Relativistic Hydrocode

Joseph Corrado
Andrew MacFadyen
Chris Tiede

December 21, 2018

1 Reconstruction Techniques

1.1 First Order in Space

At each interface the U_j^n manifests a jump which generates left and right states of the local riemann problem. The original idea by Godunov was to use a piecewise constant reconstruction of the values of the conserved variables throughout each of the cells in order to solve the riemann problem at each interface. This method works to first order in space, as it takes no information about the derivatives of the function into account. This reconstruction is given by:

$$U(x,0) = \begin{cases} U_j^n, & x < x_{j+1/2} \\ U_{j+1}^n, & x > x_{j+1/2} \end{cases}$$

When we use this reconstruction method, we only need one guard cell on either side because we are simply taking the left and right hand conserved states on either side of every boundary to compute the numerical flux across each boundary.

1.2 Total variation Diminishing Methods

The monotonicty of a solution is closely related to the appearance of oscillations. Given a numerical method, where evolution of the conserved variables forward in time is a function of the current conserved variables, a method is called monotone if:

$$\frac{\partial f}{\partial \boldsymbol{U}_j^n} \geq 0, \quad \forall \boldsymbol{U}_j^n$$

If we require that out method be monotone, this is equivalent to all of the coefficients of the linear combination used to propagate the solution forward in time need to be positive semi-definite. If we apply this requiremenet, we obtain the result that the solution will satisfy:

$$\max_j\{\boldsymbol{U}_j^{n+1}\} \leq \max_j\{\boldsymbol{U}_j^n\}, \quad \min_j\{\boldsymbol{U}_j^{n+1}\} \geq \min_j\{\boldsymbol{U}_j^n\}$$

If we have these properties for our solution, then our method will not introduce spurious new extrema in the solution as time evolves. In this sense, the monotonicity of a numerical scheme is closely related to the potential appearance of spurious oscillations in the solution across a discontinuity. In order to attenuate these oscillations, we can introduce artificial velocity. In order to measure the oscillations appearing in the solution, we define the total variation as:

$$TV(\boldsymbol{U}^n) = \sum_{j=-\infty}^{\infty} |\boldsymbol{U}_j^n - bmj - 1^n|$$

Requiring that the scheme does not produce spurious oscillations is equivalent to requiring that the total variation does not increase in time, or that the total variation is bounded by the total variation of the initial data. Given this definition of variation, a numerical method is total vriation diminishing (TVD) if our solution satisfies:

$$TV(\boldsymbol{U}^{n+1}) \leq TV(\boldsymbol{U}^n)$$

for every timestep. TVD methods differ from artificial viscosity by controlling the appearance of oscillations in the solution without explicitly introducing viscous terms in the numerical scheme, and by reducing the oscillations of the methods by the method itself.
One way to characterize schemes by the ability to not introduce or increase oscillations in the solution, we consider methods such that:

$$\boldsymbol{U}_j^n \geq \boldsymbol{U}_{j+1}^n, \quad \forall j$$

Methods that satisfy this constraint are said to be monotonicity preserving. This results in preventing the appearance of oscillations since monotonicity-preserving methods do not permit new local extrema as the value of local minima cannot decrease and the value of local maxima cannot increase. A result due to Harten states that any monotone numerical method is TVD, and any TVD method is monotonicity preserving. However, this comes at the cost that TVD methods do not extend beyond second-order accuracy, and they reduce to first-order at local extrema. If we want to construct higher-order methods, we must drop the total variation condition. In practice, the TVD property can be imposed through a slope limiter method. A theorem encapsulating the restrictions on TVD methods was given by Godunov: a linear and monotonicity-preserving scheme is at most first-order accurate.
Modern HRSC methods all possess the properties:

1. at least second order accuracy on smooth parts of the solution
2. sharp resolution of discontinuities without large smearing
3. absence of spurious oscillations in the solution
4. convergence to the true solution as the grid is refined
5. no use of artificial viscosity terms

1.3 Second Order in Space

As mentioned above, slope limiter methods can be used to improve the representation of the solution within the cell while imposing a TVD property on the method. In order to increase the order of convergence, we can perform a piecewise linear reconstruction inside each cell:

$$U_j^n(x) = U_j^n + \sigma_j^n(x - x_j), \quad x_{j-1/2} \leq x \leq x_{j+1/2}$$

The slope limiter method I used was a generalized minmod method. In this reconstruction, the slope σ is given by the minmod function evaluated on the adjacent states of the cosnerved variables. The minmod function is defined as:

$$minmod(x, y, z) = \frac{1}{4}|sign(x) + sign(y)|(sign(x) + sign(z)) \min(|x|, |y|, |z|)$$

Using this method, we need to use two cells on either side of every boundary, and so we need two guard cells on either side of the grid for this reconstruction technique. This reconstruction is given by:

$$U(x, 0) = \begin{cases} U_j^+, & x < x_{j+1/2} \\ U_{j+1}^-, & x > x_{j+1/2} \end{cases}$$

Where $U_j^+ = U_j(x_j + 1/2)$ and $U_j^- = U_j(x_{j-1/2})$. Where we are interpreting integer values of the subscript to be the bin centers, and half integer values of the subscripts refer to the boundaries between cells. The reconstructed states are then given by:

$$c^L_{i+1/2} = c_i + 0.5 minmod(\theta(c_i - c_{i-1}), 0.5(c_{i+1} - c_{i-1}), \theta(c_{i+1} - c_i))$$

$$c^R_{i+1/2} = c_{i+1} - 0.5 minmod(\theta(c_{i+1} - c_i), 0.5(c_{i+2} - c_i), \theta(c_{i+2} - c_{i+1}))$$

This generalized minmod function is a single parameter version of slope limiter, and has the property that when $\theta = 1$, it becomes more diffusive normal minmod limiter, and when $\theta = 2$, it becomes the monotonized central-difference limiter. In my code, I used $\theta = 1.5$. The minmod limiter always takes the least steep slope amongs tthose provided by the backward and forward finite-difference, while the MC limiter compares three different slopes and takes the least steep amongst the three.

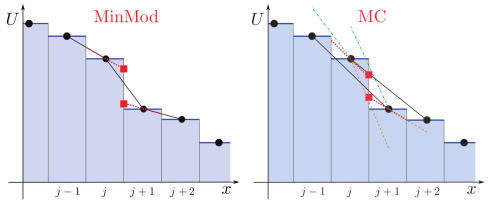

Figure 1: Minmod and MC Slope Limiters [3]

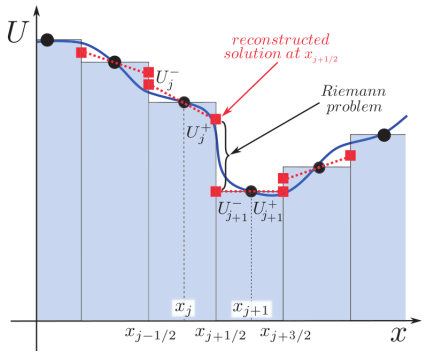

Figure 2: Schematic Representation of Boundary-Extrapolated Values, [3]

2 Conserved Variables

2.1 Nonrelativistic Case

The euler equations are given by:

$$\frac{\partial \boldsymbol{U}}{\partial t} + \frac{\partial \boldsymbol{F}}{\partial x} = 0$$

In this case, the conserved variables are given by:

$$U = (\rho, \rho v, E)^T$$

Where ρ is the fluid density, v is the velocity, P is the pressure, and E is the total energy density. The expression for the total energy density is $E = \rho\epsilon + \frac{1}{2}\rho v^2$ where ϵ is the specific internal energy. These equations are closed by the equation of state for an ideal gas, given by:

$$p = (\gamma - 1)\rho\epsilon$$

2.2 Relativistic Case

In the relativistic case, the conserved quantities are slightly different, given by $U = (D, S, \tau)^T$. In this expression, D is the rest mass density, S is the momentum density, and τ is the energy density, all as measured in the laboratory frame. The relations relating these conserved variables back to the primitives are:

$$D = \rho W$$

$$S = \rho h W^2 v$$

$$\tau = \rho h W^2 - p - \rho W$$

$$W^2 = \frac{1}{1 - v^2}$$

For these conversions, $h = 1 + \epsilon + p/\rho$ is the relativistic specific enthalpy, and ϵ is the specific internal energy.

3 Physical Fluxes

3.1 Nonrelativistic Fluxes

We can compute the physical fluxes from the expressions for the fluxes as a function of the conserved variables and primitive variables. The physical fluxes are given from the euler equations by:

$$F = \begin{bmatrix} \rho v \\ \rho v^2 + p \\ (E + p)v \end{bmatrix}$$

After retrieving the primitives from the conserved variables, we can compute the physical fluxes for each of the physical cells.

3.2 Relativistic Fluxes

In the relativistic case, we build the physical fluxes out of the new conserved variables.

$$F = (Dv, Sv + p, S - Dv)^T$$

4 Approximate Riemann Solver

Computing the exact solution to the riemann problem at the boundary of each of the cells is computationally expensive because it involves iterations over all of the cells for every timestep. In order to avoid doing this to be more computationally efficient, modern high resolution shock capturing schemes employ approximate riemann solvers to numerically approximate the solutions to the riemann problem at each of the boundaries. In my project, I used one of the simplest numerical flux functions, the FHLL flux. This flux approximates the solution to the local riemann problem as a two-state solution with an intermediate state in the middle. Approximate riemann solvers can be divided into complete and incomplete riemann solvers. Incomplete riemann solvers contain only a subset of the characteristic fields of exact solutions, whereas complete riemann solvers contain all of the characteristic fields of the exact solution. Complete riemann solvers are fully upwind and are able to capture any discontinuity or wave that forms during the evolution. Incomplete riemann solvers are usually only upwind with respect to the fastest possible waves that are produced in the riemann fan, but they are susceptible to losing information about intermediate waves.

4.1 The HLLE Numerical Solver

This solver assmumes that after the decay of the initial discontinuity, only two waves propagate in two opposite directions with velocities λ_L and λ_R, and a single constant state between them:

$$\boldsymbol{U}(x,t) = \begin{cases} \boldsymbol{U}_L, & x/t < \lambda_L \\ \boldsymbol{U}^{HLLE}, & \lambda_L < x/t < \lambda_R \\ \boldsymbol{U}_R, & x/t > \lambda_R \end{cases}$$

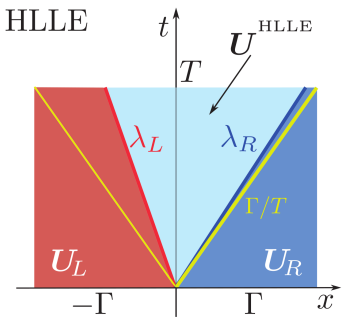

Figure 3: Spacetime control Volume for the computation of the approximate HLLE FLux, [3]

By applying the Rankine-Hugoniot conditions across the left and right waves, we can compute the constant state $\boldsymbol{U}^{HLLE}$, given that we know the signal speeds λ_L and λ_R. Doing this, we obtain the HLLE flux:

$$F_* = \frac{\lambda_R \boldsymbol{F}_L - \lambda_L \boldsymbol{F}_R + \lambda_L \lambda_R (\boldsymbol{U}_R - \boldsymbol{U}_L)}{\lambda_R - \lambda_L}$$

We use the HLLE flux in the godnuov method using the logic:

$$\boldsymbol{F}^{HLLE} = \begin{cases} \boldsymbol{F}_L, & x/t < \lambda_L \\ \boldsymbol{F}_*, & \lambda_L < x/t < \lambda_R \\ \boldsymbol{F}_R, & x/t > \lambda_R \end{cases}$$

THis riemann solver has the pro of being very simple, and that it performs well at sonic rarefactions, but due to the fact that the middle waves are ignored in the solution, it shows exceissve smearing at contact discontinuities. To use this riemann solver, we need to compute the wave speeds λ_L and λ_R in a way that is appropriate for special relativistic hydrodynamics:

$$\lambda_L = \min(0, \lambda_-(\boldsymbol{U}_L), \lambda_-(\boldsymbol{U}_R))$$

$$\lambda_R = \max(0, \lambda_+(\boldsymbol{U}_L), \lambda_+(\boldsymbol{U}_R))$$

where λ_- and λ_+ are given by the appropriate eigenvalues of the jacobian matrix.

4.2 The HLLC Numerical Solver

Although the HLLE riemann solver is more simple, it suffers from strong limitations in terms of capturing contact and tangential waves. The HLLC riemann solver was introduced in order to restore the missing information about the intermediate contact discontinuity by using two intermediate states, $\boldsymbol{U}_{L_*}$ and $\boldsymbol{U}_{R_*}$. Using this, the complete solution is given by:

$$\boldsymbol{U}(x,t) = \begin{cases} \boldsymbol{U}_L, & x/t < \lambda_L \\ \boldsymbol{U}_{L_*}, & \lambda_L < x/t < \lambda_C \\ \boldsymbol{U}_{R_*}, & \lambda_C < x/t < \lambda_R \\ \boldsymbol{U}_R, & x/t > \lambda_R \end{cases}$$

where $\lambda_L \leq 0$ and $\lambda_R \geq 0$ are the smallest and largest of the characteristics of the solution to the local Riemann problem, and λ_C is the speed of the contact discontinuity. The HLLC flux is given by:

$$\boldsymbol{F}^{HLLC} = \begin{cases} \boldsymbol{F}_L, & x/t < \lambda_L \\ \boldsymbol{F}_{L_*}, & \lambda_L < x/t < \lambda_C \\ \boldsymbol{F}_{R_*}, & \lambda_C < x/t < \lambda_R \\ \boldsymbol{F}_R, & x/t > \lambda_R \end{cases}$$

For this solution, it is required an estimate for the speeds λ_L, λ_C, and λ_R. For λ_L and λ_R, one can use the same strategy used for the HLLE riemann solver, in which we used the largest and smallest eigenvalues of the jacobian matrix. An estimate for λ_C is obtained as the speed of the contact discontinuity in the analytic solution of the two-rarefaction riemann solver.

4.3 My Code

In my code, I implemented the HLLE riemann solver for the nonrelativistic and relativistic versions. In the future I would like to implement the HLLC riemann solver as well as other approximate riemann solvers and compare their performance.

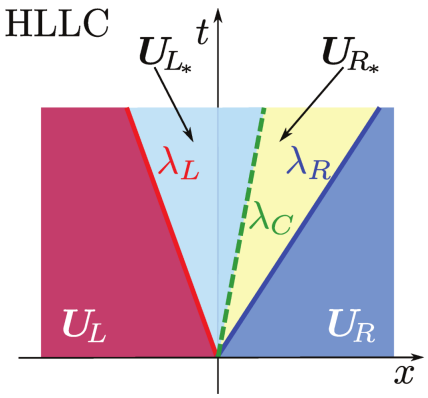

Figure 4: Spacetime control volume for the computation of approximate HLLC flux. This riemann fan is coposed of two waves with speeds λ_L and λ_R and an approximation to the contact discontinuity wave speed λ_C, [3]

5 Eigenvalues

In order to implement the FHLL flux, it was necessary to compute the minimal and maximal eigenvalues of the jacobians of the left and right states.

5.1 Nonrelativistic Case

In the nonrelativistic case, we can compute the minimal and maximumal eigenvalues of the left and right states in order to compute the numerical flux:

$$\alpha^{\pm} = \max(0, \pm\lambda^{\pm}(\boldsymbol{U}^L), \pm\lambda^{\pm}(\boldsymbol{U}^R))$$

For the nonrelativistic case, it can be shown that the minimal and maximal eigenvalues are given by:

$$\lambda^{\pm} = v \pm c_s$$

Where c_s is the nonrelativistic sound speed.

5.2 Relativistic Case

For the relativistic case in one dimension, the minimal and maximal eigenvalues correspond to their relativistic counterpart via the einstein summation formula:

$$\lambda^{\pm} = \frac{v \pm c_s}{1 \pm v c_s}$$

In addition, it was necessary for me to use the relativistic version of sound speed as given by:

$$c_s = \sqrt{\frac{(\Gamma - 1)}{h}\left(\epsilon + \frac{p}{\rho}\right)}$$

Where h is the relativistic specific enthalpy, and ϵ is the specific internal energy.

6 Ghost cells

In order to reconstruct the states to the left and right of the first and last interface, I needed to use imaginary ghost cells in which I fill the values of the conserved quantities as specified by the boundary conditions each iteration. I then could use these values of the ghost cells in order to compute the left and right states during reconstruction. For first order in space, I only needed one ghost cell, whereas for second order in space, when I did the TVD minmod reconstruction, I needed two ghost cells on either side.

6.1 First order in Space

For first order in space, we simple use the left and right states on either side of each boundary to compute the numerical flux across the boundary. Two first order, we do a simple godunov scheme without any reconstruction, and we just use the time averaged values across each of the left and right states as the left and right states for the local riemann problem. When this is the case, we only need one ghost cell on either side of the leftmost and rightmost boundary.

6.2 Second Order in Space

For second order in space, we do a minmod reconstruction of the values of each of the conerved variables or primitive variables on either side of each of the interfaces. Using the generalized minmod scheme, to reconstruct the values of the left and right states for the local riemann problem at each boundary, we need two states on either side of each boundary in order to reconstruct the states at the boundary. For this reason we need two ghost cells on either side to reconstruct the values of the state variables close to the boundaries.

7 Retrieving the Primitives from the Conserved Variables

In order to compute the physical and numerical fluxes, one needs to retrieve the primitive variables from the conserved variables at least once per timestep. For the nonrelativistic case, the prescription for computing the primitive variables comes directly from a manipulation of the euler equations.

7.1 Nonrelativistic Case

The equations for one dimensional hydrodynamics are given by:

$$\frac{\partial \boldsymbol{U}}{\partial t} + \frac{\partial \boldsymbol{F}}{\partial x} = 0$$

Where the conserved variables are:

$$\boldsymbol{U} = \begin{bmatrix} \rho \\ \rho v \\ E \end{bmatrix}$$

and the fluxes are given by:

$$\boldsymbol{F} = \begin{bmatrix} \rho v \\ \rho v^2 + P \\ (E + P)v \end{bmatrix}$$

The first primitive ρ is given by the first component of the conserved vector. The second primitive, v, is given by the second component of the conserved vector, divided by the first component of the conserved vector. The third primitive, the pressure can be obtained from the conserved variables once we introduce an equation of state in order to close the system of equations. If we assume an ideal gas equation of state:

$$p(\rho, e) = (\Gamma - 1)\rho e$$

Using the expression for the total energy density:

$$E = \rho e + \frac{1}{2}\rho v^2$$

We can use the equation of state to eliminate the specific internal energy (e). Doing this yields the expression:

$$p = \left(E - \frac{1}{2}\rho v^2\right)(\Gamma - 1)$$

which can be obtained by using the components of the conserved vector.

7.2 Relativistic Case

In the relativistic case, the conserved variables become heavily coupled via the lorentz factor as we consider velocities a substantial fraction of the speed of light. For the relativistic case, the conserved variables are given by:

$$U = \begin{bmatrix} D \\ S_j \\ \tau \end{bmatrix} = \begin{bmatrix} \rho W \\ \rho h W^2 v_j \\ D(hW - 1) - p \end{bmatrix}$$

However, trying to retrieve the primitive variables from those expressions for the conserved variables cannot be done analytically. So, we need to use a root-finding algorithm in order to retrieve the pressure every timestep. Using the pressure and the conserved variables, we can then retrieve the rest of the primitive variables.
In my code, I implemented a newton-raphson root-finding algorithm to solve for the optimal value of the pressure for each cell using the equation of state and an implicit expression for pressure of the state. The implicit expression for the pressure is of the form:

$$p - \overline{p}[\rho(\boldsymbol{U}, p), e(\boldsymbol{U}, p)] = 0$$

Where $\overline{p}$ is given by the equation of state. My expressions in order to calculate the value of the equation of state, evaluated at my current guess for the pressure, and the current values of the conserved variables are given by:

$$v^* = \frac{S_j}{\tau + p + D}$$

$$W^* = \frac{1}{\sqrt{1-(v^*)^2}}$$

$$\rho^* = \frac{D}{W^*}$$

$$\epsilon^* = \frac{\tau + D(1-W^*) + (1-(W^*)^2)p}{DW^*}$$

The implicit expression for the pressure that I used in my code is:

$$(\Gamma - 1)\rho^*\epsilon^* - p = 0$$

For the newton-raphson algorithm, I need to supply the derivative of the function for which I am trying to find the root of. The derivative of the implicit function with respect to $\overline{p}$ can be approximated using the expression [2]:

$$\frac{df}{dp} = (v^*)^2(cs^*)^2 - 1$$

Where the sound speed here is the relativistic one, and is evaluated using our current guess for the pressure and the current value of the conserved variables. This approximation has the property that it tends to the exact derivative as the solution is approached.
One issue that I encountered with this was that sometimes I would get non-physical results from the root-finding algorithm. In order to correct this, I checked for whenever there was a velocity greater than 1, or a pressure that was less than the theoretical minimum ($P_{min} = |S - \tau - D|$[1]), and corrected it by replacing the value with the minimum value for the pressure when the pressure went below, or a value for the velocity slightly below 1 whenever the velocity went above 1. However, I found that the problem was largely averted by retrieving the primitives before reconstructing the left and right conservative states, and then reconstructing the primitives directly. This method helped significantly for avoiding negative pressures, as opposed to reconstructing the conserved variables close to the boundary, and then retrieving the primitive variables from the reconstructed conserved variables.

8 Modularity

I took great lengths to improve the modular structure of my code. As a result of this, it is possible for me in the future to implement more versions of approximate riemann solvers, more varied test cases, as well as several dimensions. This provides the structure in order to perform a more inclusive comparison between different combinations of higher order methods in different scenarios.

9 Future Goals

In the future, I hope to be able to increase the dimensionality of my code to 2D in order to investigate phenomena that arise once one goes to higher dimension. In addition, I would like to implement other approximate riemann solvers, as well as different reconstruction methods in order to compare how well they represent the solution for different situations.

10 Test Cases

I have tested my code against several different test cases whose results have already been shown in other sources [4].

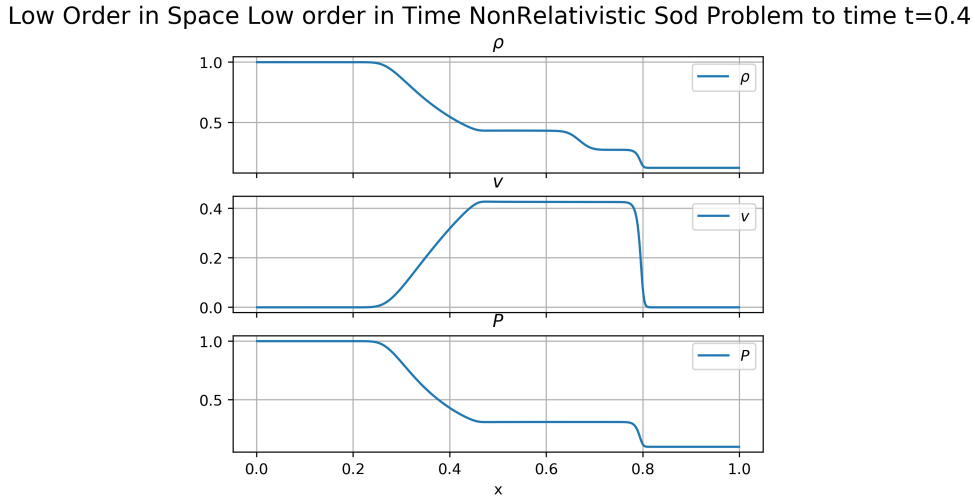

Figure 5: Nonrelativistic test case:$\rho_L = 1$,$v_L = 0$,$p_L = 1$,$\rho_R = 0.125$,$v_R = 0$,$p_R = 0.1$,$\gamma = 1.4$

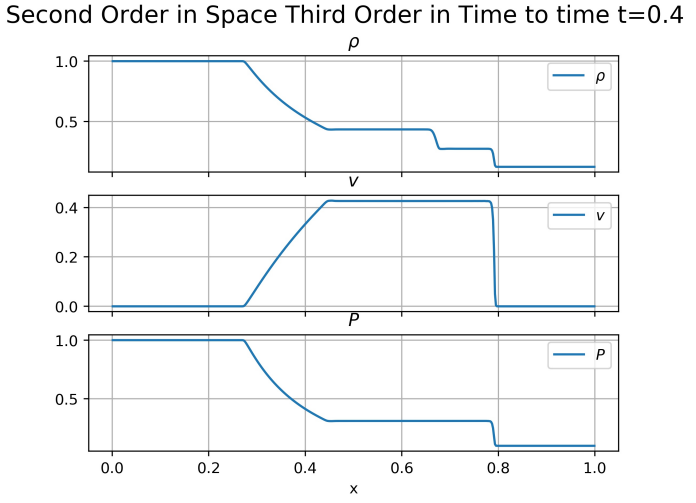

Figure 6: Nonrelativistic test case:$\rho_L = 1$,$v_L = 0$,$p_L = 1$,$\rho_R = 0.125$,$v_R = 0$,$p_R = 0.1$,$\gamma = 1.4$

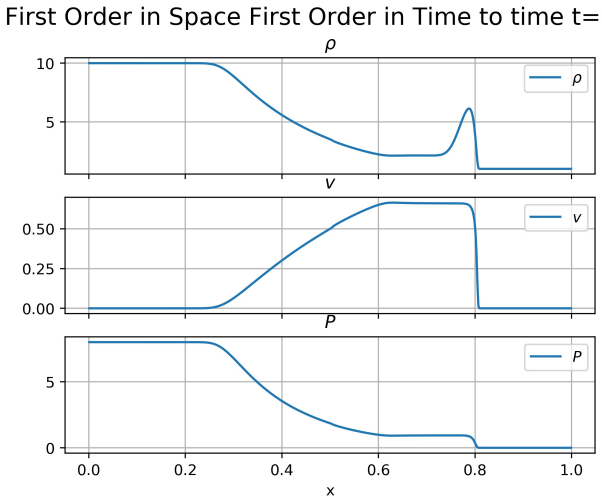

Figure 7: Relativistic Sod Problem 1:$\rho_L = 10$,$v_L = 0$,$p_L = 13.33$,$\rho_R = 1.0$,$v_R = 0$,$p_R = 1e-8$,$\gamma = 5/3$

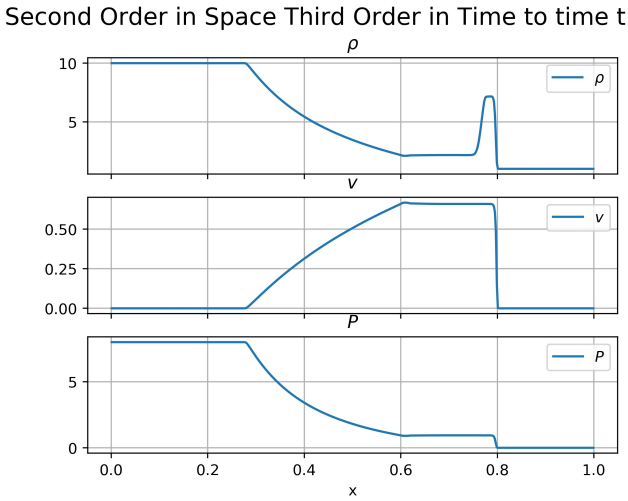

Figure 8: Relativistic Sod Problem 1:$\rho_L = 10$,$v_L = 0$,$p_L = 13.33$,$\rho_R = 1.0$,$v_R = 0$,$p_R = 1e-8$,$\gamma = 5/3$

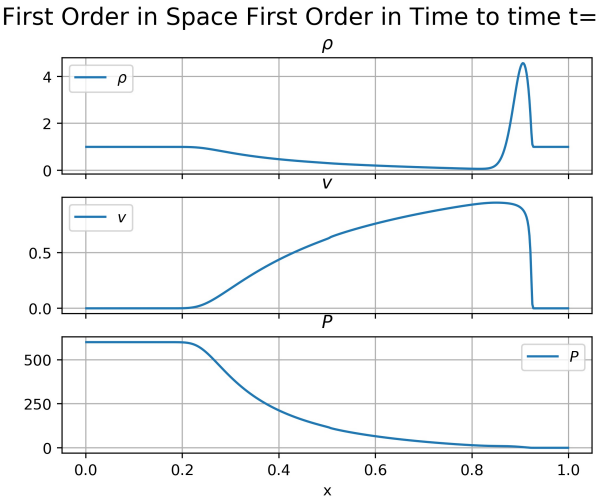

Figure 9: Relativistic Sod Problem 2:$\rho_L = 1$,$v_L = 0$,$p_L = 1000$,$\rho_R = 1.0$,$v_R = 0.0$,$p_R = 1e-2$,$\gamma = 5/3$

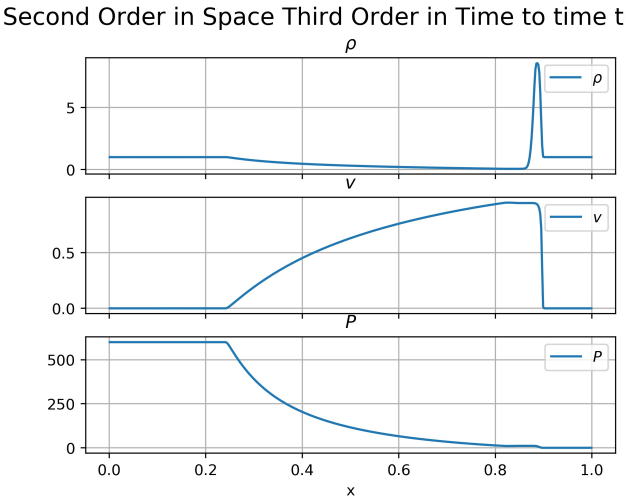

Figure 10: Relativistic Sod Problem 2:$\rho_L = 1$,$v_L = 0$,$p_L = 1000$,$\rho_R = 1.0$,$v_R = 0.0$,$p_R = 1e-2$,$\gamma = 5/3$

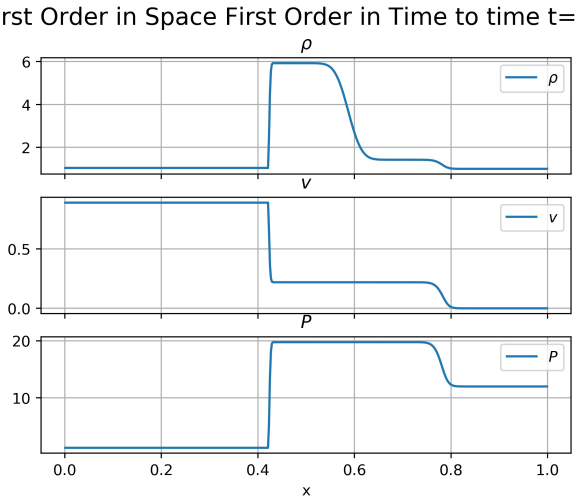

Figure 11: Relativistic Sod Problem 3:$\rho_L = 1$,$v_L = 0.9$,$p_L = 1$,$\rho_R = 1.0$,$v_R = 0$,$p_R = 10.0$,$\gamma = 4/3$

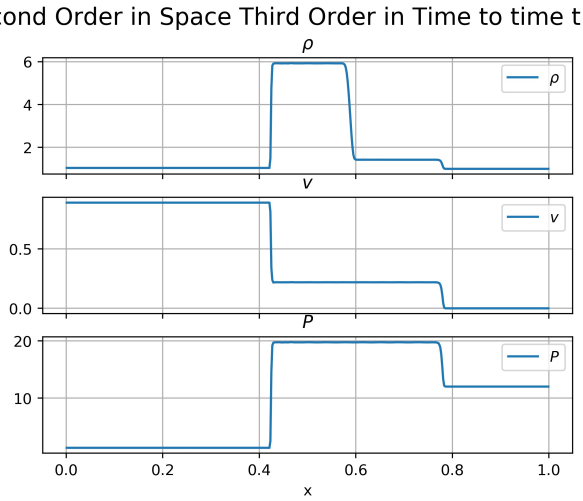

Figure 12: Relativistic Sod Problem 3:$\rho_L = 1$,$v_L = 0.9$,$p_L = 1$,$\rho_R = 1.0$,$v_R = 0$,$p_R = 10.0$,$\gamma = 4/3$

References

[1] Orhan Donmez. "Solving 1-D special relativistic hydrodynamics (SRH) equations using different numerical methods and results from different test problems". In: *Applied Mathematics and Computation* 181.1 (). DOI: https://doi.org/10.1016/j.amc.2006.01.031.

[2] Ewald Muller Jose Marti. "Numerical Hydrodynamics in Special Relativity". In: *Living Reviews in Relativity* 6.7 (), pp. 73–74. DOI: https://doi.org/10.12942/lrr-2003-7.

[3] Olindo Zanotti Luciano Rezzolla. *Relativistic Hydrodynamics*. Oxford University Press, 2013.

[4] Andrew Macfadyen Weiqun Zhang. "RAM: A Relativistic Adaptive Mesh Refinement Hydrodynamics Code". In: *The Astrophysical Journal Supplement Series* 215.1 (), pp. 5–8. DOI: arXiv:astro-ph/0505481.