N-body Techniques

Part 2

Massimo Ricotti
ricotti@astro.umd.edu

University of Maryland
Time-integration Schemes

- Clearly, Newton’s laws are IVP. Could use any method (Euler, RK4, etc.).
- But, issue is to balance accuracy vs. efficiency.
- Typically need many particles to capture dynamics correctly (e.g., in stellar system or galaxy). This consideration may be as important as accuracy of any one individual particle (exception: solar system—$N \sim 9$, $\tau \sim 10^9–10^{10}$ orbits).
- Could use Euler scheme. But we have seen it is just as easy to design 2\textsuperscript{nd}-order scheme by centering derivatives $\Rightarrow$ could use leapfrog (very stable).
Practical timestep control

- The stability criterion from the discussion of stiff systems also applies to the leapfrog integrator for the $N$-body problem.

- Can show need $\delta t < 2/\Omega$, where $\Omega^2 = |\nabla F|$ is a characteristic “interaction frequency” for a particle (in practice, need $\delta t \ll 2/\Omega$ to avoid problems).

- But $\Omega^2$ is different for every particle; can be very large for particle undergoing close interaction.

- If have to take $\Omega_{\text{max}}$, can be very restrictive.
Two solutions:

1. Use different timesteps for each particle (individual timesteps).

   E.g., $\delta t_i = \eta F_i / \dot{F}_i$ — effective, but complex implementation, and may break symplecticity of leapfrog (for example).

   More complex expressions for $\delta t_i$ can be formulated, e.g., involving higher-order derivatives of $F$. These are largely heuristics with convenient properties. It is difficult to prove analytically that one formulation is superior to another.

   Sometimes $\delta t_i$ is discretized, e.g., in factors of 2 (multisteppeing).
2 Eliminate short-timescale phenomena by modifying gravity on small scales.

   E.g., Set $\delta t = \tau_D / 30$ and/or use softening (see below).

always important to check whether simulation is giving physically meaningful results.

   Handy technique: reduce timestep by factor of 2 to see if global behavior strongly affected. If so, may have to use smaller steps.

   Beware of chaos: if state of system strongly dependent on initial conditions, change of timestep may give seemingly vastly different results. Need to monitor constants of motion to be sure.
Force evaluation

- Solving the IVP requires evaluation of the RHS of the ODEs, i.e., must compute interparticle forces.
- Will discuss PP, PM, $P^3M$, and tree methods.
- But first must consider another practical issue, related to timestep control...
Hard interactions

Recall $F_{ij} = -Gm_j (r_i - r_j) / |r_i - r_j|^3$.

Problem: if $|r_i - r_j|$ is small, $|F_{ij}|$ diverges, leading to timestep trouble as $|v_i| \to \infty$.

Physically, very close encounters occur on very short timescales, e.g., can form close binaries with very short periods.

To alleviate problem, could use “softened” forces:

$$F_{ij} = -\frac{Gm_j (r_i - r_j)}{(|r_i - r_j|^2 + \varepsilon^2)^{3/2}},$$

where $\varepsilon = \text{“softening parameter.”}$

Maximum force now $\sim Gm^2 / \varepsilon^2$. 
Physically, this eliminates possibility of forming binaries with $r < \varepsilon$.

- OK when particles represent collection of stars on similar orbits.
- Not OK if studying small clusters, where each particle represents an individual star. In this case binaries *can* form and significantly affect evolution of entire cluster.

Modern methods also sometimes use “regularization.”

- Binaries (or hierarchies) replaced by pseudo-particles until interaction with other particles becomes important.
Direct Summation (PP Method)

- Most straightforward way of evaluating $F_{ij}$.
- But number of operations $= \frac{1}{2} N(N - 1) \sim N^2$ for $N \gg 1$ (the $\frac{1}{2}$ comes from the fact that $F_{ij} = -F_{ji}$).
  
  $\therefore$ 10× more particles $\implies$ 100× more work.

- Severely limits number of particles that can be used (typically $\lesssim 10^{3-4}$).

- Motivates finding more efficient techniques.
Evaluating Forces on a Mesh (PM Method)

Idea: compute forces (per unit mass) from gravitational potential:

\[ \mathbf{F} = -\nabla \Phi, \]  

(1)

where \( \Phi = \) potential, a scalar function of \( r \).

Potential given by solution of Poisson’s equation:

\[ \nabla^2 \Phi = 4\pi G \rho. \]  

(2)

Can FD (2) to compute \( \Phi \) on a mesh, then FD (1) to compute \( \mathbf{F} \) from \( \Phi \).
In 1-D, Poisson equation is:

\[
\frac{\partial^2 \Phi}{\partial x^2} = 4\pi G \rho.
\]

This is an elliptic PDE.

Consider discretizing \( \Phi \) on a mesh, center \( \Phi_i \) and \( \rho_i \), at mesh centers \( i = 1, \ldots, N \).
Then, 2nd-order FDE for (2) is:

\[
\frac{(\Phi_{i+1} - \Phi_i) - (\Phi_i - \Phi_{i-1})}{\Delta x} \Delta x = \frac{\Phi_{i-1} - 2\Phi_i + \Phi_{i+1}}{(\Delta x)^2} = 4\pi G \rho_i \ (i = 1, \ldots, N).
\]

For \(i = 1\), need \(\Phi_0\)

For \(i = N\), need \(\Phi_{N+1}\) (boundary conditions).

For the force, just FD (1) using the same mesh:

\[
\mathcal{F}_{i+1/2} = - \left[ \frac{\Phi_{i+1} - \Phi_i}{\Delta x} \right]
\]

(need to interpolate to cell centers).