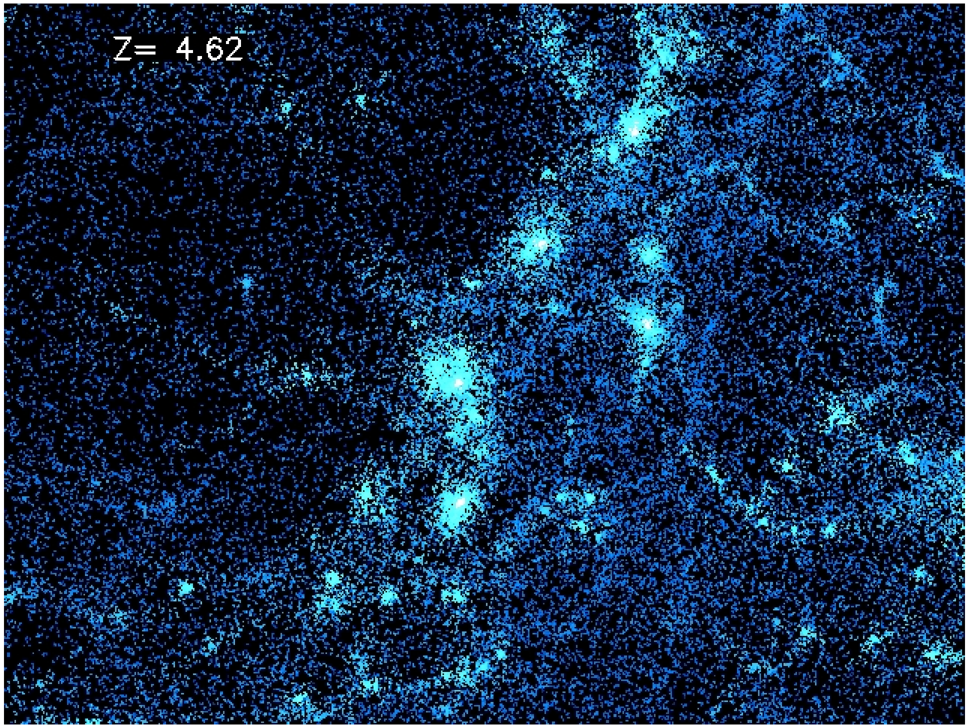
Writing a PM code

Andrey Kravtsov
andrey@oddjob.uchicago.edu
astro.uchicago.edu/˜andrey/talks/PM/pmpotsdam.pdf

July 24, 2006



Cosmological simulations: the problem

- The Universe is thought to be composed of dark matter and normal matter (the baryons), photons, neutrinos, etc.

- There should be $\sim 10^{80 \div 90}$ particles of dark matter and baryons in the observable Universe.

- In our Galaxy, the Milky Way, alone there are $\sim 10^{68}$ baryonic particles.

- Simulations, of course, cannot follow evolution of all of these particles.

- The evolution of a large number of particles of a given type can be described as evolution of a continuous field, characterized by the distribution function $f(\mathbf{x}, \dot{\mathbf{x}}, t)$ — the density of particles in phase space.

Cosmological simulations: equations I

- The evolution of f_i is described by the Boltzmann equation:

$$\frac{\partial f_i}{\partial t} + \dot{\mathbf{x}}\frac{\partial f_i}{\partial \mathbf{x}} + \ddot{\mathbf{x}}\frac{\partial f_i}{\partial \dot{\mathbf{x}}} = C\left[f_i, f_j\right];$$

- $C[f_i, f_j]$ is the collision integral, describing possible particle collisions, which can scatter particles in and out of a phase-space volume. The form of C depends on the specific properties of particles in the system (i.e., how they scatter, whether they can be created or annihilated, etc.)

- In general, a system of 6D Boltzmann equations (equations for each of the relevant components, i, of the Universe) needs to be solved.

- This is indeed done in codes describing evolution of the primordial matter and linear perturbations to solve for the initial power spectrum and properties of the CMB temperature anisotropies. In this case, the matter distribution is close to uniform and no detailed information on the 3D distribution of matter is needed. The dimensionality of the problem can thus be reduced.

Cosmological simulations: equations II

- When solving for nonlinear structure formation, however, the full 3D spatial information is needed.

- If dark matter is collisionless or nearly collisionless, $C[f] = 0$, resulting in a simpler, *collisionless Boltzmann equation*, a.k.a. *the Vlasov equation*. However, this "simpler" equation is still a 6D ODE, if we want to describe evolution of the matter in 3D space. High-dimensionality of the equation severely limits the resolution of simulations due to the large number of elements required to cover the 6D space.

- The solution of the Vlasov equation can be represented in terms of characteristic equations, which look like equations of particle motion. Characteristic equations describe lines in phase space along which the distribution function is constant.

- A complete set of characteristic equations is equivalent to the Vlasov equation. But if a representative subsample of characteristics can be followed, we can get the approximate evolution of the system.

Cosmological simulations: N-body approach

- This is the main idea behind the particle N-body approach in plasma and cosmological simulations.

- In N-body approach, the initial phase space is split into small domains, each domain is represented by a particle, and particles are evolved self-consistently using equations of particle motion:

$$\frac{d\mathbf{x}}{dt} = \mathbf{u}, \quad \frac{d\mathbf{u}}{dt} = -\nabla\Phi$$

with potential given by the Poisson equation:

$$\nabla^2\Phi = 4\pi G\rho_{\rm tot}.$$

- N-body approach is effectively a Monte Carlo technique for random sampling of the characteristics.[1]

[1] Monte Carlo techniques are very often used in the otherwise intractable high-dimensional problems, like evaluating many-dimensional integrals or likelihood and parameter estimation in many-parameter problems. Markov Chain Monte Carlo (MCMC) methods are used in such problems.

Cosmological simulations: particle initial conditions

- If we deal with cold dark matter, initial particle velocities are due solely to gravity and, in the Zeldovich approximation, depend only on particle positions. In this case only 3D space needs to be discretized.

- For other type of dark matter (hot, warm, etc.), their initial velocity distribution must also be sampled.

- The problem of initial discretization is the problem of cosmological initial conditions, covered last week. For additional information see:

  Anatoly Klypin's PM code

  Ed Bertschinger's GRAFIC2 package: a public cosmological IC code

  Grocce et al. 2006, (astro-ph/0606505)

- the number of particles and the size of simulation volume are dictated by the available computational resources and the needs of particular problem.

Cosmological simulations: particle initial conditions

"It seems probable to me that God in the beginning formed matter in solid, massy, hard, impenetrable, movable particles, of such sizes and figures, and with such other properties, and in such proportion to space, as most conduced to the end for which he formed them."

- Isaac Newton, "Opticks"

Cosmological simulations: equations III: gasdynamics

- In a fully collisional particle systems with properties of an ideal gas, the Boltzmann equation can be simplified and reduced to 3D ODEs describing ideal compressible gas.

- This is done by taking the three first moments of the Boltzmann equations — i.e., multiplying the equation by 1, $\dot{\mathbf{x}}$, and $\dot{x}^2$ and integrating over velocity space).

Cosmological simulations: equations III: gasdynamics

- These moments give the continuity, Euler, and energy equations of hydrodynamics[2] (e.g., *F. Shu "Gasdynamics"*):

$$\frac{\partial \rho}{\partial t} + \nabla \rho \mathbf{u} = 0,$$

$$\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u}\cdot\nabla)\mathbf{u} = -\nabla\Phi - \frac{\nabla P}{\rho},$$
$$\frac{\partial E}{\partial t} + \nabla\cdot[(E+P)\mathbf{u}] = -\rho\mathbf{u}\cdot\nabla\Phi.$$

- The equations above are closed with the equation of state, which is usually assumed to be that of an ideal gas:

$$\varepsilon = \frac{1}{\gamma-1}\frac{P}{\rho},$$

or to have some other simple form (i.e., polytropic EoS: $P = K\rho^{\gamma}$)

[2]Equations below do not take into account additional processes commonly included in simulations: heating and cooling of gas, star formation, etc. These are included as sink and source terms in the appropriate equations. We will consider such processes on friday.

Cosmological simulations: equations summary

- Equations of particle motion, the Poisson equation, the equations of gasdynamics above, along with the equation of state of the gas, form the set of basic equations solved in cosmological simulations:

$$\frac{d\mathbf{x}}{dt} = \mathbf{u}, \quad \frac{d\mathbf{u}}{dt} = -\nabla\Phi$$

$$\nabla^2\Phi = 4\pi G\rho_{\rm tot}.$$

$$\frac{\partial\rho}{\partial t} + \nabla\rho\mathbf{u} = 0,$$

$$\frac{\partial\mathbf{u}}{\partial t} + (\mathbf{u}\cdot\nabla)\mathbf{u} = -\nabla\Phi - \frac{\nabla P}{\rho},$$

$$\frac{\partial E}{\partial t} + \nabla\cdot[(E+P)\mathbf{u}] = -\rho\mathbf{u}\cdot\nabla\Phi,$$

$$\varepsilon = f(\rho, P).$$

- Baryonic gas and dark matter are coupled by gravity, as they are both influenced by the same gravitational potential and they both contribute to the density in the Poisson equation.

- Note that all of the equations we solve to describe the evolution of matter in the universe have their usual simple non-relativistic form. This is because most commonly the matter components relevant for late evolution move with non-relativistic velocities.

Cosmological simulations: dissipationless case

- For many problems it is warranted to ignore the baryonic effects and solve only evolution of dissipationless component (dark matter, stars).

- In this case, only equations of particle motion and the Poisson equation are solved.

- Very generally, each evolution time step of a dissipationless simulation consists of the following two main parts:

  — *gravity solver:* solve the Poisson equation and calculate particle accelerations.

  — *time integrator:* update particle positions and velocities.

- different codes differ in how these two parts are implemented.

- Today we will consider the Particle-Mesh (PM) technique for solving the Poisson equation and integrating particle trajectories.

Particle-Mesh (PM) technique: a brief history

- PM was invented in the 1950's at LANL for simulations of compressible fluid flows
- However, the first wide-spread application was for collisionless plasma simulations (for which PM was reinvented in the 1960's and popularized by Hockney and collaborators)
- In the late 70's PM was first applied for 3D cosmological simulations. The method and its descendants (such as P^3M, AP^3M, and ART) have been used in most cosmological simulations ever since [3].

[3] In the 1990's Tree codes have also enjoyed increasing popularity

Particle-Mesh (PM) technique: modern relevance

- The popularity of PM in cosmology is due to
  - ⋆ its relative algorithmic simplicity, speed (the running time scales as $\propto O(N_p)+O(N_c \ln N_c)$, where N_p is the number of particles and N_c is the number of grid cells; and low storage requirements;
  - ⋆ natural incorporation of periodic boundary conditions;

- Today the PM technique is still very much relevant because
  - ⋆ Simplicity allows for efficient parallelization $\rightarrow$ and thus simulation on largest distributed architectures;
  - ⋆ low storage requirements allow simulation of a very large numbers of particles with relatively low resolution, suitable for some problems (e.g., halo mass function). Papers based on pure PM simulations still appear.
  - ⋆ PM technique is used to solve for the large-scale potential and forces in many modern high-resolution methods (e.g., AP^{3}M, ART, TreePM[4]).

[4] now implemented inthe Gadget-2 code

Model equations

PM code solves the cosmological Poisson equation (Peebles 1980, LSS):

$$\nabla^2\Phi = 4\pi G\rho_{\rm tot} - \Lambda,$$

and equations of motion of particles

$$\frac{d\mathbf{r}}{dt} = \mathbf{u}; \quad \frac{d\mathbf{u}}{dt} = -\nabla\Phi,$$

note that all variables are defined in proper coordinates and all spatial derivatives are also taken with respect to these coordinates.

However, it is convenient to re-write the equations in *comoving* variables and make them dimensionless by choosing suitable units (I will denote variables in code units with a tilde $\tilde{}$). In the following, I will adopt the variables and units used in Anatoly Klypin's PM code. This is just an example. Feel free to choose your own - just make sure you are consistent!

$$\tilde{\mathbf{x}} \equiv a^{-1}\frac{\mathbf{r}}{r_0}, \quad \tilde{\mathbf{p}} \equiv a\ \frac{\mathbf{v}}{v_0}, \quad \tilde{\phi} \equiv \frac{\phi}{\phi_0}, \quad \tilde{\rho} = a^3\frac{\rho}{\rho_0},$$

where $\mathbf{x}$ is the comoving coordinates, $\mathbf{v} = \mathbf{u} - H\mathbf{r} = a\dot{\mathbf{x}}$ is the *peculiar velocity*[5] and ϕ is the *peculiar potential* defined as (Peebles 1980, p. 42)

$$\phi \quad = \quad \Phi + 1/2\ a\ddot{a}\,(r/a)^2 = \Phi + \frac{H_0^2}{2}\left(\Omega_{\Lambda,0} - \frac{1}{2}a^{-3}\Omega_{m,0}\right)r^2,$$

where

$$\Omega_{m,0} = \frac{8\pi G\rho_0}{3H_0^2}; \qquad \Omega_{\Lambda,0} = \frac{\Lambda}{3H_0^2}.$$

[5] $p \propto av$ is called momentum, this choice of variable allows us to get rid of the annoying $(\dot{a}/a)$ terms in equations.

The quantities with subscript zero are the units in which corresponding physical variables are measured. The unit of length, r_0, is an arbitrary scale. I will chose r_0 to be the size of the PM grid cell:

$$r_0 = \frac{L_{\rm box}}{N_g}; \quad N_g^3 = \text{total number of grid cells}$$

the rest of the units are defined as

$$\begin{aligned} t_0 &\equiv H_0^{-1}, \\ v_0 &\equiv \frac{r_0}{t_0}, \\ \rho_0 &\equiv \frac{3H_0^2}{8\pi G}\Omega_{m,0}, \\ \phi_0 &\equiv \frac{r_0^2}{t_0^2} = v_0^2. \end{aligned}$$

It is also convenient to choose the expansion factor as time variable (using expressions for the Hubble constant: $\dot{a} = aH(a)$). For this choice of variables, the Poisson equation and equations of motion can be re-written as

$$\nabla^2\phi = 4\pi G\Omega_{m,0}\rho_{crit,0}a^{-1}\delta, \quad \delta = \frac{\rho - \bar{\rho}}{\bar{\rho}},$$

$$\frac{d\mathbf{p}}{da} = -\frac{\nabla\phi}{\dot{a}}, \quad \frac{d\mathbf{x}}{da} = \frac{\mathbf{p}}{\dot{a}a^2}.$$

where δ is the overdensity in comoving coordinates and $\dot{a}$ is

$$\dot{a} = H_0 a^{-1/2}\sqrt{\Omega_{m,0} + \Omega_{k,0}a + \Omega_{\Lambda,0}a^3}; \quad \Omega_{m,0} + \Omega_{\Lambda,0} + \Omega_{k,0} = 1$$

In dimensionless variables the equations are:

$$\tilde{\nabla}^2 \tilde{\phi} = \frac{3}{2}\frac{\Omega_0}{a}\tilde{\delta},$$

$$\frac{d\tilde{\mathbf{p}}}{da} = -f(a)\tilde{\nabla}\tilde{\phi}, \quad \frac{d\tilde{\mathbf{x}}}{da} = f(a)\frac{\tilde{\mathbf{p}}}{a^2}.$$

where $\tilde{\delta} = \tilde{\rho} - 1$ and

$$f(a) \equiv H_0/\dot{a} = \left[a^{-1}\left(\Omega_{m,0} + \Omega_{k,0}a + \Omega_{\Lambda,0}a^3\right)\right]^{-1/2}.$$

These equations are used in the three main steps of a PM code:

- Solve the Poisson equation using the density field estimated with current particle positions.
- Advance momenta using the new potential.
- Update particle positions using new momenta.

Data structures

For a PM code with the second-order accurate time integration we need the minimum of *six real numbers for positions and momenta for each particle* (assuming particles have the same mass) and *one real number for the potential of each grid cell*.

The array for the potential can be shared between density and potential: first use it for density, then replace it with potential when the Poisson equation is solved. However, for simplicity you can start with two grid arrays (for density and potential). Also, you will probably need auxiliary arrays for FFT, depending on which FFT solver you choose to use.

The convenient data structures are 1D or 3D arrays for particles (e.g., six 1D arrays for `x(i), y(i), z(i), vx(i), vy(i), vz(i)`) and 3D arrays for the grid variables (e.g., `rho(i,j,k), phi(i,j,k)`).

Run your tests for (32^3, 64^3) or (64^3, 128^3) particles and cells in which case memory requirements should not be an issue.

Density Assignment

The Particle Mesh algorithms assume that particles have certain size, mass, shape, and internal density. This determines the interpolation scheme used to assign densities to grid cells. Let's define the 1D particle shape, $S(x)$, to be mass density at the distance x from the particle for cell size Δx (Hockney & Eastwood 1981). The common choices are

- Nearest Grid Point (NGP): particles are point-like and all of particle's mass is assigned to the single grid cell that contains it:

$$S(x) = \frac{1}{\Delta x}\delta\left(\frac{x}{\Delta x}\right)$$

- Cloud In Cell (CIC): particles are cubes (in 3D) of uniform density and of one grid cell size.

$$S(x) = \frac{1}{\Delta x}\begin{cases} 1, & |x| < \frac{1}{2}\Delta x \\ 0, & \text{otherwise} \end{cases}$$

- Triangular Shaped Cloud (TSC):

$$S(x) = \frac{1}{\Delta x} \begin{cases} 1 - |x|/\Delta x, & |x| < \Delta x \\ 0, & \text{otherwise} \end{cases}$$

The fraction of particle's mass assigned to a cell ijk is the shape function averaged over this cell:

$$W(x_p - x_{ijk}) = \int_{x_{ijk}-\Delta x/2}^{x_{ijk}+\Delta x/2} dx' S(x_p - x');$$

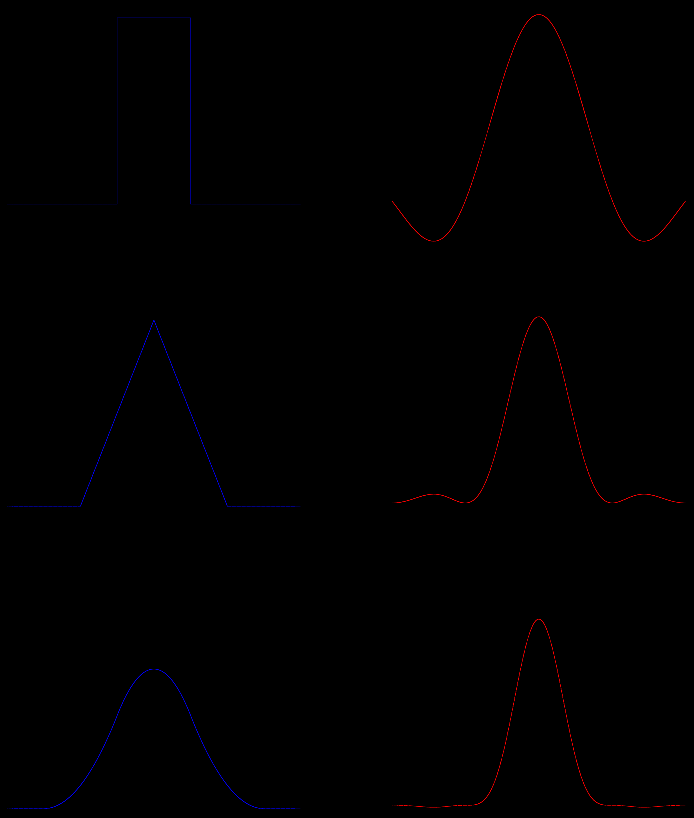
$$W(\mathbf{r}_p - \mathbf{r}_{ijk}) = W(x_p - x_{ijk}) W(y_p - y_{ijk}) W(z_p - z_{ijk});$$

The density in a cell ijk is then

$$\rho_{ijk} = \sum_{p=1}^{N_p} m_p W(\mathbf{r}_p - \mathbf{r}_{ijk})$$

In practice, of course, we loop over particles and assign the density to neighboring cells as opposed to summing over all particles for each cell as the straightforward reading of the above equation would suggest.

PM lectures by M.Gross



Implementing CIC interpolation

[6] Make sure you enforce periodic boundary conditions: $i = \mathrm{mod}(i, N_{g,1})$, etc. for j and k.

Solving the Poisson equation

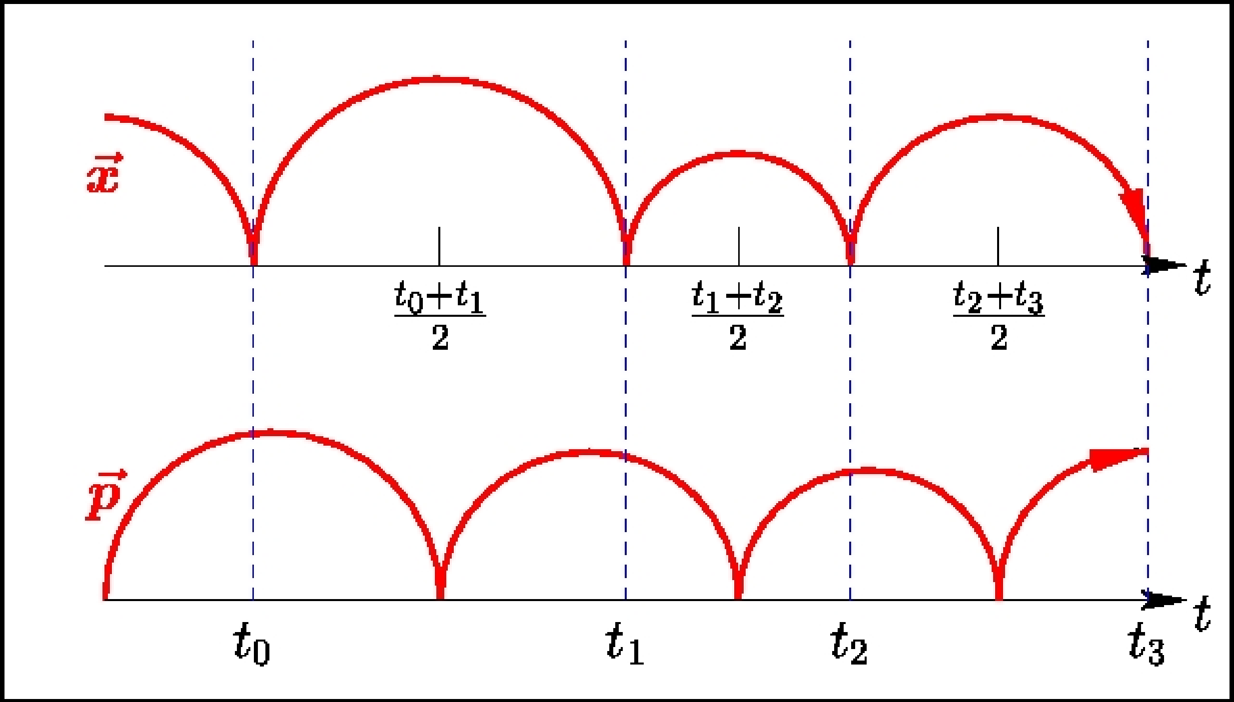
7

[7] It is customary in PM codes to discretize the Laplacian operator using the 7-point template.

[8]

[8] Be careful about normalization of the FFT. The transforms $\tilde{\delta}(\mathbf{r}) \longrightarrow \tilde{\delta}(\mathbf{k})$ and $\tilde{\delta}(\mathbf{k}) \longrightarrow \tilde{\delta}(\mathbf{r})$ should recover the original field $\tilde{\delta}(\mathbf{r})$.

Updating particle positions and velocities



This is important!

Zeldovich approximation

-

  Zeldovich 1970

-

Carrol et al. (1993)

-
-
-

[9] Bond et al. (1996)

[9] Another explanation of the success of ZA is its use of displacement field, $S(q)$, as the basis for evolution model. Density depends on the derivatives of $S(q)$ so that small (linear) displacements can correspond to large density contrasts.

A test problem: 1D collapse of a plane wave

Efstathiou et al. (1985)

Beyond the Zeldovich test:
setting up cosmological initial conditions

[10]

[11]

---

[10] An alternative is to set up initial conditions using a public code

Hu

& Sugiyama 1996 Eisenstein & Hu 1999

Project summary

- “Numerical recipes”
-
-
  - ⋆
  - ⋆

PM in cosmology: some historical references

- Efstathiou G. & Eastwood J. 1981, MNRAS **194**, 503
- Klypin A.A. & Shandarin S.F. 1983, MNRAS **204**, 891
- Centrella J. & Melott A.L. 1983, Nature **305**, 196-198
- Miller R.H., 1983, ApJ **270**, 390-409
- Efstathiou G., Davis M., White S.D.M., & Frenk C.S. 1985, ApJS **57**, 241-260

Papers with useful info on PM

- Hockney, R. W., and Eastwood, J. W. 1981, *"Computer Simulation Using Particles"*, McGraw-Hill, New York

- Klypin A.A. & Shandarin S.F. 1983, MNRAS **204**, 891

- Efstathiou G., Davis M., White S.D.M., & Frenk C.S. 1985, ApJS **57**, 241-260
  [Review and comparison of cosmological N-body methods]

- Sellwood J.A. 1987, ARA&A **25**, 151
  [Review of particle simulation methods]

- Description of Hugh Couchman's AP^{3}M code

- Bertschinger E. 1998, ARA&A **36**, 599
  [The most recent review of numerical techniques used in cosmological simulations and algorithms for setting up the initial conditions.]

- Michael Gross's, PhD Thesis
  [useful descriptions of PM and an algorithm for setting up ICs in Ch. 2 and Appendix A]

- Klypin, A. & Holtzman, J. 1997, `astro-ph/9712217`
  "Particle-Mesh code for cosmological simulations"

Web links

- Amara's Recap of Particle Simulation Methods: collection of info and links on various N-body algorithms

- Anatoly Klypin's public PM package for cosmological simulations.

- fortran PGPLOT library C