N-Body project

Computational Astrophysics

ADVICE: using a low level programming language (such as for example C/C++) can significantly improve the performances of the code you are going to build up, especially in the second task, where time integration is involved. Higher level languages as Python can be used but be aware it will typically take way longer to complete a simulation, modulo optimization tricks (NumPy, ...).

Contents

1 Task1

1.1 Step 1

- Preliminarly, verify the form of the density function $\rho(r)$ by inferring it from the particle distribution and compare it with the analytical density function described in the original paper by Hernquist (from 1990 on Astrophysical Journal available on the web). Use Poissonian error bars[1] when comparing the numerical density profile with the analytical expected values.
- Note that the initial conditions are given in a system of units in which G=1. Assume reasonable units of length and mass for your calculations (units of velocity and time follow automatically from the assumption G=1) and discuss your choice.

[1]Poissonian error: the exercise requires to compare the Ernquist density profile with the density profile of the data. The simplest way to do that is to divide the space in spherical bins (shells) and count how many particles you have in all the bins, as you would do when building a histogram. Then you can compare these values to the expected values, i.e. the average amount of particles you would expect in each bin given the Ernquist density profile. In doing that you should consider that the number of particles you count in a given shell is a random variable that follows a Poisson distribution with λ = expected number of particles (average value). When you compare the 2 values you would need some error estimate to evaluate how similar they are, and the standard error you have is the standard deviation of the Poisson distribution, i.e. $\sqrt{\lambda}$. Please be careful about how you choose your bins, in order to have reasonable results.

1.2 Step 2

- Compute the direct N-Body forces between particles (note that the array potential[i] is not needed for this purpose). Start by assuming a softening of the order of the mean interparticle separation[2] in the system, then repeat the force calculation by experimenting with different values of the softening and discuss your results.
- To check the direct force calculation result and its dependence on the softening choice, compare it with the analytical force expected based on the application of Newton's second theorem for spherical potentials[3] and plot the result (use the book "Galactic Dynamics" by Binney and Tremaine as main reference for the theoretical notions, in particular sec. 2.2 (most recent version of the book) or 2.1 (1987 version).
- Compute the relaxation timescale of the numerical model given the number of particles and the physical crossing timescale (use the half-mass radius R_{hm} and the circular velocity computed at the half-mass radius, $v_c = \sqrt{GM(R_{hm})/R_{hm}}$). Keeping in mind how the relaxation time formula is derived, do you expect varying the value of the gravitational softening to change the relaxation timescale? In particular, do you expect it to increase or decrease if the softening is increased above the interparticle separation? Can you explain why?

2 Task 2 (choice 1): tree-code

- Compute the force on particles (a subset of them would be fine) using multipole expansion (tree-code). For multipole calculation decide on a criterion (distance based) to group particles, eventually experimenting with different orders of the expansion. Finally, compare the tree-code result with the direct summation result, in terms of accuracy as well as computational cost. Try different softenings for the direct summation and different opening angles for the tree-code.
- Using an appropriate time integrator among those that you have studied and tested earlier, integrate the equation of motions using direct summation for a few systems' crossing timescales (see definition at Task 1 Step 2). Repeat the integration with different force softenings and attempt to measure the magnitude of numerical relaxation in runs with different softenings, comparing them. Try different timesteps, justifying the choice.
- OPTIONAL: Attempt to write a full gravity tree solver scheme with a time integrator of your choice and evolve the system for at least a few timesteps.

3 Task 2 (choice 2): Particle-Mesh code

Download and use the appropriate initial condition set provided. They represent a 2D stellar disk (total mass $M = 10$) with a central massive particle (black hole for instance) with mass $M_0 = 1$.

[2] Use the number of particles contained in the half-mass radius R_{hm} to estimate this value.

[3] If we assume spherical symmetry we have $\vec{F}(r) = -\frac{GM(r)}{r^3}\vec{r}$ and $M(r)$ is the mass included within the radius r.

Always assume G=1 and choose again reasonable units in order to get the correct time and length units. There are two different sets to experiment with, one with $\sim$ 1000 particles, the other with $\sim$ 10000 if your code is able to handle them. Each of the two sets come in two versions, one is very regular with particles on a grid, the other adds some noise and random distribution. The softening for time evolution is provided.

- Compute the force on particles using a Particle-Mesh method. In practice, you should build an uniform 3D grid[4], compute the density on the grid starting from the particle distribution[5], solve the Poisson equation on the grid (Fourier transform for example), assign the accelerations from the grid to the particle again. Try different grid spacings and justify them. Finally, compare the PM results with the direct summation result, in terms of accuracy as well as computational cost. Try different softenings for the direct summation and different grid spacing for the PM.

- Using an appropriate time integrator among those that you have studied and tested earlier, integrate the equation of motions using direct summation for a few orbits. Verify that the initial configuration is stable and in equilibrium (the disk should not shrink or expand) at least for a few orbits with the softening that has been provided. Try also different softenings and check what happens. Try different timesteps, justifying the choice.

- OPTIONAL: Attempt to write a full gravity PM solver scheme with a time integrator of your choice and evolve the system for at least a few timesteps. They disk may not be in equilibrium and expand/shrink right away. Why is that? (look at the acceleration comparison with different grid spacing and softening). You can adjust the velocities in the initial conditions in order the disk to be in equilibrium.

[4]You can also try to built a 2D grid but be careful of what you are really calculating (Σ vs ρ).

[5]In case you want to check, the density distribution should behave as $\Sigma = \Sigma_0 (r/0.1)^{-2}$ where $\Sigma_0 \sim 72.5$. Σ is basically the vertical sum of the volume density $\Sigma(x, y) = \sum_z \rho(x, y, z)h$ where h is the vertical grid spacing.