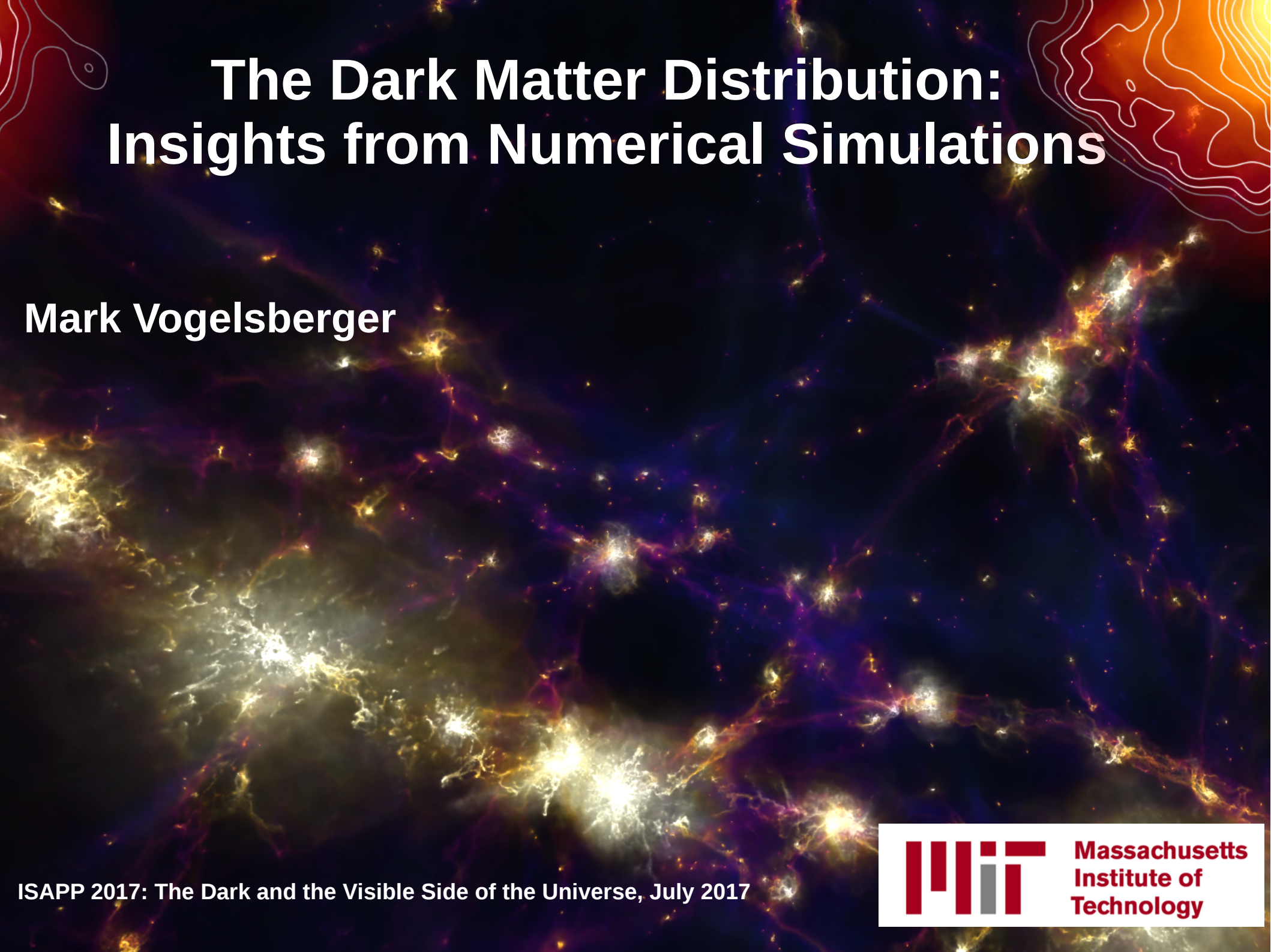
The Dark Matter Distribution: Insights from Numerical Simulations

Mark Vogelsberger



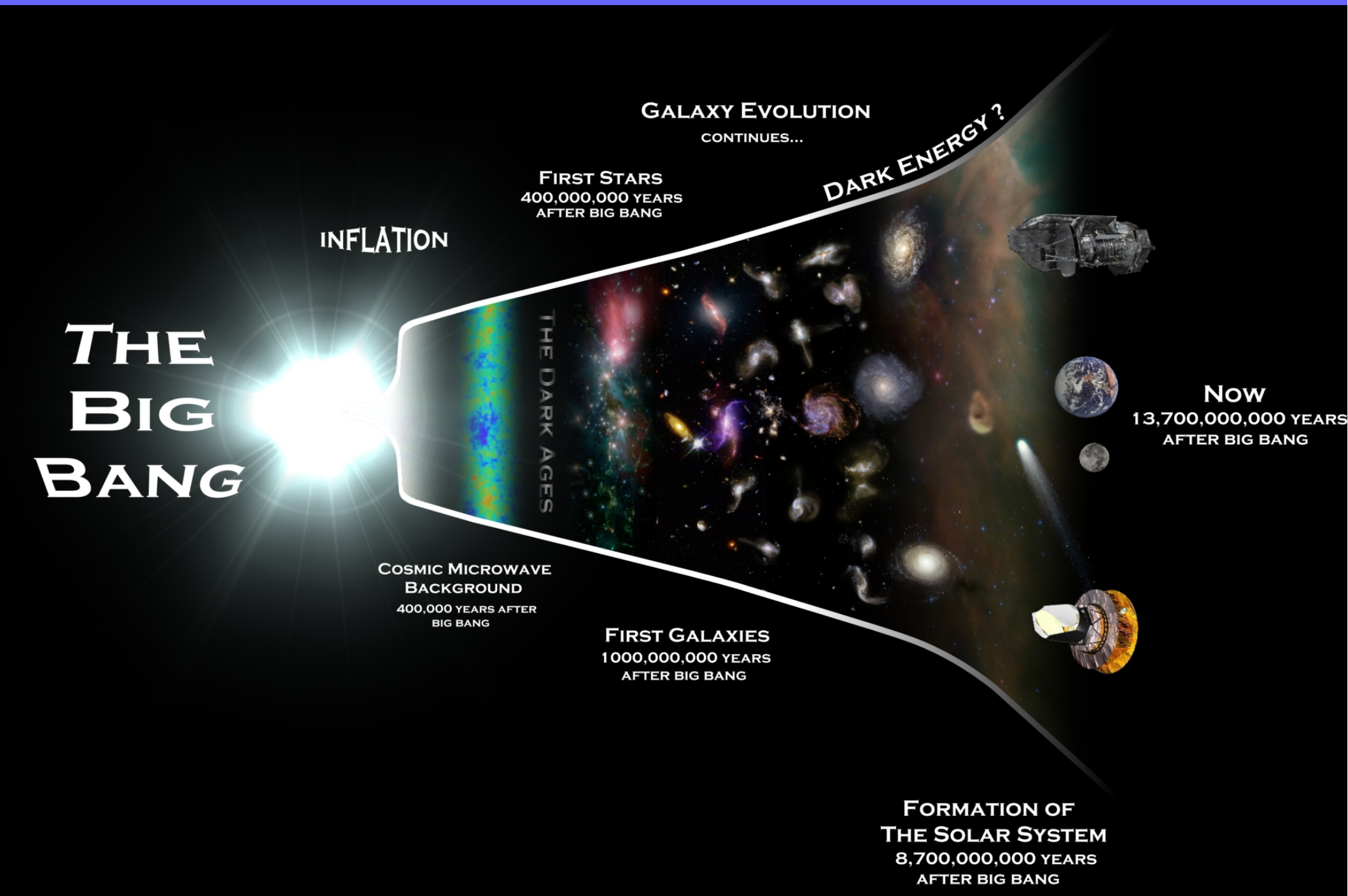
ISAPP 2017: The Dark and the Visible Side of the Universe, July 2017

Massachusetts Institute of Technology

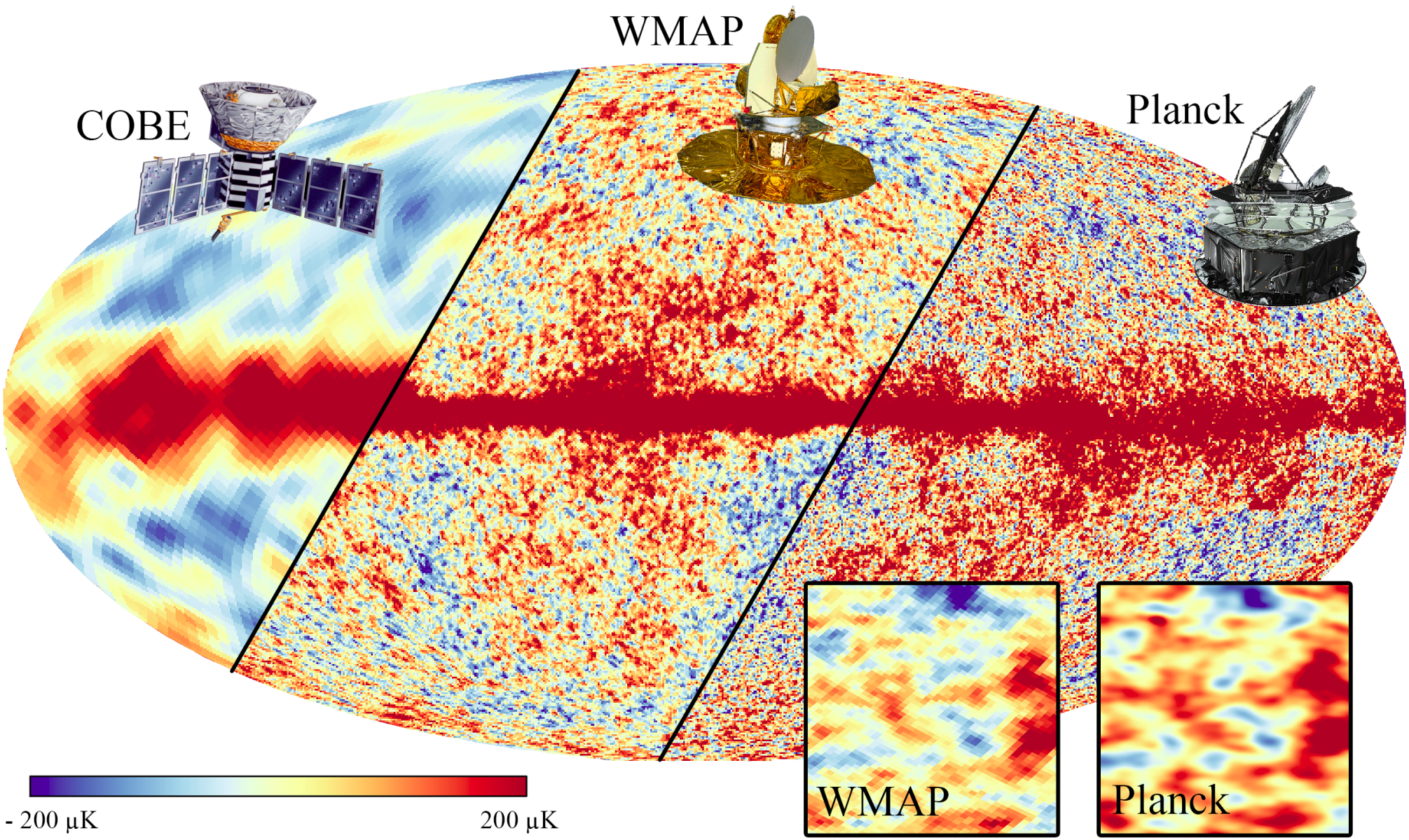
Content

Predicting the DM Distribution with Simulations

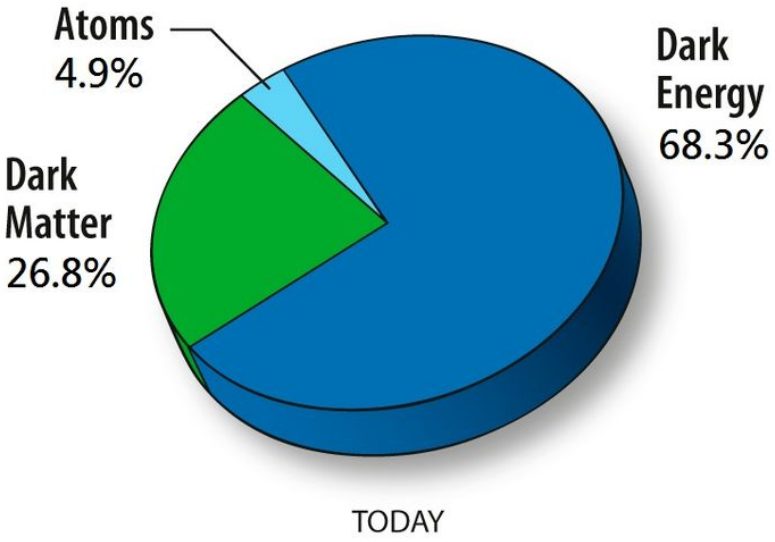
General Framework of Structure Formation



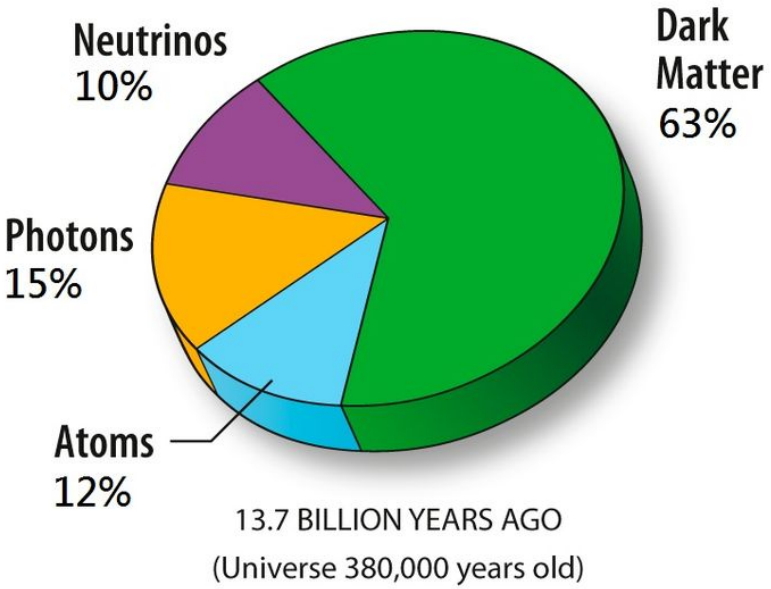
The Initial Conditions for Simulations

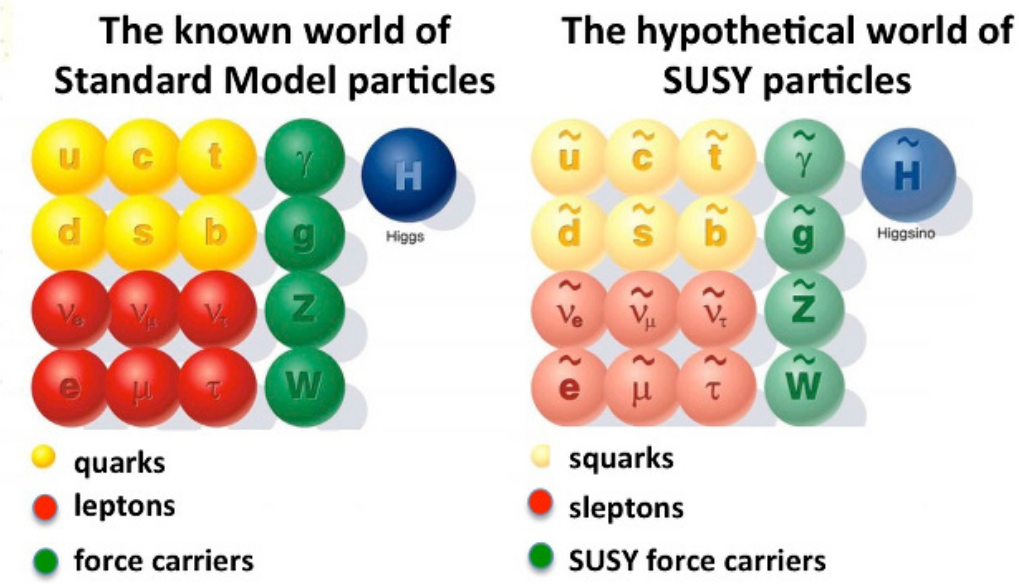


The Composition of the Universe

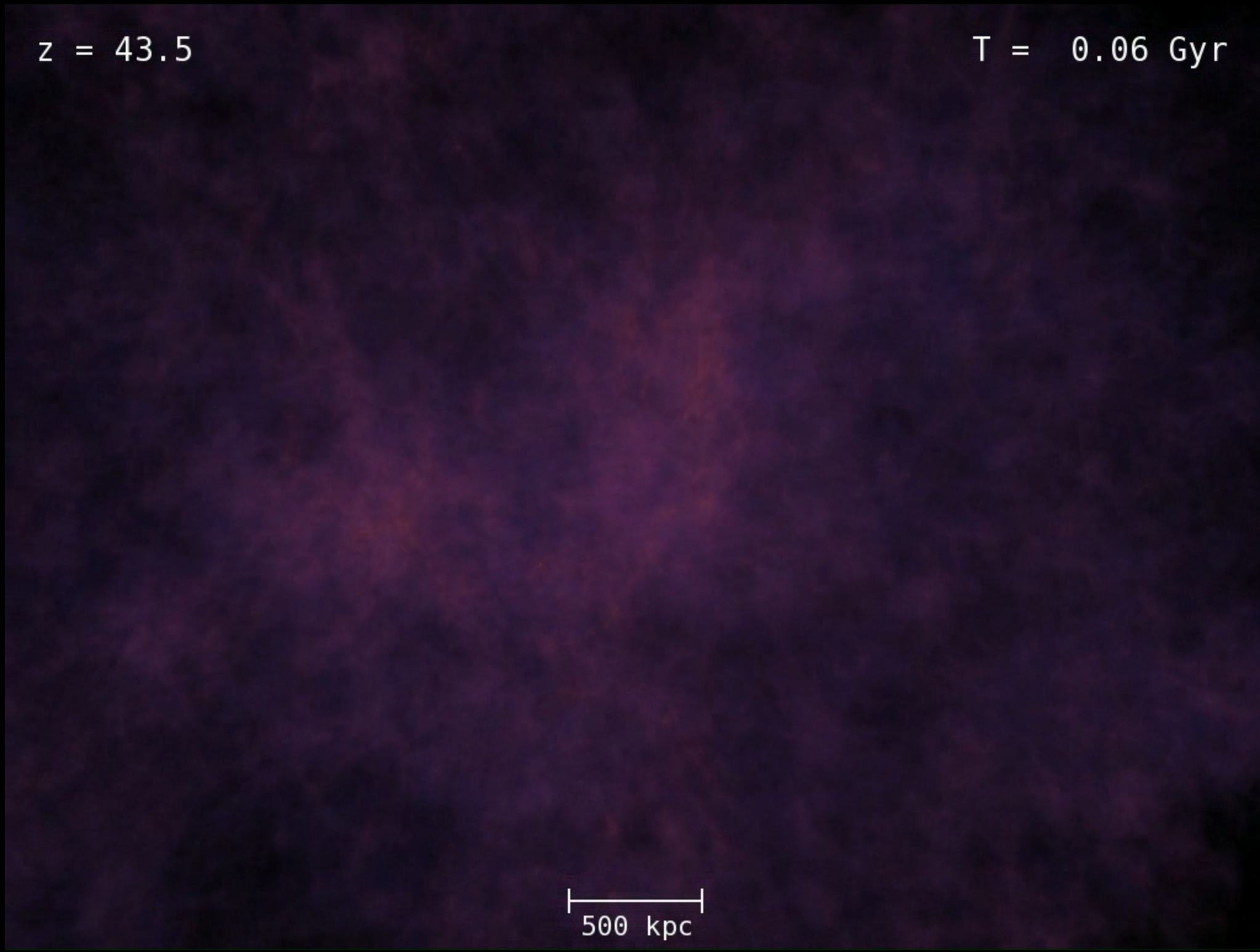


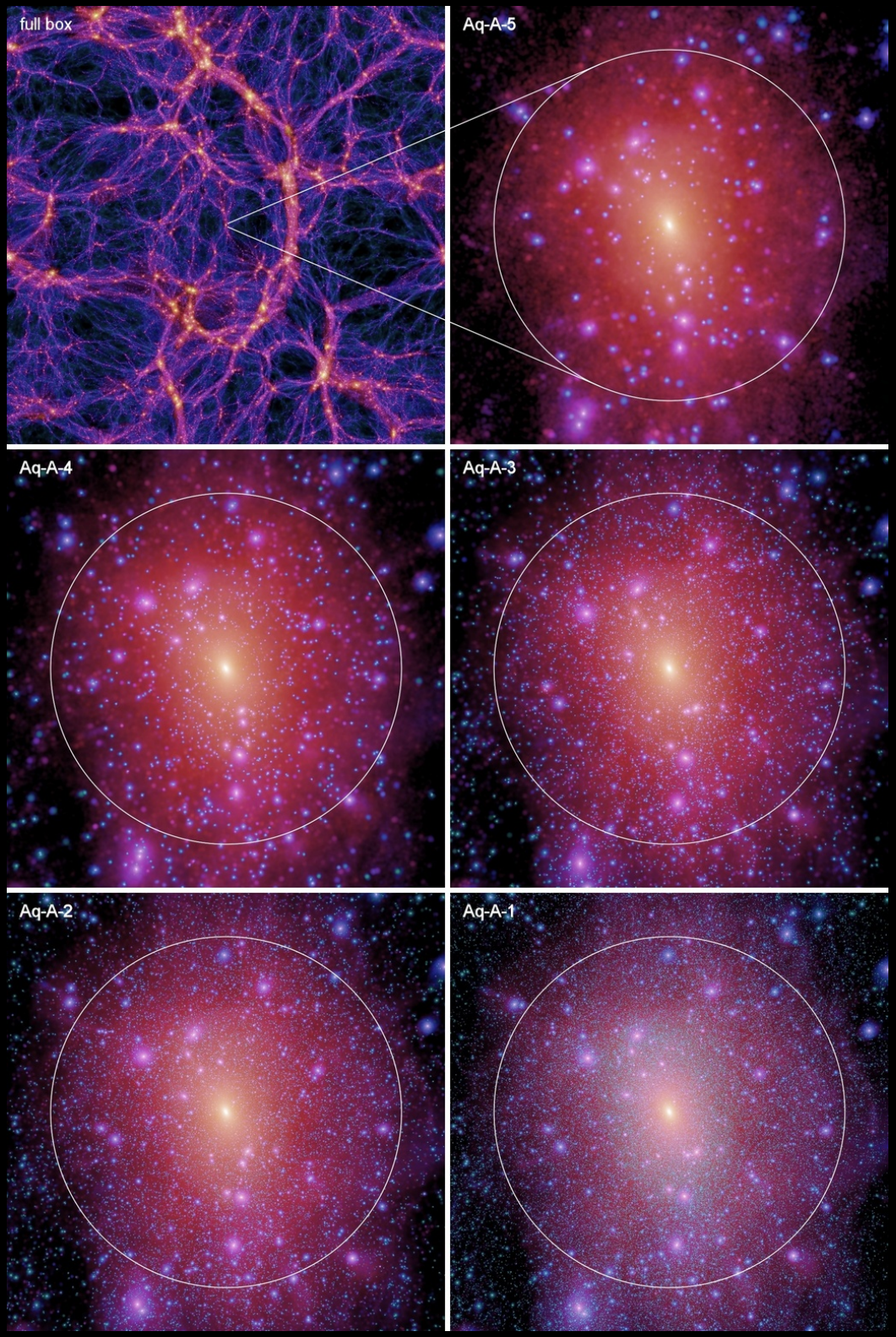
input for structure formation simulations: model DM, DE, atoms





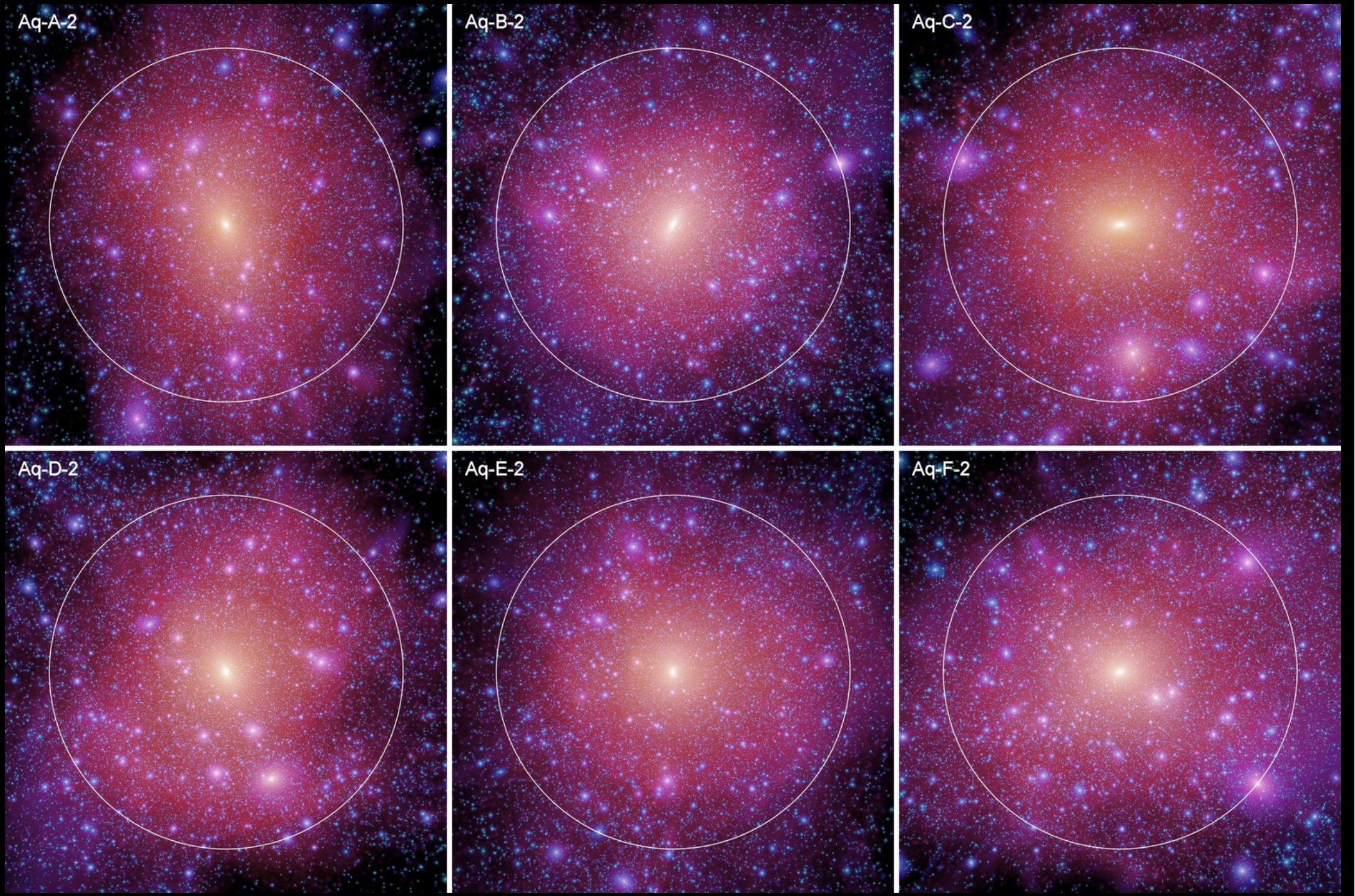
Simulations of Milky Way-like DM Halos


z = 43.5
T = 0.06 Gyr
500 kpc



zoom-in simulations of an individual Milky Way-like DM halo: different resolution levels

Springel+ 2008



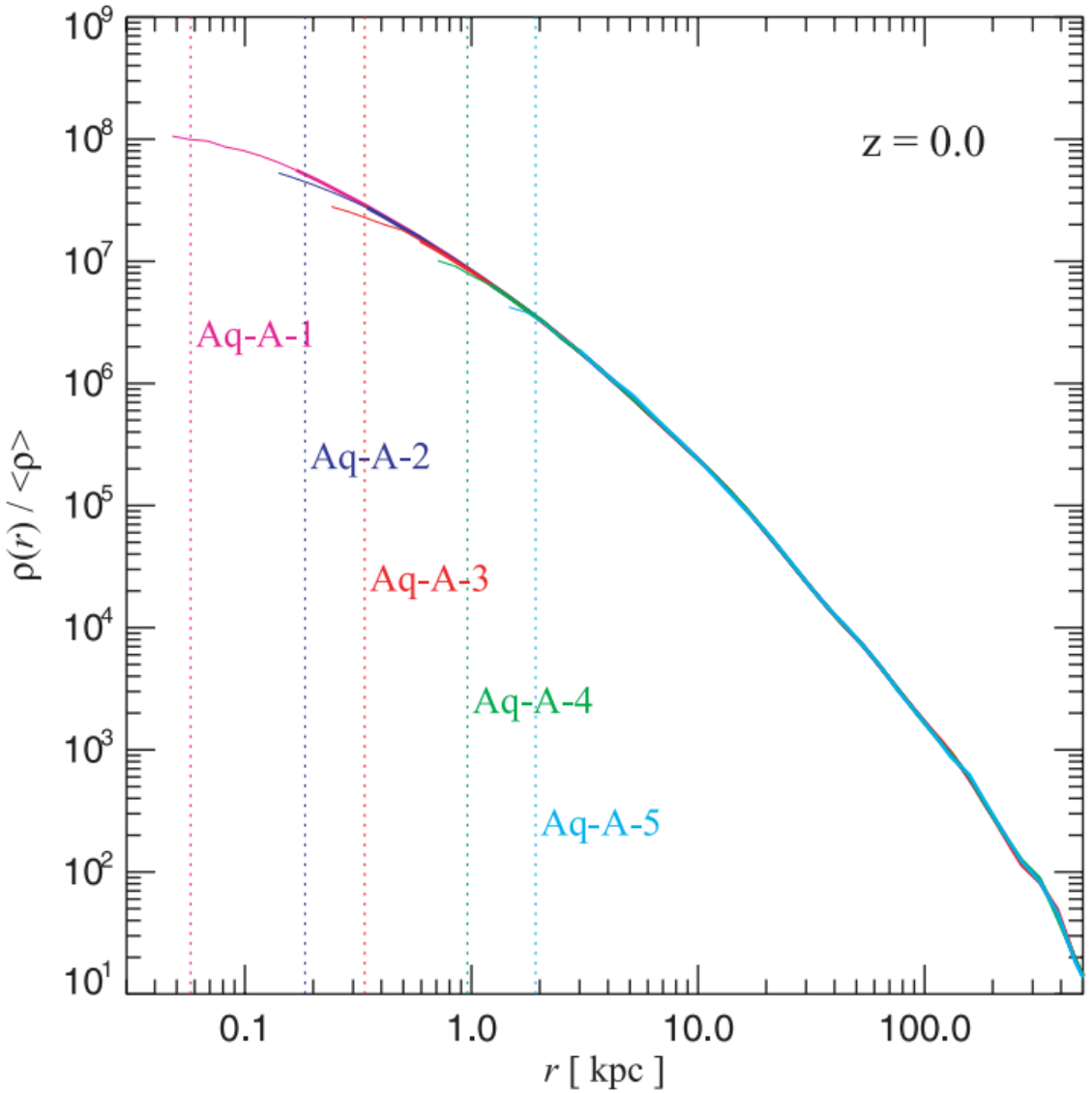
Springel+ 2008

Simulation Details

Name	m_p ($M_\odot$)	ϵ (pc)	N_{hr}	N_{lr}	M_{200} ($M_\odot$)	r_{200} (kpc)	M_{50} ($M_\odot$)	r_{50} (kpc)	N_{50}
Aq-A-1	1.712×10^3	20.5	4252 607 000	144 979 154	1.839×10^{12}	245.76	2.523×10^{12}	433.48	1473 568 512
Aq-A-2	1.370×10^4	65.8	531 570 000	75 296 170	1.842×10^{12}	245.88	2.524×10^{12}	433.52	184 243 536
Aq-A-3	4.911×10^4	120.5	148 285 000	20 035 279	1.836×10^{12}	245.64	2.524×10^{12}	433.50	51 391 468
Aq-A-4	3.929×10^5	342.5	18 535 972	634 793	1.838×10^{12}	245.70	2.524×10^{12}	433.52	6424 399
Aq-A-5	3.143×10^6	684.9	2316 893	634 793	1.853×10^{12}	246.37	2.541×10^{12}	434.50	808 479
Aq-B-2	6.447×10^3	65.8	658 815 010	80 487 598	8.194×10^{11}	187.70	1.045×10^{12}	323.12	162 084 992
Aq-B-4	2.242×10^5	342.5	18 949 101	648 874	8.345×10^{11}	188.85	1.050×10^{12}	323.60	4683 037
Aq-C-2	1.399×10^4	65.8	612 602 795	78 634 854	1.774×10^{12}	242.82	2.248×10^{12}	417.09	160 630 624
Aq-C-4	3.213×10^5	342.5	26 679 146	613 141	1.793×10^{12}	243.68	2.285×10^{12}	419.36	7110 775
Aq-D-2	1.397×10^4	65.8	391 881 102	79 615 274	1.774×10^{12}	242.85	2.519×10^{12}	433.21	180 230 512
Aq-D-4	2.677×10^5	342.4	20 455 156	625 272	1.791×10^{12}	243.60	2.565×10^{12}	435.85	9579 672
Aq-E-2	9.593×10^3	65.8	465 905 916	74 119 996	1.185×10^{12}	212.28	1.548×10^{12}	368.30	161 323 676
Aq-E-4	2.604×10^5	342.5	17 159 996	633 106	1.208×10^{12}	213.63	1.558×10^{12}	369.14	5982 797
Aq-F-2	6.776×10^3	65.8	414 336 000	712 839	1.135×10^{12}	209.21	1.517×10^{12}	365.87	223 901 216
Aq-F-3	2.287×10^4	120.5	122 766 400	712 839	1.101×10^{12}	207.15	1.494×10^{12}	363.98	65 320 572

Springel+ 2008

Spherically Averaged Density Profiles of DM Halos

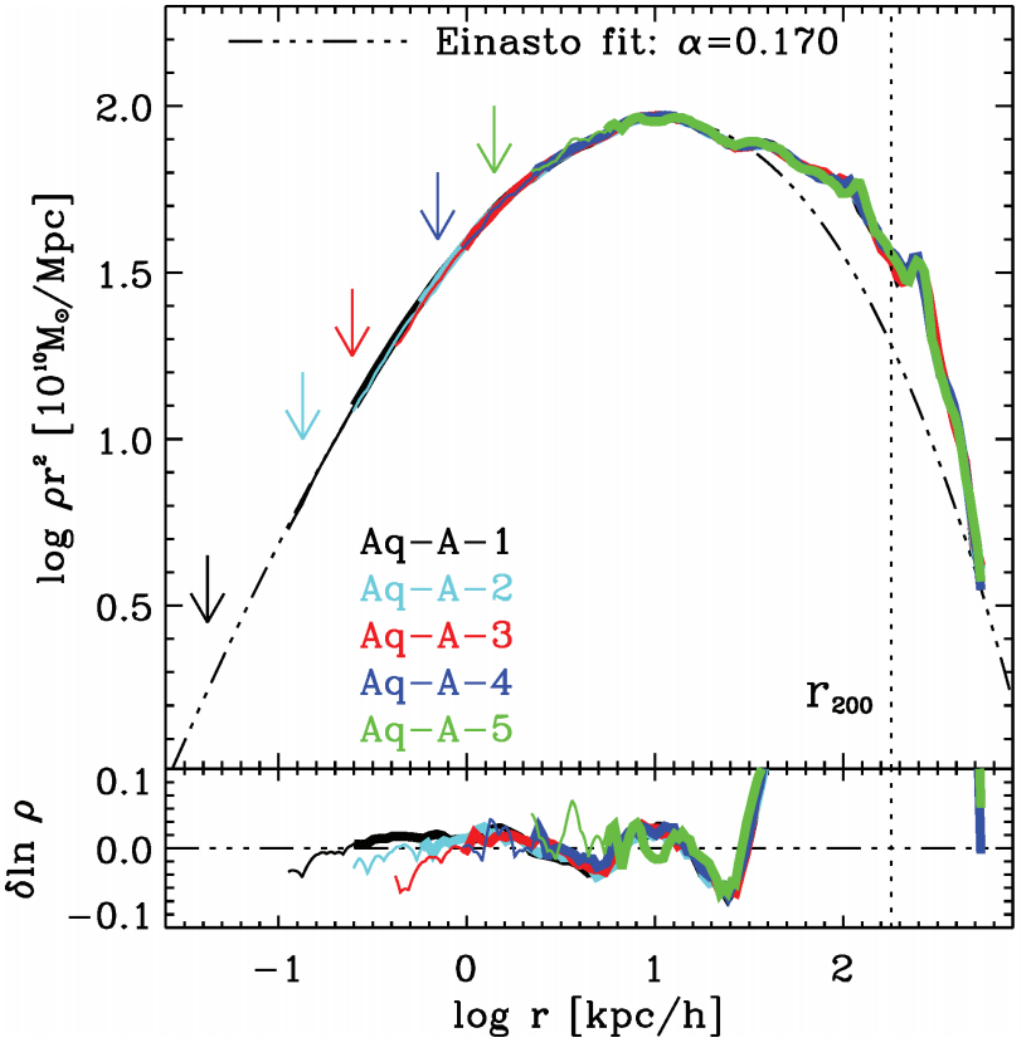


DM halos have a nearly universal spherically averaged density profile: -1 to -3 logarithmic slope

very well converged

Springel+ 2008

Density Profiles: Empirical Fitting Functions



NFW:

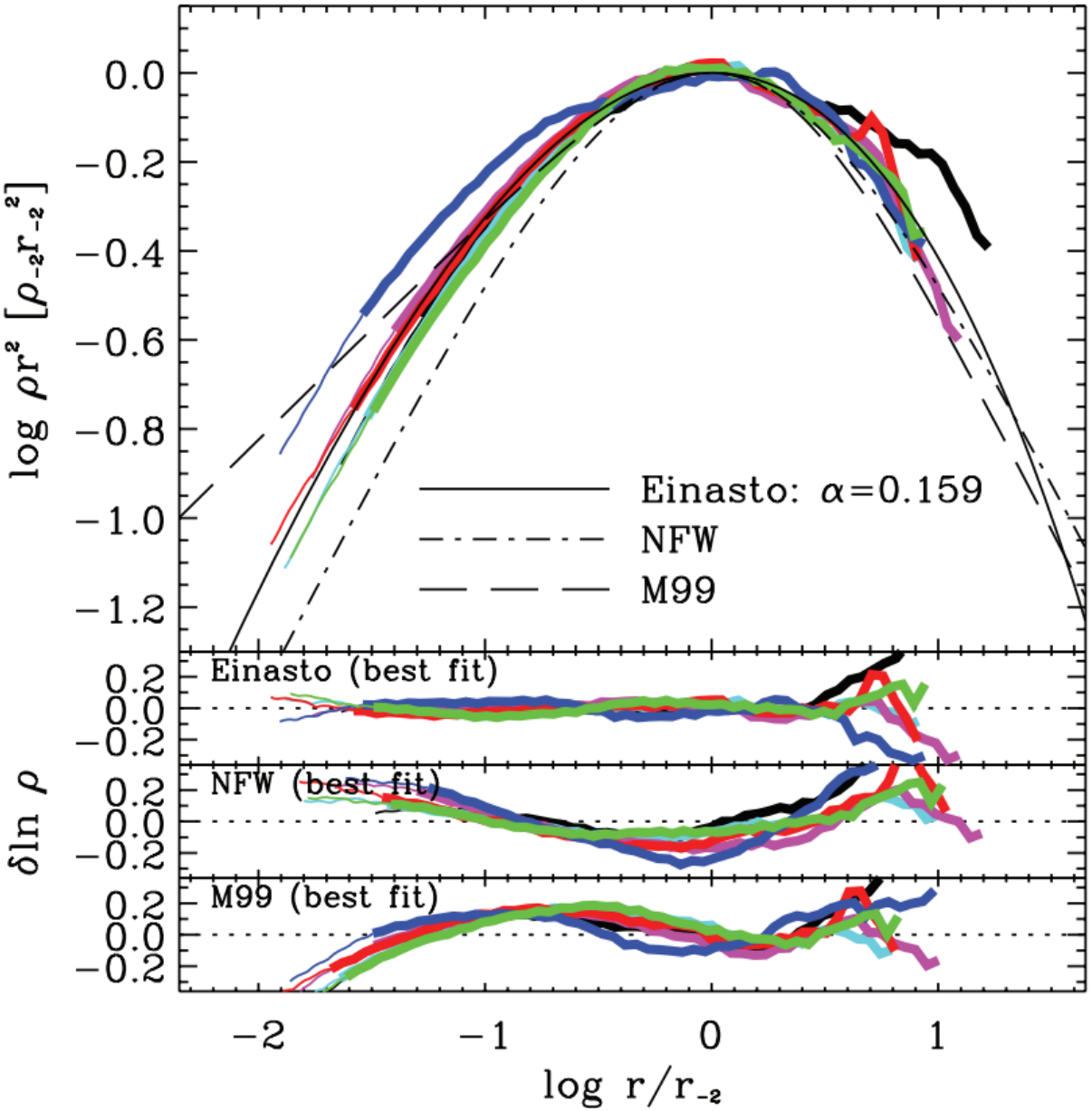
$$\rho(r) = \frac{\rho_s}{(r/r_s)(1 + r/r_s)^2}$$

Einasto:

$$\ln[\rho(r)/\rho_{-2}] = (-2/\alpha)[(r/r_{-2})^{\alpha} - 1]$$

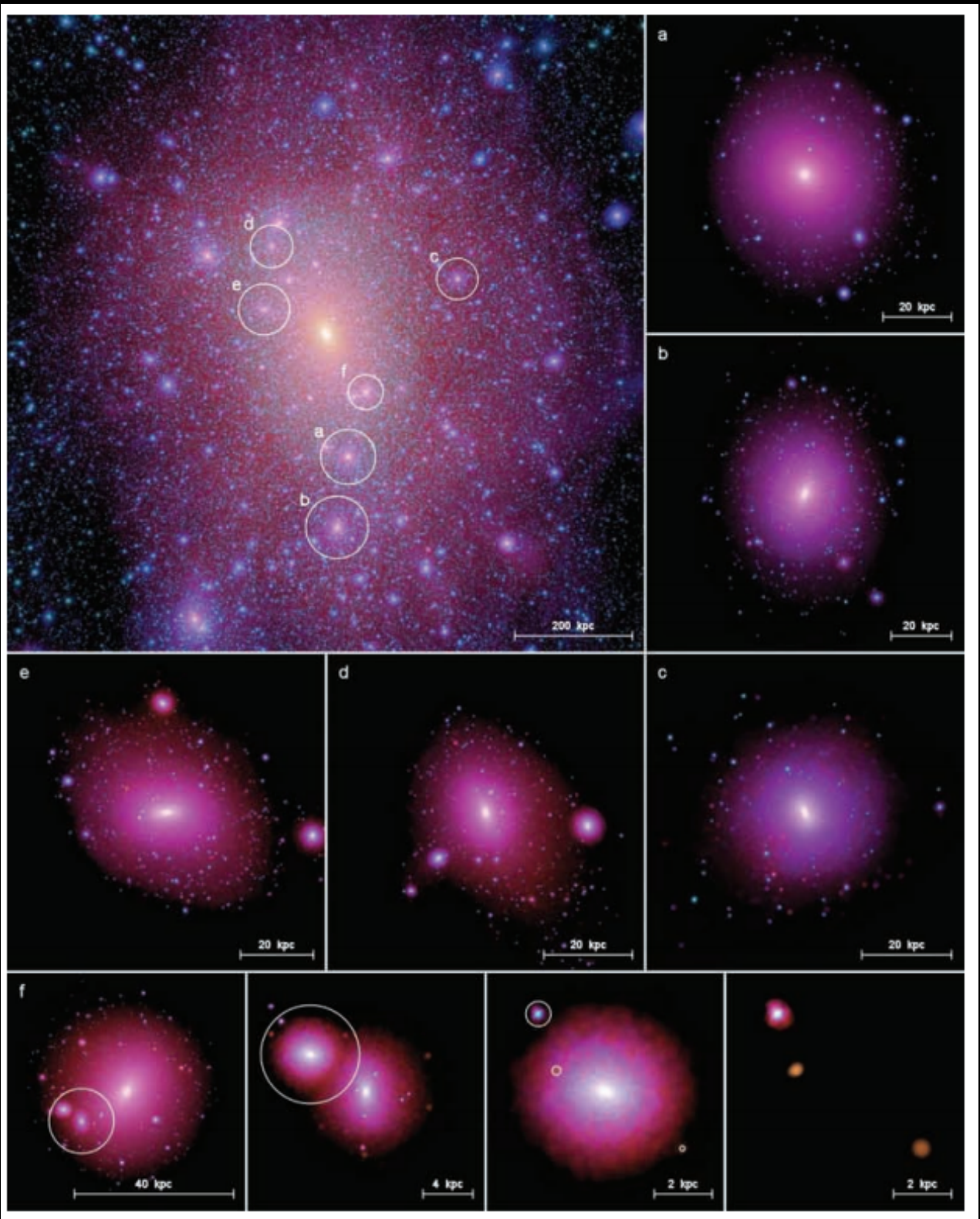
Navarro+ 2010

Density Profiles: Empirical Fitting Functions



generally NFW provides excellent fits; except for inner parts, where Einasto profiles have smaller residuals

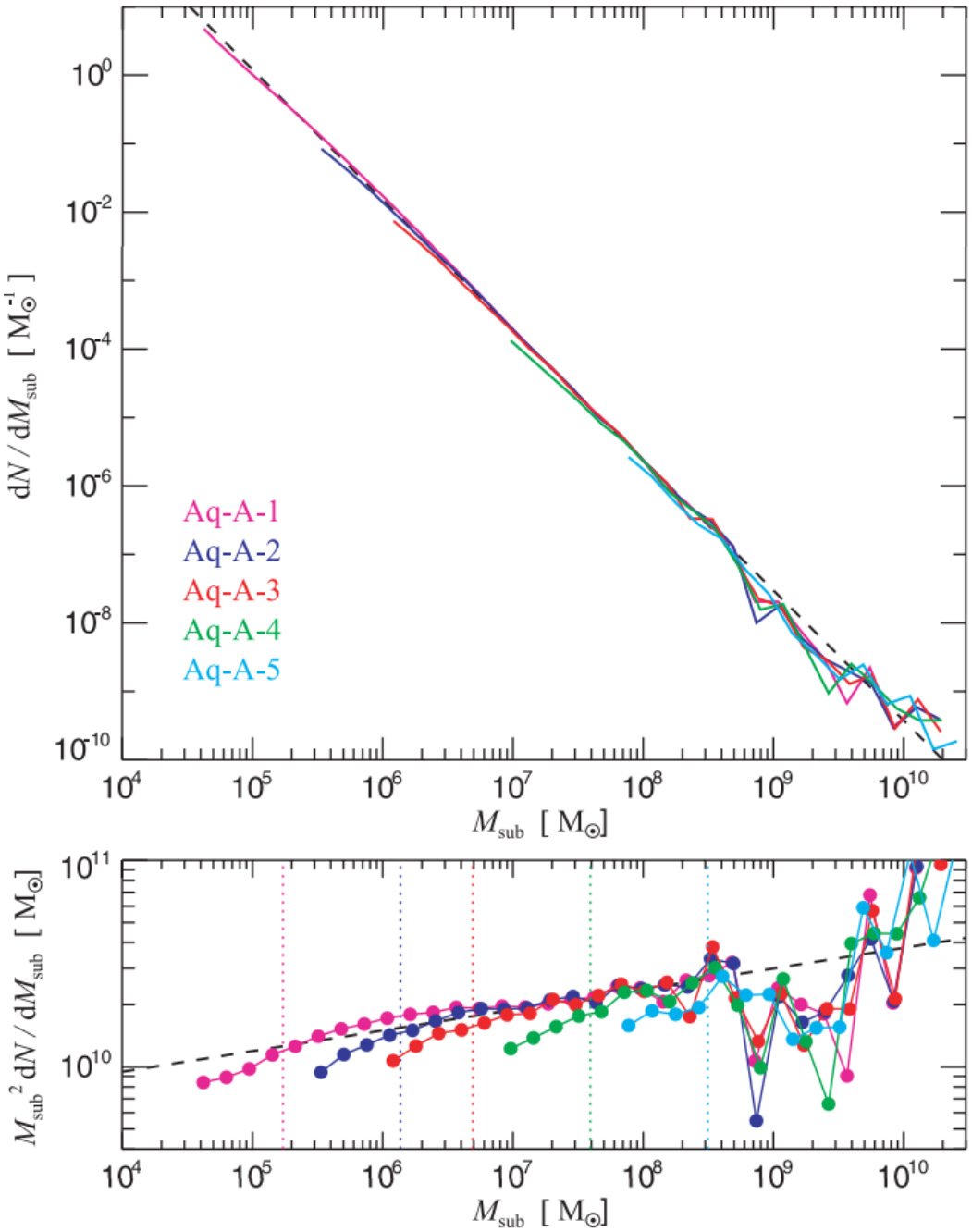
Navarro+ 2010



nested levels of subhalos:
a generic prediction of CDM

Springel+ 2008

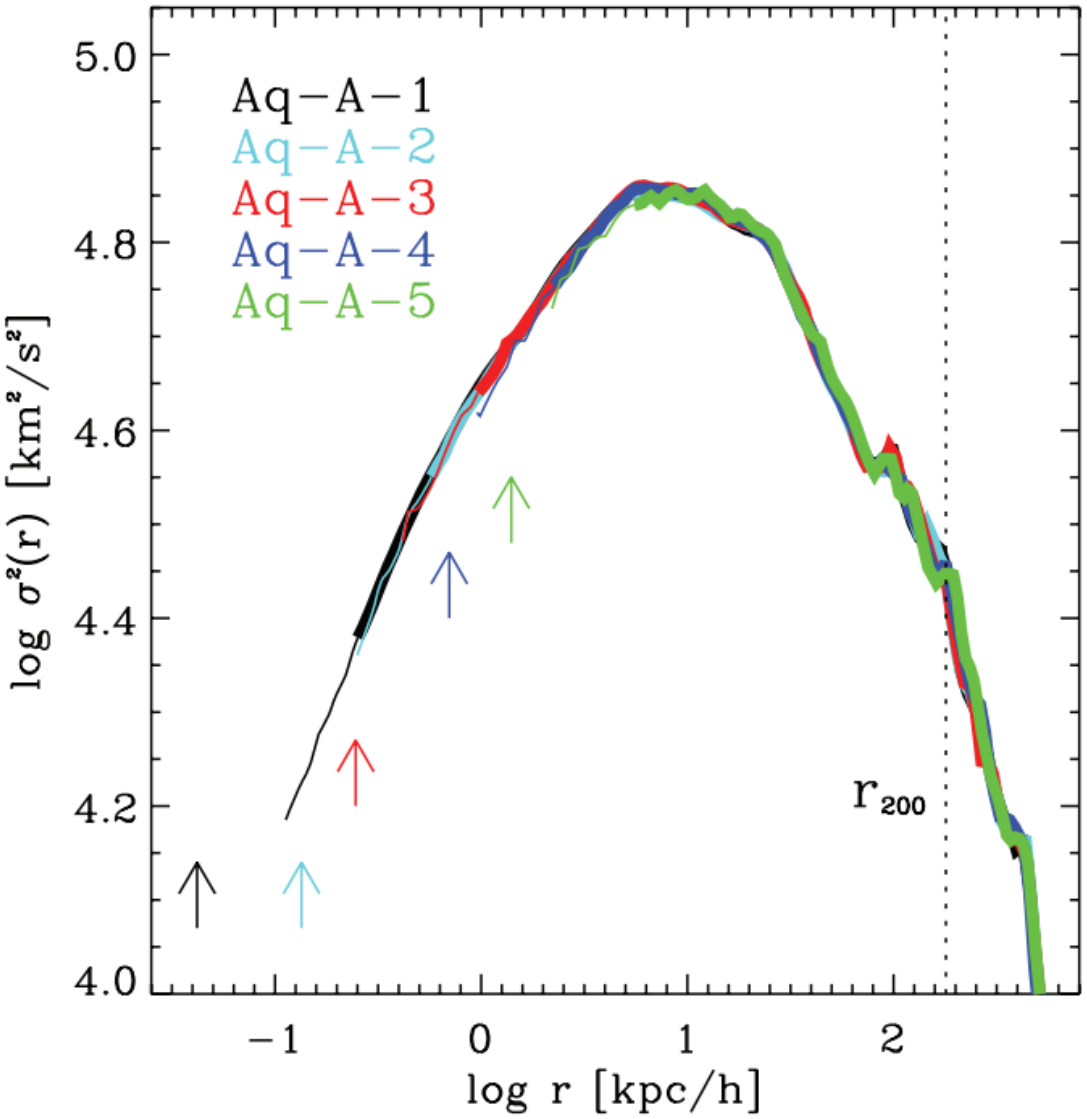
Subhalo Abundance



subhalo mass function has a slope close to -2; slightly shallower; extrapolate → annihilation boost

Springel+ 2008

Velocity Dispersion Profile



CDM halos also have generic velocity dispersion profiles

Navarro+ 2010

DM Detection

Direct Detection:

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

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

Indirect Detection:

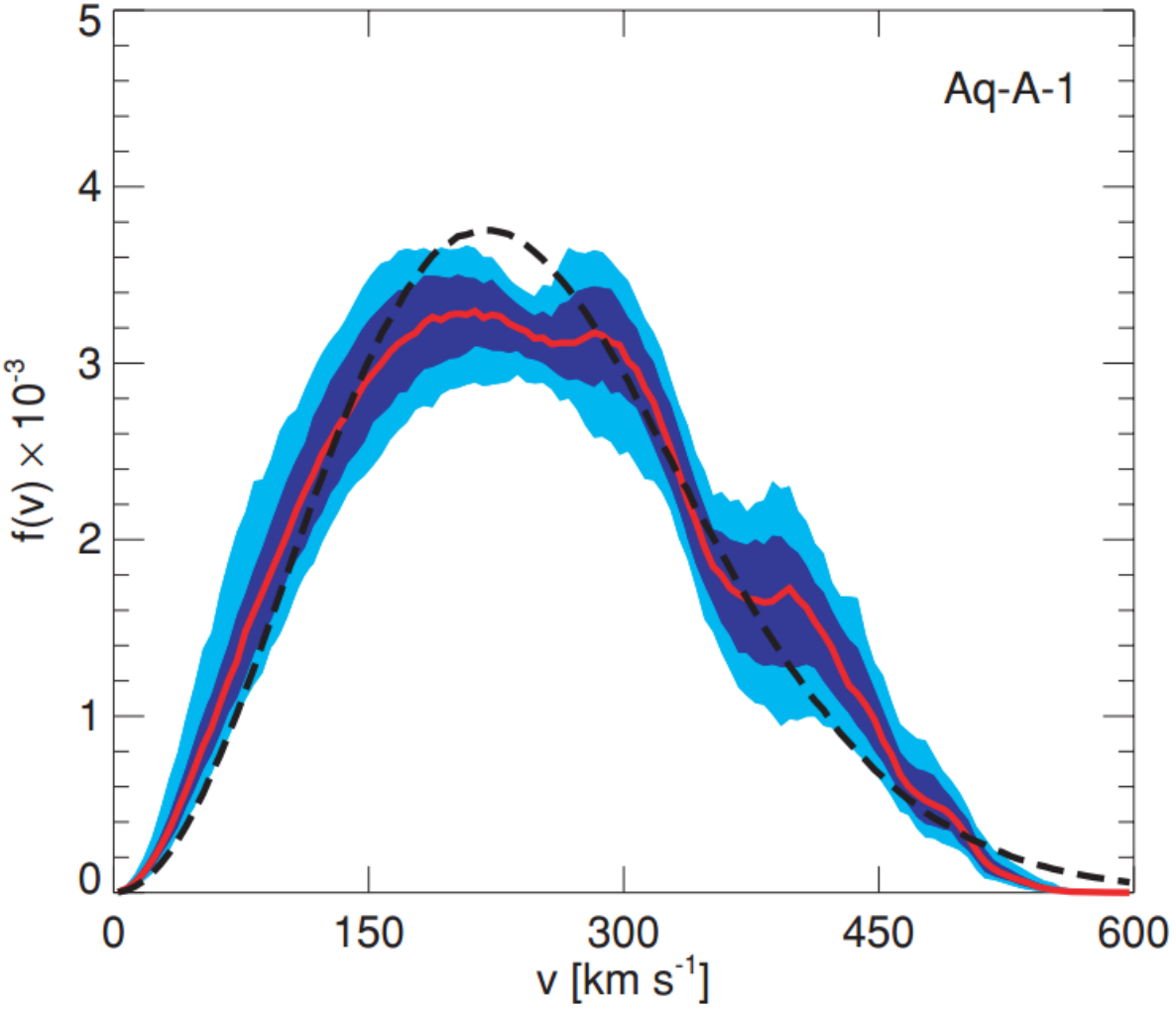
$$n = \frac{\langle \sigma v \rangle}{2 m_{\mathrm{DM}}^2} \int_V \mathrm{d}^3 x \rho_{\mathrm{DM}}^2(\mathbf{x})$$

phase-space structure of CDM halos is important for DM detection experiments: input from DM simulations

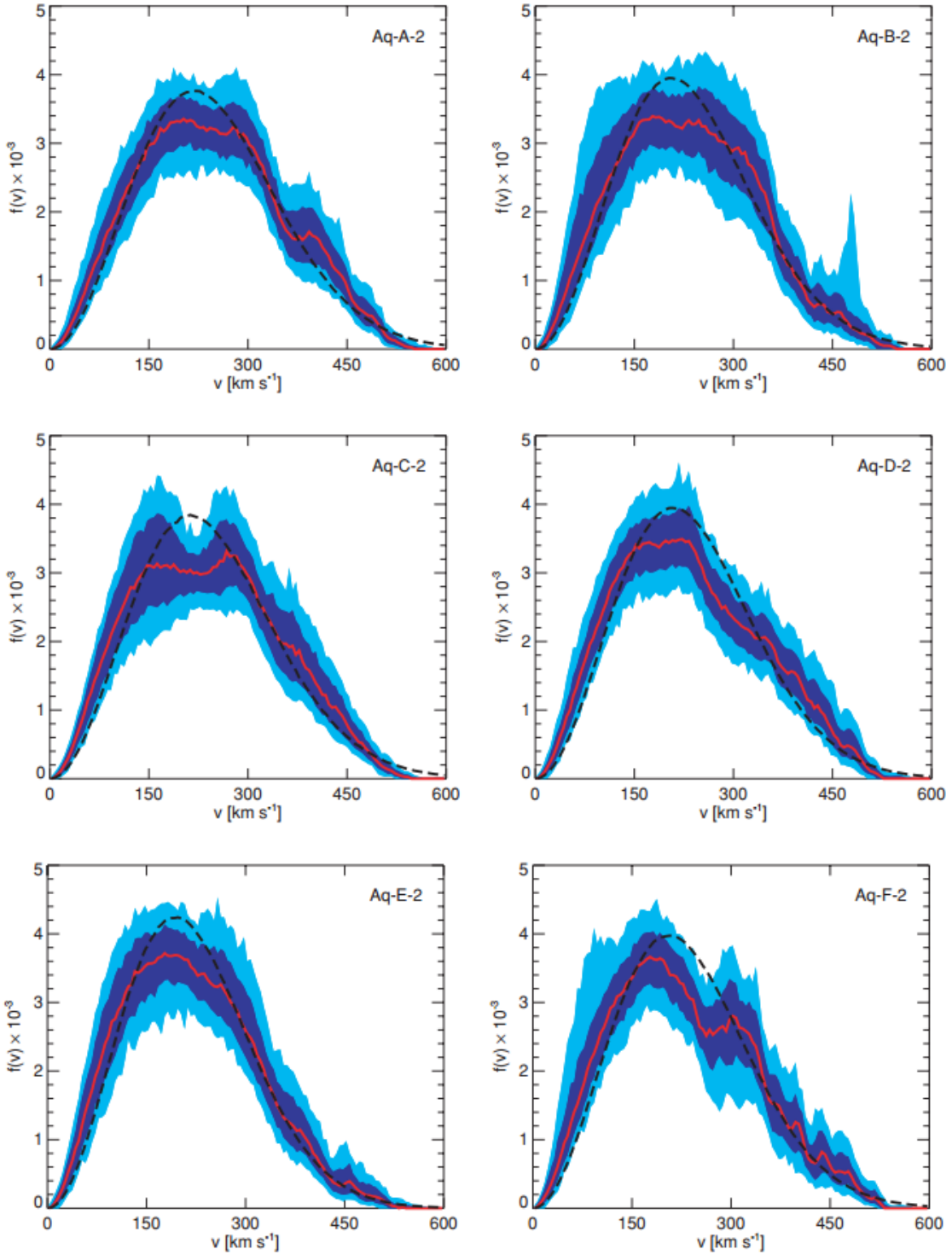
Local Velocity Distribution at Solar Circle

predicted local velocity distribution is not Maxwellian like in standard halo model

includes imprints of halo formation history



Vogelsberger+ 2009

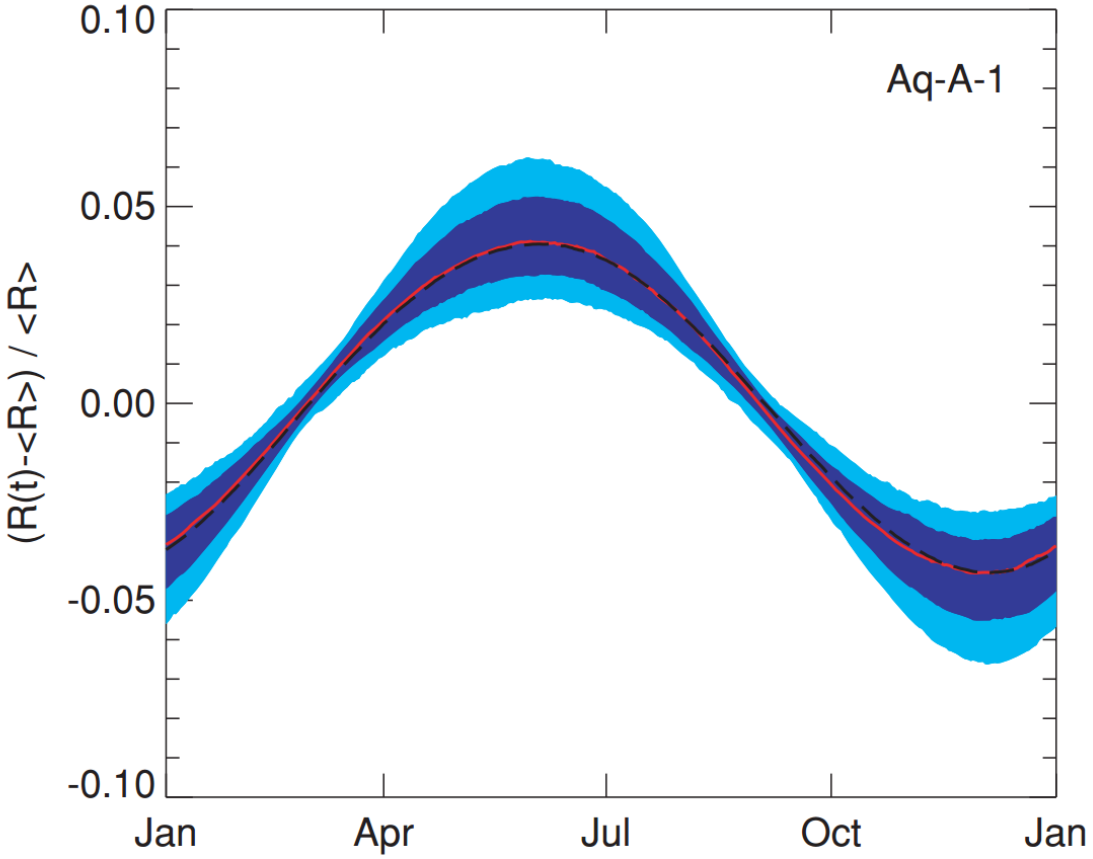


sample of six different halos: all have individually different local DM velocity distributions

DM ‘Astronomy’?

Vogelsberger+ 2009

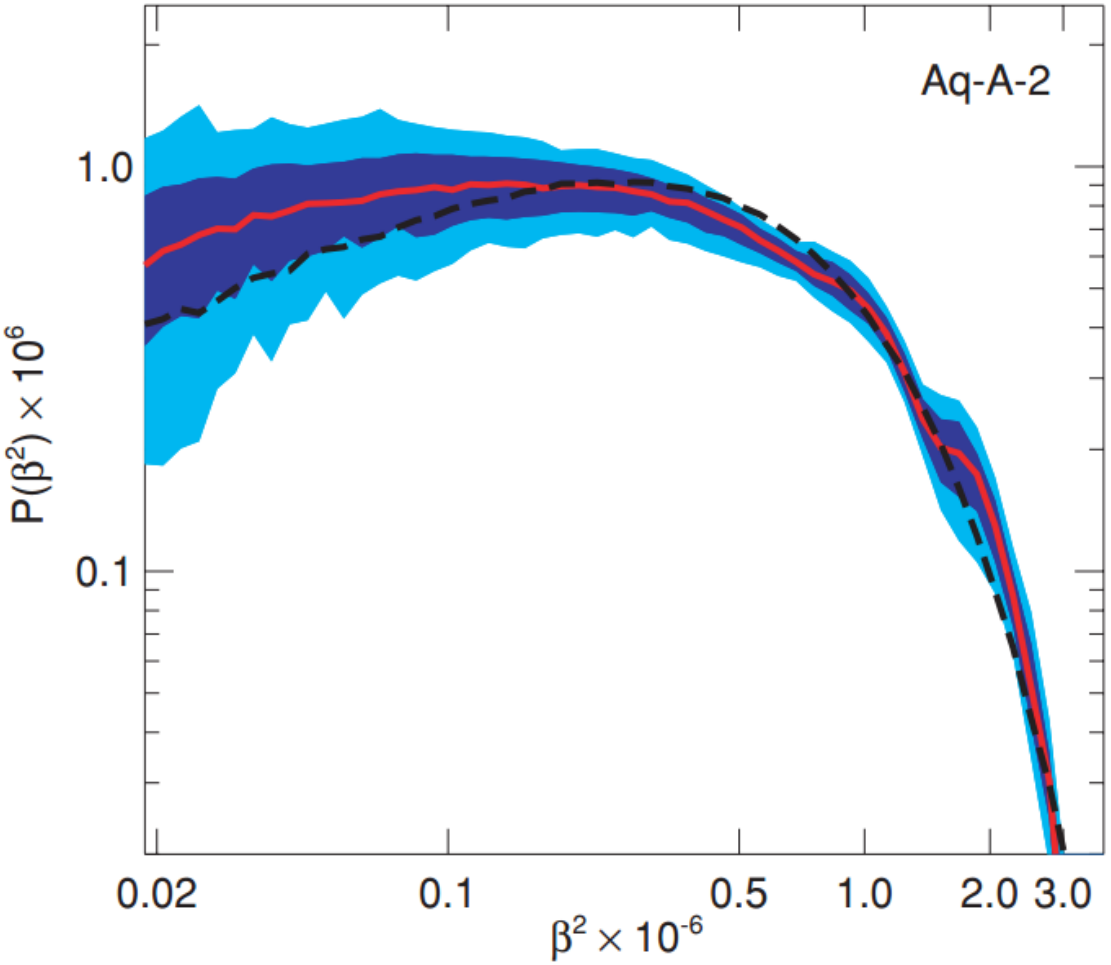
Annual Modulation Signal



predicted annual modulation signal: mock the motion of earth

Vogelsberger+ 2009

Axion Spectra

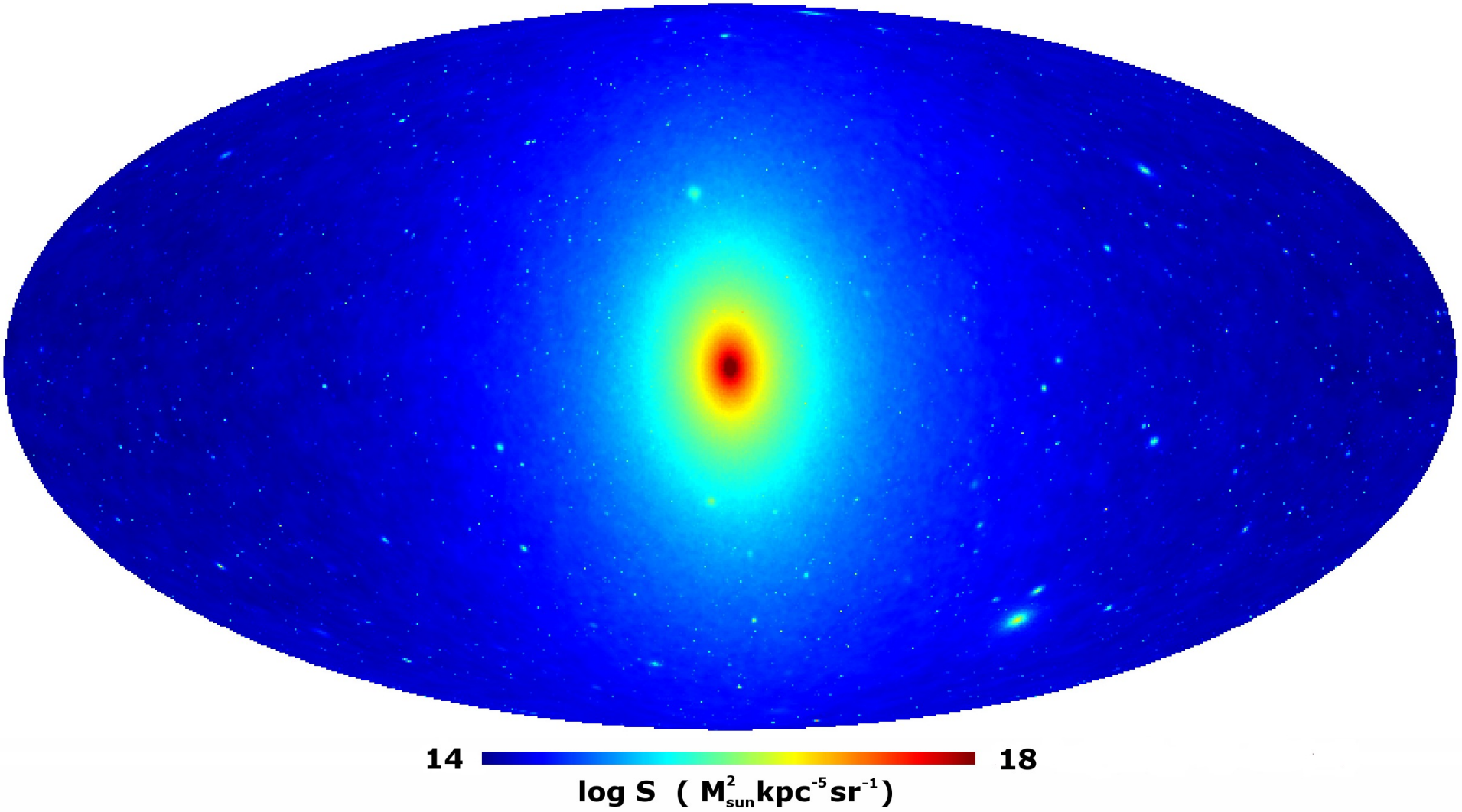


mock axion spectra

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

Vogelsberger+ 2009

Annihilation Signal



Springel+ 2008

DM Simulation Methods: A brief Overview

Description of DM

→ 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 gravitational 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

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

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

real space density derived from distribution function

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

Brute Force

→ 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 $200^3 \times 200^3 = 64$ trillion grid cells

→ assume we store 1 float = (4 x 1 = 4 Bytes) per grid cell

→ $64 \times 10^{12} \times 4$ Bytes ~ 0.25 PetaByte

→ for a two times finer grid ~ 2 PetaByte

requires 2 PetaByte = 2,000 TB = 2,000,000 GB RAM

assuming 2GB per core: requires 1,000,000 processors

top500.org (June)

Rank	Site	System	Cores	Rmax (TFlop/s)	Rpeak (TFlop/s)	Power (kW)
1	National Supercomputing Center in Wuxi China	**Sunway TaihuLight** - Sunway MPP, Sunway SW26010 260C 1.45GHz, Sunway NRCPC	10,649,600	93,014.6	125,435.9	15,371
2	National Super Computer Center in Guangzhou China	**Tianhe-2 (MilkyWay-2)** - TH-IVB-FEP Cluster, Intel Xeon E5-2692 12C 2.200GHz, TH Express-2, Intel Xeon Phi 31S1P NUDT	3,120,000	33,862.7	54,902.4	17,808
3	Swiss National Supercomputing Centre (CSCS) Switzerland	**Piz Daint** - Cray XC50, Xeon E5-2690v3 12C 2.6GHz, Aries interconnect , NVIDIA Tesla P100 Cray Inc.	361,760	19,590.0	25,326.3	2,272
4	DOE/SC/Oak Ridge National Laboratory United States	**Titan** - Cray XK7, Opteron 6274 16C 2.200GHz, Cray Gemini interconnect, NVIDIA K20x Cray Inc.	560,640	17,590.0	27,112.5	8,209
5	DOE/NNSA/LLNL United States	**Sequoia** - BlueGene/Q, Power BQC 16C 1.60 GHz, Custom IBM	1,572,864	17,173.2	20,132.7	7,890
6	DOE/SC/LBNL/NERSC United States	**Cori** - Cray XC40, Intel Xeon Phi 7250 68C 1.4GHz, Aries interconnect Cray Inc.	622,336	14,014.7	27,880.7	3,939
7	Joint Center for Advanced High Performance Computing Japan	**Oakforest-PACS** - PRIMERGY CX1640 M1, Intel Xeon Phi 7250 68C 1.4GHz, Intel Omni-Path Fujitsu	556,104	13,554.6	24,913.5	2,719
8	RIKEN Advanced Institute for Computational Science (AICS) Japan	K computer, SPARC64 VIIIfx 2.0GHz, Tofu interconnect Fujitsu	705,024	10,510.0	11,280.4	12,660
9	DOE/SC/Argonne National Laboratory United States	**Mira** - BlueGene/Q, Power BQC 16C 1.60GHz, Custom IBM	786,432	8,586.6	10,066.3	3,945
10	DOE/NNSA/LANL/SNL United States	**Trinity** - Cray XC40, Xeon E5-2698v3 16C 2.3GHz, Aries interconnect Cray Inc.	301,056	8,100.9	11,078.9	4,233

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

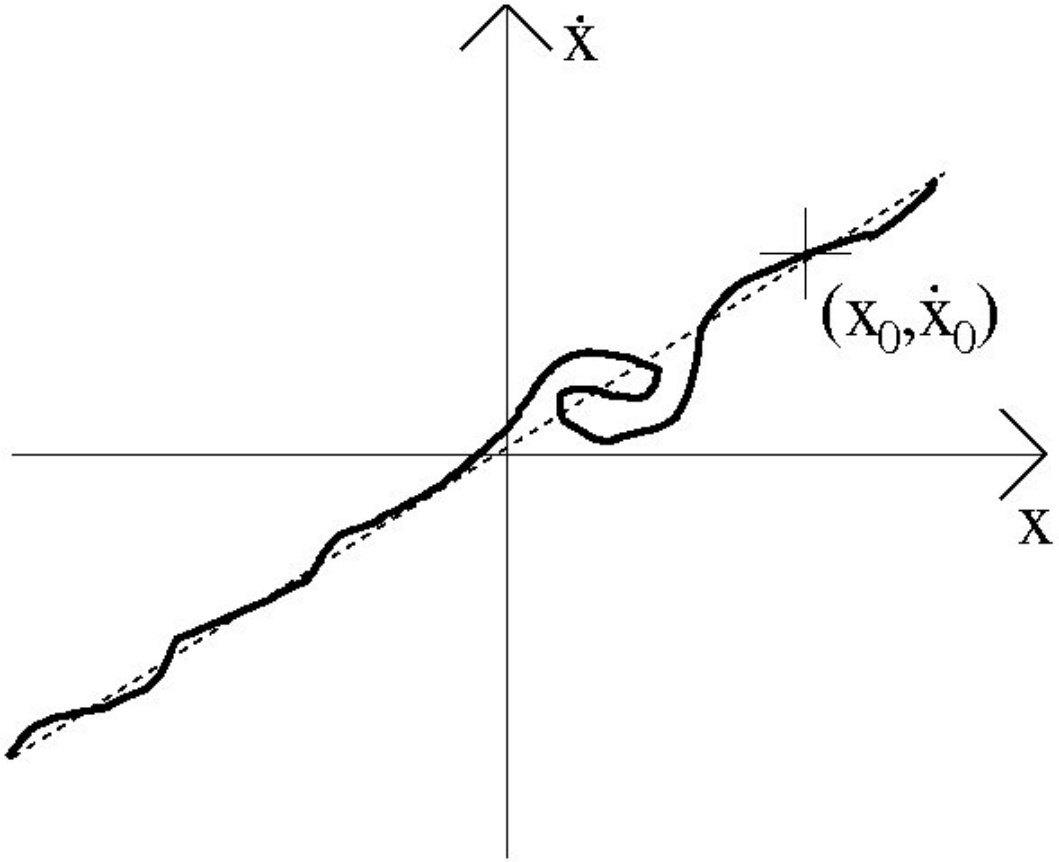
We present a scheme for numerical simulations of collisionless self-gravitating systems which directly integrates the Vlasov–Poisson equations in six-dimensional phase space. Using the results from a suite of large-scale numerical simulations, we demonstrate that the present scheme can simulate collisionless 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.

impractical for state-of-the-art simulations

Phase-Space Structure of CDM

→ 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

Monte Carlo Approach

→ discretize in terms of N simulation 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

Monte Carlo Approach

→ **small particle mass to resolve details**

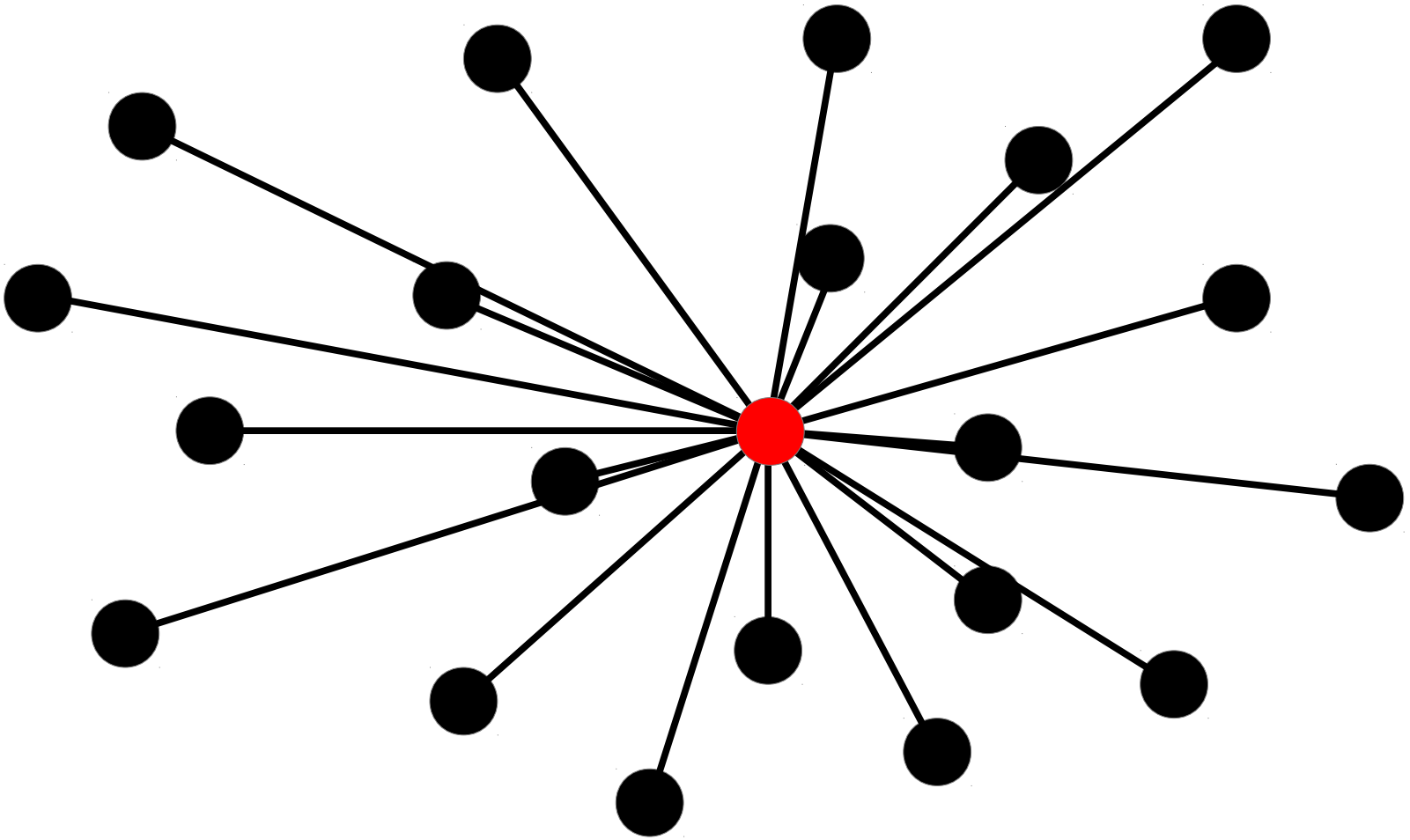
→ **large volume for representative volume**

the more particles the better

Task: Solve the N-body problem

→ **how to compute the gravitational forces efficiently and accurately?**

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

complexity $O(N^2)$

Particle Mesh Methods

→ **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 parallel FFT to efficiently carry out the two Fourier transformations

Particle Mesh Methods

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

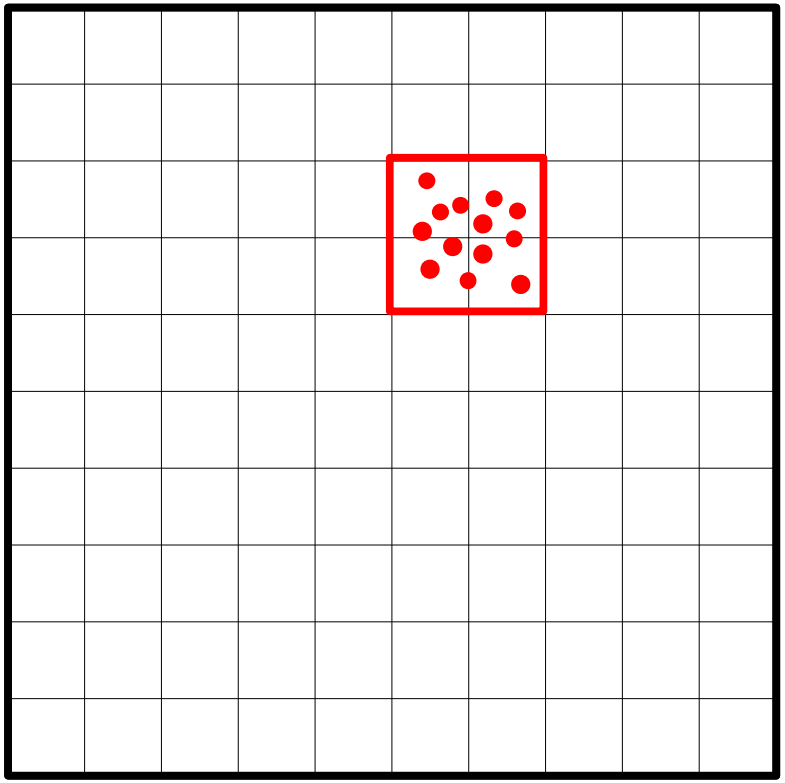
$$f^{(x)}_{i,j,k} = -\frac{4}{3}\frac{\Phi_{i+1,j,k} - \Phi_{i-1,j,k}}{2h} + \frac{1}{3}\frac{\Phi_{i+2,j,k} - \Phi_{i-2,j,k}}{4h}$$

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)

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



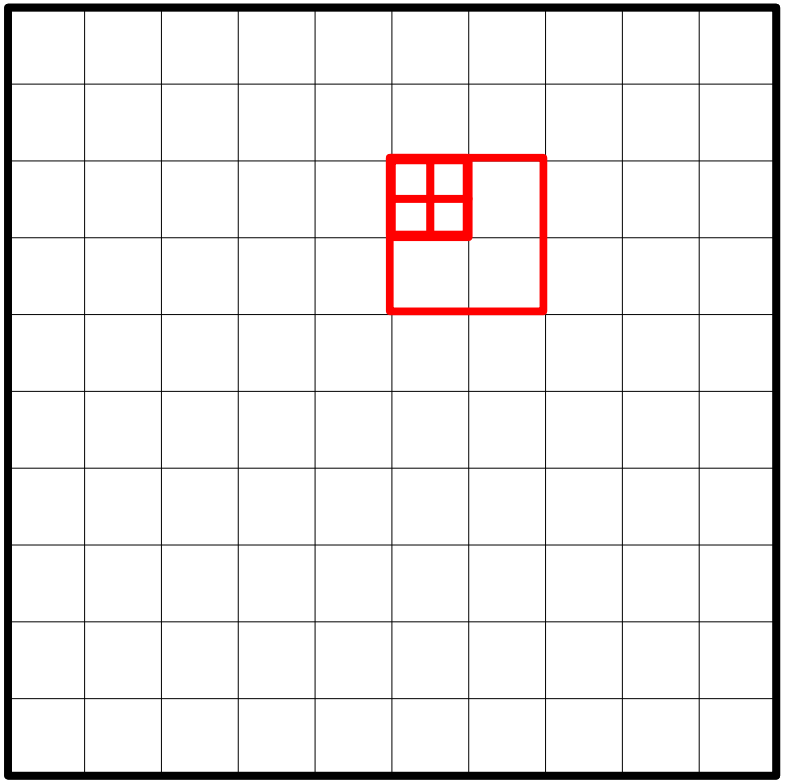
→ do a direct summation at the mesh scale

→ this increases resolution

→ increases dynamic range a lot

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

Mesh Refinement



→ additional smaller meshes for small scales

→ use refinement criteria

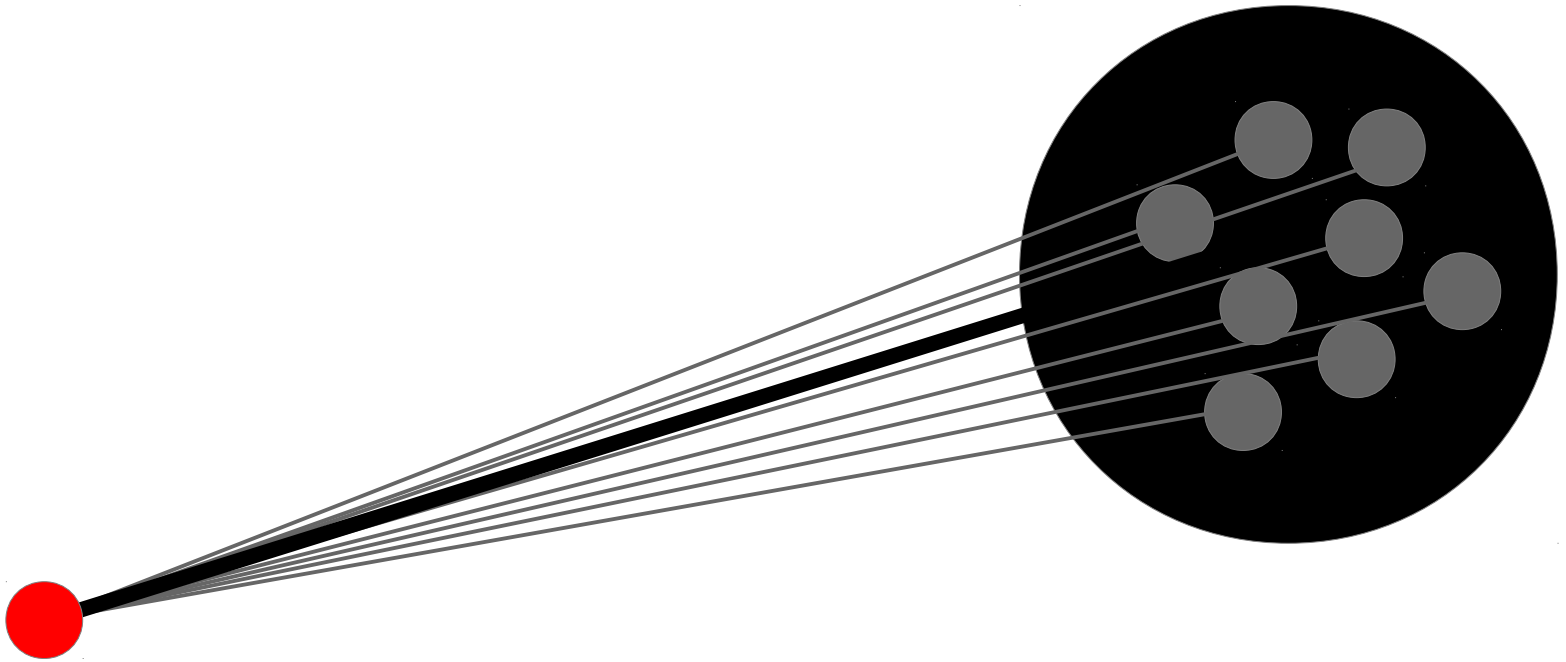
Tree-Algorithm

Task: efficient force calculation without mesh and FFT

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

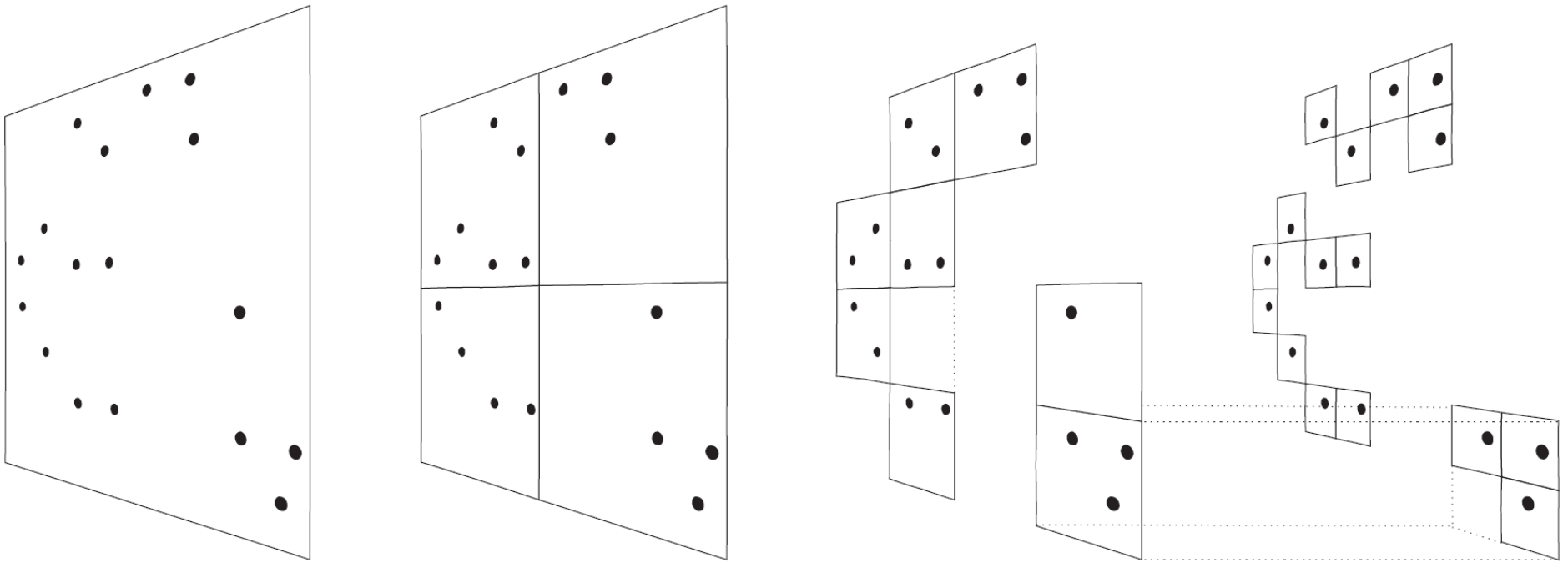
→ i.e. each particle has to calculate a pairwise force with N other particles

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



Tree-Algorithm

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 (Barnes-Hut criterion)

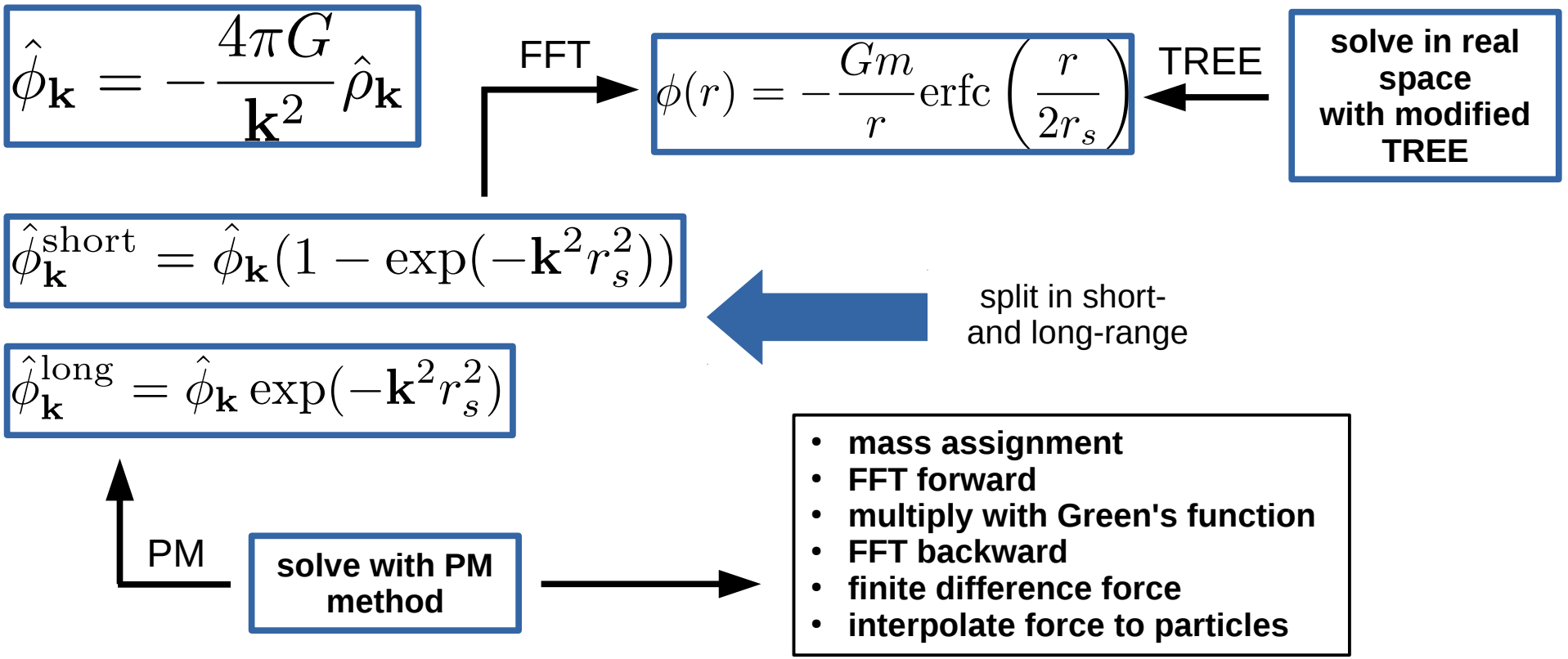
Tree-Algorithm

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



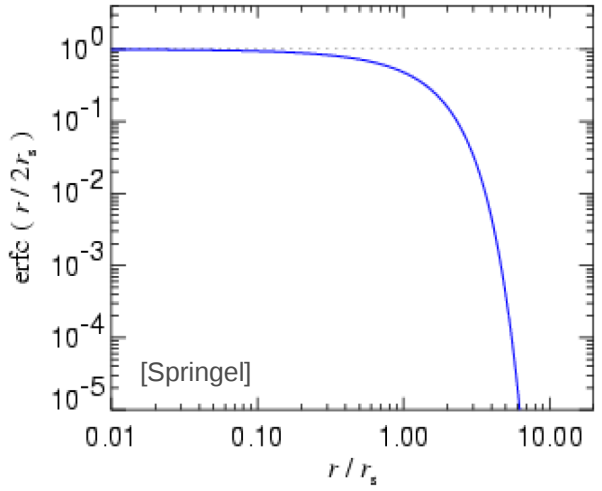
Tree-Algorithm

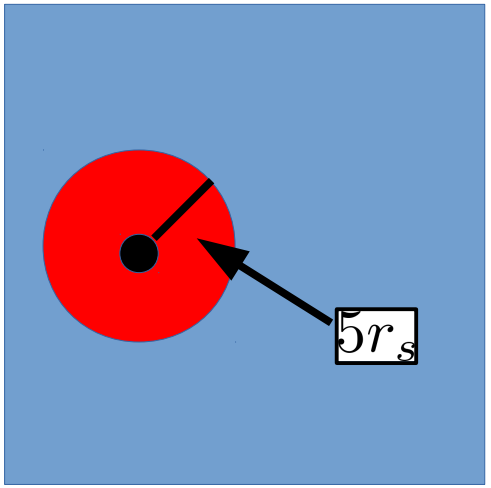
→ **erfc(x) drops very quickly: erfc(2.5)~10^{-4}**

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

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

→ **long-range contribution comes from PM part**

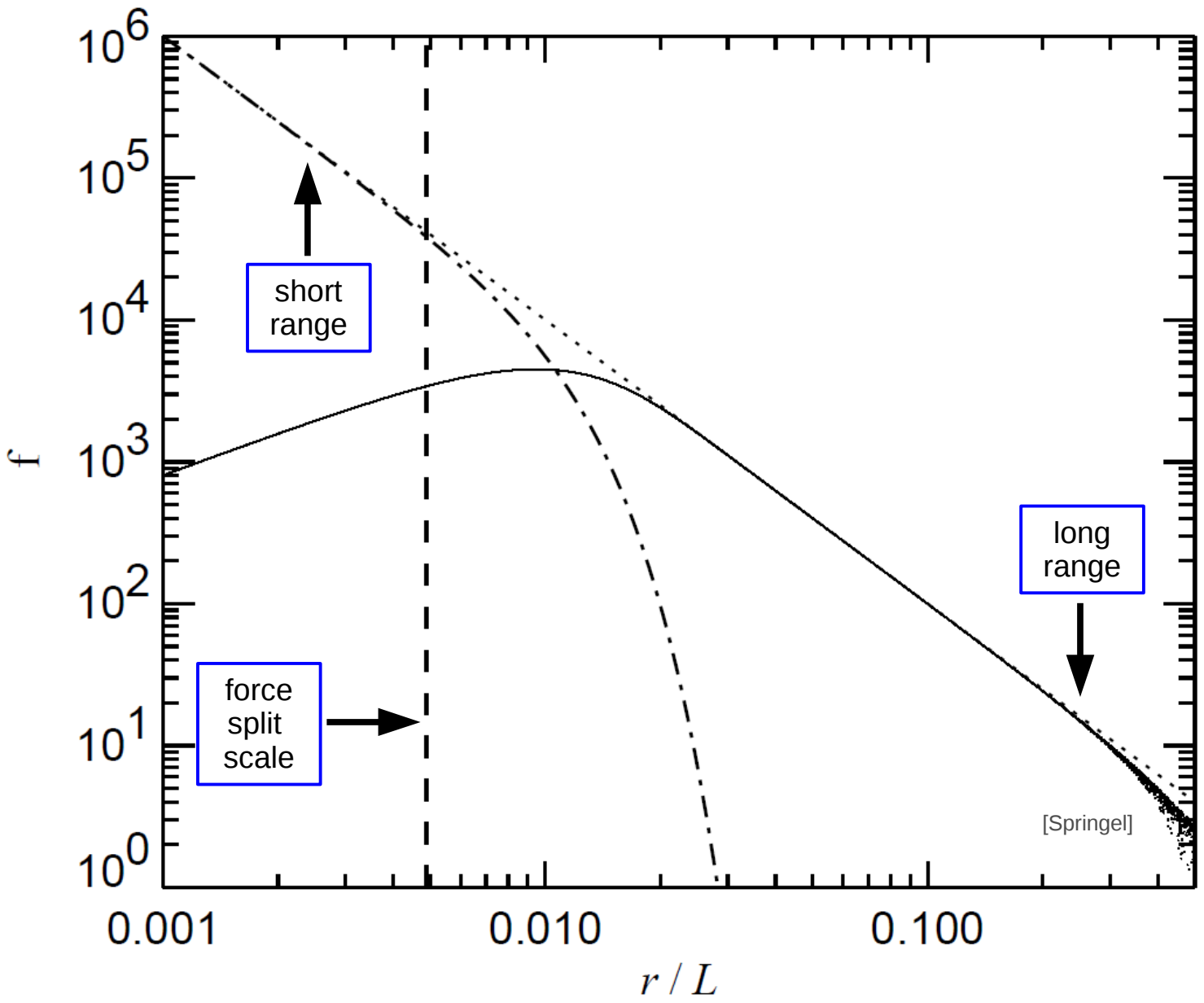




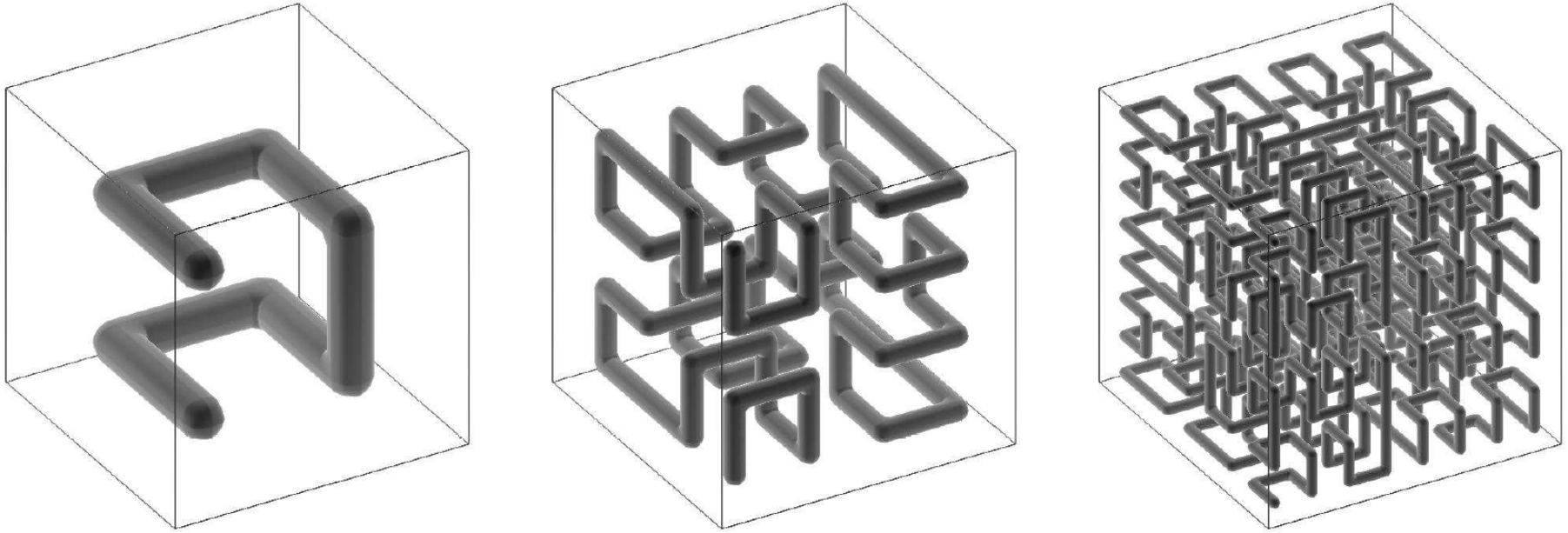
advantages:
- **accurate and fast long-range force**
- **high resolution on small scales**

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

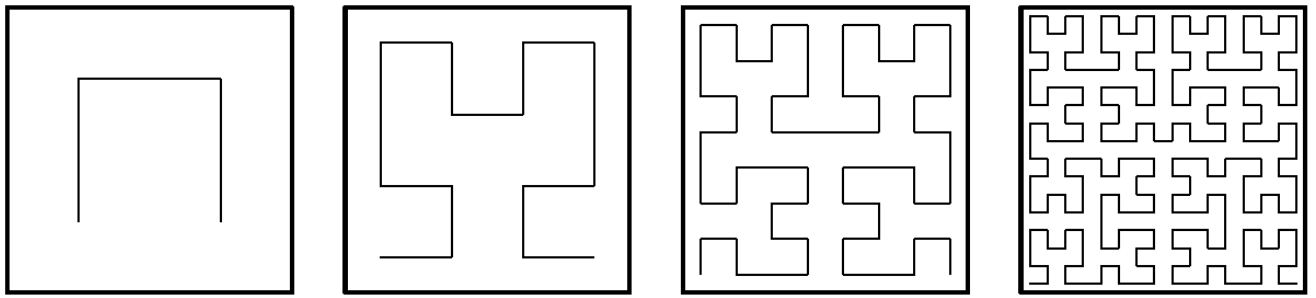
Tree-Algorithm



Parallelization



[Springel]

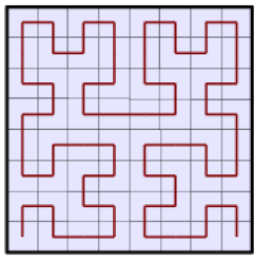


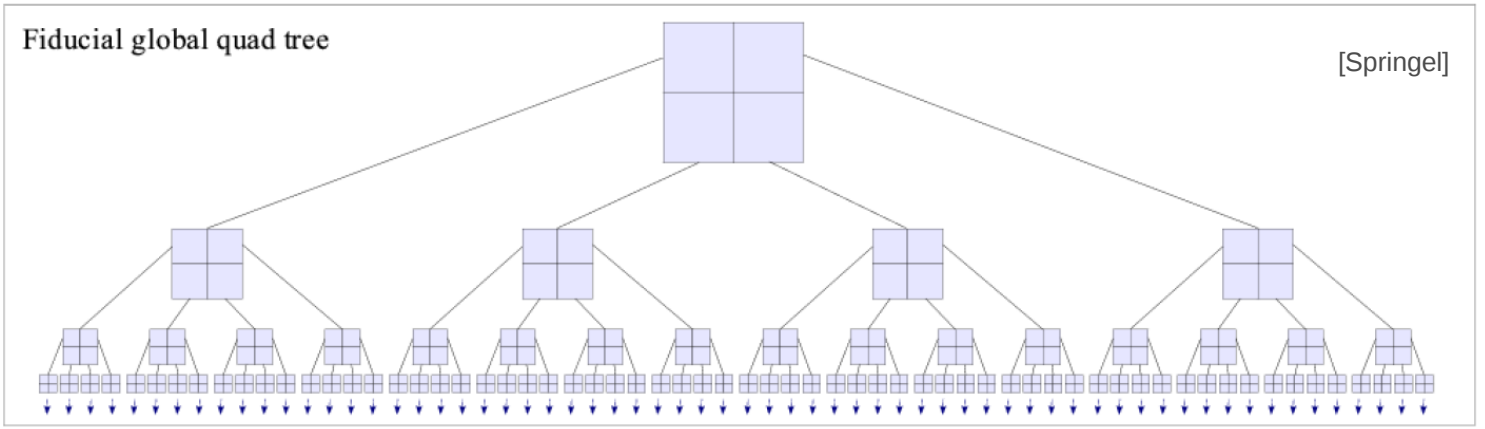
→ **sort particles along a fractal Peano-Hilbert curve**

→ **chop curve for domain decomposition**

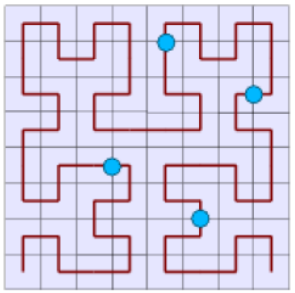
Parallelization

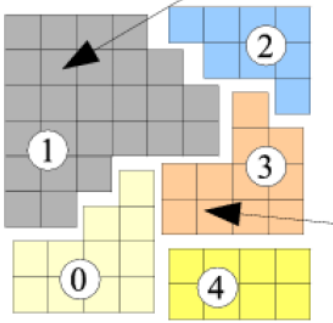
Peano-Hilbert curve

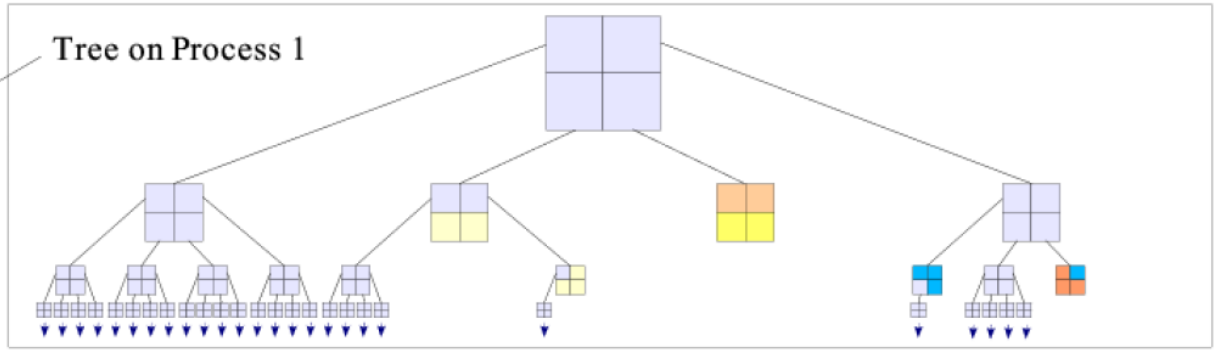


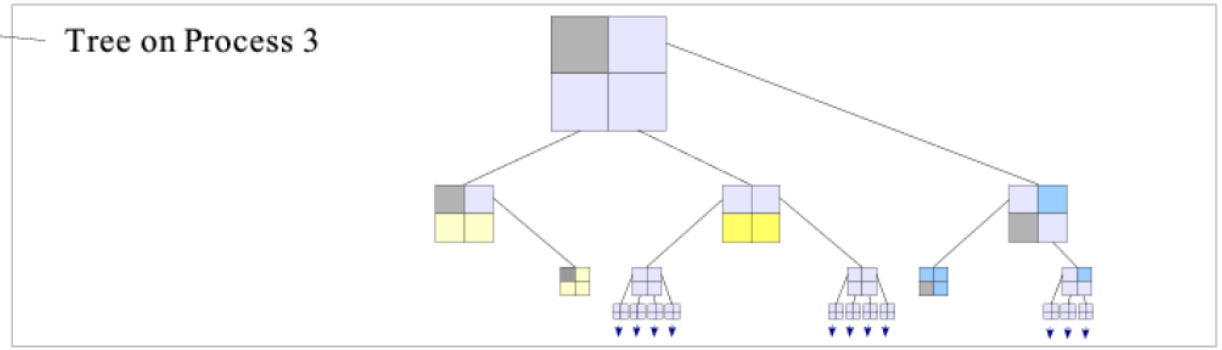


Domains are obtained by cutting the Peano-Hilbert curve into segments



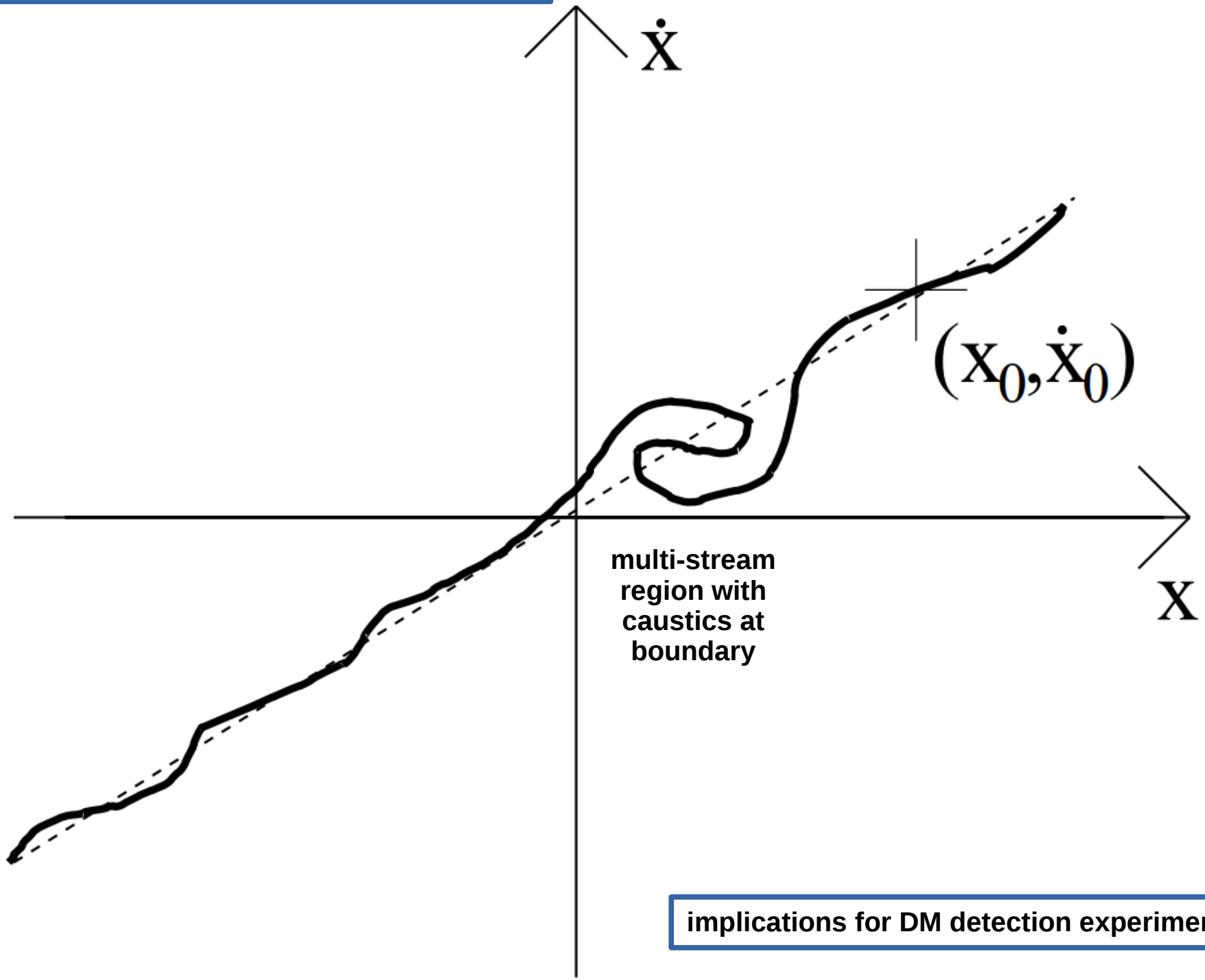






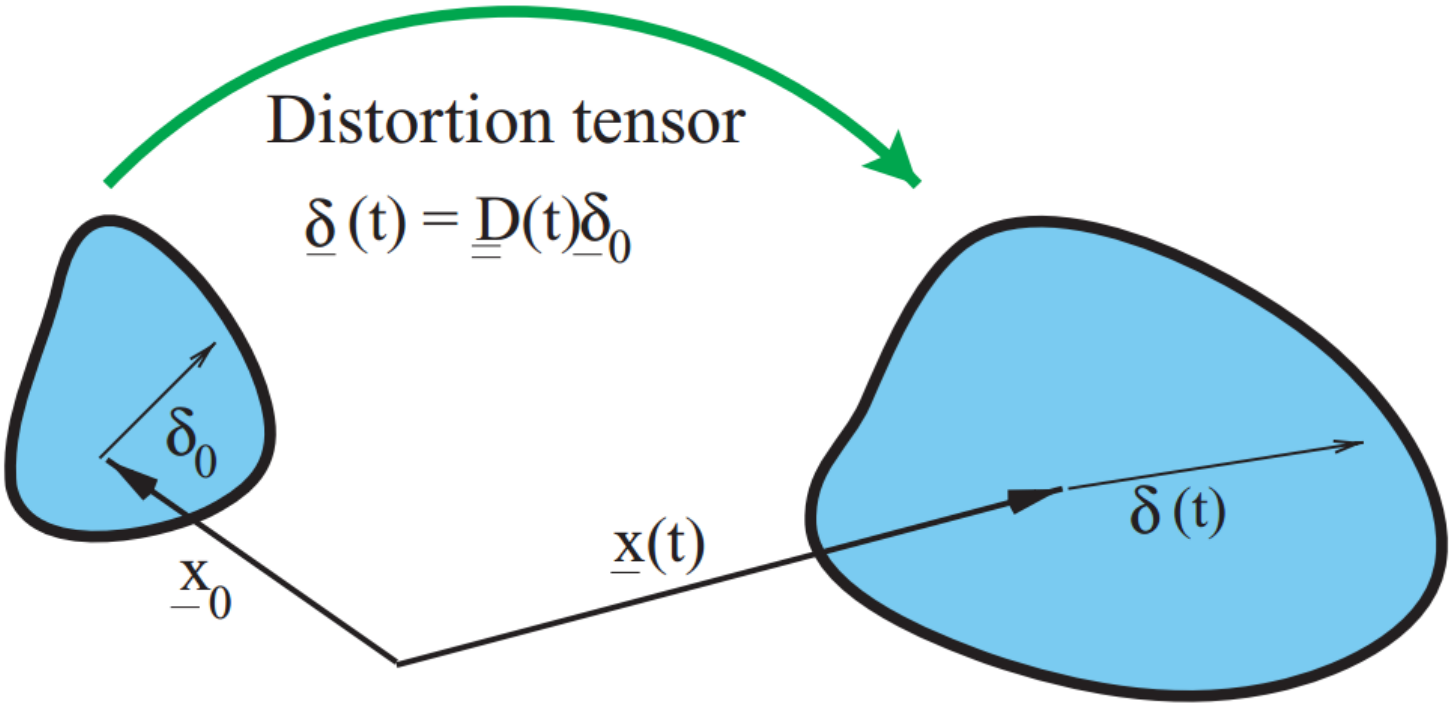
Beyond N-Body Methods

The fine-grained phase-space structure of CDM



implications for DM detection experiments?

Geodesic Deviation Equation: Resolving the Fine-Grained Structure

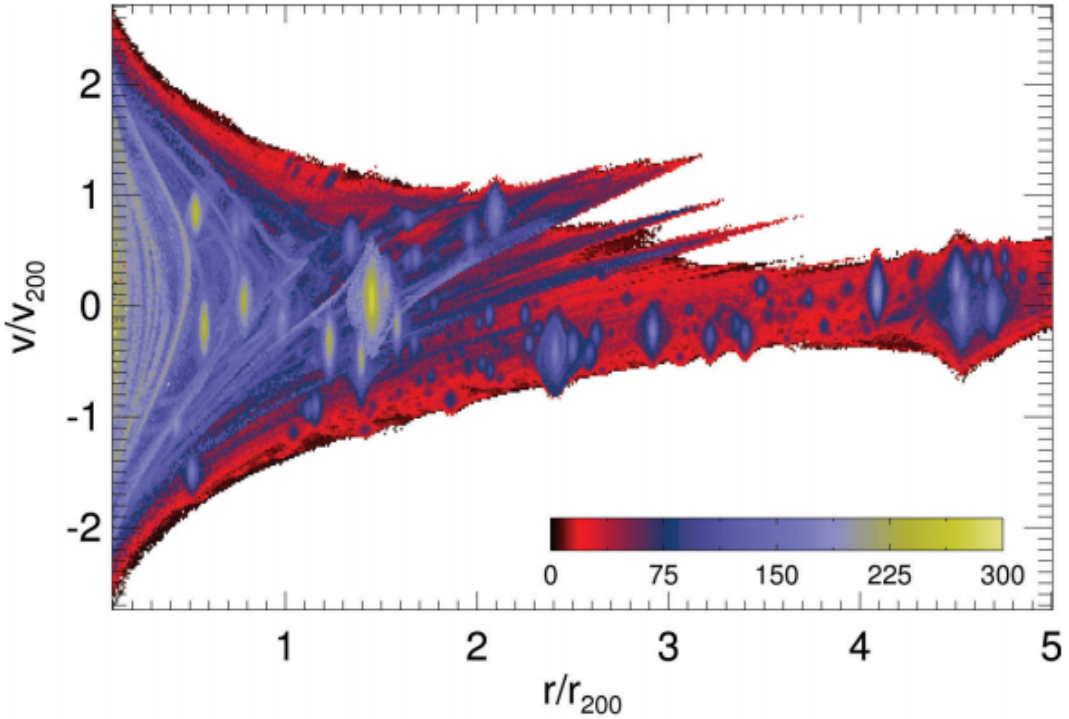


$$\overline{\overline{D}}(t;\overline{x}_0) \equiv \frac{\partial \overline{x}}{\partial \overline{x}_0}(t;\overline{x}_0)$$

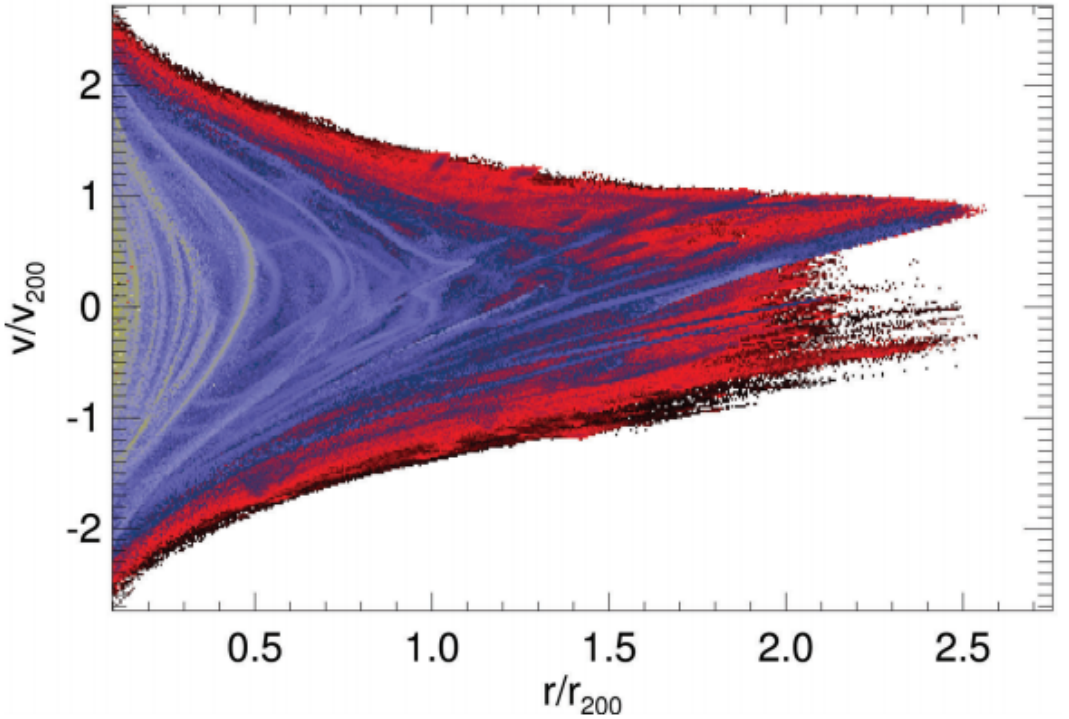
$$\overline{\delta}(t) \cong \overline{\overline{D}}(t;\overline{x}_0)\,\overline{\delta}_0$$

augment N-body equations with geodesic deviation equation

Vogelsberger+ 2007

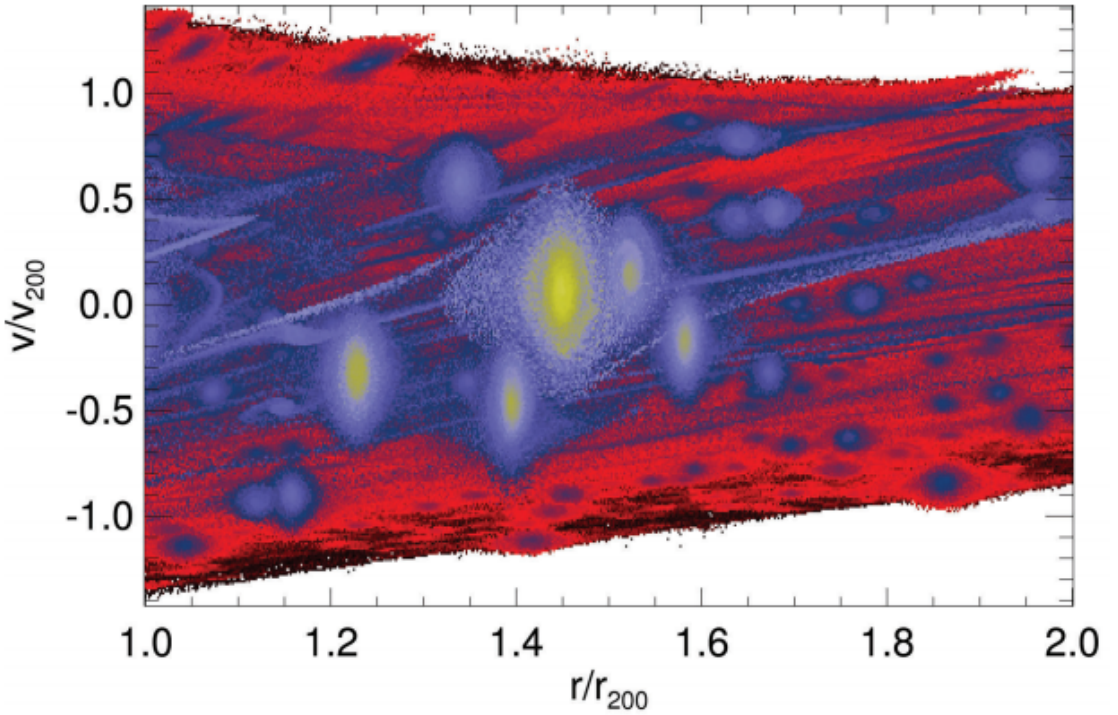


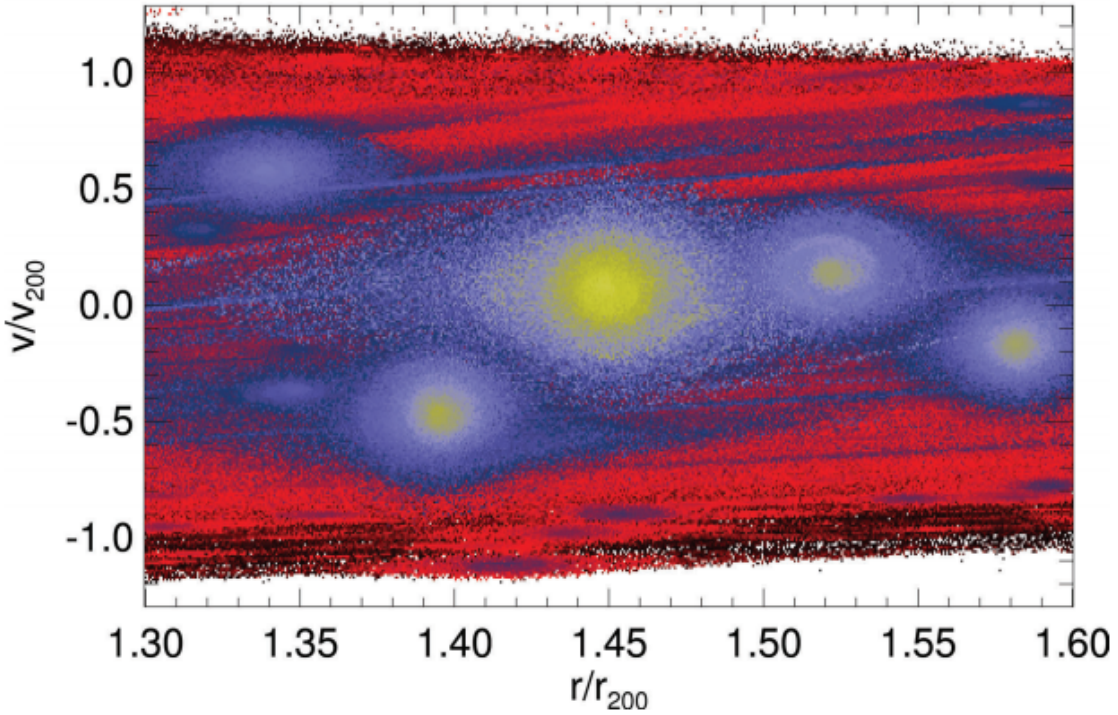
caustic crossing maps of a Milky Way-like halo



only smooth main halo: DM streams in phase-space

Vogelsberger+ 2011





caustic crossing maps near subhalos

Vogelsberger+ 2011