Final Honour School of Mathematical and Theoretical Physics Part C and MSc Mathematical and Theoretical Physics

Galactic and Planetary Dynamics

Markers: John Magorrian, Bence Kocsis

Please typeset your solutions using a font size of at least 10 pt for the main body of your report.

In this miniproject you will interpret the results of using an N-body code to simulate the dynamics of a local patch of a razor-thin disc in the shearing-sheet approximation. The N-body code is `REBOUND` (Rein & Liu 2012); instructions for downloading are at https://rebound.readthedocs.io. The shearing sheet is covered briefly in lectures. Binney (2020) goes in to more detail. Toomre & Kalnajs (1991) present a thorough analysis of N-body simulations of the shearing sheet. The first two sections of Hamilton (2021) develop similar ideas in a more general context. You should read through these papers at least once before starting the miniproject.

(a) We will drive `REBOUND` using the code overleaf (available for download here). What assumption does this make about the large-scale gravitational potential that the shearing patch is embedded in? What are the integrals of motion $\mathbf{I}$ for particles in the shearing-sheet approximation and what distribution function $f(\mathbf{I})$ is assumed in the driver to set up the initial condition?

(b) Outline the method the code uses to calculate the gravitational potential within the sheet.

(c) Explain the algorithm used to evolve particles' phase-space locations.

(d) Run the code for $N = 10^3$, 10^4 and 10^5 and for $Q = 0.2$, 0.6 and 1.4. Describe qualitatively how the evolution of the simulated sheet varies with N and Q and give brief physical explanations for the differences.

(e) In the shearing-sheet approximation how does the location of the guiding centre $(\bar{x}, \bar{y})$ of a particle change with time? How are $(\bar{x}, \bar{y})$ obtained from the particle's (x, y, v_x, v_y) coordinates? For the $Q = 1.4$, $N = 10^5$ sheet what is the ensemble-averaged distribution of *all* particles' displacements from the guiding centre radius of a typical star? Explain the method that you use to measure this.

(f) Stating clearly any results that you take from the literature and identifying any underlying assumptions, show how the star–guiding centre correlation function measured in (e) can be predicted using perturbation theory. *Note: you are not expected to attempt the numerical calculations needed to plot the result!* Would you expect the results of this prediction to agree with the measurement from the simulation?

(g) Outline the changes that would have to be made to this driver code and to `REBOUND` in order to apply the shearing-sheet approximation to a Mestel disc [$f(E, L) = L^q \mathrm{e}^{-E/\sigma^2}$] with a flat rotation curve.

References

1. Rein H., Liu S.-F., 2012, Astronomy and Astrophysics, Volume 537, A128
2. Binney J., 2020, Monthly Notices of the Royal Astronomical Society, Volume 496, p. 767
3. Toomre A., Kalnjas A.J., 1991, Dynamics of Disc Galaxies, Proceedings of the conference held 25-30 May 1991 at Varberg Castle, Sweden. Edited by B. Sundelius. Göteborgs: Göteborgs University and Chalmers University of Technology, p. 341
4. Hamilton C., 2021, Monthly Notices of the Royal Astronomical Society, Volume 501, p.3371

```
import rebound
import numpy as np
pi = np.pi
from numpy import sqrt, sin, cos

# Set up the rebound simulation.  Start a webserver at
# http://localhost:12345 so that we can see what's going on!
sim = rebound.Simulation()
sim.start_server(port=12345)

N = 100000
Q = 1.4
Omega = 1
A = 3*Omega/4
sigma = 1.0
Sigma = Omega*sigma/(3.36*Q)

boxsize = 100
mass = Sigma*boxsize**2/N
sim.ri_sei.OMEGA = Omega
sim.G = 1
sim.dt = 1e-2
sim.softening = 1.0
sim.configure_box(boxsize)
sim.N_ghost_x = 2
sim.N_ghost_y = 2
sim.integrator = "sei"
sim.boundary   = "shear"
sim.gravity    = "tree"
sim.collision  = "none"

for n in range(N):
    x = np.random.uniform(low=-boxsize/2., high=boxsize/2.)
    y = np.random.uniform(low=-boxsize/2., high=boxsize/2.)
    H = np.random.exponential(sigma**2)
    thet = np.random.uniform(0,2*pi)
    sim.add(m=mass, x=x, y=y, z=0.0,
        vx = sqrt(2*H)*cos(thet),
        vy = -2*A*x + sqrt(2*H)*sin(thet)/2,
        vz = 0.0, r=0.2)

nt = 100
Tdump = 0.1*(2*pi/Omega)
for it in range(0,nt):
    T = it*Tdump
    sim.integrate(it*Tdump)
    sigma_gc = sqrt(np.mean([ p.vx**2+(p.vy+2*A*p.x)**2 for p in sim.particles]))
    print(T,sigma_gc)
```


End of Last Page