

PiTP Summer School 2009

Plan for my lectures

Volker Springel

- Lecture 1 ► **Basics of collisionless dynamics and the N-body approach**
- Lecture 2 ► **Gravitational solvers suitable for collisionless dynamics, parallelization**
- Lecture 3 ► **More parallelization, Introduction to smoothed particle hydrodynamics**
- Lecture 4 ► **Algorithmic aspects of SPH, caveats, applications**
- Lecture 5 ► **Comparison of SPH to finite volume methods, Moving-mesh hydrodynamics**



The N-body approach to collisionless dynamics

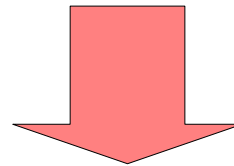
BASIC MONTE-CARLO IDEA

Collisionless
Boltzmann equation

Poisson-Vlasov System

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

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



N-body System

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

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

need large **N**

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

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

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

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

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

**Initial conditions
generation**

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

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

————▶ Then I is a solution of the CBE.

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

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

Example:

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

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

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

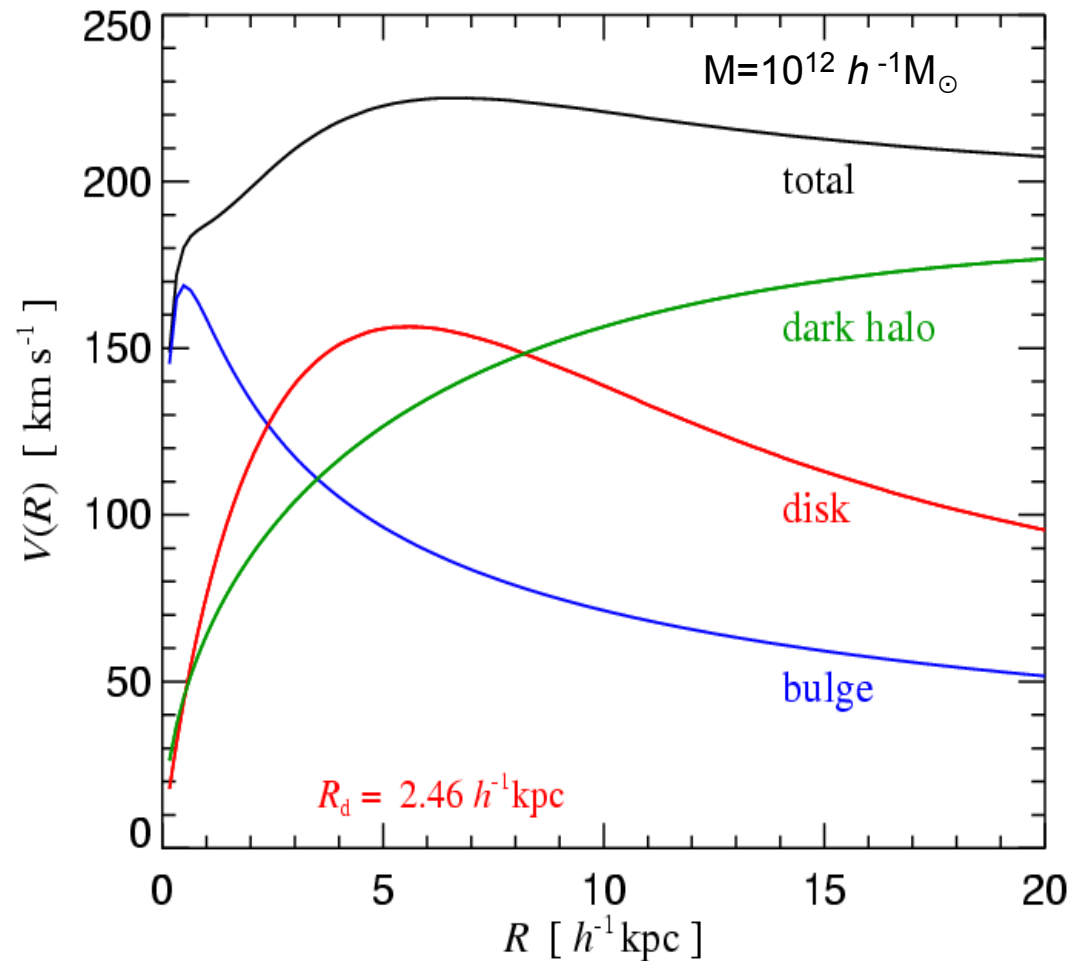
Construction of compound disk galaxies that are in dynamical equilibrium

STRUCTURAL PROPERTIES OF MODEL GALAXIES

Components:

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

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



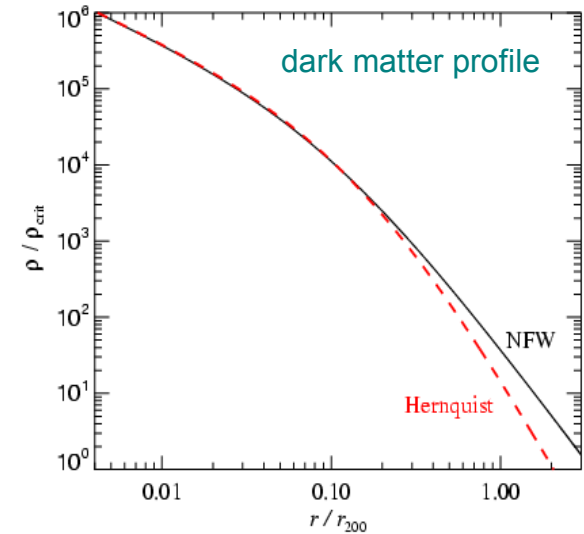
The first step in constructing an isolated galaxy model is the specification of the density structure of all mass components

DENSITY DISTRIBUTIONS OF DARK MATTER AND STARS IN BULGE AND DISK

Dark matter:

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

Hernquist or NFW profile



Stars in the disk:

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

“Isothermal sheet” with exponential profile

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

Disk scale length h determined by spin parameter of halo.

Stars in the bulge:

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

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

Gas in the disk:

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

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

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

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

MOMENT EQUATIONS FOR THE VELOCITY STRUCTURE

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

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

Then cross-moments vanish:

$$\langle v_R v_z \rangle = \langle v_z v_\phi \rangle = \langle v_R v_\phi \rangle = 0$$
$$\langle v_R \rangle = \langle v_z \rangle = 0$$

The radial and vertical moments are given by:

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

The azimuthal dispersion fulfills a separate equation:

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

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

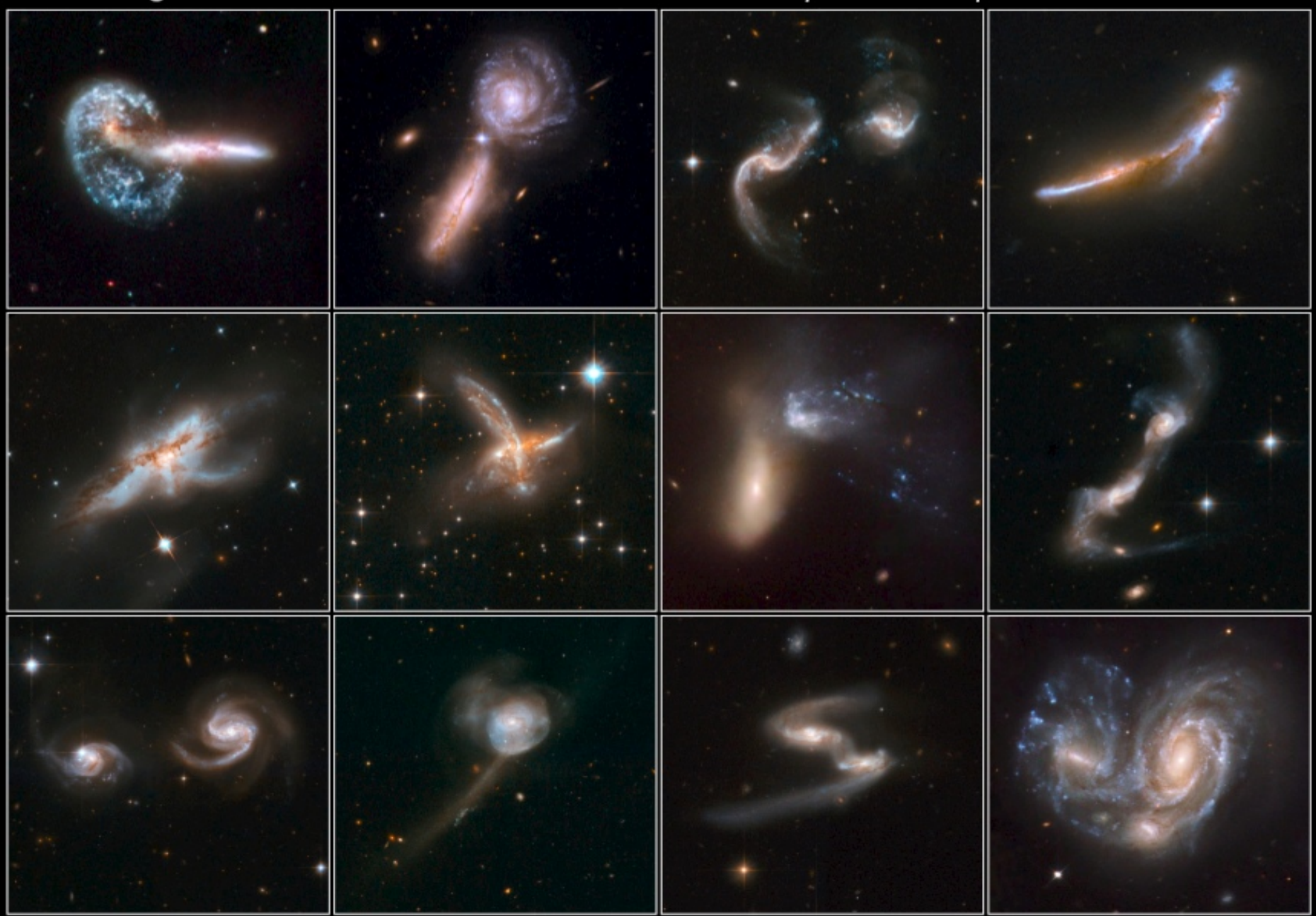
A remaining freedom lies in the azimuthal streaming $\langle v_\phi \rangle$, which is not determined by the above assumptions. For the dark matter, it can be set to zero, or to a value corresponding to a prescribed spin.

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

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

Interacting Galaxies

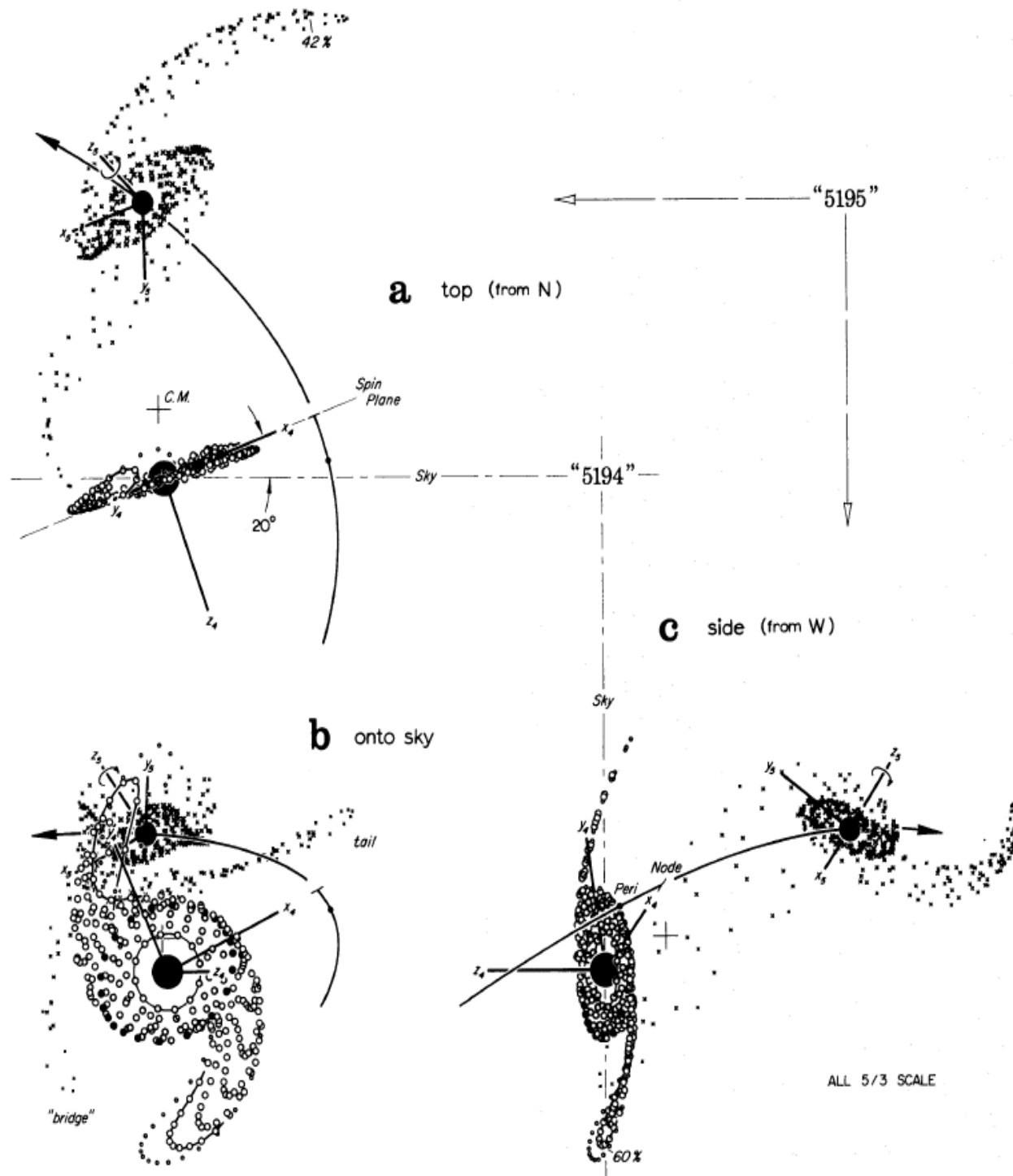
Hubble Space Telescope • ACS/WFC • WFPC2



The famous **merger hypothesis** conjectures that tidal features around galaxies occur in collisions which ultimately produce spheroidals

TOOMRE & TOOMRE (1972 !)

Restricted three-body simulations

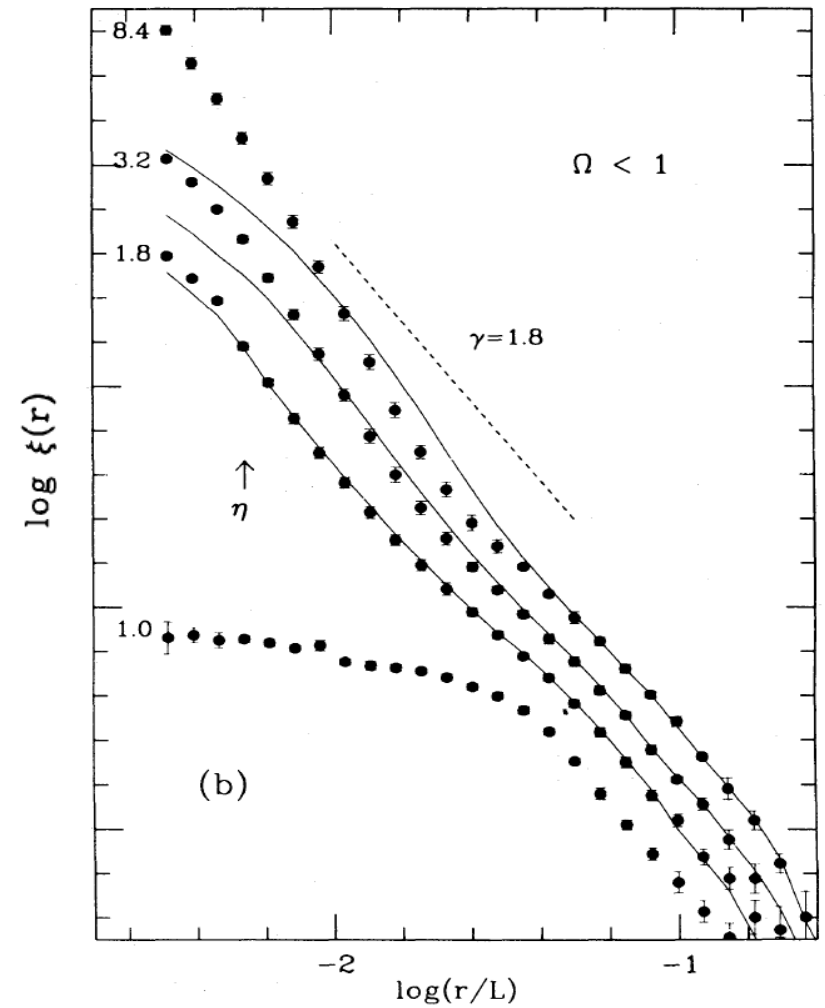
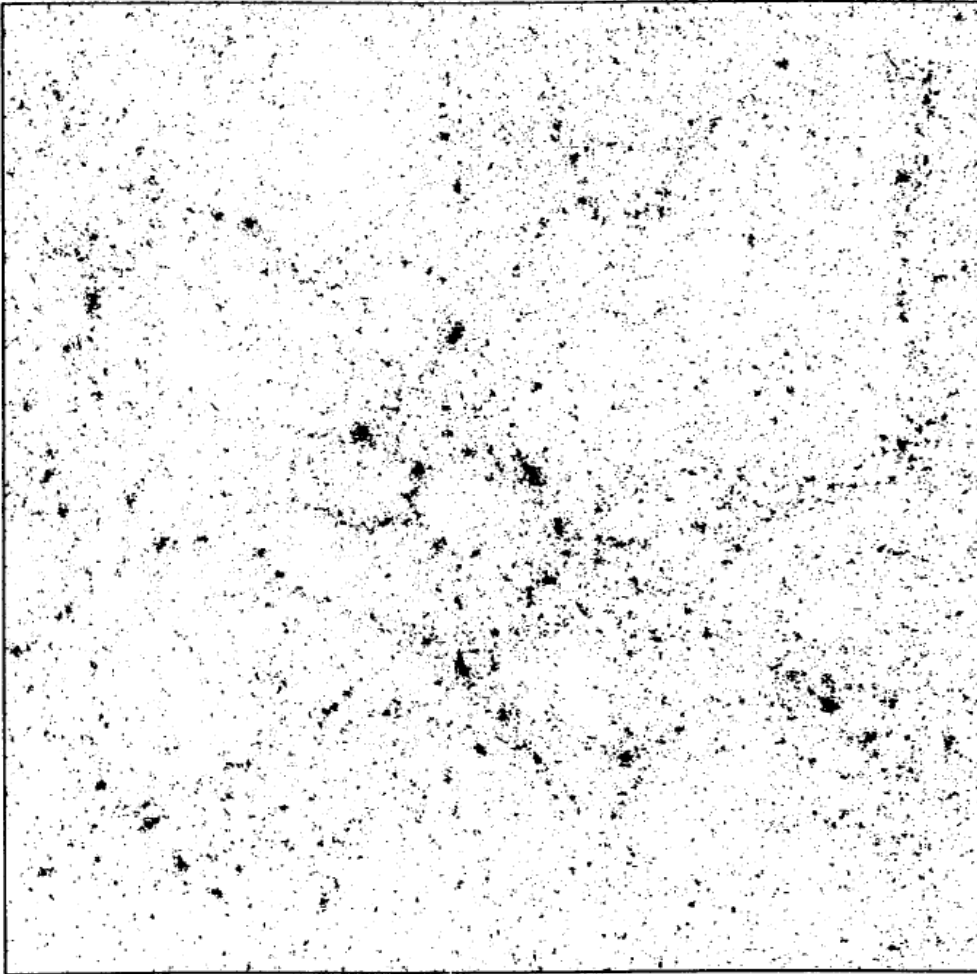


A model for the interaction of M51 and NGC 5195

More important than particle number is **physical insight and intuition**

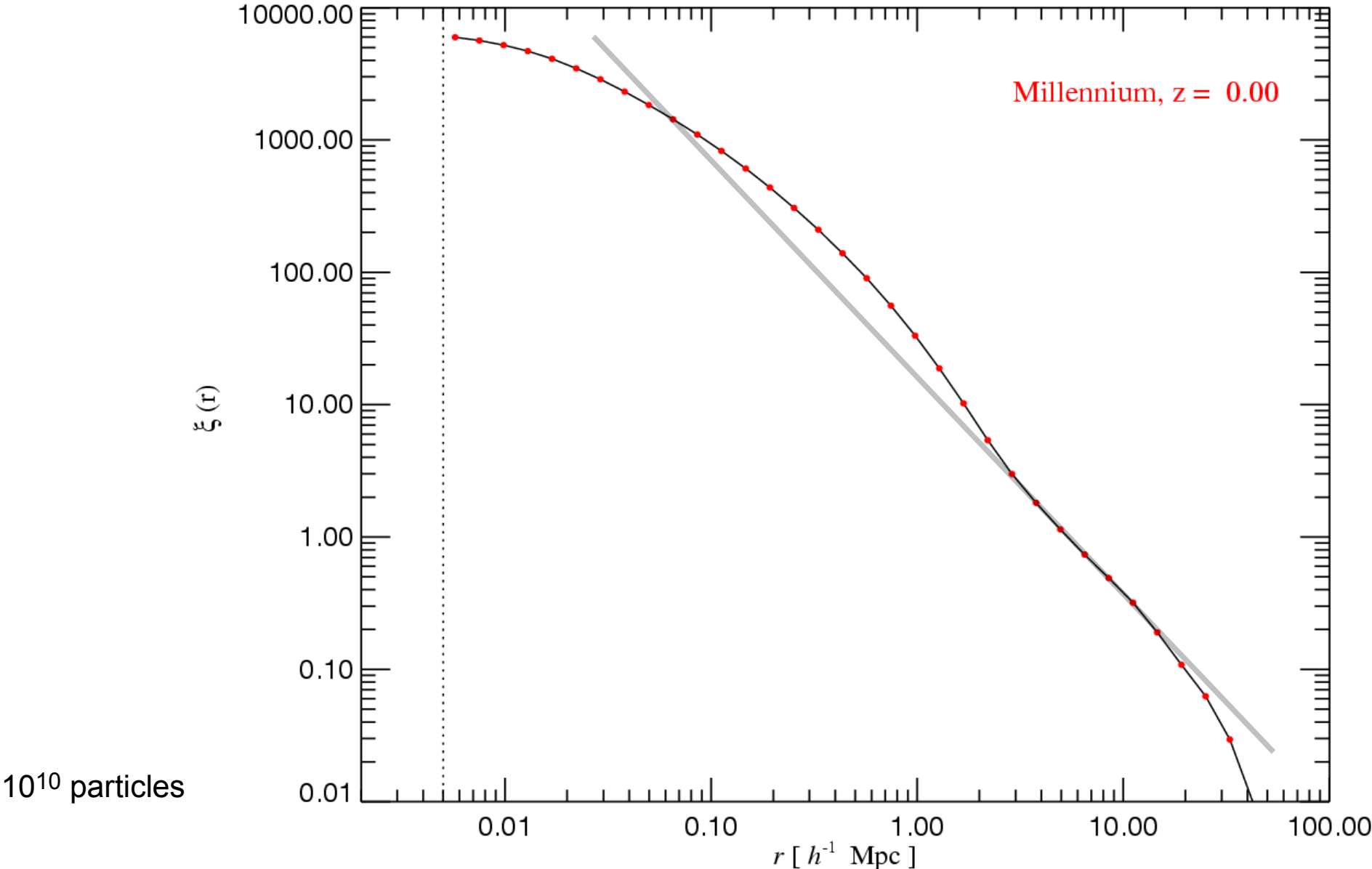
DAVIS, EFSTATHIOU, FRENK & WHITE (1985 !)

32^3 particles – the first generation of CDM simulations



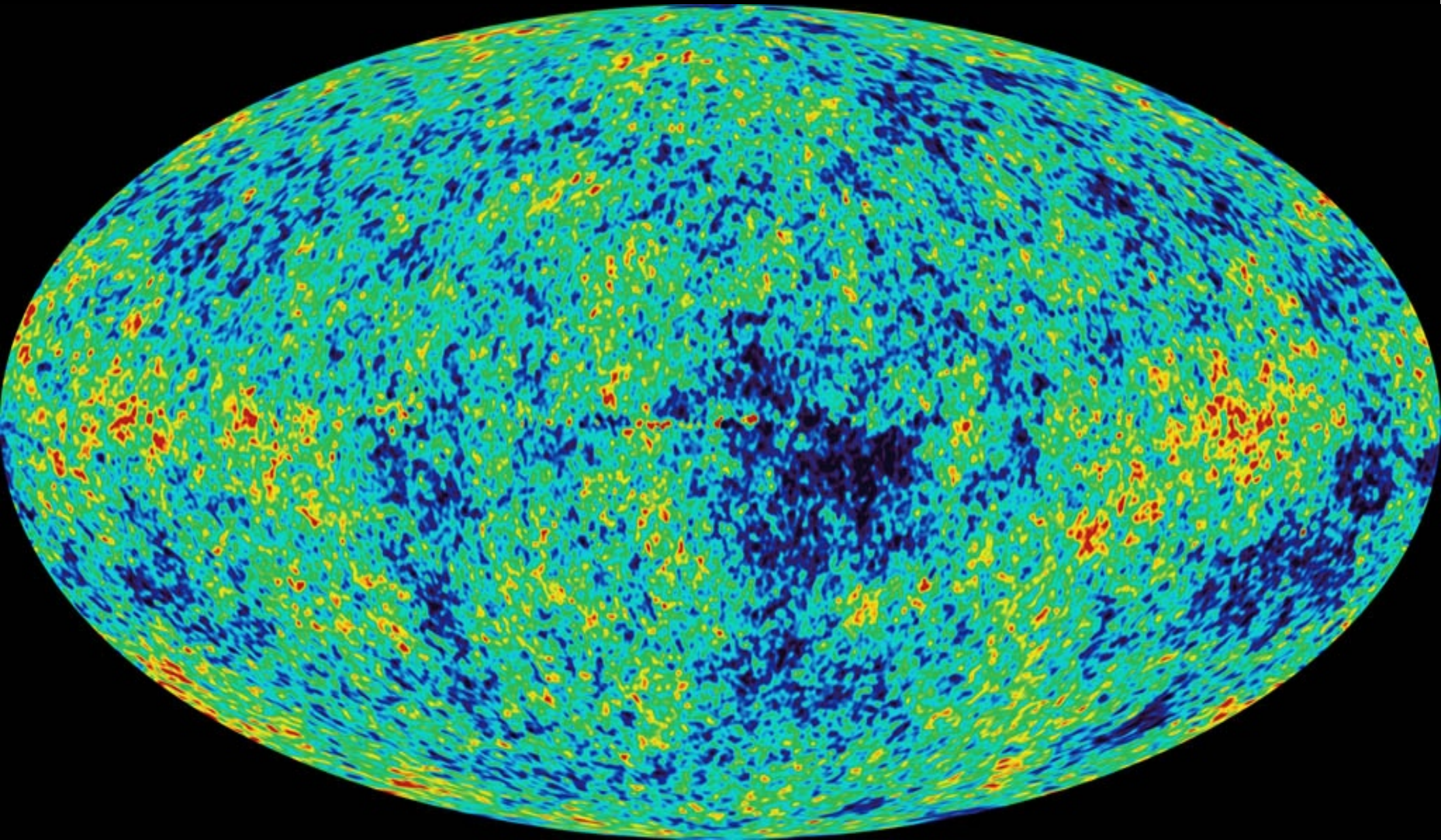
In modern simulations, the same dark matter autocorrelation function is measured, but more accurately

DARK MATTER TWO-POINT FUNCTION



The initial conditions for cosmic structure formation are directly observable

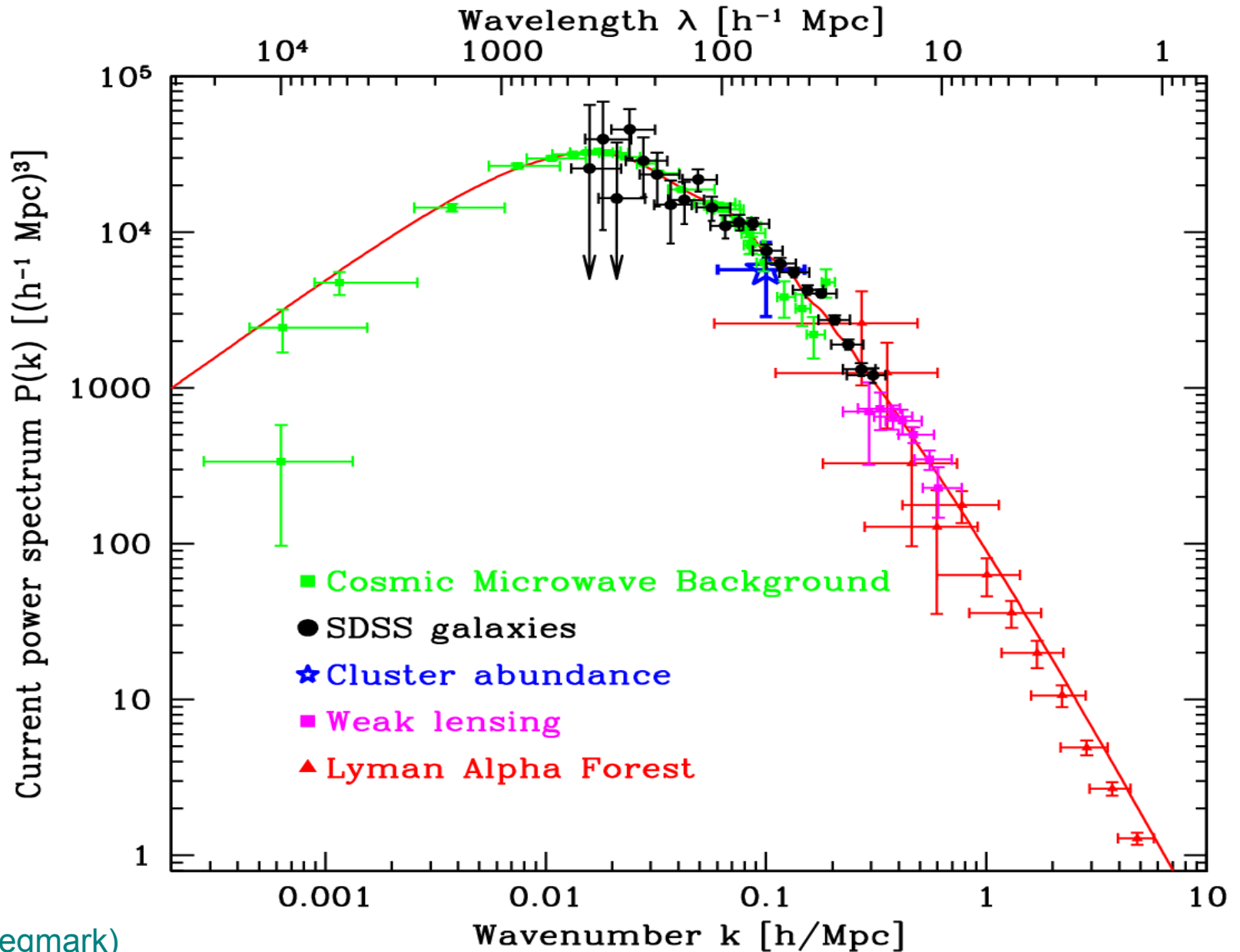
THE MICROWAVE SKY



WMAP Science Team (2003, 2006, 2008)

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

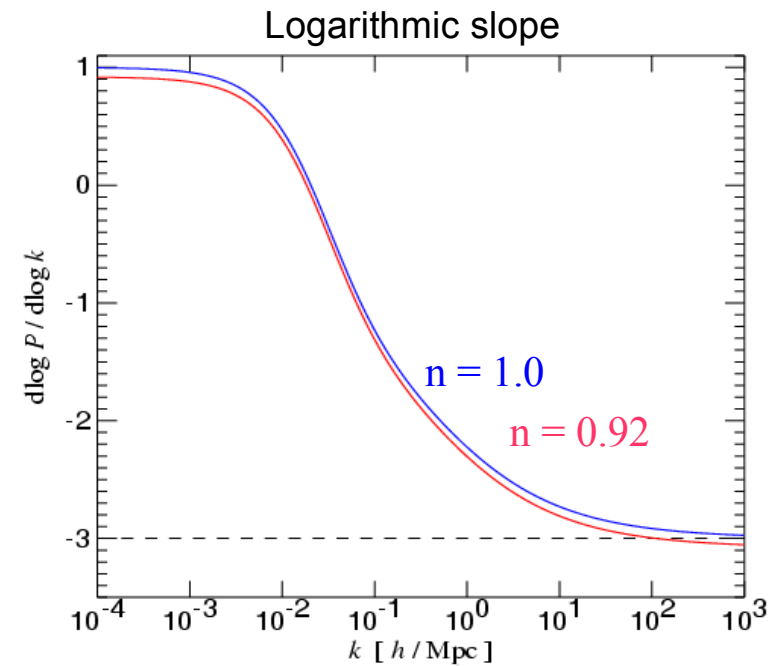
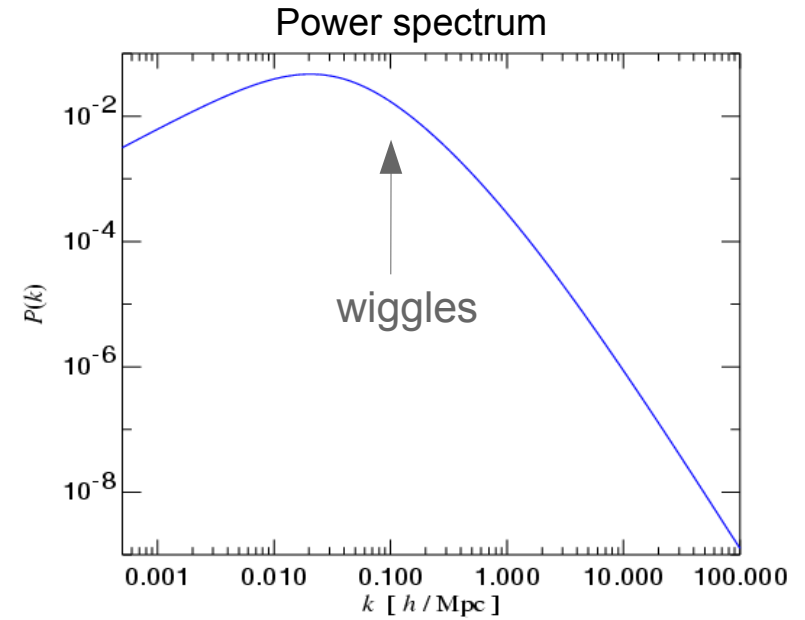
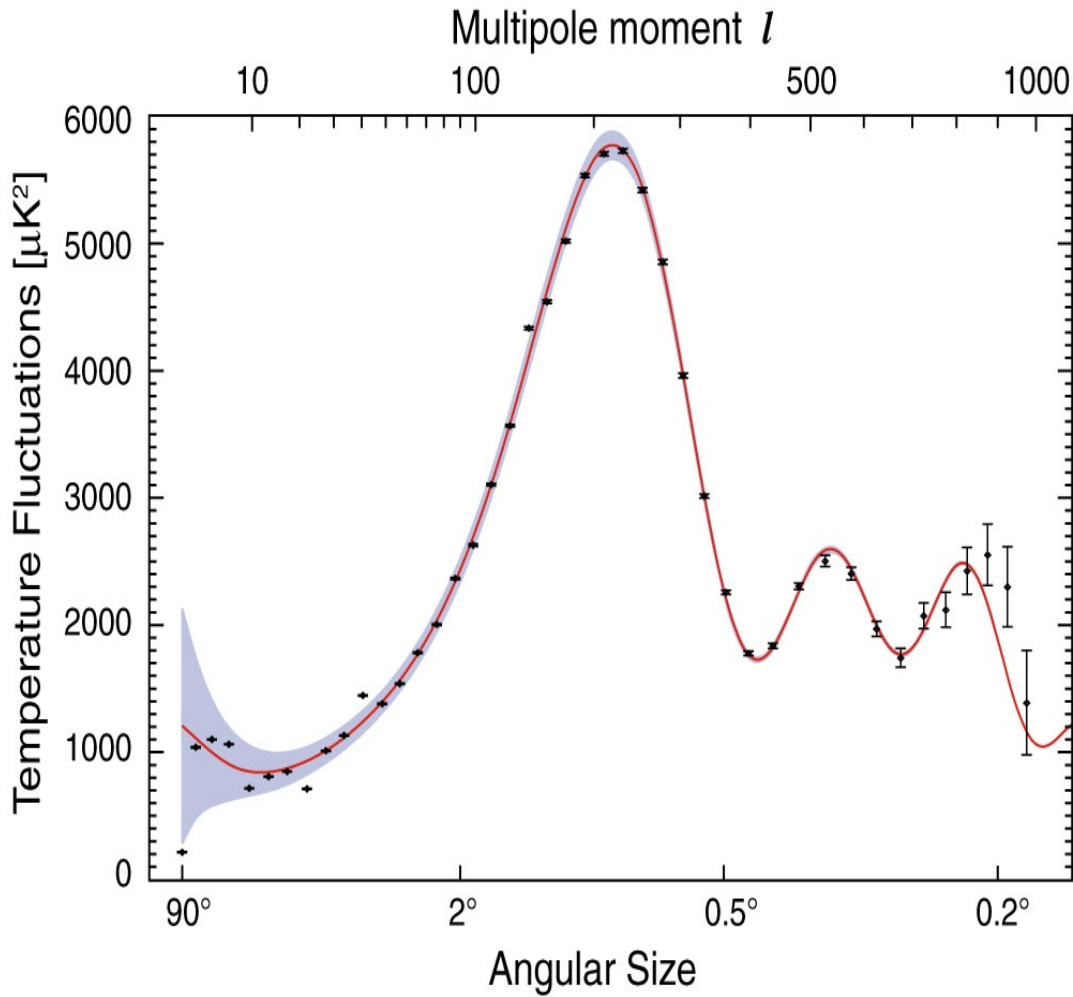
DIFFERENT PROBES OF THE MASS POWER SPECTRUM



(figure from Max Tegmark)

The linear theory power spectrum can be computed accurately

THE LINEAR POWER SPECTRUM



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

FILTERED DENSITY FIELD AND THE NORMALIZATION OF THE POWER SPECTRUM

The filtered density field:
$$\sigma^2(M, z) = D^2(z) \int_0^\infty \frac{dk}{2\pi^2} k^2 P(k) \left[\frac{3j_1(kR)}{kR} \right]^2$$

Observational input:
$$\sigma_8 = 0.74 - 0.9 \quad R = 8 h^{-1} \text{Mpc}$$

Extrapolate back to the starting redshift with the growth factor $D(z)$

This depends on cosmology.

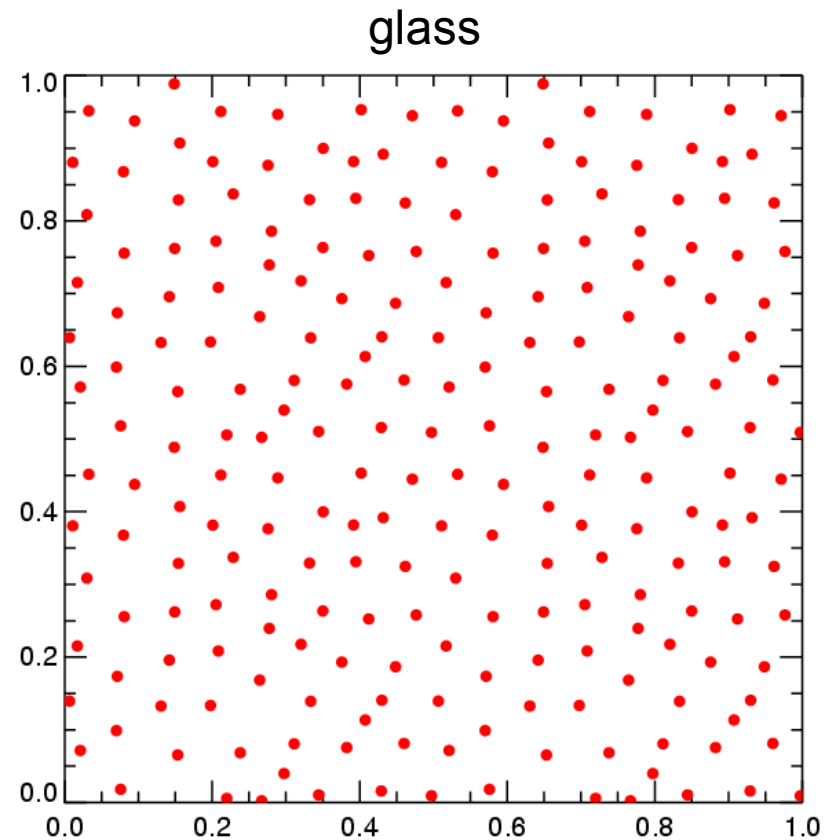
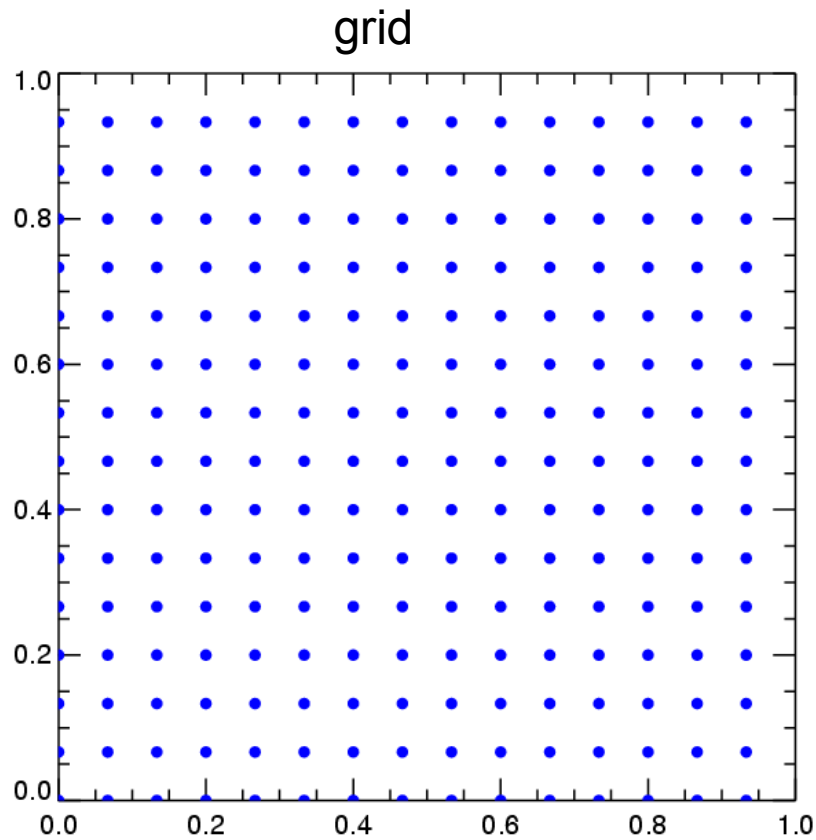
—————▶ fluctuation spectrum of initial conditions fully specified.

Aside:
$$P(k) \propto k^n \quad \rightarrow \quad \sigma^2(M) \propto M^{-(n+3)/3}$$

- Close to the critical slope, halos on very different mass scales form nearly simultaneously
- The multiplicity function of halos becomes essentially flat

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

GLASS OR CARTESIAN GRID



For CDM, the initial velocity dispersion is negligibly small.

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

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

SETTING INITIAL DISPLACEMENTS AND VELOCITIES

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

Density change due to displacements:

$$\rho(\mathbf{x}) = \frac{\rho_0}{\left| \frac{\partial \mathbf{x}}{\partial \mathbf{q}} \right|} = \frac{\rho_0}{\left| \delta_{ij} + \frac{\partial \mathbf{d}}{\partial \mathbf{q}} \right|}$$

For small displacements:

$$\left| \delta_{ij} + \frac{\partial \mathbf{d}}{\partial \mathbf{q}} \right| \simeq 1 + \nabla_{\mathbf{q}} \cdot \mathbf{d}$$

Resulting density contrast:

$$\delta(\mathbf{x}) = \frac{\rho(\mathbf{x}) - \rho_0}{\rho_0} = -\nabla_{\mathbf{q}} \cdot \mathbf{d}$$

During linear growth:

$$\begin{aligned} \delta(t) &= D(t) \delta_0 \\ \mathbf{d}(t) &= D(t) \mathbf{d}_0 \end{aligned} \quad \longrightarrow \quad \dot{\mathbf{x}} = \dot{\mathbf{d}} = \dot{a} \frac{dD}{da} \mathbf{d}_0 = \frac{\dot{a}}{a} \frac{a}{D} \frac{dD}{da} \mathbf{d}$$

Particle velocities:

$$\dot{\mathbf{x}} = H(a) f(\Omega) \mathbf{d} \quad f(\Omega) = \frac{d \ln D}{d \ln a} \simeq \Omega^{0.6}$$

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

Displacement field:

$$\nabla^2 \phi = \delta \quad \mathbf{d} = -\nabla \phi$$

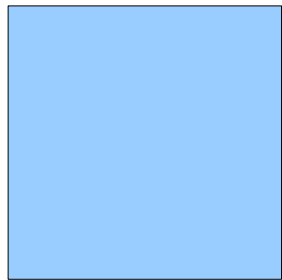
Fourier realization:

$$\phi_{\mathbf{k}} = -\frac{1}{k^2} \delta_{\mathbf{k}} \quad \mathbf{d}_{\mathbf{k}} = -i\mathbf{k} \phi_{\mathbf{k}} = \frac{i\mathbf{k}}{k^2} \delta_{\mathbf{k}} \quad \mathbf{d}_{\mathbf{k}} = -\nabla \phi = \sum_{\mathbf{k}} \frac{i\mathbf{k} \delta_{\mathbf{k}}}{k^2} \exp(i\mathbf{k}\mathbf{x})$$

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

GENERATING THE FLUCTUATIONS IN K-SPACE

Simulation box



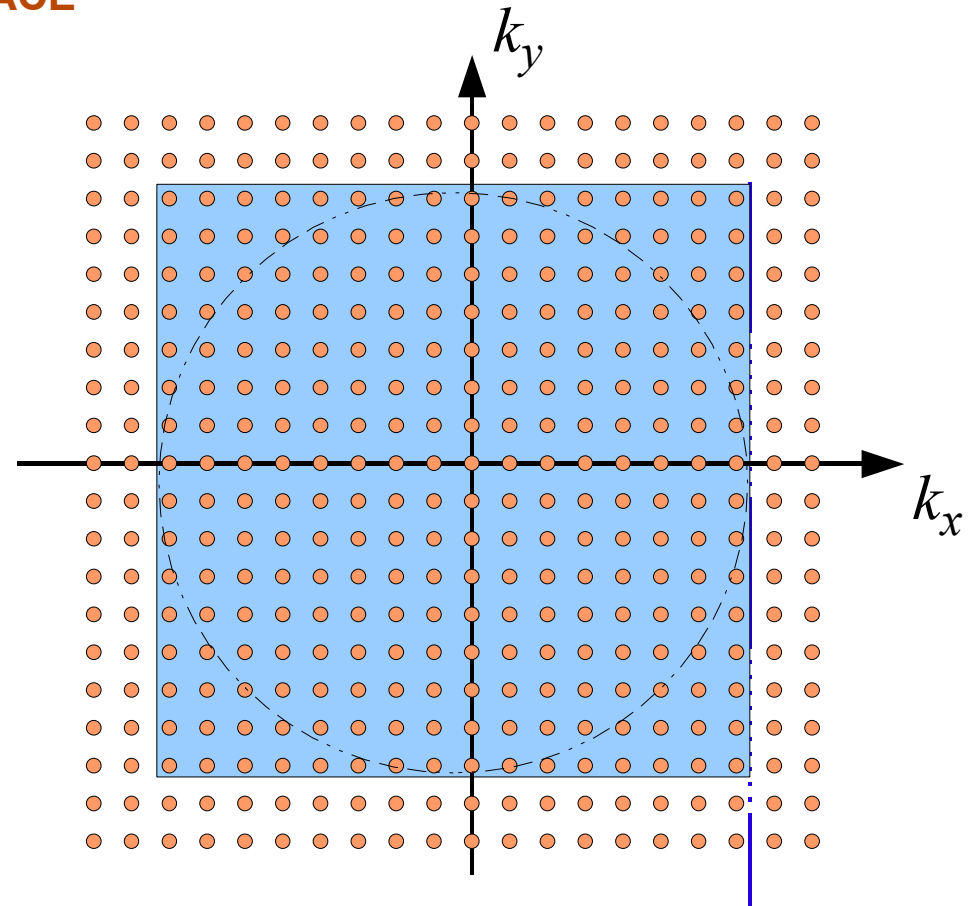
sampled with N^2 points

L

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

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

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



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

Calculating gravitational forces

Direct summation calculates the gravitational field **exactly**

FORCE ACCURACY IN COLLISIONLESS SIMULATIONS

Direct summation approach:

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

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

N^2 complexity

Are *approximate* force calculations sufficient?

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

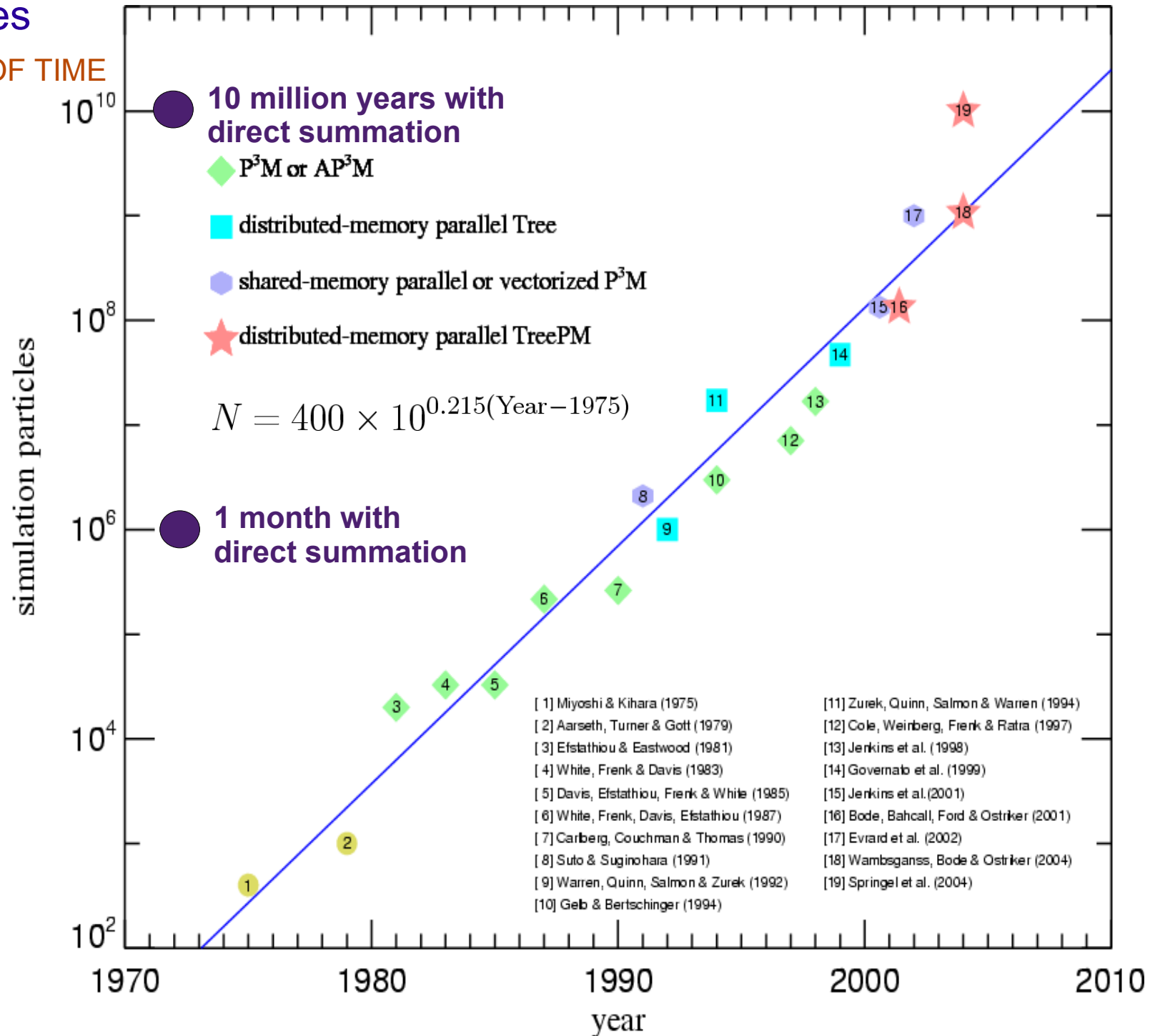
Since the N-body force field is noisy anyway, small random errors will only insignificantly reduce the relaxation time.

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

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

"N" AS A FUNCTION OF TIME

- ▶ Computers double their speed every 18 months (Moore's law)
- ▶ N-body simulations have doubled their size every 16-17 months
- ▶ Recently, growth has accelerated further.
The Millennium Run should have become possible in 2010 – we it was done in 2004.
It took ~350000 CPU hours, about a month on 512 cores.



The particle mesh (PM) force calculation

The particle-mesh method

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

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

Example for
vacuum boundaries:

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

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

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

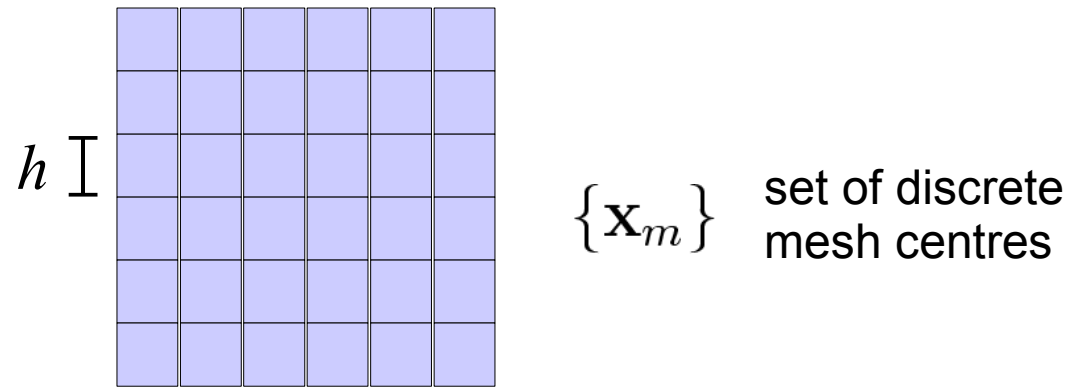
—► **Solve the potential in these steps:**

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

The four steps of the PM algorithm

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

Density assignment



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

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



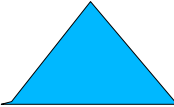
The assignment function is hence the convolution:

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

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

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

Commonly used particle shape functions and assignment schemes

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

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

Finite differencing of the potential to get the force field

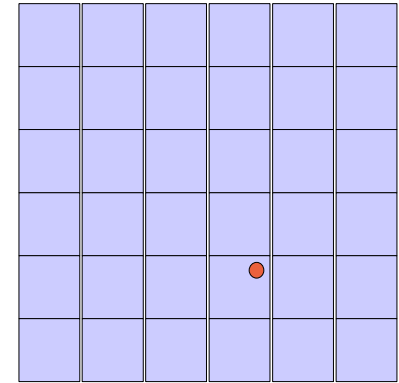
Approximate the force field $\mathbf{f} = -\nabla\Phi$ with finite differencing

2nd order accurate scheme:

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

4th order accurate scheme:

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



Interpolating the mesh-forces to the particle locations

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

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

Advantages and disadvantages of the PM-scheme

Pros: **SPEED** and simplicity

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



serious problem:

cosmological simulations cluster strongly and have a very large dynamic range

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



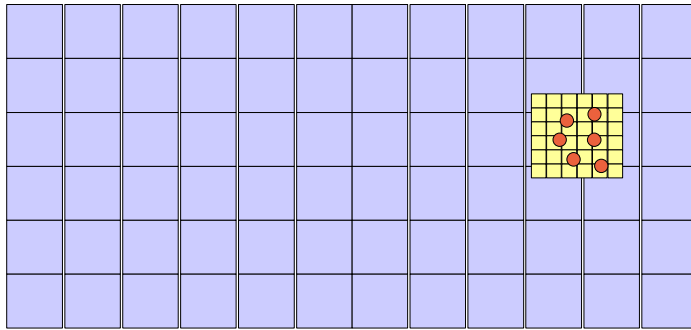
we need a method to increase the **dynamic range** available in the force calculation

Particle-Particle PM schemes (P³M)

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

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

In AP³M, mesh-refinements are placed on clustered regions



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

Codes that use AP³M: **HYDRA** (Couchman)

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

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

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

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

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

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

ART (Kravtsov)

MLAPM (Knebe)

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

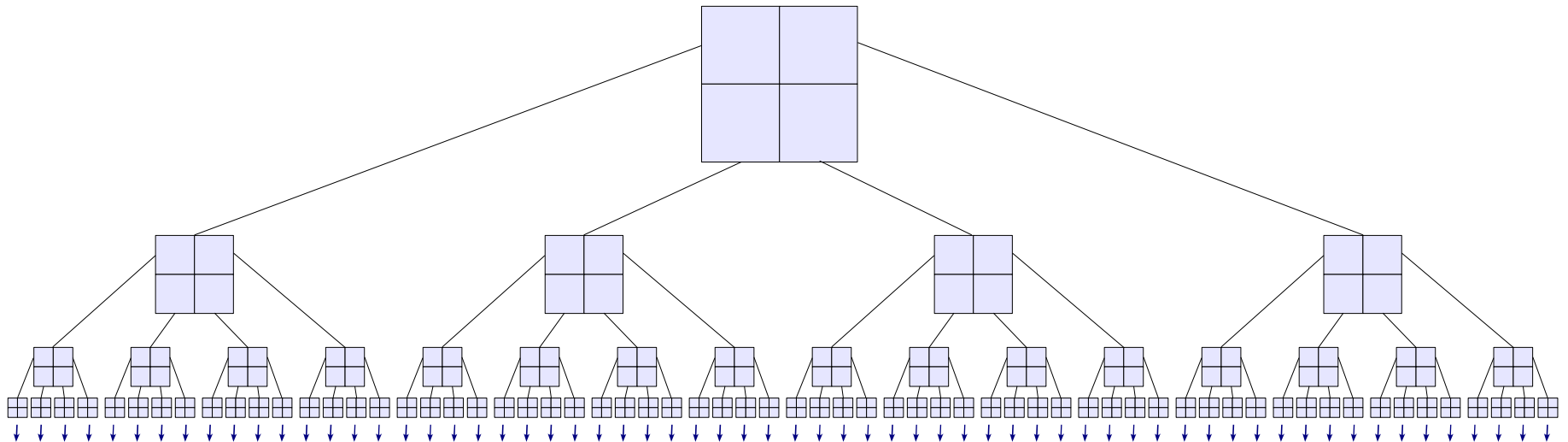
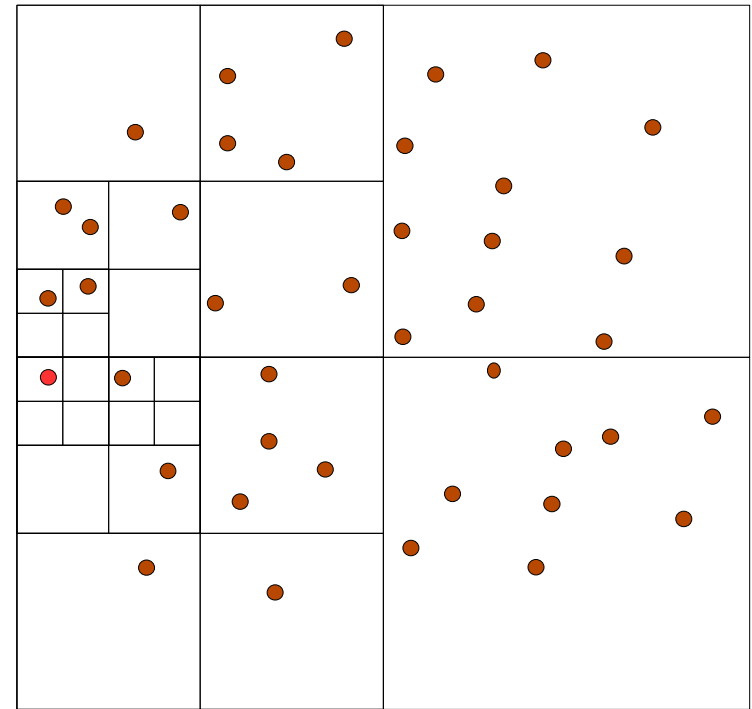
TREE algorithms

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

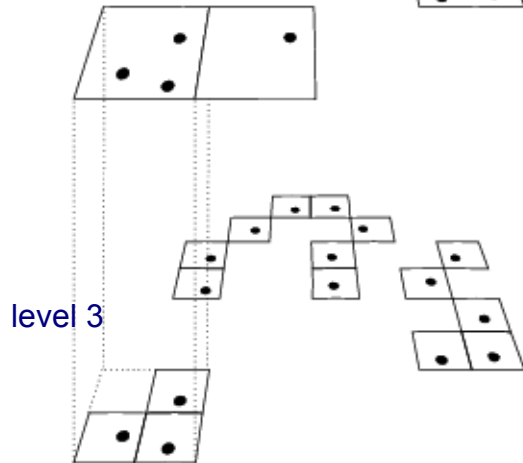
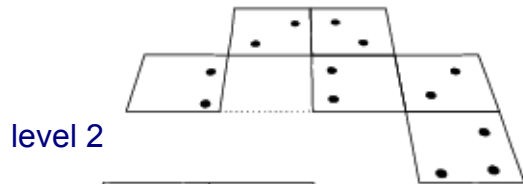
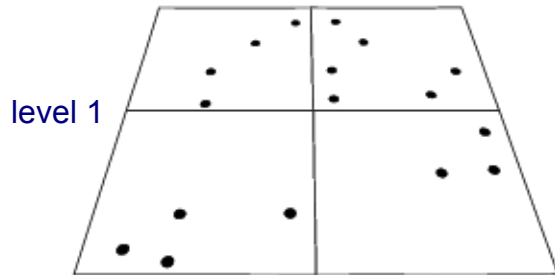
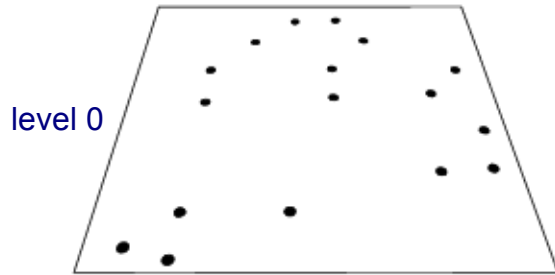
HIERARCHICAL TREE ALGORITHMS

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

→ Only $\sim \log(N)$ force terms per particle.



Oct-tree in two dimensions



Tree algorithms

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

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

We expand:

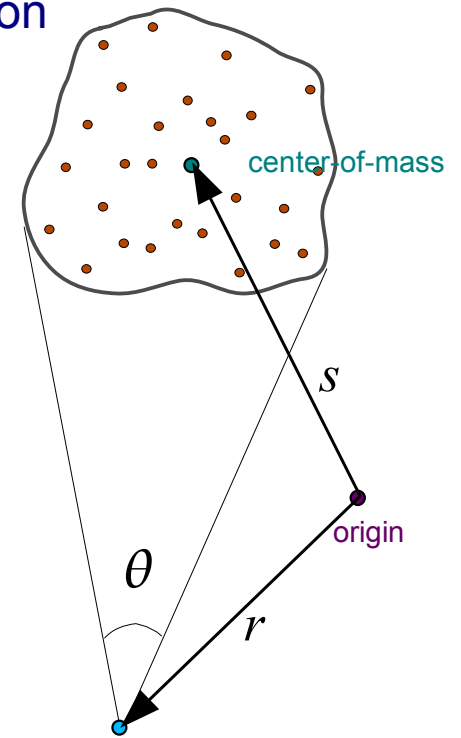
$$\frac{1}{|\mathbf{r} - \mathbf{x}_i|} = \frac{1}{|(\mathbf{r} - \mathbf{s}) - (\mathbf{x}_i - \mathbf{s})|}$$

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

and obtain:

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

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



The multipole moments are computed for each node of the tree

Monpole moment:

$$M = \sum_i m_i$$

Quadrupole tensor:

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

Resulting potential/force approximation:

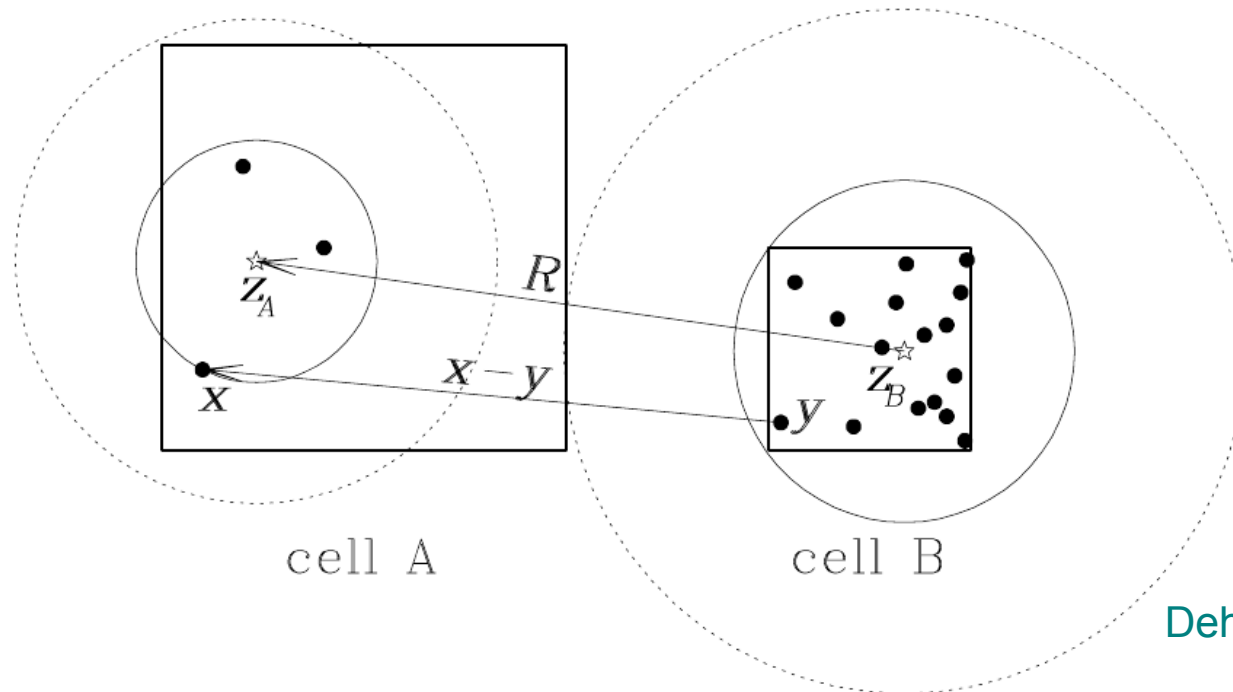
$$\Phi(\mathbf{r}) = -G \left[\frac{M}{|\mathbf{y}|} + \frac{1}{2} \frac{\mathbf{y}^T \mathbf{Q} \mathbf{y}}{|\mathbf{y}|^5} \right]$$

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

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

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

Two interacting cells:



Dehnen (2002)

- Very fast
- Manifest momentum conservation

But:

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

TreePM force calculation algorithm

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

THE TREE-PM FORCE SPLIT

Periodic peculiar potential

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

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

Poisson equation in Fourier space:

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

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

Solve with PM-method

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

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

FFT to real space

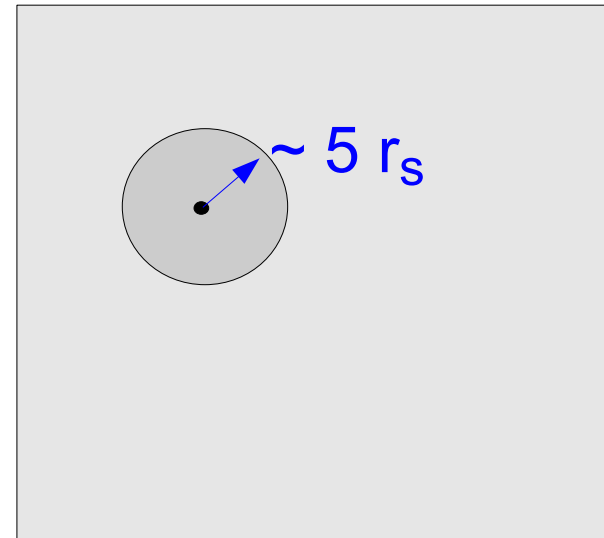
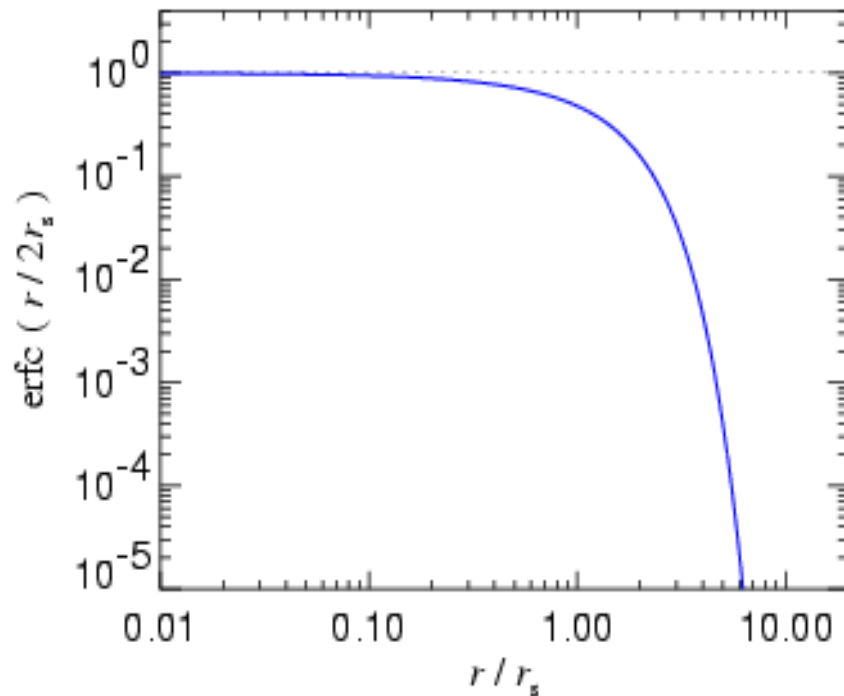
$$\phi(r) = -\frac{Gm}{r} \operatorname{erfc}\left(\frac{r}{2r_s}\right)$$

Solve in real space with TREE

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

PERFORMANCE GAIN DUE TO LOCAL TREE WALK

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



Advantages of TreePM include:

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

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

Brief comments on time integration

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

THE LEAPFROG AS A SYMPLECTIC INTEGRATOR

Separable Hamiltonian

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

Drift- and Kick-Operators

$$\mathbf{D}(\Delta t) \equiv \exp \left(\int_t^{t+\Delta t} dt \mathbf{H}_{\text{kin}} \right) = \begin{cases} \mathbf{p}_i \mapsto \mathbf{p}_i \\ \mathbf{x}_i \mapsto \mathbf{x}_i + \frac{\mathbf{p}_i}{m_i} \Delta t \end{cases}$$

$$\mathbf{K}(\Delta t) = \exp \left(\int_t^{t+\Delta t} dt \mathbf{H}_{\text{pot}} \right) = \begin{cases} \mathbf{x}_i \mapsto \mathbf{x}_i \\ \mathbf{p}_i \mapsto \mathbf{p}_i - \sum_j m_i m_j \frac{\partial \phi(\mathbf{x}_{ij})}{\partial \mathbf{x}_i} \Delta t \end{cases}$$

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

The Leapfrog

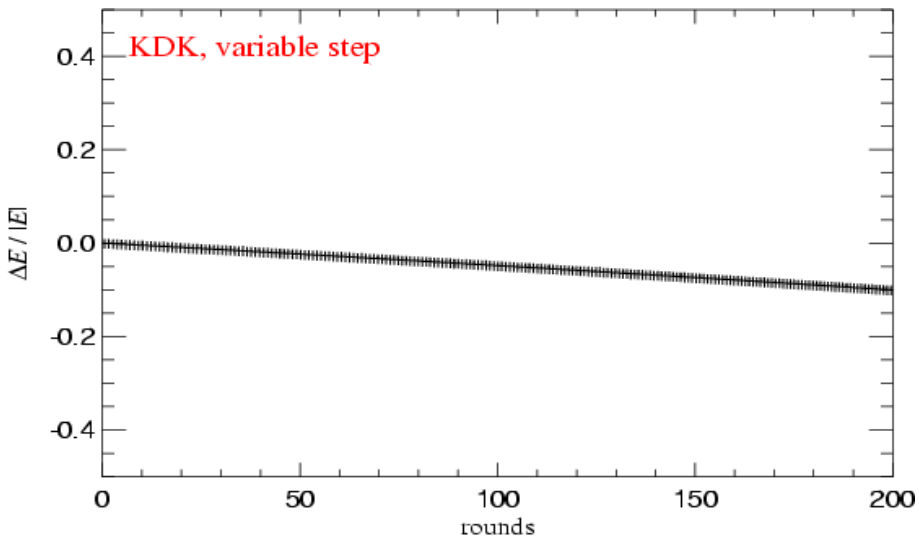
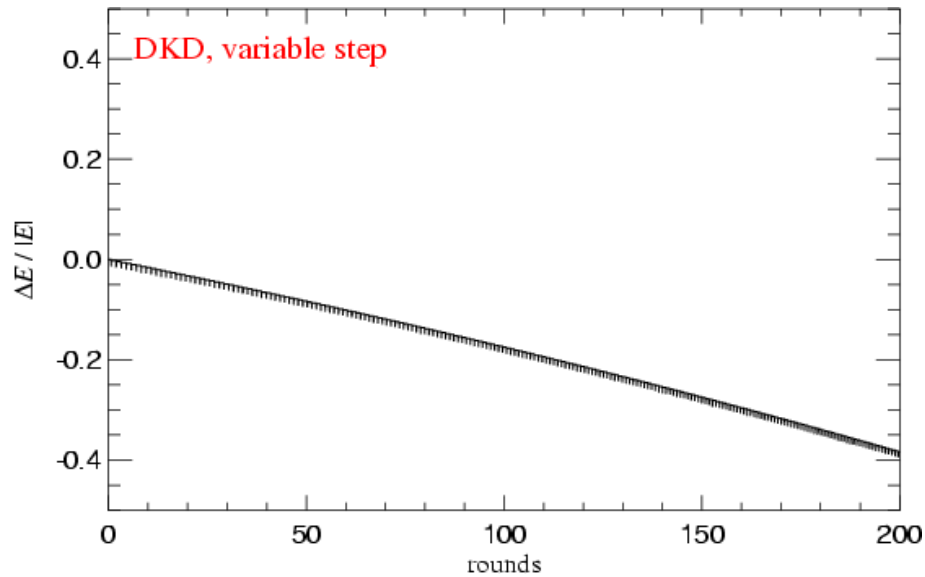
Drift-Kick-Drift: $\tilde{\mathbf{U}}(\Delta t) = \mathbf{D} \left(\frac{\Delta t}{2} \right) \mathbf{K}(\Delta t) \mathbf{D} \left(\frac{\Delta t}{2} \right)$

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

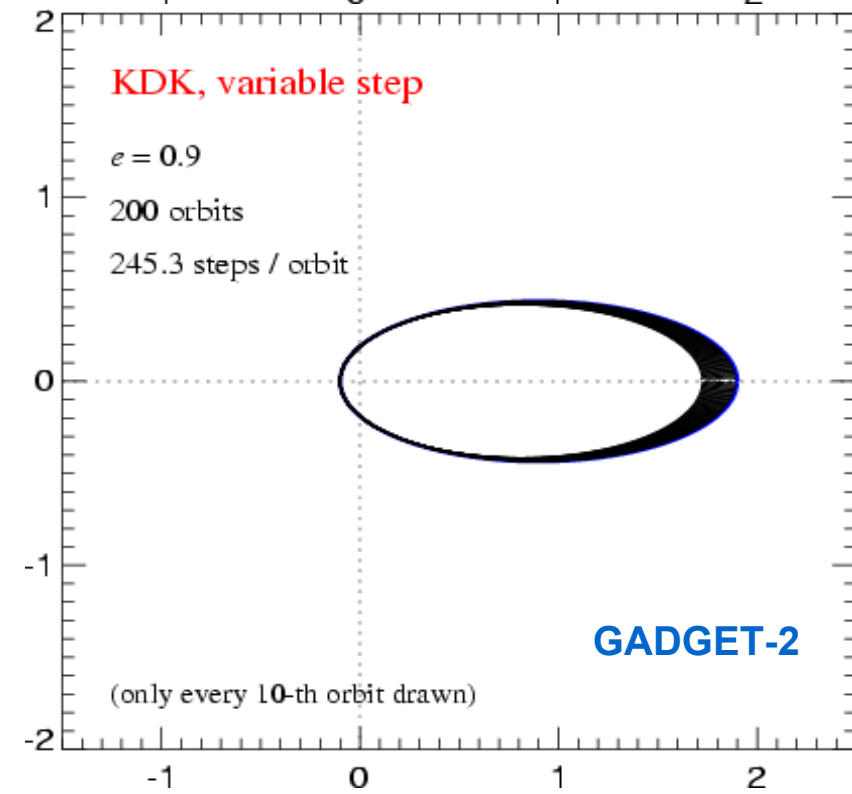
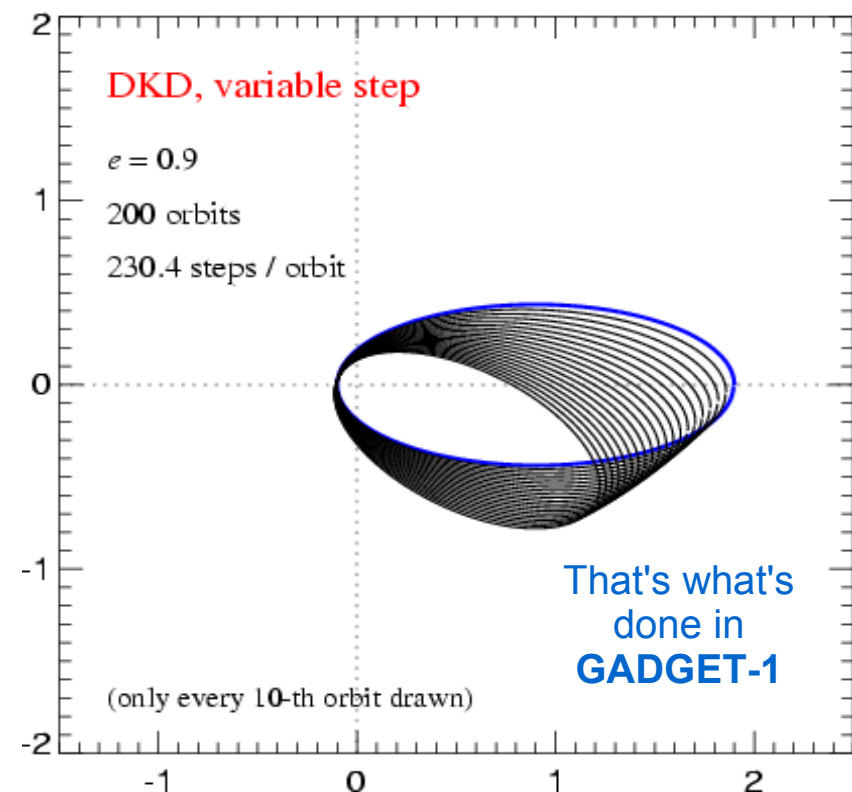
Hamiltonian of the numerical system: $\tilde{H} = H + H_{\text{err}} \quad H_{\text{err}} = \frac{\Delta t^2}{12} \left\{ \{H_{\text{kin}}, H_{\text{pot}}\}, H_{\text{kin}} + \frac{1}{2} H_{\text{pot}} \right\} + \mathcal{O}(\Delta t^3)$

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

INTEGRATING THE KEPLER PROBLEM



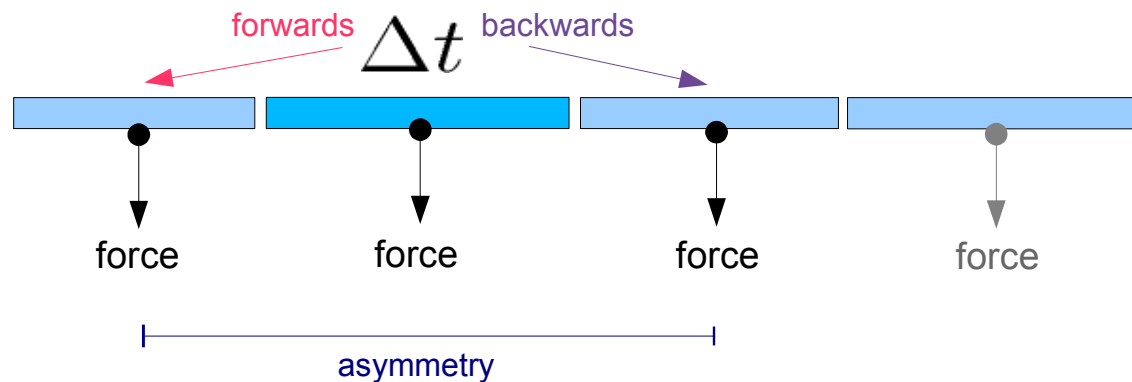
→ Going to KDK reduces the error by a factor 4, at the same cost !



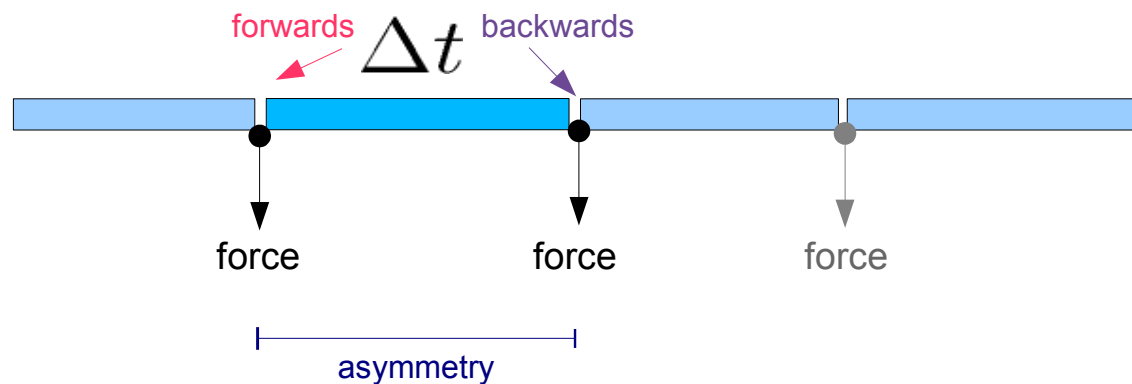
For periodic motion with adaptive timesteps, the DKD leapfrog shows more time-asymmetry than the KDK variant

LEAPFROG WITH ADAPTIVE TIMESTEP

DKD



KDK



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

THE HAMILTONIAN IN COMOVING COORDINATES

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

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

Drift- and Kick operators

$$\mathbf{D}(t + \Delta t, t) = \exp \left(\int_t^{t+\Delta t} dt \mathbf{H}_{\text{kin}} \right) = \begin{cases} \mathbf{p}_i \mapsto \mathbf{p}_i \\ \mathbf{x}_i \mapsto \mathbf{x}_i + \frac{\mathbf{p}_i}{m_i} \int_t^{t+\Delta t} \frac{dt}{a^2} \end{cases}$$

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

Choice of timestep

For linear growth, fixed step in $\log(a)$ appears most appropriate...



timestep is then a constant fraction of the Hubble time

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

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

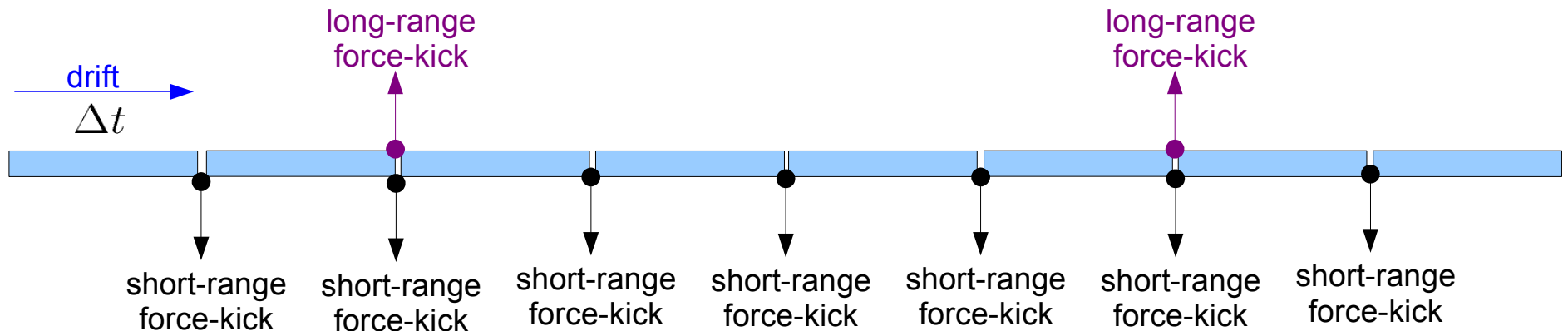
TIME INTEGRATION FOR LONG AND SHORT-RANGE FORCES

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

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

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

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



Parallelization: Domain decomposition

The maximum size of collisionless dark matter simulations with the TreePM algorithm is essentially memory bound

MEMORY REQUIREMENTS IN DIFFERENT PARTS OF THE (LEAN) GADGET-2 CODE

Particle Data

44 bytes / particle

Tree storage

40 bytes / particle

FFT workspace

24 bytes / mesh-cell

Not needed concurrently!

Special code version

84 bytes / particle

Lean-GADGET-II needs:

(Assuming 1.5 mesh-cells/particle)

Example: Simulation set-up of the Millennium Run

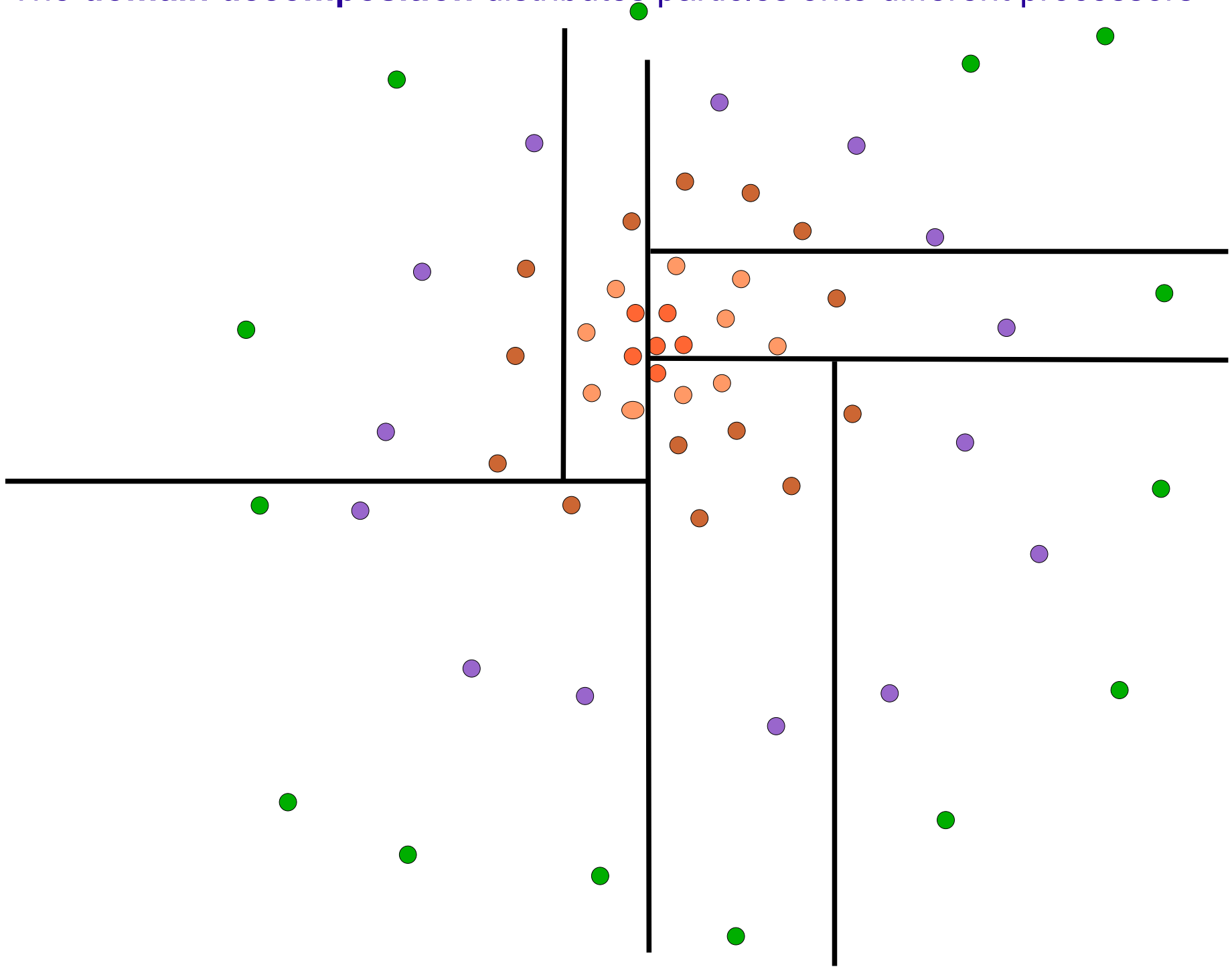
- Particle number: $2160^3 = 10.077.696.000 = \sim 10^{10}$ particles
- Boxsize: $L = 500 h^{-1} \text{ Mpc}$
- Particle mass: $m_p = 8.6 \times 10^8 h^{-1} M_\odot$
- Spatial resolution: $5 h^{-1} \text{ kpc}$
- Size of FFT: $2560^3 = 16.777.216.000 = \sim 17$ billion cells

Minimum memory requirement
of simulation code

~840 GByte

We will do ~200 billion particles soon – but how can we cope with the data?

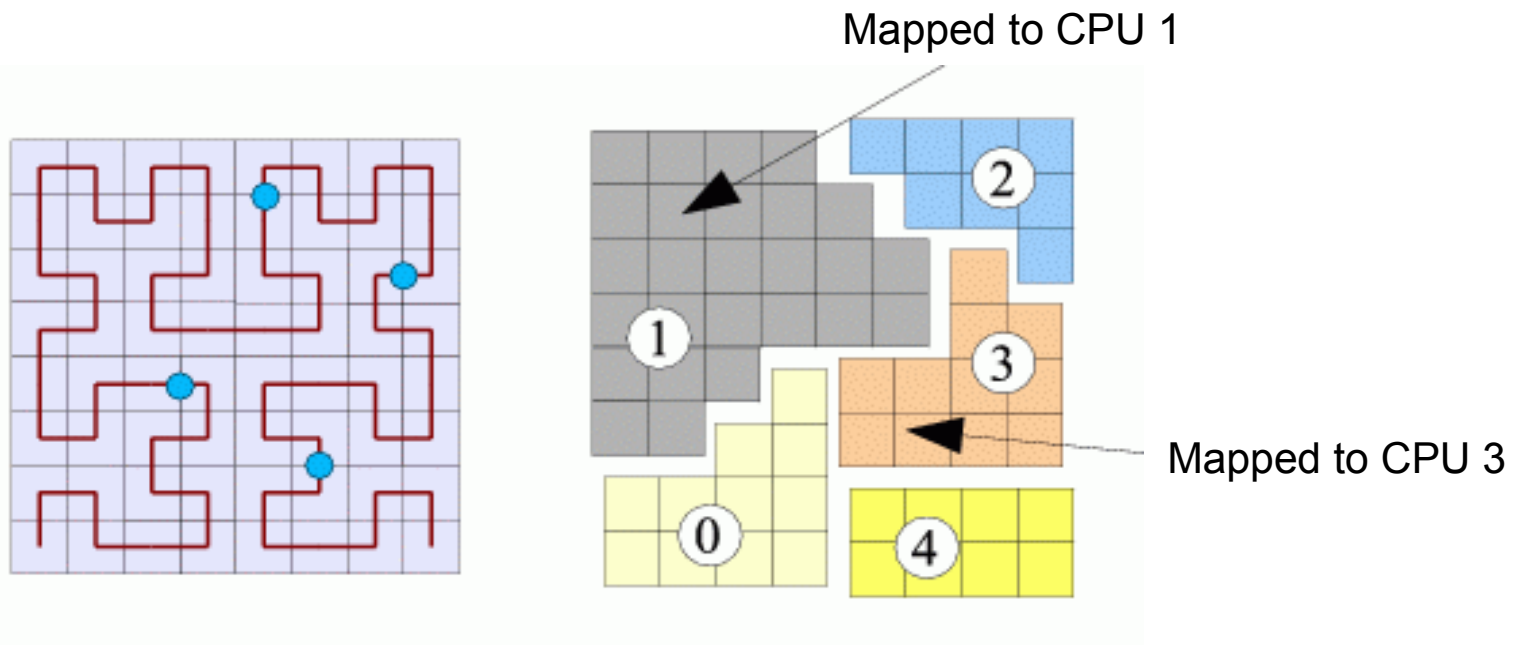
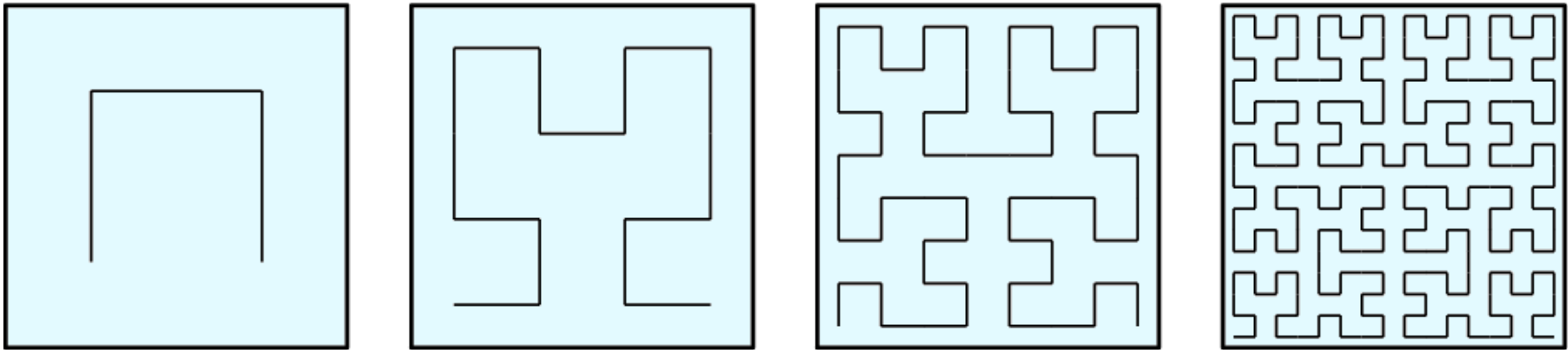
The **domain decomposition** distributes particles onto different processors



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

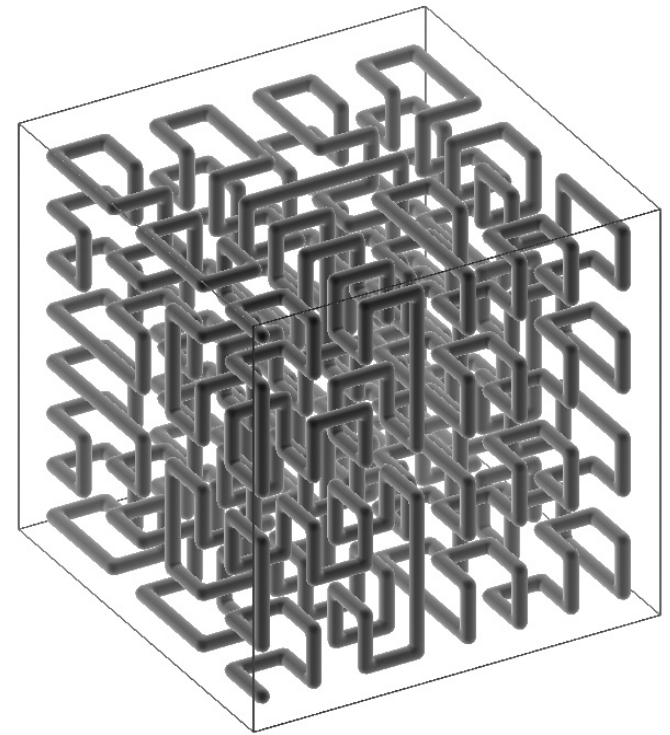
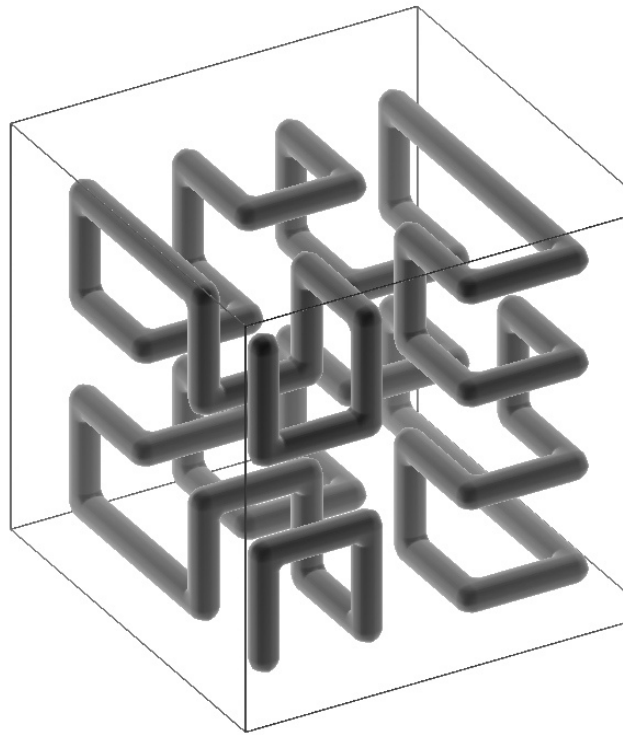
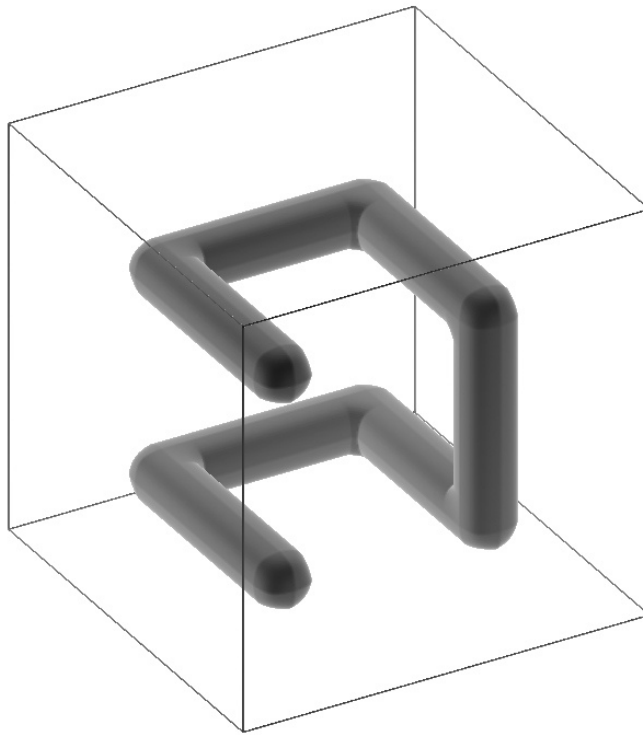
CONSTRUCTION OF A FLEXIBLE DOMAIN DECOMPOSITION WITH CACHE BENEFITS

Idea: Order the particles along a space-filling curve



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

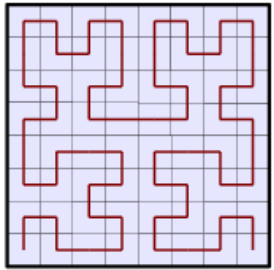
THE PEANO-HILBERT CURVE



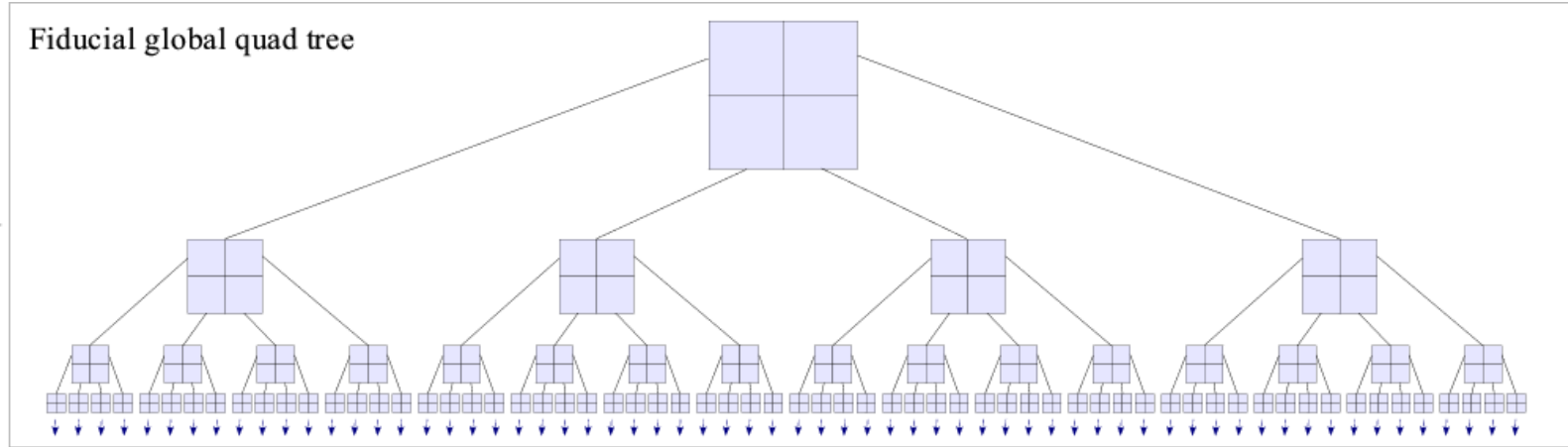
The space-filling Peano-Hilbert is used in GADGET-2 for the domain-decomposition

SPLITTING UP THE TREE FOR DIFFERENT PROCESSORS

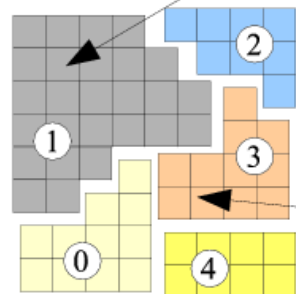
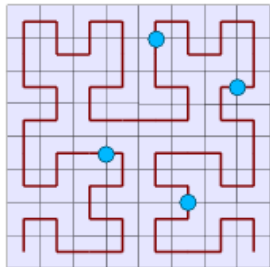
Peano-Hilbert curve



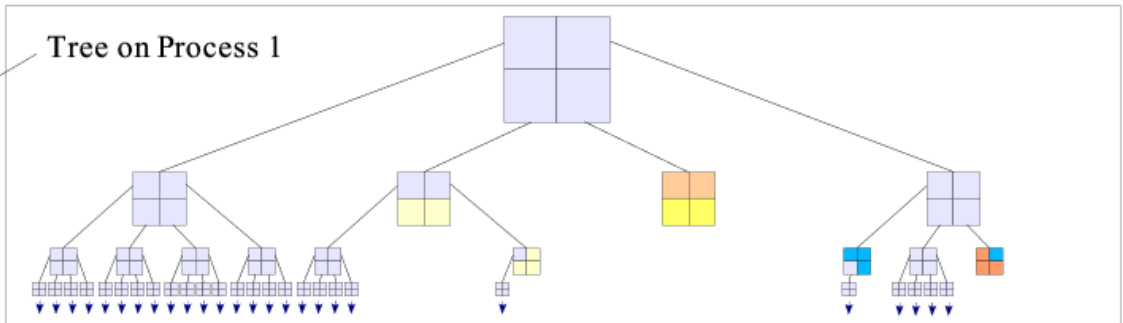
Fiducial global quad tree



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



Tree on Process 1



Tree on Process 3

