ASTRON 449, Winter 2019 – Problem Set 3

Due Thu Feb. 14, in class.

REGULAR PROBLEMS:

1. Filling the gaps in Jeans equation derivations. 1) In class, we derived an expression for the second Jeans equation in Cartesian coordinates,

$$\frac{\partial(\nu\bar{v}_j)}{\partial t} + \frac{\partial(\nu\overline{v_i v_j})}{\partial x_i} + \nu\frac{\partial\Phi}{\partial x_j} = 0. \tag{1}$$

Showing all intermediate steps, show that this equation can be rewritten in a form analog to Euler's equation for fluids:

$$\nu\frac{\partial\bar{v}_j}{\partial t} + \nu\bar{v}_i\frac{\partial\bar{v}_j}{\partial x_i} = -\nu\frac{\partial\Phi}{\partial x_j} - \frac{\partial(\nu\sigma_{ij}^2)}{\partial x_i}. \tag{2}$$

b) Starting from the Jeans equation for a steady-state spherical system in hydrostatic balance,

$$\frac{1}{\nu}\frac{\partial(\nu\sigma_{rr}^2)}{\partial r} + 2\frac{(\sigma_{rr}^2 - \sigma_{t1}^2)}{r} = -\frac{GM(r)}{r^2}, \tag{3}$$

show that

$$M(r) = -\frac{r\sigma_{rr}^2}{G}\left[\frac{d\ln\nu}{d\ln r} + \frac{d\ln\sigma_{rr}^2}{d\ln r} + 2\beta(r)\right]. \tag{4}$$

This formulation is useful because it shows how the mass enclosed is directly related to the slope of the number density and radial velocity dispersion profiles with radius.

2. BH radius of influence. For nearby elliptical galaxies, typical values are $M_{\rm BH} = 10^8$ $\mathrm{M}_\odot$ and $\sigma_{||} = 200$ km s^{-1}. Evaluate the radius of influence of such a galaxy. Calculate the angular size of the radius of influence in arcsec assuming that the galaxy is $d = 20$ Mpc away, which is also typical of galaxies in which central black holes are studied using stellar dynamics. The angular resolution at which spectroscopy can be done accurately with current instruments is about 0.1 arcsec. What does this imply for measurements of central black hole masses using stellar dynamics? Comment on the importance of the Hubble Space Telescope and adaptive optics, which provide the best angular resolutions available today.

3. Distribution function for an isothermal sphere. In this problem, you will fill the gaps in the example given in class. Consider a system with distribution function

$$f(\mathcal{E}) = \frac{\rho_1}{(2\pi\sigma^2)^{3/2}}\exp\left(\frac{\Psi - \frac{1}{2}v^2}{\sigma^2}\right), \tag{5}$$

where ρ_1 and σ are constants, and Ψ is a time-independent relative potential.

a) Show that the density distribution

$$\rho = \int d^3\mathbf{v}\, f(\mathcal{E}) = \rho_1\exp\left(\Psi/\sigma^2\right). \tag{6}$$

b) Using Poisson's equation and looking for solutions of the form $\rho = Cr^{-b}$, show that a self-consistent solution for the density distribution is the singular isothermal sphere

$$\rho = \frac{\sigma^2}{2\pi G r^2}. \tag{7}$$

Note that this solution is not unique and that it is also possible to obtain solutions that are finite at $r = 0$ by enforcing suitable boundary conditions.

c) Show that the velocity dispersion $\langle v^2 \rangle = 3\sigma^2$ everywhere. Hint: To evaluate the integrals in this problem, you may find it useful to use the result

$$\frac{1}{\sqrt{2\pi}\sigma} \int_{-\infty}^{\infty} dx \exp\left(-x^2/2\sigma^2\right) = 1 \tag{8}$$

(normalization of a Gaussian) but should otherwise show all the steps in your calculation.

4. Distribution function for an arbitrary spherical potential. a) For a general distribution function that is a function of $\mathcal{E}$ only and assuming that the mass distribution is spherically-symmetric, show that

$$\rho(r) = 4\pi \int_0^{\Psi} d\mathcal{E} f(\mathcal{E}) \sqrt{2(\Psi - \mathcal{E})}. \tag{9}$$

b) Assume that both $\rho(r)$ and $\Psi(r)$ are monotonic functions of r (otherwise, the system would have a cavity in it and would generally be unstable). Then we may consider ρ to be a function of Ψ. Show that

$$\frac{1}{\pi\sqrt{8}} \frac{d\rho}{d\Psi} = \int_0^{\Psi} d\mathcal{E} \frac{f(\mathcal{E})}{\sqrt{\Psi - \mathcal{E}}}. \tag{10}$$

Hint: Remember Leibniz's integral differentiation rule.

c) This is an Abel integral equation for $f(\mathcal{E})$. Use Abel's transform to invert it and show that

$$f(\mathcal{E}) = \frac{1}{\pi^2\sqrt{8}} \frac{d}{d\mathcal{E}} \int_0^{\mathcal{E}} \frac{d\Psi}{\sqrt{\mathcal{E} - \Psi}} \frac{d\rho}{d\Psi}. \tag{11}$$

Differentiate the integral to show that

$$f(\mathcal{E}) = \frac{1}{\pi^2\sqrt{8}} \left[\int_0^{\mathcal{E}} \frac{d^2\rho}{d\Psi^2} \frac{d\Psi}{\sqrt{\mathcal{E} - \Psi}} + \frac{1}{\sqrt{\mathcal{E}}} \left(\frac{d\rho}{d\Psi} \right)_{\Psi=0} \right]. \tag{12}$$

Since $f(\mathcal{E})$ must be ≥ 0, equation (11) shows that a spherical density distribution $\rho(r)$ can arise from a distribution function that is a function of $\mathcal{E}$ only (an "ergodic" distribution function) if, and only if, the integral on the right hand side of that equation is a monodically increasing function of $\mathcal{E}$. In practice, this requires $\rho(r)$ to drop sufficiently rapidly with radius.

d) What is the anisotropy parameter β for a system with distribution function $f = f(\mathcal{E})$? Explain your answer.

COMPUTATIONAL PROBLEMS:

Reminder concerning units: Treat Newton's constant G as a variable whose value can be modified in the code. By default, we work in dimensionless units and set $G = 1$.

In this problem set, we will start evolving realistic N-body simulations for $N \gg 1$.

C1. Direct summation code. Write a Python program that evolves the phase-space coordinates (x, y, z, v_x, v_y, v_z) of a list of N particles with masses $m_1, ..., m_N$ using the direct summation method. Design your program so that it can be run using a command of the form

```
python dsum.py input_data integr t dt eps dt_out output_base
```

where `input_data` is an ASCII file containing a list of particle masses and initial phase-space coordinates in the format

```
1 m1 x1 y1 z1 vx1 vy1 vz1
2 m2 x2 y2 z2 vx2 vy2 vz2
...
N mN xN yN zN vxN vyN vzN
```

(the first column is a particle ID that allows us to keep track of particles at different times).

As before, `integr` is a string specifying the integration algorithm (you need only implement the `leapfrog` option for this problem), `t` is the duration of the integration, and `dt` is the time-step.

In your code, evaluate gravitational forces using a Plummer softening kernel, with softening length (the constant b in class notes) specified by the command line parameter `eps`.

We want two kinds of output files for this program.

The first is a file that summarizes "global" diagnostic quantities for the system of N particles at each integration time-step. Design your program so that this global diagnostic file is written in ASCII format

```
t0 Epot0 Ekin0 Etot0 absL0
t1 Epot1 Ekin1 Etot1 absL1
...
tM EpotM Ekin EtotM absLM
```

to the file named `output_base_global.dat`, where `output_base` is the string specified in the command line, and "`_global.dat`" is a suffix indicating the global diagnostics. The diagnostic entries in this file are similar to the ones that we tracked in our code to integrate the orbit of a single

test particle, except that we now keep track of the total potential energy of the N-body system (`Epot`), its total kinetic energy (`Ekin`), as well as the total mechanical energy (`Etot`). `absL` is the magnitude of the total angular momentum vector relative to the origin.

We also need the full particle data at different times (positions and velocities, in the same format as `input_data`), but we don't want to output this full information at every integration time-step `dt` (this would produce too much data). Instead, we output the full particle data only at longer time intervals `dt_out`, also specified in the command line. The full particle data should be written to files named "`output_base_parts_x.dat`," where here `x` = 0, 1, 2, ... indicates the "snapshot number" (corresponding to simulation time `x` × `dt_out`).

Note: The most computationally expensive parts of the code involve computing the accelerations and total gravitational potential energy of the system, since each scales as $O(N^2)$. To speed up your code, you can use NumPy to perform the calculations on arrays of particles. This can be more than 10× faster than pure Python loops and may be necessary for you to evolve simulations with larger N in reasonable time.

C2. Gravitational collapse of a sphere. You will now use your code to simulate the gravitational collapse of an initial particle distribution. Download the following input files from the course web site, which you will use as initial conditions (ICs) for simulations: `homo_sph_Nx_R3_v1.dat`, where `x=200, 500, 1000`.

Each of these corresponds to a statistically homogeneous sphere of radius $R = 3$ (particle positions are random within the sphere). In each case, all particles have the same mass m such that $1/\sqrt{G\rho} = 1$ within the sphere (i.e., the free fall time is unity) and `x` specifies the number of particles N. The particles are given small initial random velocities.

a) Use your direct summation code to evolve these three sets of ICs to a final time $t = 5$ using the leapfrog integrator and a Plummer softening length `eps=0.2`. Output the full particle data at intervals `dt_out=0.1` so that you have enough information to do all the analysis you will need.

You will need to choose an integration time-step dt such that your results are well converged, which you can assess by checking conservation of total mechanical energy. The requirements for this can be quite stringent as the particle distribution collapses to the center, so you may need rather small $dt \sim 0.0001$.

b) For each N, produce a multi-panel plot as in Figure 1, showing the ICs, particle snapshots (xy maps) at $t = 0.5,\ 1,\ 2,\ 3,\ 4,\ 5$, and energy and angular momentum statistics. For the energy panel, plot $E_{\rm tot}/E_{\rm tot,0}$ as a diagnostic of total energy conservation and $E_{\rm kin}/|E_{\rm pot}|$ as a diagnostic of the partition between kinetic and potential energy.

c) We will see in class that the system can be considered "virialized" when it relaxes to a state with $E_{\rm kin}/|E_{\rm pot}| \approx 0.5$. Compare the time needed for virialization ($t_{\rm vir}$) of the initial particle distribu-

tions indicated by your simulations with estimates for the following time scales: the initial free fall time ($t_{\rm ff,0}$), the crossing time shortly after virialization ($t_{\rm cross}$), and the two-body relaxation time ($t_{\rm relax}$). By considering how $t_{\rm vir}$ and $t_{\rm relax}$ vary with N, determine whether two-body relaxation drives virialization in this simulation (justify your answer).

d) Read section 4.10.2 on phase mixing and violent relaxation in BT2 to gain insight into the process that drives virialization in your simulations.

To upload: Your three plots (for $N = 200,\ 500,\ 1000$); example global output and final ($t = 5$) particle data files for the $N = 200$ simulation; a copy of your Python code; and answers to the questions above.

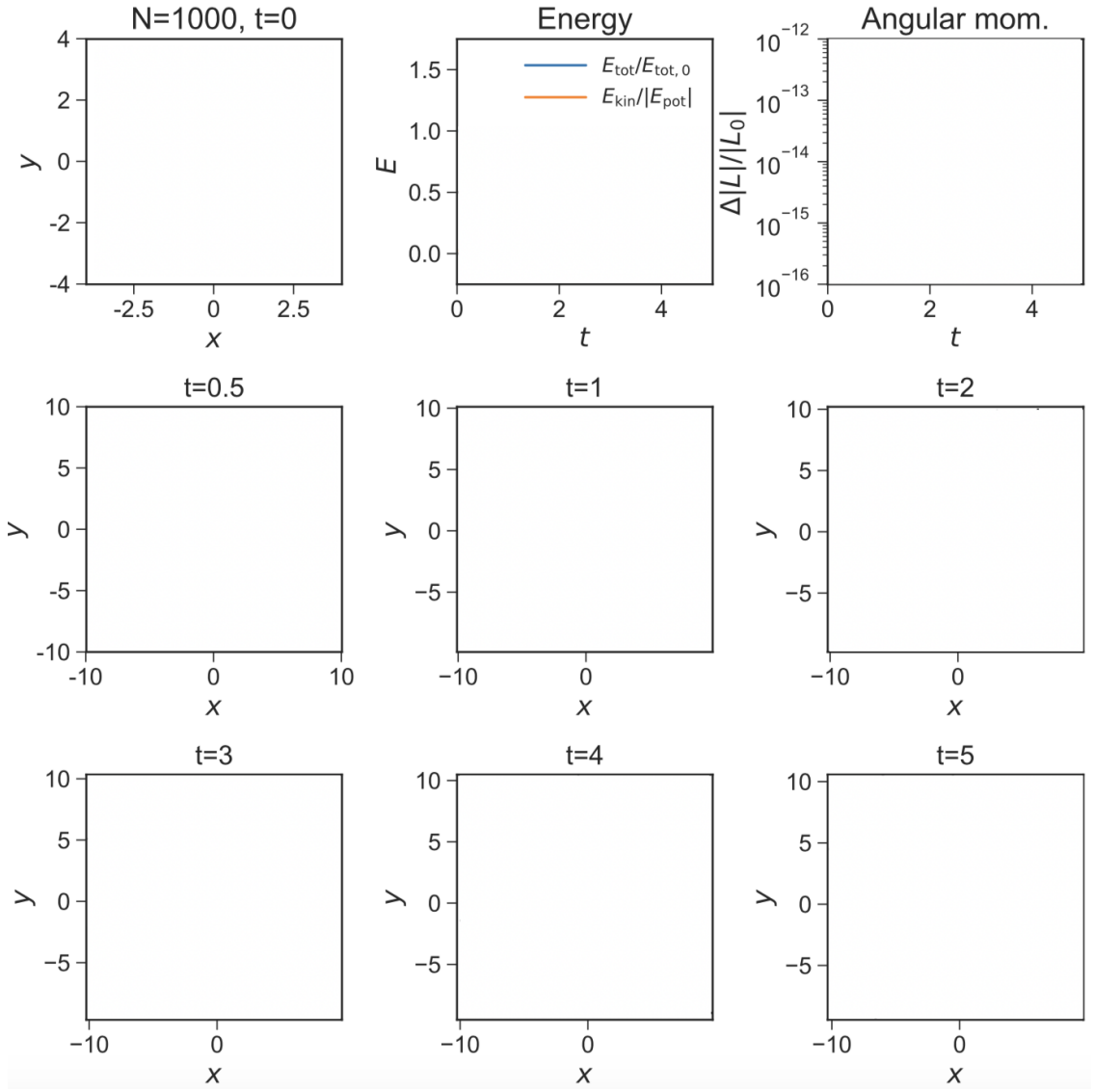

Fig. 1.— Example multi-panel plot to summarize the results of your $N-$body simulations.