A MODEL OF MIRA’S COMETARY HEAD/TAIL ENTERING THE LOCAL BUBBLE

A. Esquivel1, A. C. Raga1, J. Cantó2, A. Rodríguez-González1, D. López-Cámara1, P. F. Velázquez1, and F. De Colle3

1 Instituto de Ciencias Nucleares, Universidad Nacional Autónoma de México, Apartado Postal 70-543, 04510 México D.F., Mexico; esquivel@nucleares.unam.mx, raga@nucleares.unam.mx, d.lopez@nucleares.unam.mx, pablo@nucleares.unam.mx
2 Instituto de Astronomía, Universidad Nacional Autónoma de México, Apartado Postal 70-488, 04510 México D.F., Mexico
3 Department of Astronomy and Astrophysics, University of California Santa Cruz, Santa Cruz, CA 95064, USA; fabio@ucolick.org

1. INTRODUCTION

Mira is a well-studied binary system. Mira A is a prototypical, thermally pulsating luminous asymptotic giant branch (AGB) star (which in fact gives name to the so-called Mira class of variables), while Mira B is a much less luminous star, believed to be a white dwarf or a main-sequence star. Recently, the discovery of a cometary structure of $\sim 2\degree$ in the sky with its head centered on Mira, by Martin et al. (2007), with the GALEX satellite, has generated a renewed interest in this system. In order to compare the observations with our models, we include in Figure 1 a schematic diagram of the UV image of Martin et al. (2007).

The broadband UV filters (FWHM $\Delta\lambda = 256, 730$ Å, centered at $\lambda_c = 1516, 2267$ Å) in GALEX make it difficult to discriminate the physical mechanism responsible for the observed emission. Martin et al. (2007) suggested that most of the observed emission in the tail of Mira could be due to fluorescent H$_2$ lines, while in regions closer to the bow shock it is likely that other species such as C IV could contribute to the emission.

The distance to Mira estimated from Hipparcos is 107 pc (Knapp et al. 2003). The Mira system moves at a high speed ($\sim 125$ km s$^{-1}$) with respect to its surrounding medium, as deduced from its proper motion (Turon et al. 1993) and radial velocity (Evans 1967). The physical size of the cometary tail, assuming the Hipparcos distance, is $\sim 4\degree$.

New observations and/or re-examination of previous observations have been made in several wavelengths, including H I 21 cm (Matthews et al. 2008), IR (Ueta 2008), and optical (Meaburn et al. 2009). Recent theoretical work has been done by Wareing et al. (2007), Raga et al. (2008), and Raga & Cantó (2008). Wareing et al. (2007) performed a three-dimensional (3D) hydrodynamical simulation of the interaction between an isotropic AGB wind and the interstellar medium (ISM), modeled as a plane-parallel wind (considering a reference system in which Mira is at rest). Raga et al. (2008) included an inhomogeneous AGB wind, resulting in a more complicated shock structure for the cometary head (possibly more consistent with the observations). These two models, however, have one flaw: if the models are adjusted to produce the correct size for the cometary head, the tail that they produce is significantly broader than that of Mira’s comet.

The reason for this discrepancy with the observations lies in the fact that the ISM density in those models is estimated by assuming ram-pressure balance between Mira’s wind and the ISM at the tip of the head of the cometary structure. Therefore, the distance between the front of the bow shock and Mira is consistent with the observations. However, the tail produced by the model has a width comparable to the head, while Mira’s comet shows a considerably narrower tail (see Figure 1).

Wareing et al. (2007) speculated that Mira could have recently crossed into the Local Bubble (a tenuous region of coronal gas). Most of the tail could then have formed outside the Local Bubble, in a higher density ISM, resulting in the observed, narrow-tail structure. In this paper, we explore this possibility through a set of axisymmetric numerical simulations of a traveling star+wind system which emerges into a hot, coronal gas bubble.

The paper is organized as follows. In Section 2, we describe the model, including a review of the analytical solution of the wind/streaming environment by Wilkin (1996) and Cantó et al. (1996). The setup of the numerical simulations is presented in Section 3, followed by the results in Section 4, and a summary in Section 5. As this paper is the first publication with our new hydrodynamical code WALICXE, we include an appendix describing the code in detail, as well as some tests.

2. THE MODEL

Following Wareing et al. (2007) and Raga et al. (2008), we model Mira’s cometary tail as the interaction of a slow, dense, wind that arises from the AGB star Mira A, with a fast, lower-density, plane-parallel wind that results from the motion of Mira through the ISM (the simulations are performed in the reference frame in which Mira is at rest), as shown in the schematic diagram of Figure 2.

The structure that results from the interaction between a streaming, plane-parallel flow and a spherical wind has a
cometary shape that has been described analytically, considering conservation of linear and angular momentum in a "thin-shell" approximation (Wilkin 1996; Cantó et al. 1996). With this formalism, the shape of bow shock is found to have the form

$$R(\theta) = R_s \csc \theta \sqrt{3(1 - \theta \cot \theta)},$$

(1)

where $R$ is the spherical radius (measured from the center of the isotropic wind source), $\theta$ is the polar angle (measured from the symmetry axis, aligned with the direction toward the impinging, plane-parallel flow), and $R_s$ is the stagnation radius (or standoff distance, see Figure 2). The stagnation radius (point of closest approach to the spherical wind source) is given by the ram-pressure balance between the two winds:

$$\frac{M_w v_w}{4\pi R_s^2} = \rho_s v_s^2,$$

(2)

where $M_w$ is the mass-loss rate of Mira, $v_w$ is its terminal wind velocity, $\rho_s$ is the ambient (ISM) mass density, and $v_s$ is the velocity of the star moving through the surrounding environment.

Figure 3 shows the locus of the interaction region (location of the thin shell and shape of the bow-shock structure) for different stagnation radii. Note that for smaller $R_s$ the cometary tail becomes narrower.

3. THE NUMERICAL SETUP

We performed two hydrodynamical simulations of Mira’s tail with the new code WALICXE. The code solves the gas-dynamic equations in a block-based adaptive mesh that is designed to run in parallel on machines with distributed memory (i.e., clusters).

WALICXE can be run with a Cartesian two-dimensional (2D) mesh or an axisymmetric (cylindrical) grid. For this particular problem, we have used the axisymmetric version of the code with the hybrid HLL–HLLC Riemann solver (see the Appendix for details).

Along with the gas-dynamic equations, a rate equation for hydrogen ionization is solved. The resulting H ionization fraction is used to compute the radiative energy losses, which are obtained with the parameterized cooling function (that depends on the temperature, density, and hydrogen ionization fraction) described in Raga & Reipurth (2004).

The simulations have five root blocks of $16 \times 16$ cells each, aligned along the axial direction (z-axis). We allow seven levels of refinement, yielding an equivalent resolution of $1024 \times 5120$ (radial x axial) cells at the highest grid resolution. The computational domain extends $[0, 0.25] \times (0,1.25] \times 10^{19}$ cm in the radial and axial directions, respectively. Thus, the resolution at the highest level of refinement is $\approx 2.4 \times 10^{15}$ cm. We impose a reflective boundary condition along the symmetry axis ($r = 0$). Open boundary conditions are used at $r = 0.25 \times 10^{19}$ cm and $z = 0$.

In both models, the spherical wind has the following properties. It is injected (at every time step) in a spherical region of radius $3 \times 10^{16}$ cm, centered at $r = 0$, $z = 1.05 \times 10^{19}$ cm. The density inside the injection region follows a law $R^{-2}$ ($R$ is the spherical radius measured from the position of the wind source), scaled so that the mass-loss rate is $M_w = 3 \times 10^{-7}$ $M_\odot$ yr$^{-1}$. The wind is neutral with a temperature of $T_w = 10^6$ K and an outward, radially directed velocity of $v_w = 5$ km s$^{-1}$. The values of the physical parameters of Mira’s wind were taken from the observational studies of CO lines by Young (1995) and Ryde et al. (2000).

At the initial time ($t = 0$), both models are identical, with a plane-parallel wind that fills the computational domain (except inside the sphere in which the isotropic wind is imposed), which is replenished by an inflow condition at $z = 1.25 \times 10^{19}$ cm. The ISM is homogeneous, with a hydrogen number density...
$n_H = 1 \text{ cm}^{-3}$ (consistent with the ISM in the galactic plane, just outside the Local Bubble in the vicinity of Mira, see Vergely et al. 2001) and a velocity $v_a = 125 \text{ km s}^{-1}$ (in the $-z$ direction). The wind and ISM are neutral, except for a seed electron density assumed to arise from singly ionized C (i.e., the minimum ionization fraction is $10^{-4}$). The ISM has a temperature of $T_a = 10^3 \text{ K}$.

In model M1, we evolve these initial/boundary conditions up to an integration time of 350 kyr. Model M1 corresponds to the evolution of Mira outside of the Local Bubble, in a uniform medium dense enough to produce the narrow tail as observed.

In model M2, we simulate Mira entering the Local Bubble, a region much hotter and tenuous than the average ISM. To achieve this, after 200 kyr of evolution in the same medium as model M1, we raise the temperature of the ISM that enters the computational domain to $T_a = 10^6 \text{ K}$, lower its density to $n_H = 5 \times 10^{-2} \text{ cm}^{-3}$ (from the observations by Lallement et al. 2003), and change its ionization fraction to 1 (fully ionized hydrogen). We then let this configuration evolve up to a $t = 350 \text{ kyr}$ integration time.

The model of Mira entering the Local Bubble is well justified by the current Galactic Position of Mira $(l, b) = (168, -58)$ (Perryman et al. 1997; Wareing et al. 2007). This position puts Mira inside, but close to the edge of, the Local Bubble, as can be seen from the maps presented by Lallement et al. (2003). Given the high velocity of Mira, in the $\sim 300 \text{ kyr}$ that it takes to form a tail of 4 pc it would have traveled approximately 38 pc. This distance is enough to place the starting point of the tail formation outside the Local Bubble.

4. RESULTS

We allow the models to evolve up to an integration time of 350 kyr, which is of the order of the time needed to form a cometary tail of $\sim 4 \text{ pc}$ in length (the size of the structure observed with GALEX). Figure 4 shows the density (top half of the panels) and temperature distributions (bottom half) of M1 and M2 at the final evolutionary stage.

From the top panel in Figure 4 we can see that after 350 kyr a narrow tail, of axial and radial dimensions comparable with the observations of Mira, is formed. However, the standoff distance (in agreement with the analytical prediction in Equation (2)) is $R_s \sim 4.9 \times 10^{16} \text{ cm}$, which is much closer to the source compared to the $R_s \sim 3.1 \times 10^{17} \text{ cm}$ that can be deduced from the observations of Mira’s comet. The analytical solution for the parameters of model M1, plotted as a dotted line, traces well the region that separates the dense and cold material which arises from Mira’s wind and the hotter, lower density shocked ISM. This can be seen more clearly in the following section, where we...
separate the contribution of the wind and show column density maps.

The bottom panel of Figure 4 shows the tail that would be produced if Mira entered the Local Bubble some 145 kyr ago (i.e., model M2). The analytical solution for the parameters used for the Local Bubble is included as a dashed line. The corresponding standoff distance for model M2 is at \( R_s = 2.2 \times 10^{17} \) cm to the right of Mira, a distance that is close to what is observed (\( 3.1 \times 10^{17} \) cm).

This standoff distance and the corresponding width of the tail (dashed line) are much larger than observations. However, the change from one stationary configuration to the other is not instantaneous. As the cometary flow expands after entering the Local Bubble, the head grows rapidly. Approximately 20 kyr after the entrance of Mira into the Local Bubble, we obtain a structure that resembles remarkably well the broad-head/narrow-tail structure of Mira’s comet; such a tail remains well collimated for at least 100 kyr. It is only after 150 kyr of entering the Local Bubble that the tail becomes somewhat wider than the UV observations (see time sequence in the following section).

A clear difference in the wind/environment interface can be seen in models M1 and M2. In model M2, the outer boundary of the region filled by the stellar wind shows a more complex structure, which fragments the tail downstream from the wind source. Such structures are absent in Model M1 (see Figure 4). This difference is a result of the fact that while the wind/environment boundary has a highly supersonic velocity shear in model M1, this shear is barely transonic in model M2 (as Mira’s motion is basically sonic with respect to the hot, Local Bubble environment). It is a well-known experimental result that the spreading rate of sonic mixing layers is much larger than the spreading rate of high Mach number mixing layers (see Canto & Raga 1991); this effect is also present in our models, and therefore a more turbulent tail is seen in model M2.

### 4.1. Column Density Maps

One problem of interpreting the Martin et al. (2007) observations of Mira is that the broadband filters in GALEX do not give much information about the physical mechanisms that produce the observed emission. Martin et al. suggested that the bulk of the emission arises from \( \text{H}_2 \) molecules, which are likely to be present in the cold wind of Mira but not in the surrounding environment.

In order to make a better comparison with the observations, we have added a passive scalar in the simulations that allows us to identify the material from Mira’s wind and the ISM material. We then computed column density maps of the material from Mira’s wind (which would be molecular \( \text{H}_2 \)) and present them in the form of a time sequence in Figure 5.

In Figure 5, we see that the tail forms and reaches the observed length (of Mira’s comet) after approximately \( \sim 300 \) kyr. The ISM that enters from the right changes in model M2) to the Local Bubble parameters at \( t = 200 \) kyr, but it takes some 5 kyr to reach the two-wind interaction region. From then on, models M1 and M2 start to differ.

From the column density maps, it is clear that the stellar wind material in M1 is well confined to a region that resembles the tail of Mira for the duration of the simulation, but (as we have discussed above) this requires an ambient density that puts the axial standoff distance too close to Mira.

For model M2, as Mira crosses into the Local Bubble its wind expands toward a new steady-state configuration (which is not reached in our simulations). We found that at after only 20 kyr the stellar wind has expanded enough to form a dense structure extending \( \sim 2 \times 10^{17} \) cm upstream from the stellar wind source.

This size is in agreement with the axial extent of the head of Mira’s comet. We also see that the tail in model M2 expands at a much slower rate compared to the head, therefore reproducing the “broad-head/narrow-tail” morphology of Mira’s comet for a timescale of the order of 100 kyr. After 150 kyr inside the Local Bubble, the tail of Mira has expanded laterally beyond what is observed in the GALEX UV maps (see Martin et al. 2007 and diagram in Figure 1).

To simulate the UV emission that would arise from the \( \text{H}_2 \) molecules requires a chemical/radiation transfer calculation that is beyond the scope of this paper. We only note that if we use the standard conversion factor \( A_V \approx 10^{-21} \) (due to dust absorption), we find that the tail in our simulations has \( A_V \sim 10 \). Therefore, molecules present in the wind from Mira A would not be photodissociated in a substantial way, and the tail should indeed have a large \( \text{H}_2 \) fraction.

Two interesting issues arise from this work: one is the age of Mira’s tail itself and the other is how long it has been inside the Local Bubble (assuming that this is the case).

Regarding the age of the tail, in the original discovery of Martin et al. (2007), the authors estimate that the tail was formed in approximately 30 kyr, while the simulations of Wareing et al. (2007) suggested an age of 450 kyr. This large discrepancy is related to the rate at which Mira’s wind decelerates to merge dynamically with the ISM. Martin et al. (2007) assumed that the wind material instantly decelerates, so the 30 kyr is the time that Mira takes to traverse the length of the tail. In contrast, the 450 kyr timescale of Wareing et al. (2007) was obtained directly from numerical simulations which are, however, sensitive to viscous effects. Such effects depend mainly on two things: the density of the ISM that is used (a physical effect), and the amount of artificial viscosity (a numerical effect) needed to stabilize the code. Artificial viscosity usually overwhelms real viscosity, resulting in an enhanced deceleration, and thus the numerical estimates correspond to lower limits of the actual age.

Our simulations lie in between these two previous estimates as they require a time of \( \sim 300 \) kyr to develop Mira’s tail. This difference with the model of Wareing et al. (2007) is due to the higher ISM density of our models, which results in a higher deceleration rate.

The question of the time spent by Mira inside the Local Bubble is not well constrained by our models because the tail takes of the order of 100 kyr to expand laterally in an appreciable way. Our results suggest that Mira could have crossed into the Local Bubble between 20 and 120 kyr ago. This result is in rough agreement with the 130–225 kyr proposed by Wareing et al. (2007). In this respect, a more detailed map of the Local Bubble and an extrapolation of Mira’s motion might prove more useful.

### 5. SUMMARY

We have modeled the cometary structure observed around Mira as the interaction of the cold, dense wind from Mira with a streaming ISM. For this purpose we have used a newly developed code, which is described in detail in the Appendix. Similar simulations have been done previously (Wareing et al. 2007; Raga et al. 2008), but they fail to reproduce the broad-head/narrow-tail configuration characteristic of Mira’s comet.

The new element in our simulations is to assume that Mira has recently crossed from a (denser than previously considered) medium into the Local Bubble, a region of tenuous coronal gas.
This possibility was first suggested by Wareing et al. (2007) and is quite feasible given Mira’s current location.

By starting a stellar wind in a denser region than previous papers, we are able to reproduce the width of the observed tail of Mira’s comet. By allowing the stellar wind source to enter (at a later time) a less dense region of the ISM (i.e., the Local Bubble), we obtain an expansion of the cometary head. For the parameters of our model, the predicted cometary structure has a striking resemblance to Mira’s comet ~20 kyr after entering the Local Bubble, and the structure lasts for another ~100 kyr.

The age that we obtain for the formation of the tail is ~300 kyr. This can be compared with estimates by Martin et al. (2007, 30 kyr) and Wareing et al. (2007, 450 kyr) that assume either total or very little (respectively) deceleration of the wind as it encounters the ISM.

We should note that Meaburn et al. (2009) recently discovered a bipolar jet system in the Mira system (possibly ejected from Mira B). This collimated outflow might have an important effect in the formation of Mira’s cometary head and tail, which has not been considered in our present work, and should clearly be studied in the future.

This work has been supported by CONACYT grants 61547, 101356, and 101975. We thank Enrique Palacios, Martín Cruz,
APPENDIX

THE WALICXE CODE

Our previous adaptive grid codes (e.g., the YGUAZÚ-A, see Raga et al. 2000) were not suited for running in parallel on computers with distributed memory (i.e., clusters). In order to achieve high resolution by today’s standards, we have developed the new code WALICXE, which contains the cooling/chemistry modules of YGUAZÚ-A, but uses a new adaptive mesh designed to be easily parallelizable in clusters. At this point we only have a 2D version, with a 3D version in progress. We describe below the numerical algorithm and the adaptive mesh refinement (AMR) scheme of the WALICXE code.

A.1. Basic Equations

The 2D Euler system of equations can be written in Cartesian coordinates as

\[
\frac{\partial \mathbf{U}}{\partial t} + \frac{\partial \mathbf{F}}{\partial x} + \frac{\partial \mathbf{G}}{\partial y} = \mathbf{S},
\]

where

\[
\mathbf{U} = \begin{bmatrix} \rho \\ \rho u \\ \rho v \\ E \\ n_1 \\ n_2 \\ \vdots \\ n_r \\ n_r \end{bmatrix}, \quad \mathbf{F} = \begin{bmatrix} \rho u \\ P + \rho u^2 \\ \rho uv \\ u(E + P) \\ n_1 u \\ n_2 u \\ \vdots \\ n_r u \end{bmatrix}, \quad \mathbf{G} = \begin{bmatrix} \rho v \\ \rho vu \\ \rho v^2 \\ v(E + P) \\ n_1 v \\ n_2 v \\ \vdots \\ n_r v \end{bmatrix}, \quad \mathbf{S} = \begin{bmatrix} G - L \\ S_1 \\ S_2 \\ \vdots \\ S_r \end{bmatrix}.
\]

Here, \( \mathbf{U} \) are the so-called conserved variables, \( \rho \) is the mass density, \( (u, v) \) are the velocity components in the \( (x, y) \)-directions, and \( n_1, n_2, \ldots, n_r \) are densities of additional species that are advected passively into the fluid, which can be used to compute heating/cooling rates. \( \mathbf{F} \) and \( \mathbf{G} \) are the fluxes in the \( x \)- and \( y \)-directions, respectively,

\[
P = (n_i + n_e)kT
\]

is the thermal pressure, and the (total) energy is given by

\[
E = \frac{1}{2} \rho (u^2 + v^2) + C_v P,
\]

where \( C_v \) is the specific heat at constant volume. \( \mathbf{S} \) is the source vector that contains the energy gains and losses (\( \mathbf{G} \) and \( \mathbf{F} \) from radiative and collisional processes) and the reaction rates for the \( n_1, n_2, \ldots, n_r \) species; the source vector can also be extended to include geometrical terms for cylindrical and spherical coordinate systems. In addition, one can define a vector with the so-called primitive variables

\[
\mathbf{W} = \begin{bmatrix} \rho \\ u \\ v \\ P \\ n_1 \\ n_2 \\ \vdots \\ n_r \end{bmatrix},
\]

which are easily obtained from the conserved variables.

A.2. Discretization: Finite Areas

We will denote the indices and grid spacings of the computational cells in the \( (x, y) \)-directions by \( i, j \) and \( \Delta x, \Delta y \), respectively. The time is discretized with an index \( n \) such that \( \Delta t_n = t_{n+1} - t_n \), which in principle varies at each time step; however, to make the notation clearer we will drop the sub-index \( n \) from \( \Delta t_n \).

Integration of Equation (A1) on the area of the computational cell centered at \( i, j \) on a time interval \([t, t+\Delta t]\) yields the following exact expression:

\[
\mathbf{U}_{i,j}^{n+1} = \mathbf{U}_{i,j}^n - \frac{\Delta t}{\Delta x} \left( \frac{\mathbf{F}_{i+1/2,j}^{n+1/2} - \mathbf{F}_{i-1/2,j}^{n+1/2}}{2} \right) + \frac{\Delta t}{\Delta y} \left( \frac{\mathbf{G}_{i,j+1/2}^{n+1/2} - \mathbf{G}_{i,j-1/2}^{n+1/2}}{2} \right) + \Delta t \mathbf{S}_{i,j}^{n+1/2},
\]

where the variables \( \mathbf{U}_{i,j}^n \) are now area averages at a time \( t = t_n \):

\[
\mathbf{U}_{i,j}^n = \frac{1}{\Delta x \Delta y} \int_{y_{i-1/2}}^{y_{i+1/2}} \int_{x_{j-1/2}}^{x_{j+1/2}} \mathbf{U}(x, y, t_n) \, dx \, dy,
\]

where \( x \pm 1/2 = x_l \pm \Delta x/2, y \pm 1/2 = y_l \pm \Delta y/2 \) are the locus of the boundaries of the cell at \( x_l, y_l \). The fluxes are now time (and length) averages:

\[
\mathbf{F}_{i+1/2,j}^{n+1/2} = \frac{1}{\Delta y \Delta t} \int_{t_n}^{t_{n+1}} \int_{y_{i-1/2}}^{y_{i+1/2}} \mathbf{F}(x_{i+1/2}, y, t) \, dt \, dy,
\]

\[
\mathbf{G}_{i,j+1/2}^{n+1/2} = \frac{1}{\Delta x \Delta t} \int_{t_n}^{t_{n+1}} \int_{x_{j-1/2}}^{x_{j+1/2}} \mathbf{G}(x, y_{j+1/2}, t) \, dt \, dx.
\]

And the source terms are time and area averages:

\[
\mathbf{S}_{i,j}^{n+1/2} = \frac{1}{\Delta x \Delta y \Delta t} \int_{t_n}^{t_{n+1}} \int_{y_{i-1/2}}^{y_{i+1/2}} \int_{x_{j-1/2}}^{x_{j+1/2}} \mathbf{S}(x, y, t) \, dx \, dy \, dt.
\]

The method we have described is often referred to as finite volumes, but since we are dealing with 2D it is more accurate to call it finite areas. Of course, the expressions of Equations (A6)–(A10) are easily extended to 3D yielding a truly finite-volumes method, where the area integrals used above become volume integrals, and the length integrals become surface integrals. What remains now is to find approximations to the numerical intercell fluxes (\( \mathbf{F}_{i+1/2,j}^{n+1/2}, \mathbf{G}_{i,j+1/2}^{n+1/2} \)) and sources (\( \mathbf{S}_{i,j}^{n+1/2} \)). This can be done by solving the Riemann problem at the cell interfaces, that is, assuming a true discontinuity of the hydrodynamic variables at such interfaces (Toro 1999).
A.3. Solution Method: Finite-areas, Second-order Godunov Scheme

We use a second-order Godunov-type method to advance the solution; it uses a linear reconstruction (with slope limiters) of the primitive variables and an approximate Riemann solver to compute the fluxes at the intercell boundaries. As is customary, below we explain the method in one dimension; the expressions for the fluxes in the y-direction are analogous to those in the x-direction. The finite-volume (finite-lengths in one dimension) solution is

\[ U_i^{n+1} = U_i^n - \frac{\Delta t}{\Delta x} (F_{i+1/2}^{n+1} - F_{i-1/2}^n) + \frac{\Delta t}{2} S_i^{n+1/2}. \]  
(A11)

To obtain a second-order approximation of the intercell fluxes, we proceed as follows. First, we calculate the time step \( \Delta t \) in order to ensure that the standard Courant–Friedrichs–Lewy condition (Courant et al. 1967) is met. Then, a first-order half-time step is computed as

\[ U_i^{n+1/2} = U_i^n - \frac{\Delta t}{2\Delta x} (F_{i}^n - F_{i-1}^n) + \frac{\Delta t}{2} S_i^n, \]  
(A12)

where the fluxes are obtained solving the Riemann problem with an approximate Riemann solver (see Toro 1999), using the primitive variables \( W_i(U_i) \). That is,

\[ F_i = \text{Riemann}(W_i, W_{i+1}). \]  
(A13)

With the values of \( U_i^{n+1/2} \) from Equation (A12), we obtain a new set of primitives \( W_i^{n+1/2} \). We then use a linear reconstruction to extrapolate them to the cell boundaries:

\[ W_{L,i}^{n+1/2} = W_i^{n+1/2} - \frac{1}{2} \text{avg}(W_i^{n+1/2} - W_{i-1}^n, W_{i+1}^n - W_i^{n+1/2}), \]

\[ W_{R,i}^{n+1/2} = W_i^{n+1/2} + \frac{1}{2} \text{avg}(W_i^{n+1/2} - W_{i+1}^n, W_{i-1}^n - W_i^{n+1/2}). \]  
(A14a)

\[ W_{L,i}^{n+1/2} = W_{L,i}^{n+1/2}, \]

\[ W_{R,i}^{n+1/2} = W_{R,i}^{n+1/2}. \]  
(A14b)

where avg is an averaging function. The choice of this function is made in order to limit the slope used and avoid spurious oscillations. The code has a number of different averaging functions, the most widely known (although the most diffusive) being the minmod function.

The left and right states in Equations (A14) are used to approximate the fluxes used in Equation (A11):

\[ F_{i+1/2}^{n+1} = \text{Riemann}(W_{R,i}^{n+1/2}, W_{L,i+1}^{n+1/2}). \]  
(A15a)

\[ F_{i-1/2}^{n+1} = \text{Riemann}(W_{R,i-1/2}^{n+1/2}, W_{L,i}^{n+1/2}). \]  
(A15b)

At this point a new \( \Delta t \) is calculated, and we iterate until the desired evolutionary stage is reached. The time step is the same for all blocks, independently of their level of resolution; this has a cost in computing time, and introduces additional numerical viscosity to coarser blocks (because they are effectively run at a smaller Courant number), but makes it easier and more efficient to balance the load among different processors.

As to the Riemann solver used, the default is the HLLC algorithm, an improvement proposed by Toro et al. (1994) to the HLL (Harten et al. 1983) solver, which restores the information of the contact wave into the solver (the C stands for contact). The original HLL is also available in the code but is rarely used because it is too diffusive. In addition to this solver we have implemented a hybrid scheme that combines the HLL and HLLC fluxes to fix the so-called carbuncle or shock instability problem, which occurs when strong shocks align with the computational grid. Our HLL–HLLC scheme is very similar to those proposed by Kim et al. (2009) and Huang et al. (2010); the description and some tests are given in Appendix A.6.

A.4. The Adaptive Grid

The hydrodynamic+rate equations described above are integrated on a binary, block-based computational grid with the following characteristics.

A.4.1. Blocks (Patches)

The adaptive mesh was designed to be used for parallel computers with distributed memory (i.e., clusters). It consists of a series of “root” blocks with a predetermined (set at compile time) number of cells, \( n_x \times n_y \). Each block can be subdivided in a binary fashion into four blocks with the same number of cells of the parent block. We will refer to these new blocks as “siblings,” which have twice the resolution of their parent block. The maximum number of levels of refinement allowed (\( n_{levs} \)) is set at compile time. Thus the maximum resolution is \( \Delta = L_i/[2^{n_{levs}-1}(n_i)], \) where \( L_i \) is the size of the computational mesh on the \( k \)-th direction (i.e., \( i = x, y \)). In addition, each block has its own ghost cells, two in each direction for the second-order scheme described below, but can be changed to implement higher-order schemes. These cells would be communicated with Message Passing Interface (MPI) when adjacent blocks are in different processors.

The result is a tree structure (of blocks or patches, not individual cells): the root blocks branch into higher resolutions up to \( n_{levs} \). The equations are only solved in “leaf” blocks, that is, only on blocks that are not further refined.

A.4.2. Refinement/Coarsening Criteria

Before advancing the solution, at every time step we update the mesh. This is done by sweeping all the leaf blocks, measuring pressure and density gradients on each of them. Ifeither the density or pressure suffers a relative gradient, between two adjacent cells, above a threshold (user defined, typically \( \sim 0.05 \)) the block is flagged for refinement. In addition, we enforce that neighboring blocks differ at most on one level of refinement: blocks in proximity to a higher resolution that has been marked for further refinement are also flagged. Coarsening is also controlled by measuring gradients of density and pressure, but with a different tolerance (also user defined, typically \( \sim 1.0 \)); if all the relative gradients in the block are below this value the block is marked to be coarsened. Coarsening is only allowed when an entire group of four siblings has been marked and it is not impeded by the proximity criterion.

The actual refinement algorithm assigns the same value of the variables in the parent block to all the siblings (zeroth-order interpolation), whereas the coarsening assigns the mean value of each variable from the four siblings to their parent block.

Additional refinement criteria are easy to implement depending of the problem to be solved. One can, for instance, use a vorticity criterion, or refine above a certain value of any variable, or in certain fixed regions, or in regions labeled by a passive scalar advected into the flow.

A.4.3. Parallelization and Load Balancing

Since each of the blocks has the same number of cells, it is straightforward to send different blocks (actually a group of
blocks) to different processors. This is achieved with the MPI standard. At every time step the blocks that share border with other processors communicate their boundaries (ghost cells). Domain decomposition is made following a Hilbert ordering in similar fashion to the RAMSES code (Teyssier 2002). Each block is assigned a number according to its position in a Hilbert curve of the $n_{levs} - 1$ order. Following such a curve, the number of leaf blocks is divided evenly among the available processors, and the blocks are distributed to be solved. The tree structure is copied entirely to every processor, but only the leaf blocks are distributed. This, however, is a time-consuming operation, and the workload is only balanced every $n$ time steps. This, however, is a time-consuming operation, and the workload is only balanced every $n$ time steps. The run in panel (a) has five levels of refinement: for a maximum resolution of $\Delta x = \Delta y = 1/160$; the run in panel (b) has six levels of refinement and a maximum resolution of $\Delta x = \Delta y = 1/320$. Below each of the panels, the AMR structure (only the $[0, 3] \times [0, 1]$ region) is shown; each square corresponds to one $n_x \times n_y$ block, and in color we show the domain decomposition in eight processors. Note how the mesh is only refined at places where the flow properties change, and it remains coarse at regions where the flow is smooth.

A.5. Testing the Code

Several standard tests have been made of the code; we present here one that we believe is very illustrative: the classic double Mach reflection test problem proposed by Woodward & Colella (1984). The test was inspired by laboratory studies and consists of a planar shock that meets a reflecting surface at an angle of $\pi/3$. The reflecting surface is taken to be the $x$-axis, and the initial conditions (explained in detail in Woodward & Colella (1984) of the problem are:

$$(\rho, u, v, P) = \begin{cases} (1.4, 0, 0, 1) & \text{for } x > x_s, \\ (8, 8.25 \sin \alpha, -8.25 \cos \alpha, 116.5) & \text{otherwise}, \end{cases}$$

which correspond to a Mach 10 shock in air (using an adiabatic equation of state with $C_v = 2.5$, $\gamma = 1.4$). The problem is solved on a Cartesian domain $[0, 4] \times [0, 1]$ with the following boundary conditions: outflow boundaries in $x = 4$; reflective boundaries in $y = 0$ for $x > 1/6$; at $x = 0$ and $y = 0$ for $x < 1/6$ inflow conditions are used; and at $y = 1$ time-dependent inflow conditions are applied, tracing the position of the shock front that moves according to $x_s(t) = 1/6 + 10 \, y \, \sin \alpha + y \, \tan \alpha$. The flow configuration produces a series of shocks, two of which are almost perpendicular and are separated by a contact discontinuity, and a small jet at the base of the reflecting surface. This jet is driven by a pressure gradient and its evolution is known to be very sensitive to numerical diffusion. The results of this test can be compared with those of Woodward & Colella (1984) and many other codes which have adopted it as a standard test for 2D hydrodynamics (e.g., Mignone et al. 2007; Stone et al. 2008).

In Figure 6, we show the results of the double Mach reflection test with the HLLC solver, for an integration time of $t = 0.2$, for two different resolutions. Both runs start with four square root blocks aligned along the $x$-axis. The root blocks have $10 \times 10$ cells, covering a physical size of $(1 \times 1)$. The run in panel (a) has five levels of refinement: for a maximum resolution of $\Delta x = \Delta y = 1/160$; the run in panel (b) has six levels of refinement and a maximum resolution of $\Delta x = \Delta y = 1/320$. Below each of the panels, the AMR structure (only the $[0, 3] \times [0, 1]$ region) is shown; each square corresponds to one $n_x \times n_y$ block, and in color we show the domain decomposition in eight processors. Note how the mesh is only refined at places where the flow properties change, and it remains coarse at regions where the flow is smooth.

Another feature that can be identified is the fact that the jet formed along the reflecting surface (at $x \sim 2.4–2.8$) reaches the shock that is almost perpendicular to the surface ($x \sim 2.8$), bending it slightly at its base. This is often called “the kinked Mach stem,” and it is unphysical, caused by insufficient numerical diffusion at places where shocks closely align with the computational grid (Quirk 1994). This issue becomes particularly critical for high-resolution simulations (less numerical/artificial viscosity).

A.5.1. Parallel Performance

It is customary to measure the performance of a parallel code in terms of its scaling with the number of processors. Ideally the computing time would scale in proportion to $1/N_p$, where $N_p$ is the number of processors. In fact, many fixed grid codes show near-ideal scaling. The reason for such a good scaling is twofold: (1) most of the code is parallelizable, which restricts the communication between processors to passing boundaries and determining the time step, and (2) since the grid is fixed, each processor knows beforehand with which processors the communication will take place.

Adaptive mesh codes use a significant amount of central processing unit (CPU) time to maintain the AMR structure, and neighboring blocks can be in different processors at different times. This causes AMR codes to scale less than ideally, but the CPU time and memory requirements can remain significantly
lower than for fixed grid codes. Also, there are many variables that can impact the performance, and many of them are problem dependent. To illustrate this, we took the double Mach reflection test with an equivalent resolution of 2560 × 640 cells, on 2, 4, 8, 16, and 32 processors. To achieve the same resolution at the highest level of refinement, we used three combinations: five levels of refinement with 40 × 40 cells blocks, six levels with 20 × 20 cells, and seven levels with 10 × 10 blocks; the scaling results are presented in Figure 7.

We see from the figure a good performance of the code, almost linear for a small number of processors. As we increase the number of processors the portion of domain that is solved by each processor becomes too small and the communication time becomes important compared to the time to solve each region. Certainly, if we require an even finer resolution, the transition from ideal to not-so-ideal would happen at a larger number of processors.

It is quite noticeable that by allowing the mesh to be finer the CPU time drops dramatically. We must note that the outcome from ideal to not-so-ideal would happen at a larger number of processors. Certainly, if we require an even finer resolution, the transition from ideal to not-so-ideal would happen at a larger number of processors.

We see from the figure a good performance of the code, almost linear for a small number of processors. As we increase the number of processors the portion of domain that is solved by each processor becomes too small and the communication time becomes important compared to the time to solve each region. Certainly, if we require an even finer resolution, the transition from ideal to not-so-ideal would happen at a larger number of processors.

We see from the figure a good performance of the code, almost linear for a small number of processors. As we increase the number of processors the portion of domain that is solved by each processor becomes too small and the communication time becomes important compared to the time to solve each region. Certainly, if we require an even finer resolution, the transition from ideal to not-so-ideal would happen at a larger number of processors.

We see from the figure a good performance of the code, almost linear for a small number of processors. As we increase the number of processors the portion of domain that is solved by each processor becomes too small and the communication time becomes important compared to the time to solve each region. Certainly, if we require an even finer resolution, the transition from ideal to not-so-ideal would happen at a larger number of processors.

We see from the figure a good performance of the code, almost linear for a small number of processors. As we increase the number of processors the portion of domain that is solved by each processor becomes too small and the communication time becomes important compared to the time to solve each region. Certainly, if we require an even finer resolution, the transition from ideal to not-so-ideal would happen at a larger number of processors.

We see from the figure a good performance of the code, almost linear for a small number of processors. As we increase the number of processors the portion of domain that is solved by each processor becomes too small and the communication time becomes important compared to the time to solve each region. Certainly, if we require an even finer resolution, the transition from ideal to not-so-ideal would happen at a larger number of processors.

We see from the figure a good performance of the code, almost linear for a small number of processors. As we increase the number of processors the portion of domain that is solved by each processor becomes too small and the communication time becomes important compared to the time to solve each region. Certainly, if we require an even finer resolution, the transition from ideal to not-so-ideal would happen at a larger number of processors.
features of the other two solvers is evident, a barely noticeable kinked Mach stem, with a sharp contact surface.

REFERENCES

Cantó, J., & Raga, A. C. 1991, ApJ, 372, 646
Cantó, J., Raga, A. C., & Wilkin, F. P. 1996, ApJ, 469, 729
Courant, R., Friedrichs, K., & Lewy, H. 1967, IBM J. Res. Dev., 11, 215
Evans, D. S. 1967, in IAU Symp. 30, Determination of Radial Velocities and their Applications, ed. A. H. Batten & J. F. Heard (Dordrecht: Kluwer), 57
Harten, A., Lax, P. D., & van Leer, B. 1983, SIAM Rev., 25, 357
Huang, K., Wu, H., Yu, H., & Yan, D. 2010, Int. J. Numer. Methods Fluids (doi:10.1002/fld.2217)
Kim, S. D., Lee, B. J., Lee, H. J., & Jeung, I. 2009, J. Comput. Phys., 228, 7634
Knapp, G. R., Pourbaix, D., Platais, I., & Jorissen, A. 2003, A&A, 403, 993
Lallemand, R., Welsh, B. Y., Vergely, J. L., Crifo, F., & Sfeir, D. 2003, A&A, 411, 447
Liou, M. 2000, J. Comput. Phys., 160, 623
Martin, D. C., et al. 2007, Nature, 448, 780
Matthews, L. D., Libert, Y., Gérard, E., Le Bertre, T., & Reid, M. J. 2008, ApJ, 684, 603
Meaburn, J., López, J. A., Bounis, P., Lloyd, M., & Redman, M. P. 2009, A&A, 500, 827
Mignone, A., Bodo, G., Massaglia, S., Matsakos, T., Tesileanu, O., Zanni, C., & Ferrari, A. 2007, ApJS, 170, 228
Nishikawa, H., & Kitamura, K. 2008, J. Comput. Phys., 227, 2560
Perryman, M. A. C., et al. 1997, A&A, 323, L49
Quirk, J. J. 1994, Int. J. Numer. Methods Fluids, 18, 555
Raga, A. C., & Cantó, J. 2008, ApJ, 685, L141
Raga, A. C., Cantó, J., De Colle, F., Esquivel, A., Kajdic, P., Rodríguez-González, A., & Velázquez, P. F. 2008, ApJ, 680, L45
Raga, A. C., Navarro-González, R., & Villagrá-Muniz, M. 2000, RevMexAA, 36, 67
Raga, A. C., & Reipurth, B. 2004, RevMexAA, 40, 15
Roe, P. L. 1981, J. Comput. Phys., 43, 357
Ryde, N., Gustafsson, B., Eriksson, K., & Hinkle, K. H. 2000, ApJ, 545, 945
Stone, J. M., Gardiner, T. A., Teuben, P., Hawley, J. F., & Simon, J. B. 2008, ApJS, 178, 137
Teyssier, R. 2002, A&A, 385, 337
Toro, E. F. 1999, Riemann Solvers and Numerical Methods for Fluid Dynamics (Berlin: Springer)
Toro, E. F., Spruce, M., & Speares, W. 1994, Shock Waves, 4, 25
Turon, C., et al. 1993, Bull. Inf. Cent. Donnees Stellaires, 43, 5
Ueta, T. 2008, ApJ, 687, L33
Vergely, J., Freire Ferrero, R., Siebert, A., & Valette, B. 2001, A&A, 366, 1016
Wareing, C. J., Zijlstra, A. A., O’Brien, T. J., & Seibert, M. 2007, ApJ, 670, L125
Wilkin, F. P. 1996, ApJ, 459, L31
Woodward, P., & Colella, P. 1984, J. Comput. Phys., 54, 115
Young, K. 1995, ApJ, 445, 872