The Astrophysical Journal Supplement Series, 216:35 (14pp), 2015 February doi:10.1088/0067-0049/216/2/35

RICH: OPEN-SOURCE HYDRODYNAMIC SIMULATION ON A MOVING VORONOI MESH

Almog Yalinewich[1], Elad Steinberg[1], and Re'em Sari[1,2]
[1] Racah Institute of Physics, The Hebrew University, 91904, Jerusalem, Israel
[2] California Institute of Technology, MC 130-33, Pasadena, CA 91125, USA
Received 2014 September 24; accepted 2015 January 12; published 2015 February 10

ABSTRACT

We present here RICH, a state-of-the-art two-dimensional hydrodynamic code based on Godunov's method, on an unstructured moving mesh (the acronym stands for Racah Institute Computational Hydrodynamics). This code is largely based on the code AREPO. It differs from AREPO in the interpolation and time-advancement schemeS as well as a novel parallelization scheme based on Voronoi tessellation. Using our code, we study the pros and cons of a moving mesh (in comparison to a static mesh). We also compare its accuracy to other codes. Specifically, we show that our implementation of external sources and time-advancement scheme is more accurate and robust than is AREPO when the mesh is allowed to move. We performed a parameter study of the cell rounding mechanism (Lloyd iterations) and its effects. We find that in most cases a moving mesh gives better results than a static mesh, but it is not universally true. In the case where matter moves in one way and a sound wave is traveling in the other way (such that relative to the grid the wave is not moving) a static mesh gives better results than a moving mesh. We perform an analytic analysis for finite difference schemes that reveals that a Lagrangian simulation is better than a Eulerian simulation in the case of a highly supersonic flow. Moreover, we show that Voronoi-based moving mesh schemes suffer from an error, which is resolution independent, due to inconsistencies between the flux calculation and the change in the area of a cell. Our code is publicly available as open source and designed in an object-oriented, user-friendly way that facilitates incorporation of new algorithms and physical processes.

Key words: hydrodynamics – methods: numerical

1. INTRODUCTION

It has long been recognized that the aid of computers can greatly increase our understanding of astrophysical phenomena. Yet even with the progress of computers, solutions to some problems are still limited by computing power. One idea to achieve greater accuracy at a given computer power is using a computational mesh that moves together with the fluid (Lagrangian grid) rather than the more common static mesh (Eulerian grid). While it has not been proven that the former is better, one reason for using a Lagrangian grid is that it automatically gets denser (thus providing higher resolution) in places where matter is flowing in (e.g., behind shock fronts). Because these areas are usually the more interesting parts of the domain, the Lagrangian grids tend to give better resolution in areas of interest.

Recently, a novel method for a semi-Lagrangian Gudonov scheme, called AREPO (Springel 2010), was published. In AREPO, in contrast to arbitrary Lagrangian Eulerian (ALE) simulations, mesh points are no longer required to adhere to their neighbors, so when computational cells drift too far apart, they do not tangle but change neighbors. Another advantage of this scheme is that each flux is calculated in the reference frame of the moving edge, so advections between cells are greatly reduced. AREPO is thus able to reap most of the benefits of ALE with fewer drawbacks.

Despite a comprehensive description of the code and its test suite in Springel (2010), a more thorough comparison between semi-Lagrangian and Eulerian grids is in order. Another matter that requires more work is the coupling of external forces or sources. While the method employed in Springel (2010) to couple gravity is explained in detail, there is no simple way to extend it to arbitrary source terms. We developed our own version of the code, called RICH (which stands for Racah Institute Computational Hydrodynamics), which is written in C++ and takes after AREPO and its relativistic variant TESS (Duffell & MacFadyen 2011), with a few changes. The purpose of this paper is to present our code, compare its accuracy with other codes, do a parameter study of the mesh rounding mechanism (Llyod iterations), and explore the pros and cons of using a semi-Lagrangian grid. Because most of the algorithms were discussed in other papers (Springel 2010; Duffell & MacFadyen 2011), in this paper we will focus on the differences between our code and AREPO and TESS.

The paper is structured as follows. In Section 2 we describe the differences of our code from AREPO and TESS. One-dimensional (1D) and two-dimensional (2D) test problems are presented in Sections 3 and 4, respectively. The effect of non-Lagrangian motion is discussed in Section 6. We show that there is a resolution-independent error that arises from an inconsistency between the flux calculation and change in the area of a cell in Section 5. The question of whether a Lagrangian code is always better than a Eulerian code is addressed in Section 7. In Section 8 we summarize and discuss the advantages of using a semi-Lagrangian grid.

2. ALGORITHM MODIFICATIONS

Because our code is very similar to AREPO (Springel 2010) and TESS (Duffell & MacFadyen 2011), we will only dwell on the differences from them.

2.1. Tessellation Creation

We follow AREPO and TESS and construct the Voronoi diagram by first building the Delaunay triangulation (its dual graph), and then we translate the triangulation to the Voronoi diagram in linear time. We create the Delaunay triangulation using the point insertion method (Ledoux 2007; Springel 2010). This method adds the mesh generating points one after another and checks each time whether it falls inside a circumscribing circle of an existing triangle. One difficulty with this stage occurs when a point lies exactly on the circle because numerical round-off errors can change the result. We use a library developed by Shewchuk (1996), which employs adaptive floating-point arithmetic whenever the round-off error in the calculation may change the sign of the answer. This method was tested on a set of points arranged in a square grid (so that all of the triangles are degenerate), and the in-circle test time was about twice that of normal arithmetic. When the triangulation was tested with a random set of points, only a 10% increase in the triangulation time was observed. Constructing a Voronoi diagram with 10^6 random points takes about 5 s on an i7–2620 M CPU, and a square mesh with the same number of points takes 7 s. For comparison, using the same CPU but using the qhull algorithm with MATLAB 2013a took 14 s.

2.2. *Interpolation*

Higher-order schemes require spatial interpolation of the cell values. AREPO reconstructs the gradient in each cell using the Green–Gauss theorem, and we implement the same method. Specifically,

$$\left\langle \vec{\nabla}\phi \right\rangle_i = \frac{1}{A_i}\sum_{i\neq j} L_{ij}\left(\left[\phi_j - \phi_i\right]\frac{\vec{c}_{ij}}{r_{ij}} - \frac{\phi_i + \phi_j}{2}\frac{\vec{r}_{ij}}{r_{ij}}\right), \tag{1}$$

where ϕ_i is the quantity to reconstruct in the i cell, A_i is the cell area, L_{ij} is the length of the edge between the i and j cells, r_{ij} is the vector connecting the two mesh generating points, and c_{ij} is the vector from the midpoint between the i and j mesh generating points and the center of the edge between the cells. The summation is done among all of the cell's neighbors. Once the gradient is known, the primitive variables at the edges are reconstructed using linear extrapolation:

$$\phi_{ij} = \phi_i + \left\langle \vec{\nabla}\phi \right\rangle_i \cdot \left(\vec{L}_{\rm mid} - \vec{s}_i\right), \tag{2}$$

where $\vec{s}$ is the cell's center of mass, and $\vec{L}_{\rm mid}$ is the middle of the edge.

In order to prevent the creation of new maxima or minima, which can cause oscillations near discontinuities, a slope limiter is used. AREPO's slope limiter prevents the creation of a global maxima/minima in the sense that the extrapolated value cannot exceed the value of the highest neighbor and cannot be below the lowest neighbor. In order to achieve this, the gradient is set to be

$$\left\langle \vec{\nabla}\phi \right\rangle_i' = \alpha_i \left\langle \vec{\nabla}\phi \right\rangle_i, \tag{3}$$

where the slope limiter α_i is set to be

$$\alpha_i = \min\left(1, \psi_{ij}\right) \tag{4}$$

$$\psi_{ij} = \begin{cases} \left(\phi_i^{\max} - \phi_i\right)/\Delta\phi_{ij} & \Delta\phi_{ij} > 0 \\ \left(\phi_i^{\min} - \phi_i\right)/\Delta\phi_{ij} & \Delta\phi_{ij} < 0 \\ 1 & \Delta\phi_{ij} = 0, \end{cases} \tag{5}$$

where $\Delta\phi_{ij}$ is the difference between the interpolated value at the edge and the value at the cell center, and $\phi_i^{\max}$ and $\phi_i^{\min}$ are the maximum and minimum values among the neighbors,

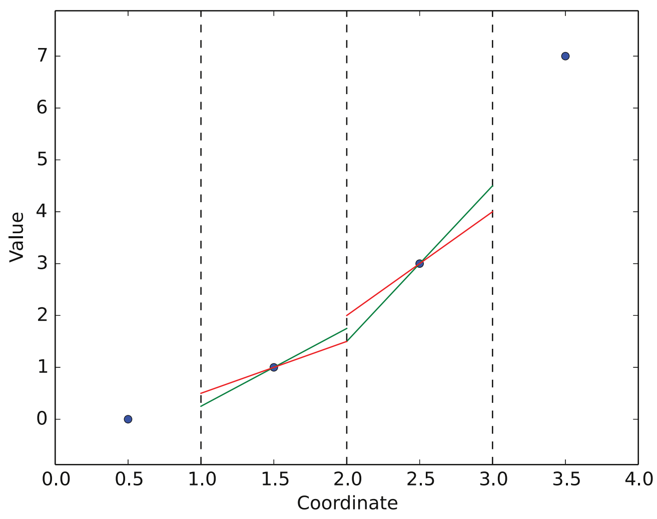

Figure 1. Example for TVD violation in AREPO's interpolation. Blue points represent the cell values, the dashed black line the cell boundaries, green lines the interpolations according to AREPO, and red lines the TVD interpolation. One can see that AREPO's method creates new local extrema, whereas the TVD method does not.

respectively. This slope limiting is not total variation diminishing (TVD; Toro 1999) because under these constraints, the gradient of interpolated values can have a different sign from the gradient of the two neighboring mesh points. To demonstrate this problem, let us assume a uniform 1D grid with four equal-sized computation cells, whose values are 0, 1, 3, and 7. Applying the method described above, we get that although the value of cell #3 is greater than cell #2, the interpolated value of cell #2 is greater than that of cell #3, as can be seen in Figure 1.

A possible remedy for this problem is presented in TESS (Duffell & MacFadyen 2011), where they employ a more restrictive "local" slope limiter, in which the extrapolated value cannot exceed any of its neighbors by some numerical factor θ:

$$\psi_{ij} = \begin{cases} \max\left(\theta\left(\phi_j - \phi_i\right)/\Delta\phi_{ij}, 0\right) & \Delta\phi_{ij} \neq 0 \\ 1 & \Delta\phi_{ij} = 0 \end{cases}. \tag{6}$$

Choosing $\theta \leqslant 0.5$ prevents TVD violations. The trade-off is that the lower θ is, the more diffusive the scheme becomes.

By default we use AREPO's scheme, unless we suspect that there is a shock front in the neighborhood of cells, in which case we use TESS' scheme instead. The quantitative criterion is when either of these two conditions is true:

$$-\nabla\mathbf{v} > \delta_v \frac{c_{s_i}}{R_i} \tag{7}$$

$$\delta p > \min\begin{cases} P_i/P_j & P_j > P_i \\ P_j/P_i & P_i > P_j, \end{cases} \tag{8}$$

where ∇ denotes the divergence operator. The default numerical factors we chose are $\theta = 0.5$, $\delta_v = 0.2$, and $\delta_p = 0.7$. In the limit $\theta \rightarrow 0$ the interpolation reduces to a first order. Hence, lowering the value of θ spoils the second-order interpolation. We note, however, that Equation 6 is used only in the vicinity of shock fronts.

2.3. Time Advancement

In order to achieve second-order accuracy, AREPO uses linear interpolation to determine the hydrodynamic variables near the edge of a cell at the beginning of a time step. By substituting the spatial derivatives into the hydrodynamic equations, it gets the time derivative of the primitive variables on the edge between cells and uses them to estimate the values at the edge at half a time step. We call this method of time advancement "extrapolated fluxes." Then, the "time-centered" primitive variables are given to the Riemann solver in order to compute second-order-accurate fluxes. Adding these fluxes to the conserved variables from the beginning of the time step yields second-order accuracy and only invokes the Riemann solver and interpolations once. The downside is that external sources have to be written in a special way so they will be second-order accurate in time as well. For instance, gravity has to use the variables from before the time step and after the time step in order to be second-order accurate. Another example is the viscosity module (Munoz 2013), where an elaborate scheme was required to attain second-order accuracy.

We use a different approach for our time integration. Following TESS, we use a "time-centered fluxes" scheme. The system is advanced by half a time step (using linear spatial interpolation), the mesh is rebuilt, and the half-time-step primitive variables are computed. Then, the time-centered fluxes are computed and are added to the conserved variables from the beginning of the time step with a full time step. The final mesh is built from advancing the mesh points a full time step, from their position at the beginning of the time step with a velocity that was calculated during the half time step. This "time-centered fluxes" scheme ensures that our time advancement is second-order accurate.

The "time-centered fluxes" scheme has the added benefit that external sources have to be only first-order accurate in time, and the time integration will automatically make them second-order accurate. The Courant stability condition also allows us to use a Courant number larger than unity, though we usually use the default arbitrary value of 0.3. The reason it is so low is to prevent the simulation from crashing for other reasons, like the depletion of energy from a cell because of a strong rarefaction wave.

The downside to our scheme is that it requires twice the computation time because it requires building the mesh and calculating the fluxes twice. In many applications, the robustness of the external forces implementation is worth the slower execution time.

2.4. HLLC Riemann Solver

The Riemann solver used in AREPO is exact. This means that calculating the flux on every edge involves a numerical solution of a single-variable equation (Toro 1999). The downside of using this solver is that it is time consuming and that it is generally not applicable to all equations of state. To avoid these difficulties, we implemented the HLLC Riemann solver (Toro 1999). This is an approximate solver, so it does not return the correct flux when there is a large difference between the values of the hydrodynamic variables in adjacent cells. However, because the Godunov method tends to smear a discontinuity across a few cells (and thus reduce the difference between adjacent cells), the values of the hydrodynamic variables before and after the discontinuity converge to the correct values within a few time steps. Also, because it only uses the energy and speed of sound, it can be used with any equation of state, not just ideal gas. In order to take into account the motion of the edges, we solve each Riemann problem in the reference frame of the moving edge.

2.5. Parallelization Scheme

Our code is made parallel by the use of the MPI interface. Our domain is decomposed by building a Voronoi diagram from CPU points that represent the different CPUs, and each CPU holds in its memory only the hydro points that are inside its Voronoi cell. In order to maintain a good load balance throughout the run, we move the CPU mesh points in a way to preserve the workload as roughly equal. Our parallelization scheme is discussed in Steinberg et al. (2014).

3. ONE-DIMENSIONAL TEST PROBLEMS

In order to test our code, we run a set of 1D and 2D test problems. For all of the 1D test problems we compute the convergence rate and an error function, which is problem specific, by repeating the tests with resolutions of 32–256 cells. A summary of all of our results is given in Table 1.

3.1. Simple Waves

We repeat the test described in Colella et al. (2006), which tests the propagation of large perturbations. The density is given by

$$\rho_0(x) = \begin{cases} 1 & |\, x - 0.3\, | > 0.5 \\ 1 + 10 \cdot \left((x-0.3)^2 - 0.5\right)^4 & |\, x - 0.3\, | < 0.5 \end{cases}, \tag{9}$$

and the pressure and velocity are chosen so that the entropy and the negative Riemann invariant would be uniform throughout the domain, so there would only be a forward-moving wave. The pressure and the velocity are given by

$$p_0(x) = \rho_0^{\gamma}(x), \tag{10}$$

$$v_0(x) = \frac{2}{\gamma - 1}\left[\sqrt{\gamma\frac{p_0(x)}{\rho_0(x)}} - \sqrt{\gamma\frac{p_0(1)}{\rho_0(1)}}\right], \tag{11}$$

where $\gamma = \frac{5}{3}$, and the problem is set up with a domain of [0, 1] using periodic boundaries. The calculation was terminated at time $t = 0.02$. In this test the error function, L, is defined as

$$L = \sum_{i=1}^{N} \frac{\left|\phi_i^{\,n} - \phi_i^{\,a}\right|}{\left|\phi_i^{\,a}\right| N}, \tag{12}$$

where $\phi_i^{\,n}$ is the numerical hydrodynamic quantity, $\phi_i^{\,a}$ is the analytical value, N is the total number of cells, and subscript i represents the value at a spatial position with index i. For the velocity we replace $\phi_i^{\,a}(x)$ in the denominator by the speed of sound. In this test, other things being equal, the Lagrangian did better than the Eulerian, and the time-centered flux did better than extrapolated fluxes.

3.2. Acoustic Waves

The acoustic waves problem checks how small perturbations propagate. When the perturbations are small, the hydrodynamic equations can be linearized and solved analytically (Landau &

Table 1
The Fit Parameters for the L Error Function for Different Tests, as Described in the Text for Our 1D and 2D Test Problems.

		Time-Centered Fluxes—RICH				Extrapolated Fluxes—AREPO			
		Eulerian		Semi-Lagrangian		Eulerian		Semi-Lagrangian	
Test Name	L Equation	α	β	α	β	α	β	α	β
Acoustic density	(16)	−1.79	−11.55	…	…	−1.97	−11.25	…	…
Acoustic pressure	(16)	−1.79	−11.55	…	…	−1.97	−11.25	…	…
Acoustic velocity	(16)	−1.79	−11.55	…	…	−1.97	−11.25	…	…
Simple wave density	(12)	−1.69	4.03	−2.28	5.09	−1.63	3.77	−1.83	3.35
Simple wave pressure	(12)	−1.79	5.39	−2.3	6.17	−1.74	5.18	−1.92	4.68
Simple wave velocity	(12)	−1.76	4.52	−2.62	6.29	−1.68	4.2	−2.22	4.87
Shock tube density	(18)	−0.85	0.56	−0.83	0.17	−0.82	0.42	−0.82	0.09
Shock tube pressure	(18)	−1.02	1.89	−1.01	1.74	−0.97	1.67	−1.01	1.6
Shock tube velocity	(18)	−1.03	0.97	−0.99	0.61	−0.94	0.61	−1	0.6
Sedov Taylor density	(31)	−0.61	0.89	−0.76	1.78	−0.61	0.86	−0.78	1.87
Sedov Taylor pressure	(31)	−0.64	4.00	−0.78	4.89	−0.65	4.02	−0.81	4.94
Sedov Taylor velocity	(31)	−0.72	2.1	−0.82	−1.18	−0.72	2.12	−0.83	2.8
Standing driven waves density	(23)	−1.87	−7.84	…	…	−1.33	−8.61	…	…
Standing driven waves pressure	(23)	−1.5	−8.62	…	…	−1.33	−8.1	…	…
Standing driven waves velocity	(23)	−2.28	−5.04	…	…	−1.44	−8.35	…	…
Gresho vortex density	(34)	−1.64	−2.96	−1.53	−3.37	−1.46	−3.42	−1.58	−3.21
Gresho vortex pressure	(34)	−1.64	−0.71	−1.54	−1.13	−1.47	−1.15	−1.52	−1.15
Gresho vortex velocity	(34)	−1.39	−2.28	−1.32	−2.57	−1.4	−2.17	−1.32	−2.6
Noh density	(27)	−0.78	2.63	−0.75	2.67	−0.76	2.54	−0.88	2.96
Noh pressure	(27)	−0.87	1.76	−0.86	1.84	−0.85	1.67	−1.06	2.36
Noh velocity	(27)	−0.76	−0.06	−0.91	0.47	−0.73	−0.15	−1.03	0.79

Notes. We Fit the Data to a Function of the Form $\log(L_1) = \alpha \ln(n) + \beta$ Where n Is the Resolution, α Is the Slope and β Is the Offset. Columns Represent the Reference to the Equation in Which L Is Defined, the Time-Advancement Scheme, and if the Mesh Points Were Allowed to Move in a Semi-Lagrangian Nature or Not.

Lifshitz 1987). Another feature of small perturbations is that the approximate Riemann solver gives results that are very close to the exact Riemann solver (Toro 1999).

The problem is set up with a domain of [0, 1], and the equation of state is of an ideal gas with adiabatic index $\gamma = \frac{5}{3}$. The boundary conditions are set to be periodic, and the initial conditions are

$$\rho_0(x) = 1 + 10^{-6}\sin(2\pi x) \tag{13}$$

$$p_0(x) = \frac{3}{5} + 10^{-6}\sin(2\pi x) \tag{14}$$

$$v_0(x) = 0. \tag{15}$$

We compare the spatial profile at time $t = 1$ (the time it takes a sound wave to go full circle) to the analytical profiles (which are identical to the initial conditions) using

$$L = \sum_{i=1}^{N} \frac{\left|\phi_i^n - \phi_i^a\right|}{10^{-6}N}. \tag{16}$$

Because the velocities are very low, there is almost no difference between the Eulerian and Lagrangian schemes; both achieve a second-order convergence rate, and we report only the Eulerian schemes.

The measured prefactors that are reported in Table 1 show that, in this test, the extrapolated fluxes time-advancement scheme has a prefactor that is about 1.5 lower than the time-centered fluxes time-advancement scheme. However, the slope of the latter is steeper than the former.

3.3. Shock Tube

The shock tube problem tests the code's ability to resolve strong shocks and discontinuities. The initial conditions are

$$\rho_0(x) = 1, \quad p_0(x) = \begin{cases} 1 & x > 0.5 \\ 10 & x < 0.5 \end{cases}, \quad v_0(x) = 0 \tag{17}$$

in the domain $x \in [0, 1]$. The exact, self-similar solution can be found by the solution of the Riemann problem (Toro 1999), and the profiles are compared to the analytical solution at time $t = 0.1$. We compare the numeric results to the analytic solution using the following error norm:

$$L = \sum_{i=1}^{N} \frac{\left|\phi_i^n - \phi_i^a\right|}{\left|\phi_i^a\right| N}, \tag{18}$$

except for the velocity, which is again normalized by the speed of sound. Table 1 show that the time-centered fluxes time-advancement scheme has the same error as the extrapolated fluxes scheme, and in both schemes the semi-Lagrangian movement is better than the Eulerian. Because this problem involves shocks, we do not get second-order convergence, but only first order.

3.4. Standing Driven Waves

The main goal of this problem is to test the accuracy of the code when coupled to external forces or sources. We start out with a smooth uniform hydrodynamic profile $\rho(x) = \rho_0 = 1$, $p(x) = p_0 = 1$, $v = v_0 = 0$ on the domain $x \in [0, 1]$ with periodic boundary conditions. Perturbations are introduced by an external acceleration

$$f(x, t) = A\sin(kx)\sin(ktv), \tag{19}$$

where $A = 10^{-4}$ has the units of acceleration, $k = 2\pi$, and $v = 0.1$. The perturbations in the hydrodynamic variables, which are obtained from the analytical solution for $A \ll 1$, are given by

$$\delta\rho = \frac{A\rho_0\cos(kx)\left(v\sin\left(c_0kt\right) - c_0\sin(ktv)\right)}{c_0k\left(c_0^2 - v^2\right)} \tag{20}$$

$$\delta p = \frac{Ac_0\rho_0\cos(kx)\left(v\sin\left(c_0kt\right) - c_0\sin(ktv)\right)}{k\left(c_0^2 - v^2\right)} \tag{21}$$

$$\delta v = \frac{Av\left(\cos(ktv) - \cos\left(ktc_0\right)\right)\sin(kx)}{k\left(c_0^2 - v^2\right)}, \tag{22}$$

where $c_0 = \sqrt{\gamma p_0/\rho_0}$ and $\gamma = \frac{5}{3}$. For this test problem we define the error function as

$$L = \sum_{i=1}^{N}\frac{\left|\phi_i^n - \phi_i^a\right|}{\left|\phi_i^a\right|AN}, \tag{23}$$

and the velocity is once again normalized by the speed of sound. The slope of the time-centered flux is in the range −1.5 to −2.3, and the slope of the extrapolated fluxes is −1.3 to −1.5.

4. TWO-DIMENSIONAL TEST PROBLEMS

4.1. *Pure Advection*

One of the benefits of having a moving mesh is that it should handle advection much better than do Eulerian codes. In this test we set the velocity and pressure to constant values and the density to some nontrivial distribution. Physically, it is equivalent to a static environment viewed from a moving reference frame. When those initial conditions are advanced by a Eulerian scheme, the features of the initial density distribution tend to diffuse. In a Lagrangian scheme we would expect no such distortion, so the error should be zero, up to numerical precision. We note that the motion of the mesh generating points slightly deviates from Lagrangian motion in order to make the cells round. Therefore, if the initial cells will not be round enough, there will be some diffusion even in the Lagrangian case.

We choose the pressure to be $p = 1$, velocity $\vec{v} = \hat{x} + \hat{y}$ (so it will not be parallel to either axis), and the density distribution to be

$$\rho(r) = \begin{cases} 100 & r < 0.2 \\ 1 & r > 0.2 \end{cases}.$$

The domain is set to be $[-0.5, 0.5]^2$ with periodic boundary conditions. The simulation is then run to time 1 (the time it takes all of the points to come full circle) and compare the initial and final snapshots of the density. The test was run with different resolutions, and in all cases the errors were consistent with machine round-off error when the mesh was allowed to move with the fluid.

When considering pure advection, the moving mesh has the great advantage of having zero error, compared to Eulerian codes where the error depends on the fluid's velocity.

4.2. *Noh Problem*

The Noh problem (Noh 1987) checks how the code handles strong shocks and highly supersonic flow. The setup for the test is a uniform density $\rho_0 = 1$, small uniform pressure $p = 10^{-6}$, and uniform radial inflow velocity $v = 1$, and the adiabatic index is set to $\gamma = \frac{5}{3}$. The analytic self-similar solution is

$$\rho(r, t) = \begin{cases} \left(\dfrac{\gamma+1}{\gamma-1}\right)^2 & r < t(\gamma-1)/2 \\ 1 + t/r & r > t(\gamma-1)/2 \end{cases} \tag{24}$$

$$v(r, t) = \begin{cases} 0 & r < t(\gamma-1)/2 \\ -1 & r > t(\gamma-1)/2 \end{cases} \tag{25}$$

$$p(r, t) = \begin{cases} \dfrac{1}{2}\dfrac{(\gamma+1)^2}{\gamma-1} & r < t/3 \\ 10^{-6} & r > t/3 \end{cases}. \tag{26}$$

One way to simulate this problem is to have the computational grid only in the first quadrant $[0, 1]^2$ and use rigid wall boundary conditions on the lower $(y = 0)$ and left $(x = 0)$ boundaries. However, we choose to take after AREPO and use the computational domain of $[-1, 1]^2$. This allows us to verify how well our code preserves reflection symmetry, which was achieved.

We use 2500 mesh generating points randomly distributed across the domain, and the boundary conditions are dictated from the analytic solution. An inflow boundary condition poses no difficulty in the case of Eulerian point motion, but in the case of a semi-Lagrangian point motion, cells close to the origin would tend to compress and shrink, thus causing the time step to plummet, while cells far away from the origin would tend to bloat, thus causing loss of precision. To remedy this problem, we use adaptive mesh refinement, like in AREPO. We split cells once their volume increases above 150% of their initial value and coarsen them when their volume drops below 25% of their initial value.

We define the error norm in this case as

$$L = \sum_{i=1}^{N}\frac{\left|\phi_i^n - \phi_i^a\right|}{\left|\phi_i^a\right|N}. \tag{27}$$

The results in Table 1 show that all of the schemes give comparable results, with Eulerian being slightly better. This is the result of the AMR scheme that derefines the cells around the shock front and the area inconsistency error that is described in Section 5.

4.3. *Sedov Taylor Explosion*

Like the shock tube problem, this problem tests the ability of the code to handle shocks. However, it differs from the shock tube in that the downstream is not uniform. The computational domain is $[-1, 1] \times [-1, 1]$. The initial conditions are

$$\rho_0(r) = 1 \tag{28}$$

$$p_0(x) = \begin{cases} 0.01 & r > 0.06 \\ 10^4 & r < 0.06 \end{cases} \tag{29}$$

$$\vec{v}_0(r) = 0. \tag{30}$$

The simulation is advanced to time $t = 0.04$, where the shock radius is approximately $r \approx 0.8$. Like in the previous problems, the error norm is calculated by summation of the absolute values of the differences from the analytic solution (Landau & Lifshitz 1987):

$$L_1 = \frac{1}{N}\sum_{i=1}^{N}\frac{\left|\phi_i^a - \phi_i^n\right|}{\left|\phi_i^a\right|}. \tag{31}$$

We found that all schemes converge slower than N^{-1} and that in general the Lagrangian schemes converge better than do Eulerian schemes.

4.4. Gresho Vortex

We repeated the simulation of the Gresho vortex problem as described in the AREPO code paper (Springel 2010). In this problem, the initial density is uniform and equal to one. The pressure is given in polar coordinates by

$$P(r, \phi) = \begin{cases} 5 + \frac{25}{2}r^2 & r < \frac{1}{5} \\ 9 + \frac{25}{2}r^2 - 20r + 4\ln(5r) & \frac{2}{5} > r > \frac{1}{5}, \\ 3 + 4\ln 2 & r > \frac{2}{5} \end{cases} \tag{32}$$

and the azimuthal velocity is

$$v_\phi(r, \phi) = \begin{cases} 5r & r < \frac{1}{5} \\ 2 - 5r & \frac{2}{5} > r > \frac{1}{5} \\ 0 & r > \frac{2}{5}. \end{cases} \tag{33}$$

The pressure balances the centrifugal force, so the variables should not change in time. In this test L is defined as

$$L = \sum_{i=1}^{N}\frac{\left|\phi_i^n - \phi_i^a\right|}{\left|\phi_i^a\right| N}. \tag{34}$$

Like Springel (2010), we found $L \propto N^{-1.5}$ and the prefactor in the time-centered fluxes time-advancement scheme was slightly better than the one in the extrapolated fluxes scheme. With the time-centered fluxes time-advancement scheme, there was no difference between Lagrangian and Eulerian grid motion as long as the calculation was performed in the reference frame where the vortex is stationary. The Lagrangian simulation gave considerably better results only when the vortex was boosted.

4.5. Kelvin–Helmholtz Instability

One of the main benefits of a semi-Lagrangian code is that it preserves contact discontinuities better than do Eulerian codes. A classic test that demonstrates this difference is the Kelvin–Helmholtz instability (Chandrasekhar 1961). This instability occurs between two superposed fluid layers moving in parallel to their interface. Our setup is identical to that described in AREPO, with a resolution of 50×50 mesh generating points in the domain $[0, 1]^2$ with periodic boundary conditions and a termination time of $t = 2$. Specifically, the pressure is set to be $P = 2.5$ throughout the domain, and the density and the velocity are given by

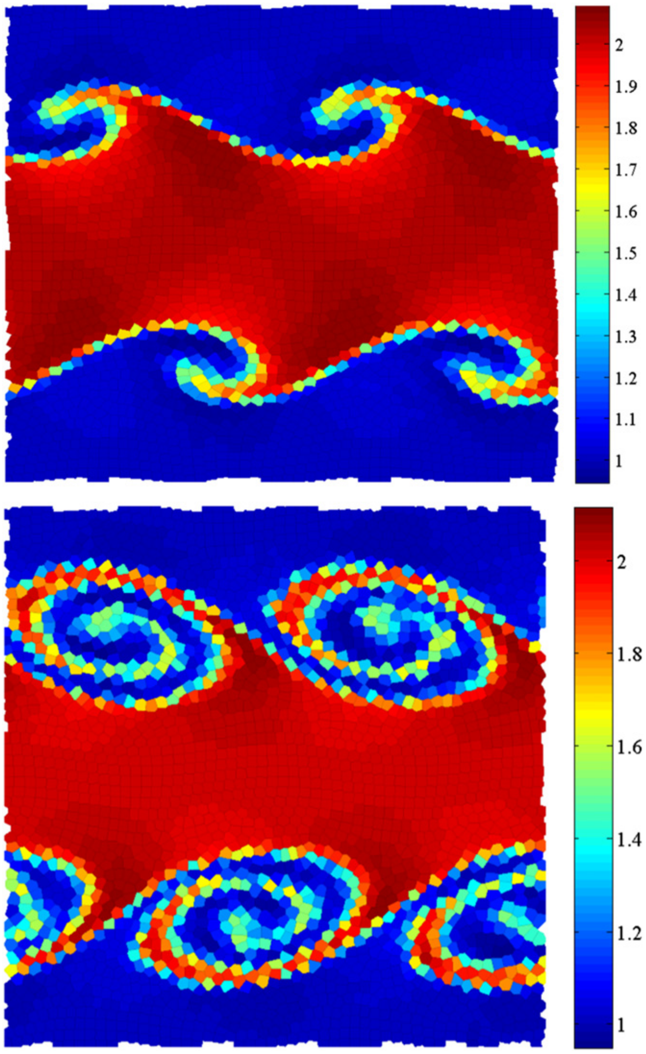

Figure 2. Density map of the Kelvin–Helmholtz problem at $t = 1$ (left) and $t = 2$ (right). The setup for this single-mode test is described in the text.

$$\rho(x, y) = \begin{cases} 1 & |y - 0.5| > 0.25 \\ 2 & |y - 0.5| < 0.25, \end{cases} \tag{35}$$

$$v_x(x, y) = \begin{cases} -0.5 & |y - 0.5| > 0.25 \\ 0.5 & |y - 0.5| < 0.25, \end{cases} \tag{36}$$

$$v_y(x, y) = 0.1\sin(4\pi x)\left(e^{-\frac{(y-0.25)^2}{2\sigma^2}} + e^{-\frac{(y-0.75)^2}{2\sigma^2}}\right), \tag{37}$$

where $\sigma = 0.05/\sqrt{2}$ and the adiabatic index is set to be $\gamma = 5/3$. Figure 2 shows two snapshots of our simulation at times $t = 1$ and $t = 2$. The snapshots seem very similar to AREPO's (Figure 32 in their paper) and preserve the discontinuity between the fluids rather well. We also verified that these snapshots do not change, even when a constant boost is added to all cells in the initial conditions.

4.6. Rayleigh–Taylor Instability

A Rayleigh–Taylor instability involves constant, uniform external force and a discontinuity between densities. Again, the setup was the same as in AREPO (Springel 2010), with a

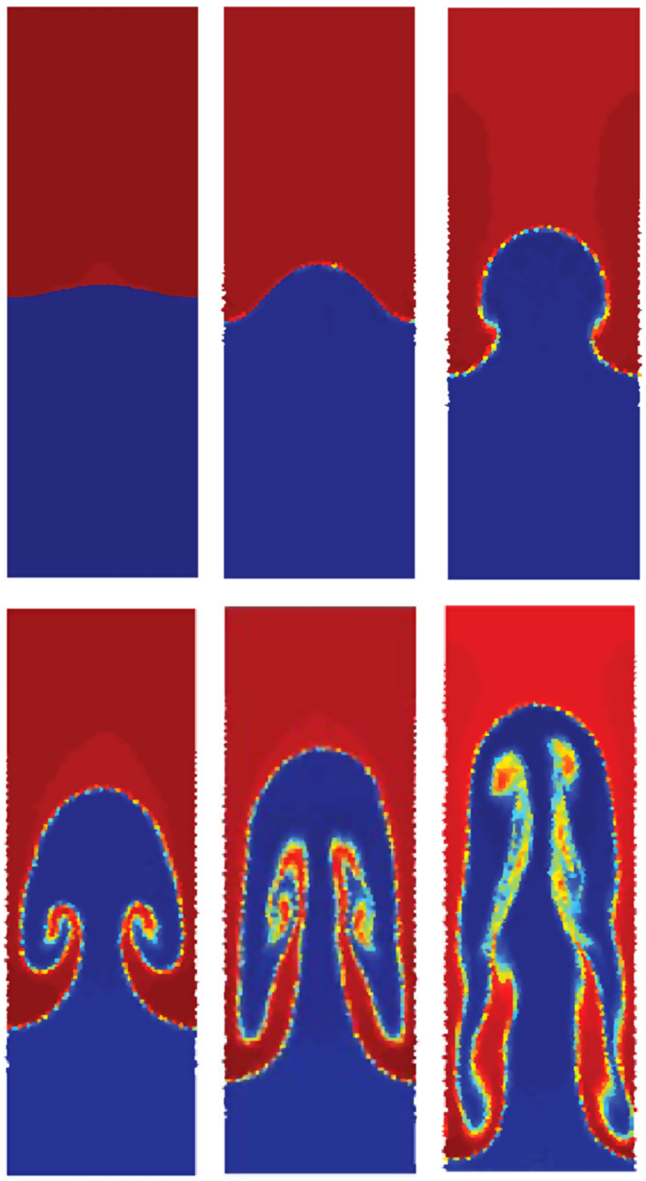

Figure 3. Density maps of the Rayleigh–Taylor instability at different times. The initial setup is given in the text.

resolution 48 × 144 in the domain $x \in [0, 0.5]$ and $y \in [0, 1.5]$ and with periodic boundary conditions for the x axis and reflecting for the$y$ axis. The initial setup is

$$v_x(x, y) = 0, \tag{38}$$

$$v_y(x, y) = w_0(1 - \cos(4\pi x))(1 - \cos(4\pi y/3)), \tag{39}$$

$$\rho(x, y) = \begin{cases} 1 & y < 0.75 \\ 2 & y > 0.75, \end{cases} \tag{40}$$

$$P(x, y) = P_0 + g(y - 0.75), \tag{41}$$

where $g = -0.1$, $P_0 = 2.5$, and $w_0 = 0.0025$. The simulation is run until $t = 15$ with an adiabatic index $\gamma = 1.4$ run until time $t = 15$. Figure 3 shows snapshots at different times of this simulation. As was shown in AREPO, the semi-Lagrangian scheme is less diffusive than the Eulerian scheme.

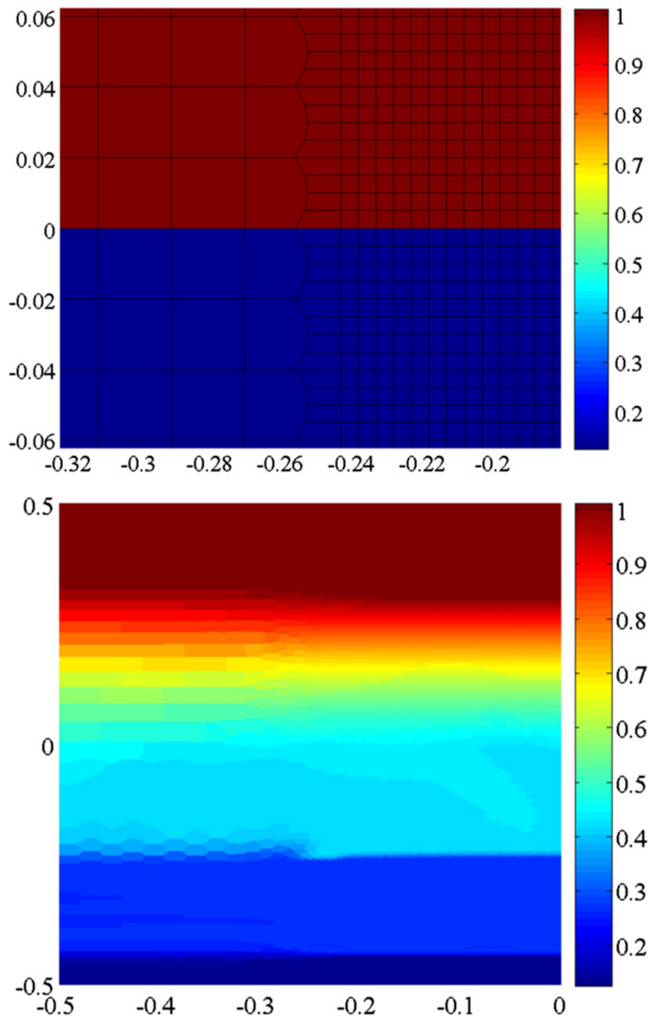

Figure 4. The initial grid setup for the 2D Sod problem (left, zoomed in) and the evolved symmetry breaking at $t = 0.25$. Color denotes density.

4.7. Sod Shock Tube With Large Cell Volume Gradient

It is common knowledge that in AMR simulations neighboring cells should be of similar size (Kravtsov et al. 1997). In a moving mesh simulation, cells of different sizes can become neighbors even without AMR. Neighbors with a large volume ratio can cause numerical errors in the simulation and even crash it.

One reason for this is that information travels farther in large cells than in small cells. To demonstrate this phenomenon, the classic 1D Sod problem is run on a 2D grid with an uneven mesh. The initial conditions are

$$\rho_0(x, y) = \begin{cases} 1.0 & : y > 0 \\ 0.125 & : y < 0' \end{cases} \tag{42}$$

$$p_0(x, y) = \begin{cases} 1.0 & : y > 0 \\ 0.1 & : y < 0' \end{cases} \tag{43}$$

$$v_0(x, y) = 0. \tag{44}$$

The initial conditions are independent of x, but the resolution is not. The domain $0 < x < -0.25$ has a resolution of 25 cells, and the domain $-0.25 < x < 0$ has a resolution of 100 cells, as shown in Figure 4. As time advances, the waves propagate at different velocities on each side of the grid, and that causes asymmetry in the hydrodynamics. Also, small cells next to large cells tend to have an aspect ratio much different from unity, which can cause numerical errors and crash the simulation in extreme cases.

However, in most simulations this is not a critical issue because the cell rounding scheme prevents most of the extreme cases of different size ratios. In the few special cases where the cell rounding scheme does not fix this issue, the problem can be remedied by splitting cells when they become much larger than their neighbors or coarsening cells when they get much smaller than their neighbors.

5. AREA INCONSISTENCY PROBLEM

Because we solve the Riemann problem in a moving reference frame, this implicity assumes that the cell is going to change its area according to

$$\dot{A}_i = -\sum_{i \neq j} L_{ij} \left(\left[\vec{w}_j - \vec{w}_i \right] \frac{\vec{c}_{ij}}{r_{ij}} - \frac{\vec{w}_i + \vec{w}_j}{2} \frac{\vec{r}_{ij}}{r_{ij}} \right), \tag{45}$$

where $\vec{w}$ is the velocity of the mesh generating point, and the rest are defined in Equation (1). However, the actual change in the cell's area is not $\dot{A}_i \Delta t$, which is accurate only to a first order in time (more exactly to a first order in the CFL number). The resulting difference between the expected change in the area and the actual change results in an error in the calculated fluxes (the scheme is still conservative). Moreover, the error is resolution independent because everything scales with the size of the cells and in principle can be of order unity. This inconsistency can be demonstrated with a very simple test problem involving a strong shock, which induces a large variation in the cell's geometry. The initial conditions are set to be

$$\rho = 1, \tag{46}$$

$$p = 0.1, \tag{47}$$

$$v_x = 1, \tag{48}$$

$$v_y = 0, \tag{49}$$

the adiabatic index is $\gamma = \frac{5}{3}$, and the problem is set up with a domain of $[-1, 1]^2$ with rigid walls, except the left wall, which has inflow boundary conditions. The inflowing material creates a shock wave that moves to the left with a velocity of $U \sim 0.448$ and has a postshock density of $\rho_\mathrm{d} \sim 3.23$. In the Lagrangian scheme, cells are compressed during the passage of the shock wave, and the area inconsistency error is largest there. We run the simulation until $t = 1.3$ with a CFL of 0.6 for various resolutions and record the maximal deviation from the analytical prediction in the downstream density (in units of ρ_d), as well as the error function that is defined as

$$L = \sum_{i=1}^{N} \frac{\left| \rho_i^{\,n} - \rho_\mathrm{d} \right|}{\rho_\mathrm{d} N}, \tag{50}$$

where the summation is done only on the downstream cells, excluding those adjacent to the rigid wall and those adjacent to the shock wave.

In Figure 5 we show L, the error function, for the two time-advancement schemes as well as for the Eulerian and Lagrangian point motion as a function of the resolution (the one-dimensional number of points). The nature of the time-centered flux time-advancement scheme solves Equation (45) to a higher accuracy than the extrapolated fluxes scheme. The time-centered flux time-advancement scheme has an error that is a factor of two less than the extrapolated fluxes scheme. For lower CFL numbers, the ratio in the error between the two time-advancement schemes only increases. Also, the error in the semi-Lagrangian schemes is constant because of the area inconsistency problem, while the Eulerian schemes have first-order convergence, as expected. This is in stark contrast to the 1D shock tube test, where the Lagrangian scheme was better because it had no area inconsistency problem. The maximal error is indeed of order unity, as can be seen in Figure 6. In fact, the maximal error increases with resolution for the Lagrangian schemes; this is because there are more cells while the probability of having a large error is constant.

Because typically the errors between time steps are uncorrelated, the cumulative error is a random walk of the error of a single time step, until the error is large enough that it is canceled by the diffusion term.

Is this error critical? Typically, large errors occur only when the cells are very "unround," otherwise the difference between the calculated change in the area and the actual change are small. The errors do not change the overall dynamics of the simulation but might cause errors on the level of a few percent in a few cells and in extreme cases an error of order unity in a handful. Nevertheless, we have no a priori guarantee that in a specific calculation these errors would be small. We note that the results shown in this section do not prove that the area inconsistency is the culprit for these errors. However, when the area difference was taken into account, the errors disappeared in the stationary problem described above. Unfortunately, they reemerged when the fix was applied to nonstationary problems. Hence, further research is required to find a robust remedy.

6. DEVIATION FROM LAGRANGIAN MOTION AND DIFFUSION

Moving the mesh generating points strictly with the fluid velocity can cause cells to become very elongated over time. This has the downside of causing the code to be unstable and even crash in extreme cases. An additional issue arises when two mesh points are close to each other, which can cause the mutual edge to have a large rotational velocity that can induce errors in the hydrodynamics (in RICH, we have implemented a safeguard against such occurences that prevents lateral motion when neighboring mesh points are too close). In order to fix this issue, AREPO has proposed to add an additional velocity to the mesh point whenever the mesh point is far from the center of the cell. The added velocity brings the mesh point closer to the cell's center. This fix is controlled by two parameters, χ, which defines in units of the cell's sound speed how fast the additional velocity is, and η, the criteria of how far in units of the cell's radius the mesh point is allowed to deviate from the cell's center before the fix is applied.

Because this additional velocity is typically not in the direction of the fluid's velocity, a non-Lagrangian motion occurs, resulting with advection between cells. The more often the fix is applied, the more advection takes place, and the higher the fix velocity is, the higher the diffusion error in the advection term. However, having a diffusion error is not necessarily a bad thing because it allows the smoothing out of small-scale errors. Mesh geometry induces errors with a wavelength comparable to the cell's size. The errors in the

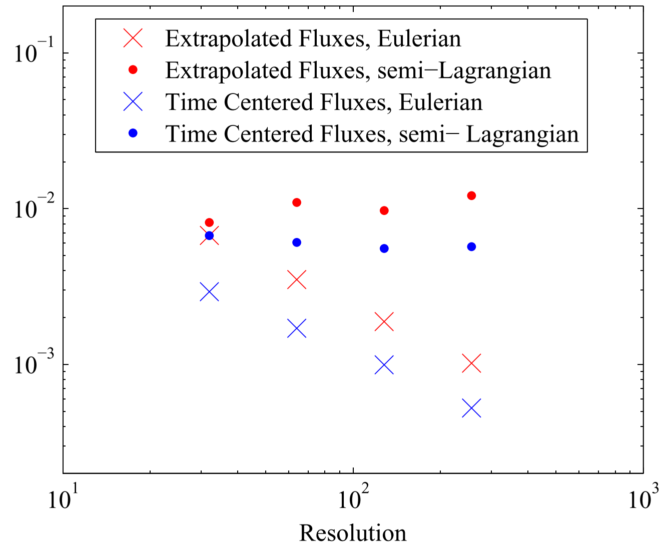

Figure 5. *L*, the error function of the density for the strong shock test, as a function of resolution for different time-advancement schemes and for Eulerian or semi-Lagrangian meshes. The initial conditions for the test are described in the text.

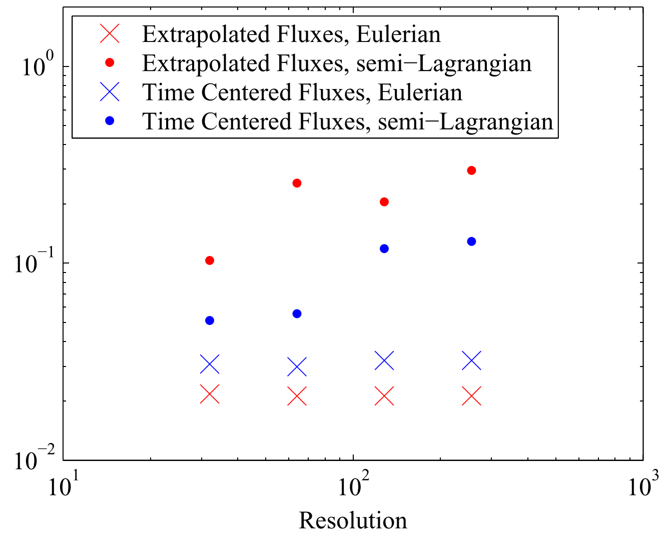

Figure 6. The maximal deviation from the analytical prediction in the downstream density (in units of ρ_d) as a function of resolution for different time-advancement schemes and for Eulerian or semi-Lagrangian meshes for the strong shock test. The initial conditions for the test are described in the text.

pressure and velocity tend to quickly adjust themselves to a smooth pattern, but errors in the density take longer to smooth out if the motion is Lagrangian. An additional concern is that if $\chi \approx 1$, it can have a negative effect on the time step because it can significantly increase the fluid's velocity relative to the edge's velocity.

In order to show the dependence of the code on the fix parameters, we run the Gresho vortex problem as presented in Section 4.4 with different parameters of the cell roundness fix and with a resolution of 30^2 cells. In Figure 7 we show *L*, the error function as described in Equation (34) of the density for values of $\chi \in [0.01, 1]$ and $\eta \in [0.001, 0.5]$. The lowest value of *L* is given approximately when $\chi = 0.15$ and $\eta = 0.02$, and we set those values to be our default choice when we run the code. These results can also be understood in the following intuitive way. There are two possible approaches to applying the correction. In the first, we postpone the interference as much as possible and apply a drastic correction once a certain threshold was crossed. In the second, we continuously apply subtle correction. Figure 7 shows that the second approach is more accurate than the first. Our experience has shown that this behavior is not unique to this problem, but recurs in other problems as well.

7. IS LAGRANGIAN BETTER?

In this section we focus on linear finite difference schemes, i.e., a recurrence relation of the form

$$y_i^{n+1} = \sum_j a_j y_j^n, \tag{51}$$

where *y* is the dependent variable, the lower index is the spatial, the upper is temporal, and a_j are constants. Such schemes can be solved analytically using Fourier transform (Richtmyer & Morton 1994). Because the hydrodynamic equations are nonlinear, such a scheme is of little use. However, in the limit of small perturbations to a uniform background, it is possible to obtain a linear approximation (Landau & Lifshitz 1987). In this limit, the hydrodynamic equations are reduced to three decoupled linear advection equations:

$$\frac{\partial s}{\partial t} + v_0 \frac{\partial s}{\partial x} = 0 \tag{52}$$

$$\frac{\partial j_\pm}{\partial t} + (v_0 \pm c_0) \frac{\partial j_\pm}{\partial x} = 0, \tag{53}$$

where $s = \log(p/\rho^\gamma) \approx \frac{\delta p}{p_0} - \gamma \frac{\delta \rho}{\rho_0}$ is the entropy and $j_\pm = \delta v \pm \frac{\delta p}{\rho_0 c_0}$ are the Riemann invariants. In the limit of small perturbations, Godunov's method is reduced to a finite difference scheme with three decoupled linear advection equations (Toro 1999).

7.1. First Order

The first-order scheme for the linear advection equation

$$\frac{\partial y}{\partial t} + v \frac{\partial y}{\partial x} = 0 \tag{54}$$

(assuming positive drift velocity *v*) is

$$y_i^{n+1} = y_i^n + \xi (y_{i-1} - y_i), \tag{55}$$

where $\xi = \frac{v \Delta t}{\Delta x}$ is the Courant–Friedrichs–Levy number, Δx is the cell size, and Δt is the time step. Denoting the imaginary number by $I = \sqrt{-1}$, to avoid confusion with the indices, and $n = \frac{t}{\Delta t} = \frac{tv}{\Delta x \xi}$, we substitute the Fourier mode

$$y(x, t) = A \sigma^{t/t_0} \exp(-Ikx) \tag{56}$$

or equivalently

$$y_i^n = A\left(\sigma^{\Delta t/t_0}\right)^n \exp(Iki\Delta x) \tag{57}$$

into the first-order finite difference scheme (Equation (55)) and have

$$\sigma^{\Delta t/t_0} = 1 - \xi(1 - \exp(-Ik\Delta x)), \tag{58}$$

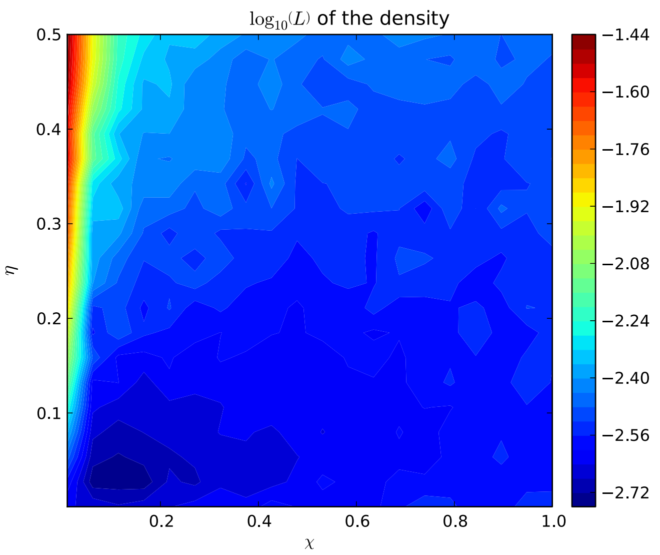

Figure 7. L, the error function of the density for the Gresho vortex test for different values of χ, the magnitude of the cell roundness fix in units of the cell's sound speed, and η, the criteria for how far the mesh point should be from the cell's center of mass in units of the cell's radius, in order to apply the fix. The initial conditions for the test and the definition of L are described Section 4.4.

$$\frac{y_i^n}{y_i^0} = [1 - \xi(1 - \exp(-Ik\Delta x))]^n . \tag{59}$$

In the limit $\Delta x \ll \frac{1}{k}$ (while ξ remains constant) Equation (59) simplifies to

$$\frac{y_i^n}{y_i^0} \approx \exp(-Iktv)\exp\left(-\frac{1}{2}(1 - \xi)k^2\Delta xtv\right). \tag{60}$$

The first term on the right-hand side is simply a shift (which happens to be the exact Fourier filter for the advection equation), and the second term is the leading term in the error caused by the finite difference. In this case, the first-order finite difference scheme introduces artificial attenuation. We note that if $\xi > 1$, then attenuation becomes amplification, and the scheme becomes unstable. If $\xi = 1$, then the second term disappears and the wave travels without distortions. This phenomenon is known as the "magic time step" (Taflove & Hagness 2005). However, it is never used in practice, mainly because, as we mentioned before, hydrodynamics involves three wave speeds (which also vary in space), so it is impossible to choose a single time step for which all CFL numbers would be 1.

7.2. Second Order

The same exercise as in Section 7.1 can be done for a second-order scheme, bearing in mind that it has to be second order in both space and time:

$$y_i^{n+1/2} = y_i^n + \frac{1}{4}\xi\left(y_{i-1}^n - y_{i+1}^n\right) \tag{61}$$

$$y_i^{n+1} = y_i^n + \frac{1}{2}\xi\left(y_{i-1}^{n+1/2} - y_{i+1}^{n+1/2}\right) \tag{62}$$

Substituting the Fourier mode (Equation (57)) yields the filter

$$\frac{y_i^n}{y_i^0} = \left(1 - I\xi\sin(k\Delta x) - \frac{1}{2}\xi^2\sin^2(k\Delta x)\right)^n . \tag{63}$$

In the limit of small Δx

$$\frac{y_i^n}{y_i^0} \approx \exp(-Ikvt)\exp\left(\frac{I}{6}\left(1 - \xi^2\right)k^3\Delta x^2tv\right). \tag{64}$$

In this case, the leading error decreases the effective propagation speed, so the numerical wave always lags behind the exact solution. This scheme is unstable for all values of the CFL number, in accordance with Godunov's theorem (Godunov 1961). In practice, this difficulty is circumvented by the use of slope limiters (Toro 1999), which introduce nonlinearity to the scheme.

7.3. Grid Motion

The formalism described above can be used to explore the effects of grid motion (which is usually chosen to be either Eulerian or Lagrangian) by repeating the calculation described above but varying the ambient velocity. In a simple linear advection equation, the higher the velocity, the less accurate the scheme will be. We recall that a snapshot of some variable is represented by a discrete set of values at a fixed position. Suppose we start out with the same initial condition and advance it to time t using two different methods. The first method uses the exact solution to the advection equation, and the second method uses the analytic-numeric method described above. This will yield two sets of values, ϕ_i^1 and ϕ_i^2, where i is the spatial index. In order to measure how close both sets are, we define the following function:

$$L_1 = \frac{1}{AN}\cdot\sum_{i=1}^{N}\left|\phi_i^1 - \phi_i^2\right|, \tag{65}$$

where N is the number of terms of x_i (and also y_i), and A is the amplitude of the wave. The latter is included so that L_1 will be dimensionless. Figure 8 shows the variation of the L_1 error norm as a function of the ambient velocity for both first- and second-order time-advance schemes for the case of a single mode as the initial condition, where the resolution is 100 cells, the wavenumber is $2\pi \cdot 10$, the time is 1, and the CFL number is 0.3. The first-order scheme seems to grow linearly and then saturates. This occurs when the wave decays to zero due to numerical viscosity, so $L_1 \leqslant 1$. In the case of the second-order scheme, the error first increases but then starts decreasing and continues to oscillate. The oscillations occur because the lag increases with the velocity, but when the phase approaches a multiple of 2π the numeric and analytic waves coincide and the error decreases. At even higher velocities the next term dominates and the error grows monotonically. The value of L_1 at early times for both schemes can be approximated analytically. We assume that the initial conditions are a single Fourier mode $y(x) = \exp(Ikx)$. We then obtain two spatial profiles at a later time t. The first profile is obtained using the exact solution to the advection equation by multiplying by $\exp(-Iktv)$. The second profile is obtained by multiplying by the filter of the first order scheme, taylor expanded for $\Delta x \to 0$ (Equation (60)). Comparing the two profiles using the L_1 norm

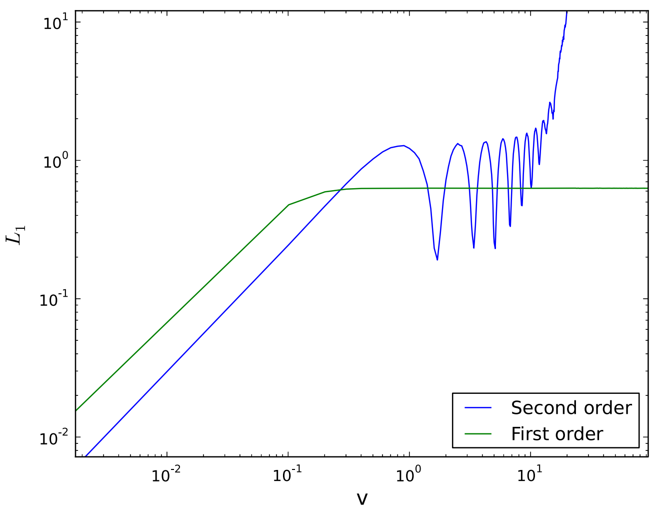

Figure 8. The variation of the L_1 norm (see Equation (65)) relative to the analytic solution as a function of the drift velocity v, for both first- and second-order schemes of the linear advection equation. The resolution is 100 cells, the initial profile was a sine wave with amplitude 1 and wavenumber $2\pi \cdot 10$, the time is 1, and the CFL coefficient is 0.3.

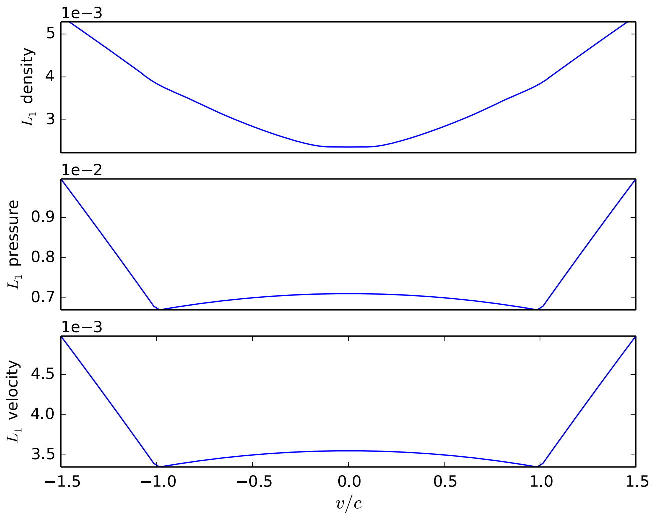

Figure 9. L_1 measure of the difference between analytic and numerical solutions as a function of the Mach number for the first-order scheme. The initial conditions for this problem were $\rho(x) = 1$, $p(x) = 1 + 10^{-3} \exp\left(-\frac{(x-1/2)^2}{0.001}\right)$, and $v(x) = 0$. The number of points is 1000, the CFL number was 0.3, and the simulation was carried on to time $t = 0.1$.

(Equation (65)) yields

$$L_1^{1o} = \left(1 - e^{-\frac{1}{2}(1-\xi)k^2\Delta x t v}\right)\frac{k}{\pi}\int_0^{\pi/k} \sin(kx)dx \\ = \frac{2}{\pi}\left[1 - \exp\left(-\frac{1}{2}(1-\xi)k^2\Delta x t v\right)\right] \quad . \tag{66}$$

We assumed that the amplitude was positive so that the analytic solution would always be larger than the numeric, and thus the absolute value can be dropped. In principle, the integration should be carried out in the range $[0, 2\pi/k]$ (i.e., over one cycle), but because of the symmetry, it is sufficient to integrate over the range $[0, \pi/k]$. A similar calculation can be performed for the second-order scheme. Again, we start out with a pure Fourier mode as initial

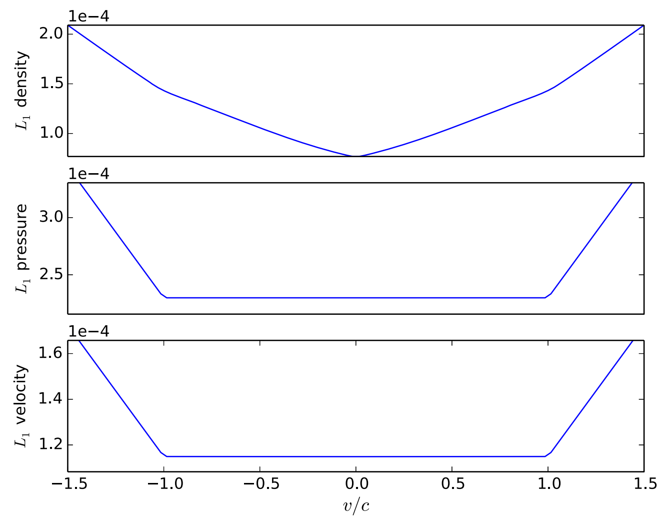

Figure 10. L_1 measure of the difference between analytic and numerical solutions as a function of the Mach number for the second-order scheme. The problem considered here is the same as in Figure 9.

conditions, $y(x) = \exp(Ikx)$. We obtain two profiles at a later time t: once using the exact filter $\exp(-Iktv)$ and a second time using the filter for the second-order scheme, Taylor expanded about $\Delta x \to 0$ (Equation (64)). In this case, the numeric solution lags behind the analytic solution:

$$L_1^{2o} = \frac{k}{2\pi}\int_0^{2\pi/k} \left|\sin(k(x - vt)) - \sin\left(k(x - v_n t)\right)\right| dx \\ = \frac{4}{\pi}\left|\sin\left(kt(v - v_n)\right)\right|, \tag{67}$$

where v denotes the drift velocity, and $v_n = v\left(\frac{1}{6}\left(1 - \xi^2\right)k^2\Delta x^2\right)$ is the numeric velocity. In contrast to the monotonous behavior of the first-order scheme, the L_1 of the second order oscillates because the lag between the waves increases until the phase difference is 2π. At that point, the error drops to zero, and the cycle repeats itself. The reason for this counterintuitive behavior is that this is based on a Taylor expansion for small t (Equation (64)). At larger values of t this approximation no longer holds, and one must resort to the complete expression for the second-order filter (Equation (63)).

In order to demonstrate the effect of drift velocity on the accuracy of a finite difference scheme, we performed the following test. We used the same initial conditions for the perturbations $\left(\delta\rho = 0,\ \delta p = 10^{-3}\exp\left(-\frac{(x-1/2)^2}{0.001}\right),\ \delta v = 0\right)$ and changed the drift velocity. For every velocity we advanced the hydrodynamic profiles to a time $t = 0.1$ using the analytic formalism described above and compared the result to the analytic profiles using the L_1 metric. The results for a first-order scheme are presented in Figure 9. The minima occur whenever one of the wave speeds becomes zero. Because the velocity and pressure propagate only through sound waves, their minima occur at $v = \pm c$. In the case of density, the dominant contribution is from the entropy wave, and a minimum only occurs at $v = 0$.

The same behavior recurs in second-order schemes, as can be seen in Figure 10. The reason for the plateau in the range $v \in [-c, c]$ is that the errors from the left and right sound waves exactly balance each other.

These results show us that, in general, a Lagrangian simulation will not always give better results than a Eulerian simulation. However, in the case of a highly supersonic flow, we expect that the error in the Lagrangian simulation would be smaller than in the Eulerian simulation. This is because the error does does not always grow while the velocity relative to the grid is in the range $[-c, c]$, but it always grows as the speed gets farther away from this range.

8. CONCLUSION

We presented our version of a hydrodynamic code on a moving Voronoi mesh. This code is similar to AREPO, with several important exceptions of a few implementation details. Our code, in its current initial form, still lacks some features available in AREPO. These features include three-dimensional geometry and individual time steps.

With our new code, we explored the question whether a simulation based on a moving mesh gives better results than static mesh. In our array of tests, a Lagrangian grid tends to give better results than a Eulerian grid. However, a more detailed one-dimensional analysis reveals some scenarios where a Eulerian grid would surpass a Lagrangian grid. The same analysis also revealed that in the case of highly supersonic flow a Lagrangian simulation would be more accurate.

Comparing the different time-advancement schemes between the codes shows that for purely hydrodynamic problems, AREPO's scheme tends to give slightly better results for small perturbations, and for external sources our time-advancement scheme gives better results for pressure and density, and AREPO does better for the velocity.

Our code is publicly available at https://code.google.com/p/huji-rich/. The open-source nature of our code allows other users to both reproduce the results presented here and run the code for their own calculations. Our code is built in a modular, object-oriented fashion to allow other users to incorporate new physics with ease.

We thank Omer Bromberg and Orly Gnat, who wrote the HLLC solver. We also thank Jonathan Granot, Udi Nakar, and Tsvi Piran for helpful discussions. E.S. is supported by an Ilan Ramon grant from the Israeli Ministry of Science. This research is supported in part by ISF, ISA, iCORE grants, and a Packard Fellowship.

APPENDIX A
ACOUSTIC WAVES

In Figure A1, we show the convergence curve for the acoustic waves test.

APPENDIX B
SIMPLE WAVES

In Figure B1, we show the convergence curve for the simple waves test.

APPENDIX C
SHOCK TUBE

In Figure C1, we show the convergence curve for the shock tube test.

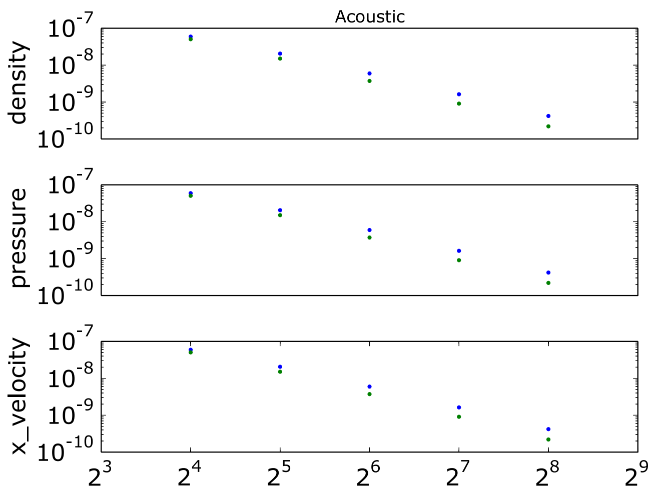

Figure A1. Variation of L_1 norm with resolution for the acoustic waves test. Blue denotes the time-centered flux (RICH), and green the extrapolated flux (AREPO).

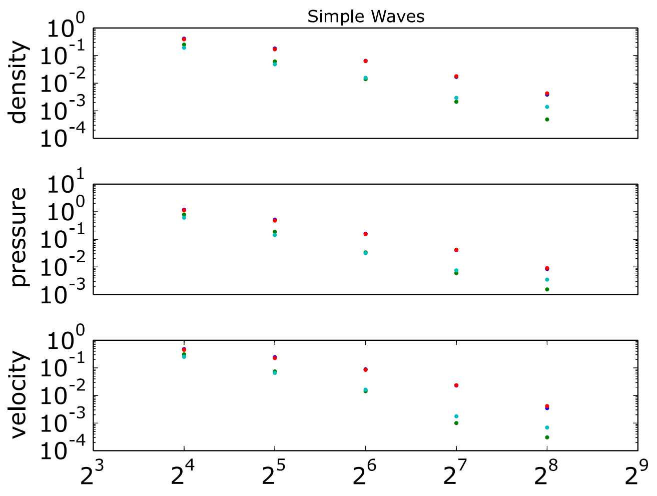

Figure B1. Variation of the L_1 norm with resolution for the simple waves test. Blue denotes the time-centered flux (RICH) with Eulerian grid, green the time-centered flux (RICH) with Lagrangian grid, red the extrapolated flux (AREPO) with Eulerian grid, and cyan the extrapolated flux (AREPO) with Lagrangian grid.

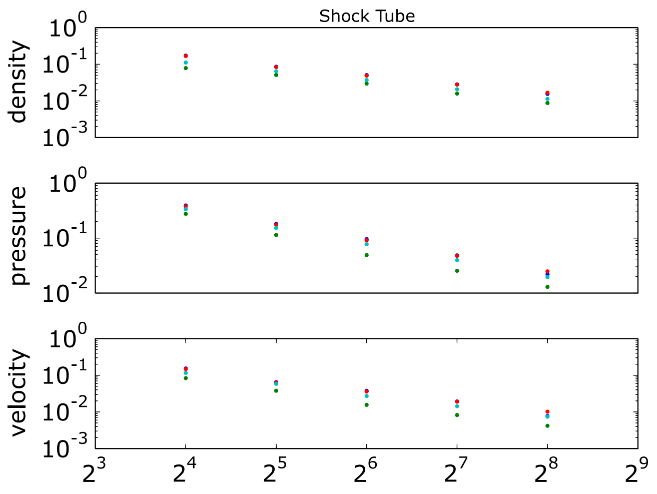

Figure C1. Variation of the L_1 norm with resolution for the shock tube test. Blue denotes the time-centered flux (RICH) with Eulerian grid, green the time-centered flux (RICH) with Lagrangian grid, red the extrapolated flux (AREPO) with Eulerian grid, and cyan the extrapolated flux (AREPO) with Lagrangian grid.

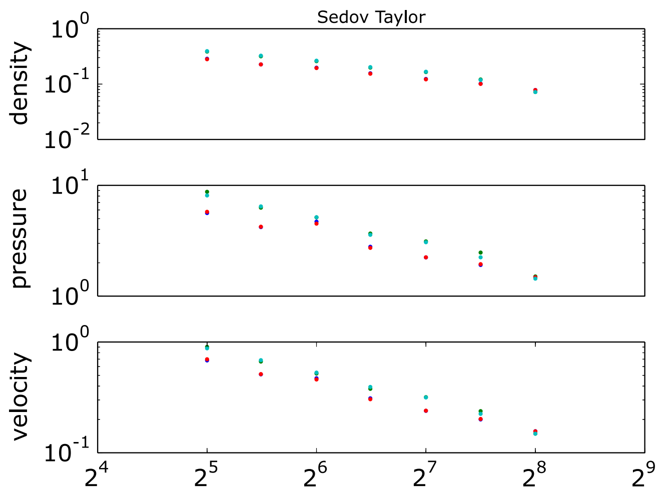

Figure D1. Variation of the L_1 norm with resolution for the Sedov Taylor test. Blue denotes the time-centered flux (RICH) with Eulerian grid, green the time-centered flux (RICH) with Lagrangian grid, red the extrapolated flux (AREPO) with Eulerian grid, and cyan the extrapolated flux (AREPO) with Lagrangian grid.

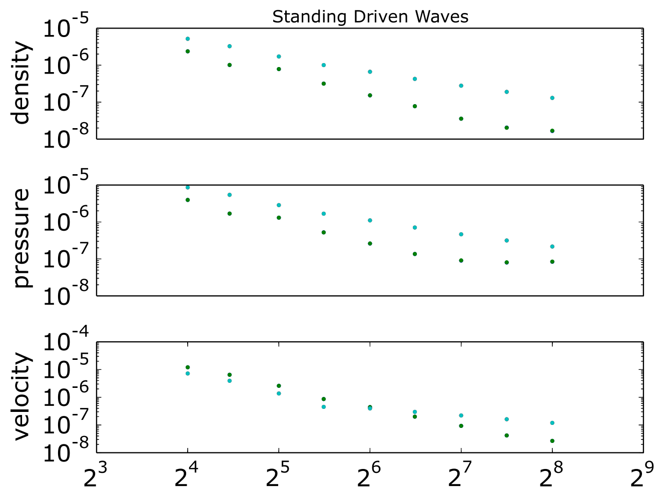

Figure E1. Variation of the L_1 norm with resolution for the standing driven waves test. Green denotes the time-centered flux (RICH), and cyan the extrapolated flux (AREPO).

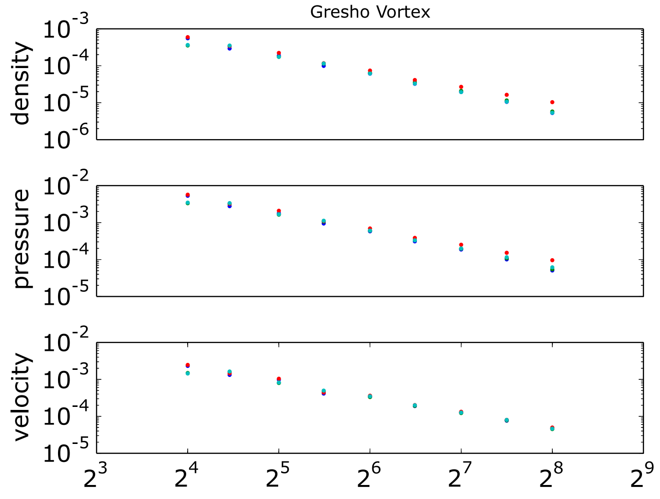

Figure F1. Variation of the L_1 norm with resolution for the Gresho vortex test. Blue denotes the time-centered flux (RICH) with Eulerian grid, green the time-centered flux (RICH) with Lagrangian grid, red the extrapolated flux (AREPO) with Eulerian grid, and cyan the extrapolated flux (AREPO) with Lagrangian grid.

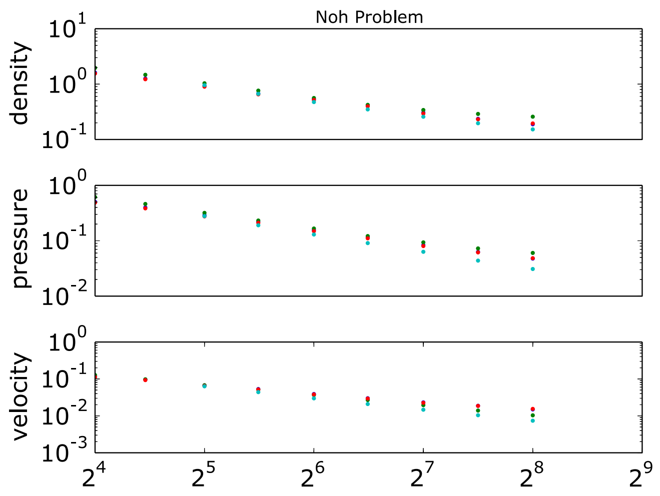

Figure G1. Variation of the L_1 norm with resolution for the Noh problem test. Blue denotes the time-centered flux (RICH) with Eulerian grid, green the time-centered flux (RICH) with Lagrangian grid, red the extrapolated flux (AREPO) with Eulerian grid, and cyan the extrapolated flux (AREPO) with Lagrangian grid.

APPENDIX D
SEDOV TAYLOR

In Figure D1, we show the convergence curve for the Sedov-Taylor test.

APPENDIX E
STANDING DRIVEN WAVES

In Figure E1, we show the convergence curve for the Standing Driven waves test.

APPENDIX F
GRESHO VORTEX

In Figure F1, we show the convergence curve for the Gresho vortex test.

APPENDIX G
NOH PROBLEM

In Figure G1, we show the convergence curve for the Noh test.

REFERENCES

Chandrasekhar, S. 1961, Hydrodynamic and Hydromagnetic Stability (New York: Dover)
Colella, P., Graves, D. T., Keen, B. J., & Modiano, D. 2006, JCoPh, 211, 347
Duffell, P. C., & MacFadyen, A. I. 2011, ApJS, 197, 15
Fryxell, B., Olson, K., Ricker, P., et al. 2000, ApJS, 131, 273
Godunov, S. 1961, Uspehi mat. Nauk, 16, 171
Hirt, C. W., Amsden, A. A., & Cook, J. L. 1974, JCoPh, 14, 227
Kravtsov, A. V., Klypin, A. A., & Khokhlov, A. M. 1997, ApJS, 111, 73
Landau, L. D., & Lifshitz, E. M. 1987, in Course of Theoretical Physics, Vol. 6, ed. L. D. Landau, & E. M. Lifshitz (2nd ed.; Portsmouth, NH: Butterworth-Heinemann)
Ledoux, H. 2007, in ISVD'07, 4th International Symposium on Voronoi Diagrams in Science and Engineering (Los Alamitos, CA: IEEE Computer Society Press), 117
Mignone, A., Bodo, G., Massaglia, S., et al. 2007, ApJS, 170, 228
Munoz, D. J. 2013, PhD Thesis, Univ. Harvard
Noh, W. F. 1987, JCoPh, 72, 78

Richtmyer, R. D., & Morton, K. W. 1994, Difference Methods for Initial-Value Problems (Malabar, FL: Krieger)
Shewchuk, J. R. 1996, Discrete Comput. Geom., 18, 305
Springel, V. 2010, MNRAS, 401, 791
Steinberg, E., Yalinewich, A., Sari, R., & Duffell, P. 2014, arXiv:1408.3196
Stone, J. M., Gardiner, T. A., Teuben, P., Hawley, J. F., & Simon, J. B. 2008, ApJS, 178, 137
Stone, J. M., & Norman, M. L. 1992, ApJS, 80, 753
Taflove, A., & Hagness, S. C. 2005, Computational Electrodynamics: The Finite-Difference Time-Domain Method (3rd ed.; Boston, MA: Artech House Publishers)
Toro, E. F. 1999, Riemann Solvers and Numerical Methods for Fluid Dynamics (Berlin: Springer)
van Leer, B. 1979, JCoPh, 32, 101
Zhang, W., & MacFadyen, A. I. 2006, ApJS, 164, 255
Ziegler, U. 1998, CoPhC, 109, 111