Available online at www.sciencedirect.com
ScienceDirect
Journal of Computational Physics 227 (2008) 3758–3780
JOURNAL OF COMPUTATIONAL PHYSICS
www.elsevier.com/locate/jcp

An adaptive multiresolution scheme with local time stepping for evolutionary PDEs

Margarete O. Domingues [a,b,*], Sônia M. Gomes [c], Olivier Roussel [d], Kai Schneider [b,e]

[a] *Laboratório Associado de Computação e Matemática Aplicada (LAC), Coordenadoria de Laboratórios Associados, Instituto Nacional de Pesquisas Espaciais (INPE), Av. dos Astronautas 1758, 12227-010 São José dos Campos, Brazil*
[b] *Laboratoire de Modélisation et Simulation Numérique en Mécanique et Génie des Procédés (MSNM-GP), CNRS and Universités d'Aix-Marseille, 38 rue Frédéric Joliot-Curie, 13451 Marseille Cedex 20, France*
[c] *Universidade Estadual de Campinas, IMECC, Caixa Postal 6065, 13083-970 Campinas SP, Brazil*
[d] *Institut für Technische Chemie und Polymerchemie (TCP), Universität Karlsruhe, Kaiserstrasse 12, 76128 Karlsruhe, Germany*
[e] *Centre de Mathématiques et d'Informatique (CMI), Université de Provence, 39 rue Frédéric Joliot-Curie 13453 Marseille Cedex 13, France*

Received 3 April 2007; received in revised form 24 September 2007; accepted 21 November 2007
Available online 23 December 2007

Abstract

We present a fully adaptive numerical scheme for evolutionary PDEs in Cartesian geometry based on a second-order finite volume discretization. A multiresolution strategy allows local grid refinement while controlling the approximation error in space. For time discretization we use an explicit Runge–Kutta scheme of second-order with a scale-dependent time step. On the finest scale the size of the time step is imposed by the stability condition of the explicit scheme. On larger scales, the time step can be increased without violating the stability requirement of the explicit scheme. The implementation uses a dynamic tree data structure. Numerical validations for test problems in one space dimension demonstrate the efficiency and accuracy of the local time-stepping scheme with respect to both multiresolution scheme with global time stepping and finite volume scheme on a regular grid. Fully adaptive three-dimensional computations for reaction–diffusion equations illustrate the additional speed-up of the local time stepping for a thermo-diffusive flame instability.

MSC: 65M50; 80A32; 76V05

Keywords: Finite volume; Adaptivity; Multiresolution; Evolutionary partial differential equation

* Corresponding author. Address: Laboratório Associado de Computação e Matemática Aplicada (LAC), Coordenadoria de Laboratórios Associados, Instituto Nacional de Pesquisas Espaciais (INPE), Av. dos Astronautas 1758, 12227-010 São José dos Campos, Brazil. Tel.: +55 12 3945 6542; fax: +55 12 3945 6375.

E-mail addresses: mo.domingues@lac.inpe.br (M.O. Domingues), soniag@ime.unicamp.br (S.M. Gomes), roussel@ict.uni-karlsruhe.de (O. Roussel), kschneid@cmi.univ-mrs.fr (K. Schneider).

0021-9991/$ - see front matter
doi:10.1016/j.jcp.2007.11.046

1. Introduction

Systems of nonlinear partial differential equations (PDEs) naturally arise from mathematical modeling of chemical–physical problems encountered in many applications, like in meteorology or chemical industry. In turbulent reactive or non-reactive flows, for instance, the solutions of these PDEs typically exhibit a multitude of active spatial and temporal scales. However, since these scales are mostly not uniformly distributed in the space-time domain, efficient numerical discretizations could take advantage of this property. Introducing some kind of adaptivity in space-time allows to reduce the computational complexity with respect to uniform discretizations, while controlling the accuracy of the adaptive discretization.

Up to the present, different approaches have been investigated to define adaptive space discretizations, some emerge from *ad hoc* criteria, others are based on more sophisticated *a posteriori* error estimators using control strategies by solving computational expensive adjoint problems [3,38]. Adaptive mesh refinement (AMR) methods introduced by Berger et al. [7] are now widely used for many applications using structured or unstructured grids, e.g., see [4,5]. However, the data compression rate is high where the solution is almost constant, but remains low where the solution is smooth.

More recently, multiresolution based schemes (MR) were developed by Harten [22,23], first for conservation laws. Harten's approach has then been extended and further developed in different directions by Cohen et al. [14], Kaibara and Gomes [27], Chiavassa and Donat [11], Müller [29], and Roussel et al. [35,36]. The main idea of the MR method is to use a multiresolution data representation. The decay of the MR coefficients yields information on local regularity of the solution. Therewith the truncation error can be estimated and coarser grids can be used in regions, where this error is small and the solution is smooth. An adaptive grid can be introduced by suitable thresholding where only significant wavelet coefficients are retained. Hence a given discretization on a uniform mesh can be accelerated as the number of costly flux evaluations is significantly reduced, while maintaining the accuracy of the discretization. The memory requirements could also be reduced, for example using a dynamic tree data structure. An overview of the different MR methods can be found, e.g., in the books of Cohen [12] and Müller [29]. A comparison between the AMR method and the adaptive multiresolution approach has been described in [13].

A bottleneck of most of these space-adaptive methods, which typically employ explicit or semi-explicit time discretizations, is that the finest spatial grid size imposes a small time step in order to fulfill the stability criterion of the time scheme. Hence, for extensive grid refinement with a huge number of refinement levels, a very small size of the time step is implied. To overcome this difficulty different strategies have been pursued to introduce adaptive time stepping for space-adaptive discretizations of PDEs. Osher and Sanders [31] introduced local time stepping for one-dimensional scalar conservation laws, where the space discretization is non-uniform but fixed. Extensions of this approach have been presented in [16] for second-order Runge–Kutta schemes using a predictor–corrector type local time stepping, which has been further improved by Tang and Warnecke [39]. Space-time mesh refinement for the one-dimensional wave equation based on the conservation of a discrete energy is proposed in [15]. Two different approaches exist in the AMR context originally proposed by Berger and Oliger [7]. For the computation of stationary solutions, a time step is used in each subdomain, without synchronization. The instationary solution is of course inconsistent, but becomes consistent when the stationary solution is reached. For the computation of instationary solutions, in the papers of Berger et al. [7,6] and also in the Ph.D. thesis of Quirk [33], only one-stage time integrations are used, either explicit or implicit. Multi-stage methods have been considered in the AMR context so far for 1D simulations only [20]. In [17,18,25], local time-stepping algorithms for discontinuous Galerkin methods are presented. In [17], each element uses its optimal time step given by the local stability condition without requiring synchronization between the elements. In the context of adaptive wavelet methods, Bacry et al. [2] first introduced a scale-dependent time step. They applied this method to linear parabolic equations and to the Burgers equation. A stability analysis of this scheme has been conducted for the heat equation in [10]. It is shown that the adaptive time-stepping strategy does not affect the stability of the scheme. More recently, Müller and Stiriba [30] presented a fully adaptive multiresolution finite volume scheme with a locally varying time stepping. For time discretization, one stage methods, either explicit or implicit Euler schemes are used. A linear combination, leading to a Crank–Nicholson scheme, yields second-order accuracy. Applications for one-dimensional conservation laws illustrate the efficiency and accuracy of the scheme. A pure space-time Galerkin approach for viscous Burgers equation where the time axis is treated like a space direction has been introduced by Alam et al. [1]. Results for one space dimension look promising, however the extension of this method to higher dimensions seems questionable as it could be expensive in memory storage.

The aim of the present paper is to develop local scale-dependent time stepping for the space-adaptive multiresolution scheme introduced in [35,36]. The idea is to obtain additional speed-up of this efficient space-adaptive scheme by introducing larger time steps at large scales, without violating the stability condition of the explicit time scheme. This procedure reduces the number of flux evaluations, due to larger time steps. The originality of our paper is to combine, in the framework of multidimensional cell-average multiresolution analysis, a local time stepping together with multi-stage Runge–Kutta methods. The synchronization we propose differs from the one introduced in [30], where one-stage methods were used.

The remainder of the paper is organized as follows. Section 2 summarizes the finite volume (FV) discretization for uniform grids. Section 3 is dedicated to the formulation of local time-stepping strategies to be used in combination with adaptive multiresolution finite volume schemes. In a simplified setting of a two-block grid, formed by two uniform grids with different grid sizes, we consider the matrix method for stability analysis of such local time-stepping schemes applied to a convection–diffusion equation in one space dimension. In Section 4, we describe the space-adaptive multiresolution finite volume method. The multiresolution local time-stepping algorithm (MR/LTS) is described in Section 5. In Section 6, different applications of this new adaptive method are presented and compared with the results obtained using FV schemes on a regular grid and multiresolution (MR) schemes with global time stepping. Their accuracy, CPU time and memory compression are discussed. We show results for the convection–diffusion equation, for the one-dimensional compressible Euler equations and for the reaction–diffusion equations in one and three space dimensions. For the latter we present a numerical simulation of a thermodiffusive flame instability in the cellular regime. Finally, conclusions are drawn and perspectives for future investigations are discussed. In the Appendix, the algorithms are presented in a form that emphasizes the way they are actually implemented within the data structure.

2. Finite volume discretization on uniform grids

We consider parabolic conservation laws in Cartesian geometry for $(x,t) \in \Omega \times [0,+\infty), \Omega \subset \mathcal{R}^d$, of the form,

$$\frac{\partial u}{\partial t} = \mathcal{D}(u, \nabla u), \tag{1}$$

with initial condition $u(x,0) = u_0(x)$, and appropriate boundary conditions. We shall consider operators of type $\mathcal{D}(u,\nabla u) = -\nabla \cdot F(u,\nabla u) + S(u)$ formed by *divergence* and *source* terms. For the applications of this paper, the flux function contains advective and diffusive parts of the form $F(u,\nabla u) = f(u) - \nu\nabla u$, with constant diffusion coefficient $\nu \geqslant 0$. The numerical model has two basic aspects: the spatial and the temporal discretization.

For the spatial discretization, in the classical finite volume formulation, we consider the computational domain composed of the union of cells $\{\Omega_i\}_{i=1}^{N}$. In each cell Ω_i, with boundary $\partial\Omega_i$, external normal η_i and volume $|\Omega_i|$, we integrate Eq. (1) using a quadrature formula, and we get

$$\frac{d\bar{u}_i}{dt} = \overline{\mathcal{D}}_i(\overline{U}(t)), \tag{2}$$

where $\overline{U}(t) = (\bar{u}_i(t))$ contains the cell-averages of the numerical solution on the computational mesh at instant t, such that

$$\bar{u}_i(t) \approx \frac{1}{|\Omega_i|}\int_{\Omega_i} u(x,t)\,dx,$$

and

$$\overline{\mathcal{D}}_i(\overline{U}(t)) \approx -\frac{1}{|\Omega_i|}\int_{\partial\Omega_i} F(u,\nabla u)\cdot\eta_i\,dx + \overline{S}_i(u).$$

A variety of methods can be distinguished in the literature, mainly, by the way the numerical flux is defined for the approximation of the flux contribution. Typically, advective and diffusive terms are approximated differently. For the advective part, we use either second-order centered or upwind AUSM + schemes [28] whereas, for the diffusive part, we always choose a second-order centered scheme. The source term is approximated by $\overline{S}_i \approx S(\bar{u}_i)$. After space discretization by means of the finite volume scheme, the result is a system of ordinary equations,

$$\frac{\mathrm{d}\overline{U}}{\mathrm{d}t}(t) = \overline{\mathcal{D}}(\overline{U}(t)), \tag{3}$$

where we assume that the action of the operator $\overline{\mathcal{D}}$ also includes the enforcement of boundary conditions. For a sequence of discrete time values $t_n = n\Delta t$, let $\overline{U}^n$ be the approximation at t_n. At the next time step, $\overline{U}^{n+1}$ is obtained by the application of a discrete evolution operator $\mathbf{E} = \mathbf{E}(\Delta t)$ such that

$$\overline{U}^{n+1} = \mathbf{E}\overline{U}^n,$$

where $\mathbf{E}$ includes the application of the spatial differential operator $\overline{\mathcal{D}}$ and the discretization in time by means of some ODE solver. For the application of the present paper we use the second-order Runge–Kutta (RK2) scheme,

$$\overline{U}^{n+1} = \overline{U}^n + \frac{\Delta t}{2}[\overline{\mathcal{D}}(\overline{U}^n) + \overline{\mathcal{D}}(\overline{U}^n + \Delta t\overline{\mathcal{D}}(\overline{U}^n))]. \tag{4}$$

3. Local time stepping: spectrum analysis

Explicit time discretizations suffer typically from the limitations of the time step due to stability reasons, e.g., for advection problems we have a CFL condition which requires that $\Delta t \propto \Delta x$, and for diffusive terms we even have $\Delta t \propto \Delta x^2$. Evolving the cell-averages on an adaptive grid with a global time step implies that the time step is limited by the stability condition of the finest scale, while for coarser scales, the time step could in principle be larger. Hence, the time step may become very small in the case where many scales are present in the solution. The basic idea for local scale-dependent time stepping is to use large time steps to advance the solution at large scales, while small time steps are used to advance the solution at fine scales. The reason is to reduce the number of operations if fine scales are only required locally, provided the stability constraint can be preserved in some extent.

For sake of clarification, and inspired by the method proposed in [20], we shall first state and analyze this kind of strategy for a convection–diffusion equation in one space dimension using a simplified setting of a grid formed by two blocks of uniform grids with different mesh sizes. Nevertheless, it should be mentioned that for a fully adaptive numerical discretization a more sophisticated analysis is required. When the eigenmodes of the problem are preserved, this simplified analysis could be applied, but this necessitates that the grids remain the same for a substantial number of time steps. For given velocity $c > 0$ and diffusion coefficient $v > 0$, we consider the problem,

$$\frac{\partial u}{\partial t} + c\frac{\partial u}{\partial x} = v\frac{\partial^2 u}{\partial x^2}, \quad x \in \Omega, \quad t > 0,$$

with $\Omega = [-1, 1]$, initial condition $u(x, 0) = u_0(x)$ and homogeneous Dirichlet boundary conditions $u(-1, t) = 0$ and $u(1, t) = 0$. We consider two schemes:

(1) a *reference single step scheme* of the form $\overline{U}^{n+1} = \mathbf{E}\,\overline{U}^n$, corresponding to a FV/RK2 discretization on a regular grid;
(2) a *local time-stepping scheme* $\overline{W}^{n+1} = \mathbf{E}_{\mathrm{LTS}}\overline{W}^n$, where $\overline{W}^n$ are vectors formed by cell-averages on two blocks of uniform grids in which the mesh size in the coarse part is twice the mesh size in the fine part. The local time-stepping strategy consists in evolving the coarse cells using FV/RK2 with time step $2\Delta t$, where Δt is the step used to advance the fine cell-averages. Interpolation of cell averages is used at the interface between the two blocks at intermediate stages.

For these two schemes, we apply the matrix method for stability analysis by requiring that the spectral radius of the iteration matrices $\mathbf{E}$, and $\mathbf{E}_{\mathrm{LTS}}$ are smaller than one. Although this method may not give sufficient stability criteria, as indicated in [24], our purpose is to visualize the stability effect of non-uniformness of time and space steps.

3.1. Reference scheme on a uniform grid

For a uniform partition of the interval $[-1, 1]$ with grid size $\Delta x = \frac{1}{N}$, let $\bar{u}_i(t)$ be approximations of the cell-averages of the solution on the cells $\Omega_i = [(i-\frac{1}{2})\Delta x, (i+\frac{1}{2})\Delta x]$, for $-N+1 \leqslant i \leqslant N-1$, with boundary conditions $\bar{u}_{-N} = \bar{u}_N = 0$. The FV semi-discrete model can be written in the conservative form,

$$\frac{\mathrm{d}\bar{u}_i}{\mathrm{d}t} = -\frac{1}{\Delta x}[F_{i+\frac{1}{2}} - F_{i-\frac{1}{2}}],$$

where $F_{i+\frac{1}{2}} \approx F(u((i+\frac{1}{2})\Delta x, t))$ denotes the numerical flux. For this model problem, we consider the second-order numerical flux,

$$F_{i+\frac{1}{2}} = c\frac{\bar{u}_{i+1} + \bar{u}_i}{2} - v\frac{\bar{u}_{i+1} - \bar{u}_i}{\Delta x},$$

which is equivalent to a central finite difference scheme

$$\frac{\mathrm{d}\bar{u}_i}{\mathrm{d}t} = -\frac{1}{\Delta x}\left[c\frac{\bar{u}_{i+1} - \bar{u}_{i-1}}{2} - v\frac{\bar{u}_{i+1} - 2\bar{u}_i + \bar{u}_{i-1}}{\Delta x}\right].$$

Introducing the mesh Reynolds number $Re = \frac{c\Delta x}{v}$, the scheme can be expressed in the form,

$$\frac{\mathrm{d}\bar{u}_i}{\mathrm{d}t} = \alpha\bar{u}_{i-1} + \beta\bar{u}_i + \delta\bar{u}_{i+1},$$

where $\beta = -\frac{2v}{\Delta x^2}$, $\alpha = \frac{v}{2\Delta x^2}[2 + Re]$ and $\delta = \frac{v}{2\Delta x^2}[2 - Re]$. For time integration, using RK2 scheme, we obtain the evolution operator,

$$\overline{U}^{n+1} = \mathbf{E}\overline{U}^n, \tag{5}$$

where $\mathbf{E} = (I + \Delta t\mathbf{S} + \frac{\Delta t^2}{2}\mathbf{S}^2) = \varphi(\Delta t\mathbf{S})$, with $\mathbf{S} = \mathbf{S}(\Delta x)$ being the $(2N-1)\times(2N-1)$ tri-diagonal matrix with β, α and δ values on the main, lower and upper diagonals, respectively. The matrix $\mathbf{S}$ is diagonalizable with eigenvalues λ_i which can be expressed in terms of the mesh Reynolds number Re and the CFL number $\sigma = c\frac{\Delta t}{\Delta x}$, by the formula,

$$\Delta t\lambda_i = -\frac{\sigma}{Re}\left(2 - \sqrt{4 - Re^2}\cos\left(\frac{i\pi}{2N}\right)\right). \tag{6}$$

Therefore, the eigenvalues of $\mathbf{E} = \varphi(\Delta t\mathbf{S})$ are given by $\varphi(\Delta t\lambda_i)$, and the spectral radius $\rho(\mathbf{E}) = \rho(\varphi(\Delta t\mathbf{S}))$ is less than one, if and only if $|\varphi(\Delta t\lambda_i)| < 1$. The curve (b) in Fig. 2 (left) corresponds to the isoline in the $Re \times \sigma$-plane of the maximum absolute eigenvalue being equal to one, i.e., $\max_i|\varphi(\Delta t\lambda_i)| = 1$, where $\Delta t\lambda_i$ is obtained from (6) using $N = 50$. For the scheme (5), the von Neumann stability region is indicated by curve (a) in Fig. 2 (left), which is a sufficient stability condition for periodic boundary problems. The estimation of this region proposed in [8] is $\sigma \leqslant \frac{Re}{2}$ for $Re \leqslant 2\sqrt{3}$, and $Re \leqslant \frac{6}{\sigma}$ for $Re > 2\sqrt{3}$, as indicated by curve (c) in Fig. 2 (left). For a pure advection problem, i.e., $v = 0$, the eigenspectrum $\lambda_j = \imath\frac{c}{\Delta x}\cos(\pi\frac{\pi j}{2N})$ of matrix $\mathbf{S}$ is purely imaginary,[1] meaning that the combination of this central finite difference scheme and RK2 is unstable.

3.2. Two-block grid with local time stepping

We consider now a grid formed by two blocks of uniform grids with different mesh sizes. The right block is a fine grid with spacing $\Delta x_1 = \Delta x$, and the left block is a coarse grid with double mesh spacing $\Delta x_0 = 2\Delta x$

[1] where $\imath = \sqrt{-1}$.

(Fig. 1). Let $\bar{u}_{0,i}$ denote the cell-averages on the coarse cells $\Omega_{0,i} = [(\mathrm{i}-\frac{1}{2})\Delta x_0, (\mathrm{i}+\frac{1}{2})\Delta x_0]$, and $\bar{u}_{1,i}$ denote the cell-averages on the fine cells $\Omega_{1,i} = [\mathrm{i}\Delta x_1, (\mathrm{i}+1)\Delta x_1]$.

Outside the interface at $x = \Delta x_1$, we compute the numerical flux using the mesh spacing of each block to get

$$\frac{\mathrm{d}\bar{u}_{0,i}}{\mathrm{d}t} = \alpha_0\bar{u}_{0,i-1} + \beta_0\bar{u}_{0,i} + \delta_0\bar{u}_{0,i+1}, \quad i \leqslant -1$$
$$\frac{\mathrm{d}\bar{u}_{1,i}}{\mathrm{d}t} = \alpha_1\bar{u}_{1,i-1} + \beta_1\bar{u}_{1,i} + \delta_1\bar{u}_{1,i+1}, \quad i \geqslant 2,$$

where $\alpha_1 = \alpha, \beta_1 = \beta, \delta_1 = \delta$, while $\alpha_0 = \frac{v}{4\Delta x^2}[1 + Re]$, $\beta_0 = -\frac{v}{2\Delta x^2}$ and $\delta_0 = \frac{v}{4\Delta x^2}[1 - Re]$ are the coefficients associated to the coarse grid block, which are expressed in terms of the parameters Re and σ associated with the fine grid. For the calculation of $\frac{\mathrm{d}\bar{u}_{1,1}}{\mathrm{d}t}$ and $\frac{\mathrm{d}\bar{u}_{0,0}}{\mathrm{d}t}$, the numerical flux $F_{\frac{1}{2}}$ at the interface $x = \Delta x_1$ is required. It is defined using cell-averages on the fine grid

$$F_{\frac{1}{2}} = \left(\frac{c}{2} - \frac{v}{\Delta x}\right)\bar{u}_{1,0} + \left(\frac{c}{2} + \frac{v}{\Delta x}\right)\bar{u}_{1,-1}.$$

To apply this formula, we approximate the virtual cell-average $\bar{u}_{1,0}$ using central Lagrange interpolation of degree 2, and compute the virtual cell-averages $\bar{u}_{0,1}$ by the exact value $\bar{u}_{0,1} = \frac{\bar{u}_{1,1}+\bar{u}_{1,2}}{2}$ to obtain

$$\bar{u}_{1,0} = \bar{u}_{0,0} - \frac{1}{8}\bar{u}_{0,-1} + \frac{\bar{u}_{1,1} + \bar{u}_{1,2}}{16}.$$

Therefore, at the grid interface we get

$$\frac{1}{\Delta x}F_{\frac{1}{2}} = \left(-\delta_1 + \frac{\alpha_1}{16}\right)\bar{u}_{1,1} + \alpha_1\bar{u}_{0,0} - \frac{\alpha_1}{8}\bar{u}_{0,-1} + \frac{\alpha_1}{16}\bar{u}_{1,2}. \tag{7}$$

Using this formula, we obtain

$$\frac{\mathrm{d}\bar{u}_{0,0}}{\mathrm{d}t} = \left(\alpha_0 + \frac{\alpha_1}{16}\right)\bar{u}_{0,-1} + \frac{3\beta_0}{2}\bar{u}_{0,0} + \left(\frac{\delta_1}{2} - \frac{\alpha_1}{32}\right)\bar{u}_{1,1} - \frac{\alpha_1}{32}\bar{u}_{1,2}$$

and

$$\frac{\mathrm{d}\bar{u}_{1,1}}{\mathrm{d}t} = \left(\beta_1 + \frac{\alpha_1}{16}\right)\bar{u}_{1,1} + \left(\delta_1 + \frac{\alpha_1}{16}\right)\bar{u}_{1,2} + \alpha_1\bar{u}_{0,0} - \frac{\alpha_1}{8}\bar{u}_{0,-1}.$$

Defining $\overline{U}_l(t) = [\bar{u}_{l,i}(t)]$, and assuming that $\Delta x = \frac{1}{2N}$, this semi-discrete FV formulation can be expressed in matrix form as

$$\frac{\mathrm{d}}{\mathrm{d}t}\begin{bmatrix}\overline{U}_0\\ \overline{U}_1\end{bmatrix} = \begin{bmatrix}S_0 & B_0\\ B_1 & S_1\end{bmatrix}\begin{bmatrix}\overline{U}_0\\ \overline{U}_1\end{bmatrix}, \tag{8}$$

where

- S_0 is a $N \times N$ tri-diagonal matrix with elements β_0 on the main diagonal, α_0 on the lower diagonal and δ_0 on the upper diagonal, except in the last row where α_0 is replaced by $\alpha_0 + \frac{\alpha_1}{16}$ and β_0 is replaced by $\frac{3\beta_0}{2}$;
- S_1 is a $(2N-1) \times (2N-1)$ tri-diagonal matrix with elements values β_1 on the main diagonal, α_1 on the lower diagonal and δ_1 on the upper diagonal, except on the first row where β_1 is replaced by $\beta_1 + \frac{\alpha_1}{16}$ and δ_1 is replaced by $\delta_1 + \frac{\alpha_1}{16}$. Furthermore, on the last row, the last element β_1 is replaced by $\beta_1 - \delta_1$, due to the right boundary condition $\bar{u}_{1,2N} = -\bar{u}_{1,2N-1}$.

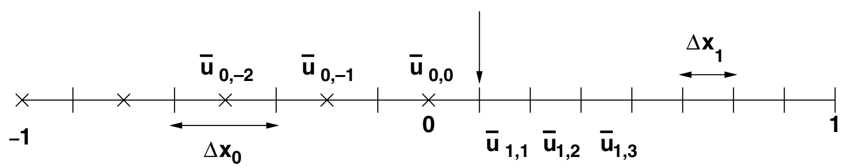

Fig. 1. Two-block grid.

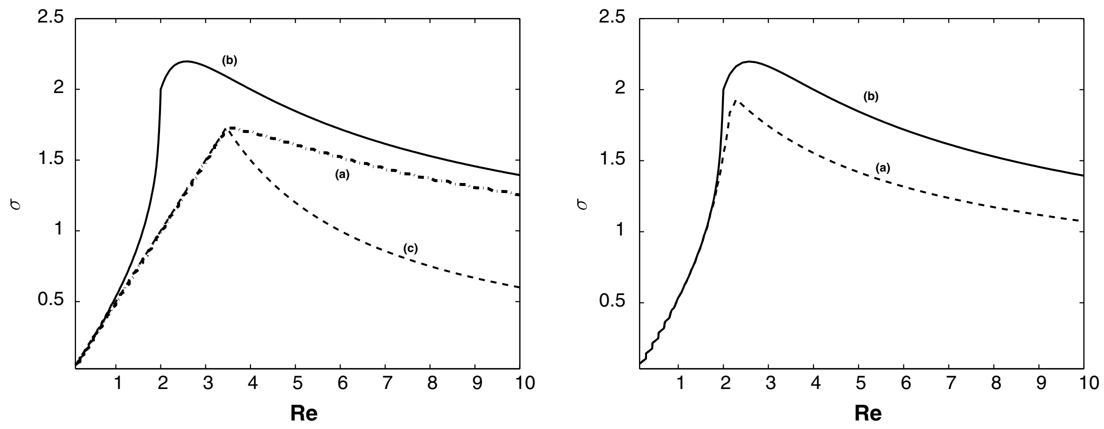

Fig. 2. Stability boundaries in the $Re \times \sigma$ plane: spectrum analysis for RK2 scheme. Left: uniform grid (a) von Neumann analysis, (b) Dirichlet BC, and (c) estimation of [8]. Right: (a) two-block grid with local time stepping, and (b) uniform fine grid, both with Dirichlet BC.

- B_0 is a $N \times (2N-1)$ matrix with all elements being equal to zero, except on the last row where the value of the first and second elements are given by $\frac{\delta_1}{2} - \frac{\alpha_1}{32}$ and $-\frac{\alpha_1}{32}$, respectively;
- B_1 is a $(2N-1) \times N$ matrix, where all elements are equal to zero, excepting the last two ones of the first row, which are given by $-\frac{\alpha_1}{8}$ and α_1, respectively.

To reduce CPU time, instead of integrating (8) with a global time step Δt, a local time-stepping procedure considers the use of two time steps. On the right side, with grid size $\Delta x_1 = \Delta x$, the cell-averages are evolved with time step $\Delta t_1 = \Delta t$, and on the left side, where the grid size is $\Delta x_0 = 2\Delta x$, we use a double time step $\Delta t_0 = 2\Delta t$. The time cycle from t^n to t^{n+2} is completed after the steps described in Section 5, and illustrated in Fig. 4. We observe that linear interpolation in time is performed to predict coarse virtual cell-averages at t^{n+1}, which are required for the time evolution on fine cells close to the interface. In summary, if $\overline{W}^n = [\overline{U}_0^n \overline{U}_1^n]^t$, where $\overline{U}_l^n$ are the vectors of the cell-averages $\bar{u}_{l,i}^n$ at the time n, this scheme can be expressed in matrix notation as follows:

$$\overline{W}^{n+2} = \mathbf{E}_{\text{LTS}} \overline{W}^n,$$

where the iteration matrix

$$\mathbf{E}_{\text{LTS}} = \begin{bmatrix} D_0 + D_1[\mathcal{T} + \mathcal{C}_1] \\ D_2[\mathcal{T}^2 + \mathcal{C}_2 + \mathcal{T}\mathcal{C}_1] \end{bmatrix}$$

is formed by the sub-matrices $D_0 = [(I + 2\Delta t S_0 + 2\Delta t^2 S_0^2), \Delta t(I + 2\Delta t S_0)B_0], D_1 = [\Delta t^2 B_0 B_1, \Delta t B_0(I + \Delta t S_1)]$, $D_2 = [0, I]$, which is a $(2N-1) \times (3N-1)$ matrix,

$$\mathcal{T} = \begin{bmatrix} \frac{1}{2}(I + \Delta t^2 B_0 B_1) & \frac{\Delta t}{2} B_0(I + \Delta t S_1) \\ \frac{\Delta t}{2}(I + \Delta t S_1)B_1 & [I + \Delta t S_1 + \frac{\Delta t^2}{2} S_1^2] \end{bmatrix}, \quad \mathcal{C}_s = \begin{bmatrix} \frac{1}{2}(I + \Delta t S_0)(I + s\Delta t S_0) & \frac{s\Delta t}{2}(I + \Delta t S_0)B_0 \\ \frac{\Delta t}{2} B_1(I + s\Delta t S_0) & \frac{s\Delta t^2}{2} B_1 B_0 \end{bmatrix}.$$

Fig. 2 (right) shows the spectrum stability curve (a) for algorithm RK2/LTS, which is the isoline where the maximum absolute eigenvalue of $\mathbf{E}_{\text{LTS}}$ is equal to one, computed numerically with $N = 20$. We can also note that this matrix stability region coincides with the one for a uniform fine grid (b) for $Re < 2$, but for larger values of Re, there is a reduction of about 20% of this region.

4. Adaptive multiresolution scheme

Here the goal consists in performing the finite volume model described in the Section 2 in a more economic fashion, by taking into account local regularity information about the numerical solution. For the implementation of this adaptive algorithm, the basic tools come from multiresolution wavelet analysis, which is denoted everywhere by the subscript MR.

The principle of the MR analysis is to represent a set of data given on a fine grid as values on a coarser grid plus a series of differences at different levels of nested dyadic grids. The differences contain the information of the solution when going from a coarse to a finer grid. In particular, these coefficients are small in regions where the solution is smooth and significant close to irregularities. Based on such a kind of information, a MR scheme considers the representation of the numerical solution $\overline{U}^n_{\mathrm{MR}} = \overline{U}^n_{L,\mathrm{MR}}$ on a sparse grid $\Gamma^n = \Gamma^n_L$, which is formed by the cells corresponding to significant wavelet coefficients. For the solution to evolve from $\overline{U}^n_{\mathrm{MR}}$ into $\overline{U}^{n+1}_{\mathrm{MR}}$, three basic steps are undertaken:

Refinement: $\overline{U}^{n+}_{\mathrm{MR}} \leftarrow \mathbf{R}\overline{U}^n_{\mathrm{MR}}$;
Evolution: $\overline{U}^{n+1}_{\mathrm{MR}} \leftarrow \mathbf{E}_{\mathrm{MR}}\overline{U}^{n+}_{\mathrm{MR}}$;
Coarsening: $\overline{U}^{n+1}_{\mathrm{MR}} \leftarrow \mathbf{T}(\epsilon)\overline{U}^{n+1}_{\mathrm{MR}}$.

The refinement operator $\mathbf{R}$ is a precautionary measure to account for possible translation or creation of finer scales in the solution between two subsequent time steps. Since the regions of smoothness or irregularities of the solution may change with time, the grid Γ^n may not be convenient anymore at the next time step t^{n+1}. Therefore, before doing the time evolution, the representation of the solution should be extended to a grid Γ^{n+}, which is expected to be a refinement of Γ^n, and to contain Γ^{n+1}. Then, a time evolution operator $\mathbf{E}_{\mathrm{MR}} = \mathbf{E}_{\mathrm{MR}}(\Delta t)$ is applied. Only the cell-averages on the computational grid Γ^{n+} are evolved in time, and the adaptive flux computation is adopted at interfaces of cells of different scale levels. Finally, a thresholding operation $\mathbf{T}(\epsilon)$ (coarsening) is applied in order to unrefine those cells in Γ^{n+} that are unnecessary for an accurate representation of $\overline{U}^{n+1}_{\mathrm{MR}}$.

4.1. Multiresolution representation of cell-averages

In the present paper, we adopt the cell-average multiresolution representation of Harten [22], which is briefly described in the following.

For the adaptive multiresolution representation of cell-averages we use a tree data structure. This structure is organized as a dynamic graded tree to compress data, while still being able to navigate through it. In the wavelet terminology, a graded tree structure corresponds to an adaptive approximation in which connectivity in the tree structure is always ensured, i.e., no hole is admitted inside the tree. We denote by Λ the set of indices of the existing tree nodes, by $\mathcal{L}(\Lambda)$ the restriction of Λ onto the leaves, and by Λ_l the restriction of Λ to a level $l, 0 \leqslant l < L$. For example, in the 1D case, $\Omega = \Omega_{0,0}$ is the root cell, and $\Omega_{l,i}$, $i \in \Lambda_l$, are the different node cells at level l. The refinement of a parent node cell $\Omega_{l,i}$ at level l produces two children nodes $\Omega_{l+1,2i}$, $\Omega_{l+1,2i+1}$ at level $\ell + 1$. We denote by $\bar{u}_{l,i}$ the cell-average value of the quantity u on the cell $\Omega_{l,i}$, and by $\overline{U}_l = (\bar{u}_{l,i})_{i\in\Lambda_l}$ the ensemble of the existing cell-average values at the level l. To compute the cell-averages of a level l from the ones of the level $l+1$, we use the *projection* operator $P_{l+1\to l} : \overline{U}_{l+1} \mapsto \overline{U}_l$ such that

$$\bar{u}_{l,i} = (P_{l+1\to l}\overline{U}_{l+1})_i = \frac{1}{2}(\bar{u}_{l+1,2i} + \bar{u}_{l+1,2i+1}). \tag{9}$$

To estimate the cell-averages of a level $l+1$ from the ones of the level l, we use the *prediction* operator $P_{l\to l+1} : \overline{U}_l \mapsto \widetilde{U}_{l+1}$. This operator gives an *approximation* $\widetilde{U}_{l+1}$ of $\overline{U}_{l+1}$ at the level $l+1$ by interpolation. This operator must satisfy two properties. First, it has to be *local*, i.e., the interpolation for a child is made from the cell-averages of its parent and the s nearest parent's neighbors in each direction. Second, it has to be *consistent with the projection*, i.e., $P_{l+1\to l} \circ P_{l\to l+1} = \mathrm{Id}$. The wavelet coefficients (details) are the differences between the exact and the predicted values,

$$\bar{d}_{l+1,2i+1} = \bar{u}_{l+1,2i+1} - \tilde{u}_{l+1,2i+1}. \tag{10}$$

If $\overline{D}_{l+1}$ is the vector of these wavelet coefficients, there is a one-to-one relation

$$\overline{U}_{l+1} \leftrightarrow (\overline{D}_{l+1}, \overline{U}_l).$$

Repeating this operation recursively on L levels, one gets the so-called multiresolution transform,

$$\overline{\mathbf{M}} : \overline{U}_L \mapsto (\overline{D}_L, \overline{D}_{L-1}, \ldots, \overline{D}_1, \overline{U}_0). \tag{11}$$

In conclusion, the knowledge of the cell-average values of all the leaves is equivalent to the knowledge of the cell-average value of the root and the details of all the other nodes of the tree structure. In the applications of this paper we use the third-order prediction scheme

$$\tilde{u}_{l+1,2i} = \bar{u}_{l,i} + \frac{1}{8}(\bar{u}_{l,i+1} - \bar{u}_{l,i-1}), \quad \tilde{u}_{l+1,2i+1} = \bar{u}_{l,i} - \frac{1}{8}(\bar{u}_{l,i+1} - \bar{u}_{l,i-1}), \tag{12}$$

which is exact for quadratic polynomials. For higher dimensions in Cartesian geometry, a tensor product approach is used. Details can be found in [36].

4.2. Conservative flux computation

In the graded tree structure, a leaf at level $l+1$ has not necessarily a neighbor at the same level. Therefore, for flux computations, virtual leaves have to be created. In such case, the flux at the interface is computed using the cell-average values of the adjacent virtual leaves, which are computed by prediction from the information at level l. In order to maintain the strict conservativity in flux computations, the ingoing flux for a parent cell (at level l) is taken as the sum of the fluxes going out of the adjacent leaves at level $l+1$ (Fig. 3), i.e.,

$$F_{l,i,j \to l,i+1,j} = F_{l+1,2i+1,2j \to l+1,2i+2,2j} + F_{l+1,2i+1,2j+1 \to l+1,2i+2,2j+1}.$$

Conservative flux evaluations in three dimensions are performed in a similar way. Details can be found in [36].

4.3. Tree updating

During the simulation, we have a dynamic tree. This means that the tree evolves in time: when needed, some nodes are added or removed. The tree updating is performed by two operations: the refinement procedure $\mathbf{R}$, which is performed by including children of some leaves in the tree structure, and the thresholding operator (coarsering) $\mathbf{T}(\epsilon)$, which consists in removing unnecessary leaves where details are smaller than a prescribed tolerance ϵ, while preserving the graded tree data structure. These two operations are performed by the following procedure:

I) For the whole tree, from the leaves to the root:
- Compute the details on the nodes $d_{l,i},\ i \in \Lambda_l$, by the multiresolution transform.
- Define the deletable cells, if the details on the corresponding nodes and their neighbors are smaller than the prescribed tolerance.

II) For the whole tree, from the leaves to the root:
- If a node and its children nodes are deletable, and the children nodes are simple leaves (i.e., without virtual children), then delete their children.
- If the node is not deletable, and it is not at the maximum level, then create the children for this node.

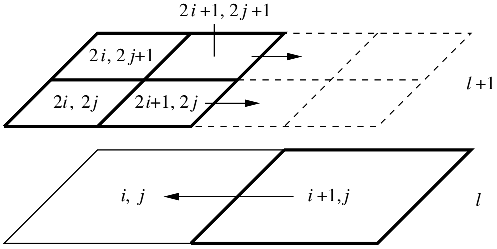

Fig. 3. Ingoing and outgoing flux computation in 2D for two different levels.

For more details on the implementation of these procedure we refer to [36].

5. Local scale-dependent time stepping

We propose a MR/LTS algorithm that uses the multiresolution framework described in the previous section. Assuming that Δt is the time step for cells in the finest scale level L, the main principle is that the cells at lower levels $L - l$ are evolved with time step $2^l \Delta t$. Consequently, if $\overline{U}^n_{\mathrm{LTS}}$ represents the numerical solution at $t^n = n\Delta t$ on the adaptive grid $\Gamma^n = \Gamma^n_L \subset \Omega_L$, then one complete time cycle of the local time-stepping evolution operator evolves the solution from t^n to t^{n+2^L}.

In Fig. 4, the basic ideas of the local scale-dependent RK2 time stepping are illustrated for one time cycle and two scale levels. First, we compute one step of RK2 with the local time step. In case the cell level is not on the finest level, we store its values and we use it to interpolate the cell-average value for the previous half local time step, that was not computed (see Fig. 4, top-left). Then, with these interpolation values, we perform the RK2 update on the values for the neighbors on the finest level if they exist (see Fig. 4, top-right). Following it, we return the cell-average value that we have already stored (see Fig. 4, bottom-left). Finally, we advance in time the finest levels and we perform the RK2 usual update (see Fig. 4, bottom-right).

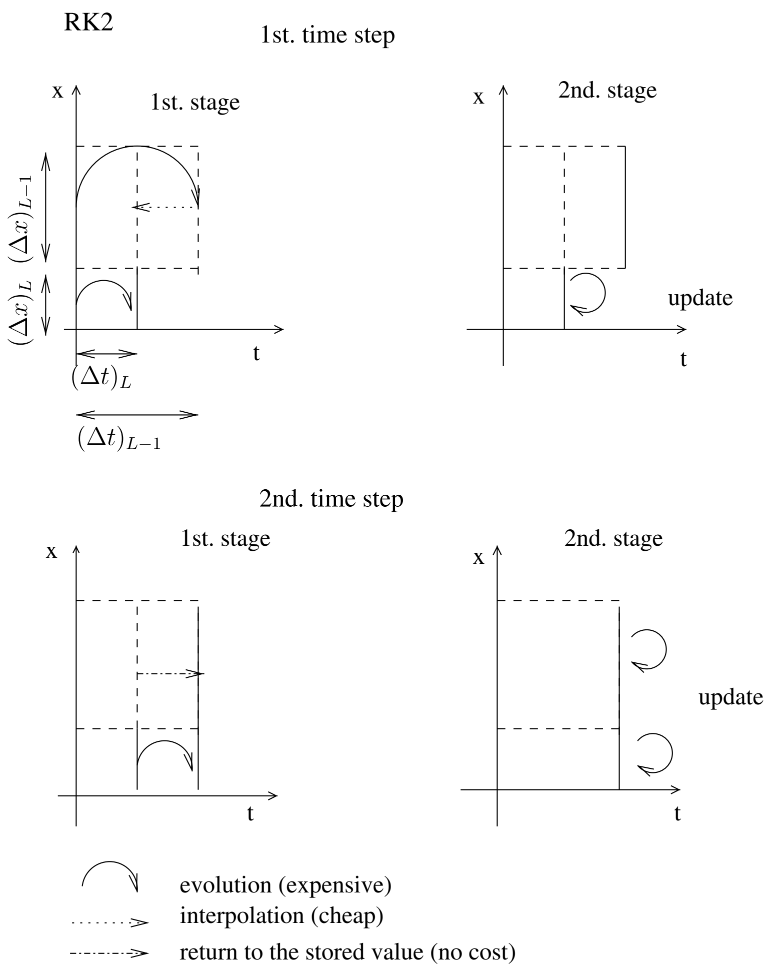

Fig. 4. Scheme of local scale-dependent time stepping.

For the whole adaptive tree, the time cycle can be summarized as follows:

I. Input leaves and time step (see Appendix, Algorithm 1).
II. Compute LTS time evolution cycle (see Appendix, Algorithm 2).
III. Adapt tree, update the tree values, combining or splitting them (see Appendix, Algorithm 5).

We recall that, since we are working with a spatially graded tree data structure, there is only one level difference between two neighbor cells. Furthermore, refinement operations are only allowed on the tree data structure when one complete time cycle on a given level is finished. On the other hand, coarsening of the mesh is forbidden during the LTS time cycle. In addition, we should mention that the employed interpolation procedure in time is not strictly conservative. Nevertheless, as shown in the next numerical results section, the accuracy of the FV scheme is mantained by the MR/LTS scheme.

To estimate the potential of the efficiency of the MR/LTS scheme with respect to the MR scheme, in the spirit of the analysis in [26,30], let $W_{\mathrm{MR/LTS}}$ and W_{MR} be the computational work required to do one time cycle using local and global time stepping, respectively. Taking into account only the number of flux computations, and assuming the adaptive grid does not change during the time cycle, we obtain

$$W_{\mathrm{MR}} \approx 2^L M, \quad W_{\mathrm{MR/LTS}} \approx \sum_{l=0}^{L} M_l 2^l,$$

where M is the total number of cells of the MR adaptive grid, and M_l is the number of cells at the level l. Defining the speed-up rate,

$$\Theta = \frac{W_{\mathrm{MR}} - W_{\mathrm{MR/LTS}}}{W_{\mathrm{MR}}},$$

we obtain $\Theta = \sum_{l=0}^{L} \alpha(l)(1 - 2^{l-L})$, where $\alpha(l) = M_l/M$ is the proportion of cells at level l. This formula reveals the dependence of the speed-up on the distribution of cells over the different levels. Since the factor $(1 - 2^{l-L})$ decreases with increasing l, a gain in efficiency may be expected if the majority of cells is placed on coarser levels. We can also express $\alpha(l) = \beta(\ell)\, 2^{d(l-L)}/\eta$, where d is the space dimension, $\eta = M/2^{dL}$ is the total compression rate of the MR grid, and $\beta(l) = M_l/2^{dl}$ is the compression rate at level l. Consequently, we have

$$\Theta = \frac{1}{\eta} \sum_{\ell=0}^{L} 2^{d(l-L)} (1 - 2^{l-L}) \beta(\ell).$$

For given compression rates η and $\beta(l)$, we can see the influence of the space dimension, since $2^{d(l-L)}$ is less significant for higher dimensions. Moreover, its influence is more sensitive at lower levels, where the contribuition of the term $(1 - 2^{l-L})$ becomes larger.

6. Numerical results

In this section, we present different numerical examples in one and three space dimensions using finite volume second-order accurate schemes with second-order Runge–Kutta (RK2) time integration. A multiresolution analysis for a cell-average of third-order prediction is used. In the following, three different methods, i.e., the finite volume reference scheme (FV), the adaptive multiresolution method (MR) and the adaptive multiresolution method with local time stepping (MR/LTS), are applied to a one-dimensional convection–diffusion equation for which the analytical solution is known and for which we conducted the stability analysis presented in Section 3. Then, we show computations for one-dimensional compressible Euler equations for a shock tube problem. Finally, we present computations for reaction–diffusion equations in one and three space dimensions, which correspond to a planar flame front and to a cellular instability inside a spherical flame initially stretched, respectively.

6.1. Convection–diffusion equation

We consider the linear convection–diffusion equation,

$$\frac{\partial u}{\partial t}+\frac{\partial u}{\partial x}=v\frac{\partial^2 u}{\partial x^2},\quad (x,t)\in[-1,1]\times[0,+\infty), \tag{13}$$

with initial condition $u(x,0)=u_0(x)$, and boundary values $u(-1,t)=1$ and $u(1,t)=0$. All the numerical solutions of this section are compared with the analytical solution (e.g., see [24])

$$u_{\mathrm{ex}}(x,t)=\frac{1}{2}\mathrm{erfc}\left(\frac{x-t}{2\sqrt{vt}}\right), \tag{14}$$

and the computations start at $u_0(x)=u_{\mathrm{ex}}(x,0.1)$, to avoid a discontinuous initial condition which would affect the accuracy. The tolerance parameter ϵ is chosen according to the formula given in [36]

$$\epsilon=\epsilon_L=5\times 10^8\frac{v2^{-3L}}{1+v2^{L+2}}. \tag{15}$$

Using $v=0.001$, and $L=9$, the plots in Fig. 5 represent the stability regions in the $Re\times\sigma$ plane for the FV, MR and MR/LTS methods obtained by checking the solution at $t=0.5$. For all schemes we performed computations for σ ranging from 0.125 to 1.875 in steps of 0.125 and for Re from 1 to 10 in steps of 1. The stability region for the FV scheme depicted in Fig. 5 (left) fits well with the theoretical one presented in Fig. 2 (left) (a). As shown in Fig. 5 (right), the stability domains for MR and MR/LTS schemes coincide. Moreover, they are quite similar to the one found for the FV scheme.

The MR/LTS discretization error $\|u_{\mathrm{ex}}-u_{\mathrm{LTS}}\|$ can be decomposed into different error contributions: the FV discretization error $\|u_{\mathrm{ex}}-u_{\mathrm{FV}}\|$, the thresholding error $\|u_{\mathrm{FV}}-u_{\mathrm{MR}}\|$ and the local time-stepping error $\|u_{\mathrm{MR}}-u_{\mathrm{LTS}}\|$. We can see in Table 1 that the MR/LTS discretization error is almost identical with the FV discretization error. The thresholding error and the local time-stepping error are two orders of magnitude smaller. Table 2 presents errors of the different schemes for different maximum scales L, showing that MR and MR/LTS computations yield the same second-order accuracy as the reference FV computation on the corresponding regular grid.

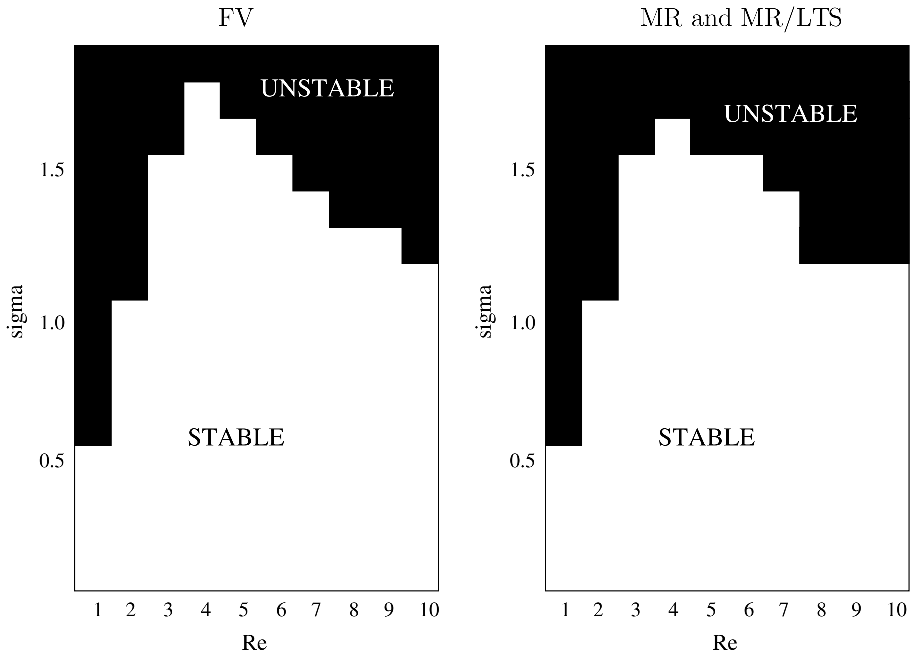

Fig. 5. Numerically computed stability regions for the convection–diffusion equation using centered FV(left), MR and MR/LTS (right) schemes, with $v=0.001$, and $L=9$.

Table 1
Convection–diffusion equation: contributions of the different errors at $t = 0.5$ for $\sigma = 0.5$, $\nu = 0.001$, $\epsilon = 1.2 \times 10^{-3}$ and $L = 9$

Terms	L_1 norm	L_2 norm	L_∞ norm
$\|u_{\mathrm{ex}} - u_{\mathrm{FV}}\|$	2×10^{-3}	2.96×10^{-4}	1.50×10^{-2}
$\|u_{\mathrm{FV}} - u_{\mathrm{MR}}\|$	3.10×10^{-5}	5.22×10^{-6}	3.80×10^{-4}
$\|u_{\mathrm{MR}} - u_{\mathrm{LTS}}\|$	4.27×10^{-5}	8.74×10^{-6}	9.15×10^{-4}
$\|u_{\mathrm{ex}} - u_{\mathrm{LTS}}\|$	1.99×10^{-3}	2.94×10^{-4}	1.55×10^{-2}

Table 2
Convection–diffusion: errors in the L_∞ and L_1 norms for FV, MR and MR/LTS methods obtained at $t = 0.5$ for $\sigma = 0.5$, $\nu = 0.001$, and $L = 9$ to 12

L	FV		MR		MR/LTS	
	L_1 norm	L_∞ norm	L_1 norm	L_∞ norm	L_1 norm	L_∞ norm
9	1.13×10^{-2}	4.79×10^{-4}	1.14×10^{-2}	2.33×10^{-3}	1.13×10^{-2}	2.39×10^{-3}
10	2.96×10^{-3}	1.25×10^{-4}	2.99×10^{-3}	6.57×10^{-4}	3.35×10^{-3}	7.01×10^{-4}
11	7.55×10^{-4}	3.17×10^{-5}	7.57×10^{-4}	1.83×10^{-4}	9.99×10^{-4}	2.24×10^{-4}
12	1.90×10^{-4}	7.99×10^{-6}	1.90×10^{-4}	4.58×10^{-5}	1.90×10^{-4}	4.60×10^{-5}

The gain in CPU time of the MR/LTS computation with respect to the MR one is illustrated in Fig. 6 (right) showing the significant gain of CPU time using the MR/LTS method, which increases with the number of levels. Moreover, it can be noticed that almost the same memory is required for both methods (Fig. 6, left).

6.2. Shock-tube problem

In the following, we consider the compressible Euler equations in one space dimension and we compute a shock-tube problem (e.g., see [37]). The governing equations read

$$\frac{\partial Q}{\partial t} + \frac{\partial F}{\partial x} = 0, \tag{16}$$

with $Q = (\rho, \rho u, \rho e)^t, F = (\rho u, \rho u^2 + p, (\rho e + p)u)^t$, where $\rho = \rho(x,t)$ is the density, $u = u(x,t)$ the velocity, $e = e(x,t)$ the energy per unit of mass and $p = p(x,t)$ the pressure. The system is completed by the equation of state for an ideal gas $p = (\gamma - 1)\rho(e - \frac{u^2}{2})$, where γ is the specific heat ratio. The initial condition is the one proposed by Sod [37]: $Q(x,0) = Q_L = (1, 0, 2.5)^t$, for $x < 0$, and $Q(x,0) = Q_R = (0.125, 0, 0.25)^t$, for $x > 0$. For

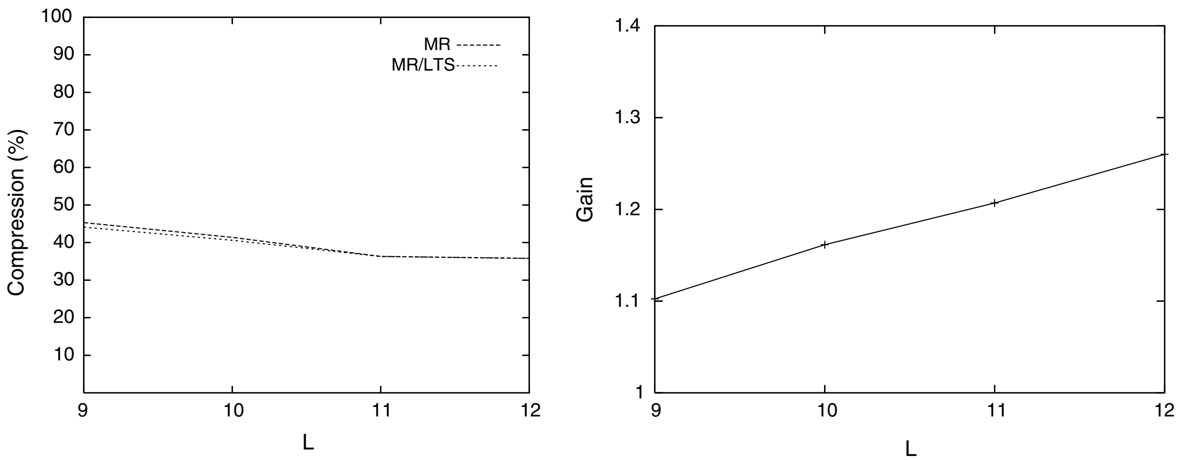

Fig. 6. Convection–diffusion: memory compression for both MR and MR/LTS methods (left), and gain in CPU time of the MR/LTS method in comparison with the MR method (right) for $\sigma = 0.5$, $\nu = 0.001$, and $L = 9$ to 12.

the simulations, the computational domain is $\Omega = [-1, 1]$, and Neumann boundary conditions are applied on both sides. For the numerical flux, we use the classical TVD second-order AUSM + scheme, together with the Van Albada limiter [28]. The physical parameters are $Ma = 1, \gamma = 1.4$, and the computations are performed until physical time $t = 0.5$. The time step is computed with $\sigma = 0.5$, and the tolerance parameter ϵ depends on the maximum scale L. For $L = 9$ and 10 we choose $\epsilon = 10^{-3}$, for $L = 11, \epsilon = 7.5 \times 10^{-4}$, and for $L = 12$ and 13, $\epsilon = 5 \times 10^{-4}$.

The results for pressure, density and velocity at $t = 0.5$ are shown for the three methods in Fig. 7, with $L = 12$, together with the adaptive MR and MR/LTS grids. We observe that all three solutions almost coincide. The adaptive MR and MR/LTS grids are locally refined in the shock region and are identical. In regions where the solution is smoother, the grids differ slightly, due to the time advancement at larger scales with larger time steps. Computing the energy $\int |u(x, 0.5)|^2 \, dx$ at the final time, the results obtained with the FV scheme are 0.18174 and 0.18170, for $L = 12$ and 13, respectively. Using MR and MR/LTS, the energy difference compared with FV scheme is, for both methods, less than 0.004% (see Table 3). Concering the computational efficiency, Table 3 shows that both adaptive MR and MR/LTS methods strongly reduce the memory requirements of the FV scheme. For instance, for $L = 13$, less than 8% of the memory is used, the MR computation

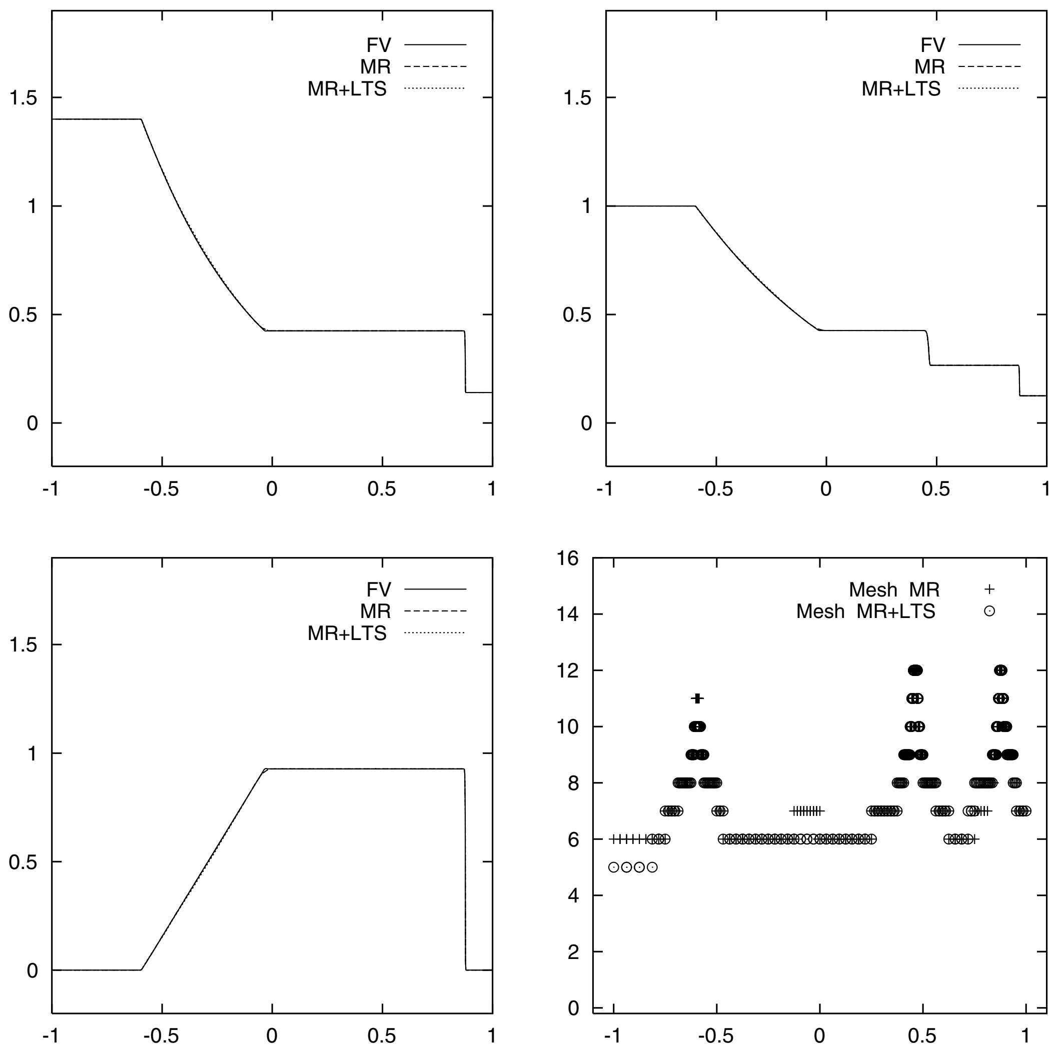

Fig. 7. Shock-tube problem: MR and MR/LTS solutions for 1D Euler equation at $t = 0.5$ with $L = 12$: pressure (top left), density (top right), velocity (bottom left), and mesh visualizations (bottom right).

Table 3
Shock-tube problem: CPU, memory compressions and energy error

Method	L	% CPU time	% Memory	% Error on energy
FV	12	100	100	0
MR		11.3	14.5	4×10^{-3}
MR/LTS		7.9	14.5	3×10^{-3}
FV	13	100	100	0
MR		6.1	7.8	1×10^{-3}
MR/LTS		3.8	7.6	4×10^{-3}

requires about 6% of the CPU time of the FV computation. The MR/LTS computation yields an additional speed-up of 38%. Note that MR/LTS is more than 1.6 faster than the MR scheme (Fig. 8), withthout significantly increasing the error on the solution (see Table 4).

6.3. Reaction–diffusion equations

Next simulations are performed for reaction–diffusion equations, which are prototypes of nonlinear parabolic equations and where the nonlinearity is in the source term.

6.3.1. Steady planar flame front

We consider a steady planar flame front, with equal heat and mass diffusions, modeled by the equation written in its 1D dimensionless form,

$$\frac{\partial T}{\partial t} + v_{\mathrm{f}} \frac{\partial T}{\partial x} = \frac{\partial^2 T}{\partial x^2} + \omega(T), \tag{17}$$

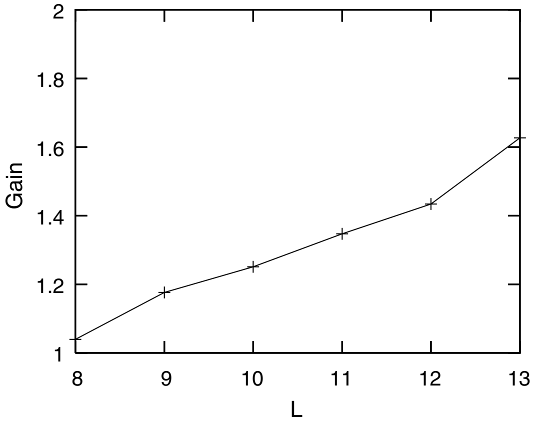

Fig. 8. Shock-tube problem: gain in CPU time of the MR/LTS method in comparison with the MR method, for different mesh sizes at $t = 0.5$ with CFL $\sigma = 0.5$.

Table 4
Steady planar flame front: estimated CPU and memory compressions for the different methods, using CFL $\sigma = 0.5$, with $\epsilon = 10^{-2}$ for 12 scales and 2×10^{-3} for 13 scales

Method	L	% CPU Time	% Memory	% Error on v_f
MR	12	14.1	9.9	0.089
MR/LTS	12	11.0	9.9	0.089
MR	13	10.1	7.0	0.022
MR/LTS	13	7.3	7.0	0.022

where the function $T = T(x,t)$ is the dimensionless temperature, normalized between 0 (fresh premixed gas) and 1 (burnt gas). The chemical reaction rate $\omega(T)$ is given by the formula

$$\omega(T) = \frac{Ze^2}{2}(1-T)\exp\frac{Ze(1-T)}{\tau(1-T)-1}, \tag{18}$$

where τ is the burnt–unburnt temperature ratio, $v_\mathrm{f} = \int_\Omega \omega\,\mathrm{d}x$ is the flame velocity, and Ze is the dimensionless activation energy (Zeldovich number).

In the simulations, we take the initial condition $T(x,0) = 1$, for $x \leqslant 1$, and $T(x,0) = \exp(1-x)$, for $x > 1$. The computational domain is $x \in [-40, 40]$, $t \in [0, 15]$, and the boundary conditions $\frac{\partial T}{\partial x}(-40,t) = 0$, and $T(40,t) = 0$ are enforced. The physical parameters are $\tau = 0.8, Ze = 10$, and we choose a CFL $\sigma = 0.5$. For accuracy reasons, the tolerance parameter ϵ is chosen depending on the maximum scale L. For $L = 8$ and 9, we take $\epsilon = 7.5 \times 10^{-2}$, for $L = 10, \epsilon = 5 \times 10^{-2}$, for $L = 11$ and 12, $\epsilon = 5 \times 10^{-2}$ and for $L = 13$, $\epsilon = 2 \times 10^{-3}$.

Fig. 9 shows a steady planar flame front. Between the burnt and the unburnt gas, the chemical reaction takes place. The adaptive grid is strongly refined in this reaction zone and, up to 13 levels, it is active in both MR and MR/LTS computations, whereas, in quiescent regions, the grid is coarsened down to 5 levels. Com-

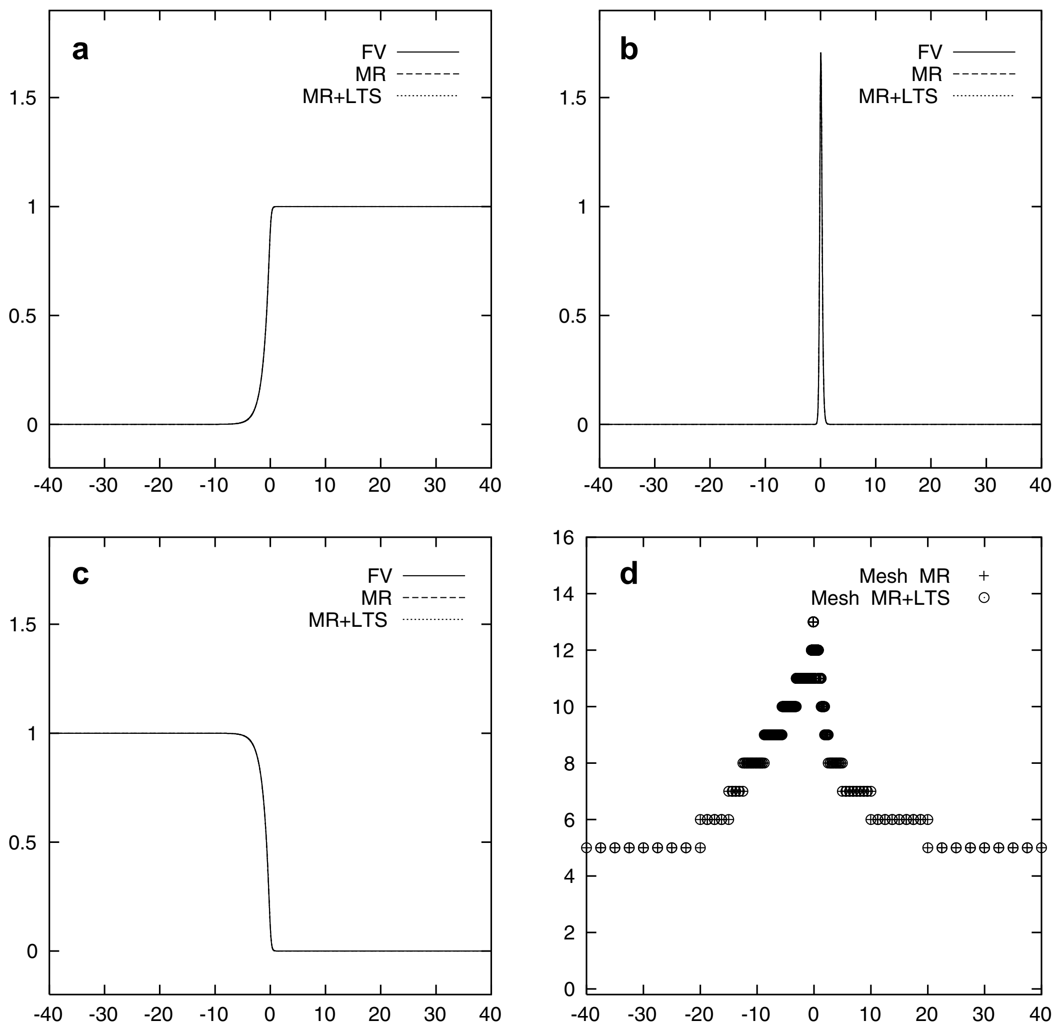

Fig. 9. Steady planar flame front: FV, MR and MR/LTS solutions for (a) temperature, (b) reaction rate, (c) concentration at $t = 15$ for the reaction–diffusion equation, using $\tau = 0.8$, $Le = 1$, $L = 13$, $\epsilon = 2 \times 10^{-3}$ and $\sigma = 0.5$. Comparison between MR and MR/LTS adaptive meshes (d).

paring the value of the flame velocity with the asymptotic value given in Peters and Warnatz [32], we observe that all methods yield values close to the asymptotic one of $v_f = 0.918$ within an error tolerance below 0.01%. For $L = 13$ levels, the adaptive MR and MR/LTS computations only require 7% of the memory compared to a FV method on a regular grid. The MR computation is 10 times faster than the FV computation and the MR/LTS scheme yields an additional speed-up of 28%, i.e., a gain of 1.38 (see Fig. 10).

6.3.2. Three-dimensional flame ball

In the following, we perform numerical simulations of three-dimensional instabilities of spherical flames initially stretched in one space direction, to check the formation of cellular patterns. A flame ball is a stationary or slowly propagating spherical flame structure in a premixed gaseous mixture. Such flames have been experimentally observed for low Lewis numbers under micro-gravity conditions [34].

The thermo-diffusive approximation is well adapted for the computation of flame balls, because the flame velocity is very small [9,21]. Considering again the constant density approximation and one-step chemical kinetics, the system of equations modeling such a flame structure is the set of two reaction–diffusion equations for the temperature T and the partial mass of the unburnt gas Y

$$\frac{\partial T}{\partial t} = \nabla^2 T + \omega - s, \quad \frac{\partial Y}{\partial t} = \frac{1}{Le}\nabla^2 Y - \omega,$$

where ω is the reaction rate

$$\omega = \omega(T, Y) = \frac{Ze^2}{2Le} Y \exp\left[\frac{Ze(T-1)}{1+\tau(T-1)}\right].$$

According to the Stefan–Boltzmann law, the heat-loss s due to radiation writes

$$s = s(T) = \kappa[(T + \tau^{-1} - 1)^4 - (\tau^{-1} - 1)^4],$$

where κ is the dimensionless radiation coefficient. The initial conditions are

$$T(r,0) = \begin{cases} 1 & \text{if } r \leqslant r_0 \\ \exp(1 - \frac{r}{r_0}) & \text{if } r > r_0, \end{cases} \quad Y(r,0) = \begin{cases} 0 & \text{if } r \leqslant r_0 \\ 1 - \exp\left(Le(1 - \frac{r}{r_0})\right) & \text{if } r > r_0, \end{cases}$$

where r_0 denotes the initial radius of the flame ball. The boundaries are sufficiently far from the flame ball, so that they have negligible influence. Hence we can use Neumann boundary conditions. This spherical initial condition is stretched in one direction and a rotation in two axes is applied. Hence, we have $r = \sqrt{\frac{X^2}{a^2} + \frac{Y^2}{b^2} + \frac{Z^2}{c^2}}$, where $X = x\cos\theta - y\sin\theta, Y = (x\sin\theta + y\cos\theta)\cos\varphi - z\sin\varphi$, and $Z = (x\sin\theta + y\cos\theta)\sin\varphi + z\cos\varphi$. The parameters of the ellipsoid are $a = b = 1.5$, $c = 3$, and the rotation angles are

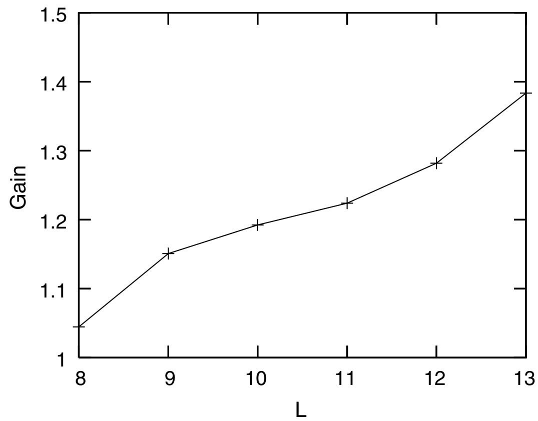

Fig. 10. Steady planar flame front: MR and MR/LTS: gain in CPU time for different mesh sizes at $t = 15$ with CFL $\sigma = 0.5$.

$\theta = \frac{\pi}{3}$ and $\varphi = \frac{\pi}{4}$ with an initial radius $r_0 = 1$. The computational domain is $\Omega = [-20, 20]^3$. For the fresh mixture, we chose a lean 6.5% H_2–air mixture, for which the Lewis number is $Le = 0.3$, $Ze = 10$ and the temperature ratio is $\tau = 0.64$. The radiative heat-loss can be increased by adding products like CF_3Br which do not modify the main chemical reaction, but increase the radiative heat-loss due to soot formation [9]. Hence, we choose a large radiation coefficient, i.e., $\kappa = 0.1$. The dimensionless elapsed time is $t = 27$.

We observe that, after the first splitting, the new cells also split until the domain is full of small balls. This can be seen in Fig. 11, which shows the splitting of the ellipsoid into two cells, a ring of splitting cells, and a moment where there are cells in different stages of splitting in the domain. In Fig. 12 we can see the projection of the cell positions used on the adaptive mesh for temperature and concentration in XZ, YZ and XY planes.

The time evolution of the global reaction rate $R = \int_\Omega \omega \, dx \, dy \, dz$, shown in Fig. 13, is very similar in both MR and MR/LTS simulations. We observe a growth until $t = 22$, when the cells touch the boundaries, and then it starts to decrease.

Table 5 shows the performances for MR and MR/LTS computations. We use a time step of 3.66×10^{-4}, which corresponds to an initial CFL $\sigma = 0.1$. To give some absolute numbers about the CPU time using an Intel(R) Xeon(TM) CPU 3.20 GHz processor, the MR computation required around 2 days and 11 h, and the MR/LTS around 2 days and 4 h to reach the physical time $t = 10$. The FV computation was performed for few iterations only to estimate the CPU time of around 89 days and 6 h that such a computation would require to reach $t = 10$.

The CPU and memory compressions of the different methods, $\epsilon = 0.05$ with 8 scales and initial CFL $\sigma = 0.1$ show that, at $t = 10$, the MR method requires less than 3% of the CPU time needed for the FV computation.

The MR/LTS yields an additional speed-up of the MR method of 15% and hence implies a total speed-up of the FV scheme by a factor 43. Both adaptive methods required only 1% of the memory which the FV computation would consume. The difference in the global reaction rate computed at $t = 10$ (when the second splitting of the flame balls starts) after 27307 iterations which gives $R = 669.1$ is less than 0.003%. Hence, using a local time-stepping procedure, we get an additional speed-up without significant loss in accuracy.

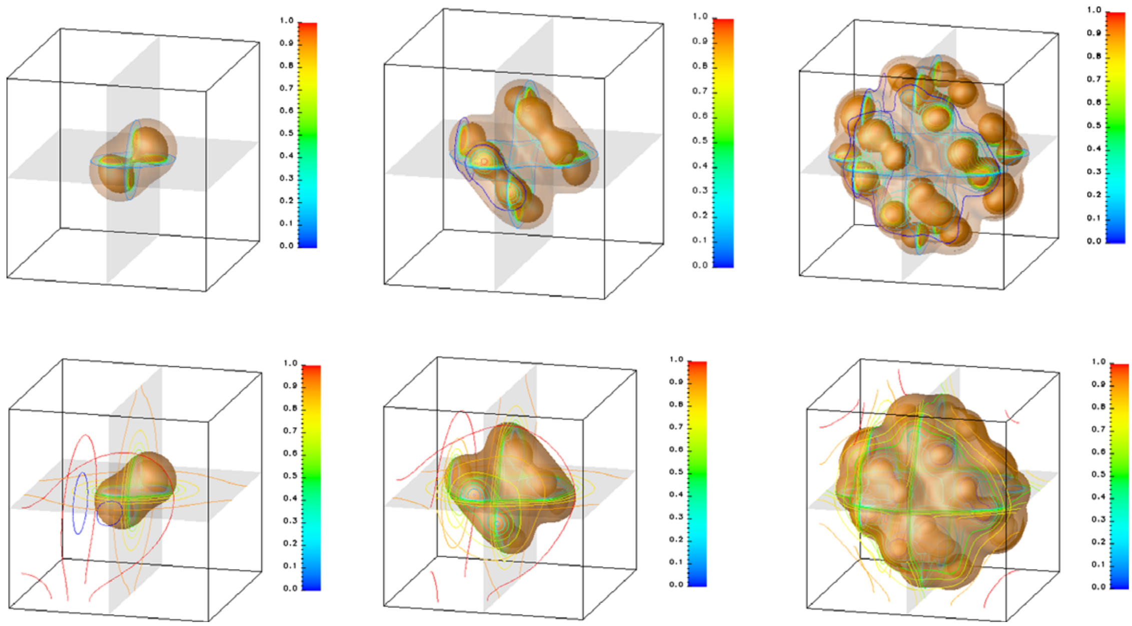

Fig. 11. Splitting flame ball using the MR/LTS method: iso-surfaces and isolines on the cut-plane for the temperature (top) and the concentration of the limiting reactant (bottom) with $L = 8$ scales, $Le = 0.3$, $Ze = 10$, $\tau = 0.64$, $\kappa = 0.1$. Left to right: physical times $t = 6.0$ (left), $t = 13.5$ (center), and $t = 21.0$ (right).

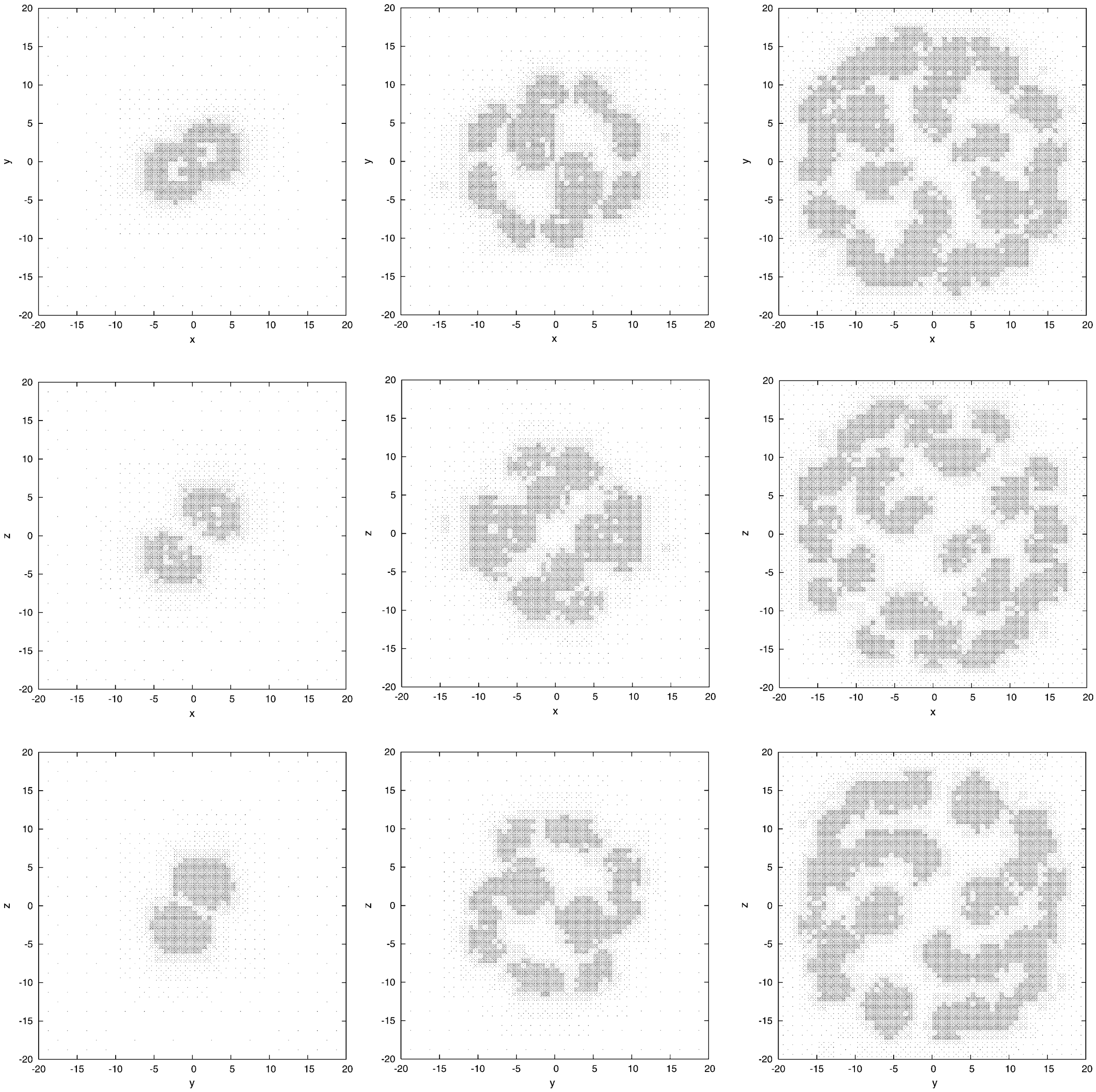

Fig. 12. Splitting flame ball: projections of the cell-centers used on the adaptive mesh in the XZ (top), YZ (middle) and XY-planes (bottom) at times $t = 6.0$ (left), $t = 13.5$ (center), and $t = 21.0$ (right).

7. Conclusion

The present paper describes an efficient space-adaptive multiresolution method with local time stepping to solve evolutionary PDEs in Cartesian geometry. It is based on a finite volume discretization with explicit time integration, both of second-order. We introduced a new local scale-dependent time-stepping method into the adaptive multiresolution scheme developed in [35,36]. In comparison to the finite volume scheme on a regular grid this new scheme allows further speed-up due to an improved time advancement using larger time steps on large scales without violating the stability condition of the explicit scheme. The number of costly flux evaluations is likewise reduced, together with the memory requirements, thanks to the dynamic tree data structure. However, as different scales evolve with different time steps, a synchronization of the tree data structure becomes necessary. The synchronization limits currently the time scheme to second-order Runge–Kutta methods as for higher order schemes this task becomes much more difficult. We have also shown that a suit-

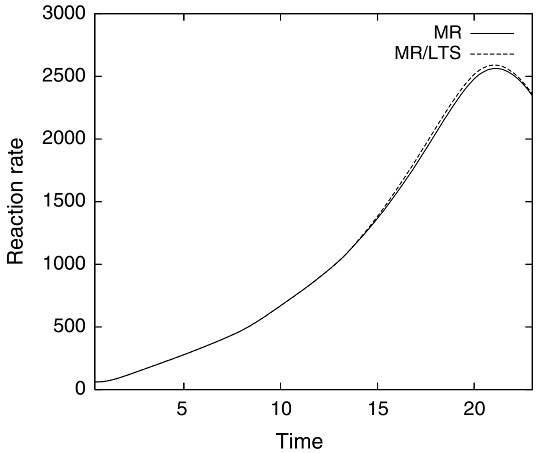

Fig. 13. Splitting flame ball: time evolution of the reaction rates for MR and MR/LTS for the 3D flame ball.

Table 5
Splitting flame ball: CPU and memory compressions for the different methods, $\epsilon = 0.05$ with 8 scales and CFL $\sigma = 0.1$

Method	% CPU Time	% Memory	R
MR	2.7	1.05	669.09
MR/LTS	2.3	1.05	669.11

The global reaction rate is R computed at $t = 10$, after 27,307 iterations.

able thresholding of the wavelet coefficients maintains the second-order accuracy of the finite volume scheme on the regular grid. The local time stepping hence represents a moderate, but significant, speed-up with unsignificant loss of accuracy.

A matrix stability analysis for a simplified two grid problem applied to a convection–diffusion equation showed that the spectral stability region of the local time-stepping scheme is approximately the same as the one corresponding to the coarse grid for a mesh Reynolds number $Re \leqslant 2$, but for higher values of this parameter there is a reduction in this stability region. We verified the theoretical prediction numerically for the adaptive scheme and showed that the stability region obtained numerically corresponds to the theoretical one. We demonstrated the efficiency of the new method for different test problems in one and three space dimensions and studied its performance by comparing the CPU time and the memory requirements with the finite volume method using uniform discretization. Fully adaptive three-dimensional computations of spherical flame instabilities reveal the applicability to practically relevant problems. The current speed-up increases with the number of required scales to represent the solution, and also depends on the computational cost of the flux evaluations. Hereby the structure of the graded tree plays a crucial role which reflects the local regularity of the solution. A better localization of small scale features of the solution leads to an unbalanced tree for which MR/LTS is most beneficial. In conclusion, the more expensive the flux evaluation and the larger the number of well-localized active scales, the better the speed-up of MR/LTS. As predicted by an analytical cost estimate, we found that the actual speed-up of local time stepping depends on the distribution of the active cells. If the majority of the cells is active on fine scales, the MR/LTS scheme is less efficient with respect to the MR scheme, whereas for few active cells on fine scales (e.g., point singularities) the speed-up becomes larger. In these cases, performing one large time step at a coarse level instead of several time steps on fine scale cells becomes more efficient.

As perspectives, we plan to extend this space-time adaptive scheme to solve the three-dimensional compressible Navier–Stokes equations to perform Coherent Vortex Simulations (CVS) [19] of turbulent flows in the weakly compressible regime. Future work will also deal with the extension to higher order space discretizations using large stencils (e.g., ENO, WENO), which will imply a modification of the tree structure. Higher order multi-stage methods for the MR/LTS scheme are not straightforward and will require further investigations.

Acknowledgments

M.O.D. thankfully acknowledges financial support from CNRS, department ST2I, the ANR project "M2TFP", Ecole Centrale de Marseille and FAPESP. S.G. thankfully acknowledges financial support from Ecole Centrale de Marseille and from CNPq – the Brazilian Research Council. O.R. and K.S. acknowledge financial support from the French–German DFG-CNRS Research Programme "LES and CVS of Complex Flows". K.S. thanks the ANR project "M2TFP" for financial support. We are also grateful to Marie Goretti Dejean for her helpful assistance, and to Igor Molina for having performed some of the computations shown in the numerical results. Finally, we would like to thank Siegfried Müller for fruitful discussions on local time stepping.

Appendix. Algorithms

In the following algorithms, $\mathcal{L}(\Lambda)$ are the leaves of the tree Λ and $\mathcal{V}(\Lambda)$ are the virtual leaves of the tree. The symbol $\tilde{}$ denotes the temporary values.

Algorithm 1 LTS: INPUT

Require: $\overline{U}^0_{\mathcal{L}(\Lambda)} = \{\bar{u}^0_{l,i} | (l,i) \in \mathcal{L}(\Lambda)\}$ {all the leaves at initial time}
Ensure: $\overline{U}^n_{\mathcal{L}(\Lambda)} = \{\bar{u}^n_{l,i} | (l,i) \in \mathcal{L}(\Lambda)\}$ {all the leaves at the time step n}
Require: $\Delta t = \Delta t_L$ {the time step on the finest level is set as reference time step}
Ensure: $\Delta t_l = 2^{L-l}\Delta t, \quad 0 < l \leqslant L$ {time step at the level l}
  LTS:CYCLE

Algorithm 2 LTS: CYCLE

---

for all iterations n **do**
  LTS: COMPUTE VIRTUAL LEAVES at iteration n (Algorithm 3)
  {STAGE 1}
  for all levels from $l = 0$ until L **do**
    for all cells i such that $0 \leqslant i < 2^l$ **do**
      if $(l,i) \in \mathcal{L}(\Lambda)$ **then**
        $p = 2^{L-l}$ {number of steps being advanced at the level l}
        if $((n-1) \bmod p = 0)$ **then**
          $\overline{D}^n_{l,i} = D(\bar{u}^n_{l,i-2}, \bar{u}^n_{l,i-1}, \bar{u}^n_{l,i}, \bar{u}^n_{l,i+1}, \bar{u}^n_{l,i+2})$ {Compute divergence}
          $\tilde{u}^{n+p}_{l,i} = \bar{u}^n_{l,i} + \Delta t_l \overline{D}^n_{l,i}$ {RK 1st stage}
          if $(l \neq L)$ **then**
            $\tilde{u}^{n+p\div 2}_{l,i} = \frac{1}{2}(\tilde{u}^{n+p}_{l,i} + \bar{u}^n_{l,i})$ {Store values at intermediary state}
  LTS: UPDATE VIRTUAL LEAVES at iteration n (Algorithm 4)
  {STAGE 2}
  for all levels from $l = 0$ until L **do**
    for all cells i such that $0 \leqslant i < 2^l$ **do**
      if $(l,i) \in \mathcal{L}(\Lambda)$ **then**
        $p = 2^{L-l}$ {the number of steps being advanced at the level l}
        if $(n \bmod p = 0)$ **then**
          $\widetilde{D}^{n+p}_{l,i} = D(\tilde{u}^{n+p}_{l,i-2}, \cdots, \tilde{u}^{n+p}_{l,i+2})$ {Compute divergence}
          $\bar{u}^{n+p}_{l,i} = \frac{\bar{u}^n_{l,i}}{2} + \frac{\tilde{u}^{n+p}_{l,i}}{2} + \frac{\Delta t_l}{2}\widetilde{D}^{n+p}_{l,i}$ {RK 2nd stage}
  LTS: ADAPT TREE at iteration n (Algorithm 5)

Algorithm 3 LTS: COMPUTE VIRTUAL LEAVES at iteration n

for all levels from $l = 0$ until L **do**
  for all cells i such that $0 \leqslant i < 2^l$ **do**
    if $(l, i) \in \mathcal{V}(\Lambda)$ **then**
      $q = i \div 2$
      {the cell $(l-1, q)$ is the parent of the cell $(l, i), i$ being odd or even}
      $\bar{u}^n_{l,i} = P_{l-1\to l}(\bar{u}^n_{l-1,q-1},\ \bar{u}^n_{l-1,q},\ \bar{u}^n_{l-1,q+1})$
      {Predict value from parent and its neighbors}

Algorithm 4 LTS: UPDATE VIRTUAL LEAVES at iteration n

for all levels from $l = 0$ until L **do**
  for all cells i such that $0 \leqslant i < 2^l$ **do**
    if $(l, i) \in \mathcal{V}(\Lambda)$ **then**
      $q = i \div 2$
      {the cell $(l-1, q)$ is the parent of the cell $(l, i), i$ being odd or even}
      $\tilde{u}^n_{l,i} \leftarrow P_{l-1\to l}(\tilde{u}^n_{l-1,q-1},\ \tilde{u}^n_{l-1,q},\ \tilde{u}^n_{l-1,q+1})$
      {Predict value from parent and its neighbors}

Algorithm 5 LTS: ADAPT TREE at iteration n

Require n
  for all levels from $l = 0$ until L **do**
    for all cells i such that $0 \leqslant i < 2^l$ **do**
      if $(l, i) \in \mathcal{L}(\Lambda)$ **then**
        $p = 2^{L-l}$ {number of steps being advanced at the level l}
        if $(n \bmod p) = 0$ **then**
          {Delete children if possible}
          if $(|d^n_{l+1,2i}| < \epsilon_{l+1})$ **and** $(|d^n_{l+1,2i+1}| < \epsilon_{l+1})$ **and** $(|d^n_{l,i}| < \epsilon_l)$ **then**
            Deallocate children: $(\bar{u}^n_{l+1,2i}), (\bar{u}^n_{l+1,2i+1})$
        else
          if $(n \bmod p \div 2 = 0)$ **then**
            {Add children when necessary}
            if $(|d^n_{l,i}| > \epsilon_l)$ **then**
              Allocate children: $(\bar{u}^n_{l+1,2i}), (\bar{u}^n_{l+1,2i+1})$
              {Predict value from parent and its neighbors}
              $\bar{u}^n_{l+1,2i} = P_{l\to l+1}(2i; \bar{u}^n_{l,i-1}, \bar{u}^n_{l,i}, \bar{u}^n_{l,i+1})$
              $\bar{u}^n_{l+1,2i+1} = P_{l\to l+1}(2i+1; \bar{u}^n_{l,i-1}, \bar{u}^n_{l,i}, \bar{u}^n_{l,i+1})$

References

[1] J. Alam, N.-K.-R. Kevlahan, O. Vasilyev, Simultaneous space-time adaptive wavelet solution of nonlinear partial differential equations, J. Comput. Phys. 214 (2006) 829–857.

[2] E. Bacry, S. Mallat, G. Papanicolaou, A wavelet based space-time adaptive numerical-method for partial-differential equations, Rairo-Math. Modell. Numer. Anal. 26 (1992) 793–834.

[3] R. Becker, R. Rannacher, An optimal control approach to error control and mesh adaptation, Acta Numer. 10 (2001) 1–102.

[4] J. Bell, M. Berger, J. Saltzmann, M. Welcome, Three-dimensional adaptive mesh refinement for hyperbolic conservation laws, SIAM J. Sci. Comput. 15 (1994) 127.

[5] M. Berger, R. LeVeque, Adaptive mesh refinement using wave-propagation algorithms for hyperbolic systems, SIAM J. Numer. Anal. 35 (1998) 2298–2316.
[6] M.J. Berger, P. Collela, Local adaptive mesh refinement for shock hydrodynamics, J. Comput. Phys. 82 (1) (1989) 64–84.
[7] M.J. Berger, J. Olinger, Adaptive mesh refinement for hyperbolic partial differential equations, J. Comp. Phys. 53 (1984) 484–512.
[8] B.L. Bihari, Multiresolution schemes for conservation laws with viscosity, J. Comput. Phys. 123 (1996) 207–225.
[9] H. Bockhorn, J. Fröhlich, K. Schneider, An adaptive two-dimensional wavelet-vaguelette algorithm for the computation of flame balls, Combust. Theory Model. 3 (1999) 1–22.
[10] J.E. Castilho, S.M. Gomes, A multilevel wavelet scheme for evolution equations with time adaptivity, Technical Report RP 09/02, Unicamp, Campinas, April 2002.
[11] G. Chiavassa, R. Donat, Point value multi-scale algorithms for 2d compressible flow, SIAM J. Sci. Comput. 23 (3) (2001) 805–823.
[12] A. Cohen, Wavelet methods in numerical analysis, in: P.G. Ciarlet, J.L. Lions (Eds.), Handbook of Numerical Analysis, vol. VII, Elsevier, Amsterdam, 2000.
[13] A. Cohen, Adaptive methods for PDE's – wavelet or mesh refinements? in: International Conference of Mathematics, Beijing, 2002.
[14] A. Cohen, S.M. Kaber, S. Müller, M. Postel, Fully adaptive multiresolution finite volume schemes for conservation laws, Math. Comp. 72 (2003) 183–225.
[15] F. Collino, T. Fouquet, P. Joly, A conservative space-time mesh refinement method for the 1-d wave equation. I. Construction, Numer. Math. 95 (2) (2003) 197–221.
[16] C. Dawson, R. Kirby, High resolution schemes for conservation laws with locally varying time steps, SIAM J. Sci. Comput. 22 (6) (2001) 2256–2281.
[17] M. Dumbser, M. Käser, E. Toro, An arbitrary high order discontinuous Galerkin method for elastic waves on unstructured meshes V: local time stepping and p-adaptivity, Geophys. J. Int. 171 (2) (2007) 695–717.
[18] C.-D.M.F. Lörcher, G. Gassner, A discontinuous galerkin scheme based on a spacetime expansion I. Inviscid compressible flow in one space dimension, J. Sci. Comput. 32 (2) (2007).
[19] M. Farge, K. Schneider, Coherent vortex simulation (CVS), a semi-deterministic turbulence model using wavelets, Flow Turbul. Combust. 66 (4) (2001) 393–426.
[20] L. Ferm, P. Löstedt, Space-time adaptive solutions of first order pdes, J. Sci. Comput. 26 (1) (2006) 83–110.
[21] W. Gerlinger, K. Schneider, J. Fröhlich, H. Bockhorn, Numerical simulations on the stability of spherical flame structures, Combust. Flame 132 (1–2) (2003) 247–271.
[22] A. Harten, Multiresolution algorithms for the numerical solution of hyperbolic conservation laws, Comm. Pure Appl. Math. 48 (1995) 1305–1342.
[23] A. Harten, Multiresolution representation of data: a general framework, SIAM J. Numer. Anal. 33 (3) (1996) 385–394.
[24] C. Hirsch, Numerical Computation of Internal and External Flows, vol. 2, John Wiley and Sons, 1990.
[25] J.E.J.E. Flaherty, R.M. Loy, M.S. Shephard, B.K. Szymanski, J.D. Teresco, L.H. Ziantz, Adaptive local refinement with octree load balancing for the parallel solution of three-dimensional conservation laws, J. Parall. Dist. Comp. 47 (2) (1997) 139–152.
[26] L. Jameson, AMR vs high order schemes, J. Sci. Comput. 18 (2003) 1–24.
[27] M. Kaibara, S.M. Gomes, A fully adaptive multiresolution scheme for shock computations, in: E.F. Toro (Ed.), Godunov Methods: Theory and Applications, Klumer Academic/Plenum Publishers, 2000.
[28] M.-S. Liou, A sequel to AUSM: AUSM+, J. Comput. Phys. 129 (1996) 364–382.
[29] S. Müller, Adaptive Multiscale Schemes for Conservation Laws, Lectures Notes in Computational Science and Engineering, vol. 27, Springer, Heidelberg, 2003.
[30] S. Müller, Y. Stiriba, Fully adaptive multiscale schemes for conservation laws employing locally varying time stepping, J. Sci. Comput. 30 (3) (2007) 493–531.
[31] S. Osher, R. Sanders, Numerical approximations to nonlinear conservation laws with locally varying time space grid, Math. Comp. 43 (1983) 321–336.
[32] N. Peters, J. Warnatz (Eds.), Numerical Methods in Laminar Flame Propagation, Notes on Numerical Fluid Mechanics, vol. 6, Vieweg, 1982.
[33] J.J. Quirk, An adaptive grid algorithm for computational shock hydrodynamics, PhD Thesis, Cranfield Institute of Technology, 1991.
[34] P.D. Ronney, Near-limit flame structures at low Lewis number, Combust. Flame 82 (1990) 1–14.
[35] O. Roussel, K. Schneider, An adaptive multiresolution method for combustion problems: application to flame ball–vortex interaction, Comp. Fluids 34 (7) (2005) 817–831.
[36] O. Roussel, K. Schneider, A. Tsigulin, H. Bockhorn, A conservative fully adaptive multiresolution algorithm for parabolic pdes, J. Comput. Phys. 188 (2003) 493–523.
[37] G.A. Sod, A survey of several finite difference methods for systems of nonlinear hyperbolic conservation laws, J. Comput. Phys. 27 (1978) 1–31.
[38] E. Süli, D.F. Mayers, An Introduction to Numerical Analysis, Cambridge University Press, 2003.
[39] H.Z. Tang, G. Warnecke, A class of high resolution schemes for hyperbolic conservation laws and convection–diffusion equations with varying time and space grids, SIAM J. Sci. Comput. 26 (4) (2005) 1415–1431.