LOCAL TIME STEPPING APPLIED TO IMPLICIT-EXPLICIT METHODS FOR HYPERBOLIC SYSTEMS

FRÉDÉRIC COQUEL†, QUANG LONG NGUYEN‡, MARIE POSTEL†, AND QUANG HUY TRAN†

Abstract. In the context of systems of nonlinear conservation laws it can be crucial to use adaptive grids in order to correctly simulate the singularities of the solution over long time ranges while keeping the computing time within acceptable bounds. The adaptive space grid must vary in time according to the local smoothness of the solution. More sophisticated and recent methods also adapt the time-step locally to the space discretization according to the stability condition. We present here such a method designed for an explicit-implicit Lagrange-projection scheme, addressing physical problems where slow kinematic waves coexist with fast acoustic ones. Numerical simulations are presented to validate the algorithms in terms of robustness and efficiency.

Key words. hyperbolic systems of conservation laws, time-varying adaptive grid, local time stepping, implicit-explicit schemes

AMS subject classifications. 35L65, 65M06, 76M25

DOI. 10.1137/070708500

1. Introduction. Our scope of interest is the accurate numerical simulation of highly nonlinear conservative laws. In our context of hydrocarbons transportation, the closure laws used by the oilfield engineers are very "sophisticated" functions: they must match experimental data and take into account a large range of operating conditions. They turn out in practice to be piecewise defined and even tabulated. The solutions of such systems of PDEs exhibit localized and moving singularities which require costly numerical schemes. At the same time realistic applications often require simulation of the phenomenon on a very long time range. As developed in [1], we are interested in multiphase flows applications presenting the particularity of two scale wavelengths: a slow kinematic one which is the phenomenon of interest for the engineers, and fast acoustic waves arising from the highly nonlinear physics. The latter will induce very small time-steps when treated explicitly. This discrepancy between kinematic and acoustic speeds can be advantageously dealt with in a Lagrange projection formulation, as detailed in [9,12]. This allows us to resolve the acoustic part of the solution with an implicit scheme, therefore relaxing the time-step bound. The kinematic components are solved explicitly, ensuring in turn the required precision.

This setup is a very good candidate for adaptive methods, such as the one introduced by Harten in the nineties [18], which was specially designed for the speed up of finite volume schemes. The gist of the method consists of analyzing the smoothness of the solution by performing a wavelet-like transformation. At the discrete level this relies on a dyadic hierarchy of grids where a multiscale transformation of the solution discretized by its mean values is performed at each time-step. The size of
the coefficients in the multiscale basis is tested against a threshold parameter. Some smoothness property imposed on the underlying biorthogonal wavelet ensures that the size of the coefficients of the solution in the wavelet basis is inversely proportional to the local smoothness of the solution (see [6]). This information is used to design an adaptive grid by locally selecting the coarsest level of discretization beyond which the coefficients are negligible. Of course the adaptive grid must evolve at each time-step in order to follow the displacement of the singularities of the solution, and this evolution strongly relies on the hyperbolic nature of the equations that ensures a finite speed of propagation. The time-step is monitored by the smallest space grid step and the CFL-like stability condition. Design and study of fully adaptive schemes can be found in [6, 23]. Use of the multiresolution technique for semi-implicit schemes is described in [13] and [1] and is also implemented in the Lagrangian case in [8, 9]. Simulations exhibit that it can significantly speed up the computation.

The goal of the present paper is to describe a further enhancement to the multiresolution technique: the local time stepping (LTS). Numerical schemes designed on nonuniform—moving or fixed—grids can be sped up by using different time-steps in different areas of the grid, according to the local mesh size. First developments in this direction can be found in [26] for fixed nonuniform grid just before the development of the adaptive mesh refinement technique in [3, 4]. In the context of the multiresolution technique previously described, it has been developed and implemented for one-dimensional scalar conservation laws in [24] and extended to the two-dimensional shallow water equations in [21], with promising CPU time gains. A macro time-step, fit for the coarsest level of discretization, is subdivided into intermediate time-steps whose duration is linked to the finest discretization. For a $K$-level dyadic grid hierarchy there are $2^K$ intermediate time-steps. At a given intermediate time-step, the solution is updated only on cells belonging to the current synchronization level or finer. Transition zones are defined in order to deal with the synchronization of the solution at the border between consecutive levels of discretization. Straightforward application of this method to our system of PDEs in the explicit scheme case was first presented in [8]. It is now extensively tested for robustness and performances in this paper.

We also endeavor here to adapt the LTS idea to our semi-implicit scheme, where two wave speeds interplay in the sense that fast acoustic waves are treated implicitly with a CFL condition number much larger than one, and kinematic waves are treated explicitly, with a time-step satisfying a standard CFL condition with a CFL number less than one. The resulting scheme departs from the original one in [24] and will be described in detail. The implicit version of the LTS algorithm presented in [24] or an alternative to it described by the same authors in [25] cannot be readily applied to our problem, where we have to deal with transient fast acoustic waves. In particular, the prediction of the adaptive grid from one intermediate time-step to the next needs to take into account the fast wave speed. Another originality of our setup is the splitting of our operator into a Lagrangian step and an Euler projection step, which requires some extra synchronization during the time integration.

We first present the model equations and the numerical algorithms on a fixed nonuniform grid. The next section summarizes the multiresolution setup and its application to a finite volume scheme, as described at length in [1, 13]. Section 4 is devoted to the LTS enhancement in the case of the explicit scheme. This part makes full use of the notations established in [24]. The semi-implicit scheme, which requires a new approach, is treated at last in section 5. Numerical simulations are described in sections 4.3 and 5.2. They allow us to compare the efficiency of the different schemes with and without LTS, in terms of accuracy and computing time.
2. Model and schemes on a uniform grid.

2.1. Physical problem in Eulerian coordinates. Over the state space

\[ \Omega_V = \{ V = (\rho, \rho Y, \rho u) \in \mathbb{R}^3, \; \rho > 0, \; Y \in [0, 1], \; u \in \mathbb{R} \} , \]

we consider the problem of finding a function of time and space \((t, x) \in \mathbb{R}^+ \times \mathbb{R} \to V(t, x) \in \Omega_V\) that solves the Euler-like model

\[
\begin{align*}
\partial_t (\rho) + \partial_x (\rho u) &= 0, \\
\partial_t (\rho Y) + \partial_x (\rho Y u) &= 0, \\
\partial_t (\rho u) + \partial_x (\rho u^2 + P) &= 0.
\end{align*}
\]

Here, \(\rho\) is the density of the mixture, \(u\) is the velocity, and \(Y\) is the gas mass fraction. The thermodynamical pressure \(P\) is given as an algebraic closure law \(P(\rho, Y)\). In the context of hydrocarbons transportation, the closure laws used by the oilfield engineers are very “sophisticated” functions: they must match experimental data and take into account a large range of operating conditions. They turn out in practice to be piecewise defined, in many instances even tabulated, and therefore highly expensive to evaluate.

The three PDEs of (2.2) express the conservation laws associated with, respectively, the mixture density, the gas density, and the total momentum. This system is a particular yet prominent case of the so-called drift-flux models for two-phase flows in pipelines [1,2,13]. It can be put under the abstract form

\[\partial_t V + \partial_x (Vu(V) + P(V)) = 0,\]

with the vector notation

\[P(V) = (0, 0, P).\]

It is known [17] that under the assumption

\[\partial_y P(\rho, Y) |_{Y} > 0,\]

system (2.3) is strictly hyperbolic, i.e., the Jacobian matrix \(\nabla_V P(V)\) corresponding to the flux \(P(V) = Vu(V) + P(V)\) has three distinct real eigenvalues. In increasing order, these eigenvalues are

\[u - c < u < u + c,\]

where \(c\) denotes the local sound speed

\[c = \sqrt{\frac{\partial_y P(\rho, Y) |_{Y}}{\partial_y \rho}}.\]

From the physics standpoint, the eigenvalues (2.6) represent speeds of wave propagation. In the context of oil transportation data, a subsonic regime is most frequently observed for which

\[|u| \ll c.\]

In other words, there are two scales of wave velocities with distinctive orders of magnitude: the fast acoustic waves \(u \pm c\) and the slow kinematic wave \(u\). Only the slow kinematic wave is worth being solved explicitly and accurately, since it corresponds to the actual transportation of matter. The fast acoustic waves do not have any interest for the engineers.
2.2. Physical problem in the arbitrary Lagrange–Euler formalism. This special feature of the physical problem must be kept in mind if we are to design an appropriate numerical scheme. In order to save the CPU time of simulations, we look for numerical schemes that solve the acoustic part of the solution in an implicit way (thus allowing for large time-steps), while computing the kinematic part of the solution in an explicit way (thus preserving accuracy on the slow wave). This splitting between slow and fast components, respectively handled by an explicit and implicit scheme, is a well-established procedure in compressible flows literature [7,27]. In the context of multiphase flows such a hybrid implicit-explicit scheme was attempted by Masella, Faille, and Gallouët [22], Faille and Heintzé [16], and Evje and Flatten [15] in the framework of VFRoe methods, then reused by Baudin, Coquel, and Tran [2] in the framework of relaxation methods.

All of the previous schemes are built directly in Eulerian coordinates. While this approach is natural, it does not ensure that the density and the mass fraction remain positive in the implicit-explicit setting. In a recent work [12], we constructed a new implicit-explicit scheme that allows us to guarantee positivity for density and the mass fraction. This new approach relies on considering the physical problem in the arbitrary Lagrange–Euler formalism [14,19]. Before explaining how to combine this new scheme with LTS, let us revisit its core ideas.

We introduce a new referential frame, in which the coordinates are denoted by $\chi$. This frame is neither the material (Lagrangian) configuration nor the laboratory (Eulerian) configuration $x$. Instead, it moves at the imposed speed $u-w$ with respect to the laboratory. Then, the velocity of the particles with respect to the moving frame, as seen from the laboratory, is equal to $w$. Let $x = x(\chi,t)$ be the correspondence between the moving frame and the laboratory frame, and let

$$J = \partial_\chi x|t$$

be the dilatation rate. Then, it is straightforward to prove [14,19] that system (2.3) is equivalent to

\begin{align}
\partial_t (J) + \partial_\chi (w) - \partial_\chi (u) &= 0, \\
\partial_t (\nabla J) + \partial_\chi (\nabla w) + \partial_\chi (\nabla P) &= 0.
\end{align}

The system above highlights the role of the displacement $w$, which is for the moment arbitrary. We are going to take advantage of it in order to design a splitting strategy.

2.3. Semidiscrete scheme. Let $\Delta t > 0$ denote the time-step. Each iteration from time $t^n$ to time $t^{n+1} = t^n + \Delta t$ is split into two steps, as shown by the diagram

$$n \rightarrow n^s \rightarrow n + 1.$$

1. The Lagrange step. Starting from $J^n = 1$, we solve

\begin{align}
\partial_t (J) - \partial_\chi (u) &= 0, \\
\partial_t (\nabla J) + \partial_\chi (\nabla u) &= 0.
\end{align}

By doing so, we have momentarily ignored the convection due to $w$ in (2.10).

2. The projection step. Starting from the intermediate state $n^s$, we set $w = u$ and solve

\begin{align}
\partial_t (J) + \partial_\chi (u) &= 0, \\
\partial_t (\nabla J) + \partial_\chi (\nabla u) &= 0.
\end{align}
By doing so, we account for the motion of the particles. A judicious choice of \( w = u \) at the discrete level leads us to \( J^{n+1} = 1 \) at the end of the step.

### 2.4. Fully discrete scheme.

To make things clearer, let us go through the update formulae at the fully discrete level. The space domain is decomposed into cells, labeled by subscripts \( j \in \mathbb{Z} \). The edges of the cells are referred to by subscripts \( j + 1/2 \). The size of cell \( j \) is \( \Delta x_j \).

1. **The Lagrange step.** Starting from \( J^n_j = 1 \), we approximate system (2.12) by

\[
\begin{align*}
J^{n+1}_j - J^n_j &= \tilde{u}_{j+1/2} - \tilde{u}_{j-1/2}, \\
(\nabla J)^{n+1}_j - (\nabla J)^n_j &= \tilde{P}_{j+1/2} - \tilde{P}_{j-1/2}.
\end{align*}
\]

(2.14a) \quad (2.14b)

Here we point out that although the practical interest lies in the implicit treatment of the Lagrange step, the explicit version allows the almost straightforward implementation of an LTS strategy. Both fully explicit and implicit-explicit schemes will therefore be studied and tested. More specifically, we have for the explicit scheme

\[
\tilde{u}_{j+1/2} = \tilde{u}(\nabla^n_j, \nabla^n_{j+1}) \quad \text{and} \quad \tilde{P}_{j+1/2} = \tilde{P}(\nabla^n_j, \nabla^n_{j+1}),
\]

and for the implicit scheme

\[
\tilde{u}_{j+1/2} = \tilde{u}(\nabla^{n+1}_j, \nabla^n_{j+1}) \quad \text{and} \quad \tilde{P}_{j+1/2} = \tilde{P}(\nabla^{n+1}_j, \nabla^n_{j+1}).
\]

(2.15) \quad (2.16)

Of course, it has to be understood that

\[
\tilde{P}(\nabla_j, \nabla_{j+1}) = (0, 0, \tilde{P}(\nabla_j, \nabla_{j+1})).
\]

The two-variable functions \( \tilde{u}(., .) \) and \( \tilde{P}(., .) \) are specific to the acoustic solver actually used. For example, the relaxation solver proposed in [12] reads

\[
\begin{align*}
\tilde{u}(\nabla_j, \nabla_{j+1}) &= \frac{u_j + u_{j+1}}{2} - \frac{P_{j+1} - P_j}{2a}, \\
\tilde{P}(\nabla_j, \nabla_{j+1}) &= \frac{P_j + P_{j+1}}{2} - \frac{u_{j+1} - u_j}{2},
\end{align*}
\]

(2.18a) \quad (2.18b)

where \( a > 0 \) denotes some frozen Lagrangian sound speed to be properly defined. While other functions are possible for \( \tilde{u} \) and \( \tilde{P} \), the advantage of the solver (2.18) lies in the possibility of guaranteeing positivity for \( a \) large enough [12]. We do not go into further details, insofar as only the generic structure (2.14)–(2.16) will be used in what follows.

2. **The projection step.** Starting from \( J^{n+1}_j \) provided by (2.14a), we approximate system (2.13) by

\[
\begin{align*}
\frac{J^{n+1}_j - J^n_j}{\Delta t} + \frac{\tilde{u}_{j+1/2} - \tilde{u}_{j-1/2}}{\Delta x_j} &= 0, \\
\frac{(\nabla J)^{n+1}_j - (\nabla J)^n_j}{\Delta t} + \frac{\nabla^{n+1}_j \tilde{u}_{j+1/2} - \nabla^n_j \tilde{u}_{j-1/2}}{\Delta x_j} &= 0,
\end{align*}
\]

(2.19a) \quad (2.19b)
with the upwind flux

\[(2.20) \quad V_{j+1/2, u_j+1/2} = V_{j} (\bar{u}_{j+1/2})^+ + V_{j+1} (\bar{u}_{j+1/2})^- \, .\]

Note that the discrete version of \( w = u \) is simply \( w_{j+1/2} = \bar{u}_{j+1/2} \). In other words, the velocity to be prescribed at the edges for the projection step is an outcome of the Lagrange step.

To convince ourselves that the combination of the Lagrange step and the projection step gives rise indeed to a consistent approximation of the initial model, let us add (2.14) and (2.19). It follows that

\[ J_{n+1,j} - J_{n,j} \Delta t = 0, \]

\[ \frac{(\nabla J)^{n+1}_j - (\nabla J)^n_j}{\Delta t} + \frac{\bar{F}_{j+1/2} - \bar{F}_{j-1/2}}{\Delta x_j} = 0, \]

where

\[ \bar{F}_{j+1/2} = V_{j} (\bar{u}_{j+1/2})^+ + V_{j+1} (\bar{u}_{j+1/2})^- + \bar{P}_{j+1/2}. \]

Since \( J^n_j = 1 \), we have \( J^{n+1}_j = 1 \), and the resulting scheme

\[ \frac{\nabla_j^{n+1} - \nabla_j^n}{\Delta t} + \frac{\bar{F}_{j+1/2} - \bar{F}_{j-1/2}}{\Delta x_j} = 0 \]

appears to be a discretization of system (2.3) in conservation form, whose numerical flux (2.22) is consistent.

2.5. Characteristic speeds and CFL conditions. Let us go back to the semi-discrete formulation (2.12)–(2.13) in order to analyze the characteristic speeds of each step. Introduce the specific volume \( \tau = 1/\rho \), and consider the vector

\[ (2.24) \quad v = (\tau, Y, u). \]

Since the first equation of (2.12b) boils down to

\[ (2.25) \quad \partial_t (\rho J) = 0, \]

we can make use of the property \( \rho J = \rho^n \) (due to \( J^n = 1 \)) in order to simplify the remaining equations of (2.12). It can be readily checked that these equations can be written as

\[ (2.26) \quad \rho^n \partial_t v + \partial_\lambda g = 0, \]

where

\[ (2.27) \quad g = (-u, 0, P) \]

has to be seen as a function of \( v \). Computing the eigenvalues of the Jacobian matrix

\[ \nabla v g(v), \]

we are in a position to show that the eigenvalues of (2.26) are

\[ (2.28) \quad -\tau^n \sqrt{-P_\tau(\tau, Y)|_Y} < 0 < \tau^n \sqrt{-P_\tau(\tau, Y)|_Y}, \]
the pressure \( P \) being now seen as a function of \((\tau, Y)\). At this point we remark that the sound speed \( c \), defined by (2.7), can also be expressed as

\[
(2.29) \quad c = \tau \sqrt{-P_\tau(\tau, Y)/Y}.
\]

Hence, the first and third eigenvalues of (2.28) are homogeneous to acoustic speeds. As a consequence, if the Lagrange step is solved explicitly with respect to time, then there should be a CFL restriction based on the acoustic speeds given by (2.28) of the type

\[
(2.30) \quad \max_j \left\{ \frac{c_n^j \Delta t}{\Delta x_j} \right\} \leq \frac{1}{2}.
\]

The characteristic speeds arising from the projection step are even easier to compute. Subtracting (2.13a) multiplied by \( V \) from (2.13b) and assuming \( J \neq 0 \), we obtain

\[
(2.31) \quad \partial_t V + \frac{u}{J} \partial_x V = 0.
\]

All components of \( V \) are transported at the same characteristic speed \( u/J \). As a consequence, since the projection step is always solved explicitly with respect to time, there must be a CFL restriction based on \( u/J \).

This CFL condition in the projection step can be more deeply investigated at the fully discrete level. Eliminating \( J_{n+1} \) from (2.19a)–(2.19b) and thanks to \( J_{n+1} = 1 \), we have

\[
(2.32) \quad V_{j+1}^{n+1} = \left[ 1 - \frac{(\bar{u}_{j-1/2})^+ \Delta t}{\Delta x_j} + \frac{(\bar{u}_{j+1/2})^- \Delta t}{\Delta x_j} \right] V_j^n + \frac{(\bar{u}_{j-1/2})^+ \Delta t}{\Delta x_j} V_{j-1}^n + \frac{(\bar{u}_{j+1/2})^- \Delta t}{\Delta x_j} V_{j+1}^n.
\]

Stability is obtained by requiring that the right-hand side be a convex combination. A sufficient condition for this to occur is obviously

\[
(2.33) \quad \max_j \max \left\{ \frac{(\bar{u}_{j-1/2})^+ \Delta t}{\Delta x_j}, -\frac{(\bar{u}_{j+1/2})^- \Delta t}{\Delta x_j} \right\} \leq \frac{1}{2}.
\]
coupling of multiresolution with the semi-implicit Euler relaxation scheme is detailed in [13] for the nondrift model and in [1] for the complete model with drift and friction. The adaptation of the method to the Lagrange projection scheme is straightforward. As in the pure Eulerian scheme [13], since the fast acoustic waves are treated implicitly, they are smoothed out very early in the computation. They are therefore well captured on coarser levels of discretization. The prediction of the adaptive grid from one time-step to the next can be handled with the same strategy as in the explicit case, based on the slow transport speed.

The important point for our present work is the enhancement to LTS. In the previous works [1,6,13] the time-step is dictated by the size of the smallest cell in the adaptive grid which enters into a CFL stability condition. We first describe how to adapt the LTS approach developed by Müller and Stiriba [24] in the framework of the explicit Lagrange projection scheme. The design of the LTS algorithm for the implicit version of the Lagrange projection method is the original part of this work and is described in detail.

First we briefly recall in section 3.1 the basics of the multiresolution analysis. It is then used, in section 3.2, to monitor a time-varying adaptive grid for a conservative finite-volume scheme. Technical details and examples can be found in [23].

3.1. Basics of multiresolution analysis. The computational domain \([0,L]\) is discretized uniformly with a mesh size \(\Delta x_0 = L/(N_0 + 1)\) such that

\[
(3.1) \quad [0,L] = \bigcup_{j=0}^{N_0} \Omega_j^0 \quad \text{with} \quad \Omega_j^0 = \left[ x_j^{0}, x_j^{0} + \Delta x_j^0 \right].
\]

Starting from this coarsest discretization labeled 0, we define a hierarchy of \(K + 1\) nested grids \((S_k)_{k=0}^{K}\) by dyadic refinement. At level \(k\) the step size is \(\Delta x_k = 2^{-k} \Delta x_0 = 2^{K-k} \Delta x_K\), the cells are \(\Omega_j^k = \Omega_{2j}^{k+1} \cup \Omega_{2j+1}^{k+1}\), and the interfaces are \(x_{2j+1/2}^{k+1} = x_{j+1/2}^{k}\), with new midpoints

\[
(3.2) \quad x_{2j+1/2}^{k+1} = \frac{1}{2} \left( x_{j+1/2}^k + x_{j+1/2}^k \right).
\]

Initially, the piecewise constant vector-valued function \(V\) is defined on the finest grid, numbered \(K\), where it is represented by the sequence of its mean values \(V^K = (V_{K,j})_j\) on the cells \(\Omega_j^K = [x_{j-1/2}^K, x_{j+1/2}^K]\). The coarsening operator \(P^{k-1}_k\) consists of cell averaging from one grid to the coarser one, i.e.,

\[
(3.3) \quad V^{k-1} = P^{k-1}_k V^K \quad \text{with} \quad V_{k-1,j} = \frac{1}{2} (V_{k,j+1} + V_{k,j+1}).
\]

The inverse operator consists of recovering the mean values on grid level \(k\), given the mean values on the coarser level \(k - 1\). This involves an approximation—or prediction—operator \(P^{k-1}_{k-1}\), which we define here as

\[
(3.4) \quad \hat{V}^k = P^{k-1}_{k-1} V^{k-1} \quad \text{with} \quad \left\{ \begin{array}{l}
\hat{V}_{k,j+1} = V_{k-1,j} - \frac{1}{2} (V_{k-1,j+1} - V_{k-1,j}), \\
\hat{V}_{k,j+1} = V_{k-1,j} + \frac{1}{2} (V_{k-1,j+1} - V_{k-1,j}).
\end{array} \right.
\]

We define the detail vector \(D^k\) with

\[
(3.5) \quad D_{k,j} = V_{k,j+1} - \hat{V}_{k,j} \quad \text{for} \quad k = 1, \ldots, K.
\]
The two vectors $V^k$ and $(V^{k-1}, D^k)$ are of same length, and we can use $D^k$ along with $V^{k-1}$ to recover $V^k$ entirely. Iterating this encoding operation from the finest level down to the coarsest provides the multiscale representation

\begin{equation}
M^K = ([0, \ldots, N_0], [0, \ldots, N_0], \ldots, [0, \ldots, 2^{K-1}N_0]).
\end{equation}

where the notation has been extended to $D^0 = V^0$. The indices of the multiscale representation $M^K$ vary in

\begin{equation}
\nabla^K = \{(0, \ldots, N_0), (0, \ldots, N_0), \ldots, (0, \ldots, 2^{K-1}N_0)\}.
\end{equation}

Thanks to the polynomial exactness of the prediction operator (3.4), a small detail in absolute value means that the solution can be locally approached by a polynomial, quadratic in our case. We can use this property to compress the function in the multiple domain by dropping all details smaller than a given threshold. To clarify this idea, we first define a threshold operator $T$ acting on the multiscale representation $M^K$, depending on a subset $\Gamma \subset \nabla^K$ of indices $\gamma = (k, j)$, by

\begin{equation}
(T(M))_\gamma = \begin{cases}
D_\gamma & \text{if } \gamma \in \Gamma,
0 & \text{otherwise}.
\end{cases}
\end{equation}

Given level-dependent threshold values $\varepsilon = (\varepsilon_k)_{k=0,\ldots,K}$, we introduce the subset $\Gamma_\varepsilon = \Gamma(\varepsilon_0, \ldots, \varepsilon_K) := \{ \gamma \text{ s.t. } |D_\gamma| \geq \varepsilon_{|\gamma|} \}$, where $|\gamma| = |(k, j)| = k$. This completes the definition of the threshold operator $T_\varepsilon := T_{\Gamma_\varepsilon}$ and gives rise to an approximating operator $\mathcal{A}_\varepsilon := \mathcal{M}^{-1} T_{\varepsilon} \mathcal{M}$ acting on the physical domain representation. In practice, we take advantage of the fact that the remaining fine-scale details will be concentrated near singularities. This is not such a trivial result because the operator $\mathcal{A}_\varepsilon$ is nonlinear, since $\Gamma_\varepsilon$ depends on $V^K$ through the threshold scheme. We refer to [5] for a thorough investigation of nonlinear approximation and the proof of the main result

\begin{equation}
\|V^K - V^K_\varepsilon\|_{L^1} \leq C \varepsilon, \quad \text{with } V^K_\varepsilon = \mathcal{A}_\varepsilon V^K,
\end{equation}

valid when

\begin{equation}
\varepsilon_0 = 0 \quad \text{and} \quad \varepsilon_k = 2^{K-k} \varepsilon \quad \text{for } k = 1, \ldots, K.
\end{equation}

Here the $L^1$ norm is the norm of the piecewise constant function on the finest grid $S_K$.

\begin{equation}
\|V^K\|_{L^1} = \Delta x_K \sum_{j=0}^{2^KN_0} |V^K_j|.
\end{equation}

For practical purposes it is necessary to enlarge the set of significant details into a gradual tree, which is defined recursively by the Algorithm 1. Choosing the grading parameter $g = 1$ ensures that the stencil of the prediction operator (3.4) always belongs to the tree. We can then define an adaptive grid $S_\varepsilon$, where the local size of the cell will be the grid step corresponding to the finest nonnegligible detail

\begin{equation}
S_\varepsilon = \{(k, j), k \in \{0, \ldots, K\}, j \in \{0, \ldots, 2^KN_0\}, \text{ s.t. } (k, |j/2|) \in \Gamma_\varepsilon \text{ and } (k+1, j) \notin \Gamma_\varepsilon \},
\end{equation}

where $[j]$ denotes the integer part of $j$. Assuming the tree $\Gamma_\varepsilon$ is gradual, the cell $(k, j)$ belongs to the adaptive grid $S_\varepsilon$ if its parent $(k-1, |j/2|)$ is too coarse (which
Algorithm 1 Gradualness of the tree.

for level $k = K - 1 \setminus 0$ do
    if $(k + 1, 2j) \in \Gamma_e$ then
        ensure that $(k, j + \ell) \in \Gamma_e$ for all $\ell = -g, \ldots, g$.
    end if
end for

is measured by the detail $|D_{k, j/2}| \geq \varepsilon_k$ and if it is itself fine enough to render the local smoothness of the solution (which is measured by $|D_{k+1, j}| < \varepsilon_{k+1}$).

The representation $V_e$ of the solution on this adaptive grid is obtained from the encoded multiscale representation $M^K = \mathcal{F}_e \mathcal{M}^K$ by the partial decoding Algorithm 2. Note, in particular, that the representation by its mean value $V_{k, j}$ on an intermediate level $k$ does not mean that the function is locally constant on this cell of width $\Delta x_k$, but simply that its mean values on the finest grid in this area can be recovered—within the $\varepsilon$ accuracy—using the mean values on this intermediate level and the reconstruction operators $P_{l-1}^l$ for $l = k + 1, \ldots, K$.

Algorithm 2 Partial decoding.

Starting from $\mathcal{F}_e \mathcal{M}^K$ on $\Gamma_e$

for level $k = 1 \uparrow K$ do
    if $(k, j) \in \Gamma_e$ then
        compute $\{V_{k+1, 2j} = \hat{V}_{k, 2j}, V_{k+1, 2j+1} = 2V_{k-1, j} - V_{k, 2j}\}$.
    end if
end for

The reverse transformation is performed with the partial encoding Algorithm 3. The grading property imposed by Algorithm 1 ensures that the complexity of the partial decoding and encoding Algorithms 2 and 3 is reduced to $O(\#\Gamma_e)$ instead of $O(2^K)$.

3.2. Application to a finite volume scheme. We now briefly describe how to use the multiresolution analysis in the context of a finite volume scheme, written in conservative form on the most refined level of discretization $K$ as

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

where $\lambda_K = \Delta t / \Delta x_k$ and the flux balance is defined as

$$B_{K,j}^n = F_{K,j+1/2}^n - F_{K,j-1/2}^n.$$

Algorithm 3 Partial encoding.

Starting from $\hat{V}$ on $S_e$

for level $k = K - 1 \setminus 0$ do
    if $(k + 1, 2j) \in \Gamma_e$ then
        compute $\{\hat{V}_{k,j} \text{ using (3.4)}, D_{k+1, 2j} = V_{k+1, 2j} - \hat{V}_{k+1, 2j} \text{ using (3.4)}\}$.
    end if
end for
The numerical flux $\mathbb{F}^n_{K,j+1/2}$ between cells $\Omega^K_j$ and $\Omega^K_{j+1}$ is computed using $r$ values of the solution on each side of the interface $x^K_{j+1/2}$:

$$\mathbb{F}^n_{K,j+1/2} = \hat{F}(\nabla^n_{K,j+1-r}, \ldots, \nabla^n_{K,j}, \nabla^n_{K,j+1}, \ldots, \nabla^n_{K,j+r}).$$

In our case it will be defined by (2.22), along with (2.15) in the explicit case, and therefore $r = 2$.

Applying the multiresolution operator $\mathcal{S}_\varepsilon$ on the finite volume scheme (3.7) with the partial encoding and decoding algorithms, we obtain a fully adaptive scheme

$$\nabla^n_{\varepsilon,k,j} = \nabla^n_{\varepsilon,k,j} - \lambda_k \mathbb{B}^n_{\varepsilon,k,j}, \quad (k,j) \in S_\varepsilon,$n

where $\lambda_k = 2^{k-K} \lambda_K$ and the same recursive definition is used for the flux balances as for the mean values of the solution. However, due to the nestedness of the dyadic grids, we can define the numerical fluxes on coarser scales $k$ from the fluxes defined at the same location on finer scales

$$\mathbb{F}^n_{k,j-1/2} = \mathbb{F}^n_{k+1,2j-1/2} = \cdots = \mathbb{F}^n_{K,2^{k-j}-1/2}.$$

Because of the conservation property, the internal fluxes cancel, and only the fluxes at the boundary of cell $\Omega^n_j$ have to be computed:

$$\mathbb{B}^n_{k,j} = \mathbb{F}^n_{k,j+1/2} - \mathbb{F}^n_{k,j-1/2} = \mathbb{F}^n_{K,2^{k-j+1}} - \mathbb{F}^n_{K,2^{k-j-1}}.$$

In practice the flux balances $\mathbb{B}^n_{\varepsilon}$ in (3.15) are computed using (3.17) on the adaptive solution $\nabla^n_{\varepsilon}$, which reduces the computation complexity significantly. It should be noted nevertheless that the fluxes on the local scale have to be computed using the data on the highest level in order to maintain the accuracy of the reference level $K$. Otherwise one might lose significantly in accuracy, unless some local reconstruction of the solution is performed, as shown in [1]. The analysis of the scheme with a local reconstruction of the data to compute the fluxes is done in the recent work by Hovhannisyan and Müller [20].

The important point in (3.15) is that the adaptive grid $S_{\varepsilon}$ must be adequate to represent the solution at both times $n$ and $n + 1$. More specifically, if we denote by $\Gamma^n_\varepsilon$ the graded tree obtained by applying $\mathcal{S}_\varepsilon$ to $\nabla^n_\varepsilon$, then $\Gamma^n_\varepsilon$ can be inflated into $\Gamma^{n+1}_\varepsilon$ containing $\Gamma^{n+1}_{\varepsilon}$ as well as $\Gamma^n_\varepsilon$, ensuring that the estimation (3.9) is valid at both times $n$ and $n + 1$, that is,

$$(3.18) \quad \|\nabla^{K,n} - \mathcal{S}_\varepsilon \nabla^{K,n}\|_{L^1} \leq C\varepsilon \quad \text{and} \quad \|\nabla^{K,n+1} - \mathcal{S}_\varepsilon \nabla^{K,n+1}\|_{L^1} \leq C\varepsilon$$

when using $\mathcal{S}_\varepsilon = \mathcal{M}^{-1} \mathcal{F}_{\Gamma^{n+1}_\varepsilon} \mathcal{M}$. Setting $\Gamma^{n+1}_\varepsilon$ to $\nabla^K$ does the trick, but it is useless in practice, for the inflated tree should be as small as possible. An economical way to ensure (3.9) both for $\nabla^n_\varepsilon$ and $\nabla^{n+1}_\varepsilon$ was heuristically described by Harten in [18] and leads to Algorithm 4.

Let us comment on the choice of $s$ and $g$: in Harten’s strategy the number of cells added on each side of important details to account for displacement of the solution during one time-step is $s = 1$, thanks to the CFL $< 1$ for explicit schemes. This is an important point of discussion in the extension of the adaptive algorithm to implicit schemes with CFL larger than one, and the value of the parameter $s$ has been discussed in [13]. It turns out that $s = 1$ remains valid in the semi-implicit case because the fast waves are severely damped out by the implicit scheme and can
Algorithm 4 Prediction of the adaptive grid $\Gamma_{\varepsilon}^{n+1}$.

Prediction:
\begin{algorithmic}
\For {level $k = K \searrow 1$}
\If {$(k,j) \in \Gamma_{\varepsilon}^n$ and $|D_j^k| \geq \varepsilon_k$}
\State $(k,j+l) \in \Gamma_{\varepsilon}^{n+1}$ for $l = -s, \ldots, s$
\If {$|D_j^k| \geq 2\varepsilon_k$}
\State $(k+1,2j)$ and $(k+1,2j+1) \in \Gamma_{\varepsilon}^{n+1}$
\EndIf
\EndIf
\EndFor
\end{algorithmic}

Gradualness:
\begin{algorithmic}
\For {level $k = K \searrow 1$}
\If {$(k,j) \in \Gamma_{\varepsilon}^{n+1}$}
\For {$|\ell| \leq g$}
\State $(k-1,\lfloor j/2 \rfloor + \ell) \in \Gamma_{\varepsilon}^{n+1}$
\EndFor
\EndIf
\EndFor
\end{algorithmic}

therefore be well represented at the coarser levels of discretization. Concerning the gradualness accounted for by the second step of Algorithm 4, the multiresolution stencil should always be readily available in order to apply the reconstruction formula (3.4), so in our case $g = 1$. In [6], it is shown that more demanding rules have to be designed on $\Gamma_{\varepsilon}^{n+1}$ in order to obtain (3.18) rigorously. In particular, it may be necessary to refine $\Gamma_{\varepsilon}^n$ on more than one level, depending on the size of the details at time $n$. Nevertheless Harten’s rules work fine in practice, and we content ourselves with them in the numerical simulations.

It is also shown in [6] that in the scalar case, (3.18) provides in turn the following estimate for the perturbation error induced by the adaptive algorithm

\begin{equation}
|V^{K,n} - V_{\varepsilon}^{K,n}|_{L^1} \leq C\varepsilon,
\end{equation}

where $V^{K,n}$ is the solution of the finite volume scheme (3.12) on the finest grid $S_K$. The adaptive solution $V_{\varepsilon}^{K,n}$, computed using (3.15), is reconstructed on the same finest grid by using the decoding Algorithm 2 on the full tree $\nabla K$, where the details on cells in $\nabla K \setminus \Gamma_{\varepsilon}^n$ have been set to zero.

We end this section by summarizing the full adaptive algorithm (see Algorithm 5) and then address the purpose of the present paper, which is to couple it with an LTS strategy.

4. LTS. In the adaptive scheme presented in the previous section the time-step is the same everywhere in the grid and is determined in order to ensure stability. It must obey some CFL condition and therefore depends on the smallest space grid size $\Delta x_K$.

We now address the problem of using different time-steps depending on the local size of the adaptive grid cell. We rely on the assumption that if the stability criterion leads to a time-step $\Delta t$ on the finest grid of size $\Delta x_K$ at level $K$, then the scheme can be applied in coarser regions where the grid size is $\Delta x_k = 2^{K-k}\Delta x$ using a
Algorithm 5 Adaptive algorithm.

Initialization: encoding of the initial solution and definition of $\Gamma^0_\varepsilon$.

for time-step $n = 0, \ldots, N - 1$ do

Prediction of $\tilde{\Gamma}^{n+1}_\varepsilon$ using Algorithm 4.

Partial decoding of $V^n_\varepsilon$ on $\tilde{S}^{n+1}_\varepsilon$ (derived from $\tilde{\Gamma}^{n+1}_\varepsilon$ using (3.12)).

Evolution of $V^n_\varepsilon$ to $V^{n+1}_\varepsilon$ on the adaptive grid $\tilde{S}^{n+1}_\varepsilon$ using (3.15).

Definition of $\Gamma^{n+1}_\varepsilon$ by partial encoding of $V^{n+1}_\varepsilon$.

end for

Decoding of $V^N_\varepsilon$ on the finest grid $S^N_\varepsilon$.

---

Fig. 1. Using a different time-step according to the grid size.

The corresponding larger time-step $\Delta t_k = 2^{K-k} \Delta t$ and still be stable, since the ratio

\begin{equation}
\lambda = \frac{\Delta t_k}{\Delta x_k}
\end{equation}

remains the same (see Figure 1). The obvious difficulty arising from this discrepancy in time-steps is that the solution is not synchronized after one time-step, and something special must be done at interfaces between regions of different grid size.

We retain here the formalism of [24] and summarize in the first section the algorithm in the explicit case. The second section is devoted to the semi-implicit scheme for which a specific implementation of the LTS is designed.

4.1. LTS for the explicit scheme. We denote by $\Delta t_0 = 2^K \Delta t$ the macro time-step, which can be used on cells at the coarsest level. The discrete times $t_n = n \Delta t_0$ are subdivided into $2^K$ intermediate time-steps $t_{n+1-\kappa}$ for $i = 1, \ldots, 2^K$ with step size $\Delta t$.

At time $t_{n+1-\kappa}$ the smallest synchronization level is determined by

\begin{equation}
k_i := \min \left\{ k; \ 0 \leq k \leq K, \ i \equiv 0 \mod 2^{K-k} \right\}.
\end{equation}

As illustrated on the three-level hierarchy example depicted in Figure 2, all cells belonging to levels finer than or equal to $k_i$ are updated during the intermediate step $i$. Two cases must be considered, depending on whether those cells belong or not to the range of dependence $\Sigma_\ell$ of finer cells. The range of dependence $\Sigma_\ell$ of a cell $\ell$ on level $k$ is determined by the stencil of the $2r$-point flux (3.14) and the multiscale local prediction (3.4) as a set of cells on level $k-1$:

\begin{equation}
\Sigma_\ell := \left\{ \lfloor (\ell - r)/2 \rfloor - 1, \ldots, \lfloor (\ell + r)/2 \rfloor + 1 \right\}.
\end{equation}

The partial updating at level $k$ acts on active cells at this level, whose indices are in

\begin{equation}
\tilde{I}_k := \left\{ (k, j); \ (k, j) \in \tilde{S}^{n+1}_\varepsilon \right\}.
\end{equation}
This set of indices is split into two disjoint sets $\bar{I}_k = C_k \cup \bar{C}_k$. The index set $C_k$ contains cells at level $k$ that can be evolved in time by one time-step with step size $\Delta t_k = 2^{K-k} \Delta t$, without having to access data on finer levels in order to compute the fluxes. The complementary set $\bar{C}_k$ contains the cells at level $k$ which are close to cells on level $k + 1$. They must be evolved along with them, with the time-step $\Delta t_{k+1} = 2^{K-k-1} \Delta t$, in order to provide the synchronized data necessary to compute all the fluxes on level $k + 1$.

To sum it up, at each intermediate time-step the index sets are designed with Algorithm 6, in accordance with the evolution of the grid $\bar{S}^{n+1}$ (detailed in the next section 4.2).

**Algorithm 6 Definition of the index sets.**

On the finest level $K$ all cells in the adaptive grid are evolved

$$C_K := \{(K,j) ; (K,j) \in \bar{S}^{n+1} \}_{K} = \bar{I}_K.$$  

for $k = K - 1 \searrow 0$ do

$$C_k := \{(k,j) \in S^{n+1} ; \bar{I}(k+1,\ell) \in \bar{S}^{n+1} : j \in \Sigma_{\ell}\},$$

$$\bar{C}_k := \bar{I}_k \setminus C_k.$$  

end for

With these definitions, illustrated in Figures 2 and 3, the evolution of the solution during a macro time-step is performed in a loop on intermediate time-steps. At each intermediate time-step $i$ the solution is updated on all active cells belonging to levels $k \geq k_i$, using time-step $\Delta t_k = 2^{K-k} \Delta t$, except for the cells in the transition regions where a half time-step is used. It is also updated on cells in the transition region of the immediately coarser level $k_i - 1$, using a half time-step (here $2^{K-k_i} \Delta t$). Thanks to the explicit nature of the scheme and of the flux computation (2.22), we can use the conservation property depicted in Figure 4 to synchronize the solution in the manner detailed in Algorithm 7.

**Fig. 2.** A three-level adaptive grid with transition zones of width 2 represented with dotted lines. The intermediate step number is denoted by $i = 1, \ldots, 4$, and $k_i$ denotes the synchronization level. The $\diamond$, $\star$, and $\bigstar$ indicate the fluxes that are computed at the corresponding time-step. No intermediate regridding is performed.

**Fig. 3.** Definition of index set $C_{K-1}$ and $C_K$, transition zone $\bar{C}_{K-1}$, and ranges of dependence $\Sigma_{\ell}$ and $\Sigma_m$ for the cells $(K,\ell)$ and $(K,m)$, in the case where $r = 2$. 

Copyright © by SIAM. Unauthorized reproduction of this article is prohibited.
Algorithm 7 Synchronized time evolution for time-step $t_{n+2^k}$.

for each intermediate time-step $i = 1, \ldots, 2^K$ do
  \begin{itemize}
  \item Intermediate regridding if $i$ is odd (Algorithm 8).
  \item Update fluxes on cells that have been modified at previous time-step:
  \end{itemize}
  for levels $k = K \setminus k_{j-1}$ do
    for $j$ s.t. $(k, j) \in \tilde{S}_{n+1}$ do
      Compute $F_{k,j-1/2}$ using (2.14)–(2.22).
    end for
  end for
  for $(k_{i-1}, 1, j) \in C_{k_{i-1} - 1}$ s.t. $(k_{i-1}, 2j - 1) \in \tilde{S}_{n+1}$ do
    Compute $F_{k_{i-1},j-1/2}$ using (2.14)–(2.22).
  end for
  \begin{itemize}
  \item Time evolution
  \end{itemize}
  for levels $k = L \setminus k_i$ do
    use full time-step $\Delta t_k$
    for $(k, j) \in C_k$ do
      \begin{align*}
      \psi_{k,j}^{n+2^k} &= \psi_{k,j}^{n+1} - \lambda(F_{k,j+1/2}^{n+2^k} - F_{k,j-1/2}^{n+2^k}) - \frac{1}{2}(F_{k,j+1/2}^{n+(i-1)2^k} - F_{k,j-1/2}^{n+(i-1)2^k}).
      \end{align*}
    end for
  end for
  for levels $k = L \setminus k_i - 1$ do
    use half time-step $\Delta t_k/2$
    for $(k, j) \in C_k$ do
      \begin{align*}
      \psi_{k,j}^{n+2^k} &= \psi_{k,j}^{n+(i-1)2^k} - \frac{1}{2}(F_{k,j+1/2}^{n+(i-1)2^k} - F_{k,j-1/2}^{n+(i-1)2^k}).
      \end{align*}
    end for
  end for
Elsewhere the solution remains unchanged.

4.2. Regridding at intermediate time-steps. An important point in the LTS algorithm is the evolution of the adaptive grid. In [24], it is shown that a prediction of the grid from one macro time-step to the next is not efficient when the number of levels is large. In one macro time-step a significant information might move by

Fig. 4. Flux conservation at interface between level $k$ and $k+1$ implies $2\tilde{F}_{k,j-1/2}^n = \tilde{F}_{k+1,2j-1/2}^n + \tilde{F}_{k+1,2j}^n$. The solution $\psi^{n+1/2}_{k,j-1}$ is used to compute $\tilde{F}_{k+1,2j-1/2}^n$. 

\begin{itemize}
\end{itemize}
Algorithm 8 Intermediate regriding at odd time-steps.

- Partial encoding of the solution from levels $K$ to $k_{i-1}$.
- Prediction:
  
  for levels $k = K \setminus k_{i-1}$ do
  
  if $(k, j) \in \Gamma^{n+(i-1)2^{-K}}_{\varepsilon}$ and $\|D_j^k\| \geq \varepsilon_k$ then
  
  $(k, j + \ell) \in \Gamma^{n+(i)2^{-K}}_{\varepsilon}$ for $|\ell| \leq s$
  
  if $\|D_j^k\| \geq 2\varepsilon_k$ then
  
  $(k + 1, 2j)$ and $(k + 1, 2j + 1) \in \Gamma^{n+(i)2^{-K}}_{\varepsilon}$
  
  end if
  
  end if
  
  end for

- Gradualness:
  
  for levels $k = K \setminus k_{i-1}$ do
  
  if $(k, j) \in \Gamma^{n+(i)2^{-K}}_{\varepsilon}$ then
  
  if $(k - 1, \lfloor j/2 \rfloor + \ell) \in \Gamma^{n+(i-1)2^{-K}}_{\varepsilon}$ for $|\ell| \leq g$ then
  
  $(k - 1, \lfloor j/2 \rfloor + \ell) \in \Gamma^{n+(i)2^{-K}}_{\varepsilon}$
  
  end if
  
  end if
  
  end for

- Partial decoding of the solution from levels $k_{i-1}$ to $K$.
- Update indices sets $C_k, \overline{C}_k$, for $k = K \setminus k_{i-1}$.

$2^K$ cells in each direction. This has to be accounted for with the parameter $s$ in the prediction Algorithm 4, leading to a prerefined grid whose size increases geometrically with the number of levels. The alternative consists of predicting the evolution of the grid at intermediate time-steps. The partial encoding/decoding and thresholding is performed on the part of the solution which has been modified, i.e., for all levels above the synchronization level at a previous intermediate time-step, in the manner described in Algorithm 8. Note that since the synchronization level is equal to $K$ for odd intermediate time-steps, no regriding can take place after them.

The prediction parameter $s$ should be equal to the maximum displacement of a singularity in one time-step. In the explicit case, it is therefore equal to 1 as in the standard multiresolution Algorithm 4. We will see that for the semi-implicit case, it is necessary to take into account the fast wave and assume a speed larger than 1. The gradualness parameter $g$ must ensure the possibility of regridding and still preserve the transition zone. It is therefore linked to the parameter $s$ through

$$g = \left\lfloor \frac{s + r}{2} \right\rfloor + 1.$$  

This rule is illustrated in Figure 5 for speeds 1, 2, and 3. For each example, the predicted tree $\Gamma_{\varepsilon}$ is represented at the beginning of the macro time-step, with the single nonnegligible detail symbolized with a ♦. Then the resulting adaptive grid $S_{\varepsilon}$ is represented at the different intermediate time-steps, along with the progression of the singularity, marked with a ♦. The dotted lines indicate the transition zones.

4.3. Numerical simulations for the explicit scheme. In this section we validate the algorithms on a test case. We compare the uniform finite volume scheme
Fig. 5. Grid prediction at intermediate time-steps. $\Gamma_\varepsilon$ indicates the tree at the beginning of the macro time-step, with a single nonnegligible detail $\bullet$. The evolution of the adaptive grid $S_\varepsilon$ includes intermediate regridding every two intermediate time-steps. The singularity is marked with a $\bigcirc$. The grid propagation speed is (a) $s = 1$, (b) $s = 2$, (c) $s = 3$.

(U) with two versions of the multiresolution algorithm: (MR) denotes the algorithm with global time stepping based on the cell size at the finest resolution level $K$, (LTS) denotes the LTS with partial regridding at each intermediate time-step.

The simulation is set in a 32km long pipeline. At the initial time the density of the mixture is $500\text{kg/m}^3$ until $x = 16\text{km}$ and $400\text{kg/m}^3$ beyond. The gas mass fraction is, respectively, 0.2 and 0.4, and the speed is uniform and equal to $-10\text{m/s}$. The transport wave moves slowly toward the left at a speed $-29\text{m/s}$, while two acoustic waves are visible on the density and speed components moving in opposite directions at roughly $-263\text{m/s}$ and $+238\text{m/s}$.

The pressure law corresponds to a perfect gas and incompressible liquid

(4.6)

$$P(\rho, \rho Y) = a^2 \frac{\rho \rho Y}{\rho_l - \rho(1 - Y)}.$$  

Figure 6 displays the density field obtained after 42s of propagation with the uniform, the multiresolution, and the LTS schemes. The grid has five levels of refinement, which amount to $J = 4096$ grid points on the smallest level of refinement.

We use a constant elementary time-step on the finest level throughout the simulation, determined by taking the minimum time-step during a previous uniform simulation. For the 4096 points simulations we use $\Delta t = 0.0096\text{s}$. The MR and LTS results are obtained with a threshold parameter $\varepsilon = 0.005$. The crosses for the multiresolution and circles for the LTS denote the level of refinement used locally to discretize the solution. In Figure 7 the range of abscissa is zoomed in the three regions of interest. The quality of approximation of the transport wave is the same for the two algorithms. The LTS does a slightly better job than the standard multiresolution as far as the acoustic waves are concerned. This is because there are less projection steps performed and therefore less diffusion.
Fig. 6. Density field and grid refinement at \( t = 42s \) for \( \varepsilon = 0.005 \). (a) Explicit multiresolution scheme (MR) against uniform solution (U) on finest (5th) level and initial solution (Ini) (b) LTS scheme (LTS). The symbols + indicate the level of refinement used locally.

Fig. 7. Density field at \( t = 42s \) for \( \varepsilon = 0.005 \). Zoom in (a) transport wave, (b) left-going acoustic wave, (c) right-going acoustic wave.

To study the performance and the robustness of the algorithm we perform a parameter study. We test two discretizations: \( J = 4096 \) points on a hierarchy of 5 levels and \( J = 8192 \) with 6 levels. The range of the \( \varepsilon \) threshold goes from 0 to \( 10^{-2} \).
For each set of parameters we compute the solution at $t = 42s$ with the 3 algorithms and calculate the relative error between the adaptive solution reconstructed on the finest level at the final time and the solution computed on the uniform finest grid.

The adaptive solution is reconstructed on the finest level by using the decoding Algorithm 2 on the full tree $\nabla^K$, where the details on cells in $\nabla^K \setminus \Gamma^K_\varepsilon$ have been set to zero. The errors are computed for the total density $\rho$ (which is sensitive to both kinematic and acoustic phenomena) and the gas mass fraction $Y$ (which is transported only by the slow kinematic wave) by the formulae

$$E_1(\rho_\varepsilon) = \frac{\sum_{j=0}^{J} |\rho_j^\varepsilon - \rho_j|}{\sum_{j=0}^{J} |\rho_j|}, \quad E_1(Y_\varepsilon) = \frac{\sum_{j=0}^{J} |Y_j^\varepsilon - Y_j|}{\sum_{j=0}^{J} |Y_j|}. \tag{4.7}$$

This measures the perturbation error introduced by the thresholding and prediction operations.

We also compute the error with respect to a reference solution computed on a uniform mesh at the 9th level of discretization (on 65536 points), that is, 8 times finer than the simulation on 8192 points. It will be considered as the exact solution. In the formulae

$$E_1^{\text{ref}}(\rho_\varepsilon) = \frac{\sum_{j=0}^{J} |\bar{\rho}_j^\varepsilon - \bar{\rho}_j|}{\sum_{j=0}^{J} |\bar{\rho}_j|}, \quad E_1^{\text{ref}}(Y_\varepsilon) = \frac{\sum_{j=0}^{J} |\bar{Y}_j^\varepsilon - \bar{Y}_j|}{\sum_{j=0}^{J} |\bar{Y}_j|}, \tag{4.8}$$

where $\bar{\rho}_j$ and $\bar{Y}_j$ denote the average of the reference solution on the 8 or 16 cells covering cell $j$. This measures the total error, combining both the thresholding and the discretization effects. In the case where $\varepsilon = 0$ it reduces to the discretization error.

In Figure 8, we display the graph of the error (4.7) with respect to the uniform solution on the finest level $K = 6$ versus the threshold parameter. For both schemes
(MR) and (LTS), the perturbation error converges to zero with respect to the threshold parameter \( \varepsilon \), which corroborates the theoretical estimate (3.19). In Figure 9, we notice that as expected, the total error (4.8) with respect to the reference solution does not converge toward zero as the perturbation error does (see Figure 8). It is bounded below by the discretization error between the uniform solution on the 6th level and the reference solution. Therefore an optimal choice for the threshold value would be such that the discretization and the perturbation error are balanced. For the present set of parameters this would be \( \varepsilon_{\text{opt}} \approx 0.00125 \). For the 5-level hierarchy the best threshold value is slightly larger \( \varepsilon_{\text{opt}} \approx 0.002 \).

In Figure 10, we study the dependence between the precision and the speed of the algorithm by displaying the relative perturbation errors (4.7) between the adaptive and uniform grid solutions as a function of the gain in computing time. The gain is the ratio between the computing time required for the uniform scheme on the finest level of discretization and the computing time required by the multiresolution (MR) or the LTS scheme (LTS) for a given threshold ratio. Each point on these curves corresponds to a different value of the threshold parameter \( \varepsilon \). The best CPU gains are achieved for the highest value of \( \varepsilon \) but also correspond to the highest error. The CPU gain can be as high as 10 for a relative error of one percent for the standard multiresolution, and the LTS goes yet more than twice as fast for a given error level.
In Figure 11, we display the error with respect to the reference solution as a function of the CPU gain for both the 4096 and 8192 point simulations. Here again, each point corresponds to a different value of $\varepsilon$. The CPU gain is computed with respect to the CPU for the uniform simulation on the 5th (respectively, 6th) level for the multiresolution (MR) and LTS (LTS) simulations on the 5-level hierarchy (resp., 6-level hierarchy). Both (MR) and (LTS) simulations converge to the error level of the uniform grid simulation when $\varepsilon$ goes to 0, which is roughly $10^{-3}$ for 4096 points and $5 \times 10^{-4}$ for 8192 points. Since the uniform simulation on 8192 points costs roughly 4 times as much as the one on 4096 points, all multiresolution simulations on the 6-level hierarchy which exhibit a CPU gain superior to 4 and an error below $10^{-3}$ are doing a better and faster job than the uniform simulation on the 5th level.

5. LTS for the semi-implicit scheme. The main difficulty to implement LTS for the semi-implicit scheme lies in the discrepancy between speeds of acoustic and kinematic waves. Actually, if we use a time-step designed on the fast wave speed instead of the slow transport speed as in (2.33), the Müller and Stiriba algorithm [24] designed for implicit schemes is convenient and can be used with CFL = 1. In that case, however, we completely miss the advantage of using an implicit scheme for the Lagrange phase of the algorithm, which is meant to enable us to use a very large time-step. As mentioned in the presentation of the model, it is usual practice, in aerodynamics as well as in the multiphase flow community, to use a semi-implicit scheme with the CFL condition designed on the speed of propagation of the phenomenon of interest. In practice, however, in order to limit the distortion of the acoustic phenomenon, we also enforce the stability condition (2.30) based on the acoustic waves’ speed but with a very large CFL number ($\text{CFL}_{\text{imp}} = 10$ to 20) instead of 0.5. This new CFL acoustic condition (2.30) and the transport CFL condition (2.33) are imposed simultaneously. Note that due to nonlinear effects the ratio between the eigenvalues corresponding to the acoustic waves and that of the transport wave can be larger than the ratio between the sound speeds in the liquid and the gas; hence the limit imposed on the time-step by the CFL_{imp} parameter can very well be met.

This bound (2.30) on $\Delta t$ enables us to design the stencil of influence $SI$ of a given location with respect to the acoustic waves, knowing that these waves can travel at most $\text{CFL}_{\text{imp}}$ cells in one time-step. At intermediate time-step $i$ all cells in levels $k \geq k_i$ are updated and therefore enter the implicit system at the Lagrangian step.
Some additional cells belonging to coarser levels \( k = k_i - 1 \) must be included in the nonlinear implicit systems (2.14)–(2.16) on each side of \( \mathcal{D}_i := \{ \tilde{\mathcal{C}}_k \}_{k=k_i,...,K} \) to compute the Lagrangian phase (see Figure 12). They are the cells in the stencil of influence of cells at the border of \( \mathcal{D}_i \). Equipped with (2.30) and (2.33) we can safely restrict this area \( SI_i \) to a zone of width \( \lceil \text{CFL}_{\text{imp}} \rceil \) on the borders of \( \mathcal{C}_{k_i} \) contiguous to \( \overline{\mathcal{C}}_{k_i-1} \), measured in number of cells of size \( \Delta x_{k_i} \). To design \( SI_i \) in practice, we introduce the distances between a given cell \((k,j)\) and a finer level \(k'\) on its right and on its left:

\[
\begin{align*}
\text{(5.1a)} & \\
& d_l(k,j,k') = \min_{\ell',(k',\ell') \in \mathcal{C}_{k'}} \left( j 2^{k' - k} - \ell' - 1 \right)^+, \\
& d_r(k,j,k') = \min_{\ell',(k',\ell') \in \mathcal{C}_{k'}} \left( \ell' - (j + 1) 2^{k' - k} \right)^+.
\end{align*}
\]

The stencil \( SI_i \) is built according to Algorithm 9, and the nonlinear system of (2.14)–(2.16) is then solved for all cells belonging to \( SI_i \) using a linearization method described for instance in [9]. The implicit step involves all cells belonging to levels \( k = k_i \) and some cells belonging to coarser levels, differing therefore from the method used in [24] where implicit systems are solved independently on each level. This could not work here where we deal with transient problems with a large CFL number.

The solution of the Lagrange phase is then used to compute the Euler projection fluxes (2.22) in a manner described in Algorithm 10. As in the explicit scheme, the cells who are updated at this stage belong to the levels above or equal to the current synchronization level \( k_i \) or are in the transition zone \( \overline{\mathcal{C}}_{k_i-1} \) of the immediately coarser level (see Figure 12).
Algorithm 9 Design of the Lagrangian step.

for levels $k = K \setminus k_i$ do
  for $(k, j) \in C_k$ do
    add $(k, j)$ to $SI_i$ for a full time-step advance $\Delta t_k$.
  end for
  for $k < K$ and $(k, j) \in C_k$ do
    add $(k, j)$ to $SI_i$ for a half time-step advance $\Delta t_k/2$.
  end for
end for

for levels $k = k_i - 1 \setminus 0$ do
  for cells $(k, j)$ s.t. $d_i(k, j, k_i) \leq \lceil \text{CFL}_{\text{imp}} \rceil$ or $d_r(k, j, k_i) \leq \lceil \text{CFL}_{\text{imp}} \rceil$ do
    add $(k, j)$ to $SI_i$ for a reduced time-step advance $\Delta t_{k_i} = \Delta t_k 2^{k-k_i}$.
  end for
end for

Algorithm 10 Euler projection fluxes (2.22).

for levels $k = K \setminus k_i$ do
  for $(k, j) \in C_k$ do
    Compute $\tilde{P}_{j-1/2}^{n+(i-1)2^{-K}}$ and $\tilde{u}_{j-1/2}^{n+(i-1)2^{-K}}$ using (2.16).
    Save fluxes at the previous intermediate time-step for transition zone end points.
    Compute fluxes using (2.22)
  end for
end for

for level $k = k_i - 1$ do
  for $(k, j) \in I_k$ and $d_i(k, j, k_i) \leq r + 1$ or $d_r(k, j, k_i) \leq r + 1$ do
    Compute $\tilde{P}_{j-1/2}^{n+i2^{-K}}$ and $\tilde{u}_{j-1/2}^{n+i2^{-K}}$ using (2.16).
    Compute fluxes using (2.22).
  end for
end for

We will now explain that cells at the boundaries of the transition zone require a special treatment at the Euler projection step. In the case of the explicit scheme the fluxes at the end of the transition zone $C_k$ are not modified, and the conservation property depicted in Figure 4 can be used in a very simple manner. In the case of the semi-implicit scheme, the solution in the transition zone has been modified during the Lagrangian step; therefore all fluxes must be computed again as depicted in Figure 13. Furthermore a special synchronization must be done on levels $k$, for $k_i \leq k < K$, at the interface between cells $(k, j)$ and $(k, j+1)$ if $d_i(k, j, k_i + 1) = r + 1$ or at the interface between cells $(k, j - 1)$ and $(k, j)$ if $d_r(k, j, k_i + 1) = r + 1$.

(i) If $d_i(k, j, k_i + 1) = r + 1$, the solution on cell $(k, j)$ is updated over $\Delta t_{k_i}/2$, while the solution on cell $(k, j+1)$ is updated over $\Delta t_{k_i}$. The flux $\tilde{F}_{k,j+1/2}^{n+(i-1)2^{-K}}$ has been computed using the synchronized Lagrange step solution and can be used straightforwardly in (2.23) for cell $(k, j+1)$. However, solution on cell $(k, j)$ has already been advanced with a half time-step at a previous intermediate time-step $i' = i - 2^{K-k}$, using flux $\tilde{F}_{k,j+1/2}^{n+(i'-1)2^{-K}}$. This quantity has not been tampered with.
already been advanced with a half time-step at a previous intermediate time-step \(i\), the solution on cell \((k, j)\) has been computed using the synchronized Lagrange step solution and can be used straightforwardly in (2.23) for cell \((k, j)\).

(ii) Similarly if \(d_t(k, j, k+1) = r+1\), the solution on cell \((k, j-1)\) is updated over \(\Delta t_k\), while the solution on cell \((k, j)\) is updated over \(\Delta t_k/2\). The flux \(\overline{F}_{k,j-1/2}^{n+(i-1)2^{-K},t}\) has been computed using the synchronized Lagrange step solution and can be used straightforwardly in (2.23) for cell \((k, j-1)\). However, solution on cell \((k, j)\) has already been advanced with a half time-step at a previous intermediate time-step \(i' = i - 2^{K-k}\), using flux \(\overline{F}_{k,j-1/2}^{n+(i'-1)2^{-K},t}\). Therefore, on \((k, j)\) (2.23) should be replaced by

\[
\psi_{k,j}^{n+2^{-K}} = \psi_{k,j}^{n+(i-1)2^{-K}} - \frac{\Delta t}{2\Delta x} \left( \overline{F}_{k,j+1/2}^{n+(i-1)2^{-K},t} - \overline{F}_{k,j-1/2}^{n+(i-1)2^{-K},t} \right),
\]

The time evolution Algorithm 11 takes into account this special synchronization.

**Algorithm 11 Time evolution and synchronization.**

```alg
for levels \(k = L \geq k_l\) do
    full time-step \(\Delta t_k\) for \((k, j) \in C_k\)
    \[
    \psi_{k,j}^{n+2^{-K}} = \psi_{k,j}^{n+(i-1)2^{-K}} - \frac{\Delta t}{2\Delta x} \left( \overline{F}_{k,j+1/2}^{n+(i-1)2^{-K},t} - \overline{F}_{k,j-1/2}^{n+(i-1)2^{-K},t} \right).
    \]
end for

for levels \(k = L \geq k_l - 1\) do
    half time-step for \((k, j) \in \overline{C}_k\)
    \[
    \psi_{k,j}^{n+2^{-K}} = \psi_{k,j}^{n+(i-1)2^{-K}} - \frac{\Delta t}{2\Delta x} \left( \overline{F}_{k,j+1/2}^{n+(i-1)2^{-K},t} - \overline{F}_{k,j-1/2}^{n+(i-1)2^{-K},t} \right).
    \]
    if \(k > k_l\) then
        special evolution equations should be used at the end of transition zone: (5.3)
        if \(d_t(k, j, k+1) = r+1\) and (5.2) if \(d_t(k, j, k+1) = r+1\).
    end if
    Elsewhere the solution remains unchanged
end for
```

Algorithm 12 sums up the semi-implicit method. As in the explicit case, it includes a regridding step which we will now discuss.
Algorithm 12 Synchronized time evolution.

for intermediate time-step $i = 1, \ldots, 2^K$ do
  Intermediate regridding if $i$ is odd.
  Solve Lagrange step (see Algorithm 9).
  Compute Euler projection fluxes (see Algorithm 10).
  Time evolution and synchronization (see Algorithm 11).
end for

5.1. Grid prediction and regridding. The time-step used in the semi-implicit scheme is designed according to the stability condition (2.33), which ensures that the slow transport wave can be handled with a maximum principle by the explicit projection step. For this part of the solution, the prediction of the tree assuming a displacement of a singularity of at most one cell per time-step is relevant. The fast acoustic waves being treated implicitly at the Lagrange step, a singularity in this part of the solution will smooth out in all the domain of computation in a single time-step, losing, of course, most of its singular nature. To quantify this property we study the simple case of the first-order forward implicit scheme applied to a linear scalar equation with speed one
\[
\frac{\partial U}{\partial t} + \frac{\partial U}{\partial x} = 0.
\]
\[U_{j+1}^n = U_j^n - \lambda(U_{j+1}^{n+1} - U_j^{n+1}).\]  \hspace{1cm} (5.4)

We consider a solution at time $t_n$ built from a single detail at position $J$ on level $K-1$ in the multiresolution domain:
\[\forall^0 = 0, \quad \mathbb{D}^1 = \cdots = \mathbb{D}^{K-2} = 0, \quad \mathbb{D}_{K-1,j} = \delta^J_j, \quad \mathbb{D}^K = 0.\] \hspace{1cm} (5.5)

After transformation into the physical space using the decoding operator $\mathcal{M}^{-1}$, the solution is advanced in time using the implicit scheme (5.4) and then transformed back in the multiscale representation using the encoding operator $\mathcal{M}$. We display in Figure 14(a) different details of this solution on level $K-1$ as a function of the coefficient $\lambda = \Delta t / \Delta x$. The strength of the initial detail $\mathbb{D}_{K-1,j}$ decreases rapidly and is below 10\% of its initial value for $\lambda > 10$. The next detail in the direction of the propagation $\mathbb{D}_{K-1,j+1}$ has a significant amplitude for $\lambda \in [0, 5]$. We then represent the two following details: $\mathbb{D}_{K-1,j-1}$ and $\mathbb{D}_{K-1,j+2}$. They are already very small, and all others are negligible. The curve corresponding to $\mathbb{D}_{K-1,j+1}$ indicates that

![Fig. 14. Details at time $\Delta t$ as a function of $\lambda$. (a) $D_{K-1,j+m}$ and (b) $D_{K-2,j/2+m}$ for $m = -1, \ldots, 2$.](image-url)
detail \( \mathbb{D}_{K-1,J} \) has propagated at a speed higher than 1 in a manner that cannot be completely neglected at level \( K - 1 \). In Figure 14(b), the details at the coarser level \( K - 2 \) are displayed. The strength of the detail on cell \( J/2 + 1 \) is comparable to that on the cell \( J/2 \), which also means that some of the solution has actually moved with speed 2. This elementary example illustrates that the details which are transported at fast speed are also damped by the implicit scheme, which is good because it means that they will be captured by coarser levels in the multiresolution hierarchy. Nevertheless, to ensure that the regridding at intermediate time-steps can well capture the part of the solution which has moved fast, we increase the speed \( s \) to 3 instead of 1 in the grid prediction step of Algorithm 8. This has the effect of broadening the tree, thanks to the gradualness rule (4.5), in the manner described in Figure 5(c).

5.2. Numerical simulations for the semi-implicit scheme. To illustrate the algorithm presented above we use the same test case as in subsection 4.3 for the explicit scheme. The uniform scheme provides us with a time-step \( \Delta t = 0.065 \) s, ensuring stability throughout the simulation. This is roughly seven times larger than the one used for the explicit scheme simulations.

Figure 15 displays the density field computed with the uniform scheme and the adaptive schemes (MR) and (LTS). The zoom in the three different waves displayed in Figure 16 show the robustness of the new algorithm. The kinematic wave is reproduced as well, if not better as with the standard (MR) scheme. The acoustic waves are also well handled, even though they are more smoothed out with the LTS algorithm than with the standard multiresolution. The grids displayed above the density field in Figure 15 show that the resolution of both schemes is comparable in the vicinity of the acoustic waves.

A parameter study for varying values of the threshold \( \varepsilon \) is presented in Figures 17, 18, 19, and 20. In Figure 17 we display the relative perturbation error (4.7) between the density or the gas mass fraction obtained using the uniform scheme or the different adaptive schemes. This error decreases monotonously for both the MR and LTS
schemes. We also observe that the relative error on the gas mass fraction is two orders of magnitude smaller than the error on the density. This is quite normal since the gas mass fraction is driven by the slow kinetic wave and therefore entirely computed during the second step of the scheme with an explicit and accurate scheme. The adaptive grid retains almost all details in the vicinity of the discontinuity in the gas mass fraction, which accounts for a much smaller error. On the other hand the density sees acoustic waves which are more sensitive to the adaptivity of the grid because they are computed with the implicit scheme.

We notice, however, that the curves of the perturbation errors reach a flat for very small values of the tolerance $\varepsilon$ while they were converging toward 0 in the explicit case (see Figure 8). This comes from using a Riemann problem as a test case. As soon
as the threshold tolerance $\varepsilon$ is different from 0, however small, the grid is coarsened at initial time. Some global coefficients of the implicit system of (2.14)–(2.16) (for instance coefficient $a$ in equations (2.18)) have different values when computed on the adaptive grid. The discrepancy is amplified in the LTS case where the coefficients computed on this initial coarse grid are used during a full macro time-step. This phenomenon disappears when there is no constant part in the solution like in the test case proposed in [10].
The total relative errors with respect to the exact solution are computed using in fact the reference solution on a fine grid, corresponding to the 9th level of discretization. They are displayed on Figure 18. As in the explicit case, they are bounded below by the discretization error between the uniform solution on the 6th level and the reference solution. The optimal value for the threshold tolerance $\varepsilon$ could be deduced from the error curve on the gas mass fraction as the value leading to a perturbation error comparable to the discretization error. In our case, this would be somewhere between $10^{-4}$ and $10^{-2}$.

Figure 19 displays the same relative perturbation errors versus the gain in CPU time. For the best error level that can be achieved with the LTS algorithms we have a CPU gain of more than 17, while the gain using the standard multiresolution is around 11.

Eventually we display, in Figure 20, the total error between the adaptive solution and the exact solution. The abscissa is the gain in CPU with respect to the uniform solution on the finest level. As expected, both schemes MR and LTS converge to the error level corresponding to the error $E_u$ between the uniform solution on the finest level, the 5th or the 6th, and the reference solution on the 9th level. Compared to the similar curves with the explicit scheme in Figure 11, the LTS enhancement seems to be more advantageous, knowing that the gas mass fraction is the only component where precision really matters: all values of threshold tried on the 6-level hierarchy lead to a relative error below that obtained with the uniform scheme on the 5th level.

In order to compare all the schemes tested in this study, we collect in Table 5.1 the performances of the six-level simulations, in term of computing times, for both the explicit and implicit schemes. The numbers in parentheses are the ratio between the uniform and the adaptive performances. The gain is significant between the uniform and adaptive scheme: 9 in the explicit case and 6 in the implicit one. With the LTS enhancement it becomes spectacular: 15 in both case. The number of calls to the equilibrium state laws is also reported in the table, along with the computing times. These figures indicate the potential performance of the algorithms when realistic thermodynamical closure laws will be used—instead of the model one (4.6). First tests in the explicit case [11] show that the correlation between the CPU and the number of calls to closure laws is very good when using realistic laws.

Comparing Figures 10 and 19 shows that the LTS enhancement in terms of CPU compared to both the MR and uniform schemes is larger in the explicit case than in the implicit one. Of course this observation made on a given test case can only be qualitatively generalized, considering that the overall complexity of the adaptive methods is higher in the implicit case than in the explicit one. One should also note that for the test case presented in this table, we have used for the explicit scheme a constant time-step $\Delta t = 0.0048$ and for the implicit one $\Delta t = 0.0325$ that is roughly 7 times larger. These values are the minimum values provided by the stability conditions.
(2.33) throughout a uniform computation. The larger time-step in the implicit case is counterbalanced by the cost of solving the linear systems (2.14)–(2.16), and both schemes end up costing roughly the same in terms of computing times in the uniform case. For reasons described at length in [12], in the implicit case, letting the time-step vary with the solution allows us to use larger time-steps as the solution gets smoother. This feature is currently being implemented within the LTS framework for realistic test cases including time-varying boundary conditions.

6. Conclusion. We have described in detail several algorithms to compute the solution of a one-dimensional multiphase flow, in the framework of a Lagrange–Euler projection formulation of the equations. Both explicit and semi-implicit schemes are presented, and for each one, two adaptive enhancement methods are described. The first one is the standard mutiresolution method already implemented in more complicated cases. It is tested here versus the LTS algorithm.

The robustness of the two algorithms has been checked by doing a parameter study. Several values have been tried of the threshold parameter $\varepsilon$ and for the number of levels. The relative errors between the different adaptive solutions and the uniform one present the expected behaviors. The benefits of the LTS enhancement in terms of computing time are very encouraging both the explicit and implicit case.

Acknowledgments. The authors wish to thank S. Müller and S. M. Kaber for fruitful discussions on the redaction of this paper. The authors also thank the referees for their numerous and constructive remarks.

REFERENCES

[11] F. Coquel, Q.-L. Nguyen, M. Postel, and Q.-H. Tran, Performances de la multirésolution avec ou sans pas de temps local sur des cas tests réalisistes d’écoulements diphasiques, in


