A (partially implemented) hydrodynamics code

Daniel Boyea

Physics 6810

April 2023

Professor. Bundschuh

1 Overivew

Here, I describe some of the structure of the code here, and since the program is heavy on the equations, I also note the critical equations. See the README.md file for a description of how to excecute the program.

A note about Julia (since I am not sure how much you have used/seen the language). Julia uses unicode (as a language feature even), so the source code does use unicode symbols for variables (like ρ, ϕ, etc.). Also, Julia is compiled at runtime so there is no need to worry about makefiles. Julia also has some very abbreviated synatx for arithmatic (like `2x^2`) and vectorization (just add a .).

2 Structure

The main body of the code is in the `src/` directory. This directory includes the files

- `GalaxySim.jl`. This just imports and exports other pieces of the project.
- `evolve.jl` contains the main loop of the simulation, including the leapfrog integration scheme and time-step criteria
- `gal_files.jl` writes the simulation outputs to files. (Unfortunantly, other io to files for testing are scattered through the project)
- `density.jl` contains routines for density estimation.
- `gravity.jl` calculates the gravity
- `physics.jl` all the rest of the physics (hydrodynaics, viscosity, etc.)
- `particles.jl` definition of the Particle struct

- `params.jl` struct to read in Params (stored in `init/` directory)
- `constants.jl` Physical constants in cgs (which the code uses internally)

3 Physics

Lagrangian derivative

$$\frac{d}{dt} = \frac{\partial}{\partial t} - \vec{v} \cdot \nabla \tag{1}$$

The hydrodynamic equations using lagrangian derivatives are

$$\mathbf{a} = \frac{-1}{\rho} \nabla \cdot \mathbf{P} \tag{2}$$

$$u = \frac{3}{2} \frac{R}{\mu} T \frac{du}{dt} = \tag{3}$$

Smoothed Particle Hydrodynamics (SPH) is a varient of the Lagrangian method for solving hydrodynamics equations. Instead of dividing space up into a grid, each particle is followed and the physical properties are calculated in the frame of each particle.

First, we need a way to estimate the density at any given point from the distribution of nearby points. We do this using a weighted sum over the nearby points of a particle. The density of particle j is

$$\rho_j = \sum_i m_i W(r_{i,j}, h_j), \tag{4}$$

where the sum index i is over all points and $W(r, h)$ is the kernel weight at a distance r with a smoothing length h.

3.1 Hydrodynamics

3.2 Viscosity

3.3 Gravity

3.4 Star Formation and Feedback

4 Implementation

I follow a variety of sources to use standard smoothed particle hydrodynamics (SPH) to implement the physics (Monaghan, 1992, 2005; Price & Monaghan, 2007; Price et al., 2018; Pasetto et al., 2010; Price, 2012; Springel et al., 2001). The idea (as you probably know) is to estimate the density with a kernel The kernel also has a smoothing length h, which should represent the mass inside the smoothing sphere, i.e.

$$h = \eta \left(\frac{m}{\rho} \right)^{1/3} \tag{5}$$

where η is density parameter. This system can be solved using Newton-Raphsons method. I follow Monaghan (2005) and use the function

$$f(h) = \rho - \rho_{\text{new}} \tag{6}$$

where ρ is calculated from the current value of h in Eq. 5, and ρ_{new} is calculated from the summation above.

So each new h is found with

$$h_{\text{new}} = h - \frac{f(h)}{f'(h)} \tag{7}$$

Other physics (like the change in density, position, etc.) are the standard SPH equations (Monaghan, 2005, 1992), except gravity is done following SPH using a smoothed kernel as described in Price & Monaghan (2007).

$$\left.\frac{du}{dt}\right|_P = \frac{P_j}{\Omega_j \rho_j^2} \sum_i m_i (\mathbf{v}_j - \mathbf{v}_i) \cdot \nabla_j W_i j \tag{8}$$

Integration is leapfrog, so

- $x \to x + v\, dt/2$
- $v \to v + a\, dt/2$
- calculate a from current half-step v and x
- $v \to v + a\, dt/2$
- $x \to x + v\, dt/2$

5 Bibliography

Monaghan, J. J. 1992, ARA&A, 30, 543, doi: 10.1146/annurev.aa.30.090192.002551

—. 2005, Reports on Progress in Physics, 68, 1703, doi: 10.1088/0034-4885/68/8/R01

Pasetto, S., Grebel, E. K., Berczik, P., Spurzem, R., & Dehnen, W. 2010, A&A, 514, A47, doi: 10.1051/0004-6361/200913240

Price, D. J. 2012, Journal of Computational Physics, 231, 759, doi: 10.1016/j.jcp.2010.12.011

Price, D. J., & Monaghan, J. J. 2007, MNRAS, 374, 1347, doi: 10.1111/j.1365-2966.2006.11241.x

Price, D. J., Wurster, J., Tricco, T. S., et al. 2018, PASA, 35, e031, doi: 10.1017/pasa.2018.25

Springel, V., Yoshida, N., & White, S. D. M. 2001, New A, 6, 79, doi: 10.1016/S1384-1076(01)00042-2