Shocks and flux limiter

Julen Expósito

I. Introduction

In the hydrodynamic systems, governed by the fluid equations, it is common to develop discontinuities due to their non-linearity, which is a problem while we try to solve them numerically since we want the solution to be differentiable.

This makes difficult the study of many interesting physical processes like the supernova explosions, whose feedback is critical to simulate galaxy formation appropriately.

To be able to handle shocks in fluid simulations we can use a flux limiter. The first order schemes for fluid equation numerical solving tends to increase non-physical dissipation and simulate the system with low resolution, while a second order scheme gives us better resolution, but tends to create artefacts around physical discontinuities, of which first order schemes don't suffer. A flux limiter changes the resolution for derivatives calculation depending on the conditions around the point where we are solving the fluid equations, giving us the best of both worlds and thus allowing us to correctly simulate the shocks.

In this project, first, we have developed a simple shock configuration with the Riemann initial value problem, then we have implemented a flux limiter, and finally, we have used it to do a simple simulation of a supernova explosion.

II. Sod shock

The Riemann problem in 1D is an initial value problem where the initial conditions are determined with two constant values of the variables of the system separated by a finite discontinuity. For a 1D fluid, this means that the ICs are

$$\rho, P, v = \begin{cases} \rho_L, P_L, v_L & if \quad x < x_d \\ \\ \rho_R, P_R, v_R & if \quad x > x_d \end{cases}$$

To make these initial conditions differentiable we have approximated them with a function, for a variable u:

$$u = u_R + \frac{1}{2}\left(1 - \tan\left[\frac{x - x_d}{\epsilon \Delta x}\right]\right)(u_L - u_R),$$

where ϵ is the size of the shock in Δx units.

We also need to limit the domain, so $x_i \leq x \leq x_f$. The boundary conditions, in this case, impose derivative equal to zero. This can be easily implemented numerically just by defining the numerical domain as $x_i - 0.5dx < x < x_f + 0.5dx$ (so, for an array of the variable u, $u[1:-1]$ corresponds to the physical points and $u[0]$ and $u[-1]$ are outside the physical domain) and imposing

$$u[0] = u[1], \qquad u[-1] = u[-2].$$

We have solved this problem using two different numerical schemes; a first order scheme (Lax-Friedrich, LF from now on) and a second order scheme (Lax-Wendroff-Ritchmyer, LWR from now on). The initial parameters are

$$\rho_L = 1, \qquad \rho_R = 0.2,$$

$$P_L = 3.5, \qquad P_R = 0.5,$$

$$v_L = 0, \qquad v_R = 0.$$

We have used a CFL number of 0.9 and $\epsilon = 5$. Note that the physical system corresponds to $\epsilon \to 0$, but these numerical schemes are incapable of handling a very small value of ϵ since we need to smooth more the physical discontinuity to avoid numerical artefacts (this, as we will see, will change with the flux limiter). An ϵ value of 5 with a Δx sufficiently small let us obtain fine results for this simple system, though.

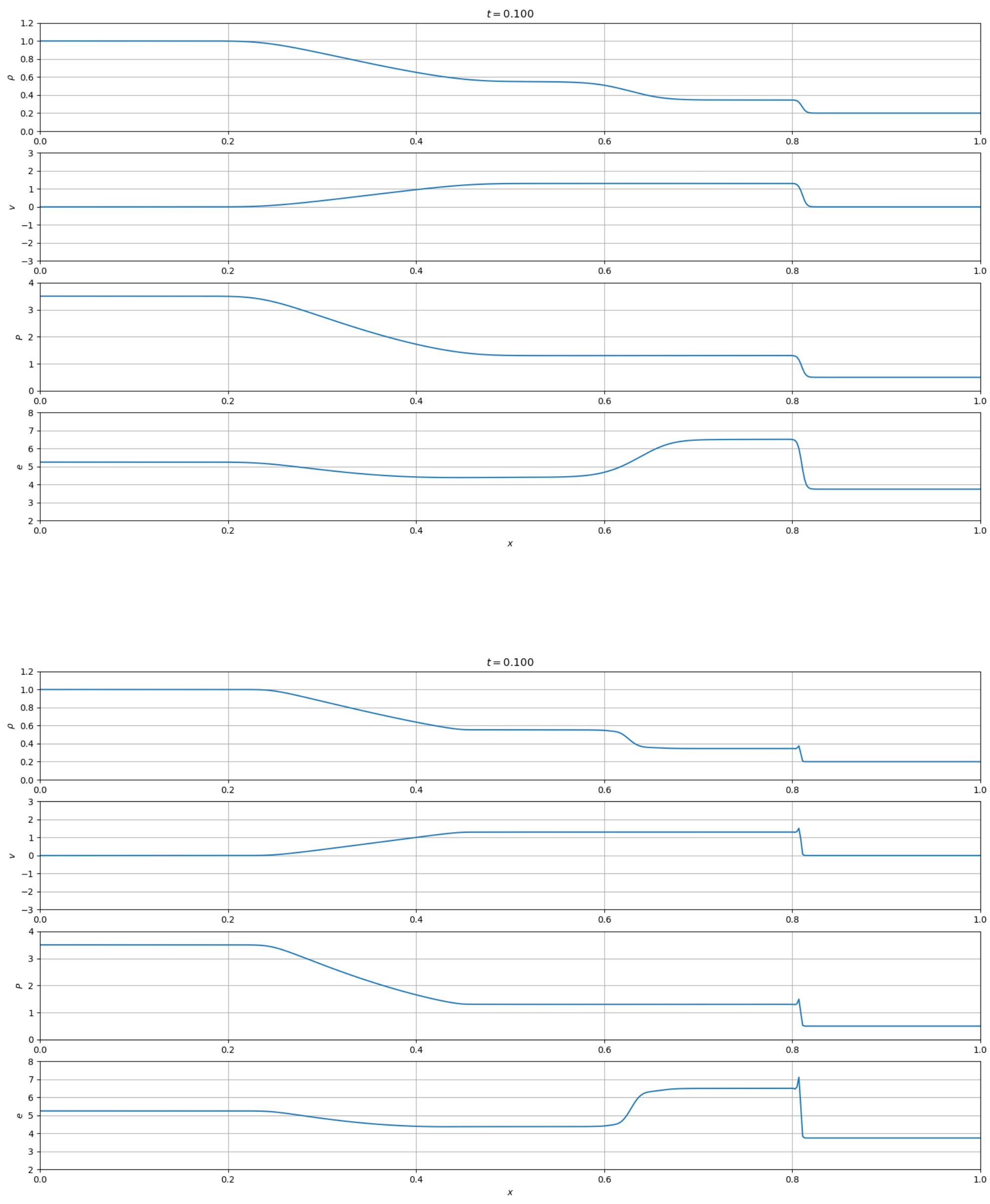
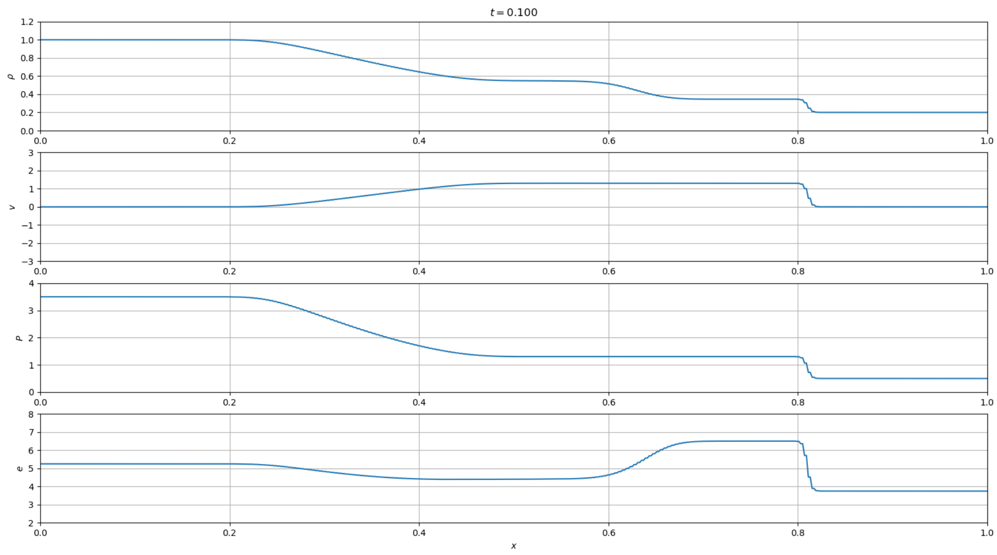


Figure 1: *Density, flux velocity, pressure and internal energy for a simulated sod shock with a CFL number of 0.9, $\epsilon = 5$ and $N = 500$ points. Up: LR scheme. Down: LWR scheme.*

There can be observed 3 different fluid states, as expected. We expect the shock (the abrupt descent of density at the right, which is moving in that direction) to be extremely thin in the physical system. As it can be seen, the first order scheme flattens it more than the second order scheme, but this scheme forms non-physical peaks as the shock due to its incapability of maintaining monotonicality.

As we have said before, these problems can be handled with a flux limiter.

Before going forward, it is interesting to see the effect of running this schemes with $\epsilon \to 0$:



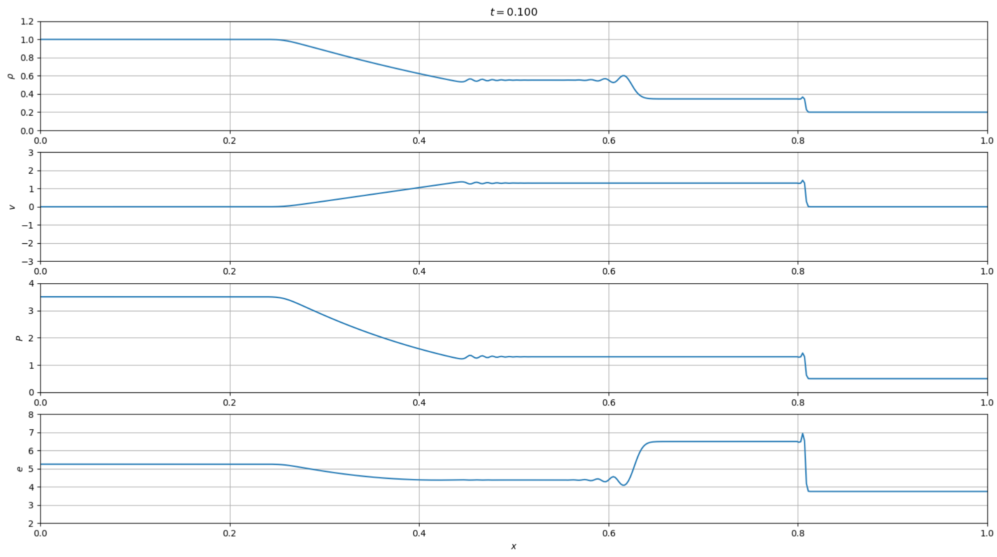


Figure 2: *Density, flux velocity, pressure and internal energy for a simulated sod shock with a CFL number of 0.9,* $\epsilon = 10^{-10}$ *and* $N = 500$ *points. Up: LR scheme. Down: LWR scheme.*

We can see clear problems with both schemes. The number and size of the non-physical artefacts has increased in response to the abrupt change in values the initial conditions have around x_d. Note, in any case, how the LWR scheme has develop more and worse artefacts than LF, which only forms an staggered shape around the shock (and also have the usual extra dissipation, of course).

III. Limiter

The second order scheme forms artefacts at the points with high gradient, so we are interested in lowering the resolution at these points while maintaining a high resolution in smooth flows. To detect the gradients we use the relative gradient

$$r_i = \frac{u_i - u_{i-1}}{u_{i+1} - u_i}.$$

Note that r will tend to zero when we encounter steep gradients only for increasing gradients. For decreasing ones, its absolute value will tend to infinity. An r close to 1 implies a smooth region where the gradient isn't changing significantly.

The flux limiter is a function $\phi(r)$ that indicates the degree of contribution of low order and high order flow based on the relative gradient value. As our shock moves only in one direction, there is no problem in considering that a value of $\phi(r)$ close to 0 means there is a steep gradient, while for greater values of $\phi(r)$ we have smoother flows (i.e. $\phi(r)$ can ignore that $r >> 1$ also implies steep gradients).

The numerical implementation of r needs to handle the situation where $u_{i+1} - u_i$ is 0. In Python, this will give us an $\pm$ *Inf.* if $u_i - u_{i-1}$ isn't zero, which can be handled, if needed, imposing a maximum absolute value for r. If $u_i - u_{i-1} = 0$, Python will give us a NaN (Not a Number). Note that $r_i = 0/0$ means that in the position x_i the gradient isn't changing, so r has a finite real solution of 1. Therefore, the code interprets all NaNs as 1.

The scheme has the form

$$u_i^{n+1} = u_i^n - \frac{\Delta t}{\Delta x}\left(F_{i+1/2}^{n+1/2} - F_{i-1/2}^{n+1/2}\right),$$

where, in terms of the first order ($f_{i,l}^n$) and second order ($f_{i,h}^n$) fluxes:

$$F_{i+1/2}^{n+1/2} = [1 - \phi(r_i)]\, f_{i+1/2,l}^{n+1/2} + \phi(r_i) f_{i+1/2,h}^{n+1/2},$$

$$F_{i-1/2}^{n+1/2} = [1 - \phi(r_i)]\, f_{i-1/2,l}^{n+1/2} + \phi(r_i) f_{i-1/2,h}^{n+1/2}.$$

The implementation of the second order flux is direct from the explicit LWR scheme. For the first order fluxes, we have

$$f_{i+1/2,l}^{n+1/2} = \frac{1}{2} f_{i+1}^n - \frac{\Delta x}{2\Delta t}\left(u_{i+1}^n - u_i^n\right),$$

$$f_{i-1/2,l}^{n+1/2} = \frac{1}{2} f_{i-1}^n + \frac{\Delta x}{2\Delta t}\left(u_{i-1}^n - u_i^n\right).$$

Note that these fluxes recover the LF scheme for $\phi(r) = 0$. On the other hand, for $\phi(r) = 1$ we recover the LWR scheme.

For systems like these, a well performing flux limiter is the Minmod limiter:

$$\phi(r) = \max\left[0, \min(1, r)\right],$$

which returns 0 for any negative value of r, what we want because that implies a peak, where the first order scheme's contribution should maximize and the second order scheme's contribution should minimize. It also returns 1 for any value of r greater than 1, so the code will maximize the resolution for smooth fluxes and very high relative fluxes (which we have already argued why it isn't a problem for systems where the shocks only go in one direction).

Note that the use of the Minmod eliminates the need of avoiding the infinite values for r since $\phi(r)$ will handle them without any problem. Even so, our code does have a limit for the absolute value of r to avoid overflows while making use of flux limiters that do direct operations with r. This won't change the values the Minmod returns since all $r > 1$ will lead to $\phi(r) = 1$.

Let's see how the code performs with the minmod limiter and $\epsilon \to 0$:

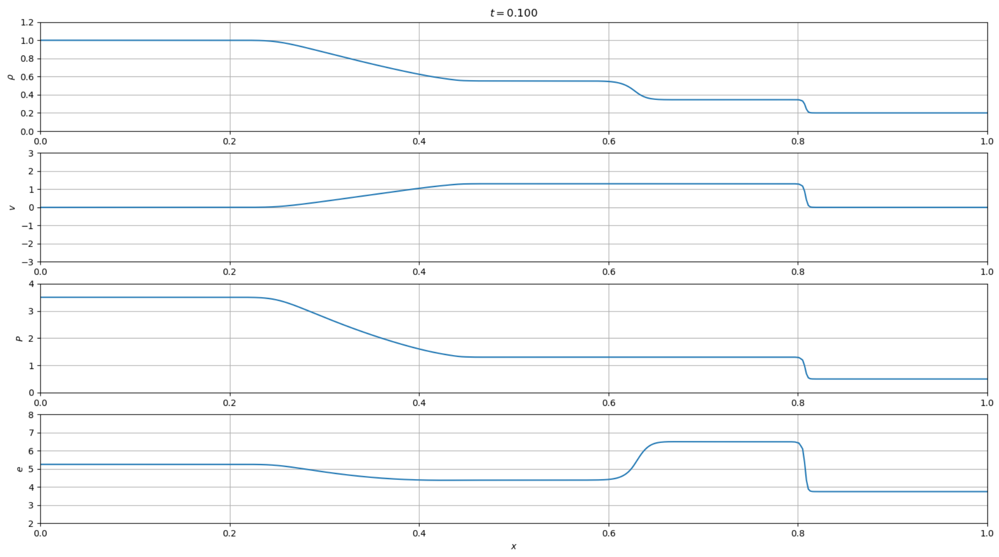


Figure 3: *Density, flux velocity, pressure and internal energy for a simulated sod shock with a CFL number of 0.9,* $\epsilon = 10^{-10}$ *and* $N = 500$ *points. Minmod flux limiter.*

The artefacts are nowhere to be seen. Note that the extra dissipation characteristic of the LF scheme has decreased too, as expected.

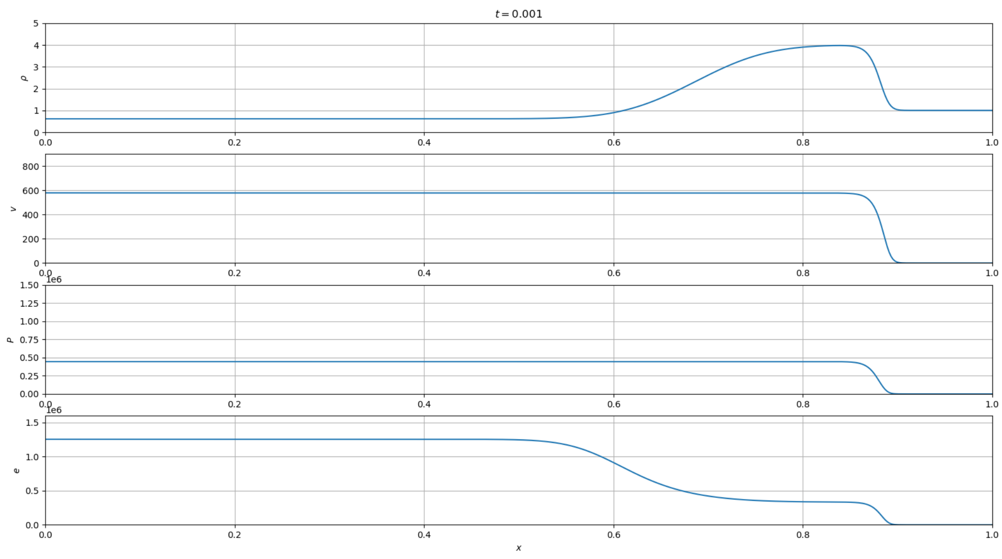
IV. Supernova explosion

We can simulate a supernova explosion with this same code. Specifically, we can simulate it as a Riemann problem where the initial pressure on one side is much greater than the pressure on the other. Let's have $x_d = 0.1$ and

$$\rho_L = 1, \qquad \rho_R = 1,$$

$$P_L = 10^6, \qquad P_R = 1,$$

$$v_L = 0, \qquad v_R = 0.$$



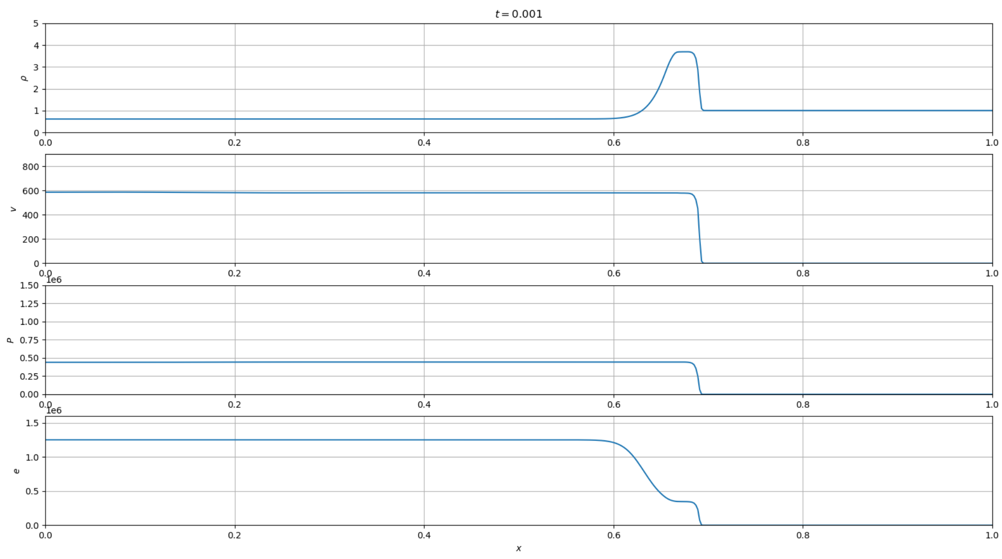
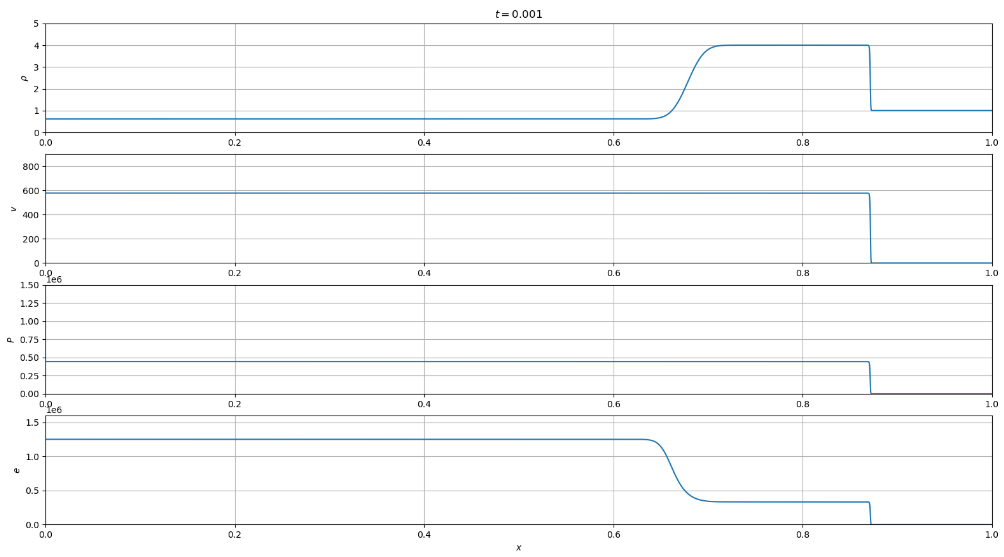


Figure 4: *Density, flux velocity, pressure and internal energy for a simulated supernova explosion with a CFL number of 0.9 and $N = 500$ points. Up: LR scheme, $\epsilon = 5$. Down: Minmod flux limiter, $\epsilon = 10^{-10}$.*

In this case, a much notorious difference can be observed between the schemes. The LWR isn't represented because the artefacts it creates grow very fast and destroys rapidly the simulation.

The non-physical dissipation effect of the LF scheme is clearly seen here. The use of the flux limiter avoids it. Also, the shock seems to be slower than the one simulated by the LF scheme. In the next section, we will discuss if this is a problem of the limiter or a problem of the LF scheme. The density contrast is different too: the LF scheme predicts a value of 3.97, while the simulation with the flux limiter predicts a value of 3.69. The analytical solution gives a density contrast of 4, so the LF scheme is performing better in this regard.

It may be a good idea to increase the spatial resolution. Let's try to simulate the shock with $N = 10^4$ points:



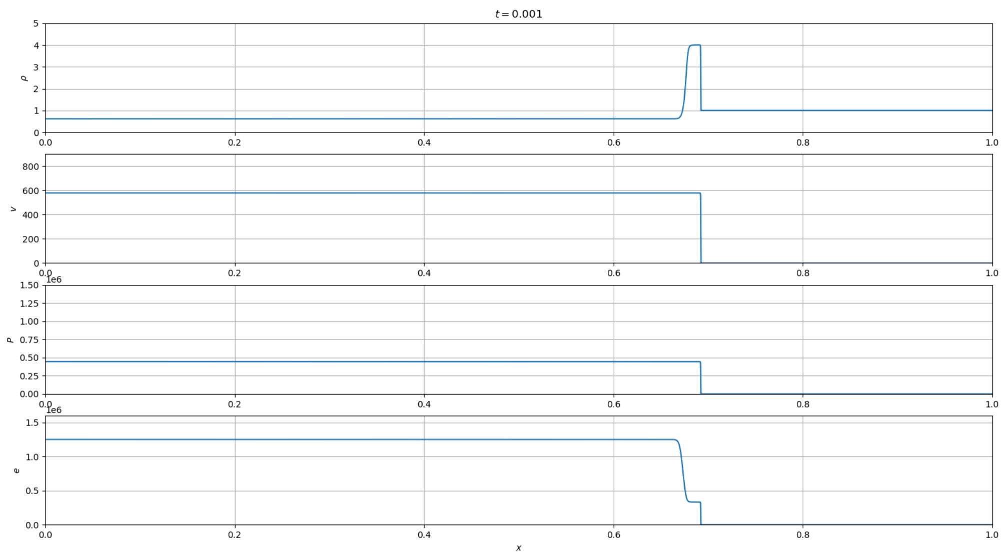


Figure 5: *Density, flux velocity, pressure and internal energy for a simulated supernova explosion with a CFL number of 0.9 and $N = 10^4$ points. Up: LR scheme, $\epsilon = 5$. Down: Minmod flux limiter, $\epsilon = 10^{-10}$.*

Note how thin the shock front really is. In this case, both LF and Minmod give a density contrast of 4.00, which indicates that the incorrect result of the simulation with the flux limiter was only a resolution problem. However, the shock velocity difference is maintained.

V. Mach numbers

The Mach number of the shock is the velocity of the shock divided by the sound velocity of the unperturbed medium. In the Riemann problem and applying the Rankine-Hugoniot conditions, the analytical Mach number of the shock will be:

$$M_0 = \left[\frac{\gamma+1}{2\gamma}\left(\frac{p_1}{p_0} + \frac{\gamma-1}{\gamma+1}\right)\right]^{1/2},$$

where p_1 is the pressure in the post-shock region and p_0 the pressure of the unperturbed medium.

We can find the approximate shock position finding the x_i where $|dv/dx|$ has its maximum value. The shock region, even if it is small, covers a finite space (a multiple of Δx), so, in order to find a point we know correspond to the post-shock region (but, in the case of the sod shock, a point that is also at the right of the contact discontinuity) and a point we know correspond to the region where the shock doesn't have arrived yet, we have to decrease and increase a certain number of Δx to the shock position x_i. That number will depend on the total number of points N.

In the code, we have defined a post-shock region point as a point $N/10$ to the left of the shock position and an unperturbed region point as a point $N/10$ to the right of the shock position. This works fine in the two studied systems.

The Mach number should be constant, but it is expected the value to change a little with time due to the dependence of the simulated p_0 and p_1 values, which will be only approximately constant. The best calculation of the analytical Mach number is, then, to calculate it at all simulated times after the shock is fully created and propagating normally and take the mean value. In the code, it is assumed the simulation will stop sufficiently late for half of the stopping time to be a good starting point for Mach number calculation.

Numerically, we have to obtain the velocity of the shock looking for the change with time of the shock position. For this, we have created an array with all the shock positions after the shock is fully created and propagating normally (as said, the code consider this to be fulfilled at half of the stop time). As the shock velocity is constant, we can then use this array to fit the function $x(t) = vt + c$. The slope of the line is the shock velocity.

Then, we have calculated the numerical Mach number taking the shock velocity and dividing it by the sound speed at the end of the sound speed array (because that position corresponds to the unperturbed medium).

For the sod shock:

M_0	Sod shock (LF)	Sod shock (LRW)	Sod shock (minmod)
Analytical solution	1.504	1.514	1.511
Numerical solution	1.518	1.515	1.498

Table 1: *Analytical and numerical mach numbers calculated in simulations with a CFL number of 0.9, $\epsilon = 5$ and $N = 500$ points.*

The results are pretty consistent.

M_0	Supernova (LF)	Supernova (minmod)
Analytical solution	597.196	594.990
Numerical solution	597.836	457.666

Table 2: *Analytical and numerical mach numbers calculated in simulations with a CFL number of 0.9, $\epsilon = 10^{-10}$ and $N = 500$ points.*

As can be seen, the supernova shock is extremely supersonic. The results for the LF scheme are consistent, but the numerical Mach number of the supernova simulated with a flux limiter is slower than the analytically predicted value. The limiter is artificially slowing down the shock.

VI. Appendix

i. Constant derivatives as boundary conditions.

In this project, we have imposed derivatives equal to zero as boundary conditions, but it is interesting to think about how to implement constant derivatives.

The centered derivative for the first point of the physical domain is, if we remember how we have created our numerical domain array:

$$\frac{u[1]-u[0]}{dx}.$$

As we want the derivative to be constant at the boundary, we impose

$$\frac{u[1]-u[0]}{dx}=\frac{u[2]-u[1]}{dx},$$

$$u[0]=2u[1]-u[2].$$

For the end of the array we can apply the same:

$$\frac{u[-1]-u[-2]}{dx}=\frac{u[-2]-u[-3]}{dx},$$

$$u[-1]=2u[-2]-u[-3].$$

ii. Other flux limiters.

It may be interesting to try other flux limiters for comparison purposes. The minmod limiter only compares values and select one following criterion (it isn't a continuous function of r; it only copies its value and limit it from 0 to 1), but we could have, for example, an injective flux limiter such as

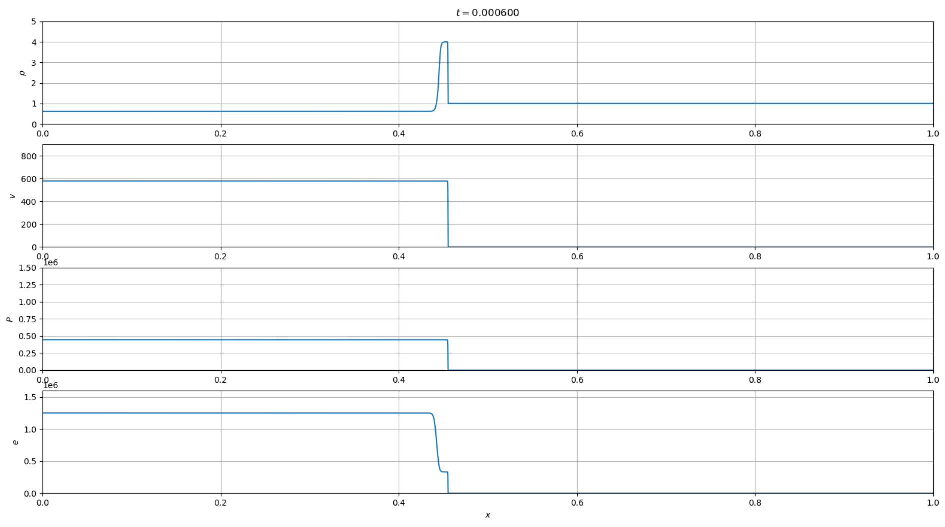
$$\phi(r)=\frac{r^2+r}{r^2+1}.$$

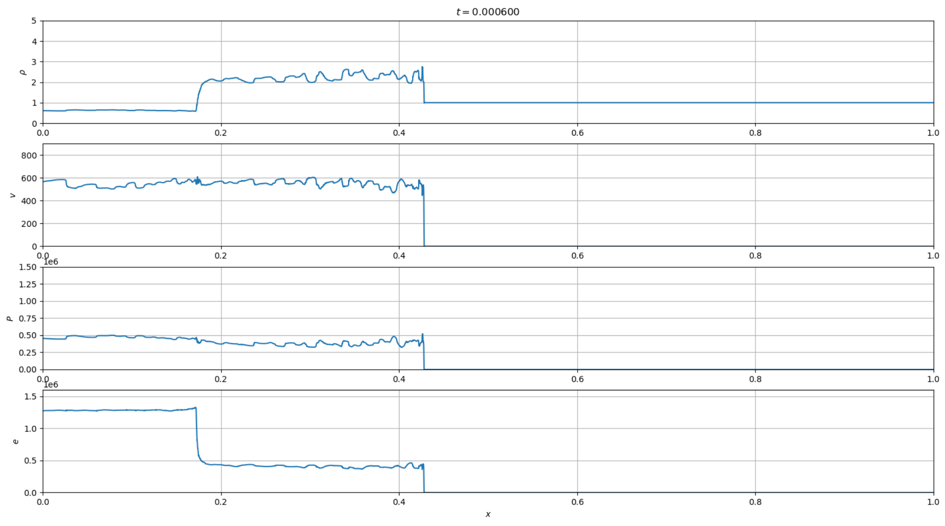
This is the Van Albada flux limiter. Note that, as the minmod limiter, it returns a value of 1 for $r\to\infty$, but in this case, it can return values outside the [0,1] range for values of the relative gradient outside that same range. Also, it doesn't have a linear relation with r inside the [0,1] range, but a concave one.

We can also expand the Minmod limiter to greater possible values. See the Osher limiter, for example:

$$\phi(r)=\max\left[0,\min(1,\beta)\right], \qquad (1\leq\beta\leq 2).$$

Let's see how these limiters perform in comparison with the Minmod limiter:





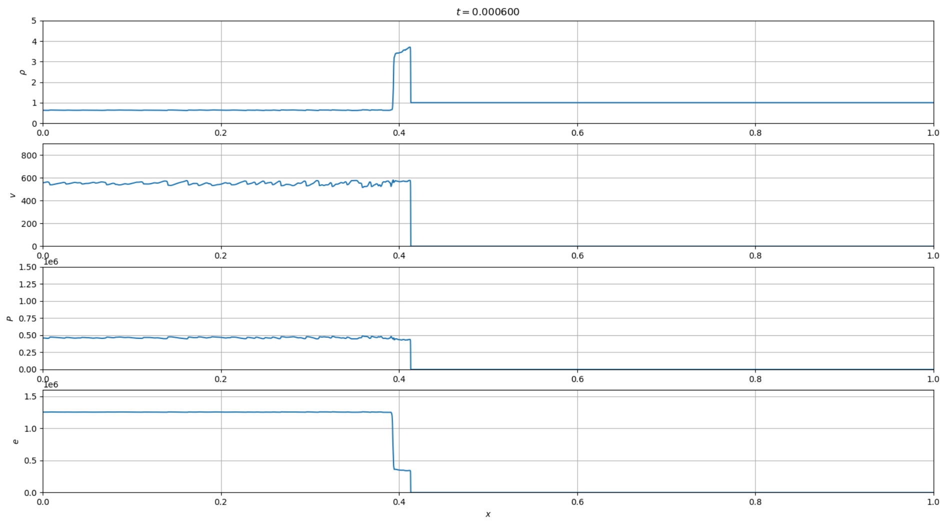


Figure 6: *Density, flux velocity, pressure and internal energy for a simulated supernova explosion with a CFL number of 0.9, $\epsilon = 10^{-10}$ and $N = 10^4$ points. In order from top to bottom: Minmod, Osher ($\beta = 2$) and Van Albada limiters.*

It seems these systems don't respond well to limiters that return values outside the [0,1] range.