Journal of Scientific Computing, Vol. 30, No. 3, March 2007 (© 2006)
DOI: 10.1007/s10915-006-9102-z

Fully Adaptive Multiscale Schemes for Conservation Laws Employing Locally Varying Time Stepping

Siegfried Müller[1] and Youssef Stiriba[2]

Received May 4, 2004; accepted (in revised form) May 9, 2006; Published online June 16, 2006

In recent years the concept of fully adaptive multiscale finite volume schemes for conservation laws has been developed and analytically investigated. Here the grid adaptation is performed by means of a multiscale analysis. So far, all cells are evolved in time using the same time step size. In the present work this concept is extended incorporating locally varying time stepping. A general strategy is presented for explicit as well as implicit time discretization. The efficiency and the accuracy of the proposed concept is verified numerically.

KEY WORDS: Multiscale techniques; local grid refinement; locally varying time stepping; finite volume schemes; conservation laws.

1. INTRODUCTION

The solution of hyperbolic conservation laws typically exhibits locally steep gradients and large regions where it is smooth. To account for the highly nonuniform spatial behavior, we need numerical schemes that adequately resolve the different scales, i.e., use a high resolution only near sharp transition regions and singularities but a moderate resolution in regions with smooth, slowly varying behavior of the solution.

For this purpose, numerical schemes have been discussed or are under current investigation that aim at adapting the *spatial* grid to the local behavior of the flow field. A standard strategy is to base local mesh refinements on *local indicators* which are typically related to gradients in the flow field (see [9, 11]), or local residuals (see [31, 42, 43]). Although

[1] Institut für Geometrie und Praktische Mathematik, RWTH Aachen, 52056 Aachen, Germany. E-mail: mueller@igpm.rwth-aachen.de
[2] Departament Enginyeria Mecànica, Universitat "Rovira i Virgili" - ETSEQ, Av. Paisos Catalans, 26, 43007 Tarragona, Spain. E-mail: youssef.stiriba@urv.net

these concepts turn out to be very efficient in practice they offer no reliable error control. For this purpose, *a posteriori error estimates* have been derived which aim at equilibrating local errors (see [33]).

In the early 1990s Harten [28] proposed to use *multiresolution techniques* in the context of hyperbolic conservation laws. He employed these techniques to transform the arrays of cell averages associated with any given finite volume discretization into a different format that reveals insight into the local behavior of the solution. The cell averages on a given highest level of resolution (reference mesh) are represented as cell averages on some coarse level where the fine scale information is encoded in arrays of *detail coefficients* of ascending resolution. By means of the multiresolution analysis the flux evaluation is controlled, i.e., cheap finite differences are employed in regions where the solution is smooth. By this strategy the computation is accelerated and the solution remains within the same accuracy as the *reference scheme*, i.e., the scheme on the finest computational mesh that uses the expensive flux evaluation throughout the entire domain. However, since one works still on a uniform mesh the computational complexity stays proportional to the number of cells on the finest grid. So far, Harten's concept has been successfully implemented for two-dimensional Cartesian meshes [12, 13, 18, 19, 40], curvilinear meshes [23], and unstructured meshes [1, 14, 20].

Parallel to Harten's original idea a modified approach has been developed by Müller and co-authors [21, 27, 37] that is aiming at reducing the computational costs with regard to both computational time *and* memory requirements but still maintaining the accuracy of the reference scheme. In contrast to this, the *detail coefficients* will be used here to create *locally refined meshes* on which the discretization is performed. Of course, the crux in this context is to arrange this procedure in such a way that at no stage of the computation there is ever made use of the fully refined uniform mesh. A central mathematical problem is then to show that the solution on the adapted mesh is of the same accuracy as the solution on the reference mesh. By now the fully adaptive multiscale concept has been applied by several groups with great success to different real-world applications, e.g., 2D/3D–steady state computations of compressible fluid flow around air wings modeled by the Euler and Navier–Stokes equations, respectively, as well as fluid-structure interactions on block-structured curvilinear grid patches [15, 16], nonstationary wave interactions in two-phase fluids on 2D Cartesian grids for Euler equations [2, 3, 24, 37], backward–facing step on 2D triangulations [22] and simulation of a flame ball modeled by reaction–diffusion equations on 3D Cartesian grids [41].

A short-coming of this approach is the lack of *temporal adaptivity*, i.e., all cell averages are evolved in time by the same time step size τ. For reasons of stability we are therefore obliged to choose τ such that the CFL condition for the cells on the *finest* mesh is satisfied. However, for cells corresponding to a coarser discretization we may use a larger time step to meet the local CFL condition. Therefore, it is natural to use locally varying time stepping.

First results on *local time stepping* have been published by Osher and Sanders [39] for one-dimensional scalar conservation laws. Here the space discretization is fixed but nonuniform. Each element is evolved in time either by an entire time step or a fixed number of smaller time steps. They thoroughly analyzed a first-order spatial discretization with a local forward Euler time stepping scheme. This work has been extended in [25] where a maximum principle was proven for a local forward Euler method when limited slopes are included. Moreover, they showed that the main ideas may be extended to second-order in time by TVD Runge–Kutta methods. Recently, similar ideas were considered in [44] for hyperbolic conservation laws where the solution increment is projected at each local time step.

About the same time, Berger and Oliger [11] proposed the by now classical Adaptive Mesh Refinement (AMR) technique. Here refined grids are laid over regions of the coarse mesh. In particular, the grids need not to be nested but can have a different orientation than the coarse grid. This allows for a local alignment of the grid with anisotropic effects such as shocks. Each refinement level is propagated with its own time step. Information is passed between the grids using injection and interpolation techniques. This approach has been investigated in a series of papers and applied to multidimensional hyperbolic systems of conservation laws (see [6–10]).

In the present work, we are now concerned with the modification of the adaptive multiscale scheme such that we may evolve cell averages on level l by its own level-dependent time step size τ_l. After one time step, the new data then correspond to different times. This procedure is adequate for steady-state problems. Due to this nonuniformity of the temporal propagation front this is no longer admissible for instationary problems because this would result in wrong shock positions. In this case, we have to synchronize the coarse and fine grid solution to obtain an overall conservative scheme. This subject was intensively investigated by Berger et al. in several papers (see [7, 8, 11]). In the context of adaptive multiscale finite volume scheme this has to be adjusted to the requirement that the resulting scheme provides a spatial accuracy that is comparable to the spatial accuracy of the reference mesh. This point of view is similar to the

method presented in [45] for second-order partial differential equations, which exhibit a smooth solution.

Opposite to the AMR framework (cf. [11]), we will successively propagate the data in time starting on the *finest* refinement level instead of the *coarsest*. The synchronization then takes place after having performed one time step on the coarsest level which is referred to as the *macro time step*. If we use a high number of refinement levels, a shock may have a large range of influence within one macro time step. To resolve the shock adequately we either have to refine *a priorly* a large region on the finest level or we have to perform grid refinement on sublevels to track the shock position within one macro time step. The latter strategy is preferable because the overhead for the grid adaptation on the intermediate time levels is by far compensated by the reduced number of flux evaluations on the finest grid.

Since the underlying fully adaptive multiscale concept can be applied to multidimensional scalar and systems of conservation laws based on an explicit or implicit reference finite volume scheme, the strategy of incorporating locally varying time stepping can be applied to all of these problems as well. This makes our strategy a general concept.

The outline of the paper is as follows. In Sec. 2, we start with a summary of the standard fully adaptive multiscale concept recalling its core ingredients, namely, the multiscale analysis and the local grid adaptation. In Sec. 3, we then outline the concept for incorporating locally varying time stepping. Here we first consider an explicit time integration. The main ingredients are (i) a conservation–preserving flux evaluation near interface points, (ii) the computation of appropriate prediction values on coarser levels, (iii) the synchronization of the time evolution, and (iv) the local grid refinement on the intermediate time levels to track appropriately the movement of discontinuities. These ideas are then extended to an implicit time integration. Numerical results verify the efficiency and the accuracy of our method (see Sec. 4). Note that all concepts are presented for one-dimensional problems only to simplify the presentation. However, the concepts also work for multidimensional problems as has been verified for 2D Euler and shallow water computations (cf. [34, 35, 38]). We conclude with some remarks on open questions and future work.

2. FULLY ADAPTIVE MULTISCALE SCHEMES

We briefly summarize the fully adaptive multiscale finite volume scheme for conservation laws. To simplify the presentation of the basic ideas we confine to the 1D case. The multidimensional case is presented in detail in [37]. For this purpose we consider a scalar conservation law

$$u_t(t,x) + (f(u(t,x)))_x = 0, \quad t > 0, \quad x \in \mathbb{R}, \tag{1}$$

which is governed by the initial data

$$u(0,x) = u_0(x), \quad x \in \mathbb{R}. \tag{2}$$

Note that in case of a bounded computational domain we additionally have to supply boundary conditions as well. Since these will cause no conceptual limitation in the design of the multiscale scheme we confine to initial value problems. A conservative finite volume discretization of this problem can be written in the form

$$v_k^{n+1} + \theta\,\lambda\,B_k^{n+1} = v_k^n - (1-\theta)\,\lambda\,B_k^n, \quad \lambda := \tau/h. \tag{3}$$

Here space and time are uniformly discretized by h and τ, respectively. Note that the time discretization is explicit for $\theta = 0$ and implicit for $0 < \theta \leqslant 1$. Conservation means that the flux balance B_k^n has the form

$$B_k^n := F\left(v_{k-p+1}^n, \ldots, v_{k+p}^n\right) - F\left(v_{k-p}^n, \ldots, v_{k+p-1}^n\right) = F_{k+1} - F_k, \tag{4}$$

where the function $F(u_1, \ldots, u_{2p})$ is the numerical flux function.

To improve the efficiency of the finite volume scheme without loss of accuracy we employ multiresolution techniques. For this purpose, we first recall the basic ideas of the underlying multiscale concept. This is employed to construct a locally refined grid on which finally the time evolution is performed. In Sec. 3, we will see that some steps of the grid adaptation procedure have to be adapted to the needs of locally varying time stepping.

2.1. Multiscale Analysis

A finite volume discretization is typically working on a sequence of cell averages. In order to analyze the local regularity behavior of the data, we employ the concept of biorthogonal wavelets [17]. This approach may be seen as a natural generalization of Harten's discrete framework [4, 5, 29]. For reasons of simplicity only uniform refinements in one space dimension are considered here. Note that the framework presented here is not restricted to this simple configuration but can also be applied to *unstructured* grids and *irregular* grid refinements. Details can be found in [37].

Grid hierarchy. Let be $\mathcal{G}_l := \{V_{l,k}\}_{k \in I_l}, l \in \mathbb{N}_0, I_l = \mathbb{Z}$, a sequence of different grids corresponding to different resolution levels. These meshes are composed of the intervals $V_{l,k} = [x_{l,k}, x_{l,k+1}]$ determined by the grid

points $x_{l,k} = 2^{-l}k, k \in \mathbb{Z}$. We note that with increasing refinement level l the interval length $h_l = 2^{-l}$ becomes smaller. Obviously, the resulting grid hierarchy is *nested*, i.e., $\mathcal{G}_l \subset \mathcal{G}_{l+1}$, because of the subdivision condition

$$V_{l,k} = V_{l+1,2k} \cup V_{l+1,2k+1}, \quad \forall l \in \mathbb{N}_0, \quad k \in \mathbb{Z}. \tag{5}$$

Box function and cell averages. Relative to the partitions $\mathcal{G}_l$ we introduce the so-called *box function*

$$\tilde{\phi}_{l,k}(x) := \frac{1}{|V_{l,k}|}\chi_{V_{l,k}}(x) = \begin{cases} 2^l, & x \in V_{l,k}, \\ 0, & x \notin V_{l,k} \end{cases} \tag{6}$$

defined as the L^1–scaled characteristic function with respect to $V_{l,k}$. By $|V|$ we denote the volume of a cell V. Then the averages of a scalar, integrable function $u \in L^1(\Omega)$ can be interpreted as an inner product, i.e.,

$$\hat{u}_{l,k} := \langle u, \tilde{\phi}_{l,k} \rangle_\Omega \qquad \text{with} \quad \langle u, v \rangle_\Omega := \int_\Omega u\, v\, dx. \tag{7}$$

Obviously the nestedness of the grids as well as the linearity of integration imply the two–scale relations

$$\tilde{\phi}_{l,k} = \sum_{r \in \mathcal{M}^0_{l,k}} m^{l,0}_{r,k}\, \tilde{\phi}_{l+1,r} \qquad \text{and} \quad \hat{u}_{l,k} = \sum_{r \in \mathcal{M}^0_{l,k}} m^{l,0}_{r,k}\, \hat{u}_{l+1,r}, \tag{8}$$

where the refinement set is defined by $\mathcal{M}^0_{l,k} := \{2k, 2k+1\} \subset I_{l+1}$ and the mask coefficients turn out to be $m^{l,0}_{r,k} := |V_{l+1,r}|/|V_{l,k}| = 0.5$.

Wavelets and details. In order to detect singularities of the solution, we consider the difference of the cell averages corresponding to different resolution levels. For this purpose, we introduce the wavelet functions $\tilde{\psi}_{l,k}$ as linear combinations of the box functions, i.e.,

$$\tilde{\psi}_{l,k} := \sum_{r \in \mathcal{M}^1_{l,k} \subset I_{l+1}} m^{l,1}_{r,k}\, \tilde{\phi}_{l+1,r}. \tag{9}$$

The construction of the wavelets is subject to certain constraints, namely, the wavelet functions $\Psi_l := (\psi_{l,k})_{k \in I_l}$ build a completion of the basis system $\Phi_l := (\phi_{l,k})_{k \in I_l}$, they are locally supported, provide vanishing moments and there exists a biorthogonal system. For details, we refer to the *concept of stable completions* (see [17]). Then we can perform a change of basis between $\Phi_l \cup \Psi_l$ and Φ_{l+1}, i.e.,

$$\tilde{\phi}_{l+1,k} = \sum_{r \in \mathcal{G}^0_{l,k} \subset I_l} g^{l,0}_{r,k}\, \tilde{\phi}_{l,r} + \sum_{r \in \mathcal{G}^1_{l,k} \subset I_l} g^{l,1}_{r,k}\, \tilde{\psi}_{l,r}. \tag{10}$$

By means of the wavelet functions we introduce the *detail coefficients*

$$d_{l,k} := \langle u, \tilde{\psi}_{l,k} \rangle_\Omega. \tag{11}$$

These coefficients inherit the two-scale relation

$$d_{l,k} = \sum_{r \in \mathcal{M}^1_{l,k}} m^{l,1}_{r,k}\, \hat{u}_{l+1,r} \tag{12}$$

from its functional counterpart (9).

Multiscale transformation. The ultimate goal is to transform the array of cell averages $\boldsymbol{u}_L := (\hat{u}_{L,k})_{k \in I_L}$ corresponding to a finest uniform discretization level into a sequence of coarse grid data $\boldsymbol{u}_0 := (\hat{u}_{0,k})_{k \in I_0}$ and details $\boldsymbol{d}_l := (d_{l,k})_{k \in I_l}, l = 0, \ldots, L-1$, representing the successive update from a coarse resolution to a high resolution. According to (8) and (9), we obtain two-scale relations for the coefficients inherited from the two-scale relations of the box functions and the wavelet functions

$$\hat{u}_{l,k} = \sum_{r \in \mathcal{M}^0_{l,k}} m^{l,0}_{r,k}\, \hat{u}_{l+1,r}, \qquad d_{l,k} = \sum_{r \in \mathcal{M}^1_{l,k}} m^{l,1}_{r,k}\, \hat{u}_{l+1,r} \tag{13}$$

and

$$\hat{u}_{l+1,k} = \sum_{r \in \mathcal{G}^0_{l,k}} g^{l,0}_{r,k}\, \hat{u}_{l,r} + \sum_{r \in \mathcal{G}^1_{l,k}} g^{l,1}_{r,k}\, d_{l,r}. \tag{14}$$

Applying the relations (13) iteratively the array $\hat{\boldsymbol{u}}_L$ is successively decomposed. We refer to this transformation as *multiscale transformation*. It is reversed by the *inverse multiscale transformation* (14).

Cancellation property. It can be shown that the details become small with increasing refinement level when the underlying function is smooth

$$|d_{l,k}| \leqslant C\, 2^{-l\,M}\, \|u^{(M)}\|_{L^\infty(V_{l,k})}. \tag{15}$$

Obviously, the details decay with a rate at least of $2^{-l\,M}$ provided the function u is differentiable and the wavelets have M vanishing moments. This motivates to neglect all sufficiently small details in order to compress the original data.

Example Finally we give an example for the above multiscale setting in case of a dyadic grid refinement of the real axis, i.e., $\Omega = \mathbb{R}$ and $I_l = \mathbb{Z}$ for $l = 0, \ldots, L$. This simplifies the computation of the wavelets because

the mask coefficients are independent of the level and the position. Otherwise we have to modify the wavelet construction near boundaries and ensure that the support is fully contained in the flow field. Following the wavelet construction with

$$M = 2s + 1 \tag{16}$$

vanishing moments presented in [36, Secs. 2.5.2 and 3.8], we obtain for the mask coefficients

$$g^{l,0}_{k,2j+i} = \begin{cases} \delta_{k,j} - (-1)^i l_{k-j+s}, & k \in \{j-s, \dots, j+s\}, \\ 0, & \text{elsewhere}, \end{cases}$$

$$g^{l,1}_{k,2j+i} = \begin{cases} (-1)^i, & k = j, \\ 0, & \text{elsewhere}, \end{cases}$$

$$m^{l,0}_{2j+i,k} = \begin{cases} 0.5, & k = j, \\ 0, & \text{elsewhere}, \end{cases}$$

$$m^{l,1}_{2j+i,k} = \begin{cases} 0.5\,(l_{j-k+s} + (-1)^i \delta_{k,j}), & l \in \{j-s, \dots, j+s\}, \\ 0, & \text{elsewhere} \end{cases}$$

for $k, j \in I_l$ and $i \in \{0, 1\}$. The coefficients $l_i, i = 0, \dots, s$, are summarized in Table I. Hence the corresponding index sets of nonvanishing entries are determined by

$$\mathcal{M}^0_{l,k} = \{2k, 2k+1\}, \quad \mathcal{M}^1_{l,k} = \{2(k-s), \dots, 2(k+s)+1\}, \tag{17}$$

$$\mathcal{G}^0_{l,k} = \{\lfloor k/2 \rfloor - s, \dots, \lfloor k/2 \rfloor + s\}, \quad \mathcal{G}^1_{l,k} = \{\lfloor k/2 \rfloor\}. \tag{18}$$

2.2. Local Grid Adaptation

By means of the details, we now determine a locally refined grid. Since the grid adaptation tool is supposed to dynamically adapt the mesh to an underlying flow field, we start with data corresponding to a certain time step n. At this time step, the locally refined grid is characterized by the index set $\mathcal{G}^n_{L,\epsilon} \subset \{(l,k)\,;\, k \in I_l,\ l = 0, \dots, L\}$, i.e., $\Omega = \cup_{(l,k) \in \mathcal{G}^n_{L,\epsilon}} V_{l,k}$. It

Table I. Coefficients

s	l_0	l_1	l_2	l_3	l_4
0	0				
1	−1/8	0	1/8		
2	3/128	−11/64	0	11/64	−3/128

is required that the set $\mathcal{G}^n_{L,\epsilon}$ has the structure of a graded tree, i.e., neighboring cells differ at most by one level of refinement. The grid is provided with cell averages $\{\hat{u}^n_{l,k}\}_{(l,k)\in\mathcal{G}^n_{L,\epsilon}}$. Then the grid adaptation procedure consists of six steps. Note that it can be realized with an optimal complexity, i.e., the number of floating point operations is proportional to the number of cells in the adaptive grid. In particular, we never access to the *finest* mesh.

Local multiscale transformation. In a first step, we perform a multiscale analysis of the data at hand. For this purpose, we proceed level by level from fine to coarse according to (13). Note that the two-scale transformation is performed locally only for the indices corresponding to the adaptive grid instead of the full levels. In particular, applying the local two-scale transformation can be interpreted as a successive coarsening of the grid where fine-grid cells are agglomerated to a coarse-grid cell and the difference information is stored by the detail coefficients.

Thresholding. The idea is simply to discard all coefficients $d^n_{l,k}$ that fall in absolute value below a certain threshold. For this purpose, we introduce the index set

$$\mathcal{D}^n_{L,\epsilon} := \left\{(l,k)\ ;\ |d^n_{l,k}| > \varepsilon_l,\ \ k\in I_l,\ \ l\in\{0,\ldots,L-1\}\right\}$$

corresponding to what will be referred to as *significant details*. Here $\varepsilon_l = 2^{l-L}\varepsilon$ is a level–dependent threshold value which is smaller on coarser levels. The choice of the threshold parameter ε is discussed in [36].

Prediction of significant details. To perform the evolution step, we have to determine the adaptive grid on the *new* time level. Since the corresponding averages, respectively, details are not yet available, we have to *predict* all details on the new time level $n+1$ that may become significant due to the evolution by means of the details on the *old* time level n. In order to guarantee, the adaptive scheme to be *reliable* in the sense that no significant future feature of the solution is missed, the prediction set $\tilde{\mathcal{D}}^{n+1}_{L,\epsilon}$ has to satisfy

$$\mathcal{D}^n_{L,\epsilon} \cup \mathcal{D}^{n+1}_{L,\epsilon} \subset \tilde{\mathcal{D}}^{n+1}_{L,\epsilon}, \tag{19}$$

where, of course $\mathcal{D}^{n+1}_{L,\epsilon}$ is not known at the old time level. In [28], Harten suggests a heuristic approach that could not be rigorously verified to satisfy (19). However, in [21], a slight modification of Harten's prediction strategy has been shown to lead to a reliable prediction strategy in the sense of (19).

Grading. In order to perform the grid adaptation procedure *level by level* we need that the index set of significant details corresponds to a

graded tree, i.e., for any significant detail on level l there are significant details in the neighborhood but on the next lower level

$$(l,k)\in\mathcal{D}_{L,\epsilon} \quad \Rightarrow \quad (l-1,r)\in\mathcal{D}_{L,\epsilon} \quad \forall r\in\{\lfloor k/2\rfloor - q,\ldots,\lfloor k/2\rfloor + q\}. \tag{20}$$

Here the parameter q depends on the number s related to the number of vanishing moments (16) and on the stencil width p of the numerical flux (4) (cf. [36, p. 68, 70]). In particular, this implies that the levels of neighboring cells differ at most by one. Since the sets $\mathcal{D}^n_{L,\epsilon}$ and $\mathcal{D}^{n+1}_{L,\epsilon}$, respectively, are in general not graded, we have to apply in addition a grading procedure. This will slightly inflate the index set of significant details but has so far been observed not to spoil the complexity reduction of floating point operations in any significant way.

Grid adaptation. Then we exploit the inflated set $\tilde{\mathcal{D}}^{n+1}_{L,\epsilon}$ to determine an associated index set $\mathcal{G}^{n+1}_{L,\epsilon}$, which characterizes the adaptive grid at the new time level. The index set $\mathcal{G}^{n+1}_{L,\epsilon}$ is initialized by all indices of the coarsest discretization. Then, traversing through the levels from coarse to fine we proceed as follows: if $(l,k)\in\tilde{\mathcal{D}}^{n+1}_{L,\epsilon}$ then the cell $V_{l,k}$ is locally refined, i.e., the index (l,k) is removed from $\mathcal{G}^{n+1}_{L,\epsilon}$ and the indices of the subcells on the finer level are added to $\mathcal{G}^{n+1}_{L,\epsilon}$. Finally we obtain the locally adapted grid which naturally corresponds to the leaves of the graded tree of significant details.

Local inverse multiscale transformation By the previous step, the grid has locally changed due to local refinement and coarsening. In order to determine the cell averages $\{\hat{u}^n_{l,k}\}_{(l,k)\in\mathcal{G}^{n+1}_{L,\epsilon}}$, we employ a local inverse multiscale transformation according to (14) interrelating the local cell averages $\{\hat{u}^n_{l,k}\}_{(l,k)\in\mathcal{G}^n_{L,\varepsilon}}$ and the significant details $\{d^n_{l,k}\}_{(l,k)\in\tilde{\mathcal{D}}^{n+1}_{L,\varepsilon}}$. Again we proceed level by level from coarse to fine where we locally replace a cell average on the coarse scale by the cell averages of its subcells. This is done whenever there is a significant detail associated to this coarse cell in $\tilde{\mathcal{D}}^{n+1}_{L,\epsilon}$.

2.3. Application to Reference Finite Volume Scheme

Finally we have to present the time evolution on the locally refined grid. Note that the ultimate goal is to obtain an efficient algorithm that is as accurate as the reference scheme (3) performed on the uniform finest grid. Here the crucial point is the flux computation on the adaptive grid. The basic idea is to apply the multiscale transformation to the reference scheme.

$$v^{n+1}_{L,k} + \theta\,\lambda_L\,B^{n+1}_{L,k} = v^n_{L,k} - (1-\theta)\,\lambda_L\,B^n_{L,k}, \quad \lambda_L := \frac{\tau}{h_L}, \quad k\in I_L. \tag{21}$$

Here $v^n_{L,k}$ denote the numerical approximations at time t_n. The flux balances $B^n_{L,k}$ are defined according to (4).

Then we introduce the cell averages $v^n_{l,k}$ on the coarser scales $l=0,\dots,L-1$

$$v^n_{l,k} := \sum_{r\in\mathcal{M}^0_{l,k}} m^{l,0}_{r,k}\, v^n_{l+1,r} = \frac{1}{2}\left(v^n_{l+1,2k} + v^n_{l+1,2k+1}\right) \tag{22}$$

for an arbitrary time level t_n according to (8).

Applying the multiscale transformation (13), we obtain discrete evolution equations for these cell averages

$$v^{n+1}_{l,k} + \theta\,\lambda_l\, B^{n+1}_{l,k} = v^n_{l,k} - (1-\theta)\,\lambda_l\, B^n_{l,k}, \quad \lambda_l := \frac{\tau}{h_l}. \tag{23}$$

Here the local flux balances $B^n_{l,k}$ are recursively defined by

$$B^n_{l,k} := \sum_{r\in\mathcal{M}^0_{l,k}} \frac{|V_{l,k}|}{|V_{l+1,r}|}\, m^{l,0}_{r,k}\, B^n_{l+1,r} = \sum_{r\in\mathcal{M}^0_{l,k}} B^n_{l+1,r} = F^n_{l,k+1} - F^n_{l,k}, \tag{24}$$

where we employ (4) and (8). Due to the nestedness of the grids (5) the numerical fluxes on level l coincide with the numerical fluxes on the higher scales, i.e.,

$$F^n_{l,k} = F^n_{l+1,2k} = \cdots = F^n_{L,2^{L-l}k} \equiv F\left(v^n_{L,2^{L-l}k-p},\dots,v^n_{L,2^{L-l}k+p-1}\right). \tag{25}$$

Since the solution is only available on the adaptive grid, this formula can not be directly applied for the flux evaluation. For this purpose, the data on the finest level have to be reconstructed locally from the coarse-scale information. In case of dyadic grid refinement where the mask coefficients do not depend on level and position one might compute *a priorly* constant transfer operators by which the reconstruction can be performed very efficiently (cf. [20]). For non-dyadic grid refinements or curvilinear grid patches, this is no longer feasible. Furthermore, formula (25) only holds in the one-dimensional case. For multidimensional problems, hanging nodes occur in the locally refined grid and the fluxes on different resolution levels do not coincide. Then the local fluxes are defined by the sum of all fluxes on the higher level whose cell interfaces build a composition of the local cell interface.

Since the numerical divergence on the coarser levels is recursively defined by (24) we further conclude

$$B^n_{l,k} = \sum_{i=0}^{2^{L-l}-1} B^n_{L,2^{L-l}k+i} = F^n_{L,2^{L-l}(k+1)} - F^n_{L,2^{L-l}k} \equiv F^n_{l,k+1} - F^n_{l,k}, \tag{26}$$

i.e., the local numerical divergence is determined by the fluxes on the *finest* scale. Then the fully adaptive scheme in 1D reads

$$v_{l,k}^{n+1} + \theta\, \lambda_l\, B_{l,k}^{n+1} = v_{l,k}^{n} - (1-\theta)\, \lambda_l\, B_{l,k}^{n}, \quad (l,k) \in \mathcal{G}_{L,\epsilon}^{n+1}, \tag{27}$$

where the flux balances $B_{l,k}^{n}$ are determined by (26) and (25).

3. LOCAL TIME STEPPING

So far no local time stepping is incorporated in the fully adaptive multiscale scheme as summarized in Sec. 2.3. As we conclude from (23) all cell averages are evolved in time by the same time step size τ. Therefore, τ has to be chosen such that the CFL condition for the cells on the *finest* mesh holds. Note that for cells on the coarser scales we may use $\tau_l = 2^{L-l}\,\tau, l = 0, \ldots, L-1$, to satisfy locally the CFL condition. In the sequel, we explain how to modify the adaptive multiscale scheme such that we may evolve cell averages on level l by its own time step size τ_l. For this purpose, we first consider the explicit scheme before extending the ideas to the implicit scheme.

3.1. Explicit Local Time Stepping

The main issues arising in local time stepping concern (i) the conservative flux computation at *interface points*, i.e., grid points where the neighboring cells are located on different refinement levels, (ii) the computation of a prediction value for the flux computation at intermediate time levels near interface points, (iii) the synchronization of time evolution for cells on different levels that are propagated with level-dependent time stepping, and (iv) the prediction of an appropriate adaptive grid when performing intermediate time steps on higher levels.

3.1.1. Conservation–Preserving Flux Computation

Propagating each cell with its own time step size leads to a nonsynchronization of the flow field. For instationary problems we therefore have to synchronize the coarse and fine grid solution at certain time levels. According to the definition of τ_l the data can be synchronized naturally at the times corresponding to the coarsest discretization τ_0. In the context of multiscale schemes, the flux synchronization problem is directly related to the conservation property of the finite volume scheme, i.e., at each interface point only one flux is computed for both of the adjacent cells.

For this purpose, we investigate the correlation of the flux balances at interface points in some detail.

For reasons of simplicity, we consider now only two refinement levels to outline the basic ideas, i.e., a fine grid (level $l+1$) and a coarse grid (level l). This situation is sketched in Fig. 1. Then the extension to the multilevel case can be performed recursively (see Sec. 3.1.3). Let us assume that we know the data on level $l+1$ at time t_n. We now perform two time steps according to (23) where we apply our reference scheme on level $l+1$ for the cells $r \in \mathcal{M}_{l,k}^0$ with time step size $\tau \equiv \tau_{l+1} = 2\,\tau_l$, i.e.,

$$v_{l+1,r}^{n+1/2} = v_{l+1,r}^{n} - \overline{\lambda}_{l+1}\, B_{l+1,r}^{n}, \tag{28}$$

$$v_{l+1,r}^{n+1} = v_{l+1,r}^{n+1/2} - \overline{\lambda}_{l+1}\, B_{l+1,r}^{n+1/2} \tag{29}$$

with

$$\overline{\lambda}_{l+1} := \frac{\tau_{l+1}}{h_{l+1}} = \frac{0.5\,\tau_l}{0.5\,h_l} = \overline{\lambda}_l. \tag{30}$$

Here the flux balances $B_{l+1,r}^{n}$ and $B_{l+1,r}^{n+1/2}$ are computed according to (26) by means of the data $v_{l+1,r}^{n}$ and $v_{l+1,r}^{n+1/2}$ corresponding to time t_n and $t_{n+1/2} = t_n + \tau_{l+1}$, respectively. These correspond to the numerical fluxes indicated by $\circ$ in Fig. 1. We now replace $v_{l+1,r}^{n+1/2}$ on the right-hand side of (29) by (28). Hence (29) can be rewritten as

$$v_{l+1,r}^{n+1} = v_{l+1,r}^{n} - \overline{\lambda}_{l+1} \left(B_{l+1,r}^{n+1/2} + B_{l+1,r}^{n} \right). \tag{31}$$

Again we apply to (31) the multiscale transformation (13) and obtain the discrete evolution equations for cell averages on the coarser level

$$v_{l,k}^{n+1} = v_{l,k}^{n} - \overline{\lambda}_l\, \overline{B}_{l,k}^{n}, \tag{32}$$

where the local flux balance $\overline{B}_{l,k}^{n}$ is recursively defined by

$$\overline{B}_{l,k}^{n} := \sum_{r \in \mathcal{M}_{l,k}^0} m_{r,k}^{l,0} \left(B_{l+1,r}^{n+1/2} + B_{l+1,r}^{n} \right) = \overline{F}_{l,k+1}^{n} - \overline{F}_{l,k}^{n} \tag{33}$$

with the numerical fluxes

$$\overline{F}_{l,k}^{n} := \frac{1}{2} \left(F_{l+1,2k}^{n+1/2} + F_{l+1,2k}^{n} \right). \tag{34}$$

Here we use (8) and (26). These fluxes correspond to $\bullet$ in Fig. 1.

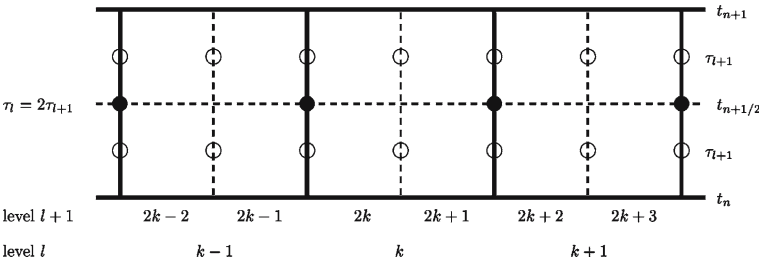

Fig. 1. Two-scale grid in space and time.

In principle, the multiscale scheme tells us how to determine the flux balances and numerical fluxes on coarser levels. As we conclude from (33) and (34) they are determined by the average of their counterparts on the finer level. This suggests to proceed from *fine* to *coarse* when propagating the cell averages on different scales. At interface points we then have to compute the flux balances and numerical fluxes by the information already determined on the higher level. This situation is sketched locally in Fig. 2.

3.1.2. Prediction Value

When computing the numerical flux $F^{n+1/2}_{l+1,2k}$ according to (25) we access to cell averages at the intermediate time $t_{n+1/2}$. In the literature different approaches have been investigated to provide some *prediction value* for these cells at the intermediate time level.

A naive strategy would be to use the available information on a previous time level, i.e., for the coarse cells (level l) we use the data of the old time step t_n as a prediction whereas for the finer cells (level $l+1$) we can already use the new value at the intermediate time $t_{n+1/2}$ (see Fig. 3). This approach has been considered as a motivating example by Osher and Sanders [39]. Recently, Tang and Warnecke [44] verify that this naive approach is inconsistent at interfaces separating two global domains corresponding to two different resolutions. For this purpose, they apply a

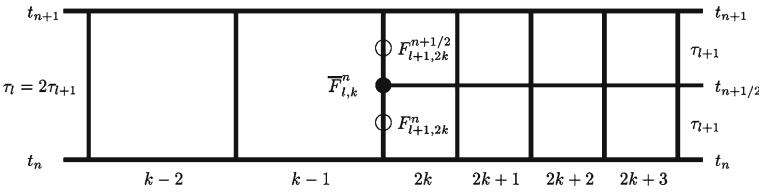

Fig. 2. Locally refined grid in space and time.

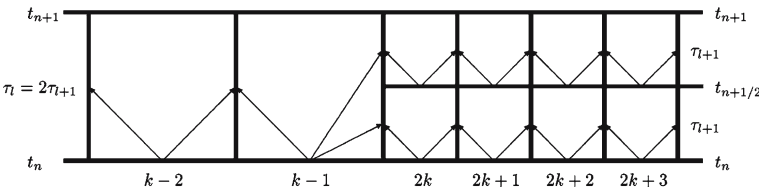

Fig. 3. Flux computation on a locally refined grid in space and time with a naive prediction value.

first-order upwind scheme to a linear advection equation. We also used this approach. But numerical simulations verified that the mixing of time scales caused small perturbations at interface points. These are detected by the multiscale analysis and result in an increase of significant details, i.e., the adaptive grid is locally inflated, and, hence, the efficiency of the computation is degraded.

An alternative strategy has been introduced by Berger and Oliger [11]. They are using interpolation techniques, i.e., first the cells on the coarser level are evolved in time. Then a prediction value is determined at the intermediate time level evaluating an interpolation formulae computed by the values of the old and new time. This results in two numerical fluxes at interface points. To ensure the conservation property of the scheme, a so-called synchronization step is necessary to compensate for the flux difference (see also Sec. 3.1.7).

Since we proceed levelwise from *fine* to *coarse* we use a *predictor-corrector approach* similar to Osher and Sanders [39]. However, we prefer a different representation more suited to an efficient implementation of the resulting algorithm whereas the predictor–corrector representation is preferable for analytical investigations. In Sec. 3.1.6, we will address this in more detail. In the following, we outline the basic ideas by means of the situation sketched in Fig. 4. First of all, we introduce the index sets $\mathcal{C}_j \subset \mathcal{G}_{L,\epsilon}^{n+1}$, $j = l, l+1$, of the cells on level j that can be evolved in time by *one* time step with step size τ_j, i.e., these cells are *not* involved in the flux computation on level $j+1$ at the intermediate time level corresponding to τ_{j+1}. For this purpose, we consider the local flux function (25) with $L = l+1$. Then we perform for each of the data $v_{L,2k+r}^n$, $r = -p, \ldots, p-1$, the local multiscale transformation (14). From the supports (18) we then conclude

$$\mathcal{C}_l = \{(l,i)\,;\, i < \lfloor (2k-p)/2 \rfloor - s\}, \qquad \mathcal{C}_{l+1} = \{(l+1,i)\,;\, i \geqslant 2k\}.$$

Note that on the highest level $(l+1)$ no cells are excluded. Furthermore we introduce set of all cells on level j and the complement sets $\overline{\mathcal{C}}_j$

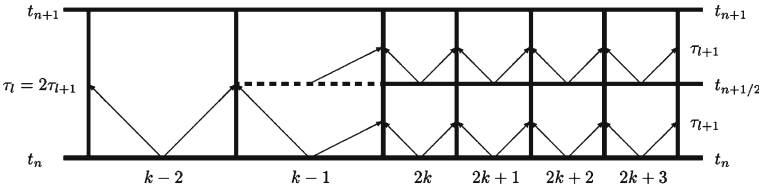

Fig. 4. Flux computation on a locally refined grid in space and time with an accurate prediction value.

of cells on level j not contained in $\mathcal{C}_j$

$$\tilde{I}_j := \left\{ (j,k')\,;\ (j,k') \in \mathcal{G}_{L,\epsilon}^{n+1} \right\}, \qquad \overline{\mathcal{C}}_j := \tilde{I}_j \backslash \mathcal{C}_j .$$

For the particular situation considered here they turn out to be

$$\overline{\mathcal{C}}_l = \{(l,k)\,;\ \lfloor (2k-p)/2 \rfloor - s \leqslant i \leqslant k-1\}, \quad \overline{\mathcal{C}}_{l+1} = \emptyset .$$

Then the time evolution consists of two intermediate steps. In the first step, we evolve all cells in $\mathcal{C}_{l+1}$ by a *full* step with τ_{l+1} and to compute the prediction values for all cells on level l contained in $\overline{\mathcal{C}}_l$ we perform a *half* step also using τ_{l+1}, i.e.,

$$v_{l',k'}^{n+1/2} = v_{l',k'}^{n} - \frac{\tau_{l+1}}{h_{l'}} \left(F_{l',k'+1}^{n} - F_{l',k'}^{n} \right), \quad (l',k') \in \mathcal{C}_{l+1} \cup \overline{\mathcal{C}}_l . \tag{35}$$

In a second step, we then perform a *full* step on level $j = l, l+1$ for all cells in the sets $\mathcal{C}_j$ with τ_j and a *half* step for the prediction values on level l contained in $\overline{\mathcal{C}}_l$, i.e.,

$$\begin{aligned} v_{l',k'}^{n+1} &= v_{l',k'}^{n+(l'-l)/2} - \frac{\tau_{l'}}{h_{l'}} \left(F_{l',k'+1}^{n+(l'-l)/2} - F_{l',k'}^{n+(l'-l)/2} \right), \quad (l',k') \in \mathcal{C}_{l+1} \cup \mathcal{C}_l, \\ v_{l,k'}^{n+1} &= v_{l,k'}^{n+1/2} - \frac{\tau_{l+1}}{h_l} \left(F_{l,k'+1}^{n+1/2} - F_{l,k'}^{n+1/2} \right), \quad (l,k') \in \overline{\mathcal{C}}_l . \end{aligned} \tag{36}$$

Here the numerical fluxes are determined by (25) where we use either the data at time t_n or $t_{n+1/2}$, respectively.

If we plug in (35) into (36) then we may rewrite the time evolution in one (macro) time step corresponding to τ_l as

$$v_{l',k'}^{n+1} = v_{l',k'}^{n} - \frac{\tau_l}{h_{l'}} \left(\overline{F}_{l',k'+1}^{n} - \overline{F}_{l',k'}^{n} \right), \quad (l',k') \in \mathcal{G}_{L,\epsilon}^{n+1} . \tag{37}$$

For the flux computation $\overline{F}_{l',k'}^{n}$, see also (34), we distinguish three cases:

(i) at the fine level ($l' = l + 1$)

$$\overline{F}^n_{l+1,k'} = \frac{1}{2}\left(F^{n+1/2}_{l+1,k'} + F^n_{l+1,k'}\right), \tag{38}$$

(ii) at the coarse level ($l' = l$) away from interface points

$$\overline{F}^n_{l,k'} = F^n_{l,k'}, \tag{39}$$

(iii) at the coarse level ($l' = l$) coinciding with an interface point

$$\overline{F}^n_{l,k'} = \frac{1}{2}\left(F^{n+1/2}_{l+1,2k'} + F^n_{l+1,2k'}\right) = \overline{F}^n_{l+1,2k'}. \tag{40}$$

Finally we note that (35) can be considered a *prediction step* and (36) a *correction step*.

3.1.3. Synchronization of Time Evolution

In the previous sections, we derived a conservation–preserving and accurate strategy for the flux evaluation at interface points. Note that due to the grading of our adaptive grid (see Sec. 2.2), the refinement level of the adjacent cells differs by at most one. Therefore, it has been sufficient to outline the local time stepping strategy for a two-level grid only. We now have to explain how to incorporate this concept into the time evolution of the adaptive grid. The basic idea is to evolve each cell on level l with the level-dependent time discretization $\tau_l = 2^{L-l}\,\tau_L, l = 0, \ldots, L$. Obviously, all cell averages correspond to the same integration time after having performed 2^l time steps with τ_l, i.e., the cells are *synchronized*. This is schematically sketched in Fig. 5. Therefore, one macro time step with $\tau_0 = 2^L\,\tau_L$ consists of 2^L intermediate time steps $t_{n+i\,2^{-L}}, i = 1, \ldots, 2^L$, with step size τ_L. At time $t_{n+i\,2^{-L}}$ the *smallest synchronization level* is determined by

$$l_i := \min\{l\,;\, 0 \leqslant l \leqslant L,\ i \mod 2^{L-l} = 0\},$$

i.e., in this step we have to evolve all cells sitting on the levels $l = l_i, \ldots, L$ to ensure the synchronization of the time evolution.

According to the multiscale analysis the time evolution is performed first for the cells on the highest level and then successively for the coarser levels. By this procedure we ensure that the fluxes at the intermediate time level have already been computed when determining the fluxes (40) at the interface points on the coarser level.

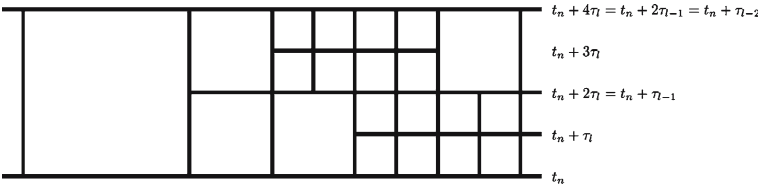

Fig. 5. Synchronization on multilevel grid.

The details of the synchronized time evolution is summarized in the following Algorithms. First of all, we have to initialize the index sets $\mathcal{C}_l, l = 0, \ldots, L$ and the index set of the numerical fluxes $\mathcal{F}$.

Algorithm 1 (Initialization on levels $l = 0, \ldots, L$)
For each macro time step we first have to initialize

(a) the index sets $\mathcal{C}_l, l = 0, \ldots, L$, of cells on level l that can be evolved in time by *one* time step with step size τ_l, i.e.,

$$\mathcal{C}_L := \left\{ (L,k)\,;\ (L,k) \in \tilde{\mathcal{G}}_{L,\epsilon}^{n+1} \right\},$$

$$\mathcal{C}_l := \left\{ (l,k) \in \tilde{\mathcal{G}}_{L,\epsilon}^{n+1}\,;\ \nexists\, (l+1,r) \in \tilde{\mathcal{G}}_{L,\epsilon}^{n+1} :\ k \in \Sigma_r \right\}, \quad l < L$$

with the range of dependence Σ_r determined by the stencil of the flux computation (25) and the local inverse multiscale transformation (14) as

$$\Sigma_r := \{ \lfloor (r-p)/2 \rfloor - s, \ldots, \lfloor (r+p)/2 \rfloor + s \};$$

these cells are *not* involved in the flux computation on level $l+1$ at the intermediate time level corresponding to τ_{l+1};

(b) the complement sets, i.e.,

$$\overline{\mathcal{C}}_l := \tilde{I}_l \backslash \mathcal{C}_l, \qquad \tilde{I}_l := \left\{ (l,k)\,;\ (l,k) \in \mathcal{G}_{L,\epsilon}^{n+1} \right\};$$

(c) the index set $\mathcal{F}$ of all numerical fluxes to be computed for the current adaptive grid, i.e.,

$$\mathcal{F} := \left\{ (l,k)\,;\ (l,k) \in \tilde{\mathcal{G}}_{L,\epsilon}^{n+1} \text{ or } (l,k+1) \in \tilde{\mathcal{G}}_{L,\epsilon}^{n+1} \right\}.$$

After having initialized these sets we may perform the time evolution for each of the intermediate time steps $i = 1, \ldots, 2^L$.

Algorithm 2 (Synchronized time evolution for time step $t_{n+i\,2^{-L}}$)
For each intermediate time step we have to perform the following steps:

(1) Flux computation:

(a) For the levels $l = l_{i-1}, \ldots, L$ we determine the numerical fluxes with respect to the data of the previous intermediate time step corresponding to $t_{n+(i-1)\,2^{-L}}$, i.e.,

(i) no interface point ($(l+1, 2k) \notin \mathcal{F}$)

$$F_{l,k}^{n+(i-1)\,2^{-L}} = F\left(v_{L,2^{L-l}k-p}^{n+(i-1)\,2^{-L}}, \ldots, v_{L,2^{L-l}k+p-1}^{n+(i-1)\,2^{-L}}\right),$$

(ii) interface point ($(l+1, 2k) \in \mathcal{F}$)

$$F_{l,k}^{n+(i-1)\,2^{-L}} = F_{l+1,2k}^{n+(i-1)\,2^{-L}};$$

this procedure has to be performed from *fine* to *coarse*;

(b) For level $l = l_{i-1} - 1$ we have to update the numerical fluxes at the interface points to maintain the conservation property, i.e.,

$$F_{l_{i-1}-1,k}^{n+(i-1)\,2^{-L}} = F_{l_{i-1},2k}^{n+(i-1)\,2^{-L}};$$

(c) For the levels $l = 0, \ldots, l_{i-1} - 1$ the numerical fluxes are unchanged *except* for the interface points on level l_{i-1} (see step 1b), since the cell averages on these levels have not changed in the previous intermediate time step, i.e.,

$$F_{l,k}^{n+(i-1)\,2^{-L}} = F_{l,k}^{n+(i-2)\,2^{-L}};$$

(2) Time Evolution:

(a) For the levels $l = l_i, \ldots, L$ we perform a *full* time step with τ_l for the cells $(l, k) \in \mathcal{C}_l$, i.e.,

$$v_{l,k}^{n+i\,2^{-L}} = v_{l,k}^{n+(i-1)\,2^{-L}} - \frac{\tau_l}{h_l}\left(F_{l,k+1}^{n+(i-1)\,2^{-L}} - F_{l,k}^{n+(i-1)\,2^{-L}}\right);$$

(b) For the levels $l = l_i - 1, \ldots, L$ we perform a *half* time step with $\tau_{l+1} = \tau_l/2$ for the cells $(l, k) \in \overline{\mathcal{C}}_l$, i.e.,

$$v_{l,k}^{n+i\,2^{-L}} = v_{l,k}^{n+(i-1)\,2^{-L}} - \frac{\tau_{l+1}}{h_l}\left(F_{l,k+1}^{n+(i-1)\,2^{-L}} - F_{l,k}^{n+(i-1)\,2^{-L}}\right);$$

(c) For the levels $l = 0, \ldots, l_i - 1$ the cell averages are unchanged *except* for the cells on level l_{i-1} contained in $\overline{C}_{l_i-1}$ (see step 2b), i.e.,

$$v_{l,k}^{n+i\,2^{-L}} = v_{l,k}^{n+(i-1)\,2^{-L}}.$$

3.1.4. Prediction of Details

So far we only considered the correct flux treatment at interfaces of two different discretizations. This affects the correct transport of information and the stability of the approximation. In addition to this, we have to be concerned with the quality of the approximation. According to the fully adaptive multiscale concept presented in Sec. 2.3, we are still aiming at the accuracy provided by the reference scheme (21) on the finest uniform discretization. In particular, after one macro-time step using the local time stepping procedure with $\tau_0 = 2^L \tau_L$ we would like to have as good an approximation as having performed 2^L time steps with the reference scheme using the step size τ_L. Therefore, we have to make sure that the solution is adequately resolved at the old time t_n and the new time step $t_{n+1} = t_n + 2^L \tau_L = t_n + \tau_0$. For the original fully adaptive scheme this is ensured by the prediction step of the grid adaption (see Sec. 2.2). The prediction of the details ensures that a significant information can only move by at most one cell on the *finest* level. However, employing the same strategy for the local time stepping strategy this information could move up to one cell on the *coarsest* mesh. This would result in a completely underresolution of discontinuities on the new time level. To account for this we have to modify the prediction step of the details (19) such that the prediction set $\tilde{\mathcal{D}}_{L,\epsilon}^{n+1}$ satisfies the modified reliability condition

$$\bigcup_{i=0}^{2^L} \mathcal{D}_{L,\epsilon}^{n+i\,2^{-L}} \subset \tilde{\mathcal{D}}_{L,\epsilon}^{n+1}, \tag{41}$$

where the sets $\mathcal{D}_{L,\epsilon}^{n+i\,2^{-L}}$ correspond to the significant details of the solution at times $t_{n+i\,2^{-L}} = t_n + i\,\tau_L, i = 0, \ldots, 2^L$.

One option could be to resolve the *whole* range of influence characterized by the maximal and minimal characteristic speeds, respectively. This can be realized by refining 2^l cells (to the left and the right) in a neighborhood of a significant detail on level l instead of only one cell. Hence, we take into account that an information can move by one *coarse* cell instead of one *fine* cell only. Unfortunately, this results in a tremendous overhead of work on the higher levels as can be concluded from Fig. 6. There, we

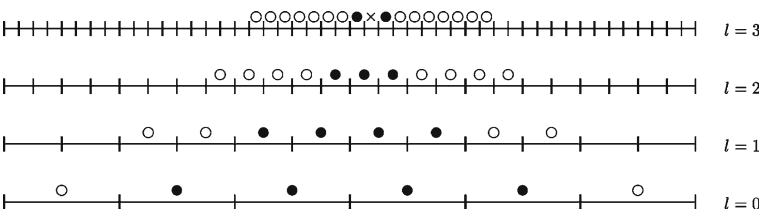

Fig. 6. Influence of different prediction strategies.

sketch the influence of the modified prediction strategy in comparison to the old one after having performed the grading step. Here we consider one significant detail denoted by $\times$ on level $l=3$. Then we mark by $\bullet$ the cells corresponding to the old strategy. The additional cells $\circ$ characterize the inflation due to the modification. We note that the graded tree is much more inflated, in particular, on the higher levels. Obviously, this strategy ensures that all effects are properly resolved on the new time level after having performed the macro-time step. However, the efficiency degrades significantly.

To optimize the efficiency we suggest an alternative approach. The idea is to perform additional grid adaptation steps according to Sec. 2.2 after each synchronization. In particular, after having propagated the data on levels $l=l_i,\dots,L$, which correspond to 2^l time steps with τ_l, we apply the grid adaptation on this part of the grid only instead of the whole adaptive grid. By this procedure it is possible to track, for instance, the shock position on the intermediate time levels instead of a *priori* refining the whole range of influence. In other words, we make sure that a significant information on level l can only propagate by at most *one* cell on *this* level when performing the evolution step with τ_l. To realize this we only have to perform minor changes in the grid adaptation algorithms where we replace the coarsest level 0 by the current synchronization level l_i, i.e., level l_i is considered to be the coarsest level. The data and the cells on the coarser levels $0,\dots,l_i-1$ are not affected. Moreover, we replace the index sets $\mathcal{G}_{L,\epsilon}$ and $\mathcal{D}_{L,\epsilon}$ by $\mathcal{G}_{l_i,L,\epsilon}$ and $\mathcal{D}_{l_i,L,\epsilon}$, respectively, to indicate that these sets only correspond to the levels $l_i,\dots,L$. Note that in practice we do not need additional data structures but can work on the same data structures as before. The modified grid adaptation strategy is summarized in the following algorithm.

Algorithm 3 (Partial grid adaptation on levels $l_i, \ldots, L$ at time $t_{n+i\,2^{-L}}$)

(1) Local multiscale transformation performing the loop over l from L down to $l_i + 1$;
(2) Thresholding performing the loop over l from l_i to $L - 1$;
(3) Prediction of significant details performing the loop over all $(l, k) \in \mathcal{D}^{n+i\,2^{-L}}_{l_i, L, \epsilon}$;
(4) Grading performing the loop over l from L down to $l_i + 1$;
(5) Grid adaptation performing the loop over l from l_i to $L - 1$ where the grid $\mathcal{G}^{n+(i+1)\,2^{-L}}_{l_i, L, \epsilon}$ for the new time step is initialized by the cells of the old grid $\mathcal{G}^{n+i\,2^{-L}}_{L, \epsilon}$ corresponding to the coarser levels $0, \ldots, l_i - 1$;
(6) Local inverse multiscale transformation performing the loop over l from l_i to $L - 1$.

When performing this partial grid adaptation on the levels $l_i, \ldots, L$, we have to make sure that a cell on level l_i that is adjacent to a cell of level $l_i - 1$ is not refined. Otherwise there will be a level-2-transition in the adaptive grid that does not fit the assumptions of the algorithms realizing the multiscale transformation (see Sec. 2.1). This can be avoided by increasing the grading parameter by 1 (see Sec. 2.2).

3.1.5. Algorithm

Finally we have to combine the time evolution at the intermediate time steps $t_{n+i\,2^{-L}}$ and the partial grid adaptation at the synchronization levels $l = l_i, \ldots, L$. For this purpose, we have to combine appropriately the Algorithms 2 and 3. The complete macro-time step is summarized in the following algorithm:

Algorithm 4 (Explicit local time stepping)

(1) Initialization:
    (a) Grid adaptation:
    Perform grid adaptation according to Algorithm 3 on the levels $0, \ldots, L$ providing the grid $\tilde{\mathcal{G}}^{n+2^{-L}}_{L, \epsilon}$ and the corresponding cell averages $v^n_{l,k}$ at time t_n;
    (b) Index sets:
    Determine the index sets $\mathcal{C}_l, \overline{\mathcal{C}}_l, l = 0, \ldots, L$ and the index set of the numerical fluxes $\mathcal{F}$ according to Algorithm 1.

For each intermediate time step $i = 1, \dots, 2^L$ we then perform the following steps:

(2) Synchronization:
The time evolution at time $t_{n+i\,2^{-L}}$ is performed according to Algorithm 2;

(3) Partial grid adaptation:
After having synchronized the data on levels $l = l_i, \dots, L$, we adapt the grid on these levels to the new data according to Algorithm 3;
due to the grid adaptation the redistribution of cells makes it necessary to reinitialize the sets $\mathcal{C}_l, \overline{\mathcal{C}}_l, l = l_i - 1, \dots, L$ and the index set $\mathcal{F}$ on the levels $l = l_i, \dots, L$ according to Algorithm 1.

3.1.6. Remarks on the strategy of Osher and Sanders

Assume that the locally adapted grid $\mathcal{G}_{L,\epsilon}^{n+1}$ is static for all intermediate time steps of the time evolution described in Algorithm 2. Then we may rewrite the algorithm using the predictor–corrector representation by Osher and Sanders [39]. For this purpose, we introduce the notation $M := 2^L, \sigma_j := 2^{-L}, j = 1, \dots, M-1, n_j := \sum_{i=1}^{j} \sigma_i = j\,2^{-L}$, and $\tilde{\lambda}_l := \tau_0/h_l$. In addition, we introduce the set

$$\mathcal{C}^i := \{(l,k) \in \mathcal{G}_{L,\epsilon}^{n+1} \backslash \mathcal{C}_{l_i-1} \,;\, 0 \leqslant l \leqslant l_i - 1\}, \quad i = 1, \dots, M.$$

Then for each $i = 1, \dots, M-1$ the *predictor* is defined by

$$v_{l,k}^{n+n_i} = \begin{cases} v_{l,k}^{n+n_{i-1}}, & (l,k) \in \mathcal{C}^i, \\ v_{l,k}^{n+n_{i-1}} - \tilde{\lambda}_l \sum_{j=0}^{i-1} \sigma_{j+1} \left(F_{l,k+1}^{n+n_{j-1}} - F_{l,k}^{n+n_{j-1}} \right), & (l,k) \notin \mathcal{C}^i \end{cases}$$

and the *corrector* is

$$v_{l,k}^{n+1} = v_{l,k}^{n} - \tilde{\lambda}_l \sum_{j=0}^{M-1} \sigma_{j+1} \left(F_{l,k+1}^{n+n_{j-1}} - F_{l,k}^{n+n_{j-1}} \right) \equiv v_{l,k}^{n} - \frac{\tau_l}{h_l} \left(\overline{F}_{l,k+1}^{n} - \overline{F}_{l,k}^{n} \right).$$

For an example, see the two-level scheme in Sec. 3.1.2.

3.1.7. Remarks on AMR

In [11], the classical AMR strategy has been originally introduced. In this context, local time stepping and flux synchronization has been investigated by Berger [7, 8]. In the following, we would like to point out the main differences of the concept proposed in this work.

In the AMR setting the propagation of the cells is performed levelwise where first the cells on the coarse level are evolved and then the one on the finer levels. This results in two numerical fluxes at the interface (see Fig. 2), namely $\overline{F}^n_{l,k}$ for the cell $V_{l,k-1}$ and $F^n_{l+1,2k}$, $F^{n+1/2}_{l+1,2k}$ for the cell $V_{l+1,k}$, respectively. To ensure the conservation property of the scheme, a so-called synchronization step is necessary to compensate for the difference

$$\overline{F}^n_{l,k} - \frac{1}{2}\left(F^n_{l+1,2k} + F^{n+1/2}_{l+1,2k}\right)$$

in the coarse cell $V_{l,k-1}$. In the present setting, this synchronization step is superfluous because we proceed levelwise from fine to coarse and compute $\overline{F}^n_{l,k}$ by the already computed information on the higher level (see (34)).

Furthermore, we perform a *half step* to determine a prediction in the coarse cell $V_{l,k-1}$ for the computation of the flux $F^{n+1/2}_{l+1,2k}$ (see Fig. 6). This is different in the AMR setting. There a prediction value $v^{n+1/2}_{l,k-1}$ is computed by means of some interpolation between the data $v^n_{l,k-1}$ and $v^{n+1}_{l,k-1}$.

In order to reduce the number of interfaces where a flux synchronization is needed the AMR setting typically works on grid patches composed of cells corresponding to one level. Hence, the fluxes have to be synchronized only at the boundaries of the patches. This also simplifies the implementation. However, the size of the patches has to be chosen such that, for instance, a shock is not leaving this patch when performing the macro-time step. Otherwise the shock could be underresolved on a coarser grid patch. To realize this either the patches have to be large or the macro-time step has to be sufficiently small. Therefore, the number of refinement levels is typically moderate in practice. Since, we perform grid adaptation as well on the intermediate time steps we are able to track the shock successively without *a priorly* refining the whole domain of influence of the shock. Therefore, we also can handle a large number of refinement levels that corresponds to a large macro-time step without inflating the adaptive grid too much.

Finally, we note that the grid patches on which AMR is typically working need not to be nested but can have a different orientation than the coarse grid. This allows for a local alignment of the grid with anisotropic effects such as shocks. In contrast to this, the multiscale setting is based on a *nested* grid hierarchy.

3.2. Implicit Local Time Stepping

So far we considered only explicit local time stepping schemes. Note that for implicit schemes time restrictions also occur due to nonlinear

stability (TVD properties), convergence of the Newton scheme (initial guess) and relaxation schemes for solving linear problems, relaxation processes due to nonequilibrium effects, anisotropies in the grid, etc. Therefore, a local time stepping may also be helpful for implicit schemes in case of instationary problems. In analogy to the explicit case, we first confine ourselves to two refinement levels, i.e., a fine grid (level $l+1$) and a coarse grid (level l). Let us assume that we know the data on level $l+1$ at time step t_n. We now perform two time steps where we apply our implicit reference scheme on level $l+1$ with time step size $\tau \equiv \tau_{l+1} = 2^{-1}\,\tau_l$ according to (23), i.e.,

$$v_{l+1,r}^{n+1/2} + \theta\,\overline{\lambda}_{l+1}\,B_{l+1,r}^{n+1/2} = v_{l+1,r}^{n} - (1-\theta)\,\overline{\lambda}_{l+1}\,B_{l+1,r}^{n}, \tag{42}$$

$$v_{l+1,r}^{n+1} + \theta\,\overline{\lambda}_{l+1}\,B_{l+1,r}^{n+1} = v_{l+1,r}^{n+1/2} - (1-\theta)\,\overline{\lambda}_{l+1}\,B_{l+1,r}^{n+1/2}, \tag{43}$$

where $\overline{\lambda}_{l+1}$ is defined by (30).

Here the flux balances are computed according to (26) by means of the data $v_{l+1,r}^{n}$, $v_{l+1,r}^{n+1/2}$ and $v_{l+1,r}^{n+1}$ corresponding to the times t_n, $t_{n+1/2} = t_n + \tau_{l+1}$ and $t_{n+1} = t_n + 2\,\tau_{l+1}$, respectively. We now replace $v_{l+1,r}^{n+1/2}$ on the right-hand side of (43) by (42). Then we obtain

$$v_{l+1,r}^{n+1} + \theta\,\overline{\lambda}_{l+1}\left(B_{l+1,r}^{n+1} + B_{l+1,r}^{n+1/2}\right) = v_{l+1,r}^{n} - (1-\theta)\,\overline{\lambda}_{l+1}\left(B_{l+1,r}^{n+1/2} + B_{l+1,r}^{n}\right). \tag{44}$$

Note that (44) is only a nonlinear problem for the data $v_{l+1,r}^{n+1}$ provided that the data on the intermediate time level $t_{n+1/2}$ are known. Applying the multiscale transformation (13)–(44) again yields the discrete evolution equations for cell averages on the coarser level

$$v_{l,k}^{n+1} + \theta\,\overline{\lambda}_l\,\overline{B}_{l,k}^{n+1} = v_{l,k}^{n} - (1-\theta)\,\overline{\lambda}_l\,\overline{B}_{l,k}^{n}, \quad \overline{\lambda}_l := \frac{\tau_l}{h_l} = \overline{\lambda}_{l+1}. \tag{45}$$

In analogy to (33) the local flux balances and numerical fluxes are recursively defined by

$$\overline{B}_{l,k}^{n+i} := \sum_{r \in \mathcal{M}_{l,k}^{0}} m_{r,k}^{l,0}\left(B_{l+1,r}^{n+(i+1)/2} + B_{l+1,r}^{n+i/2}\right) = \overline{F}_{l,k+1}^{n+i} - \overline{F}_{l,k}^{n+i}, \tag{46}$$

$$\overline{F}_{l,k}^{n+i} := \frac{1}{2}\left(F_{l+1,2k}^{n+(i+1)/2} + F_{l+1,2k}^{n+i/2}\right) \tag{47}$$

with $i = 0, 1$. Again the multiscale scheme determines the computation of the flux balances and the numerical fluxes on coarser levels proceeding from *fine* to *coarse*. In particular, we note that the computation of $\overline{F}_{l,k}^{n+i}$ is

in agreement with (34). As for the explicit case (see Sec. 3.1.1), we have to provide some *prediction value* $\tilde{v}_{l,k}^{n+1/2}$ to compute the numerical flux $F_{l+1,2k}^{n+i/2}$ at interface points in (47).

3.2.1. Prediction Value

For the implicit case we have to reconsider the strategies to determine appropriate prediction values for computing the numerical flux $F_{l+1,2k}^{n+1/2}$ near interface points. According to (42), a nonlinear problem has to be solved to determine $v_{l+1,r}^{n+1/2}$. Proceeding this way we cannot gain in computational efficiency using locally varying time stepping. To overcome this bottleneck we use some heuristics. In principle, we proceed as in the explicit case (see Sec. 3.1.2). However, the *full steps* in (35) and (36), respectively, are replaced by solving a nonlinear problem for each level separately. For the prediction value we then use the information of the previous time step instead of performing a *half step*. For the situation sketched in Fig. 2, we hence proceed as follows: in the first step, we evolve all cells on level $l+1$ by a full step with τ_{l+1}, i.e., we solve the nonlinear problem

$$v_{l',k'}^{n+1/2} + \theta\,\overline{\lambda}_{l+1}\,B_{l',k'}^{n+1/2} = v_{l',k'}^{n} - (1-\theta)\,\overline{\lambda}_{l+1}\,B_{l',k'}^{n},\quad (l',k') \in \tilde{I}_{l+1},$$

where at interface points we use the prediction

$$v_{l',k'}^{n+1/2} = v_{l',k'}^{n},\quad (l',k') \in \overline{C}_l.$$

In a second step, we determine the data on the new time level t_{n+1}. For this purpose, we solve two nonlinear problems for the cells on level l and $l+1$, separately for each level, i.e.,

$$\begin{aligned} v_{l',k'}^{n+1} + \theta\,\overline{\lambda}_{l+1}\,B_{l',k'}^{n+1} &= v_{l',k'}^{n+1/2} - (1-\theta)\,\overline{\lambda}_{l+1}\,B_{l',k'}^{n+1/2},\quad (l',k') \in \tilde{I}_{l+1}, \\ v_{l',k'}^{n+1} + \theta\,\overline{\lambda}_{l}\,B_{l',k'}^{n+1} &= v_{l',k'}^{n} - (1-\theta)\,\overline{\lambda}_{l}\,B_{l',k'}^{n},\quad (l',k') \in \tilde{I}_{l}. \end{aligned} \tag{48}$$

Again the solution process is from *fine* to *coarse*. In our computations the nonlinear problems are solved approximately by applying the Newton scheme. As initial guess, we use the data of the previous (intermediate) time step. Note that we never solve a nonlinear problem for both levels at a time. A coupling of the systems in (48) only exists via the flux computation at interface points according to (47) and (25). Furthermore, if we choose $\theta = 0$ then the resulting explicit scheme does not coincide with the explicit local time stepping scheme in Sec. 3.1.2. In this case, the prediction corresponds to the naive strategy discussed above.

Finally, we like to remark that the above strategy is some heuristic approach to compute a prediction value at grid interface points. There might be other strategies that work as well, for instance, one might think of solving the coupled system (48) for both levels l and $l+1$. However, fully decoupling the nonlinear systems will result in smaller problems to be solved iteratively. This will be preferable in case of multidimensional systems where the nonlinear problems might become huge. There we can save both in computational time and memory.

3.2.2. Synchronization of Time Evolution

The synchronization of the time evolution in the multilevel case is similar to the explicit case as described in Sec. 3.1.3. In principle, we may apply the Algorithms 1 and 2 where we remove any operation on the sets $\overline{\mathcal{C}}_l$ and perform all operations on $\overline{\mathcal{C}}_l$ for the sets $\tilde{\mathcal{I}}_l$. First of all, we have to initialize the index sets $\tilde{\mathcal{I}}_l, l=0,\dots,L$ and the index set of the numerical fluxes $\mathcal{F}$ according to Algorithm 1. Then we may perform the time evolution for each of the intermediate time steps $i=1,\dots,2^L$.

Algorithm 5 (Synchronized time evolution for time step $t_{n+i\,2^{-L}}$)
For each intermediate time step we have to perform the following steps for the levels $l=l_i,\dots,L$ proceeding from *fine* to *coarse*:

(1) Flux computation:

(a) For the levels $l_i \leqslant l \leqslant L$ we determine the numerical fluxes with respect to the data of the previous intermediate time step corresponding to $t_{n+(i-1)\,2^{-L}}$, i.e.,

(i) no interface point $((l+1,2k)\notin\mathcal{F})$

$$F_{l,k}^{n+(i-1)\,2^{-L}} = F\left(v_{L,2^{L-l}k-p}^{n+(i-1)\,2^{-L}},\dots,v_{L,2^{L-l}k+p-1}^{n+(i-1)\,2^{-L}}\right),$$

(ii) interface point $((l+1,2k)\in\mathcal{F})$

$$F_{l,k}^{n+(i-1)\,2^{-L}} = F_{l+1,2k}^{n+(i-1)\,2^{-L}};$$

(b) For the levels $0 \leqslant l < l_i$ the numerical fluxes are unchanged, since the cell averages on these levels have not changed in the previous intermediate time step, i.e.,

$$F_{l,k}^{n+(i-1)\,2^{-L}} = F_{l,k}^{n+(i-2)\,2^{-L}};$$

(2) Time Evolution:

(a) For the levels $l_i \leqslant l \leqslant L$ we perform a *full* time step with τ_l for the cells $(l,k) \in \tilde{I}_l$ solving the nonlinear problem

$$v_{l,k}^{n+i\,2^{-L}} + \theta\,\bar{\lambda}_l\, B_{l,k}^{n+i\,2^{-L}} = v_{l,k}^{n+(i-1)\,2^{-L}} - (1-\theta)\bar{\lambda}_l\, B_{l,k}^{n+(i-1)2^{-L}};$$

(b) For the levels $0 \leqslant l < l_i$ the cell averages are unchanged, i.e.,

$$v_{l,k}^{n+i\,2^{-L}} = v_{l,k}^{n+(i-1)\,2^{-L}}.$$

3.2.3. Algorithm

As in the explicit case, we combine the time evolution at the intermediate time steps $t_{n+i\,2^{-L}}$ and the partial grid adaptation at the synchronization levels $l = l_i, \ldots, L$. For the prediction of the significant details, we use the same strategy as for the explicit case (see Sec. 3.1.4). Therefore, the algorithm for one macro-time step using implicit local time stepping is similar to Algorithm 4. Only minor changes are necessary, namely, (i) in steps, 1b and 3, we initialize the sets $\tilde{\mathcal{I}}_l$ instead of the sets $\mathcal{C}_l$ and (ii) in step 2, we replace Algorithm 2 for the explicit time stepping algorithm by Algorithm 5.

4. NUMERICAL RESULTS

The benefits of the proposed concept are verified by several numerical investigations. For this purpose, we perform several parameter studies for the inviscid Burger equation in 1D where we investigate the efficiency in terms of CPU time and the accuracy by comparisons with the exact solution. Here, we consider an instationary problem and a quasi-steady state problem, respectively, applying an explicit and implicit time discretization as well. Note that all computations have been performed on a PC with an Intel Pentium IV processor and 2.8 GHz.

4.1. Reference Scheme

To verify the efficiency and the accuracy of the proposed local time stepping strategy we perform several numerical simulations for the inviscid Burger equation, i.e., $f(u) = 0.5\,u^2$ in (1). For the reference scheme (3) we consider a first and second-order finite volume scheme, respectively. The numerical flux is chosen to be the Engquist–Osher (EO) flux (see [26]),

$$F^{\mathrm{EO}}(v_L, v_R) = \frac{1}{2}\left(f(v_L) + f(v_R) - \int_{v_L}^{v_R} |f'(u)|\, du \right). \tag{49}$$

For the first-order scheme the states v_L and v_R are determined by the cell data at the interface. To improve spatial and temporal accuracy we employ a piecewise linear ENO reconstruction according to [30]. For a non-equidistant grid this reads

$$v_L = \begin{cases} w_{-1}, & \text{First-order,} \\ w_{-1} + \overline{m}\left(\Delta w_0, \Delta w_{-1}\right)\left(h_{-1}/2 - \tau\, f'(w_{-1})\right), & \text{Second-order,} \end{cases}$$

$$v_R = \begin{cases} w_0, & \text{First-order,} \\ w_0 - \overline{m}\left(\Delta w_1, \Delta w_0\right)\left(h_0/2 + \tau\, f'(w_0)\right), & \text{Second-order} \end{cases} \tag{50}$$

with the divided differences Δw_i and the function $\overline{m}$ defined by

$$\Delta w_i := \frac{w_i - w_{i-1}}{h_i + h_{i-1}}, \qquad \overline{m}(a, b) := \begin{cases} a\,, & |a| \leqslant |b|, \\ b\,, & |a| > |b|. \end{cases}$$

Here the stencil is determined by the values w_i and the corresponding discretization lengths h_i, $i = -2, \ldots, 1$. Note that the term corresponding to the time discretization τ guarantees second-order in time for the explicit scheme. For an implicit scheme the time derivative is discretized applying the second-order Crank–Nicholson scheme. Therefore, this term is suppressed. For the computation of the local numerical fluxes we employ an unstructured flux computation, i.e., we do not access to the data of the finest level but to the p next neighbors to the left and the right of a cell $V_{l,k}$ corresponding to the adaptive grid. In practice, this in general does not affect the accuracy but preserves the computational complexity (see [21]).

For the implicit time discretization scheme we choose $\theta = 0.5$ in (3), i.e., second-order time discretization. In each time step, the nonlinear system is solved iteratively by the Newton scheme which is initialized by the data of the previous (intermediate) time step. In each Newton step, we solve a linear problem. The matrix is determined by computing the Jacobian of the EO flux (49). Note that in the case of the second-order scheme we do not take into account the derivative of the ENO reconstruction. Therefore each row of the Jacobian has at most three nonvanishing entries. Enumerating the unknowns from left to right results in a tridiagonal matrix. This system can be efficiently solved by an exact solver. For multidimensional problems this is no longer feasible. In this case, we have to choose some iterative solver and an appropriate preconditioner to avoid time restrictions due to the linear problem. The Newton iteration terminates when the error has dropped below a certain tolerance value. For the

local time stepping strategy, we choose a level-dependent tolerance determined by the threshold value, i.e., $\text{tol} = \epsilon_l$, to solve the nonlinear problem on level l. Otherwise we put $\text{tol} = \epsilon_L$ because the nonlinear problem is to be solved for all refinement levels simultaneously. The number of Newton steps is limited by 15 that is never reached in our computations.

To investigate the performance of the proposed concept we consider an unsteady problem and a quasi-steady state problem, respectively.

4.2. Wave Interaction

The first test configuration is determined by the piecewise constant initial data

$$u(x,0) = \begin{cases} 3, & x < 0.1, \\ -2, & 0.1 \leqslant x < 0.5, \\ 5, & 0.5 \leqslant x < 0.9, \\ -5, & x \geqslant 0.9. \end{cases} \tag{51}$$

The exact solution to this problem is composed of a right-running shock wave (left) and a stationary shock wave (right) separated by a rarefaction wave. These waves start to interact. In the end there is only one moving shock.

We apply the explicit (implicit) adaptive multiscale scheme with and without locally varying time stepping to this problem. The computations are distinguished by EXP (IMP) and EXP-LTS (IMP-LTS). The computational domain is $\Omega = [0, 1]$ and the integration time is $T = 0.5$ s. The coarsest discretization is $h_0 = 0.05$. The time discretization is fixed determined by the CFL number. For EXP we choose $\text{CFL}_L = 0.5 = 2^{-L}\,\text{CFL}_0$ on the finest discretization level and $\text{CFL}_0 = 0.5 = 2^{-L}\,\text{CFL}_L$ for EXP-LTS on the coarsest level. This implies that for EXP we have to perform 2^L time steps with $\tau = \tau_L$. This corresponds to one macro-time step with EXP-LTS using $\tau = \tau_0 = 2^L \tau_L$. For the grid adaptation we choose wavelets with three vanishing moments. The threshold value is fixed by $\epsilon = 0.001$.

In Fig. 7, the solution is shown exemplarily for some characteristic times recorded in Table II. The right figures show the exact solution and an approximate solution with EXP-LTS using $L = 10$ refinement levels. The corresponding adaptive grids are presented in the figures on the left. Here we plot the cell center with respect to the refinement level l

In Table III, we summarize the CPU times for different computations with first and second-order EXP and EXP-LTS where we vary the number of refinement levels L. We note that EXP-LTS becomes much faster with increasing number of refinement levels in comparison to EXP. Speed-up rates up to a factor of about 8 have been obtained. Similar results

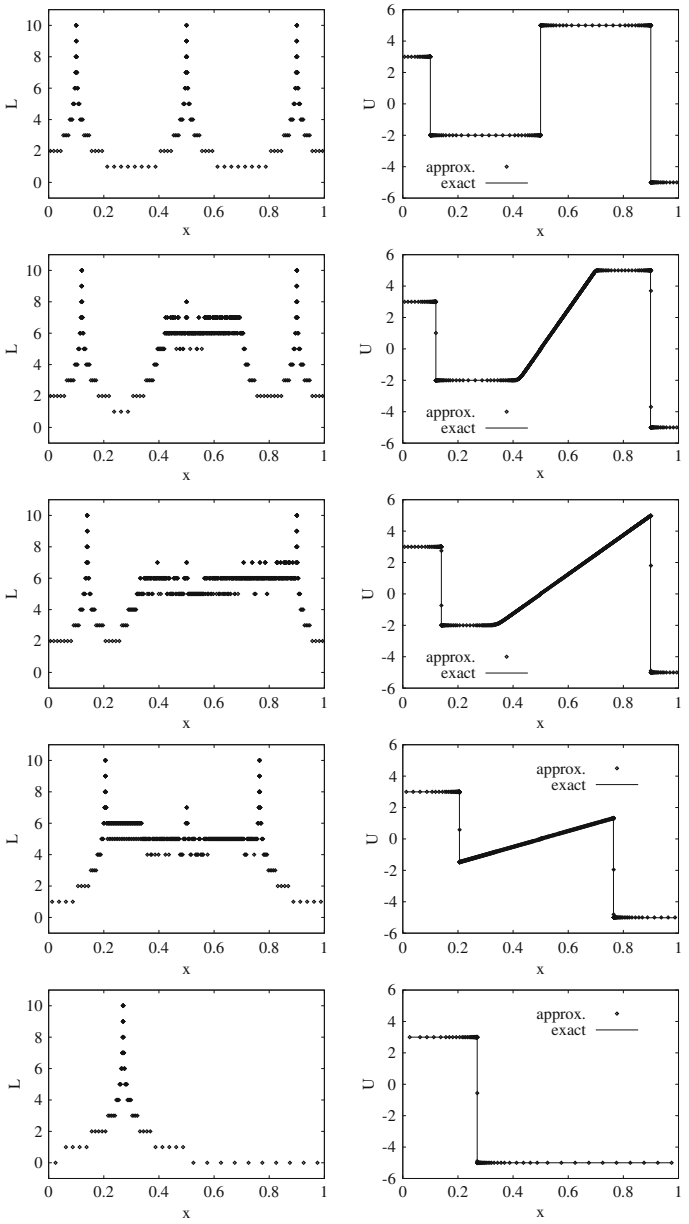

Fig. 7. Test configuration 1: Adaptive grid (left), exact and approximate solution (right) at time $T = 0.00,\ 0.04,\ 0.08,\ 0.20,$ and 0.48.

Table II. Characteristic times (s) for test configuration 1

$T_0 = 0.00$	Initial data
$T_1 = 0.04$	Before interaction
$T_2 = 0.08$	After interaction of rarefaction wave with right shock wave
$T_3 = 0.20$	After interaction of rarefaction wave with left shock wave
$T_4 = 0.48$	Single shock

Table III. CPU time (s) for explicit discretization

	First-order scheme		Second-order scheme	
L	EXP	EXP-LTS	EXP	EXP-LTS
5	3.58	0.94	4.01	1.14
6	8.47	1.87	9.28	2.21
7	20.24	3.61	22.00	4.40
8	47.09	7.67	51.87	8.78
9	111.35	14.95	119.67	17.68
10	251.49	31.50	272.78	36.11

are obtained for the implicit discretization (see Table IV). Because of the instationary behavior of the solution the CFL number is bounded by at most 1 for an explicit as well as an implicit time discretization. Therefore, all computations with an explicit scheme are faster than using the implicit analog. However, there are applications in engineering such as the shock buffet case where the flow field is instationary because the shock is oscillating at moderate speed due to fluid–structure interactions. Moreover, if there is some "stiffness" encountered in the problem, for instance, some relaxation process due to chemical reactions, then the time restriction comes from the relaxation time. In this case, an implicit discretization will relax this time restriction significantly.

Furthermore, we note that the speed-up rates crucially depend on the distribution of the cells over the refinement levels. If the bulk of cells is sitting on the higher levels the number of flux computations is not significantly reduced and the gain in CPU time is moderate. In general, the number of flux computations for global time stepping F_{NLTS} and local time stepping F_{LTS} are approximately related by

Table IV. CPU time (s) for implicit discretization

	First-order scheme		Second-order scheme	
L	IMP	IMP-LTS	IMP	IMP-LTS
5	25.94	4.63	27.53	5.76
6	67.83	8.98	73.20	10.12
7	107.09	17.23	107.25	19.44
8	258.19	35.81	279.77	40.45
9	756.11	87.76	753.91	89.68
10	1853.15	202.65	2160.50	193.97

$$F_{\mathrm{LTS}} \approx F_{\mathrm{NLTS}} \sum_{l=0}^{L} 2^{l-L}\, \alpha_l,$$

where we neglect the fluxes at the boundary. Here $\alpha_l = \tilde{N}_l/\tilde{N}$ is the ratio of cells on level l and the number of all cells in the adaptive grid, i.e., $\tilde{N} = \sum_{l=0}^{L} \tilde{N}_l$. In particular, for Cartesian grid hierarchies this relation is independent of the spatial dimension. Obviously, the approximate speed-up factor $\left(\sum_{l=0}^{L} 2^{l-L}\, \alpha_l\right)^{-1}$ approaches 1 if the bulk of cells is sitting on the higher scales whereas it becomes large (at most: 2^L) if the cells on lower scales are dominating. This is in agreement with Jameson's observation that adaptive mesh refinement methods can be effective provided that "roughly no more than one-third of the domain should be at the finest grid spacing" (see [32]).

Besides the efficiency we are interested in the accuracy of the local time stepping scheme. Therefore, we investigate the error between the exact solution, $\hat{u}$, and the numerical approximations, v. Since the ultimate goal is to maintain the accuracy of the reference scheme we compute the error on the uniform finest refinement level. For this purpose, we map the numerical solution corresponding to the adaptive grid to the reference grid where we apply the full inverse multiscale transformation and use a value of zero for all nonsignificant details. The error is measured in the weighted l_1-norm, i.e.,

$$\|\boldsymbol{v}^n - \hat{\boldsymbol{u}}^n\| := \sum_{k \in I_L} h_L\, |v^n_{L,k} - \hat{u}^n_{L,k}|.$$

In Tables V (explicit) and VI (implicit) the error is exemplarily recorded for the computation with $L = 10$ refinement levels. The results correspond to the four times T_1, T_2, T_3, and T_4 given in Table II. We note that the

Table V. Error in weighted l_1-norm for explicit discretization, $L = 10$

	First-order scheme		Second-order scheme	
Time	EXP	EXP-LTS	EXP	EXP-LTS
T_1	1.30×10^{-2}	4.70×10^{-3}	1.25×10^{-2}	3.00×10^{-3}
T_2	2.42×10^{-2}	6.43×10^{-3}	2.32×10^{-2}	4.07×10^{-3}
T_3	2.85×10^{-2}	6.64×10^{-3}	2.71×10^{-2}	5.38×10^{-3}
T_4	2.30×10^{-5}	1.9×10^{-5}	3.30×10^{-5}	2.30×10^{-5}

Table VI. Error in weighted l_1-norm for implicit discretization, $L = 10$

	First-order scheme		Second-order scheme	
Time	IMP	IMP-LTS	IMP	IMP-LTS
T_1	1.39×10^{-2}	1.01×10^{-2}	1.27×10^{-2}	8.64×10^{-3}
T_2	2.80×10^{-2}	1.62×10^{-2}	2.22×10^{-2}	1.49×10^{-2}
T_3	3.14×10^{-2}	1.83×10^{-2}	2.53×10^{-2}	1.79×10^{-2}
T_4	4.00×10^{-5}	1.22×10^{-2}	3.90×10^{-5}	1.19×10^{-2}

error of the computations with local time stepping are in general smaller, except for IMP-LTS at time T_4 where the solution exhibits only two constant states separated by a shock. Since we perform less time steps for coarser cells we locally reduce the number of grid adaptation steps. In particular, the number of threshold steps is reduced in these cells. Therefore, the accumulative threshold error over all time steps is smaller. Moreover, the schemes with locally varying time stepping exhibit also less numerical diffusion.

Altogether we conclude for the local time stepping scheme that we can significantly improve the efficiency of the adaptive multiscale scheme and we even gain in accuracy.

4.3. Slowly Moving Shock Wave

A second test configuration is determined by the initial data

$$u(x, 0) = \begin{cases} 1, & x < 0.5, \\ -0.99, & x \geqslant 0.5. \end{cases} \tag{52}$$

The exact solution is a slowly mowing shock wave propagating at shock speed $s = (u_l + u_r)/2 = 0.005$. This configuration is well suited to underline the benefits of an implicit discretization.

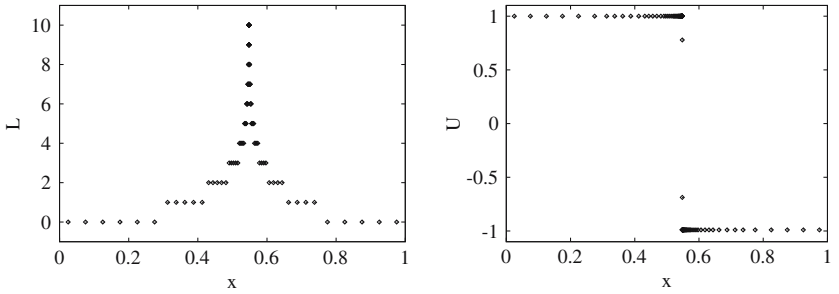

Fig. 8. Test configuration 2: Adaptive grid (left), approximate solution (right) at time $T = 10$.

The computational domain is $\Omega = [0, 1]$ and the integration time is $T = 10$ s. The coarsest discretization is $h_0 = 0.05$. For the explicit scheme the time discretization is determined by the CFL number. For EXP we choose $\mathrm{CFL}_L = 0.64$ on the finest discretization level and $\mathrm{CFL}_0 = 0.64$ for EXP-LTS on the coarsest level. Since the maximum characteristic speed $\max_{x \in \Omega}\{|f'(u_0(x))|\} = 1$ is much larger than the shock speed we may use a higher CFL number for the implicit discretization. Here we use $\mathrm{CFL}_L = 4$ (IMP) and $\mathrm{CFL}_0 = 4$ (IMP-LTS), respectively. Note that the shock wave will *not* move by more than *one* cell although the CFL number is larger than 1. Therefore, the prediction strategy in Sec. 2.2 has *not* to be adjusted to the higher CFL number. For the grid adaptation we again choose wavelets with three vanishing moments. The threshold value is fixed by $\epsilon = 0.001$.

In Fig. 8, the solution is shown at time $T = 10$. The right figure shows the approximate solution with the second-order IMP-LTS using $L = 10$ refinement levels. The figures on the left show the corresponding adaptive grid. Here we again plot the cell center with respect to the refinement level l.

From Tables VII and VIII, we conclude for the explicit and implicit scheme a similar performance of the speed-up rates as for test case 1 in Sec. 4.2. However, the implicit computation is faster due to the higher CFL number. This indicates that the implicit strategy might be useful for steady state problems arising, for instance, in fluid flow around airplane wings. Here the question remains open how to control automatically the CFL number.

5. CONCLUSION AND OUTLOOK

We have developed a general concept to incorporate local time stepping into fully adaptive multiscale finite volume schemes. In particular, we

Table VII. CPU time (s) for explicit scheme

	First-order scheme		Second-order scheme	
L	EXP	EXP-LTS	EXP	EXP-LTS
5	6.75	1.29	9.68	1.63
6	15.43	2.61	22.19	2.68
7	35.75	5.41	51.34	6.55
8	79.90	10.82	113.34	13.15
9	175.93	21.60	253.44	25.28
10	388.62	43.48	467.75	52.60

Table VIII. CPU time (s) for implicit scheme

	First-order scheme		Second-order scheme	
L	IMP	IMP-LTS	IMP	IMP-LTS
5	3.01	0.84	3.60	0.89
6	7.32	1.53	8.51	1.69
7	16.57	3.20	19.85	3.44
8	40.40	6.38	46.27	6.78
9	92.72	12.74	105.92	13.60
10	211.01	26.18	239.85	26.30

are able to track the position of discontinuities on the intermediate time levels where we apply the grid adaptation strategy on a subrange of all available refinement levels. Numerical investigations for one-dimensional scalar conservation laws verify the efficiency and the accuracy of the proposed concept. First 2D Euler computations using an explicit time discretization show the benefits of the concept also for multidimensional systems (see [38]). More details on the extension of the concept to higher dimensional problems and their efficient implementation can be found in [35].

The analytical justification is still open. In analogy to previous work in the context of AMR [7, 8, 11] and the predictor–corrector approach [25, 39, 44] stability and consistency have to be investigated. In addition, we have to verify the reliability of the strategy for predicting significant details. This issue is important to bound uniformly the perturbation error introduced by the thresholding. For the global time stepping scheme this was done in [21, 37]. In particular, we have to analyze the connection between the details and the time derivatives of the solution. This might also be helpful in the design of an automatic time step control for steady state problems.

In the future, we will incorporate the presented local time stepping strategy into the flow solver QUADFLOW [16] that is being developed for large scale computations of compressible fluid flow and fluid–structure interaction. In particular, this will enable us to investigate the above strategy for multidimensional systems using an implicit time discretization. First results on moving grids employing multiscale space and time adaptivity can be found in [34].

ACKNOWLEDGMENTS

This work has been performed with funding by the Deutsche Forschungsgemeinschaft in the Collaborative Research Centre SFB 401 *Flow Modulation and Fluid-Structure Interaction at Airplane Wings* of the RWTH Aachen, University of Technology, Aachen, Germany, and the "Ramón y Cajal" program of the Ministerio de Educacion y Ciencia, Spain.

REFERENCES

1. Abgrall, R. (1997). Multiresolution analysis on unstructured meshes: applications to CFD. In Chetverushkin, B. et al. (eds.), *Experimentation, Modelling and Computation in Flow, Turbulence and Combustion*, Wiley, New York.
2. Andreae, S., Ballmann, J., and Müller, S. (2005). Wave processes at interfaces. In Warnecke, G. (ed.), *Analysis and Numerics for Conservation Laws*, Springer, Berlin, pp. 1–26.
3. Andreae, S., Ballmann, J., Müller, S., and Voß, A. (2003). Dynamics of collapsing bubbles near walls. In Hou, T., and Tadmor E. (eds.), *Hyperbolic Problems: Theory, Numerics, Applications*, Springer Verlag, Berlin, pp. 265–272.
4. Arandiga, F., Donat, R., and Harten, A. (1998). Multiresolution based on weighted averages of the hat function I: linear reconstruction techniques. *SIAM J. Numer. Anal.* **36**(1), 160–203.
5. Arandiga, F., Donat, R., and Harten, A. (1999). Multiresolution based on weighted averages of the hat function II: non-linear reconstruction techniques. *SIAM J. Sci. Comput.* **20**(3), 1053–1093.
6. Bell, J., Berger, M., Saltzman, J., and Welcome, M. (1994). Three-dimensional adaptive mesh refinement for hyperbolic conservation laws. *SIAM J. Sci. Comput.* **15**(1), 127–138.
7. Berger, M. (1985). Stability of interfaces with mesh refinement. *Math. Comp.* **45**, 301–318.
8. Berger, M. (1987). On conservation at grid interfaces. *SIAM J. Numer. Anal.* **24**, 967–984.
9. Berger, M., and Colella, P. (1989). Local adaptive mesh refinement for shock hydrodynamics. *J. Comp. Phys.* **82**, 64–84.
10. Berger, M., and LeVeque, R. (1998). Adaptive mesh refinement using wave-propagation algorithms for hyperbolic systems. *SIAM J. Numer. Anal.* **35**(6), 2298–2316.
11. Berger, M., and Oliger, J. (1984). Adaptive mesh refinement for hyperbolic partial differential equations. *J. Comp. Phys.* **53**, 484–512.
12. Bihari, B., and Harten, A. (1995). Application of generalized wavelets: An adaptive multiresolution scheme. *J. Comp. Appl. Math.* **61**, 275–321.

13. Bihari, B., and Harten, A. (1997). Multiresolution schemes for the numerical solution of 2–D conservation laws I. *SIAM J. Sci. Comput.* **18**(2), 315–354.
14. Bihari, B., Ota, D., Liu, Z., and Ramakrishnan, S. (2002). The multiresolution method on general unstructured meshes. *AIAA J.* **40**(7), 1323–1330.
15. Bramkamp, F., Gottschlich-Müller, B., Hesse, M., Lamby, P., Müller, S., Ballmann, J., Brakhage, K.H., and Dahmen, W. (2003). *H*-adaptive multiscale schemes for the compressible Navier–Stokes equations – polyhedral discretization, data compression and mesh generation. In Ballmann, J. (ed.), *Flow Modulation and Fluid-Structure-Interaction at Airplane Wings, Numerical Notes on Fluid Mechanics*, Vol. 84, Springer, Berlin, pp. 125–204.
16. Bramkamp, F., Lamby, P., and Müller, S. (2004). An adaptive multiscale finite volume solver for unsteady an steady state flow computations. *J. Comp. Phys.* **197**(2), 460–490.
17. Carnicer, J., Dahmen, W., and Peña, J. (1996). Local decomposition of refinable spaces and wavelets. *Appl. Comput. Harmon. Anal.* **3**, 127–153.
18. Chiavassa, G., and Donat, R. (2001). Point value multiresolution for 2D compressible flows. *SIAM J. Sci. Comput.* **23**(3), 805–823.
19. Chiavassa, G., Donat, R., and Marquina, A. (2002). Fine–mesh numerical simulations for 2D riemann problems with a multilevel scheme. In Warnecke, G., Freistühler, H. (eds.), Hyperbolic Problems: Theory, Numerics, Applications, Birkhäuser, pp. 247–256.
20. Cohen, A., Dyn, N., Kaber, S., and Postel, M. (2000). Multiresolution finite volume schemes on triangles. *J. Comp. Phys.* **161**, 264–286.
21. Cohen, A., Kaber, S., Müller, S., and Postel, M. (2003). Fully Adaptive Multiresolution Finite Volume Schemes for Conservation Laws. *Math. Comp.* **72**(241), 183–225.
22. Cohen, A., Kaber, S., and Postel, M. (2002). Multiresolution analysis on triangles: application to gas dynamics. In Warnecke, G., Freistühler, H. (eds.), *Hyperbolic Problems: Theory, Numerics, Applications*, Birkhäuser, pp. 257–266.
23. Dahmen, W., Gottschlich–Müller, B., and Müller, S. (2000). Multiresolution schemes for conservation laws. *Numer. Math.* **88**(3), 399–443.
24. Dahmen, W., Müller, S., and Voß, A. (2005). Riemann problem for the Euler equations with non-convex equation of state including phase transitions. In Warnecke, G.(ed.), *Analysis and Numerics for Conservation Laws*, Springer, Berlin, pp. 137–162.
25. Dawson, C., and Kirby, R. (2001). High resolution schemes for conservation laws with locally varying time steps. *SIAM J. Sci. Comput.* **22**(6), 2256–2281.
26. Engquist, B., and Osher, S. (1981). One-sided difference approximations for nonlinear conservation laws. *Math. Comp.* **36**, 321–352.
27. Gottschlich–Müller, B., and Müller, S. (1999). Adaptive finite volume schemes for conservation laws based on local multiresolution techniques. In Fey, M., and Jeltsch R. (eds.), *Hyperbolic Problems: Theory, Numerics, Applications*, Birkhäuser, pp. 385–394.
28. Harten, A. (1995). Multiresolution algorithms for the numerical solution of hyperbolic conservation laws. *Comm. Pure Appl. Math.* **48**(12), 1305–1342.
29. Harten, A. (1996). Multiresolution representation of data: a general framework. *SIAM J. Numer. Anal.* **33**(3), 1205–1256.
30. Harten, A., Engquist, B., Osher, S., and Chakravarthy, S. (1987). Uniformly high order accurate essentially non–oscillatory schemes III. *J. Comp. Phys.* **71**, 231–303.
31. Houston, P., Mackenzie, J., Süli, E., and Warnecke, G. (1999). A posteriori error analysis for numerical approximations of Friedrichs systems. *Numer. Math.* **82**, 433–470.
32. Jameson, L. (2003). AMR vs high order schemes. *J. Sci. Comput.* **18**(1), 1–24.
33. Kröner, D., and Ohlberger, M. (1999). A posteriori error estimates for upwind finite volume schemes for nonlinear conservation laws in multi dimensions. *Math. Comp.* **69**(229), 25–39.

34. Lamby, P., Massjung, R., Müller, S., and Stiriba, Y. (2005). Inviscid flow on moving grids with multiscale space and time adaptivity (2005). Submitted for publication in Proceedings of 6th European Conference on Numerical Methods and Advanced Mathematics, July, 18–22, Santiago de Compostela, Spain.
35. Lamby, P., Müller, S., and Stiriba, Y. (2005). Solution of shallow water equations using fully adaptive multiscale schemes. *Int. J. Numer. Methods Fluids* **49**(4), 417–437.
36. Müller, S. (2002). Adaptive multiresolution schemes. In Herbin, B., and Kröner, D. (eds.), *Finite Volumes for Complex Applications*, Hermes Science, Paris.
37. Müller, S. (2002). Adaptive multiscale schemes for conservation laws. *Lecture Notes on Computational Science and Engineering*, vol. 27, Springer, Berlin.
38. Müller, S., and Stiriba, Y. (2004). Fully adaptive multiscale schemes for conservation laws employing locally varying time stepping. IGPM–Report 238, RWTH Aachen.
39. Osher, S., and Sanders, R. (1983). Numerical approximations to nonlinear conservation laws with locally varying time and space grids. *Math. Comp.* **41**, 321–336.
40. Rault, A., Chiavassa, G., and Donat, R. (2003). Shock-vortex interactions at high Mach numbers. *J. Sci. Comput.* **19**, 347–371.
41. Roussel, O., Schneider, K., Tsigulin, A., and Bockhorn, H. (2003). A conservative fully adaptive multiresolution algorithm for parabolic PDEs. *J. Comput. Phys.* **188**(2), 493–523.
42. Sonar, T.V., and Hannemann, D.H. (1994). Dynamic adaptivity and residual control in unsteady compressible flow computation. *Math. Comput. Model.* **20**, 201–213.
43. Sonar, T.E.S. (1998). A dual graph–norm refinement indicator for finite volume approximations of the Euler equations. *Numer. Math.* **78**, 619–658.
44. Tang, H., and Warnecke, G. (2006). A class of high resolution schemes for hyperbolic conservation laws and convection-diffusion equations with varying time and space grids. *J. Comp. Math.* **24**(2), 121–140.
45. Trompert, R., and Verwer, J. (1993). Analysis of the implicit Euler local uniform grid refinement method. *SIAM J. Sci. Comput.* **14**(2), 259–278.