Final Project: 2D Hydro Code with HLLC Riemann Solver

Austin McDowell

Introduction

For my final project I wrote a two-dimensional hydrodynamics code with a Harten-Lax-van Leer Contact (HLLC) Riemann solver that is third order in time and second order in space. A two-dimensional code allows for a wider variety of test problems and is, of course, a better represenation of actual phenomena and the HLLC Riemann solver is essential for higher dimensionality.

I will first describe the details of the HLLC Riemann solver and include a test to prove that it is functional. Then, I will discuss the changes needed to make the code 2D and ways I could improve the code in the future. Next, I will discuss the test problems that I ran.

HLLC Solver

The 2D conservation laws are a straightforward generalization of the 1D equations we saw in homework 3

$$\frac{\partial \mathbf{U}}{\partial t} + \frac{\partial \mathbf{F}}{\partial x} + \frac{\partial \mathbf{G}}{\partial y} = 0 \tag{1}$$

where $\mathbf{U}=(\rho, \rho v_x, \rho v_y, E)$, $\mathbf{F}=(\rho v_x, \rho v_x^2 + P, \rho v_x v_y, (E+P)v_x)$ is the flux in the x-direction, and $\mathbf{G}=(\rho v_y, \rho v_x v_y, \rho v_y^2 + P, (E+P)v_y)$ is the flux in the y-direction. In this case the energy density is given by $E = \rho e + \frac{1}{2}\rho(v_x^2 + v_y^2)$ and the evolution of $\mathbf{U}$ is given by

$$L(\mathbf{U}) = \frac{d\mathbf{U}_{i,j}}{dt} = -\frac{\mathbf{F}_{i+1/2,j} - \mathbf{F}_{i-1/2,j}}{\Delta x} - \frac{\mathbf{G}_{i,j+1/2} - \mathbf{G}_{i,j-1/2}}{\Delta} y \tag{2}$$

where $U_{i,j}$ is the value of the conserved quantities in zone (i,j) and $\mathbf{F}_{i+1/2,j}$, $\mathbf{F}_{i-1/2,j}$, $\mathbf{G}_{i,j+1/2}$, and $\mathbf{G}_{i,j-1/2}$, are the fluxes at the zone boundaries. Then we can express the time evolution of $\mathbf{U}$ using the forward Euler method

$$\mathbf{U}^{n+1} = \mathbf{U}^n + L(\mathbf{U})\Delta t \tag{3}$$

where n denotes the timestep number.

We must then solve the Riemann problem to find the values for $\mathbf{F}_{i+1/2,j}$, $\mathbf{F}_{i-1/2,j}$, $\mathbf{G}_{i,j+1/2}$, and $\mathbf{G}_{i,j-1/2}$. Previously, this was done using the HLL approximate Riemann solver, which assumed a constant state for the central, starred region and used continuity of flux across the characteristic lines to find the flux in the starred region (as shown in Figure (1)).

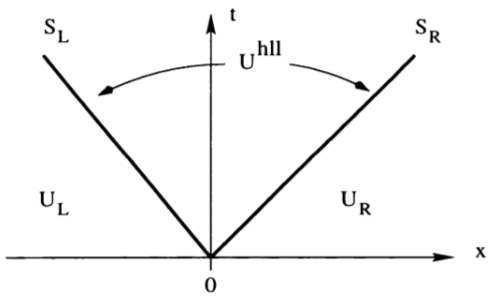

Figure 1: Diagram depicting HLL treatment of the Riemann problem. The central region U^{hll} is assumed to be constant, and flux is found using the quantities in the left and right states (U_L, U_R) and the characteristic wave speeds (S_L, S_R).

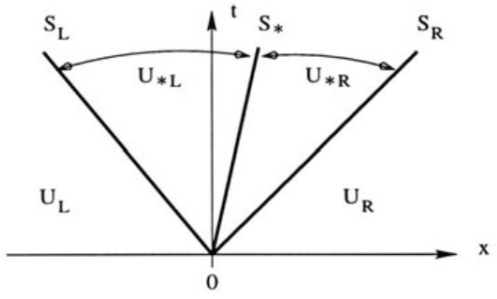

Figure 2: Diagram depicting HLLC treatment of the Riemann problem. The two central regions (U_{*L}, U_{*R}) are found using the quantities in the left and right states (U_L, U_R) and the characteristic wave speeds (S_L, S_R, S_*).

The HLLC approximate Riemann solver divides the central region into two states U_{*L} and U_{*R} which are separated by a characteristic with wave speed S_* as shown in Figure (2). The condition of continuity across each characteristic can be written as

$$\mathbf{F}_{*L} - S_L \mathbf{U}_{*L} = \mathbf{F}_L - S_L \mathbf{U}_L \tag{4}$$

$$\mathbf{F}_{*R} - S_* \mathbf{U}_{*R} = \mathbf{F}_{*L} - S_* \mathbf{U}_{*L} \tag{5}$$

$$\mathbf{F}_{*R} - S_R \mathbf{U}_{*R} = \mathbf{F}_R - S_R \mathbf{U}_R \tag{6}$$

We then assume

$$\begin{aligned} v_{x,*L} = v_{x,*R} &= S_* \\ P_{*L} = P_{*R} &= P_* \\ v_{y,*L} = v_{y,L}, \quad v_{y,*R} &= v_{y,R} \end{aligned} \tag{7}$$

which, in combination with the continuity conditions, allows us to write

$$S_L \mathbf{U}_{*L} - \mathbf{F}_{*L} = S_L \mathbf{U}_L - \mathbf{F}_L \tag{8}$$

$$S_R \mathbf{U}_{*R} - \mathbf{F}_{*R} = S_R \mathbf{U}_R - \mathbf{F}_R \tag{9}$$

where the right hand side of each equation is known. We can then solve for $\mathbf{U}_{*R}$ and $\mathbf{U}_{*L}$

$$\mathbf{U}_{*K} = \rho_K \left(\frac{S_K - u_K}{S_K - u_*} \right) \begin{bmatrix} 1 \\ S_* \\ v_{y,K} \\ \frac{E_K}{\rho_K} + (S_* - u_K) \left[S_* + \frac{P_K}{\rho_K (S_K - u_K)} \right] \end{bmatrix} \tag{10}$$

where K stands for either L or R. We can then use equations (4) and (6) to solve for $\mathbf{F}_{*L}$ and $\mathbf{F}_{*R}$, respectively. However, we still need to determine the values for S_L, S_R, and S_*. For S_L and S_R we use

$$S_L = v_{x,L} - c_{s,L} \quad ; \quad S_R = v_{x,R} + c_{s,R} \tag{11}$$

where $c_{s,K}$ is the sounds speed on either the right or left side. From our condition that $P_{*L} = P_{*R} = P_*$ we can find the equation for S_*.

$$S_* = \frac{P_R - P_L + \rho_L v_{x,L}(S_L - v_{x,L}) - \rho_R v_{x,R}(S_R - v_{x,R})}{\rho_L (S_L - v_{x,L}) - \rho_R (S_R - v_{x,R})} \tag{12}$$

The advantage of the HLLC solver is that it preserves contact discontinuities that would be diffused using the HLL method. For example, if there is a discontinuity in density, constant pressure, and zero velocity, the HLL solver will find that ρ_* is equal to the average of the two different densities and that the flux into the starred region is non-zero. For this steady state, we would expect the flux to be exactly zero. Therefore the HLL solver is inherently diffusive.

An example of this is shown below in figures (3) and (4). The initial conditions are

$$\begin{aligned} \rho_L &= 1.4 \\ \rho_R &= 1.0 \\ P_L = P_R &= 1.0 \\ v_L = v_R &= 0 \end{aligned} \tag{13}$$

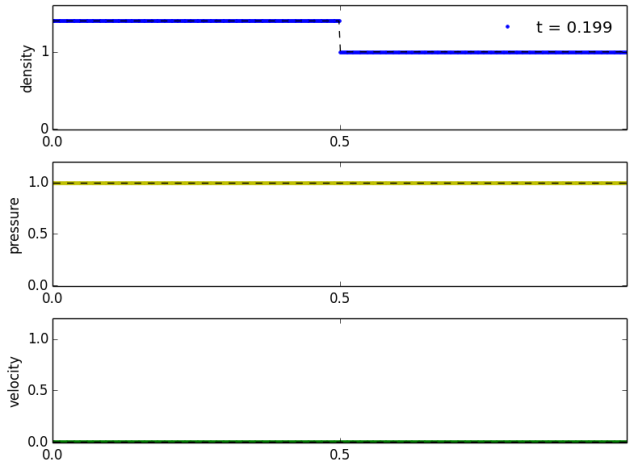

Figure 3: Steady state solution at $t \sim 0.2$ with the HLLC Riemann solver. The discontinuity in density is perfectly maintained.

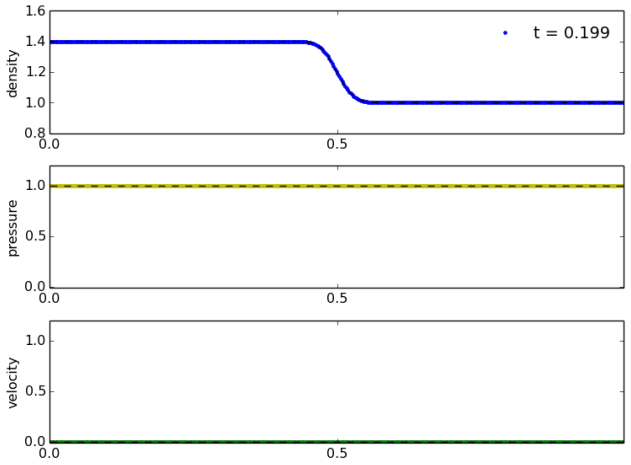

Figure 4: "Steady" solution at $t \sim 0.2$ with the HLL Riemann solver. At late times we see that the discontinuity has been smoothed due to numerical diffusion.

We can see that at $t \sim 0.2$ the initial discontinuity has been diffused by the HLL solver, but is perfectly preserved with the HLLC solver. This preservation is crucial in 2D test problems where contact discontinuities are important.

2D Adjustments

As mentioned above, upgrading to 2D adds ρv_y to $\mathbf{U}$, $\rho v_x v_y$ to $\mathbf{F}$, and introduces the flux in the y direction, $\mathbf{G}$. However, when solving the Riemann problem for $\mathbf{G}_*$, there are a few changes to the solutions presented above. First, the continuity of the primitive variables becomes

$$\begin{aligned} v_{y,*L} = v_{y,*R} &= S_* \\ P_{*L} = P_{*R} &= P_* \\ v_{x,*L} = v_{x,L}, \quad v_{x,*R} &= v_{x,R} \end{aligned} \tag{14}$$

Next, the solution for $\mathbf{U}_{*K}$ changes to

$$\mathbf{U}_{*K} = \rho_K \left(\frac{S_K - u_K}{S_K - u_*} \right) \begin{bmatrix} 1 \\ v_{x,K} \\ S_* \\ \frac{E_K}{\rho_K} + (S_* - u_K) \left[S_* + \frac{P_K}{\rho_K (S_K - u_K)} \right] \end{bmatrix} \tag{15}$$

and S_L and S_R become

$$S_L = v_{y,L} - c_{s,L} \quad ; \quad S_R = v_{y,R} + c_{s,R} \tag{16}$$

. Finally, the equation for S_* becomes

$$S_* = \frac{P_R - P_L + \rho_L v_{y,L}(S_L - v_{y,L}) - \rho_R v_{y,R}(S_R - v_{y,R})}{\rho_L (S_L - v_{y,L}) - \rho_R (S_R - v_{y,R})} \tag{17}$$

With these adjustments we can solve for $\mathbf{G}_*$ in the same way we solved for $\mathbf{F}_*$ above.

Additionally, to make the code higher order in space we implement the piecewise linear method (PLM) described in homework 3. However, now PLM must be applied in both the x and y directions in order to interpolate the values on each zone face.

Improvements

I will also list a few improvements to the code that I would like to make in the future.

Parallelization

The first major improvement would be to parallelize the code. Since the evolution of each zone only depends on its neighbors, the problem is well suited to parallelization. Additionally, parallelization would allow for faster computation of larger grids. To implement this, I would use the Message Passing Interface (MPI) MPI4Py and divide the entire computational grid into even segments for each processor to handle. Each grid chunk, however, would need two additional ghost zones on the edges. This way, the values of those zones could be updated and passed between processors to ensure that each chunk evolves correctly.

Moving Mesh

I would also like to update to a moving computational grid in order to improve the code's accuracy in capturing contact discontinuities. The first change in the code would be in the flux in each starred region of the HLLC solver. In the moving mesh case, we must also consider the 'amount' of conserved quantities advected with the movement of the mesh. To do so the flux would change

$$\mathbf{F}_* \rightarrow \mathbf{F}_* - (\vec{w} \cdot \hat{n}) \mathbf{U}_* \tag{18}$$

where $\vec{w}$ is the zone face's velocity and $\hat{n}$ is the zone face's normal vector. With a stationary mesh the two starred states are separated by the characteristic $x = t = 0$, however, with a moving mesh this characteristic becomes $x/t = \vec{w} \cdot \hat{n}$. Additionally, the HLLC Riemann solver is able to accurately calculate advective flux and numerical error is minimized when using the HLLC solver with the moving mesh technique.

Results

In order to test the code, I ran two test problems: the spherical blast wave and the Kelvin-Helmholtz instability. The results of each test are summarized below.

Blast Wave

The first test problem I performed was the spherical blast wave. This system is evolved on a uniform grid ranging from $x = [-0.5, 0.5]$, $y = [-0.5, 0.5]$ and with $\gamma = 5/3$. The initial conditions are

$$\begin{aligned} v_x = v_y &= 0 \\ \rho &= 1.0 \\ P &= 0.1 \end{aligned} \tag{19}$$

but then we set the pressure within a circle of radius 0.1 equal to 10. This high pressure difference causes a central explosion as shown below in figures (5), (6), and (7).

We can check the accuracy of our solution by comparing it to the known, self-similar Sedov-Taylor solution to a spherically expanding blast wave. It has been shown that the radius of the blast scales as

$$r \sim t^{2/5} \tag{20}$$

which, with the knowledge of an initial radius r_0 at t_0 allows us to write

$$r = r_0 \left(\frac{t}{t_0} \right)^{2/5} \tag{21}$$

In figure (5) the black circle denotes the shock position given by the Sedov-Taylor solution (with $r_0 = 0.1$). We see that the predicted solution expands more quickly than our calculated solution. This may be due to some error with how v_x and v_y are calculated. However, we are still able to obtain a blast that expands in a seemingly symmetric manner.

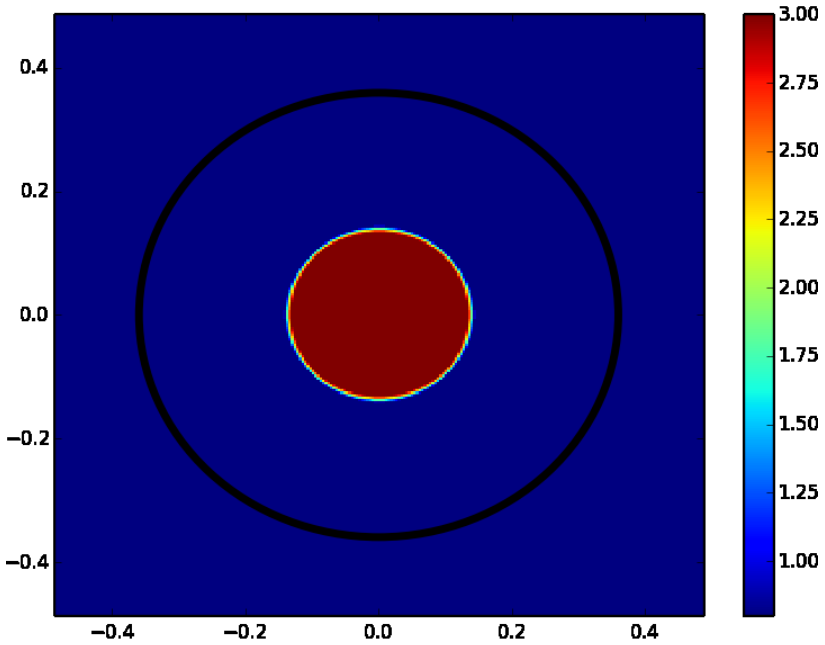

Figure 5: Pressure plot of isentropic blast wave around $t \sim 0.015$. Solid black line shows position of shock front given by the self-similar Sedov Taylor solution

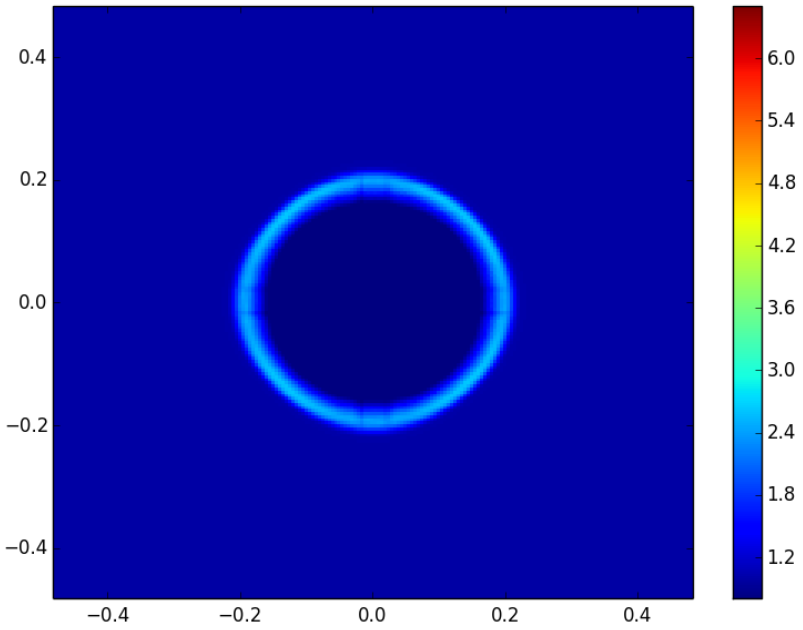

Figure 6: Density plot of isentropic blast wave around $t \sim 0.048$

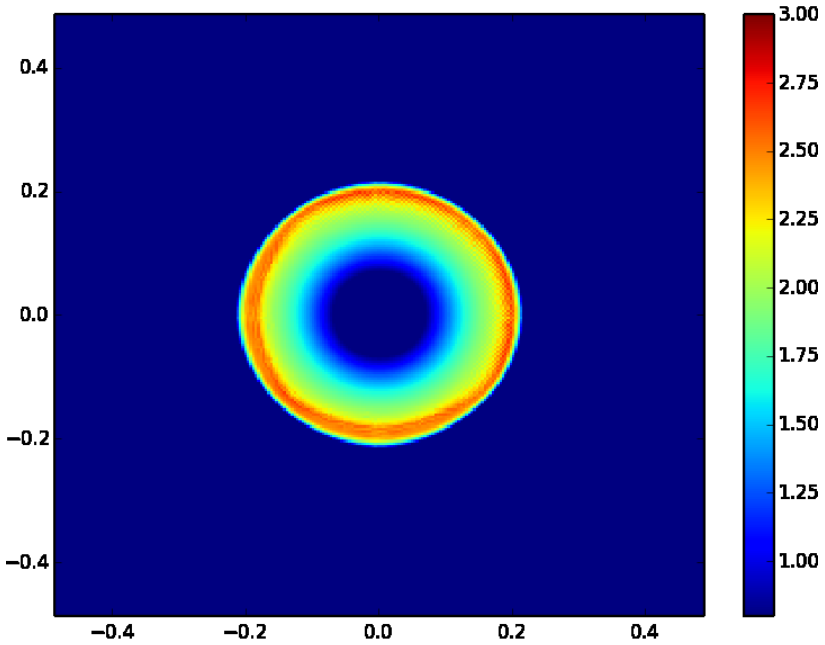

Figure 7: Pressure plot of isentropic blast wave around $t \sim 0.048$

With more time I would have liked to implement periodic boundary conditions so I could observe the interaction of the reflected explosion. I would also have liked to figure out exactly why the Sedov-Taylor solution does not match up with the calculated solution.

Kelvin-Helmholtz Instability

The next test I ran was the Kelvin-Helmholtz instability. This instability occurs when two fluids of different density shear past one another. This problem was calculated on a uniform grid from $x = [0, 1]$ and $y = [0, 1]$ with $\gamma = 5/3$. The pressure everywhere is set to $P = 2.5$, the density and v_x in the central region between $0.25 < y < 0.75$ are set to 2.0 and 0.5, respectively. Elsewhere, ρ and v_x are set to 1.0 and -0.5, respectively. v_y is given by

$$v_y = f(y) * g(x) \tag{22}$$

where

$$g(x) = \omega_0 \sin(4\pi x) \tag{23}$$

$$f(y) = e^{-\frac{(y-0.25)^2}{2\sigma^2}} + e^{-\frac{(y-0.75)^2}{2\sigma^2}} \tag{24}$$

, $w_0 = 0.1$, and $\sigma = 0.05/\sqrt{2}$. These initial conditions are chosen to amplify modes in the velocity that drive the instability. A plot of density after ~ 1562 timesteps is shown below in figure (8). We can see that the shearing has produced spirals at the interface between the two fluids. These spirals should be symmetric on the top and bottom, however, we do not see this symmetry here. This is likely due to an error in v_x and/or v_y. It's possible that the two velocity components have been swapped at some point in the code, but the author is unable to find where this may have happened.

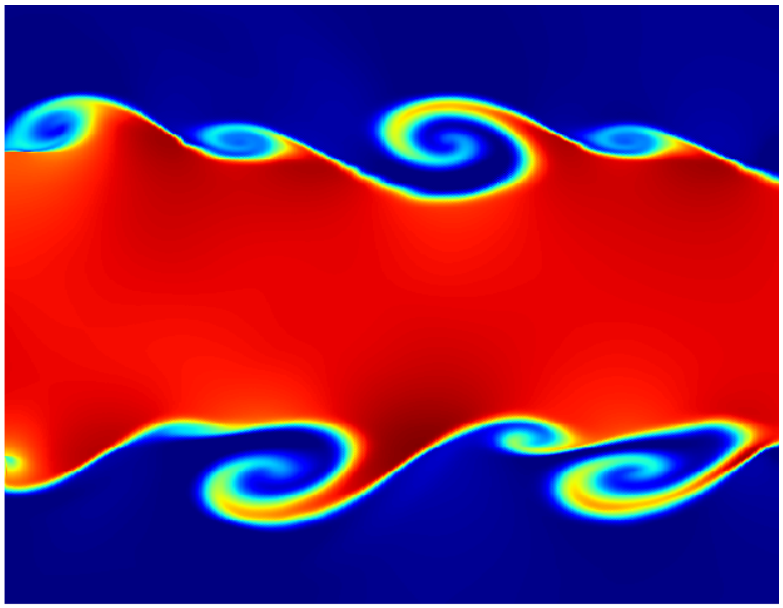

Figure 8: Density plot of Kelvin-Helmholtz instability after ~ 1562 timesteps

The plot below shows the advection of a passive scalar, q, set in the initial conditions. The evolution of q is given by

$$\frac{\partial(\rho q)}{\partial t} + \frac{\partial(\rho q v_x)}{\partial x} = 0 \quad ; \quad \frac{\partial(\rho q)}{\partial t} + \frac{\partial(\rho q v_y)}{\partial y} = 0 \tag{25}$$

and the flux found by the HLLC solver is

$$(\rho q)_{*K} = \rho_K \left(\frac{S_K - v_{i,K}}{S_K - S_*} \right) q_K \tag{26}$$

where K again refers to L or R and v_i is either v_x or v_y.

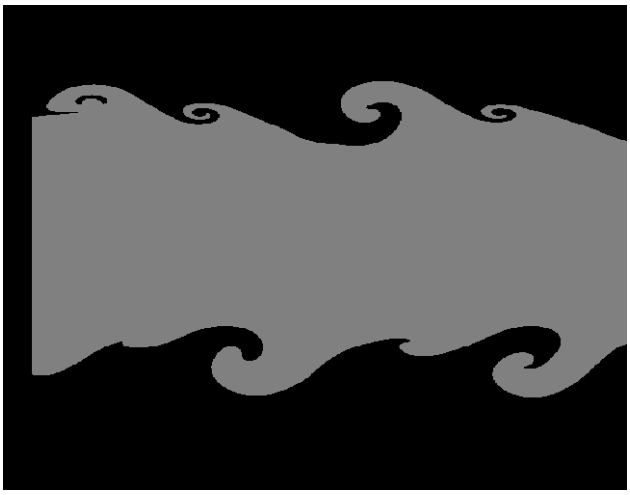

Figure 9: Passive scalar advection in the Kelvin-Helmholtz instability after ~ 1215 timesteps

The passive scalar does not affect the evolution of the system and is evolved only to observe the mixing that occurs between the two fluids.

Going forward I would like to fix the issue that causes the asymmetry in the instability. I would also like to compare the growth rate of the spirals to the analytic solution found by Bodo et al. 2004.

References

"Riemann Solvers and Numerical Methods for Fluid Dynamics". F. Toro, Eleuterio
"The Athena Code Test Page". Stone, Jim. www.astro.princeton.edu/∼jstone/Athena/tests
"Thesis". Duffell, Paul. duffell.org/media/thesis.pdf