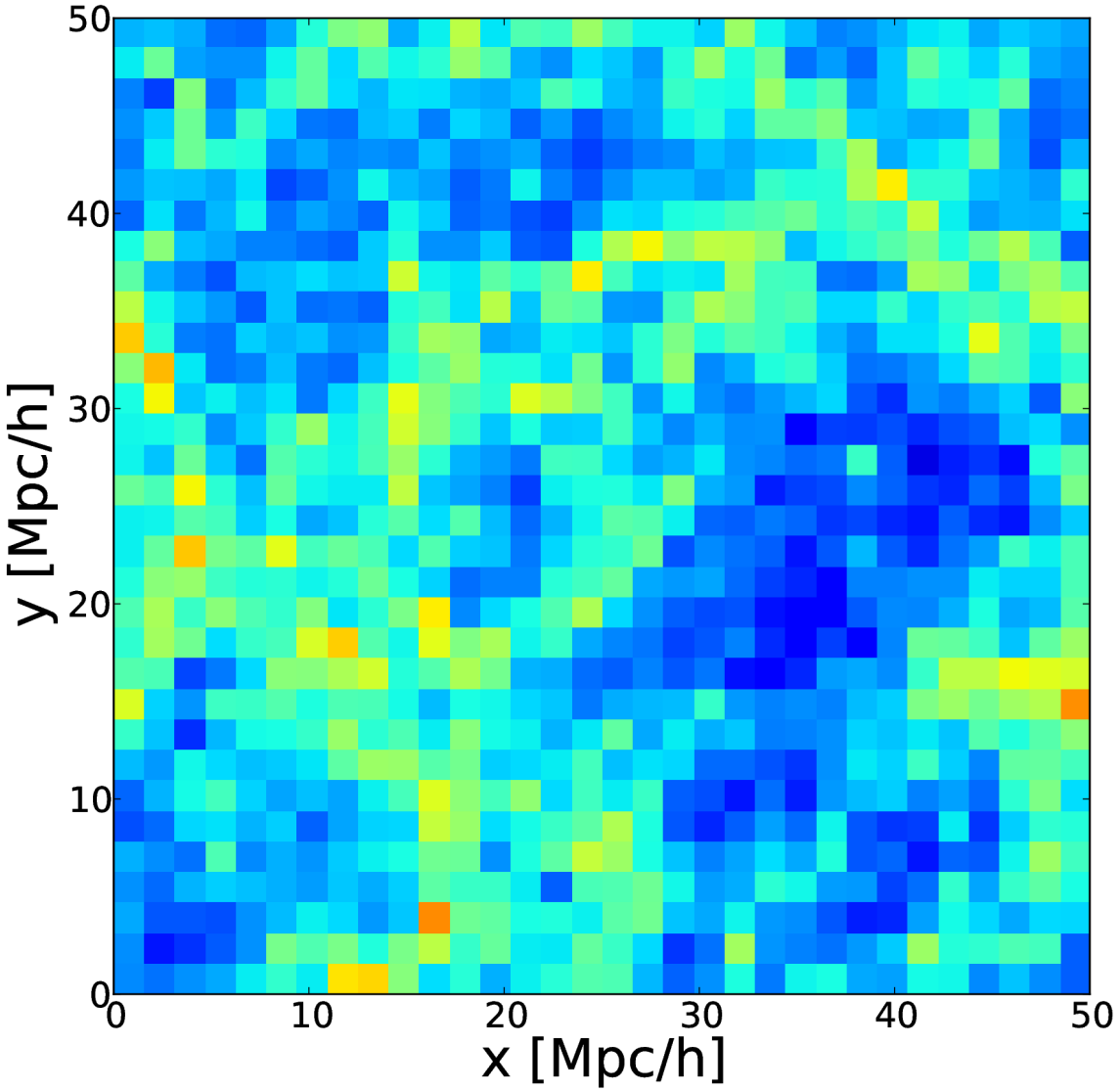
Z = 5

Simple Post-Processing: DM Density Map



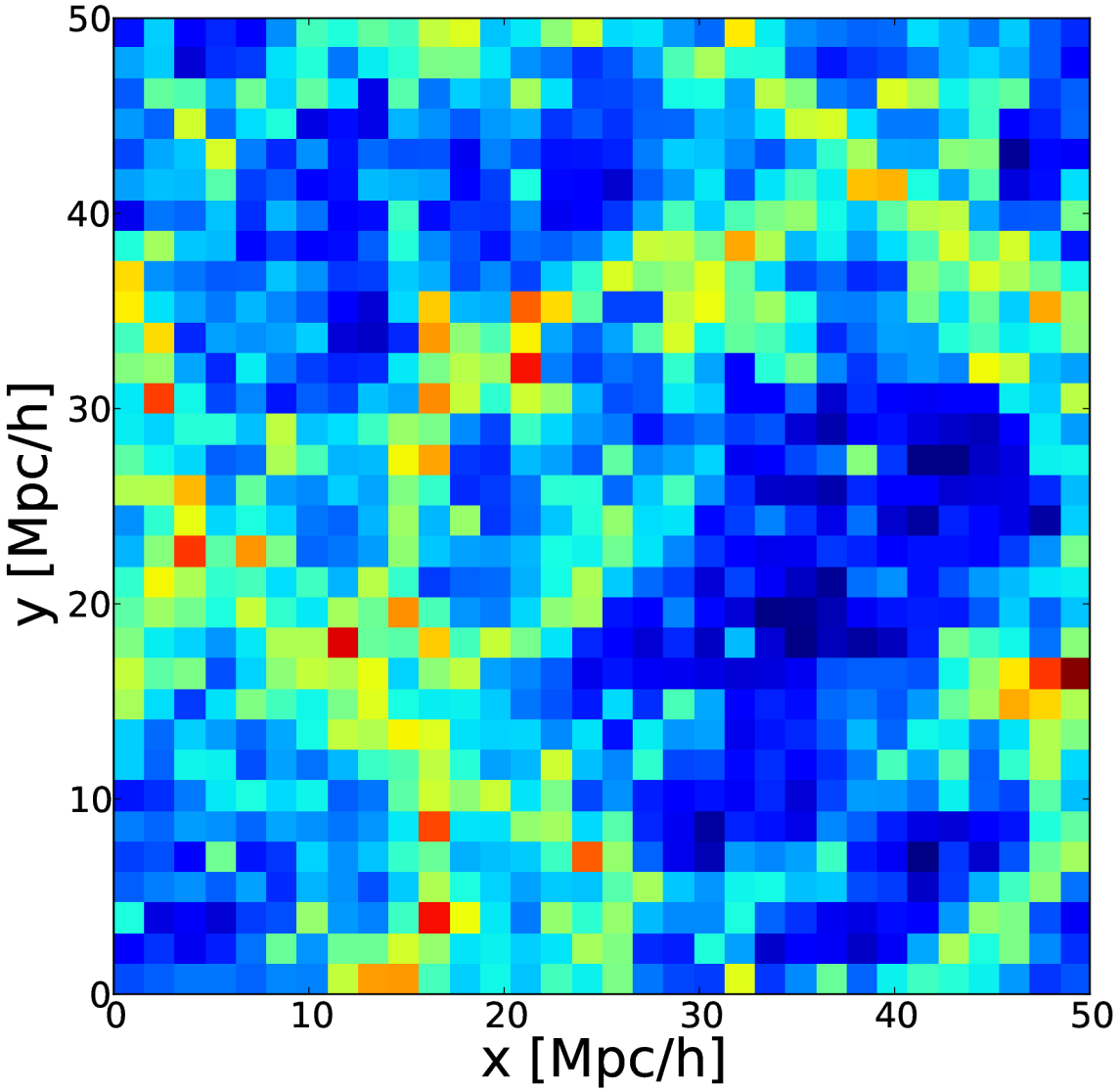
Z = 2

Simple Post-Processing: DM Density Map



Z = 1

Simple Post-Processing: DM Density Map



Z = 0

Illustris Project

- **(75 Mpc/h)^3 volume with 2 x 1820^3 particles/cells**

- **Three simulations:**
  DM only (done), non-radiative, full physics (@ z=0.4)

MV+ (in prep)
Genel+ (in prep)
Sijacki+ (in prep)