The Astrophysical Journal Supplement Series, 170:228–242, 2007 May
© 2007. The American Astronomical Society. All rights reserved. Printed in U.S.A.

PLUTO: A NUMERICAL CODE FOR COMPUTATIONAL ASTROPHYSICS

A. Mignone,[1,2] G. Bodo,[2] S. Massaglia,[1] T. Matsakos,[1] O. Tesileanu,[1] C. Zanni,[3] and A. Ferrari[1]

Received 2006 November 5; accepted 2007 January 28

ABSTRACT

We present a new numerical code, PLUTO, for the solution of hypersonic flows in 1, 2, and 3 spatial dimensions and different systems of coordinates. The code provides a multiphysics, multialgorithm modular environment particularly oriented toward the treatment of astrophysical flows in presence of discontinuities. Different hydrodynamic modules and algorithms may be independently selected to properly describe Newtonian, relativistic, MHD, or relativistic MHD fluids. The modular structure exploits a general framework for integrating a system of conservation laws, built on modern Godunov-type shock-capturing schemes. Although a plethora of numerical methods has been successfully developed over the past two decades, the vast majority shares a common discretization recipe, involving three general steps: a piecewise polynomial reconstruction followed by the solution of Riemann problems at zone interfaces and a final evolution stage. We have checked and validated the code against several benchmarks available in literature. Test problems in 1, 2, and 3 dimensions are discussed.

Subject headings: hydrodynamics — methods: numerical — MHD — relativity–shock waves

Online material: color figures

1. INTRODUCTION

Theoretical models based on direct numerical simulations have unveiled a new way toward a better comprehension of the rich and complex phenomenology associated with astrophysical plasmas.

Finite difference codes such as ZEUS (Stone & Norman 1992a, 1992b) or NIRVANA+ (Ziegler 1998) inaugurated this novel era and have been used by an increasingly large fraction of researchers nowadays. However, as reported in Falle (2002), the lack of upwinding techniques and conservation properties have progressively moved scientist's attention toward more accurate and robust methods. In this respect, the successful employment of the so-called high-resolution shock-capturing (HRSC) schemes have revealed a mighty tool to investigate fluid dynamics in nonlinear regimes. Some of the motivations behind their growing popularity is the ability to model strongly supersonic flows while retaining robustness and stability. The unfamiliar reader is referred to the books of Toro (1997), LeVeque (1998), and references therein for a more comprehensive overview.

Implementation of HRSC algorithms is based on a conservative formulation of the fluid equations and proper upwinding requires an exact or approximate solution (Roe 1986) to the Riemann problem, i.e., the decay of a discontinuity separating two constant states. This approach dates back to the pioneering work of Godunov (1959), and it has now become the leading line in developing high-resolution codes examples of which include FLASH (Fryxell et al. 2000 for reactive hydrodynamics), the special relativistic hydro code GENESIS (Aloy et al. 1999), the versatile advection code (VAC; Tóth 1996), or the new NIRVANA (Ziegler 2004).

Most HRSC algorithms are based on the so-called reconstruct-solve-average (RSA) strategy. In this approach volume averages are first reconstructed using piecewise monotonic interpolants inside each computational cell. A Riemann problem is then solved at each interface with discontinuous left and right states, and the solution is finally evolved in time. It turns out that this sequence of steps is quite general for many systems of conservation laws, and therefore, it provides a general framework under which we have developed a multiphysics, multialgorithm, high-resolution code, PLUTO. The code is particularly suitable for time-dependent, explicit computations of highly supersonic flows in the presence of strong discontinuities, and it can be employed under different regimes, i.e., classical, relativistic unmagnetized, and magnetized flows. The code is structured in a modular way, allowing a new module to be easily incorporated. This flexibility turns out to be quite important, since many aspects of computational fluid dynamics are still in rapid development. Besides, the advantage offered by a multiphysics, multisolver code is also to supply the user with the most appropriate algorithms and, at the same time, provide interscheme comparison for a better verification of the simulation results. PLUTO is entirely written in the C programming language and can run on either single processor or parallel machines, the latter functionality being implemented through the message passing interface (MPI) library. The code has already been successfully employed in the context of stellar and extragalactic jets (Bodo et al. 2003; Mignone et al. 2004, 2005a), radiative shocks (Mignone 2005; Massaglia et al. 2005), accretion disks (Bodo et al. 2005; Tevzadze et al. 2006), magneto-rotational instability, relativistic Kelvin-Helmholtz instability, and so forth.

The paper is structured as follows: in § 2 we give a description of the code design; in § 3 we introduce the physics modules available in the code; in § 4 we give a short overview on source terms and nonhyperbolicity; and in § 5 the code is validated against several standard benchmarks.

2. CODE DESIGN

PLUTO is designed to integrate a general system of conservation laws that we write as

$$\frac{\partial \boldsymbol{U}}{\partial t} = -\nabla \cdot \mathbf{T}(\boldsymbol{U}) + \boldsymbol{S}(\boldsymbol{U}). \tag{1}$$

Here $\boldsymbol{U}$ denotes a state vector of conservative quantities, $\mathbf{T}(\boldsymbol{U})$ is a rank 2 tensor, the rows of which are the fluxes of each component

[1] Dipartimento di Fisica Generale "Amedeo Avogadro," Università degli Studi di Torino, via Pietro Giuria 1, 10125 Torino, Italy.
[2] INAF/Osservatorio Astronomico di Torino, Strada Osservatorio 20, 10025 Pino Torinese, Italy.
[3] Laboratoire d'Astrophysique de l'Observatoire de Grenoble, 414 Rue de la Piscine, 38041 Grenoble Cedex 09, France.

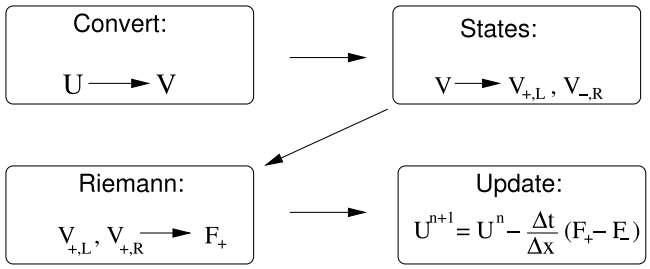

FIG. 1.—Simplified flow diagram of the reconstruct-solve-average (RSA) strategy: first, volume averages $\boldsymbol{U}$ are more conveniently mapped into primitive quantities $\boldsymbol{V}$. Left and right states $\boldsymbol{V}_{+,L}$ and $\boldsymbol{V}_{-,R}$ are constructed inside each zone by suitable variable interpolation and/or extrapolation. A Riemann problem is then solved between $\boldsymbol{V}_{+,L}$ and $\boldsymbol{V}_{+,R}$ to compute the numerical flux function $\boldsymbol{F}_+$ at cell interfaces and the solution is finally advanced in time.

of $\boldsymbol{U}$, and $\boldsymbol{S}(\boldsymbol{U})$ defines the source terms. Additional source terms may implicitly arise when taking the divergence of $\mathbf{T}(\boldsymbol{U})$ in a curvilinear system of coordinates. An arbitrary number of advection equations may be added to the original conservation law (eq. [1]).

Although the components of $\boldsymbol{U}$ are the primary variables being updated, fluxes are more conveniently computed using a different set of physical quantities, which we take as the primitive vector $\boldsymbol{V}$. This choice is supported, moreover, by the fact that interpolation on primitive variables enforces physical constraints such as pressure positivity and subluminal speeds in the case of relativistic flows.

Numerical integration of the conservation law (1) is achieved through shock-capturing schemes using the finite volume (FV) formalism where *volume averages* evolve in time. Generally speaking, these methods are comprised of three steps: an interpolation routine followed by the solution of Riemann problems at zone edges and a final evolution stage. In PLUTO, this sequence of steps provides the necessary infrastructure of the code; see the schematic diagram in Figure 1.

At the higher level, the original system of equations is integrated by following this general sequence of steps, regardless any knowledge of the physics involved. The explicit form of $\boldsymbol{U}$, $\boldsymbol{V}$, $\mathbf{T}(\boldsymbol{U})$, and $\boldsymbol{S}(\boldsymbol{U})$, on the other hand, depends on the particular physical module selected. Thus, at the lower level, a physical module collects the set of algorithms required to compute the terms involved in the discretization of the right-hand side of equation (1). This set should provide one or more Riemann solver(s), mapper routines for the conversion between primitive and conservative variables, a flux routine giving the components of $\mathbf{T}(\boldsymbol{U})$ in each direction, a source term function (if any), and a routine to compute the maximum and minimum characteristic speeds of the Jacobian matrix. Of course, additional features may be easily added by exploiting the independent modularity.

From the user's perspective, a particular configuration can be defined through a friendly interface entirely written in the Python scripting language. The interface allows the user to specify all problem-dependent attributes and algorithms, such as number of dimensions, geometry, physics module, reconstruction method, time stepping integration, and so forth.

2.1. *Notations*

PLUTO employs logically rectangular grids in a generic system of orthogonal curvilinear coordinates $(x^1,\ x^2,\ x^3)$ (see Table 1). Let N_1, N_2, and N_3 be the number of points in the three directions. The lower and upper coordinate bounds of zone $(i,\ j,\ k)$ are $(x^1_{i-1/2},\ x^2_{j-1/2},\ x^3_{k-1/2})$ and $(x^1_{i+1/2},\ x^2_{j+1/2},\ x^3_{k+1/2})$, respectively, where $1 \le i \le N_1$, $1 \le j \le N_2$, and $1 \le k \le N_3$. The zone widths

TABLE 1

SYSTEMS OF COORDINATES (I.E., COORDINATES, VOLUMES, AND AREAS) ADOPTED IN PLUTO

PLUTO	Cartesian	Polar	Spherical
x^1	x	r	r
x^2	y	ϕ	θ
x^3	z	z	ϕ
$\Delta\mathcal{V}^1$	Δx	$\Delta^2 r$	$\Delta^3 r$
$\Delta\mathcal{V}^2$	Δy	$r\Delta\phi$	$\Delta^3 r\Delta\mu$
$\Delta\mathcal{V}^3$	Δz	Δz	$\Delta^3 r\Delta\mu\Delta\phi$
A^1_+	1	r_+	r_+^2
A^2_+	1	1	$\Delta^2 r s_+$
A^3_+	1	1	$\Delta^2 r\Delta\theta$

NOTES.—Here r is the cylindrical or spherical radius, $0 \le \phi \le 2\pi$ is the azimuthal angle, and $0 \le \theta \le \pi$ is the polar angle. Here $\Delta^n r = (r_+^n - r_-^n)/n$, $\Delta\mu = \cos\theta_- - \cos\theta_+$, and $s_+ = \sin\theta_+$, where + or − refer to the right or left zone interface, respectively.

are then simply given by $\Delta x^1_i = x^1_{i+1/2} - x^1_{i-1/2}$, $\Delta x^2_j = x^2_{j+1/2} - x^2_{j-1/2}$, and $\Delta x^3_k = x^3_{k+1/2} - x^3_{k-1/2}$. The mesh spacing can be either uniform or (geometrical or logarithmic) stretched.

Parallelization is achieved through domain decomposition: the global domain is divided in subdomains with an equal number of points and each of the subdomains is assigned to a processor. By default the subdomains are created as maximally cubic; however, the user, at run time can specify a different strategy for the creation of the subdomains, imposing that a given direction cannot be subdivided. In the code, parallelization is handled through the MPI library.

Grid adaptivity techniques are also being provided. A one-dimensional adaptive mesh refinement (AMR) module based on the Berger & Colella (1989) method is distributed with the code.[4] However, since the main goal of this paper is to focus on the code modularity and its performance, AMR implementation will be described in future works.

In order to avoid formulas with cluttered notations we omit integer-valued subscripts when referring to three-dimensional quantities. Thus, $\boldsymbol{V}_{i,j,k}$ becomes simply $\boldsymbol{V}$. We introduce the standard two-point difference one-dimensional operator

$$\boldsymbol{\mathcal{L}}^d(\boldsymbol{V}) = -\frac{1}{\Delta\mathcal{V}^d}\left(A^d_+\boldsymbol{F}^d_+ - A^d_-\boldsymbol{F}^d_-\right) + \boldsymbol{S}^d, \qquad (2)$$

where $d = 1, 2, 3$ is a given direction and $A^d_\pm$ and $\Delta\mathcal{V}^d$ are, respectively, the cell's right (+) and left (−) interface areas and cell volume in that direction (see Table 1). Here $\pm \equiv (i \pm \frac{1}{2},\ j, k)$, $(i,\ j \pm \frac{1}{2},\ k)$, $(i,\ j,\ k \pm \frac{1}{2})$ when $d = 1$, 2, and 3, respectively. Scalar components of the vectors appearing in equation (2) will be denoted with a pair of square brackets; e.g., $\mathcal{L}^d_{[m_\phi]}$ is the component of $\boldsymbol{\mathcal{L}}^d$ contributing to the m_ϕ equation. Furthermore, since several of the building block algorithms are one-dimensional, we also omit the superscript d when unnecessary.

The numerical flux functions $\boldsymbol{F}_\pm$ in equation (2) follow the solution of one-dimensional Riemann problems at cell interfaces. Since more than one Riemann solver may be available in each physics module, we set, without loss of generality,

$$\boldsymbol{F}_+ = \mathcal{R}\left(\boldsymbol{V}_{+,L},\ \boldsymbol{V}_{+,R}\right), \qquad (3)$$

[4] Extension to multiple spatial dimensions is currently being developed using the CHOMBO library available at http://seesar.lbl.gov/ANAG/chombo/.

where $\mathcal{R}$ is the Riemann solver being used and $\boldsymbol{V}_{+,L}$, $\boldsymbol{V}_{+,R}$ are suitable left and right states at the zone edges perpendicular to the x-direction.

2.2. *Reconstruction*

Interpolation routines are designed to reconstruct a piecewise polynomial approximations $\boldsymbol{\mathcal{P}}(\boldsymbol{x})$ to $\boldsymbol{V}$ inside each cell starting from its cell averages:

$$\boldsymbol{V}_{\pm,S} = \mathcal{I}(\boldsymbol{\mathcal{P}},\ \boldsymbol{V}), \tag{4}$$

where $S = L$ ($S = R$) at $\boldsymbol{x} = \boldsymbol{x}_+$ ($\boldsymbol{x} = \boldsymbol{x}_-$). Here $\mathcal{I}$ is an interpolation routine designated to provide left and right edge interpolated values inside each cell, that is, $\boldsymbol{V}_{+,L} = \lim_{\boldsymbol{x}\to\boldsymbol{x}_+}\boldsymbol{\mathcal{P}}(\boldsymbol{x})$ and $\boldsymbol{V}_{-,R} = \lim_{\boldsymbol{x}\to\boldsymbol{x}_-}\boldsymbol{\mathcal{P}}(\boldsymbol{x})$, where the "L" and "R" subscripts refer to the sides of the interface.

Reconstruction methods have to satisfy monotonicity constraints in order to avoid spurious oscillations in proximity of discontinuities and steep gradients, see the books from Toro (1997), LeVeque (1998), and references therein. For second-order linear interpolants, for instance, one has

$$\boldsymbol{V}_{\pm,S} = \boldsymbol{V} \pm \frac{\Delta\tilde{\boldsymbol{V}}}{2}, \tag{5}$$

with the slopes $\Delta\tilde{\boldsymbol{V}}$ computed following a limiting procedure applied to primitive or characteristic variables. In the latter case one has

$$\Delta\tilde{\boldsymbol{V}} = \sum_k \Delta\tilde{w}_k \boldsymbol{r}_k, \quad \Delta\tilde{w}_k = \lim\left(\Delta w_{k,+},\ \Delta w_{k,-}\right), \tag{6}$$

where $\boldsymbol{r}_k$ and $\boldsymbol{l}_k$ are, respectively, the right and left eigenvectors of the primitive form of the equations, $\Delta w_{k,\pm} = \pm\boldsymbol{l}_k \cdot (\boldsymbol{V}_{i\pm1} - \boldsymbol{V}_i)$ are forward (+) and backward (−) derivatives and k labels the kth characteristic field. Different slope limiters (lim) are characterized by distinct steepening properties and can be independently assigned to each primitive variable or characteristic field.

At the time of this writing, PLUTO allows one to choose between either flat (first-order in space), linear (second-order), third-order convex ENO (Del Zanna & Bucciantini 2002), parabolic reconstructions (as in Mignone et al. 2005b), or the fifth-order finite difference WENO scheme of Jiang & Shu (1996) (only for the hydrodynamics equations).

2.3. *Riemann Solver*

As already anticipated, computation of the numerical flux function $\boldsymbol{F}_+$ at a zone edge (x_+) requires the solution $\boldsymbol{U}(x,\ t)$, for $t > t_0$, to the initial value problem

$$\boldsymbol{U}(x,\ t_0) = \begin{cases} \boldsymbol{U}_{+,L} & \text{if } x < x_+, \\ \boldsymbol{U}_{+,R} & \text{if } x > x_+, \end{cases} \tag{7}$$

complemented with the one-dimensional evolutionary equation for $\boldsymbol{U}$. Since this section deals with interface quantities, we omit, in this section only, the + subscripts.

The exact solution to the Riemann problem (eq. [7]) involves the decay of a set of nonlinear waves and can be a rather cumbersome task to achieve. With the exception of few simple cases, existing Riemann solvers routinely involved in upwind schemes are based on different levels of approximation.

The Lax-Friedrichs Rusanov flux is robust, but also the most diffusive solver and is available for all modules. It computes the fluxes according to

$$\boldsymbol{F} = \frac{1}{2}\left[\boldsymbol{f}_L + \boldsymbol{f}_R - |\lambda_{\max}|(\boldsymbol{U}_R - \boldsymbol{U}_L)\right], \tag{8}$$

where $\boldsymbol{f} = \hat{\boldsymbol{e}}^d \cdot \mathbf{T}(\boldsymbol{U})$ is the projection of the tensor flux on the $\hat{\boldsymbol{e}}^d = (\delta_{1d},\ \delta_{2d},\ \delta_{3d})$ unit vector, δ_{ij} is Kronecker-Delta symbol, and $|\lambda_{\max}|$ is the largest local signal velocity.

A somewhat different approximation comes from the HLL-family solvers:

$$\boldsymbol{F} = \begin{cases} \boldsymbol{f}_L & \text{if } \lambda_1 > 0, \\ \boldsymbol{f}_k & \text{if } \lambda_k\lambda_{k+1} < 0\ (k = 1,\ \ldots,\ N-1), \\ \boldsymbol{f}_R & \text{if } \lambda_N < 0, \end{cases} \tag{9}$$

where the solution to the Riemann problem is approximated by N waves with $\lambda_{k+1} > \lambda_k$ and $k = 1,\ \ldots,\ N-1$ and separated by $N+1$ states. The $\boldsymbol{f}_k$ are computed from a suitable "parameterization" of the Rankine-Hugoniot jump conditions across each wave:

$$\lambda_k(\boldsymbol{U}_{k+1} - \boldsymbol{U}_k) = \boldsymbol{f}_{k+1} - \boldsymbol{f}_k, \tag{10}$$

with $\boldsymbol{U}_{N+1} = \boldsymbol{U}_R$, $\boldsymbol{U}_1 = \boldsymbol{U}_L$ and similarly for $\boldsymbol{f}_{N+1}$ and $\boldsymbol{f}_1$. The HLL and HLLC solvers can be consistently derived following this recipe with $N = 2$ and 3, respectively; see Toro et al. (1994) and Batten et al. (1997) for the hydro equations, Li (2005) for the MHD equations, and Mignone & Bodo (2005, 2006) for the relativistic equations. Similarly, we have also implemented the multistate ($N = 5$) HLLD solver of Miyoshi & Kusano (2005) in the MHD module. The HLL approach has some attractive features in that it guarantees positivity of pressure and, in the case of relativistic flows, it preserves the condition $|\boldsymbol{v}| < 1$.

Linearized Riemann solvers are more accurate since the averaging process inherent to equation (8) or equation (9) is removed and all jumps are computed by linearization around some average state. The Roe solver computes the numerical fluxes according to

$$\boldsymbol{F} = \frac{1}{2}\left[\boldsymbol{f}_L + \boldsymbol{f}_R - \sum_k |\lambda_k| \mathbf{L}_k \cdot (\boldsymbol{U}_L - \boldsymbol{U}_R)\mathbf{R}_k\right], \tag{11}$$

where the rows (columns) of $\mathbf{L}$ ($\mathbf{R}$) are the left (right) eigenvectors of the Jacobian $\partial\boldsymbol{f}(\boldsymbol{U})/\partial\boldsymbol{U}$.

The advection upstream splitting method (AUSM, originally proposed Liou & Steffen 1993) provides an alternative and efficient upwinding strategy by splitting the flux into convective and pressure terms, respectively associated with linear and nonlinear fields:

$$\boldsymbol{F} = \boldsymbol{\Phi} v_n + \boldsymbol{p}, \tag{12}$$

where v_n is a suitably defined convective velocity and the $\boldsymbol{p}$ flux is governed by the acoustic speed. The original AUSM scheme has been substantially improved in the work of Liou (1996, 2006) and Wada & Liou (1997).

The most accurate approach consists of directly solving the whole set of Rankine-Hugoniot jump conditions to find $\boldsymbol{V}^*$, from which the flux can be computed,

$$\boldsymbol{F} = \boldsymbol{f}(\boldsymbol{V}^*). \tag{13}$$

TABLE 2
NUMBER OF RIEMANN PROBLEMS PER CELL PER TIME STEP REQUIRED BY DIFFERENT TIME MARCHING SCHEMES IN $N_d = 1, 2, 3$ DIMENSIONS

Time Marching	1D	2D	3D	$C_a^{\max}$
MH (S)	1	2	3	1
MH (U)	-	4	12	1
ChTr (S)	1	2	3	1
ChTr (U)	-	4	12	1
RK2 (S)	2	4	6	1[a]
RK2 (U)	3	6	9	$1/\sqrt{N_d}$
RK3 (S)	2	4	6	1
RK3 (U)	3	6	9	$1/\sqrt{N_d}$

NOTES.—(S) or (U) stand for dimensionally split or unsplit algorithm, MH and ChTr are the MUSCL-Hancock and Characteristic tracing schemes, and RK2 and RK3 are the second- and third-order Runge-Kutta schemes. The rightmost column gives the maximum allowed Courant number.

[a] High-order interpolants (PPM, CENO3) may require a lower limit.

However, this approach is computationally expensive since it generally involves the solution of highly nonlinear equations.

Different physics modules come with different sets of Riemann solvers, and additional methods of solution may be easily accommodated. We warn that more accurate Riemann solvers (such linearized or exact schemes) may introduce insufficient numerical dissipation for certain flow configurations. Sporadically, this could lead to a number of numerical pathologies such as the carbuncle phenomena, odd-even coupling or lack of positivity-preserving properties. See the work by Quirk (1994) for a comprehensive review.

2.4. *Temporal Evolution*

Time marching algorithms provide a general (i.e., physics-independent) framework for the discretization of the left-hand side of equation (1). For example, in the simplest case of forward Euler discretization one has

$$\frac{\boldsymbol{U}^{n+1} - \boldsymbol{U}^n}{\Delta t} = \boldsymbol{\mathcal{L}}^n, \tag{14}$$

where $\boldsymbol{\mathcal{L}}^n$ is the right-hand side operator from equation (2) or a sum of them for dimensionally split or unsplit schemes, respectively. The time step Δt is limited by the Courant-Friedrichs-Lewy (CFL; Courant et al. 1928) condition:

$$\Delta t = C_a \min_d \left(\frac{\Delta l^d_{\min}}{\left|\lambda^d_{\max}\right|} \right), \tag{15}$$

with $\Delta l^d_{\min}$ and $\lambda^d_{\max}$ being, respectively, the smallest cell length and largest signal velocity in the d direction. C_a identifies the Courant number, restricted by the conditions given in the last column of Table 2.

PLUTO provides a number of time-marching schemes for the explicit numerical integration of the conservation law (1). We discriminate between (1) fully discrete, zone-edge extrapolated and (2) semidiscrete methods. Evolution in more than one dimension may be achieved by either operator splitting (Strang 1968) or fully multidimensional integration.

2.4.1. *Zone-Edge Extrapolated Methods*

Zone-edge extrapolated methods achieve second-order temporal accuracy by midpoint in time quadrature,

$$\boldsymbol{U}^{n+1} = \boldsymbol{U}^n + \Delta t \boldsymbol{\mathcal{L}}^{n+1/2}, \tag{16}$$

and thus are based on a single step. For directional splitting methods one has $\boldsymbol{\mathcal{L}}^{n+1/2} \equiv \boldsymbol{\mathcal{L}}^d\left(\boldsymbol{V}^{n+1/2}\right)$, whereas $\boldsymbol{\mathcal{L}}^{n+1/2} = \sum_d \boldsymbol{\mathcal{L}}^d\left(\boldsymbol{V}^{n+1/2}\right)$ in the case of a dimensionally unsplit method. The input states for the Riemann solver are computed by Taylor expansion of the primitive variables at half-time step,

$$\boldsymbol{V}^{n+1/2}_{\pm,S} = \boldsymbol{V}^n \pm \frac{1}{2}\left(\mathbf{I} - \frac{\Delta t}{\Delta x}\mathbf{A}^n\right)\Delta\tilde{\boldsymbol{V}}^n, \tag{17}$$

where $\mathbf{A}^n$ is the matrix of the quasi-linear form of the equations and $\mathbf{I}$ is the identity matrix. An alternative form that does not require the primitive form of the equations can be written in terms of conservative variables. Both variants yield the well-known MUSCL-Hancock scheme (MH; van Leer 1974; Toro 1997). When eigenvalues and eigenvectors are available, upwind limiting may be used to select only those characteristics that contribute to the effective left and right states. This approach is employed, for instance, in the original PPM advection scheme of Colella & Woodward (1984), and we refer to this as characteristic tracing (ChTr). These methods require boundary conditions to be assigned at the beginning of the time step only.

The advantage offered by dimensionally split single-step algorithms is the lower computational cost (only one Riemann solver per cell per direction is required; see Table 2) and storage requirement. An alternative, dimensionally unsplit version of these schemes, however, adopts the corner transport upwind (CTU) method of Colella (1990) and Saltzman (1994) and is more expensive. In this case, an extra correction term is needed in equation (17); in two dimensions, for example, the input states for the Riemann problem are modified to

$$\tilde{\boldsymbol{U}}^{n+1/2}_{\pm,S} = \boldsymbol{U}^{n+1/2}_{\pm,S} + \frac{\Delta t}{2}\boldsymbol{\mathcal{L}}^{t,n+1/2}, \tag{18}$$

with $\boldsymbol{\mathcal{L}}^{t,n+1/2}$ being the right-hand side operator corresponding to the transverse direction. The CTU method has been recently extended to magnetohydrodynamics in the work of Crockett et al. (2005), Gardiner & Stone (2005), and Fromang et al. (2006), and to relativistic hydrodynamics by Mignone et al. (2005b). As in the dimensionally split case, the CFL stability condition is not affected by the dimensionality of the problem (i.e., $C_a < 1$; see also Table 2). Contrary to its dimensionally split version, on the other hand, this method is computationally more expensive since 4 instead of 2 (in two dimensions) and 12 instead of 3 (in three dimensions) Riemann problems must be solved at each interface; see Table 2.

2.4.2. *Semidiscrete Methods*

Semidiscrete methods are based on the classical method of lines, where the spatial discretization is considered separately from the temporal evolution that is left continuous in time. In this framework equation (1) is discretized as a regular ordinary differential equation. PLUTO implements the second- and third-order total variation diminishing (TVD) Runge-Kutta schemes of Gottlieb & Shu (1996). The second-order method (RK2) advances the system of conservation law (1) as

$$\boldsymbol{U}^* = \boldsymbol{U}^n + \Delta t \boldsymbol{\mathcal{L}}^n, \tag{19}$$

$$\boldsymbol{U}^{n+1} = \frac{1}{2}(\boldsymbol{U}^n + \boldsymbol{U}^* + \Delta t \boldsymbol{\mathcal{L}}^*), \tag{20}$$

with $\mathcal{L}^n \equiv \mathcal{L}^d(\boldsymbol{V}^n)$ or $\mathcal{L}^n = \sum_d \mathcal{L}^d(\boldsymbol{V}^n)$ in the case of a dimensionally split or unsplit method, respectively. The third-order Runge-Kutta method (RK3) may also be used, at the cost of an additional step:

$$\boldsymbol{U}^* = \boldsymbol{U}^n + \Delta t \mathcal{L}^n, \tag{21}$$

$$\boldsymbol{U}^{**} = \frac{1}{4}(3\boldsymbol{U}^n + \boldsymbol{U}^* + \Delta t \mathcal{L}^*), \tag{22}$$

$$\boldsymbol{U}^{n+1} = \frac{1}{3}(\boldsymbol{U}^n + 2\boldsymbol{U}^{**} + 2\Delta t \mathcal{L}^{**}). \tag{23}$$

For this class of methods, input states for the Riemann solver are given by the output of the interpolation routine, see § 2.2. Besides, boundary conditions must be assigned before each step. A total of two and three Riemann problems per cell per direction must be solved by the RK2 and RK3 marching scheme, respectively. Furthermore, fully unsplit Runge-Kutta integrators require a stronger time step limitation; see Table 2.

3. PHYSICS MODULES

PLUTO is distributed with four independent physics modules for the explicit numerical integration of the fluid equations under different regimes and conditions. The hydrodynamics (HD), magnetohydrodynamics (MHD), relativistic (RHD), and relativistic MHD (RMHD) modules solve, respectively, the Euler equations of gas dynamics, the ideal/resistive MHD equations, the energy-momentum conservation laws of a special relativistic perfect gas, and the equations for a (special) relativistic magnetized ideal plasma.

In what follows, ρ, p, and E denote, respectively, the proper density, thermal pressure, and total energy density. Vector fields such as $\boldsymbol{m} \equiv (m_1,\ m_2,\ m_3)^T$, $\boldsymbol{v} \equiv (v_1,\ v_2,\ v_3)^T$, and $\boldsymbol{B} \equiv (B_1, B_2,\ B_3)^T$, define the momentum density, velocity, and magnetic field. Finally, Γ will be used to define the ratio of specific heats for the ideal equation of state.

3.1. *The Hydrodynamics (HD) Module*

This module implements the equations of classical fluid dynamics with an ideal equation of state. The conservative variables $\boldsymbol{U}$ and the flux tensor are

$$\boldsymbol{U} = \begin{pmatrix} \rho \\ \boldsymbol{m} \\ E \end{pmatrix}, \quad \mathbf{T}(\boldsymbol{U}) = \begin{pmatrix} \rho\boldsymbol{v} \\ \boldsymbol{m}\boldsymbol{v} + p\mathbf{I} \\ (E+p)\boldsymbol{v} \end{pmatrix}^T, \tag{24}$$

where $\boldsymbol{m} = \rho\boldsymbol{v}$ is the momentum density and $\mathbf{I}$ is the unit, 3×3 tensor. The total energy density E is related to the gas pressure p by the ideal gas closure:

$$E = \frac{p}{\Gamma - 1} + \frac{|\boldsymbol{m}|^2}{2\rho}. \tag{25}$$

The set of primitive variables $\boldsymbol{V} \equiv (\rho,\ \boldsymbol{v},\ p)^T$ is given by density, velocity $\boldsymbol{v}$, and thermal pressure p.

This module comes with a set of several Riemann solvers, including the nonlinear Riemann solver based on the two-shock approximations (Colella & Woodward 1984; Fryxell et al. 2000), the Roe solver (Toro 1997), the AUSM+ scheme (Liou 1996), the HLL (Einfeldt et al. 1991), HLLC (Toro et al. 1994) solvers, and the Lax-Friedrichs solver (Rusanov 1961).

The HD module contains an implementation of the fast Eulerian transport algorithm for differentially rotating disk (FARGO; Masset 2000) on polar grids. The FARGO scheme allows much larger time steps than the standard integration where the Courant condition is traditionally limited by the fast orbital motion at the inner boundary.

3.2. *The Magnetohydrodynamics (MHD) Module*

The MHD module deals with the equations of classical ideal or resistive magnetohydrodynamics (MHD). In the ideal case, $\boldsymbol{U}$ and $\mathbf{T}$ may be written as

$$\boldsymbol{U} = \begin{pmatrix} \rho \\ \boldsymbol{m} \\ \boldsymbol{B} \\ E \end{pmatrix}, \quad \mathbf{T}(\boldsymbol{U}) = \begin{bmatrix} \rho\boldsymbol{v} \\ \boldsymbol{m}\boldsymbol{v} - \boldsymbol{B}\boldsymbol{B} + p_t\mathbf{I} \\ \boldsymbol{v}\boldsymbol{B} - \boldsymbol{B}\boldsymbol{v} \\ (E+p_t)\boldsymbol{v} - (\boldsymbol{v}\cdot\boldsymbol{B})\boldsymbol{B} \end{bmatrix}^T, \tag{26}$$

with $\boldsymbol{m} = \rho\boldsymbol{v}$ and $p_t = p + |\boldsymbol{B}|^2/2$ being the total (thermal + magnetic) pressure, respectively. The additional constraint $\nabla \cdot \boldsymbol{B} = 0$ complements the magnetic field evolution (see § 3.2.1). Resistivity is introduced by adding appropriate parabolic terms to the induction and energy equations; see § 4.3.

Available equations of state implemented are the ideal gas law,

$$E = \frac{p}{\Gamma - 1} + \frac{1}{2}\left(\frac{|\boldsymbol{m}|^2}{\rho} + |\boldsymbol{B}|^2\right), \tag{27}$$

and the isothermal equation of state $p = c_s^2\rho$, where c_s is the (constant) isothermal speed of sound.

The set of primitive variables is the same one used for the HD module, with the addition of magnetic fields. The user can choose among the following available Riemann solvers: the Roe solver of Cargo & Gallice (1997), the HLL (Janhunen 2000), HLLC (Li 2005), HLLD (Miyoshi & Kusano 2005), and the Lax-Friedrichs solvers.

3.2.1. *Solenoidal Constraint*

The solution to the MHD equations must fulfill the solenoidal constraint, $\nabla \cdot \boldsymbol{B} = 0$, at all times. Unfortunately, it is well known that numerical scheme do not naturally preserve this condition unless special discretization techniques are used. Among the variety of monopole control strategies proposed in literature (for a review see Tóth 2000), we have implemented (1) the eight wave formulation (Powell 1994; Powell et al. 1999) and (2) the constrained transport (CT) of Balsara & Spicer (1999) and Londrillo & del Zanna (2004). The CT framework has been incorporated into the unsplit CTU integrator following the recent work by Gardiner & Stone (2005). A similar approach is used by Teyssier et al. (2006).

In the first strategy, the magnetic field has a cell-centered representation and an additional source term is added to the MHD equation. The discretization of the source term is different depending on the Riemann solver (following Janhunen 2000), a feature that we found to greatly improve robustness.

In the CT formulation, on the other hand, the induction equation is integrated directly using the Stokes theorem and the magnetic field has a staggered collocation.

3.3. *The Relativistic Hydrodynamics (RHD) Module*

This module deals with the motion of an ideal fluid in special relativity. The equations are given by particle number conservation and energy-momentum conservation (Landau & Lifshitz 1959). Conservative variables and tensor flux are

$$\boldsymbol{U} = \begin{pmatrix} D \\ \boldsymbol{m} \\ E \end{pmatrix}, \quad \mathbf{T}(\boldsymbol{U}) = \begin{pmatrix} D\boldsymbol{v} \\ \boldsymbol{m}\boldsymbol{v} + p\mathbf{I} \\ \boldsymbol{m} \end{pmatrix}^T, \tag{28}$$

where $D = \gamma\rho$ is the laboratory density and $\gamma = (1 - |\boldsymbol{v}|^2)^{-1/2}$ is the Lorentz factor. For convenience, velocities are normalized to the speed of light.

The relation between conserved and primitive variables $\boldsymbol{V} \equiv (\rho,\ \boldsymbol{v},\ p)^T$ is more involved than its classical counterpart. Specifically, we have

$$D = \gamma\rho, \quad \boldsymbol{m} = \rho h\gamma^2\boldsymbol{v}, \quad E = \rho h\gamma^2 - p, \tag{29}$$

where the specific enthalpy $h \equiv h(p,\ \rho)$ depends on the equation of state. The inverse map requires the solution of a nonlinear equation. Ryu et al. (2006) address this topic for different choices of $h(p,\ \rho)$.

Two equations of state are available for this module: the constant-Γ ideal gas law

$$h = 1 + \frac{\Gamma}{\Gamma - 1}\Theta, \tag{30}$$

with $\Theta = p/\rho$, and the equation of state (TM henceforth) already introduced in Mignone et al. (2005b):

$$h = \frac{5}{2}\Theta + \sqrt{\frac{9}{4}\Theta^2 + 1}, \tag{31}$$

which satisfies Taub's inequality (Taub 1948) and approximates (within $\lesssim$4%) the single-specie relativistic perfect gas (for a thorough discussion, see Mignone et al. [2005b], Ryu et al. [2006], and references therein).

The two-shock nonlinear Riemann solver described in Mignone et al. (2005b) has been incorporated into the set of available Riemann solvers, together with the recently developed HLLC algorithm by Mignone & Bodo (2005). The HLL and Lax-Friedrichs Riemann solvers have also been coded.

3.4. *The Relativistic Magnetohydrodynamics (RMHD) Module*

The motion of an ideal relativistic magnetized fluid is described by conservation of mass and energy momentum, and by Maxwell's equations; see, for example, Anile & Pennisi (1987) and Anile (1989). PLUTO implements the equation of special relativistic magnetohydrodynamics where the vector of conservative variables and respective fluxes can be cast as

$$\boldsymbol{U} = \begin{pmatrix} D \\ \boldsymbol{m} \\ \boldsymbol{B} \\ E \end{pmatrix}, \quad \mathbf{T}(\boldsymbol{U}) = \begin{pmatrix} D\boldsymbol{v} \\ w_t\gamma^2\boldsymbol{v}\boldsymbol{v} - \boldsymbol{b}\boldsymbol{b} + \mathbf{I}p_t \\ \boldsymbol{v}\boldsymbol{B} - \boldsymbol{B}\boldsymbol{v} \\ \boldsymbol{m} \end{pmatrix}^T, \tag{32}$$

where $w_t = \rho h + b_m^2$, $b_m^2 = |\boldsymbol{B}|^2/\gamma^2 + (\boldsymbol{v}\cdot\boldsymbol{B})^2$, and $\boldsymbol{b} = \boldsymbol{B}/\gamma + \gamma(\boldsymbol{v}\cdot\boldsymbol{B})\boldsymbol{v}$, and $p_t = p + b_m^2/2$ is the total (thermal + magnetic) pressure.

The components of $\boldsymbol{U}$ are related to the primitive variables $\boldsymbol{V} \equiv (\rho,\ \boldsymbol{v},\ p,\ \boldsymbol{B})$ through

$$D = \rho\gamma, \tag{33}$$

$$\boldsymbol{m} = \left(\rho h\gamma^2 + |\boldsymbol{B}|^2\right)\boldsymbol{v} - (\boldsymbol{v}\cdot\boldsymbol{B})\boldsymbol{B}, \tag{34}$$

$$E = \rho h\gamma^2 - p + \frac{|\boldsymbol{B}|^2}{2} + \frac{|\boldsymbol{v}|^2|\boldsymbol{B}|^2 - (\boldsymbol{v}\cdot\boldsymbol{B})^2}{2}. \tag{35}$$

The inverse map is highly nonlinear and different methods of solution have been proposed when a constant Γ law is adopted (for a recent review, see Noble et al. 2006). In PLUTO, we follow the approach described in Mignone & Bodo (2006), which was recently generalized to include the TM equation of state described in § 3.3. The interested reader should also refer to the work by Mignone & Mc Kinney (2007).

The RMHD module may be used with the Lax-Friedrichs, HLL, or the recently developed HLLC Riemann solver; see Mignone & Bodo (2006). Since the induction equation takes the same form as in the classical case, the divergence of the magnetic field can be kept under control by any of the methods already introduced in the MHD module; see § 3.2.1.

4. SOURCE TERMS AND NONHYPERBOLICITY

In addition to the homogeneous terms previously described, a number of additional features may be added in the code.

Local source terms are functions of the variables themselves but not of their derivatives and are included either during the advection step or via operator splitting. Examples include the centripetal and Coriolis terms implicitly arising when working in polar or spherical coordinates, external forces (e.g., gravity) or optically thin radiative losses.

Nonideal effects such as viscosity, resistivity and thermal conduction, on the other hand, introduce parabolic corrections to the equations and involve the solution of diffusion equations.

4.1. *Geometrical Source Terms*

The divergence of a rank 2 tensor in curvilinear coordinates breaks down the homogeneous properties of equation (1). This loss of conservation comes from the additional source terms inevitably introduced by those unit vectors that do not have fixed orientation in space (see, for instance, the Appendix in Mignone et al. 2005b). In the case of symmetric or antisymmetric tensors, however, some of the source terms can be eliminated by properly rearranging the derivatives. This ensures conservation of the angular momenta rather than the linear ones.

For all physics modules previously introduced, for example, the components of the flux tensor in the momentum equation form a 3×3 rank 2 symmetric tensor $M_{ij} + p\delta_{ij} = M_{ji} + p\delta_{ji}$, where p is the isotropic pressure term. However, since $\nabla\cdot(\mathbf{I}p) \equiv \nabla p$, the differencing terms involving pressure are separately treated as gradient components and never contribute to the source terms.

The symmetric character of $\mathbf{M}$ leads to further simplifications in polar coordinates, since the ϕ-component of the divergence of $\mathbf{M}$ may be written as

$$(\nabla\cdot\mathbf{M})_\phi = \frac{1}{r^2}\frac{\partial\left(r^2 M_{r\phi}\right)}{\partial r} + \frac{1}{r}\frac{\partial M_{\phi\phi}}{\partial\phi} + \frac{\partial M_{z\phi}}{\partial z}. \tag{36}$$

This leads to the so-called angular momentum-conserving form, where the radial contribution to m_ϕ is computed by the new operator

$$\mathcal{L}^r_{[m_\phi]} = -\frac{1}{r\Delta\mathcal{V}^r}\left(r^2_+ F^r_{+,[m_\phi]} - r^2_- F^r_{-,[m_\phi]}\right), \qquad (37)$$

thus leaving the radial component of momentum as the only nonhomogeneous equation, with $S_{[m_r]} = M_{\phi\phi}/r$. Similar considerations lead to the angular momentum conserving form in spherical geometry.

Geometrical source terms are integrated explicitly during the advection step by adding their contribution to the right-hand side operator in equation (2). In the semidiscrete approach, this is straightforward. For edge-extrapolated methods, this requires (1) augmenting equation (17) with the cell-center source term for a half-step, and then (2) averaging the resulting time-centered left and right edge values to compute the $\boldsymbol{S}$ for the final conservative step.

4.2. *Optically Thin Cooling*

For many astrophysical applications, radiative processes may become important during the evolution. This is particularly true when the cooling timescale becomes comparable to or smaller than the typical timescale for the dynamical evolution.

Optically thin radiative losses are treated as local source terms depending on the temperature, density and an arbitrary number of ions of different elements (e.g., H, He, C, N, O, and so on). The ionization fractions are advected with the fluid and are subject to collisional ionization, recombination, and charge exchange processes.

Currently, we have implemented a cooling module (O. Tesileanu et al. 2007, in preparation) similar to the one coded by Raga et al. (1997) extending the temperature range of applicability (2000 K $\lesssim T \lesssim 2\times10^5$ K) by using an increased number of ion species. This module employs 28 ions and is valid for densities up to 10^5 cm^{-3}.

Operator splitting is employed to evolve the chemical network in time. A second-order fully implicit scheme is adopted in regions of rapid variations, whereas a second-order explicit integrator is used otherwise. A comprehensive description is outside the goal of this work and will be available in future works.

4.3. *Treatment of Parabolic Terms*

Parabolic terms introduce second-order spatial derivatives and their treatment requires the solution of diffusion equations. Typical examples include electric resistivity,

$$\frac{\partial \boldsymbol{B}}{\partial t} + \nabla\times(-\boldsymbol{v}\times\boldsymbol{B} + \eta\boldsymbol{J}) = 0, \qquad (38)$$

$$\frac{\partial E}{\partial t} + \nabla\cdot[(E+p_t)\boldsymbol{v} - (\boldsymbol{v}\cdot\boldsymbol{B})\boldsymbol{B} + \eta\boldsymbol{J}\times\boldsymbol{B}] = 0, \qquad (39)$$

where $\boldsymbol{J} = \nabla\times\boldsymbol{B}$ is the current and η is the resistivity; or thermal conduction,

$$\frac{\partial E}{\partial t} + \nabla\cdot\left(\boldsymbol{F}^{\rm adv} - \kappa\nabla T\right) = 0, \qquad (40)$$

where $\boldsymbol{F}^{\rm adv}$ is the energy advection flux and κ is the thermal conductivity coefficient. These terms may be included in the original conservation law with the further time step limitation $\Delta t = \min(\Delta t^{\rm ad},\ \Delta t^{\rm par})$, where $\Delta t^{\rm ad}$ is the advective time step (eq. [15]) and

$$\Delta t^{\rm par} < 0.5 \min_{d=1,2,3}\left[\frac{\left(\Delta x^d\right)^2}{\max(\sigma)}\right] \qquad (41)$$

is the diffusion step. Here σ characterizes the physical process, e.g., η or κ.

For advection-dominated problems (i.e., $\Delta t^{\rm ad} \gtrsim \Delta t^{\rm par}$), equation (41) does not pose severe restrictions and diffusion terms can be treated explicitly. We achieve this by adding centered in space, second-order finite difference approximations to the right-hand side operator $\mathcal{L}^d$; see equation (2).

However, for diffusion-dominated problems and/or increasingly high-resolution simulations, $\Delta t^{\rm par}$ will eventually drop below the typical advection scale. In such situations the explicit integration can be considerably accelerated using the super time stepping (STS) method (Alexiades et al. 1996; O'Sullivan & Downes 2006). In STS, diffusion terms are included via operator splitting, and the solution vector is evolved over a super time step ΔT consisting of N smaller substeps, the stability of which is closely related to the properties of Chebyshev polynomials. It can be proved (Alexiades et al. 1996) that

$$\Delta T \to N^2 \Delta t^{\rm par}, \qquad (42)$$

which makes STS almost N times faster than the standard explicit scheme. Thus, if ΔT is taken to be the advective time step, STS will require (approximately) $\left(\Delta t^{\rm ad}/\Delta t^{\rm par}\right)^{1/2}$ iterations rather than $\Delta t^{\rm ad}/\Delta t^{\rm par}$, typical of a normal subcycling explicit time stepping. This approach offers dramatic ease of implementation over implicit schemes.

Resistivity has been extensively tested through the direct comparison with analytical solutions of linear diffusion problems of magnetic field. Our results confirm the efficiency and accuracy of the STS approach already highlighted by previous investigators. This has encouraged further experiments toward nonlinear problems (e.g., Spitzer-like conductivity), where preliminary results suggest that the STS methodology may be successfully applied. This issue will be presented in future works.

5. CODE VERIFICATION

PLUTO has been validated against several benchmarks typically adopted for other numerical schemes. Several algorithms for relativistic flows presented in Mignone et al. (2005b) and Mignone & Bodo (2005, 2006) are now part of the code and the related verification problems will not be repeated here. In what follows, we propose additional tests aimed to check the code performances on problems of different nature, geometry, and dimension. Computational details for each test are given in Table 3.

5.1. *Double Mach Reflection of a Strong Shock*

The initial condition for this test problem (Woodward & Colella 1984) consists of a planar shock front making an angle $\alpha = \pi/3$ with a reflecting wall, taken to be the x-axis:

$$\left(\rho,\ v_x,\ v_y,\ p\right) = \begin{cases} (1.4,\ 0,\ 0,\ 1) & \text{for } x > x_s(0), \\ (8,\ 8.25,\ -8.25,\ 116.5) & \text{otherwise,} \end{cases} \qquad (43)$$

where $x_s(t) = (10t/\sin\alpha + 1/6 + y/\tan\alpha)$ is the shock coordinate. The ideal equation of state with $\Gamma = 1.4$ is used throughout

TABLE 3
NUMERICAL SCHEMES ADOPTED IN THE DIFFERENT TEST PROBLEMS DESCRIBED IN THE TEXT

Test	Module	Case	Geometry	Dimensions	Interpolation	Solver	Time Stepping	C_a
Double Mach Reflection (§ 5.1).....	HD	(a)	Cartesian	2	Parabolic	HLLC	RK3 (S)	0.8
		(b)			CENO3	HLLC	RK3 (S)	0.8
		(c)			WENO5	Roe F-S	RK3 (S)	0.8
		(d)			Parabolic	Two-Shock	ChTr (S)	0.8
		(e)			Linear	Roe	MH (U)	0.8
		(f)			Linear	HLL	MH (U)	0.8
Underexpanded Jet (§ 5.2).....	HD	(a)	Cylindrical	2	Linear	AUSM+	MH (U)	0.4
		(b)			CENO3	HLL	RK3 (S)	0.4
Rotated Shock Tube (§ 5.3)	MHD	(a)	Cartesian	2.5	Linear	Roe	RK2 (U)	0.4
		(b)			Linear	HLLD	RK2 (U)	0.4
		(c)			Linear	HLL	RK2 (U)	0.4
Fast Rotor (§ 5.4).....	MHD	(a)	Cartesian	2	Parabolic	HLLC	ChTr (U)	0.6
		(b)	Polar		Linear	Roe	RK3 (U)	0.4
Magnetized Torus (§ 5.5)	MHD	(a)	Spherical	2.5	Linear	HLLD	RK2 (U)	0.4
		(b)	Spherical		Parabolic	HLL	RK3 (U)	0.4
		(c)	Cylindrical		Linear	HLLD	RK2 (U)	0.4
		(d)	Cylindrical		Parabolic	HLL	RK3 (U)	0.4
Spherical Shock Tube (§ 5.5).....	RHD	(a)	Spherical	1	Parabolic	Two-Shock	ChTr (S)	0.8
		(b)	Spherical	1	Linear	HLL	MH (S)	0.8
		(c)	Cartesian	3	Linear	HLLC	ChTr (S)	0.8
Magnetized Blast Wave (§ 5.7).....	RMHD	(a)	Cartesian	3	Linear	HLL	RK2 (U)	0.2
		(b)	Cylindrical	2	Linear	HLL	MH (U)	0.4

NOTES.—Here ChTr stands for the characteristic tracing, RK2 and RK3 are for the second and third Runge-Kutta time stepping, respectively, and MH is the MUSCL-Hancock time marching scheme. Split or unsplit schemes are denoted with (S) or (U). The rightmost column gives the CFL number C_a.

the calculation. The computational domain is the rectangle $[0, 4] \times [0, 1]$ covered with a uniform grid with $1/\Delta x = 1/\Delta y = 480$. Outflow boundary conditions apply at $x = 4$, and reflective conditions are imposed at the bottom boundary $y = 0$ for $x > 1/6$. Fluid variables are kept constant to their initial values at $y = 0$ for $x < 1/6$ and at the leftmost boundary $x = 0$. At the top boundary, $y = 1$, the exact motion of the shock is prescribed. We carry out integration until $t = 0.4$ using the six different combinations of algorithms described in Table 3. Panels in the left column of Figure 2 show, from top to bottom, the results obtained with high-order (>2) interpolants: piecewise parabolic reconstruction (case *a*), third-order central ENO (*b*), and the WENO scheme of Jiang & Shu (1996) (*c*). Panels on the right show the original PPM scheme of Colella & Woodward (1984) (case *d*; *top*) and the second-order unsplit Muscl-Hancock with a Roe solver (case *e*; *middle*) and with the HLL solver (case *f*; *bottom*).

After the reflection, a complicated flow structure develops with two curved reflected shocks propagating at directions almost orthogonal to each other and a tangential discontinuity separating them. At the wall, a pressure gradient sets up a denser fluid jet propagating along the wall. Kelvin-Helmholtz instability patterns may be identified with the "rolls" developing at the slip line. This feature is visible only for some of the selected schemes. Indeed, the use of high-order interpolants, such as PPM (cases *a*, *d*) or WENO (*c*), and the ability of the Riemann solvers to resolve contact and shear waves result in smaller numerical viscosity when compared to a second-order slope-limited reconstruction and/or a more approximate Riemann solver. In this respect, case *a* shows the smallest amount of dissipation, whereas the HLL solver (case *f*; *bottom right*) has the largest numerical diffusion.

In terms of CPU time, we found that cases *a*–*f* followed the ratios 1.82 : 1.55 : 8.52 : 0.94 : 1.24 : 1, that is, with the fifth-order WENO schemes being by far the most expensive (more than a factor of 8 compared to the simple HLL) and the original PPM (split) scheme being the fastest. In doing the comparison, however, one has to bear in mind that the amount of computing time depends, in the first place, by the number of Riemann problems solved in each zone at each time step, as shown in Table 2. This number is 6 for cases *a*, *b*, and *c*, but 4 for cases *e* and *f* (which are dimensionally unsplit), and only 2 for the original dimensionally split PPM scheme, case *d*. Thus, one can safely conclude, comparing cases *e* and *f*, that the Roe solver is by a factor $\sim 1/4$ slower than HLL or, by comparing cases *a*, *b*, and *c*, that the CENO and PPM are considerably faster than WENO, although the combinations of algorithms in case *a* better resolves small-scale structures.

Finally, we note that computations carried with more accurate schemes (with the exception of the fifth-order WENO) exhibit a slight tendency to form a spurious "kinked" Mach stem on the *x*-axis. This feature is engendered by a numerical flaw (Quirk 1994) caused by insufficient numerical dissipation and it becomes particularly dramatic when the resolution is further increased.

5.2. *Underexpanded Jet*

Code validation, in addition to the typical tests described above, can be carried out against laboratory experiments of underexpanded free jets as well. This kind of experiments consists of injecting, through a converging nozzle, a gas at stagnation pressure p_0 into a chamber kept at lower pressure p_c. The shock structure that forms is typically axially symmetric and consists of a quasi-stationary Mach-disk, located at a distance z_M from the nozzle, and of a barrel and reflected shocks (Young 1975). After series of experiments with different gases, an empirical expression was found (Adamson & Nicholls 1959; Bier & Schmidt 1961; Ashkenas & Sherman 1965) that relates the Mach-disk location z_M with the pressure ratio (Young 1975):

$$z_M = 1.34 r^* \sqrt{\frac{p_0}{p_c}}, \tag{44}$$

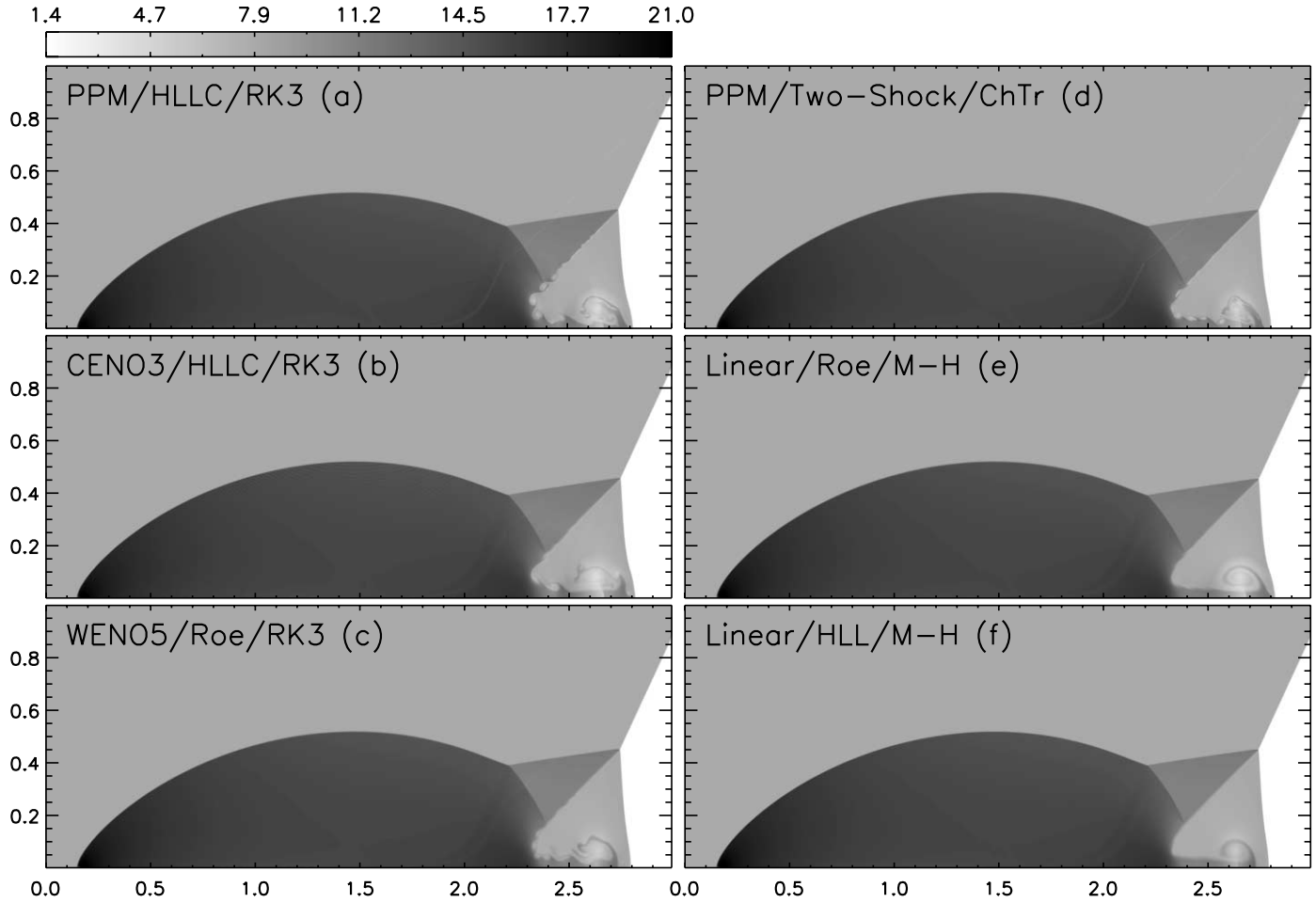

FIG. 2.—Density maps for the double Mach reflection test at $t = 0.2$. Each panel shows results obtained with the different combinations of schemes listed in Table 3. The mesh size ($1/\Delta x = 1/\Delta y = 480$) and Courant number ($C_a = 0.8$) are the same for all cases. For the sake of clarity only the region $[0,\ 3] \times [0,\ 1]$ is shown. [*See the electronic edition of the Supplement for a color version of this figure*.]

with r^* the effective sonic nozzle radius. Furthermore, Knuth (1964) obtained for the Mach number $M_{\rm axis}$ on the jet axis the following empirical expression:

$$M_{\rm axis} \approx (2.2)^{(\Gamma-1)/2}\left[\Gamma(\Gamma-1)\right]^{-(\Gamma-1)/4}\left(\frac{\Gamma+1}{\Gamma-1}\right)^{(\Gamma+1)/4}\left(\frac{z}{2r^*}\right)^{\Gamma-1}. \tag{45}$$

We have carried out two-dimensional numerical simulations in cylindrical coordinates on a uniform grid $r \in [0, 40]$, $z \in [0, 80]$, with 20 zones per beam radius. The grid has been further extended in both directions (up to $r = 80$ and $z = 160$) by adding a second patch of geometrically stretched zones (100 and 200 zones in r and z, respectively) in order to avoid spurious reflections at the boundaries. Free outflow is set at the outer boundaries and reflective conditions are imposed on the axis $r = 0$ and for $r > 1$ at $z = 0$. Considering the actual nozzle as the injection zone at $z = 0, r \le 1$, we have obtained the values of the pressure p^* and density ρ^* by employing the isentropic laws for a perfect gas (Shames 1983) for a converging nozzle with stagnation pressure p_0 and density ρ_0:

$$p^* = p_0\left(\frac{2}{\Gamma+1}\right)^{\Gamma/(\Gamma-1)}, \quad \rho^* = \rho_0\left(\frac{2}{\Gamma+1}\right)^{1/(\Gamma-1)}. \tag{46}$$

We choose $p_0/p_c = 2 \times 10^3$ and $\rho_0/\rho_c = 2 \times 10^4$ (typical of an argon jet in helium at ambient temperature) and accordingly choose a sonic injection velocity at the nozzle ($M^* = 1$). Densities, velocities, and lengths are normalized to the ambient ρ, sound speed, and beam radius, respectively.

Figure 3 shows the density maps for the two selected numerical schemes (see Table 3) at $t = 240$. The results obtained with the

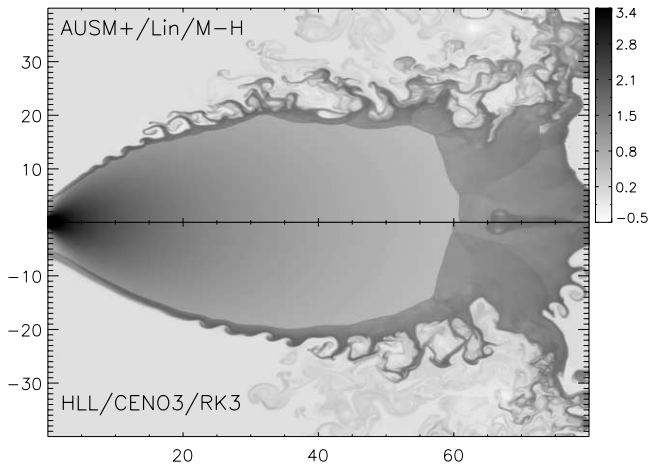

FIG. 3.—Density logarithms for the underexpanded jet at $t = 240$, for the linear Muscl-Hancock CTU with the AUSM+ solver (*top*) and the RK3 with HLL solver and third-order CENO interpolation (*bottom*). [*See the electronic edition of the Supplement for a color version of this figure*.]

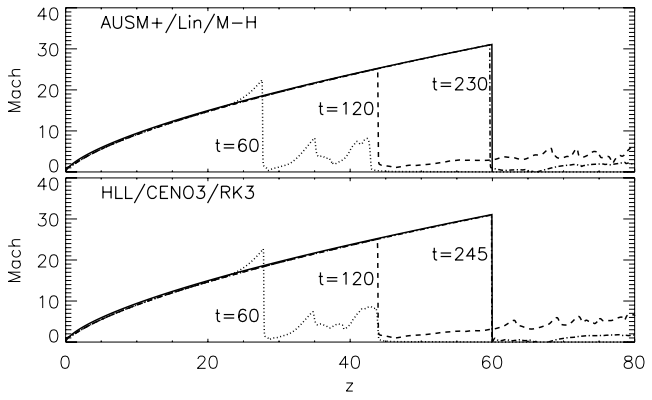

FIG. 4.—Axial Mach numbers plotted at $t = 60, 120, 230$ (for Muscl Hancock) and at $t = 60, 120, 245$ (for RK3) using dotted, dashed and dash-dotted lines, respectively. Because of the small-amplitude oscillations around the equilibrium position, the last time is not the same. The solid line gives the empirical relation (45) for $0 < z < z_M$, where z_M is given by eq. (44).

AUSM+ Riemann solver (scheme *a*) disclose a somewhat greater amount of small scale structure than the HLL integration (scheme *b*). This is not surprising because of the former's ability to properly capture contact and shear waves. On the contrary, the latter greatly simplifies the wave pattern of the Riemann fan at the cost of introducing extra numerical dissipation. This is compensated, to some level, by the choice of the third-order interpolant (CENO3), which, nevertheless, is better combined with an equally accurate time marching scheme (RK3). The overall balance, therefore, favors the second-order scheme since only four Riemann problems must be solved instead of the six required by RK3; see Table 2. This conclusion is supported not only by the decreased numerical viscosity inherent to scheme *a*, but also by the reduced computational cost, which sees scheme *b* being ~1.6 times slower than scheme *a*.

In both cases we see that the Mach number profiles plotted in Figure 4 exhibit an excellent agreement with the empirical relation (45). It should be emphasized that the saturated position of the Mach disk oscillates around the experimental value, equation (44), by a few percent. The oscillations are caused by disturbances originated at the intersection between the Mach disk and the barrel shock surrounding the jet.

5.3. *Rotated MHD Shock Tube Problem*

This test sets up a Riemann problem between left and right states given, respectively, by

$$\begin{pmatrix} \rho \\ v_\parallel \\ v_\perp \\ v_z \\ p_g \\ B_\parallel \\ B_\perp \\ B_z \end{pmatrix}_L = \begin{pmatrix} 1.08 \\ 1.2 \\ 0.01 \\ 0.5 \\ 0.95 \\ 2/\sqrt{4\pi} \\ 3.6/\sqrt{4\pi} \\ 2/\sqrt{4\pi} \end{pmatrix}, \quad \begin{pmatrix} \rho \\ v_\parallel \\ v_\perp \\ v_z \\ p_g \\ B_\parallel \\ B_\perp \\ B_z \end{pmatrix}_R = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 2/\sqrt{4\pi} \\ 4/\sqrt{4\pi} \\ 2/\sqrt{4\pi} \end{pmatrix}, \tag{47}$$

where parallel and perpendicular components refer to the direction of structure propagation. The ratio of specific heats is $\Gamma = 5/3$. The line $2y = (\frac{1}{2} - x)$ (corresponding to a rotation angle of $\alpha = \tan^{-1}2$ with respect to the *y*-axis) is chosen as the surface of

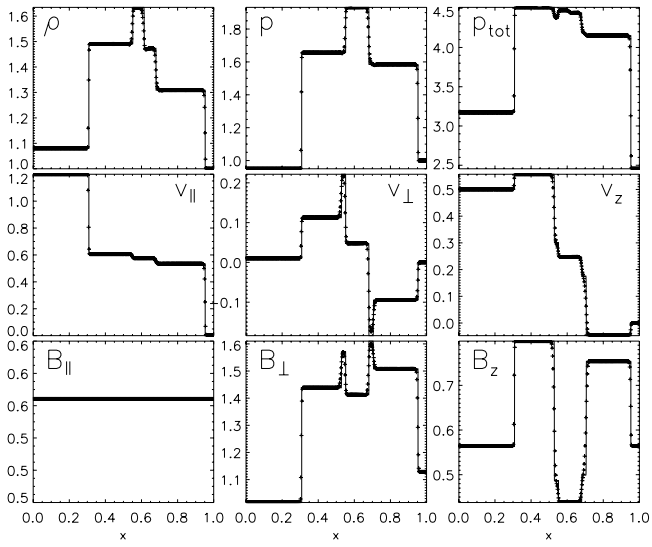

FIG. 5.—Rotated MHD shock-tube problem at $t = 0.2 \cos \alpha$. Symbols give the solution computed on a two-dimensional grid with 400×2 zone, while the solid line gives a reference solution computed for the nonrotated version at a resolution of 8192 zones.

discontinuity separating the two constant states and the computational domain $[0,\ 1] \times [0,\ 1/N_x]$ is discretized with $N_x \times 2$ zones, as in Li (2005). This setup yields a mesh spacing ratio $\Delta x/\Delta y = 2$, thus ensuring that the initial magnetic field is divergence-free (CT is used to advect magnetic fields). At the boundaries in the *y*-direction we impose translational invariance by applying shifted boundary condition; i.e., for any pair of indexes $(i,\ j)$ spanning the ghost zones, we set $V_{i,j} = V_{i-1,j+1}$ at the lower *y*-boundary and $V_{i,j} = V_{i+1,j-1}$ at the upper *y*-boundary.

The structure involves three-dimensional vector fields, and the outcoming wave pattern is bounded by two fast shocks (located at $x \approx 0.3$ and $x \approx 0.95$) enclosing two rotational discontinuities (visible only in the magnetic field), two slow shocks and one contact wave in at $x \approx 0.62$. Figure 5 shows a cut along the *x*-axis at $t = 0.2 \cos \alpha$ together with the high-resolution reference solution (*solid line*) computed from a one-dimensional integration at $t = 0.2$. The second-order unsplit Runge-Kutta method with a CFL number of 0.4 together with second-order linear interpolation on primitive variables and the Riemann solver of Roe were used for integration.

Computations were repeated at different resolutions starting from $N_x = 50$ (lowest) up to $N_x = 1600$ (highest), by doubling the number of zones each time. The performances and accuracies of the Roe, HLLD, and HLL Riemann solvers were compared in terms of errors (in L_1 norm) and CPU time, as shown in Figure 6. As indicated, HLLD and Roe yields comparable errors (9.53×10^{-4} and 9.36×10^{-4} at the highest resolution), whereas the HLL solver results in errors that are ~22% larger. In terms of CPU time, the HLLD solver is considerably faster than the Roe scheme and this is reflected in the right panel of Figure 6. Indeed, for a given accuracy, the computing time offered by HLLD significantly improves over the Roe solver and it is still slightly better than the cheaper HLL scheme. In this sense, HLLD offers the best trade-off between accuracy and efficiency.

A caveat: the previous considerations hold, strictly speaking, for the particular combinations of algorithms applied to the problem in question and should not be trusted as general statements. One may find, for example, that if interpolation is carried on characteristic variables (instead of primitive) the overall trend that favors HLL over Roe (in terms of CPU time) is reversed. Still, our results

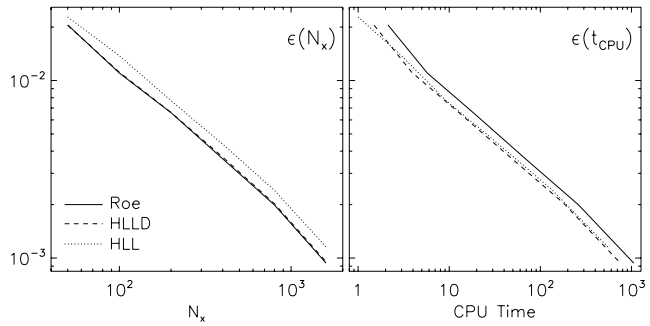

FIG. 6.—Errors in L_1 norm ($\epsilon_{L_1}(q) \equiv \sum |q_{ij} - q_{ij}^{\rm ex}|\Delta x_i \Delta y_j$, where q_{ij} and $q_{ij}^{\rm ex}$ are the numerical and analytical solutions on the mesh) for the rotated MHD shock-tube problem at $t = 0.2 \cos\alpha$. In order to consider all discontinuities ϵ was computed as the arithmetic average of $\epsilon_{L_1}(\rho)$ and $\epsilon_{L_1}(B_z)$. On the left, errors are plotted as function of the mesh size, whereas the left panel gives the errors as function of the CPU time, normalized to the fastest integration. Solid, dashed and dotted lines refer to computations carried with the Roe, HLLD, and HLL Riemann solvers, respectively.

illustrate how the choice of an algorithm or another may considerably bear on important issues of accuracy and efficiency.

5.4. *MHD Fast Rotor Problem*

A rapidly spinning ($\omega = 20$) cylinder with higher density is embedded in a static background medium with uniform pressure ($p = 1$), threaded by a constant magnetic field ($B_x = 5/(4\pi)^{1/2}$). Hydrodynamical variables smoothly change their values between the disk and the external environment using a taper function:

$$(\rho,\ v_\phi) = \begin{cases} (10,\ \omega r) & \text{for } r < r_0, \\ (1 + 9f,\ f\omega r_0) & \text{for } r_0 \le r \le r_1, \\ (1,\ 0) & \text{otherwise.} \end{cases} \qquad (48)$$

The disk has radius $r_0 = 0.1$, and the taper function $f = (r_1 - r)/(r_1 - r_0)$ terminates at $r_1 = 0.115$. The initial radial velocity is zero everywhere and the adiabatic index is $\Gamma = 1.4$. This problem has been considered by several authors; see for example Balsara & Spicer (1999), Londrillo & del Zanna (2004), Li (2005), and references therein.

Because of the geometrical setting, we have setup the problem in both Cartesian and polar coordinates. On the Cartesian grid, the domain is the square $[-\frac{1}{2},\ \frac{1}{2}]^2$ with outflow boundary conditions applied everywhere. On the polar grid, we choose $0.05 < r < 0.5, 0 < \phi < 2\pi$ with periodic boundary conditions in ϕ and zero-gradient at the outer radial boundary. At the inner boundary we set $\partial_r(v_r/r) = \partial_r(v_\phi/r) = 0$. For each geometry, we adopt the combination of algorithms listed in Table 3 and compare two different strategies to control the solenoidal constraint $\nabla \cdot \boldsymbol{B} = 0$, i.e., the CT and Powell methods.

Figure 7 shows density and magnetic pressure contours computed at $t = 0.15$ on the Cartesian and polar grids at a resolution of 400^2 zones. As the disk rotates, strong torsional Alfvén waves form and propagate outward carrying angular momentum from the disk to the ambient. Our results fully agree with those previously recovered by the above-mentioned authors and computations obtained with the CT and Powell scheme shows excellent agreement. Despite the higher complexity of the CT scheme where both staggered and zone-centered magnetic field are evolved in time, the computational cost turned out to be comparable (CT/Powell $\sim$ 1.05) for each system of coordinates. Moreover, the solutions computed in the two different geometries show nearly identical patterns, although polar coordinates (even with their intrinsically higher numerical viscosity at higher radii, due to the diverging nature of the grid) better fit the geometrical configuration of the problem. However, computations carried out on the polar grid were $\sim$5 times slower than the Cartesian one, because of the more severe time step limitation at the inner boundary, where the grid narrows. These considerations can considerably affect the choice of geometry, specially in long-lived simulations.

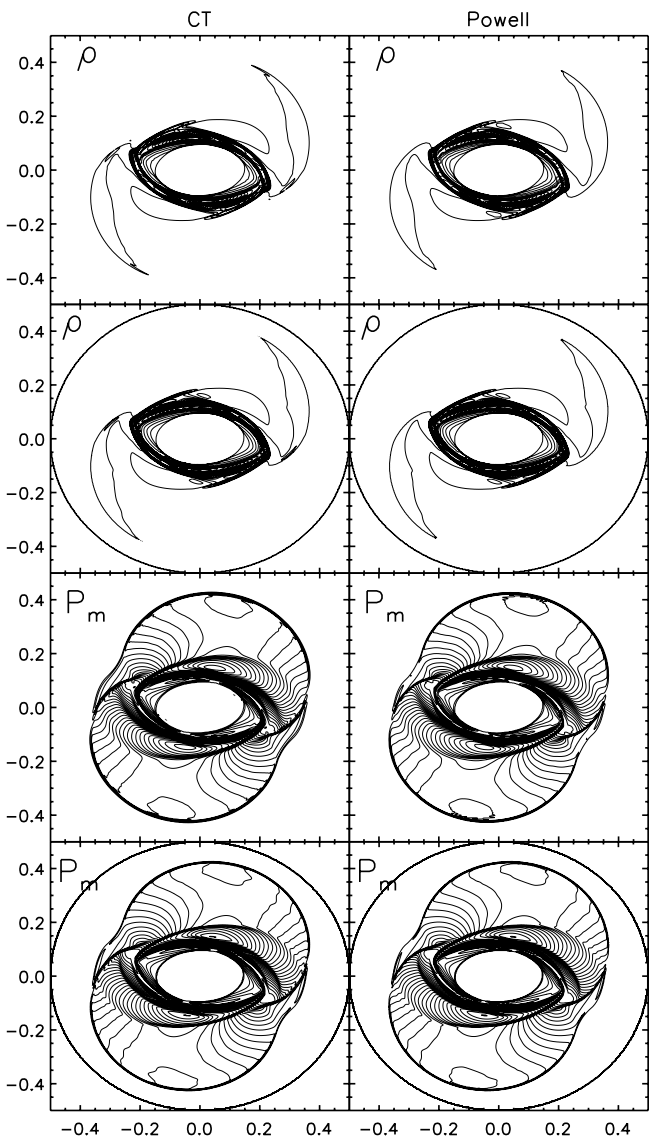

FIG. 7.—Contour maps for the MHD rotor problem at $t = 0.15$. Left and right panels show, respectively, computations carried with the CT and Powell's method in Cartesian coordinates (first and third rows from top) and polar coordinates (second and fourth rows from top). Thirty equally spaced contour levels are used for density ($0.483 < \rho < 13.21$, first and second rows) and magnetic pressure ($0.0177 < |\boldsymbol{B}|^2/2 < 2.642$, third and fourth rows).

5.5. *Magnetized Accretion Torus*

We now discuss an application of the code to a problem of astrophysical interest, along the lines of Hawley (2000). The problem involves a magnetized, constant angular momentum ($\Omega \propto (r \sin\theta)^{-2}$) torus in a (properly normalized) pseudo-Newtonian gravitational potential, $\Phi = -(r-1)^{-1}$. The torus has an equilibrium configuration described by the integral relation

$$\frac{\Gamma p}{(\Gamma - 1)\rho} = C - \Phi - \frac{1}{2}\frac{l_{\rm kep}^2}{r^2 \sin^2\theta}. \qquad (49)$$

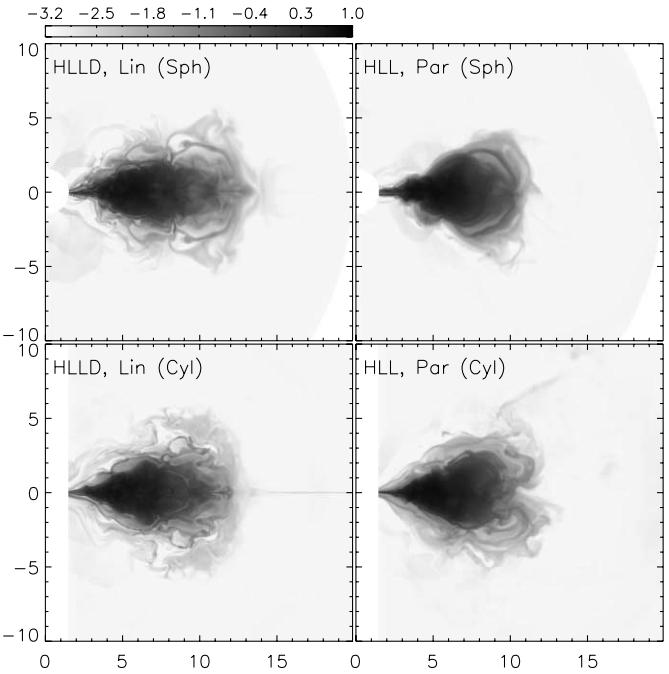

FIG. 8.—Density logarithm for the magnetized accretion torus at $t/\Delta T = 5.6$. Computations with the HLLD Riemann solver (*left*) and the HLL solver (*right*) are shown in spherical (*top*) and cylindrical (*bottom*) geometries. The region $0 < r < 1.5$ is excluded from computations. [*See the electronic edition of the Supplement for a color version of this figure.*]

Here C is determined by the radial location of the inner edge of the torus ($r = 3$ in our case) and $l_{\rm kep} = r^{3/2}/(r-1)$ is the Keplerian specific angular momentum at the pressure maximum ($r = 4.7$) where the orbital period is $\Delta T = 50$. Density and pressure are tied by the polytropic relation $p = K\rho^{\Gamma}$, with $\Gamma = 5/3$. A poloidal magnetic field is initialized inside the torus using the ϕ component of the potential vector $A_\phi \propto \min[\rho(r,\ \phi) - 5,\ 0]$, normalized by the condition $\min(2p/|\boldsymbol{B}|^2) = 10^2$.

We solve the problem in spherical as well as cylindrical coordinates. The spherical grid used for the problem is $1.5 \le r \le 20$ and $0 \le \theta \le \pi$, with 320×256 uniform zones covering the region $1.5 \le r \le 11.5$ and $\pi \le 4\theta \le 3\pi$. The remaining regions were covered by a geometrically stretched grid with 72 zones. Outflow boundaries are imposed at the innermost and outermost radii, whereas axisymmetry holds at $\theta = 0$ and π. In cylindrical coordinates, the computational box is $0 \le r \le 20$ and $-20 \le z \le 20$, with a uniform grid 320×320 in the region $1.5 \le r \le 11.5$ and $-5 \le z \le 5$. Axisymmetry boundary conditions are imposed at the axis, while all the rest are set to outflow. In addition, each case is further investigated by adopting (1) the HLLD solver with second-order slope limited reconstruction—cases *a* and *c* in Table 3—and (2) the HLL solver combined with piecewise parabolic reconstruction (cases *b* and *d* in Table 3). Second- and third-order Runge Kutta schemes are used for integration, respectively.

Figure 8 shows the density logarithm at $t/\Delta T = 5.6$ for the four cases, respectively. Initially, the strong shear generates a toroidal magnetic field component leading to a pressure-driven expansion of the torus and accretion takes place at $t/\Delta T \approx 1$. The growth of the poloidal field develops into a highly turbulent regime accounted for by the magneto rotational instability (MRI; Balbus & Hawley 1991) at $t/\Delta T \approx 4$. The effect is a net transport and redistribution of angular momentum.

Even though linear reconstruction presumably introduces more diffusion than parabolic interpolation, the HLLD solver, being more accurate, results in computations with a higher level of fine structure (compared to HLL) around the tori during the turbulent phase. Besides, by normalizing the CPU time to case *c*, we found the ratio $a:b$ and $c:d$ to be 1:1.75 (in cylindrical coordinates) and 1.55 : 2.42 (in spherical coordinates), respectively. Indeed, since higher than second-order interpolants are computationally more expensive and are used in conjunction with third-order time accurate schemes, the HLL integration comes at an extra computational cost. This suggests that the use of a more accurate Riemann solver can balance the lower order of reconstruction in space and result, at the same time, in more efficient computations. Moreover, note that the equatorial symmetry breaks down more rapidly when parabolic reconstruction is employed, but it is retained to a higher level for the second-order case. This is likely due to the larger number of operations introduced by the PPM reconstruction step, inevitably leading to faster-growing round off errors.

Finally we note that computations in spherical geometry suffer from the major drawbacks of introducing excessive numerical viscosity at the outer radii and to severely limiting the time step at the inner boundary. Both aspects are indeed confirmed in our numerical tests. On the other hand, a cylindrical system of coordinate does not experience a similar behavior and thus a higher turbulence level is still observed at larger radii.

5.6. *Three-dimensional Spherically Symmetric Relativistic Shock Tube*

An initial spherical discontinuity separates a high-pressure region ($p = 10^3$ for $r < 0.4$) from a uniform medium with $p = 1$ (for $r > 0.4$). The density is set to unity everywhere, and the gas is initially static.

Computations proceed by investigating the influence of two different equations of state (EoS), the constant Γ law with $\Gamma = 5/3$ (eq. [30]) and the TM equation (31). Moreover, we compare the performances of the PPM scheme (with the Two-Shock Riemann solver) with the simpler MH (with the HLL solver), which avoids characteristic decomposition of the equations (schemes *a* and *b* in Table 3). Spherical symmetry is assumed in the one-dimensional calculation.

The ensuing wave pattern (Fig. 9, *solid line*) is comprised of a left rarefaction wave, a contact discontinuity in the middle and a right going shock. The overall morphology is significantly affected by the choice of the EoS. Indeed, when the more realistic EoS is adopted, waves propagate at smaller velocities, and this is particularly evident at the head and the tail points of the left-going rarefaction wave. This results in a stronger adiabatic expansion for the constant Γ gas and in a denser shell between the shock and the trailing contact wave for the TM EoS.

Figure 10 reports the L_1 norm errors (for density) and the normalized CPU time as functions of the resolution $\Delta r^{-1} = 50 \times 2^n$ ($0 \le n \le 5$) for both schemes and for the selected equations of state. Errors are computed against a reference solution of 16×10^3 zones. The cheaper scheme *b* is certainly faster, although almost twice the resolution should be employed to achieve comparable accuracy with scheme *a*. The opposite trend is found for the PPM scheme, which is slower by almost a factor of 2 when compared to MH/HLL. Employment of the TM EoS results in an additional computational cost of about 25%–30% independently of the Riemann solver. However, the simple results obtained here suggest that a more realistic EoS can significantly impact the solution and thus justify the direct use of equation (31) instead of equation (30). Further reading on this topic, together with its extension to RMHD, may be found in Ryu et al. (2006) and Mignone & Mc Kinney (2007).

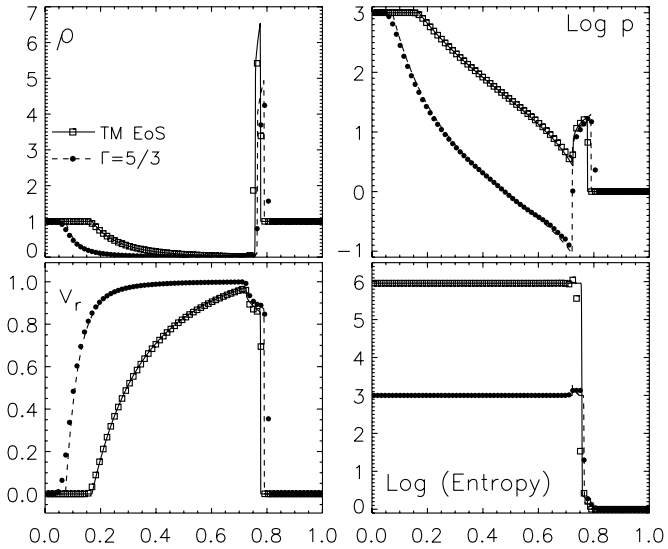

FIG. 9.—Spherically symmetric relativistic shock tube at $t = 0.4$. Solid and dashed lines give the one-dimensional reference solutions for the TM and $\Gamma = 5/3$ equations of state, respectively. Density (ρ), pressure logarithm ($\log p$), radial velocity (v_r) and entropy logarithm are shown. Overplotted filled circles and empty boxes show the same quantities along the main diagonal in the three-dimensional simulation. The CFL number is 0.8 and 128^3 zones have been used.

Spherically symmetric problems can serve as useful verification benchmarks to tests the code performance on three-dimensional Cartesian grids. For this purpose, we use the second-order ChTr scheme with the recently developed HLLC Riemann solver (Mignone & Bodo 2005) (scheme *c*) and adopt a resolution of 128^3 computational zones. By exploiting the symmetry, computations are carried out on one octant ($[0, 1]^3$) only by adopting reflecting boundary conditions. Results are overplotted using symbols in Figure 9. Small propagation structures such as the thin density shell pose serious computational challenges. Indeed we see that the maximum density peaks achieve ~85% and ~83% of the reference values for $\Gamma = 5/3$ and the TM equation of state, respectively. The smooth rarefaction wave is correctly captured at constant entropy with very small overshoots at the back. Results at higher resolution (not shown here) show increasingly better agreement.

Scaling performances have been tested for this problem on different parallel platforms, all yielding satisfactory scaling properties. Figure 12 below shows, for instance, the normalized execution time on the IBM SP Cluster 1600 p5-575 with $N_{\rm pr} = 2^p$ processors ($1 \le p \le 5$) for the same problem at a resolution of 128^3, indicating an almost ideal scaling ($\propto 1/N_{\rm pr}$).

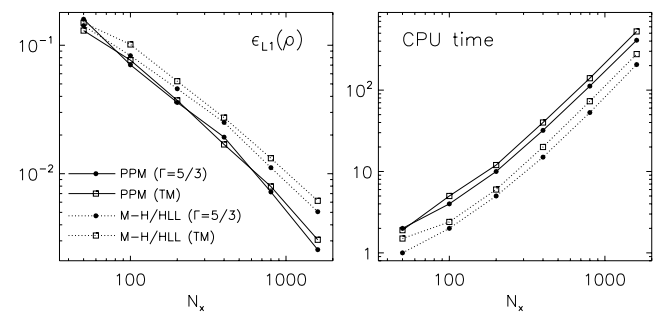

FIG. 10.—L_1 norm errors (*on the left*) and CPU time (*on the right*) for the one-dimensional spherically symmetric relativistic shock tube as function of the resolution. Solid and dotted lines label computations carried with the PPM scheme *a* and the MH scheme *b*. Results pertaining to different equations of state are marked with filled circles ($\Gamma = 5/3$) and boxes (TM EoS). The CPU time has been normalized to the fastest computation.

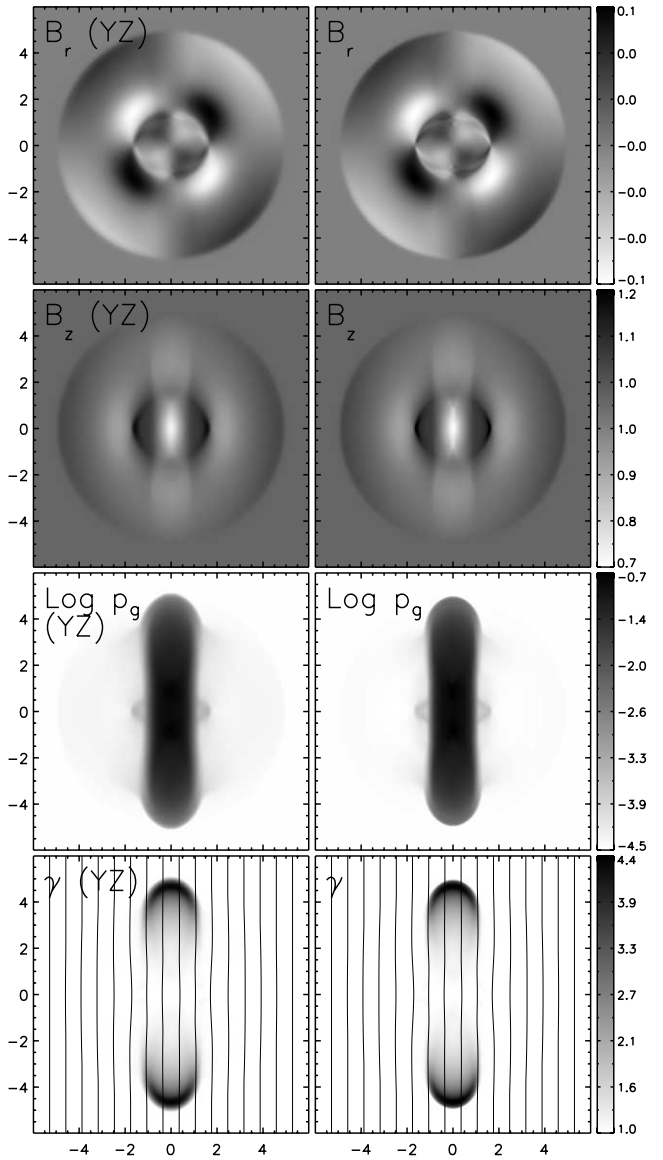

FIG. 11.—Relativistic magnetized blast wave at $t = 0.4$ in Cartesian (*left panels*; cuts in the *y*-*z*-plane at $x = 0$) and cylindrical (*right panels*) coordinates. Shown in the uppermost panels are the cylindrical radial and vertical magnetic field distributions with equally spaced color levels ranging from −0.052 to 0.052 (for B_r) and from 0.74 to 1.17 (for B_z). The bottom panels show, respectively, thermal pressure (in log scale) and the Lorentz factor with contours ranging from −4.54 to −0.74 and from 1 to 4.45. Magnetic field lines are overplotted on the Lorentz factor distribution. [*See the electronic edition of the Supplement for a color version of this figure.*]

5.7. *Three-dimensional Relativistic Magnetized Blast Wave*

A spherical region with density $\rho = 10^{-2}$ and thermal pressure $p = 1$ is embedded in a static uniform medium with $\rho = 10^{-4}$ and 3×10^{-5}. The sphere is centered around the origin and has radius $r = 0.8$. A linear smoothing function is applied for $0.8 < r < 1$. The whole region is threaded by a constant vertical magnetic field in the *z*-direction, $B_z = 1$. The ideal equation of state with specific heat ratio $\Gamma = 4/3$ is used. A similar setup in two-dimensional planar geometry was earlier considered by Komissarov (1999).

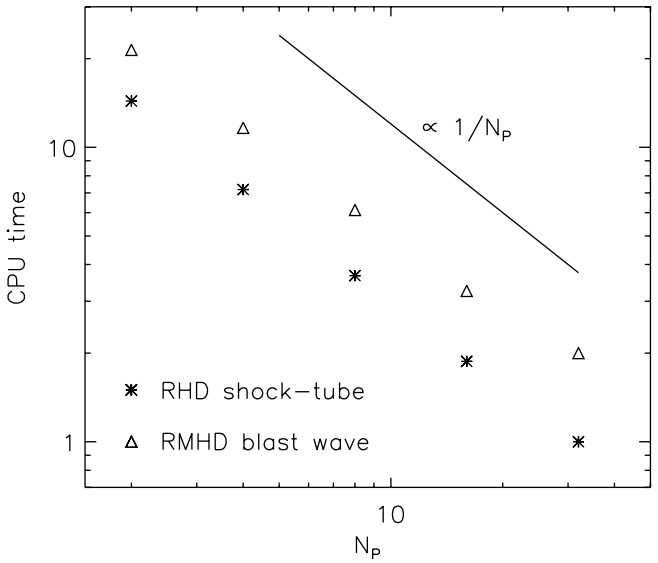

FIG. 12.—Scaling of PLUTO on 2, 4, 8, 16 and 32 processors for the three-dimensional relativistic shock tube interaction (*stars*) and the RMHD blast wave (*triangles*) test problems, respectively normalized to the 32 and 16 processor execution time. A grid of 128^3 was used. The perfect scaling slope ($\propto 1/N_{\rm pr}$, where $N_{\rm pr}$ is the number of processors) is shown as the solid line. Computation were performed on the IBM SP Cluster 1600 p5-575.

Our computational grid employs 512^3 zones on the box $[-6, 6]^3$. By exploiting the symmetric nature of the problem we reduce the computations to one octant ($0 < x,\ y,\ z < 6$) at half the resolution This is easily achieved by symmetrizing all quantities with respect to the coordinate planes and flipping the signs of ($B_x,\ v_x$) at $y = 0$, ($B_y,\ v_y$) at $x = 0$, ($v_z,\ B_x,\ B_y$) at $z = 0$.

Figure 11 shows the magnetic field distribution, thermal pressure, and Lorentz factor for the two configurations at $t = 4$. As a reference solution, computations on a cylindrical axisymmetric grid with $[512 \times 1024]$ zones are also shown. The expanding region is delimited by a fast forward shock propagating radially at almost the speed of light. Because of the strong magnetic confinement a jetlike structure develops in the z direction with a maximum Lorentz factor of $\gamma_{\rm max} \approx 4.5$. This problem is particularly challenging because of the very small plasma $\beta = 2p/|b_m|^2 = 6 \times 10^{-5}$ and instructive in checking the robustness of the code and the algorithm response to different kinds of degeneracies.

Scaling tests have been conducted in a way similar to § 5.6 and corresponding performance are shown in Figure 12.

6. CONCLUSIONS

In the context of astrophysical problems, where high-Mach number compressible flow dynamics plays a crucial role, a sizable number of numerical codes is now available to the community of scientists. Most of them are based on the reconstruct-solve-average strategy typical of the high-resolution shock-capturing Godunov-type codes. The reasons for choosing one code or another can be the most diverse, however, one can attempt to list a number of characteristics that a code should hold to be appealing for the skilled user, but not necessarily expert in numerics:

1. *A modular structure: the possibility to easily code and combine different algorithms to treat different physics*.—PLUTO offers a number of features that can be combined together. Besides four physical modules (e.g., Newtonian/relativistic hydrodynamics with or without magnetic fields), gravity, radiative cooling, resistivity, and multidimensional geometries, equations of state may be included where required.

2. *Provide the user with a number of numerical schemes tested against the most severe benchmarks*.—Several algorithms (e.g., Riemann solvers, interpolations, choice of boundary conditions and so on) have been coded in PLUTO and their choice is dictated by the problem at hand and/or by efficiency and robustness issues.

3. *Portability*.—We have successfully ported PLUTO to a number of different Unix-based systems among which IBM sp5/sp4, SGI Irix, Linux Beowulf clusters, Power Macintosh. In addition the code can run in either serial, single-processor, or parallel machines.

4. *Grid adaptivity: the ability to resolve flow features with different spatial and temporal scales on the same computational domain*.—The code provides a one-dimensional AMR integrator and multidimensional extensions are being developed by taking advantage of the CHOMBO library.[5] This feature will be distributed with future versions of the code.

5. *Last but not least, user friendliness*.—A simple user-interface based on the Python scripting language is available to setup a physical problem in a quick and self-explanatory way. The interface is conceived to minimize the coding efforts left to the user.

The code together with its documentation is freely distributed and it is available at the Web site.[6]

We would like to thank S. Ritta for his useful contributions to the underexpanded jet test and E. Beltritti for extensively testing the code performances on several platforms. The present work was partially supported by the European Community Marie Curie Actions-Human resource and mobility within the JETSET network under contract MRTN-CT-2004 005592. Numerical calculations were partly performed in CINECA (Bologna, Italy) thanks to the support by INAF.

A. M. would like to thank R. Rosner, T. Linde, and T. Plewa and all the people at the FLASH center at the University of Chicago for their helpful suggestions and discussions during the early development of the code.

[5] Library available at http://seesar.lbl.gov/ANAG/chombo.
[6] See http://plutocode.to.astro.it.

REFERENCES

Adamson, T. C., & Nicholls, J. A. 1959, J. Aerospace Sci., 26, 16
Alexiades, V., Amiez, G., & Gremaud, P. 1996, Commun. Num. Meth. Eng., 12, 31
Aloy, M. A., Ibáñez, J. M., Martí, J. M., & Müller, E. 1999, ApJS, 122, 151
Anile, A. M. 1989, Relativistic Fluids and Magnetofluids (Cambridge: Cambridge Univ. Press)
Anile, A. M., & Pennisi, S. 1987, Ann. Inst. Henri Poincaré, 46, 127
Ashkenas, H., & Sherman, F. S. 1965, Rarefied Gas Dynamics, Vol. 2, ed. J. H. de Leeuw (New York: Academic), 84
Balbus, S. A., & Hawley, J. F. 1991, ApJ, 376, 214
Balsara, D. S., & Spicer, D. S. 1999, J. Comput. Phys., 149, 270
Batten, P., Clarke, N., Lambert, C., & Causon, D. M. 1997, SIAM J. Sci. Comput., 18, 1553
Berger, M. J., & Colella, P. 1989, J. Comput. Phys., 82, 64
Bier, K., & Schmidt, Z. 1961, Z. Angew. Phys., 13, 493
Bodo, G., Chagelishvili, G., Murante, G., et al. 2005, A&A, 437, 9
Bodo, G., Rossi, P., Mignone, A., Massaglia, S., & Ferrari, A. 2003, NewA Rev., 47, 557
Cargo, P., & Gallice, G. 1997, J. Comput. Phys., 136, 446
Colella, P. 1990, J. Comput. Phys., 87, 171

Colella, P., & Woodward, P. R. 1984, J. Comput. Phys., 54, 174
Courant, R., Friedrichs, K. O., & Lewy, H. 1928, Math. Ann., 100, 32
Crockett, R. K., Colella, P., Fisher, R. T., Klein, R. I., & McKee, C. F. 2005, J. Comput. Phys., 203, 422
Del Zanna, L., & Bucciantini, N. 2002, A&A, 390, 1177
Einfeldt, B., Roe, P. L., Munz, C. D., & Sjogreen, B. 1991, J. Comput. Phys., 92, 273
Falle, S. A. E. G. 2002, ApJ, 577, L123
Fromang, S., Hennebelle, P., & Teyssier, R. 2006, A&A, 457, 371
Fryxell, B., Olson, K., Ricker, P., et al. 2000, ApJS, 131, 273
Gardiner, T. A., & Stone, J. M. 2005, J. Comput. Phys., 205, 509
Godunov, S. K. 1959, Matematicheskii Sbornik, 47, 271
Gottlieb, S., Shu, C.-W. 1996, NASA CR-201591 ICASE Rep. 96-50, 20 (Washington: NASA)
Hawley, J. F. 2000, ApJ, 528, 462
Janhunen, P. 2000, J. Comput. Phys., 160, 649
Jiang, G.-S., & Shu, C.-W. 1996, J. Comput. Phys., 126, 202
Knuth, E. L. 1964, Dept. Engineering Rep. 64 (Los Angeles: Univ. California)
Komissarov, S. S. 1999, MNRAS, 303, 343
Landau, L. D., & Lifshitz, E. M. 1959, Fluid Mechanics (Oxford: Pergamon)
LeVeque, R. J., Mihalas, D., Dorfi, E. A., & Müller, E. 1998, Computational Methods for Astrophysical Flow, ed. O. Steiner & A. Gautschy (Berlin: Springer)
Li, S. 2005, J. Comput. Phys., 203, 344
Liou, M.-S. 1996, J. Comput. Phys., 129, 364
———. 2006, J. Comput. Phys., 214, 137
Liou, M.-S., & Steffen, C. 1993, J. Comput. Phys., 107, 23
Londrillo, P., & del Zanna, L. 2004, J. Comput. Phys., 195, 17
Massaglia, S., Mignone, A., & Bodo, G. 2005, A&A, 442, 549
Masset, F. 2000, A&AS, 141, 165
Mignone, A. 2005, ApJ, 626, 373
Mignone, A., & Bodo, G. 2005, MNRAS, 364, 126
———. 2006, MNRAS, 368, 1040
Mignone, A., Massaglia, S., & Bodo, G. 2004, Ap&SS, 293, 199
———. 2005a, Space Sci. Rev., 121, 21
Mignone, A., & McKinney, J. C. 2007, MNRAS, in press
Mignone, A., Plewa, T., & Bodo, G. 2005b, ApJS, 160, 199
Miyoshi, T., & Kusano, K. 2005, J. Comput. Phys., 208, 315
Noble, S. C., Gammie, C. F., McKinney, J. C., & Del Zanna, L. 2006, ApJ, 641, 626
O'Sullivan, S., & Downes, T. P. 2006, MNRAS, 366, 1329
Powell, K. G. 1994, Rep. 94-24 (Langley: ICASE)
Powell, K. G., Roe, P. L., Linde, T. J., Gombosi, T. I., & de Zeeuw, D. L. 1999, J. Comput. Phys., 153, 284
Quirk, J. J. 1994, Int. J. Num. Methods Fluids, 18, 555
Raga, A. C., Mellema, G., & Lundqvist, P. 1997, ApJS, 109, 517
Roe, P. L. 1986, Annu. Rev. Fluid Mech., 18, 337
Rusanov, V. V. 1961, J. Comput. Math. Phys. USSR, 1, 267
Ryu, D., Chattopadhyay, I., & Choi, E. 2006, ApJS, 166, 410
Saltzman, J. 1994, J. Comput. Phys., 115, 153
Shames, I. H. 1983, Mechanics of Fluids (2nd ed.; Columbus: McGraw-Hill)
Stone, J. M, & Norman, M. L. 1992a, ApJS, 80, 753
———. 1992b, ApJS, 80, 791
Strang, G. 1968, SIAM J. Numer. Anal., 5, 506
Taub, A. H. 1948, Phys. Rev., 74, 328
Tevzadze, A., Bodo, G., Rossi, P., Mignone, A., & Chagelishvili, G. 2006, A&A, submitted
Teyssier, R., Fromang, S., & Dormy, E. 2006, J. Comput. Phys., 218, 44
Toro, E. F. 1997, Riemann Solvers and Numerical Methods for Fluid Dynamics (Berlin: Springer)
Toro, E. F., Spruce, M., & Speares, W. 1994, Shock Waves, 4, 25
Tóth, G. 1996, Astrophys. Lett. Commun., 34, 245
———. 2000, J. Comput. Phys., 161, 605
van Leer, B. 1974, J. Comput. Phys., 14, 361
Wada, Y., & Liou, M.-S. 1997, SIAM J. Sci. Computing, 18, 633
Woodward, P. R., & Colella, P. 1984, J. Comput. Phys., 54, 115
Young, W. S. 1975, Phys. Fluids, 18, 1421
Ziegler, U. 1998, Comput. Phys. Comm., 109, 111
———. 2004, J. Comput. Phys., 196, 393