Final Project

This final assignment is a capstone hydrodynamic modeling project in which you will work in a small group to set up and carry out calculations using the *Gizmo* code.

Gravitational Collapse++

In this project you will use GIZMO to model gravitational collapse with different initial conditions and compare the results. Gravitational collapse problems, while useful for testing code results, are also critical for the formation of structure across astrophysics. In this project you will start with the simplest case and add additional complexity. To set up your initial conditions, you can build on *Make_IC.py* and/or the cloud initializer developed by Mike Grudic, *MakeCloud.py* (`https://github.com/mikegrudic/MakeCloud`) to be useful.

1. **Uniform collapse**.

a.) Set up a uniform sphere of gas with mass $M = 1M_{\odot}$, radius, $R = 7.8 \times 10^{15}$ cm and uniform temperature of T =10 K. Assume the sphere is surrounded by a low-density ambient medium that is 100 times lower than the sphere density. Adopt periodic boundary conditions, such that the box size is 4 times larger than the gas sphere. I suggest starting with a resolution of 10^4 cells in the sphere, which should run relatively quickly.

Set up the compile flags to meet the following conditions:

- MFM solver.
- The domain is periodic but gravity is not.
- The gas uses an adiabatic equation of state that is very very close to isothermal.
- Set the equation of state (EOS) function so the adiabatic sound speed (squared) is defined explicitly in the Config.sh file.

Check your setup units carefully!

b.) The collapse of the above sphere can be described analytically (approximately) as the collapse of a pressureless uniform cloud (see Truelove et al. 1998 §3.2.1). Run your setup for at least 0.5 freefall times and compare with the analytic result predicted for the maximum gas density. (You may need to adjust the zero-point of the analytic solution). How well does GIZMO model the collapse, i.e., what is the accuracy?

2. **Uniform Collapse with Rotation and Realistic Thermodynamics**.

Real astrophysical gas follows complex thermodynamics that regulate the temperature via a variety of atomic and molecular processes. Rerun the above collapse using the GIZMO

'cooling' function instead of an idealized EOS. Compare your result to the analytic prediction. How well does the collapse follow the solution now? Make a plot of the gas temperature at an early and late time. Based on the documentation/method papers what are the main heating/cooling processes acting?

Note: you will need to download and put an additional file in your run directory.

3. **Rotating Collapse**

a.) A slightly more realistic setup will have non-zero initial velocities and some amount of angular momentum. Initialize the sphere above with solid body rotation where $\Omega = 10^{-12}$ radians s^{-1}. Re-run the collapse problem and compare against the analytic solution. How well does the solution match now? How well does GIZMO conserve angular momentum over time?

b.) Increase the initial rotational rate. How does the evolution and rate of collapse change when the rotational energy is 1%, 10%, and 50% of the gravitational energy?

4. **Magnetized Collapse** (*Optional*)

Add an initially uniform magnetic field $\mathbf{B} = (0, 0, B_z)$ with $B_z = 1$ Gauss. How does the additional magnetic pressure affect the evolution and rate of collapse?

Project Report

Write up your results in a project report. Includes some background on the GIZMO code, describe the initial conditions of the runs, and presents the results.

Notes on Initial Conditions

There are a several different ways to set up the particle / cell placement, e.g., random sampling or a uniform grid. Another option is to use a lattice 'glass' structure; this is uniform configuration of particles that is created by distributing them by minimizing the system energy (we briefly discussed this when we discussed SPH initialization). A python numpy array file of lattice particle positions is available here on Stampede2: */scratch/00653/tg458122/glass_orig.npy*. See the *MakeCloud.py* code for how you might scale this to make your setup.