A 1D Lagrangian Hydrodynamics Code

Alun Rees

1 Introduction

Hydrodynamic simulation codes play an important role in the study of Plasma physics, particularly in the fields of Inertial Confinement Fusion and Astrophysics. They can be used to design experiments and to predict the behaviour of astrophysical bodies that are hard to observe. The fundamental principle of Hydrodynamic codes is to solve the Euler or Navier-Stokes equations, which describe the time evolution of fluid parameters as a set of first order differential equations. For the purpose of my code, we will focus on the Euler equations. These equations are a result of the three conservation laws, and are described below.

Conservation of Mass:

$$\frac{\partial \rho}{\partial t} + \nabla \cdot \rho \vec{u} = 0 \tag{1}$$

Conservation of Momentum:

$$\frac{\partial}{\partial t}(\rho \vec{u}) + \nabla \cdot \left(\vec{u} \otimes (\rho \vec{u})\right) + \nabla P = 0 \tag{2}$$

Conservation of energy:

$$\frac{\partial}{\partial t}(\rho e_T) + \nabla \cdot \left(\vec{u}\ (\rho e_T + P)\right) = 0 \tag{3}$$

where ρ is the density, $\vec{u}$ is the velocity vector, P is the thermodynamic pressure, e_i is the specific internal energy, and $e_T = e_i + 1/2u^2$ is the specific total energy. For an ideal gas, the specific internal energy, e_i and the thermodynamic pressure, P, and the heat capacity ratio, γ, are linked by the ideal gas equation:

$$e_i = \frac{P}{\rho(\gamma - 1)} \tag{4}$$

The ideal gas equation provides an Equation of State - a closure relation for the Euler equations, which does not bring in a new dependence for the parameters. In reality, the equation of state is much more complicated for simulating high energy density physics, but for the purposes of this code, it will suffice.

2 Theory

In order to solve the Euler equations, it is important to first choose a frame of reference for which to solve the equations in. These are the most commonly adopted methods:

- Eulerian methods
- Riemann Solver
- Lagrangian Method
- Lagrangian Remap
- Arbitrary Lagrangian Eulerian (ALE) Method

The most basic and fundamental methods are the Eulerian and Lagrangian - these provide the framework for all of the other methods and each have their advantages and disadvantages.

2.1 Eulerian Method

The Eulerian method discretises the fluid and has a grid of cells which are fixed in space and allow the fluid to flow through it. Using the above equations, it is possible to simulate a fluid so long the flux between neighbouring cells is taken into consideration.

The main advantage in using an Eulerian code is robustness. Unlike their Lagrangian counterparts, Eulerian methods do not struggle to complete computations when the flow becomes complicated. The physics within an Eulerian code is usually simpler to expand than Lagrangian methods, due to the fact that the grid is known, and orthogonal. This simplicity of the grid means that the computational cost of a single time-step should be less than the Lagrangian code. However, in practice, Eulerian codes can be dimensionally split, so this advantage can be reversed. Other disadvantages include numerical diffusion.

2.2 Lagrangian Method

The Lagrangian approach to hydrodynamics treats the fluid as a collection of cells, and evaluates the time evolution of each of these fluid elements. By treating the fluid in this manner, then the mass of each cell is conserved, but its position and volume changes as the system evolves.

By implementing this system, we are effectively defining each parameter of the fluid as a function of time and density, i.e pressure can be defined as: $p \mapsto p(x,t)$. It is important to construct a grid for the system and be aware of the indices of each parameter and where they exist relative to each other.

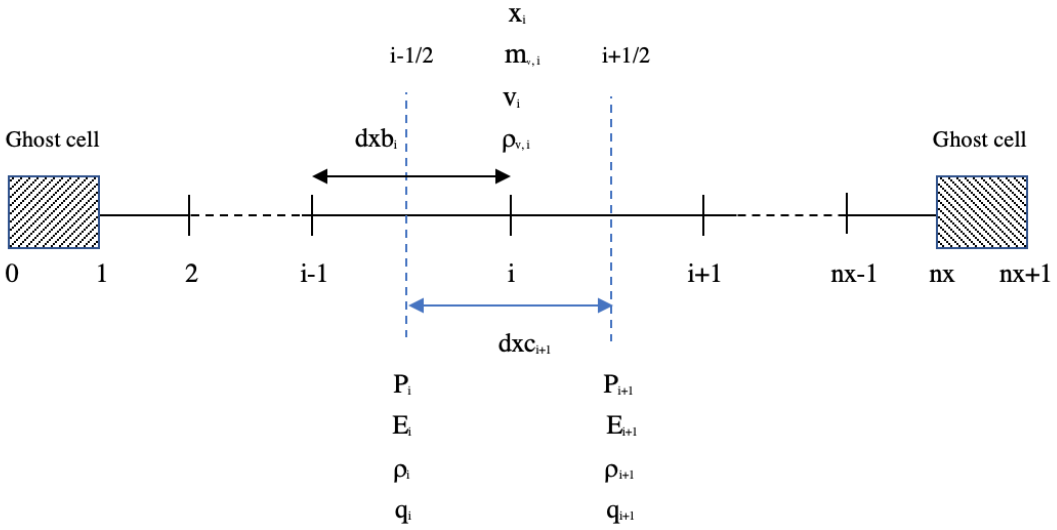

Figure 1: Schematic diagram showing the structure of the grid implemented as part of this Hydrodynamic code.

Note that the diagram shows ghost cells at either end of the domain. These are necessary to implement boundary conditions into the computational form of the Euler equations (discussed in Section 3). Boundary conditions set the physical properties of the simulation at the domain edge. Common forms of boundary conditions include: reflective boundaries, periodic, and solid wall. For the purposes of this simulation, we used a periodic boundary. This is defined by setting the values of the ghost cell to the values of the cell before, i.e for Pressure:

$$P_0^t = P_1^t \tag{5}$$

$$P_{nx+1}^t = P_{nx}^t \tag{6}$$

The equations to compute the time evolution of the fluid parameters in the Lagrangian frame can be written as follows:

$$\frac{d\rho}{dt} = -\rho \nabla \cdot \vec{u} \tag{7}$$

$$\rho \frac{d\vec{u}}{dt} = -\nabla P \tag{8}$$

$$\rho \frac{de}{dt} = -P \nabla \cdot \vec{u} \tag{9}$$

where all the terms are as previously described in the introduction.

2.3 The Rankine-Hugoniot equations

Hydrodynamic codes need to be able to deal with shock waves and their characteristic discontinuities - and to allow them to evolve correctly and yield the correct solutions, know as the Rankine-Hugoniot conditions.

The Rankine-Hugoniot conditions, describe the relationship between the states on either side of a shock wave in the flow of a one dimensional fluid. The equations are a result of the 3 conservation laws, and can be written as follows:

- Conservation of Mass:
$$\rho_1 u_1 = \rho_2 u_2 = m \tag{10}$$
- Conservation of Momentum:
$$\rho_1 u_1^2 + p_1 = \rho_2 u_2^2 + p_2 \tag{11}$$
- Conservation of Energy:
$$h_1 + \frac{u_1^2}{2} = h_2 + \frac{u_2^2}{2} \tag{12}$$

where m is the mass flow rate per unit area, ρ_i is the mass density of the fluid, u_i denotes the fluid velocity, p_i is the pressure, and h_i is the specific enthalpy. Note that the subscript, a_1, denotes the value of some parameter, a, upstream of the wave, and a_2 is the value of the downstream parameter as shown in Figure 2.

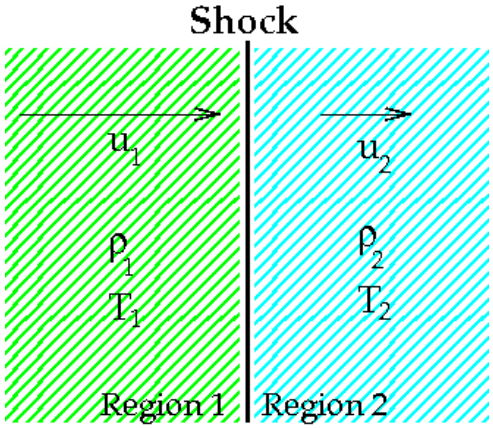

Figure 2: A schematic diagram of a shock wave situation with the density, velocity, and temperature indicated for each region.

These equations can be used to find the exact solution of the Sod shock-tube problem, a test Hydrodynamic scenario, which is discussed in Section 5.

3 Computational Methods

In order to solve the time evolution of the fluid parameters, we utilize a finite difference method to solve the Euler equations. There are several variations of this method and each have different orders of accuracy.

3.1 First Order Scheme

Here are the basic equations for a 1D upwind finite difference scheme:

$$v_i^{t+1} = v_i^t - \frac{dt}{2}\frac{\left(P_{i+1}^t - P_i^t\right)}{M_{v,i}} \tag{13}$$

$$x_i^{t+1} = x_i^t + \frac{dt}{2}v_i^t \tag{14}$$

$$e_i^{t+1} = e_i^t - \frac{dt}{2}P_i^t \times \frac{\left(v_i^t - v_{i-1}^t\right)}{M_{v,i}} \tag{15}$$

$$dxb_i^{t+1} = dxb_i^t + \frac{dt}{2} \times \left(v_i^t - v_{i-1}^t\right) \tag{16}$$

$$\rho_i^{t+1} = \frac{m_i}{dxb_i^{t+1}} \tag{17}$$

3.2 Predictor-Corrector Scheme

The idea behind the predictor corrector methods is to use a suitable combination of an explicit and an implicit technique to obtain a method with better convergence characteristics. The method calculates an estimate value at the half timestep, and then uses these values to get a more accurate value at the full timestep.

In practice, the equations used for the Lagrangian Predictor-Corrector method are as follows:

Predictor

$$v_i^{t+1/2} = v_i^t - \frac{dt}{2}\frac{\left(P_{i+1}^t - P_i^t\right)}{M_{v,i}} \tag{18}$$

$$x_i^{t+1/2} = x_i^t + \frac{dt}{2}v_i^t \tag{19}$$

$$e_i^{t+1/2} = e_i^t - \frac{dt}{2}P_i^t \times \frac{\left(v_i^t - v_{i-1}^t\right)}{M_{v,i}} \tag{20}$$

$$dxb_i^{t+1/2} = dxb_i^t + \frac{dt}{2} \times \left(v_i^t - v_{i-1}^t\right) \tag{21}$$

$$\rho_i^{t+1/2} = \frac{m_i}{dxb_i^{t+1/2}} \tag{22}$$

Corrector

$$v_i^{t+1} = v_i^{t+1/2} - \frac{dt}{2}\frac{\left(P_{i+1}^{t+1/2} - P_i^{t+1/2}\right)}{M_{v,i}} \tag{23}$$

$$x_i^{t+1} = x_{i+1/2}^t + \frac{dt}{2}v_i^{t+1/2} \tag{24}$$

$$e_i^{t+1} = e_i^{t+1/2} - \frac{dt}{2}P_i^{t+1/2} \times \frac{\left(v_i^{t+1/2} - v_{i-1}^{t+1/2}\right)}{M_{v,i}} \tag{25}$$

$$dxb_i^{t+1} = dxb_i^{t+1/2} + \frac{dt}{2} \times \left(v_i^{t+1/2} - v_{i-1}^{t+1/2}\right) \tag{26}$$

$$\rho_i^{t+1} = \frac{m_i}{dxb_i^{t+1}} \tag{27}$$

3.3 CFL Condition

The Courant-Friedrichs-Lewy (CFL) condition is a condition for the stability of unstable numerical methods that model convection or wave phenomena. As such, it plays an important role in Computational Fluid Dynamics and is relevant to the stability of this code.

Formally, the CFL condition is defined as: *the full numerical domain of dependence must contain the physical domain of dependence*. Therefore, the CFL condition expresses that the distance that any information travels during the timestep length within the mesh must be lower than the distance between mesh elements. In other words, information from a given cell or mesh element must propagate only to its immediate neighbours and the entanglement of nearby cells is restricted.

Generally, the Courant number (C), the physical parameter used to implement the CFL condition, is described by the following equation:

$$C = a\frac{\Delta t}{\Delta x} \tag{28}$$

where Δt is the timestep, Δx is the length between mesh elements, and a, is the velocity magnitude described by,

$$a = c_s + v \tag{29}$$

where v is the velocity of the fluid element and c_s is the sound speed given by:

$$c_s = \sqrt{\frac{\gamma P}{\rho}} \tag{30}$$

It follows that for any explicit simple linear convection problem, the Courant number must be equal to or smaller than 1, otherwise, the numerical viscosity would be negative, i.e $C \leq 1$.

Using this restriction on the Courant number, C, it is possible to calculate a timestep value for which the simulation is stable. Note that the CFL condition has to be valid across the entire spatial domain for each timestep, therefore we must consider the points at which it might fails - high velocities (large a), and small distance between neighbouring cells (small Δx). Therefore, we use the following equation to calculate the timestep for the j^{th} iteration:

$$\Delta t^j = C\frac{min(\Delta x^j)}{max(a^j)} \leq \frac{min(\Delta x^j)}{max(a^j)} \tag{31}$$

This condition sets a lower criteria for the stability of the dynamics, and when other phenomenon are considered such as shock waves and artificial viscosity, then a more restrictive condition must be put in place. This can usually be solved by significantly lowering the Courant number.

4 Artificial Viscosity

It is not possible to resolve a shock discontinuity in a first-principles physics code with a finite resolution. Due to the sharp gradients involved this creates distinct numerical oscillations in predictions due to overshoots if the shock is not effectively considered. This was originally addressed by von Neumann and Richtmyer who developed a quadratic viscosity formulation to smear the shock across multiple cells. The method was intended to complete the Rankine-Hugoniot jump conditions and remain negligible away from shocks.

The artificial viscosity provides a means of modelling discontinuous shocks by smearing it across a rapidly varying but continuous transition region while maintaining the Rankine-Hugoniot jump conditions. However, it must be properly implemented to ensure it critically damps the system.

Since the original form of artificial viscosity, first described by von Neumann and Richtmyer, then several alternative forms of artificial viscosity have been developed. The original equation for artificial viscosity, q, was given by:

$$q_{nl} = c_2 \rho (\Delta \mathbf{v})^{\mathbf{2}} \tag{32}$$

where c_2 is a constant of order unity, ρ is the density and $\Delta \mathbf{v}$ is the change in velocity. Note that the q value is zero in the case where $\Delta \mathbf{v} < \mathbf{0}$.

One of the most successful forms of artificial viscosity was developed by Kuropatenko, which contains linear and non-linear components and is given by:

$$q_{Kur} = \rho \left\{ c_2 \frac{(\gamma+1)}{4} |\Delta \mathbf{v}| + \sqrt{c_2^2 \Big(\frac{\gamma+1}{4} \Big)^2 (\Delta \mathbf{v})^2 + c_1^2 c_s^2} \right\} |\Delta \mathbf{v}| \tag{33}$$

where γ is the heat capacity ratio, c_1 is a constant which is usually set to unity, and c_s is the sound speed of the medium.

In order to successfully implement the artificial viscosity, it is necessary to add the q term whenever P is included, i.e $P \mapsto P + q$. This transforms the previous equations such that:

$$v_i^{t+1} = v_i^t - \frac{dt}{2} \frac{\big(P_{i+1}^t + q_{i+1}^t - P_i^t - q_i^t \big)}{M_{v,i}} \tag{34}$$

$$e_i^{t+1} = e_i^t - \frac{dt}{2} \big(P_i^t + q_i^t \big) \times \frac{\big(v_i^t - v_{i-1}^t \big)}{M_{v,i}} \tag{35}$$

and the sound speed is now defined by:

$$c_s = \sqrt{\frac{\gamma \big(P + q \big)}{\rho}} \tag{36}$$

Note that this will have a knock-on effect on the value of the artificial viscosity, q_{Kur}, and will also lower the value of each timestep iteration, Δt.

5 Results

In order to test the accuracy and function of a Hydrodynamics code, it is commonplace to use a test problem. There are several popular test problems such as:

- Sod Shock-Tube
- Sedov Explosion
- Isentropic Vortex
- Double Mach Reflection
- Shu-Osher

Only some of these problems are applicable to 1D Hydrodynamic codes, so for the purposes of testing my code, I will only be using the Sod Shock-Tube problem, and a simple Energy diffusion problem. In discussing these test, I will set out the initial conditions of the problem, present the results and discuss them.

5.1 Energy Diffusion

Diffusion is the process in which particles move from an area of high concentration to an area of low concentration. This is a relatively simple process and is used as a first stepping stone to ensure that the simulation is developing in time as expected, it is for this reason that I chose it as the first problem to test my code against.

Initially, we place a Gaussian energy distribution in the fluid that is centered in the middle of the spatial domain such that:

$$e_i = 1.0 + 2.0 \times \exp\left(\frac{-(x_i - 0.5)^2}{0.00\dot{6}}\right) \tag{37}$$

where x_i is the position of each cell vertex, which extends from 0 to 1. We also define the initial fluid as stationary, $v_i = 0$, isopycnic, $\rho = 1.0$, and ideal, such that we can use the ideal gas law: $p_i = \rho_i e_i(\gamma - 1)$. We also set $\gamma = 1.4$.

The initial fluid and its parameters are shown below in Figure 2.

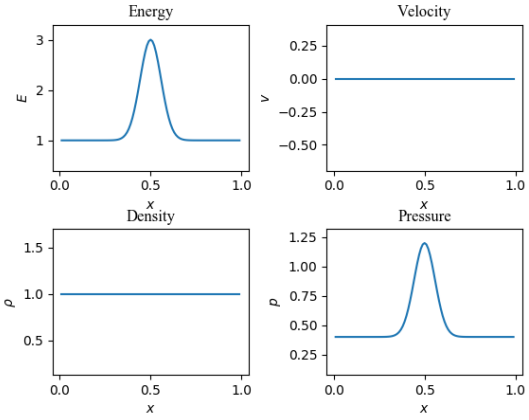

Figure 3: Initial conditions of the Energy diffusion problem. Note that the top left figure shows the Gaussian energy distribution.

As I was hoping, the energy distribution diffuses successfully as the simulation runs. The results are shown in Figure 3 below.

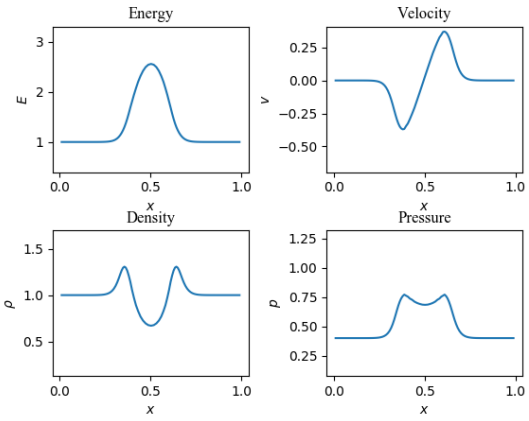

Figure 4: A snapshot of the simulation after 100 timesteps, at $t \approx 5.906 \times 10^{-2}$ s. The plot shows successful diffusion.

5.2 Sod's Shock Tube

This is one of the most popular tests for checking the implementation of artificial viscosity. The problem sets up a step function such that a shock-wave will form in the fluid. The parameters are described as follows:

$$e(x) = \begin{cases} 2.50J & \text{for } x < 0.5, \\ 2.00J & \text{for } x \geq 0.5 \end{cases}$$

$$\rho(x) = \begin{cases} 1.0 & \text{for } x < 0.5, \\ 0.1 & \text{for } x \geq 0.5 \end{cases}$$

$$P(x) = \begin{cases} 1.0 & \text{for } x < 0.5, \\ 0.1 & \text{for } x \geq 0.5 \end{cases}$$

Note that the conditions for $P(x)$ can be derived from the ideal gas equation. Initially, the fluid is stationary, $v_i = 0$, as is shown in Figure 4, and we assume that the tube is infinitely long, and that the fluid is inviscid. The spatial grid of the problem consists of 500 real cells and 2 ghost cells, and runs until 0.2s.

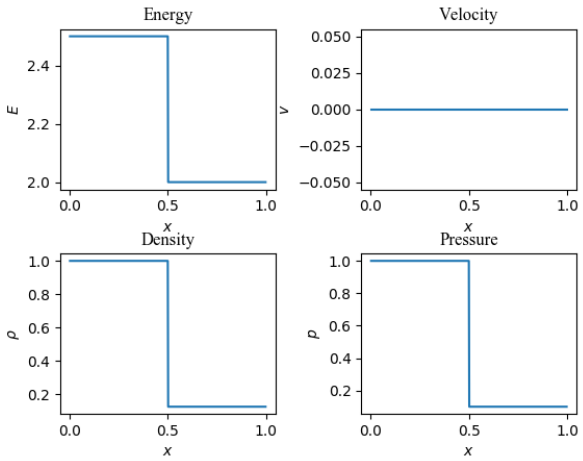

Figure 5: Initial conditions of the Sod Shock-Tube problem.

The simulation yields the following results at time, t=0.2s:

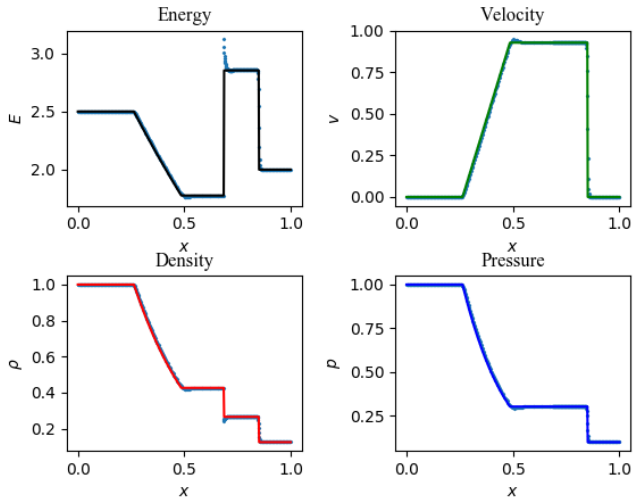

Figure 6: Initial conditions of the Energy diffusion problem. Note that the top left figure shows the Gaussian energy distribution.

As expected of a shock wave problem, the simulation results yield the Rankine-Hugoniot relations (as described in Section 2) and consists of 5 regions:

1. Left side of initial state
2. Expansion/Rarefaction wave
3. Steady state to left of contact
4. Steady state to right of contact
5. Right side of initial state

These regions are shown in Figure 6 below.

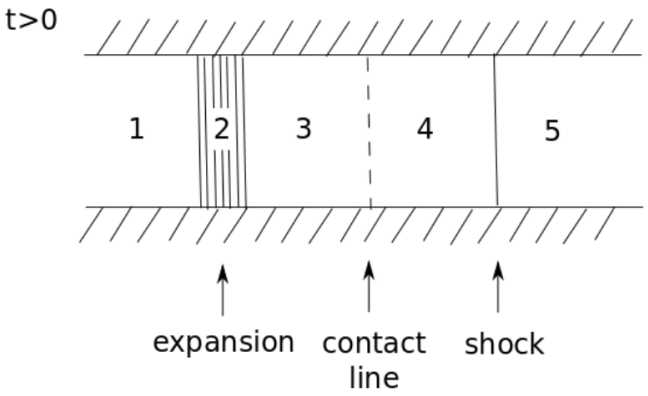

Figure 7: Schematic diagram showing the development of regions in the Sod shock-tube problem. Note that, in reality, the contact line is invisible, but it does separate two fluids of different entropy.

Notice that behind the shock front, but in front of the contact discontinuity, the energy of the fluid has increased above its pre-shock value, due to shock-heating. In order to treat the shock properly, we need to treat this problem as a multi-material problem or implement artificial conductivity.