Class 20. N-body Techniques, Part 3

The PM Method, Continued

There are several distinct steps in PM process:

1. Assign particles to mesh to compute ρ_i.
2. Get boundary conditions for Φ (Φ_0 and Φ_{N+1}).
3. Solve discretized version of Poisson's equation.
4. Compute $\boldsymbol{\mathcal{F}}$ from discretized version of force equation.

Step 1: Assigning particles to mesh

Discuss two schemes here:

1. Nearest Grid Point (NGP) scheme:
   - Assign entire mass of particle to grid zone that contains it.
   - E.g., discretize space into N zones in x-dimension:

     Set $\rho_i = n_i m/\Delta x$, where n_i = number of particles in cell i (equal mass).
   - Leads to a very coarse distribution of ρ_i:

2. Charge-In-Cell (CIC) or Particle-In-Cell (PIC):
   - Assign a "shape" or "cloud" to particle.
   - Assume a distribution of ρ inside this shape.
   - Then distribute mass to zones according to overlap.
   - E.g., assume particle has top-hat ρ distribution, width w, height $\rho_0 = m/w$:

Then (in 1-D), $\int_{-\infty}^{\infty} \rho(x)\, dx = m$. Distribute mass of particle according to overlap:

Leads to smoother ρ_i.

- Can adopt more complex shapes for density. E.g.,

  Triangle
  Gaussian
  etc.

  Higher-order "shapes" introduce ringing into system.

Step 2: Boundary conditions

- Given ρ_i, $i = 1, ..., N$, need a boundary value for Φ, i.e., need Φ_0 and Φ_{N+1}.
- Often can use periodic BC, i.e., $\Phi_0 = \Phi_N$, $\Phi_{N+1} = \Phi_1$. Appropriate for, e.g., cosmology simulations.
- Otherwise, standard to use multipole expansion (e.g., Jackson 1975) to compute potential on boundary due to mass in each cell.
  - Often, first (monopole) term is good enough:

$$\Phi_B(\mathbf{r}) = -\sum_{i=1}^{N} \frac{Gm_i}{|\mathbf{r} - \mathbf{r}_i|}.$$

  - See Binney & Tremaine Eq. 2-122 for full series (involves spherical harmonics).

Step 3: Solve Poisson's equation

- Can see that discretized equation

$$\frac{\Phi_{i-1} - 2\Phi_i + \Phi_{i+1}}{(\Delta x)^2} = 4\pi G\rho_i$$

leads to tri-diagonal (tri-di) matrix:

$$\begin{bmatrix} -2 & 1 & & & & \\ 1 & -2 & 1 & & & \\ & 1 & -2 & \ddots & & \\ & & \ddots & \ddots & \ddots & \\ & & & 1 & -2 & 1 \\ & & & & 1 & -2 \end{bmatrix} \begin{bmatrix} \Phi_1 \\ \Phi_2 \\ \Phi_3 \\ \vdots \\ \vdots \\ \Phi_N \end{bmatrix} = \begin{bmatrix} 4\pi G\rho_1(\Delta x)^2 - \Phi_0 \\ 4\pi G\rho_2(\Delta x)^2 \\ 4\pi G\rho_3(\Delta x)^2 \\ \vdots \\ \vdots \\ 4\pi G\rho_N(\Delta x)^2 - \Phi_{N+1} \end{bmatrix}.$$

- There is an extremely efficient algorithm for solving tri-di systems.
  - Write discretized system as:

$$a_i \Phi_{i-1} + b_i \Phi_i + c_i \Phi_{i+1} = d_i.$$

  - Then forward elimination gives (Hockney & Eastwood, p. 185):[1]

$$w_1 = \frac{c_1}{b_1} \qquad w_i = \frac{c_i}{b_i - a_i w_{i-1}},$$

    ($i = 2, 3, ..., N-1$), and,

$$g_1 = \frac{d_1}{b_1} \qquad g_i = \frac{d_i - a_i g_{i-1}}{b_i - a_i w_{i-1}}.$$

  - Backsubstitution:

$$\begin{aligned} \Phi_N &= g_N \\ \Phi_i &= g_i - w_i \Phi_{i+1}, \end{aligned}$$

    with $i = N-1, N-2, ..., 1$.
  - If a, b, c constant, can precompute w_i and $1/(b_i - a_i w_{i-1})$.
  - If $a = 1$, $b = -2$, $c = 1$, only need $4N$ operations.
  - For periodic BC, even simpler method possible (Hockney & Eastmood, p. 35).

Step 4: Force interpolation

- Once potential is given, must compute force (per unit mass) from $\boldsymbol{\mathcal{F}} = -\nabla\Phi$.
- In 1-D, $\mathcal{F} = -\partial\Phi/\partial x \Rightarrow$ FDE $\mathcal{F}_{i+1/2} = -(\Phi_{i+1} - \Phi_i)/\Delta x$.
  - Forces centered at cell boundaries:

- Must interpolate forces to particle positions.
- Linear interpolation simplest. For each particle, position $x_{i-1/2} < x < x_{i+1/2}$, compute:

$$\mathcal{F}(x) = \mathcal{F}_{i-1/2} + \left(\frac{x - x_{i-1/2}}{\Delta x}\right)\left(\mathcal{F}_{i+1/2} - \mathcal{F}_{i-1/2}\right).$$

- Higher-order interpolation used in conjunction with higher-order charge-assignment schemes.

[1] Also see `tridag()` (*NRiC* §2.4).

We now have ingredients necessary for a 1-D PM code:

1. Particle assignment;
2. Boundary conditions;
3. Solve Poisson's equation;
4. Force interpolation.

Result is $\boldsymbol{\mathcal{F}}$ for every particle.

Generalizing to 3-D

- Generalizing to 3-D is straightforward:
    1. Particle assignment: use NGP; or for PIC, particle is sphere.
    2. BCs: periodic, or use 3-D multipole expansion.
    3. Solve Poisson's equation in 3-D (see below).
    4. Interpolate $\boldsymbol{\mathcal{F}}$ in 3-D (easy).
- Poisson's equation in 3-D:

$$\frac{\partial^2\Phi}{\partial x^2} + \frac{\partial^2\Phi}{\partial y^2} + \frac{\partial^2\Phi}{\partial z^2} = 4\pi G\rho.$$

- Discretize Φ in 3-D:

$$\begin{aligned} \Phi(x,y,z) &\rightarrow \Phi_{i,j,k}, \\ \rho(x,y,z) &\rightarrow \rho_{i,j,k}. \end{aligned}$$

- FDE becomes:

$$\frac{\Phi_{i-1,j,k} - 2\Phi_{i,j,k} + \Phi_{i+1,j,k}}{(\Delta x)^2} + \frac{\Phi_{i,j-1,k} - 2\Phi_{i,j,k} + \Phi_{i,j+1,k}}{(\Delta y)^2} + \frac{\Phi_{i,j,k-1} - 2\Phi_{i,j,k} + \Phi_{i,j,k+1}}{(\Delta z)^2} = 4\pi G\rho_{i,j,k}.$$

- Can be written in matrix form:

$$a_i\Phi_{i,j,k-1} + b_i\Phi_{i,j-1,k} + c_i\Phi_{i-1,j,k} + d_i\Phi_{i,j,k} + e_i\Phi_{i+1,j,k} + f_i\Phi_{i,j+1,k} + g_i\Phi_{i,j,k+1} = h_i,$$

  where $i = 1, ..., N_x$, $j = 1, ..., N_y$, $k = 1, ..., N_z$ and

$$\begin{aligned} c_i = e_i &= 1/(\Delta x)^2 & d_i &= -2\left[(1/\Delta x)^2 + (1/\Delta y)^2 + (1/\Delta z)^2\right] \\ b_i = f_i &= 1/(\Delta y)^2 & h_i &= 4\pi G\rho_{i,j,k} \text{ (modulo BCs)} \\ a_i = g_i &= 1/(\Delta z)^2 \end{aligned}$$

- Leads to very large sparse banded matrix:

$$
\begin{bmatrix}
d & e & & & & f & & & & g & & \\
c & \cdot & \cdot & & & & \cdot & & & & \cdot & \\
& \cdot & \cdot & \cdot & & & & \cdot & & & & g \\
& & \cdot & \cdot & \cdot & & & & f & & & \\
& & & \cdot & \cdot & \cdot & & & & \cdot & & \\
b & & & & \cdot & d & e & & & & \cdot & \\
& \cdot & & & & c & d & \cdot & & & & f \\
& & \cdot & & & & \cdot & \cdot & \cdot & & & \\
& & & b & & & & \cdot & \cdot & \cdot & & \\
a & & & & \cdot & & & & \cdot & \cdot & \cdot & \\
& \cdot & & & & \cdot & & & & \cdot & \cdot & e \\
& & a & & & & b & & & & c & d
\end{bmatrix}
$$

  - Dimension is $(N_x N_y N_z) \times (N_x N_y N_z)$!
  - $\Longrightarrow$ even very small problem $(20^3) \rightarrow$ large matrix 8000×8000.
  - "Reasonable" sized problem $(100^3) \rightarrow 10^6 \times 10^6$ matrix!
  - Clearly need efficient ways to solve matrix:
    1. Relaxation schemes — guess solution, then relax (Cf. *NRiC* §19.5–19.6).

       E.g., "Successive Over-Relaxation" (SOR), "Alternating-Direction Implicit" (ADI), multi-grid (use exact solution on coarse grid as initial guess for iterative solution on fine grid), etc.
    2. Sparse banded solvers, e.g., conjugate gradient method (*NRiC*, §2.7).
    3. Fourier methods — solution of FDE in Fourier space is very simple, then can inverse Fourier transform solution back to real space (*NRiC* §19.4).
       - Very powerful, but requires periodic BCs.

Summary: PM Method

- What is advantage of PM code?
  - Force solving scales as $\mathcal{O}(N_g)$, where N_g = number of mesh grid points.
  - Leapfrog scales as $\mathcal{O}(N_p)$, where N_p = number of particles.
  - Work associated with leapfrog $\ll$ solving Poisson's equation.

    $\therefore$ can afford very large N_p, e.g., N_p 10^{6-8} with $N_g \sim 10^{4-6}$.
  - Not good for correlated systems (in which 2-body encounters important) but great for uncorrelated systems (where it takes the place of softening).