Adaptive Multiresolution for Finite Volume Solutions of Gas Dynamics

A. Cohen, S.M. Kaber, M. Postel

Université Pierre & Marie Curie, Boîte courrier 187, 75252 Paris cedex 05, France

Abstract

Multiresolution analysis (MR) is used to improve the CPU and memory performance of a finite volume scheme. Departing from Harten’s original scheme we present a fully adaptive scheme in the sense that at a given time, the solution is represented in a compressed form by a set of significant wavelet coefficients. Numerical benchmarks for Euler’s system of compressible gas dynamics are performed on triangular meshes.

Key words: multiresolution analysis, finite volumes, adaptivity

1 Introduction

We are interested in solving systems of nonlinear hyperbolic conservation laws written in their general 2D form as

$$\partial_t u + \text{Div}_x (f(u(t,x))) = 0, \quad u \in \mathbb{R}^m, \quad x \in \mathbb{R}^2, \quad t > 0.$$  

with initial value $u(t = 0, x) = u_0(x)$. The main difficulties are related to the appearance of discontinuities in the solutions of such equations. These discontinuities can appear in finite time even for smooth initial and boundary conditions data. The position of these discontinuities varies generally with time, which is of course an additional difficulty.

In order to represent accurately the discontinuous solutions it is necessary to discretize the computational domain on very fine grids. Furthermore, to ensure efficiency and high precision on these fine grids, costly nonlinear solvers are used. On the other hand, in the most frequent case, the discontinuities

Email address: postel@ann.jussieu.fr (M. Postel).
URL: www.ann.jussieu.fr/~postel (M. Postel).

Preprint submitted to Elsevier Preprint 17 July 2001
are localized in only a small region of the total domain. The fine discretiza-
tion which is necessary in the vicinity of the discontinuities is superfluous
elsewhere. The costly numerical fluxes (like ENO methods for instance) could
be advantageously replaced by uniform higher order schemes wherever the
solution is smooth enough.

The goal of this work is to use multiresolution strategies to build up an efficient
and robust adaptive algorithm which solves the above difficulties. Strongly in-
fuenced by wavelet theory, multiresolution analysis of a function relies on its
coding/decoding on a hierarchy of nested grids. Representations on consec-
utive grids provide means of compressing the solution as well as smoothness
indicators. In the framework of hyperbolic conservation laws, the combina-
tion of multiresolution with finite volume method appears first in (Harten,
1994). Since this pioneering work, the idea was profitably explored and developed in
many directions. The method was extended to Cartesian 2D grids in (Bihari
and Harten, 1997) for scalar problems and more recently in (Chiavassa and
Donat, 1999) for gas dynamics problems. In (Dahmen et al., 2000), a bior-
onthogonal wavelet formulation is used and curvilinear grids are treated. In (Abgrall
and Harten, 1994; Abgrall, 1997), and (Schröder-PandZ et al., 2000) the prob-
lem of unstructured grids for Finite Volume schemes is tackled while similar
MR concepts are applied to triangular cell centered schemes in (Cohen et al.,
2000). All these contributions have in common an inherent limitation which is
the actual encoding of the solution on a very fine uniform grid imposed by the
desired precision. We describe here a fully adaptive scheme where the solution
is computed on a time dependent adaptive grid where the fine resolution level
is used locally according to the local current smoothness of the solution. This
approach has been studied in details in (Cohen et al., 2001) for the 1D case,
both from the theoretical and numerical point of view in term of error analy-
sis. We summarize briefly the method in the case of triangular Finite Volume
and present the latest numerical developments consisting in 2D benchmarks
for the compressible Euler equations.

2 Multiresolution meets Finite Volume

In the context of Finite Volume schemes, the natural discretization of (1) uses
cell averages: the spatial domain is partitioned into “triangles” \( \Omega _{\gamma } \) \( \gamma \in \mathcal{S} \), with
\( \mathcal{S} \) the set of indices. One is interested in computing approximations \( u_{\gamma }^{n} \) of the
averages of \( u \) at discrete times \( n \Delta t \), over triangles \( \Omega _{\gamma } \). Applying the divergence
theorem to (1) leads to the Finite Volume scheme

\[
  u_{\gamma }^{n+1} = u_{\gamma }^{n} - \frac{\Delta t}{|\Omega _{\gamma }|} \sum_{\mu} |\Gamma _{\gamma ,\mu }| F_{\gamma ,\mu }^{n},
\]

(2)
where the numerical fluxes $F^n_{\gamma\mu}$ are approximations of $\overline{F}^n_{\gamma\mu}$ which denotes the average flux across interface $\Gamma_{\gamma\mu} = \Omega_\gamma \cap \Omega_\mu$ between time steps $n\Delta t$ and $(n+1)\Delta t$.

**Cell average multi-resolution.**

Finite Volume multi-resolution is based on nested finite volume discretizations. For $j = 0, 1, \cdots, J$, we are given regular disjoint partitions $(\Omega_\gamma)_{\gamma \in S_j}$ of $\mathbb{R}^2$ (or of a domain of interest) such that each $\Omega_\gamma$, with $\gamma = (k, j) \in S_j$, is the union of a finite number of cells $\Omega_{\mu}$, with $\mu = (k, j+1) \in S_{j+1}$. The case of triangle subdivisions is represented in figure 1. In order to keep track of the scale level $j$ associated to an index $\gamma = (k, j)$, we shall make use of the notation $|\gamma| := j$ if $\gamma \in S_j$.

Consider a vector $U_j := (u_\gamma)_{\gamma \in S_j}$ of discrete data on the grid $S_j$ representing the cell-averages $u_\gamma := |\Omega_\gamma|^{-1} \int_{\Omega_\gamma} u(x) dx$ of some function $u \in L^1(\mathbb{R}^2)$.

The **projection** operator $P^j_{j-1}$ maps $U_j$ to $U_{j-1}$. Since the partitions $S_j$ are nested we obtain easily the averages at the coarser level $u_\gamma = |\Omega_\gamma|^{-1} \sum_{\mu \in \Omega_\gamma} |\Omega_\mu| u_\mu$ with summation on $\mu \in \{ |\mu| = |\gamma| + 1, \Omega_\mu \subset \Omega_\gamma \}$. It is clear that one can derive $U_{j-1}, U_j, \cdots, U_0$ from the data $U_j$ by iterative application of the projection operators $P^j_{j-1}$.

The **prediction** operator $P^j_{j+1}$ maps $U_j$ to an approximation $\hat{U}_{j+1}$ of $U_{j+1}$. In contrast to the projection operator, there is an infinite number of choices for the definition of $P^j_{j+1}$. In all the numerical simulations we use the following linear reconstruction (3) fully detailed in Cohen et al. (2000),

\[
\begin{align*}
    \hat{u}^{j+1}_{00} &= u^j_0, \\
    \hat{u}^{j+1}_{01} &= u^j_0 + (u^j_2 + u^j_3 - 2u^j_1)/6, \\
    \hat{u}^{j+1}_{02} &= u^j_0 + (u^j_1 + u^j_3 - 2u^j_2)/6, \\
    \hat{u}^{j+1}_{03} &= u^j_0 + (u^j_1 + u^j_2 - 2u^j_3)/6.
\end{align*}
\]

(3)

It is suitably local (the stencil $\Sigma_\lambda$ of cells in $S_{\lambda+1}$ used to predict $u_\lambda$ contains only four triangles) and consistent ($P^j_{j+1}P^j_{j+1} = \text{Identity}$). We can define the prediction error at level $j$ as the differences between the exact and predicted values, $d_\mu := u_\mu - \hat{u}_\mu$. Using the consistency assumption, we define the detail vector $D_j = (d_\mu)_{\mu \in \Sigma_j}$, where $\Sigma_j$ is a selection of details such that there is a one to one correspondence between $U_j$ and $M_j = (U_0, D_1, D_2, \cdots, D_j)$. Using the
local structure of the projection and prediction operators, we can implement the **multiscale transformation** $M : U_J \mapsto M_J$, and its inverse $M^{-1}$ with optimal complexity $O(N_J)$, where $N_J := \#(S_J)$ represents the cardinality of $U_J$.

The main interest of decomposing $U_J$ into $M_J$ is that this new representation is often more appropriate for **data compression**. Given a set $\Lambda \subset \nabla^J$ of indices $\lambda$, we define a truncation operator $T_\Lambda$ acting on multiscale representations, that leaves unchanged the component $d_\lambda$ if $\lambda \in \Lambda$ and replaces it by 0 otherwise. In practice, we are typically interested in sets $\Lambda$ obtained by **thresholding**; given a set of level-dependent threshold $(\varepsilon_0, \varepsilon_1, \ldots, \varepsilon_J)$, we set $\Lambda = \Lambda(\varepsilon_0, \varepsilon_1, \ldots, \varepsilon_J) := \{ \lambda \text{ s.t. } |d_\lambda| \geq \varepsilon_\lambda \}$, and define the corresponding thresholding operator $T_\Lambda$. Applying $T_\Lambda$ on the multiscale decomposition of $U_J$ amounts in building a nonlinear approximation $A_\Lambda U_J$, where the operator $A_\Lambda$ is given by $A_\Lambda := M^{-1}T_\Lambda M$.

Both the **accuracy** and the **stability** of the thresholding scheme (3) have been extensively studied in (Cohen et al., 2000). The smoothness of the solution provides a bound for the details $|d_\lambda|$. A level dependent threshold $\varepsilon_j := 2^{2(j-1)}\varepsilon$, allows to control the effect of thresholding by the following estimate $\|U_J - A_\Lambda U_J\| \leq C\varepsilon$. It is very important that the set of preserved indices $\Lambda$ has a **tree structure** to ensure the isomorphism property of $M$ and furthermore some **gradedness** property in order to ensure good computational properties to the encoding/decoding algorithm. In practice it consists in adding to the tree all the triangles belonging to the reconstruction stencils of the triangles $\Omega_j$ already in the tree because $|d_\lambda| \geq 2^{2(j-1)}\varepsilon$ and we denote by $\Lambda_\varepsilon$ the resulting set of indices.

### 3 From Harten’s scheme to fully adaptive schemes

To link the multiresolution analysis with the Finite Volume scheme presented above, we rewrite (2) on the finest resolution level as

$$V_J^{n+1} = V_J^n - B_J^n := E_J V_J^n,$$

where $V_J^n := (u^n_\gamma)_{\gamma \in S_J}$ is the vector representing the Finite Volume numerical solution at time $n \Delta t$ and $B_J^n := (b^n_\gamma)_{\gamma \in S_J}$ with $b^n_\gamma := \frac{\Delta t}{|\Omega_\gamma|} \sum_{\mu} |\Gamma_{\gamma\mu}|F^n_{\gamma\mu}$ the numerical flux balance for the cell $\Omega_\gamma$. The increment $b^n_\gamma$ depends locally and nonlinearly on the numerical solution. If $\overline{U}_J$ is the cell average vector of the exact solution $u$ at time $n \Delta t$, the numerical error incurred using standard Finite Volume is $e_n := \|V_J^n - \overline{U}_J^n\|$. The goal of our multiresolution scheme is to compute a numerical solution $U_J^n$ with significant CPU and memory gain over the reference Finite Volume scheme, while the additional error $a_n := \|U_J^n - \overline{U}_J^n\|$ remains within a prescribed accuracy so that $\|U_J^n - V_J^n\| \approx e_n$. 

4
Both Harten’s multiresolution scheme and our fully adaptive scheme are based on the intuitive idea, introduced in (Harten, 1994), that the set of significant wavelet coefficients of the numerical solution evolves “slowly” from one time step to the other. More precisely, if $\Lambda^n$ is the graded tree obtained from the application of the thresholding to some numerical approximation $U^n_j$ of $\bar{U}^n$, one can enlarge $\Lambda^n$ into a graded tree $\Lambda^{n+1}$ which contains both $\Lambda^n$ and $\Lambda^{n+1}$ so that, if $U^{n+1}_j = E_j U^n_j$, we have $\|U^n_j - A_{\Lambda^n} U^n_j\| \leq C\varepsilon$ and $\|U^{n+1}_j - A_{\Lambda^{n+1}} U^{n+1}_j\| \leq C\varepsilon$.

This heuristics is justified rigorously in the uniform Cartesian grid case (Cohen et al., 2001) by imposing demanding rules to the enlarging procedure. However at this moment it has not been yet extended to the unstructured triangular grid and we therefore present numerical results obtained using the following enlarging procedure initially described by Harten:

- If $\Omega_\lambda$ is in $S_{\Lambda^n}$, then its neighbors at the same scale (all triangles sharing one or two vertices with $\Omega_\lambda$) are included in $S_{\Lambda^{n+1}}$.
- If $\Omega_\lambda$ is in $S_{\Lambda^n}$ and if $|d_\lambda| > 4\varepsilon$, then its four children are also included.

By (4), we see that Harten’s heuristics also implies $\|B^n_j - A_{\Lambda^{n+1}} B^n_j\| \leq C\varepsilon$, i.e., the flux balance is also well represented by $\tilde{\Lambda}^{n+1}$. Given a prescribed tolerance $\varepsilon > 0$, the scheme proposed in (Harten, 1994) consists in using the compressed vector $A_{\tilde{\Lambda}^{n+1}} B^n_j$ in place of $B^n_j$: the cell averages are now evolved according to $U^{n+1}_j = U^n_j - A_{\Lambda^{n+1}} B^n_j$. Of course, $B^n_j$ is now the numerical flux balance computed from $U^n_j$ which differs from $V^n_j$. As explained in the introduction, the goal of this modification is to save computational cost through a smaller number of numerical flux evaluations. Harten’s scheme achieves this up to a point since the solution is nevertheless computed everywhere on the uniform finest grid, although in smooth area, it is computed using interpolation of the solution computed on coarser grids. In contrast, the adaptive scheme presented below operates on a compressed representation of the numerical solution $U^n_j$: at time step $n\Delta t$, the non-zero detail coefficients in the multiscale decomposition of $U^n_j$ are confined to a graded tree $\Lambda_n \subset \nabla_j$, so that $U^n_j$ is exactly represented by its coefficients $(d^n_\lambda)_{\lambda \in \Lambda_n}$ or by its cell averages $(u^n_\lambda)_{\lambda \in S(\Lambda_n)}$ on the corresponding adaptive discretization.

Given $\Lambda_n$ and $U^n_j$ (represented by $(d^n_\lambda)_{\lambda \in \Lambda_n}$ or $(u^n_\lambda)_{\lambda \in S(\Lambda_n)}$), we derive $\Lambda_{n+1}$ and $U^{n+1}_j$ by the following steps:

- **Refinement.** A new set $\tilde{\Lambda}_{n+1}$ containing $\Lambda_n$ is constructed based on the magnitude of the coefficients $|d^n_\lambda|$, $\lambda \in \Lambda_n$ according to the growing rules described above. The vector $(d^n_\lambda)_{\lambda \in \Lambda_n}$ is extended by setting $d^n_\lambda = 0$ for $\lambda \in \tilde{\Lambda}_{n+1} \setminus \Lambda_n$. Applying $M^{-1}_{\Lambda_{n+1}}$, we derive the refined averages $(u^n_\lambda)_{\lambda \in S(\tilde{\Lambda}_{n+1})}$.

- **Computation.** A first numerical solution $\tilde{U}^{n+1}_j$ at time $(n+1)\Delta t$, discretized on $S(\tilde{\Lambda}_{n+1})$, is computed by $\tilde{u}^{n+1}_\lambda = u^n_\lambda - \hat{b}_\lambda$, $\lambda \in S(\tilde{\Lambda}_{n+1})$. The adaptive flux...
balance vector \( \tilde{\hat{u}}_{\Lambda}^{n+1} \lambda \in S(\tilde{\Lambda}_{n+1}) \) is directly computed from \( u_{\tilde{\Lambda}}^{n+1} \lambda \in S(\tilde{\Lambda}_{n+1}) \). In the 1D case we have devised a fast method to reconstruct locally the solution on the finest level. It is based on the uniformity of the fine grid and cannot be extended simply to the unstructured grid case. We use instead a linear reconstruction of the solution on the edges of the triangles where we need the fluxes \( F_{\tilde{\gamma}, \tilde{u}} \) entering in the evaluation of \( \tilde{\hat{u}}_{\Lambda} \) (see (Durlofsky et al., 1992)). This scheme is not strictly in the scope where the error bound \( a_n < C\varepsilon \) obtained in (Cohen et al., 2001) can be used even though the numerical experiments tend to show that it preserves the overall accuracy in a similar way.

- **Thresholding.** Applying \( M_{\tilde{\Lambda}_{n+1}} \) to \( (\tilde{\hat{u}}_{\Lambda}^{n+1}) \lambda \in S(\tilde{\Lambda}_{n+1}) \), we derive \( (d_{\Lambda}^{n+1}) \lambda \in \tilde{\Lambda}_{n+1} \). We define \( U_j^{n+1} \) by thresholding \( \tilde{U}_j^{n+1} \) according to \( U_j^{n+1} = A_j \tilde{U}_j^{n+1} \), and the new set \( \tilde{\Lambda}_{n+1} \subset \tilde{\Lambda}_{n+1} \) to be the corresponding set \( \Lambda_c \) of preserved coefficients.

4 **Numerical tests**

We consider the 2D Euler equations of gas dynamics

\[
\begin{align*}
\partial_t u + \partial_x f(u) + \partial_y g(u) &= 0, \quad \text{with} \quad u = (p, \rho u, \rho v, e)^T, \\
&= f(u) = u p + p(0, 1, 0, u)^T, \\
g(u) = v u + p(0, 0, 1, v)^T,
\end{align*}
\]

where \( \rho \) is the density of the gas, \( (u, v)^T \) its velocity, and \( p \) its pressure. The energy \( e = p/\gamma + \rho(u^2 + v^2)/2 \) with \( \gamma = 1.4 \). We first study the reflection of an oblique shock on the lower side of a rectangular domain: \( \{ 0 \leq x \leq 4.129, 0 \leq y \leq 1 \} \). The initial flow is given by

\( \rho = 1, \quad u = 2.9, \quad v = 0 \) and \( p = 1/1.4 \).

The boundary conditions are:

- inflow on the side \( x = 0 \) (those of the initial flow).
- outflow on the side \( x = 4.129 \).
- reflection on the lower side \( y = 0 \).
Fig. 3. Hybrid grid at time t=4, four levels at most, \(\varepsilon = 0.1\), 12349 triangles.

- fixed values on the upper side \(y = 1\) (those of the exact solution):
  \(p = 1.7,\quad u = 2.61932,\quad v = -0.506339\quad \text{and}\quad p = 1.52824\)

The exact solution is formed by an incident shock with angle 29° and a reflected
shock with angle 23.28°. The solution after the second shock is:

\(p = 2.68732,\quad u = 2.40148,\quad v = 0\quad \text{and}\quad p = 2.93413\)

The coarsest grid is composed of \(N_0 = 820\) triangles. Four levels at most are
used for the computations summarized in figure 2 in terms of memory, CPU
and relative L1-error computed with respect to the Finite Volume solution on
the finest grid which has 69,700 triangles. The three curves exhibit the expected
trends in the 1st order Roe scheme case. The error curve in the second order
flux case does not decay as well as in the 1st order case for small \(\varepsilon\) values.
The reference solution is closer to the piecewise constant exact solution in that
case which makes the thresholding of details less sensitive to \(\varepsilon\).

An example of hybrid grid is represented in figure 3 which shows that the
shock is well located. More details on this numerical simulation are available
on the web page. The second experiment simulates a supersonic flow past
a double ellipse (see Arminjon et al. (1997)) starting from a constant field
\(p = 1,\quad p = 1/5.6\) and a unit velocity with an angle of attack 20° with the
horizontal and \(M_\infty = 2\). In figure 4 we show the coarse grid and the hybrid
grid obtained with a threshold at time 0.1. The triangles with two vertices on
the ellipse boundary are always subdivided down to the finest level. At each
level the mid point for the two vertices lying on the ellipse is projected on the
ellipse. Since this contradicts the consistency assumption of the prediction,
the thresholding is deactivated on these cells. Nevertheless, the memory gain
is 7.28 and the CPU gain 3.2 with comparison to the Finite Volume solution
computed on the uniform finest grid.

5 Conclusion

The good performance of the numerical tests illustrates the feasibility and in-
terest of the method. However this also indicates the need for a better underlying
Finite Volume scheme, which would better compare with the uniform fine
resolution in the smooth areas. Work in this direction is currently in progress.
References

Multiresolution analysis on triangles: application to conservation laws

A. Cohen, S.M. Kaber, M. Postel
Laboratoire d’analyse numérique
Université Pierre et Marie Curie (Paris 6), France
http://www.ann.jussieu.fr

Abstract A multiresolution algorithm is coupled with a finite volume scheme to solve scalar bidimensional conservation laws. The originality resides in the adaptivity of the multiresolution decomposition, which takes into account the possible appearance of discontinuities and their displacement. Numerical simulations on triangular meshes point out the advantages of the method in terms of CPU and memory costs.

Key Words: Multiresolution - finite volumes - adaptive scheme.

1. Introduction

We are interested in this work in solving conservation laws on polygonal domains. It is well known that such equations can develop localized discontinuities in finite time. In such areas where the solution is not smooth a fine resolution is necessary and furthermore, higher order schemes for flux computations are necessarily nonlinear - including for instance ENO reconstruction. These are very costly techniques and it seems reasonable to use all available information on the local smoothness of the solution to decide whether they should be used or not. This can be done within multiresolution framework. The original idea of combining the advantages of this method - data compression, smoothness indicators - into conservation laws solvers in order to reduce the number of flux computations is due to Harten [Har94]. Initially implemented in one dimension, it was then extended to bidimensional cartesian grids, and smooth deformations of rectangular grids, (see references in [CDKP99]). Preliminary approach for unstructured grid is explained in [Abg97]. In the case of triangular meshes, which are more flexible in modeling complex geometries, a complete implementation with a detailed analysis of the encoding/decoding algorithm can be found in [KP99, CDKP99].

In this former approach, the solution is represented everywhere on a uniform fine grid and the multiscale analysis is used to speed up the flux computations in the smooth areas. What we propose here is a fully adaptive scheme which
make use of the compressed solution - encoded at each time step by its most significant coefficients. This approach gives way to new difficulties, for instance in the analysis of the stability and precision. These points have been studied in detail in the one dimension case in [CKP99] and are currently under investigation in the triangular meshes case.

The outline of the paper is as follows: we recall in section 2 the multiresolution and the finite volume algorithms for triangular meshes which are then combined to produce the adaptive scheme presented in section 3. Numerical simulations and analysis of performances on test cases follows in section 4.

2. Multiresolution analysis and finite volumes

We briefly describe a multiscale transformation of a function described by its mean - and not point wise as usual - values on a triangular mesh. The equation is defined on a polygonal domain Ω which is discretized by a coarse grid Ω^{0} made of N_{0} triangles. A hierarchy of nested grids Ω^\ell (0 ≤ \ell ≤ L) is built. The grid Ω^{\ell+1} is obtained by dividing each triangle of Ω^\ell into four smaller triangles by joining the three edges midpoints. Triangles from the level \ell are denoted by T_{k}^{\ell} for 1 ≤ k ≤ N_{\ell} = 4^{\ell}N_{0}. We denote by \bar{u}_{k}^{\ell} the average of a function u on the triangle T_{k}^{\ell} and \bar{u}^{\ell} = (\bar{u}_{k}^{\ell})_{k=1}^{N_{\ell}}. The mean values \bar{u}^{\ell+1} of a function u being given on the Ω^{\ell+1} grid, the mean values on the coarser grid Ω^\ell can be computed by

\[ \bar{u}_{i}^{\ell} = \frac{1}{|T_{i}^{\ell}|} \sum_{j=0}^{3} P_{i,j}^{\ell+1} \bar{u}_{i,j}^{\ell+1}, \]

where T_{i,j}^{\ell+1} are the four triangles of Ω^{\ell+1} obtained by subdividing the triangle T_{i}^{\ell}. The center triangle is denoted by T_{i}^{\ell+1}. The relation (1) defines a projection operator P_{i}^{\ell+1} from the resolution level \ell + 1 on the level \ell by P_{i}^{\ell+1} \bar{u}^{\ell+1} = \bar{u}^{\ell}. Multiscale decomposition relies on the existence of this operator as well as a prediction operator Q_{i}^{\ell+1} from the resolution level \ell to the level \ell + 1. They should be compatible in the sense that P_{i}^{\ell+1}Q_{i}^{\ell+1} = Id. The predicted values \bar{u}^{\ell+1} = Q_{i}^{\ell+1} \bar{u}^{\ell} are approximations of the exact mean values \bar{u}^{\ell+1} obtained by local linear combinations of values at level \ell. Details on every triangle are defined as the differences between exact and predicted values \bar{u}_{i,j}^{\ell+1} = \bar{u}_{i,j}^{\ell+1} - \bar{u}_{i,j}^{\ell+1}, for j = 1, \ldots, 3. The multiscale representation of \bar{u}^{L} is denoted by \bar{u}_{MR} = (\bar{u}_{i,j}^{0}, \ldots, \bar{u}_{i,j}^{L-1}). Going from \bar{u}^{L} to \bar{u}_{MR} and reverse is achieved through wavelet-type algorithms described in [CDKP99]. The two types of data representation have the same memory requirement. However, in the multiscale representation case, the amount of data can be reduced by *thresholding* the sufficiently small details. More precisely this consists in setting \bar{u}_{i,j}^{\ell} = 0 wherever ∥\bar{u}_{i,j}^{\ell}\parallel is smaller than a level dependent tolerance \varepsilon_{\ell}. We refer to [CDKP99] for justification. In the same reference, the stability and
convergence of a class of reconstruction operators are studied. This leads to the following optimal choice for $Q_t^{L+1}$

\[
\begin{align*}
\tau_{t+1}^L & = \tau_{t}^L, \\
\tau_{t+1}^{L+1} & = \tau_{t+1}^{L+1} - (\bar{\tau}_{t+1}^L + \bar{\tau}_{t+1}^L - 2\bar{\tau}_{t+1}^L)/6, \\
\tau_{t+1}^{L+2} & = \tau_{t+1}^{L+2} - (\bar{\tau}_{t+1}^L + \bar{\tau}_{t+1}^L - 2\bar{\tau}_{t+1}^L)/6, \\
\tau_{t+1}^{L+3} & = \tau_{t+1}^{L+3} - (\bar{\tau}_{t+1}^L + \bar{\tau}_{t+1}^L - 2\bar{\tau}_{t+1}^L)/6.
\end{align*}
\]

(2)

We now recall how the numerical treatment of the 2D scalar conservation law

\[ \partial_t u + \nabla \cdot \mathbf{f}(u) = 0, \]  

(3)
can be improved within the multiresolution framework. The unknown $u(x, y, t)$ is defined on a polygonal set $\Omega$, for $t > 0$. An initial condition $u(x, y, 0) = u_0(x, y)$ is imposed as well as boundary conditions on the edges of $\Omega$. We first consider that equation (3) is solved by a finite volumes scheme on the fine grid $\Omega^L$. The costly step of this scheme is the flux evaluation $\mathbf{F}_k^{L+1} \approx \frac{1}{|\Gamma_k^L|} \int_{\Gamma_k^L} \nabla \cdot \mathbf{f}(\mathbf{u}) \, dx dy$. Since the solution can present discontinuities - even with smooth initial data - the computations may have to be locally very precise. On the other hand, the pointwise evaluation of a function known by its mean values requires a reconstruction step (using the operator $\mathcal{R}$ in the flux evaluation, not to be mistaken with the prediction $Q_t^{L+1}$ defined by (2)). This step can also be very costly if ENO techniques are used for instance. It is therefore crucial to limit the number of these flux computations and this is where multiscale analysis comes into play. In Harten’s framework, the details provided by the multiscale decomposition of the solution are used as criteria to choose the flux computation method as follows. Starting from the coarsest level where fluxes on all triangle edges are precisely computed. The levels are explored in turn: wherever details are large, fluxes are computed using precise and costly numerical flux approximations $\mathbf{F}_k^L = \frac{1}{|\Gamma_k^L|} \sum_j \mathbf{F}_{k,j}^L$, where

\[ |\Gamma_{k,j}^L| \mathbf{F}_{k,j}^L := \sum_{\Gamma_h \subset \Gamma_{k,j}^L} |\Gamma_h^L| \mathbf{F}_{h,m}^L, \]  

with $\mathbf{F}_{h,m}^L$ denoting the numerical flux across the edge $\Gamma_h^L$ (of length $|\Gamma_h^L|$) between two triangles belonging to the finest grid $\Omega^L$. Elsewhere, the mean values of flux divergences are computed by interpolating the same quantities on the immediately coarser level, using the prediction operator $\Theta_{L+1}$. This algorithm was successfully implemented in [KP99, CDKP99]. It leads to a CPU reduction by a factor 2 to 3 depending on the test case. Its precision is governed by the precision of the underlying finite volume scheme on $\Omega^L$ since at each time step the time evolution of the solution is performed on this grid. Therefore the gain in CPU time remains linked to this discretization,
and there is no memory saving. A way to overcome these two limitations is provided by a fully adaptive scheme in which the solution is computed on a nonuniform time dependent grid. In the next section we modified the initial scheme in order to compute the solution on such a hybrid grid, fine where large fluctuations are foreseen and coarse in the smooth areas. We refer to [CKMP99] for a detailed analysis of the algorithm in the one dimensional case. We are principally interested here in showing the potential of the method in the case of multidimensional grids - and more precisely triangular ones.

3. Fully adaptive schemes

We introduce a hybrid grid $\mathcal{H}_n^0 \subset \cup_\ell \Omega^\ell$ defined at time step $t_n$ by selecting subsets out of each $\Omega^\ell$. We will show how this grid is built in such a way that it reflects the smoothness of the solution. Since this regularity is time dependent, the hybrid grid will have to evolve in time, this will also be described. Eventually we will explain how the finite volume scheme is performed on such a grid, so that both the solution mean values and the details in the multiresolution analysis evolve correctly in time.

We first describe how $\mathcal{H}_n^0$ is built using the initial data $u_0$. It is initially set equal to $\Omega^0$. Until the maximum level of resolution if reached, each triangle of $\mathcal{H}_n^0$ is tested to decide whether its subdivisions should be added to $\mathcal{H}_n^0$. The criterion is the comparison of the mean value of the solution on the four subdivisions with the values predicted by interpolation $Q_{\ell+1}^\ell$. A smoothness indicator $R_{\ell}^{k,j}$, associated to a triangle $T_{k,j}^\ell$, is activated ($R_k^\ell = 1$) if one of the four details $d_{k,j}^\ell$ is greater in absolute value than the threshold tolerance $\varepsilon_\ell$. The triangles whose indicators are activated are actually divided into four triangles which are themselves included into the computing grid $\mathcal{H}_n^0$. Another requirement is to preserve the grid nestedness. As a corollary we must always be able to interpolate at the finer level. This has two consequences:

- If a triangle $T_{k,j}^\ell$ is divided because $R_k^\ell = 1$, all its neighbors on the same level $\ell$ are included in $\mathcal{H}_n^0$, even if their smoothness indicators are not activated.
- An auxiliary grid $\mathcal{G}_n^0$ containing $\mathcal{H}_n^0$ is defined: If a triangle belongs to $\mathcal{H}_n^0$ for one of the two previous reasons, then all its neighbors at level $\ell$ belong to $\mathcal{G}_n^0$. On triangles belonging to $\mathcal{G}_n^0 \setminus \mathcal{H}_n^0$ no actual computation takes place and the mean values of the solution will be estimated by $Q_{\ell+1}^\ell$.

When the initial data is given by its dual multiresolution representation, pathological cases with non zero details only on the fine level can occur. To handle such cases, all the triangles should be actually divided up to the finest level, even if the details are negligible at a coarser level, and the thresholding should be done iteratively starting from the finest level. This is in practice very difficult to realize without giving up all the advantage of the multiresolution in terms of memory savings. In the test cases we have studied so far, the initial solution is given by an explicit expression, which ensures that the tree structure of $\mathcal{H}_n^0$ is achieved by the previous algorithm which explores and refines the levels.
starting from the coarsest.

The hybrid grid may have to be modified at each time step in order to follow the discontinuities propagation and capture shocks formation. To this effect, we define a larger grid $\mathcal{H}_{e+1}^n$, containing $\mathcal{H}_e^n$, whose time evolution can be predicted. We use here the hyperbolicity of the problem ensuring that if the CFL condition is respected the discontinuities do not propagate further than one space step in one time step. To foresee possible propagation in any direction the term of neighbors of a triangle is extended to all triangles sharing a vertices with it, and not only the three triangles having a common edge and participating in the interpolation scheme $\mathcal{Q}_{e+1}$. A corresponding grid $\mathcal{G}_{e+1}^n$ containing $\mathcal{H}_{e+1}^n$ as well as all its neighbors at each level is also defined. At each time step the mean values of the solution on the coarsest grid and the details corresponding to triangles in $\mathcal{H}_e^n$ are modified (as detailed in algorithm 1). Whenever one of the four details of a triangle of $\mathcal{H}_e^n$ is higher than the level tolerance, its smoothness indicator and that of all its neighbors is activated. If a detail is higher than a still larger tolerance, the corresponding indicators at the finer level are also activated. These operation can make triangles previously in $\mathcal{G}_{e}^n \setminus \mathcal{H}_e^n$ part of $\mathcal{H}_{e+1}^n$ and even create new subdivisions at the finer level. The nestedness of the new grid $\mathcal{H}_{e+1}^n$ is ensured as in the initial step - which can induce the creation of subdivisions in $\mathcal{G}_{e+1}^n$. Eventually some triangles of $\mathcal{G}_{e}^n \setminus \mathcal{H}_e^n$ may now be useless and therefore removed.

An important difference with Harten's algorithm is that the computing grid $\mathcal{H}_e^n$ is incomplete. In other words, the depth of resolution is spatially variable and defined for a triangle $T_e^l$ of level $l$ by a local depth $\ell$. This has direct consequences on the precision of scheme. In both approaches the precise numerical flux evaluations are performed only on the interfaces of the hybrid grid. In Harten's case the solution is everywhere available on the finest grid so the maximum precision can be achieved. In the adaptive case, the solution could theoretically be computed down to the finer level by applying the $\mathcal{Q}_{e+1}^n$ as many times as necessary starting from $\ell$ where it is actually known. This would be completely inefficient in practice since it would destroy all the benefits of the multiresolution. On the other hand, brutal use of the available mean values on the hybrid grid to compute the fluxes leads to an order of approximation governed by the coarser level. This problem is explained in details in the one dimension case in [CKMP99] and we simply summarize here the main conclusions. We have so far two ways to tackle this difficulty:

- In one dimension it is possible to reconstruct directly - that is linearly - mean values at the finest level from values at a not immediately coarser level - as soon as the details are null on a sufficient number of consecutive intervals on this coarse level (four intervals in the case of linear reconstruction). Taking into account the areas of high gradient where actual decoding must be performed leads to a complexity in $N \log(N)$ where $N$ is the cardinal of $\mathcal{H}_e^n$.

- The second method consists in using a higher order ENO type scheme to compute the fluxes at points where the solution is known by its mean values on
a coarse or intermediate grid level. In that case the complexity is that of the hybrid grid multiplied by a fixed coefficient which depends on the complexity of the ENO reconstruction. The precision is that of the coarse grid to a power related to the ENO reconstruction order.

From the parameter study performed in [CKMP99], it seems that the first method is the most efficient one if we compare the performances for a given accuracy. However its extension to the triangular grid case seems more complicated. In this preliminary implementation we have used the same ENO type reconstruction as in [CDKP99]. We summarize in the following adaptive algorithm (1), how the finite volumes scheme is used to compute the time evolution of the solution and the details of the multiresolution on the hybrid computing grid \( \mathcal{H}_e \).

**Algorithm 1** Finite volumes + adaptative multiscale

- Initialization of \( \mathcal{H}_e^0 \) and \( \mathbf{u}_M^0 \)
- Loop on time step \( n = 0, 1, ... 

\[
\begin{align*}
\Delta t & \quad \text{criterion} \\
\end{align*}
\]

4. Numerical experiments

To illustrate the performances of the algorithm we first present the test case of a linear flux on an initial discontinuous solution. We choose the function

\[
u_0(x, y) = 1 \quad \text{if} \quad \sqrt{x^2 + y^2} < 0.1 \quad \text{and zero elsewhere on the unit square with periodic boundary conditions.}
\]

Performances in term of CPU time, memory requirements and precision are evaluated for the adaptive multiresolution scheme and the reference finite volumes scheme on the finest level. The numerical fluxes are computed with the second order ENO method already implemented in [KP99, CDKP99]. It requires a small CFL number chosen here equal to 0.2 on the finest grid. There are 50 triangles on the coarsest level. We first show results corresponding to a multiresolution decomposition on a maximum of 5 levels. In this case the finest grid on which the finite volumes reference solution is computed counts 12800 triangles. The direction of propagation is parallel to
the 0z axis and the solution is computed over one period of time. Figure 1. shows the initial solution and the solution after one period computed with the adaptive scheme on a maximum of five levels with a total of 1209 triangles. The shape of the solution is well preserved, and the finer levels are used only in the vicinity of the discontinuity. On the next group of figures 2, the performances of the adaptive scheme are displayed for different maximum number of levels. For each value of this parameter $L$, the standard finite volume solution on the finest grid - level $L$ - is computed. CPU time and memory requirements along with the error with the exact solution in the $L_1$ norm can be compared. The precision of the adaptive scheme in this case is very satisfying since the two error curves are always close to each other. Actually, in this test case, the source of possible errors is the discontinuity and in its vicinity both schemes are similarly discretized. As far as the memory requirement are concerned the advantages of the adaptive case are indisputable. We are actually restricted to a few number of levels because the finite volume scheme will not run on our work station for a grid finer than the 5th level ($L = 4$). We can predict that for 204800 triangles it would require about 13 hours of CPU. In this limit case $L = 6$, the adaptive scheme actually discretizes the solution on 2198 triangles, among which only 1228 belong to the finest grid and takes only 1.5 hour to run. One remarks that the CPU time does not depend linearly upon the number of triangles because some expensive overhead computing must be done which is inherent to the multiscale decomposition like the creation and the deletion of triangles. The adaptive scheme is nevertheless much faster than the standard finite volume scheme, all the more so as the number of levels in the multiresolution increases.

5. Conclusions

A fully adaptive scheme based on the multiresolution decomposition of the solution is proposed to improve the performances of a standard finite volume algorithm. The mean values of the solution are computed on a time varying grid made of cells belonging to different levels of decomposition according to the local smoothness. The preliminary numerical tests which are presented exhibit very good performances in term of CPU and memory savings. The immediate future works consists in developing the vector case algorithm necessary to handle gas dynamics. From a more theoretical point of view, we are currently studying the convergence of the scheme.

Bibliography

Figure 1: Multiresolution solution on five levels at time $t=0$ and $t=1$ (one period).

Figure 2: Comparison of CPU times, memory occupation and precision for finite volumes and multiresolution algorithms.


Multiresolution analysis on triangles: application to gas dynamics

Albert Cohen, Sidi Mahmoud Kaber, Marie Postel

Abstract. Multiresolution analysis is used to improve the performances of a Finite Volumes scheme. Two schemes coupling Multiresolution and Finite Volumes are presented. One is a generalisation of Harten’s original scheme for triangles. The other scheme is fully adaptive in the sense that at a given time, the solution is represented in a compressed form by a set of significant wavelets coefficients. The two schemes are applied to solve the Euler’s system of gas dynamics.

1. Introduction

The main difficulties in solving nonlinear conservation laws arise from the discontinuities which the solutions of such equations may develop in finite time. The fact that the position of these discontinuities varies generally with time is of course an additional difficulty. In order to represent accurately the discontinuous solutions it is necessary to discretize the computational domain on very fine grids. Furthermore to ensure efficiency and high precision on these fine grids costly nonlinear solvers are used. In many problems the discontinuities are localized in only a small region of the total domain. The fine discretization which is necessary in the vicinity of the discontinuities is in fact a luxury everywhere else. The costly numerical fluxes could be advantageously replaced by simpler solvers wherever the solution is smooth enough. A. Harten proposed to use multiresolution analysis (MR) in order to speed up Finite Volume (FV) schemes. See [H] and the references therein, in particular [A] for applications to triangular meshes.

Let us detail the original strategy proposed in [H]. At the start one is given a FV scheme associated to a grid $\Omega^L$ which is the finest one in a hierarchy of nested grids $\Omega^{\ell}$ for $\ell = 0, \ldots, L$. At time $t_n$ the approximate solution is represented by its averages $(\bar{u}_k^n)$ on the various cells of $\Omega^L$. The values $(\bar{u}_k^{n+1})_k$ are evaluated by the FV scheme through the computations of the fluxes on $\Gamma^L$, the set of all interfaces between the cells of $\Omega^L$. The basic idea is to use a wavelet-like multiscale decomposition of the solution at time $t_n$ as a smoothness indicator in order to reduce the computations of the fluxes on $\Gamma^L$. In the regions where the details defined below by (2)- are small (i.e. below some preassigned threshold), the flux is assumed to
be smooth enough so that we can replace its exact evaluation on the finest level \( L \) by an interpolation from its values on the coarser levels (see Algorithm 2). This saves CPU since the precise computation of the flux is expensive, while the interpolation from coarse to fine meshes is cheap.

In section 2, we define a MR transform suited for triangulations. In section 3, we combine it with a FV scheme to obtain an extension of Harten’s MR scheme to triangles. In section 4, we present an adaptive scheme that takes full advantage of the multiscale decomposition of the solution.

2. Multiresolution analysis (MR) on triangles

In [CDKP], we have defined a multiscale method adapted to cell-averages given on triangular meshes. Starting from a coarse triangulation \( \Omega_0^0 \), we define a hierarchy of nested grids \( \Omega^\ell (1 \leq \ell \leq L) \) by dividing each triangle \( T_k^\ell \) of \( \Omega^\ell \) into 4 triangles \( (T_{k,j}^{\ell+1})_{j=0}^3 \) of \( \Omega^{\ell+1} \). The mean value of the function \( u \) on the triangle \( T_k^\ell \) is denoted by \( \bar{u}_k^\ell = \frac{1}{|T_k^\ell|} \int_{T_k^\ell} u \) and the array of all \( (\bar{u}_k^\ell)_{k=1,\ldots,N_t} \) is denoted \( \bar{u}^\ell \).

2.1. The Multiscale decomposition

Two fundamental operators are defined.

- A projection operator \( \mathcal{P}^\ell_{\ell+1} \) from the resolution level \( \ell + 1 \) to the coarser one \( \ell \), that maps \( \bar{u}^{\ell+1} \) onto \( \bar{u}^\ell \). This operator is simply defined by

\[
\bar{u}^\ell = \frac{1}{|T_i^\ell|} \sum_{j=0}^3 \left| P_{i,j}^{\ell+1} \right| \bar{u}_{i,j}^{\ell+1}.
\]

- A prediction or reconstruction operator \( \mathcal{Q}^\ell_{\ell+1} \) from the resolution level \( \ell \) to the finer level \( \ell + 1 \). This prediction operator should satisfy the “consistency condition” \( \mathcal{P}^{\ell+1}_{\ell+1} \mathcal{Q}^\ell_{\ell+1} = Id \). That is to say

\[
\sum_{j=0}^3 \left| P_{i,j}^{\ell+1} \right| \bar{u}_{i,j}^{\ell+1} = \left| T_i^\ell \right| \bar{u}_{i}^\ell, \tag{1}
\]

where \( \bar{u}_{i,j}^{\ell+1} \) are the values predicted by \( \mathcal{Q}_{\ell+1}^{\ell+1} \). We only consider here reconstruction operators for which the predicted values \( \bar{u}_{i,j}^{\ell+1} \) are given by a local linear combination of some immediately coarser values; \( \bar{u}_{i,j}^{\ell+1} = \sum_{m \in \Omega(j)} a_m \bar{u}_m^\ell \), where \( \Omega(j) \) corresponds to a neighborhood (in \( \Omega^\ell \)) of \( T_j^\ell \). Related to the prediction operator are the prediction errors or details

\[
d_{i,j}^\ell = \bar{u}_{i,j}^{\ell+1} - \bar{u}_{i,j}^{\ell} \quad \text{for} \quad j = 0, \ldots, 3. \tag{2}
\]

Note that (1) implies a linear dependence for four details associated to the same triangle. This allows to compute only three details per triangle, for example \( d_{i,j}^\ell := \left(d_{i,j}^\ell\right)^3_{j=0} \).
Remark. The details can be written as \( d_{j,k}^l = \langle u, \psi_{l,k}^j \rangle = \int\int u \psi_{l,k}^j \), where the wavelet \( \psi_{l,k}^j \) is defined by \( \psi_{l,k}^j = \varphi_{l+1,k} - \sum_{m \in \mathbb{N} \backslash \{0\}} a_m \varphi_{l,k}^m \), and \( \varphi_{l,k}^j = \frac{1}{\sqrt{2^l}} \chi_{2^l \cdot 2^j} \) are the scaling functions. We will refer to the details as wavelets coefficients.

Denoting by \( d^i \) the array of all the \( d_{j,k}^l \), the two representations

- \( \tilde{u}^L \) (all the averages on the finest level)
- \( \tilde{u}^0 \cup \{ d_{j,k}^l, \ell = 0, \ldots, L - 1 \} \) (averages on the coarsest level + details)

are equivalent in so far as they provide the same information and use the same memory requirement. Using the local structure of the projection and prediction operators, one can easily implement the multiscale transformation \( \Phi: \tilde{u}^L \mapsto (\tilde{u}^0, d_{L-1}^L, \ldots, d_{0}^{L-1}) \), and it is inverse \( \Phi^{-1} \) with optimal complexity.

2.2. Compression

One of the main interest of decomposing \( \tilde{u}^L \) into \( \Phi \tilde{u}^L \) is that this new representation is more appropriate for data compression. Let us define

\( \Lambda := \Lambda(\varepsilon_0, \varepsilon_1, \ldots, \varepsilon_{L-1}) = \{ \lambda = (\ell, j, k) \text{ s.t. } |d_{\lambda}^l| \geq \varepsilon_\ell \} \)

and the associated truncation operator \( \Xi_{\Lambda} \) that leaves unchanged the component \( d_{\lambda}^l \) if \( \lambda \in \Lambda \) and replaces it by 0 otherwise. In all the computations performed, the level depending threshold parameters are \( \varepsilon_\ell := \frac{2^d}{2^L - \ell} \varepsilon \) for a fixed \( \varepsilon \) (here \( d \) is the space dimension). The parameter \( \varepsilon \) controls the truncation error resulting from the thresholding [CDKP]. Applying \( \Xi_{\Lambda} \) on the multiscale decomposition of \( \tilde{u}^L \) amounts in building a nonlinear approximation \( \Phi^{-1} \Phi \tilde{u}^L \) in which the details in the finest scales are concentrated near the isolated singularities. Such compression properties are however related to additional assumptions on the prediction operator: polynomial accuracy and multiscale stability. Consult [CDKP, CKMP] and the references therein for a detailed presentation of this problem.

2.3. Smoothness indicators

The prediction operator \( \Omega_{k+1}^l \) has polynomial exactness of order \( n \) if for all \( u \in P_n \) (polynomials of degree \( n \)), we have \( \tilde{u}^L = \tilde{u}^\ell \), i.e. \( d_{L-1}^L = 0 \) which expresses the orthogonality of \( \psi_{l,k}^j \) with \( P_n \). Therefore, if \( u \) has local smoothness \( C^s \), \( s > 0 \) (say in a neighborhood of \( T_{k,j}^l \)), then \( |d_{k,j}^l| \leq C 2^{-(\ell \min(s,n+1))} \). In order to justify the use of the details as smoothness indicators, we need a converse property: small details should indicate that the encoded function is locally smooth. We also need some stability in the sense that we can control in some prescribed norm the perturbation of \( u \) resulting from thresholding the small details, see [CDKP].

3. Harten’s algorithm

We now present a generalisation for triangles of the Harten’s MR scheme. For cartesian grids and Finite Differences, see [CD]. The starting point is a classical FV scheme for solving the conservation law \( \partial_t u + \text{div} f(u) = 0 \) on the finest grid \( \Omega^L \). Here \( u(x, y, t) \) is a scalar function, but generalisation to systems is straightforward.
The FV scheme computes at time $t_n$ approximate averages $\bar{u}_k^{L,n} \simeq \mathfrak{A}(T_k^{L})u(., t_n)$ of the solution $u$ by Algorithm 1.

| Step 1. Reconstruction: Use a reconstruction operator $\mathfrak{R}$ to obtain point values from cell averages.  
| Step 2. Flux evaluation: compute $\mathfrak{D}_k^{L,n}$, an approximation of $\mathfrak{D}_k^{L,n} = \mathfrak{A}(T_k^{L})\text{div}f(\mathfrak{R})$.  
| Step 3. Advance in time: $\bar{u}_k^{L,n+1} = \bar{u}_k^{L,n} - \Delta t \mathfrak{D}_k^{L,n}, k = 1, \ldots, N_L$. |

We only discuss here the Step 2 of this algorithm. The flux evaluations are based on the remark that, by the divergence theorem

$$\mathfrak{D}_k^{L,n} = \frac{1}{|T_k^{L}|} \sum_j |\Gamma_{k,j}^L| f_{k,j}^{L,n}, \quad \text{with} \quad f_{k,j}^{L,n} = \frac{1}{|\Gamma_{k,j}^L|} \int_{\Gamma_{k,j}^L} f(\mathfrak{R}(\sigma)) \cdot n_{k,j} d\sigma. \quad (3)$$

Here $\Gamma_{k,j}^L$ denotes interfaces between $T_k^{L}$ and $T_j^{L}$. Therefore, $\mathfrak{D}_k^{L,n}$ can be computed by applying (3) to some approximations $f_{k,j}^{L,n}$ of $f_{k,j}$.  

### 3.1. Multilevel computation of the fluxes

We now explain how the multiscale decomposition of the solution is used to speed up the flux evaluation, through a modification in Step 2 of the FV algorithm. To this effect we define $\mathfrak{D}_k^{\ell,n} = \mathfrak{A}(T_k^{\ell})\text{div}f(\mathfrak{R})$ for $0 \leq \ell \leq L$. These mean values can again be computed as in (3) and approximated by

$$\mathfrak{D}_k^{\ell,n} = \frac{1}{|T_k^{\ell}|} \sum_j |\Gamma_{k,j}^{\ell}| f_{k,j}^{\ell,n}, \quad \text{where} \quad f_{k,j}^{\ell,n} = \frac{1}{|\Gamma_{k,j}^{\ell}|} \sum_{m \in \Gamma_{k,j}^{\ell}} |\Gamma_m^{L}| f_{m}^{L,n}. \quad (4)$$

The MR representation of the solution is used to avoid the evaluation of $\mathfrak{D}_k^{L,n}$ on the finest grid wherever it is possible. In the regions (let say around the triangle $T_k^{L}$) where the solution is smooth, $\mathfrak{D}_k^{L,n}$ is computed by interpolation of the $\mathfrak{D}_j^{\ell,n}$’s corresponding to some coarser levels $\ell < L$. In the region where the solution is not smooth enough, $\mathfrak{D}_k^{L,n}$ will be computed accurately using a precise (and expensive) numerical flux. Let $\Omega^{\ell,n+1}$ denote a subset of $\cup_{\ell=0}^{L} \Omega^{\ell}$ formed by all the triangles $T_k^{\ell}$ over which the numerical solution is not regular enough. This hybrid grid (built by Algorithm 3) is the key ingredient of the following algorithm which is an MR version of Step 2 of Algorithm 1.
Algorithm 2: Multilevel computation of the fluxes.
Step 2-1. Compute the set $\Omega^{r,n+1}$ (see Algorithm 5).
Step 2-2. Compute the $\bar{S}^0$'s on the coarsest grid $\Omega^0$ using (4).
Step 2-3. For $\ell = 1 \land L$, compute the approximate $\bar{S}^\ell$'s by
    If $d_{k,j}^\ell \in \Omega^{r,n+1}$ Then
        $\bar{S}^\ell_{k,j}$ is accurately computed using (4) as in Step 2-2.
    Else
        $\bar{S}^\ell_{k,j}$ is approximately computed by interpolation of
        the values $\bar{S}^{\ell-1}$, using the prediction operator $\Omega^{\ell-1}$.

The first test in the algorithm takes into account the propagation of information (which is limited by the CFL condition). The second test takes into account the possible appearance of discontinuities.

Algorithm 3: The hybrid grid.
Step 2-1-1. Initialize $\Omega^{r,n+1} = \Omega^0$
Step 2-1-2. For $\ell = L - 1 \land 0$
    For $k = 1, \ldots, N_\ell$
        If $|u_{k,j}^\ell| \geq \varepsilon_\ell$ for one $j$ Then
            add $T_{k,j}^\ell$ to $\Omega^{r,n+1}$ if $T_{k,j}^\ell$ shares a vertex with $T_k^\ell$.
        If $|u_{k,j}^\ell| \geq 8\varepsilon_\ell$ Then
            add $T_{k,j}^{\ell+1}$ to $\Omega^{r,n+1}$.

3.2. Numerical tests (2D Euler equations of gas dynamics)
In all the computations, we have used the following reconstruction operator (see the figure 1 for the notations) introduced in [CDKP]:

$$\begin{align*}
\bar{u}_{0,0} &= \bar{u}_0 \\
\bar{u}_{0,2} &= \bar{u}_0 + (\bar{u}_1 + \bar{u}_3 - 2\bar{u}_2)/6 \\
\bar{u}_{0,1} &= \bar{u}_0 + (\bar{u}_2 + \bar{u}_3 - 2\bar{u}_1)/6
\end{align*}$$

Reflection of a shock. We first study the reflection of an oblique shock on the lower side of a rectangular domain. The exact solution and the boundary conditions of the simulation are shown figure 2. The coarsest grid is composed of $N_0 = 200$ triangles. The MR parameters are $\varepsilon = 0.01, L = 3$. The CPU time is $T_{MR} = 0.8 T_{FV}$. The CPU performances of Harten's scheme are disappointing on this example. To illustrate the adaptive computation of the fluxes, we plot on figure 3 what we called 'hybrid grid' consisting in all the triangles for which the flux computations are done precisely using the (supposed to be expensive) numerical solver instead of the (supposed to be cheaper) interpolation. The shape of the hybrid grid indicates that memory savings would be very interesting if the time evolution was be performed on an adaptive grid instead of the finest grid.

Mach 3 wind tunnel with a step. This model problem has been examined by several authors. The coarsest grid is composed of $N_0 = 126$ triangles. Four levels are used for the computations. The FV solution computed on the finest grid and the MR solution are displayed figure 4 (density at time $t = 4$). The CPU time for the FV run is $T_{FV} = 13944$ seconds. The MR parameters are $\varepsilon = 0.05, L = 3$. The CPU
time is $T_{MR} = 9931 = 0.71 T_{FV}$. Here again the CPU gain is not tremendous, but the discontinuities are clearly localized on the hybrid grid (not displayed) which is a promising property, in view of the adaptive scheme.

3.3. Limitations of Harten’s scheme

Although the flux is computed adaptively, the evolution of the solution at each time step still takes place on the finest grid $\Omega^L$. For these reasons, the computational gain is limited. In the next section, we describe a fully adaptive algorithm for which the memory storage is proportional to the number of wavelet coefficients describing the solution. The first results using adaptive MR schemes were presented in [GMM].

4. A Fully adaptive scheme

The goal of the adaptive algorithm is to save the maximal amount of computational time and memory space while maintaining the order of accuracy achieved by the reference FV scheme on the finest mesh.

4.1. The algorithm

At a given time $t_n$ the solution $u^n$ is now represented on a adaptive grid $\Lambda_n$ made of triangles belonging to different resolution levels. This approach gives way to new difficulties, for instance in the analysis of the stability and precision [CKMP]. Given the numerical solution represented in a compressed form by a set $\Lambda_n$ of significant wavelets coefficients i.e. $u^n = \sum_{\lambda \in \Lambda_n} d^n_\lambda \psi_\lambda$, the adaptive strategy consists in three steps:

1. Refinement. Predict from $\Lambda_n$ a larger set $\bar{\Lambda}_{n+1}$, such that $\Lambda_n \subset \bar{\Lambda}_{n+1}$, which is adapted to describe both $u^n$ and $u^{n+1}$ with the required accuracy. The new set of indices $\Lambda_{n+1}$ is defined from $\Lambda_n$ as follows:
   • $\Lambda_n$ is padded by a “security margin” determined by the CFL condition.
   • According to the magnitude of the details $d^n_\lambda$, we possibly add (at position $k$ and scale $\ell$) new wavelets coefficients located at the same space-location but with finer scales $\ell'$ such that $\ell < \ell' \leq \ell^* \leq L$. The precise computation of the local refinement level $\ell^*$ is given in [CKMP].

2. Evolution. Compute from $u^n$ an intermediate solution $\bar{u}^{n+1}$ defined from the set $\bar{\Lambda}_{n+1}$: $\bar{u}^{n+1} = \sum_{\lambda \in \bar{\Lambda}_{n+1}} d^{n+1}_\lambda \psi_\lambda$. We refer to [CKP] for a complete description of this step.

3. Thresholding. Use a thresholding procedure to obtain the new set $\Lambda^{n+1} \subset \bar{\Lambda}^{n+1}$ and the new approximation $u^{n+1} = \sum_{\lambda \in \Lambda^{n+1}} d^{n+1}_\lambda \psi_\lambda$.

An important feature of the scheme is that the set $\Lambda_n$ has a tree structure. This structure is crucial since it allows a one to one correspondence between the truncated multiscale decomposition of $\bar{u}^n$ and its cell averages on the adaptive triangulation associated to the set $\Lambda_n$ [CKMP].
4.2. Numerical tests

In a previous work [CKP], we have shown the efficiency of the adaptive scheme for scalar 2D problems. The adaptive computations were (up to 10 times) faster than the pure FV ones. We consider here the 1D gas dynamics system. 

**Sod’s shock tube.** The coarse discretization of the computing interval consists in 200 subdivisions. The finest one is seven times finer. The standard FV solution on this fine grid is our reference solution. On figure 5, we show the density ($t=0.26$) computed on the finest grid with the FV scheme (this computation took $T_{FV} = 693$ seconds) along with the MR solution ($L = 4, \varepsilon = 10^{-4}$ and the adaptive grid. For $\varepsilon = 10^{-3}$ (not displayed) the finest level of resolution is used exclusively near the four singularities of the density. This induces some slight oscillations near the contact discontinuity. The $L_1$-error is $E_1 = .00297$ and the computing time is $T_{MR} = 86 \approx T_{FV}/8$ seconds. For $\varepsilon = 10^{-4}$ (figure 5), the finer levels of resolution are used more often, leading to a better accuracy $E_1 = .00158$. In particular, the oscillations near the discontinuities have disappeared and the computing time remains interesting $T'_{MR} = 121 \approx T_{FV}/5.7$.

**Blast waves.** We now test the code to solve the difficult problem of interacting blast waves. The results are shown on figure 6 for 5 levels of resolution. We see that the fine levels are used only where the two waves interact, even though they may have been used at intermediate times. The computing time for several runs are summarized in the following table.

<table>
<thead>
<tr>
<th># levels</th>
<th>MR ($\varepsilon = 0.002$)</th>
<th>FV</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>CPU</td>
<td>memory</td>
</tr>
<tr>
<td>2</td>
<td>1329</td>
<td>207</td>
</tr>
<tr>
<td>3</td>
<td>2966</td>
<td>465</td>
</tr>
<tr>
<td>4</td>
<td>10677</td>
<td>621</td>
</tr>
<tr>
<td>5</td>
<td>38458</td>
<td>717</td>
</tr>
</tbody>
</table>

As far as the memory requirement is concerned, the advantages of the adaptive algorithm are indisputable. The CPU gain however is not so spectacular between the MR computation and the FV run on the corresponding finest level. This poor performance is due to the overhead costs in the MR program. We are using a C++ prototype, which is not optimized at all and in particular uses the standard dynamic memory allocation functions. This is not a sensitive issue when the grid is created once and for all as in the standard FV scheme but becomes very costly for the time adaptive grid book-keeping.

5. Concluding remarks

A FV Multiscale scheme has been applied to solve classical gas dynamics tests. We have shown the effectiveness of the method on triangular meshes but also its limits. Another FV Multiscale scheme is presented. This new scheme takes full advantage of the multiscale representation of the solution: at each time step, the solution is represented in a compressed form by a set of significant wavelets coefficients
(or equivalently on an adaptive grid). Not only the fluxes but also the whole solution is computed adaptively. As far as the memory requirement is concerned, the advantages of the adaptive algorithm are indisputable. The improvement in terms of CPU is however not as spectacular as in the scalar case treated in [CKP] due to big overhead costs in the MR program. The big difference with the present tests, in terms of programming, is that we now handle vectorial objects for which the use of optimized dynamic memory allocation functions is crucial. This technical problem is currently under investigation.

\[ \begin{align*} T_0^t & \quad T_1^t \\ T_0^t & \quad T_0^{t+1} \\ T_2^t & \quad T_3^t \end{align*} \]

**Figure 1.** Subdivision of a triangle $T_0^t$.

**Figure 2.** Exact solution (and boundary conditions) of the reflection problem.

**References**


The hybrid grid

Figure 3. A burst view of a hybrid grid.

FV solution, time t=4

MR solution, time t=4

Figure 4. FV solution of the step problem and MR Harten’s solution.

**Figure 5.** Shock tube: density (left y-axis) and adaptive grid (right).

**Figure 6.** Blast Waves: density (left y-axis) and adaptive grid (right).


Laboratoire d’analyse numérique, Université Pierre & Marie Curie, 75252 Paris Cedex 05, France

E-mail address: cohen, kaber, postel@ann.jussieu.fr
Multiresolution using triangles

S.M. Kaber †  M. Postel ‡

December 17, 1998

Abstract
Multiresolution of a polygonal domain consists here in building a hierarchy of two dimensional unstructured and nested mesh. Functions are represented on a given level by their mean value on each triangle and the difference with a value computed by interpolating the mean values at the coarser level. Data compression is achieved by truncating these differences with a given tolerance. The notion of hybrid mesh - made of triangles from various resolution levels - is also precisely defined.

Résumé
La multirésolution pour un domaine polygonal est traitée ici en construisant une hiérarchie de maillages non structurés imbriqués. Une fonction est représentée sur un maillage donné par ses valeurs moyennes sur chaque triangle et par les différences entre ces dernières et des valeurs calculées par interpolation sur le maillage plus grossier. La compression de données est obtenue en tronquant ces différences en dessous d’une tolérance. La notion de maillage hybride - composé de triangles de différents niveaux de résolution - est aussi définie précisément.

Keywords
Multiresolution - Unstructured mesh - Data compression.

†A.S.C.I. Bat. 506 Université Paris Sud 91405 Orsay Cedex (ould@ann.jussieu.fr), and also
‡Laboratoire d’Analyse Numérique, UMR CNRS 7598, BC 187, 4 place Jussieu, 75252 Paris Cedex 05, FRANCE (postel@ann.jussieu.fr).
Contents

1 Introduction 3

2 Encoding/Decoding 4
   2.1 Encoding .................................................. 4
   2.2 Decoding .................................................. 7
   2.3 Examples .................................................. 7

3 The hybrid grid 9
   3.1 Examples .................................................. 23

4 Conclusions 24
1 Introduction

Multiresolution analysis is a technique of data representation which has several interesting aspects at the boundary between different domains like wavelet decomposition, CFD, etc... Our goal is to couple multiresolution on unstructured grids with algorithms to compute the solutions of conservation laws. We first present an algorithm to encode information and detect singularities. This enables us to define a hybrid grid where the encoded function is represented on cells pertaining to different levels according to the local regularity of the function. The next step is the design of an algorithm to compute the solution of an hyperbolic equation using this hybrid grid. In order to do this the hybrid grid must be able to evolve in time along with the solution: this implies at each time step encoding - decoding from the hybrid grid to the fine grid. Eventually, the actual implementation of the hyperbolic solver - which includes the computation of the fluxes on the hybrid grid has been postponed to a future publication.

Multiresolution on unstructured grids has been tackled by Abgrall and Harten in [AH] along with a justification of the criteria for the choice of stencils on which they compute the interpolated values. We present here an alternative algorithm for the encoding of the data on triangles. The coupling of Multiresolution with CFD has been addressed in [BH97] in the case of cartesian grids. In this framework it boils down to tensorial products of the one dimensional algorithms. In one dimension, the complete problem has first been successfully tested with wavelets in [MPR91]. It has then been solved in the multiresolution context in [Har95] and [Har94]. However in these two papers, the evolution of the solution in time is computed on the finest grid. This is not what we propose to do in two-D, as stated above, we expect that a better efficiency can be achieved by computing the time evolution on the hybrid grid.

Multiresolution coding-encoding algorithms are often classified into "point values" and "cell averages" with important advantages for both solutions. For details on point values resolution, consult [Coh]. Since we aim in the near future at the resolution of conservation laws with finite volumes algorithms - whose natural output is the mean value of the solution on the grid cells - we will concentrate on this approach.

We first define useful quantities and notations:
A function $u$ is defined on a polygonal domain $\Omega \subset \mathbb{R}^2$. Let $\Omega^L$ be a given coarse triangulation of $\Omega$. The triangulation $\Omega^\ell$ ($\ell = 0, \ldots, L - 1$) is obtained from $\Omega^{\ell+1}$ by dividing each triangle in smaller triangles. Therefore $\Omega^0$ denotes the finest triangulation. Let $T^\ell_i$ denote a generic triangle in $\Omega^\ell$

$$\Omega^\ell = \bigcup_{i=1}^{N_i} T^\ell_i.$$  

The measure of a triangle $T$ is noted

$$|T| = \int_T dx dy.$$  

The centroid of the triangle $T^\ell_i$ is denoted by $c^\ell_i$, its coordinates are

$$x^\ell_i = \frac{1}{|T^\ell_i|} \int_{T^\ell_i} x dx dy, \quad y^\ell_i = \frac{1}{|T^\ell_i|} \int_{T^\ell_i} y dx dy.$$  

We define also the average operator $A$ on the triangles

$$A(T)w = \frac{1}{|T|} \int_T w(x,y) dx dy.$$
and the mean values of $u$ on the triangles

$$\bar{u}_i^\ell = \frac{1}{|T_i^\ell|} \int_{T_i^\ell} u(x, y) \, dx \, dy = A(T_i^\ell) u.$$  

Eventually we denote $\bar{u}^\ell$ the collection of values $(\bar{u}_i^\ell)$. We have stated in the introduction that we work on unstructured grids. This is true in the sense that the coarse grid need not be cartesian. On the other hand even with a complicated geometry the grid can be as structured as the coarser grid allows it in the following sense: given $N^L$ triangles of a coarse grid $\Omega^L$ and $L \geq 1$ the maximum number of different grids, the triangulation $\Omega^\ell$ ($\ell = 0, \ldots, L - 1$) is obtained from $\Omega^{\ell+1}$ by dividing each triangle into $q$ smaller triangles. In the sequel $q$ is fixed to four, and the small triangles are obtained by setting three new vertices at the mid-points of the initial triangle edges. The grid $\Omega^\ell$ is composed of $N^\ell = 4^{L-\ell}N^L = 4^{-\ell}N^0$ triangles. Note that this refinement algorithm is well adapted to finite volume implementation but would correspond to non conformous grid in a Finite Element framework.

2 Encoding/Decoding

In this section we explain how the knowledge of a function by its cell average values on a grid of a given level - within the hierarchy introduced above - enables the computation of its cell average on a coarser grid (Encoding) or on a finer grid (Decoding).

2.1 Encoding

In the case of cell average representation the knowledge of the function on the grid $\Omega^{\ell-1}$ enables its representation on the immediately coarser grid $\Omega^\ell$ in the following way: we denote $T_{i,j}^{\ell-1}$ ($j = 0, \ldots, 3$) the four triangles of $\Omega^{\ell-1}$ composing the triangle $T_i^\ell$, by convention $T_{i,3}^\ell$ is the central triangle as shown on figure 1.

$$T_i^\ell = \bigcup_{j=0}^3 T_{i,j}^{\ell-1}.$$  

The mean values of $u$ on the coarse grid $\Omega^\ell$ are computed naturally by

$$\bar{u}_i^\ell = \frac{1}{|T_i^\ell|} \sum_{j=0}^3 |T_{i,j}^{\ell-1}| \bar{u}_{i,j}^{\ell-1}. \quad (1)$$  

Note here that the same triangle - and the quantities defined on it - can be denoted by $T_i^\ell$ within its own level and by $T_{i,k}^{\ell+1}$ as the $k^{th}$ subdivision (out of four) of the $j^{th}$ triangle at the level $\ell + 1$. Going from the grid $\Omega^\ell$ to the finer grid $\Omega^{\ell-1}$ is not as straightforward. It is actually at this level that our method differs from the references cited in the introduction. Let $(S_{i,j}^\ell)_{j=0}^3$ be the vertices of the triangle $T_i^\ell \in \Omega^\ell$. For each vertices $S_{i,j}^\ell$, we associate a polynomial of degree lower or equal to $N$ defined by

$$p_{i,j}^\ell(x, y) = a_{0,0} + \sum_{k=1}^N \sum_{m+n=k} a_{n,m}(x - x_{i}^\ell)^n(y - y_{j}^\ell)^m, \quad (2)$$  

where the reals $a_{n,m} \equiv (a_{i,j})_{n,m}$ are to be optimally determined. In the adopted frame work of mean value based multiresolution this is done by imposing that the mean values of $p_{i,j}^\ell$ and of
the function \( u \) coincide on a set of neighbour triangles of \( S_{i,j}^l \). Let \( \mathcal{V}_{i,j}^l \) be this set of triangles, such that all its members belong to \( \Omega^l \), \( \mathcal{V}_{i,j}^l \subset \Omega^l \). Imposed conditions on \( p_{i,j}^l \) are

\[
\mathcal{A}(T)p_{i,j}^l = \mathcal{A}(T)u, \quad \forall T \in \mathcal{V}_{i,j}^l. \tag{3}
\]

Depending on the degree of the polynomial and on the number of neighbours that we are ready to consider the coefficients \( a_{n,m} \) will be over or under determined. For instance, if the refinement of the grids is done in the way previously described - by dividing each triangle into four smaller - it is easy to see that, except for some points of the initial coarse grid and for points on the frontiers, one vertex usually belongs to six triangles. These six triangles could form the set \( \mathcal{V}_{i,j}^l \) of 'natural' neighbours and enable the determination - using (3) - of the six coefficients defining a polynomial of degree two. For points with less than six neighbours, further triangles need to be considered and added to \( \mathcal{V}_{i,j}^l \), chosen for instance among the neighbours' neighbours. This last group of triangles being larger than needed the selection will have to be done following some other criterium (see for instance [AH]). From another standpoint, only three triangles are necessary to define a polynomial of degree one - supposing that the equations provided by (3) are independent. Therefore here again, for any vertex belonging to a refinement grid, there are too many neighbours - six - out of which only three should be chosen. This choice of the stencil on which the approximation is computed is very important and will be illustrated in one of the numerical simulations. In this type of situations we choose an alternative method: the equation (3) is imposed on \( T_i^l \in \Omega^l \) and verified in the least square sense only for all the other triangles of the \( \mathcal{V}_{i,j}^l \).

We now detail the case of the degree \( N = 1 \), since the numerical simulations shown thereafter are done using it. The expression (2) reduces to

\[
p_{i,j}^l(x,y) = a_{i,j}^l(x - x_i^l) + b_{i,j}^l(y - y_i^l) + c_{i,j}^l. \tag{4}
\]

Imposing (3) on \( T_i^l \) leads to

\[
c_{i,j}^l = \mathcal{A}(T_i^l)u = \bar{a}_i^l. \tag{5}
\]

The cardinal of \( \mathcal{V}_{i,j}^l \) is almost always greater than 2, hence \( p_{i,j}^l \) is required to satisfy (3) in the least square sense: find \( (a, b) = (a_{i,j}^l, b_{i,j}^l) \) minimizing

\[
\sum_{T \in \mathcal{V}_{i,j}^l \setminus T_i^l} |\mathcal{A}(T)p - \mathcal{A}(T)u|^2 \quad \text{with} \quad p(x, y) = a(x - x_i^l) + b(y - y_i^l) + \bar{a}_i^l.
\]

This leads to a \( 2 \times 2 \) linear system whose matrix is equal to \( B^TB \) with

\[
B = \begin{pmatrix}
    x_i^l - x_i^l & y_i^l - y_i^l \\
    \vdots & \vdots \\
    x_k^l - x_i^l & y_k^l - y_i^l \\
    \vdots & \vdots
\end{pmatrix} \quad \text{with} \quad k \neq i.
\]

It is therefore symmetric positive. Furthermore it is definite if at least two centroids \( G_k^l \) are not aligned with the current centroid \( G_i^l \). Let \( T^l \equiv T^l(\bar{a}_i^l) \) be the operator \( T^l : \Omega^{l-1} \mapsto \mathbb{R} \) defined by

\[
T^l(T_{i,j}^{l-1}) = \begin{cases}
  \mathcal{A}(T_{i,j}^{l-1})p_{i,j}^l \quad & j = 0, 1, 2 \\
  \frac{1}{|\mathcal{V}_{i,j}^l|} \left\{ |T_i^l| \bar{a}_i^l - \sum_{k=0}^{j-3} |T_{i,j}^{l-1-k}| T^l(T_{i,j}^{l-1-k}) \right\} & j = 3.
\end{cases}
\]
An approximation of the mean values \( u \) on the small triangles \( T^\ell_{i,j} \) can be computed by,
\[
\bar{u}^\ell_{i,j} = I^\ell(T^\ell_{i,j}), \quad j = 0, \ldots, 3.
\]
(7)
For each vertices \( S^\ell_{i,j} \), we define the differences \( d^\ell = \{d^\ell_{i,j} \mid 0 \leq i \leq N^\ell \} \) with
\[
d^\ell_{i,j} = \bar{u}^\ell_{i,j} - \bar{u}^{\ell-1}_{i,j}.
\]
(8)
Since the cell average values on consecutive levels are linked by
\[
\sum_{j=0}^{3} |T^\ell_{i,j}| \bar{u}^{\ell-1}_{i,j} = |T^\ell_{i,j}| \bar{u}^\ell_{i,j},
\]
we get from (1), (8) and (9), that
\[
\sum_{j=0}^{3} |T^\ell_{i,j}| d^{\ell-1}_{i,j} = 0.
\]
(10)

**Algorithm 2.1**

- \( u \) is known by its cell average values on the finest grid \( \Omega^0 \)
- **For** \( \ell = 1 \rightarrow L 
  
  1. Compute \( \bar{u}^\ell \) by (1)
  2. From \( \bar{u}^\ell \), compute \( I^\ell \) on \( \Omega^\ell \) by (7) and the differences \( d^{\ell-1} \) by (8)
- **end For**

This algorithm defines a one to one transformation between, on the one hand \( \Lambda^0 \) values \( \bar{u}_i^0 \) and on the other hand \( \Lambda^L \) values \( \bar{u}_i^L \) along with the differences \( d^{\ell-1}, \ldots, d^0 \).

We explain now by looking at one particular triangle why the information requires the same amount of storage in the two configurations (before or after Encoding).

For one triangle \( T^1_i \):

- The four mean values \( \bar{u}^0_{i,j} \), \( 0 \leq j \leq 3 \) are known
- Compute \( \bar{u}^1 \) using (1) and store into \( \bar{u}^0_{i,3} \)
- Compute \( I^1(T^0_{i,j}) \) for \( 0 \leq j \leq 2 \) and replace \( \bar{u}^0_{i,j} \) by \( I^1(T^0_{i,j}) \)
- No need to store \( d^0_{i,3} \), since relation (10) shows that the four \( d^0_{i,j} \) are linked.

It is now time to motivate all this hard work. Encoding the information using the above algorithm can be used to save storage space if the function is smooth or on the other hand to detect where it is singular. The method to save space is very simple: if \( |d^0_{i,j}| < \varepsilon\ell \) it is thrown away entirely setting \( d^0_{i,j} = 0 \). The criterion \( \varepsilon\ell \) varies with the grid level \( \varepsilon\ell = \lambda^{-\ell}\varepsilon \) \( (\lambda > 1) \), so that the finer the grid the easier it is to verify and eventually to drop some differences. Let \( \mathcal{E}^\ell \) be the set of coefficients actually computed and saved at level \( \ell \), i.e., such that \( |d^0_{i,j}| > \varepsilon\ell \). The "multiresolution analysis" of \( \bar{u}^0 \) is the data set comprising \( \bar{u}^L \) and the coefficients \( \mathcal{E}^{\ell-1}, \ldots, \mathcal{E}^0 \). It is denoted by
\[
\bar{u}^\varepsilon = \mathcal{M}^\varepsilon \bar{u}^0 \equiv (\varepsilon, \mathcal{E}^0, \ldots, \mathcal{E}^{\ell-1}, \bar{u}^\varepsilon_{i,j}).
\]

The former encoding algorithm is modified to introduce the "truncation" operation.
2.2 Decoding

The function \( u \) is known on the grid \( \Omega^L \), say \( u^L \), along with the differences \( d^{L-1}, \ldots, d^0 \). The following algorithm computes \( u^{L-1}, \ldots, u^0 \) or more symbolically \( \tilde{\alpha}^{L,0} = \mathcal{M}^{-1} \tilde{\alpha}^L \):

Algorithm 2.2

\[ \begin{align*}
\bullet \ & \text{For } \ell = L \setminus 1 \ \text{Calcul de } \tilde{\alpha}^{L,\ell-1} \\
\qquad \bullet \ & \text{For } i = 1 \to N^{L-1} \\
\qquad \qquad & \tilde{\alpha}^{L,\ell-1}_i = d^{L,\ell-1}_i + I^\ell(T^L_{i,j}^{-1}) \\
\qquad \end{align*} \]

\[ \text{end For } i \]

\[ \text{end For } \ell \]

For \( \varepsilon = 0 \), the transformation is "one to one" and we get back exactly \( \tilde{\alpha}^0 \). For \( \varepsilon \neq 0 \), the difference between \( \tilde{\alpha}^0 \) and \( \tilde{\alpha}^0 = \mathcal{M}^{-1} \mathcal{M}^\varepsilon \tilde{\alpha}^0 \) can be estimated as follows: The interpolated value \( I^\ell(T^L_{i,j}^{-1}) \) is given by the polynomial expression defined in (4). It depends linearly on the mean values \( \bar{u}_k \) of \( u \) on all the triangles \( T_k \) belonging to \( \mathcal{T}^\ell_{i,j} \)

\[ I^\ell(T^L_{i,j}^{-1}) = \sum_{k \in \mathcal{T}^\ell_{i,j}} \alpha^\ell_{i,j,k} \bar{u}_k \]

The coefficients \( \alpha^\ell_{i,j,k} \) depend only on the geometry and can therefore be bounded by \( C \) independently of the level \( \ell \) and \( \varepsilon \). We have then

\[ |\tilde{\alpha}^\ell - \tilde{\alpha}^{\varepsilon,\ell}| \leq \|d^\ell - d^{\varepsilon,\ell}\| + C|\tilde{\alpha}^{\ell+1} - \tilde{\alpha}^{\varepsilon,\ell+1}|. \]

Using the fact that

\[ |d^\ell - d^{\varepsilon,\ell}| \leq \lambda^{-\ell} \varepsilon, \]

we obtain easily that

\[ |\tilde{\alpha}^0 - \tilde{\alpha}^{\varepsilon,0}| \leq C \varepsilon. \]

In the former inequalities the sup norm is the discrete norm

\[ \|v^\ell\| = \max_{0 \leq t \leq N^\ell} |v^\ell_t|. \]

2.3 Examples

Except where otherwise specified, in all the examples the functions are given on the unit square \( \Omega = [0,1] \times [0,1] \). They are represented on \( \Omega \) on at most five levels of discretization.

The errors \( E_1, E_2 \) and \( E_\infty \) are computed on the fine grid \( \Omega^0 \). For \( p = 1 \) and 2 they are defined by

\[ |\Omega^0| E_p = \left( \sum_{T \in \Omega^0} \left( \int_T \tilde{u}^0_T - \mathcal{A}(T) u_T \right)^p \right)^{1/p}. \]

The integral \( \mathcal{A}(T)w \) is evaluated by the simple quadrature formula

\[ \mathcal{A}(T) \simeq |T| w(x_T, y_T), \]
hence $E_1$ and $E_2$ are evaluated by

$$|\Omega^p| E_p \simeq \left( \sum_{T \in \Omega^p} |T| |u_T^0 - u(x_T, y_T)|^p \right)^{1/p}.$$  

The sup error is computed by

$$E_\infty = \max_{T \in \Omega^p} |\tilde{u}_T^0 - A(T) u| \simeq \max_{T \in \Omega^p} |\tilde{u}_T^0 - u(x_T, y_T)|.$$  

Example 2.1 We first study a smooth function a square domain $[0,1] \times [0,1]$  

$$u^0(x, y) = \sin(x) \sin(2y).$$  

(11)  

On figure 2 and figure 3 the error is represented as a function of the tolerance. On figure 2 the full range of $\varepsilon$ is represented from $10^{-1}$ to $10^{-2}$, while on figure 3 only a close-up for $\varepsilon$ between $10^{-1}$ to $10^{-3}$ is represented. The sup error is of course much higher than the two other norms but it also reaches a maximum. This is due to the fact that the solution is smooth, therefore the error that one does by interpolating linearly on a triangle is somewhat related to the size of this triangle. The minimum error level at which the step is reached corresponds to the maximum difference $\delta_i^L$ on all the triangles and all the allowed resolution levels. It is also interesting to notice its dependency with respect with the number of resolution levels. On figure 4 we represent the sup error with respect to $\varepsilon$ for five and six levels. Five levels beeing the case of the two former figures. The sup error reaches a maximum first for six levels of resolution while it goes on increasing for five levels. The efficiency or compression rate $\nu$ is defined as

$$\nu = \frac{\text{card}(\Omega_0)}{\text{card}(\Omega_L) + \sum_{\ell=0}^{L-1} \text{card}(D_\varepsilon^\ell)}$$  

(12)  

where $D^\varepsilon = \{T^\varepsilon / u^\varepsilon_{i,j} \neq 0\}$. It is plotted on figure 5 as a function of $\varepsilon$ again for five and six levels of resolution. As expected the efficiency improves if the number of resolution levels increases.

To visualize the efficiency and give an idea of how it can be used to detect singularities we define for a given tolerance level $\varepsilon$, the hybrid grid $\Omega^\varepsilon$, as a set of nonoverlapping cells belonging to different levels of resolution. It is built with the following algorithm:

**Algorithm 2.3**

- $\Omega^\varepsilon = \Omega^0$
- For $\ell = 1 \rightarrow L$
  - For $i = 0 \rightarrow N^\ell$
    - If $|u_{i,j}^{\ell-1}| < \varepsilon^\ell$ and $T_{i,j}^{\ell-1} \in \Omega^\varepsilon$ for all $j = 0, \ldots 3$
      - THEN
        - $\Omega^\varepsilon = \Omega^\varepsilon \setminus \bigcup T_{i,j}^{\ell-1}$
        - $\Omega^\varepsilon = \Omega^\varepsilon \cup T_{i}^{\ell}$
      - END IF

8
\[ u(x,y) = e^{-5(x-0.5)^2} e^{-10(y-0.5)^2} \] (13)

Example 2.2 The second example is also defined on a square domain but the function presents a sharp extremum in the middle.

For this example we show the corresponding set of figures: the figure 9 and the figure 10 show the different errors as a function of the tolerance level \( \varepsilon \), and the figure 11 shows the compression rate \( \nu \) also as a function of \( \varepsilon \). The hybrid grid for \( \varepsilon = 10^{-2} \) (respectively \( \varepsilon = 10^{-3} \)) is represented on figure 12 (resp. figure 13).

Example 2.3 Eventually we study an example on a more complicated geometry as shown on figure 14 on which we represent the hybrid grid obtained for the following function

\[
\begin{cases}
1 & \text{if } \sqrt{(x-2)^2 + (y-1)^2} < 0.5 \\
0 & \text{elsewhere}
\end{cases}
\]

We start from a very coarse grid with only eight elements and allow seven levels of resolution. The truncation level for the differences is \( \varepsilon = 10^{-2} \). To fix the ideas, the 1931 triangles of the hybrid grid (figure 14) gives the same precision (up to tolerance \( \varepsilon \)) than the 32768 triangles of the finer grid.

The compression rate - which is not exactly the ratio between these two numbers but is computed with formula (12) - is here equal to 16.89. The hybrid grid is also represented in a 3D mode (figure 15) with the level number as third coordinate for each triangle.

On figure 16 we represent the hybrid grid for \( \varepsilon = 10^{-2} \) but with a reconstruction polynomial computed using exactly three triangles and solving the linear system provided by (3) instead of the least square definition using all neighbours. The differences are non negligible and prove that the reconstruction scheme is important and should be improved, at least by using a quadratic interpolation instead of a linear one.

3 The hybrid grid

The encoding-decoding algorithm presented in the previous section provides a compacted representation of a dataset and a way to save storage space in the smoothness areas. We would like now to develop another representation which avoids the systematic representation on the finest grid and establish the transformation between the coarsest level of resolution \( \Omega_L \) and the hybrid grid \( \Omega_h \), which contains the information on the spatial irregularities. In other words, we should be able, knowing the function \( u \) on the coarse grid \( \Omega_L \), and for a given real \( \varepsilon > 0 \), to find a hybrid triangulation \( \Omega^\varepsilon \) made of elements from different triangulations \( \Omega_L \) and such that \( \|u^\varepsilon - u^0\| \leq \varepsilon \) where \( u^\varepsilon \) denotes the function defined for all \( T \in \Omega^0 \) by

- if \( T \in \Omega^\varepsilon \), \( \mathcal{A}(T)u^\varepsilon = \mathcal{A}(T)u^0 \)
Figure 1: Division of the triangle $T_i^f$

Figure 2: Errors vs $\epsilon$ for the full range $[0, 0.01]$. Encoding-Decoding Algorithm. Example (2.1)
Figure 3: Errors vs $\varepsilon$ for small $\varepsilon$ in $[0, 10^{-3}]$. Encoding-Decoding Algorithm. Example (2.1)

Figure 4: Sup error vs $\varepsilon$ for five and six levels at most of resolution. Example (2.1)
Figure 5: Efficiency - compression ratio vs $\varepsilon$ for example (2.1)

- if $T \notin \Omega^e$, $\mathcal{A}(T)u^e$ is computed by some interpolation of the values of $u^e$ on the triangles of $\Omega^e$ which are neighbours of $T$.

First, we precise the criterium for the division of a triangle of $\Omega^\ell$ into four triangles of $\Omega^{\ell-1}$: for each vertex $S_{i,j}^{\ell}$, we consider the polynomial defined by (4). The three coefficients are again determined by imposing that the mean values of $p_{i,j}^{\ell}$ and of the function $u$ coincide on a set $\mathcal{V}_{i,j}^{\ell}$ of neighbour triangles of $S_{i,j}^{\ell}$. Refering to paragraph 2.1, we encounter the same problems of over- (or under-) determination of the polynomial - depending on its degree. Here again its coefficients will be the solution of a least-squared problem.

The novelty is that now the triangles of $\mathcal{V}_{i,j}^{\ell}$ can belong to $\Omega^\ell$ itself but also to coarser levels, as shown on figure 17.

$$\mathcal{V}_{i,j}^{\ell} \subset \bigcup_{k=\ell}^{L} \Omega^k.$$  

This possibility increases the chances of aligned centroids for neighbour triangles. In some cases, corresponding to points on the boundary with few neighbours, the least squared matrix may therefore become singular. Whenever this happen the polynomial cannot be defined for the given vertex. We assume - for geometrical reasons - that this will happen for at most one vertex in a triangle and compute the polynomials for the two others vertices. Then, using relation (9), we affect the same mean values to both the center small triangle and the triangle for wich no polynomial could be defined.

If for one of the vertices $S_{i,j}^{\ell}$ the error $\varepsilon_{i,j}^{\ell} = |d_{i,j}^{\ell-1}|$ is larger than a tolerance $\varepsilon_{\ell}$, the triangle $T_{i}^{\ell}$ is divided, and the mean values of $u$ are computed on the small triangles

$$\bar{u}_{i,j}^{\ell-1} = \mathcal{A}(T_{i,j}^{\ell-1})u.$$  

The criterium $\varepsilon_{\ell}$ varies again with the grid level $\varepsilon_{\ell} = \lambda^{-\ell}\varepsilon$. ($\lambda > 1$). For a given $\varepsilon > 0$ the algorithm is the following

Algorithm 3.1
Figure 6: Hybrid grid for $\varepsilon = 0.01$ example (2.1). Algorithm (2.3)
Grille mixte

seuil = 0.001

Figure 7: Hybrid grid for $\varepsilon = 0.001$ example (2.1). Algorithm (2.3)
Figure 8: Hybrid grid for $\varepsilon = 0.0001$ example (2.1). Algorithm (2.3)
Figure 9: Errors vs $\varepsilon$ example (2.2)

Figure 10: Errors vs $\varepsilon$
\( \Omega^c = \Omega^L \)

- For \( \ell = L \setminus 1 \)
  - For \( i = 1 \) \( \neq N^\ell \)
    - If \( T_i^\ell \in \Omega^c \)
      - If \( \epsilon_{i,j} > \varepsilon^\ell \) for at least one of the \( j = 0, \ldots, 2 \)
        - \( T_i^\ell \) is divided, \( \bar{a}_{i,j}^{\ell-1} \) are computed and \( \Omega^\ell \) is updated.
      - For \( j = 0 \) \( \neq 3 \)
        - \( \bar{a}_{i,j}^{\ell-1} = A(T_i^{\ell-1})u \)
      - end For \( j \)
    - \( \Omega^c = \Omega^c \setminus T_i^\ell \)
    - \( \Omega^c = \Omega^c \cup \left( \bigcup_{j=0}^{3} T_i^{\ell-1} \right) \)
  - end If
  - end For \( i \)
- End For \( \ell \)

The grid \( \Omega^c \) being built, the efficiency of algorithm 3.1 is tested by computing \( u \) on \( \Omega^0 \) (starting from \( u \) given on the grid \( \Omega^c \)) and comparing it with \( u^0 \). For that purpose, all triangles belonging to \( \Omega^c \) are divided as many times as needed to reach the finest level of discretization. Meanwhile, dividing the triangle \( T_i^\ell \) as shown in figure 1, an approximation of the mean values \( u \) on the small triangles \( T_{i,j}^{\ell-1} \) is computed by (7)

**Algorithm 3.2**

- Mean values of \( u \) are known on the grid \( \Omega^c \)
Figure 12: Hybrid grid for example 2.2, $\varepsilon = 0.01$
Figure 13: Hybrid grid for example 2.2, $\epsilon = 0.001$
Figure 14: Hybrid grid for example 2.3, $\varepsilon = 0.01$
Grille mixte
seuil = 0.01

Figure 15: Hybrid grid in 3D mode for example 2.3, $\varepsilon = 0.01$
Figure 16: Hybrid grid for example 2.3, $\varepsilon = 0.01$. Interpolation on three neighbours
• For \( \ell = L \setminus 0 \),
  
  – Computation of \( u \) on \( \Omega^\ell \).
  
  – For \( i = 1 \setminus N^\ell \)
    
    * If \( T \in \Omega^{x,\ell} = \Omega^x \cap \Omega^\ell \), \( T \) is divided and the mean values of \( u \) are computed on its subdivisions by \( T \).
  
  – end For \( i \)

• end For \( \ell \)

3.1 Examples

We first apply the algorithm defined above to the last example of the previous section. We get exactly the same hybrid grid as using the Code-Decode algorithm. In the two other examples the functions are given on the unit square \( \Omega = [0, 1] \times [0, 1] \). They are represented on \( \Omega \) with at most four levels of discretization.

Example 3.1 We take again the smooth function of example 2.1, \( u^O(x, y) = \sin(x)\sin(2y) \). The performances obtained for varying tolerance level on the finest grid are gathered in the table (1).

<table>
<thead>
<tr>
<th>( \varepsilon )</th>
<th>( E_\infty )</th>
<th>( E_1 )</th>
<th>( E_2 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( 10^{-2} )</td>
<td>( 0.295889 \times 10^{-1} )</td>
<td>( 0.146022 \times 10^{-2} )</td>
<td>( 0.228351 \times 10^{-2} )</td>
</tr>
<tr>
<td>( 10^{-3} )</td>
<td>( 0.127874 \times 10^{-1} )</td>
<td>( 0.309395 \times 10^{-3} )</td>
<td>( 0.513387 \times 10^{-3} )</td>
</tr>
<tr>
<td>( 10^{-4} )</td>
<td>( 0.746338 \times 10^{-2} )</td>
<td>( 0.167857 \times 10^{-6} )</td>
<td>( 0.174172 \times 10^{-3} )</td>
</tr>
</tbody>
</table>

Table 1: Errors as a function of the tolerance level \( \varepsilon \) for the function (2.1).

The figure 18 represents the hybrid grid for \( \varepsilon = 0.01 \), the figure 19 represents the isovales of the solution interpolated on the finest grid using the values on the hybrid grid and the figure 20 represents the isovales of the exact solution.

Example 3.2 The example of a discontinuous function is now considered. A standard example in gas dynamics is provided by the reflection of a shock in a compressible fluid (See [Col90]). Euler’s equations of gasdynamics are:

\[
\begin{align*}
\partial_t u + \partial_x f(u) + \partial_y g(u) &= 0 \quad (15) \\
u(x, y, 0) &= u_0(x, y) \quad (16)
\end{align*}
\]

with \( u = (\rho, \rho u, \rho v, e) \). The fluxes \( f \) and \( g \) are defined by

\[
\begin{align*}
f &= (\rho u, p + \rho u^2, \rho uv, (e + p)u) \quad (17) \\
g &= (\rho v, \rho uv, p + \rho v^2, (e + p)v) \quad (18)
\end{align*}
\]

\( \rho \) is the fluid density, \( u \) and \( v \) are respectively the components of the velocity, \( p \) is the pressure, the gas is assumed to be perfect and polytropic, \( p = (\gamma - 1)(e - \frac{1}{2}\rho(u^2 + v^2)) \), \( \gamma = 1.4 \). \( e \) is the energy per volume unit, \( e = \rho E + \frac{1}{2}\rho(u^2 + v^2) \), \( E \) is the internal energy per unit mass: \( (\gamma - 1)E = p/\rho \).
We use the reflection of an oblique shock on the lower side of a rectangular domain defined by $0 \leq \xi \leq 4.12829$, $0 \leq \eta \leq 1$. The initial flow is given by

$$\rho = 1, \quad u = 2.9, \quad v = 0 \quad \text{and} \quad p = \frac{1}{1.4}$$

The boundary conditions are:

- **inflow boundary conditions on** $\xi = 0$; all the variables are fixed taking the same values as at initial time.
- **outflow boundary conditions on the side** $\xi = 4.12829$; no variable is imposed.
- **reflection condition on the lower side** $\eta = 0$:

$$v(\xi, 0) = 0$$

- **fixed values on the upper side** $\eta = 1$ (those of the exact solution):

$$\rho = 1.7, \quad u = 2.61932, \quad v = -0.506339 \quad \text{and} \quad p = 1.52824$$

The exact solution is formed by an incident shock with angle $29^\circ$ and a reflected shock with angle $23.28^\circ$, see figure 21. The constant state after the second shock is:

$$\rho = 2.68732, \quad u = 2.40148, \quad v = 0 \quad \text{and} \quad p = 2.93413$$

Since the admissible error is smaller than the function jump, in the neighborhood of the discontinuity the solution is not interpolated but computed on the finest possible grid. On the other hand since the solution is piecewise constant everywhere else the linear approximation is exact. Hence the actual error is always zero, whatever its definition ($E_\infty$, $E_1$ and $E_2$). The figure 22 shows the hybrid grid obtained using a tolerance criterium equal to 0.01 and at most 4 levels of discretization.

4 Conclusions

In the near future we plan to improve the encoding-decoding algorithm by implementing the quadratic interpolation for the reconstruction. This should increase the compression ratio for a fixed error level. The next step will be of course the actual treatment of a hyperbolic equation (as 3.2) in the multiresolution framework. This implies the computation of the fluxes and the time evolution of the hybrid grid.

Acknowledgements. We thank Albert Cohen for his interest in our work and his numerous suggestions.

References


List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Division of the triangle $T_i^k$</td>
<td>10</td>
</tr>
<tr>
<td>2</td>
<td>Errors vs $\varepsilon$ for the full range $[0, 0.01]$. Encoding-Decoding Algorithm. Example (2.1)</td>
<td>10</td>
</tr>
<tr>
<td>3</td>
<td>Errors vs $\varepsilon$ for small $\varepsilon$ in $[0, 10^{-3}]$. Encoding-Decoding Algorithm. Example (2.1)</td>
<td>11</td>
</tr>
<tr>
<td>4</td>
<td>Sup error vs $\varepsilon$ for five and six levels at most of resolution. Encoding-Example (2.1)</td>
<td>11</td>
</tr>
<tr>
<td>5</td>
<td>Efficiency - compression ratio vs $\varepsilon$ for example (2.1)</td>
<td>12</td>
</tr>
<tr>
<td>6</td>
<td>Hybrid grid for $\varepsilon = 0.01$ example (2.1). Algorithm (2.3)</td>
<td>13</td>
</tr>
<tr>
<td>7</td>
<td>Hybrid grid for $\varepsilon = 0.001$ example (2.1). Algorithm (2.3)</td>
<td>14</td>
</tr>
<tr>
<td>8</td>
<td>Hybrid grid for $\varepsilon = 0.0001$ example (2.1). Algorithm (2.3)</td>
<td>15</td>
</tr>
<tr>
<td>9</td>
<td>Errors vs $\varepsilon$ example (2.2)</td>
<td>16</td>
</tr>
<tr>
<td>10</td>
<td>Errors vs $\varepsilon$</td>
<td>16</td>
</tr>
<tr>
<td>11</td>
<td>Efficiency versus $\varepsilon$ for example 2.2</td>
<td>17</td>
</tr>
<tr>
<td>12</td>
<td>Hybrid grid for example 2.2, $\varepsilon = 0.01$</td>
<td>18</td>
</tr>
<tr>
<td>13</td>
<td>Hybrid grid for example 2.2, $\varepsilon = 0.001$</td>
<td>19</td>
</tr>
<tr>
<td>14</td>
<td>Hybrid grid for example 2.3, $\varepsilon = 0.01$</td>
<td>20</td>
</tr>
<tr>
<td>15</td>
<td>Hybrid grid in 3D mode for example 2.3, $\varepsilon = 0.01$</td>
<td>21</td>
</tr>
<tr>
<td>16</td>
<td>Hybrid grid for example 2.3, $\varepsilon = 0.01$. Interpolation on three neighbours</td>
<td>22</td>
</tr>
<tr>
<td>17</td>
<td>example of $W_i^k$</td>
<td>27</td>
</tr>
<tr>
<td>18</td>
<td>Multiresolution of a smooth function: mixed grid</td>
<td>27</td>
</tr>
<tr>
<td>19</td>
<td>Multiresolution of a smooth function: interpolated solution</td>
<td>28</td>
</tr>
<tr>
<td>20</td>
<td>Multiresolution of a smooth function: exact solution</td>
<td>29</td>
</tr>
<tr>
<td>21</td>
<td>exact solution of the shock reflection problem.</td>
<td>29</td>
</tr>
<tr>
<td>22</td>
<td>Multiresolution of a reflected shock</td>
<td>30</td>
</tr>
</tbody>
</table>
Figure 17: Example of $V^E$.

Figure 18: Multiresolution of a smooth function: mixte grid
Figure 19: Multiresolution of a smooth function: interpolated solution
Figure 20: Multiresolution of a smooth function: exact solution

Figure 21: exact solution of the shock reflection problem.
Figure 22: Multiresolution of a reflected shock
Adaptive Multiresolution for Finite Volume Solutions of Gas Dynamics

A. Cohen, S.M. Kaber, M. Postel

Université Pierre & Marie Curie, Boîte courrier 187, 75252 Paris cedex 05, France

Abstract

Multiresolution analysis (MR) is used to improve the CPU and memory performance of a finite volume scheme. Departing from Harten’s original scheme we present a fully adaptive scheme in the sense that at a given time, the solution is represented in a compressed form by a set of significant wavelet coefficients. Numerical benchmarks for Euler’s system of compressible gas dynamics are performed on triangular meshes.

Key words: multiresolution analysis, finite volumes, adaptivity

1 Introduction

We are interested in solving systems of nonlinear hyperbolic conservation laws written in their general $2D$ form as

$$\partial_t u + \text{Div}_x(f(u(t,x))) = 0, \quad u \in \mathbb{R}^m, \quad x \in \mathbb{R}^2, \quad t > 0. \quad (1)$$

with initial value $u(t = 0, x) = u_0(x)$. The main difficulties are related to the appearance of discontinuities in the solutions of such equations. These discontinuities can appear in finite time even for smooth initial and boundary conditions data. The position of these discontinuities varies generally with time, which is of course an additional difficulty.

In order to represent accurately the discontinuous solutions it is necessary to discretize the computational domain on very fine grids. Furthermore, to ensure efficiency and high precision on these fine grids, costly nonlinear solvers are used. On the other hand, in the most frequent case, the discontinuities

Email address: postel@ann.jussieu.fr (M. Postel).
URL: www.ann.jussieu.fr/~postel (M. Postel).

Preprint submitted to Elsevier Preprint 17 July 2001
are localized in only a small region of the total domain. The fine discretization which is necessary in the vicinity of the discontinuities is superfluous elsewhere. The costly numerical fluxes (like ENO methods for instance) could be advantageously replaced by uniform higher order schemes wherever the solution is smooth enough.

The goal of this work is to use multiresolution strategies to build up an efficient and robust adaptive algorithm which solves the above difficulties. Strongly influenced by wavelet theory, multiresolution analysis of a function relies on its coding/decoding on a hierarchy of nested grids. Representations on consecutive grids provide means of compressing the solution as well as smoothness indicators. In the framework of hyperbolic conservation laws, the combination of multiresolution with finite volume method appears first in (Harten, 1994). Since this pioneering work, the idea was profitably explored and developed in many directions. The method was extended to Cartesian 2D grids in (Bihari and Harten, 1997) for scalar problems and more recently in (Chiavassa and Donat, 1999) for gas dynamics problems. In (Dahmen et al., 2000), a biorthogonal wavelet formulation is used and curvilinear grids are treated. In (Abgrall and Harten, 1994; Abgrall, 1997), and (Schröder-Pandet et al., 2000) the problem of unstructured grids for Finite Volume schemes is tackled while similar MR concepts are applied to triangular cell centered schemes in (Cohen et al., 2000). All these contributions have in common an inherent limitation which is the actual encoding of the solution on a very fine uniform grid imposed by the desired precision. We describe here a fully adaptive scheme where the solution is computed on a time dependent adaptive grid where the fine resolution level is used locally according to the local current smoothness of the solution. This approach has been studied in details in (Cohen et al., 2001) for the 1D case, both from the theoretical and numerical point of view in term of error analysis. We summarize briefly the method in the case of triangular Finite Volume and present the latest numerical developments consisting in 2D benchmarks for the compressible Euler equations.

2 Multiresolution meets Finite Volume

In the context of Finite Volume schemes, the natural discretization of (1) uses cell averages: the spatial domain is partitioned into “triangles” \((\Omega_\gamma)_{\gamma \in S}\), with \(S\) the set of indices. One is interested in computing approximations \(u^n_\gamma\) of the averages of \(u\) at discrete times \(n \Delta t\), over triangles \(\Omega_\gamma\). Applying the divergence theorem to (1) leads to the Finite Volume scheme

\[
u^{n+1}_\gamma = u^n_\gamma - \frac{\Delta t}{|\Omega_\gamma|} \sum_\mu |\Gamma_{\gamma, \mu}| F^n_{\gamma, \mu},
\]

(2)
where the numerical fluxes $F_{\gamma,\mu}^n$ are approximations of $\overline{F}_{\gamma,\mu}^n$ which denotes the average flux across interface $\Gamma_{\gamma,\mu} = \Omega_\gamma \cap \Omega_\mu$ between time steps $n\Delta t$ and $(n + 1)\Delta t$.

**Cell average multiresolution.**

Finite Volume multiresolution is based on nested finite volume discretizations. For $j = 0, 1, \cdots, J$, we are given regular disjoint partitions $(\Omega_\gamma)_{\gamma \in S_j}$ of $\mathbb{R}^2$ (or of a domain of interest) such that each $\Omega_\gamma$, with $\gamma = (k, j) \in S_j$, is the union of a finite number of cells $\Omega_\mu$, with $\mu = (k, j + 1) \in S_{j+1}$. The case of triangle subdivisions is represented in figure 1. In order to keep track of the scale level $j$ associated to an index $\gamma = (k, j)$, we shall make use of the notation $|\gamma| := j$ if $\gamma \in S_j$.

Consider a vector $U_j := (u_\gamma)_{\gamma \in S_j}$ of discrete data on the grid $S_j$ representing the cell-averages $u_\gamma := |\Omega_\gamma|^{-1} \int_{\Omega_\gamma} u(x) dx$ of some function $u \in L^1(\mathbb{R}^2)$.

![Figure 1. Subdivisions $(\Omega^{j+1}_k)_{k=0,\ldots,3}$ of a triangle $\Omega^j_0$ with neighbors $(\Omega^j_k)_{1 \leq k \leq 3}$](image)

The **projection** operator $P^j_{j-1}$ maps $U_j$ to $U_{j-1}$. Since the partitions $S_j$ are nested we obtain easily the averages at the coarser level $u_\gamma = |\Omega_\gamma|^{-1} \sum_{\mu} |\Omega_\mu| u_\mu$ with summation on $\mu \in \{ |\mu| = |\gamma| + 1, \Omega_\mu \subset \Omega_\gamma \}$. It is clear that one can derive $U_{j-1}, U_{j-2}, \cdots, U_0$ from the data $U_j$ by iterative application of the projection operators $P^j_{j-1}$.

The **prediction** operator $P^j_{j+1}$ maps $U_j$ to an approximation $\hat{U}_{j+1}$ of $U_{j+1}$. In contrast to the projection operator, there is an infinite number of choices for the definition of $P^j_{j+1}$. In all the numerical simulations we use the following linear reconstruction (3) fully detailed in Cohen et al. (2000),

$$
\begin{align*}
\hat{u}^{j+1}_{\Omega_0} &= u^j_0, \\
\hat{u}^{j+1}_{\Omega_1} &= u^j_0 + (u^j_2 + u^j_3 - 2u^j_1)/6, \\
\hat{u}^{j+1}_{\Omega_2} &= u^j_0 + (u^j_1 + u^j_3 - 2u^j_2)/6, \\
\hat{u}^{j+1}_{\Omega_3} &= u^j_0 + (u^j_1 + u^j_2 - 2u^j_3)/6.
\end{align*}
$$

(3)

It is suitably local (the stencil $\Sigma_\lambda$ of cells in $S_{\lambda-1}$ used to predict $u_\lambda$ contains only four triangles) and consistent ($P^j_{j+1}P^j_{j+1} = \text{Identity}$). We can define the prediction error at level $j$ as the differences between the exact and predicted values, $d^j_\mu := u_\mu - \hat{u}_\mu$. Using the consistency assumption, we define the detail vector $D_j = (d^j_\mu)_{\mu \in \Sigma_j}$, where $\Sigma_j$ is a selection of details such that there is a one to one correspondence between $U_j$ and $M_j = (U_0, D_1, D_2, \cdots, D_J)$. Using the
local structure of the projection and prediction operators, we can implement the multiscale transformation $M : U_J \mapsto M_J$, and its inverse $M^{-1}$ with optimal complexity $O(N_J)$, where $N_J := \#(S_J)$ represents the cardinality of $U_J$.

The main interest of decomposing $U_J$ into $M_J$ is that this new representation is often more appropriate for data compression. Given a set $\Lambda \subset \mathcal{V}^J$ of indices $\lambda$, we define a truncation operator $T_\Lambda$ acting on multiscale representations, that leaves unchanged the component $d_\lambda$ if $\lambda \in \Lambda$ and replaces it by 0 otherwise. In practice, we are typically interested in sets $\Lambda$ obtained by thresholding: given a set of level-dependent threshold $(\varepsilon_0, \varepsilon_1, \cdots, \varepsilon_J)$, we set $\Lambda = \Lambda(\varepsilon_0, \varepsilon_1, \cdots, \varepsilon_J) := \{ \lambda \text{ s.t. } |d_\lambda| \geq \varepsilon_{|\lambda|} \}$, and define the corresponding thresholding operator $T_\Lambda$. Applying $T_\Lambda$ on the multiscale decomposition of $U_J$ amounts in building a nonlinear approximation $A_\Lambda U_J$, where the operator $A_\Lambda$ is given by $A_\Lambda := M^{-1}T_\Lambda M$.

Both the accuracy and the stability of the thresholding scheme (3) have been extensively studied in (Cohen et al., 2000). The smoothness of the solution provides a bound for the details $|d_\lambda|$. A level dependent threshold $\varepsilon_j := 2^{(j-j_0)}\varepsilon$, allows to control the effect of thresholding by the following estimate $\|U_J - A_\Lambda U_J\| \leq C\varepsilon$. It is very important that the set of preserved indices $\Lambda$ has a tree structure to ensure the isomorphism property of $M$ and furthermore some gradedness property in order to ensure good computational properties to the encoding/decoding algorithm. In practice it consists in adding to the tree all the triangles belonging to the reconstruction stencils of the triangles $\Omega_j$ already in the tree because $|d_\lambda| \geq 2^{(j_0-j)}\varepsilon$ and we denote by $\Lambda_\varepsilon$ the resulting set of indices.

3 From Harten’s scheme to fully adaptive schemes
To link the multiresolution analysis with the Finite Volume scheme presented above, we rewrite (2) on the finest resolution level as
\[ V_{J}^{n+1} = V_J^n - B_J^n := E_J V_J^n, \]
where $V_J^n := (u_J^\gamma)_{\gamma \in S_J}$ is the vector representing the Finite Volume numerical solution at time $n\Delta t$ and $B_J^n := (b_J^\gamma)_{\gamma \in S_J}$ with $b_J^\gamma := \frac{\Delta t}{|\Omega_j|} \sum_{\mu} |\Gamma_{\gamma,\mu}| F_{\gamma,\mu}^n$ the numerical flux balance for the cell $\Omega_j$. The increment $b_J^\gamma$ depends locally and nonlinearly on the numerical solution. If $\overline{U}_J$ is the cell average vector of the exact solution $u$ at time $n\Delta t$, the numerical error incurred using standard Finite Volume is $e_n := \|V_J^n - \overline{U}_J^n\|$. The goal of our multiresolution scheme is to compute a numerical solution $\hat{U}_J^n$ with significant CPU and memory gain over the reference Finite Volume scheme, while the additional error $a_n := \|\hat{U}_J^n - V_J^n\|$ remains within a prescribed accuracy so that $\|\hat{U}_J^n - \overline{U}_J^n\| \approx e_n$. 

4
Both Harten’s multiresolution scheme and our fully adaptive scheme are based on the intuitive idea, introduced in (Harten, 1994), that the set of significant wavelet coefficients of the numerical solution evolves “slowly” from one time step to the other. More precisely, if $\Lambda^n_\varepsilon$ is the graded tree obtained from the application of the thresholding to some numerical approximation $U^n_j$ of $\overline{U}^n$, one can enlarge $\Lambda^n_\varepsilon$ into a graded tree $\tilde{\Lambda}^{n+1}_\varepsilon$ which contains both $\Lambda^n_\varepsilon$ and $\Lambda^{n+1}_\varepsilon$ so that, if $U^n_{j+1} = E_j U^n_j$, we have $\|U^n_j - A^n_{\tilde{\Lambda}^{n+1}_\varepsilon} U^n_j\| \le C \varepsilon$ and $\|U^{n+1}_j - A^n_{\tilde{\Lambda}^{n+1}_\varepsilon} U^n_j\| \le C \varepsilon$.

This heuristics is justified rigorously in the uniform Cartesian grid case (Cohen et al., 2001) by imposing demanding rules to the enlarging procedure. However at this moment it has not been yet extended to the unstructured triangular grid and we therefore present numerical results obtained using the following enlarging procedure initially described by Harten

- If $\Omega_\Lambda$ is in $S_{\tilde{\Lambda}^n_\varepsilon}$, then its neighbors at the same scale (all triangles sharing one or two vertices with $\Omega_\Lambda$) are included in $S_{\tilde{\Lambda}^{n+1}_\varepsilon}$.
- If $\Omega_\Lambda$ is in $S_{\tilde{\Lambda}^n_\varepsilon}$ and if $|d_\Lambda| > 4 \varepsilon$, then its four children are also included.

By (4), we see that Harten’s heuristics also implies $\|B^n_j - A^n_{\tilde{\Lambda}^{n+1}_\varepsilon} B^n_j\| \le C \varepsilon$, i.e. the flux balance is also well represented by $\tilde{\Lambda}^{n+1}_\varepsilon$. Given a prescribed tolerance $\varepsilon > 0$, the scheme proposed in (Harten, 1994) consists in using the compressed vector $A^n_{\tilde{\Lambda}^{n+1}_\varepsilon} B^n_j$ in place of $B^n_j$: the cell averages are now evolved according to $U^n_{j+1} = U^n_j - A^n_{\tilde{\Lambda}^{n+1}_\varepsilon} B^n_j$. Of course, $B^n_j$ is now the numerical flux balance computed from $U^n_j$ which differs from $V^n_j$. As explained in the introduction, the goal of this modification is to save computational cost through a smaller number of numerical flux evaluations. Harten’s scheme achieves this up to a point since the solution is nevertheless computed everywhere on the uniform finest grid, although in smooth area, it is computed using interpolation of the solution computed on coarser grids. In contrast, the adaptive scheme presented below operates on a compressed representation of the numerical solution $U^n_j$: at time step $n \Delta t$, the non-zero detail coefficients in the multiscale decomposition of $U^n_j$ are confined to a graded tree $\Lambda_n \subset \nabla J$, so that $U^n_j$ is exactly represented by its coefficients $(d^n_\Lambda)_{\Lambda \in \Lambda_n}$ or by its cell averages $(u^n_\Lambda)_{\Lambda \in S(\Lambda_n)}$ on the corresponding adaptive discretization.

Given $\Lambda_n$ and $U^n_j$ (represented by $(d^n_\Lambda)_{\Lambda \in \Lambda_n}$ or $(u^n_\Lambda)_{\Lambda \in S(\Lambda_n)}$), we derive $\Lambda_{n+1}$ and $U^n_{j+1}$ by the following steps:

- **Refinement.** A new set $\tilde{\Lambda}_{n+1}$ containing $\Lambda_n$ is constructed based on the magnitude of the coefficients $|d^n_\Lambda|, \Lambda \in \Lambda_n$ according to the growing rules described above. The vector $(d^n_\Lambda)_{\Lambda \in \Lambda_n}$ is extended by setting $d^n_\Lambda = 0$ for $\Lambda \in \tilde{\Lambda}_{n+1} \setminus \Lambda_n$. Applying $M^{n+1}_{\tilde{\Lambda}_{n+1}}$, we derive the refined averages $(u^n_\Lambda)_{\Lambda \in S(\tilde{\Lambda}_{n+1})}$.
- **Computation.** A first numerical solution $\tilde{U}^{n+1}_j$ at time $(n+1) \Delta t$, discretized on $S(\tilde{\Lambda}_{n+1})$, is computed by $\tilde{u}^{n+1}_\Lambda = u^n_\Lambda - \tilde{b}^n_\Lambda, \Lambda \in S(\tilde{\Lambda}_{n+1})$. The adaptive flux
balance vector $(\mathbf{b}^n \lambda, \tilde{u}^n \lambda, \tilde{u}^n \lambda)^T$ is directly computed from $(u^N \lambda, \tilde{u}^N \lambda, \tilde{u}^N \lambda)^T$. In the 1D case we have devised a fast method to reconstruct locally the solution on the finest level. It is based on the uniformity of the fine grid and cannot be extended simply to the unstructured grid case. We use instead a linear reconstruction of the solution on the edges of the triangles where we need the fluxes $F^n_{\gamma_n}$ entering in the evaluation of $\mathbf{b}^n \lambda$ (see (Durlofsky et al., 1992)). This scheme is not strictly in the scope where the error bound $a_n < \varepsilon$ obtained in (Cohen et al., 2001) can be used even though the numerical experiments tend to show that it preserves the overall accuracy in a similar way.

- **Thresholding.** Applying $\mathcal{M}_{\Lambda_{n+1}}$ to $(\mathbf{b}^n \lambda, \tilde{u}^n \lambda, \tilde{u}^n \lambda)^T$, we derive $(d^n \lambda, \tilde{u}^n \lambda, \tilde{u}^n \lambda)^T$. We define $U^+_{\gamma_{n+1}}$ by thresholding $\tilde{U}^n \lambda$ according to $U^+_{\gamma_{n+1}} = A \tilde{U}^n \lambda$, and the new set $\Lambda_{n+1} \subset \Lambda_{n+1}$ to be the corresponding set $\Lambda_{\gamma}$ of preserved coefficients.

### 4 Numerical tests

We consider the 2D Euler equations of gas dynamics

$$
\partial_t \mathbf{u} + \nabla \cdot \mathbf{f}(\mathbf{u}) + \partial_z \mathbf{g}(\mathbf{u}) = 0, \quad \text{with} \quad \mathbf{u} = (p, u, v, e)^T,
$$

$$
\mathbf{f}(\mathbf{u}) = \mathbf{u} p + (0, 1, 0, u)^T, \quad \mathbf{g}(\mathbf{u}) = \mathbf{v} p + (0, 0, 1, v)^T,
$$

where $\rho$ is the density of the gas, $(u, v)^T$ its velocity, and $p$ its pressure. The energy $e = p/(\gamma - 1) + \rho(u^2 + v^2)/2$ with $\gamma = 1.4$. We first study the reflection of an oblique shock on the lower side of a rectangular domain: $0 \leq x \leq 4.129, 0 \leq y \leq 1$. The initial flow is given by

- $\rho = 1, \quad u = 2.9, \quad v = 0$ and $p = 1/1.4$.

The boundary conditions are:

- inflow on the side $x = 0$ (those of the initial flow).
- outflow on the side $x = 4.129$.
- reflection on the lower side $y = 0$. 

![Fig. 2. Performances of MR, memory (triangles) and CPU (s), L1 error for Roe flux (left) and 2nd order flux (right)](image-url)
Fig. 3. Hybrid grid at time $t=4$, four levels at most, $\varepsilon = 0.1$, 12349 triangles.

- fixed values on the upper side $y = 1$ (those of the exact solution):
  
  $\rho = 1.7$, $\quad u = 2.61932$, $\quad v = -0.506339$ and $\quad p = 1.52824$

  The exact solution is formed by an incident shock with angle $29^\circ$ and a reflected
  shock with angle $23.28^\circ$. The solution after the second shock is:
  
  $\rho = 2.68732$, $\quad u = 2.40148$, $\quad v = 0$ and $\quad p = 2.93413$

  The coarsest grid is composed of $N_0 = 820$ triangles. Four levels at most are
  used for the computations summarized in figure 2 in terms of memory, CPU
  and relative L1-error computed with respect to the Finite Volume solution on
  the finest grid which has 69,700 triangles. The three curves exhibit the expected
  trends in the 1st order Roe scheme case. The error curve in the second order
  flux case does not decay as well as in the 1st order case for small $\varepsilon$ values.
  The reference solution is closer to the piecewise constant exact solution in that
  case which makes the thresholding of details less sensitive to $\varepsilon$.

  An example of hybrid grid is represented in figure 3 which shows that the
  shock is well located. More details on this numerical simulation are available
  on the web page. The second experiment simulates a supersonic flow past a
  double ellipse (see Arminjon et al. (1997)) starting from a constant field
  $\rho = 1$, $p = 1/5.6$ and a unit velocity with an angle of attack $20^\circ$ with the
  horizontal and $M_\infty = 2$. In figure 4 we show the coarse grid and the hybrid
  grid obtained with a threshold at time 0.1. The triangles with two vertices on
  the ellipse boundary are always subdivided down to the finest level. At each
  level the mid point for the two vertices lying on the ellipse is projected on the
  ellipse. Since this contradicts the consistency assumption of the prediction,
  the thresholding is disactivated on these cells. Nevertheless, the memory gain
  is 7.28 and the CPU gain 3.2 with comparison to the Finite Volume solution
  computed on the uniform finest grid.

5 Conclusion

The good performance of the numerical tests illustrates the feasibility and in-
terest of the method. However this also indicates the need for a better underly-
ing Finite Volume scheme, which would better compare with the uniform fine
resolution in the smooth areas. Work in this direction is currently in progress.
References


Multiresolution analysis on triangles: application to conservation laws

A. Cohen, S.M. Kaber, M. Postel
Laboratoire d'analyse numérique
Université Pierre et Marie Curie (Paris 6), France
http://www.anm.jussieu.fr

Abstract A multiresolution algorithm is coupled with a finite volume scheme to solve scalar bidimensional conservation laws. The originality resides in the adaptivity of the multiresolution decomposition, which takes into account the possible appearance of discontinuities and their displacement. Numerical simulations on triangular meshes point out the advantages of the method in terms of CPU and memory costs.

Key Words: Multiresolution - finite volumes - adaptive scheme.

1. Introduction

We are interested in this work in solving conservation laws on polygonal domains. It is well known that such equations can develop localized discontinuities in finite time. In such areas where the solution is not smooth a fine resolution is necessary and furthermore, higher order schemes for flux computations are necessarily nonlinear - including for instance ENO reconstruction. These are very costly techniques and it seems reasonable to use all available information on the local smoothness of the solution to decide whether they should be used or not. This can be done within multiresolution framework. The original idea of combining the advantages of this method - data compression, smoothness indicators - into conservation laws solvers in order to reduce the number of flux computations is due to Harten [Har94]. Initially implemented in one dimension, it was then extended to bidimensional cartesian grids, and smooth deformations of rectangular grids, (see references in [CDKP99]). Preliminary approach for unstructured grid is explained in [Aeb97]. In the case of triangular meshes, which are more flexible in modeling complex geometries, a complete implementation with a detailed analysis of the encoding/decoding algorithm can be found in [KP99, CDKP99].

In this former approach, the solution is represented everywhere on a uniform fine grid and the multiscale analysis is used to speed up the flux computations in the smooth areas. What we propose here is a fully adaptive scheme which
make use of the compressed solution - encoded at each time step by its most significant coefficients. This approach gives way to new difficulties, for instance in the analysis of the stability and precision. These points have been studied in detail in the one dimension case in [CKMP99] and are currently under investigation in the triangular meshes case.

The outline of the paper is as follows: we recall in section 2 the multiresolution and the finite volume algorithms for triangular meshes which are then combined to produce the adaptive scheme presented in section 3. Numerical simulations and analysis of performances on test cases follows in section 4.

2. Multiresolution analysis and finite volumes

We briefly describe a multiscale transformation of a function described by its mean - and not point wise as usual - values on a triangular mesh. The equation is defined on a polygonal domain $\Omega$ which is discretized by a coarse grid $\Omega^0$ made of $N_0$ triangles. A hierarchy of nested grids $\Omega^\ell$ ($0 \leq \ell \leq L$) is built. The grid $\Omega^{\ell+1}$ is obtained by dividing each triangle of $\Omega^\ell$ into four smaller triangles by joining the three edges midpoints. Triangles from the level $\ell$ are denoted by $T^\ell_k$ for $1 \leq k \leq N_\ell = 4^\ell N_0$. We denote by $\bar{u}^\ell_k$ the average of a function $u$ on the triangle $T^\ell_k$ and $\bar{u}^\ell = (\bar{u}^\ell_k)^{N_\ell}_{k=1}$. The mean values $\bar{u}^{\ell+1}$ of a function $u$ being given on the $\Omega^{\ell+1}$ grid, the mean values on the coarser grid $\Omega^\ell$ can be computed by

$$\bar{u}^\ell_i = \frac{1}{|T^\ell_i|} \sum_{j=0}^3 |T^\ell_{i,j}| \bar{u}^{\ell+1}_{i,j},$$

where $T^\ell_{i,j}$ are the four triangles of $\Omega^{\ell+1}$ obtained by subdividing the triangle $T^\ell_i$. The center triangle is denoted by $T^\ell_{i,0}$. The relation (1) defines a projection operator $P^{\ell+1}_T$ from the resolution level $\ell + 1$ on the level $\ell$ by $P^{\ell+1}_T \bar{u}^{\ell+1} = \bar{u}^\ell$. Multiscale decomposition relies on the existence of this operator as well as a prediction operator $Q^{\ell+1}_T$ from the resolution level $\ell$ to the level $\ell + 1$. They should be compatible in the sense that $P^{\ell+1}_T Q^{\ell+1}_T = I_d$.

The predicted values $\bar{u}^{\ell+1} = Q^{\ell+1}_T \bar{u}^\ell$ are approximations of the exact mean values $\bar{u}^{\ell+1}$ obtained by local linear combinations of values at level $\ell$. Details on every triangle are defined as the differences between exact and predicted values $d^{\ell+1}_{i,j} = \bar{u}^{\ell+1}_{i,j} - \bar{u}^{\ell+1}_{i,j}$, for $j = 1, \ldots, 3$. The multiscale representation of $\bar{u}^L$ is denoted by $\bar{u}_{MR} = (\bar{u}^0, d^1, \ldots, d^{L-1})$. Going from $\bar{u}^L$ to $\bar{u}_{MR}$ and reverse is achieved through wavelet-type algorithms described in [CDKP99]. The two types of data representation have the same memory requirement. However, in the multiscale representation case, the amount of data can be reduced by thresholding, the sufficiently small details. More precisely this consists in setting $d^\ell_{i,j} = 0$ wherever $|d^\ell_{i,j}|$ is smaller than a level dependent tolerance $\varepsilon^\ell$. We refer to [CDKP99] for justification. In the same reference, the stability and
The convergence of a class of reconstruction operators are studied. This leads to the following optimal choice for $Q_{l+1}^e$

$$
\begin{align*}
\tau_{0,0}^e &= \tilde{u}_0^e, \\
\tau_{0,1}^e &= \tilde{u}_0^e + \frac{1}{6} (\tilde{u}_0^e + \tilde{u}_0^e - 2 \tilde{u}_0^e), \\
\tau_{0,2}^e &= \tilde{u}_0^e + \frac{1}{6} (\tilde{u}_0^e + \tilde{u}_0^e - 2 \tilde{u}_0^e), \\
\tau_{0,3}^e &= \tilde{u}_0^e + \frac{1}{6} (\tilde{u}_0^e + \tilde{u}_0^e - 2 \tilde{u}_0^e).
\end{align*}
$$

(2)

We now recall how the numerical treatment of the 2D scalar conservation law

$$
\partial_t u + \text{div} f(u) = 0,
$$

(3)
can be improved within the multiresolution framework. The unknown $u(x, y, t)$ is defined on a polygonal set $\Omega$, for $t > 0$. An initial condition $u(x, y, 0) = u_0(x, y)$ is imposed as well as boundary conditions on the edges of $\Omega$. We first consider that equation (3) is solved by a finite volumes scheme on the fine grid $\Omega^L$. The costly step of this scheme is the flux evaluation $\tilde{F}_{k,n}^L \approx \frac{1}{|T_k|} \int_{T_k} \text{div} f(R) \, dx dy$. Since the solution can present discontinuities - even with smooth initial data - the computations may have to be locally very precise. On the other hand, the point wise evaluation of a function known by its mean values requires a reconstruction step (using the operator $R$ in the flux evaluation, not to be mistaken with the prediction $Q_{l+1}^f$ defined by (2)). This step can also be very costly if ENO techniques are used for instance. It is therefore crucial to limit the number of these flux computations and this is where multiscale analysis comes into play. In Harten’s framework, the details provided by the multiscale decomposition of the solution are used as criteria to choose the flux computation method as follow. Starting from the coarsest level where fluxes on all triangle edges are precisely computed. The levels are explored in turn: wherever details are large, fluxes are computed using precise and costly numerical flux approximations $\tilde{F}_k = \frac{1}{|T_k|} \sum_j F_{k,j}^L$, where $\sum_j F_{k,j}^L$ denotes the numerical flux across the edge $\Gamma_{k,j}$ (of length $|\Gamma_{k,j}|$) between two triangles belonging to the finest grid $\Omega^L$. Elsewhere, the mean values of flux divergences are computed by interpolating the same quantities on the immediately coarser level, using the prediction operator $Q_{l+1}^f$. This algorithm was successfully implemented in [KP99, CDKP99]. It leads to a CPU reduction by a factor 2 to 3 depending on the test case. Its precision is governed by the precision of the underlying finite volume scheme on $\Omega^L$ since at each time step the time evolution of the solution is performed on this grid. Therefore the gain in CPU time remains linked to this discretization,
and there is no memory saving. A way to overcome these two limitations is
provided by a fully adaptive scheme in which the solution is computed on a
nonuniform time dependent grid. In the next section we modified the initial
scheme in order to compute the solution on such a hybrid grid, fine where large
fluctuations are foreseen and coarse in the smooth areas. We refer to [CKMP99]
for a detailed analysis of the algorithm in the one dimensional case. We are
principally interested here in showing the potential of the method in the case
of multidimensional grids - and more precisely triangular ones.

3. Fully adaptive schemes

We introduce a hybrid grid $\mathcal{H}_\ell^0 \subset \cup_{\ell} \Omega^\ell$ defined at time step $t_n$ by selecting
subsets out of each $\Omega^\ell$. We will show how this grid is built in such a way that it
reflects the smoothness of the solution. Since this regularity is time dependent,
the hybrid grid will have to evolve in time, this will also be described. Eventu-
ally we will explain how the finite volume scheme is performed on such a grid,
so that both the solution mean values and the details in the multiresolution
analysis evolve correctly in time.

We first describe how $\mathcal{H}_\ell^0$ is built using the initial data $u_0$. It is initially set
equal to $\Omega^0$. Until the maximum level of resolution if reached, each triangle
of $\mathcal{H}_\ell^0$ is tested to decide whether its subdivisions should be added to $\mathcal{H}_\ell^0$.
The criterion is the comparison of the mean value of the solution on the four
subdivisions with the values predicted by interpolation $Q^\ell_{\ell+1}$. A smoothness
indicator $R^\ell_{i,j}$ associated to a triangle $T^\ell_{i,j}$ is activated ($R^\ell_{i,j} = 1$) if one of the four
details $d^\ell_{i,j}$ is greater in absolute value than the threshold tolerance $\varepsilon_{\ell}$. The
triangles whose indicators are activated are actually divided into four triangles
who are themselves included into the computing grid $\mathcal{H}_\ell^0$. Another requirement
is to preserve the grid nestedness. As a corollary we must always be able to
interpolate at the finer level. This has two consequences:

- If a triangle $T^\ell_{i,j}$ is divided because $R^\ell_{i,j} = 1$, all its neighbors on the same
  level $\ell$ are included in $\mathcal{H}_\ell^0$, even if their smoothness indicators are not activated.

- An auxiliary grid $Q^0_{\ell}$ containing $\mathcal{H}_\ell^0$ is defined: If a triangle belongs to the
  grid $\mathcal{H}_\ell^0$ for one of the two previous reasons, then all its neighbors at level $\ell$
  belong to $Q^0_{\ell}$. On triangles belonging to $Q^0_{\ell} \setminus \mathcal{H}_\ell^0$ no actual computation
  takes place and the mean values of the solution will be estimated by $Q^0_{\ell+1}$.

When the initial data is given by its dual multiresolution representation,
pathological cases with non zero details only on the fine level can occur. To
handle such cases, all the triangles should be actually divided up to the finest
level, even if the details are negligible at a coarser level, and the thresholding
should be done iteratively starting from the finest level. This is in practice very
difficult to realize without giving up all the advantage of the multiresolution
in terms of memory savings. In the test cases we have studied so far, the initial
solution is given by an explicit expression, which ensures that the tree structure
of $\mathcal{H}_\ell^0$ is achieved by the previous algorithm which explores and refines the levels
starting from the coarsest.

The hybrid grid may have to be modified at each time step in order to follow the discontinuities propagation and capture shocks formation. To this effect, we define a larger grid $\mathcal{H}_t^{n+1}$, containing $\mathcal{H}_t^n$, whose time evolution can be predicted. We use here the hyperbolicity of the problem ensuring that if the CFL condition is respected the discontinuities do not propagate further than one space step in one time step. To foresee possible propagation in any direction the term of neighbors of a triangle is extended to all triangles sharing a vertices with it, and not only the three triangles having a common edge and participating in the interpolation scheme $\mathcal{Q}_t^{\ell+1}$. A corresponding grid $\mathcal{G}_t^{n+1}$ containing $\mathcal{H}_t^{n+1}$ as well as all its neighbors at each level is also defined. At each time step the mean values of the solution on the coarsest grid and the details corresponding to triangles in $\mathcal{H}_t^n$ are modified (as detailed in algorithm 1). Whenever one of the four details of a triangle of $\mathcal{H}_t^n$ is higher than the level tolerance, its smoothness indicator and that of all its neighbors is activated. If a detail is higher than a still larger tolerance, the corresponding indicators at the finer level are also activated. These operation can make triangles previously in $\mathcal{G}_t^n \setminus \mathcal{H}_t^n$ part of $\mathcal{H}_t^{n+1}$ and even create new subdivisions at the finer level. The nestedness of the new grid $\mathcal{H}_t^{n+1}$ is ensured as in the initial step - which can induce the creation of subdivisions in $\mathcal{G}_t^{n+1}$. Eventually some triangles of $\mathcal{G}_t^{n} \setminus \mathcal{H}_t^n$ may now be useless and therefore removed.

An important difference with Harten's algorithm is that the computing grid $\mathcal{H}_t^n$ is incomplete. In other words, the depth of resolution is spatially variable and defined for a triangle $T^l$ of level $\ell$ by a local depth $\hat{\ell}$. This has direct consequences on the precision of scheme. In both approaches the precise numerical flux evaluations are performed only on the interfaces of the hybrid grid. In Harten's case the solution is everywhere available on the finest grid so the maximum precision can be achieved. In the adaptive case, the solution could theoretically be computed down to the finer level by applying the $\mathcal{Q}_t^{\ell+1}$ as many times as necessary starting from $\hat{\ell}$ where it is actually known. This would be completely inefficient in practice since it would destroy all the benefits of the multiresolution. On the other hand, brutal use of the available mean values on the hybrid grid to compute the fluxes leads to an order of approximation governed by the coarser level. This problem is explained in details in the one dimension case in [CKMP99] and we simply summarize here the main conclusions. We have so far two ways to tackle this difficulty:

• In one dimension it is possible to reconstruct directly - that is linearly - mean values at the finest level from values at a not immediately coarser level - as soon as the details are null on a sufficient number of consecutive intervals on this coarse level (four intervals in the case of linear reconstruction). Taking into account the areas of high gradient where actual decoding must be performed leads to a complexity in $N \log(N)$ where $N$ is the cardinal of $\mathcal{H}_t^n$.

• The second method consists in using a higher order ENO type scheme to compute the fluxes at points where the solution is known by its mean values on
a coarse or intermediate grid level. In that case the complexity is that of the hybrid grid multiplied by a fixed coefficient which depends on the complexity of the ENO reconstruction. The precision is that of the coarse grid to a power related to the ENO reconstruction order.

From the parameter study performed in [CKMP99], it seems that the first method is the most efficient one if we compare the performances for a given accuracy. However its extension to the triangular grid case seems more complicated. In this preliminary implementation we have used the same ENO type reconstruction as in [CDKP99]. We summarize in the following adaptive algorithm (1), how the finite volumes scheme is used to compute the time evolution of the solution and the details of the multiresolution on the hybrid computing grid $\mathcal{H}_e$.

**Algorithm 1** Finite volumes + adaptive multiscale

- Initialization of $\mathcal{H}_e$ and $\mathbf{u}^0_{MR}$
- Loop on time step $n = 0, 1, \ldots$

\[
\begin{cases}
\begin{aligned}
\text{Determination of } & \tilde{\mathcal{H}}_{n+1}^l \\
\text{Partial decoding} & \\
\text{Evolution on the coarse grid: } & \forall \mathbf{T}_k^l \in \Omega^l \cap \tilde{\mathcal{H}}_{n+1}^l \\
\begin{aligned}
\tilde{\mathbf{p}}_{k,j}^n &= \frac{1}{\Omega_{k,j}^l} \sum_j \mathbf{p}_{k,j}^l \tilde{\mathbf{r}}_{k,j}^l \\
\mathbf{u}_{k,j}^{n+1} &= \tilde{\mathbf{u}}_{k,j}^n - \Delta t \tilde{\mathbf{f}}_{k,j}^n
\end{aligned} \\
\end{aligned}
\end{cases}
\]

- Loop on levels $l = 0, \ldots, L - 1$

\[
\begin{cases}
\begin{aligned}
\forall \mathbf{T}_k^l \in \Omega^l \cap \tilde{\mathcal{H}}_{n+1}^l & / \mathbf{T}_{k,j}^{l+1} \in \tilde{\mathcal{H}}_{n+1}^l \\
\begin{aligned}
\tilde{\mathbf{p}}_{k,j}^{l+1,n} &= \frac{1}{\Omega_{k,j}^{l+1}} \sum_j \mathbf{p}_{k,j}^{l+1} \tilde{\mathbf{r}}_{k,j}^{l+1} \\
\mathbf{u}_{k,j}^{l+1,n} &= \tilde{\mathbf{u}}_{k,j}^{l+1,n} - \Delta t \tilde{\mathbf{f}}_{k,j}^{l+1,n}
\end{aligned} \\
\end{aligned}
\end{cases}
\]

4. Numerical experiments

To illustrate the performances of the algorithm we first present the test case of a linear flux on an initial discontinuous solution. We choose the function $u_0(x, y) = 1$ if $\sqrt{x^2 + y^2} < 0.1$ and zero elsewhere on the unit square with periodic boundary conditions. Performances in term of CPU time, memory requirements and precision are evaluated for the adaptive multiresolution scheme and the reference finite volumes scheme on the finest level. The numerical fluxes are computed with the second order ENO method already implemented in [KP99, CDKP99]. It requires a small CFL number chosen here equal to 0.2 on the finest grid. There are 50 triangles on the coarsest level. We first show results corresponding to a multiresolution decomposition on a maximum of 5 levels. In this case the finest grid on which the finite volumes reference solution is computed counts 12800 triangles. The direction of propagation is parallel to
the Oz axis and the solution is computed over one period of time. Figure 1 shows the initial solution and the solution after one period computed with the adaptive scheme on a maximum of five levels with a total of 1209 triangles. The shape of the solution is well preserved, and the finer levels are used only in the vicinity of the discontinuity. On the next group of figures 2, the performances of the adaptive scheme are displayed for different maximum number of levels. For each value of this parameter \( L \), the standard finite volume solution on the finest grid - level \( L \) - is computed. CPU time and memory requirements along with the error with the exact solution in the \( L_1 \) norm can be compared. The precision of the adaptive scheme in this case is very satisfying since the two error curves are always close to each other. Actually, in this test case, the source of possible errors is the discontinuity and in its vicinity both schemes are similarly discretized. As far as the memory requirement are concerned the advantages of the adaptive case are indisputable. We are actually restricted to a few number of levels because the finite volume scheme will not run on our work station for a grid finer than the 5th level (\( L = 4 \)). We can predict that for 204800 triangles it would require about 13 hours of CPU. In this limit case \( L = 6 \), the adaptive scheme actually discretizes the solution on 2198 triangles, among which only 1228 belong to the finest grid and takes only 1.5 hour to run. One remarks that the CPU time does not depend linearly upon the number of triangles because some expensive overhead computing must be done which is inherent to the multiscale decomposition like the creation and the deletion of triangles. The adaptive scheme is nevertheless much faster than the standard finite volume scheme, all the more so as the number of levels in the multiresolution increases.

5. Conclusions

A fully adaptive scheme based on the multiresolution decomposition of the solution is proposed to improve the performances of a standard finite volume algorithm. The mean values of the solution are computed on a time varying grid made of cells belonging to different levels of decomposition according to the local smoothness. The preliminary numerical tests which are presented exhibit very good performances in term of CPU and memory savings. The immediate future works consists in developing the vector case algorithm necessary to handle gas dynamics. From a more theoretical point of view, we are currently studying the convergence of the scheme.

Bibliography

Figure 1: Multiresolution solution on five levels at time $t=0$ and $t=1$ (one period)

Figure 2: Comparison of CPU times, memory occupation and precision for finite volumes and multiresolution algorithms


Multiresolution analysis on triangles: 
application to gas dynamics

Albert Cohen, Sidi Mahmoud Kaber, Marie Postel

Abstract. Multiresolution analysis is used to improve the performances of a Finite Volumes scheme. Two schemes coupling Multiresolution and Finite Volumes are presented. One is a generalization of Harten’s original scheme for triangles. The other scheme is fully adaptive in the sense that at a given time, the solution is represented in a compressed form by a set of significant wavelets coefficients. The two schemes are applied to solve the Euler’s system of gas dynamics.

1. Introduction

The main difficulties in solving nonlinear conservation laws arise from the discontinuities which the solutions of such equations may develop in finite time. The fact that the position of these discontinuities varies generally with time is of course an additional difficulty. In order to represent accurately the discontinuous solutions it is necessary to discretize the computational domain on very fine grids. Furthermore to ensure efficiency and high precision on these fine grids costly nonlinear solvers are used. In many problems the discontinuities are localized in only a small region of the total domain. The fine discretization which is necessary in the vicinity of the discontinuities is in fact a luxury everywhere else. The costly numerical fluxes could be advantageously replaced by simpler solvers wherever the solution is smooth enough. A. Harten proposed to use multiresolution analysis (MR) in order to speed up Finite Volume (FV) schemes. See [H] and the references therein, in particular [A] for applications to triangular meshes.

Let us detail the original strategy proposed in [H]. At the start one is given a FV scheme associated to a grid \( \Omega^L \) which is the finest one in a hierarchy of nested grids \( \Omega^\ell \) for \( \ell = 0, \ldots, L \). At time \( t_n \) the approximate solution is represented by its averages \( \langle \bar{u}^n \rangle_k \) on the various cells of \( \Omega^L \). The values \( \langle \bar{u}^{n+1} \rangle_k \) are evaluated by the FV scheme through the computations of the fluxes on \( \Gamma^L \), the set of all interfaces between the cells of \( \Omega^L \). The basic idea is to use a wavelet-like multiscale decomposition of the solution at time \( t_n \) as a smoothness indicator in order to reduce the computations of the fluxes on \( \Gamma^L \). In the regions where the details-defined below by (2)- are small (i.e. below some preassigned threshold), the flux is assumed to
be smooth enough so that we can replace its exact evaluation on the finest level \( L \) by an interpolation from its values on the coarser levels (see Algorithm 2). This saves CPU since the precise computation of the flux is expensive, while the interpolation from coarse to fine meshes is cheap.

In section 2, we define a MR transform suited for triangulations. In section 3, we combine it with a FV scheme to obtain an extension of Harten’s MR scheme to triangles. In section 4, we present an adaptive scheme that takes full advantage of the multiscale decomposition of the solution.

2. Multiresolution analysis (MR) on triangles

In [CDKP], we have defined a multiscale method adapted to cell-averages given on triangular meshes. Starting from a coarse triangulation \( \Omega^0 \), we define a hierarchy of nested grids \( \Omega^\ell (1 \leq \ell \leq L) \) by dividing each triangle \( T_k \) of \( \Omega^0 \) into 4 triangles \( \{T_{k,j}\}_{j=0}^3 \) of \( \Omega^{\ell+1} \). The mean value of the function \( u \) on the triangle \( T_k \) is denoted by \( \bar{u}_k^\ell = \mathbb{A}(T_k^\ell) u = \frac{1}{|T_k^\ell|} \int_{T_k^\ell} u \) and the array of all \( (\bar{u}_k^\ell)_{k=1,...,N_T} \) is denoted \( \bar{u}^\ell \).

2.1. The Multiscale decomposition

Two fundamental operators are defined.

- A projection operator \( \Psi_\ell^{\ell+1} \) from the resolution level \( \ell + 1 \) to the coarser one \( \ell \), that maps \( \bar{u}^{\ell+1} \) onto \( \bar{u}^\ell \). This operator is simply defined by

  \[
  \bar{u}_i^\ell = \frac{1}{|T_i^\ell|} \sum_{j=0}^3 |T_{i,j}^{\ell+1}| \bar{u}_{i,j}^{\ell+1}.
  \]

- A prediction or reconstruction operator \( \Omega_\ell^{\ell+1} \) from the resolution level \( \ell \) to the finer level \( \ell + 1 \). This prediction operator should satisfy the “consistency condition” \( \Psi_\ell^{\ell+1} \Omega_\ell^{\ell+1} = Id \). That is to say

  \[
  \sum_{j=0}^3 |T_{i,j}^{\ell+1}| \bar{u}_i^{\ell+1} = |T_i^\ell| \bar{u}_i^\ell,
  \]

where \( \bar{u}_i^{\ell+1} \) are the values predicted by \( \Omega_\ell^{\ell+1} \). We only consider here reconstruction operators for which the predicted values \( \bar{u}_i^{\ell+1} \) are given by a local linear combination of some immediately coarser values; \( \bar{u}_i^{\ell+1} = \sum_m \mathcal{M}(j) \alpha_m \bar{u}_m^\ell \), where \( \mathcal{M}(j) \) corresponds to a neighborhood (in \( \Omega^0 \)) of \( T_j^\ell \). Related to the prediction operator are the prediction errors or details

  \[
  d_{i,j}^\ell = \bar{u}_{i,j}^{\ell+1} - \bar{u}_{i,j}^\ell \quad \text{for } j = 0,\ldots,3.
  \]

Note that (1) implies a linear dependence for four details associated to the same triangle. This allows to compute only three details per triangle, for example \( d_{i,j}^\ell = \{d_{i,j}^0, d_{i,j}^1, d_{i,j}^2\}_{j=1}^3 \).
Remark. The details can be written $d_{j,k}^l = \langle u, \varphi_{j,k}^l \rangle := \int_{\mathbb{R}^2} u \varphi_{j,k}^l$, where the wavelet $\varphi_{j,k}^l$ is defined by $\varphi_{j,k}^l = \varphi_{j+1,k}^l - \sum_{m \in 2^j \mathbb{Z}^2} \alpha_m \varphi_m^l$, and $\varphi_j^l = \frac{1}{2^j} \chi_{T^l_a}$ are the scaling functions. We will refer to the details as wavelets coefficients.

Denoting $d^l$ the array of all the $d_{j,k}^l$, the two representations
- $\tilde{u}^L$ (all the averages on the finest level)
- $\tilde{u}^0 \cup \{d^l, \ell = 0, \ldots, L-1\}$ (averages on the coarsest level + details)

are equivalent in so far as they provide the same information and use the same memory requirement. Using the local structure of the projection and prediction operators, one can easily implement the multiscale transformation $\mathcal{M} : \tilde{u}^L \mapsto (\tilde{u}^0, d^0, \ldots, d^{L-1})$, and it’s inverse $\mathcal{M}^{-1}$ with optimal complexity.

2.2. Compression

One of the main interest of decomposing $\tilde{u}^L$ into $\mathcal{M} \tilde{u}^L$ is that this new representation is more appropriate for data compression. Let us define

$$\Lambda := \{\lambda(\varepsilon_0, \varepsilon_1, \ldots, \varepsilon_{L-1}) = \{ \lambda = (\ell, j, k) \text{ s.t. } |d_{j,k}^l| \geq \varepsilon_l \}$$

and the associated truncation operator $\mathcal{T}_\Lambda$ that leaves unchanged the component $d_{j,k}$ if $\lambda \in \Lambda$ and replaces it by 0 otherwise. In all the computations we performed, the level depending threshold parameters are $\varepsilon_\ell := 2^{-\ell(s-L)} \varepsilon$ for a fixed $\varepsilon$ (here $d$ is the space dimension). The parameter $\varepsilon$ controls the truncation error resulting from the thresholding [CDK]. Applying $\mathcal{T}_\Lambda$ on the multiscale decomposition of $\tilde{u}^L$ amounts in building a non-linear approximation $\mathcal{M}^{-1} \mathcal{T}_\Lambda \mathcal{M} \tilde{u}^L$ in which the details in the finest scales are concentrated near the isolated singularities. Such compression properties are however related to additional assumptions on the prediction operator: polynomial accuracy and multiscale stability. Consult [CDK, CKMP] and the references therein for a detailed presentation of this problem.

2.3. Smoothness indicators

The prediction operator $\Omega_{k+1}^l$ has polynomial exactness of order $n$ if for all $u \in \mathcal{P}_n$ (polynomials of degree $n$), we have $\bar{u}^l = \tilde{u}^l$, i.e. $d^{l-1} = 0$ which expresses the orthogonality of $\psi_{j,k}^l$ with $\mathcal{P}_n$. Therefore, if $u$ has local smoothness $C^s$, $s > 0$ (say in a neighborhood of $T_{k,j}$), then $|d_{k,j}^l| \leq C 2^{-\ell \min(s,n-1)}$. In order to justify the use of the details as smoothness indicators, we need a converse property: small details should indicate that the encoded function is locally smooth. We also need some stability in the sense that we can control in some prescribed norm the perturbation of $u$ resulting from thresholding the small details, see [CDK].

3. Harten’s algorithm

We now present a generalisation for triangles of the Harten’s MR scheme. For cartesian grids and Finite Differences, see [CD]. The starting point is a classical FV scheme for solving the conservation law $\partial_t u + \text{div} f(u) = 0$ on the finest grid $\Omega^L$. Here $u(x, y, t)$ is a scalar function, but generalisation to systems is straightforward.
The FV scheme computes at time $t_n$ approximate averages $\bar{u}_k^{L,n} \approx \mathcal{A}(T_k^L)u(.,t_n)$ of the solution $u$ by Algorithm 1.

Algorithm 1: A General FV scheme.

Initialization: $\bar{u}_k^{L,0} = \mathcal{A}(T_k^L)u_0$, for $k = 1, \ldots$.

Remarks: for each time step $n$,

Step 1. Reconstruction: Use a reconstruction operator $\mathcal{R}$ to obtain point values from cell averages.

Step 2. Flux evaluation: compute $\hat{\mathcal{D}}_k^{L,n}$, an approximation of

$$
\mathcal{D}_k^{L,n} = \mathcal{A}(T_k^L) \text{div} f(\mathcal{R}).
$$

Step 3. Advance in time:

$$
\bar{u}_k^{L,n+1} = \bar{u}_k^{L,n} - \Delta t \hat{\mathcal{D}}_k^{L,n}, \quad k = 1, \ldots, N_L.
$$

We only discuss here the Step 2 of this algorithm. The flux evaluations are based on the remark that, by the divergence theorem

$$
\mathcal{D}_k^{L,n} = \frac{1}{|T_k^L|} \sum_j |\Gamma_{k,j}^L| \int_{\Gamma_{k,j}^L} f_{k,j}^{L,n} \mathbf{n}_{k,j} d\sigma.
$$

(3)

Here $\Gamma_{k,j}^L$ denotes interfaces between $T_k^L$ and $T_j^L$. Therefore, $\hat{\mathcal{D}}_k^{L,n}$ can be computed by applying (3) to some approximations $f_{k,j}^{L,n}$ of $f_{k,j}^{L,n}$.

3.1. Multilevel computation of the fluxes

We now explain how the multiscale decomposition of the solution is used to speed up the flux evaluation, through a modification in Step 2 of the FV algorithm. To this effect we define $\mathcal{D}_k^{L,n} = \mathcal{A}(T_k^L) \text{div} f(\mathcal{R})$ for $0 \leq \ell \leq L$. These mean values can again be computed as in (3) and approximated by

$$
\hat{\mathcal{D}}_k^{L,n} = \frac{1}{|T_k^L|} \sum_j |\Gamma_{k,j}^L| \int_{\Gamma_{k,j}^L} f_{k,j}^{L,n} \mathbf{n}_{k,j} d\sigma.
$$

where $f_{k,j}^{L,n} = \frac{1}{|\Gamma_{k,j}^L|} \sum_m |\Gamma_{m}^L| f_{m}^{L,n}$. (4)

The MR representation of the solution is used to avoid the evaluation of $\hat{\mathcal{D}}_k^{L,n}$ on the finest grid wherever it is possible. In the regions (let say around the triangle $T_k^L$) where the solution is smooth, $\hat{\mathcal{D}}_k^{L,n}$ is computed by interpolation of the $\hat{\mathcal{D}}_j^{L,n}$s corresponding to some coarser levels $\ell < L$. In the region where the solution is not smooth enough, $\hat{\mathcal{D}}_k^L$ will be computed accurately using a precise (and expensive) numerical flux. Let $\Omega^{\text{ref}}_{\ell+1}$ denote a subset of $\cup_{\ell=0}^\ell \Omega^\ell$ formed by all the triangles $T_k^\ell$ over which the numerical solution is not regular enough. This hybrid grid (built by Algorithm 3) is the key ingredient of the following algorithm which is an MR version of Step 2 of Algorithm 1.
Algorithm 2: Multilevel computation of the fluxes.
Step 2-1. Compute the set $\Omega^{r,n+1}$ (see Algorithm 5).
Step 2-2. Compute the $\tilde{F}^0$'s on the coarsest grid $\Omega^0$ using (4).
Step 2-3. For $\ell = 1 \land L$, compute the approximate $\tilde{F}^\ell$'s by
   \[ \begin{align*}
   \text{If } d^\ell_{k,j} \in \Omega^{r,n+1} \text{ Then} \\
   & \tilde{F}^\ell_{k,j} \text{ is accurately computed using (4) as in Step 2-2.} \\
   \text{Else} \\
   & \tilde{F}^\ell_{k,j} \text{ is approximately computed by interpolation of} \\
   & \text{the values } \tilde{F}^{\ell-1}, \text{ using the prediction operator } \Omega^{\ell-1}.
   \end{align*} \]

The first test in the algorithm takes into account the propagation of information (which is limited by the CFL condition). The second test takes into account the possible appearance of discontinuities.

Algorithm 3: The hybrid grid.
Step 2-4-1. Initialize $\Omega^{r,n+1} = \Omega^0$.
Step 2-4-2. For $\ell = L - 1 \land 0$,
   \[ \begin{align*}
   \text{For } k = 1, \ldots, N_{\ell} \\
   \text{If } |d^\ell_{k,j}| \geq \varepsilon_{\ell} \text{ for one } j \text{ Then} \\
   & \text{add } T^\ell_{k} \text{ to } \Omega^{r,n+1} \text{ if } T^\ell_{k} \text{ shares a vertex with } T^\ell_{k}. \\
   \text{If } |d^\ell_{k}| \geq 8\varepsilon_{\ell} \text{ Then} \\
   & \text{add } T^\ell_{k} \text{ to } \Omega^{r,n+1}.
   \end{align*} \]

3.2. Numerical tests (2D Euler equations of gas dynamics)
In all the computations, we have used the following reconstruction operator (see the figure 1 for the notations) introduced in [CDKP]:
\[ \begin{align*}
   \bar{\sigma}_{0,0} &= \bar{\sigma}_0 \\
   \bar{\sigma}_{0,2} &= \bar{\sigma}_0 + (\bar{\sigma}_1 + \bar{\sigma}_3 - 2\bar{\sigma}_2)/6 \\
   \bar{\sigma}_{0,1} &= \bar{\sigma}_0 + (\bar{\sigma}_1 + \bar{\sigma}_2 - 2\bar{\sigma}_3)/6
\end{align*} \]

**Reflection of a shock.** We first study the reflection of an oblique shock on the lower side of a rectangular domain. The exact solution and the boundary conditions of the simulation are shown figure 2. The coarsest grid is composed of $N_0 = 200$ triangles. The MR parameters are $\varepsilon = 0.01, L = 3$. The CPU time is $T_{MR} = 0.8 T_{FV}$. The CPU performances of Harten’s scheme are disappointing on this example. To illustrate the adaptive computation of the fluxes, we plot on figure 3 what we called ‘hybrid grid’ consisting in all the triangles for which the flux computations are done precisely using the (supposed to be expensive) numerical solver instead of the (supposed to be cheaper) interpolation. The shape of the hybrid grid indicates that memory savings would be very interesting if the time evolution was be performed on an adaptive grid instead of the finest grid.

**Mach 3 wind tunnel with a step.** This model problem has been examined by several authors. The coarsest grid is composed of $N_0 = 126$ triangles. Four levels are used for the computations. The FV solution computed on the finest grid and the MR solution are displayed figure 4 (density at time $t = 4$). The CPU time for the FV run is $T_{FV} = 13944$ seconds. The MR parameters are $\varepsilon = 0.05, L = 3$. The CPU
time is $T_{MR} = 9931 = 0.71 T_{FV}$. Here again the CPU gain is not tremendous, but the discontinuities are clearly localized on the hybrid grid (not displayed) which is a promising property, in view of the adaptive scheme.

3.3. Limitations of Harten’s scheme
Although the flux is computed adaptively, the evolution of the solution at each time step still takes place on the finest grid $\Omega^L$. For these reasons, the computational gain is limited. In the next section, we describe a fully adaptive algorithm for which the memory storage is proportional to the number of wavelet coefficients describing the solution. The first results using adaptive MR schemes were presented in [GMM].

4. A Fully adaptive scheme
The goal of the adaptive algorithm is to save the maximal amount of computational time and memory space while maintaining the order of accuracy achieved by the reference FV scheme on the finest mesh.

4.1. The algorithm
At a given time $t_n$ the solution $u^n$ is now represented on a adaptive grid $\Lambda_n$ made of triangles belonging to different resolution levels. This approach gives way to new difficulties, for instance in the analysis of the stability and precision [CKMP]. Given the numerical solution represented in a compressed form by a set $\Lambda_n$ of significant wavelet coefficients i.e. $u^n = \sum_{\lambda \in \Lambda_n} d^n_\lambda \psi_\lambda$, the adaptive strategy consists in three steps:

1. Refinement. Predict from $\Lambda_n$ a larger set $\tilde{\Lambda}_{n+1}$ such that $\Lambda_n \subset \tilde{\Lambda}_{n+1}$, which is adapted to describe both $u^n$ and $u^{n+1}$ with the required accuracy. The new set of indices $\Lambda_{n+1}$ is defined from $\Lambda_n$ as follows:
   - $\Lambda_n$ is padded by a “security margin” determined by the CFL condition.
   - According to the magnitude of the details $d^n_\lambda$, we possibly add (at position $k$ and scale $\ell$) new wavelets coefficients located at the same space-location but with finer scales $\ell'$ such that $\ell < \ell' \leq \ell^* \leq L$. The precise computation of the local refinement level $\ell^*$ is given in [CKMP].

2. Evolution. Compute from $u^n$ an intermediate solution $\tilde{u}^{n+1}$ defined from the set $\tilde{\Lambda}_{n+1}$; $\tilde{u}^{n+1} = \sum_{\lambda \in \tilde{\Lambda}_{n+1}} d^{n+1}_\lambda \psi_\lambda$. We refer to [CKP] for a complete description of this step.

3. Thresholding. Use a thresholding procedure to obtain the new set $\Lambda^{n+1} \subset \tilde{\Lambda}^{n+1}$ and the new approximation $u^{n+1} = \sum_{\lambda \in \Lambda^{n+1}} d^{n+1}_\lambda \psi_\lambda$.

An important feature of the scheme is that the set $\Lambda_n$ has a tree structure. This structure is crucial since it allows a one to one correspondence between the truncated multiscale decomposition of $\tilde{u}$ and its cell averages on the adaptive triangulation associated to the set $\Lambda_n$ [CKMP].
4.2. Numerical tests

In a previous work [CKP], we have shown the efficiency of the adaptive scheme for scalar 2D problems. The adaptive computations were (up to 10 times) faster than the pure FV ones. We consider here the 1D gas dynamics system.

**Sod’s shock tube.** The coarse discretization of the computing interval consists in 200 subdivisions. The finest one is seven times finer. The standard FV solution on this fine grid is our reference solution. On figure 5, we show the density (t=0.26) computed on the finest grid with the FV scheme (this computation took $T_{FV} = 603$ seconds) along with the MR solution ($L = 4, \varepsilon = 10^{-4}$ and the adaptive grid. For $\varepsilon = 10^{-5}$ (not displayed) the finest level of resolution is used exclusively near the four singularities of the density. This induces some slight oscillations near the contact discontinuity. The $L_1$-error is $E_1 = 0.0297$ and the computing time is $T_{MR} = 86 \simeq T_{FV}/8$ seconds. For $\varepsilon = 10^{-4}$ (figure 5), the finer levels of resolution are used more often, leading to a better accuracy $E_1 = 0.0158$. In particular, the oscillations near the discontinuities have disappeared and the computing time remains interesting $T'_{MR} = 121 \simeq T_{FV}/5.7$.

**Blast waves.** We now test the code to solve the difficult problem of interacting blast waves. The results are shown on figure 6 for 5 levels of resolution. We see that the fine levels are used only where the two waves interact, even though they may have been used at intermediate times. The computing time for several runs are summarized in the following table.

<table>
<thead>
<tr>
<th># levels</th>
<th>MR ($\varepsilon = 0.002$)</th>
<th>FV</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>CPU</td>
<td>memory</td>
</tr>
<tr>
<td>2</td>
<td>1329</td>
<td>207</td>
</tr>
<tr>
<td>3</td>
<td>2096</td>
<td>465</td>
</tr>
<tr>
<td>4</td>
<td>10677</td>
<td>621</td>
</tr>
<tr>
<td>5</td>
<td>38458</td>
<td>717</td>
</tr>
</tbody>
</table>

As far as the memory requirement is concerned, the advantages of the adaptive algorithm are indisputable. The CPU gain however is not so spectacular between the MR computation and the FV run on the corresponding finest level. This poor performance is due to the overhead costs in the MR program. We are using a C++ prototype, which is not optimized at all and in particular uses the standard dynamic memory allocation functions. This is not a sensitive issue when the grid is created once and for all as in the standard FV scheme but becomes very costly for the time adaptive grid book-keeping.

5. Concluding remarks

A FV Multiscale scheme has been applied to solve classical gas dynamics tests. We have shown the effectiveness of the method on triangular meshes but also its limits. Another FV Multiscale scheme is presented. This new scheme takes full advantage of the multiscale representation of the solution: at each time step, the solution is represented in a compressed form by a set of significant wavelets coefficients.
(or equivalently on an adaptive grid). Not only the fluxes but also the whole solution is computed adaptively. As far as the memory requirement is concerned, the advantages of the adaptive algorithm are indisputable. The improvement in terms of CPU is however not as spectacular as in the scalar case treated in [CKP] due to big overhead costs in the MR program. The big difference with the present tests, in terms of programming, is that we now handle *vectorial objects* for which the use of optimized dynamic memory allocation functions is crucial. This technical problem is currently under investigation.

![Figure 1. Subdivision of a triangle $T_0^L$.](image)

![Figure 2. Exact solution (and boundary conditions) of the reflection problem.](image)

References


The hybrid grid

Figure 3. A burst view of a hybrid grid.

FV solution, time $t=4$

MR solution, time $t=4$

Figure 4. FV solution of the step problem and MR Harten’s solution.

Figure 5. Shock tube: density (left y-axis) and adaptive grid (right).

Figure 6. Blast Waves: density (left y-axis) and adaptive grid (right).


Laboratoire d’analyse numérique,
Université Pierre & Marie Curie,
75252 Paris Cedex 05, France
E-mail address: cohen, kaber, postel@ann.jussieu.fr
Finite volume schemes on triangles coupled with multiresolution analysis

Sidi Mahmoud KABER, Marie POSTEL

Laboratoire d’analyse numérique, Université Pierre-et-Marie-Curie, 4, place Jussieu, 75252 Paris cedex 05, France
E-mail: {kaber,postel}@cer.jussieu.fr

(Réçu le 10 février 1999, accepté le 8 mars 1999)

Abstract. A multiresolution procedure is used to reduce the costs of flux evaluations in a finite volume scheme. A two-dimensional hyperbolic conservation law is solved on the finest grid among a hierarchy of nested grids. The mean values of the solution on triangles of a given grid are estimated from the coarser level using an original reconstruction algorithm. The size of the differences between the mean values and their reconstruction is a local regularity criterium and dictates the choice of the flux computation method. Numerical experiments with computing time comparisons are presented. © Académie des Sciences/Elsevier, Paris

Schémas volumes finis sur des triangles couplés avec une analyse multirésolution

Résumé. Nous utilisons une technique de multirésolution pour réduire le coût de calcul des flux d’un schéma volumes finis. Une loi de conservation bidimensionnelle est résolue sur le maillage triangulaire le plus fin dans une hiérarchie de maillages imbriqués. Les valeurs moyennes de la solution sur chaque triangle d’un niveau donné sont estimées à partir du niveau plus grossier par un algorithme de reconstruction original. L’amplitude des différences entre les valeurs moyennes et leur reconstruction sert de critère de régularité pour choisir localement la méthode de calcul des flux. Nous présentons des exemples numériques avec des comparaisons de temps de calcul. © Académie des Sciences/Elsevier, Paris

Version française abrégée

On s’intéresse à la résolution de lois de conservation sur des domaines polygonaux et dans des cas où la solution présente des discontinuités, soit déjà dans la solution initiale dans le cas linéaire, soit induites par la formation de chocs dans le cas non linéaire. Ces discontinuités nécessitent l’emploi de techniques d’ordre élevé pour calculer les flux. On utilise ici l’algorithme ENO (Essentiellement non

Note présentée par Olivier PIRONNEAU.
oscillant) proposé dans [6]. Cet algorithme est très coûteux puisqu’il nécessite pour chaque triangle la sélection d’un polynôme parmi trois polynômes d’interpolation linéaire sur trois couples de triangles voisins. On voudrait limiter l’emploi de cet algorithme aux seules zones où les irrégularités de la solution l’exigent. Pour cela on utilise des techniques de multirésolution déjà testées en dimension un dans [7], en dimension deux et dans le cas de maillages cartésiens dans [3], pour des déformations de maillages rectangulaires dans [5]. La multirésolution pour des maillages bidimensionnels non structurés est décrite dans [2], [1]. Nous avons étudié dans [9] les problèmes de codage et décodage sur des maillages triangulaires. Dans la même référence, nous avons défini la notion de grille hybride constituée de triangles appartenant à des triangulations différentes. Nous prévoyons, dans un futur proche, d’utiliser ces grilles hybrides pour résoudre des équations aux dérivées partielles. Le travail que nous présentons ici est une première étape dans cette direction.

Le domaine $\Omega \subset \mathbb{R}^2$ sur lequel l’équation (1) est posée est doté d’une triangulation initiale grossière $\Omega^L$. On construit une hiérarchie de triangulations imbriquées $\Omega^\ell$ ($0 \leq \ell \leq L$) en divisant chaque triangle en quatre triangles plus petits. Une fonction définie sur $\Omega$ est discrétisée sur chaque niveau de maillage par ses valeurs moyennes. Le passage des moyennes sur une grille $\Omega^{\ell-1}$ aux moyennes sur la grille plus grossière $\Omega^\ell$ est fait par (2). On définit, par (3), un opérateur de reconstruction des valeurs moyennes sur $\Omega^{\ell-1}$ en fonction des valeurs moyennes au niveau immédiatement plus grossier $\Omega^\ell$. Une fonction quelconque peut alors être représentée sur le maillage $\Omega^{\ell-1}$ par les différences entre ses valeurs moyennes et les valeurs prédites à partir des valeurs moyennes au niveau $\Omega^\ell$. Ces différences ou détails sont définies par (4). Cette représentation décrite dans la section 2 peut être utilisée pour compresser l’information en ne conservant à chaque niveau que les détails supérieurs à un seuil de tolérance donné, comme expliqué dans [9]. Ici nous utilisons ces détails pour décider, pour chaque triangle et à chaque niveau, de la technique d’approximation des flux.

La discrétisation choisie pour résoudre l’équation (1) sur la grille fine $\Omega^0$ est de type volumes finis (voir (5)) : on doit donc connaître sur chaque triangle de $\Omega^0$ la moyenne de la divergence du flux. On utilisera pour cela un algorithme proposé dans [1]. On commence par calculer de manière précise les moyennes des divergences sur le maillage $\Omega^L$ en les transformant en intégrales de bord et en utilisant les schémas ENO. On va ensuite progressivement vers le maillage le plus fin : à un niveau donné, sur un triangle où les détails sont petits, la valeur moyenne de la divergence du flux est calculée à partir des valeurs moyennes au niveau immédiatement plus grossier, et ce par la reconstruction (3). Si les détails ne sont pas négligeables, la valeur moyenne de la divergence du flux est calculée de manière précise par intégration de bord et schémas ENO. Une fois les moyennes des divergences calculées sur le niveau le plus fin, on fait avancer la solution du schéma volumes finis, sur ce maillage fin, avec un schéma de Heun. On renvoie à [8] pour plus de précisions.

Dans la dernière section, on présente des tests numériques pour une condition initiale discontinue. Le gain en temps de calcul par rapport à une méthode classique, où tous les flux sont calculés sur le maillage le plus fin, varie entre deux et trois en fonction de la précision recherchée.

1. Introduction

The solution $u(x, y, t)$ of the equation

$$\partial_t u + \nabla \cdot f(u) = 0$$

is sought in $\Omega \subset \mathbb{R}^2$. It verifies the initial condition $u(x, y, 0) = u_0(x, y)$ and possibly boundary conditions as well. The modern numerical schemes when used to solve (1) give an oscillations
Finite volume schemes on triangles and multiresolution free solution. The main disadvantage of these methods is their CPU cost due to the complex flux computations they need. This cost can be reduced using a multiresolution analysis to detect the singularity zones of the numerical solution and computing accurately the fluxes only in these regions. This approach has been originally exploited in the one dimensional case by Harten in several papers. In [7] for instance he computes shock solutions for nonlinear equation with high order numerical fluxes. In two dimensions Bihari and Harten [3] solve scalar non-linear equations on Cartesian grids using tensorial products. A thorough formalization in the more general case of block structured grids can be found in Dahmen et al. [5]. As far as unstructured meshes are concerned, the multiresolution encoding/decoding of a two-dimensional scalar function has already been approached by Abgrall and Harten [2], [1] for general grid hierarchy. We have studied the case of triangular meshes in [9], where we precise the polynomial reconstructions and define the hybrid grid. We plan to use this concept in a future work to solve a PDE on a time varying grid made of cells belonging to different grids according to the local regularity of the solution. We present here an intermediate step in this direction with an emphasis on the difficulties arising from the triangular grid.

Let $\Omega_L$ be a coarse triangulation of the computational domain. We build a hierarchy of nested triangulations $\Omega^\ell$ ($\ell = 0, \ldots, L$), where each grid $\Omega^\ell$ is obtained from $\Omega^{\ell+1}$ by dividing the triangles into four smaller ones using the midpoint rule. The equation (1) is solved on the finest grid $\Omega^0$. At time step $t_n$, the numerical solution $u^n$ is known by its mean values $\bar{u}^n$ on the triangles of $\Omega^0$. This solution can also be represented by its mean values on the triangles of the different levels. To be able to recover the solution on the finest grid, one has to compute the differences between these mean values and values computed by interpolation on the immediately coarser level (see Algorithm 1). The size of these differences, or details, is the criterium to choose the fluxes computation method: on the triangles where the details are small the solution is assumed to be smooth and the flux divergences are computed by interpolation from the coarser level. Conversely, on triangles where the details are larger than a prescribed threshold, fluxes are computed with a simplified second order Essentially non-oscillatory (ENO) scheme. This idea is already available in [1] where a simplified implementation is proposed with only two levels of discretization.

We introduce now some notations. A generic triangle of $\Omega^\ell$ is noted $T_k^\ell$. The boundary of $T_k^\ell$ is $\partial T_k^\ell$, the common edge to triangles $T_k^\ell$ and $T_j^\ell$ is noted $\Gamma_{k,j}$. The outward normal to triangle $T_k^\ell$ on the edge $\Gamma_{k,j}$ is $n_{k,j}$. We also refer to the mean operator $\mathcal{A}$ on triangles; $\bar{u}_k = \mathcal{A}(T_k^\ell)u$ stands for the mean value of the function $u$ on the triangle $T_k^\ell$. Finally, $\bar{u}^\ell$ denotes the array of all the averages $\bar{u}_k^\ell$.

In Section 2, we present the encoding algorithm and the smoothness indicators. These indicators are used in Section 3 to build a numerical scheme. We apply successfully this scheme in Section 4.

2. Encoding

In the case of cell average representation the knowledge of the function on the grid $\Omega^{\ell-1}$ enables its representation on the immediately coarser grid $\Omega^\ell$ by

$$\bar{u}_i^\ell = \frac{1}{|T_i^\ell|} \sum_{j=0}^3 |T_{i,j}^{\ell-1}| \bar{u}_{i,j}^{\ell-1},$$

(2)

where $T_{i,j}^{\ell-1}$ ($j = 0, \ldots, 3$) are the four triangles of $\Omega^{\ell-1}$ composing the triangle $T_i^\ell$ ($T_{i,0}^{\ell-1}$ is the central one). For each child triangle $T_{i,j}^{\ell-1}$ of the triangle $T_i^\ell$, we associate the following stable second
order reconstruction on four triangles (see [4]):

\[
\begin{align*}
\bar{u}_{i,0}^{t-1} &= \bar{u}_{i}^t, \\
\bar{u}_{i,1}^{t-1} &= \bar{u}_{i}^t + (\bar{u}_{2}^t + \bar{u}_{3}^t - 2\bar{u}_{i}^t)/6, \\
\bar{u}_{i,2}^{t-1} &= \bar{u}_{i}^t + (\bar{u}_{1}^t + \bar{u}_{3}^t - 2\bar{u}_{2}^t)/6, \\
\bar{u}_{i,3}^{t-1} &= \bar{u}_{i}^t + (\bar{u}_{1}^t + \bar{u}_{2}^t - 2\bar{u}_{3}^t)/6.
\end{align*}
\]  

(3)

where \(\bar{u}_{j}^t\) for \(j = 1, 2, 3\) denote the averages of \(u\) over the three triangles \(T_{j}^t\) who share an edge with the current triangle \(T_{i}^t\) and numbered so that \(T_{j}^t\) does not share an edge with the corresponding subdivision \(T_{j}^{t-1}\). This reconstruction is one among a family of possible second order reconstructions. It has been selected to ensure the stability of the scheme as shown in [4].

For each triangle \(T_{i}^t\), we define the four differences

\[d_{i,j}^t = \bar{u}_{i,j}^{t-1} - \bar{u}_{i,j}^t \quad \text{for} \quad j = 0, \ldots, 3.\]

(4)

and the following encoding and decoding algorithm

**Algorithm 1.** Encoding/decoding

- \(u\) is known by its cell average values on the finest grid \(\Omega^0\)
- For \(\ell = 1 \rightarrow L\)
  1. **Coarsening:** compute \(\bar{u}^{\ell}\) using (2).
  2. **Reconstruction:** from \(\bar{u}^{\ell}\), compute \(\bar{u}^{\ell-1}\) on \(\Omega^{\ell-1}\) using (3).
  3. **Details:** compute the details \(d_{i}^{\ell}\) using (4).

\[\text{end For } \ell\]

Encoding the information using this algorithm can be used to save storage space if the function is smooth as detailed in [9] or on the other hand to detect where the function is singular. This last feature is the one that we develop here. The size of the details \(d_{i}^{\ell}\) are tested against a level dependent threshold to define local smoothness indicators \(\tilde{\kappa}_{i}^{\ell}\). At each time step we rely on the hyperbolicity of the problem to safely predict these indicators for the next time step for each triangle of each level. This is a straightforward extension of the one-dimensional case and we refer to [8] for details.

3. Numerical scheme

We explain now how we solve equation (1) on the finest grid \(\Omega^0\). Let \(\bar{u}^{n}_{i}\) be an approximation, at time \(t_{n}\), of the average of \(\mathcal{A}(T_{i}^{n})u(\cdot, t_{n})\). At each time step, one computes \(\bar{u}^{n+1}_{i}\) by

\[
\bar{u}^{n+1}_{i} = \bar{u}^{n}_{i} + \delta t \mathcal{D}^{n}_{i}(\mathcal{R}), \quad \text{with} \quad \mathcal{D}^{n}_{i}(\mathcal{R}) \equiv -\frac{1}{|T_{i}^{n}|} \int_{\Gamma_{i}^{n}} \text{div} \mathcal{F}(\mathcal{R}) \, d\Gamma.
\]

(5)

Here \(\mathcal{R} \equiv \mathcal{R}(\cdot; \bar{u})\) is a reconstruction operator applied to cell averages to obtain point values (consult [6] for the design of this operator on triangular meshes). By the divergence theorem \(\mathcal{D}^{n}_{i}(\mathcal{R})\) can be computed by

\[
\mathcal{D}^{n}_{i}(\mathcal{R}) = \sum_{j} \frac{|T_{k,j}^{n}|}{|T_{i}^{n}|} \tilde{f}_{k,j}^{n}, \quad \tilde{f}_{k,j}^{n} = \frac{1}{|\Gamma_{k,j}^{n}|} \int_{\Gamma_{k,j}^{n}} \mathcal{F}(\mathcal{R}(\sigma)) \mathbf{n}_{k,j} \, d\sigma.
\]

(6)

Each integral over \(\Gamma_{k,j}^{n}\) is the sum of integrals \(\tilde{f}_{m,j}^{n}\) over some edges of the finest grid. The computation of the \(\tilde{f}_{m,j}^{n}\)'s is detailed at the end of this section. We wish to avoid this costly step unless it is necessary and achieve this using an original idea of [1].
Algorithm 2. – The tree algorithm
1. Compute the $D^L_i$'s on the coarsest grid $\Omega^L$ using (6).
2. For each level $\ell = L - 1 \leq 0$, compute the $D^\ell_i$'s from the $D^{\ell+1}_i$'s, by:
   \[ \text{if } \hat{h}_\ell = 1, \text{ compute } D^\ell_i \text{ using (6);} \]
   \[ \text{if } \hat{h}_\ell = 0, \text{ the } D^\ell_i \text{'s are computed from the } D^{\ell+1}_i \text{ using (3).} \]

Once all the $D^\ell_i$ are computed either by integration or by reconstruction, equation (5) is advanced in time, on the finest grid.

We now explain the computations of the flux across the edges of $\Omega^0$ and assume in the sequel the equation (1) to be linear $\partial_t u + a \cdot \nabla u = 0$ with convection term $a$. For such equations, we use the approximation $f^+_\ell = F_+(u) + F_-(u)$ with $F_+(u) = [\max(a \cdot u, 0)]u$ and $F_-(u) = [\min(a \cdot u, 0)]u$. $u_k$ and $u_j$ are two values of $u$ chosen from each side of the edge $\Gamma^0_{k,j}$. We tested the reconstructions proposed in [6], namely one first order scheme and two second order ENO schemes based on linear reconstruction. The most compressive one is very costly. It implies a selection among three possible linear polynomials. We verified in [8] the announced numerical order of 1.55.

4. Test

In this section we show results that validate the multiresolution algorithm coupled with the ENO scheme of [6]. We illustrate the numerical efficiency by studying simultaneously the error and the computing time for different values of the tolerance $\varepsilon$. The initial condition is $u_0(x, y) = 1$ if $(x - 0.5)^2 + (y - 0.5)^2 < r^2$ and $u_0(x, y) = 0$ elsewhere with $r = 0.25$. The velocity is $a = (1, 1)$ and periodic boundary conditions are set on the unit square. Three different types of computations are performed using four, five or six levels of resolution. The coarsest mesh is composed of 50 triangles on the sixth level. The CFL condition is unchanged $\Delta t/h = 0.1$ and we translated the disk over one period ($t = 1$). For each discretization a computation without multiresolution is performed on the finest level using the compressive ENO flux evaluation. On Figure 1 the $L_1$ norm of the error is displayed as a function of $h$ for tolerance levels $\varepsilon = 25.10^{-5}$ and $\varepsilon = 5.10^{-4}$. In the case $\varepsilon = 0$ the multiresolution is not used and all the fluxes are computed by integral evaluations on the finer grid. The three discretizations $h = 0.025$, $0.0125$ and $0.00625$ correspond to computations done using four, five and six levels starting from the same coarse grid of 50 triangles. This figure indicates that the multiresolution does not deteriorate the rate of convergence – even though at a given discretization the error increases with the tolerance $\varepsilon$. On Figure 2 we compare simultaneously the CPU time and the precision as a function of $\varepsilon$. The computations are done using six levels of multiresolution. The CPU time is reduced by roughly a factor of two compared with a computation done entirely on the finest grid. From an other point of view, the tolerance level should remain small since the accuracy depends linearly on it and since the gain in CPU time with a high tolerance is not significant compared with what a small tolerance leads to. For a computation on five levels for instance, the CPU time required for a full computation on the finer grid with the compressive ENO scheme is 2149 seconds. If the multiresolution reconstruction is performed with a tolerance $\varepsilon = 25.10^{-5}$ it falls down to 1437 seconds. Eventually, if a centered Lax–Wendroff scheme is used instead of ENO scheme on the fine grid wherever the solution is smooth enough, the CPU time is again reduced to 1007 seconds.

5. Conclusions

This work describes the coupling of multiresolution on triangles with finite volume schemes. The multiresolution analysis is used in order to apply ENO reconstruction only when this costly procedure
S.M. Kaber, M. Postel

is really needed. The numerical simulations show a CPU reduction. Our next goal, currently under investigation, is now to solve the equations, not on the finest grid, but on a hybrid grid composed of triangles from various grids \( \Omega'. \) It will provide a significant speed up as well as important memory savings.

![Figure 1](image1.png) ![Figure 2](image2.png)

Figure 1. - \( L_1 \) error as a function of the discretization \( h \) for different tolerances: \( \varepsilon = 0 \) (+), \( \varepsilon = 9.10^{-5} \) (x) and \( \varepsilon = 5.10^{-5} \) (*).

Figure 2. - CPU time and \( L_1 \) error as a function of the tolerance \( \varepsilon \) for discretization \( h = 0.0005 \).

Acknowledgements. The authors thank A. Cohen, R. Abgrall and S. Müller for profitable discussions.

References

Finite volume schemes on triangles coupled with multiresolution analysis

Sidi Mahmoud KABER, Marie POSTEL

Laboratoire d'analyse numérique, Université Pierre-et-Marie-Curie, 4, place Jussieu, 75252 Paris cedex 05, France
E-mail: { kaber, postel } @ dms.jussieu.fr

(Reçu le 10 février 1999, accepté le 8 mars 1999)

Abstract. A multiresolution procedure is used to reduce the costs of flux evaluations in a finite volume scheme. A two-dimensional hyperbolic conservation law is solved on the finest grid among a hierarchy of nested grids. The mean values of the solution on triangles of a given grid are estimated from the coarser level using an original reconstruction algorithm. The size of the differences between the mean values and their reconstruction is a local regularity criterium and dictates the choice of the flux computation method. Numerical experiments with computing time comparisons are presented. © Académie des Sciences/Elsevier, Paris

Schémas volumes finis sur des triangles couplés avec une analyse multirésolution

Résumé. Nous utilisons une technique de multirésolution pour réduire le coût de calcul des flux d'un schéma volumes finis. Une loi de conservation bidimensionnelle est résolue sur le maillage triangulaire le plus fin dans une hiérarchie de maillages imbriqués. Les valeurs moyennes de la solution sur chaque triangle d'un niveau donné sont estimées à partir du niveau plus grossier par un algorithme de reconstruction originale. L'amplitude des différences entre les valeurs moyennes et leur reconstruction sert de critère de régularité pour choisir localement la méthode de calcul des flux. Nous présentons des exemples numériques avec des comparaisons de temps de calcul. © Académie des Sciences/Elsevier, Paris

Version française abrégée

On s'intéresse à la résolution de lois de conservation sur des domaines polygonaux et dans des cas où la solution présente des discontinuités, soit déjà dans la solution initiale dans le cas linéaire, soit induites par la formation de chocs dans le cas non linéaire. Ces discontinuités nécessitent l'emploi de techniques d'ordre élevé pour calculer les flux. On utilise ici l'algorithme ENO (Essentiellement non...
S.M. Kaber, M. Postel

Cet algorithme est très coûteux puisqu'il nécessite pour chaque triangle la sélection d'un polynôme parmi trois polynômes d'interpolation linéaire sur trois couples de triangles voisins. On voudrait limiter l'emploi de cet algorithme aux seules zones où les irrégularités de la solution l'exigent. Pour cela on utilise des techniques de multirésolution déjà testées en dimension un dans [7], en dimension deux et dans le cas de maillages cartésiens dans [3], pour des déformations de maillages bidimensionnels non structurés est décrite dans [2], [1]. Nous avons étudié dans [9] les problèmes de codage et décodage sur des maillages triangulaires. Dans la même référence, nous avons défini la notion de grille hybride constituée de triangles appartenant à des triangulations différentes. Nous prévoyons, dans un futur proche, d'utiliser ces grilles hybrides pour résoudre des équations aux dérivées partielles. Le travail que nous présentons ici est une première étape dans cette direction.

Le domaine $\Omega \subset \mathbb{R}^2$ sur lequel l'équation (1) est posée est doté d'une triangulation initiale grossière $\Omega^L$. On construit une hiérarchie de triangulations imbriquées $\Omega^\ell$ ($0 \leq \ell \leq L$) en divisant chaque triangle en quatre triangles plus petits. Une fonction définie sur $\Omega$ est discrétisée sur chaque niveau de maillage par ses valeurs moyennes. Le passage des moyennes sur une grille $\Omega^{\ell-1}$ aux moyennes sur la grille plus grossière $\Omega^\ell$ est fait par (2). On définit, par (3), un opérateur de reconstruction des valeurs moyennes sur $\Omega^{\ell-1}$ en fonction des valeurs moyennes au niveau immédiatement plus grossier $\Omega^\ell$. Une fonction quelconque peut alors être représentée sur le maillage $\Omega^{\ell-1}$ par les différences entre ses valeurs moyennes et les valeurs prédites à partir des valeurs moyennes au niveau $\Omega^\ell$. Ces différences ou détails sont définies par (4). Cette représentation décrite dans la section 2 peut être utilisée pour compresser l'information en ne conservant à chaque niveau que les détails supérieurs à un seuil de tolérance donné, comme expliqué dans [9]. Ici nous utilisons ces détails pour décider, pour chaque triangle et à chaque niveau, de la technique d'approximation des flux.

La discrétisation choisie pour résoudre l'équation (1) sur la grille fine $\Omega^F$ est de type volumes finis (voir (5)) : on doit donc connaître sur chaque triangle de $\Omega^F$ la moyenne de la divergence du flux. On utilisera pour cela un algorithme proposé dans [1]. On commence par calculer de manière précise les moyennes des divergences sur le maillage $\Omega^L$ en les transformant en intégrales de bord et en utilisant les schémas ENO. On va ensuite progressivement vers le maillage le plus fin : à un niveau donné, sur un triangle où les détails sont petits, la valeur moyenne de la divergence du flux est calculée à partir des valeurs moyennes au niveau immédiatement plus grossier, et ce par la reconstruction (3). Si les détails ne sont pas négligeables, la valeur moyenne de la divergence du flux est calculée de manière précise par intégration de bord et schémas ENO. Une fois les moyennes des divergences calculées sur le niveau le plus fin, on fait avancer la solution du schéma volumes finis, sur ce maillage fin, avec un schéma de Heun. On renvoie à [8] pour plus de précisions.

Dans la dernière section, on présente des tests numériques pour une condition initiale discontinue. Le gain en temps de calcul par rapport à une méthode classique, où tous les flux sont calculés sur le maillage le plus fin, varie entre deux et trois en fonction de la précision recherchée.

1. Introduction

The solution $u(x, y, t)$ of the equation

$$\partial_t u + \nabla \cdot f(u) = 0$$  \hspace{1cm} (1)

is sought in $\Omega \subset \mathbb{R}^2$. It verifies the initial condition $u(x, y, 0) = u_0(x, y)$ and possibly boundary conditions as well. The modern numerical schemes when used to solve (1) give an oscillations...
Finite volume schemes on triangles and multiresolution

free solution. The main disadvantage of these methods is their CPU cost due to the complex flux computations they need. This cost can be reduced using a multiresolution analysis to detect the singularity zones of the numerical solution and computing accurately the fluxes only in these regions. This approach has been originally exploited in the one dimension case by Harten in several papers. In [7] for instance he computes shock solutions for nonlinear equation with high order numerical fluxes. In two dimensions Bihari and Harten [3] solve scalar non-linear equations on Cartesian grids using tensorial products. A thorough formalization in the more general case of block structured grids can be found in Dahmen et al. [5]. As far as unstructured meshes are concerned, the multiresolution encoding/decoding of a two-dimensional scalar function has already been approached by Abgrall and Harten [2], [1] for general grid hierarchy. We have studied the case of triangular meshes in [9], where we precise the polynomial reconstructions and define the hybrid grid. We plan to use this concept in a future work to solve a PDE on a time varying grid made of cells belonging to different grids according to the local regularity of the solution. We present here an intermediate step in this direction with an emphasis on the difficulties arising from the triangular grid.

Let $\Omega^L$ be a coarse triangulation of the computational domain. We build a hierarchy of nested triangulations $\Omega^l$ ($l = 0, \ldots, L$), where each grid $\Omega^l$ is obtained from $\Omega^{l+1}$ by dividing the triangles into four smaller ones using the midpoint rule. The equation (1) is solved on the finest grid $\Omega^0$. At time step $t_n$, the numerical solution $u^\nu$ is known by its mean values $\bar{u}^{n,\nu}$ on the triangles of $\Omega^0$. This solution can also be represented by its mean values on the triangles of the different levels. To be able to recover the solution on the finest grid, one has to compute the differences between these mean values and values computed by interpolation on the immediately coarser level (see Algorithm 1). The size of these differences, or details, is the criterium to choose the fluxes computation method: on the triangles where the details are small the solution is assumed to be smooth and the flux divergences are computed by interpolation from the coarser level. Conversely, on triangles where the details are larger than a prescribed threshold, fluxes are computed with a simplified second order Essentially non-oscillatory (ENO) scheme. This idea is already available in [1] where a simplified implementation is proposed with only two levels of discretization.

We introduce now some notations. A generic triangle of $\Omega^l$ is noted $T^l_i$. The boundary of $T^l_i$ is $\partial T^l_i$, the common edge to triangles $T^l_i$ and $T^l_j$ is $\Gamma^l_{ij}$. The outward normal to triangle $T^l_i$ on the edge $\Gamma^l_{ij}$ is $n^l_{ij}$. We also refer to the mean operator $\mathcal{A}$ on triangles; $\bar{u}^l_{ij} = \mathcal{A}(T^l_i) u$ stands for the mean value of the function $u$ on the triangle $T^l_i$. Finally, $\bar{u}^l$ denotes the array of all the averages $\bar{u}^l_{ij}$.

In Section 2, we present the encoding algorithm and the smoothness indicators. These indicators are used in Section 3 to build a numerical scheme. We apply successfully this scheme in Section 4.

2. Encoding

In the case of cell average representation the knowledge of the function on the grid $\Omega^{l-1}$ enables its representation on the immediately coarser grid $\Omega^l$ by

$$\bar{u}^l_i = \frac{1}{|T^l_i|} \sum_{j=0}^3 |T^l_{i,j}| \bar{u}^{l-1}_{i,j},$$

(2)

where $T^l_{i,j} (j = 0, \ldots, 3)$ are the four triangles of $\Omega^{l-1}$ composing the triangle $T^l_i$ ($T^l_{i,0}$ is the central one). For each child triangle $T^l_{i,j}$ of the triangle $T^l_i$, we associate the following stable second

819
order reconstruction on four triangles (see [4]):

\[
\begin{align*}
\tilde{u}_{i,0}^{\ell+1} &= \tilde{u}_{i}^{\ell}, \\
\tilde{u}_{i,1}^{\ell+1} &= \tilde{u}_{i}^{\ell} + (\tilde{u}_{i}^{\ell} + \tilde{u}_{i}^{\ell} + 2\tilde{u}_{i}^{\ell})/6, \\
\tilde{u}_{i,2}^{\ell+1} &= \tilde{u}_{i}^{\ell} + (\tilde{u}_{i}^{\ell} + \tilde{u}_{i}^{\ell} + 2\tilde{u}_{i}^{\ell})/6, \\
\tilde{u}_{i,3}^{\ell+1} &= \tilde{u}_{i}^{\ell} + (\tilde{u}_{i}^{\ell} + \tilde{u}_{i}^{\ell} + 2\tilde{u}_{i}^{\ell})/6.
\end{align*}
\] (3)

where \(\tilde{u}_{i}^{\ell} \) for \(j = 1, 2, 3\) denote the averages of \(u\) over the three triangles \(T_{i}^{j}\) who share an edge with the current triangle \(T_{i}^{0}\) and numbered so that \(T_{i}^{j}\) does not share an edge with the corresponding subdivision \(T_{i}^{j-1}\). This reconstruction is one among a family of possible second order reconstructions. It has been selected to ensure the stability of the scheme as shown in [4].

For each triangle \(T_{i}^{j}\), we define the four differences

\[d_{i,j}^{\ell} = \tilde{u}_{i,j}^{\ell+1} - \tilde{u}_{i,j}^{\ell+1} \quad \text{for } j = 0, \ldots, 3.\] (4)

and the following encoding and decoding algorithm

**Algorithm 1. – Encoding/decoding**

- \(u\) is known by its cell average values on the finest grid \(\Omega^{0}\)
- **For** \(\ell = 1 \rightarrow L\)
  1. **Coarsening:** compute \(\tilde{u}^{\ell}\) using (2).
  2. **Reconstruction:** from \(\tilde{u}^{\ell}\), compute \(\tilde{u}^{\ell+1}\) on \(\Omega^{\ell+1}\) using (3).
  3. **Details:** compute the details \(d_{i,j}^{\ell}\) using (4).
- **end For** \(\ell\)

Encoding the information using this algorithm can be used to save storage space if the function is smooth as detailed in [9] or on the other hand to detect where the function is singular. This last feature is the one that we develop here. The size of the details \(d_{i,j}^{\ell}\) are tested against a level dependent threshold to define local smoothness indicators \(\hat{h}_{i}^{\ell}\). At each time step we rely on the hyperbolicity of the problem to safely predict these indicators for the next time step for each triangle of each level. This is a straightforward extension of the one-dimensional case and we refer to [8] for details.

### 3. Numerical scheme

We explain now how we solve equation (1) on the finest grid \(\Omega^{0}\). Let \(\tilde{u}^{n} \) be an approximation, at time \(t_{n}\), of the average of \(A(T_{k}^{0})u(\cdot, t_{n})\). At each time step, one computes \(u_{k}^{n+1}\) by

\[
\tilde{u}_{k}^{n+1} = \tilde{u}_{k}^{n} + b t D_{k}^{n}(\mathcal{R}), \quad \text{with} \quad D_{k}^{n}(\mathcal{R}) \equiv -\frac{1}{|T_{k}^{0}|} \int_{T_{k}^{0}} \text{div} f(\mathcal{R}) \, dx \, dy.
\] (5)

Here \(\mathcal{R} \equiv \mathcal{R}(\cdot; \tilde{u})\) is a reconstruction operator applied to cell averages to obtain point values (consult [6] for the design of this operator on triangular meshes). By the divergence theorem \(D_{k}^{n}(\mathcal{R})\) can be computed by

\[
D_{k}^{n}(\mathcal{R}) = \sum_{j} \frac{|T_{k,j}^{0}|}{|T_{k}^{0}|} \tilde{f}_{k,j}^{n}, \quad \tilde{f}_{k,j}^{n} = \frac{1}{|T_{k,j}^{0}|} \int_{T_{k,j}^{0}} f(\mathcal{R}(\sigma)) n_{k,j} \, d\sigma.
\] (6)

Each integral over \(T_{k,j}^{0}\) is the sum of integrals \(\tilde{f}_{m,j}^{n}\) over some edges of the finest grid. The computation of the \(\tilde{f}_{m,j}^{n}\)'s is detailed at the end of this section. We wish to avoid this costly step unless it is necessary and achieve this using an original idea of [1].

820
Algorithm 2. - The tree algorithm
1. Compute the $D^L_k$'s on the coarsest grid $\Omega^L$ using (6).
2. For each level $\ell = L - 1 \leq 0$, compute the $D^{\ell+1}_k$'s from the $D^\ell_k$'s, by:
   \begin{align*}
   &\text{if } \hat{h}_k = 1, \text{ compute } D^\ell_k \text{ using (6)}; \\
   &\text{if } \hat{h}_k = 0, \text{ the } D^{\ell}_k \text{'s are computed from the } D^{\ell+1}_k \text{'s using (3)}. \end{align*}

Once all the $D^\ell_k$'s are computed either by integration or by reconstruction, equation (5) is advanced in time, on the finest grid.

We now explain the computations of the flux across the edges of $\Omega^0$ and assume in the sequel the equation (1) to be linear $\partial_t u + a \cdot \nabla u = 0$ with convection term $a$. For such equations, we use the approximation $f_{ij}^0 \approx F_+\left(\bar{u}_i\right) + F_-\left(\bar{u}_j\right)$ with $F_+(u) = \max(a \cdot n, 0)|u|$ and $F_-(u) = \min(a \cdot n, 0)|u|$. $\bar{u}_i$ and $\bar{u}_j$ are two values of $u$ chosen from each side of the edge $\Gamma^0_{k,ij}$. We tested the reconstructions proposed in [6], namely one first order scheme and two second order EN0 schemes based on linear reconstruction. The most compressive one is very costly. It implies a selection among three possible linear polynomials. We verified in [8] the announced numerical order of 1.55.

4. Test

In this section we show results that validate the multiresolution algorithm coupled with the EN0 scheme of [6]. We illustrate the numerical efficiency by studying simultaneously the error and the computing time for different values of the tolerance $\varepsilon$. The initial condition is $u_0(x, y) = 1$ if $(x - 0.5)^2 + (y - 0.5)^2 < r^2$ and $u_0(x, y) = 0$ elsewhere with $r = 0.25$. The velocity is $a = (1, 1)$ and periodic boundary conditions are set on the unit square. Three different types of computations are performed using four, five or six levels of resolution. The coarsest mesh is composed of 50 triangles - 51200 on the sixth level. The CFL condition is unchanged $\Delta t/h = 0.1$ and we translated the disk over one period ($t = 1$). For each discretization a computation without multiresolution is performed on the finest level using the compressive EN0 flux evaluation. On Figure 1 the $L_1$ norm of the error is displayed as a function of $h$ for tolerance levels $\varepsilon = 25.10^{-5}$ and $\varepsilon = 5.10^{-4}$.

The three discretizations $h = 0.025, 0.0125$ and $0.00625$ correspond to computations done using four, five and six levels of resolution. The coarsest mesh is composed of 50 triangles - 51200 on the sixth level. The CFL condition is unchanged $\Delta t/h = 0.1$ and we translated the disk over one period ($t = 1$). For each discretization a computation without multiresolution is performed on the finest level using the compressive EN0 flux evaluation. On Figure 1 the $L_1$ norm of the error is displayed as a function of $h$ for tolerance levels $\varepsilon = 25.10^{-5}$ and $\varepsilon = 5.10^{-4}$. In the case $\varepsilon = 0$ the multiresolution is not used and all the fluxes are computed by integral evaluations on the finer grid. The three discretizations $h = 0.025, 0.0125$ and $0.00625$ correspond to computations done using four, five and six levels starting from the same coarse grid of fifty triangles. This figure indicates that the multiresolution does not deteriorate the rate of convergence – even though at a given discretization the error increases with the tolerance $\varepsilon$. On Figure 2 we compare simultaneously the CPU time and the precision as a function of $\varepsilon$. The computations are done using six levels of multiresolution. The CPU time is reduced by roughly a factor of two compared with a computation done entirely on the finest grid. From an other point of view, the tolerance level should remain small since the accuracy depends linearly on it and since the gain in CPU time with a high tolerance is not significant compared with what a small tolerance leads to. For a computation on five levels for instance, the CPU time required for a full computation on the finer grid with the compressive EN0 scheme is 2149 seconds. If the multiresolution reconstruction is performed with a tolerance $\varepsilon = 25.10^{-5}$ it falls down to 1437 seconds. Eventually, if a centered Lax–Wendroff scheme is used instead of EN0 scheme on the fine grid wherever the solution is smooth enough, the CPU time is again reduced to 1007 seconds.

5. Conclusions

This work describes the coupling of multiresolution on triangles with finite volume schemes. The multiresolution analysis is used in order to apply EN0 reconstruction only when this costly procedure
S.M. Kaber, M. Pastel

is really needed. The numerical simulations show a CPU reduction. Our next goal, currently under investigation, is now to solve the equations, not on the finest grid, but on a hybrid grid composed of triangles from various grids $\Omega'$. It will provide a significant speed up as well as important memory savings.

![Figure 1](image1.png)  
Figure 1. - $L_1$ error as a function of the discretization $h$ for different tolerances: $\varepsilon = 0$ (+), $\varepsilon = 25 \times 10^{-5}$ (x) and $\varepsilon = 5 \times 10^{-1}$ (*).

![Figure 2](image2.png)  
Figure 2. - CPU time and $L_1$ error as a function of the tolerance $\varepsilon$. Discretization $h = 0.00025$.

Acknowledgements. The authors thank A. Cohen, R. Abgrall and S. Müller for profitable discussions.

References


MULTIRESOLUTION FINITE VOLUME SCHEME FOR A MIXED HYPERBOLIC-ELLIPTIC SYSTEM. *

OLIVER GROSCHANS 1 AND MARIE POSTEL 2

Abstract. A mixed hyperbolic-elliptic system arises in the context of liquid-vapor phase transition modelling. It has been studied numerically with an explicit finite volume method based on a relaxation scheme which must resolve very precisely the very steep phase transition region. This requirement leads to prohibitive cost when a uniform discretization is used. Multiresolution is a good candidate to improve the performances of such schemes and we examine the possibility of implementing the algorithms developed originally for strictly hyperbolic systems.

AMS Subject Classification. 65-10, 76T05, 76M12.

1. Introduction

This work is part of a general project of modeling the thermovector fluid in the reactor of a nuclear plant. In the first phase of the research, a set of equations was established, taking into account the thermodynamical effects near the critical point, the capillarity phenomena and the fluid reaction on the wall of the reactor. Some preliminary numerical studies were performed whose conclusions were that these so called “second gradient” equations are very hard to solve [8].

The first difficulty is the modeling of the phase transition by a continuous variation of the density - whose shape is important for the engineer although it occurs on a very fine scale. The presence of second and third derivatives in the density requires a special numerical treatment, as well as the instability arising from the particular thermodynamical closure law. This theme of research was first proposed by CEA Grenoble at Cemracs 98.

Our interest in this problem was really aroused by the first difficulty which can be viewed as the challenge of describing very accurately a phenomenon arising in a very limited time dependent area of a relatively large domain. Setting aside the other inherent technical difficulties, this is the typical setting for applying a multiscale technique such as the one initially introduced by Harten [7] for hyperbolic equations and lately developed by Cohen [2],[3], Dahmen [5], Donat [1].

Keywords and phrases: Multiresolution, finite volumes

* The authors thank Albert Cohen and Frederic Coquel (LAN) for their help on this project during the Cemracs99, and Olivier Lebahge (CEA, Grenoble) for his sponsorship

1 SFB 329, IWR, INF 368, Universität Heidelberg, D-69120 Heidelberg, Germany
e-mail: O.Groshans@iwr.uni-heidelberg.de

2 Laboratoire d'Analyse Numérique, Université Pierre et Marie Curie, Boîte courrier 187, 75252 Paris Cedex 05, France
e-mail: postel@ann.jussieu.fr

© EDP Sciences, SMAI 2001
The techniques rely on the basic observation that systems of conservation laws generally require very complicated and costly schemes, because they are nonlinear and may therefore develop singularities even starting from smooth initial data. On the other hand, these costly schemes, like for instance ENO type flux reconstructions, are really necessary only in the vicinity of the discontinuities and could be advantageously replaced by simpler high order centered schemes in the smooth regions. This requires an efficient way to detect the singularities which brings immediately in mind wavelet decomposition techniques. The original multiresolution algorithm proposed by Harten relies on wavelet theory but is really adapted to the particular case of solutions of finite volumes schemes, that is functions known by their mean values on a grid. In broad lines, it consists in building a hierarchy of nested grids. The solution given on a uniform grid can be described by its mean values on the immediately coarser grid, plus the differences between the fine representation and its approximation by interpolation from the coarse one. Repeating along the succession of levels provides the encoded representation of the solution. The size of the differences provides smoothness indicators and the underlying interpolation operator can be used — wherever these differences are negligible — to compute the evolution of the solution on the coarse grid only.

In this paper, we describe more precisely the multiresolution method adapted to the specific finite volume scheme designed to solve the second gradient equations. Since the hyperbolicity assumption used in the original design of the method is not valid here, numerical experiments have been performed to study the stability and robustness of the method. So far the original Harten scheme has been implemented and the solution is computed on the finest level everywhere at each time step. This prototype is meant to be developed into a more complicated algorithm were the solution will be computed on a completely adaptive grid. For sake of simplicity we have only consider the 1D, isothermal case. Simultaneously, in this second stage taking place during Cemacs’99, several projects aimed towards extending the first years results [4] to the case of the non-isothermal form of the equations, 1D, 2D, and implicit schemes, and are being discussed in [6].

2. The Second Gradient Equations

The initial system of equations

$$\begin{align*}
\partial_t \rho + \partial_x (\rho u) &= 0 \\
\partial_t (\rho u) + \partial_x (\rho u^2 + P(\rho)) &= \lambda \partial_x (\rho^2) \\
\partial_t (\rho N) + \partial_x (\rho N u) &= \sqrt{\lambda} \partial_x u \\
\partial_t (\rho N) + \partial_x (\rho N^2 + P(\rho) + \frac{3}{2} N^2 \rho^4) &= -\sqrt{\lambda} \partial_x (\rho^2 N)
\end{align*}$$

(1)

where $P$ is for instance the Van der Waals pressure law, has been studied in D. Jamet’s thesis [8]. During Cemacs’98 F. Coquel and O. Groshans transformed this system by introducing an auxiliary unknown $N$ into

$$\begin{align*}
\partial_t \rho + \partial_x (\rho u) &= 0 \\
\partial_t (\rho N) + \partial_x (\rho N u) &= \sqrt{\lambda} \partial_x u \\
\partial_t (\rho N) + \partial_x (\rho N^2 + P(\rho) + \frac{3}{2} N^2 \rho^4) &= -\sqrt{\lambda} \partial_x (\rho^2 N)
\end{align*}$$

and studied in detail its numerical behaviour. For simplicity of presentation let us consider the following equations

$$\begin{align*}
\partial_t \rho + \partial_x (\rho u) &= 0 \\
\partial_t (\rho u) + \partial_x (\rho u^2 + P(\rho)) &= 0 \\
\partial_t (\rho N) + \partial_x (\rho N u) &= \sqrt{\lambda} \partial_x u \\
\partial_t (\rho N) + \partial_x (\rho N^2 + P(\rho) + \frac{3}{2} N^2 \rho^4) &= -\sqrt{\lambda} \partial_x (\rho^2 N)
\end{align*}$$

(2)

with $p(\rho) = P(\rho) + \frac{3}{2} \lambda (\partial_x \rho)^2 - \lambda \rho (\partial_x \rho)$ which are equivalent to (1). Now all the difficulties are hidden in this extremely complex pressure $p(\rho)$. For more details please refer to [4, 6].
3. Numerical Scheme — A Relaxation Method

To solve the system (2) we apply a relaxation method and relax on the pressure \( p(\rho) \). This leads to

\begin{align*}
\partial_t \rho + \partial_x (\rho u) &= 0 \\
\partial_t (\rho u) + \partial_x (\rho u^2 + \Pi) &= 0 \\
\partial_t (\rho \Pi) + \partial_x (\rho u \Pi + a^2 u) &= \frac{\rho}{\varepsilon} (P(\rho, N) - \Pi)
\end{align*}

(3)

where \( a = \rho^2 \frac{\partial p}{\partial \rho} \). For \( \varepsilon \to 0 \) we get the original system (2) back and \( \varepsilon \to \infty \) gives us the relaxed system. The algorithm is now as follows: First we solve the \( \varepsilon \to \infty \) system, that is we obtain values for \( \rho, \rho u, \rho \Pi \) at \( t_{n+1} \), and then we let \( \varepsilon \to 0 \) which essentially means computing \( \Pi^{n+1} \) as \( \rho(t_{n+1}) \) because \( \rho, \rho u \) at \( t_{n+1} \) are equal to \( \rho, \rho u \) at \( t_{n+1} \).

We use an explicit finite volume scheme to solve the equations (3) in the case \( \varepsilon \to \infty \). Since \( \Pi \) can be viewed as a dependent variable in this algorithm, we denote by \( U \) the vector of conservative quantities \( (\rho, \rho u)^T \) and by \( F \) the flux function

\[ F(x, t) = (\rho u, \rho u^2 + \Pi)^T \]

The computational domain is subdivided into \( N \) intervals \( I_k = [x_{k-1/2}, x_{k+1/2}] \) of length \( h \). The numerical solution is an approximation of the mean values \( \bar{U}^n_k \approx U^n_k = \mathcal{A}(I_k)U(\cdot, t_n) \), where \( \mathcal{A}(I)u \) denotes the mean operator on an interval \( I \)

\[ \mathcal{A}(I)u = \frac{1}{|I|} \int_I u(x)dx. \]

The numerical solution is computed in its conservative form as

\[ \bar{U}^{n+1}_k = \bar{U}^n_k - \frac{\Delta t}{h} \left( \bar{f}^n_{k+1/2} - \bar{f}^n_{k-1/2} \right) \]

where \( \bar{f}^n_{k-1/2} \) is the numerical flux at point \( x_{k-1/2} \), i.e. an approximation of the flux across the intervals boundary

\[ \bar{f}^n_i \approx \bar{f}^n_i \approx \frac{1}{\Delta t} \int_{n\Delta t}^{(n+1)\Delta t} F(i\Delta x, t)dt. \]

There are two difficulties in the algorithm just above. The flux evaluation is one of them: the evolution of the two independent quantities \( \rho \) and \( u \) using system (3) requires to solve at each time step the local Riemann problem for the homogeneous system obtained when letting \( \varepsilon \) go to \( \infty \) in (3). Furthermore, computing the initial condition for the next time step amounts to relax the system (3) with \( \varepsilon \to 0 \). In other words, knowing \( \Pi^{n+1} \) and \( \Pi^{n+1} \), compute \( \Pi^{n+1} \) we need the third derivative of \( \rho^{n+1} \). Therefore the evaluation of \( \Pi \) at one point will require the values of \( \bar{\rho} \) and \( \bar{u} \) at four neighboring points. We refer to [6] for the details of derivation and summarize the computation of the numerical fluxes by

\[ \bar{f}^{n+1}_{i+1/2} = \bar{F} \left( \bar{U}^n_{i+1}, \bar{U}^n_{i+1}, \bar{U}^n_{i+1}, \bar{U}^n_{i+2} \right) \]

(4)

It is therefore crucial to limit the number of those flux evaluations and the multiresolution can play an important role here. The domain is subdivided in several levels of resolution nested within each other. Each interval is the union of two intervals from the finer level. The method consists in solving the problem everywhere on the finest grid while taking advantage of smoothness wherever possible to reduce flux evaluations (4). At each time step, details provided by the multiresolution analysis of the solution are used to choose the flux computation.
method. In the areas where the solution is smooth enough, fluxes at the edges of the interval are interpolated from the fluxes at the edges of intervals on the coarser level. Correspondingly, in areas of strong variations, fluxes are computed precisely using (4) with the finite volume solution on the finest grid as input values.

In the following section we recall Harten’s algorithms as developed in [7] and references within. We first describe the encoding / decoding algorithms for a function represented by its mean values or by point values. Then we show how they are used in the multiresolution coupled with finite volumes scheme.

4. Finite volume scheme and multiresolution. Harten’s algorithm

In this section we introduce notations and recall the original form of Harten’s algorithm found in [7]. We refer the reader to this article for details and justifications. The computational domain $\Omega$ is divided at each level $\ell$ of resolution in $N_{\ell}$ intervals $I^\ell_k = [x^\ell_{k-1/2}, x^\ell_{k+1/2}]$ of length $h^\ell$. The levels are nested within each other starting from the coarse grid $\Omega^0$ until the finest grid $\Omega^L$. Intervals on one level $\ell$ all have the same length $h^\ell$ and are divided by two to form the immediately finer level $\ell + 1$: $I^\ell_k = I^\ell_{2k+1} \cup I^\ell_{2k+1}$. We therefore have $x^\ell_{k+1/2} = x^\ell_{k-1/2}$ and $x^{\ell+1}_{2k+1} = (x^\ell_{k-1/2} + x^\ell_{k+1/2})/2$. We note $\bar{U}^\ell_k = \mathcal{A}(I^\ell_k)$ the mean values of the vector function $U$ on each interval $k$ of a level $\ell$. Harten describes two methods to encode a function with a multiresolution analysis depending on whether the function is known through its mean values on the intervals (algorithm 1) or through its point values at the intervals bounds. In fact the finite volume scheme will use both representations: the mean value one for the solution $\bar{U}^{\ell,n}_k$ itself, and the point value one for the fluxes $\bar{f}^{\ell,n}_k$ since these last quantities may actually be considered as primitives of $F(x)U$ evaluated at the discretization points $x^\ell_{k+1/2}$.

**Algorithm 1. Encoding (mean values)**

$U$ is known through its approximated mean values $\bar{U}^\ell$ on finest grid $\Omega^\ell$ for $\ell = L - 1 \searrow 0$

1. Compute $\bar{U}^{t,\ell}_k = (\bar{U}^{t+1,\ell}_{2k} + \bar{U}^{t+1,\ell}_{2k+1})/2$ for $k = 1, \ldots, N_{\ell}$
2. Compute $\bar{U}^{t+1,\ell}_k$ on $\Omega^{t+1,\ell}$: $\bar{U}^{t+1,\ell}_{2k} = \mathcal{I}(I^{t+1,\ell}_{2k}; \bar{U}^{t,\ell}_k)$ for $k = 1, \ldots, N_{\ell}$
3. Compute details between $\Omega^{t+1,\ell}$ and $\Omega^{t,\ell}$: $d^\ell_k = \bar{U}^{t+1,\ell}_{2k} - \bar{U}^{t,\ell}_k$ for $k = 1, \ldots, N_{\ell}$

**End of for $\ell$**

We note $\bar{U}_{MR} = (\bar{U}^0, d^0, \ldots, d^{L-1})$ the multiresolution representation of function $\bar{U}$, namely its mean values on the coarse grid plus the details allowing to reconstruct it on the finest grid using the following algorithm:

**Algorithm 2. Decoding (mean values)**

$U$ is known by its mean values $\bar{U}^0$ on the coarsest grid $\Omega^0$ and all the details $d^\ell$ for $\ell = 0, \ldots, L - 1$ for $\ell = 0 \searrow L - 1$

1. Interpolation: compute $\bar{U}^{t+1,\ell}_k$ $\bar{U}^{t+1,\ell}_{2k} = \mathcal{I}(I^{t+1,\ell}_{2k}; \bar{U}^{t,\ell}_k)$ for $k = 1, \ldots, N_{\ell}$
2. Reconstruction: compute $\bar{U}^{t+1,\ell}_k$ on $\Omega^{t+1,\ell}$: $\bar{U}^{t+1,\ell}_{2k} = d^\ell_k + \bar{U}^{t+1,\ell}_{2k}$ and $\bar{U}^{t+1,\ell}_{2k+1} = 2\bar{U}^{t+1,\ell}_k - \bar{U}^{t+1,\ell}_{2k}$

**End of for $\ell$**

In the two previous algorithms, the linear interpolation operator $\mathcal{I}$ is obtained by imposing polynomial exactness of degree $s$. In the numerical simulations, we will set $s = 1$, which means that to go from a level $l$ to the level $l + 1$ we use a three cell averages interpolation, and as explicit formulas we have
\[
\tilde{U}_{2k}^{\ell+1} = \frac{1}{8}(\tilde{U}_{k-1}^{\ell} + 8\tilde{U}_k^{\ell} - \tilde{U}_{k+1}^{\ell}) \quad \text{inside the domain}
\]

\[
\tilde{U}_0^{\ell+1} = \frac{1}{8}(11C_0^{\ell} - 4C_1^{\ell} + C_2^{\ell}) \quad \text{on the left boundary}
\]

\[
\tilde{U}_{n+1}^{\ell+1} = \frac{1}{8}(-\tilde{U}^{\ell}_{n-2} + 4\tilde{U}^{\ell}_{n-1} + 5\tilde{U}^{\ell}_n) \quad \text{on the right boundary}
\]

We denote by \(M\) the encoding operator such that \(U_M = (C^0, d^1, ..., d^{L-1}) = M \tilde{U}^L\). If \(\Lambda = \{(k, \ell)\}\) is a set of double indices such that \(\cup_{k \in \Lambda} \Omega_k \subset \bigcup_{k \in \Omega} \tilde{\Omega}^{\ell}\), we denote by \(T_\Lambda\) the thresholding operator on an encoded function such that \(T_\Lambda(C^0, d^1, ..., d^{L-1}) = (C_0^\ell, d_1^\ell, ..., d_{L-1}^\ell)\) with \(d_\lambda = 0\) if \(\lambda \notin \Lambda\) and we introduce a level dependent thresholding based on a tolerance \(\varepsilon\) which defines a set of indices \(\Lambda_\varepsilon = \{\lambda = (k, \ell), \|p^\ell_\lambda\| > 2\delta\varepsilon\}\). It is easy to recover the basic result \(\|\tilde{U}^L - \tilde{U}\|_C < \varepsilon\) where \(T_\varepsilon = M^{-1}T_\Lambda M\).

Similar algorithms exist for a function \(f\) known this time through its values \(f_{k-1/2}^{\ell}\) at points \(x_{k-1/2}\). The finite volume mean values being primitives of the flux function at interval end points, the corresponding order of interpolation for the fluxes on a level \(\ell + 1\) is a third degree polynomial depending on the values of the fluxes at four points on the level \(\ell\) as shown below:

![Table 1. flux interpolation](image)

The explicit formulas are given as

\[
\begin{align*}
\hat{f}_{2k+1/2}^{\ell+1} &= \frac{1}{16}(-\hat{f}_{k-3/2}^{\ell} + 9\hat{f}_{k-1/2}^{\ell} + 9\hat{f}_{k+1/2}^{\ell} - \hat{f}_{k+3/2}^{\ell}), \quad \text{inside the domain} \\
\hat{f}_{2k+1/2}^{\ell+1} &= \frac{1}{16}(5\hat{f}_{k-3/2}^{\ell} + 15\hat{f}_{k-1/2}^{\ell} - 5\hat{f}_{k+1/2}^{\ell} + \hat{f}_{k+3/2}^{\ell}), \quad \text{on the left boundary} \\
\hat{f}_{2n+1/2}^{\ell+1} &= \frac{1}{16}(5\hat{f}_{n-3/2}^{\ell} - 5\hat{f}_{n-1/2}^{\ell} + 15\hat{f}_{n+1/2}^{\ell} + 5\hat{f}_{n+3/2}^{\ell}), \quad \text{on the right boundary}.
\end{align*}
\]

Similarly to the thresholding described above for the mean value encoding, we can define a thresholding for a point value function which will amount to use the interpolated values (5) whenever they are close enough to the true values. As it is explained and justified in details in [2], the multiresolution scheme relies on the construction of a set \(\Lambda_\varepsilon^{n+1}\) which contains both \(\Lambda^n\) and \(\Lambda_\varepsilon^{n+1}\). The solution at time \(t^{n+1}\) on \(\Lambda_\varepsilon^{n+1}\) is therefore close enough to the thresholded solution on \(\Lambda_\varepsilon^{n+1}\). The construction of this set \(\Lambda_\varepsilon^{n+1}\) will be further detailed and justified. Since the thresholded solutions at both time \(t^n\) and \(t^{n+1}\) are within the prescribed tolerance \(\varepsilon\) it is straightforward to obtain that the point values fluxes can be thresholded on the same set of indices. In the case where the initial finite volume scheme is \(L^1\) contractive, it is shown that the error between the multiresolution solution and the standard finite volume one is in \(C(t^n)\varepsilon\). A natural choice for the parameter \(\varepsilon\) is a value which makes this last estimate of the same order as the intrinsic error estimate of the finite volume scheme (typically in \[2^{-L^2/2}\]).

The following algorithm summarizes the full multiresolution scheme:

**Algorithm 3.** Finite Volumes + Multiresolution
Initialization $\tilde{u}_k^{L,0} = A(I_k^L)u_0 \quad \forall I_k^L \in \Omega^L$
Coding $\tilde{u}_k^{L,0} \rightarrow \tilde{u}_k^{3n}_{MR} = (\tilde{u}_0^0, d^0, \ldots, d^L-1)$. (see algorithm 1)

Loop on time step $n$
- Determination of $\tilde{\chi}_n^{n+1}$ and thresholding $u_{MR}^n \rightarrow u_{MRc}^n$ (see algorithm 4)
- Decoding $u_{MR}^n \rightarrow u_{MR}^L$ (see algorithm 2)
- Flux computation for $I_k^0 \in \Omega^0$ on coarse grid, using fine data
  \[ f_{k-1/2}^0 = F(\tilde{a}_{2k}^L, \tilde{u}_{2k}^L) \quad (4) \]
- Compute solution at time $n+1$ on coarse grid $\Omega^0$
  \[ u_{k+1,0}^n = \tilde{u}_k^{n,0} - \frac{\Delta t}{\Delta x} (f_{k+1/2}^n - f_{k-1/2}^n) \]
- Loop on intervals $I_k^0 \in \Omega^\ell$
  \[ f_{k-1/2}^\ell = f_{k+1/2}^\ell \quad (5) \]
  \[ f_{k+1/2}^\ell = F(x_{k+3/2}, \tilde{u}_{k+1}^\ell) \]
- else precise computation, using fine data
  \[ f_{k+1/2}^\ell = F(\tilde{a}_{2k+2}^0, \tilde{a}_{2k+1}^0) \quad (4) \]
  \[ d_{k+1,n+1}^\ell = d_{k,n+1}^\ell - \frac{\Delta t_{x,\ell}}{\Delta x_{x,\ell}} \left(f_{k+1/2}^\ell - f_{k+1/2}^\ell - f_{k+1/2}^\ell\right) \]
- End if
- End of loop on intervals.
- End of loop on levels.

End of loop on time step.

We now recall the intermediate algorithm of thresholding and determination of the smoothness indicators, that is the set of indices $\tilde{\Lambda}_n^{n+1}$. This is actually a tricky part in the whole algorithm. It relies on the size of the details at the given time and in the initial Harten’s setting also on the fact that the equations are hyperbolic. This means that if the CFL condition is fulfilled spatial gradients will not propagate faster than one grid point per time step. It also allows to estimate the increase of the details in one time step meaning that not more than one level of refinement should be necessary from one time step to the next. The hyperbolicity assumption is not verified in the case of the second gradient equations and the robustness of the following strategy for the thresholding has been verified numerically:

Algorithm 4. thresholding and smoothness indicators $\tilde{\Lambda}_n^\ell$

Initialization $\tilde{\Lambda}_0^{n+1} = \bigcup_{\ell=0}^L \Omega^\ell$

Loop on levels $\ell = L \setminus 1$
- Tolerance $\varepsilon_{\ell} = 2^\ell \varepsilon$
- Loop on $I_k^\ell \in \Omega^\ell$
  - if $|I_k^\ell| < \varepsilon_{\ell}$ then $d_{k}^\ell = 0$ and $I_k^\ell \notin \tilde{\Lambda}_n^{n+1}$
  - else
    - $I_{k+m}^\ell \in \tilde{\Lambda}_n^{n+1}$ for $|m| < 2$
    - if $|I_k^\ell| > 2^\ell \varepsilon_{\ell}$ and if $\ell < L$ then $I_{k+1}^\ell \cup I_{k+1}^\ell \in \tilde{\Lambda}_n^{n+1}$
  - End if
- End of loop on intervals.
End of loop on levels.

This algorithm defines a thresholding operator $T_l$ which acts on the multiresolution representations of $\tilde{U}_M$. The thresholding parameter $\epsilon$ is level dependent and for the moment we choose $\epsilon_l = 2^{l-1}\epsilon$ which means we are keeping less details if go to a coarser level. To have a kind of measure for the efficiency of the thresholding we define

$$\text{rate of compression} = \frac{n_L}{\#(\tilde{A}_l) + 2^{-L}n_L} \tag{6}$$

This quantity depends on time since the set of “non smooth indices” $\tilde{D}_l$ is computed anew at each time step.

5. Numerical experiments

We study the behaviour of our algorithm on the time evolution of an equilibrium solution with reflecting boundary conditions.

For this we compute an equilibrium solution on the finest level (Fig. 1). The computations are done on

![Equilibrium solution](image1.png)

**Figure 1.** Equilibrium solution

a domain of $2.5 \cdot 10^{-5}m$ and 1280 cells on the finest level. We compute a time evolution of the equilibrium solution on the finest level without using any multiresolution. It is this reference solution to which we compare the solutions of the multiresolution algorithm. A time step of $7.0 \cdot 10^{-8}$ is used and comparisons are made after $1 \cdot 10^3$ and $1 \cdot 10^6$ time steps.

For zero threshold, that is all coefficients are kept, we cannot see any difference between the multiresolution and the reference solution in the density variable in (Fig. 2). The velocity profile already indicates, that this variable behaves differently than the density. After $10^4$ time steps everything seems to be fine, but after $10^6$ time steps it is obvious, that the multiresolution algorithm introduces some additional roundoff error (the multiresolution solution is the upper curve in the graph), even without doing any thresholding. For a threshold
of $10^{-7}$ the density profile looks still fine (Fig. 3), both solutions, the reference and the multiresolution, are printed on top of each other, so it is not possible to distinguish between both. The situation is different for the velocity variable (Fig. 3). Already after $10^{-4}$ time steps the velocity variable shows a considerable deviation from the reference solution. But interestingly, this difference is not amplified, when we evolve in time. Even after $10^6$ time steps, the difference is of the same order and also bounded in space to the region of the interface.

This different behaviour of the density and the velocity variable can also be observed for higher values of the threshold. The following two figures (Fig. 4) show the solution for a threshold of $10^{-3}$ and again, the difference in the velocity variable is large, but does not get amplified. Again, the same can be seen for an even larger threshold of $10^{-3}$ (Fig. 5).

To obtain a quantitative view of the error between the reference and the multiresolution solution, we have used different ways to calculate the error of the density and the velocity variable. For physical applications the
surface tension of the interface is a very important value which should not change if we use a multiresolution algorithm. The surface tension is given by $f(|\nabla \rho|^2)$ and therefore, we use $f(\nabla \rho - \nabla \rho_m)^2$ to measure the error in the gradient of the density between the reference solution $\rho_r$ and the multiresolution solution $\rho_m$. The results are shown in (Fig. 6). The different points in the graph correspond to different number of time steps. Even with a high threshold, the error is negligible. This figure also shows, that even with a relatively small threshold we can obtain a high compression rate which also means a low number of exact flux evaluations!

Since the error of the velocity variable seems to be bounded, we look for the behaviour of the maximum of the difference between the reference and the multiresolution solution, that is $\max(v_r - v_m)$. In (Fig. 6) a similar behaviour of the error of the velocity in this measure compared to the error of the density variable can be observed.
6. Conclusions

The multiresolution algorithm has been successfully implemented in the case of second gradient equations. Its robustness and performances in terms of flux computation savings have been numerically tested and the results are very encouraging. We are now studying an improvement of this scheme consisting in computing the solution directly on an hybrid grid — fine where the solution varies a lot and coarser in smooth areas. This fully adaptive scheme has been extensively studied both from the theoretical and numerical sides in the case of hyperbolic equations in [3] and its implementation in the case of second gradient equations is under progress.

References


Figure 6. Error of the velocity solution depending on different thresholds.
Multiresolution Schemes on Triangles for Scalar Conservation Laws

Albert Cohen, Nira Dyn, Sidi Mahmoud Kaber, and Marie Postel

Laboratoire d’Analyse Numérique, Université Pierre et Marie Curie, 75252 Paris Cedex 05, France;
School of Mathematics, University of Tel Aviv, Israel

E-mail: cohen@ann.jussieu.fr, niradyn@math.tau.ac.il, kaber@ann.jussieu.fr, postel@ann.jussieu.fr

Received June 18, 1999; revised November 16, 1999

This paper proposes a multiresolution procedure adapted to triangular cell-averages to improve the performance of finite volume schemes by reducing flux evaluation cost, using the approach introduced by A. Harten. A specific coarse-to-fine prediction scheme is proposed that ensures the stability of the computations, even when a large number of scales are involved. Numerical tests are presented that illustrate the computational gain as well as the order of accuracy of the scheme.

Key Words: multiresolution; triangular mesh; conservation laws.

1. INTRODUCTION

Multiscale methods are a powerful tool in mathematical analysis and applications such as signal processing and numerical simulation. The theoretical background underlying these methods has been substantially reinforced since the emergence of wavelet theory in the 1980s.

One of the particular interests of multiscale discretizations into wavelet bases is that, by a simple thresholding of its coefficients in such a basis, a function is automatically represented by a coarse scale discretization, together with some additional details at finer scales which are only needed near the singularity of the function. This is directly used in signal processing for data compression purposes. In the area of numerical simulation, this suggests that multiscale methods can be used to approximate the solution of a physical problem at a low memory and computational cost, if it is smooth except at some isolated singularities.

Note that the first applications of multilevel techniques in numerical simulation had a different objective: in the context of elliptic problems, multigrid methods were developed since the 1970s for the purpose of preconditioning rather than compression (see [6, 8] for a general survey of multiscale and wavelet methods in numerical analysis).
FULLY ADAPTIVE
MULTIRESOLUTION FINITE VOLUME SCHEMES
FOR CONSERVATION LAWS

ALBERT COHEN, SIDI MAHMOUD KABER, SIEGFRIED MÜLLER,
AND MARIE POSTEL

Abstract. The use of multiresolution decompositions in the context of finite
volume schemes for conservation laws was first proposed by A. Harten for the
purpose of accelerating the evaluation of numerical fluxes through an adaptive
computation. In this approach the solution is still represented at each time
step on the finest grid, resulting in an inherent limitation of the potential gain
in memory space and computational time. The present paper is concerned with
the development and the numerical analysis of fully adaptive multiresolution
schemes, in which the solution is represented and computed in a dynamically
evolved adaptive grid. A crucial problem is then the accurate computation of
the flux without the full knowledge of fine grid cell averages. Several solutions
to this problem are proposed, analyzed, and compared in terms of accuracy
and complexity.

1. Introduction

1.1. The background. The present work is concerned with the numerical solution
of the Cauchy problem for hyperbolic systems of conservation laws of the form
\begin{equation}
\partial_t u + \text{Div}_x (f(u(t, x))) = 0, \quad u \in \mathbb{R}^m, \quad x \in \mathbb{R}^d, \quad t > 0,
\end{equation}
with initial value $u(t=0, x) = u_0(x)$. It is well known that the exact solution to
such equations may develop singularities in finite time even when $u_0$ is smooth, so
that one needs to consider weak solutions. In the scalar case, uniqueness of such
solutions in $L^\infty_t ([0, +\infty[, \mathbb{R}^1_x)$ is ensured by appending certain entropy conditions.
It is also well known that the presence of singularities results in numerical difficul-
ties when it comes to the standard discretization of (1). In the case of the most
commonly used finite volume schemes, this is reflected by the fact that convergence
to the entropy solution can only easily be proved for low order monotone schemes,
and that even for high order schemes, the rate of convergence of the approximate
solution $u_h$ associated to a mesh size $h$ is inherently limited by the lack of global
smoothness of the solution: since it is at most in $BV$, one can certainly not expect a
better global estimate than $\|u - u_h\|_{L^1} \leq C(t) h$, and moreover the existing schemes

Received by the editor May 30, 2000 and, in revised form, February 6, 2001.
2000 Mathematics Subject Classification. Primary 41A58, 65M50, 65M12.
Key words and phrases. Conservation laws, finite volume schemes, adaptivity, multiresolution,
wavelets.
The work of S. Müller has been supported by the EU–TMR network “Wavelets in Numerical
Simulations”.
can only be proved to converge with the suboptimal rate $h^{1/2}$. We refer to [25] and [26] for a general account on these theoretical and numerical difficulties.

In this context, the use of adaptive methods appears as a natural idea to improve the computational efficiency. Intuitively, a good adaptive method should combine high order approximation in the smooth regions of the solution together with mesh refinement near the singularities. In the finite element setting, this is typically illustrated by $h$-$p$ methods (see e.g., [4]) in which one seeks an equilibrium between the high order rate $CH_p$ in the smooth regions and the low order rate $ch^q$, where $h \ll H$ and $q \ll p$. For nonstationary problems such as (1), a specific difficulty is that the singularities move as time progresses, so that the adapted mesh should be updated at each time step. Such “moving mesh techniques” are feasible (see e.g., [27]), but the numerical analysis of their performance is delicate, as well as their practical implementation in more than one dimension.

In this paper, we shall present an alternative strategy for adaptive discretizations, based on a coupling of multiresolution representation and finite volume schemes, which allows a relatively simple implementation and a rigorous error analysis.

1.2. Multiresolution methods. The use of multiresolution methods in numerical simulation has long been associated with multigrid techniques for the optimal preconditioning of elliptic operator equations. In recent years, the emergence of wavelet bases, and the observation of their dramatic compression properties for fairly general functions, has motivated the development of multiresolution methods for the purpose of adaptive computations for PDE’s (see [10] and [15] for surveys on wavelets in the context of numerical simulation). In the context of hyperbolic conservation laws, the use of adaptive multiresolution methods was initially proposed in the pioneering work by Berger and Collela [3] and Harten [21, 22]. Harten’s approach, which is closer in spirit to the present work, can be summarized as follows (more details will be given in Section 2). Given a finite volume scheme which operates on a fine mesh, the multiresolution representation of the numerical solution in terms of cell averages on a coarse mesh and wavelet coefficients at intermediate scales is used at each time step to indicate the local smoothness of the solution. This information is used to accelerate the scheme by saving on the evaluation of the numerical flux, which is exactly computed on the finest mesh only in the regions of poor smoothness (or high gradients), otherwise computed approximately from its exact computation on coarser meshes. While this strategy was initially developed for one-dimensional structured grids, several contributions have made it operational for various types of multivariate finite volume meshes (Cartesian [5, 8], curvilinear [16, 30], triangular [13, 31] and unstructured [1]). Several remarks should be made concerning such a strategy:

- The goal is not to improve the accuracy but rather to gain computational time while staying within the same accuracy as the reference finite volume scheme. One of the most attractive features of this approach is that it can be thought as an “accelerating” device that can be superimposed on an existing code without requiring significant changes to the existing structure.
- This strategy is particularly effective for high resolution methods (such as the ENO schemes introduced in [23]) in which the flux computations are heavy and dominate the overall computational cost. In such a case, the observed gain in computational time can typically reach up to 5, depending on the test configuration at hand. However, this gain is inherently limited by the
fact that the evolution still takes place at the finest resolution level: the
multiresolution structure is not used to compute and represent the solution
in a compressed form, but only to accelerate the flux computations. In turn,
both the computational cost and the memory storage size remain proportional
to the cardinality of the finest mesh. In this sense Harten’s approach should
be qualified as “semi-adaptive”.

• The numerical analysis of the error produced by the approximate flux com-
putation is not fully understood. In particular, it typically relies on heuristic
arguments of the following type: if \( \Lambda_n(\varepsilon) \) is the set of wavelet coefficients
above some threshold \( \varepsilon \) for the numerical solution at time step \( n \), one is able
to “predict” a set \( \tilde{\Lambda}_{n+1}(\varepsilon) \) which both contains \( \Lambda_n(\varepsilon) \) and \( \Lambda_{n+1}(\varepsilon) \). To our
knowledge, the existing strategies for constructing the set \( \tilde{\Lambda}_{n+1}(\varepsilon) \) have not
yet been fully justified by rigorous analysis.

1.3. Objectives and outline of the paper. The objectives of the present paper
are twofold. First, we shall provide a fully adaptive algorithm in which both the
computational cost of one time step and the memory storage are proportional
to the number of wavelet coefficients describing the solution. Second, we shall provide
an error analysis for this scheme, which in particular includes a rigorous proof of
the above heuristics.

The general form of the algorithm can be roughly summarized in the following
way, which is typical of wavelet-based schemes. Given \( u^n \) the numerical solution
at time \( n\Delta t \), represented in a compressed form by a set \( \Lambda_n \) of wavelets (i.e., \( u^n = \sum_{\lambda \in \Lambda_n} d^n_\lambda \psi_\lambda \)), we produce \( u^{n+1} \) and \( \Lambda_{n+1} \) at time \( (n+1)\Delta t \) by three basic steps:

• **Refinement:** We produce from \( \Lambda_n \) a larger set \( \tilde{\Lambda}_{n+1} \) such that \( \Lambda_n \subset \tilde{\Lambda}_{n+1} \),
  which is potentially adapted to describe both \( u^n \) and \( u^{n+1} \) with the required
  accuracy.

• **Evolution:** We compute from \( u^n \) an approximate solution \( \tilde{u}^{n+1} = \sum_{\lambda \in \tilde{\Lambda}_{n+1}} d^{n+1}_\lambda \psi_\lambda \)
at the next time step.

• **Thresholding:** We use a thresholding procedure on \( \tilde{u}^{n+1} \) to obtain the set
  \( \Lambda^{n+1} \subset \tilde{\Lambda}^{n+1} \) and approximate solution \( u^{n+1} = \sum_{\lambda \in \Lambda_{n+1}} d^{n+1}_\lambda \psi_\lambda \).

An important feature of the scheme is that the wavelet coefficients of \( \Lambda_n \) or
\( \tilde{\Lambda}_{n+1} \) are always organized in a *tree structure*: whenever an index \( \lambda \) is included
in such a set, all other indices corresponding to the same spatial locations and to
coarser resolutions are also included. With this particular structure, the data of the
coefficients \( d^n_\lambda \) for \( \lambda \in \Lambda_n \) turns out to be equivalent to the discretization of \( u^n \) by its
cell averages on an adaptive discretization \( S(\Lambda_n) \) (with the same cardinality as \( \Lambda_n \)).
As in Harten’s approach we impose a maximal resolution level, and our goal is to
remain within the order of accuracy achieved by the reference finite volume scheme
on this finest mesh, while saving the maximal amount of computational time and
memory space. However, the saving is potentially much more substantial since it
is now proportional to the compression rate of the solution which is often smooth
except at isolated singularities. It is also interesting to note that, in contrast to
Harten’s approach, the savings can be very effective in the context of a low order
finite volume scheme. A first algorithm of this type was proposed in [20], and is
currently being tested for 2D problems with tensor product type discretizations and
parametric patches to adapt the geometry of the domain. Since one of the main goals of the present paper is to come out with a rigorous error analysis of such adaptive algorithms, we had to reconsider, in detail, the three basic steps of the scheme.

While the error produced by the thresholding step is easily analyzed, provided that the multiscale reconstruction algorithm has some stability properties, the two other steps deserve more attention. The refinement step is meant to guarantee the accuracy of approximation at the next time step. Our analysis will reveal that this is indeed possible by exploiting certain features of the finite volume scheme as well as the characterization of local Hölder smoothness from the decay properties of wavelet coefficients. The resulting refinement strategy is slightly different from the one proposed in [22], which was mostly based on heuristic arguments. It is actually more severe and thus more costly. As we shall see in the numerical examples section, Harten’s refinement strategy is, in practice, sufficient to guarantee the accuracy of approximation at the next time except in certain pathological situations where our refinement strategy seems necessary. Concerning the evolution step, the main difficulty is to obtain an accurate flux evaluation in order to update correctly the cell average on the adaptive mesh $S(\hat{A}_{n+1})$ associated to $\hat{A}_{n+1}$, without the knowledge of the fine grid cell averages. Note that this problem does not occur in Harten’s approach, since these fine grid data are used to describe the solution and are thus available during the computation. A first idea could be to apply on the adaptive mesh the same finite volume scheme which is used at the finest level. Unfortunately, this results in an important loss of accuracy in the case of low order finite volume schemes, since the error is then dominated by the size of the coarsest cell which appears in the adaptive mesh $S(\hat{A}_{n+1})$ (see e.g., [29] for this type of error estimate). We shall propose several strategies in order to obtain a sharper evaluation of the flux, as well as the desired error estimate. In turn we shall derive a rigorous error analysis, in the sense that we are able to tune the thresholding and refinement strategy in order to ensure a prescribed distance between the numerical solutions obtained by the reference finite volume scheme and by the adaptive algorithm. For the sake of simplicity we present this analysis in the 1D scalar case. Its extension to the case of systems and multidimensional Cartesian grids is essentially technical, while it is probably much more delicate for unstructured grids. Note that such an analysis provides an a priori error estimate between the adaptive solution and the exact solution, only if such an estimate is available for the reference finite volume scheme, i.e., essentially in the case of low order schemes for scalar equations.

The resulting scheme will meet the above mentioned objectives of adaptivity, combining a low order scheme on the finest mesh in the regions of large variations with higher order approximations on coarser meshes in the regions of smoothness. As it will be demonstrated on some benchmark numerical tests, the gain in computational time and memory size can reach very large values (above 30 in some of our examples), while remaining within the order of accuracy of the reference finite volume scheme. These tests also allowed us to notice that our analysis is satisfactory in terms of estimating the error produced at each time step by thresholding and estimating the grid from the previous time step, but becomes much too pessimistic in large time since these errors do not fully accumulate.

The paper is organized as follows. In Section 2, we give an account on multiresolution representations in the context of finite volume discretizations, and we
discuss the issue of data compression. In Section 3, we recall Harten’s multiresolution scheme for conservation laws, and we present the fully adaptive scheme with different strategies for an accurate flux computation. The core of the numerical analysis of this scheme is given in Section 4. Numerical tests for 1D and 2D (scalar and system) problems are presented in Section 5 together with a discussion concerning the practical relevance of the refinement strategy and error estimate. We end by listing some remaining trends and perspectives in Section 6.

Acknowledgment. The authors are grateful to the anonymous referee who greatly helped improve the initial version of this paper, in particular the discussion on the relevance of the error estimates.

2. Multiresolution meets finite volumes

2.1. Finite volume schemes. In the context of finite volume schemes, the natural discretization of the solution to (1) is by its cell averages: the spatial domain being partitioned into “cells” \((\Omega_\gamma)_{\gamma \in S}\)—typically intervals in 1D, triangles or quadrangles in 2D—one is interested in computing approximations \(u^n_\gamma\) of the exact averages of \(u\) at discrete times \(n \Delta t\), i.e.,

\[
u^n_\gamma \approx \overline{u}^n_\gamma := |\Omega_\gamma|^{-1} \int_{\Omega_\gamma} u(x, n \Delta t) dx.
\]

By the divergence theorem, the exact average satisfies

\[
u^{n+1}_\gamma = \nu^n_\gamma - |\Omega_\gamma|^{-1} \int_{n \Delta t}^{(n+1) \Delta t} \int_{\partial \Omega_\gamma} f(u(x, t)) n_\gamma(x) dx dt,
\]

where \(n_\gamma(x)\) is the outer normal vector to \(\Omega_\gamma\). This can be rewritten as

\[
u^{n+1}_\gamma = \nu^n_\gamma - \Delta t |\Omega_\gamma|^{-1} \sum_{\mu \neq \gamma} |\Gamma_{\gamma,\mu}| F^{n}_{\gamma,\mu},
\]

where the sum is over all \(\mu \neq \gamma\) such that the interface \(\Gamma_{\gamma,\mu} = \Omega_\gamma \cap \Omega_\mu\) is not trivial and where \(F^{n}_{\gamma,\mu}\) denotes the average flux across this interface between \(n \Delta t\) and \((n + 1) \Delta t\), i.e.,

\[
F^{n}_{\gamma,\mu} := |\Delta t|^{-1} |\Gamma_{\gamma,\mu}|^{-1} \int_{n \Delta t}^{(n+1) \Delta t} \int_{\Gamma_{\gamma,\mu}} f(u(x, t)) n_\gamma(x) dx dt.
\]

The conservative structure of the equation is expressed by the balance \(F^{n}_{\gamma,\mu} + F^{n}_{\mu,\gamma} = 0\). A finite volume scheme has the form

\[
u^{n+1}_\gamma = \nu^n_\gamma - \Delta t |\Omega_\gamma|^{-1} \sum_{\mu} |\Gamma_{\gamma,\mu}| F^{n}_{\gamma,\mu},
\]

where the numerical fluxes \(F^{n}_{\gamma,\mu}\) are approximations of \(F^{n}_{\gamma,\mu}\) computed from the values \(u^\lambda_n, \lambda \in S\). The scheme is said to be conservative if these approximations also satisfy \(F^{n}_{\gamma,\mu} + F^{n}_{\mu,\gamma} = 0\). In the simplest (low order) schemes, \(F_{\gamma,\mu}\) is typically a function of \(u^n_\gamma\) and \(u^n_\mu\), while higher order schemes require additional values \(u^n_\nu\) corresponding to neighboring cells \(\Omega_\nu\).

In the context of nonsmooth solutions of scalar conservation laws, certain finite volume schemes—mostly first order accurate—can be proved to converge to the entropy solution of (1) in the \(L^1\) norm, with rate \(O(h^{1/2})\) for Cartesian grids [29] and \(O(h^{1/4})\) for unstructured grids [9], where \(h := \max_{\gamma \in S} \text{diam}(\Omega_\gamma)\). The use of
higher order schemes is both limited by the theoretical difficulties in analyzing their convergence to the entropy solution and by the important computational cost of evaluating the fluxes $F_{\gamma \mu}$. This last difficulty was the initial motivation for the use of multiresolution techniques in the context of finite volume schemes.

2.2. **Finite volume multiresolution and wavelets.** Finite volume multiresolution is based on considering nested finite volume discretizations: for $j = 0, 1, \cdots, J$, we are given regular disjoint partitions $(\Omega_\gamma)_{\gamma \in S_j}$ of $\mathbb{R}^d$ (or of a domain of interest) such that each $\Omega_\gamma$, $\gamma \in S_j$, is the union of a finite number of cells $\Omega_\mu$, $\mu \in S_{j+1}$. The index $j$ refers to the scale level in the sense that there exist fixed constants $c, C$ such that

$$c2^{-j} \leq \text{diam}(c_\gamma) \leq C2^{-j}, \quad \gamma \in S_j,$$

where $c_\gamma$ (resp. $C_\gamma$) are balls contained in (resp. containing) $\Omega_\gamma$. In order to keep track of the scale $j$ associated to an index $\gamma$, we shall make use of the notation

$$|\gamma| := j \text{ if } \gamma \in S_j.$$

**Basic example:** For sake of simplicity we shall often refer to the most basic example of the univariate dyadic intervals

$$\Omega_\gamma = \Omega_{j,k} := [2^{-j}k, 2^{-j}(k+1)], \quad \gamma \in S_j := \{(j,k) ; \ k \in \mathbb{Z}\},$$

which clearly fulfills all the above prescriptions.

2.2.1. **Projection.** Consider a vector $U_j := (u_\gamma)_{\gamma \in S_j}$ of discrete data on the grid $S_j$. If we think of this vector as representing the cell-averages of some function $u \in L^1(\mathbb{R}^d)$, i.e.,

$$u_\gamma := |\Omega_\gamma|^{-1} \int_{\Omega_\gamma} u(x) dx,$$

it is natural to introduce a straightforward projection operator $P^j_{j-1}$, which maps $U_j$ to $U_{j-1}$, by the simple remark that since the partitions $S_j$ are nested, we obtain the averages at the coarser level by

$$u_\gamma = |\Omega_\gamma|^{-1} \sum_{|\mu| = |\gamma|+1, \Omega_\mu \subset \Omega_\gamma} |\Omega_\mu| u_\mu.$$

In the univariate dyadic case this amounts in half-summing the averages at the finer level, i.e., $u_{j-1,k} = (u_{j,2k} + u_{j,2k+1})/2$. It is clear that from the data of $U_j$ one can derive $U_{j-1}, U_{j-2}, \cdots, U_0$ by iterative application of the operators $P^j_{j-1}$.

2.2.2. **Prediction.** We next introduce a prediction operator $P^j_{j-1}$, which maps $U_{j-1}$ to an approximation $\hat{U}_j$ of $U_j$. In contrast to the projection operator, there is an infinite number of choice for definition of $P^j_{j-1}$, but we impose at least two basic constraints:

- The prediction is local, i.e., $\hat{u}_\mu$ depends on the values $u_\gamma$ on a finite stencil $R_\mu$ surrounding $\Omega_\mu$, i.e., such that

$$R_\mu \subset \{ \gamma : |\gamma| = |\mu| - 1 \text{ and } \text{dist}(\Omega_\gamma, \Omega_\mu) \leq M2^{-|\mu|} \},$$

for some fixed $M$. 


The prediction is consistent with the projection in the sense that
\[ |\Omega_\gamma|u_\gamma = \sum_{|\mu|=|\gamma|+1, \Omega_\mu \subset \Omega_\gamma} |\Omega_\mu|\hat{u}_\mu, \]
i.e., it is conservative with respect to the coarse grid cell averages, or equivalently \( P_j^{j-1} P_j^{j-1} = \text{Id} \). Note that this property implies that the stencil \( R_\mu \) must contain the unique index \( \gamma \) such that \( |\mu| = |\gamma| + 1 \) and \( \Omega_\mu \subset \Omega_\gamma \).

A trivial example of such a reconstruction operator is by simply taking \( \hat{u}_\mu = \hat{u}_\gamma \), if \( \Omega_\mu \subset \Omega_\gamma \).

Note that we do not a priori impose the linearity of the prediction operator. However, this property will be helpful in the subsequent numerical analysis, together with other additional assumptions on \( P_j^{j-1} \) (stability and accuracy) that we shall address further.

2.2.3. Multiresolution decomposition. We can define the prediction error at level \( j \) as the differences between the exact and predicted values, i.e.,
\[ d_\mu := u_\mu - \hat{u}_\mu. \]
From the consistency assumption, we see that this error satisfies the dependence relations
\[ \sum_{|\mu|=|\gamma|+1, \Omega_\mu \subset \Omega_\gamma} |\Omega_\mu|d_\mu = 0. \]
This redundancy suggests that we select a set \( \nabla_j \subset S_j \) obtained by removing for each \( \gamma \in S_{j-1} \) one \( \mu \in S_j \) such that \( \Omega_\mu \subset \Omega_\gamma \). Then, defining the detail vector \( D_j = (d_\mu)_{\mu \in \nabla_j} \), it is clear that there is a one-to-one correspondence between \( U_j \) and \( (U_{j-1}, D_j) \) which can be implemented using the operators \( P_j^{j-1} \) and \( P_j^{j-1} \). In the univariate dyadic case the detail vector can be simply defined as \( D_j = (d_{j,k})_{k \in \mathbb{Z}} \) with \( d_{j,k} = (u_{j,2k} - \hat{u}_{j,2k}) \).

By iteration of this decomposition, we obtain a multiscale representation of \( U_J \) in terms of \( M_J = (U_0, D_1, D_2, \ldots, D_J) \). Using the local structure of the projection and prediction operators, we can implement the multiscale transformation
\[ M : U_J \mapsto M_J \]
and its inverse \( M^{-1} \) with optimal complexity \( O(N_J) \), where \( N_J := \#(S_J) \) represents the dimension of \( U_J \).

2.2.4. Wavelets. In the case where \( P_j^{j-1} \) is linear, i.e.,
\[ \hat{u}_\mu := \sum_{\gamma} c_{\mu,\gamma} u_\gamma, \]
\( M \) and \( M^{-1} \) are simple changes of bases. If the \( U_j \) are given by \( (10) \), using the wavelet terminology, we can write
\[ u_\gamma := \langle u, \varphi_\gamma \rangle, \]
where the dual scaling function \( \varphi_\gamma \) is simply
\[ \varphi_\gamma := |\Omega_\gamma|^{-1} \chi_{\Omega_\gamma}, \]
and
\begin{equation}
\hat{d}_\mu = u_\mu - \hat{u}_\mu = \langle u, \tilde{\varphi}_\mu \rangle - \sum_\gamma c_{\mu,\gamma} \langle u, \tilde{\varphi}_\gamma \rangle = \langle u, \tilde{\psi}_\mu \rangle,
\end{equation}
where the dual wavelet \( \tilde{\psi}_\mu \) is given by
\begin{equation}
\tilde{\psi}_\mu := \tilde{\varphi}_\mu - \sum_\gamma c_{\gamma,\mu} \tilde{\varphi}_\gamma.
\end{equation}
In the rest of this paper, in order to describe in a simple way the multiresolution vector, we define \( \nabla J := \bigcup_{j=0}^{J} \nabla_j \) with \( \nabla_0 := S_0 \) and write
\begin{equation}
M_J = (d_\lambda)_{\lambda \in \nabla J} = (\langle u, \tilde{\psi}_\lambda \rangle)_{\lambda \in \nabla J},
\end{equation}
where we have set \( d_\lambda = u_\lambda \) and \( \tilde{\psi}_\lambda = \tilde{\varphi}_\lambda \) if \( \lambda \in \nabla_0 \).

In the case of a structured grid, e.g., in the univariate dyadic case, it is natural to impose a simple translation invariant structure on the prediction operator, i.e., a rule of the form
\begin{equation}
\hat{u}_{j,k} = \sum_mC_{k-2m}u_{j-1,m},
\end{equation}
with some possible additional adaptations near the boundary of the domain. Since we have \( \tilde{\varphi}_{j,k} = \Omega_{j,k}^{-1} \chi_{\Omega_{j,k}} = 2^j \tilde{\varphi}(2^j \cdot - k) \) with \( \tilde{\varphi} := \chi_{[0,1]} \), this results in the usual structure \( \tilde{\psi}_{j,k} := 2^j \psi(2^j \cdot - k) \) for the wavelets. Note that the dual scaling functions and wavelets are here normalized in \( L^1 \). In more general situations, we also have \( \| \tilde{\varphi}_\gamma \|_{L^1} = 1 \) by (20) and \( \| \tilde{\psi}_\lambda \|_{L^1} \leq C \) independently of \( \lambda \) in view of (22), if we assume a uniform bound on the prediction coefficients \( c_{\mu,\gamma} \) (which always exists for the prediction operators which have been used for our schemes).

In the univariate dyadic setting, note that the simple prediction operator defined by (14) leads to the well-known Haar system
\begin{equation}
\tilde{\psi}_{j,k} := 2^j (\chi_{\Omega_{j+1,2k}} - \chi_{\Omega_{j+1,2k+1}}).
\end{equation}

### 2.3. Compression

One of the main interests in decomposing \( U_J \) into \( M_J \) is that this new representation is often more appropriate for data compression. This property will be crucial for the practical efficiency of our scheme.

Given a set \( \Lambda \subset \nabla J \) of indices \( \lambda \), we define a truncation operator \( T_\Lambda \) acting on multiscale representations that leaves unchanged the component \( d_\lambda \) if \( \lambda \in \Lambda \) and replaces it by 0, otherwise. In practice, we are typically interested in sets \( \Lambda \) obtained by thresholding: given a set of level-dependent threshold \( (\varepsilon_0, \varepsilon_1, \ldots, \varepsilon_J) \), we set
\begin{equation}
\Lambda = \Lambda(\varepsilon_0, \varepsilon_1, \ldots, \varepsilon_J) := \{ \lambda \text{ s.t. } |d_\lambda| \geq \varepsilon_{|\lambda|} \}
\end{equation}
and define the corresponding thresholding operator \( T_\Lambda \).

Applying \( T_\Lambda \) on the multiscale decomposition of \( U_J \) amounts to building a nonlinear approximation \( A_\Lambda U_J \), where the operator \( A_\Lambda \) is given by
\begin{equation}
A_\Lambda := M^{-1} T_\Lambda M.
\end{equation}
Here \( A_\Lambda \) is a nonlinear operator since \( \Lambda \) depends of \( U_J \) according to (26). When \( \Lambda \) is fixed independently of \( U_J \), it is a standard linear projection. A substantial survey on nonlinear approximation—in particular by thresholding procedures—can be found in [18]. One of its main interests to us is the ability to resolve adaptively piecewise smooth functions with a small number of parameters, since we expect...
that the unthresholded details in the finest scale are only concentrated near the isolated singularities. Such compression properties are however tied to additional assumptions on the prediction operator $P^{j-1}_j$: polynomial accuracy and multiscale stability.

2.3.1. Accuracy. This first assumption means that the prediction has some prescribed order $N > 0$ of accuracy or equivalently is exact for polynomials of degree $N - 1$, i.e., if $u \in \Pi_{N-1}$, then $u_\gamma = \hat{u}_\gamma$ for all $\gamma$. In other words, for all $u \in \Pi_{N-1}$ and for all $\lambda \in \nabla J$, we have

$$\langle u, \hat{\psi}_\lambda \rangle = d_\lambda = 0,$$

i.e., the first $N$ moments of the dual wavelets are zero. Such a property has an immediate consequence on the size of the $d_\lambda$ in the smooth regions: if $u$ has $C^s$ smoothness within the support $\tilde{\Sigma}_\lambda$ of $\hat{\psi}_\lambda$ for some $s \leq N$, we can use that $d_\lambda = \langle u - p, \hat{\psi}_\lambda \rangle$ for all $p \in \Pi_{N-1}$ to estimate this coefficient by

$$|d_\lambda| \leq \inf_{p \in \Pi_{N-1}} \| u - p \|_{L^\infty(\tilde{\Sigma}_\lambda)} \| \hat{\psi}_\lambda \|_{L^1},$$

$$\leq C \inf_{p \in \Pi_{N-1}} \| u - p \|_{L^\infty(\tilde{\Sigma}_\lambda)}$$

$$\leq C 2^{-s|\lambda|} |u|_{C^s(\tilde{\Sigma}_\lambda)}.$$

Here we have used the properties of local polynomial approximation on $\tilde{\Sigma}_\lambda$ which has size of order $O(2^{-|\lambda|})$ and the fact that the $\hat{\psi}_\lambda$ are normalized in $L^1$. The fast decay of the detail coefficients in the smooth regions will thus be ensured if $N$ is sufficiently large.

Note that the prediction operator (14) associated with the Haar system is only exact for constants, i.e., the multiresolution is “first order accurate”. A possible way to raise accuracy is to define $P^{j+1}_j$ through a procedure of polynomial reconstruction, which is easy to operate for structured grids. For the sake of simplicity, let us restrict to the univariate dyadic case: consider the centered stencil $(u_{j,k-M}, \ldots, u_{j,k+M})$ and define the unique polynomial $p_{j,k}$ of degree $2M$ such that

$$2^j \int_{\Omega_j} p_{j,k}(x) dx = u_{j,l}, \quad l = k - M, \ldots, k + M.$$

Then we simply define the prediction in the two half intervals of $\Omega_{j,k}$ by using the averages of $p_{j,k}$, i.e.,

$$\hat{u}_{j+1,2k} = 2^{j+1} \int_{\Omega_{j+1,2k}} p_{j,k}(x) dx \quad \text{and} \quad \hat{u}_{j+1,2k+1} = 2^{j+1} \int_{\Omega_{j+1,2k+1}} p_{j,k}(x) dx.$$

Clearly this process is exact for polynomials of degree $2M$, i.e., has accuracy order $N = 2M + 1$. It is also clear that raising the order imposes larger stencils. In our 1D simulations, we have used the case $M = 1$, i.e., a third order accurate multiresolution. In this case, the prediction is explicitly given by

$$\hat{u}_{j+1,2k} = u_{j,k} + \frac{1}{8}(u_{j,k-1} - u_{j,k+1}) \quad \text{and} \quad \hat{u}_{j+1,2k+1} = u_{j,k} + \frac{1}{8}(u_{j,k+1} - u_{j,k-1}).$$

2.3.2. Multiscale stability. The second assumption means that we are able to control the effect of thresholding on the resulting approximation error between $U_J$ and $A_{\Lambda}U_J$, in some prescribed norm. If the prediction is linear, this amounts to understanding the individual contribution of each detail coefficient $d_\lambda$ to the reconstruction on the fine mesh by $M^{-1}$. This contribution is given by $d_\lambda \Psi_{J,\lambda}$,
where $\Psi_{J,\lambda}$ is the corresponding (discrete) vector of the basis associated to the multiscale decomposition, which is obtained by applying $M^{-1}$ to the Dirac vector $M_{\lambda} := (\delta_{\lambda,\mu})_{\mu \in \nabla J}$.

Such a reconstruction can be decomposed in two steps: one first reconstructs from $M_{\lambda}$ a cell-average vector on the grid $S_{|\lambda|}$, then iteratively applies on this vector the prediction operator $P_{j}^{J-1}$ for $j = |\lambda| + 1, \ldots, J$ without adding details. For example, in the univariate dyadic case if $\lambda = (j, k)$, then $\Psi_{J,\lambda} = \Psi_{j,k}$ is explicitly defined by

$$
\Psi_{j,k} = P_{j}^{J-1}P_{j-1}^{J-2} \cdots P_{j+1}^{0}(0, \cdots, 0, 1, 0, \cdots, 0),
$$

with 1 at position $2k$ and $-1$ at position $2k + 1$, or equivalently by $\Psi_{j,k} := \Phi_{j,2k} - \Phi_{j,2k+1}$, where

$$
\Phi_{j,k} = P_{j}^{J-1}P_{j-1}^{J-2} \cdots P_{j+1}^{0}(0, \cdots, 0, 1, 0, \cdots, 0),
$$

with 1 at position $k$. We are thus interested in the stability of the iterative applications of the prediction operators $P_{j}^{j-1}$.

This problem is particularly well understood for structured grids, e.g., in the univariate dyadic case, which allows a similar refinement process from scale to scale. In such cases, a natural way to address this problem is by analyzing the convergence of $\Psi_{J,\lambda}$ (viewed as piecewise constant functions on the grid $S_{J}$) to limit functions $\psi_{\lambda}$ as the refinement level $J$ goes to $+\infty$. We thus consider an infinite hierarchy of discretizations $(S_{j})_{j \geq 0}$, and we define the full set of indices

$$
\nabla := \bigcup_{j \geq 0} \nabla_{j}.
$$

The study of the limit functions to such refinement processes (or subdivision algorithms) is a well-known task in computer-aided geometric design and wavelet theory, and it is particularly understood in the case of uniform refinements on structured grids. We refer to [19] or [7] for general surveys on subdivision algorithms and to [17] or [10] for their relations to wavelets, and we simply recall here some basic facts. If the subdivision process converges at least in $L^{1}$, then one can check that the limit functions $\psi_{\lambda}$ constitute together with $\tilde{\psi}_{\lambda}$ a biorthogonal wavelet system similar to those introduced in [12]: an arbitrary function $u \in L^{1}$ can be synthesized according to

$$
u = \sum_{j \geq 0} \sum_{|\lambda| = j} \langle u, \tilde{\psi}_{\lambda} \rangle \psi_{\lambda},
$$

and we have the duality relations

$$
\langle \tilde{\psi}_{\lambda}, \psi_{\mu} \rangle = \delta_{\lambda,\mu}.
$$

The synthesis and analysis functions $\psi_{\lambda}$ and $\tilde{\psi}_{\lambda}$ are called the primal and dual wavelets.

In the univariate dyadic case, assuming that the prediction operator has the Toeplitz structure (24), the primal wavelets have the general form

$$
\psi_{j,k} = \psi(2^j \cdot -k).
$$

In more general cases, we also have the $L^\infty$ normalisation, i.e., $\|\psi_{\lambda}\|_{L^\infty} \leq C$ independently of $\lambda$, provided that the subdivision process converges in $L^\infty$. 
At the discrete level \( J \), for \( |\lambda| \leq J \), the vector \( \Psi_{J,\lambda} \) coincides with the cell-averages of \( \psi_{\lambda} \) at level \( J \), i.e., \( \Psi_{J,\lambda} = \{(\psi_{\lambda}, \hat{\varphi}_{\gamma})\}_{\gamma \in S_J} \). Defining at level \( J \) the normalized \( \ell^1 \) metric by

\[ \|U_J\| := 2^{-d_J} \sum_{\lambda \in S_J} |u_{\lambda}|, \]

which is equivalent to the \( L^1 \) norm of the corresponding piecewise constant function, we see that

\[ \|\Psi_{J,\lambda}\| \leq C\|\psi_{\lambda}\|_{L^1} \leq C2^{-d|\lambda|}. \]

We can therefore control the effect of thresholding by the following estimate

\[ \|U_J - A_\Lambda U_J\| = \left\| \sum_{\lambda \in \Lambda} d_\lambda \Psi_{J,\lambda} \right\| \leq C \sum_{\lambda \notin \Lambda} |d_\lambda|2^{-d|\lambda|} = C \sum_{|d_\lambda| \leq \epsilon |\lambda|} |d_\lambda|2^{-d|\lambda|}. \]

In the following, we shall always use a threshold of the type

\[ \epsilon_J := 2^{d_J} \eta, \]

which amounts to keeping the largest \( L^1 \) contributions \( |d_\lambda \Psi_{\lambda}| \) (or \( |d_\lambda \psi_{\lambda}|_{L^1} \)). This strategy—which was already proposed by Harten in [22]—finds some rigorous justification in the theory of nonlinear approximation: roughly speaking, for a large variety of norms \( \| \cdot \|_X \), one can prove that an approximation of a function \( u \) by an \( N \)-term combination of wavelets \( u_N \) which is nearly optimal when measured in \( X \) (i.e., \( \|u - u_N\|_X \leq C\|u - v_N\|_X \) for any other \( N \)-term combination \( v_N \)) is simply obtained as the truncated expansion of \( u \), keeping only the \( N \) largest \( \|d_\lambda \hat{\psi}_\lambda\|_X \) (see e.g., [18] or [10] for such types of results). Since we are targeting an \( L^1 \) error estimate, it is thus natural to use such a level dependent threshold in order to minimize the number \( N \) of degrees of freedom for the prescribed \( L^1 \) accuracy, and it is thus also crucial to use prediction operators such that the primal wavelets are at least in \( L^1 \).

If we work on a finite domain, we can then go further by deriving the estimate

\[ \|U_J - A_\Lambda U_J\| \leq C\#(\nabla^J)\eta = C\#(S_J)\eta \leq C2^{Jd}\eta. \]

This last estimate led Harten to the more specific choice \( \eta = 2^{-dJ}\epsilon \), i.e.,

\[ \epsilon_J := 2^{d(J-J)}\epsilon, \]

in order to ensure a thresholding error of prescribed order \( \epsilon \).

Note however that (43) is very crude since it bounds all the thresholded \( L^1 \) components by \( \eta \) while many of them might be much smaller. In other words the choice of \( \Lambda \) by (26) and (44) might keep too many coefficients for the target accuracy. A smarter thresholding strategy, which also ensures an error of order \( \epsilon \) while minimizing the number of preserved coefficients, is the following: sort the indices \( \lambda \) into a sequence \( (\lambda(n))_{n \geq 0} \) such that the sequence \( d^*_n := 2^{-|\lambda(n)|}\|d_{\lambda(n)}\| \) is nondecreasing, find the largest \( N \) such that \( \sum_{n=0}^N d^*_n \leq \epsilon \), and then define \( \Lambda := \{ (\lambda(n)) : n > N \} \). Note that this still amounts to defining \( \Lambda \) according to (26) with \( \epsilon_J = 2^{d_J}\eta \) but with \( \eta \) possibly much larger than \( 2^{-dJ}\epsilon \). In the schemes that will be presented further, we shall nevertheless use (44). One of the specificities of this threshold is that it also ensures an error estimate of order \( \epsilon \) in the sup-norm.
(and by interpolation in all other discrete $\ell^p$ norms) if the primal wavelets $\psi_\lambda$ are in $L^\infty$. Indeed, we have

$$\|U_J - A_\Lambda U_J\|_{\ell^\infty} = \|\sum_{|\lambda| \leq \varepsilon_r} d_\lambda \Psi_{J,\lambda}\|_{\ell^\infty}$$

\begin{equation}
(45)
\end{equation}

$$\leq \sum_j = 0 \|\sum_{|d_\lambda| \leq \varepsilon_r, |\lambda| = j} d_\lambda \Psi_{J,\lambda}\|_{\ell^\infty}$$

$$\leq C \sum_j = 0 \sup_{|\lambda| = j} |d_\lambda \Psi_{J,\lambda}|_{\ell^\infty}$$

$$\leq C \sum_j = 0 \varepsilon_j \leq C\varepsilon,$$

where we have used the fact that the $\psi_\lambda$ and $\Psi_{J,\lambda}$ are normalized in the sup-norm, and that at fixed level $|\lambda| = j$, the $\Psi_{J,\lambda}$ do not overlap too much in the sense that

\begin{equation}
(46)
\end{equation}

$$\|\sum_{|\lambda| = j} d_\lambda \Psi_{J,\lambda}\|_{\ell^\infty} \leq C \sup_{|\lambda| = j} \|d_\lambda \Psi_{J,\lambda}\|_{\ell^\infty}.$$

The integrability of the primal scaling functions and wavelets is in some sense a minimal requirement for controlling the effect of thresholding. However, the analysis of our scheme will also rely on stronger smoothness properties of these functions, in relation with the following fact: if the $\psi_\lambda$ are in $C^r$, there is a converse to (29) which says that such a decay property is an effective local smoothness indicator. More precisely, if $\Sigma$ is a given domain and if for some $s < r$ we have the estimate $|\langle u, \psi_\lambda \rangle| \leq C 2^{-s|\lambda|}$ for all $\lambda \in \nabla$ such that the support of $\psi_\lambda$ intersects $\Sigma$, then $u$ has $C^s$ smoothness on $\Sigma$. Here $C^s$ is the usual Hölder class when $s$ is fractional, and when $s$ is an integer it should be replaced by the Besov space $B^s_{\infty,\infty}$ (which is then slightly larger than $C^s$; see e.g., [28, 24, 10, 17] for such results). We shall use a discrete counterpart of such a result (Lemma 4.3) in the analysis of the adaptive schemes that we shall develop in Secton 3.

Note that in the case of the prediction operator (14) associated with the Haar system, we have $\hat{\psi} = \psi$ and therefore the primal wavelet has no Hölder smoothness.

In contrast, it is known that the limit functions associated with (32) have $C^r$ smoothness for all $r < 1$.

2.3.3. Tree-structured compression. In our particular context, we shall be interested in that the set of preserved indices $\Lambda$ has a tree structure. In order to define such structures properly, we introduce the following terminology: if $\Omega_\mu \subset \Omega_\gamma$ with $|\gamma| = |\mu| - 1$, we say that $\mu$ is a “child” of $\gamma$ and that $\gamma$ is the “parent” of $\mu$. Note that by the definition of $\nabla_j$, if $\gamma$ has $N(\gamma)$ children, $N(\gamma) - 1$ of them are in $\nabla$, i.e., represent a detail. We call these the “detail children” of $\gamma$.

**Definition 2.1.** A set of indices $\Lambda \in \nabla$ is a tree if it holds that:

(i) the fundamental level $\nabla_0 = S_0$ is contained in $\Lambda$.

(ii) if $\mu$ and $\nu$ are detail children of the same $\gamma$, then $\mu \in \Lambda$ if $\nu \in \Lambda$.

(iii) if $\gamma$ is such that its detail children are in $\Lambda$, then the parent of $\gamma$ has the same property.

In the univariate dyadic case, this definition can be rewritten in the following simpler form: $\nabla_0 \in \Lambda$ and

\begin{equation}
(47)
\end{equation}

$$(j, k) \in \Lambda \Rightarrow (j - 1, [k/2]) \in \Lambda.$$

In particular, assumption (ii) is useless since each $\gamma = (j, k)$ has only one detail child, corresponding to the detail $d_{j,k}$.

The importance of tree structures is in that they are associated in a natural way to a “hybrid” discretization by cells of various levels. We first define the leaves $\mathcal{L}(\Lambda)$
of a tree $\Lambda$ as those $\lambda \in \Lambda$ which have no children in $\Lambda$. Clearly the $\Omega_\lambda$, $\lambda \in \mathcal{L}(\Lambda)$, are disjoint but they do not form a partition, i.e., they do not tile the whole domain. For example, in the case where the leaves are all at level $j$, i.e., $\mathcal{L}(\Lambda) = \nabla_j$, we do not have a partition since $\nabla_j$ was built as a strictly smaller subset of $\mathcal{S}_j$ in order to have a nonredundant detail vector. In order to have a partition, we need to add to $\mathcal{L}(\Lambda)$ all those $\mu$ which have a common parent with some $\lambda \in \mathcal{L}(\Lambda)$ but are not in $\nabla_j$ (in the case $\mathcal{L}(\Lambda) = \nabla_j$, this corresponds to complete $\nabla_j$ into $\mathcal{S}_j$). The resulting set $\mathcal{S}(\Lambda)$ corresponds to an adaptive partition of the space domain $(\Omega_\lambda)_{\lambda \in \mathcal{S}(\Lambda)}$, which can also be obtained by an iterative refinement process: starting from the coarse partition $(\Omega_\lambda)_{\lambda \in \mathcal{S}_0}$, we subdivide a cell $\Omega_\lambda$ of the current partition whenever the detail children of $\lambda$ are in $\Lambda$. We also define the larger set $\mathcal{R}(\Lambda)$, corresponding to all the $\Omega_\lambda$ produced by the adaptive refinement process even at intermediate stages, i.e., all the $\Omega_\lambda$ which are unions of $\Omega_\mu$ with $\mu \in \mathcal{S}(\Lambda)$. One easily checks that

$$\#(\mathcal{S}(\Lambda)) = \#(\Lambda) \leq \#(\mathcal{R}(\Lambda)) \leq 2\#(\Lambda).$$

(48)

We present in Figure 1 an example of a tree $\Lambda$ together with the corresponding adaptive mesh $\mathcal{S}(\Lambda)$ in the univariate dyadic case.

A family of prediction operators $P_j^{j-1}$ being fixed, we shall also be interested in the following elaboration of a tree structure which corresponds to a certain amount of grading in the hybrid mesh.

**Definition 2.2.** A tree $\Lambda$ is graded if for all $\mu \in \Lambda$, the prediction stencil $R_\mu$ is contained in $\mathcal{R}(\Lambda)$.

Note that in the case of the predictions operators (14), a tree is always graded. We present in Figure 2 an example of a graded tree in the univariate dyadic case, with respect to the prediction operator (32), which is the smallest graded tree containing the tree in Figure 1. In this case, the grading property takes the simpler form

$$\quad (j, k) \in \Lambda \Rightarrow (j-1, \lfloor k/2 \rfloor + l) \in \Lambda, \quad l = -1, 0, 1.$$

(49)

The interest of the grading assumption is due to the following result.

**Proposition 2.3.** If $\Lambda$ is a graded tree, there exists an isomorphism $\mathcal{M}_\Lambda$ which maps the cell averages $(u_\lambda)_{\lambda \in \mathcal{S}(\Lambda)}$ of any function $u$ to its detail coefficients $(d_\lambda)_{\lambda \in \Lambda}$.
Both adaptive decomposition and reconstruction operators (i.e., $M_\Lambda$ and $M_\Lambda^{-1}$) can be implemented in $O(\#(\Lambda))$ operations.

**Proof.** Given $(d_\lambda)_{\lambda \in \Lambda}$, we compute $(u_\lambda)_{\lambda \in R(\Lambda)}$ by a coarse-to-fine algorithm. We first remark that for the maximal level $u$ the union of $(\lambda \in \Lambda : |\lambda| > J - 1)$ and $(\lambda \in \Lambda : |\lambda| = J)$ is the union of $(\lambda \in \Lambda : |\lambda| > J - 1)$ and of the projection of $(\lambda \in \Lambda : |\lambda| = J)$. We then compute $(\tilde{u}_\lambda)_{\lambda \in R(\Lambda) \cap \nabla_j}$, and thus the details $(d_\lambda)_{\lambda \in \Lambda \cap \nabla_j}$. The cost of this step is $O(\#(R(\Lambda) \cap S_j))$ operations. The complete decomposition algorithm is thus performed in $O(\#(R(\Lambda))) = O(\#(\Lambda))$ operations.

**Remark 2.4.** For a nongraded tree $\Lambda$, one can still easily prove the following: if $u$ has all its details $d_\lambda = 0$ for $\lambda \notin \Lambda$, then there exists an isomorphism between the cell averages $(u_\lambda)_{\lambda \in S(\Lambda)}$ and the coefficients $(d_\lambda)_{\lambda \in \Lambda}$. However, the complexity of implementing this isomorphism is $O(\#(\tilde{\Lambda}))$, where $\tilde{\Lambda}$ is the smallest graded tree containing $\Lambda$. This is due to the fact that the reconstruction of a $u_\lambda$ always requires the knowledge of the $u_\gamma$ for $\gamma \in R_\lambda$.

In the following, we shall always consider data compression on graded trees, which we obtain by possibly enlarging the set of nonthresholded coefficients: we define $\Lambda_\varepsilon$ to be the smallest graded tree containing the set $\{\lambda : |d_\lambda| \geq \varepsilon|\lambda|\}$, where $\varepsilon$ is given by (44). We define the corresponding tree approximation operator $A_\varepsilon := A_{\Lambda_\varepsilon}$. It is clear that we have

$$
||U_J - A_\varepsilon U_J|| \leq C\varepsilon,
$$

since we are using a larger set of coefficients than with a simple thresholding.

3. **From Harten’s Scheme to Fully Adaptive Schemes**

Both Harten’s multisolution scheme and the fully adaptive scheme that we shall develop to numerically solve (1) rely on a finite volume scheme (6) which operates on the finest resolution level. We rewrite (6) as

$$
V_{j+1}^{n+1} = V_{j+1} - B_{j}^{n+1},
$$

where $V_{j}^{n+1} := (u_\gamma^n)_{\gamma \in S_j}$ is the vector representing the numerical solution at time $n\Delta t$ and $B_{j}^{n+1} := (b_{j}^{n+1})_{\gamma \in S_j}$ with $b_{j}^{n+1} := \frac{\Delta t}{|\Omega_\gamma|} \sum_{\mu \in I(\gamma)} \Gamma_{\gamma,\mu} F_{\gamma,\mu}^{n+1}$ the numerical flux balance for the cell $\Omega_\gamma$. The increment $b_{j}^{n+1}$ depends locally on the numerical solution, i.e., it can be expressed as

$$
b_{j}^{n+1} := \frac{\Delta t}{|\Omega_\gamma|} F(u_\mu^n : \mu \in I(\gamma)),
$$
where \( I(\gamma) \) represents the finite volume stencil attached to \( \gamma \) and \( F \) the numerical flux balance function.

We denote by \( E_j \) the corresponding discrete (nonlinear) evolution operator associated to this standard scheme, so that, assuming that we initialize the scheme on the exact cell average vector \( U^0_j \) of \( u_0 \), we have

\[
V^n_j = E_j V^{n-1}_j = \cdots = E^n_j U^0_j.
\]

If \( U^n_J \) is the cell average vector of the exact solution \( u \) at time \( n \Delta t \), we define the error by

\[
e_n := \| V^n_n - U^n_J \|.
\]

For low order schemes applied to scalar conservation laws, it is usually possible to derive an a priori error estimate \( e_n \leq C n \Delta t^{J/2} \) in 1D and \( e_n \leq C n \Delta t^{J/4} \) in several space dimensions, while in more general situations we can only rely on the practical application of the scheme to evaluate this quantity. Note that the time step is typically limited by a CFL condition

\[
\Delta t \leq C 2^{-J},
\]

where the constant \( C \) typically depends on the supremum of the first order space derivatives of the flux function over the range \( |v| \leq \|u_0\|_{L^\infty} \).

The goal of the multiresolution schemes that we shall now investigate is to compute a numerical solution \( U^n_J \) with a significant CPU gain over the reference finite volume scheme, while the additional error

\[
a_n := \| U^n_J - V^n_n \|
\]

remains within a prescribed accuracy. It is of course natural that this accuracy should be chosen of the same order as the error estimate available for \( e_n \).

3.1. Harten’s heuristics. Both Harten’s multiresolution scheme and our fully adaptive scheme are based on the intuitive idea, introduced in [22], that the set of significant wavelet coefficients of the numerical solution evolves “slowly” from one time step to the other. More precisely, if \( \Lambda^n \) is the graded tree obtained from the application of \( A_\tau \) to some numerical approximation \( U^n_j \) of \( U^n_j \), one can summarize this idea as follows.

**Assumption 3.1** (Harten’s heuristics). One can enlarge \( \Lambda^n \) into a graded tree \( \Lambda^{n+1}_\varepsilon \) which contains both \( \Lambda^n \) and \( \Lambda^{n+1}_\varepsilon \) so that, if \( U^{n+1}_j = E_j U^n_j \), we have

\[
\| U^n_j - A_\varepsilon \Lambda^{n+1}_\varepsilon U^n_j \| \leq C \varepsilon \quad \text{and} \quad \| U^{n+1}_j - A_\varepsilon \Lambda^{n+1} U^{n+1}_j \| \leq C \varepsilon,
\]

i.e., \( \Lambda^{n+1}_\varepsilon \) is adapted for describing the solution at both \( n \Delta t \) and \( (n+1)\Delta t \).

Some comment should be made about the meaning of “enlarge” in the above statement. Clearly this statement is true if we simply define \( \Lambda^{n+1}_\varepsilon \) as the full set \( \nabla^J \), but such a choice has no interest from the point of view of adaptivity: the set \( \Lambda^{n+1}_\varepsilon \) should not be substantially larger than \( \Lambda^n \). The enlargement strategy proposed by Harten in [22] typically consists in “growing” the tree according to the size of the current detail coefficients. More precisely, in the univariate dyadic case, Harten proposes the following rules:

- If \( (j, k) \) is in \( \Lambda^n \), then its neighbors at the same scale \( (j, k+1) \) and \( (j, k-1) \) are included in \( \Lambda^{n+1}_\varepsilon \).
• If \((j, k)\) is in \(\Lambda^n_\varepsilon\) and if \(|d_{j,k}| > 2^N \varepsilon\), where \(N\) is the order of accuracy of the prediction operator (i.e., the number of vanishing moments of the dual wavelets), then its children \((j+1, 2k + k')\), \(k' = 0, 1\), are also included.

The first rule takes into account the finite velocity transport of the solution, assuming that some CFL condition is imposed on the time step. The second rule aims to foresee the formation of discontinuities, using the detail coefficients as numerical smoothness indicators and assuming that the future loss of this numerical smoothness can be detected on the coarse scales.

The above heuristics plays a crucial role in the error analysis of dynamically adaptive multiresolution schemes. We shall devote part of Section 4 to providing a rigorous proof of (57), which will necessitate some changes in the growing rules. In particular, we shall possibly need several levels of refinement when \(|d_{j,k}|\) is very large, and the number of these levels will also depend on the degree of Hölder smoothness of the synthesis wavelets \(\psi_\lambda\).

3.2. Harten’s multiresolution scheme. By (51), we see that Assumption 3.1 also implies

\[
\|B^n_j - A_{\Lambda^n_{\varepsilon}+1} B^n_J\| \leq C \varepsilon,
\]

i.e., the flux balance is also well represented by \(\tilde{\Lambda}_n^{n+1}\). Here we have used the fact that, once \(\Lambda^n_{\varepsilon}+1\) is fixed, \(A_{\Lambda^n_{\varepsilon}+1}\) is a linear operator. Given a prescribed tolerance \(\varepsilon > 0\), the scheme proposed by Harten in [22] consists in using the compressed vector \(A_{\Lambda^n_{\varepsilon}+1} B^n_J\) in place of \(B^n_J\): the cell averages are now evolved according to

\[
U^{n+1}_J = U^n_J - A_{\Lambda^n_{\varepsilon}+1} B^n_J.
\]

Of course, \(B^n_J\) is now the numerical flux balance computed from \(U^n_J\) which differs from \(V^n_J\). As explained in the introduction, the goal of this modification is to save computational cost through a smaller number of numerical flux evaluation. For this purpose, we notice that according to Proposition 2.3, we can reconstruct \(A_{\Lambda^n_{\varepsilon}+1} B^n_J\) from the averages \((b^n_{\lambda})_{\mu \in S(\Lambda^n_{\varepsilon}+1)}\) of \(B^n_J\) on the corresponding adaptive discretization. Such averages are defined from \(B^n_J\) by

\[
b^n_{\lambda} := \sum_{|\gamma| = j, \Omega_{\varepsilon} \subset \Omega_{\lambda}} \frac{|\Omega_{\gamma}|}{|\Omega_{\lambda}|} \int_{\Omega_{\gamma}} b^n_{\gamma},
\]

but in view of the definition of \(b_{\gamma}\) as a numerical flux balance, they can also be defined as the global numerical flux balance for \(\Omega_{\lambda}\), i.e.,

\[
b^n_{\lambda} := \frac{\Delta t}{|\Omega_{\lambda}|} \sum_{|\gamma| = |\mu| = j, \Omega_{\gamma} \subset \Omega_{\lambda}, \Gamma_{\gamma, \mu} \subset \partial \Omega_{\lambda}} \int_{\Gamma_{\gamma, \mu}} F^n_{\gamma, \mu}.
\]

In other words, the computation of \(A_{\Lambda^n_{\varepsilon}+1} B^n_J\) only requires the evaluation of the numerical fluxes \(F^n_{\gamma, \mu}\) corresponding to the edges \(\Gamma_{\gamma, \mu}\) at the finest level which are contained in the edges of the adaptive mesh \(S(\Lambda^n_{\varepsilon}+1)\). Some remarks are in order at this point:

• In the one dimensional case where edges are reduced to points, the number of numerical flux evaluation is directly proportional to \(#(\Lambda^n_{\varepsilon}+1)\). This is no more true in several dimensions, since a coarse cell \(\Omega_{\lambda}\) in the adaptive mesh might have a large number of fine grid edges \(\Gamma_{\gamma, \mu}\) contained in its boundary \(\partial \Omega_{\lambda}\).
The number of flux evaluation is nevertheless substantially reduced since we do not need to compute the fluxes for all the edges $\Gamma_{\gamma\mu}$ which are inside $\Omega_\lambda$.

- Once the $(b_\lambda)_{\mu \in S(\tilde{\Lambda}_n)}$ are computed, the reconstruction of $A_{\tilde{\Lambda}_{n+1}} B_n^J$ on the finest grid is performed by iterating the prediction operator without adding details. In the one dimensional case it can be shown that this is equivalent to computing the remaining fluxes $F_{\gamma\mu}$ by an iterative point value interpolation process derived from the prediction operator (see, e.g., [22]).

- The tolerance parameter $\varepsilon$ monitors the loss of accuracy of the scheme when compared to the standard finite volume scheme which corresponds to $\varepsilon = 0$. In particular, it can be easily shown (see Section 4) that if the reference finite volume scheme is $\ell^1$-contractive, i.e., $\|E f U - E f V\| \leq \|U - V\|$, and under Assumption 3.1, then the additional error $a_n$ can be estimated by $C n \varepsilon$. In principle, raising $\varepsilon$ is beneficial for the computational cost since it reduces $\#(\tilde{\Lambda}_n)$ However, note that this also modifies the numerical solution at the next time step, which could possibly lead to larger sets of coefficients if the new numerical solution is less smooth.

- As already mentioned in the Introduction, the computational savings remains inherently limited by the fact that the evolution by (59) takes place on the finest discretization $S_J$ and therefore the complexity of one step is still $O(N_J)$. In particular, the numerical solution $U^J_n$ is stored at the finest discretization level rather than in a compressed form, which is also a limitation in terms of memory space. Note that the knowledge of $U^J_n$ at the finest level is needed to compute the $b_\lambda$ according to (61).

We shall now turn to the development, analysis, and practical testing of fully adaptive schemes which aim to circumvent these limitations, while preserving the interesting features of Harten’s scheme, such as controlling the accuracy with respect to a reference finite volume scheme.

### 3.3. General structure of the fully adaptive scheme.

The adaptive scheme operates on a compressed representation of the numerical solution $U^J_n$: at time step $n \Delta t$, the nonzero detail coefficients in the multiscale decomposition of $U^J_n$ are confined to a graded tree $\Lambda_n \subset \nabla_J$ so that $U^J_n$ is exactly represented by its coefficients $(d_\lambda^J)_{\lambda \in \Lambda_n}$ or by its cell averages $(u^J_\lambda)_{\lambda \in S(\Lambda_n)}$ on the corresponding adaptive discretization.

Before discussing the derivation of $(U^J_{n+1}, \Lambda_{n+1})$ from $(U^J_n, \Lambda_n)$, a few words are in order concerning the initialization of the scheme: ideally, we would like to start from the exact cell-averages $\overline{u}_J$ of $u_0$ and define $U^J_0$ by a first thresholding step

$$U^J_0 := A_c \overline{u}_J. \tag{62}$$

This first step thus obliges us to consider the full vector $\overline{u}_J$ in order to decompose it and find the set $\Lambda_0$. In practice, it might happen that this is not feasible due to the size of $\overline{u}_J$, and that one needs a more direct access to a compressed representation. This is typically done through some a priori analysis of the initial value $u_0$. In particular, if $u_0$ is provided by an analytic expression or if we have some information on the local size of its derivatives, the estimate (29) can be used to avoid the computation of most details which are below threshold. With such a strategy, we expect to obtain $\Lambda_0$ and $(d^0_\lambda)_{\lambda \in \Lambda_0}$ or $(u^0_\lambda)_{\lambda \in S(\Lambda_0)}$ with a memory and computational cost which remains proportional to $\#(\Lambda_0)$. 

Given $\Lambda_n$ and $U^n_j$ (represented by $(d^n_\lambda)_{\lambda \in \Lambda_n}$ or $(u^n_\lambda)_{\lambda \in S(\Lambda_n)}$), we derive $\Lambda_{n+1}$ and $U^{n+1}_j$ by the following steps:

- **Refinement.** A new set $\tilde{\Lambda}_{n+1}$ containing $\Lambda_n$ is constructed based on the magnitude of the coefficients $|d^n_\lambda|$, $\lambda \in \Lambda_n$, according to some growing rules that we shall describe below. The vector $(d^n_\lambda)_{\lambda \in \Lambda_n}$ is extended by setting $d^n_\lambda = 0$ for $\lambda \in \tilde{\Lambda}_{n+1} \setminus \Lambda_n$. Applying $M^{-1}_{\tilde{\Lambda}_{n+1}}$, we derive the refined averages $(u^{n+1}_\lambda)_{\lambda \in S(\tilde{\Lambda}_{n+1})}$.

- **Computation.** A first numerical solution $\tilde{U}^{n+1}_j$ at time $(n+1)\Delta t$, discretized on $S(\tilde{\Lambda}_{n+1})$, is computed by

$$\tilde{u}^{n+1}_\lambda = u^n_\lambda - \tilde{b}^n_\lambda, \quad \lambda \in S(\tilde{\Lambda}_{n+1}).$$

The adaptive flux balance vector $(\tilde{b}^n_\lambda)_{\lambda \in S(\tilde{\Lambda}_{n+1})}$ is directly computed from $(u^n_\lambda)_{\lambda \in S(\tilde{\Lambda}_{n+1})}$ according to one of the strategies that we shall describe in subsection 3.5.

- **Thresholding.** Applying $M^{\tilde{\Lambda}_{n+1}}$ to $(\tilde{u}^{n+1}_\lambda)_{\lambda \in S(\tilde{\Lambda}_{n+1})}$, we derive $(\tilde{d}^{n+1}_\lambda)_{\lambda \in \tilde{\Lambda}_{n+1}}$. We define $U^{n+1}_j$ by thresholding $\tilde{U}^{n+1}_j$ according to

$$U^{n+1}_j = \varepsilon_{\tilde{\Lambda}} \tilde{U}^{n+1}_j,$$

and the new set $\Lambda_{n+1} \subset \tilde{\Lambda}_{n+1}$ to be the corresponding set $\Lambda_{\varepsilon}$ of preserved coefficients.

In contrast to the multiresolution scheme described in the previous section, all these steps only involve the compressed representation. Two crucial aspects are the refinement rules that define $\tilde{\Lambda}_{n+1}$ and the computation of the adaptive flux balance vector $(\tilde{b}^n_\lambda)_{\lambda \in S(\tilde{\Lambda}_{n+1})}$ without the knowledge of $U^n_j$ at the finest resolution level. We address these issues in the next two subsections.

### 3.4. The refinement process

In order to define $\tilde{\Lambda}_{n+1}$, we introduce several notations. First, assuming that the wavelets $\psi_\lambda$ has $C^r$ Hölder smoothness (in the case where $r$ is an integer we mean by this that the $r - 1$ derivative of $\psi_\lambda$ is Lipschitz continuous) and that the dual wavelets $\tilde{\psi}_\lambda$ have $N$ vanishing moments, we fix some $s > 1$ such that $r < r + 1$. Note that in the classical constructions of wavelets on structured grid, one always has $r < N$, so that we also have $s < N + 1$.

If $\lambda \in \Lambda_n$, we define an index $n(\lambda)$ as the unique integer such that

$$2^n(\lambda) |\varepsilon_{\lambda}| < |d^n_\lambda| \leq 2^{n(\lambda)+1} |\varepsilon_{\lambda}|.$$

Recall that $|\varepsilon_{\lambda}|$ is given by (44). Our growing procedure will take into account the size of $d^n_\lambda$ in the sense that we shall typically perform $\max\{n(\lambda); 0\}$ iterative refinements of the adaptive mesh in the neighborhood of $\Omega_\lambda$. The index $n(\lambda)$ is thus a measure of the “pollution effect” induced in the finer scales (if $n(\lambda) > 0$) at the next time step by $d^n_\lambda$. We also need that $\tilde{\Lambda}_{n+1}$ takes into account a pollution effect in space, which corresponds to the propagation of singularities between time $n\Delta t$ and $(n+1)\Delta t$.

In order to describe this growing procedure in a more precise way, we define $\Sigma_\lambda$ as the union of the cells of $S_j$ which constitutes the support of the discrete wavelet $\psi^j_\lambda$. In other words, $\Sigma_\lambda$ is the region of the finest grid where $U^n_j$ is influenced by the coefficient $d^n_\lambda$. Note that this support is in general contained in (not necessarily equal to) the support of the continuous primal wavelet $\psi_\lambda$ which is obtained by
letting \( J \) go to infinity in the subdivision process. We recall that \( \tilde{\Sigma}_\lambda \) denotes the support of \( \tilde{\psi}_\lambda \), i.e., the space region which influences the coefficient \( d_\lambda \).

In order to take into account the influence of the evolution operator, we use the following notation: if \( \Sigma \) is a set of cells, we define by \( \Sigma^- \) the set of \( \mu \in \Sigma \) such that \( \mu \in I(\gamma) \) for some \( \gamma \in \Sigma \). Recall that \( I(\gamma) \) is the finite volume stencil attached to \( \gamma \). In other words \( \Sigma^- \) is the backward influence domain of \( \Sigma \) for one time step of the reference finite volume scheme. Similarly, we define the forward influence domain \( \Sigma^+ \) of \( \Sigma \) as the set of \( \mu \in \Sigma \) such that \( I(\mu) \cap \Sigma \) is not empty. These influence domains are typically the union of \( \Sigma \) and a layer of fine grid cells, the width of which depends on the size of the finite volume stencil.

For \( \lambda \in \Lambda_n \), we define the influence set of \( \lambda \) by

\[
\Lambda_\lambda := \{ \mu \in \nabla J \text{ s.t. } |\mu| \leq |\lambda| + n(\lambda) \text{ and } \hat{\Sigma}_\mu \cap \Sigma_\lambda \neq \emptyset \},
\]

i.e., all the \( \mu \) of scale level less than \(|\lambda| + n(\lambda)\) such that the detail \( d_\mu \) of \( E_j U_j^\gamma \) is influenced by \( d_\lambda \). Note that the property \( \hat{\Sigma}^-_\mu \cap \Sigma_\lambda \neq \emptyset \) is equivalent to \( \bar{\Sigma}^-_\mu \cap \Sigma^+_\lambda \neq \emptyset \).

We then define \( \tilde{\Lambda}_{n+1} \) by adding to \( \Lambda_n \) all the influence trees:

\[
\tilde{\Lambda}_{n+1} := \Lambda_n \cup \left[ \bigcup_{\lambda \in \Lambda_n} \Lambda_\lambda \right].
\]

In Section 4, we shall see that with such a definition for \( \Lambda_{n+1} \), one can rigorously prove Harten’s heuristics expressed by Assumption 3.1 (in the context of multiresolution decomposition based on structured grids).

**Remark 3.2.** One can easily check that from its definition the influence set \( \Lambda_\lambda \) is necessarily a graded tree. Therefore \( \tilde{\Lambda}_{n+1} \) defined by (66) automatically inherits the structure of a graded tree.

**Remark 3.3.** In view of (65), we shall reduce the number of refinements \( n(\lambda) \) if we take \( s \) as large as possible, with the limitation \( s < r + 1 \). This is slightly different from the heuristic strategy of Harten described in subsection 3.1, which corresponds to taking \( s = N + 1 \) and to limit the refinements to at most one level. It is thus important that the smoothness \( r \) of the primal wavelets is not too small. Recalling that \( r < N \) in all wavelet constructions, we see that our strategy introduces more refinements than Harten’s strategy. As we shall see in Section 5, Harten’s refinement strategy is in practice sufficient to guarantee the needed accuracy of approximation at the next time. However, we shall exhibit particular initial data for which our strategy becomes necessary.

### 3.5. Computing the numerical flux accurately.

When computing the flux balance vector \( (b^\lambda_\lambda)_{\lambda \in S(\tilde{\Lambda}_{n+1})} \), we are facing the difficulty that we do not have at our disposal the representation of \( U_j^\gamma \) by the finest cell averages \( (u^\gamma_\gamma)_{\gamma \in S_j} \), which would allow us to compute the exact vector \( (b^\lambda_\lambda)_{\lambda \in S(\tilde{\Lambda}_{n+1})} \) according to (61). Several strategies are available to cope with this difficulty.

#### 3.5.1. Exact local reconstruction.

This first strategy is based on the remark that according to (61) the computation of the exact vector \( (b^\lambda_\lambda)_{\lambda \in S(\tilde{\Lambda}_{n+1})} \) does not require the knowledge of all the finest cell averages \( (u^\gamma_\gamma)_{\gamma \in S_j} \), but only of those which are involved in the evaluation of the fluxes \( F_{\gamma,\mu} \) for the \( \Gamma_{\gamma,\mu} \) which are part of the edges of the adaptive mesh \( S(\tilde{\Lambda}_{n+1}) \). We denote by \( T(\tilde{\Lambda}_{n+1}) \) the subset of \( S_j \) corresponding to these particular cells.
The idea is then simply to reconstruct \((u^n_\gamma)_\gamma \in \mathcal{T}(\tilde{\Lambda}^{n+1})\) from the available data, i.e., 
\((d^n_\lambda)_\lambda \in \tilde{\Lambda}^{n+1}\) or \((u^n_\lambda)_\lambda \in \Sigma(\tilde{\Lambda}^{n+1})\). Using the local structure of the prediction operator, this can be performed with much less computational cost than the reconstruction of the full \((u^n_\gamma)_\gamma \in \mathcal{S}_f\).

This idea is particularly effective in the univariate case. As an example, consider the univariate dyadic case with the third order accurate prediction (32). Suppose that the adaptive grid \(\mathcal{S}(\Lambda^{n+1})\) coincides with the coarsest grid \([2, 2]\), i.e., \(U^n_J\) is given by its averages \(u^n_{0,k}\) on the intervals \([k, k + 1], k \in \{-2, -1, 0, 1\}\), and that we want to compute the exact numerical flux at the point \(x = 0\). If the numerical flux function in \(E_J\) only depends on a two-cell stencil surrounding the point of interest, then we need to reconstruct the cell averages \(u^n_{J,-1}\) on \([-2^{-J}, 0]\) and \(u^n_{J,0}\) on \([0, 2^{-J}]\). From (32), we see that the cell averages \(u^n_{J,k}, k \in \{-2, -1, 0, 1\}\), can be locally reconstructed by \(J\) iterations of the linear equations

\[
\begin{align*}
    u^n_{J,-2} &= u^n_{J-1,-1} + \frac{1}{8}(u^n_{J-1,-2} - u^n_{J-1,0}), \\
    u^n_{J,-1} &= u^n_{J-1,-1} + \frac{1}{8}(u^n_{J-1,0} - u^n_{J-1,-1}), \\
    u^n_{J,0} &= u^n_{J-1,0} + \frac{1}{8}(u^n_{J-1,-1} - u^n_{J-1,1}), \\
    u^n_{J,1} &= u^n_{J-1,1} + \frac{1}{8}(u^n_{J-1,-1} - u^n_{J-1,-1}),
\end{align*}
\]

i.e., \(J\) applications of a simple \(4 \times 4\) matrix \(M\). More generally, we see that if the adaptive mesh is locally refined up to some level \(j\) in the neighborhood of one of its points \(x\), the computation of the flux at \(x\) will require \(J - j\) applications of \(M\). Therefore, it will be sufficient to store the powers \((M, M^2, \ldots, M^J)\) in order to have direct access to the fine grid cell averages which are needed for the flux computation. In turn, the computational cost of the local reconstruction is optimal with respect to the adaptive discretization since \|#(T(\hat{\Lambda}^{n+1}))\| is of the same order as \|#(\Lambda^{n+1})\| in the one dimensional case.

**Remark 3.4.** The case where the adaptive grid is not locally uniform around \(x\) is not a real problem since the graded tree assumption ensures that at most two adjacent levels \((j, j + 1)\) are represented in the four cells of the adaptive grid surrounding this point. Therefore, by one application of the prediction operator, we immediately recover the four cell averages at the uniform level \(j + 1\) which are needed to initialize the local reconstruction.

**Remark 3.5.** With such a local reconstruction process, our scheme meets the general requirements of adaptivity which were raised in the introduction, namely combining high order approximation in the smooth regions together with mesh refinement near the singularities, even if the reference finite volume scheme is low order accurate. Indeed, high order approximation in the smooth regions is here ensured by the polynomial exactness in the multiresolution prediction operator rather than by the finite volume scheme itself.

As we shall see in Section 4, the error produced by the adaptive scheme with such local reconstructions can be analyzed in a relatively simple way under the assumption that the reference finite volume scheme is \(\ell^1\)-contractive.

In the multivariate case this strategy is still feasible but the computational cost and memory storage are no more optimal since in this case \|#(T(\hat{\Lambda}^{n+1}))\| might be substantially larger than \|#(\Lambda^{n+1})\| (and yet still substantially smaller than \(N_J\)) as noticed in subsection 3.2. A possibility is then to renounce the exact computation of the adaptive flux balance vector and use an approximation \((\tilde{h}^n_\lambda_{\lambda \in \tilde{\Lambda}^{n+1}})\) which
can be constructed from the available data with an optimal computational cost and memory storage. In the following we briefly describe two strategies of that type.

3.5.2. Direct evaluation. An obvious possibility for the computation of \((\tilde{b}^n_{\lambda})_{\lambda \in \mathcal{S}(\tilde{\Lambda}^{n+1})}\) from the \((u^n_{\lambda})_{\lambda \in \mathcal{S}(\tilde{\Lambda}^{n+1})}\) is to apply the numerical flux function of the reference finite volume scheme directly to the values \(u^n_{\lambda}\) in order to directly evaluate the flux on the edges of the adaptive discretization. This approach is very inaccurate in comparison to the previous one in the case where the numerical flux function is low order accurate, e.g., first order, since it generates errors in \(O(H)\) where \(H\) is the mesh size in the coarsest regions of the adaptive grid \(S(\tilde{\Lambda}^{n+1})\).

Therefore, this second strategy can only be successful if a high order scheme, such as an ENO scheme, is applied in the coarse regions. In this context, a rigorous error analysis seems more difficult due to the lack of available results concerning the stability and convergence of high order ENO schemes on uniform grids. In practice, we have observed that such a strategy essentially exhibits the same performances as the exact local reconstruction strategy, with an optimal computational cost and memory storage, proportional to \(\#(\Lambda^{n+1}_{\varepsilon})\).

We end by describing a third hybrid strategy which can be viewed as a combination of the two previous ones. The idea is to associate to each edge \(E\) of \(S(\tilde{\Lambda}^{n+1})\) a fixed finite number of fine grid edges \(\Gamma_{\gamma,\mu}\) contained in \(E\) and to reconstruct exactly the fine scale averages only on the subset \(\tilde{T}(\tilde{\Lambda}^{n+1}_{\varepsilon}) \subset T(\tilde{\Lambda}^{n+1}_{\varepsilon})\) which is needed to evaluate the fluxes \(F_{\gamma,\mu}\) corresponding to these \(\Gamma_{\gamma,\mu}\). The next step consists in evaluating the flux across the edges of \(S(\tilde{\Lambda}^{n+1})\) from these exact values by means of high order quadrature, which can also be thought of as a high order interpolation of the flux along the edges. Since the cardinality of \(\tilde{T}(\tilde{\Lambda}^{n+1}_{\varepsilon})\) is of the same order as \(\#(\Lambda^{n+1}_{\varepsilon})\), the cost of the local reconstruction is optimal even in several dimensions.

4. Error analysis

For the analysis of Harten’s multiresolution scheme and of our fully adaptive scheme, we are interested in controlling the additional error \(a_n\) by some prescribed accuracy. For this, we shall need the following assumption on the reference finite volume scheme.

**Assumption 4.1.** There exists some fixed \(C \geq 0\) such that the reference scheme satisfies

\[
\|E_J U - E_J V\| \leq (1 + C\Delta t)\|U - V\|,
\]

for all \(U, V\).

In the case where \(C = 0\), this assumption means that the scheme is \(\ell^1\)-contractive, a property which is achieved by several first order accurate schemes. Recall that even for such schemes, we expect that the fully adaptive scheme provides a high order decay of the error with respect to the number of degrees of freedom.

4.1. Cumulative error analysis. Let us first consider Harten’s multiresolution scheme as described in subsection 3.2. In this case, following the analysis in [22], we can bound the error with respect to the reference finite volume scheme according to

\[
a_n \leq \|E_J U^{n-1}_J - E_J V^{n-1}_J\| + \|E_J U^{n-1}_J - U^n_J\| \\
\leq (1 + C\Delta t)a_{n-1} + c_n,
\]
where
\begin{equation}
    c_n := \| B_j^{n-1} - A_{\tilde{\Lambda}} B_j^{n-1} \|
\end{equation}
represents the cumulative error at each time step, which can be thought of as a "refinement error" since it measures how well the flux vector \( B_j^{n-1} \) is approximated by the adaptive set \( A_{\tilde{\Lambda}} \) which was refined from \( \Lambda_n^{n-1} \). Under Assumption 3.1, which we shall rigorously prove in subsection 4.3 based on the refinement rule of subsection 3.4, this cumulative error is bounded by \( C\varepsilon \). In the case where the reference scheme is \( \ell^1 \)-contractive, this immediately leads to the estimate
\begin{equation}
    a_n \leq \sum_{k=1}^{n} c_k \leq C n\varepsilon.
\end{equation}

At fixed time \( T = n\Delta t \), this leads to
\begin{equation}
    a_n \leq C \frac{T}{\Delta t}\varepsilon,
\end{equation}
Under the more general Assumption 4.1, we derive
\begin{equation}
    a_n \leq \sum_{k=0}^{n-1} C\varepsilon (1 + C\Delta t)^k = C\varepsilon [(1 + C\Delta t)^n - 1]/\Delta t \leq C\varepsilon \frac{\varepsilon}{\Delta t} [e^{CT} - 1],
\end{equation}
and thus the same estimate as (72) if \( T \) is not large. Note that based on (55), we typically take \( \Delta t \) proportional to \( 2^{-J} \) so that these estimates are in \( O(\frac{T}{\Delta t}\varepsilon) \). This suggests that we take \( \varepsilon \) of order \( 2^{-(1+\gamma)J} \) if the error estimate available between the reference finite volume scheme and the exact solution is in \( CT2^{-J} \).

Let us now turn to the fully adaptive scheme that was proposed in subsection 3.3. We shall only consider here the version of this scheme where the flux is computed by exact local reconstruction as proposed in subsection 3.5.1. In order to compare the fully adaptive scheme with the reference scheme, we shall still evaluate \( a_n := \| U^n_j - V^n_j \| \), where \( U^n_j \) corresponds to the solution produced by the adaptive scheme reconstructed on the finest mesh \( S_j \). One should keep in mind that the adaptive scheme really operates on the compressed representation of \( U^n_j \), i.e., on the \((d^n_{\lambda})_{\lambda \in \Lambda_n^0}\) or \((u^n_{\lambda})_{\lambda \in S(\Lambda_n)}\). For the purpose of error analysis, we may yet summarize the action of the adaptive scheme on \( U^n_j \) by the following simple observation.

**Proposition 4.2.** If \( U^n_j \) is the result of the adaptive scheme based on exact local reconstruction, we have
\begin{equation}
    U^{n+1}_j := A_{\varepsilon} A^{n+1}_{\tilde{\Lambda}} E_j U^n_j,
\end{equation}
where \( A^{n+1}_{\tilde{\Lambda}} := M^{-1} T^{n+1}_{\tilde{\Lambda}} M \) is the linear approximation operator based on discarding the coefficients not in \( \tilde{\Lambda}_{n+1} \) and \( A_{\varepsilon} \) is the tree-structured compression operator.

**Proof.** Since we use the exact numerical flux in the evolution of the averages \((u^n_{\lambda})_{\lambda \in S(\Lambda_n)}\) by (63), this step amounts to transforming \( U^n_j \) into \( \tilde{U}^{n+1}_j := A^{n+1}_{\tilde{\Lambda}} E_j U^n_j \). After the thresholding step, \( U^{n+1}_j \) is thus given by (74).

Using this observation, we can perform a cumulative error analysis of the same type as for Harten’s scheme. We now have
\begin{align*}
a_n & \leq \| E_j U_j^{n-1} - E_j V_j^{n-1} \| + \| E_j U_j^{n-1} - U_j^n \| \\
& \leq (1 + C\Delta t)a_{n-1} + d_n.
\end{align*}
The new cumulative error $d_n := \| E_j U_j^{n-1} - U_j^n \|$ can be further estimated by

$$d_n \leq \| E_j U_j^{n-1} - \tilde{U}_j^n \| + \| \tilde{U}_j^n - U_j^n \| = c_n + t_n,$$

where we have

$$c_n := \| E_j U_j^{n-1} - A_{\hat{\Lambda}_n} E_j U_j^{n-1} \| = \| B_j^{n-1} - A_{\hat{\Lambda}_n} B_j^{n-1} \|,$$

since $A_{\hat{\Lambda}_n} U_j^{n-1} = U_j^n$, and

$$t_n := \| \tilde{U}_j^n - U_j^n \| = \| \tilde{U}_j^n - A_{\varepsilon} \tilde{U}_j^n \|.$$

The term $c_n$ is thus exactly the same refinement error as in Harten’s scheme and, as already explained, will be bounded by $C\varepsilon$ if we apply the refinement rule proposed in subsection 3.4. The additional term $t_n$, corresponding to the thresholding error, is always bounded by $C\varepsilon$ according to (50). Therefore the fully adaptive scheme satisfies exactly the same error bounds as Harten’s scheme, i.e., (72) in the case of an $\ell^1$-contractive reference scheme or (73) for a more general scheme which satisfies Assumption 4.1.

It remains for us to show how the refinement strategy proposed in subsection 3.4 ensures the validity of Harten’s heuristics or equivalently the estimate of the cumulative error $c_n$ by $C\varepsilon$. This is the purpose of the next subsection.

4.2. A rigorous setting for Harten’s heuristic. According to (41), in order to bound $c_n$ by $C\varepsilon$, it suffices to show that for $\mu \notin \hat{\Lambda}_{n+1}$ with $\hat{\Lambda}_{n+1}$ defined by (67) we have the estimate

$$| d_{\mu}(B_j^n) | \leq C\varepsilon |\mu| = C2^d |\mu| \varepsilon,$$

where $d_{\mu}(B_j^n)$ is the detail coefficient of the numerical flux balance vector $B_j^n$ computed from $U_j^n$.

In order to prove this estimate, we shall restrict our discussion to multiresolution representations based on structured finite volume meshes. For the sake of simplicity, we shall present the proof in the univariate dyadic case, i.e., $S_j$ defined by (9). The generalization of the proof to the multivariate structured meshes is straightforward but involves much heavier notation. Much more difficult seems to be the generalization of our results to unstructured meshes. In particular, our way of proof, through Lemma 4.3, 4.4 and 4.5 below, involve three ingredients: (i) the smoothness of the primal wavelets $\psi_{\lambda}$, (ii) finite difference operators of possibly high order, and (iii) the shift invariance of the discrete evolution operator $E_j$. All these ingredients are essentially well defined in the case of structured meshes, and a proper generalization of them to the unstructured case is not clearly available.

In the simple univariate dyadic case, we can identify a vector of fine grid data $U_j = (u_{\gamma})_{\gamma \in S_j}$ to a uniform sampling:

$$u_{\gamma} \sim U_j(k) \text{ for } \gamma = (J,k).$$

We define the finite difference of order $M$ and step $K$ of a vector $U_j$ at the sampling point $k$ on the finest grid by

$$\Delta_K^M U_j(k) = \sum_{m=0}^{M} (-1)^m \binom{M}{m} U_j(k + mK),$$
and the corresponding finite difference stencil as the union of those cells contributing to this quantity, i.e.,

\[(81)\quad S(M, K, k) := \{\gamma = (J, k + mK) ; m = 0, \cdots, M\}.
\]

Recall that the refinement process proposed in subsection 3.4 involves some \( s > 0 \) such that \( s < r + 1 \), assuming that the wavelets \( \tilde{\psi}_\lambda \) have \( C^r \) Hölder smoothness and that the dual wavelets \( \tilde{\psi}^\lambda \) have \( N \) vanishing moments \((N > r)\). We also define \( R \) as the unique integer such that

\[(82)\quad R - 1 < r \leq R.
\]

In particular, we have \( R \leq N \). We can estimate the details \( d_\mu(U_J) \) of a vector \( U_J \) from its finite differences of order \( N \) and step \( 2^{J-|\mu|} \) which stencil is contained in the support \( \tilde{\Sigma}_\mu \) of \( \tilde{\psi}^\mu \) according to the following result.

**Lemma 4.3.** Let \( \mu \in \mathcal{V}_J \) for \( j \in \{1, \cdots, J\} \) and let \( K = 2^{J-|\mu|} \). Then we have

\[(83)\quad |d_\mu(U_J)| \leq C \sup \{|\Delta^N_{K}U_J(k)| ; k \text{ s.t. } S(N, K, k) \subset \tilde{\Sigma}_\mu\}.
\]

**Proof.** The support \( \tilde{\Sigma}_\mu \) is of the form

\[(84)\quad \tilde{\Sigma}_\mu = \{\Omega_\gamma, \gamma = (J, 2^J-|\mu|p + k) ; k = 0, \cdots, 2^J-|\mu|m - 1\},
\]

where \( p \) is an integer that indicates the space position of \( \mu \) and \( m \) is a fixed integer which depends on the size of the prediction stencil. From its definition \( d_\mu \) is a linear combination of cell averages at scale \( 2^{-\mu} \) of the form

\[(85)\quad d_\mu = \sum_{j=0}^{m-1} \rho_j \left( 2^{J-|\mu|} \sum_{0 \leq k < 2^{J-|\mu|} - 1} U_J(2^{J-|\mu|}(p + j) + k) \right).
\]

In the case of a prediction operator of the type (24), the coefficients \( \rho_j \) are independent of \( \mu \). In fact we shall only make use of the fact that \( \sum_j |\rho_j| \) is bounded independently of \( \mu \). We remark that since the dual wavelets have \( N \) vanishing moments, \( d_\mu = 0 \) if \( U_J(k) \) represents the cell averages of some polynomial of degree \( N - 1 \). Equivalently, \( d_\mu = 0 \) if \( U_J(k) = p(k) \) with \( p \in \mathcal{H}_{N-1} \), and the vector \( \rho = (\rho_j)_{j=0,\cdots,m-1} \) is thus orthogonal to the vectors \((j^k)_{j=0,\cdots,m-1}\) for \( k = 0, \cdots, N - 1 \). Therefore, we can express this vector as

\[(86)\quad \rho := \sum_{l=0}^{m-1-N} \sigma_l e_l,
\]

where \((e_l)_{l=0,\cdots,m-1-N}\) is a basis of the orthogonal complement to such vectors. A natural choice is given by defining

\[(87)\quad e_0(j) := \binom{R}{j}(-1)^j, \quad j = 0, \cdots, N, \quad e_0(j) = 0, \quad j > N \text{ or } j < 0,
\]

and taking for \( e_l \) its shifted versions \( e_l = e_0(j - l) \). Therefore a combination of the type \( \sum_{j=0}^{m-1} \rho_j f(j) \) can be rewritten in terms of \( N \)th order differences according to

\[(88)\quad \sum_{j=0}^{m-1} \rho_j f(j) = \sum_{j=0}^{m-1-N} \sigma_j \Delta^N_j f(j),
\]
where $\sum_j |\sigma_j|$ is also bounded independently of $\mu$. Combining (85) and (88) gives
\begin{equation}
|d_\mu| = |\sum_{j=0}^{m-1-N} \sigma_j \left(2^{|\mu|-J} \sum_{0 \leq k < 2^{|\mu|-1}} \Delta_K^N U_j(2^{|\mu|}(p + j) + k)\right)| \leq (\sum_j |\sigma_j|) \sup \{ |\Delta_K^N U_j(k)| : k = 2^{|\mu|} n, \cdots, 2^{|\mu|}(p + mN) - 1\},
\end{equation}
i.e., (83) with $C = \sum_j |\sigma_j|$.

Such a result can be viewed as a discrete counterpart to (29) (one can actually derive (29) from (83) since the finite differences of order $R$ can be controlled by the local H"older smoothness). Note that (83) also holds for all $M \leq N$ in place of $N$, in particular since $R \leq N$, we have
\begin{equation}
|d_\mu(U_j)| \leq C \sup \{ |\Delta_K^R U_j(k)| : k \text{ s.t. } S(R, K, k) \subset \tilde{\Sigma}_\mu \}.
\end{equation}
In view of (90), the estimate (78) on $d_\mu(B_j^p)$ will be ensured if we bound by $C\varepsilon_{|\mu|}$ the differences $\Delta_K^R B_j^p(k)$ for $k$ such that $S(R, K, k) \subset \tilde{\Sigma}_\mu$, $K := 2^{|\mu|}$. Our first step will be to estimate these differences in terms of the differences $\Delta_K^M U_j^p(k)$ for $1 \leq M \leq R$ and $k$ such that $S(M, K, k)$ is contained in the backward influence set $\tilde{\Sigma}_\mu$ of $\Sigma_\mu$. In the univariate dyadic case, $B_j^p(k)$ is related to $U_j^p$ by a formula of the type
\begin{equation}
B_j^p(k) = F(U_j^p(k - a), \cdots, U_j^p(k + b)),
\end{equation}
where $\{k - a, \cdots, k + b\}$ is the computation stencil associated to $k$. We thus need to understand the interplay between the difference operator $\Delta_K^R$ and the numerical flux balance function $F(v_0, \cdots, v_{a+b})$. For this we shall need two additional assumptions on the reference scheme. The first one expresses a control of the sup-norm.

**Assumption 4.4.** There exists some fixed $C \geq 0$ such that the reference scheme satisfies
\begin{equation}
\|E_j V\|_{\ell^\infty} \leq (1 + C\Delta t)\|V\|_{\ell^\infty},
\end{equation}
for all $V$.

In many instances we have $C = 0$, i.e., the sup-norm is diminished by the scheme. More generally, at fixed time $T = n\Delta t$, the above assumption leads to
\begin{equation}
\|V_j^p\|_{\ell^\infty} \leq (1 + CT/n)^n \|V_j^0\|_{\ell^\infty} \leq C(T)\|u_0\|_{L^\infty},
\end{equation}
with $C(T)$ behaving like $e^{CT}$. We can also control the sup-norm $\|U_j^p\|_{\ell^\infty}$ in a similar manner for Harten’s scheme and for the fully adaptive scheme in the following sense: assuming that the estimates (78) hold at the previous time steps $0, 1, \cdots, n - 1$, we have committed at each of these time steps an additional error in the sup-norm which is bounded by $C\varepsilon$ according to (45). Therefore we still have
\begin{equation}
\|U_j^p\|_{\ell^\infty} \leq C(T)\|u_0\|_{L^\infty} + Cn\varepsilon.
\end{equation}
Note that since $\varepsilon$ will be chosen in such a way that $n\varepsilon$ is of the same order as the error estimate $e_n$, this additional term is negligible in comparison to $C(T)\|u_0\|_{L^\infty}$.
In brief, we can assume that $\|U_j^p\|_{\ell^\infty}$ is bounded by a constant depending on $\|u_0\|_{L^\infty}$ and time $T = n\Delta t$ in order to prove the estimate (78) for the next time step. The second assumption concerns the smoothness of the numerical flux balance function $F$. 

\textit{FULLY ADAPTIVE MULTIRESOLUTION FINITE VOLUME SCHEMES 207}
**Assumption 4.5.** The numerical flux function \( F(v_0, \cdots, v_{a+b}) \) is \( C^{R-1} \) with \( D^{R-1}F \) Lipschitz continuous, and its derivatives \( D^kF, \ k = 1, \cdots, R, \) are bounded in the sense that

\[
(95) \quad \sup_{|v_i| \leq C(T)\|u_0\|_{L^\infty}} |D^kF(v_0, \cdots, v_{a+b})| \leq C_k,
\]

where the constant \( C_k \) depends on \( C(T), \) and possibly of \( \|u_0\|_{L^\infty}, \) but not of the finest resolution level \( J. \)

For practically all classical schemes, these assumptions are at least fulfilled at order \( R = 1, \) i.e., \( F \) is Lipschitz continuous and the left hand side of (95) is typically bounded by \( \sup_{|u| \leq \sup|u_0|} |Df| \frac{\Delta x}{\Delta t}, \) which is controlled independently of the time and space step, as well as of \( \|u_0\|_{L^\infty}, \) due to the CFL condition. Only a few numerical schemes for conservation laws have a numerical flux function which has more than \( C^1 \) smoothness. An typical example is the Lax-Friedrich scheme (see, e.g., [26]) in which \( F \) has the same smoothness as \( f. \) For this type of scheme, (95) holds for values of \( k \) which depend on the smoothness of \( f \) itself. Note that in the case of a flux function given by a quadratic polynomial, e.g., Burgers equation, the \( C_k \) then become trivial for \( k > 2. \) With these additional assumptions, we can estimate the \( |\Delta^R_K B^j_\nu(k)| \) for \( k \) such that \( S(R, K, k) \subset \Sigma_\mu \) in terms of the finite differences of \( U^\nu_j \) of lower order, according to the following result.

**Lemma 4.6.** Define \( D_M(U_j, K, \Sigma_\mu) := \sup\{|\Delta^M_K U^\nu_j(k)| : S(M, K, k) \subset \Sigma_\mu\}. \) Then have

\[
(96) \quad D_R(B^\nu_j, K, \Sigma_\mu) \leq C \sup \left\{ \prod_{j=1}^R [D_j(U^\nu_{\nu_j}, K, \Sigma^-_{\nu_j})]^{l_j} ; \right. \\
\left. = (l_1, \cdots, l_R) \in \mathbb{N}^R \text{ s.t. } \sum_j j l_j = R \right\},
\]

where the constant \( C \) depends on the derivative bounds \( C_k \) in (95).

**Proof.** We want to evaluate \( \Delta^R_K B^j_\nu(k) = \Delta^R_K F(U^\nu_j(-a), \cdots, U^\nu_j(+b))(k) \) for any \( k \) such that \( S(R, K, k) \subset \Sigma_\mu \) in terms of the finite differences of \( U^\nu_j \) at lower order. For this purpose we introduce the vector \( P = (p_0, \cdots, p_{a+b}) \) of Lagrange polynomials of degree \( R \) defined by the interpolation conditions

\[
(97) \quad p_i(m) = U^\nu_j(k - a + i + mK), \ m = 0, \cdots, R,
\]

and the function

\[
(98) \quad G(x) := F(P(x)) = F(p_0(x), \cdots, p_{a+b}(x)).
\]

With such a definition, note that we have

\[
(99) \quad \Delta^R_K B^j_\nu(k) = \Delta^R_K G(0),
\]

and therefore

\[
(100) \quad |\Delta^R_K B^j_\nu(k)| \leq C \sup_{x \in [0, R]} |G^{(R)}(x)|.
\]
Using the chain rule for differentiation, we remark that the $R$th derivative $G^R$ has the general expression

$$
G^R(x) = \sum_{k=1}^{R} \sum_{j_1 \geq 1, \ldots, j_k = R} c_{(j_1, \ldots, j_k)} D^k F(P(x))[P^{(j_1)}(x), \ldots, P^{(j_k)}(x)].
$$

Therefore, we obtain that

$$
\sup_{x \in [0, R]} |G^R(x)| \leq C \sup_{x \in [0, R]} \sup \left\{ \prod_{j=1}^{R} |P^{(j)}(x)|^{l_j} ; l_j > 0 \text{ s.t. } \sum_j j l_j = R \right\},
$$

where the constant $C$ depends on the $c_{(j_1, \ldots, j_k)}$ above and on

$$
\sup_{x \in [0, R], k=1, \ldots, R} |D^k F(P(x))|.
$$

With $P(x)$ defined by (97), we have $\sup_{x \in [0, R]} |P(x)| \leq C \| U^J_a \|_{L^\infty}$ and therefore $C$ only depends on the derivative bounds $C_k$ in (95).

We then remark that a polynomial $p$ of degree $R$ always satisfies $p^{(R)}(x) = \Delta^R_x p(0)$ and more generally, for $j \leq R$,

$$
\sup_{x \in [0, R]} |p^{(j)}(x)| \leq C \sup_{m=0, \ldots, R-j} |\Delta^j_x p(m)|.
$$

Combined with (102) and (100) this implies that $|\Delta^R_K B^2_j(k)|$ is bounded by

$$
\sup \left\{ \prod_{j=1}^{R} |\Delta^j_K U^J_a (k - a + i + mK)|^{l_j} ; 0 \leq i \leq a + b, 0 \leq m \leq R - j, l_j > 0 \text{ s.t. } \sum_j j l_j = R \right\},
$$

up to a multiplicative constant $C$ which only depends on the derivative bounds $C_k$ in (95). Taking the supremum over all $k$ such that $S(R, K, k) \subset \Sigma_{\mu}$, we thus obtain (96).

We are thus interested in estimating the right hand side of (96) under the assumption that $\mu \notin \hat{\Lambda}_{n+1}$. For this purpose, we shall use the following result expressing that the finite differences are controlled from the size of the wavelet coefficients.

**Lemma 4.7.** Let $K = 2^{J-|\mu|}$ and let $r$ be such that $\psi_{\lambda} \in C^r$. Then, for $M > 0$ we have

$$
|\Delta^M_K U^J_a (k)| \leq C \sum_{l=0}^{J} 2^{-\min\{M, r\}(|\mu| - l)_+} \times \sup \{|d_\lambda| ; \lambda \text{ s.t. } |\lambda| = l, \Sigma_{\lambda} \cap S(M, K, k) \neq \emptyset\},
$$

where $(|\mu| - l)_+ := \max\{0; |\mu| - l\}$.

**Proof.** We can expand $U^J_a$ in terms of the discrete wavelets $\Psi^J_\lambda$

$$
U^J_a = \sum_{\lambda \in \Psi^J_a} d_\lambda \Psi^J_\lambda,
$$
so that we have
\[
|\Delta^M K U_J(k)| \leq \sum_{\lambda \in \nabla J} |d_\lambda \Delta^M K \Psi^J(\lambda)|. \tag{107}
\]

Note that the $\Delta^M K \Psi^J(\lambda)$ are zero if $\Sigma_\lambda$ and $S(M, K, k)$ are disjoint, i.e., the summation only carries over those $\lambda$ such that $|\Sigma_\lambda \cap S(M, K, k)| \neq 0$. Recall that the discrete wavelets $\Psi^J_\lambda$ are exactly the cell averages of the continuous wavelets $\psi_\lambda$, i.e.,
\[
\Psi^J_\lambda(k) = 2^J \int_{2^{-j}k}^{2^{-j}(k+1)} \psi_\lambda(x) dx. \tag{108}
\]

In particular, at some fixed level $|\lambda| = j$, the $\Psi^J_\lambda$ do not overlap too much in the sense of (46).

A similar property clearly holds with $\Delta^M K \Psi^J_\lambda$ in place of $\Psi^J_\lambda$ so that (107) implies the estimate
\[
|\Delta^M K U_J(k)| \leq \sum_{l=0}^J \sup \{|d_\lambda| \|\Delta^M K \Psi^J_l\|_{L^\infty} ; \lambda \text{ s.t. } |\lambda| = l, \Sigma_\lambda \cap S(M, K, k) \neq \emptyset\}. \tag{109}
\]

Since the primal wavelets are in $C^r$, we have the standard inverse estimates
\[
\|\Delta^M h \psi_\lambda\|_{L^\infty} \leq C\|\psi_\lambda\|_{L^\infty}[\min \{1, h2|\lambda|\}]^{\min\{M, r\}} \tag{110}
\]
(recall that the primal wavelets are normalized in $L^\infty$) and thus, if $|\lambda| = l$ and $h = 2^{-|\mu|}$,
\[
\|\Delta^M h \psi_\lambda\|_{L^\infty} \leq C2^{-\min\{M, r\}(|\mu|-l)_+}. \tag{111}
\]

By averaging this estimate on the cells of $S_J$, we obtain a similar estimate for $\Delta^M K \Psi^J_\lambda$, i.e.,
\[
\|\Delta^M K \Psi^J_l\|_{L^\infty} \leq C2^{-\min\{M, r\}(|\mu|-l)_+}, \tag{112}
\]
if $|\lambda| = l$. Combined with (109), this leads to (105).

Using this lemma, we shall evaluate the differences $|\Delta^M K U^n_J(k)|$ involved in the right hand side of (96), according to the following result.

**Lemma 4.8.** Let $\mu \notin \hat{\Lambda}_{n+1}$, $\hat{\Xi} := 2^{-|\mu|}$, and let $k$ be such that $S(M, K, k) \subset \hat{\Xi}$. If $M \geq R$, we have the estimate
\[
|\Delta^M K U^n_J(k)| \leq C_{|\mu|}, \tag{113}
\]
while if $M < R$, we have the estimate
\[
|\Delta^M K U^n_J(k)| \leq C_{|\mu|}[M/R]. \tag{114}
\]

In the second case (114), the constant $C$ depends on $C(T)$ and $\|u_0\|_{L^\infty}$.

**Proof.** Since $S(M, K, k) \subset \hat{\Xi}$, we can derive from (105) the estimate
\[
|\Delta^M K U^n_J(k)| \leq C \sum_{l=0}^J 2^{-\min\{M, r\}(|\mu|-l)_+} \sup \{|d_\lambda(U^n_J)| ; \lambda \text{ s.t. } |\lambda| = l, \Sigma_\lambda \cap \hat{\Xi}_\mu \neq \emptyset\}. \tag{115}
\]
Note that the above sum is also restricted to the $\lambda$ which belong to $\Lambda_n$ since otherwise $d_\lambda(U^n_j) = 0$. From the definition of $\hat{\Lambda}_{n+1}$, we know that if $\mu \notin \hat{\Lambda}_{n+1}$ and if $\lambda \in \Lambda_n$ is such that $\Sigma_\lambda \cap \hat{\Sigma}_\mu^- \neq \emptyset$, then necessarily $|\mu| > |\lambda| + n(\lambda)$, i.e.,

$$(116) \quad |d_\lambda(U^n_j)| \leq \varepsilon |\lambda| 2^{(|\mu|-|\lambda|)s} = 2^{|\lambda|-J+((|\mu|-|\lambda|)s)} \varepsilon.$$  

We first consider the case where $M \geq R$, i.e., $\min\{M, r\} = r$. Combining (115) and (116) we obtain

$$(117) \quad |\Delta_K U^n_J(k)| \leq C\varepsilon 2^{-J} \sum_{l=0}^{J} 2^{-r(|\mu|-l)+2(|\mu|-l)s+l} = C\varepsilon 2^{-J}[A + B],$$  

with

$$(118) \quad A = 2^{(s-r)|\mu|} \sum_{l=0}^{|\mu|^{-1}} 2^{l(1-s+r)}$$  

and

$$(119) \quad B = 2^{(|\mu|s)} \sum_{l=|\mu|}^{J} 2^{l(1-s)}.$$  

Since $1 < s < r + 1$, both terms get estimated by $C2^{|\mu|}$, which proves (113).

Let us now turn to the case where $M < R$, i.e., $\min\{M, r\} = M$. In this case we use the additional estimate

$$(120) \quad |d_\lambda(U^n_j)| \leq C\|U^n_j\|_{L^\infty}.$$  

If $p > 1$ we can combine this with (116) to obtain the estimate

$$(121) \quad |d_\lambda(U^n_j)|^p \leq C\varepsilon |\lambda| 2^{(|\mu|-|\lambda|)s} = 2^{(|\lambda|-J+((|\mu|-|\lambda|)s)} \varepsilon,$$  

where $C$ is proportional to $[C(T)\|u_0\|_{L^\infty}]^{p-1}$. We set $p := R/M$ and take some $\beta > 0$ such that $1 + \beta < s < 1 + R - \beta$ (which is always feasible since $1 < s < 1 + R$). We then elevate (115) to the power $p$ and use Hölder’s inequality to obtain

$$(122) \quad |\Delta_K U^n_J(k)|^p \leq C \left[ \sum_{l=0}^{J} 2^{-M(|\mu|-l)+} \sup \{|d_\lambda(U^n_j)| ; \lambda \text{ s.t. } |\lambda| = l, \Sigma_\lambda \cap \hat{\Sigma}_\mu^- \neq \emptyset\} \right]^p$$  

$$\leq C \sum_{l=0}^{J} 2^{-R(|\mu|-l)+2^\beta(|\mu|-l)} \sup \{|d_\lambda(U^n_j)|^p ; \lambda \text{ s.t. } |\lambda| = l, \Sigma_\lambda \cap \hat{\Sigma}_\mu^- \neq \emptyset\}$$  

$$\leq C\varepsilon 2^{-J} \sum_{l=0}^{J} 2^{-R(|\mu|-l)+2^\beta(|\mu|-l)} 2^(|\mu|-l)s+l$$  

$$= C\varepsilon 2^{-J}[A + B]$$  

with

$$(123) \quad A = 2^{(s-R+\beta)|\mu|} \sum_{l=0}^{|\mu|^{-1}} 2^{l(1-s+R-\beta)}$$
and

\[ B = 2^{|\mu|}(s-\beta) \sum_{l=|\mu|} J \sum_{l=|\mu|} 2^l(1-s+\beta). \]  

(124)

Since \( 1 + \beta < s < 1 + R - \beta \), both terms get estimated by \( C2^{|\mu|} \), which proves (114).

Clearly, the above result combined with Lemma 4.6 and (90) yields the desired result that we summarize below.

**Theorem 4.9.** If \( \mu \notin \tilde{\Lambda}_{n+1} \), then (78) holds with the constant \( C \) depending on \( C(T) \), \( \|u_0\|_{L^\infty} \) and on the bounds \( C_k \) in (95).

### 5. Numerical tests

One originality and difficulty in the actual implementation of the algorithm presented in subsection 3.3 is the fulfillment of the graded tree property. In order to design an efficient data structure that is well suited for our purposes there are four basic criteria that should be taken into account:

(i) fast random access, e.g., check whether an element already exists;
(ii) fast insert and delete of elements, i.e., avoid copying and sorting of elements within the data structure;
(iii) fast dynamic memory allocation and extension, because the memory requirement changes during the computation; and
(iv) support of group information, i.e., the connection of data corresponding to a common refinement level should be maintained.

The numerical simulations presented in this section have been performed using a C++ code which answers the previous requirements through two main data structures: the *cells* and the *edges*. Each one contains references to parent and children. The *cells* also refer to their neighbors and their *edges* and the *edges* refer to their neighboring *cells*. The recursiveness is another important ingredient in the graded tree algorithm. Other types of data structures (see for instance [35]) can be used based on hash tables instead of trees.

#### 5.1. Tests for 1D scalar equations.

The first set of numerical tests aims to compare different versions of the adaptive algorithm. For this purpose we consider Burgers equation with smooth initial data

\[ \begin{cases} 
\partial_t u + \partial_x u^2/2 = 0, \\
u(x, 0) = 2 + \sin(\pi x).
\end{cases} \]  

(125)

With such an initial data, a shock develops at time \( t = 1/\pi \). The simulations are performed on the \([0, 1]\) interval with periodic boundary conditions. The multiresolution simulations are performed on a hierarchy of eight nested uniform grids \( S_0 \subset S_1 \subset \cdots \subset S_7 \), similar to (9) up to a rescaling: the coarsest level \( j = 0 \) already includes 20 cells. The coarse-to-fine prediction operator is given by (32). The reference finite volume scheme operates on the finest level uniform grid \( S_7 \) which contains \( 20 \times 2^7 = 2560 \) cells.
We consider two types of reference schemes corresponding to two different numerical fluxes: a first order Roe scheme

\[
F_{\lambda,\mu} = \bar{f}(u_\lambda, u_\mu) = \frac{1}{2} \left[ f(u_\lambda) + f(u_\mu) - |\bar{a}(u_\lambda, u_\mu)| (u_\mu - u_\lambda) \right],
\]

and a second order scheme where the input to the previous \( \bar{f} \) function are the point values of an ENO reconstruction (based on the mean values on four neighboring cells), evaluated at the left and right side of the interface point between \( \Omega_\lambda \) and \( \Omega_\mu \).

The adaptive algorithm based on these two reference schemes as described in subsection 3.3, are implemented with two different strategies for the computation of the adaptive flux balance \( \tilde{b}_\lambda^a \) in (63): exact local reconstruction as explained...
in subsection 3.5.1 or direct evaluation applying the above flux functions on the available data as explained in subsection 3.5.2. We thus test four different adaptive schemes.

In order to estimate the error produced by the different algorithms, we compute an “exact solution” at \( t = 0.5 \) with a second order finite volume scheme on a very fine grid of 10240 cells and \( \Delta t = 6.25 \times 10^{-6} \). The performances are summarized in Figure 3.

The performances of the four adaptive schemes are reflected by the four curves which relate the evaluated \( L^1 \) error with the CPU cost. The plotted points on each of these curves correspond to different values of the threshold \( \varepsilon \), from \( 10^{-2} \) down to 0. The horizontal straight line indicates the error for the reference finite volume solution on \( S_7 \), while the vertical line indicates its CPU cost. Several comments can be made:

- Certain points are sitting on the right of the vertical line, in which case the adaptive scheme costs more—in terms of CPU cost—than the uniform reference scheme. They correspond to very small threshold values \( (10^{-5}, 10^{-6}, \text{and } 0) \) for which the over cost due to the bookkeeping and the dynamic memory allocation is not compensated for by adaptivity. The points on the left of the vertical line correspond to computations which are faster than the uniform reference scheme.
- When the fluxes are computed with the first order Roe scheme, the strategy of direct evaluation gives bad results in the sense that the deterioration of the accuracy when raising the threshold is such that the error of the adaptive scheme is several orders of magnitude above the error of the reference scheme whenever some CPU saving is achieved. On the other hand the exact local flux evaluation allows us to preserve the accuracy of the reference scheme with CPU savings up to a factor 20.
- When the fluxes are computed with the second order scheme, both strategies give good results. It appears that the local reconstruction has a cost which is higher if the compression parameter \( \varepsilon \) is large. More precisely, the direct evaluation curve is slightly below the local reconstruction one for a small CPU. This means that one can get a given—but poor—precision faster, using the direct evaluation rather than local reconstruction. In the “safe region” (i.e., when \( \varepsilon \) is small enough so that \( a_n \) is close to \( e_n \)) both algorithms exhibit the same performances. Since the error analysis relies on the local reconstruction step, we will always perform it in the subsequent 1D computations. On the other hand it is reassuring to see that from the practical point of view it is not absolutely necessary. This somehow justifies the use of the direct evaluation in the 2D case where so far we have no easy way of performing local reconstruction.

5.2. Sharpness of the error analysis. Before proceeding further with more numerical tests, we shall use the first set of numerical tests in order to check the reliability of our error analysis. Recall that this analysis was based on estimating the cumulative error \( d_n := \| E_j U_j^{n-1} - U_j^n \| \) by the sum of a thresholding error \( t_n := \| U_j^n - A U_j^n \| \) and a grid refinement error \( e_n := \| B_j^{n-1} - A_{j-1} B_j^{n-1} \| \), both of which are controlled by an estimate in \( O(\varepsilon) \). We can address two questions:
• How sharp is the estimate of $d_n$ at each time step? In particular, do we have that $t_n$ and $c_n$ are indeed behaving like $O(\varepsilon)$, and do these two errors really accumulate in $d_n$?

• How sharp is the estimate of the error $a_n$ between the numerical solutions based on the adaptive and reference scheme? In particular do the errors $(d_n)_{n \geq 0}$ really accumulate as time grows, i.e., is $a_n$ of the same order as $\sum_{k=0}^{n} d_k$?

In order to answer these questions, we have made the exact computation of the quantities $a_n$, $d_n$, $c_n$, and $t_n$ for various values of $\varepsilon$ and $n$ for a fixed number of refinement levels (six levels besides the coarse one). Our observations will also allow us to discuss the refinement strategy.

5.2.1. *Thresholding and prediction error at each time step.* We begin by addressing the first question. Computing $d_n$, $c_n$, and $t_n$, we observed that while these

![Graph showing error estimates $d_n$, $c_n$, and $t_n$ as a function of $\varepsilon$. Harten strategy (top) and our more severe strategy (bottom).](image)
quantities fluctuate very rapidly with time, they keep the same order of magnitude. In order to compare this order of magnitude with \( \varepsilon \), we considered averages of these quantities over 20 time steps around the time of interest \( t^n \):

\[
\bar{d}_n := \frac{1}{20} \sum_{i=n-10}^{n+10} d_i, \quad \bar{c}_n := \frac{1}{20} \sum_{i=n-10}^{n+10} c_i, \quad \text{and} \quad \bar{t}_n := \frac{1}{20} \sum_{i=n-10}^{n+10} t_i.
\]

For the sake of simplicity, we will use the original notation \( a_n \) for \( \bar{a}_n \) from now on. Figure 4 displays the dependence of these quantities with respect to \( \varepsilon \), when using Harten’s refinement strategy and our refinement strategy which guarantees an estimate of \( c \) by \( O(\varepsilon) \), respectively. We choose a time when the shock is completely formed, \( t^n = 0.5 \), reached after \( n = 2000 \) time steps.

Two observations can be made. We first observe that these error terms indeed behave like \( O(\varepsilon) \), and that the thresholding term \( c_n \) and the prediction term \( t_n \) do indeed accumulate in \( d_n \). In this sense our analysis was sharp.

Second, we notice that the error is dominated by the thresholding term \( t \) even when using Harten’s refinement strategy. For this reason, it does not seem worthwhile to apply our refinement strategy, which is too severe and costly, and we have continued to use Harten’s strategy in subsequent numerical tests. We can give an intuitive explanation of why Harten’s strategy is sufficient here. The piecewise smooth structure of the numerical (and exact) solution implies a specific numerical organization of the coefficients \( d^n_\lambda \) in the trees \( \Lambda^n \): these coefficients decrease with scale and are close to the threshold level \( \varepsilon_\lambda \) when \( \lambda \) belongs to the leaves of the tree, i.e., \( \lambda \in \mathcal{L}(\Lambda^n) \). Therefore, at the next time step, large coefficients are not created more than one refinement level above these leaves. In subsection 5.2.3, we shall yet provide an example where this organization is violated and where our refinement strategy becomes strictly necessary to ensure the estimate of the refinement error term by \( O(\varepsilon) \).

5.2.2. Accumulation of the error with time. Next we address the second question, i.e., comparing the error \( a_n \) with the sum \( \tilde{a}_n := \sum_{k=1}^{n} d_k \). For various values of \( \varepsilon \) our observation is essentially the following: \( a_n \) and \( \tilde{a}_n \) are only comparable for

![Figure 5. Error estimate \( a_n \) and cumulative \( \tilde{a}_n = \sum_i d_i \) for \( \varepsilon = 10^{-4} \) as a function of time](image-url)
small $n$, i.e., small time. This is illustrated in Figure 5 which shows that for large $n$, the behavior of $a_n$ is not linear anymore. The unreliability of our estimate for large time is not surprising in this example. Since in any case we expect that the error remains bounded by $2\|u_0\|_{L^\infty}$, the $d_k$ cannot really accumulate when reaching large values of $k$.

5.2.3. A case where Harten’s refinement is not enough. We last want to show that Harten’s refinement strategy can be however “defeated”, yet in rather pathological situations. We consider here an initial data $V_0^J$ such that its multiscale decomposition vector $M_0^J = \mathcal{M}V_0^J$ has all its coordinates equal to zero except for a single index $\lambda$ in the coarsest scale: $d_\lambda = 1$ with $\lambda \in \nabla_0$. The adaptive algorithm will thus be initialized with the singleton $\Lambda_0 = \{\lambda\}$. For small values of $\varepsilon$, i.e., when $|d_\lambda|$ is much larger than $\varepsilon$, both refinement strategies differ for the first time step: our strategy will refine a number of levels proportional to the order of magnitude separating 1 and $\varepsilon$, while Harten’s strategy will refine at most one level.

Figure 6 displays the dependence of the prediction error $c_1$ with respect to $\varepsilon$ for both refinement strategies. We can see that as $\varepsilon$ gets smaller, Harten’s refinement strategy is not sufficient to guarantee an estimate in $O(\varepsilon)$. This example is yet pathological, and after a small number of time steps, we are back in the situation which was encountered in our first test: the organization of the numerically significant multiscale coefficients of the solution is such that Harten’s refinement strategy can be applied without threatening the estimate on $c_n$.

5.3. Tests for 1D systems. The next set of numerical tests consists in applying the adaptive multiresolution to the classical “shock-tube” problem of 1D gas dynamics modeled by the Euler equations with initial piecewise constant data

\[
\begin{align*}
\partial_t \omega + \partial_x f(\omega) &= 0, \\
\omega(x,0) &= \begin{cases} 
\omega_L, & x < 0, \\
\omega_R, & x > 0.
\end{cases}
\end{align*}
\]

(127)
Here
\[ \omega = \begin{pmatrix} \frac{\rho}{m} \\ \frac{m}{E} \end{pmatrix}, \quad f(\omega) = \begin{pmatrix} \frac{\rho u}{m} \\ \frac{\rho u^2 + p}{u(E + p)} \end{pmatrix}, \]
where \( \rho, m, E \) are the density, momentum and energy, respectively. The velocity \( u = m/\rho \) and the pressure \( p \) are related through the equation of state for a polytropic gas,
\[ p = (\gamma - 1)(E - \rho u^2/2), \quad \gamma = 1.4. \]

The reference finite volume scheme uses Roe’s numerical flux along with a minmod type slope limiter interpolation computed independently on each physical component (see [33] for details). We present the performance of the multiresolution scheme on the classical Sod’s benchmark case on the interval \([-1, 1]:\]
\[ \omega_L = \begin{pmatrix} 1 \\ 0 \\ 2.5 \end{pmatrix}, \quad \omega_R = \begin{pmatrix} 0.125 \\ 0 \\ .25 \end{pmatrix}. \]
The coarsest grid \( S_0 \) consists in 200 cells. The reference finite volume solution is computed on \( S_4 \) which has 12800 cells. The CPU cost for this solution is \( t_{\text{cpu}} = 6647 \) seconds. Here we apply the adaptive algorithm only with the exact local reconstruction strategy, and compute the \( L^1 \) error between the adaptive solution and the reference solution. We also study the effect of limiting the maximal scale level to \( J = 4 \) or \( J = 5 \) in the adaptive algorithm. The time step for these simulations depends on the finest grid step size \( \Delta x = \frac{\Delta t}{\text{CPU}} \) and on the solution itself. It is computed on the reference solution so that at each time iteration it ensures the CFL condition
\[ \Delta t < N_{\text{CFL}} \frac{\Delta x}{\max_{i=1,2,3}(|\lambda_i|)} \]
where the \( \lambda_i \) for \( i = 1, 2, 3 \) are the three eigenvalues of the Jacobian matrix of the flux function. In our case, \( \max_{i=1,2,3}(|\lambda_i|) = |c| + |u| \) with
\[ c = \sqrt{(\gamma - 1)(e + p)/\rho - \frac{1}{2}u^2}. \]
We set \( N_{\text{CFL}} = 0.3 \), which leads to \( \Delta t = 4.10^{-5} \) for \( J = 7 \). In this case 6500 time iterations are necessary to reach \( t = 0.26 \). For \( J = 5 \) and \( J = 4 \) we need, respectively, 3250 and 1625 iterations. In each case, we try several values for the threshold parameter \( \varepsilon \). In Figure 7, we show the density at time \( t = 0.26 \) computed with the reference scheme on \( S_6 \), along with the adaptive solution with maximal level \( J = 4 \) and threshold \( \varepsilon = 10^{-3} \) on the top and \( \varepsilon = 10^{-4} \) on the bottom. The adaptive solution is represented on the adaptive grid \( S(A^n) \) as a step function (we did not plot its reconstruction on the finest level since it cannot be visually distinguished from the reference solution). In both figures we also show the varying depth of the multiresolution analysis which reads on the bottom \( y \)-axis. In order to understand the effectiveness of adaptivity, it is natural to compare these solutions with the one obtained by application of the reference scheme on \( S_4 \), i.e., taking \( \varepsilon = 0 \), in terms of CPU saving and of the \( L^1 \) error with respect to the uniform solution on \( S_6 \) (viewed here as the exact solution). For this solution on \( S_4 \) the CPU
time is 693 seconds and the $L^1$ error is 0.00151. For $\varepsilon = 10^{-4}$, the error is 0.00158 which is barely above the previous one, while the CPU time drops down to 121 seconds. For $\varepsilon = 10^{-3}$, the error is 0.00297 while the CPU time drops down to 86 seconds. In this last case, the finest level of resolution $J = 4$ is very seldom used in the adaptive mesh—in fact exclusively at the four locations of highest gradient for the density $\rho$. This induces some slight oscillation near the contact discontinuity, around $x = 0.25$. For $\varepsilon = 10^{-4}$, the oscillations near the discontinuities have disappeared. We summarize all the tests in Figure 8: for each of the three curves corresponding to the maximal levels $J = 4, 5,$ and $6$, we plot the error with the reference solution versus the CPU and memory requirement. A vertical straight line indicates the CPU and memory required to compute the reference solution. Similar to the experiment on Burgers equation, we observe that these curves exhibit a flat region corresponding to the range of $\varepsilon$ that ensures an error of the same order as the error of the uniform scheme on $S_J$, i.e., taking $\varepsilon = 0$. It is of interest to choose the largest threshold in this range in order to minimize the CPU time as well as the memory. We also see that raising $\varepsilon$ above this range has a dramatic effect on
the error while not reducing much more the CPU and memory since the left part of the curves are almost vertical.

5.4. Tests in 2D. We end with some numerical tests in 2D based on hierarchical triangular meshes. Here we use the prediction operator introduced in [13] and the direct evaluation strategy for the adaptive algorithm. Some first numerical tests for a linear advection problem have been presented in [14] together with a detailed description of the algorithm. Here, we consider the 2D version of Burgers equation

\[
\partial_t u + \frac{1}{2} \partial_x u^2 + \frac{1}{2} \partial_y u^2 = 0
\]

with initial condition

\[
u_0(x, y) = 0.3 + 0.7 \sin \left( \frac{\pi}{2} (x + y) \right)
\]

and periodic boundary conditions. The numerical flux in that case is

\[
F_{\lambda, \mu} = \frac{1}{2} (u_\mu (au_\mu)^+ - u_\lambda (-au_\lambda)^+),
\]

where \(a = \mathbf{n} \cdot (1, 1)^t\), with \(\mathbf{n}\) the conveniently oriented normal.

The computing domain is the square \([-1, 1]^2\) meshed at the coarsest level by 800 triangles, and we use five levels of multiresolution. The occupancy of the different
levels varies with time, and depends of course on the thresholding parameter. Because the initial condition is very smooth, the finest level of resolution is hardly used at time $t = 0$, except for very small thresholding $\varepsilon = 0.001$ and of course for the finite volume reference solution. At final time $t = 1$, the shock has developed and finer triangles are created in this region. This is clear in Figure 9, which represents the two hybrid grids at initial and final time for $\varepsilon = 0.02$. For clarity of the figure, only the first 3 levels of resolution are represented. Table 1 compares the performances of the adaptive and reference finite volume scheme for different values of the threshold. For each value of $\varepsilon$, the solution at final time is reconstructed on the finest level to allow the computation of the error with the reference solution. As expected, the error increases with the threshold value with the predicted $O(\varepsilon)$ behavior. The CPU times as well as the number of triangles on the initial and final hybrid grids are also listed.

It is interesting to note that for a large threshold, the number of triangles has increased at final time, while for a small threshold this feature is reversed. This can
Table 1. Performances of the adaptive scheme for 2D Burgers equation

<table>
<thead>
<tr>
<th>ε</th>
<th>error</th>
<th>CPU</th>
<th># tri. t = 0</th>
<th># tri. t = 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.02</td>
<td>0.0176698</td>
<td>1636</td>
<td>3680</td>
<td>13162</td>
</tr>
<tr>
<td>0.01</td>
<td>0.00668124</td>
<td>2401</td>
<td>11040</td>
<td>14506</td>
</tr>
<tr>
<td>0.005</td>
<td>0.00468717</td>
<td>2728</td>
<td>13600</td>
<td>15209</td>
</tr>
<tr>
<td>0.001</td>
<td>0.00114192</td>
<td>6078</td>
<td>51360</td>
<td>20840</td>
</tr>
<tr>
<td>0.0005</td>
<td>0.000800985</td>
<td>8207</td>
<td>59040</td>
<td>25451</td>
</tr>
<tr>
<td>FV</td>
<td>0</td>
<td>10098</td>
<td>204800</td>
<td>204800</td>
</tr>
</tbody>
</table>

be explained as follows. At time \( t = 1 \), the shock is completely formed and the exact solution of (128)–(129) is almost piecewise affine. Since the prediction operator is exact for affine functions, all wavelet coefficients are either almost zero away from the shock or large near the shock. Therefore, as we lower the threshold, the number of triangles at final time \( N(\varepsilon, 1) \) grows slower than the number of triangles at initial time \( N(\varepsilon, 0) \). For \( \varepsilon \leq 10^{-3} \), \( N(\varepsilon, 1) \) becomes smaller than \( N(\varepsilon, 0) \).

6. Conclusions and perspectives

The adaptive multiresolution scheme we have developed allows on several benchmark tests some substantial memory and CPU savings while remaining within the same accuracy as the reference finite volume scheme. In the case of multiresolution based on a structured grid, the underlying wavelet framework provides the appropriate tools for a rigorous error analysis of this scheme. We end by listing a number of remaining trends which are currently under investigation.

**Trend 1: Multiresolution in several dimensions.** There exist many ways of building finite volume multiresolution and wavelets in several dimensions. However, many restrictions appear if one wishes to guarantee the accuracy, stability, and smoothness properties in the sense described in subsection 2.3. In the particular case of a uniform rectangular discretization, one inherits these properties from a given univariate finite volume multiresolution by means of tensor product strategies. This approach was recently extended in [16] to curvilinear discretization obtained from uniform meshes by smooth parametric maps. In this context, particular adaptations of the prediction operator (based on stable completion techniques introduced in [6]) are needed to preserve accuracy. In the case of uniform triangular discretization, finite volume multiresolution with accuracy, stability, and smoothness properties has been obtained in [13]. In the case of unstructured meshes much less is known: while finite volume multiresolution can be designed with accuracy properties (see [1]), a general strategy to achieve stability and smoothness properties is still an open question.

**Trend 2: Nonlinear multiresolution.** As we pointed out in Section 2, the prediction operator need not be linear. The idea of allowing a nonlinear prediction operator was inspired by the ENO schemes of [23] which involve a data dependent selection of the computation stencil in order to avoid spurious oscillations in the neighborhood of shocks while preserving high order accuracy. Similarly, a nonlinear multiresolution will use a data dependent selection of the prediction stencil. One of the objectives of this approach is to improve on the accuracy and compression properties by avoiding the presence of singularities within the prediction stencil. Many open problems remain concerning the stability of such decompositions which
can no longer be thought of as a change of basis. We refer to [2] for an introduction to nonlinear multiresolution and to error control algorithms, which cope with the possible lack of stability of such representations.

**Trend 3: Implicit schemes.** The adaptive multiresolution techniques that we have developed in this paper should be adaptable to implicit time discretization which are more appropriate for a certain class of problems, in particular convergence to a steady state solution. In addition, if one applies linearization techniques on the implicit part according to

\[ U_j^{n+1} = U_j^n + F(U_j^{n+1}) \approx U_j^n + F(U_j^n) + DF(U_j^n)[U_j^{n+1} - U_j^n], \]

the multiresolution representation might turn out to be useful to precondition the system

\[ (I - DF(U_j^n))U_j^{n+1} = U_j^n + F(U_j^n) - DF(U_j^n)U_j^n, \]

which needs to be solved at each time step. The adaptive multiscale concept has recently been incorporated to an implicit solver for the numerical simulation of steady state problems arising in aviation (see [36]).

**Trend 4: Super-resolution and time-space adaptivity.** The accuracy of the adaptive scheme we have developed is inherently limited by the finest resolution level \( J \) which is supposed to be fixed. In an adaptive context, it is tempting to remove this constraint in order to improve accuracy by additional refinements near the singularities. If such refinements are only local, this should not affect the memory cost as well as the complexity of one time step. However, this would clearly affect the overall computational cost since the time step is tied to the finest discretization level according to the CFL condition. A natural way to cope with this difficulty should be by introducing time-space adaptivity, i.e., using local time steps which depend on the level of refinement. In this perspective, the property of graded trees which we have imposed should play a crucial role, since it imposes a slow variation of the time step (at most by a factor two) between two adjacent cells on the adaptive mesh.

**Trend 5: Optimality analysis.** Nonlinear approximation theory provides a natural benchmark for adaptive schemes. In the context of multiresolution or wavelet schemes, this is expressed in the following way: if we knew the exact solution \( u(x) \) at some time \( T \), a nearly optimal adaptive approximation by \( N \) wavelet coefficients in a prescribed error norm \( \| \cdot \| \) could be obtained by its truncated expansion

\[ u_N = \sum_{\lambda \in E_N(u)} d_\lambda \psi_\lambda, \quad d_\lambda := \langle u, \tilde{\psi}_\lambda \rangle, \]

where \( E_N(u) \) is the set of indices corresponding to the \( N \) largest \( \|d_\lambda \psi_\lambda\| \) (see [18] for this type of result). An optimal adaptive scheme should thus produce approximate solutions \( \tilde{u}_N \) (where the number \( N \) of nonzero wavelet coefficients depends on the choice of the tolerance \( \varepsilon \)) such that \( \|u - \tilde{u}_N\| \) behaves like \( \|u - u_N\| \) as \( N \) grows. Some optimality results of this type have recently been obtained in [11] for adaptive wavelet schemes in the context of linear elliptic problems. In the case of our algorithm, we expect to be further from this benchmark due to the accumulation of error with time, and the inherent limitation of the accuracy by the finest level.
REFERENCES


Lauratoire d’Analyse Numérique, Université Pierre et Marie Curie, BC 187, 75252 Paris cedex 05, France

E-mail address: cohen@ann.jussieu.fr

Laboratoire d’Analyse Numérique, Université Pierre et Marie Curie, BC 187, 75252 Paris cedex 05, France

E-mail address: kaber@ann.jussieu.fr

Institut für Geometrie und Praktische Mathematik, RWTH, Templergraben 55, D-52056 Aachen, Germany

E-mail address: mueller@igpm.rwth-aachen.de

Laboratoire d’Analyse Numérique, Université Pierre et Marie Curie, BC 187, 75252 Paris cedex 05, France

E-mail address: postel@ann.jussieu.fr