Chapter 2

GENESIS: A high–resolution code for 3D RHD

2.1. Introduction

Numerical relativistic hydrodynamics (RHD) has experienced an important step forward in recent years when modern high–resolution shock–capturing (HRSC) techniques began to be applied to solve the equations of RHD in conservation form. Prior to the advent of HRSC techniques the field was dominated for more than one decade by Wilson (1979)'s approach to relativistic hydrodynamics. This approach relies on the use of artificial viscosity in order to handle the discontinuities (shocks, contact discontinuities, etc.) that may appear in the flow numerically. However, techniques based on artificial viscosity are prone to severe numerical difficulties when simulating ultrarelativistic flows (see, e.g., Centrella & Wilson 1984). Using modern HRSC techniques instead, these difficulties are overcome (see, e.g., Donat *et al.* 1998) allowing one to simulate challenging relativistic astrophysical phenomena like, e.g., relativistic jets or gamma-ray bursts (GRB hereafter).

The need for a relativistic treatment in the framework of astrophysical jets is justified because flow velocities as large as 99.5% of the speed of light (Lorentz factors > 10) are required – according to the nowadays accepted standard model (see 3.1.2) – to explain the apparent superluminal motion observed at parsec scales in many jets of extragalactic radio sources associated to active galactic nuclei. Similar arguments applied to the galactic superluminal sources GRS1915+105 (Mirabel & Rodriguez 1994) and GROJ1655–40 (Tingay et al. 1995) allow one to infer intrinsic velocities of $\approx 0.9c$ in the jets of these sources. Further independent indication of highly relativistic speeds can be inferred from the intraday variability occurring in more than a quarter of all compact extragalactic radio sources (Krichbaum, Quirrenbach & Witzel 1992). If the observed intraday radio variability is intrinsic and results from incoherent synchrotron radiation (according to Begelman, Rees & Sikora 1994), the associated jets must have bulk Lorentz factors in the range $\sim 30 - 100$.

Another astrophysical phenomenon which also involves flows with velocities very close to the speed of light are the GRBs. The standard fireball model (see 4.1.1) assumes that the fireball that originates the GRB is accelerated to reach Lorentz factors $> 10^2$ (Cavallo & Rees 1978; Piran, Shemi & Narayan 1993). In addition to this high velocities, the thermodynamics of the GRB production needs a relativistic treatment (for example, the energies involved in the GRB production are ~ 1 MeV per particle –Rees 1997–), which for practical purposes means that a consistent relativistic equation of state (EOS) must be used. Furthermore, this relativistic thermodynamics is closely linked to the dynamical evolution of any relativistic plasma, governed by the RHD equations, through the value of the enthalpy as was pointed out by Martí *et al.* (1997) in the context of relativistic jets.

In the following we describe the main features of a special relativistic 3D hydrodynamic code, which is based on explicit HRSC methods, and which is a considerably extended version of the special relativistic 2D hydrodynamic code developed by Martí, Müller & Ibáñez (1994) and by Martí *et al.* (1995). The code has been designed modularly which allows one to use different reconstruction algorithms and Riemann solvers. As it is the final goal of our work to simulate relativistic jets and GRBs in three spatial dimensions, the code has successfully been subjected to an intensive testing in the ultrarelativistic regime (see Section (2.4)). In particular, GENESIS has successfully passed the spherical shock reflection test (simulated in 3D Cartesian coordinates) involving flow Lorentz factors larger than 700 (see §(2.4.3)).

The chapter is organized as follows. In Section (2.2), we introduce the 3D equations of RHD in Cartesian coordinates in differential and discretized forms. The latter have been implemented into our 3D RHD code GENESIS. Detailed information about the structure and the main features of the code is given in Section (2.3). Several 1D, 2D and 3D relativistic test problems computed with GENESIS are described in Section (2.4). The performance of GENESIS on scalar and multi-processor computers is analyzed in Section (2.5).

A summary of the chapter containing our main conclusions can be found in Section (2.6). In Appendix A, we give the spectral decomposition of the three dimensional system of RHD equations with explicit expressions for the eigenvalues and the right- and left-eigenvectors. Appendix B shows how to get an efficient implementation of flux formulae in multidimensional relativistic hydrodynamical codes paying special attention to the explicit formulae for the numerical viscosity of Marquina's (Donat & Marquina 1996) flux formula. The Appendix C describes the explicit algorithm to recover the primitive variables form the conserved ones.

2.2. Equations of RHD in conservation form

The evolution of a relativistic perfect fluid is described by five conserved quantities: rest mass density, D, momentum density, $\mathbf{S}$, and energy density, τ (all of them measured in the laboratory frame and in natural units, i.e., the speed of light $c = 1$),

$$
\begin{aligned}
D &= \rho W && (2.1)\\
S^j &= \rho h W^2 v^j \quad (j = 1, 2, 3) && (2.2)\\
\tau &= \rho h W^2 - p - \rho W, && (2.3)
\end{aligned}
$$

where the Lorentz factor $W = (1 - \mathrm{v}^2)^{-1/2}$ and $\mathrm{v}^2 = \delta_{ij} v^i v^j$ (the Einstein summation convention is used here, and δ_{ij} is the Kronecker symbol). Furthermore, ρ is the rest–mass density, p the pressure and h the specific enthalpy given by $h = 1 + \varepsilon + p/\rho$ with ε being the specific internal energy. The components of the vector of variables $\mathbf{w} \equiv (\rho, v^i, \varepsilon)^T$ are called *primitive* or physical variables.

The relativistic Euler equations form a system of conservation laws (see, e.g., Font *et al.* 1994) which can be written in Cartesian coordinates as

$$\frac{\partial D}{\partial t} + \sum_{j=1}^{3} \frac{\partial}{\partial x^j}(Dv^j) = 0 \tag{2.4}$$

$$\frac{\partial S^i}{\partial t} + \sum_{j=1}^{3} \frac{\partial}{\partial x^j}(S^i v^j + \delta^{ij} p) = 0 \qquad \text{(i=1, 2, 3)} \tag{2.5}$$

$$\frac{\partial \tau}{\partial t} + \sum_{j=1}^{3} \frac{\partial}{\partial x^j}(S^j - Dv^j) = 0 \tag{2.6}$$

or, equivalently, as

$$\frac{\partial \mathbf{U}}{\partial t} + \sum_{i=1}^{3} \frac{\partial \mathbf{F}^i}{\partial x^i} = 0\,, \tag{2.7}$$

where the vector of unknowns $\mathbf{U}$ (i.e., the conserved variables) is given by

$$\mathbf{U} = \Big(D,\, S^1,\, S^2,\, S^3,\, \tau\Big)^T\,, \tag{2.8}$$

and the fluxes are defined by

$$\mathbf{F}^i = \Big(Dv^i, S^1 v^i + p\delta^{1i}, S^2 v^i + p\delta^{2i}, S^3 v^i + p\delta^{3i}, S^i - Dv^i\Big)^T\,. \tag{2.9}$$

The system (2.7) of partial differential equations is closed with an equation of state $p = p(\rho, \varepsilon)$. Anile (1989) has shown that system (2.7) is hyperbolic for causal equations of state, i.e., for those where the local sound speed, c_s, defined by

$$hc_s^2 = \frac{\partial p}{\partial \rho} + (p/\rho^2)\frac{\partial p}{\partial \epsilon}, \tag{2.10}$$

satisfies $c_s < 1$.

The structure of the characteristic fields corresponding to the nonlinear system of conservation laws (2.7) has explicitly been derived in Donat *et al.* (1998) and is summarised in Appendix A.

In order to evolve system (2.7) numerically, one has to discretize the state vector $\mathbf{U}$ within computational cells. The temporal evolution of the state vector is determined by the flux balance across the zone interfaces of each cell and the contribution of source terms. Using a method of lines (see, e.g., LeVeque 1991), our discretization reads

$$\begin{aligned}\frac{d\mathbf{U}_{i,j,k}}{dt} &= -\frac{1}{\Delta x}\left(\tilde{\mathbf{F}}^x_{i+\frac{1}{2},j,k} - \tilde{\mathbf{F}}^x_{i-\frac{1}{2},j,k}\right) - \frac{1}{\Delta y}\left(\tilde{\mathbf{F}}^y_{i,j+\frac{1}{2},k} - \tilde{\mathbf{F}}^y_{i,j-\frac{1}{2},k}\right) \\ &\quad -\frac{1}{\Delta z}\left(\tilde{\mathbf{F}}^z_{i,j,k+\frac{1}{2}} - \tilde{\mathbf{F}}^z_{i,j,k-\frac{1}{2}}\right) + \mathbf{S}_{i,j,k} \equiv L(\mathbf{U})\,, \end{aligned} \tag{2.11}$$

where Latin subscripts i, j and k refer to the x, y and z coordinate direction, respectively. $\mathbf{U}_{ijk}$ and $\mathbf{S}_{i,j,k}$ are the mean values of the state and source vector (if non zero) in the corresponding three-dimensional cell, while $\tilde{\mathbf{F}}^x_{i\pm\frac{1}{2},j,k}$, $\tilde{\mathbf{F}}^y_{i,j\pm\frac{1}{2},k}$ and $\tilde{\mathbf{F}}^z_{i,j,k\pm\frac{1}{2}}$ are the numerical fluxes at the respective cell interface. Finally, $L(\mathbf{U})$ is a short hand notation of the spatial operator in our method.

At this stage, our system of conservation laws is a system of ordinary differential equations which can be integrated with a large number of algorithms. We have chosen a multi-step Runge–Kutta (RK) method developed by Shu & Osher (1988) which can provide second (RK2) and third (RK3) order in time. The explicit form of the algorithms is (subindexes (i, j, k) are ommited to clarify the notation):

1. Prediction step (common for both RK2 and RK3):

$$\mathbf{U}^{(1)} = \mathbf{U}^n + \Delta t L(\mathbf{U}^n) \tag{2.12}$$

2. Depending on the order do:

   - RK2:

$$\mathbf{U}^{n+1} = \frac{1}{\alpha}\left(\beta\mathbf{U}^n + \mathbf{U}^{(1)} + \Delta t L(\mathbf{U}^{(1)})\right), \tag{2.13}$$

   being $\alpha = 2$ and $\beta = 1$.

   - RK3:

$$\begin{aligned}\mathbf{U}^{(2)} &= \frac{1}{\alpha}\left(\beta\mathbf{U}^n + \mathbf{U}^{(1)} + \Delta t L(\mathbf{U}^{(1)})\right) \\ \mathbf{U}^{n+1} &= \frac{1}{\beta}\left(\beta\mathbf{U}^n + 2\mathbf{U}^{(2)} + 2\Delta t L(\mathbf{U}^{(2)})\right), \end{aligned} \tag{2.14}$$

(2.15)

   in this case, $\alpha = 4$ and $\beta = 3$.

2.3. The relativistic hydrodynamic code GENESIS

2.3.1. Code structure

The special relativistic multidimensional hydrodynamic code GENESIS described in detail in the following is a 3D extension of the 2D HRSC hydrodynamic code developed by some of the authors. The 2D code has been successfully used for the simulation of relativistic jets (Martí *et al.* 1994, 1995, 1997; Gómez *et al.* 1995, 1997). The main structural features of Martí *et al.*'s code has been kept, but there are important changes in the computational part. Besides the addition of the third spatial dimension, a large effort has been made to *minimise memory requirements* and to *optimize the performance* of the code as well as to enhance its *portability*.

Like its predecessor, GENESIS evolves the equations of RHD in conservation form using a finite volume approach in Cartesian coordinates. In accordance with the method of lines, we split the discretization process in two parts. First, we only discretize the differential equations in space, i.e. the problem remains continuous in time. This leads to a system of ordinary differential equations (ODEs) in time (2.11). The numerical fluxes between adjacent cells required for the time integration are obtained by solving the appropriate 1D Riemann problems along the coordinate directions (spatial sweeps). High-order spatial accuracy is achieved by applying a high-order interpolation procedure in space, while high-order accuracy in time is obtained by using high-order ODE solvers.

GENESIS integrates the 3D RHD equations on uniform grids in each spatial direction. In order to have a flexible code GENESIS is programmed to allow for different boundary conditions, spatial reconstruction algorithms, Riemann solvers, ODE solvers for the time integration and external forces. The user selects these options at the preprocessor level, which reduces the number of *if*–clauses inside the nested 3D loops to a minimum, and thereby maximizing the code's efficiency.

Making the selection at the preprocessing stage has allowed us to obtain a code, which is independent of a specific (shared memory) machine architecture. Hence, it runs on different types of machines and processors. Up to now, we have tested GENESIS on SGI platforms (INDY workstations, Power Challenge and Cray-Origin 2000 arrays), on HP machines (712 workstations and J280 computers), and on a

CRAY-JEDI multiprocessor system. As a next step we plan to port GENESIS on a CRAY-T3E massively parallel computer.

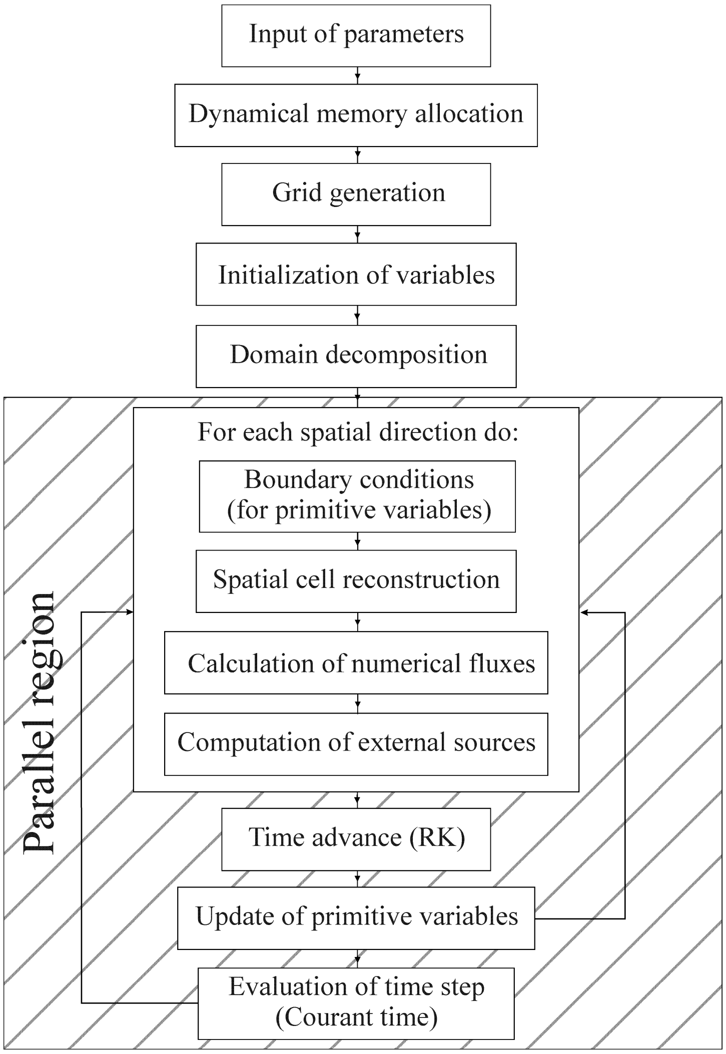

Fig. 2.1.— Flow diagram of GENESIS.

The flow diagram of GENESIS is shown in figure 2.1. Details of the major components of GENESIS are discussed in the following subsections.

2.3.2. Memory requirements

The current version of GENESIS, which is written in FORTRAN 90 has the capability of allocating memory *dynamically*, i.e. the number of computational cells can be chosen at run time. Reducing the RAM requirements of a 3D hydrodynamic code is obviously crucial. In GENESIS multidimensional variables are responsible for about 99% of the code's memory requirement. Thus, the number of these 3D arrays has to be kept at the absolute minimum possible. In its present version, GENESIS only requires three sets of five 3D arrays each, consisting of one set of conserved variables at the beginning of each time level ($\mathbf{U}^n$), another set of primitive variables and a third set of scratch variables ($\tilde{\mathbf{U}}$). The time integration scheme (eqs. 2.12–2.14) which results in the updated values of the conserved variables at the next time level ($\mathbf{U}^{n+1}$) then reads:

1. Prediction step (common for RK2 and RK3):

$$\tilde{\mathbf{U}} = \mathbf{U}^n + \Delta t L(\mathbf{U}^n)$$

2. Depending on the order of accuracy of the time integration scheme do:

RK2:

$$\begin{aligned} \tilde{\mathbf{U}} &= \tilde{\mathbf{U}} + \Delta t L(\tilde{\mathbf{U}}), \\ \mathbf{U}^{n+1} &= \frac{1}{\alpha}\left(\beta \mathbf{U}^n + \tilde{\mathbf{U}}\right), \end{aligned}$$

with $\alpha = 2$ and $\beta = 1$, or

RK3:

$$\begin{aligned} \tilde{\mathbf{U}} &= \tilde{\mathbf{U}} + \Delta t L(\tilde{\mathbf{U}}), \\ \tilde{\mathbf{U}} &= \frac{1}{\alpha}\left(\beta \mathbf{U}^n + \tilde{\mathbf{U}}\right), \\ \tilde{\mathbf{U}} &= \tilde{\mathbf{U}} + \Delta t L(\tilde{\mathbf{U}}), \\ \mathbf{U}^{n+1} &= \frac{1}{\beta}\left(\beta \mathbf{U}^n + 2\tilde{\mathbf{U}}\right), \end{aligned}$$

with $\alpha = 4$ and $\beta = 3$.

Quantities like entropy, internal energy, sound speed or Lorentz factor are implemented as FORTRAN scalars. Consequently, GENESIS needs about 1 Gbyte of RAM memory to handle a grid of $100 \times 100 \times 720$ (in double precision).

2.3.3. Domain decomposition

The technique of domain decomposition is used to optimize the parallelization of the code and to guarantee its performance in real applications, too. It is also the first step towards the development of a parallel version of GENESIS which runs efficiently on parallel computers with distributed memory.

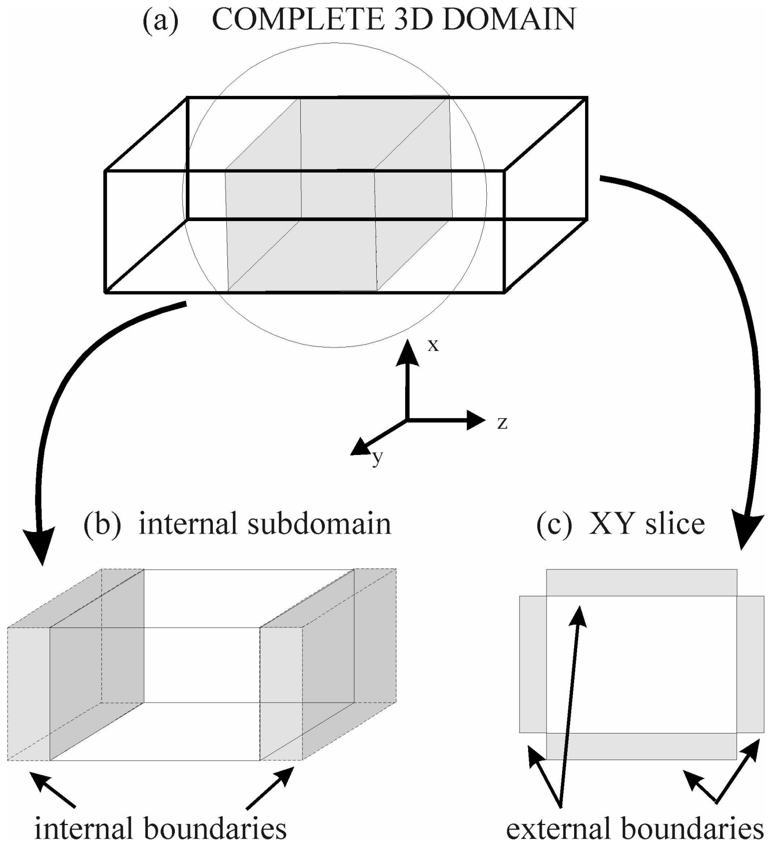

Fig. 2.2.— (a) Complete three dimensional computational domain, showing a typical subdomain (in grey). (b) Zoom of the previous subdomain including its internal boundaries. These regions overlap with contiguous subdomains. (c) Cut through the computational grid along the X-Y plane displaying the external boundaries.

The physical domain is split along *one* arbitrary spatial direction (z, in the present version) in a set of subdomains (i.e. *slices*, see Fig. 2.2a) of similar computational load. The subdomains are then distributed across processors. Numerical fluxes at subdomain boundaries are calculated by providing the appropriate internal and external boundary conditions (see Fig. 2.2b,c, respectively, and § 2.3.4).

2.3.4. Boundary conditions

The computational grid is extended in each coordinate in positive and negative direction by four so-called *ghost* zones, which provide a convenient way to implement different types of boundary conditions. These boundary conditions have to be provided in each spatial sweep for all primitive variables. In GENESIS several types of boundary conditions are available including reflecting, inflow, outflow, time-dependent and analytically prescribed boundary conditions.

Flow conditions at subdomain boundaries must be provided, too, in order to calculate numerical fluxes at subdomain interfaces. Hence, subdomains are also enlarged by four ghost zones in each coordinate direction. Note that these ghost zones do overlap with adjacent subdomains (see Fig. 2.2). The internal boundary conditions in these overlapping regions are defined by copying the corresponding values of the respective adjacent subdomain. For NS subdomains and $NX \times NY \times NZ$ computational zones the number of overlapping cells is $(4+4) \times (NS-1) \times NX \times NY$, i.e. the fraction of overlapping cells is $8 \times (NS-1)/NZ$. Hence, for $NS = 16$ and $NZ = 1000$ (typical of a jet simulation) the fraction of overlapping cells is about 12%.

2.3.5. Spatial reconstruction

In order to improve the spatial accuracy of the code, we interpolate the values of the pressure, the proper rest–mass density and the spatial components of the four–velocity (Wv^i) within computational cells. These reconstructed variables are afterwards used to compute the numerical fluxes. Because of the monotonicity of the reconstruction procedures (see below) used in GENESIS, the occurrence of unphysical (i.e., negative) values in the reconstructed profiles of pressure and density are avoided. In addition, reconstructing the spatial components of the four-velocity with monotonic schemes, also prevents the occurrence of unphysical values of the flow velocity, i.e., the flow velocity always remains smaller than the speed of light even in multidimensional calculations.

GENESIS provides, at the preprocessing level, four different types of reconstruction schemes: piecewise constant, linear using the minmod function of Van Leer (1979), parabolic using the piecewise parabolic method, PPM, of Colella & Woodward (1984; see also Martí & Müller 1996) or hyperbolic using the piecewise hyperbolic method, PHM, of Marquina (1994).

2.3.6. Source terms

Gravity, local radiative processes, etc., are coupled with hydrodynamics through terms on the right hand side of the RHD equations (i.e., via the source terms, $\mathbf{S}_{i,j,k}$, in Eq. (2.11)). GENESIS integrates such terms assuming piecewise constant profiles for the source functions.

2.3.7. Computation of the numerical fluxes

In this paper we use a variant of Marquina's flux formula (see Donat & Marquina 1996) which has already been shown to work properly in the simulation of relativistic jets in 2D (Martí *et al.* 1997).

The approach followed by Donat & Marquina (1996) relies on the extension of the entropy–satisfying scalar numerical flux of Shu & Osher (1989) to hyperbolic systems of conservation laws. Given the spectral decomposition of the RHD equations (see Appendix A), the implementation of Marquina's scheme is straightforward.

The original Marquina's algorithm computes the contribution to the numerical viscosity of each characteristic field in a different way depending on whether the corresponding eigenvalue (characteristic speed) does change its sign between the left and right states or whether it does not. However, instead of using the original algorithm, we only consider that part which corresponds to characteristic speeds changing their signs between the left and right states of every numerical interface. The modified algorithm has a larger numerical viscosity, but it is more stable and does not involve any *if*-clause. Hence, it can easily be vectorized.

In the 2D version used in Martí *et al.* (1997) the left eigenvectors of the Jacobians are calculated numerically by inverting the matrix of right eigenvectors. In GENESIS we use the analytical expressions for the left eigenvectors, which allow one to simplify the computation of the numerical viscosity terms.

The explicit expressions for the numerical fluxes ($\tilde{\mathbf{F}}^i$, $i \in x, y, z$ in Eq. 2.11) as a function of the local (reconstructed) primitive and conserved variables are given in Appendix B. Besides its influence on the efficiency of the code, the use of explicit expressions for the left eigenvectors also leads to analytical cancellations in the computation of the numerical viscosity causing a damping of the growth of round-off errors and an improvement of the overall accuracy of the code. Previous versions of GENESIS, in which numerical fluxes were calculated without the use of analytical expressions, suffered from a growth of round-off errors due to the large number of operations involved and due to the finite precision of floating point arithmetics. This growth of errors manifests itself in a gradual loss of symmetry in initially perfectly symmetric problems. Our experience shows that the analytical manipulation of the expressions of the numerical flux together with their appropriate symmetrization (i.e., using commutating formulas for the components of the velocity parallel to cell interfaces) allows one to achieve a perfect numerical symmetry (see §2.3.10 and §2.4.2).

2.3.8. Time advance and time step computation

Time integration is carried out by two different total variation diminishing RK methods developed in Shu & Osher (1988). The user can choose, at preprocessing level, between the RK2 and RK3 algorithm (see eqs. 2.12–2.14). Results of similar quality can be obtained either with the RK3 algorithm or with RK2 using smaller time steps. Nevertheless, for a given time step, the computational cost of RK3 is about a factor 1.5 larger than that of RK2.

As in any explicit hydrodynamic code, time steps are limited for stability reasons by the Courant-Friedrichs-Levy (CFL) condition, which is computed using the characteristic speeds. At the end of each time step the size of the new time step is determined as the minimum of the time steps of all subdomains. This requires a global operation across all subdomains. Experience has shown that acceptable CFL numbers lie in the interval $[0.1, 0.8]$. CFL numbers larger than 0.8 can lead to post shock oscillations.

2.3.9. Recovering primitive variables

The solution of the Riemann problem requires knowledge of the value of the pressure and its thermodynamic derivatives. Given the functional dependence between conserved and primitive variables (see eq. (2.2)), the recovering procedure can not be formulated in closed form. Instead a kind of iterative method must be used, which is very time consuming. Hence, usage of the recovering procedure should be reduced to the absolute minimum. Therefore, primitive variables are consistently updated from the mean values of the conserved variables after each Runge-Kutta step and their values are stored in a set of 3D arrays.

Our approach is the same as that of Martí, Ibáñez & Miralles (1991) and that of Martí *et al.* (1997). Its explicit form can be found in Appendix C (see also Martí & Müller 1996). The iterative recovering procedure is based on a second order accurate Newton-Raphson method to solve an implicit equation for the pressure.

In zones where the flow conditions change smoothly the typical number of iterations ranges from 1 to 3 when a relative accuracy of 10^{-10} is requested. There exist zones, however, inside shocks or near strong gradients, where the number of iterations required is larger depending on the strength of the shock or the steepness of the gradient. For example, in the shock reflection test in 3D, the shock zone needs about 4 to 8 iterations.

2.3.10. Some notes on code structure

We have taken special care in designing a numerical code that accurately preserves any symmetries present in the initial data. This is an important point for a code aimed to study, for example, the stability and long term evolution of initially axisymmetric jets.

There exist two potential sources of numerical *asymmetries* in our code, both of them are related to the fact that floating point arithmetics is not associative. One cause of asymmetries is due to the computation of numerical fluxes in spatial sweeps, which violates what we call henceforth *sweep–level symmetry* (SLS). In order to guarantee SLS the expressions by which the numerical fluxes are evaluated have been symmetrized (see §2.3.7).

A second source of (numerically caused) asymmetry arises specifically in 3D codes using directional splitting. It can only be avoided, if the code has a property which we call *sweep–coherence symmetry* (SCS). It refers to the symmetry of the integration algorithm with respect to the order in which the 1D-sweeps are performed. This symmetry property of the algorithm becomes crucial if an initially spherically symmetric state is considered. We found that its initial symmetry is lost unless special care is taken in the calculation of the Lorentz factor (in the numerical flux routine), which involves the summation of the squares of the three velocity components. To guarantee a perfect sweep–coherence symmetry of the algorithm the addition of the vector components has to be performed in a cyclic manner, i.e. in the X–sweep the components are summed up in x, y, z order, in the Y–sweep in y, z, x order, and finally in the Z–sweep in z, x, y order. Due to the stochastic nature of round-off errors, a violation of the sweep–coherence symmetry manifests itself only in the last few significant digits of the state variables, if the number of time steps is not too large (less than about 3000; see section §2.4.3).

Given that round–off errors grow sufficiently slow and that they do not interact with the truncation errors due to the finite difference scheme (which can render the scheme *unstable*), GENESIS does keep the symmetry of an initial state at an acceptable level. We have also tried to develop a version of GENESIS with a perfect 3D symmetry (limited by the Cartesian discretization). For this purpose, we applied the *extended partial precision* technique in the computation of expressions in which the associative property should be satisfied. The procedure was successful, but increased the total computational costs by more than 30%. All the results presented in the following have been obtained without making use of such a technique.

2.4. Code Testing

The capabilities of GENESIS to solve problems in special relativistic hydrodynamics are checked by means of three tests calculations that involve strong shocks and a wide range of flow Lorentz factors. In these test runs an ideal gas equation of state with an adiabatic exponent γ has been used. All results presented in this section have been obtained with the PPM reconstruction procedure and the relativistic Riemann solver based on Marquina's flux formula (see previous section for details).

2.4.1. Mildly Relativistic Riemann Problem (MRRP)

In the first test we consider the time evolution of an initial discontinuous state of a fluid at rest. The initial state is given by $\rho_L = 10$, $\epsilon_L = 2$, $v_L = 0$, $\gamma_L = 5/3$, $\rho_R = 1$, $\epsilon_R = 10^{-6}$, $v_R = 0$ and $\gamma_R = 5/3$, where the subscript L (R) denotes the state to the left (right) of the initial discontinuity. This test problem has been considered by several authors in the past (in 1D by Hawley, Smarr & Wilson 1984, Schneider *et al.* 1993, Martí & Müller 1996, Wen, Panaitescu & Laguna 1997; in 2D by Martí *et al.* 1997). It involves the formation of an intermediate state bounded by a shock wave propagating to the right and a transonic rarefaction propagating to the left. The fluid in the intermediate state moves at a mildly relativistic speed ($v = 0.72c$) to the right. Flow particles accumulate in a dense shell behind the shock wave compressing the fluid by a factor of 5 and heating it up to values of the internal energy much larger than the rest-mass energy. Hence, the fluid is extremely relativistic from a thermodynamical point of view, but only mildly relativistic dynamically.

To change this intrinsically one dimensional test problem into a multidimensional one we have rotated the initial discontinuity (normal to the x-axis) by an angle of 45^o around the y-axis, and then again by an angle of 45^o around the z-axis. Gas states L and R are placed within a cube of major diagonal equal to 1 that constitutes the 3D numerical grid.

The analytical solution to this test problem can be found in Martí & Müller (1994). Our analysis is restricted to the flow conditions along the major diagonal of the numerical grid, which is normal to the initial discontinuity. Figure 2.3 shows the solution along the major diagonal at time $t = 0.5$. The shock is captured in two to three zones in accordance with the capabilities of HRSC methods. The transonic rarefaction has a smooth profile across the sonic point located at $x = 0.5$, and exhibits sharp corners. The contact discontinuity is spread out over roughly three zones.

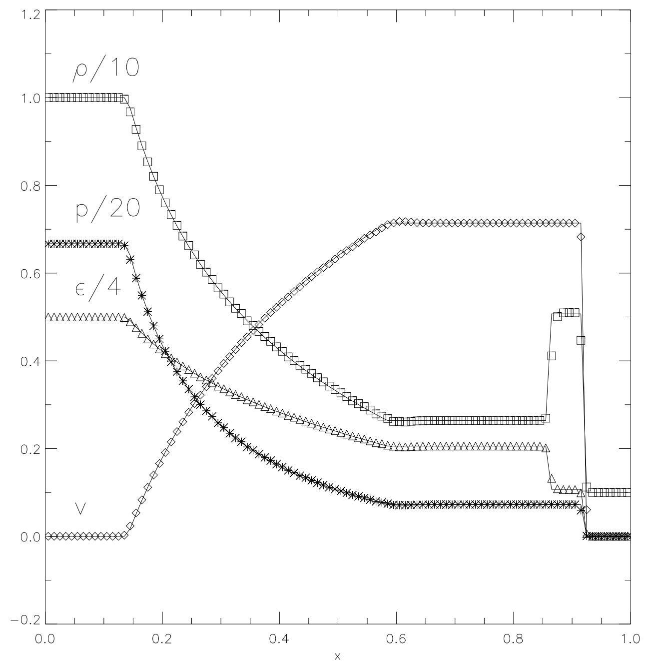

Fig. 2.3.— Numerical and exact solution of the mildly relativistic Riemann test problem (MRRP) described in the text after 0.5 time units. The computed one–dimensional distributions of proper rest–mass density, pressure, specific internal energy and flow velocity are shown, in normalized units, with discrete symbols. Continuous lines depict the corresponding exact solution. The simulation was performed on a grid of 100^3 zones. The CFL number was set equal to 0.6 and a second–order Runge-Kutta was used for time integration.

Cells	Pressure	Density	Velocity
40^3	8.0(2.0)E–2	1.1(0.3)E–1	0.9(0.4)E–2
60^3	5.2(0.4)E–2	9.8(0.8)E–2	1.1(0.3)E–2
80^3	4.5(0.2)E–2	9.2(0.5)E–2	1.1(0.1)E–2
100^3	3.7(0.4)E–2	7.0(0.9)E–2	7.0(2.0)E–3
150^3	2.5(0.2)E–2	4.8(0.7)E–2	5.0(2.0)E–3

Table 2.1: Absolute global errors (L_1 norm) of the primitive variables for the mildly relativistic Riemann test problem (MRRP) for different grids at $t = 0.5$. As the errors are dominated by those zones located inside the shock and as the grid resolution is still poor even on the finest grid, we have repeated every calculation four times varying t within an interval $t \pm \delta t$ (δt being of the order of one Courant time) and calculated the mean errors. In parentheses we give the standard root mean square deviation of the errors ($\sigma_{\text{n-1}}$).

The absolute global errors[1] of pressure, density and velocity are given in Table 2.1 for different grid resolutions at $t = 0.5$. Table 2.1 implies a convergence rate of slightly less than 1 when comparing the errors obtained on the coarsest (40^3) and the largest (150^3) grid. This behavior is expected for multidimensional problems involving discontinuities (see, e.g. LeVeque 1991).

2.4.2. Relativistic Planar Shock Reflection (RPSR)

This 1D test problem involves the propagation of a strong shock wave generated when two cold gases, moving at relativistic speeds in opposite directions, collide. The problem has been considered as a test for almost any new relativistic hydrodynamic code (Centrella & Wilson 1984; Hawley, Smarr & Wilson 1984; Martí & Müller 1994;

[1] in L_1 norm given by $\epsilon_{\text{abs}} = \sum_{i,j,k} \left| \mathbf{w}^n_{i,j,k} - \mathbf{w}(\mathbf{x}_{i,j,k}, t_n) \right| \Delta x_i \Delta y_j \Delta z_k$, where $\mathbf{w}^n_{i,j,k}$ and $\mathbf{w}(\mathbf{x}_{i,j,k}, t_n)$ are the numerical and exact solution, respectively

Eulderink & Mellema 1994; Falle & Komissarov 1996).

After the collision of the two gases, two shock waves are created in the plane of symmetry of the physical domain propagating in opposite directions. The inflowing gas is heated in the shocks and comes to a rest. The exact solution of this Riemann problem was obtained by Blandford & McKee (1976).

The initial data are $\rho_L = 1$, $\epsilon_L = 2.29\,10^{-5}$, $v_L = v_i$, $\rho_R = 1$, $\epsilon_R = 2.29\,10^{-5}$ and $v_R = -v_i$, where v_i is the inflow velocity of the colliding gas.

Figure 2.4 shows the numerical solution at $t = 2.0$ on the left half of a grid having a total of 401 zones. The results obtained in the right half of the grid are strictly symmetric with respect to the collision point ($x = 0$), i.e., the sweep–level symmetry (SLS; see section §2.3.10) is exactly fulfilled. Near $x = 0$, the numerical solution shows small errors (of the same order as the mean error in the post-shock state, 0.3%) which are due to the *wall heating* phenomenon (Noh 1987) characterized by an overshooting of the internal specific energy and an undershooting of the proper rest-mass density.

In Table 2.2 we give the global absolute errors (L_1 norm) of the primitive variables for different grids at $t = 2.0$ and for an inflow velocity $v_i = 0.999c$. We find a convergence rate about equal to one (see columns 5-7) for all variables.

We can use this test problem to check the robustness of GENESIS in the ultra-relativistic regime. To simplify notation, we define the quantity $\nu = 1 - v_i$, which tends to zero when v_i tends to one. Table 2.3 contains the relative global errors of the primitive variables at $t = 2.0$ for a set of calculations performed on a grid of 401 zones, where we have varied ν from 10^{-1} to 10^{-11}. The latter value corresponds to a Lorentz factor $W = 2.24 \times 10^5$. The relative error of the primitive variables shows a weak dependence on the inflow velocity. It never exceeds 3.5% and for $\nu \geq 10^{-9}$ it is smaller than 1%.

The PPM parameters (see Colella & Woodward 1984) have been tuned to minimize the number of zones within the shock without introducing unacceptable numerical post–shock oscillations. Fig. 2.5 demonstrates that there are no numerical post-shock oscillations for $\nu \leq 10^{-5}$ when the shock is captured by 2 to 3 zones.

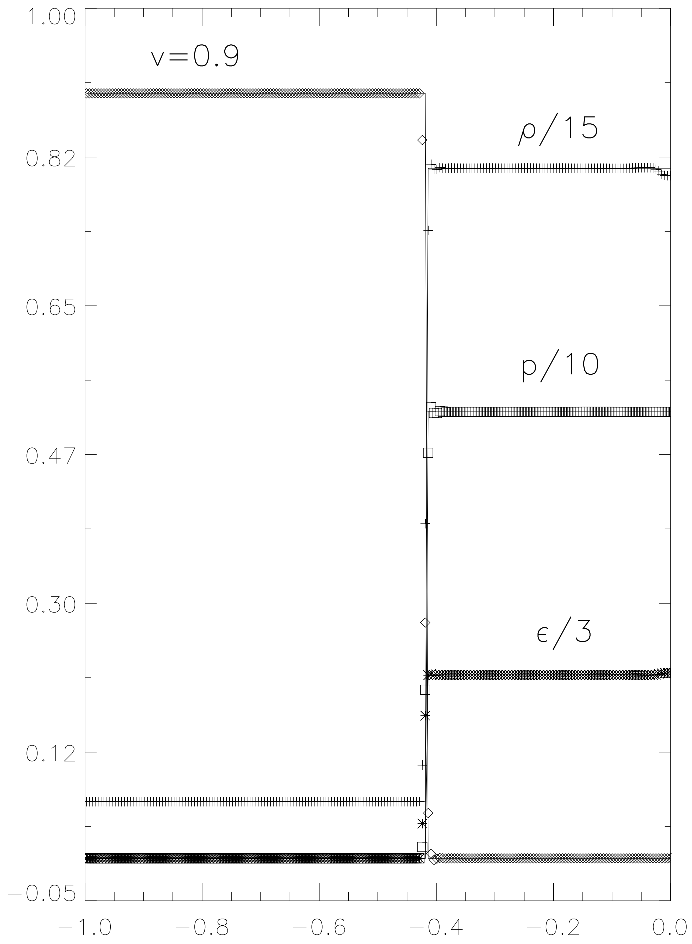

Fig. 2.4.— Numerical and exact solution of the relativistic planar shock reflection problem (RPSR) described in the text after 2.0 time units. The computed distributions of proper rest–mass density, pressure, specific internal energy and flow velocity are shown, in normalized units, with discrete symbols, for an inflow velocity of the colliding gases equal $v_i = 0.9$. Continuous lines depict the corresponding exact solution. The simulation was performed on a grid of 401 zones spanning the interval $[-1, 1]$ with both gases colliding in the middle of the grid at $x = 0$. Only the left half of the grid is shown. The CFL number was set equal to 0.3 and a second–order Runge–Kutta was used for time integration.

2.4.3. Relativistic Spherical Shock Reflection (RSSR)

The initial setup consists of a spherical inflow at speed v_i (which might be ultrarelativistic) colliding at the centre of symmetry of a sphere of radius unity. For a hydrodynamic code in Cartesian coordinates this is a 3D test problem, which allows one to evaluate the directional splitting technique as well as the symmetry

Cells	Pressure	Density	Velocity	r_ρ	r_p	r_v
101	19.3(0.3)E+0	290.8(0.4)E–2	2.4(0.1)E–2			
201	10.8(0.2)E+0	147.2(0.7)E–2	10.1(0.4)E–3	0.99	0.84	1.26
401	49.2(0.7)E–1	85.0(1.0)E–2	92.8(0.8)E–4	0.80	1.14	0.14
801	25.2(0.2)E–1	37.3(0.1)E–2	3.4(0.1)E–3	1.19	0.97	1.44
1601	13.8(0.1)E–1	187.4(0.7)E–3	17.3(0.4)E–4	0.99	0.87	0.98

Table 2.2: Absolute global errors (L_1 norm) of the primitive variables (columns 2-4) and the corresponding convergence rates (columns 5-7) for the relativistic planar shock reflection test problem (RPSR) for different grids at $t = 2.0$. The test runs have been performed with a Courant number equal to 0.1 and the third order accurate Runge-Kutta time integration method (RK3). In parenthesis we give the standard root mean square deviation of the errors (see also Table 2.1).

properties of the algorithm. Figure 2.6 shows the numerical results for $v_i = 0.9c$ on a grid of 101^3 zones at $t = 2.0$. The shock capturing properties of GENESIS, which we have already demonstrated in 1D, are retained in this genuine multidimensional case. Two or three zones are required to handle the shock wave. The pressure and proper rest–mass density have global relative errors of about 12% and 8% respectively.

Ultrarelativistic flows have been explored by increasing the inflow Lorentz factor. Table 2.4 gives the growth of the relative global errors[2] on a fixed grid size of 81^3 zones for v_i in the range $0.9c$ to $0.999999c$ (the latter inflow velocity corresponding to a Lorentz factor $W \approx 707$). The relative global errors are acceptable (considering the inherent difficulty of the test and the resolution of the experiments) and do not grow dramatically with the Lorentz factor. The observed growth can be explained by the fact that the errors are dominated by the shock region and that the shock strength increases with the Lorentz factor.

[2] $\epsilon_{\rm rel} = \epsilon_{\rm abs} / \sum_{i,j,k} |\mathbf{w}(\mathbf{x}_{i,j,k}, t_n)| \; \Delta x_j \Delta y_j \Delta z_k$

ν	Pressure	Density	Velocity
10^{-1}	90.7(0.5)E–4	96.6(0.5)E–4	80.3(0.5)E–4
10^{-3}	58.0(0.8)E–4	72.0(0.8)E–4	12.6(0.1)E–3
10^{-5}	100.3(0.5)E–5	79.3(0.5)E–4	72.0(0.8)E–4
10^{-7}	61.0(0.8)E–4	93.0(0.1)E–4	85.6(0.1)E–4
10^{-9}	65.2(0.1)E–4	103.0(0.1)E–4	81.3(0.5)E–4
10^{-11}	141.0(0.1)E–5	340.1(0.1)E–4	325.7(0.5)E–5

Table 2.3: Relative global errors (L_1 norm) of the primitive variables for the planar shock reflection test problem (RPSR) on a grid of 401 zones at $t = 2.0$. The quantity ν is defined as $\nu = 1 - v_i$. The test runs have been performed with a Courant number equal to 0.1 and the third order accurate Runge-Kutta time integration method (RK3). In parenthesis we give the standard root mean square deviation of the errors (see also Table 2.1).

The CFL factors used in the last two tests of this series are unusually small (0.019 and 0.005) which is due to the strength of the shock and the relatively small grid resolution (compared with the 1D case). It is noticeable that for $v_i = 0.999999c$ the errors are considerably larger (last entry in Table 2.4). This has two reasons. Firstly, the global relative errors decrease with time in the RSSR test problem. Secondly, we could not continue the run with $v_i = 0.999999c$ beyond 1.5 time units, because interaction with the grid boundaries became severe causing the code to crash. Hence, $v_i = 0.999999c$ must be considered as the maximum inflow velocity in the RSSR test problem, which the present code can handle properly (for the resolution used). The symmetry properties of the RSSR solution are very well maintained by GENESIS, even though the number of timesteps was very large ($>$ 30000) in the last two test runs.

The absolute global errors (L_1 norm) and the convergence rates of the primitive variables at $t = 2.0$ are displayed in Table 2.5. Obviously, the errors are much larger in the 3D test than in the corresponding 1D one. This can be explained considering

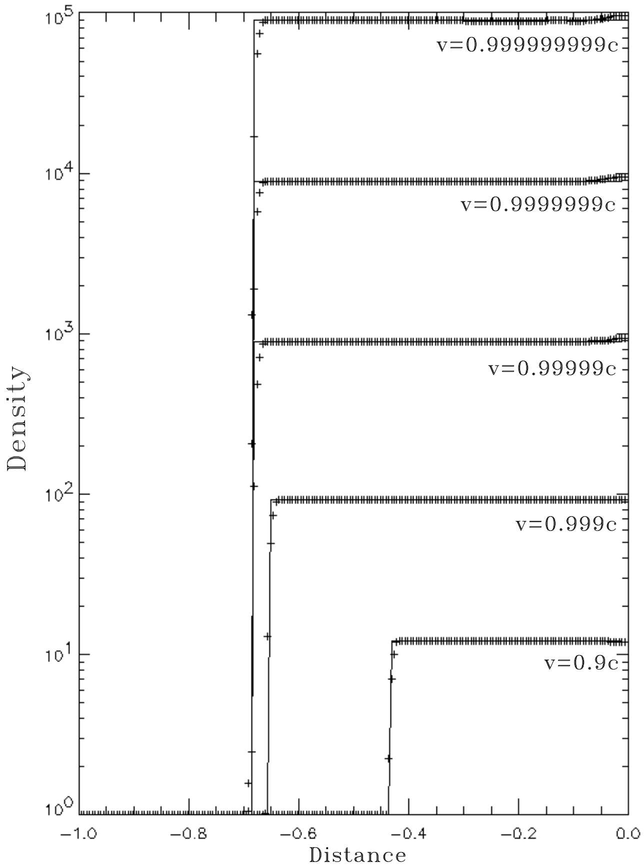

Fig. 2.5.— Density jump (in logarithmic scale) for different inflow velocities in the relativistic planar shock reflection problem (RPSR), over an equally spaced grid of 401 zones at $t = 2.0$. As in the previous Figure, only the left half of the grid is shown. Solid lines represent the exact solution while symbols refer to numerical values. A third–order Runge–Kutta was used for time integration.

that (i) the grids are coarser than in 1D, and that (ii) the jumps in pressure and density across the shock are nearly a factor of 30 larger in the 3D test than in the planar case.

The preservation of the sweep-level symmetry (SLS; see section §2.3.10) is reflected in the symmetry of the one dimensional profiles in Fig. 2.6. Moreover, a comparison of the profiles in X and Y direction in Fig. 2.6 shows the capability of

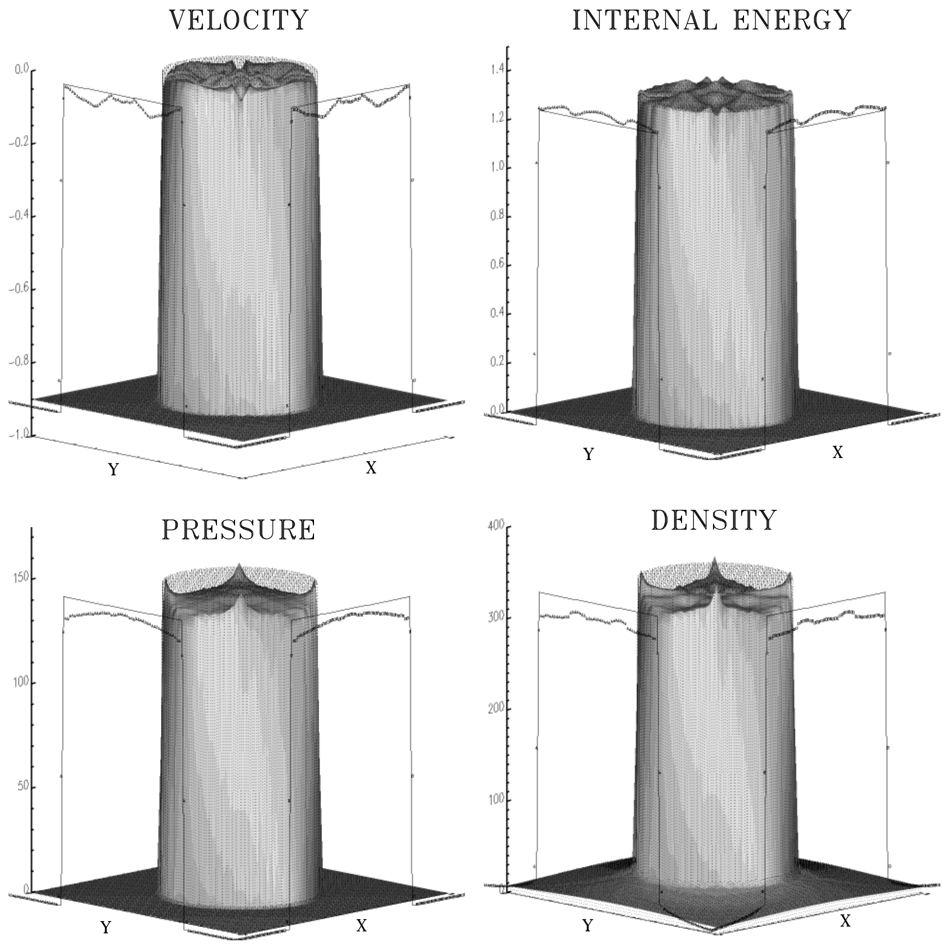

Fig. 2.6.— Intensity plots of proper rest–mass density, pressure, specific internal energy and flow velocity over the plane XY at $z = 0$ in the relativistic spherical shock reflection test problem (RSSR) described in the text, after 2.0 time units. Shaded surfaces represent the numerical results while dotted surfaces are the exact solution. One dimensional plots along X and Y axes are projected on the front sides of the pictures. Symbols inside the one dimensional plots are numerical values; solid lines represent the exact solution on the same axis. The test was ran using a CFL equal to 0.2 and a third–order Runge–Kutta for time integration.

ν	Pressure (%)	Density (%)	Velocity (%)
10^{-1}	15.8	10.5	0.82
10^{-3}	19.9	22.1	3.07
10^{-5}	22.1	27.8	3.89
$10^{-6\,(a)}$	32.2	39.1	1.91

Table 2.4: Growth of relative global errors of the primitive variables for the relativistic spherical shock reflection test problem (RSSR) for different inflow velocities at $t = 2.0$. The four test runs have been performed with RK3 and Courant numbers 0.1, 0.1, 0.019 and 0.005, respectively. The quantity ν has the same meaning as in Table 2.3.

[a] The run time for this test is 1.5.

Cells	Pressure	Density	Velocity	r_ρ	r_p	r_v
41^3	11.8(0.2)E+0	30.3(0.4)E+0	80.0(3.0)E–3			
61^3	76.5(0.7)E–1	20.1(0.2)E+0	55.8(0.6)E–3	1.09	1.03	0.91
81^3	57.5(0.8)E–1	15.5(0.2)E+0	41.0(0.8)E–3	1.01	0.92	1.09
101^3	45.2(0.8)E–1	12.5(0.1)E+0	32.4(0.5)E–3	0.99	0.97	1.07

Table 2.5: Absolute global errors (L_1 norm) and convergence rates of the primitive variables for the relativistic spherical shock reflection test problem (RSSR) for different grids at $t = 2.0$. The test runs have been performed with a Courant number equal to 0.1 and the third order accurate Runge-Kutta time integration method (RK3). In parenthesis we give the standard root mean square deviation of the errors (see also Table 2.1).

the code to maintain the sweep-coherence symmetry (SCS), too.

2.5. Code Performance

We have parallelized GENESIS in order to run on multiprocessor computers with shared memory. Apart from the initial setup of variables, the grid generation and the output, the rest of the program is organized in a 4-level nested loop. The outermost loop runs from one to the total number of sub-domains, assigning one sub-domain to each processor. This procedure allows an almost complete parallelization of the code employing the corresponding parallelization directives (see Fig. 2.1).

The MRRP and RSSR tests have been run for different grids on a SGI Cray-Origin 2000 computer. Tables 2.6 and 2.7 show the total execution time for every run as a function of the number of CPUs used. We also give the *speed up* factor, defined as the CPU ratio between a one processor run and one using several processors in parallel. This factor is a measure of the degree of parallelization of the code and should ideally be equal to the number of CPUs used. The tables also contain the execution time per cell and time iteration (TCI). The TCI for a given number of processors is nearly independent of the number of computational cells, and can be used as a time unit to estimate the total execution time needed in a particular simulation.

According to the data shown in Tables 2.6 and 2.7 the TCI is about $7.6\ 10^{-5}$, $2.1\ 10^{-5}$ and $1.3\ 10^{-5}$ seconds for 1, 4 and 8 processors, respectively. A significant drop of the performance is noticeable for a grid of 64^3 zones due to the phenomenon of *cache trashing*, because in this case the dimensions of the 3D matrices are multiples of the size of cache lines. Hence, different 3D matrices are mapped into the same set of cache lines, and every time the program needs to reference a new 3D matrix all cache lines are updated.

Concerning the speed up factor, it is noticeable from Tables 2.6 and 2.7 that it increases with the number of grid points, because the 3D nested loops consume a larger percentage of the total CPU time when the number of grid zones is larger. The maximum speed up factors are 3.7 and 6.5 for 4 and 8 CPUs, respectively. We also notice a *super linear* behavior, for the largest grid, for the MRRP test problem. As typical 3D simulations are performed with zone numbers larger than the ones used in the test runs, we expect to reach even larger speed up factors in these applications.

# cells	# CPUs	Time	Speed up	# iter	TCI	Mflops
44^3	1	3.91E2		86	5.34E–5	64.73/——
	4	1.13E2	3.48		1.54E–5	59.03/236.11
	8	6.02E1	6.50		8.22E–6	58.58/468.65
64^3	1	3.85E3		118	1.24E–4	30.05/——
	4	1.84E3	2.09		5.94E–5	16.25/65.00
	8	1.49E3	2.59		4.81E–5	10.46/83.70
84^3	1	5.56E3		150	6.26E–5	62.04/——
	4	1.56E3	3.57		1.75E–5	56.77/227.08
	8	8.46E2	6.58		9.52E–6	53.99/431.88
104^3	1	1.31E4		183	6.35E–5	62.72/——
	4	3.66E3	3.57		1.78E–5	57.02/228.08
	8	2.36E3	5.54		1.15E–5	45.47/363.75
154^3	1	8.94E4		265	9.23E–5	45.12/——
	4	1.84E4	4.87		1.90E–5	54.92/219.68
	8	1.15E4	7.80		1.18E–5	44.74/357.91
	16	7.39E3	12.09		7.64E–6	35.90/574.41

Table 2.6: Performance of GENESIS for the mildly relativistic Riemann test problem (MRRP) on different grids. The test runs are stopped at $t = 0.5$, and are performed with a Courant number equal to 0.8 and the second order accurate Runge-Kutta time integration (RK2) method. Times are measured in seconds on a SGI Cray–Origin 2000. The last column displays the number of Mflops per processor and the total number of Mflops. One notices that the efficiency per processor in parallel mode (Speed Up/CPUs) multiplied by the number of Mflops in sequential mode is equal to the number of Mflops in parallel mode. Megaflops are calculated using SGI's *Perfex Tool*.

The number of Mflops (millions of floating point operations per second) achieved by the code is about 60 on one processor (R10000) of a SGI Cray–Origin 2000 computer. The theoretical peak speed of such a processor is 400 Mflops. For com-

# cells	# CPUs	Time	Speed up	# iter	TCI	Mflops
45^3	1	8.73E2		114	8.40E–5	62.53/——
	4	2.53E2	3.45		2.44E–5	56.88/225.67
	8	1.51E2	5.78		1.45E–5	50.15/401.21
65^3	1	4.94E3		198	9.08E–5	62.27/——
	4	1.50E3	3.29		2.76E–5	52.88/211.54
	8	1.10E3	4.49		2.02E–5	37.54/300.28
85^3	1	2.15E4		369	9.49E–5	61.76/——
	4	6.13E3	3.51		2.71E–5	55.40/221.60
	8	3.41E3	6.30		1.51E–5	51.35/410.79
105^3	1	9.92E4		890	9.63E–5	62.09/——
	4	2.78E4	3.57		2.70E–5	56.41/225.65
	8	1.57E3	6.31		1.53E–5	48.97/391.79

Table 2.7: Performance of GENESIS for the relativistic spherical shock reflection test problem (RSSR) on different grids. The test runs are stopped at $t = 2.0$, and are performed with a Courant number varying from 0.8 (45^3 grid) to 0.2 (105^3 grid). The third order accurate Runge-Kutta time integration (RK3) method has been used. Times are measured in seconds on a SGI Cray–Origin 2000. The last column displays the number of Mflops per processor and the total number of Mflops. One notices that the efficiency per processor in parallel mode (Speed Up/CPUs) multiplied by the number of Mflops in sequential mode is equal to the number of Mflops in parallel mode. Megaflops are calculated using SGI's *Perfex Tool*.

parison, Pen (1998) reports a performance of 48 Mflops for his 3D adaptive moving mesh classical hydrodynamic code using a SGI Power Challenge machine with R8000 processors (300 Mflops theoretical peak speed).

Finally, we compare the performance of GENESIS achieved on the PA8000 processor of Hewlett Packard with that obtained on the R10000 processor of Silicon Graphics. For the comparison we used a HP J280 workstation equipped with a

PA8000 processor with a 180 MHz clock and a cache memory of 512 Kbytes and a SGI Cray-Origin 2000 equipped with a R10000 processor with a 195 MHz clock and 4 Mbytes of cache memory. The test problem selected for the comparison was the relativistic spherical shock reflection test (RSSR) with an inflow velocity of $0.9c$. Test runs were done with four different grids. The resulting execution times per zone and time step (TCI) are given in Table 2.8.

# cells	Machine	Time	# iter	TCI
45^3	SGI	8.73E2	114	8.40E-5
	HP	2.11E3	101	2.29E-4
65^3	SGI	4.94E3	198	9.08E-5
	HP	1.33E4	220	2.20E-4
85^3	SGI	2.15E4	369	9.49E-5
	HP	4.76E4	357	2.17E-4
105^3	SGI	9.92E4	890	9.63E-5
	HP	2.16E5	879	2.12E-4

Table 2.8: Performance of GENESIS for the relativistic spherical shock reflection test (RSSR) on different grids and machines.

We find that $\mathrm{TCI}_{HP} \approx 2\times \mathrm{TCI}_{SGI}$. From Table 2.8 we can infer a general trend. The TCIs obtained on both machines tend to become similar when the number of zones increases. Furthermore, the TCI for the HP machine is nearly independent of the number of zones, while the TCI for the SGI machine increases with that number. This behavior may result from the fact that the problem size always leads to an overflow of the cache memory on the HP workstation, while this does not generally happen for the larger cache memory of the SGI machine.

2.6. Conclusions

We have described the main features of a novel three dimensional, high–resolution special relativistic hydrodynamic code GENESIS based on relativistic Riemann solvers. We have discussed several test problems involving strong shocks in three dimensions which GENESIS has passed successfully. The performance of GENESIS on single and multiprocessor machines (HP J280 and SGI Cray–Origin 2000) has been investigated. Typical simulations (in double precission) with up to $7\,10^6$ computational cells can be performed with 1Gbyte of RAM memory with a performance of $\approx 7\,10^{-5}$ s of CPU time per zone and time step (on a SCI Cray-Origin 2000 with a R10000 processor). Currently we are working on a version of GENESIS suited for massively parallel computers with distributed memory (like, e.g., Cray T3E).

GENESIS has been designed to handle highly relativistic flows. Hence, it is well suited for three dimensional simulations of relativistic jets. First results will be presented in a separate paper (Aloy et al. 1999b). Further applications envisaged are the simulation of relativistic outflows from merging compact objects (see, e.g., Ruffert et.al. 1997), from hypernovae (Paczynski 1998), or collapsars (MacFadyen & Woosley 1998). In all these models ultra-relativistic outflow is thought to occur and to play a crucial role in the generation of gamma-ray bursts.