REFINING A RELATIVISTIC, HYDRODYNAMIC SOLVER: ADMITTING ULTRA-RELATIVISTIC FLOWS

J. P. BERNSTEIN† AND P. A. HUGHES‡

Abstract. We have undertaken the simulation of hydrodynamic flows with bulk Lorentz factors in the range $10^2$–$10^6$. We discuss the application of an existing relativistic, hydrodynamic primitive-variable recovery algorithm to a study of pulsar winds, and, in particular, the refinement necessary to admit such ultra-relativistic flows. We show that the use of an analytical quartic root finder is required for Lorentz factors above $10^2$, but that an iterative quartic root finder, which is known to be robust for Lorentz factors up to at least 25, offers a 24% speed advantage. We demonstrate the existence of a simple diagnostic allowing for a hybrid primitives recovery algorithm that includes an automatic, real-time toggle between the iterative and analytical methods. We further determine the accuracy of the iterative and hybrid algorithms for a comprehensive selection of input parameters.

Key words. methods: numerical, hydrodynamics, relativity, pulsars: general

AMS subject classifications. 85-08, 85A30, 76Y05, 65Y20

1. Introduction. Hydrodynamic simulations have been widely used to model a broad range of physical systems. When the velocities involved are a small fraction of the speed of light and gravity is weak, the classical Newtonian approximation to the equations of motion may be used. However, these two conditions are violated for a host of interesting scenarios, including, for example, heavy ion collision systems [6], relativistic laser systems [3], and many from astrophysics [9] (and references therein), that call for a fully relativistic, hydrodynamic (RHD) treatment. The methods of solution of classical hydrodynamic problems have been successfully adapted to those of a RHD nature, albeit giving rise to significant complication; in particular, the physical quantities of a hydrodynamic flow (the rest-frame mass density, $n$, pressure, $p$, and velocity, $v$) are coupled to the conserved quantities (the laboratory-frame mass density, $R$, momentum density, $M$, and energy density, $E$) via the Lorentz transformation. The fact that modern RHD codes typically evolve the conserved quantities necessitates the recovery of the physical quantities (often referred to as the “primitive variables”) from the conserved quantities in order to obtain the flow velocity. Thus, the calculation of the primitives from the conserved variables has become a critical element of modern RHD codes [10].

In this paper, we present a method for recovering the primitive variables from the conserved quantities representing special relativistic, hydrodynamic (SRHD) flows with bulk Lorentz factors $(\gamma = (1 - v^2)^{-1/2})$, where $v$ is the bulk flow velocity normalized to the speed of light) up to $10^6$. We started with a module from an existing SRHD code used to simulate flows with $\gamma \leq 25$, as described in Duncan & Hughes [4]. Admitting flows with such ultra-relativistic Lorentz factors as $10^6$ required significant refinement to the method used in the existing code to calculate the flow velocity from the conserved quantities. In particular, such extreme Lorentz factors lead to severe numerical problems such as effectively dividing by zero and subtractive cancellation. In §2 we discuss the formalism of recovering the primitives from within the context of

*This work is supported by a NASA Graduate Student Researchers Program grant.
†Department of Astronomy, University of Michigan, 500 Church Street, 830 Dennison, Ann Arbor, MI 48109-1042 (jbernst@umich.edu).
‡Department of Astronomy, University of Michigan, 500 Church Street, 830 Dennison, Ann Arbor, MI 48109-1042 (phughes@umich.edu).
J. P. BERNSTEIN AND P. A. HUGHES

the Euler equations. In §3, we elucidate the details of the refinement to this formalism necessitated by ultra-relativistic flows. We present the refined primitives algorithm in §4 and our results in §5.

2. Recovering the primitive variables from \( R, M, \) and \( E \). In general, recovering the primitives from the conserved quantities reduces to solving a quartic equation, \( Q(v) = 0 \), for the flow velocity in terms of \( R, M, \) and \( E \). Implementation typically involves a numerical root finder to recover the velocity via Newton-Raphson iteration which is very efficient and provides robustness because it is straightforward to ensure that the computed velocity is always less than the speed of light. This is a powerful method that is independent of dimensionality and symmetry. The latter point follows directly from the fact that symmetry is manifest only as a source term in the Euler equations and does not enter into the derivation of \( Q(v) \) (see the axisymmetric example below). Dimensional generality arises because the impact of any set of coordinates \( x_i, i = 1, 2, 3 \cdots, m \), is the distribution of \( M \) into \( m \) components \( M_{x_i} \).

However, one may always write \( M = \sqrt{\sum M_{x_i}^2} \) and recover coordinate generality.

In the case of magnetohydrodynamic (MDH) flows, there are, of course, additional considerations. However, non-magnetic (RHD) simulations still have a significant role to play in astrophysics, e.g., extragalactic jets and pulsar wind nebulae.

As an example, consider the case of the axisymmetric, relativistic Euler equations, which we apply to pulsar winds and which have been applied to extragalactic jets.

In cylindrical coordinates \( \rho \) and \( z \), and defining the evolved-variable, flux, and source vectors

\[
U = (R, M_\rho, M_z, E)^T,
\]

\[
F^\rho = (Rv^\rho, M_\rho v^\rho + p, M_z v^\rho, (E + p)v^\rho)^T,
\]

\[
F^z = (Rv^z, M_\rho v^z, M_z v^z + p, (E + p)v^z)^T,
\]

\[
S = (0, p/\rho, 0, 0)^T,
\]

the Euler equations may be written in almost-conservative form as:

\[
\frac{\partial U}{\partial t} + \frac{1}{\rho} \frac{\partial}{\partial \rho} (\rho F^\rho) + \frac{\partial}{\partial z} (F^z) = S.
\]

The pressure is given by the ideal gas equation of state \( p = (\Gamma - 1)(e - n) \), where \( e \) and \( \Gamma \) are the rest-frame total energy density and the adiabatic index. Note that the velocity and pressure appear explicitly in the relativistic Euler equations, in addition to the evolved variables, and pressure and rest density are needed for the computation of the wave speeds that form the basis of typical numerical hydrodynamic solvers, such as that due to Godunov. We obtain these values by performing a Lorentz transformation where the rest-frame values are required:

\[
R = \gamma n,
\]

\[
M_\rho = \gamma^2 (e + p)v^\rho,
\]

\[
M_z = \gamma^2 (e + p)v^z,
\]

\[
E = \gamma^2 (e + p) - p,
\]

\[
\gamma = (1 - v^2)^{-1/2},
\]

(2.2)

where \( v^2 = (v^\rho)^2 + (v^z)^2 \) and \( M^2 = \gamma^4 (e + p)^2 [(v^\rho)^2 + (v^z)^2] = \gamma^4 (e + p)^2 v^2 \). When the adiabatic index is constant, combining the above equations with the equation of state
creates a closed system which yields the following quartic equation for $v$ in terms of $Y \equiv M/E$ and $Z \equiv R/E$:

$$Q(v) = (\Gamma - 1)^2(Y^2 + Z^2)v^4 - 2\Gamma(\Gamma - 1)Y v^3 + \left[\Gamma^2 + 2(\Gamma - 1)Y^2 - (\Gamma - 1)^2Z^2\right]v^2 - 2\Gamma Y v + Y^2 = 0.$$  

(2.3)

Component velocities, and the rest-frame total energy and mass densities are then given by:

$$v^\rho = \text{sign}(M^\rho)v, \quad v^z = M_z v^\rho, \quad e = E - M^\rho v^\rho - M_z v^z, \quad n = \frac{R}{\gamma}.$$  

This type of formalism enjoys diverse application, in both special and general relativistic settings, from 3D simulations of extragalactic jets [8], to theories of the generation of gamma-ray bursts [16] and the collapse of massive stars to neutron stars and black holes [13].

3. Refinement of the root finder to admit ultra-relativistic flows. A particular implementation of the above has been previously applied to relativistic galactic jets with $\gamma \leq 25$ [4]. The ultra-relativistic nature of pulsar winds necessitated an investigation of the behavior of the primitives algorithm upon taking $\gamma >> 1$. We found that beyond $\gamma \sim 10^2$ the algorithm suffers a severe degradation in accuracy that worsens with increasing Lorentz factor until complete breakdown occurs due to the failure of the Newton-Raphson iteration process used to calculate the flow velocity.

The problem lies in the shape of the quartic, $Q(v)$, one must solve to calculate the primitive variables. The quartic equation as derived using the velocity directly as a variable exhibits two roots for typical physical parameters of the flow (see Fig. 3.1). In general, for $\gamma < 10^2$, the two roots are sufficiently separated on the velocity axis such that the Newton-Raphson (N-R) iteration method converges to the correct zero very quickly and accurately\(^1\). In fact, N-R iteration can be so efficient that it is more desirable to use this method than it is to calculate the roots of the quartic analytically (see §4.2). However, as the Lorentz factor of the flow increases, the roots move progressively closer together and the minimum in $Q(v)$ approaches zero. Eventually, the minimum equals zero to machine accuracy which causes $dQ/dv = 0$ to machine accuracy resulting in a divide by zero and the Newton-Raphson method fails (see Fig. 5.2).

A simple and highly effective solution (see §5 for details) is to rewrite the velocity quartic, $Q(v)$ (Eqn. 2.3), in terms of the Lorentz factor (i.e., make the substitution $v^2 = 1 - \gamma^{-2}$) to obtain the quartic equation in $\gamma$ (recall $Y \equiv M/E$ and $Z \equiv R/E$):

$$Q(\gamma) = \Gamma^2(1 - Y^2)\gamma^4 - 2\Gamma(\Gamma - 1)Z\gamma^3 + \left[2\Gamma(\Gamma - 1)Y^2 + (\Gamma - 1)^2Z^2 - \Gamma^2\right]\gamma^2 + 2\Gamma(\Gamma - 1)Z\gamma - (\Gamma - 1)^2(Y^2 + Z^2) = 0.$$  

(3.1)

\(^1\)For $Y < 0.9$ and $Z > 10^{-5}$ (corresponding to $\gamma < 2$), the roots approach each other sufficiently such that the incorrect root is selected (see §4).
Fig. 3.1. The left-hand plots show the shape of the Lorentz factor quartic over a run of Lorentz factors for a mildly relativistic flow ($\gamma_0 = 1.5$) and an ultra-relativistic flow ($\gamma_0 = 10^6$). The right-hand plots show the shape of the velocity quartic over a run of velocity for a mildly relativistic flow ($\gamma_0 \approx 1.5$) and a highly (but not ultra) relativistic flow ($\gamma_0 \approx 10^2$). The crosses mark the location of the physical root. From the plot in the lower right, one can see the onset of the zero derivative problem as the roots are not distinguishable from each other or the local minimum even on a scale of $10^{-13}$, which begins to encroach on the limit of 8-byte accuracy.

As Fig. 3.1 exemplifies, $Q(\gamma)$ exhibits a single root for the physical range $\gamma \geq 1$. However, Newton-Raphson iteration also fails in this case at high Lorentz factors because of the steepness of the rise in $Q(\gamma)$ through the root.

Thus, we are forced to use an analytical method of solving a quartic. Below, we discuss our implementation.

### 3.1. Solving a quartic equation.

We use the prescription due to Bronshtein & Semendyayev [1] in order to analytically solve for the roots of a quartic. We chose this method because it provides equations for the roots of the quartic that are the most amenable (of the methods surveyed) to integration into a computational environment. We proceed as follows\(^2\).

Given a quartic equation in $x$:

$$a_4 x^4 + a_3 x^3 + a_2 x^2 + a_1 x + a_0 = 0, \quad a_n \in \mathbb{R}, \quad a_4 \neq 0,$$

(normalizing the equation (dividing by $a_4$) and making the substitution $y = x + \frac{a_3}{4a_4}$)

\(^2\)In order to provide a complete picture of our method, which includes steps not found in Bronshtein & Semendyayev [1], we reproduce some sections of that text.
results in the reduced form:

\[ y^4 + Py^2 + Qy + R = 0, \]

where, defining \( \tilde{a}_n \equiv a_n/a_4 \):

\[ P \equiv -\frac{3}{8} \tilde{a}_3^2 + \tilde{a}_2, \]

\[ Q \equiv \left( \frac{\tilde{a}_3}{2} \right)^3 - \left( \frac{\tilde{a}_2}{2} \right) \tilde{a}_2 + \tilde{a}_1, \]

\[ R \equiv -3 \left( \frac{\tilde{a}_3}{4} \right)^4 + \left( \frac{\tilde{a}_3}{4} \right)^2 \tilde{a}_2 - \left( \frac{\tilde{a}_3}{4} \right) \tilde{a}_1 + \tilde{a}_0. \]

These coefficients allow the definition of the cubic resolvent:

\[ u^3 + 2Pu^2 + (P^2 - 4R)u - Q^2 = 0, \]

(3.3)

upon whose solutions the solutions of the original quartic (Eqn. 3.2) depend. The product of the solutions of the cubic resolvent \( u_1u_2u_3 = Q^2 \) must be positive by Vieta’s theorem. The characteristics of the quartic’s roots depend on the nature of the roots of the cubic resolvent (see Tab. 3.1).

**Table 3.1**

<table>
<thead>
<tr>
<th>Solutions of the cubic resolvent</th>
<th>Solutions of the quartic equation</th>
</tr>
</thead>
<tbody>
<tr>
<td>All real and positive</td>
<td>All real</td>
</tr>
<tr>
<td>All real, one positive</td>
<td>Two complex conjugate (cc) pairs</td>
</tr>
<tr>
<td>One real, one cc pair</td>
<td>Two real, one cc pair</td>
</tr>
</tbody>
</table>

Given the solutions of the cubic resolvent \( u_1, u_2, \) and \( u_3 \), the solutions of the quartic (Eqn. 3.2) are

\[ x_1 = \frac{1}{2}(\sqrt{u_1} + \sqrt{u_2} + \sqrt{u_3}) - \frac{a_3}{4a_4}, \]

\[ x_2 = \frac{1}{2}(\sqrt{u_1} - \sqrt{u_2} - \sqrt{u_3}) - \frac{a_3}{4a_4}, \]

\[ x_3 = \frac{1}{2}(-\sqrt{u_1} + \sqrt{u_2} - \sqrt{u_3}) - \frac{a_3}{4a_4}, \]

\[ x_4 = \frac{1}{2}(-\sqrt{u_1} - \sqrt{u_2} + \sqrt{u_3}) - \frac{a_3}{4a_4}. \]

(3.4)

### 3.2. Solving a cubic equation.

The equations of the previous section reduce the problem of solving a quartic equation to that of solving a cubic equation (i.e., the cubic resolvent of Eqn. 3.3).

Once again following Bronshtein and Semendyayev \[\text{II}\] (note the similarity to the method in the previous section), given a cubic equation:

\[ b_3u^3 + b_2u^2 + b_1u + b_0 = 0, \quad b_n \in \mathbb{R}, \quad b_3 \neq 0, \]

(3.5)
normalizing the equation and making the substitution \( v = u + b_2/3b_3 \) results in the reduced form:

\[
v^3 + pv + q = 0,
\]

where, defining \( \tilde{b}_n \equiv b_n/b_3 \):

\[
p \equiv -\frac{1}{3} \tilde{b}_2^2 + \tilde{b}_1, \\
q \equiv 2 \left( \frac{b_2}{3} \right)^3 - \left( \frac{b_2}{3} \right) \tilde{b}_1 + \tilde{b}_0.
\]

These coefficients allow the definition of the discriminant:

\[
D \equiv \left( \frac{p}{3} \right)^3 + \left( \frac{q}{2} \right)^2,
\]

upon which the characteristics of the solutions of the cubic equation depend (see Tab. 3.2).

**Table 3.2**
The dependence of the solutions of a cubic equation on the sign of the discriminant (assuming a real variable).

<table>
<thead>
<tr>
<th>D</th>
<th>Solutions of the cubic equation</th>
</tr>
</thead>
<tbody>
<tr>
<td>positive</td>
<td>one real, one complex conjugate pair</td>
</tr>
<tr>
<td>negative</td>
<td>all real and distinct</td>
</tr>
<tr>
<td>= 0</td>
<td>all real, two (one, if ( p = q = 0 )) distinct</td>
</tr>
</tbody>
</table>

Given \( p, q, \) and \( D \), Cardano’s formula for the reduced form of the cubic leads to the solutions of the original cubic (Eqn. 3.5):

\[
\begin{align*}
 u_1 &= s + t - \frac{b_2}{3b_3}, \\
 u_2 &= -\frac{1}{2}(s + t) - \frac{b_2}{3b_3} + \frac{\sqrt{3}}{2}(s - t), \\
 u_3 &= -\frac{1}{2}(s + t) - \frac{b_2}{3b_3} - \frac{\sqrt{3}}{2}(s - t),
\end{align*}
\]

(3.6)

where:

\[
\begin{align*}
 s &= \sqrt[3]{-\frac{1}{2}q + \sqrt{D}}, \\
 t &= \sqrt[3]{-\frac{1}{2}q - \sqrt{D}}, \\
 i &= \sqrt{-1}.
\end{align*}
\]

If \( D \leq 0 \), the cubic has three real roots, subject to the following two subcases, and the four real roots of the quartic follow directly from Eqn. 3.4. If \( D = 0 \), then \( s = t \) and the cubic has three real solutions that follow directly from Eqn. 3.6 from
which one can see that two are degenerate. If $D < 0$, the cubic has three distinct real roots. Obtaining these solutions via Eqn. 3.6 requires intermediate complex arithmetic. However, this may be circumvented by making the substitutions:

$$\rho = \sqrt{-\left(\frac{p}{3}\right)}$$

$$\cos(\phi) = -\frac{q}{2\rho},$$

in which case the solutions of the cubic (Eqn. 3.5) are:

$$u_1 = 2\sqrt[3]{\rho} \cos\left(\frac{\phi}{3}\right) - \frac{b_2}{3b_3},$$

$$u_2 = 2\sqrt[3]{\rho} \cos\left(\phi + \frac{2\pi}{3}\right) - \frac{b_2}{3b_3},$$

$$u_3 = 2\sqrt[3]{\rho} \cos\left(\phi + \frac{4\pi}{3}\right) - \frac{b_2}{3b_3}.$$

(3.7)

If $D > 0$, then the cubic has one real root and a pair of complex conjugate roots and the quartic has two real roots and a pair of complex conjugate roots (see Tab. 3.1). Finding the roots of the quartic involves intermediate complex arithmetic which may be circumvented as follows. Defining:

$$R \equiv -\frac{1}{2}(s + t) - \frac{b_2}{3b_3},$$

$$C \equiv \frac{\sqrt{3}}{2}(s - t),$$

Eqn. 3.6 may be rewritten as:

$$u_1 = s + t - \frac{b_2}{3b_3},$$

$$u_2 = R + iC,$$

$$u_3 = R - iC.$$

Next, we have $u_{2,3} = \sqrt{R^2 + C^2}e^{\pm iC/R}$. We then obtain the roots of the quartic from Eqn. 3.4

$$x_{1,2} = \frac{\sqrt{u_1}}{2} - \frac{a_3}{4a_4} \pm \frac{\sqrt{R^2 + C^2} \cos\left(\frac{C}{2R}\right)}{C},$$

$$x_{3,4} = \frac{\sqrt{u_1}}{2} - \frac{a_3}{4a_4} \pm i\frac{\sqrt{R^2 + C^2} \sin\left(\frac{C}{2R}\right)}{C}.$$ 

(3.8)

Note that $x_1$ and $x_2$ are the two real solutions.

4. The refined primitives algorithm. Using the method above we created a SRHD primitive algorithm called “REST_FRAME”. Given the speed advantage of the iterative root finder (see 4.2), it a desirable choice over the analytical method within its regime of applicability, i.e., for low Lorentz factors. As Fig. 5.2 shows, the iterative root finder is accurate to order $10^{-4}$ (see 4.4) for a sizable region of parameter space including all $R/E$ such that $\log(R/E) \geq -(7/9) \times \log(1 - M/E) - 7$. Therefore, for a given $M/E$ and $R/E$, we check if this inequality is true; if (not) so, we call the (analytical) iterative root finder (see 4.4).
4.1. Pseudo-code. REST_FRAME calculates the primitive variables given the conservative variables and the adiabatic index as represented in the following pseudo-code (note this is a 2D example):

PROCEDURE REST_FRAME
RECEIVED FROM PARENT PROGRAM: \(Y, Z\)
RETURNED TO PARENT PROGRAM: \(\gamma, v, C\)

Comment: recall \(Y \equiv \frac{M}{E}\) and \(Z \equiv \frac{R}{E}\)
Comment: \(C\) is returned \(< 0\) for code failures

GLOBAL VARIABLE: \(\Gamma\)
SET VALUE OF \(m_{\text{underflow}}\)
SET VALUE OF \(v_{\text{tol}}\)

Comment: determines iterative method velocity accuracy
Comment: we set \(v_{\text{tol}} = 10^{-8}, 10^{-10}, 10^{-12}, 10^{-14}\)
Comment: for \(- \log(1 - Y) < 8.3, < 10.3, < 12.3\), otherwise, respectively

SET \(M = \sqrt{M_x^2 + M_y^2}\)
IF \(M < m_{\text{underflow}}\) THEN
\(v = 0, \gamma = 1\)
Comment: avoids code failure if \(v\) is numerically zero
ELSE
TEST FOR UNPHYSICAL PARAMETERS
IF PASSED, SET \(C\) NEGATIVE AND RETURN
IF \(\log(R/E) \geq -(7/9) \times \log(1 - M/E) - 7\), THEN
CALL ITERATIVE_QUARTIC\((Y, Z, v_{\text{tol}}, v, C)\)
Comment: updates \(v_{n-1}\) to \(v_n\) using \(n\) cycles of Newton-Raphson iteration
Comment: returns \(v = v_n\) when \(|v_n - v_{n-1}| \leq v_{\text{tol}}\)
IF \(C < 0\), THEN
COMMENT: this means the iteration failed to converge
RETURN
ELSE
\(\gamma = \sqrt{\frac{1}{1 - v^2}}\)
END IF
ELSE
CALL ANALYTICAL_QUARTIC\((Y, Z, \gamma)\)
Comment: calculates \(\gamma\) using analytical solution – see below
\(v = \sqrt{1 - \frac{1}{\gamma^2}}\)
END IF
END IF
END PROCEDURE REST_FRAME

PROCEDURE ANALYTICAL_QUARTIC
Comment: see §3.1 for equations
RECEIVED FROM PARENT PROGRAM: \(Y, Z\)
RETURNED TO PARENT PROGRAM: \(\gamma\)

GLOBAL VARIABLE: \(\Gamma\)
\(a_3 = 2\Gamma(\Gamma - 1)Z(Y^{-2} + 1)\)
\(a_2 = (\Gamma^2 - 2\Gamma(\Gamma - 1)Y^2 - (\Gamma - 1)^2Z^2)(Y^{-2} + 1)\)
\[ \hat{a}_1 = -a_3 \]
\[ \hat{a}_0 = (\Gamma - 1)^2(Y^2 + Z^2)(Y^{-2} + 1) \]
\[ \hat{a}_4 = 1 + Y^2 - a_0 - a_2 \]

Comment: coefficients recast to counter subtractive cancellation – see §5

NORMALIZE COEFFICIENTS TO \( a_4 \)

Comment: e.g., \( a_{3N} = a_3/a_4 \)

CALCULATE CUBIC RESOLVENT COEFFICIENTS
CALCULATE DISCRIMINANT, \( D \)

IF \( D \leq 0 \) THEN

WRITE ERROR MESSAGE AND STOP

Comment: exploration suggests \( D \leq 0 \) is unphysical but formal proof is elusive
Comment: thus, we leave \( D \leq 0 \) uncoded with a error flag just in case

ELSE

Comment: \( D > 0 \) ⇒ \( Q(\gamma) \) has 2 real roots (see Tab. 3.1 & 3.2)

CALCULATE ROOTS OF CUBIC RESOLVENT

Comment: the cubic has one real root and a pair of complex conjugate roots

IF REAL ROOT < 0, SET REAL ROOT = 0

Comment: the real root cannot be less than zero analytically
Comment: numerically, however, it can have a very small negative value

CALCULATE THE TWO REAL ROOTS OF THE QUARTIC
TEST FOR TWO OR NO PHYSICAL ROOTS

IF PASSED, WRITE ERROR MESSAGE, AND RETURN
IF FAILED, SET \( \gamma = \) PHYSICAL ROOT

END IF

END PROCEDURE ANALYTICAL QUARTIC

4.2. Code timing. Using the Intel Fortran library function CPU_TIME, we calculated the CPU time required to execute \( 5 \times 10^7 \) calls to REST_FRAME for \( Y = 0.9975 \) & \( Z = 1 \times 10^{-4} \) (\( \gamma \sim 10 \)) using the Newton-Raphson iterative method with \( Q(v) \) and 8-byte arithmetic, the analytical method with \( Q(\gamma) \) and both 8-byte and 16-byte arithmetic (we investigated the use of 16-byte arithmetic due to an issue with subtractive cancellation – see §3). The CPU time for each of these scenarios is 29.5, 36.5 (averaged over ten runs and rounded to the nearest half second), and \( \sim 11650 \) seconds (one run only), respectively. This indicates that while using the 8-byte analytical method is satisfactory, it is advantageous to use the iterative method when Lorentz factors are sufficiently low, and that the use of 16-byte arithmetic is a nonviable option. This result is not surprising as the accuracy of Newton-Raphson iteration improves by approximately one decimal place per iterative step \[4\], and the relative inefficiency of 16-byte arithmetic is a known issue.

4.3. Input parameters for PWNe. We will implement our refined primitives algorithm within the context of simulating the interaction of a light, fast pulsar wind with a dense, slow-moving ambient medium arising from the high-space velocity that is typical of pulsars \[2\]. This interaction gives rise to the classic structure of forward and reverse shocks separated by a contact surface \[17\]. Pulsar winds have bulk Lorentz factors on the order of \( 10^6 \) with values of \( 10^2-10^6 \) realized. Though we will identify a physical region of hydrodynamic parameter space applicable to this system, in the next section we consider a comprehensive region of parameter space to assess the robustness of our routine for a range of applications.

We specify an initial state through the cross-flow velocity, \( v_a \) in units of the speed
of light, Mach number, \( \mu_a \), and mass density, \( n_a \), of the ambient medium, and the Lorentz factor, \( \gamma_o \), pressure, \( p_o \), and mass density, \( n_o \), of the wind (or, more generally, the “outflow”). The ambient velocity flow arises from the space velocity of the pulsar, which is typically 400-500 km s\(^{-1} \). We adopt a value of 500 km s\(^{-1} \), which implies \( v_a = (5/3) \times 10^{-3} \). We further select values of \( \mu_a = 5, 50, 500 \) representing ambient-medium sound speeds of 100, 10, and 1 km s\(^{-1} \), respectively. The value of \( n_a \) is arbitrary and \( n_o \) is scaled accordingly. We are interested in \( \gamma_o = 10^2 - 10^6 \).

The outflow streams relativistically into the ambient medium generating a strong shock. We derive a value for \( p_o \) from the assumption that the outflow is interacting with the ambient medium, requiring that the momentum flux be comparable on either side of this shock. This is the fundamental premise of our study; if the fluxes were not comparable, then either the ambient flow or outflow would dominate and the problem would be uninteresting. The momentum flux of the ambient medium and outflow are, respectively:

\[
F_{M,a} = n_a v_a^2 + p_a, \\
F_{M,o} = \gamma_o^2 (e_o + p_o) v_o^2 + p_o.
\]

For our ultra-relativistic outflow, \( p_o \gg n_o \Rightarrow e_o \rightarrow 3p_o \), and \( v_o \rightarrow 1 \), and, for the ambient medium, we have \( n_a v_a^2 \gg p_a \). Applying these conditions, and noting that \( \gamma_o^2 p_o \gg p_a \), gives:

\[
p_o \sim n_a \left( \frac{v_a^2}{2 \gamma_o} \right)^2 \sim 10^{-19} \text{ for } \gamma_o = 10^6, n_a = 1.
\]

We are then free to pick any \( n_o \) provided the conditions of a light, relativistic outflow are met, i.e., \( n_a, p_o \gg n_o \). We select \( n_o = 10^{-l} p_o, 3 < l < 6 \). This clearly satisfies \( p_o \gg n_o \) and one may verify it satisfies \( n_a \gg n_o \) by noting that the equation for \( p_o \) above implies \( n_o \gg p_o \) since \( \gamma_o^2 \gg v_a^2 \) for the flows of interest here. In what follows, we consider a comprehensive set of parameters that is of general interest beyond our application to PWNe.

5. Results. The input parameters for our primitives algorithm are the ratios of the laboratory-frame momentum and mass densities to the laboratory-frame energy density (recall \( Y \equiv M/E \) and \( Z \equiv R/E \)) both of which must be less than unity in order for solutions of Eqn. 22 to exist. In addition, the condition \( Y^2 + Z^2 < 1 \) must be met. Along with the fact that \( Y \) and \( Z \) must also be positive, this defines the comprehensive and physical input parameter space to be \( 0 < Y, Z < 1 \) such that \( Y^2 + Z^2 < 1 \). We tested the accuracy of our iterative and hybrid primitives algorithms within this space as follows.

First, we elected to use the quantities \(- \log(1 - Y)\) and \( \log(Z) \) to define the accuracy-search space because we are most interested in light, highly relativistic flows and these two quantities span \( 0.9 < Y < 1 \) and \( Z < 1 \) for all positive values greater than 1 and negative values less than -1, respectively. We selected \( 0 < - \log(1 - Y) < 13 \) and \(-13 < \log(Z) < 0 \) corresponding to Lorentz factors (\( \gamma \)) between 1 and \( 2 \times 10^6 \). We chose a range with a maximal \( \gamma \) slightly above \( 1 \times 10^6 \) in order to completely bound the PWN parameter space defined in the previous section.

Setting the relativistic value \( \Gamma = 4/3 \) and using 1300 points for both \(- \log(1 - Y)\) and \( \log(Z) \), we tested the accuracy of REST_FRAME by passing it \( Y \) and \( Z \), choosing \( E = 1 \), and using the returned primitive quantities to derive the calculated energy density \( E_c \), and calculating the difference \( |1 - E_c/E| \equiv \delta E/E \). We chose this estimate
of the error because $\delta E/E \sim \delta \gamma/\gamma$ and $\delta \gamma/\gamma$ is tied to the accuracy of the numerical, hydrodynamic technique (see the final paragraph in this section).

Our results for the Newton-Raphson (N-R) iterative method and the hybrid method are shown in Figs. 5.2 & 5.3 where white, light grey, medium grey, dark grey, and hatched regions correspond to accuracy of order at least $10^{-4}$, at least $10^{-3}$, worse than $10^{-3}$, failure, and unphysical input ($Z^2 \geq 1 - Y^2$), respectively. We chose an accuracy of order $10^{-4}$ as the upper cutoff because N-R iteration returns accuracies on this order for $\gamma < 25$ and relativistic, hydrodynamic simulations of galactic jets by Duncan and Hughes [4] produced robust results for Lorentz factors of at least 25 using N-R iteration. An additional result of interest is that the ultra-relativistic approximation for $v$ (i.e., taking $R = 0$ thereby reducing $Q(v) = 0$ to a quadratic equation) manages an accuracy of at least $10^{-4}$ for a large portion of the physical $Y-Z$ plane (see Fig. 5.1).

Fig. 5.1. The accuracy (estimated as $\delta E/E$) of the ultra-relativistic approximation of the flow velocity where white, light grey, medium grey, and hatched regions correspond to an accuracy of order at least $10^{-4}$, at least $10^{-3}$, worse than $10^{-3}$, and unphysical input ($R^2/E^2 \geq 1 - M^2/E^2$), respectively. Note that the Lorentz factor varies from order 1 at the far left to order $10^6$ at the far right. The accuracy degradation at the extreme right is due to the fact that the fractional error in the Lorentz factor is proportional to the fractional error in the velocity divided by $1 - v^2$ which diverges as $v \to 1$.

Fig. 5.2 shows the accuracy of the N-R iterative method. There are several noteworthy features. First is the presence of a sizable region representing $Y < 0.999999$ ($\gamma < 500$) and $Z > 5 \times 10^{-8}$ within which accuracy is generally significantly better than $10^{-4}$. Second is that N-R iteration is unreliable due to sporadic failures for all $Y$ & $Z$ such that $Z < 5 \times 10^{-8}$ and for an ever increasing fraction of $Z > 5 \times 10^{-8}$ as $Y$ increases until accuracy becomes unacceptable or the code fails outright for all $Y$ & $Z$ such that $Y > 0.999999$. Failures are due to divide by zero (see §3) or nonconvergence within a reasonable number of iterations. In addition, though N-R iteration has been widely established as the primitives recovery method of choice for flows with Lorentz factors less than order $10^2$, we found for $Y < 0.9$ ($\gamma < 2$) and $Z > 10^{-5}$ our N-R algorithm suffered an unacceptable degradation in accuracy. Our original method for
choosing the initial velocity estimate \( v_i \) places \( v_i \) between the roots \( v_1 \) and \( v_2 \equiv v \) with the latter being the physical root and \( v_2 > v_1 \), and in this region \( v_i \) is sufficiently close to \( v_1 \) that N-R iteration converges to the incorrect root. Specifically, the common approach \([4],[12]\) to estimating an initial velocity is to bracket \( v \) with an upper and lower bound:

\[
\begin{align*}
v_{\text{max}} &= \min(1, Y + \delta), \\
\delta &\sim 10^{-6}, \\
v_{\text{min}} &= \frac{\Gamma - \sqrt{\Gamma^2 - 4(\Gamma - 1)Y^2}}{2Y(\Gamma - 1)},
\end{align*}
\]

where \( \delta \sim 10^{-6} \) and \( v_{\text{min}} \) is derived by setting \( R = 0 \) when deriving \( Q(v) \) (i.e., by taking the ultra-relativistic limit). The initial velocity is then \( v_i = \frac{(v_{\text{min}} + v_{\text{max}})}{2} + \eta \), where \( \eta = (1 - Z)(v_{\text{min}} - v_{\text{max}}) \) for \( v_{\text{max}} > \epsilon \) and \( \eta = 0 \) otherwise (\( \epsilon \) order \( 10^{-9} \)).

These definitions guarantee that the physical root \( v \) is in the range \( v_{\text{min}} < v < v_{\text{max}} \), which leads to convergence to the incorrect root upon the convergence of the roots.

Our solution (used in constructing Fig. 5.2) is a simpler initial estimate of \( v_i = v_{\text{max}} \), which guarantees that \( v_i \) is “uphill” from \( v \) for all physical \( Y - Z \) space and that N-R iteration converges on \( v \).

Fig. 5.2. The accuracy (estimated as \( \delta E/E \)) of the Newton-Raphson (N-R) iterative primitives algorithm where white, light grey, medium grey, dark grey, and hatched regions correspond, respectively, to an accuracy of order at least \( 10^{-4} \), at least \( 10^{-3} \), worse than \( 10^{-3} \), failure, and unphysical input \( (R^2/E^2 \geq 1 - M^2/E^2) \). Note that the Lorentz factor varies from order 1 at the far left to order \( 10^6 \) at the far right. There is a sizable white region representing \( M/E < 0.999999 \) \( (\gamma < 500) \) and \( R/E > 5 \times 10^{-8} \) within which accuracy is generally significantly better than \( 10^{-4} \). N-R iteration is unreliable due to sporadic failures for all \( M/E \) and \( R/E \) such that \( R/E < 5 \times 10^{-8} \) and for an ever increasing fraction of \( R/E > 5 \times 10^{-8} \) as \( M/E \) increases until accuracy becomes unacceptable or the code fails outright for \( M/E \) and \( R/E \) such that \( M/E > 0.999999 \). Failures are due to divide by zero (see \S 3) or nonconvergence within a reasonable number of iterations.

Fig. 5.3 shows that our hybrid algorithm REST_FRAME is accurate to at least \( 10^{-4} \) for all but a smattering of \( Z \) at the highest \( Y \). In fact, it is significantly more accurate over the majority of the physical portion of the \( Y - Z \) plane. The space between the parallel lines represents the PWN input parameters discussed in the
previous section. The accuracy degradation for $Y$ closest to 1 ($\gamma \sim 10^6$) is due to subtractive cancellation in the fourth-order coefficient of $Q(\gamma)$ (see Eqs. 3.1) as $Y \to 1$. We find that multiplying $Q(\gamma)$ by $(Y^2 - Y^{-2})$ and rewriting the new $a_4(\tilde{a}_4)$ in terms of the new $a_2(\tilde{a}_2)$ and new $a_0(\tilde{a}_0)$, e.g., $\tilde{a}_4 = 1 + Y^2 - \tilde{a}_0 - \tilde{a}_2$, improves the accuracy somewhat, but does not entirely mitigate the problem. The issue of accuracy loss at large Lorentz factors in 8-byte primitives algorithms is a known issue [11] for which we know of no complete 8-byte solution. Employing 16-byte arithmetic provides spectacular accuracy, but introduces an unacceptable increase in run time (see §4.2).

**Fig. 5.3.** The accuracy (estimated as $\delta E/E$) of the hybrid primitives algorithm where white, light grey, and hatched regions correspond, respectively, to an accuracy of order at least $10^{-4}$, at least $10^{-3}$, and unphysical input ($R^2/E^2 \geq 1 - M^2/E^2$). Note that the Lorentz factor varies from order 1 at the far left to order $10^6$ at the far right. The space between the parallel lines represents PWNe input parameter space. The accuracy degradation at the extreme right is due to subtractive cancellation in the 4th-order coefficient of the Lorentz-factor quartic as $M/E \to 1$.

The issue of what constitutes an acceptable error in the calculated Lorentz factor is driven by the fact that a fractional error in $\gamma$ translates to the same fractional error in $p$ and $n$ which are needed to calculate the wave speeds that form the basis of the numerical, hydrodynamic technique. Our SRHD solver uses a Godunov scheme [5] which approximates the solution to the local Riemann problem by employing an estimate of the wave speeds. We do not know a priori how accurately this estimate needs to be. Thus, we will proceed with 8-byte simulations of pulsar winds confidently with the knowledge that we can use a known-solution shock tube problem [14] to validate the accuracy of the computation of well-defined flow structures as we approach the highest Lorentz factors. It is also noteworthy that while $\gamma = 10^6$ is the canonical bulk Lorentz factor for pulsar winds, $\gamma = 10^4$ and $10^5$ are still in the ultra-relativistic regime, and it may very well prove to be that these Lorentz factors are high enough to elucidate the general ultra-relativistic, hydrodynamic features of such a system. The hybrid algorithm achieves accuracies of at least $10^{-6}$ for $\gamma \sim 10^5$, which is safely in the acceptable accuracy regime.
6. Conclusion. We discussed the application of an existing special relativistic, hydrodynamic (SRHD) primitive-variable recovery algorithm to ultra-relativistic flows (Lorentz factor, $\gamma$, of $10^2$–$10^6$) and the refinement necessary to the numerical velocity root finder. We found that the velocity quartic, $Q(v)$, exhibits dual roots in the physical velocity range that move progressively closer together for larger $\gamma$ leading to a divide by zero and the failure of the Newton-Raphson iteration method employed by the existing primitives algorithm. Our solution is to recast the quartic to be a function, $Q(\gamma)$, of $\gamma$. We demonstrated that $Q(\gamma)$ exhibits only one physical root. However, Newton-Raphson iteration also fails in this case at high $\gamma$, due to the extreme slope of the quartic near the root, necessitating the use of an analytical numerical root finder.

Our timing analysis indicates that using $Q(\gamma)$ with the 8-byte analytical root finder increases run time by 24% compared to using $Q(v)$ with the 8-byte iterative root finder (based on 10 trial runs), while using $Q(\gamma)$ with the 16-byte analytical root finder balloons run time by a factor of approximately 400. The iterative root finder is accurate to order $10^{-4}$ for a sizable region of parameter space including all $R/E$ such that $\log(R/E) \geq -(7/9) \times \log(1-M/E) - 7$. Therefore, for a given $M/E$ and $R/E$, we check if this inequality is true and call the iterative or analytical root finder accordingly. In addition, our exploration of parameter space suggests that the discriminant of the cubic resolvent (as defined in this paper) will always be positive for physical flows. Therefore, we did not include code for negative discriminants in our routine. Formal proof remains elusive, however, leaving potential for future work.

We have shown that REST_FRAME is capable of calculating the primitive variables from the conserved variables to an accuracy of at least $O(10^{-4})$ for Lorentz factors up to $10^6$, with significantly better accuracy for Lorentz factors $\leq 10^5$, and slightly worse (order $10^{-3}$) for a small portion of the space corresponding to the highest Lorentz factors. We traced the degradation in accuracy for larger Lorentz factors to the effect of subtractive cancellation. Past studies have shown that an accuracy of order $10^{-4}$ is capable of robustly capturing hydrodynamic structures. Therefore, we will proceed with 8-byte SRHD simulations with confidence, but with more caution for the highest Lorentz factors.

Acknowledgments. This work is supported by NASA Graduate Student Researchers Program grant #NGT5-159.

REFERENCES


