N-body Techniques
Part 3

Massimo Ricotti
ricotti@astro.umd.edu

University of Maryland
The PM Method, Continued

There are several distinct steps in PM process:

1. Assign particles to mesh to compute $\rho_i$.
2. Get boundary conditions for $\Phi$ ($\Phi_0$ and $\Phi_{N+1}$).
3. Solve discretized version of Poisson’s equation.
4. Compute $\mathcal{F}$ 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:
     
     ![Diagram of discretized space]

     
     Set $\rho_i = n_i m / \Delta x$, where $n_i = \text{number of particles in cell } i$
     (equal mass).

   - Leads to a very coarse distribution of $\rho_i$:
2 Cloud-In-Cell (CIC) or Particle-In-Cell (PIC):

- Assign a “shape” or “cloud” to particle.
- Assume a distribution of $\rho$ inside this shape.
- Then distribute mass to zones according to overlap.
- E.g., assume particle has top-hat $\rho$ 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 $\rho_i$.

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

- Triangle
- Gaussian
- etc.

Higher-order “shapes” introduce ringing into system.
Step 2: Boundary conditions

- Given $\rho_i, \ i = 1, \ldots, N$, need a boundary value for $\Phi$, i.e., need $\Phi_0$ and $\Phi_{N+1}$.

- 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:
    \[
    \Phi_B(r) = -\frac{GM}{|r - r_{CM}|}.
    \]

- See Binney & Tremaine (second ed), Ch. 24, Eq. 2.95 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:

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

\[ w_1 = \frac{c_1}{b_1}, \quad w_i = \frac{c_i}{b_i - a_i w_{i-1}}, \]

\[ (i = 2, 3, ..., N - 1), \text{ and,} \]

\[ g_1 = \frac{d_1}{b_1}, \quad g_i = \frac{d_i - a_i g_{i-1}}{b_i - a_i w_{i-1}}. \]

\[ ^a \text{Also see tridag}() \text{ (}NRiC\text{ §2.4).} \]
Backsubstitution:

\[ \Phi_N = g_N \]
\[ \Phi_i = g_i - w_i \Phi_{i+1}, \]

with \( i = N - 1, N - 2, \ldots, 1 \).

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 & Eastwood, p. 35).
Step 4: Force interpolation

- Once potential is given, must compute force (per unit mass) from \( \mathbf{F} = -\nabla \Phi \).

- In 1-D, \( \mathbf{F} = -\frac{\partial \Phi}{\partial x} \Rightarrow \text{FDE} \ F_{i+1/2} = -\frac{(\Phi_{i+1} - \Phi_i)}{\Delta x} \).

- Forces centered at cell boundaries:

  \[
  \begin{array}{c|c|c}
  \Phi_{i-1} & F_{i-1/2} & \Phi_i \\
  \hline
  \Phi_i & F_{i+1/2} & \Phi_{i+1}
  \end{array}
  \]

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

\[
F(x) = F_{i-1/2} + \left( \frac{x - x_{i-1/2}}{\Delta x} \right) (F_{i+1/2} - F_{i-1/2}).
\]

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

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

1. Particle assignment;
2. Boundary conditions;
3. Solve Poisson’s equation;

Result is \( F \) for every particle.
Generalizing to 3-D

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 $\mathcal{F}$ 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 $\Phi$ in 3-D:

\[ \Phi(x, y, z) \rightarrow \Phi_{i,j,k}, \]
\[ \rho(x, y, z) \rightarrow \rho_{i,j,k}. \]

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 \( i = 1, \ldots, N_x \), \( j = 1, \ldots, N_y \), \( k = 1, \ldots, N_z \) and

\[
c_i = e_i = 1/(\Delta x)^2, \quad d_i = -2 \left[ (1/\Delta x)^2 + (1/\Delta y)^2 + (1/\Delta z)^2 \right],
\]
\[
b_i = f_i = 1/(\Delta y)^2, \quad h_i = 4\pi G \rho_{i,j,k} \,(\text{modulo BCs}),
\]
\[
a_i = g_i = 1/(\Delta z)^2.
\]
Leads to very large sparse banded matrix:

\[
\begin{bmatrix}
  d & e & f & g \\
  c & & & \\
  & & & g \\
  & & f & \\
  b & d & e & \\
  & c & d & f \\
  & & & \\
  a & & & e \\
  & a & b & c & d \\
\end{bmatrix}
\]

Dimension is \((N_x N_y N_z) \times (N_x N_y N_z)!\)
even very small problem ($20^3$) $\rightarrow$ large matrix $8000 \times 8000$.

“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 $O(N_g)$, where $N_g = \text{number of mesh grid points}$.
  - Leapfrog scales as $O(N_p)$, where $N_p = \text{number of particles}$.
  - Work associated with leapfrog $\ll$ solving Poisson’s equation.
    - ∴ can afford very large $N_p$, e.g., $N_p \sim 10^{6-8}$ 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).