The Astrophysical Journal, 867:112 (18pp), 2018 November 10 https://doi.org/10.3847/1538-4357/aae5f9

Relativistic Hydrodynamics with Wavelets

Jackson DeBuhr[1], Bo Zhang[1], Matthew Anderson[2], David Neilsen[3], Eric W. Hirschmann[3], Temistocle Grenga[4], and Samuel Paolucci[5]
[1] School of Informatics, Computing, and Engineering, Indiana University, Bloomington, IN 47404, USA; debuhj@gmail.com, zhang416@indiana.edu
[2] Intelligent Systems Engineering Department, School of Informatics, Computing, and Engineering, Indiana University, Bloomington, IN 47404, USA; andersmw@indiana.edu
[3] Department of Physics and Astronomy, Brigham Young University, Provo, UT 84602, USA; david.neilsen@byu.edu, ehirsch@byu.edu
[4] Institute for Technical Combustion, RWTH Aachen University, Aachen, D-52056, Germany; t.grenga@itv.rwth-aachen.de
[5] Department of Aerospace and Mechanical Engineering, University of Notre Dame, South Bend, IN 46556, USA; paolucci@nd.edu
Received 2018 January 8; revised 2018 September 24; accepted 2018 September 26; published 2018 November 6

Abstract

Methods to solve the relativistic hydrodynamic equations are important in a large number of astrophysical simulations, which may be very dynamic and involve multiscale features. This requires computational methods that are highly adaptive and capable of automatically resolving numerous localized features and instabilities that emerge across the computational domain and over many temporal scales. While this has been historically accomplished with adaptive-mesh-refinement-based methods, alternatives using wavelet bases and the wavelet transformation have recently achieved significant success in adaptive representation for advanced engineering applications. The current work presents a new method, extending the wavelet adaptive multiresolution representation method, for the integration of the relativistic hydrodynamic equations using iterated interpolating wavelets and introduces a highly adaptive implementation for multidimensional simulation. The wavelet coefficients provide a direct measure of the local approximation error for the solution and place collocation points that naturally adapt to the fluid flow while providing good conservation of fluid quantities. The resulting implementation, OAHU, is applied to a series of demanding 1D and 2D problems that explore high Lorentz factor outflows and the formation of several instabilities, including the Kelvin–Helmholtz instability and the Rayleigh–Taylor instability.

Key words: gamma-ray burst: general – hydrodynamics – methods: numerical – relativistic processes

1. Introduction

Relativistic fluids are used to model a variety of systems in high-energy astrophysics, such as neutron stars, accretion onto compact objects, supernovae, and gamma-ray burst outflows. Consequently, methods to solve the relativistic hydrodynamic equations are important in a large number of astrophysics simulations. Because of the many physical length scales present when simulating astrophysical phenomena, these methods must also be highly adaptive and capable of automatically resolving many localized emerging features and instabilities throughout the computational domain across many temporal scales. For Eulerian fluid methods, this has been historically accomplished with adaptive mesh refinement (AMR) based methods (Berger & Oliger 1984; Anderson et al. 2006; Zhang & MacFadyen 2006). Other approaches include smoothed particle hydrodynamics (Rosswog 2009, 2010, 2015; Springel 2010a), solving the fluid equations on a moving Voronoi mesh (Springel 2010b; Duffell & MacFadyen 2011), and meshless Lagrangian methods (Hopkins 2015; Hopkins & Raives 2016). An alternative, the so-called wavelet adaptive multiresolution representation (WAMR) method, based on wavelets and the fast wavelet transform, has recently achieved significant success in adaptive representation for advanced engineering applications (Paolucci et al. 2014a, 2014b). This has inspired work to investigate their application in relativistic hydrodynamics. The current work presents a new method for the integration of the relativistic hydrodynamic equations using iterated interpolating wavelets and introduces a highly adaptive implementation for multidimensional simulations called OAHU.

We are developing the OAHU code to address the challenge of simulating multiscale astrophysical phenomena. A key component of this code is that we combine a robust high-resolution shock-capturing method with an unstructured grid of collocation points that conforms to the features of the solution provided by the WAMR method. This grid adaptivity is realized by expanding functions in a wavelet basis and adding refinement only where the solution has small-scale features.

Wavelets allow one to represent a function in terms of a set of basis functions that are localized both spatially and with respect to scale. In comparison, spectral bases are infinitely differentiable but have global support; basis functions used in finite-difference or finite-element methods have small compact support but poor continuity properties. Wavelets with compact support have been applied to the solutions of elliptic, parabolic, and hyperbolic partial differential equations (PDEs; Glowinski et al. 1990; Beylkin 1992; Latto & Tenenbaum 1990; Qian & Weiss 1993a, 1993b; Dahmen et al. 1997; Beylkin & Coult 1998; Holmström 1999; Alpert et al. 2002; Alam et al. 2006; Urban 2009; Chegini & Stevenson 2011). Wavelets have also been applied to the solutions of integral equations (Alpert et al. 1993). We note that when applied to nonlinear equations, some of these previous (e.g., wavelet-Galerkin) methods will map the space of wavelet coefficients onto the physical space and there compute the nonlinear terms. They then project that result back to the wavelet coefficient space using analytical quadrature or numerical integration. Our approach is rather to combine collocation methods with wavelets, thus allowing us to operate in a single space (Vasilyev et al. 1995; Bertoluzza & Naldi 1996; Vasilyev & Paolucci 1996, 1997; Vasilyev & Bowman 2000; Regele & Vasilyev 2009).

In astronomy, wavelets have seen extensive use in analysis tasks, from classifying transients (Powell et al. 2015; Varughese et al. 2015), to image processing (Mertens & Lobanov 2015), to finding solutions to nonlinear initial value problems (Kazemi Nasab et al. 2015). However, they have not, to our knowledge, seen much use in solving PDEs in astrophysics.

This paper reports on an initial version of OAHU that concentrates on initial tests of the fluid equations as solved on an adaptive wavelet grid and demonstrates the viability and robustness of this method for highly relativistic astrophysical problems. The organization of this paper is as follows. In Section 2 we briefly describe the relativistic fluid equations and the numerical methods used to solve them. Section 3 presents 1D tests of the resulting scheme. In Section 4 we present the results of applying our method to 2D problems. Section 5 presents a stringent test of our method as applied to a relativistic outflow that develops Rayleigh–Taylor generated turbulence. Finally, in Section 6 we summarize results and make note of future work suggested by the method.

2. Methods

This section describes some of the numerical approaches and algorithms used in OAHU. The relativistic fluid equations are written in units where the speed of light is set to unity, $c = 1$. Repeated Greek indices sum over all spacetime coordinates, 0, 1, 2, 3, and repeated Latin indices sum over the spatial coordinates, 1, 2, 3.

2.1. Relativistic Hydrodynamics

In general relativity the spacetime geometry is described by a metric tensor $g_{\mu\nu}$, and we write the line element in Arnowitt Deser Misner (ADM) form as

$$ds^2 = g_{\mu\nu}\, dx^\mu dx^\nu \tag{1}$$

$$= (-\alpha^2 + \beta_i\beta^i)dt^2 + 2\beta_i\, dt dx^i + \gamma_{ij}\, dx^i dx^j. \tag{2}$$

Here α and β^i are functions that specify the coordinates and γ_{ij} is the 3-metric on spacelike hypersurfaces. While our code is written for a completely generic spacetime, the tests presented in this paper are all performed in flat spacetime. To simplify the presentation, we write the equations in special relativity (i.e., flat spacetime) in general curvilinear coordinates, and we set $\alpha = 1$ and $\beta^i = 0$. In Cartesian coordinates, the flat space metric is the identity $\gamma_{ij} = \mathrm{diag}(1, 1, 1)$, but γ_{ij} is generally a function of the curvilinear coordinates.

A perfect fluid in special relativity is described by a stress-energy tensor of the form

$$T_{\mu\nu} = hu_\mu u_\nu + Pg_{\mu\nu}, \tag{3}$$

where h is the total enthalpy of the fluid

$$h = \rho(1 + \varepsilon) + P. \tag{4}$$

The fluid variables ρ, ε, u^μ, and P are the rest mass density, the specific internal energy, the four-velocity, and the pressure of the fluid, respectively. Once an equation of state of the form $P = P(\rho, \varepsilon)$ is adopted, the equations determining the matter dynamics are obtained from the conservation law $\nabla_\mu T^\mu{}_\nu = 0$ and the conservation of baryons $\nabla_\mu(\rho u^\mu) = 0$.

We introduce the three-velocity of the fluid, v^i, and the Lorentz factor W by writing the four-velocity as

$$u^\mu = (W,\, Wv^i)^T. \tag{5}$$

The four-velocity has a fixed magnitude $u_\mu u^\mu = -1$, which gives the familiar relation between W and v^i,

$$W^2 = \frac{1}{1 - v_i v^i}. \tag{6}$$

We introduce a set of *conserved* variables that represent conserved fluid quantities

$$D \equiv \rho W \tag{7}$$

$$S_i \equiv hW^2 v_i \tag{8}$$

$$\tau \equiv hW^2 - P - \rho W. \tag{9}$$

These quantities correspond in the Newtonian limit to the rest mass density, the momentum, and the kinetic energy of the fluid, respectively. When using curvilinear coordinates, it is convenient to work with the *densitized* variables, which include the geometric factor $\sqrt{\gamma}$:

$$\tilde{D} \equiv \sqrt{\gamma} D \tag{10}$$

$$\tilde{S}_i \equiv \sqrt{\gamma} S_i \tag{11}$$

$$\tilde{\tau} \equiv \sqrt{\gamma}\tau, \tag{12}$$

where $\gamma = \det \gamma_{ij}$. In terms of these fluid variables, the relativistic fluid equations are

$$\partial_t \tilde{D} + \partial_i(\tilde{D}v^i) = 0 \tag{13}$$

$$\partial_t \tilde{S}_j + \partial_i(v^i \tilde{S}_j + \sqrt{\gamma} P \gamma^i{}_j) = {}^3\Gamma^i{}_{kj}(v^k \tilde{S}_i + \sqrt{\gamma} P \gamma^k{}_i) \tag{14}$$

$$\partial_t \tilde{\tau} + \partial_i(\tilde{S}^i - \tilde{D}v^i) = 0, \tag{15}$$

where ${}^3\Gamma^i{}_{jk}$ are the Christoffel symbols associated with the spatial metric γ_{ij}.

The fluid equations of motion can be written in balance law form

$$\partial_t \boldsymbol{u} + \partial_i \boldsymbol{f}^i(\boldsymbol{u}) = \boldsymbol{s}(\boldsymbol{u}), \tag{16}$$

where $\boldsymbol{u}$ is the state vector of conserved variables and $\boldsymbol{f}^i$ are the fluxes

$$\boldsymbol{u} = \begin{pmatrix} \tilde{D} \\ \tilde{S}_i \\ \tilde{\tau} \end{pmatrix}, \quad \boldsymbol{f}^i = \begin{pmatrix} \tilde{D}v^i \\ v^i \tilde{S}_j + \sqrt{\gamma} P \gamma^i{}_j \\ \tilde{S}^i - \tilde{D}v^i \end{pmatrix}. \tag{17}$$

The fluid equations in curvilinear coordinates have geometric source terms, which are included in $\boldsymbol{s}$.

Finally, the system of equations is closed with an equation of state. We use the Γ-law equation of state

$$P = (\Gamma - 1)\rho\, \varepsilon, \tag{18}$$

where Γ is the adiabatic constant.

2.2. Sparse Field Representation

In this section we describe the construction of a sparse, adaptive representation of our functions or fields. Two of the essential ingredients in this construction include the iterative interpolation introduced by Deslauriers & Dubuc (1989) and the wavelet representation of Donoho (1992). We will describe

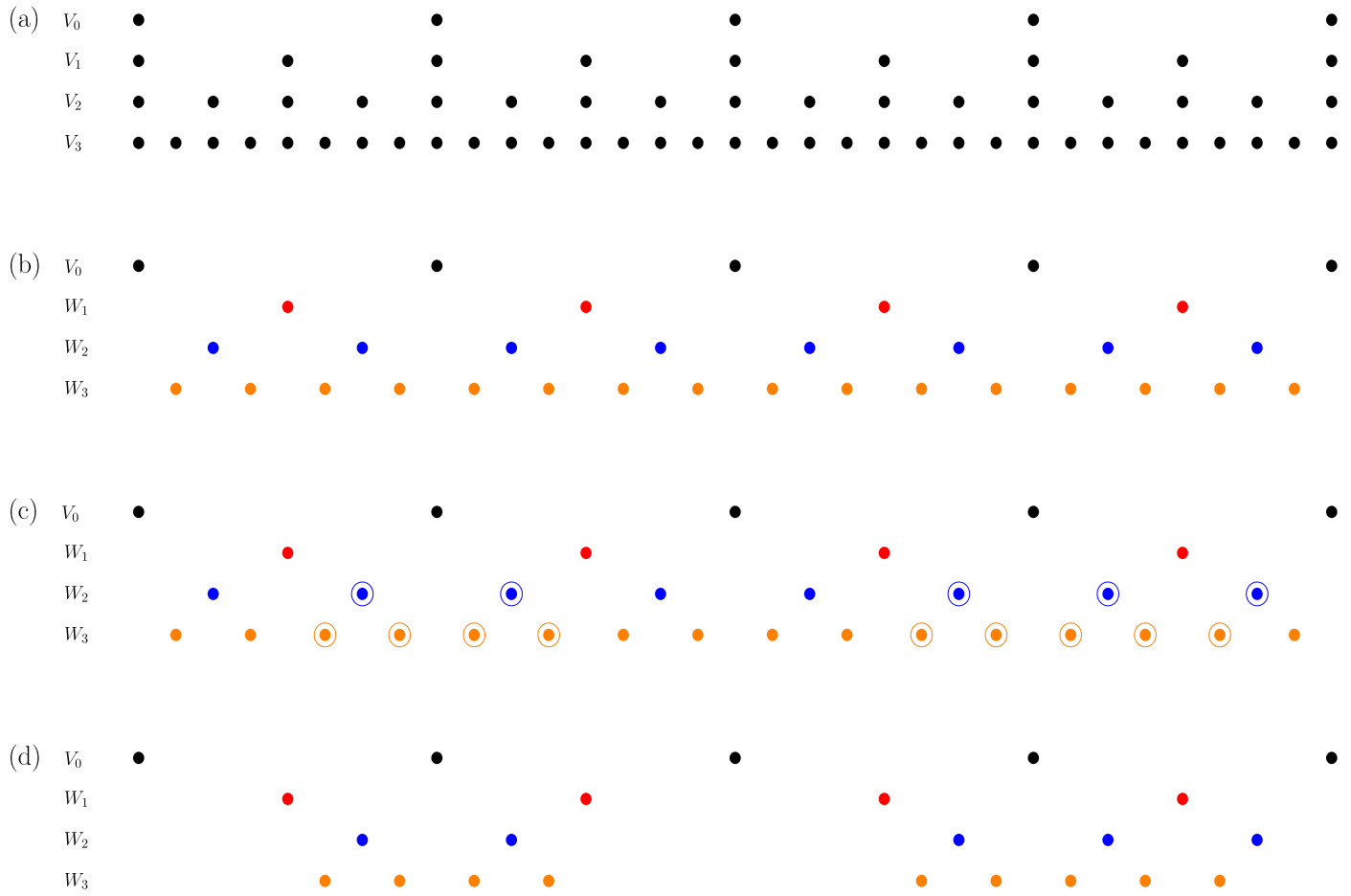

Figure 1. (a) Nested grids, V_i; in particular, $V_0 \subset V_1 \subset V_2 \subset V_3$. (b)$V_0$ and the complementary spaces of V_j (for $j = 1, 2, 3$), called W_j. The process of thresholding first calculates wavelet coefficients based on the absolute difference between the field on level $j+1$ (W_{j+1}) and the interpolated value from the previous level j (W_j). Those points for which these wavelet coefficients exceed some allowed tolerance, ϵ, are kept, while the remainder are discarded. (c) Examples of such points for which the wavelet coefficients exceed ϵ are given by open circles surrounding the grid points. (d) Example of the final sparse grid on thresholding as the circled points are kept and the remainder discarded. Note that the grid points belonging to V_0 and W_1 are always kept and are always part of the final grid.

both. Our presentation largely follows that in Holmström (1999). Additional details regarding the WAMR method can be found in Wirasaet & Paolucci (2005), Rastigejev & Paolucci (2006), and Paolucci et al. (2014b). For simplicity we treat the 1D case in this section. The generalization to higher dimensions is considered in the next section.

We begin by defining a base grid in the familiar way. For a 1D domain of size L, we introduce $N+1$ evenly spaced points with spacing $\Delta x = L/N$. This base grid we refer to as V_0. The grid points on the base grid are given coordinate values $x^{0,k} = k\Delta x$, where k is an integer that indexes the points on the grid.

We construct a refined grid, V_1, in an analogous manner but now with $2N+1$ points and coordinate locations $x^{1,k} = k\Delta x/2$. Clearly, we can continue such a construction of ever more refined grids, which will result in a set of nested grids given by

$$V_j = \{x^{j,k} : x^{j,k} = 2^{-j}k\Delta x\}. \tag{19}$$

Note that the integer k now indexes points within the various grid levels. Notice that all the grid points in grid V_j will appear in all higher-level grids V_l (where $l > j$). Grid points with even k at level j will likewise be in the grid at level $j-1$. Figure 1 gives an illustration of the grid.

If we know that the values of a function or field, f, on the grid at level j are $f^{j,k} = f(x^{j,k})$, we can extend these values to higher levels of the grid by exploiting this repetition of the grid points from a lower-level grid into a higher-level grid. For those points in V_{j+1} that are also in V_j, we just copy the value from the coarser grid: $f^{j+1,2k} = f^{j,k}$. The previous is also the means by which the field values can be restricted to coarser levels: points at coarser levels have values copied from finer levels.

For grid points in V_{j+1} not already present in V_j, we use interpolation from known values of V_j to approximate $f^{j+1,2k+1}$. In particular, we take the nearest p field values from grid V_j and interpolate using

$$\tilde{f}^{j+1,2k+1} = \sum_m h_{j,m}^{j+1,2k+1} f^{j,m}, \tag{20}$$

where we use the tilde to indicate that these are interpolated values. In the current work, we take the interpolation coefficients, $h_{j,k}^{j+1,2k+1}$, to be those for Lagrange interpolation from level j to level $j+1$. In practice, for a given k, only a small number of these coefficients are nonzero. Choosing the

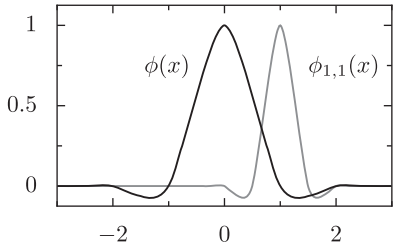

Figure 2. Fundamental solution of the iterated interpolation for $p = 4$, $\phi(x)$ (black) and a basis element in level one, $\phi_{1,1}(x)$ (gray). Both the scaling and wavelet functions are scaled, translated versions of the fundamental solution.

$p = 4$ closest points results in

$$\tilde{f}^{j+1,2k+1} = -\frac{1}{16}f^{j,k-1} + \frac{9}{16}f^{j,k} + \frac{9}{16}f^{j,k+1} - \frac{1}{16}f^{j,k+2}. \quad (21)$$

Strictly speaking, the previous expression applies only in the interior of the grid, where the grid points on V_j are symmetric around the interpolated point on V_{j+1}. Near the boundaries, the interpolation stencil is no longer symmetric around the refined point, and the coefficients in the sum will be different, but are still those appropriate to Lagrange interpolation.

As described, all field values can thus be provided on grid V_{j+1}. The same procedure (copy and interpolation) can then be iterated to advance the field to V_{j+2}. In this way, any level of refinement can be achieved starting from our initial set of known field values, e.g., $f^{j,k}$. More generally, when this iterated interpolation is allowed to continue from an initial set or sequence of values *ad infinitum*, it can be shown that a continuous function on the interval $[0, L]$ is the result. Details about the regularity and convergence of such iterated interpolation functions can be found in Donoho (1992).

This iterated interpolation process is linear and can produce functions, which suggests a natural set of basis functions for iterated interpolating functions. These can be formed from a very simple set of sequences composed mostly of zeros. In particular, consider a sequence of field values on level j that are zero at every grid point except one. At the sole exception, the location of which we will call $x^{j,l}$, the field value is 1. We can represent such a sequence as $\phi_{j,k}(x^{j,l}) = \delta^l_k$, a so-called Kronecker sequence. Now perform the iterated interpolation process with this sequence. The function that results, call it $\phi_{j,k}(x)$, has a number of properties, among which is the two-scale relation:

$$\phi_{j,k}(x) = \sum_l c^l \phi_{j+1,l}(x). \quad (22)$$

A step in the interpolation will produce a sequence on V_{j+1}, which can be written as a linear combination of the Kronecker sequences on V_{j+1}. Indeed, the coefficients c^l come straightforwardly out of the interpolation itself. For $p = 4$ these are $c^l = \{..., 0, -1/16, 0, 9/16, 1, 9/16, 0, -1/16, 0, ...\}$. Note that each of these functions is a scaled, translated version of a single function $\phi(x)$, illustrated in Figure 2, called the fundamental solution of the interpolation:

$$\phi_{j,k}(x) = \phi(2^j x/\Delta x - k). \quad (23)$$

This property of the fundamental solution of the iterated interpolation also defines this function as the autocorrelation of the Daubechies scaling function (Saito & Beylkin 1993).

The scaled and translated versions of the fundamental solution, often referred to as scaling functions, can now be used to create bases for each level of the grid separately. Note, however, that the two-scale relation prevents the full set from being a basis on the entire set of collocation points across all grids. In particular, for each location in the grid, $x^{j,k}$, there is one associated scaling function. As already remarked on, certain locations are represented on multiple levels, for example, $x^{j,k} = x^{j+m,2^m k}$ for all $m > 0$. In order to form a basis for the full grid composed of a coarse level and all refined levels, we have to circumvent this redundancy. To that end, we define a new space of grid points W_{j+1}, for $j > 0$, as the complementary space of V_j, such that

$$V_{j+1} = V_j \oplus W_{j+1}. \quad (24)$$

Clearly, W_{j+1} are those points in V_{j+1} that are not in V_j, or

$$W_j = \{x^{j,k} : x^{j,k} = 2^{-j}k\Delta x,\ k \text{ odd}\}. \quad (25)$$

Note that the W_j grids are the V_j grids with the points from coarser levels removed (see the colored points in Figure 1). On defining the set of grids $\{V_0, W_j\}$, we now have each point represented exactly once. Forming a basis for these points is achieved by taking those scaling functions, $\phi_{j,k}(x)$, that correspond to the points in the base grid, as well as to the points in the modified refined grids, W_j. With this basis, we can now represent a field, u, as follows:

$$u(x) = \sum_{k\in S_0} u^{0,k}\phi_{0,k}(x) + \sum_{j=1}^{\infty}\sum_{k\in S_j} d^{j,k}\psi_{j,k}(x), \quad (26)$$

where $S_0 = \{0, 1, ..., N\}$ is the index set for grid V_0, $S_j = \{1, 3, ..., 2^{j+1}N - 1\}$ is the index set for grid W_j, and $u^{0,k}$ and $d^{j,k}$ are expansion coefficients. The functions $\psi_{j,k}(x)$ are defined in terms of the scaling functions as

$$\psi_{j,k}(x) = \phi_{j,k}(x) \quad (27)$$

and are referred to as wavelet functions.

Equation (26) is the interpolating wavelet expansion of the field. Intuitively, the expansion contains the coarse picture (level 0) and refinements of that picture at successively finer levels. The expansion coefficients $u^{0,k}$ are just the field values at the base level points: $u^{0,k} = u(x^{0,k})$. The coefficients $d^{j,k}$ are referred to as *wavelet coefficients*. At any given grid point, they are found by comparing the interpolation from the previous level to the field value, $u^{j,k} = u(x^{j,k})$. In particular, if we denote the interpolated value coming from level j at a level $j + 1$ grid point, $\tilde{u}^{j+1,k} = P(x^{j+1,k}, j)$, then the wavelet coefficient is simply the difference between the two:

$$d^{j,k} = u^{j,k} - P(x^{j,k}, j - 1). \quad (28)$$

Intuitively, the wavelet coefficient measures the failure of the field to be the interpolation from the previous level. This expression also defines the fast wavelet transformation, which begins with field values on the multilevel grid and produces wavelet coefficients. The transformation can easily be inverted

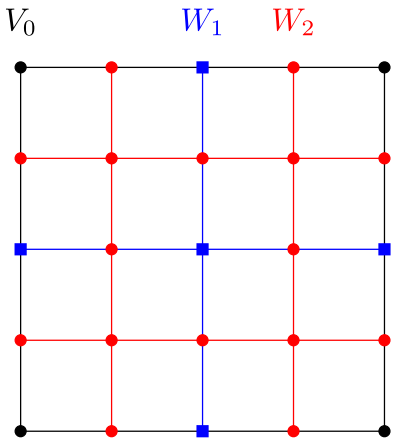

Figure 3. Slice of the 3D grid. The black circles are those in the base grid, the blue squares belong to level 1, and the red circles belong to level 2. Notice that some points at level 1 line up with points at level 0 and that some points at level 2 line up with points at level 1.

by rearranging the equation and computing field values given the wavelet coefficients.

Note that we have two descriptions of the field on the full, multilevel grid. One is composed of the set of values $\{u^{j,k}\}$ and is sometimes referred to as the point representation in the literature. The second, called the wavelet representation, is made up of the set of values $\{u^{0,k}, d^{j,k}\}$. Both representations can be made sparse via a process of thresholding. If we start with the wavelet representation and pick a particular threshold, ϵ, we can form the so-called sparse wavelet representation by removing those points from the grid whose wavelet coefficients are below some threshold: $|d^{j,k}| < \epsilon$ (see Figure 1). This is equivalent to discarding those points from the grid that are well approximated by interpolation (Wirasaet & Paolucci 2005; Wirasaet 2007; Paolucci et al. 2014a, 2014b). This naturally pares down the number of points in the grid and introduces an a priori error bound on the representation of the field (Donoho 1992). We will refer to those points whose values are kept as "essential" points. In practice, level 0 points are always kept and are therefore always essential. The field values at these essential points form what has been called the sparse point representation.

2.3. Higher Dimensions

In multiple dimensions, the construction is not much more involved. The basis functions are taken to be the products of the 1D functions:

$$\phi_{j,\boldsymbol{k}}(x, y, z) = \phi_{j,k_x}(x)\,\phi_{j,k_y}(y)\,\phi_{j,k_z}(z). \tag{29}$$

In the previous, $\boldsymbol{k} = (k_x, k_y, k_z)$ is the set of three indices required to label a 3D grid (see Figure 3). Another way to construct these functions is by interpolation in multiple dimensions. Depending on the location of the level $j+1$ point in the level j grid, this interpolation will involve p, p^2, or p^3 terms (see Figure 4). The wavelet coefficient for a point is again computed as the difference between the field value and the interpolated value; it is just that the interpolation will now contain more terms.

The rest of the method goes through as one might expect. The field is expanded in terms of these multidimensional basis

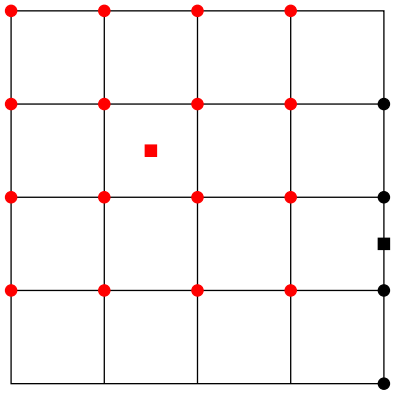

Figure 4. Portion of a slice of the 3D grid at level j. Shown are those points at level j (circles) that contribute to the wavelet transformation for the marked points at level $j+1$ (squares). Depending on the relative placement of the level $j+1$ points to the level j grid, the wavelet transformation will use p (black) or p^2 (red) points. In three dimensions this extends to cases that also use p^3 points.

functions:

$$u(x, y, z) = \sum_{\boldsymbol{k}} u^{0,\boldsymbol{k}}\phi_{0,\boldsymbol{k}}(x, y, z) + \sum_{j=1}^{\infty}\sum_{\boldsymbol{k}} d^{j,\boldsymbol{k}}\psi_{j,\boldsymbol{k}}(x, y, z). \tag{30}$$

Here the wavelet basis function is again a product of the 1D basis elements:

$$\psi_{j,\boldsymbol{k}}(x, y, z) = \phi_{j,\boldsymbol{k}}(x, y, z). \tag{31}$$

The sparse representation is formed by removing those points whose wavelet coefficients have a magnitude less than the prescribed error threshold, ϵ.

2.4. Conservation

It is possible to measure the conservation of the fields being evolved using the wavelet basis. In particular, the quantity

$$\int_{\Sigma} u(x)dx, \tag{32}$$

where Σ is the computational domain, is straightforward to compute using the standard expansion of the field in the wavelet basis. Making the substitution yields

$$\int_{\Sigma} u(x)dx = \sum_{k\in S_0} u^{0,k}\int_{\Sigma}\phi_{0,k}(x)dx + \sum_{j=1}\sum_{k\in S_j} d^{j,k}\int_{\Sigma}\psi_{j,k}(x)dx. \tag{33}$$

Given the integrals of the basis functions, we can easily compute the total amount of some quantity.

That each basis element is a scaled, translated version of the fundamental solution to the interpolation implies that, ignoring edges of the computational domain for the moment,

$$\int\phi_{j,k}(x)dx = \frac{\Delta x}{2^j}\int\phi(x)dx. \tag{34}$$

The prefactor takes into account the difference in spacing for the two functions. It is easy to show that the integral of $\phi(x)$ is

1. Thus,

$$\int \phi_{j,k}(x)dx = \frac{\Delta x}{2^j}. \tag{35}$$

We can use this in the expansion of the field to write

$$\int_\Sigma u(x)dx = \sum_{k\in S_0} u^{0,k}\Delta x + \sum_{j=1}\sum_{k\in S_j} d^{j,k}\frac{\Delta x}{2^j}. \tag{36}$$

This expression allows the monitoring of the conserved quantities during the simulation. In every case examined, the conservation is good to the level of the chosen ϵ.

Near the edges of the computational grid, the basis elements no longer have unit integrals. To compute these integrals, it is necessary to make use of the two-scale relation for the basis to compute partial integrals:

$$I_a^b \equiv \int_a^b \phi(x)dx, \tag{37}$$

where a, b are integers, one of which might be infinite. Some are simple owing to the symmetry of the basis (e.g., $I_0^\infty = 0.5$), but others require setting up a linear system using the two-scale relation. Once the set of partial integrals is computed, the basis elements near the edges are written as the sum of an extended basis that stretches past the edge of the computational domain. It is an exercise in algebra to show that each of the original basis elements modified by the edges can be written as a sum of these extended basis elements. The extended basis elements are again translated, scaled versions of the fundamental solution, so we can use the partial integrals for only the interval inside the original domain to compute the integral of the original basis elements. For the case of $p = 4$, with superscripts labeling the location in the grid, we find that

$$\begin{aligned} I_j^0 &= \frac{121}{360}\frac{\Delta x}{2^j}, \\ I_j^1 &= \frac{462}{360}\frac{\Delta x}{2^j}, \\ I_j^2 &= \frac{303}{360}\frac{\Delta x}{2^j}, \\ I_j^3 &= \frac{374}{360}\frac{\Delta x}{2^j}. \end{aligned} \tag{38}$$

Similar expressions hold at the largest k values.

In multiple dimensions, because the basis functions are simple products, the integral of the basis is just the product of the integrals in each direction.

2.5. Relativistic Hydrodynamics Using a Sparse Wavelet Representation

We use the sparse wavelet representation presented in Sections 2.2 and 2.3 to generate the unstructured grid for the computational domain. The grid has both a point representation of the function, i.e., function values defined at each point, and a wavelet representation given by the coefficients in the wavelet expansion. We could use either representation to construct a numerical scheme to solve the fluid equations. However, the wavelet representation would be very inefficient, as even multiplying two functions requires several operations of products and sums of the wavelet coefficients. On the other hand, the point representation allows us to use proven, conventional numerical methods for the relativistic fluid equations. Thus, we choose to build our numerical scheme on the point representation.

In the remaining part of Section 2 we outline the numerical methods we use for solving the relativistic fluid equations. This information is given to specify our particular numerical scheme, but the treatment is brief, as these methods have been widely used in the literature. The wavelet coefficients are used to adapt the grid and keep only those field values that are essential to capturing the structures in the fluid flow. They are also used to fill in field values that are not essential, but which are needed for some of the stencils in the fluid methods. But they are not used to advance the state of the fluid.

2.6. Fluid Methods

With our interest in relativistic fluid flows, we choose a numerical algorithm that satisfies the following conditions: (I) The fluid density and pressure can span several orders of magnitude with fluid speeds near the speed of light, so we choose a robust high-resolution shock-capturing method. (II) For optimal resolution we choose high-order reconstruction methods for the fluid variables, and we implement both PPM and MP5 reconstructions. (III) The calculation of characteristic variables is computationally intensive for relativistic fluids, especially for relativistic MHD, so we choose a central scheme with an approximate Riemann solver. (IV) When combining the fluid equations with other physical fields, such as the gravitational field of the Einstein equations, it is convenient to use a finite-difference fluid method.

In this section we assume a uniform grid of points labeled with an index, i.e., $x_j = x_{\rm min} + j\Delta x$, and a function evaluated at point x_j has the shorthand notation $f_j = f(x_j)$.

Our numerical method for solving the fluid equations is based on the finite-difference Convex ENO (CENO) scheme of Liu & Osher (1998) for conservation laws. For simplicity we present the method for a 1D problem. The extension to multiple dimensions is done by differencing each dimension in turn, as discussed in Section 2.7. We write the conservation law in semi-discrete form as

$$\frac{d\boldsymbol{u}_i}{dt} = -\frac{\Delta t}{\Delta x}(\hat{\boldsymbol{f}}_{i+1/2} - \hat{\boldsymbol{f}}_{i-1/2}), \tag{39}$$

where $\hat{\boldsymbol{f}}_{i+1/2}$ is a consistent numerical flux function

$$\hat{\boldsymbol{f}}_{i+1/2} = \hat{\boldsymbol{f}}(\boldsymbol{u}_{i-k}, \ldots, \boldsymbol{u}_{i+m}) \tag{40}$$

$$\boldsymbol{f}(\boldsymbol{u}) = \hat{\boldsymbol{f}}(\boldsymbol{u}, \ldots, \boldsymbol{u}). \tag{41}$$

Liu and Osher base the CENO method on the local Lax–Friedrichs (LLF) approximate Riemann solver, and they use an ENO interpolation scheme to calculate the numerical flux functions $\boldsymbol{f}_{i+1/2}$. In previous work we have found that the CENO scheme is too dissipative to reproduce the normal modes of neutron stars (Anderson et al. 2006), so we use the HLLE numerical flux (Harten et al. 1983; Einfeldt 1988) in place of LLF, and we use higher-order finite-volume reconstruction methods, such as PPM (Colella & Woodward 1984) and MP5 (Suresh & Huynh 1997).

The numerical flux $\hat{\boldsymbol{f}}_{i+1/2}$ requires the fluid state at $\boldsymbol{u}_{i+1/2} = \boldsymbol{u}(x_{i+1/2})$. We use the fluid variables near this point to reconstruct both left and right states at the midpoint, $\boldsymbol{u}^\ell_{i+1/2}$ and $\boldsymbol{u}^r_{i+1/2}$, respectively. The numerical flux can then be written

in terms of these new states as $\boldsymbol{f}_{i+1/2} = \boldsymbol{f}(\boldsymbol{u}^{\ell}_{i+1/2}, \boldsymbol{u}^{r}_{i+1/2})$. We have implemented piecewise linear (TVD) reconstruction, the piecewise parabolic method (PPM), and MP5 reconstruction. The MP5 reconstruction method usually gives superior results compared to the other methods, so we have used this reconstruction for all tests in this paper, except for Case IV below. As we use a central scheme for the approximate Riemann solver, we reconstruct each fluid variable separately. Moreover, given the difficulty of calculating primitive variables (ρ, v^i, P) from the conserved variables (D, S_i, τ), we reconstruct the primitive variables (ρ, v^i, P) and then calculate corresponding conserved variables.

The MP5 method is a polynomial reconstruction of the fluid state that preserves monotonicity (Suresh & Huynh 1997; Mösta et al. 2014). It preserves accuracy near extrema and is computationally efficient. The reconstruction of a variable q proceeds in two steps. We first calculate an interpolated value for the state $q^{\ell}_{i+1/2}$, called the original value. In the second step, limiters may be applied to the original value in order to prevent oscillations. This produces the final limited value. We will describe this procedure. The original value at the midpoint is first calculated as

$$q^{\ell}_{i+1/2} = \frac{1}{60}(2q_{i-2} - 13q_{i-1} + 47q_i + 27q_{i+1} - 3q_{i+2}). \tag{42}$$

We then compute a value, called q^{MP}, to aid in determining whether the reconstruction step is monotonicity preserving. This value is

$$q^{MP} = q_i + \mathrm{minmod}(q_{i+1} - q_i, \tilde{\alpha}(q_i - q_{i-1})), \tag{43}$$

where $\tilde{\alpha}$ is a constant, which we set to be $\tilde{\alpha} = 4.0$. The minmod function gives the argument with the smallest magnitude when both arguments have the same sign,

$$\mathrm{minmod}(x, y) = \frac{1}{2}(\mathrm{sgn}\,(x) + \mathrm{sgn}\,(y))\min(|x|, |y|). \tag{44}$$

With these calculated values, they are then compared via the following inequality:

$$(q^{\ell}_{i+1/2} - q_i)(q^{\ell}_{i+1/2} - q^{MP}) \leqslant \varpi|q|. \tag{45}$$

In this expression, $\varpi = 10^{-10}$ and $|q|$ is the L_2 norm of q_i over the stencil points $\{q_{i-2}, \ldots, q_{i+2}\}$. If the inequality holds, no limiter is necessary and the original value is used as is. If the inequality Equation (45) does not hold, a limiter is applied to the original value. (Note that the $|q|$ factor does not appear in the original algorithm, but we follow Mösta et al. 2014 in adding this term to account for the wide range of scales in the different fluid variables.)

The limiter is applied as follows. We first compute the second derivatives

$$D^{-}_i = q_{i-2} - 2q_{i-1} + q_i \tag{46}$$

$$D^{0}_i = q_{i-1} - 2q_i + q_{i+1} \tag{47}$$

$$D^{+}_i = q_i - 2q_{i+1} + q_{i+2} \tag{48}$$

and combinations

$$D^{M4}_{i+1/2} = \mathrm{minmod}(4D^0_i - D^+_i, 4D^+_i - D^0_i, D^0_i, D^+_i) \tag{49}$$

$$D^{M4}_{i-1/2} = \mathrm{minmod}(4D^0_i - D^-_i, 4D^-_i - D^0_i, D^0_i, D^-_i). \tag{50}$$

The minmod function is easily generalized for an arbitrary number of arguments as

$$\mathrm{minmod}(z_1, \ldots, z_k) = s\,\min(|z_1|, \ldots, |z_k|), \tag{51}$$

where

$$\begin{aligned} s = {} & \frac{1}{2}(\mathrm{sgn}(z_1) + \mathrm{sgn}(z_2)) \\ & \times \left| \frac{1}{2}(\mathrm{sgn}(z_1) + \mathrm{sgn}(z_3)) \times \ \ldots \right. \\ & \left. \times \ \frac{1}{2}(\mathrm{sgn}(z_1) + \mathrm{sgn}(z_k)) \right|. \end{aligned} \tag{52}$$

We then compute the following quantities:

$$q^{UL} = q_i + \alpha(q_i - q_{i-1}) \tag{53}$$

$$q^{AV} = \frac{1}{2}(q_i + q_{i+1}) \tag{54}$$

$$q^{MD} = q^{AV} - \frac{1}{2}D^{M4}_{i+1/2} \tag{55}$$

$$q^{LC} = q_i + \frac{1}{2}(q_i - q_{i-1}) + \frac{4}{3}D^{M4}_{i-1/2}, \tag{56}$$

to obtain limits for an accuracy-preserving constraint

$$q_{\min} = \max\,(\min\,[q_i, q_{i+1}, q^{MD}], \min\,[q_i, q^{UL}, q^{LC}]) \tag{57}$$

$$q_{\max} = \min\,(\max\,[q_i, q_{i+1}, q^{MD}], \max\,[q_i, q^{UL}, q^{LC}]). \tag{58}$$

Finally, the limited value for the midpoint is

$$\begin{aligned} q^{\ell,\mathrm{Lim}}_{i+1/2} = {} & q^{\ell}_{i+1/2} + \mathrm{minmod}[q_{\min} \\ & - q^{\ell}_{i+1/2}, q_{\max} - q^{\ell}_{i+1/2}]. \end{aligned} \tag{59}$$

To compute the right state $q^{r}_{i-1/2}$, we repeat the algorithm but reflect the stencil elements about the center, replacing $\{q_{i-2}, q_{i-1}, q_i, q_{i+1}, q_{i+2}\}$ with $\{q_{i+2}, q_{i+1}, q_i, q_{i-1}, q_{i-2}\}$.

The HLLE approximate Riemann solver is a central-upwind flux function that uses the maximum characteristic speeds in each direction to calculate a solution to the Riemann problem (Harten et al. 1983; Einfeldt 1988):

$$\hat{\boldsymbol{f}}^{\mathrm{HLLE}}_{i+1/2} = \frac{\lambda^+_r \boldsymbol{f}(\boldsymbol{u}^{\ell}_{i+1/2}) - \lambda^-_{\ell} \boldsymbol{f}(\boldsymbol{u}^{r}_{i+1/2})}{\lambda^+_r - \lambda^-_{\ell}} + \frac{\lambda^+_r \lambda^-_{\ell}(\boldsymbol{u}^{r}_{i+1/2} - \boldsymbol{u}^{\ell}_{i+1/2})}{\lambda^+_r - \lambda^-_{\ell}}, \tag{60}$$

where λ^+ and λ^- represent the largest characteristic speeds at the interface in the right and left directions, respectively.

The largest and smallest characteristic speeds of the relativistic fluid in flat spacetime in the direction x^i are

$$\begin{aligned} \lambda_{\pm} = {} & \frac{1}{1 - v^2 c_s^2}\{v^i(1 - c_s^2) \\ & \pm \sqrt{c_s^2(1 - v^2)[\gamma^{ii}(1 - v^2 c_s^2) - v^i v^i(1 - c_s^2)]}\,\}, \end{aligned} \tag{61}$$

where the sound speed c_s is

$$c_s^2 = \frac{1}{h}\left(\frac{\partial P}{\partial \rho}\right)\Bigg|_{\epsilon} + \frac{P}{\rho^2 h}\left(\frac{\partial P}{\partial \epsilon}\right)\Bigg|_{\rho}. \tag{62}$$

2.7. Time Integration

The conservation equations are written in semi-discrete form:

$$\begin{aligned}\frac{d\boldsymbol{u}_{i,j,k}}{dt} = &-\frac{1}{\Delta x}(\hat{\boldsymbol{f}}^1{}_{i+1/2,j,k} - \hat{\boldsymbol{f}}^1{}_{i-1/2,j,k}) \\ &-\frac{1}{\Delta y}(\hat{\boldsymbol{f}}^2{}_{i,j+1/2,k} - \hat{\boldsymbol{f}}^2{}_{i,j-1/2,k}) \\ &-\frac{1}{\Delta z}(\hat{\boldsymbol{f}}^3{}_{i,j,k+1/2} - \hat{\boldsymbol{f}}^3{}_{i,j,k-1/2}) \\ &+\boldsymbol{s}(\boldsymbol{u}_i),\end{aligned} \tag{63}$$

where $\hat{\boldsymbol{f}}_{i+1/2}$ is the numerical flux. The flux functions in each direction are evaluated separately. The sparse wavelet representation leads to a scheme for integrating a system of differential equations in time by using the method of lines. The coefficients in the expansion become time dependent and can be integrated in time using any standard time integrator. In this work, the classical fourth-order Runge–Kutta method is used, which has a CFL coefficient $\lambda_0 = 2/3$. The velocity and characteristic speeds of the fluid are bounded by the speed of light, so the time step $\lambda = c\Delta t/\Delta x$ is bounded by the CFL coefficient $\lambda \leqslant \lambda_0$.

As the physical state evolves during the simulation, the set of essential points will change. This means that the method needs to support the promotion of a point to becoming essential and the demotion of a point from being essential. To allow for such changes to the set of essential points, so-called *neighboring points* are added to the grid (Paolucci et al. 2014b). These are those points adjacent to an essential point at the next finer level. In a sense, these points are sentinels waiting to become essential. Given that the points at level 0 are always essential, the points at level 1 will all be at least neighboring, and in this way, both the level 0 and level 1 grids will be fully occupied. Both neighboring and essential points participate in time integration and, for this reason, are called *active* points.

At the end of each time step, every active point has its wavelet transformation computed. Essential points that no longer exceed the error threshold are demoted, and neighboring points that exceed the threshold are promoted. Neighboring points promoted to being essential points will thus require their own neighboring points at the next finer level. In this way, as the solution develops features on finer scales, the grid adapts, adding points to the grid exactly where the resolution is needed. Initially, the field values for a neighboring point can be taken from the initial conditions. Neighbors added after the initial time slice are given field values from the inverse wavelet transformation. That is, they are given a wavelet coefficient of zero, so their fields are equal to the interpolation from the previous level.

Finally, there is a third class of grid points, called *nonessential* points, that are required to fill out wavelet or other computational stencils of essential and neighboring points. Nonessential points do not participate in time integration and are given values via interpolation.

In practice, an upper limit on refinement, $j_{\max}$, needs to be specified. To this end, the wavelet expansion of the fields given in Equations (26) and (30) is truncated at some finite number of terms such that the upper limit in those series is set to $j_{\max}$. During an evolution, it can happen that the solution naturally attempts to refine past $j_{\max}$. There are two options in this case. The first is for the code to complain and die, and the second is for the code to complain and warn the user that the solution may have an error higher than the set threshold. Taking the former approach assures that the error bounds implied by the chosen ϵ are not violated, because if the grid attempts to add a point at level $j_{\max}+1$, it is because the point is needed. However, if there is a discontinuity in the solution, no level of refinement will be sufficient to satisfy the refinement criterion. A possible solution to this problem is to smooth over discontinuities in initial data and to add some viscosity to prevent their formation during a simulation. However, we are interested in sharp features that develop during the simulation, and it is for that reason that we use a fluid reconstruction and numerical flux function that allow for these sharp features. Thus, with a method that allows for discontinuities, we can never fully satisfy the refinement criterion if a discontinuity develops, and so the grid will want to refine forever. As a result, we take the practical step of limiting the maximum refinement level.

2.8. Primitive Solver

An important aspect of any relativistic hydrodynamics code is the inversion between the conserved and the primitive variables. Because the equations are written in conservation form, the conserved variables are the evolved variables. However, the primitive variables are needed as part of the calculation. For Newtonian fluids, this inversion from the conserved to the primitive variables is algebraic, can be done in closed form, and results in a unique solution for the primitive variables provided that the conserved variables take on physical values. Such is not the case in relativistic situations. As a result, a number of related procedures can be found in the literature to effect this inversion (Duez et al. 2005; Noble et al. 2006; Etienne et al. 2012). Due to its importance, we sketch our approach to performing the inversion.

For this discussion, we revert to the undensitized form of the conserved variables (D, S_i, τ). In terms of the fluid primitives (ρ, v^i, P), and for our chosen (Γ-law) equation of state (with $1 < \Gamma \leqslant 2$), the conserved variables are given by Equations (7)–(9). The inversion can be reduced to a single equation for $x = hW^2/(\tau + D)$, namely,

$$-\left(x - \frac{\Gamma}{2}\right)^2 + \frac{\Gamma^2}{4} - (\Gamma - 1)\beta^2 = (\Gamma - 1)\,\delta\,\sqrt{x^2 - \beta^2}, \tag{64}$$

where we have defined

$$\beta^2 = \frac{S^2}{(\tau + D)^2}, \qquad \delta = \frac{D}{\tau + D}. \tag{65}$$

Note that the left-hand side, call it $f(x)$, is a downward-pointing quadratic while the right-hand side includes the square root of another quadratic, $g(x) = x^2 - \beta^2$, which has roots $\pm|\beta|$. Solving for x amounts to finding the intersections of $f(x)$ and $(\Gamma - 1)\delta\sqrt{g(x)}$. For a physical solution, $x > |\beta|$, the largest root of $g(x)$. Therefore, there is a single intersection bracketed

Table 1
Initial State for Six Riemann Problems

	Case	v^x	v^y	ρ	p	Γ
I	$x < 0$	0	0	10	13.33	4/3
	$x > 0$	0	0	1	10^{-6}	
II	$x < 0$	0	0	1	1000	5/3
	$x > 0$	0	0	1	0.01	
III	$x < 0$	−0.2	0	0.1	0.05	4/3
	$x > 0$	0.2	0	0.1	0.05	
IV	$x < 0$	$1 - 10^{-6}$	0	0.001	$3.333 \cdot 10^{-9}$	4/3
	$x > 0$	$-1 + 10^{-6}$	0	0.001	$3.333 \cdot 10^{-9}$	
V	$x < 0$	0	0.0	1.0	1000	5/3
	$x > 0$	0	0.99	1.0	0.01	
VI	$x < 0$	0	0.9	1.0	1000	5/3
	$x > 0$	0	0.9	1.0	0.01	

by this root of $g(x)$ and the larger root of $f(x)$, namely, $|\beta| < x < x^*$, where

$$x^* = \frac{\Gamma}{2} + \left[\frac{\Gamma^2}{4} - (\Gamma - 1)\beta^2\right]^{1/2}. \tag{66}$$

That this root of $f(x)$ is real is guaranteed by the dominant energy condition, given here as $\beta^2 < 1$. Hence, a unique solution to the primitive inversion exists in the pure hydrodynamics case provided that the conserved variables satisfy this inequality. Using a straightforward Newton's method allows us to solve for the primitive variables in virtually every case. On the rare occasions that the inequality is violated, rescaling β^2 to bring it within physical bounds is sufficient to allow the primitive solve to proceed to a solution.

3. One-dimensional Tests

Lora-Clavijo et al. (2015) outline a number of simple 1D Riemann problems for relativistic hydrodynamics. We report on some of these same test problems below. For each Riemann problem, the base grid was chosen to have $N = 10$, and we have varied both the maximum number of refinement levels, $j_{\max}$, and the value of ϵ. In each case, the overall domain is symmetric about $x = 0$ with the left and right fluid states initially separated there. The primitive fields on each side of the problem for the six cases are given in Table 1.

In each case, the results obtained with OAHU match closely the exact solution. As an example, Figure 5 shows the results for Case I at $t = 0.4$ with $N = 10$, $j_{\max} = 10$, and $\epsilon = 10^{-5}$. The solution found matches the exact solution extremely well. The final grid has adapted to the features that form during the simulation. The final state shown has only 471 points out of a possible 10,241 points, giving this simulation a very high effective resolution at a savings of over 95%.

The bottom right panel of Figure 5 gives the level of each point in the grid. This figure is characteristic of the refinement of the wavelet method. When the solution is smooth, there is very little refinement, and where the solution exhibits sharp features, the refinement proceeds to higher levels. For the very sharp features in this example, the refinement proceeds to the highest level allowed for the particular simulation. A discontinuity in the solution will generically refine as far as is allowed by the simulation. There is no smooth approximation at any resolution to a sharp transition.

It is interesting to explore how the quality of the solution depends on the allowed maximum refinement level, $j_{\max}$, and on the refinement criterion, ϵ. Figure 6 shows two close-up views of features for Case I run for three different values of $j_{\max}$. For each one, $N = 10$ and $\epsilon = 10^{-5}$. The overshoot in the density for $j_{\max} = 10$ is not an artifact of the adaptive scheme we employ; when run at an equivalent resolution in unigrid ($N = 5120$ and $j_{\max} = 1$), the same overshoot is present. This is not a surprise, but is due to the use of a finest grid that is too coarse to represent the smallest scale present in the equations being solved. Strictly speaking, in order to obtain a solution with error below ϵ, the maximum level of refinement should not be reached by the adaptive algorithm. Of course, for a solution with a discontinuity such as Case I, the maximum refinement level, $j_{\max}$, will always be reached. A similar close-up is shown in Figure 7 giving a comparison of a set of runs with $N = 10$, $j_{\max} = 8$, and various values of ϵ. As ϵ decreases, the solution matches more closely the exact solution. However, there is little difference between $\epsilon = 10^{-4}$ and 10^{-5}. This demonstrates the interplay between $j_{\max}$ and ϵ. In this case, the sharpened refinement criterion would drive more refinement near this feature, but the maximum refinement level has already been reached.

In order to provide an error measure for the simulations, we calculate L_1 norms of the errors in the primitive variables. For unstructured and nonuniform grids that cover some region V, we base our discrete L_1 norm on the Riemann sum of the difference between the computed field f and the exact solution to the Riemann problem, $f_{\rm ex}$:

$$L_1(f) = \frac{1}{V}\sum_k |f(x_k) - f_{\rm ex}(x_k)|\, \Delta V_k, \tag{67}$$

where k is a general index that labels each point and ΔV_k approximates the differential volume element at x_k and, in practice, is the size of the cell at x_k.

Errors for Case I are given in Table 2. In particular, the top half of the table shows the L_1 norms for a uniform grid, while the bottom half shows the L_1 norms for a wavelet adaptive grid that has been made sparse via thresholding at the level of $\epsilon = 10^{-5}$. The first two columns provide the resolution of the simulation in terms of both the highest allowed refinement level and the total number of points used. For unigrid, this is $N_T = 10 \cdot 2^{j_{\max}} + 1$, while for the wavelet adaptive grid, N_T is the number of points kept as part of the sparse grid after thresholding. The rightmost column provides an average order of convergence (OC) for the three primitive variables as $j_{\max}$ increases.

Clearly, increasing the maximum refinement level for the same ϵ tends to decrease the overall error. We also experimented with changing ϵ for each of the cases. As can be seen from Figure 7 for Case I, and which is consistent across all of the cases, as ϵ is decreased, the error decreases. However, as already mentioned, depending on its value and the value of $j_{\max}$, the refinement may reach its allowed maximum, but the additional refinement that would be generated by the smaller ϵ is not realized, leading only to modest accuracy gains.

We performed similar simulations in one dimension for the remaining cases. All were done varying $j_{\max}$ from 5 to 10 and were compared against equivalent unigrid simulations. The results presented are for $\epsilon = 10^{-5}$. We include error and

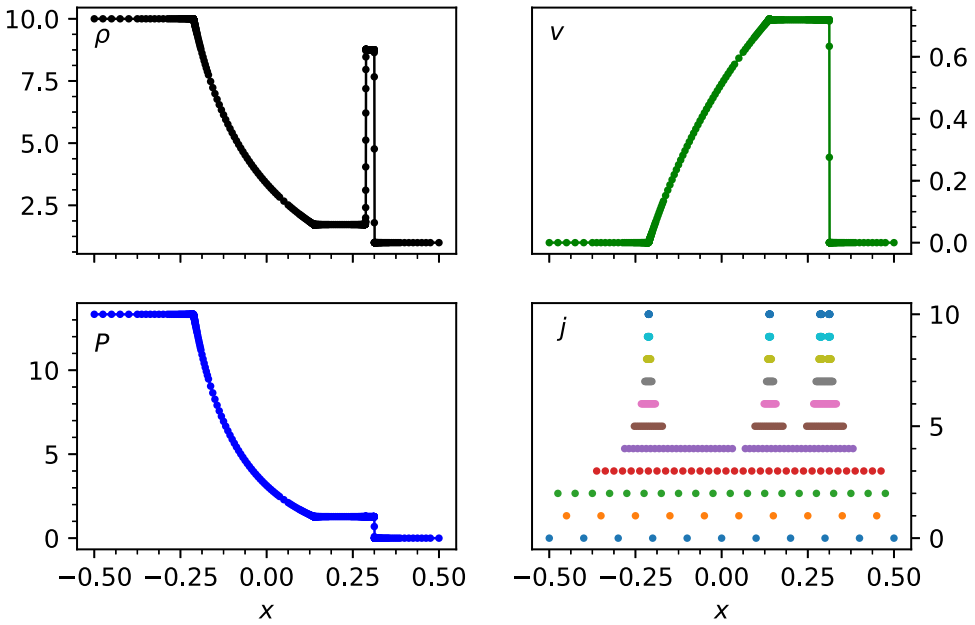

Figure 5. Comparison of the simulation results to the exact solution for the Case I Riemann problem (see Table 1). The exact solution is plotted at $t = 0.4$ as a line, with points from the numerical solution on a wavelet grid. This simulation was performed with $N = 10$, $j_{\max} = 10$, and $\epsilon = 10^{-5}$. The fluid density, velocity, and pressure all match the exact solution well. In the bottom right panel all points used in the simulation are plotted by position and level. As shown in this panel, the grid adapts to the features of the solution for ρ, v, and P, with refinement concentrated around nondifferentiable points of the solution. The resulting wavelet grid is not hand-tuned; instead, it adapts to the evolution of the fluid. At the time shown, only 471 out of a possible 10,241 grid points are occupied.

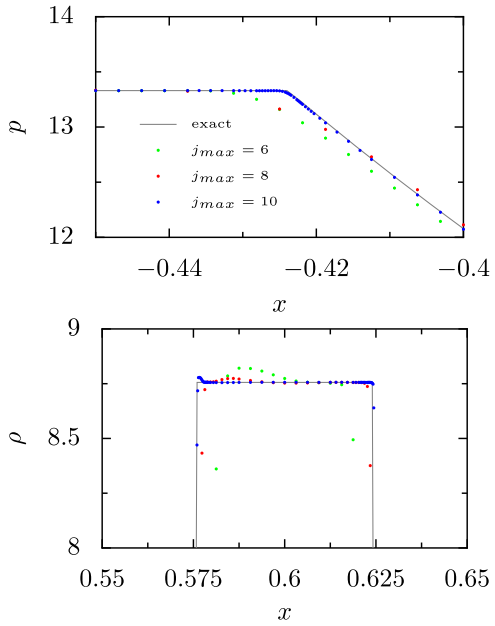

Figure 6. Detail of pressure at one end of the rarefaction wave (top) and the density in the shock (bottom) for Case I, with three different values of $j_{\max}$. As the maximum refinement level is increased, the wavelet solution matches the exact solution more closely.

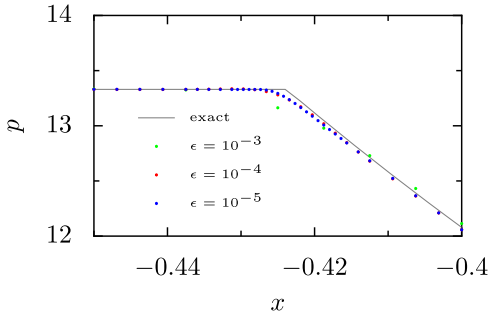

Figure 7. Detail of the pressure similar to the top panel of Figure 6, but with fixed $j_{\max} = 8$ and a varying ϵ. The refinement criterion has already pushed the refinement to the maximum level with $\epsilon = 10^{-4}$, and so there is little change as ϵ is decreased to 10^{-5}.

convergence information in Tables 3–7. They are broadly consistent with the results in Case I and with each other. They demonstrate that the code converges to the exact solution with a first-order convergence rate that is consistent with our chosen methods as applied to fluid systems containing shocks and discontinuities.

Some additional, specific comments can be made regarding these results. Note that the much smaller errors for Case III are

Table 2
L_1 Errors and Convergence Order for Case I

$j_{\max}$	N_T	$L_1(v)$ (10^{-3})	$L_1(\rho)$ (10^{-3})	$L_1(P)$ (10^{-3})	OC
5	321	3.36	57.1	21.3	⋯
6	641	1.68	29.2	10.7	0.99
7	1281	0.819	15.2	5.33	0.99
8	2561	0.388	7.95	2.66	1.01
9	5121	0.173	4.21	1.31	1.03
10	10241	0.0711	2.38	0.668	1.05
5	185	3.37	57.2	21.5	⋯
6	249	1.70	29.3	11.0	0.97
7	312	0.846	15.4	5.75	0.96
8	369	0.419	8.22	3.13	0.93
9	413	0.207	4.52	1.82	0.89
10	471	0.106	2.73	1.19	0.80

Table 3
L_1 Errors and Convergence Order for Case II

$j_{\max}$	N_T	$L_1(v)$ (10^{-2})	$L_1(\rho)$ (10^{-2})	$L_1(P)$ (10^{-2})	OC
5	321	1.06	9.24	171	⋯
6	641	0.599	5.22	87.2	0.87
7	1281	0.280	2.70	43.7	1.02
8	2561	0.150	1.43	21.9	0.94
9	5121	0.0850	0.820	11.0	0.87
10	10241	0.0425	0.434	5.52	0.97
5	191	1.06	9.24	173	⋯
6	238	0.604	5.22	91.1	0.86
7	286	0.286	2.71	48.7	0.98
8	335	0.157	1.43	27.7	0.87
9	408	0.0923	0.827	17.3	0.74
10	467	0.0501	0.442	12.1	0.77

Table 4
L_1 Errors and Convergence Order for Case III

$j_{\max}$	N_T	$L_1(v)$ (10^{-4})	$L_1(\rho)$ (10^{-4})	$L_1(P)$ (10^{-4})	OC
5	321	7.09	1.34	0.790	⋯
6	641	3.51	0.671	0.393	1.01
7	1281	1.74	0.333	0.195	1.01
8	2561	0.867	0.166	0.0968	1.01
9	5121	0.432	0.0828	0.0482	1.00
10	10241	0.216	0.0415	0.0241	1.00
5	213	7.09	1.34	0.790	⋯
6	257	3.52	0.672	0.394	1.01
7	279	1.76	0.336	0.196	1.00
8	321	0.881	0.170	0.0987	0.99
9	379	0.448	0.0879	0.0502	0.97
10	423	0.233	0.0476	0.0263	0.92

Table 5
L_1 Errors and Convergence Order for Case IV

$j_{\max}$	N_T	$L_1(v)$ (10^{-2})	$L_1(\rho)$ (10^{-2})	$L_1(P)$ (10^{-2})	OC
5	321	0.718	2.70	687	⋯
6	641	0.386	1.53	391	0.84
7	1281	0.211	0.858	222	0.84
8	2561	0.0959	0.477	128	0.92
9	5121	0.0523	0.245	63.3	0.95
10	10241	0.0297	0.143	36.3	0.80
5	125	0.718	2.70	687	⋯
6	217	0.308	1.26	320	1.14
7	393	0.211	0.858	222	0.55
8	749	0.100	0.454	123	0.95
9	1479	0.0495	0.243	62.3	0.97
10	2779	0.0305	0.139	37.1	0.75

not surprising, as this test contains two rarefaction waves, and although there are sharp features in the exact solution, there are no discontinuities. For Case IV, which had initial Lorentz factors in excess of 500, we were not able to evolve the system with MP5 reconstruction and used PPM instead. While this is consistent with Lora-Clavijo et al. (2015), we do not have an explanation for it. For all other simulations, we successfully used MP5 reconstruction. The final two cases, Case V and Case VI, are distinguished from the first four by the presence of

Table 6
L_1 Errors and Convergence Order for Case V

$j_{\max}$	N_T	$L_1(v)$ (10^{-2})	$L_1(\rho)$ (10^{-2})	$L_1(P)$ (10^{-2})	OC
5	321	0.436	15.4	186	⋯
6	641	0.191	7.58	92.8	1.07
7	1281	0.110	4.15	46.9	0.88
8	2561	0.0479	2.11	23.3	1.06
9	5121	0.0270	1.15	11.9	0.89
10	10241	0.0113	0.578	6.04	1.08
5	258	0.437	15.4	187	⋯
6	423	0.193	7.58	95.1	1.06
7	493	0.113	4.15	50.1	0.86
8	561	0.0518	2.12	27.3	0.99
9	721	0.0314	1.16	16.3	0.78
10	1038	0.0160	0.587	10.7	0.85

Table 7
L_1 Errors and Convergence Order for Case VI

$j_{\max}$	N_T	$L_1(v)$ (10^{-2})	$L_1(\rho)$ (10^{-2})	$L_1(P)$ (10^{-2})	OC
5	321	2.32	37.5	166	⋯
6	641	1.36	25.1	84.2	0.78
7	1281	0.699	13.7	43.2	0.93
8	2561	0.350	6.95	21.5	0.99
9	5121	0.184	3.80	10.6	0.94
10	10241	0.104	2.23	5.61	0.84
5	204	2.34	38.5	171	⋯
6	270	1.28	24.5	87.1	0.83
7	406	0.695	13.7	48.5	0.86
8	572	0.339	6.96	26.9	0.95
9	774	0.183	3.77	16.4	0.83
10	1050	0.108	2.22	11.8	0.67

transverse velocities in the shock tube. This adds another level of complexity to the problem and renders Case VI, in particular, especially challenging. Note that our code continues to show convergence to the exact solution for both unigrid and the wavelet adaptive grid. Indeed, the wavelet adaptive grid produces errors and convergence rates comparable to the unigrid simulations while using an order of magnitude fewer overall grid points. This is remarkable confirmation of the utility of this method for solving time-dependent PDEs.

Both Case V and Case VI are shown at $t = 0.4$ in Figures 8 and 9. A key aspect of the wavelet adaptive grid is shown in the bottom right panel of the latter figure, where the level of each point in the grid is given. This again shows the characteristic structure of the grids resulting from this method. For the particular case of this hard relativistic shock, there are clearly discontinuities, as well as other sharp features associated with this Riemann problem. Near the discontinuities, the grid refines to its maximum allowed level as will be expected. However, near the sharp, yet continuous, feature in the vicinity of $x \approx -0.3$, note that additional resolution is needed and the sparse grid includes more points than neighboring smooth regions. However, the particular refinement level needed is reached without necessarily going all the way to $j_{\max}$.

In order to further explore the effect of ϵ on solution error, we ran the hard relativistic shock tube for the three values of $\epsilon = \{10^{-4}, 10^{-5}, 10^{-6}\}$ and varied $j_{\max}$ from 5 to 10 on a base grid of $N = 10$. The effects of ϵ on the convergence of the solution error

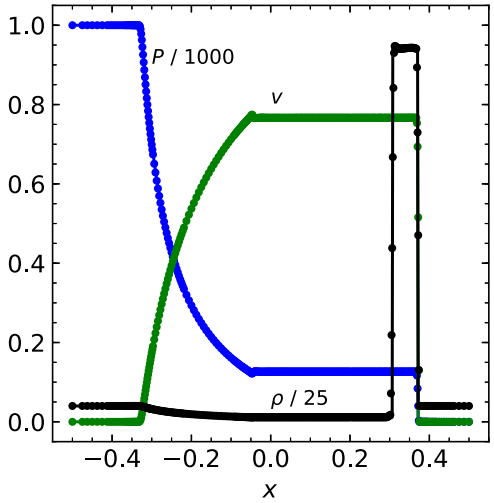

Figure 8. Solution for Case V and easy relativistic Riemann problem with a transverse velocity. The exact solution is plotted at $t = 0.4$ with lines, and the points come from the numerical solution for $j_{\max} = 6$ and $\epsilon = 10^{-5}$.

are illustrated in Figure 10. Numerical errors in solutions of the Riemann problem are concentrated at a few locations where the solution is discontinuous. As a result, decreasing ϵ does not provide a solution with significantly lower error. Indeed, the wavelet expansion identifies the nondifferentiable regions with a large value of ϵ, and these regions are then refined all the way to $j_{\max}$. Decreasing ϵ simply increases the size of the refined region about each nondifferentiable point in the solution. However, the advantages of the wavelet method become clear on comparison with the errors that arise from a run on a uniform grid. The error is roughly the same when considered by $j_{\max}$. But the convergence with respect to total number of grid points is improved, by as much as an order of magnitude. Thus, a judiciously chosen value of ϵ can result in a convergent, adaptive scheme that gives errors comparable to those from a unigrid evolution but using far fewer grid points.

To conclude this section, we note that we have also monitored the conservation of the evolved variables during a simulation. For these test cases, the conservation is as good as the specified ϵ. This is to be expected, as the representation keeps details only if those details are larger than ϵ. In each of the cases above, the relative drift in the conserved quantities is on the order of ϵ for that case.

4. Two-dimensional Tests

In the previous section we presented data showing that OAHU converges to the exact solution for six different 1D Riemann problems. In this section we consider four 2D problems. We consider the convergence of a smooth, isentropic solution in two dimensions and a 1D shock tube problem rotated in the xy-plane. We then look at two problems that have been widely reported in the literature, a 2D Riemann problem and the Kelvin–Helmholtz instability. It is worth restating, however, that our approach does not use genuinely multidimensional numerical methods. Rather, multidimensional problems are solved line by line in each dimension, as in Equation (63).

4.1. Convergence of an Isentropic Smooth Solution

The shock tube tests in the previous section all have discontinuous data, and thus the convergence to the exact solution is only first order. For smooth data, OAHU is expected to demonstrate second-order convergence. Although the time integrator (RK4) and reconstruction (MP5) are higher order, the overall method based on reconstruction of the primitive variables is only second order for smooth data.

To test the convergence of our code in two dimensions with smooth initial data, we evolve a simple Gaussian pulse with the initial parameters

$$\rho = \rho_0 + Ae^{-(x^2+y^2)/\sigma^2} \tag{68}$$

$$P = \kappa\rho^{\Gamma} \tag{69}$$

$$v^x = v^y = 0. \tag{70}$$

Here we set $\Gamma = 5/3$, $\rho_0 = A = \kappa = 1$, and $\sigma = 1.5$. The base configuration has $N_x = N_y = 51$ on the domain $x, y \in [-4, 4]$, and it was evolved to time $t = 4$ with the Courant factor $\lambda = 0.25$. A 1D slice of the initial density profile and the solution at four subsequent times is shown in Figure 11. The data are evolved with outgoing boundary conditions, and the parameters were chosen such that shocks would not form on the computational domain. In regions without shocks, the fluid flow is isentropic, and this implies that $P/\rho^{\Gamma} = \kappa$ is a constant everywhere in the fluid. Following Duffell (2016), we define the error function

$$\mu = \frac{P}{\rho^{\Gamma}} - \kappa, \tag{71}$$

and we compute $L_1(\mu)$ on the computational domain. We repeated this simulation and analysis for runs with $N_x = N_y =$ 101, 201, 401, 801, 1601 points. Figure 12 shows $L_1(s)$ for different resolutions as a function of t. This error converges to second order as expected for our algorithm.

4.2. Rotated One-dimensional Riemann Problem

As a second test of OAHU in multiple dimensions, we evolve initial data for the 1D shock tube given in Case II, but rotated in the xy-plane by 45° so that the discontinuity lies along a diagonal line on the grid. Outflow boundary conditions are used on all boundaries, and the Courant number is $\lambda = 0.25$. The solution along a line perpendicular to the initial discontinuity at time $t = 0.4$ is shown in Figure 13.

We computed this solution on grids with $N_x = N_y =$ 81, 161, 321, 641, 1281 points and computed convergence to the exact solution. These data are in Table 8 and compare well with the 1D results for this case in Table 3.

4.3. Two-dimensional Riemann Problem

A 2D Riemann problem is constructed by setting a different constant-fluid state in each quadrant of the domain. Outflow boundary conditions are imposed on each boundary. Labeling the quadrants northwest (NW), northeast (NE), southwest (SW), and southeast (SE), we choose parameters first used by Del Zanna & Bucciantini (2002):

$$(\rho, v^x, v^y, P) = \begin{cases} (0.1, 0, 0, 0.01) & \text{NW} \\ (0.1, 0.99, 0, 1) & \text{NE} \\ (0.5, 0, 0, 1) & \text{SW} \\ (0.1, 0, 0.99, 1) & \text{SE} \end{cases}, \tag{72}$$

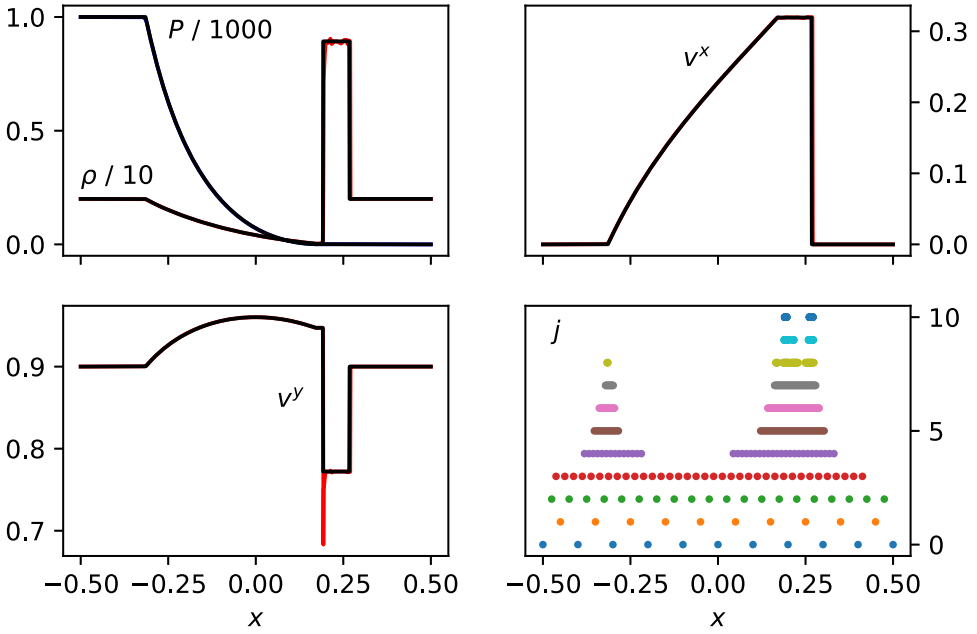

Figure 9. Comparison of the simulation results to the exact solution at $t = 0.6$ for Case VI, a hard relativistic shock with transverse velocity. The numerical solution is plotted first as a red line, and then the exact solution is plotted as a black line. Differences between the two solutions can be identified where the red line is visible. This simulation was performed with $N = 10$, $j_{\max} = 10$, and $\epsilon = 10^{-4}$. The density and pressure are plotted in the top left panel, and the velocity components v^x and v^y are plotted in the top right and bottom left panels, respectively. The largest error in the solution is in the transverse component of the velocity at the contact discontinuity. The bottom right panel shows the points used in the simulation, plotted by location and grid level j. Again, refinement is concentrated at locations where the solution features change rapidly (nondifferentiable). The resulting wavelet grid is not hand-tuned; instead, it adapts to the evolution of the fluid. At the time shown, only 1050 out of a possible 10,241 grid points are occupied.

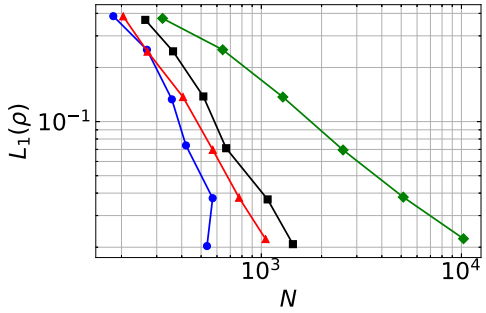

Figure 10. L_1 error in ρ for the hard relativistic shock tube (Case VI) as a function of N, the number of points used in the simulation. The blue line (circles) is for runs with $\epsilon = 10^{-4}$ as $j_{\max} = 5, \ldots, 10$. The red line (triangles) is for runs with $\epsilon = 10^{-5}$, and the black line (squares) is for $\epsilon = 10^{-6}$. The green line (diamonds) is for a uniform grid.

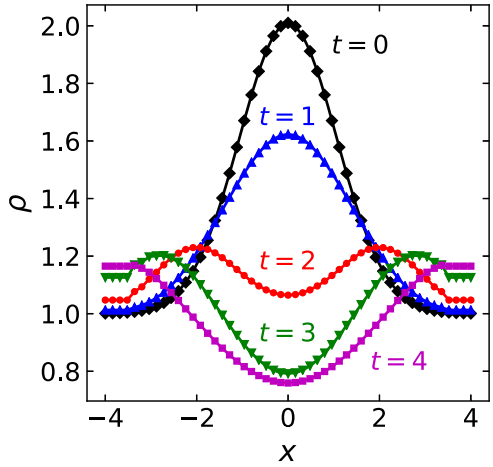

Figure 11. Initial density profile (black diamonds) along the line $x = 0$ for the convergence tests with an isentropic smooth solution. The solutions at subsequent times $t = 1$ (blue triangles), $t = 2$ (red circles), $t = 3$ (green inverted triangles), and $t = 4$ (magenta squares) are also plotted.

with $\Gamma = 5/3$. We performed two simulations on uniform grids of 401^2 points, the first with minmod reconstruction and the second with MP5, and a third simulation with an adaptive grid and MP5 reconstruction. The solution at $t = 0.4$ is shown in Figure 14. The minmod solution looks very similar to previously reported results (Lucas-Serrano et al. 2004; Zhang & MacFadyen 2006; Lora-Clavijo et al. 2015).

There are two stationary contact discontinuities, one along the boundary between the NW and SW quadrants and another between the SW and SE quadrants. Both become diffused and generate some numerical artifacts in the SW quadrant. A bow shock propagates along the diagonal in the SW quadrant, with a complicated structure behind it. In the NE quadrant, two curved shocks move outward, and there is a narrow, high-density structure that forms along the diagonal. As expected with MP5 reconstruction, all of the features in the solution are more sharply defined. The complicated structure behind the bow shock now displays rolls. These were hinted at in the solution using the U-PPM method in Zhang & MacFadyen (2006). Significantly, these rolls are not evident in simulations with minmod reconstruction, even if the resolution is increased by a factor of four. Note, too, that the MP5 solution has high-frequency numerical noise between the curved shocks in the NE quadrant.

The final panel in Figure 14 shows the solution calculated on a grid with $j_{\max} = 6$, a grid with an effective resolution of 641 points in each direction. The adaptive solution also uses MP5 and is very similar to the uniform-grid solution. The largest

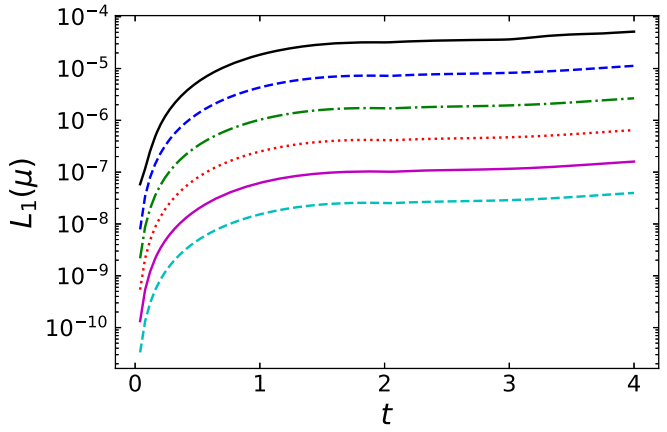

Figure 12. Second-order convergence of our code in a 2D test for an isentropic smooth solution. We plot $L_1(\mu)$, where μ is defined by Equation (71), over the computational domain as a function of time (t) for grids with $N = 51^2$ (top curve), 101^2, 201^2, 401^2, 801^2, and 1601^2 points (bottom curve).

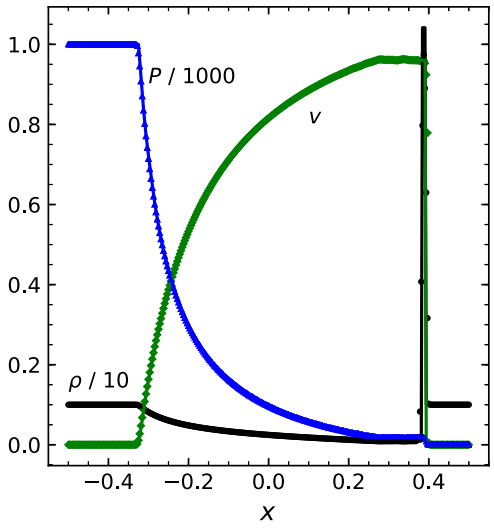

Figure 13. Solution for Case II at $t = 0.4$ computed in two dimensions. The initial data were rotated by 45° in the xy-plane so that the initial discontinuity is along a diagonal of the grid. The density is plotted with black circles, the pressure with blue triangles, and the velocity normal to the discontinuity with green diamonds. The exact solutions are plotted as lines in the respective colors. The computational grid has 321^2 points with uniform resolution.

Table 8
L_1 Errors and Convergence Order for a 2D Version of Case II along the Diagonal

N_T	$L_1(v)$ (10^{-2})	$L_1(\rho)$ (10^{-2})	$L_1(P)$ (10^{-2})	OC
81	2.03	14.5	316	...
161	1.24	10.1	146	0.78
321	0.601	5.37	67.0	1.03
641	0.269	3.32	36.1	0.92
1281	0.144	2.35	16.2	0.85

difference is the lack of symmetry across the diagonal in the complicated structure behind the bow shock. This appears to occur because the computational grid during the evolution is not exactly symmetric, a consequence of the local refinement criteria based on a sparse wavelet representation. This panel plots only the actual grid points used in the simulation, about 32% of the total available points. The grid outside of the shocks is sparsely populated, especially in the NW and SE quadrants. While the refinement naturally adapts to the solution features, in this case we see that it is also susceptible to refining on numerical noise, such as between the curved shocks in the NE quadrant. Refinement is triggered when the solution does not smoothly converge at a given length scale. Since numerical noise is resolution dependent, the grid in noisy regions will be highly refined. This highlights the importance of numerical methods that minimize numerical noise.

4.4. Relativistic Kelvin–Helmholtz Instability

We applied OAHU to the relativistic Kelvin–Helmholtz instability in two dimensions. For comparison, we have used identical initial conditions as in Radice & Rezzolla (2012). The computational domain is taken to be a periodic box from $x = -0.5$ to $x = 0.5$ and from $y = -1$ to $y = 1$. The shear is introduced via a counterpropagating flow in the x-direction,

$$v^x(y) = \begin{cases} V_s \tanh[(y - 0.5)/a], & y > 0 \\ -V_s \tanh[(y + 0.5)/a], & y \leqslant 0. \end{cases} \quad (73)$$

Here $a = 0.01$ is the thickness of the shear layer and $V_s = 0.5$. A small perturbation of the velocity transverse to the shear layer seeds the instability:

$$v^y(x, y) = \begin{cases} A_0 V_s \sin(2\pi x)\exp[-(y - 0.5)^2/\sigma], & y > 0 \\ -A_0 V_s \sin(2\pi x)\exp[-(y + 0.5)^2/\sigma], & y \leqslant 0, \end{cases} \quad (74)$$

where $A_0 = 0.1$ and $\sigma = 0.1$. For this test, $\Gamma = 4/3$ and the pressure is initially constant, $P = 1$. The density is given by a profile similar to v^x superposed on a constant as follows:

$$\rho(y) = \begin{cases} \rho_0 + \rho_1 \tanh[(y - 0.5)/a], & y > 0 \\ \rho_0 - \rho_1 \tanh[(y + 0.5)/a], & y \leqslant 0, \end{cases} \quad (75)$$

where $\rho_0 = 0.505$ and $\rho_1 = 0.495$.

The density of this system is illustrated in Figure 15 at time $t = 3$ for a run having a base grid size of $(N_x, N_y) = (40, 80)$ and a maximum refinement level of $j_{\max} = 6$. Thus, the effective grid size is 2560×5120. The refinement criterion was $\epsilon = 10^{-4}$. The final state is consistent with the results in Radice & Rezzolla (2012). The conservation of the fluid is consistent with the chosen ϵ: D suffers a relative change of 6.13×10^{-6}, and τ suffers a relative change of 1.04×10^{-4}. Note the appearance of the secondary whirl along the shear boundary. In accord with the results of Radice & Rezzolla (2012), we find that the number and appearance of these secondary instabilities depend on the maximum resolution employed in our simulation, supporting their finding that these secondary instabilities are numerical artifacts.

5. Rayleigh–Taylor Instability in a Relativistic Outflow

One interesting problem in relativistic fluid dynamics is the interaction of a relativistic blast wave of ejecta from a gamma-ray burst (GRB) explosion with the surrounding interstellar medium (ISM). As the shock expands adiabatically into the ISM, it loses thermal energy and eventually decelerates. This results in a double-shock system with a forward shock (traveling into the ISM), a contact discontinuity, and a reverse shock (moving into the ejecta). Levinson (2009, 2010) showed that the contact discontinuity is unstable to the Rayleigh–Taylor instability. The turbulence generated by this instability

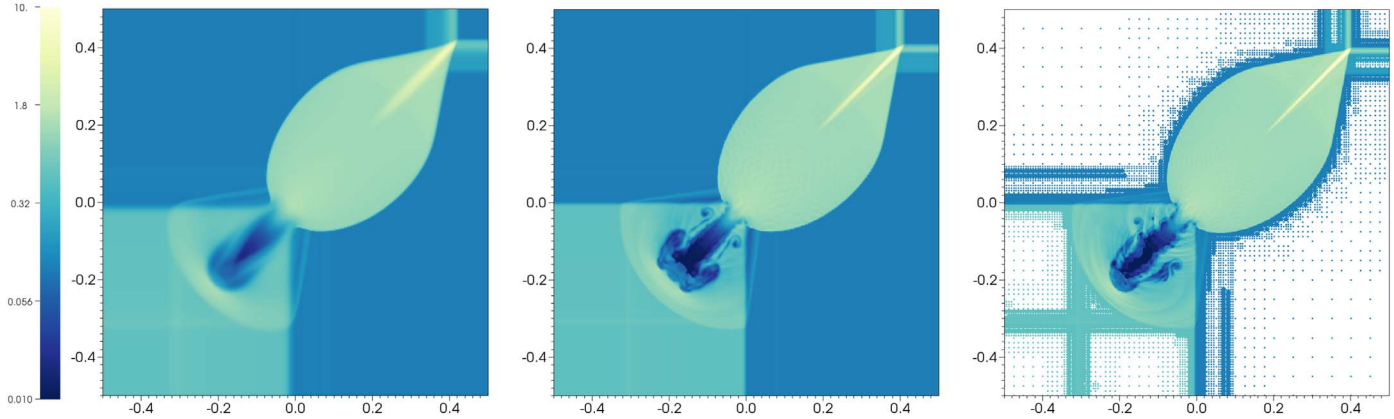

Figure 14. Parameter ρ with a logarithmic scale at time $t = 0.4$ in the 2D Riemann problem. The first two panels show ρ computed on a uniform grid of 401^2 points with two different reconstructions, minmod (left) and MP5 reconstruction (middle). The right panel shows the solution with a wavelet grid with six levels of refinement. In this panel only individual points used in the simulation are plotted, demonstrating how the grid refinement conforms to the features of the solution.

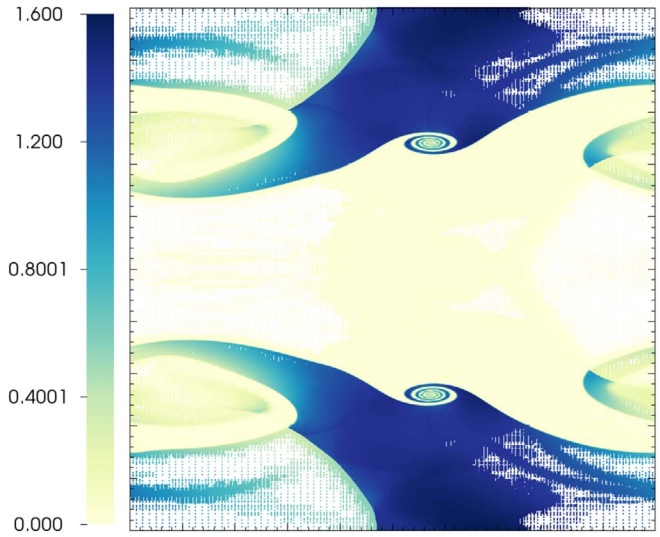

Figure 15. Relativistic Kelvin–Helmholtz instability at time $t = 3$. The points are colored by their density. This simulation used $(N_x, N_y) = (40, 80)$, $j_{\max} = 6$, and $\epsilon = 10^{-4}$. Compare with Radice & Rezzolla (2012). This figure was created by plotting only individual points used in the simulation, and it demonstrates the fine-scale refinement that naturally conforms to the features of the solution.

can amplify magnetic fields and the emission from the thin shell of material behind the forward shock. Duffell and MacFadyen have studied this system with numerical simulations in a series of papers (Duffell & MacFadyen 2011, 2013, 2014), finding that the Rayleigh–Taylor instability can disrupt the forward shock for soft equations of state, which might be typical of radiative systems (Duffell & MacFadyen 2014).

Simulating the relativistic outflow of GRB ejecta constitutes an especially challenging numerical test for an adaptive relativistic fluid code. Relativistic effects compress the width of the thin shell by the Lorentz factor squared, $\Delta r/r \approx 1/W^2$. Capturing the Rayleigh–Taylor instability that forms at the contact discontinuity within the shell thus requires very high resolution within a thin shell that propagates outward with a velocity near the speed of light. Duffell and MacFadyen succeeded in simulating this system using an elegant moving mesh code, TESS (Duffell & MacFadyen 2011). The computational cells in TESS are allowed to move with the fluid, giving very high resolution in the shell and at the shocks. As a final test, we repeat the decelerating shock test to demonstrate the adaptive capability of the wavelet approach for relativistic hydrodynamics.

We use the initial data for the decelerating shock given in Duffell & MacFadyen (2011). The initial data are spherically symmetric, but this symmetry is broken by the instability, so we perform the simulation in cylindrical coordinates. We label these coordinates $\{s, z\}$, where s is the cylindrical radius and the spherical radius r is given by $r^2 = s^2 + z^2$. The initial data for the spherical explosion are

$$\rho = \begin{cases} \rho_0 (r_0/r_{\min})^{k_0} & r < r_{\min} \\ \rho_0 (r_0/r)^{k_0} & r_{\min} < r < r_0 \\ \rho_0 (r_0/r)^{k} & r_0 < r, \end{cases} \tag{76}$$

where the different parameters are chosen to be

$$k_0 = 4, \quad r_0 = 0.1, \quad r_{\min} = 0.001. \tag{77}$$

A spherical explosion into a medium with a power-law dependence, $\rho \simeq r^{-k}$, will decelerate if $k < 3$. So we choose $k = \{0, 1, 2\}$ for our runs below. The pressure is given by

$$P = \begin{cases} e_0 \rho/3 & \text{for } r < r_{\exp} \\ 10^{-6}\rho & \text{for } r > r_{\exp}, \end{cases} \tag{78}$$

where $r_{\exp} = 0.003$ and the constant e_0 is chosen such that the outgoing shock has a Lorentz factor of $W \approx 10$. We set $e_0 = \{6, 4, 6\}$ for $k = \{0, 1, 2\}$, respectively. The adiabatic index is set to $\Gamma = 4/3$.

Three different cases $k = \{0, 1, 2\}$ are simulated to $t = 0.8$, and the solutions at this time are shown in Figure 16 and 17. For $k = \{0, 1\}$ and a base grid of $(N_x, N_y) = (10, 10)$, 10 levels of refinement were used. For the more challenging $k = 2$ case, the equivalent of 11 refinement levels were required to provide the resolution seen in Figure 17. Each case was run with a refinement criterion value of $\epsilon = 10^{-4}$. The Lorentz factor for the shock waves is $\sim$12. All simulations were evolved in their entirety in 2D from the initial conditions using the wavelet method described in this work. We note that the conservation of all conserved variables was externally monitored throughout the simulation. Variation in the conservation of D amounted to less than 0.001%, while variation in the conservation of τ was somewhat larger at 0.8%, visibly due to boundary effects along the z-axis.

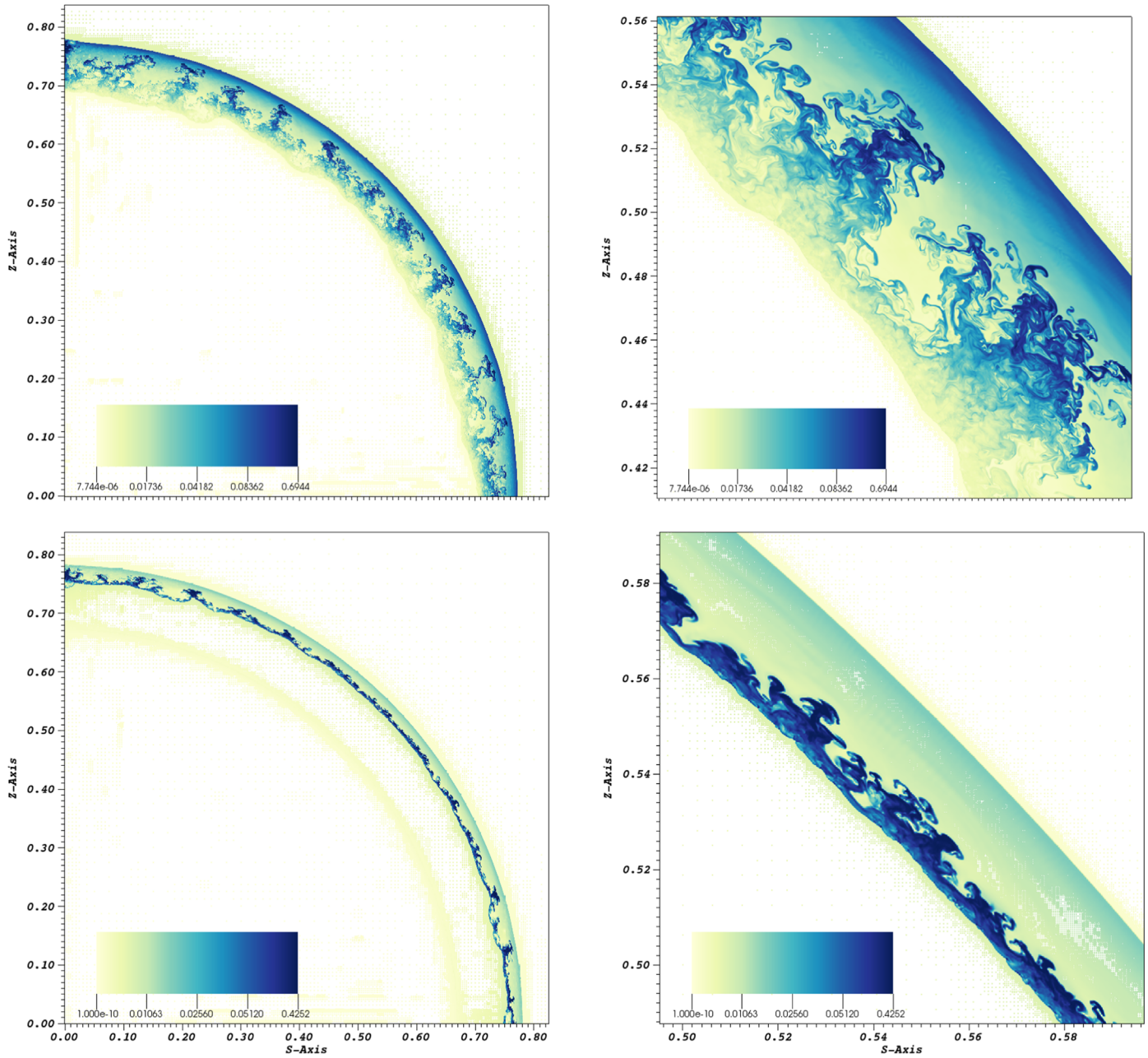

Figure 16. Top panels: $k = 0$ spherical explosion with a decelerating relativistic shock and a mass excess; bottom panels: $k = 1$ case. After a coasting period, the solution develops two shocks separated by a contact discontinuity unstable to the Rayleigh–Taylor instability. Results are shown at $t = 0.8$ for $j_{\max} = 10$ with $\epsilon = 10^{-4}$. This figure also demonstrates the adaptivity of the method; outside the blast wave there is little refinement, while inside the blast the small-scale features drive high levels of refinement, leading to a well-resolved Rayleigh–Taylor instability.

As k is increased, the width of the blast wave decreases. For $k = 0$ the RT instability is well resolved. For $k = 1$, the instability is evident, though with less detail than $k = 0$. In the $k = 2$ case, the width of the blast wave is too narrow to properly resolve the substructures in the instability. That there is an instability is apparent, but it lacks the definition of even the $k = 1$ case. It is likely that these features will be better resolved by further increasing $j_{\max}$ for the $k = 2$ simulation. Indeed, we did so but found that parallelization issues need to be better addressed in order for time- and resource-efficient simulations to be performed. Such is left for future work.

We see little evidence that the Rayleigh–Taylor fingers perturb the forward shock. Indeed, for cases $k = 0$ and $k = 1$, the deepest off-axis turbulent fingers approach the leading edge but remain behind it. A possible exception can be hinted at on the axis where there may be a disturbance of the forward shock. However, we strongly suspect this to be a numerical artifact arising from the boundary treatment on the axis. For a falloff of $k = 2$, while again the resolution is insufficient to be completely definitive, it remains the case that the turbulent region approaches the forward shock, but there appears to be no portion of that region that reaches all the way to the shock. This should be compared with Duffell & MacFadyen (2014), where with different equations of state the Rayleigh–Taylor fingers can penetrate the forward shock. In particular, we agree with their results for an adiabatic constant of $\Gamma = 4/3$ in which no collision arises between the turbulence and the shock front. However, they do find that for a softer equation of state with

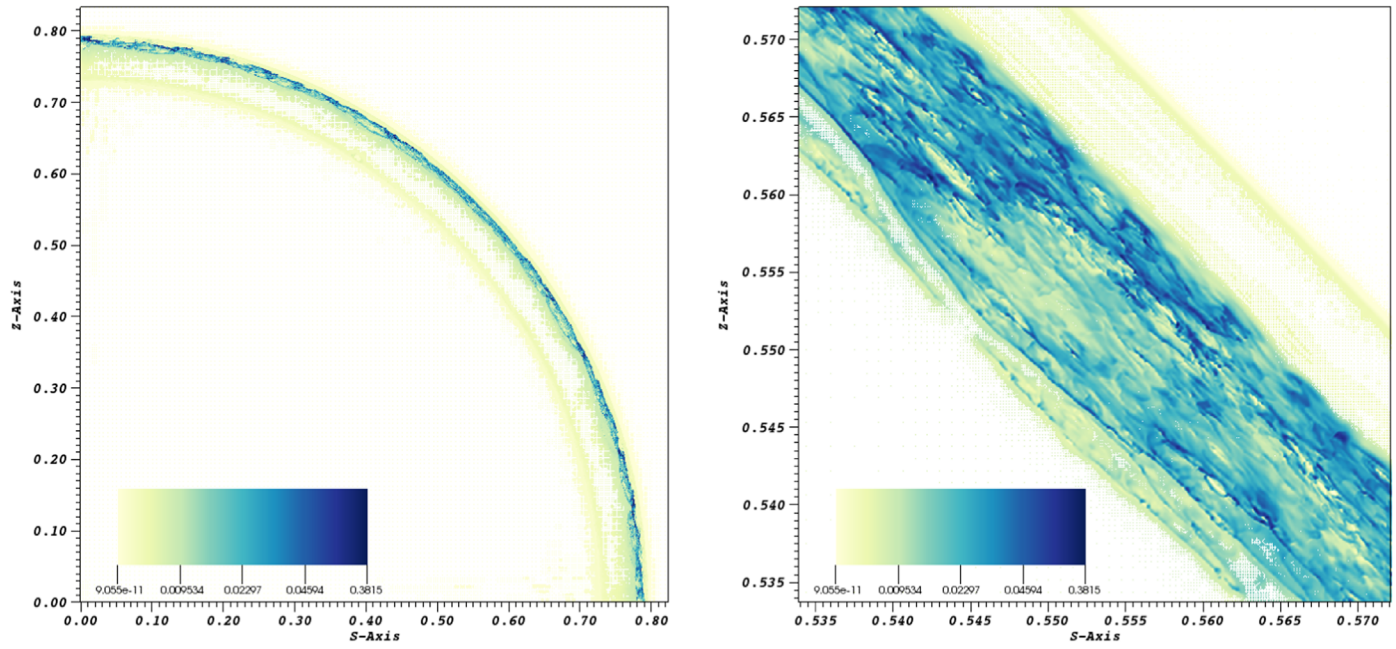

Figure 17. Case $k = 2$ spherical explosion with a decelerating relativistic shock and a mass excess. Results are shown at $t = 0.8$ for $j_{\max} = 8$ with $\epsilon = 10^{-4}$ and a base grid of $(N_x, N_y) = (80, 80)$.

$\Gamma = 1.1$ collision can occur and the Rayleigh–Taylor-induced turbulence can interact with and push the shock front forward.

6. Summary

Motivated by the need to efficiently resolve the many emerging localized features and instabilities present in astrophysics simulations such as the merger of two neutron stars, this work has presented a wavelet-based approach for solving the relativistic hydrodynamic equations. The resulting implementation of this approach, OAHU, has reproduced a number of results in relativistic hydrodynamics, including the 1D shock tube tests of Lora-Clavijo et al. (2015), the relativistic Kelvin–Helmholtz instability (Beckwith & Stone 2011; Radice & Rezzolla 2012), and the Rayleigh–Taylor instability resulting from a gamma-ray burst outflow model (Duffell & MacFadyen 2011). Unlike AMR based on nested boxes, the unstructured grid of collocation points in the wavelet approach conforms to highly localized solution features without creating the box-shaped numerical artifacts typically present in nested box AMR simulations. Further, the approach presented here demonstrates the efficiency and utility of using the coefficients from the wavelet transformation to drive refinement without requiring problem-specific a priori refinement criteria.

The capability and potential of the wavelet method as described here can now be directly applied to any number of other relativistic astrophysical systems and their model equations. These include relativistic magnetohydrodynamics (MHD) and the full, dynamical equations of general relativity. Indeed, combining these will permit robust adaptivity while using largely familiar algorithms for simulating multiscale problems such as the merger of binary compact objects and the formation of black holes, as well their gravitational wave production and emissions from their relativistic outflows (Fernando et al. 2018).

It is a pleasure to thank our long-term collaborators Luis Lehner, Steven L. Liebling, and Carlos Palenzuela, with whom we have had many discussions on adaptive methods for relativistic hydrodynamics. We acknowledge Thomas Sterling, with whom we have discussed the parallel implementation of OAHU. We also thank collaborators Milinda Fernando, Hari Sundar, and Brandon Vernon. This material is based on work supported by the Department of Energy, National Nuclear Security Administration, under award No. DE-NA0002377, the Department of Energy under award No. DE-SC0008809, the National Science Foundation under award Nos. PHY-1308727 and PHY-1607356, and NASA under award No. BL-4363100.

References

Alam, J. M., Kevlahan, N. K.-R., & Vasilyev, O. V. 2006, JCoPh, 214, 829
Alpert, B., Beylkin, G., Coifman, R., & Rokhlin, V. 1993, SIAM J. Sci. Comput., 14, 159
Alpert, B., Beylkin, G., Gines, D., & Vozovoi, L. 2002, JCoPh, 182, 149
Anderson, M., Hirschmann, E. W., Liebling, S. L., & Neilsen, D. 2006, CQGra, 23, 6503
Beckwith, K., & Stone, J. M. 2011, ApJS, 193, 6
Berger, M. J., & Oliger, J. 1984, JCoPh, 53, 484
Bertoluzza, S., & Naldi, G. 1996, Appl. Comput. Harmon. A., 3, 1
Beylkin, G. 1992, SJNA, 6, 1716
Beylkin, G., & Coult, N. 1998, Appl. Comput. Harmon. A., 5, 129
Chegini, N., & Stevenson, R. 2011, SJNA, 49, 182
Colella, P., & Woodward, P. R. 1984, JCoPh, 54, 174
Dahmen, W., Kurdila, A., & Oswald, P. 1997, Multiscale Wavelet Methods for Partial Differential Equations (Cambridge, MA: Academic Press)
Del Zanna, L., & Bucciantini, N. 2002, A&A, 390, 1177
Deslauriers, G., & Dubuc, S. 1989, Constr. Approx., 5, 49
Donoho, D. L. 1992, Interpolating Wavelet Transforms, Tech. Rep. 408 (Stanford, CA: Stanford Univ.)
Duez, M. D., Liu, Y. T., Shapiro, S. L., & Stephens, B. C. 2005, PhRvD, 72, 024029
Duffell, P. C. 2016, ApJS, 226, 2
Duffell, P. C., & MacFadyen, A. I. 2011, ApJS, 197, 15
Duffell, P. C., & MacFadyen, A. I. 2013, ApJ, 775, 87
Duffell, P. C., & MacFadyen, A. I. 2014, ApJL, 791, L1
Einfeldt, B. 1988, SJNA, 25, 294
Etienne, Z. B., Liu, Y. T., Paschalidis, V., & Shapiro, S. L. 2012, PhRvD, 85, 064029
Fernando, M., Neilsen, D., Lim, H., Hirschmann, E., & Sundar, H. 2018, arXiv:1807.06128

Glowinski, R., Lawton, W., Ravachol, M., & Tenenbaum, E. 1990, in Proc. Ninth Int. Conf. on Computing Methods in Applied Science and Engineering, ed. R. Glowinski & A. Lichnewsky (Philadelphia, PA: SIAM), 55
Harten, A., Lax, P. D., & van Leer, B. 1983, SIAMR, 25, 35
Holmström, M. 1999, SIAM J. Sci. Comput., 21, 405
Hopkins, P. F. 2015, MNRAS, 450, 53
Hopkins, P. F., & Raives, M. J. 2016, MNRAS, 455, 51
Kazemi Nasab, A., Kılıçman, A., Pashazadeh Atabakan, Z., & Leong, W. J. 2015, NewA, 34, 178
Latto, A., & Tenenbaum, E. 1990, CRAS, 311, 903
Levinson, A. 2009, ApJL, 705, L213
Levinson, A. 2010, GApFD, 104, 85
Liu, X.-D., & Osher, S. 1998, JCoPh, 142, 304
Lora-Clavijo, F. D., Cruz-Osorio, A., & Guzmán, F. S. 2015, ApJS, 218, 24
Lora-Clavijo, F. D., Cruz-Pérez, J. P., Guzmán, F. S., & González, J. A. 2013, RMxF, E59, 28
Lucas-Serrano, A., Font, J. A., Ibanez, J. M., & Marti, J. M. 2004, A&A, 428, 703
Mertens, F., & Lobanov, A. 2015, A&A, 574, A67
Mösta, P., Mundim, B. C., Faber, J. A., et al. 2014, CQGra, 31, 015005
Noble, S. C., Gammie, C. F., McKinney, J. C., & Del. Zanna, L. 2006, ApJ, 641, 626
Paolucci, S., Zikoski, Z. J., & Grenga, T. 2014a, JCoPh, 272, 842
Paolucci, S., Zikoski, Z. J., & Wirasaet, D. 2014b, JCoPh, 272, 814
Powell, J., Trifirò, D., Cuoco, E., Heng, I. S., & Cavaglià, M. 2015, CQGra, 32, 215012
Qian, S., & Weiss, J. 1993a, ApMaL, 6, 47
Qian, S., & Weiss, J. 1993b, JCoPh, 106, 155
Radice, D., & Rezzolla, L. 2012, A&A, 547, A26
Rastigejev, Y. A., & Paolucci, S. 2006, IJNMF, 52, 749
Regele, J. D., & Vasilyev, O. V. 2009, IJCFD, 23, 503
Rosswog, S. 2009, NewAR, 53, 78
Rosswog, S. 2010, JCoPh, 229, 8591
Rosswog, S. 2015, LRCA, 1, 1
Saito, N., & Beylkin, G. 1993, ITSP, 41, 3584
Springel, V. 2010a, ARA&A, 48, 391
Springel, V. 2010b, MNRAS, 401, 791
Suresh, A., & Huynh, H. T. 1997, JCoPh, 136, 83
Urban, K. 2009, Wavelet Methods for Elliptic Partial Differential Equations (Oxford: Oxford Univ. Press)
Varughese, M. M., von Sachs, R., Stephanou, M., & Bassett, B. A. 2015, MNRAS, 453, 2848
Vasilyev, O. V., & Bowman, C. 2000, JCoPh, 165, 660
Vasilyev, O. V., & Paolucci, S. 1996, JCoPh, 125, 498
Vasilyev, O. V., & Paolucci, S. 1997, JCoPh, 138, 16
Vasilyev, O. V., Paolucci, S., & Sen, M. 1995, JCoPh, 120, 33
Wirasaet, D. 2007, PhD diss., Univ. Notre Dame
Wirasaet, D., & Paolucci, S. 2005, ATJFE, 127, 656
Zhang, W. Q., & MacFadyen, A. I. 2006, ApJS, 164, 255