GrAN
Grup de recerca d'anàlisi numèrica

Numerical Experiments with Point Value Multiresolution for 2D Compressible Flows

Guillaume Chiavassa and Rosa Donat

October 28, 1999

GrAN report 99-02

VNIVERSITAT DE VALÈNCIA
Departament de Matemàtica Aplicada
Universitat de València
46100 Burjassot, València (SPAIN)

Numerical Experiments with Point Value Multiresolution for 2D Compressible Flows [1]

Guillaume Chiavassa and Rosa Donat [2]

October 28, 1999

Abstract: The numerical simulation of physical problems modeled by systems of conservation laws is difficult due to the presence of discontinuities in the solution. High order shock capturing schemes combine sharp numerical profiles at discontinuities with a highly accurate approximation in smooth regions, but usually their computational cost is quite large. Following the idea of A. Harten [14, 4], we present in this paper a method to reduce the execution time of such simulations. It is based on a point value multiresolution transform that is used to detect regions with singularities. In these regions, an expensive high resolution shock capturing scheme (ENO or Marquina of order three) is applied to compute the numerical flux at cell interfaces. In smooth regions a cheap polynomial interpolation is used to deduce the value of the numerical divergence from values previously obtained on lower resolution scales. This method is applied to solve the two-dimensional compressible Euler equations for two classical configurations. The results are analyzed in terms of quality and efficiency.

[1] This work has been supported in part by TMR Research Network Contract FMRX-CT98-0184 and by Spanish DGICYT PB97-1402.

[2] Departament de Matemàtica Aplicada. Universitat de València. València. (Spain). e-mail: `guillaum@canada.matapl.uv.es, donat@uv.es`

1 Introduction

The computation of solutions to hyperbolic systems of conservation laws has been a very active field of research for the last 20 to 30 years; as a result, a variety of methods are now available that are able to compute accurate numerical approximations to the physically relevant solution. The latest addition to the pool of numerical methods for hyperbolic conservation laws are the modern High Resolution Shock Capturing (HRSC) schemes. These schemes succeed in computing highly accurate numerical solutions, typically second or third order in smooth regions, while maintaining sharp, oscillation free, numerical profiles at discontinuities.

State of the art shock capturing schemes usually perform a 'delicate art-craft' on the computation of the numerical flux function. A typical computation involves at least one eigenvalue-eigenvector decomposition of the Jacobian matrix of the system, and the approximation of the values of the numerical solution at both sides of each cell interface, obtained via some appropriately chosen approximating functions. The numerical result is very often spectacular in terms of resolution power, but the computational effort tends to be also quite spectacular.

Without doubt, the computational speed of the latest personal computers and workstations have made possible that an increasing number of researchers become interested in HRSC methods and, as a result, HRSC methods are now being tested in a variety of scenarios (e.g. [6, 10, 18] and references therein) that involve hyperbolic systems of conservation laws.

The memory capabilities in personal computers and workstations also allow the users to employ very fine meshes in two dimensional simulations. When these are uniform, the implementation of many of these shock capturing schemes is quite straightforward. In this framework, in which the basic code structure is relatively simple, the computational time becomes the main drawback in the numerical computation. As an example we notice that a typical computation of a 2D jet configuration in [18] is 10-50 days in an HP710 or 1 to 5 days in an Origin 2000 with 64 processors.

It is well known, however, that the heavy-duty flux computations are only needed because non-smooth structures may develop spontaneously in the solution of a hyperbolic system of conservation laws and evolve in time.

This basic observation has lead researchers to develop a number of techniques that aim at reducing the computational effort. Among these techniques we find traditional hybridization, shock tracking and adaptive mesh refinement (AMR).

Shock tracking and AMR techniques (often combined with one another) are very effective at obtaining high resolution numerical approximations, but the computational effort is transferred to the programming and the data structure.

Starting with the pioneering work of Ami Harten [14], another set of techniques aiming to reduce the computational effort associated to state of the art HRSC methods has entered the scene.

The starting point is the observation that the information contained in a multi-scale decomposition of a numerical approximation can be used to determine whether the solution is locally well resolved. In smoothness regions, the costly exact value of an HRSC numerical flux can be replaced by an approximate value obtained by an inexpensive interpolation from a coarser grid.

Different applications of this basic principle have been explored by several authors [14, 4, 20, 5, 1, 2, 16]. In all cases, a cell-average multi-scale analysis (see [14] for definitions and details) is used, since in finite-volume schemes the numerical values are naturally considered as approximations to the cell-averages of the true solution.

In this paper we present an alternative based on a point-value multiresolution analysis of the numerical data. The motivation comes from ENO schemes, a class of HRSC methods that compute directly numerical approximations to the point values of the solution, instead of its cell averages. In a multi-scale decomposition of a discrete set within the point-value framework, the scale coefficients are just interpolation errors and can be directly used as 'regularity sensors'. When the grid is uniform and the interpolation technique is constructed using a tensor product approach, it is very simple to analyze the smoothness information contained in the interpolation errors.

The end result is a transparent algorithm that can be implemented fairly easily and that can be incorporated into existing hydro-dynamical codes in a straightforward manner. In a way, it works as an AMR procedure (see section 4), but in reality it is far from being an AMR technique. The different grids are being used to analyze the smoothness of the numerical solution, and to compute the numerical divergence *on the finest grid* in a multi-level fashion. The finest grid determines all the parameters in the computation (like, for example the CFL number), and this avoids some of the problems related to the use of AMR in Computational Fluid Dynamics (CFD) codes.

Notice that the highest resolution grid needs to be always available in order to compute the values of the numerical divergence with the HRSC scheme when they are needed. This may be seen as a disadvantage, and it is certainly so in some cases. However, using the values of the numerical solution in the direct computation has some nice features. First, it avoids the use of complicated data structures, which is very useful when one is trying to incorporate the algorithm into an existing code. Second, the availability of the numerical solution on the finest grid guarantees that the 'delicate art-craft' involved in the direct evaluation of the numerical fluxes (via a sophisticated HRSC scheme) is performed adequately.

In most 2D (and some 3D) computations, memory requirements are not a problem, but the computational effort involved in the most sophisticated HRSC methods leads to large running times. Our point of view is that multilevel techniques of the type described in this paper can benefit users of HRSC schemes by allowing them to obtain numerical solutions on very fine grids at a cost which is very similar to that of their original scheme on a much coarser mesh.

2 General framework: The 2D setting

Let us assume that we are given a two dimensional system of hyperbolic conservation laws

$$\partial_t \vec{U} + \vec{f}(\vec{U})_x + \vec{g}(\vec{U})_y = \vec{0}, \tag{1}$$

where $\vec{U}$ is the vector of conserved quantities.
We follow the notation of [19], and consider a Cartesian grid $\mathcal{G}^0 = \{(x_i = i\delta x, y_j = j\delta y), \quad i = 0,..,Nx \ j = 0,..,Ny\}$ and a conservative semi-discrete scheme:

$$\frac{d\vec{U}_{ij}}{dt} + \mathcal{D}(\vec{U})_{ij} = 0. \tag{2}$$

The computation of the numerical divergence:

$$\mathcal{D}(\vec{U})_{ij} = \frac{\vec{F}_{i+1/2,j} - \vec{F}_{i-1/2,j}}{\delta x} + \frac{\vec{G}_{i,j+1/2} - \vec{G}_{i,j-1/2}}{\delta y}, \tag{3}$$

is the heart of the method. One typically has $\vec{F}_{i+1/2,j} = \vec{F}(\vec{U}_{i-k,j}, .., \vec{U}_{i+m,j})$,
$\vec{G}_{i,j+1/2} = \vec{G}(\vec{U}_{i,j-k}, .., \vec{U}_{i,j+m})$, where $\vec{F}(\vec{w}_1, .., \vec{w}_{k+m})$ and $\vec{G}(\vec{w}_1, .., \vec{w}_{k+m})$ are the numerical flux

functions: the trademark of the scheme and where most of the computational effort in the numerical scheme goes.
In our numerical experiments we use two different schemes, with significantly different flux formulas in terms of computational effort,

- the classical ENO method of order 3 [19], using Roe solver and the piecewise polynomial ENO reconstruction procedure on both sides of each cell interface,
- the Marquina's scheme of order 3 [7], which uses a more complicated flux splitting formula and a piecewise hyperbolic reconstruction procedure [17]. This scheme is more expensive with respect to cpu time but it avoids (or reduces) certain numerical pathologies associated to the Roe solver [7, 12].

The time evolution is performed using a third order Runge-Kutta method as proposed in [19], thus at each time step $t_n = n\delta t$ we obtain approximations to the values of $\vec{U}$ at the grid points : $\vec{U}_{ij}^n \approx \vec{U}(x_i, y_j, t_n)$.

2.1 Interpolatory multiresolution

Multi-scale decompositions within the point-value framework were initially introduced by Harten [13] (and also independently developed by Sweldens [21]), and have been extensively analyzed in a series of papers [15, 3]. Here we only present a brief summary to clarify the notation in the remainder of the paper.

One first defines a set of nested grids $\{\mathcal{G}^l,\ l = 1, .., L\}$ by

$$(x_i, y_j)\ \in\ \mathcal{G}^l \Longleftrightarrow (x_{2^l i}, y_{2^l j})\ \in\ \mathcal{G}^0 \tag{4}$$

The values of a function v on $\mathcal{G}^0$ (which is considered the finest resolution level), $(v_{ij}^0)_{i,j}$, are the input data. Due to the embedding of the grids, the representation of the function on the coarser grid $\mathcal{G}^l$, its point values on $\mathcal{G}^l$, is:

$$v_{ij}^l = v_{2^l i\ 2^l j}^0 \quad i = 0, .., Nx/2^l \quad j = 0, .., Ny/2^l \tag{5}$$

To recover the representation of v on $\mathcal{G}^{l-1}$ from the representation on $\mathcal{G}^l$ (the next coarser grid), the following procedure is used :

- a set of approximated values is first computed :

  $$\begin{array}{rcl} \tilde{v}_{ij}^{l-1} & = & v_{i/2\ j/2}^l \quad \text{if } (x_i, y_j) \in\ \mathcal{G}^l \\ \tilde{v}_{ij}^{l-1} & = & I[(x_i, y_j); v^l] \quad \text{if } (x_i, y_j) \in \mathcal{G}^{l-1} \backslash\ \mathcal{G}^l \end{array} \tag{6}$$

  where $I(.;.)$ denotes an r^{th} order polynomial interpolation.
- The difference between the exact values (5), v_{ij}^{l-1}, and $\tilde{v}_{ij}^{l-1}$ is then represented by the details, or wavelet coefficients :

  $$d_{ij}^l = v_{ij}^{l-1} - \tilde{v}_{ij}^{l-1} \quad (x_i, y_j) \in \mathcal{G}^{l-1} \tag{7}$$

  Observe that $d_{2p,2q}^l = 0$, because of the interpolation property. Thus even-even detail coefficients are never computed.

– Relations (6) and (7) lead immediately to

$$\begin{aligned} v_{ij}^{l-1} &= v_{i/2\ j/2}^{l} \quad \text{if } (x_i, y_j) \in \mathcal{G}^l \\ v_{ij}^{l-1} &= I[(x_i, y_j); v^l] + d_{ij}^l \quad \text{if } (x_i, y_j) \in \mathcal{G}^{l-1} \setminus \mathcal{G}^l \end{aligned} \tag{8}$$

Applying this procedure from $l = 1$ to L, gives an equivalence between the discrete set v^0 and its multiresolution representation : $(v^L, d^L, .., d^1)$.

Remark 2.1 *In our numerical experiments we use a tensor-product interpolation procedure of order 4 ($r = 4$). The corresponding formulae come from standard two dimensional polynomial interpolation; explicit details can be found for example in [4] (section 3).*

This framework is probably the simplest one, because the detail coefficients are simply interpolation errors, thus their size depends directly on the local regularity of v. This piece of information will help us to localize the non-smooth structures of the solution.

3 The Multilevel algorithm

As observed in [4], the original idea of a multilevel computation of the numerical flux function (in one dimension) described by Harten in [14] cannot be used in a robust and general manner in 2D. The key point is then to observe that it is the *numerical divergence* the quantity that should be adapted to the multilevel computation.

The situation can be easily understood when we consider the simplest ODE solver: Euler's method. When applied to the semi-discrete formulation (2), we get

$$\vec{U}_{ij}^{n+1} - \vec{U}_{ij}^{n} = -\delta t\ \mathcal{D}(\vec{U})_{ij}^{n}. \tag{9}$$

It is thus clear that treating the numerical divergence with the same framework as the sets $\vec{U}_{ij}^{n,n+1}$ is the most reasonable thing to do.

In [4], the cell-average multiresolution framework was used to analyze the discrete data at each time level, as a consequence the numerical divergence was interpolated (when this was appropriate) in the sense of cell-averages.
In this paper we propose to use the point-value multiresolution framework to as the basic setting in the multilevel strategy. This allows us to use the scale coefficients directly as regularity sensors in the multi-scale scheme.

Aside from the fact that the multiresolution framework in our multilevel algorithm is different from that of [4], there are only a few differences between our main strategy and that of [4], thus we only mention the main steps of the algorithm, pointing out the differences along the way.

3.1 Description

First the numerical divergence is evaluated at all points on the coarsest grid $\mathcal{G}^L$ using the numerical flux function of the prescribed HRSC scheme. Then, for the finer grids, $\mathcal{D}$ is evaluated by the same solver **or** with a cheap interpolation procedure using the values obtained on the coarser grids. The choice depends on the regularity analysis of the approximate solution, made with the help of its multiresolution representation.

The main ingredients of the algorithm are the following :

- The *multiresolution transform* described in section 2.1 to obtain the wavelet coefficients of $\vec{U}^n$.
- A *thresholding algorithm* which associates to each wavelet coefficient a boolean flag, b^l_{ij}, whose value (0 or 1) will determine the choice of the procedure to evaluate $\mathcal{D}$. The goal is to use this flag to mark out the non-smooth regions of $\vec{U}^n$ *and* $\vec{U}^{n+1}$.

  For a given tolerance parameter ϵ, the tolerance at level l is defined as $\epsilon_l = \epsilon 2^{-l}$. Starting from a zero value for all b^l_{ij}, one applies for each detail coefficient the following two tests:

$$\begin{aligned} \text{if } |d^l_{ij}| &\geq \epsilon_l \Longrightarrow b^l_{i-k\ j-m} = 1 \quad k,m = -2,..,2 \\ \text{if } |d^l_{ij}| &\geq 2^{r+1}\epsilon_l \text{ and } l > 1 \Longrightarrow b^{l-1}_{2i-k\ 2j-m} = 1 \quad k,m = -1,0,1 \end{aligned} \tag{10}$$

  The first test takes into account the propagation of information (recall that the propagation of 'real' information is limited by the CFL condition). The second one aims at detecting shock formation. In a smooth region the local rate of decay of the detail coefficients is determined by the accuracy of the interpolation and the local regularity of the function. The test measures if the decay rate is that of a smooth function, if this is not the case, compression leading to shock formation might be taking place and the location is also flagged (see [4] for specific details on both tests).

- The *multilevel evaluation* of the numerical divergence.
  For all points $(x_i, y_j) \in \mathcal{G}^L$, $\mathcal{D}^L(\vec{U})_{ij}$ is computed with the prescribed HRSC scheme. Once the divergence is known on $\mathcal{G}^l$, its value on $\mathcal{G}^{l-1}$, $\mathcal{D}^{l-1}(\vec{U})$, is evaluated using the boolean flag:

$$\begin{aligned} &\text{if } b^l_{ij} = 1 \qquad \text{compute } \mathcal{D}^{l-1}(\vec{U})_{ij} \text{ directly (with HRSC method).} \\ &\text{if } b^l_{ij} = 0 \qquad \mathcal{D}^{l-1}(\vec{U})_{ij} = I[(x_i, y_j); \mathcal{D}^l(\vec{U})]. \end{aligned} \tag{11}$$

Letting l go from L to 1 gives us the values of $\mathcal{D}(\vec{U})$ on the finest grid $\mathcal{G}^0$, which are needed by the ODE solver.

Remark 3.1 *The direct evaluation of $\mathcal{D}^{l-1}(\vec{U})_{ij}$, is performed by computing the numerical flux functions using the values of $\vec{U}$ on $\mathcal{G}^0$:*
$\vec{F}(\vec{U}_{2^{l-1}i-k,j}, .., \vec{U}_{2^{l-1}i+m,j})$ and $\vec{G}(\vec{U}_{i,2^{l-1}j-k}, .., \vec{U}_{i,2^{l-1}j+m})$.
This fact is important to ensure the accuracy of the scheme, in terms of the finest grid-size, as well as the application of the numerical flux function to the 'correct' values in a neighborhood of singularities (users of HRSC methods are well aware that the numerical flux formulas are sometimes very sensitive to the (often reconstructed) values fed into the computation).
As a consequence, the finest grid, $\mathcal{G}^0$, is always needed and no memory savings is obtained in comparison to the direct method (without multiresolution).

	$\vec{U}$ multiresolution	ρ multiresolution
MR transform	0.09	0.024
Maximum	0.014	–
Thresholding	0.026	0.027
Total	0.13	0.051

Table 1: *Cpu time in seconds for the overhead steps of the multilevel algorithm. First column for all the components of $\vec{U}$, second one only for the density.*

The possibility to derive a fully adaptive algorithm has been explored recently (see [5] for one dimensional experiments) but so far, it seems to lead to complex data structures which increase considerably the difficulty of using such methods for two or three dimensional physical problems.
From our point of view, one important advantage of the algorithm described here is the relative simplicity in which it can be incorporated into existing CFD codes without altering their basic structure.

3.2 Details of implementation

In the original work of Harten [14], the flag coefficients are obtained using a multiresolution transform for each component of the vector $\vec{U}$. The thresholding algorithm is then applied to the largest resulting wavelet coefficient, i.e $\tilde{d}^l_{ij} = max(|d_{ij}(\rho)|, |d_{ij}(m_x)|, |d_{ij}(m_y)|, |d_{ij}(E)|)$.

For the Euler equations of gas dynamics, the density retains all the possible non-smooth structures of the flow (shocks, contact discontinuities and corners of rarefaction waves), thus it seems to be sufficient to derive the flag only from the density multiresolution representation. This modification has been tried first by Sjögreen [20] and our experiments show that no significant differences are noted in the quality of the numerical results, thus our numerical tests compute the boolean flag using only the multi-scale information of the density.
In table 1, we present the cpu time measured for both methods for an initial grid $\mathcal{G}^0$ of 512×128 points and 5 levels for the multiresolution transform. We observe a reduction of the computational time by a factor of 2 in that case.

The second modification concerns the time evolution. The ODE solver we use is the third order TVD Runge-Kutta method introduced by Shu and Osher [19]. This semi-discrete approach has also been carried out in [4] and [20]. There, a flag vector is computed before each intermediate step between t^n and t^{n+1}. As we will see it is possible to avoid this computation for the last step. The third order Runge-Kutta is defined as follows:

$$\begin{aligned} \vec{U}^* &= \vec{U}^n - \delta t\ \mathcal{D}(\vec{U}^n) \\ \vec{U}^{**} &= (3\vec{U}^n + \vec{U}^* - \delta t\ \mathcal{D}(\vec{U}^*))/4 \\ \vec{U}^{n+1} &= (\vec{U}^n + 2\vec{U}^{**} - 2\delta t\ \mathcal{D}(\vec{U}^{**}))/3, \end{aligned} \tag{12}$$

and the intermediate steps can be represented on the time axis as in figure 1. Since $\vec{U}^*$ is an order 1 approximation of $\vec{U}^{n+1}$, and therefore contains similar non-smooth structures at the same places, the mask coefficients obtained from its multiresolution transform could be used to compute $\vec{U}^{n+1}$

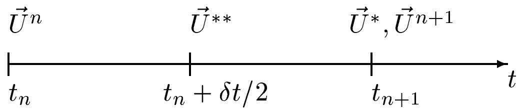

Figure 1: *Representation on the time axis of the intermediate steps of the third order Runge-Kutta method.*

from $\vec{U}^{**}$, instead of deriving it from the $\vec{U}^{**}$ multiresolution transform. This modification reduces the computational cost of the multilevel algorithm while keeping the same quality in the numerical results.

In [20], Sjögreen presents a different version of the multilevel algorithm. Its version used a "dimension by dimension" algorithm instead of a full bi-dimensional one. This means that for the fluxes in the x-direction, $\vec{F}_{i+1/2,j}$, a one dimensional multilevel algorithm is applied to each grid line $j = j_0$. Then, the same procedure is applied to each line $i = i_0$ to compute the y-direction fluxes.
The major advantage comes from the fact that only one dimensional procedures are used for the multiresolution transform, the thresholding algorithm and the interpolation process.

We have implemented also Sjögreen version in the point value context and compared with our bi-dimensional algorithm. From the quality point of view, the results are very similar and the percentage of fluxes computed by the solver is the same in both cases. Nevertheless, Sjögreen's version turns out to be less efficient than the two dimensional one and, in our implementation, a factor of 1.6 is observed between the corresponding cpu times. The difference could be explained by the fact that each point of the domain is visited two times by the multiresolution transform and thresholding algorithm (for the x and y flux computations) and that this algorithm requires more memory access.

3.3 Error Analysis

In [14] Harten gives error estimates for the multilevel algorithm for the one dimensional scalar case. Such estimations assume hypotheses on the underlying shock capturing method that are not satisfied by the ENO or Marquina schemes, but give a first idea of the stability of algorithm.
If we call v_{ref} the solution computed by a l_1-contractive scheme [14] and v_{mult} the solution computed using the multilevel algorithm based on the same scheme, Harten proves that (under appropriate assumptions)

$$\|v_{ref} - v_{mult}\|_1 \leq C\ \epsilon, \tag{13}$$

assuming that ϵ_l, the tolerance at level l, is defined as $\epsilon_l = 2^{-l}\epsilon$.
This estimate has been derived in the cell-average context, but the same arguments hold for our point value approach. An inequality like (13) guarantees that the error associated to the multilevel computation is directly controlled by the tolerance ϵ. In section 4, we perform a series of numerical

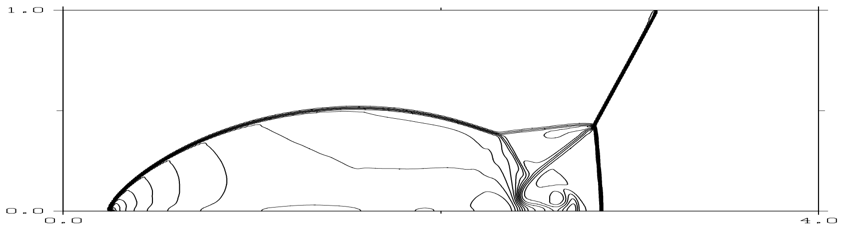

Figure 2: *Density reference solution for the test A at time* $t = 0.2$*, obtained with* 512×128 *points and Marquina's scheme without multiresolution*

tests that point out that a similar estimate holds for the multilevel scheme we consider.

4 Numerical experiments

This section is devoted to the presentation and analysis of the results obtained with the multilevel algorithm.

4.1 Description of the test cases

In order to test the multilevel algorithm, we consider two classical configurations governed by the two-dimensional Euler equations.

Test A : *Double Mach Reflection of a strong shock*

The problem begins with a 10 Mach shock which makes a 60^o angle with the bottom wall of a tunnel 4 units long and 1 unit high. Initially the shock extends from the point $x = \frac{1}{4}$ at the bottom of the domain, to the top boundary. The reflecting wall begins at $x = \frac{1}{4}$ on this bottom wall. The post-shock conditions are $\vec{U}_{left} = (8., 57.1597, -33.0012, 563.544)$ and the air ahead the shock is undisturbed and has density 1.4 and pressure 1. Post-shock values are assigned at the boundaries located to the left of the shock. At the right end of the domain outflow conditions are applied and the top values at the right of the shock are those of the undisturbed air. The top conditions change with time to describe the exact motion of the initial shock. The behavior of the solution and the description of the flow structure are detailed in [22].
The density obtained at time $t = 0.2$ with Marquina's scheme *without* multiresolution and with a grid of 512×128 points is displayed in figure 2. The complex structures which appear are difficult to compute and their proper representation is a good challenge for the multilevel algorithm.

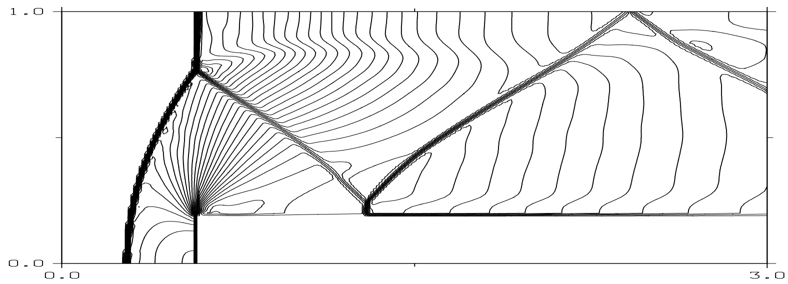

Figure 3: *Density reference solution for the test B at time* $t = 4$, *obtained with* 256×80 *points and Marquina's scheme without multiresolution*

Test B : *Mach 3 wind tunnel with a step*

This test, introduced by Emery [9] 30 years ago, has received a lot of attention during the last decade in order to analyze and compare numerical difference methods.
The problem begins with an uniform Mach 3 flow in a tunnel containing a step. The tunnel is 3 units long and 1 unit wide, and the step is located 0.6 units from the left-hand end of the tunnel and is 0.2 units high. Inflow boundary conditions are applied at the left of the domain and outflow conditions occur at the right end. Along all the walls of the tunnel reflecting boundary conditions are applied. A detailed description of the different structures of the flow is presented in [22].
In figure 3, we display the density distribution at time $t = 4$ obtained with Marquina's scheme *without* multiresolution (the number of points in the computational grid is 256×80). This simulation will serve as the reference for the multilevel results.

4.2 Computational setting and test results

In this first set of experiments we apply the multilevel algorithm with Marquina's scheme to compute the solution of the previous test problems. The *reference solutions* are the solutions obtained by direct computation on the uniform grid $\mathcal{G}^0$ presented on figures 2 and 3. The parameters for each multilevel computation are specified below.

Test A : The initial grid $\mathcal{G}^0$ has 512×128 points, we consider $L = 5$ levels and the tolerance parameter is set to 5×10^{-3}.
Figure 4 shows the density field of the computed solution at time $t = 0.1$ and $t = 0.2$ and also the corresponding adaptive grids used to compute it. These grids are obtained representing only the points of $\mathcal{G}^0$ where the numerical divergence is computed directly with the HRSC scheme.

Test B : The initial grid $\mathcal{G}^0$ has 256×80 points, the number of levels is 4 and the tolerance parameter is also 5×10^{-3}. On figures 5 and 6 we have represented the density fields for three different times

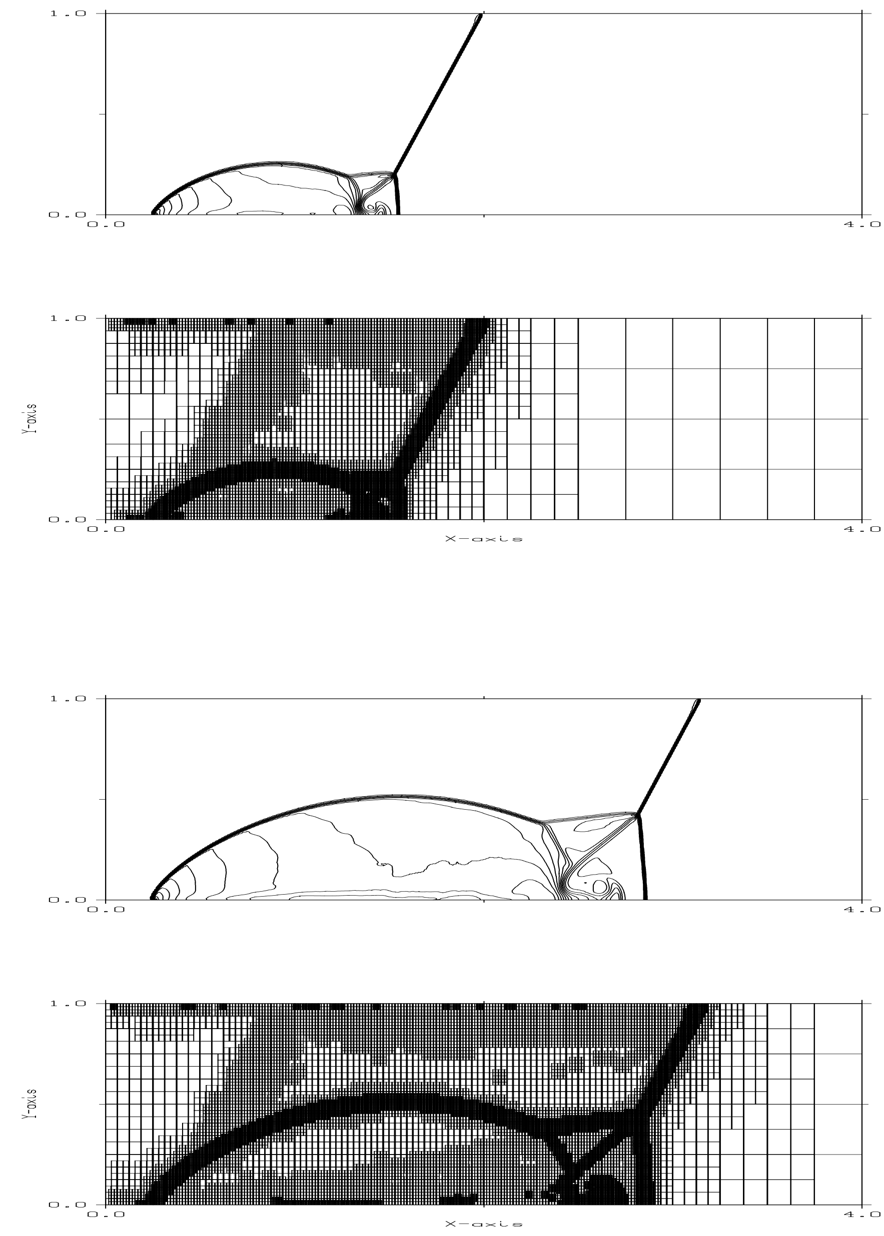

Figure 4: *Density field of the multilevel solution and adaptive grids for the test A at time* $t = 0.1$ *(top) and* $t = 0.2$ *(bottom). Initial grid* $\mathcal{G}^0$ *contains* 512×128 *points,* $L = 5$ *levels, and* $\epsilon = 5 \times 10^{-3}$.

: $t = 0.5, 1.5$ and 4.

In general terms, we observe that the smoothness analysis performed on the wavelet coefficients is able to localize correctly the non-smooth structures of the flow. The numerical results show that a direct evaluation has been performed in the neighborhood of all singularities, as well as in the shock formation process. It should be observed that the multilevel solution is 'visually' *no worse* (this will be quantified later) than the reference solution.
The plots of the adaptive grids are also interesting. As observed in [22], for test A, the results are marred by small numerical errors due to the initialization and boundary conditions. As a consequence, there is some refinement in smooth regions that is probably unnecessary, since no shock formation or evolution is taking place. This phenomenon occurs for the both reference and multilevel simulations, but the multilevel scheme clearly shows the occurrence and relative importance of these errors.
Notice also the refinement appearing at reflecting walls in both tests. The problem of dealing with reflecting boundary conditions in high resolution simulations has been addressed by various authors in recent papers, and here the multilevel algorithm can also help to detect which areas of the computational domain are displaying a numerical behavior susceptible of improvement.

4.3 Quality and Efficiency

When computing numerical approximations with the multilevel scheme, we shall be concerned with two basic questions: *Quality* and *efficiency.*
The question of quality will be analyzed by measuring, the discrete l_1-norm of the difference between the multilevel solution $\vec{U}^n$ and the reference one, $\vec{U}^n_{ref}$. This difference can be represented by any of the variables involved in the computation. For example for the density we compute

$$e_1^\rho = \frac{1}{N_p} \sum_{i=0}^{N_x} \sum_{j=0}^{N_y} |\rho_{ij}^n - \rho_{ref_{ij}}^n| / \|\rho_{ref}^n\|_{l_1}, \tag{14}$$

where $N_p = (N_x + 1) \times (N_y + 1)$ represents the total number of points.
To examine the relation between the tolerance ϵ and the error in the non-scalar case and to compare with the estimate (13), the multilevel algorithm is applied to test A with 128×32 points for $\mathcal{G}^0$, 3 levels and different values for ϵ. The error is measured, for the density ρ and the pressure p, at time 0.2 and results are presented on figure 7.
Both e_1^ρ and e_1^P decrease with ϵ more rapidly than the estimate (13), the decreasing order seems to be here approximately 1.6 instead of 1.
The improvement can be due to the fact that in practical computations many detail coefficients are much smaller than their truncation parameter ϵ_l (this cannot be taken into account in the theoretical estimates without some a priori knowledge on the solution), and to the fact that the neighbors of coefficients larger than the tolerance are also considered. Hence more coefficients are involved around discontinuities which may lead to a better accuracy in the reconstruction process.
The results of figure 7 imply that we can control the quality of the multilevel simulations by adjusting the tolerance to suit our needs.

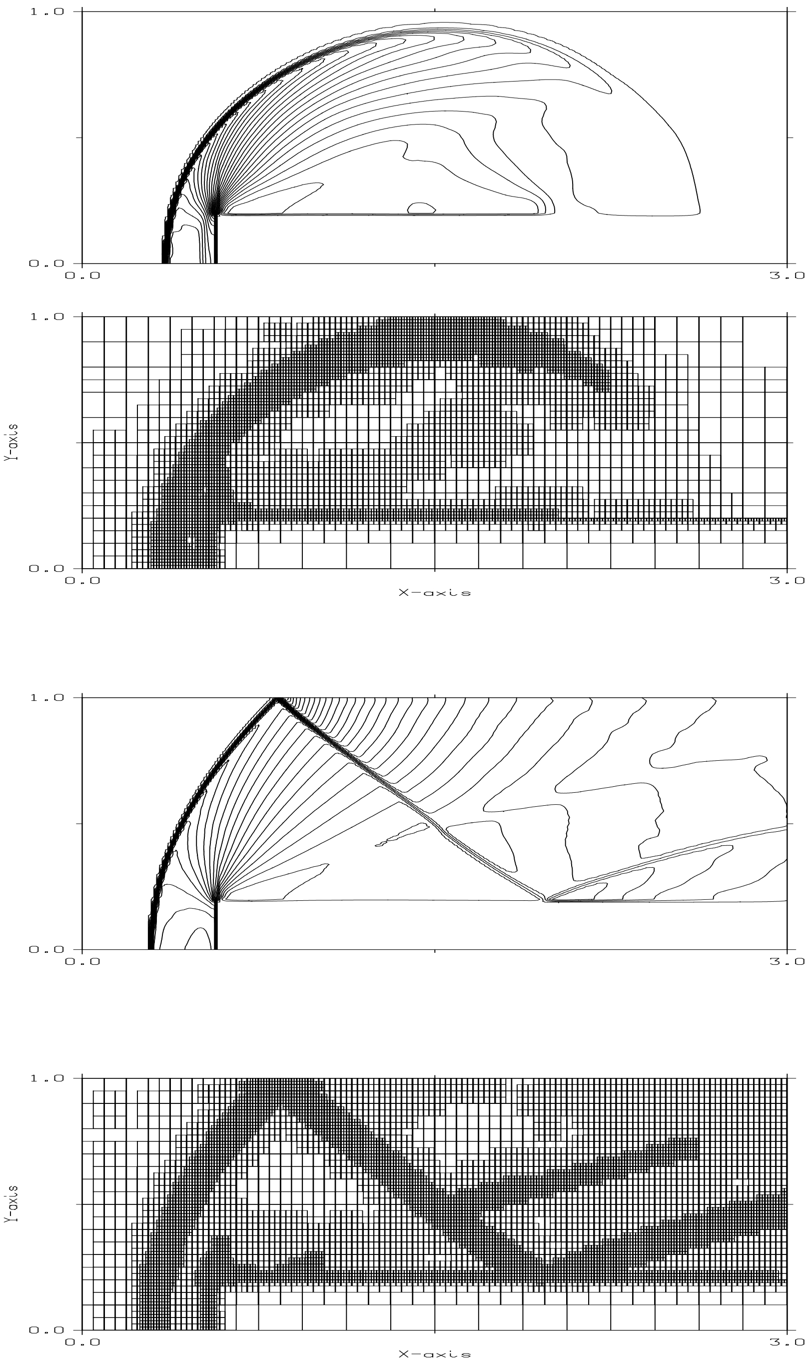

Figure 5: *Density field of multilevel solution and adaptive grids for the test B at time $t = 0.5$ (top) and $t = 1.5$ (bottom). Initial grid $\mathcal{G}^0$ contains 256×80 points, $L = 4$ levels, and $\epsilon = 5 \times 10^{-3}$.*

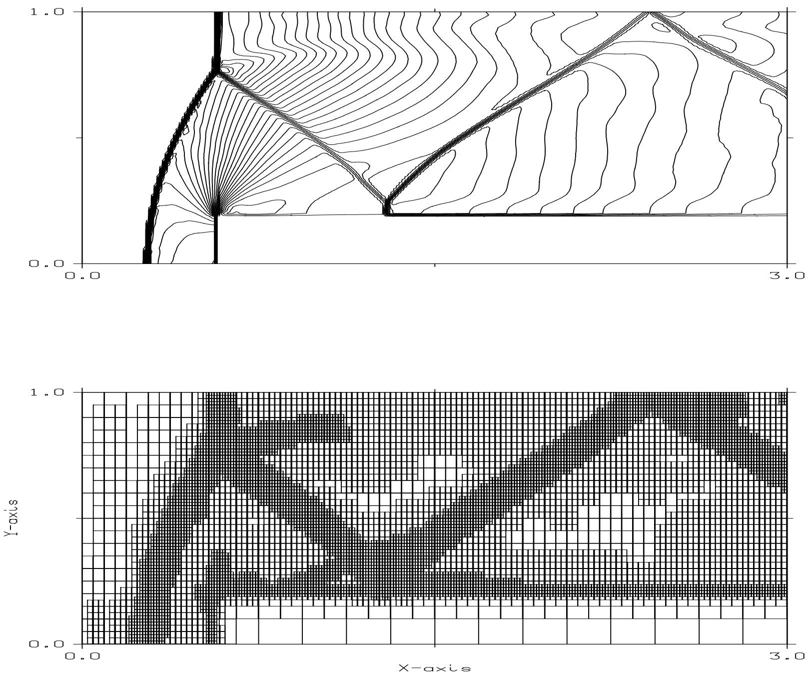

Figure 6: *Same as figure 5 for the time $t = 4$.*

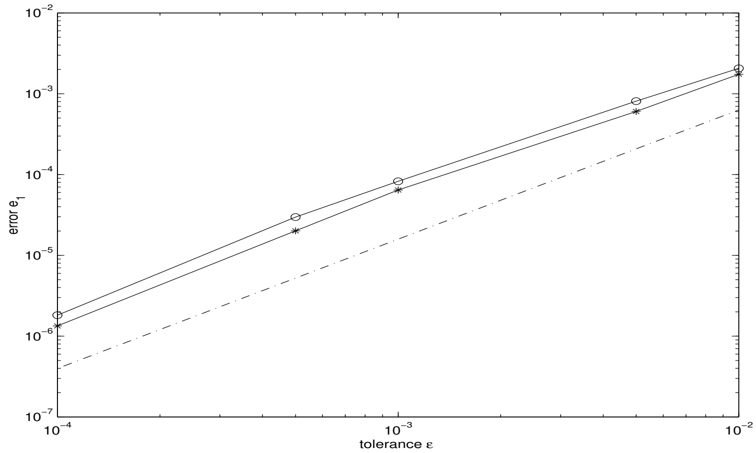

Figure 7: *Error e_1, for the density $(*)$ and the pressure (o), between the multilevel algorithm and the reference one versus the tolerance ϵ. The dotted line represents the curve of equation: $\epsilon^{1.6}$.*

grid size $\mathcal{G}^0$	$\%f_{\min}$ – $\%f_{\max}$	cpu gain θ
128×32	17.6 – 52.7	1.7
256×64	8.9 – 33.2	2.45
512×128	4.5 – 23.2	3.8

Table 2: *Percentage of resolved flux and cpu gain for test A at time* $t = 0.2$

grid size $\mathcal{G}^0$	$\%f_{\min}$ – $\%f_{\max}$	cpu gain θ
128×40	7 – 69.5	0.9
256×80	2.8 – 45	1.4

Table 3: *Same as table 2 for test B at time* $t = 4$

To evaluate the efficiency of the multilevel algorithm, we shall measure, in each one of our numerical test cases, the cpu gain for a given iteration, θ_{iter}, and the gain for the global simulation, θ.
Introducing t_{ref}^{iter} and t_{mr}^{iter} as the cpu times at iteration *iter* for respectively the reference and the multilevel algorithm, the gains are defined by :

$$\theta_{iter} = \frac{t_{ref}^{iter}}{t_{mr}^{iter}} \quad \text{and} \quad \theta = \frac{\sum t_{ref}^{iter}}{\sum t_{mr}^{iter}}. \tag{15}$$

It is obvious that the efficiency of the multilevel algorithm is directly related to the savings in the direct evaluation of the numerical divergence. An important quantity is thus the percentage of numerical divergences computed directly per time step, $\%f$.

Table 2 (for test A) and table 3 (for test B) show the gain in global cpu time as well as the maximum and minimum values of $\%f$. Observe that, for a given test, the finer the grid, the smaller the number of direct flux evaluations (i.e. the smaller $\%f$), since the number of computational cells involved in a non-smooth structure is independent (more or less) of the underlying computational grid.

The relation between the gain in execution time and the parameter $\%f$ can be further examined on figure 8, where we show the time evolution of θ and $\%f$ for both tests. It can be observed that the behavior of θ is roughly inversely proportional to that of $\%f$.

In table 4 we show the cpu time for one step of the multilevel algorithm and one stage of the Runge-Kutta method. These results have been obtained with test A and 512×128 points in $\mathcal{G}^0$ (here $\%f$ has its maximum value, 23%). It may be noticed that the overhead caused by the multiresolution transform and the threshold represents only a small part of the total cpu time, 2%, and that most of the time is spent in the numerical divergence evaluation, 96%.

These observations lead to the conclusion that the multilevel algorithm allows us to compute with very fine grids at the cost of a much coarser one (the speed-up allows us for example, to reduce

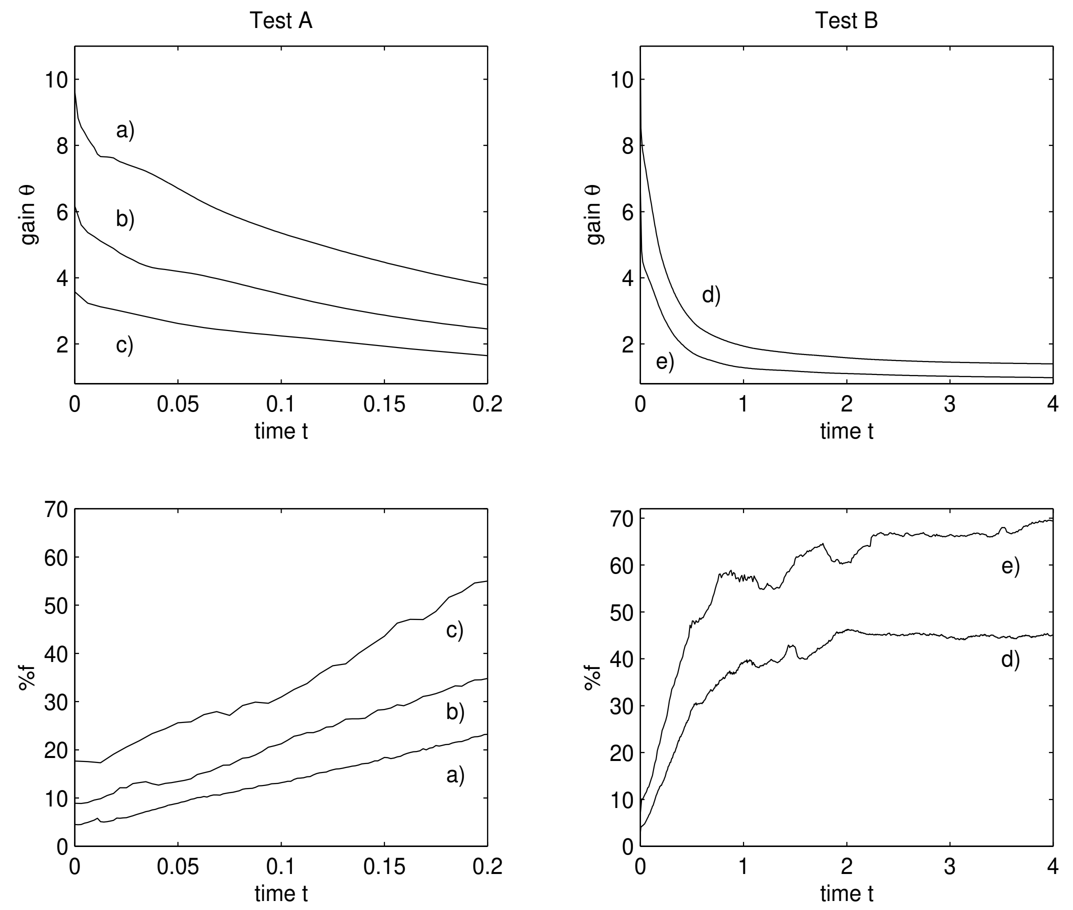

Figure 8: *Time evolution of* θ *(top) and* $\%f$ *(bottom) for test A (left) and test B (right) and for different initial grid* $\mathcal{G}^0$*. a)* 512×128*, b)* 256×64*, c)* 128×32*, d)* 256×80*, e)* 128×40.

	Multilevel algorithm	Reference algorithm
Transform	0.06	–
Thresholding	0.08	–
Divergence evaluation	6.9	13.8
Other	0.15	0.15
Total	7.2	13.95

Table 4: *Cpu time in seconds for the different steps of the multilevel and reference algorithms for one Runge-Kutta stage. These values are obtained with the test A and the largest grid* 512×128 *and with* $\%f = 23$.

grid size $\mathcal{G}^0$	$\%f_{\min}$ – $\%f_{\max}$	cpu gain θ
128×32	17.6 – 54.2	1.54
256×64	8.9 – 38.4	2.2
512×128	4.5 – 25.7	2.9

Table 5: *Percentage of resolved fluxes and cpu gain for test A with ENO fluxes at time* $t = 0.2$

the cpu time from 3 hours with the direct simulation[1] to 47 minutes for test A with the largest grid).

4.4 Comparison with ENO schemes

Let us consider now the ENO scheme of order three described by Shu and Osher in [19], and repeat the numerical tests for case A. It should be mentioned that some entropy corrections, as proposed in [11], are needed near the reflecting wall to avoid the occurrence of a carbuncle phenomenon in the case of the finest grid; these corrections are unnecessary for Marquina's scheme.
Figure 9 displays the outcome of the numerical experiments (same parameters as in figure 2) and table 5 reports the minimum and maximum percentage of ENO-computed numerical divergences and the gain for the global simulation.

Comparing with the results of table 2, we remark that the gain is not as large as in the case of Marquina's scheme, but remains significant. This is to be expected, since the cost of a direct evaluation of the numerical divergence by Marquina's scheme is higher than that of the ENO scheme (by a factor of 2 in our implementation). As pointed out by Sjögreen [20], the expected gain in the efficiency of the multilevel algorithm is larger for more expensive flux formulas. A simple comparison between tables 2 and 5 shows this to be the case.
For the same computation, a slightly larger percentage of ENO-computed numerical divergences are needed. This could be explained by the fact that the numerical solution (the multilevel as well as the reference one) obtained with Marquina's scheme does a better job at localizing the non-smooth

[1]all the simulations are done with a PC 350-Mhz

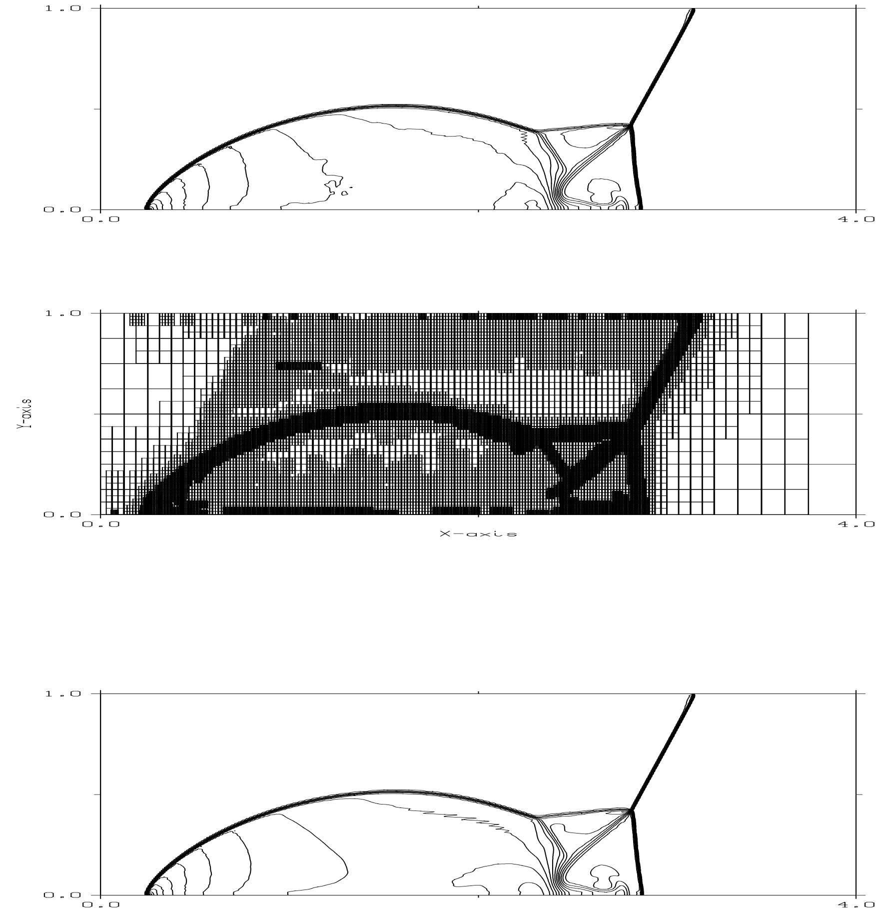

Figure 9: *Density field of the multilevel solution and adaptive grids for the test A with ENO fluxes at time* $t = 0.2$ *(top) and for the reference solution (bottom) computed with ENO scheme. Initial grid* $\mathcal{G}^0$ *contains* 512×128 *points,* $L = 5$ *levels, and the tolerance is* 5×10^{-3}.

structures of the numerical solution, especially at the small region of the double Mach reflection. In the plots of the adaptive grid structures for both computations (figures 4 and 9) it can be clearly appreciated the difference in resolution between the two schemes in the aforementioned region.

It is worth giving some computing times for the sake of comparison: In the simulations of figures 4 and 9 (the grid is 512×128), the *multilevel* simulation with Marquina's solver takes 2877 seconds while the *reference* ENO simulation takes 4980 seconds. It means, in this case, a significant saving in time (of a factor 1.7) *with* an effective increase of quality.

For test-case B, the Roe-matrix-related numerical instabilities that appear at the base of the bow shock (see e.g. [8]) lead to a crash of the code for the finest grid, unless the entropy is corrected at the bottom wall, as prescribed in [11]. The conclusions of the numerical tests with test-case B and coarser grids are absolutely similar to those reported already and will not be shown in this paper.

To end this computational study, in figure 10 we have represented the gain for one iteration, θ_{iter} (15), versus the percentage of resolved fluxes $\%f$, for both Marquina and ENO schemes. This representation is more or less independent of the considered test case since the time evolution is not taken into account.

We remark that the gain is much more important for Marquina's scheme and small values of $\%f$. Observe also that the difference is reduced when this percentage increases, a fact that could be easily understood considering the following (crude) estimate of θ_{iter}

$$\begin{aligned} \theta_{iter} &= \frac{N_p t_f}{t_{mr} + t_{thres} + N_f t_f + (N_p - N_f) t_I} \\ &= \frac{t_f}{\frac{1}{N_p}(t_{mr} + t_{thres}) + \lambda t_f + (1-\lambda) t_I} \end{aligned} \tag{16}$$

where t_f is the cpu time to compute one value of the numerical divergence with the solver, t_I the cpu time for one interpolation and t_{mr} and t_{thres} denote respectively the multiresolution transform and thresholding cpu times (which are essentially negligible, as shown in table 4). N_p is the total number of grid points, N_f represents the number of points where the numerical divergence is evaluated directly, and $\lambda = N_f/N_p$.
Taking the same percentage of resolved fluxes for both schemes, i.e $\%f^M = \%f^E (= 100 * \lambda)$, we can write :

$$\begin{aligned} \frac{\theta^M_{iter}}{\theta^E_{iter}} &= \frac{t^M_f}{t^E_f} \frac{\frac{1}{N_p}(t_{trans} + t_{thres}) + \lambda t^E_f + (1-\lambda) t_I}{\frac{1}{N_p}(t_{trans} + t_{thres}) + \lambda t^M_f + (1-\lambda) t_I} \\ &\sim 2 \frac{\lambda + (1-\lambda)\frac{t_I}{t^E_f}}{2\lambda + (1-\lambda)\frac{t_I}{t^E_f}} \end{aligned} \tag{17}$$

since in our implementation $t^M_f / t^E_f \sim 2$.
The function

$$g(\lambda) = 2\frac{\lambda + (1-\lambda)\beta}{2\lambda + (1-\lambda)\beta} \tag{18}$$

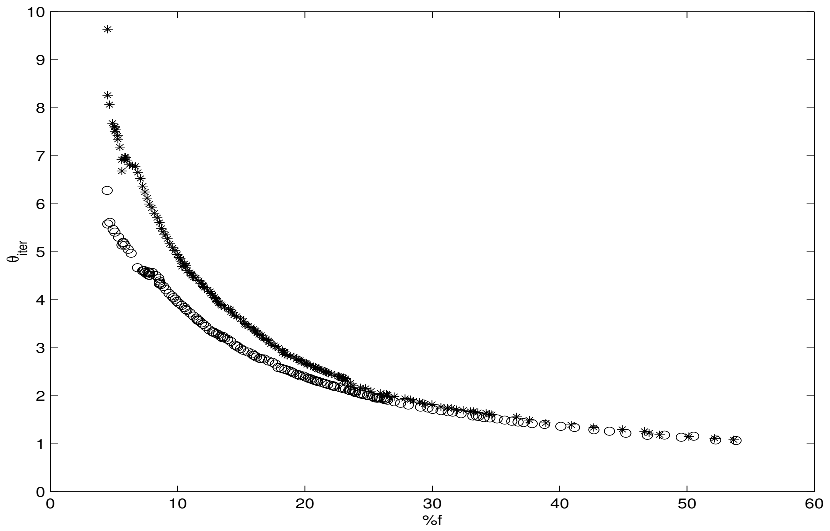

Figure 10: *Gain per iteration θ_{iter} versus the percentage of resolved fluxes $\%f$, for Marquina $(*)$ and ENO (o) schemes.*

where $\beta := t_I/t_f^E$, is monotonically decreasing, and approaches 1 when λ tends to 1. Moreover, the smaller is the ratio β, the faster is the convergence to this limit value 1. In our computations, the ratio β is approximately $\frac{1}{56}$, which leads to $g(.4) = 1.01$ and explains the behavior observed on figure 10.

From figure 10 one remarks also that for both cases, if $\%f \geq 60\%$ the multilevel algorithm is no longer computationally competitive.

5 Conclusions

We have presented a multilevel algorithm designed to reduce the high computational cost associated to some HRSC schemes for hyperbolic systems of conservation laws.

The scheme follows the basic multilevel strategy of Bihari and Harten in [4], but it is built upon the interpolatory multiresolution framework, instead of the cell-average framework as in previous works [1, 2, 4, 5, 20]. This simplifies somewhat the algorithm and its interpretation. We have also introduced some modifications with respect to earlier works [4, 20] that lead to improvements in the overhead due to the multiresolution transform.

We test our multilevel algorithm on two well known tests for the two dimensional Euler equations and observe a significant reduction in the computational time with respect to the reference computation. The efficiency of the multilevel algorithm is, of course, problem dependent. As pointed out by Sjögreen, the more expensive the flux evaluation, the more efficient the method is.
The algorithm we propose is one alternative that can be *easily* incorporated into an existing code.

Its potential users might be researchers performing computational tests with state of the art HRSC methods and using uniform grids (other possibilities with non-uniform grids and the cell-average framework have also been explored [1, 2]). The methods offers the possibility to obtain a high-resolution numerical solution on a very fine grid, at the cost of the user's own numerical technique on a much coarser mesh. The conditionant is that the discrete values at the highest resolution level need to be always available.

References

[1] R. Abgrall. Multiresolution in unstructured meshes : Application to CFD. *Numerical methods for fluid dynamics, Oxford University Press*, 5, 1996.

[2] R. Abgrall, S. Lantéri, and T. Sonar. Eno schemes for compressible fluid dynamics. *ZAMM, Z. angew. Math. Mech.*, 79:3–28, 1999.

[3] F. Arandiga, R. Donat, and A. Harten. Multiresolution based on weighted averages of the hat function II: non-linear reconstruction techniques. *SIAM J. Sci. Comput.*, 1998.

[4] B.L Bihari and A. Harten. Multiresolution schemes for the numerical solutions of 2D conservation laws. *SIAM J. Sci. Comp.*, 18(2):315–354, 1997.

[5] W. Dahmen, B. Gottschlich-Müller, and S. Müller. Multiresolution schemes for conservation laws. *Technical report*, Bericht Nr 159 IGMP, RWTH Aachen, 1998.

[6] A. Dolezal and S.S.M Wong. *J. Comp. Phys.*, 120:266, 1995.

[7] R. Donat and A. Marquina. Capturing shock reflexion: An improved flux formula. *J. Comp. Phys.*, 25, 1996.

[8] R. Donat and A. Marquina. Computing strong shocks in ultrarelativistic flows: A robust alternative. *International Series of Numerical Mathematics*, 129, 1999.

[9] A. E. Emery. *J. Comp. Phys.*, 2, 1968.

[10] F. Eulderink. Phd. thesis. *University of Leiden*, 1993.

[11] R. Fedkiw, A. Marquina, and B. Merriman. An isobaric fix for the overheating problem in multimaterial compressible flows. *J. Comp. Phys.*, 148, 1999.

[12] R. Fedkiw, B. Merriman, R. Donat, and S. Osher. The penultimate scheme for systems of conservation laws: Finite difference ENO with Marquina's flux splitting. *UCLA CAM Report*, January 1997.

[13] A. Harten. Discrete multiresolution analysis and generalized wavelets. *J. Appl. Numer. Math.*, 12:153–192, 1993.

[14] A. Harten. Multiresolution algorithms for the numerical solution of hyperbolic conservation laws. *Comm. Pure Appl. Math.*, 48, No 12, 1995.

[15] A. Harten. Multiresolution representation of data: a general framework. *SIAM J. Numer. Anal.*, 33(3):1205–1256, 1996.

[16] S.M. Kaber and M. Postel. Finite volume schemes on triangles coupled with multiresolution analysis. *C.R. Acad. Sci., Série I, Paris*, 328:817–822, 1999.

[17] A. Marquina. Local piecewise hyperbolic reconstruction of numerical fluxes for nonlinear scalar conservation laws. *SIAM J. Sci. Comput.*, 15, 1994.

[18] J.M Marti, E. Muller, J.A. Font, J.M. Ibanez, and A. Marquina. *Astrophys. J.*, 479:151–163, 1997.

[19] C. W Shu and S. J. Osher. Efficient implementation of essentially non-oscillatory shock-capturing schemes. *J. Comp. Phys.*, 83, 1989.

[20] B. Sjögreen. Numerical experiments with the multiresolution scheme for the compressible euler equations. *J. Comp. Phys.*, 117, 1995.

[21] W. Sweldens. The lifting scheme: A construction of second generation wavelets. *SIAM J. Math. Anal.*, 29:511–546, 1997.

[22] P. R. Woodward and P. Colella. The numerical simulation of two-dimensional fluid flow with strong shocks. *J. Comp. Phys.*, 54, 1984.