Numerical Cosmology
& Galaxy Formation

Lecture 4: Gravity algorithms

Benjamin Moster
Outline of the lecture course

- Lecture 1: Motivation & Historical Overview
- Lecture 2: Review of Cosmology
- Lecture 3: Generating initial conditions
- **Lecture 4: Gravity algorithms**
- Lecture 5: Time integration & parallelization
- Lecture 6: Hydro schemes - Grid codes
- Lecture 7: Hydro schemes - Particle codes
- Lecture 8: Radiative cooling, photo heating
- Lecture 9: Subresolution physics
- Lecture 10: Halo and subhalo finders
- Lecture 11: Semi-analytic models
- Lecture 12: Example simulations: cosmological box & mergers
- Lecture 13: Presentations of test simulations
Computational Cosmology

- Cosmological model + initial conditions + simulation code = galaxies
Computational Cosmology

- Cosmological model + initial conditions + simulation code = galaxies
Computational Cosmology

- Cosmological model + initial conditions + simulation code = galaxies

On cosmological scales:
Gravity is the dominant process

1. Generation the initial conditions
2. Running the simulation
3. Analyzing the data
The N-body approach

- $N$ bodies are used to sample the evolution of the Universe

- What is a simulation particle?

- Most of the matter in the Universe is Dark Matter, e.g. WIMPS ($\sim 100$ GeV)

- Box with $B=50$ Mpc has total mass of $\sim 10^{16} M_{\odot}$ so: $10^{71}$ WIMP particles

- With $N=1024^3$ particles the simulation particle mass is $10^7 M_{\odot}$

One particle represents $\sim 10^{60}$ WIMPs…
Dark Matter

- Dark Matter particles are collisionless
- The evolution is driven by the mean potential rather than two-body interactions of dark matter particles

\[ \sigma \approx 10^{-45} \text{cm}^2 \]
\[ m_{DM} \approx 10^2 \text{GeV} \approx 10^{-22} \text{g} \]
\[ \rho_c \approx 10^{-30} \text{g/cm}^3 \]
\[ \rho_c = \frac{N m_{DM}}{V} = n m_{DM} \quad n \approx 10^{-8} / \text{cm}^3 \]
\[ \lambda = \frac{1}{n \sigma} \approx \frac{1}{10^{-8} 10^{-45}} \text{cm} = 10^{53} \text{cm} \approx 10^{33} \text{Mpc} \]
Particle collisions

- Particle collisions are unwanted when modeling a collisionless system (dark matter) part of the system when modeling gas.
- But they can enter the system due to numerical limitations. Does representing $\sim 10^{60}$ DM particles by 1 simulation particle have unwanted side effects?
Particle collisions

- Particle collisions are unwanted when modeling a collisionless system (dark matter) part of the system when modeling gas.

- But they can enter the system due to numerical limitations. Does representing $\sim 10^{60}$ DM particles by 1 simulation particle have unwanted side effects?
Relaxation time of an N-body system

- Timescale on which 2-body processes are important in N-body system

\[ \Delta v_\perp = \frac{1}{m} \int F_\perp dt = \int \frac{Gm}{x^2 + b^2} \frac{b}{\sqrt{x^2 + b^2}} \frac{dx}{v} = \frac{2Gm}{bv} \]
Relaxation time of an N-body system

- During one crossing: $dn$ particles encountered with impact parameter between $b$ and $b+db$:

\[
dn \approx \frac{2\pi b \, db}{\pi R^2} N
\]

- Individual encounters add incoherently:

\[
(\Delta v_\perp)^2 = \int \left( \frac{2Gm}{bv} \right)^2 \, dn = \frac{8G^2 m^2}{R^2 v^2} N \int \frac{db}{b} = \frac{8G^2 m^2}{R^2 v^2} N \ln \left( \frac{b_{\text{max}}}{b_{\text{min}}} \right)
\]

$b_{\text{max}} = R$ given by system size

$b_{\text{min}}$ controls maximum deflection and should be $\sim v$

\[
\frac{2Gm}{b_{\text{min}} v} = v \quad \rightarrow \quad b_{\text{min}} = \frac{2Gm}{v^2}
\]
Relaxation time of an N-body system

- Typical velocity: \( v^2 = \frac{GNm}{R} \)

- So we get:

\[
b_{\text{min}} = \frac{2Gm}{v^2} = \frac{2R}{N}
\]

- Per crossing time the mean square change is:

\[
(\Delta v_\perp)^2 = \frac{8G^2m^2}{R^2v^2} N \ln \left( \frac{b_{\text{max}}}{b_{\text{min}}} \right) = \frac{8v^2}{N} \ln \left( \frac{N}{2} \right)
\]

- And the 2-body relaxation time is:

\[
t_{\text{relax}} = t_{\text{cross}} n_{\text{relax}} = t_{\text{cross}} \frac{v^2}{(\Delta v_\perp)^2} = t_{\text{cross}} \frac{N}{8 \ln(N/2)}
\]
## Relaxation time of an N-body system

- **Examples:**

<table>
<thead>
<tr>
<th>System</th>
<th>$N$</th>
<th>$t_{\text{cross}}$</th>
<th>$t_{\text{relax}}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Star cluster</td>
<td>$10^5$</td>
<td>0.5 Myr</td>
<td>0.5 Gyr</td>
</tr>
<tr>
<td>Stars in galaxy</td>
<td>$10^{11}$</td>
<td>0.01/$H_0$</td>
<td>$5\times10^6/H_0$</td>
</tr>
<tr>
<td>dark matter in galaxy</td>
<td>$10^{11}$</td>
<td>0.1/$H_0$</td>
<td>$10^{63}/H_0$</td>
</tr>
<tr>
<td>galaxy in low-res simulation</td>
<td>1000</td>
<td>0.1/$H_0$</td>
<td>2/$H_0$</td>
</tr>
</tbody>
</table>

*Collisionless* and *somewhat collisional* indicate the nature of the interactions within each system.
The Bullet cluster

collisionless dark matter

collisionless dark matter

collisional gas
Dynamical friction

- Low resolution: collision (acceleration)
- High resolution: dynamical friction (deceleration)
Dynamical friction

- Low resolution: collision (acceleration)
- High resolution: dynamical friction (deceleration)

How can we eliminate this effect?
Gravitational softening

- For collisionless systems we need to ensure: $t_{\text{sim}} \ll t_{\text{relax}}$
  - prevent large angle scattering / singularity
  - prevent formation of bound pairs

- Want to integrate equations of motion with low-order scheme and reasonably large timesteps

- Need to soften force law on small scales:

$$F_j = \frac{G m_j}{a^2} \sum \frac{m_i (\vec{x}_i - \vec{x}_j)}{((\vec{x}_i - \vec{x}_j)^2 + \epsilon^2)^{3/2}}$$
Gravitational softening

- Softening determines overall force resolution of the simulation.
- For given particle number, the mean integrated square error of force accuracy has a minimum at an optimal softening.
- Increase $N$ and decrease $\epsilon$ according to $N \epsilon^3 = \text{cst.}$
Gravitational softening

- **Too small softening:**
  - system may become collisional
  - time integration is more expensive
  - artificial heating

- **Too large softening:**
  - loss of spatial resolution in the simulations

- **Typical value in cosmological simulations:**
  - 2% - 4% of the mean-interparticle distance $B/N^{1/3}$
Size of cosmological simulation over time

- Moore’s law: computers double speed every 18 month
- Particle numbers in simulations double every 16-17 month
- Only possible with algorithms that scale close to $O(N)$
Requirements

• We want
  large $N$: high resolution (otherwise small objects are not resolved)
  large $B$: large volume (otherwise no rare objects)

• Need efficient self-gravity algorithms with scaling close to $\sim N$

• Need to be able to run efficiently on 1000s of CPU cores

• Should be memory and communication efficient

• Should automatically adapt the size of the timestep to the relevant dynamical time
Overview of self-gravity algorithms

- Direct summation
- Particle-mesh codes
- Particle-particle / particle-mesh codes (P³M)
- Tree codes
- Tree particle-mesh codes (e.g. used in GADGET)
- Multigrid relaxation (e.g. used in RAMSES)
- Fast multipole codes...
Direct summation

• Most accurate method

\[ \rho(\vec{r}) = \sum_{i}^{N} m_i \delta_D(\vec{r} - \vec{r}_i) \]

• Easy to code!

\[ \vec{F}_i(\vec{r}_i) = \sum_{i \neq j}^{N} \frac{Gm_i m_j}{(r_i - r_j)^3} (\vec{r}_i - \vec{r}_j) \quad \forall i \in N \]

• Extremely time consuming!

\[ N \times N = N^2 \]
The particle-mesh (PM) method

• Numerically integrate the Poisson equation on a grid

\[ \Delta \Phi = 4\pi G \bar{\rho}_0 \delta a^{-1} \quad \rightarrow \quad -k^2 \hat{\Phi}_k = 4\pi G \bar{\rho}_0 \delta_k a^{-1} \]

• Calculate mass density on grid

• Get Fourier transform of density contrast

• Solve Poissons equation on grid

• Transform potential back to real space

• Compute gradient by finite differencing

• Interpolate forces back to particles
How to get the density on a grid?

- We need to assign a density to each grid point

ID example:
How to get the density on a grid?

- We need to assign a density to each grid point
  1D example:
  - Nearest grid point (NGP)
  - total mass is assigned to nearest grid point
  - density is then $\rho = \frac{m}{H}$
How to get the density on a grid?

- We need to assign a density to each grid point

1D example:

- Cloud in Cell (CIC)

- $m(1 - d/H)$ is assigned to each grid point

- Each particle can contribute to neighboring cell
How to get the density on a grid?

- We need to assign a density to each grid point
  1D example:

- Triangular Shaped Cloud (TSC)

- particle mass is assigned to several grid point
The particle-mesh (PM) method

- Numerically integrate the Poisson equation on a grid
  \[ \Delta \Phi = 4\pi G \bar{\rho}_0 \delta a^{-1} \quad \Rightarrow \quad -k^2 \hat{\Phi}_k = 4\pi G \bar{\rho}_0 \delta_k a^{-1} \]

- Calculate mass density on grid

- Get Fourier transform of density contrast

- Solve Poisson's equation on grid

- Transform potential back to real space

- Compute gradient by finite differencing

- Interpolate forces back to particles
The particle-mesh (PM) method

- Numerically integrate the Poisson equation on a grid

\[ \Delta \Phi = 4\pi G \bar{\rho}_0 \delta a^{-1} \quad \rightarrow \quad -k^2 \hat{\Phi}_k = 4\pi G \bar{\rho}_0 \delta_k a^{-1} \]

- Calculate mass density on grid

- Get Fourier transform of density contrast

- Solve Poisson's equation on grid

- Transform potential back to real space

- Compute gradient by finite differencing

- Interpolate forces back to particles

**Pros**: fast and simple

**Cons**: spatial resolution limited to grid resolution

Transform potential back to real space

Compute gradient by finite differencing

Interpolate forces back to particles
The tree method

- Group distant particles together and use their multiple expansion

- Generating the tree:
The tree method

- Group distant particles together and use their multiple expansion
- Generating the tree:
The tree method

• Group distant particles together and use their multiple expansion

• Generating the tree:
The tree method

- Group distant particles together and use their multiple expansion
- Generating the tree:
The tree method

- Opening criteria:
  - Barnes-Hut: \( \frac{L}{D} < \theta \)
  - Min-distance: \( \frac{L}{D} < \theta \)
  - B_{\text{max}}: \( \frac{b_{\text{max}}}{D} < \theta \)
The tree method

- Opening criteria:
  - Barnes-Hut: $\frac{L}{D} < \theta$
  - Bmax: $\frac{b_{\text{max}}}{D} < \theta$

**Pros:** higher resolution in high density regions

**Cons:** less efficient than PM on large scales
The tree-PM method

- Tree-PM algorithm (used e.g. in Gadget):
  - combine tree and PM methods to get the advantages of both
  - split forces in long range & short range part in Fourier space

\[
\begin{align*}
\delta \phi_k^{\text{long}} &= \delta \phi_k \exp(-k^2 r_s^2) \\
\delta \phi_k^{\text{short}} &= \delta \phi_k (1 - \exp(-k^2 r_s^2)) \\
\end{align*}
\]

- long range forces
- short range forces

\[
\phi^{\text{short}} = -G \sum_i \frac{m_i}{r_i} \text{erfc} \left( \frac{r_i}{2r_s} \right)
\]

- solve with particle-mesh method
- solve with tree code
The tree-PM method

• Tree-PM algorithm (used e.g. in Gadget):
  • combine tree and PM methods to get the advantages of both
  • split forces in long range & short range part in Fourier space

**Pros:** high resolution, fast, $N \log(N)$ scaling

\[ \delta \frac{\partial^2 \psi}{\partial \xi^2} = \frac{G}{s^2} \sum_i \frac{m_i}{r_i^2} \left( \text{erfc} \left( \frac{r_i}{2r_s} \right) \right) \]

solve with particle-mesh method

in real space (assuming large $N_{\text{grid}}$)

\[ \phi^{\text{short}} = -G \sum_i \frac{m_i}{r_i} \text{erfc} \left( \frac{r_i}{2r_s} \right) \]

solve with tree code
Up next

- Lecture 1: Motivation & Historical Overview
- Lecture 2: Review of Cosmology
- Lecture 3: Generating initial conditions
- Lecture 4: Gravity algorithms
- Lecture 5: Time integration & parallelization
- Lecture 6: Hydro schemes - Grid codes
- Lecture 7: Hydro schemes - Particle codes
- Lecture 8: Radiative cooling, photo heating
- Lecture 9: Subresolution physics
- Lecture 10: Halo and subhalo finders
- Lecture 11: Semi-analytic models
- Lecture 12: Example simulations: cosmological box & mergers
- Lecture 13: Presentations of test simulations