INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN ENGINEERING
Int. J. Numer. Meth. Engng 2009; **78**:652–670
Published online 27 November 2008 in Wiley InterScience (www.interscience.wiley.com). DOI: 10.1002/nme.2501

An adaptive multiresolution method for parabolic PDEs with time-step control

M. O. Domingues[1, 2, *, †], O. Roussel[3] and K. Schneider[2, 4]

[1]*Laboratório Associado de Computação e Matemática Aplicada (LAC), Coordenadoria dos Laboratórios Associados (CTE), Instituto Nacional de Pesquisas Espaciais (INPE), Av. dos Astronautas, 1758, 12227-010 São José dos Campos, Brazil*
[2]*Laboratoire de Modélisation en Mécanique et Procédés Propres (M2P2), CNRS and Universités d'Aix-Marseille, 38, rue Joliot-Curie, 13451 Marseille Cedex 20, France*
[3]*Institut für Technische Chemie und Polymerchemie (TCP), Universität Karlsruhe, Kaiserstr. 12, 76128 Karlsruhe, Germany*
[4]*Centre de Mathématiques et d'Informatique (CMI), Université de Provence, 39, rue Joliot-Curie, 13453 Marseille Cedex 13, France*

SUMMARY

We present an efficient adaptive numerical scheme for parabolic partial differential equations based on a finite volume (FV) discretization with explicit time discretization using embedded Runge–Kutta (RK) schemes. A multiresolution strategy allows local grid refinement while controlling the approximation error in space. The costly fluxes are evaluated on the adaptive grid only. Compact RK methods of second and third order are then used to choose automatically the new time step while controlling the approximation error in time. Non-admissible choices of the time step are avoided by limiting its variation. The implementation of the multiresolution representation uses a dynamic tree data structure, which allows memory compression and CPU time reduction. This new numerical scheme is validated using different classical test problems in one, two and three space dimensions. The gain in memory and CPU time with respect to the FV scheme on a regular grid is reported, which demonstrates the efficiency of the new method. Copyright © 2008 John Wiley & Sons, Ltd.

Received 4 September 2007; Revised 14 July 2008; Accepted 8 October 2008

KEY WORDS: adaptivity; multiresolution; finite volume; Runge–Kutta; partial differential equation; time-step control

*Correspondence to: M. O. Domingues, Laboratório Associado de Computação e Matemática Aplicada (LAC), Coordenadoria dos Laboratórios Associados (CTE), Instituto Nacional de Pesquisas Espaciais (INPE), Av. dos Astronautas, 1758, 12227-010 São José dos Campos, Brazil.
†E-mail: margarete.olivera.domingues@gmail.com, mo.domingues@lac.inpe.br

Contract/grant sponsor: ANR Project 'M2TFP'
Contract/grant sponsor: DFG-CNRS Research Program 'LES in complex flows'
Contract/grant sponsor: CNRS Department ST21
Contract/grant sponsor: Ecole Centrale Marseille
Contract/grant sponsor: CNPq

Copyright © 2008 John Wiley & Sons, Ltd.

1. INTRODUCTION

The mathematical modelling of many chemical–physical problems encountered in industrial or environmental applications often leads to a system of non-linear partial differential equations (PDEs). In many cases, their solutions exhibit a multitude of spatial and temporal scales, which are, however, not uniformly distributed in the space–time domain, e.g. in turbulence or in combustion processes. Efficient numerical discretizations could then take advantage of this property by introducing some kind of adaptivity in space and time, in order to reduce the computational complexity of direct approaches, which use uniform discretizations. A key question of many adaptive schemes is how to estimate and control the error of the solution with respect to the solution computed on the finest regular grid where the quality of the approximation can be estimated in many cases.

There have been different efforts to define adaptive space discretizations, some emerge from *ad hoc* criteria, others are based on more elaborated *a posteriori* error estimators using control strategies by solving adjoint problems [1, 2]. We mention the adaptive mesh refinement methods introduced by Berger and Oliger [3], multiresolution (MR)-based schemes, which have been first developed by Harten for conservation laws [4, 5]. Harten's approach was extended and further developed by Cohen *et al.* [6], Chiavassa and Donat [7], Roussel *et al.* [8] and Roussel and Schneider [9]. Starting with a classical discretization, either finite volumes (FV) or finite differences, the main idea of MR methods is to balance the truncation error of the underlying scheme on a finest regular grid and the threshold error introduced when discarding small detail coefficients of the MR decomposition of the solution. An overview of the different methods can be found, e.g. in [10].

The sparse point representation (SPR) method was introduced by Holmström [11]. It combines the simplicity and accuracy of traditional finite difference schemes with the ability of wavelet coefficients to characterize the local regularity of functions. The idea is to represent functions by their point-values corresponding to significant wavelet coefficients. This is similar to Harten's approach using an MR based on point-values and locally uniform finite differences. Other adaptive wavelet methods were proposed with many similarities to the SPR method, e.g. the filter bank method [12] and the second generation wavelet collocation method [13] may be considered as generalizations of the SPR method.

A different but related philosophy was proposed by Domingues *et al.* [14] using adaptive block representations (ABRs). The idea is to decompose the mesh into blocks. Each block has a uniform grid and the same number of grid points. However, the refinement level may differ between the blocks. Nevertheless, the blocks are organized into a tree structure, e.g. in one space dimension we have a binary tree, in two dimensions a quad tree and in three dimensions an oct tree. The root of the tree corresponds to the whole domain on the coarsest level. On the first level, the nodes correspond to 2^d blocks (where d denotes the space dimension), the grid being refined by a factor 2 in each direction. Each sub-block may then be decomposed into 2^d sub-blocks with the grid again being refined by a factor 2 in each direction. This process can be iterated down to the finest level. The refinement criterion is based on the wavelet coefficients of a one-level decomposition of each sub-block. The computations are then performed on the leaves of the tree using finite differences on the locally refined grid with some extra virtual points, to be able to evaluate the finite difference stencils near the boundary. This method could be efficiently implemented on parallel computers.

The main drawback of adaptive MR methods with explicit time discretization is that, for highly non-linear problems, the time step required to ensure the numerical stability cannot be determined analytically. For such problems, computations are usually performed using a much smaller time

Copyright © 2008 John Wiley & Sons, Ltd.

DOI: 10.1002/nme

step than the one required, which significantly increases the cost in terms of CPU time. Recently, Ferm and Löstedt [15] proposed a time-step control for adaptive mesh refinement methods coupled with a Runge–Kutta–Fehlberg (RKF) scheme to choose automatically the size of the time step. Results have been presented for first-order hyperbolic conservation laws (Burgers, Euler and the wave equation) in one space dimension.

The aim of the present paper is to develop a time-stepping scheme with automatic error control for the space-adaptive MR scheme presented in Roussel *et al.* [8] and Roussel and Schneider [9]. The time integration uses embedded RK methods of second and third order, which allow an estimation of the local error in time. An original technique is also proposed to avoid non-admissible choices of the time step, which occur in classical embedded RK methods. The adaptivity is obtained by a MR decomposition, which automatically detects the local regularity of the solution and hence guarantees dynamic grid adaption to track the solution in space and scale. The costly numerical fluxes are evaluated on this locally refined grid, while ensuring strict conservation. The implementation uses graded tree data structures, which allow an efficient representation of the solution on adaptive grids with reduced memory requirements. Numerical experiments in one, two and three dimensions validate this method and demonstrate its efficiency.

The paper is organized as follows: Section 2 summarizes the space discretization using finite volumes for both regular and adaptive meshes. The time discretization uses compact embedded RK methods for time-step control. Section 3 presents applications of the adaptive methods and comparisons with the results obtained using a FV discretization on a regular grid and discusses the accuracy, CPU time compression and efficiency. We show results for the viscous Burgers equation in one space dimension. For reaction–diffusion equations, we present results for a cellular flame instability in two and three space dimensions. Finally, conclusions are drawn and perspectives for future investigations are illuminated.

2. SPACE AND TIME DISCRETIZATION

2.1. Adaptive MR methods using FVs

We consider the initial value problem for parabolic conservation laws on $(\mathbf{x}, t) \in \Omega \times [0, +\infty)$, $\Omega \subset \mathbb{R}^d$, of the form

$$\frac{\partial u}{\partial t} = \mathscr{D}(u, \nabla u), \quad u(\mathbf{x}, 0) = u_0(\mathbf{x}) \tag{1}$$

in $d = 1, 2, 3$ space dimensions, with appropriate boundary conditions, where $\mathscr{D}(u, \nabla u) = -\nabla \cdot F(u, \nabla u) + S(u)$ denotes the divergence and source term operator. The flux F is decomposed into advective and diffusive terms, i.e.

$$F(u, \nabla u) = F_{\text{adv}}(u) + F_{\text{dif}}(\nabla u) = f(u) - \nu \nabla u \tag{2}$$

where ν denotes the diffusivity, which is here assumed to be constant and positive. To discretize (1), we use a classical FV formulation in the standard conservative form.

The domain Ω corresponds to a parallelepiped in d dimensions. It is partitioned into cells $(\Omega_i)_{i \in \Lambda}$, $\Lambda = \{1, \ldots, i_{\max}\}$. We then denote by $\bar{u}_i(t)$ the cell-average of a given quantity u on Ω_i

Copyright © 2008 John Wiley & Sons, Ltd.

DOI: 10.1002/nme

at the instant t

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

where $|\Omega_i| = \int_{\Omega_i} \mathrm{d}\mathbf{x}$ is the volume of the cell. Integrating (1) on Ω_i yields

$$\frac{\mathrm{d}\bar{u}_i}{\mathrm{d}t}(t) = \bar{\mathscr{D}}_i(t) \tag{3}$$

where

$$\bar{\mathscr{D}}_i = -\frac{1}{|\Omega_i|} \int_{\partial\Omega_i} F(u, \nabla u) \cdot \eta_i \, \mathrm{d}\mathbf{x} + \bar{S}_i$$

In the following, we describe the space discretization and the time integration applied to (3).

2.1.1. Numerical flux. In the 1D case, Ω_i is an interval $[x_{i-1/2}, x_{i+1/2}]$ of length $\Delta x_i = x_{i+1/2} - x_{i-1/2}$. Equation (3) becomes

$$\bar{\mathscr{D}}_i = -\frac{1}{\Delta x_i}(\bar{F}_{i+1/2} - \bar{F}_{i-1/2}) + \bar{S}_i \tag{4}$$

where $\bar{F}$ is the numerical flux. Advective and diffusive terms are approximated differently. For the advective part, we use Roe's scheme [16] with a second-order ENO interpolation (see e.g. [4]), whereas, for the diffusive part, we choose a second-order accurate centered scheme. The source term is approximated by $\bar{S}_i \approx S(\bar{u}_i)$, which yields second-order accuracy in case of a linear source term.

The extension to higher dimensions in Cartesian geometries is performed by tensor products. For the 2D case, $\Omega_{i,j}$ is a rectangle with a volume $|\Omega_{i,j}| = \Delta x_i \, \Delta y_j$. Equation (3) can be written as

$$\frac{\mathrm{d}\bar{u}_{i,j}}{\mathrm{d}t}(t) = \bar{\mathscr{D}}_{i,j}(t) \tag{5}$$

where

$$\bar{\mathscr{D}}_{i,j} = -\frac{1}{\Delta x_i}(\bar{F}_{i+1/2,j} - \bar{F}_{i-1/2,j}) - \frac{1}{\Delta y_j}(\bar{F}_{i,j+1/2} - \bar{F}_{i,j-1/2}) + \bar{S}_{i,j}$$

and for the 3D case $\Omega_{i,j,k}$ is a rectangular parallelepiped with volume $|\Omega_{i,j,k}| = \Delta x_i \, \Delta y_j \Delta z_k$. The same numerical flux as in the 1D case is applied in each direction of the 2D and 3D cases.

2.1.2. MR representation. Our starting point is the cell-average MR representation [4]. The nodes are cell-average values and two operators are defined to navigate through the tree. A complete description of the 1D MR representation is given in this subsection and a brief description explains how to extend it to higher dimensions in Cartesian geometry using a tensor product approach.

We use the scheme proposed and implemented in [8] based on a graded tree structure. In the following, we denote by Λ the set of the indices of the existing nodes, by $\mathscr{L}(\Lambda)$ the restriction of Λ to the leaves and by Λ_l the restriction of Λ to a level l, $0 \leqslant l < L$. For the 1D case, we denote by $\Omega = \Omega_{0,0}$ the root cell, $\Omega_{l,i}$, $0 \leqslant l < L$, $i \in \Lambda_l$ the different node cells, $\bar{u}_{l,i}$ the cell-average value of u

Copyright © 2008 John Wiley & Sons, Ltd.

DOI: 10.1002/nme

for the cell $\Omega_{l,i}$ and $\bar{U}_l = (\bar{u}_{l,i})_{i\in\Lambda_l}$ the set of the existing cell-average values at level l. To estimate the cell-averages of a level l from the ones of the level $l+1$, we use the projection operator $P_{l+1\to l}$

$$P_{l+1\to l} : \bar{U}_{l+1} \mapsto \bar{U}_l \tag{6}$$

This operator is exact and unique, given that the parent cell-average is nothing but the weighted average of the children cell-averages. For a regular grid structure in one dimension, it is simply defined by the mean value

$$\bar{u}_{l,i} = (P_{l+1\to l}\bar{U}_{l+1})_i = \tfrac{1}{2}(\bar{u}_{l+1,2i} + \bar{u}_{l+1,2i+1}) \tag{7}$$

To estimate the cell-averages of a level $l+1$ from the ones of the level l, we use the prediction operator $P_{l\to l+1}$.

$$P_{l\to l+1} : \bar{U}_l \mapsto \tilde{U}_{l+1} \tag{8}$$

This operator gives an approximation of $\bar{U}_l$ at the level $l+1$ by interpolation. It is not unique, nevertheless, in order to be applicable in the dynamic graded tree structure as defined above, this operator must satisfy two properties:

- 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 neighbors of the parent in each direction.
- It has to be consistent with the projection, i.e. $P_{l+1\to l} \circ P_{l\to l+1} = \mathrm{Id}$.

For the regular grid structure in one dimension, we use as prediction operator a polynomial interpolation of the cell-averages, like the one proposed by Harten [4], with accuracy order of the MR method $r = 3$, which corresponds to a polynomial interpolation of degree 2. With this accuracy, only one nearest neighbor of the parent is required in each direction. Hence,

$$\begin{aligned} \tilde{u}_{l+1,2i} &= I(\bar{U}_l; l+1, 2i) = \bar{u}_{l,i} - \tfrac{1}{8}(\bar{u}_{l,i+1} - \bar{u}_{l,i-1}) \\ \tilde{u}_{l+1,2i+1} &= I(\bar{U}_l; l+1, 2i+1) = \bar{u}_{l,i} + \tfrac{1}{8}(\bar{u}_{l,i+1} - \bar{u}_{l,i-1}) \end{aligned} \tag{9}$$

The detail is the difference between the exact and the predicted values. In the 1D case, it is defined as

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

Since the sum of the details for all the brothers of a parent cell are equal to zero by definition, these coefficients are redundant [4]. Given that a parent has 2^d children, only $2^d - 1$ details are independent.

Thus, the knowledge of the cell-average value on the 2^d children is equivalent to the knowledge of the cell-average value of the parent and these $2^d - 1$ independent details. This can be expressed in one dimension by

$$(\bar{u}_{l+1,2i}, \bar{u}_{l+1,2i+1}) \longleftrightarrow (\bar{d}_{l+1,2i}, \bar{u}_{l,i})$$

For more details on this equivalence, we refer to Harten [4]. For a given level l, it can be summarized by

$$\bar{U}_l \longleftrightarrow (\bar{D}_l, \bar{U}_{l-1})$$

where $\bar{D}_l$ denotes the vector of the details on the level l.

Copyright © 2008 John Wiley & Sons, Ltd.

DOI: 10.1002/nme

Repeating this operation recursively on L levels, one gets the so-called MR transform on the cell-average values [4]

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

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. More details on the 2D and 3D scheme and its implementation are presented in [8].

Thresholding of the MR representation leads to a tree structure adapted to the solution. This procedure consists in removing leaves where details are smaller than a prescribed tolerance ε, while preserving the graded tree data structure. This is performed by the threshold operator $\mathbf{T}(\varepsilon)$. After thresholding, one more level is added as security zone, to account for the evolution of the solution in the tree representation at the next time step. The choice for an accurate tolerance is discussed in [8].

2.1.3. Conservative flux computation. When a leaf at level l has no neighbor at the same level in a given direction (left or right), the flux at the interface is computed using the cell-average value of the adjacent virtual leaf, which is computed by projection from its parent at the level $l-1$ [8].

In order to ensure conservation of the scheme in the flux computations, the ingoing flux for this parent cell (at the level $l-1$) is taken as the sum of the fluxes going out of the adjacent leaves at the level l, i.e. in 2D

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

where F denotes the convective and diffusive fluxes defined in (2).

2.2. Time integration

There are different procedures to introduce time-step control for ordinary differential equations (ODEs). In the mid 1960s, Fehlberg proposed an embedded RK method for solving ODEs, when he discovered a fifth-order method with six function evaluations while another combination of these functions yields a fourth-order scheme [17–19]. The main idea of this method is to use two RK schemes of different order. Basically, one estimates the error by computing the difference between a solution calculated with a given scheme and the one obtained using a scheme with a higher order of accuracy. This so-called *truncation error* can then be exploited to control the global error by changing the time-step size. Recently, the RKF method has been applied to PDEs using second- and third-order RK formulas [15, 20]. Numerical results have been presented for first-order hyperbolic PDEs in one space dimension.

Inspired by this work, we develop here a simplified and more efficient time-step control for the multidimensional-adaptive MR method presented in [8]. This time scheme uses second- and third-order embedded RK formulas of compact form, which allow to reduce the computational cost, since the flux evaluations of the second-order scheme can be recycled for the third-order one. Instead of estimating an optimal error control, which is very costly, the predicted time step for the next iteration is limited to avoid numerical instabilities. This choice avoids extra computations and circumvents additional storage.

Copyright © 2008 John Wiley & Sons, Ltd.

DOI: 10.1002/nme

For the time-step control, we use RK methods of order $p-1$ and p. Omitting the spatial subscript everywhere, the order $p-1$ scheme reads as follows:

$$\check{u}^{m+1} = \check{u}^m + \sum_{i=1}^{p} \check{b}_i \kappa_i \tag{12}$$

where

$$\begin{aligned} \kappa_1 &= \Delta t\, \bar{\mathscr{D}}(\check{u}^m) \\ \kappa_2 &= \Delta t\, \bar{\mathscr{D}}(c_2 \Delta t, \check{u}^m + a_{21}\kappa_1) \\ &\vdots \\ \kappa_p &= \Delta t\, \bar{\mathscr{D}}(c_p \Delta t, \check{u}^m + a_{p1}\kappa_1 + \cdots + a_{p\,p-1}\kappa_{p-1}) \end{aligned} \tag{13}$$

and the pth order formula is denoted by

$$\hat{u}^{m+1} = \hat{u}^m + \sum_{i=1}^{p+1} \hat{b}_i \kappa_i \tag{14}$$

where $a_{i,j}$, $\check{b}_i$, $\hat{b}_i$ and c_i are the RKF coefficients.

The estimation of the truncation error is defined as the difference between approximations of order p and $p-1$, and hence yields

$$\delta_{\text{old}} = \|\hat{u}^{m+1} - \check{u}^{m+1}\|_\infty \tag{15}$$

By construction, this truncation error δ_{old} scales as $(\Delta t)^p$. Using a time step Δt_{old}, an error δ_{old} is produced, i.e. δ_{old} is a function of Δt_{old}, and similarly a new step Δt_{new} produces an error δ_{desired}, i.e. δ_{desired} is a function of Δt_{new}.

Both are then related by

$$\frac{\Delta t_{\text{new}}}{\Delta t_{\text{old}}} = \left| \frac{\delta_{\text{desired}}}{\delta_{\text{old}}} \right|^{1/p}$$

Therefore, if we fix δ_{desired}, it follows that, if $\delta_{\text{old}} < \delta_{\text{desired}}$, then the time step can be increased, and if $\delta_{\text{old}} \geqslant \delta_{\text{desired}}$, then the time step must be decreased. Hence, this procedure allows to adjust automatically the step size in order to achieve a prescribed accuracy in time.

Note that, for systems of PDEs, u has more than one component, and we thus take the maximum over the components, which have been normalized by dividing by their averages of the computational domain. The reason is to compare quantities of similar order of magnitude.

Nevertheless, when $(\Delta t)_{\text{new}}$ is increased too much, the new predicted value may fail to meet the desired accuracy. In contrast to [15], this is not allowed in the present method, as we cannot go back to the previous time step once the solution at the new time step is computed, due to the fact that we are using a low storage memory model. Hence it is decided here to limit the increase of the time step by introducing a safety factor ($\mathscr{S}$). The new time step $(\Delta t)_{\text{new}}$ is chosen such that

$$-\frac{\mathscr{S}}{2} \leqslant \frac{(\Delta t)_{\text{new}} - (\Delta t)_{\text{old}}}{(\Delta t)_{\text{old}}} \leqslant \frac{\mathscr{S}}{2}$$

Copyright © 2008 John Wiley & Sons, Ltd.

DOI: 10.1002/nme

Using a more stringent limiter, i.e. a safety factor, the choice of non-admissible time steps can be avoided. The drawback of the limiter is that, when the initial time step is far from the ideal time step, some CPU time could be wasted, since the time step cannot be increased sufficiently fast. To overcome this, we introduce a time-dependent limiter $\mathscr{S}=\mathscr{S}(t)$ with an exponential decay during the first time steps, i.e.

$$\mathscr{S}(t)=(\mathscr{S}_0-\mathscr{S}_{\min})\exp\left(-\frac{t}{\Delta t}\right)+\mathscr{S}_{\min}$$

The behavior of the limiter $\mathscr{S}(t)$ for $t=0$ is the maximal allowed variation $\mathscr{S}_0$ and, for $t\to\infty$, it is $\mathscr{S}_{\min}$, where $\mathscr{S}_{\min}<\mathscr{S}_0$. Motivated from our numerical experiments, we use $\mathscr{S}_0=0.1$ and $\mathscr{S}_{\min}=0.01$ for all case studies presented in Section 3. This means that we allow 10% of variation of the time step in the initial time step and after few iterations we allow only 1%.

The implementation of RK 3(2) is no more costlier than the classical RK3, thanks to the fact that we use a compact scheme based on the RK2 result to calculate RK3 without any additional flux evaluation. Omitting the spatial subscript everywhere, the RK2 method implemented here is

$$\begin{aligned}\bar{u}^{*}&=\bar{u}^{n}\Delta t\bar{\mathscr{D}}(\bar{u}^{n})\\ \bar{u}^{n+1}&=\tfrac{1}{2}[\bar{u}^{n}+\bar{u}^{*}+\Delta t\bar{\mathscr{D}}(\bar{u}^{*})]\end{aligned}\tag{16}$$

and the RK3 method is

$$\begin{aligned}\bar{u}^{**}&=\tfrac{1}{4}[3\bar{u}^{n}+\bar{u}^{*}+\Delta t\bar{\mathscr{D}}(\bar{u}^{*})]\\ \bar{u}^{n+1}&=\tfrac{1}{3}[\bar{u}^{n}+2\bar{u}^{**}+2\Delta t\bar{\mathscr{D}}(\bar{u}^{**})]\end{aligned}\tag{17}$$

The storage requirement of the RKF 3(2) is slightly larger than the usual RK3 method because it is also necessary to store the second-stage of the RK2 computation.

2.3. Space–time integration

Now we briefly summarize the MR algorithm for the time evolution of the solution computed on the adaptive grid. First, depending on the initial condition, an initial graded tree is created by performing successive refinement where details are above the given threshold. Then, given the graded tree structure, a time evolution is made on the leaves, computing RK2 and RK3 for a given initial time step. In addition, using that result, the time adaption is made to compute the time step for the next iteration. After that the details are computed by the MR transform. A thresholding operation is applied, while respecting the graded tree structure. After having applied an inverse MR transform to get back the cell averages, another time adaption is made to compute the time step for the next iteration.

Denoting by $\bar{\mathbf{E}}(\Delta t^n)$ the discrete-time evolution operator, Δt^n being the time step at the iteration n, the algorithm can schematically be summarized by

$$\bar{U}^{n+1}=\bar{\mathbf{M}}^{-1}\cdot\mathbf{T}(\varepsilon)\cdot\bar{\mathbf{M}}\cdot\bar{\mathbf{E}}(\Delta t^n)\cdot\bar{U}^{n}\tag{18}$$

where $\bar{\mathbf{M}}$ is the MR transform operator, $\bar{\mathbf{M}}^{-1}$ its inverse and $\mathbf{T}(\varepsilon)$ the threshold operator with tolerance ε.

An appropriate choice of the tolerance ε allows to equilibrate the perturbation and discretization error and therewith to maintain the approximation order of the scheme on the regular grid [8].

Copyright © 2008 John Wiley & Sons, Ltd.

DOI: 10.1002/nme

3. NUMERICAL RESULTS

In this section, we present numerical examples in one, two and three space dimensions using FV second-order accurate schemes with third-order RK time integration. In the case of an adaptive space discretization, an MR analysis of order $r=3$ is used. The time control is done by the RK 3(2) method. In the following, the different methods are applied to the viscous Burgers equation in one space dimension and reaction–diffusion equations in two and three space dimensions. We compare MR with time-step control with MR computations using a fixed time step and also with the FV reference scheme on a fixed grid and fixed time step. This allows to measure the speed-up of the adaptive schemes with respect to the FV scheme. To assess the memory reduction of the adaptive MR computations with or without fixed time step, the ratio between the number of cells in the adaptive and regular fine grids is also given.

3.1. Viscous Burgers equation in one dimension

First, we consider the viscous Burgers equation

$$\frac{\partial u}{\partial t}+u\frac{\partial u}{\partial x}=\nu\frac{\partial^2 u}{\partial x^2}$$

where $u=u(x,t)$, $t\geqslant 0$, $x\in[-1,1]$ and $\nu=10^{-2}/\pi$. The above equation is completed with periodic boundary conditions. The initial condition is given by

$$u(x,0)=-\sin(\pi x)$$

Figure 1 shows the initial condition and the numerical solution at $t=1.6037/\pi$ computed with the MR method with time-step control (MR/CTS). Table I summarizes the memory and CPU time compressions for the five different methods using three different maximal scales $L=11$, 12 and 13, which correspond to grids of size 2^L. First, we observe that all space and time-adaptive methods require less CPU time than the FV method using a fixed time step. For $L=11$, we find that the adaptive MR method with a fixed time step (MR) only requires 55% of the CPU time and 42.7% of the memory compared with the FV method. The adaptive MR method with time-step control (MR/CTS) yields a further reduction of CPU time by a factor four with respect to MR, i.e. only 14% of the CPU time of the FV computations is needed.

Concerning the memory requirements, both space-adaptive methods yield comparable results, i.e. less than 44% of FV memory is used. For an increasing number of scales, both memory and CPU time compression are improved (Table I). For example, for $L=13$ scales, we only need 5 and 28% of the CPU time with respect to the FV method, and 17.6 and 19.4% of the memory for MR/CTS and MR, respectively. The above results illustrate that MR/CTS yields a speed-up of more than 5 with respect to MR and of 20 with respect to FV.

To verify the precision of the different numerical methods, we compare the slope of the numerical solution $|\partial u/\partial x|_{\max}$ at $\pi t=1.6037$ with the slope of the exact solution given in [21], which yields 152. Table I shows that the slope of the numerical solutions is close to the one of the exact solution, and that the precision is improved for an increasing number of scales. The introduction of adaptive time stepping does not significantly affect the memory compression and the precision of the computation. It can also be noticed that, when the solution exhibits small scales, the desired accuracy in time has to be reduced in order to avoid numerical instabilities, like spurious oscillations.

Copyright © 2008 John Wiley & Sons, Ltd.

DOI: 10.1002/nme

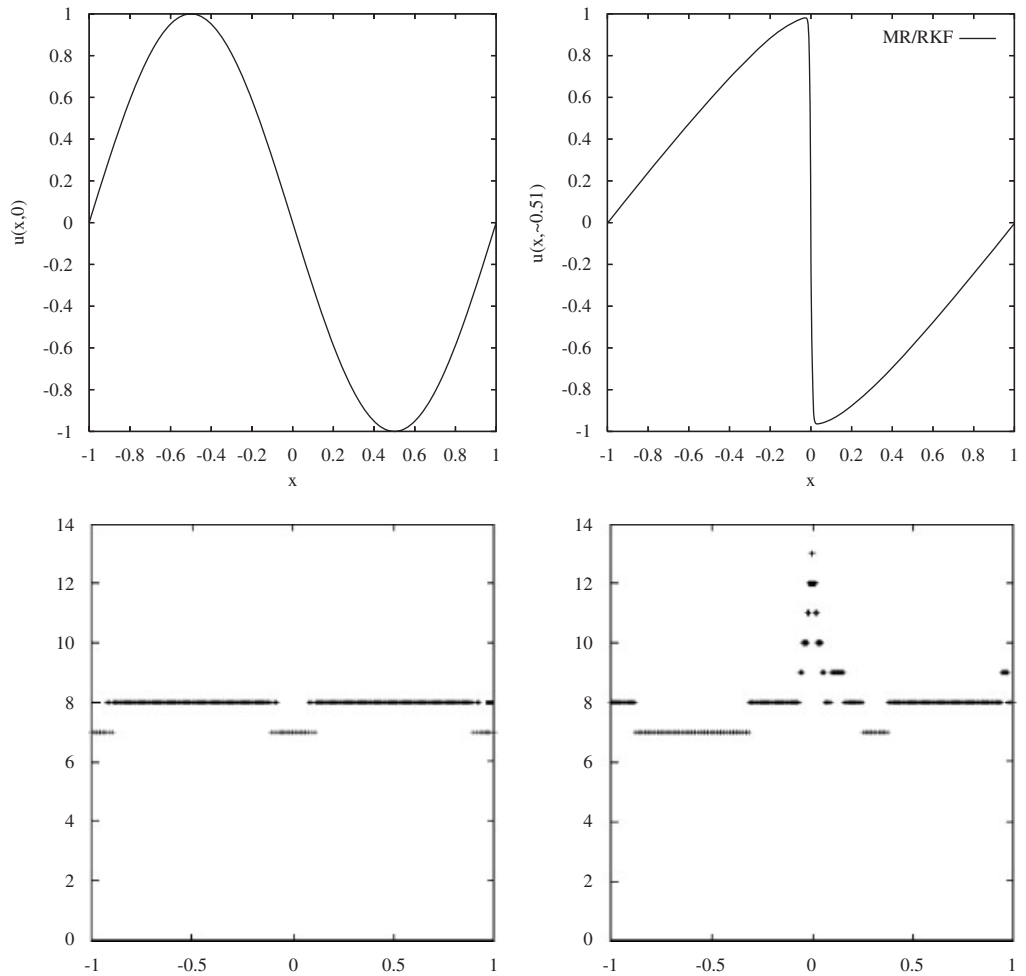

Figure 1. Top: initial condition and numerical solution of the Burgers equation at $t = 1.6037/\pi \approx 0.51$, computed using the MR/CTS method with $L = 13$ levels. Bottom: corresponding adaptive grids.

In Figure 2, we plot the evolution of the time step for the MR/CTS method, using different numbers of maximum scales, $L = 11, 12$ and 13. For late times, i.e. $t > 0.4$, we see that the time step is decreasing for increasing spatial resolution. At earlier times the evolutions strongly depend on the maximum number of scales and we find different behavior during the evolution.

3.2. Diffusion–reaction equations in 2D

In the following sections, we perform numerical simulations of cellular instabilities of flame balls. 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 [22].

Copyright © 2008 John Wiley & Sons, Ltd.

DOI: 10.1002/nme

Table I. CPU and memory compression for the different methods, using an initial CFL=0.4, $\varepsilon = 10^{-2}$ and the limiter parameters $\mathscr{S}_{\min} = 0.01$, and $\mathscr{S}_0 = 0.10$.

Method	Scales	#Steps	%CPU time	%Memory	$\lvert\partial u/\partial x\rvert_{\max}$	δ_{desired}
FV	11	18346	100	100	149.6	
MR	11	18346	55	42.7	149.5	
MR/CTS	11	3697	14	43.3	147.5	10^{-5}
FV	12	70767	100	100	151.7	
MR	12	70767	37	26.2	149.8	
MR/CTS	12	13370	8	27.2	149.4	10^{-6}
FV	13	277839	100	100	153.2	
MR	13	277839	28	19.4	151.8	
MR/CTS	13	56741	5	17.6	150.7	10^{-6}

At $\pi t = 1.6037$ we compare the slope of the numerical solutions with the exact one $\lvert\partial u/\partial x\rvert_{\max} = 152$.

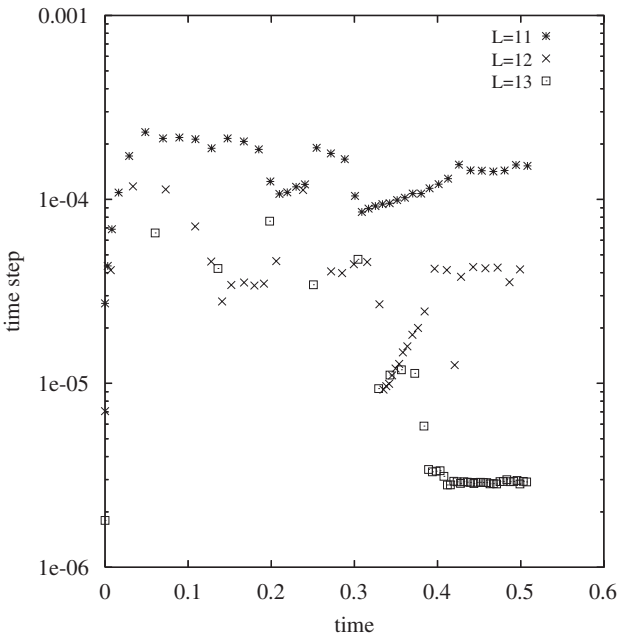

Figure 2. Evolution of the time step for the viscous Burgers equation from $t = 0$ to $t = 0.51$, using MR/CTS method with $L = 11, 12$ and 13.

The thermodiffusive approximation is well adapted for the computation of flame balls, because the flame velocity is very small [23]. We consider also a constant density approximation and one-step chemical kinetics. The system of equations modelling such a flame structure is a set of

Copyright © 2008 John Wiley & Sons, Ltd.

DOI: 10.1002/nme

two diffusion–reaction equations of the form [23]

$$\begin{aligned}\frac{\partial T}{\partial t} &= \nabla^2 T + \omega - s \\ \frac{\partial Y}{\partial t} &= \frac{1}{Le}\nabla^2 Y - \omega\end{aligned} \tag{19}$$

where $T = T(\mathbf{x}, t)$ denotes the dimensionless temperature, $Y = Y(\mathbf{x}, t)$ the partial mass of the limiting reactant and Le the Lewis number. The dimensionless reaction rate is

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

where Ze denotes the Zeldovich number and α the burnt–unburnt temperature ratio. The heat loss due to radiation follows the Stefan–Boltzmann law and is written as

$$s = s(T) = \gamma[(T + \alpha^{-1} - 1)^4 - (\alpha^{-1} - 1)^4]$$

where γ is the dimensionless radiation coefficient. The global reaction rate is

$$R(t) = \int_\Omega \omega \, \mathrm{d}x \, \mathrm{d}y$$

The initial conditions are

$$T(r, 0) = \begin{cases} 1 & \text{if } r \leqslant r_0 \\ \exp\left(1 - \dfrac{r}{r_0}\right) & \text{if } r > r_0 \end{cases} \tag{20}$$

$$Y(r, 0) = \begin{cases} 0 & \text{if } r \leqslant r_0 \\ 1 - \exp\left[Le\left(1 - \dfrac{r}{r_0}\right)\right] & \text{if } r > r_0 \end{cases} \tag{21}$$

where r_0 denotes the initial radius of the flame ball. The boundaries are far enough from the flame ball, so that they have a negligible influence. Hence we decided to use Neumann boundary conditions.

The circular initial condition is perturbed by stretching the circle in one direction. We also apply a rotation, so that the symmetry axes of the problem are not lined up with the grid. Hence we have

$$r = \sqrt{\frac{X^2}{a^2} + \frac{Y^2}{b^2}}$$

where

$$\begin{aligned} X &= x \cos\theta + y \sin\theta \\ Y &= -x \sin\theta + y \cos\theta \end{aligned}$$

Copyright © 2008 John Wiley & Sons, Ltd.

DOI: 10.1002/nme

In the computations, the Lewis number is $Le=0.3$, the Zeldovich number is $Ze=10$ and the temperature ratio $\alpha=0.64$. The aspect ratio of the ellipse is given by $a=2$, $b=1$, the rotation angle is $\theta=-\pi/6$, and the initial radius is $r_0=1$. The computational domain is $\Omega=[-30,30]^2$ and the elapsed time is $t=15$. The initial time step is calculated using the stability condition of the heat equation, while neglecting the non-linear reaction and radiation terms, i.e.

$$(\Delta t)_0=\frac{c_\lambda}{2}\lambda(\Delta x)^2$$

where $\lambda=\min(1,Le)$. For the computations with fixed time step, this value is used for every time iteration. Usually, we have $c_\lambda<0.5$. However, due to the non-linearity of both reaction and radiation source terms, this choice in fact does not guarantee the stability of the method. For such problems, numerical methods with adaptive time stepping are well adapted.

Figure 3 shows the time evolution of the isotherms, together with the corresponding adaptive grids, computed with the MR/CTS method. We observe that the elliptical flame structure is splitting and growing in space. At the two extremities of the long axis of the ellipse, the tangential reactant flux tends to increase the concentration of the fresh gas and with it the chemical reaction. Therefore, the flame is propagating faster in the direction of the long axis of the ellipse. As a consequence, the flame is splitting at the two extremities of the short axis of the ellipse. Owing to the radiative heat loss, we also observe a local extinction of the reaction at these extremities. These two cells split again into smaller cells. This phenomenon is the so-called cellular flame instability.

In Table II, CPU time and memory compressions are given for the three methods, FV, MR and MR/CTS, using $L=8,9$ scales, which corresponds to a maximal resolution of 256×256 and 512×512 grid points, respectively. Using MR with RK3, we only require 28.69% of the CPU time required by the fine-grid computation, and using MR/CTS with RK3(2), we only need 10.13% of CPU time, compared with FV. We observe that the memory compression is not affected by introducing time adaptivity, i.e. less than 14% coefficients are needed for adaptive methods in comparison with the FV method. The memory requirement of MR/CTS is slightly larger than the one of MR due to the additional storage of the RKF 3(2) method with respect of RK3.

Concerning the precision, we find that the global reaction rate agrees well with the one obtained with the FV computation, whatever the chosen adaptive method (cf. Table II). Using nine scales, the error on the global reaction rate is less than 0.15% for the MR method and 0.3% for the MR/CTS method.

On the left side of Figure 4, we plot the time evolution of the time step, showing 1 out of 100 steps for the MR/CTS method with $L=9$ scales. For this test-case, we observe that the time step tends toward a value around 1.3×10^{-3}.

3.3. Diffusion–reaction equations in three dimensions

Now, we solve system (19), in three space dimensions, with a flame ball initially stretched to study 3D cellular flame instabilities. Hence, as initial condition, we choose a spherical flame stretched in one direction and apply a rotation on two axes, so that

$$r=\sqrt{\frac{X^2}{a^2}+\frac{Y^2}{b^2}+\frac{Z^2}{c^2}}$$

Copyright © 2008 John Wiley & Sons, Ltd.

DOI: 10.1002/nme

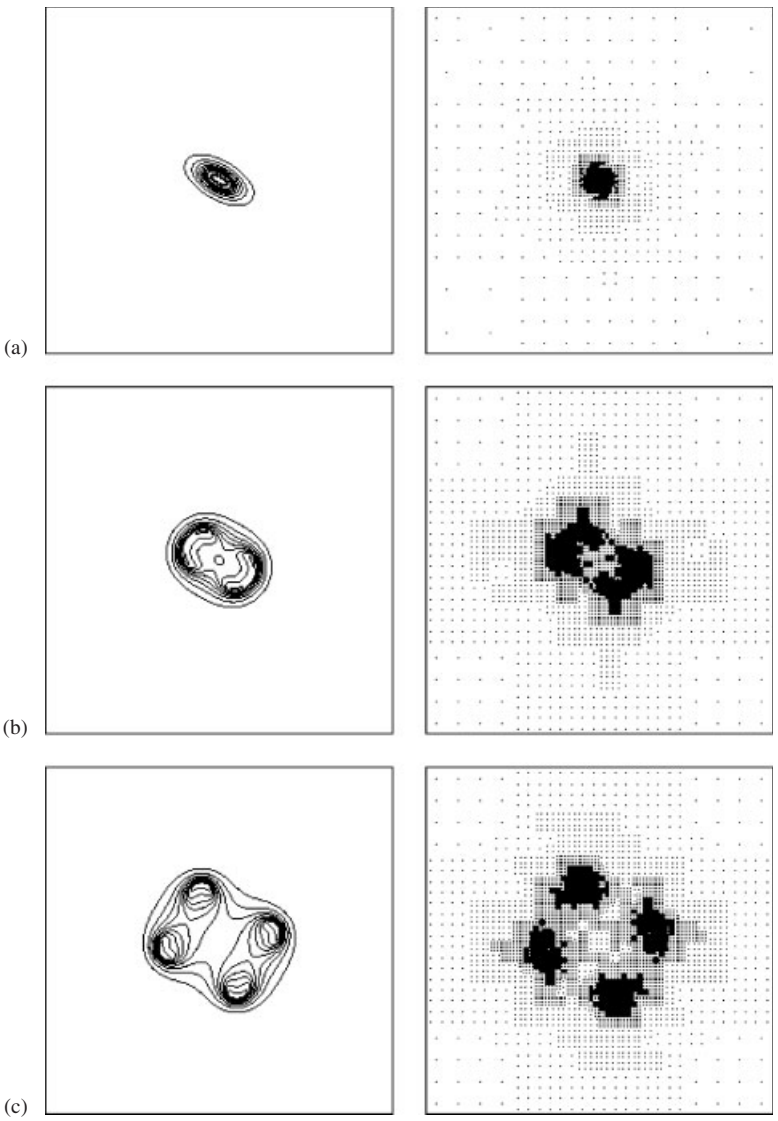

Figure 3. Numerical simulation of a 2D splitting flame ball using the MR/CTS method, $L=9$ scales, $Le=0.3$, $Ze=10$, $\alpha=0.64$, $\gamma=0.05$. Isotherms from $T=0.1$ to 1 every 0.1 at $t=0$ (a); $t=7.5$ (b); and $t=15$ (c), are shown (left) with their corresponding adaptive meshes (right).

where

$$X = x\cos\theta - y\sin\theta$$

$$Y = (x\sin\theta + y\cos\theta)\cos\varphi - z\sin\varphi$$

$$Z = (x\sin\theta + y\cos\theta)\sin\varphi + z\cos\varphi$$

Copyright © 2008 John Wiley & Sons, Ltd.

DOI: 10.1002/nme

Table II. Numerical simulation of a 2D splitting flame ball.

Method	#Steps	%CPU time	%Memory	ε	δ_0	R
$L=8$						
FV	6069	100.00	100.00			38.73
MR	6069	28.69	13.29	10^{-2}		38.66
MR/CTS	3311	10.13	13.21	10^{-2}	10^{-4}	38.55
$L=9$						
FV	24273	100.00	100.00			39.94
MR	24273	11.43	5.05	10^{-2}		39.88
MR/CTS	12027	7.58	5.03	10^{-2}	10^{-5}	39.78

CPU and memory compressions for the different methods, using initial $c_\lambda=0.3$, with 8 or 9 scales, with limiter parameters $\mathscr{S}_{\min}=0.01$ and $\mathscr{S}_0=0.10$. The global reaction rate R is given at $t=15$.

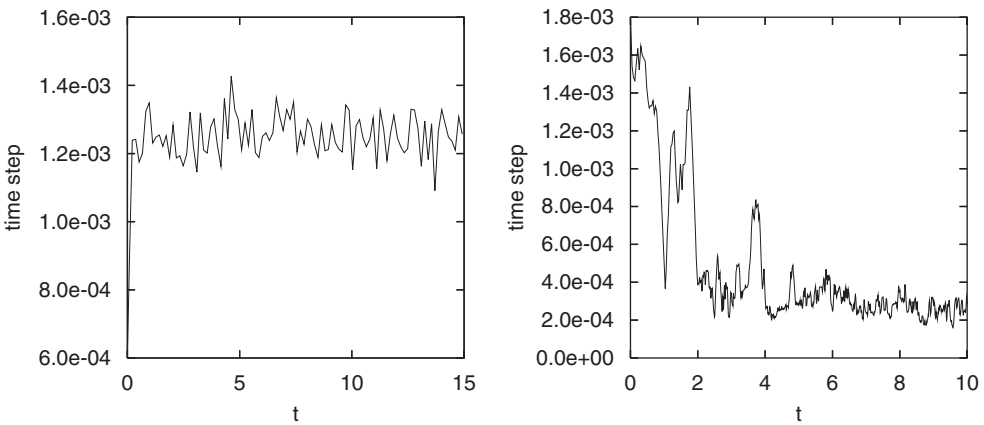

Figure 4. Evolution of the time step using the MR/CTS method for the splitting flame ball, with $L=9$ scales in two dimensions (left), with $L=8$ scales in three dimensions (right), where $\mathscr{S}_{\min}=0.01$, and $\mathscr{S}_0=0.10$.

The parameters of the ellipsoid are $a=b=1.5$, $c=3$, and the rotation angles are $\theta=\pi/3$ and $\varphi=\pi/4$. The initial radius is $r_0=1$. The computational domain is $\Omega=[-20,20]^3$ and we use $L=8$ scales. As for the 2D case, the Lewis number is $Le=0.3$, the Zeldovich number $Ze=10$ and the temperature ratio $\alpha=0.64$. To accelerate the splitting, we chose a larger radiation coefficient, i.e. $\gamma=0.1$. The dimensionless elapsed time is $t=10$.

In Figure 5, we observe the ellipsoid splitting along its two shortest axes and growing in space. After the first splitting, the new cells split again. In Table III, we give the performances for both MR and MR/CTS computations. To give some absolute values about the CPU time using a Pentium 4 (tm) 2.4 GHz processor, the MR computation required around 8 days and 4 h, and the MR/CTS around 5 days and 2 h. The FV computation was performed on a few iterations only to estimate the CPU time that such computation would require, which is about 379 days using the same computer. The memory requirement of MR/CTS is also slightly larger than the

Copyright © 2008 John Wiley & Sons, Ltd.

DOI: 10.1002/nme

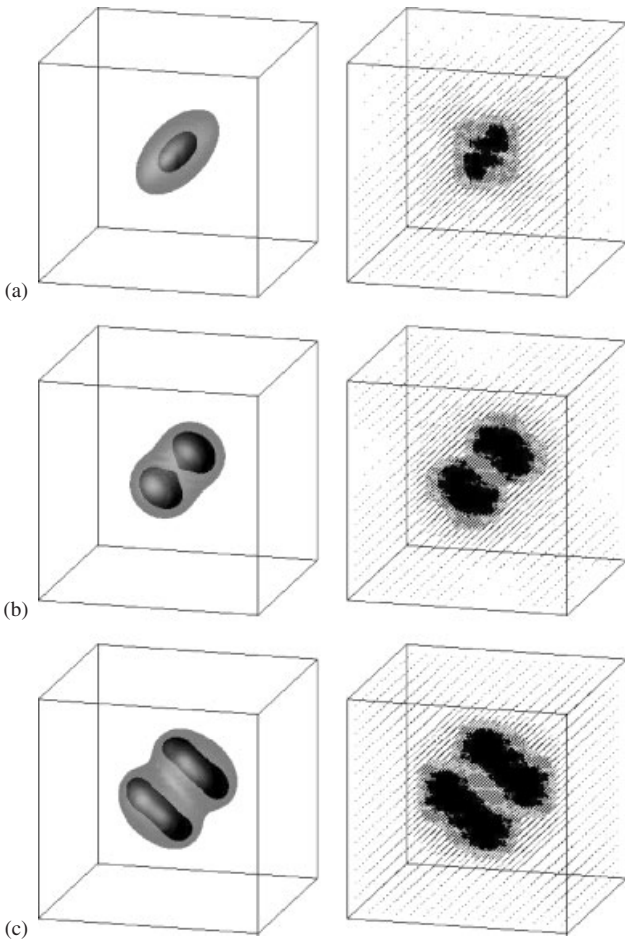

Figure 5. Numerical simulation of a 3D splitting flame ball using the MR/CTS method, $L=8$ scales, $Le=0.3$, $Ze=10$, $\alpha=0.64$, $\gamma=0.1$ are shown. Isotherms $T=0.5$ (black) and $T=0.1$ (gray) at $t=0$ (a); $t=5$ (b); and $t=10$ (c) are presented (left) with corresponding adaptive meshes (right) for these times.

one of MR due to the additional storage of the RKF 3(2) method with respect of RK3. The MR/CTS implementation is significantly more efficient that the MR; however, slightly less accurate than MR.

We remark that the MR/CTS method reduces the CPU time by a factor 1.6, in comparison with the MR method. The difference on the global reaction rate remains small, i.e. less than 0.3%. On the right side of Figure 4, the time evolution of the time step is plotted. We find that the time step is decreased in the computation and tends toward a value around 3×10^{-4}. In fact, the main part of the gain in CPU time of this method is obtained for $t<4$, where a larger time step can be used, due to the fact that the local flame velocity is smaller. Hence, for problems where no

Copyright © 2008 John Wiley & Sons, Ltd.

DOI: 10.1002/nme

Table III. Numerical simulation of a 3D splitting flame ball.

Method	#Steps	%CPU time	%Memory	ε	δ_0	R
MR	62 501	2.16	1.05	0.05		677.1
MR/CTS	28 871	1.35	1.24	0.05	10^{-4}	675.0

CPU and memory compressions for the different methods, using initial $c_\lambda = 0.5$, with $L = 8$ scales, and limiter parameters $\mathscr{S}_{\min} = 0.01$ and $\mathscr{S}_0 = 0.10$. The global reaction rate R given at $t = 10$.

absolute stability criterion exists, an adaptive time stepping enables us to reduce significantly the required CPU time.

4. CONCLUSIONS AND PERSPECTIVES

The present paper describes a new space-adaptive method with time-step control to solve multidimensional PDEs. It is based on a finite volume (FV) discretization with explicit time stepping. The previously developed adaptive multiresolution (MR) scheme [8, 9] has been further improved using an automatic time-step control. We demonstrated the efficiency of the new method for different test problems in one, two and three space dimensions. We studied its performance by comparing the CPU time and the memory requirements with the FV method using a uniform discretization and a third-order Runge–Kutta (RK) scheme. The implemented time-step control method uses a low storage in memory. For efficiency reasons, we introduced a limiter on the time step. This avoids inappropriate choices of the time step, i.e. large time steps, which may generate numerical instabilities of the computation. The limiter is a function of time and it is usually larger during the first time steps. It also avoids that inappropriate initial time steps increase the CPU time. The computational extra cost of the time-step control is very low because we use compact formulas to perform the time evolution. We showed that this new method reduces significantly the CPU time while controlling accuracy. Another feature of this technique, applied to non-linear problems for which it is difficult to establish stability criteria, is that the blow up of explicit computations can be avoided.

The time step in the above presented method is fixed for all scales, and hence the finest spatial scale present in the computations imposes the maximal time-step size, in order to guarantee the stability of the explicit RK 3(2) method. The maximal time-step size of the scheme is restricted by the finest spatial scale present in the computations, in order to guarantee the stability of the explicit RK 3(2) method. In [24] we introduced scale-dependent time stepping for the adaptive MR method presented in [8]. The underlying idea is that the time step on larger scales can be increased successively, without violating the stability requirement at a given scale. This allows an additional speed-up due to a reduced number of flux evaluations, since small time steps are only needed locally in regions where small scales are active. In a more recent work on 2D compressible Euler equations [25], we applied the time-step control to the local time-stepping scheme after one complete local time cycle, i.e. when the solution has been completely advanced with the time step of the largest scale and becomes synchronized again. In the cases considered we found that both accuracy and CPU time requirement of the scheme were improved. In a future work, we plan to compute 3D flame balls using local time stepping with time-step control.

Copyright © 2008 John Wiley & Sons, Ltd.

DOI: 10.1002/nme

Future applications of adaptive time schemes are ABRs, where the mesh is replaced on blocks of different refinement levels according to a tree structure. This has been proposed in [14] using finite differences and interpolatory wavelets. This approach could also be extended to FV schemes coupled with a cell-average MR method. The advantage of the ABR is that it is well suited for parallel computing. The different blocks, which are of the same size, can be distributed on different processors, while guaranteeing a load-balance between them.

On a longer term, we plan to extend this space–time-adaptive scheme to the 3D compressible Navier–Stokes equations to perform coherent vortex simulations [26] of compressible turbulent flows.

ACKNOWLEDGEMENTS

This work was supported by the ANR Project 'M2TFP' and the DFG-CNRS Research Program 'LES in complex flows'. M. D. thankfully acknowledges financial support from CNRS Department ST21, and from Ecole Centrale Marseille. We are thankful to Sônia Gomes for fruitful discussions and to Siegfried Müller for useful comments on the manuscript. We are also grateful to Dominique Fougere for her helpful computational assistance.

REFERENCES

1. Becker R, Rannacher R. An optimal control approach to error control and mesh adaptation. *Acta Numerica* 2001; **10**:1–102.
2. Süli E, Mayers DF. *An Introduction to Numerical Analysis*. Cambridge University Press: Cambridge, 2003.
3. Berger MJ, Oliger J. Adaptive mesh refinement for hyperbolic partial differential equations. *Journal of Computational Physics* 1984; **53**:484–512.
4. Harten A. Multiresolution algorithms for the numerical solution of hyperbolic conservation laws. *Communications on Pure and Applied Mathematics* 1995; **48**:1305–1342.
5. Harten A. Multiresolution representation of data: a general framework. *SIAM Journal on Numerical Analysis* 1996; **33**:385–394.
6. Cohen A, Kaber SM, Müller S, Postel M. Fully adaptive multiresolution finite volume schemes for conservation laws. *Mathematics of Computation* 2003; **72**:183–225.
7. Chiavassa G, Donat R. Point value multi-scale algorithms for 2d compressible flow. *SIAM Journal on Scientific Computing* 2001; **23**:805–823.
8. Roussel O, Schneider K, Tsigulin A, Bockhorn H. A conservative fully adaptive multiresolution algorithm for parabolic PDEs. *Journal of Computational Physics* 2003; **188**:493–523.
9. Roussel O, Schneider K. An adaptive multiresolution method for combustion problems: application to flame ball–vortex interaction. *Computers and Fluids* 2005; **34**:817–831.
10. Cohen A. Wavelet methods in numerical analysis. In *Handbook of Numerical Analysis*, Ciarlet PG, Lions JL (eds), vol. VII. Elsevier: Amsterdam, 2000.
11. Holmström M. Wavelet based methods for time dependent PDEs. *Ph.D. Thesis*, Uppsala University, 1997.
12. Walden J. A general adaptive solver for hyperbolic PDEs based on filter bank subdivisions. *Applied Numerical Mathematics* 2000; **33**:317–325.
13. Vasilyev OV, Browman C. Second generation wavelet collocation method for the solution of partial differential equations. *Journal of Computational Physics* 2000; **165**:660–693.
14. Domingues MO, Gomes SM, Diaz LMA. Adaptive wavelet representation and differentiation on block-structured grids. *Applied Numerical Mathematics* 2003; **47**:421–437.
15. Ferm L, Löstedt P. Space–time adaptive solutions of first order PDEs. *Journal of Scientific Computing* 2006; **26**:83–110.
16. Roe PL. Approximate Riemann solvers, parameter vectors, and difference schemes. *Journal of Computational Physics* 1981; **43**:357–372.
17. Fehlberg E. New high-order Runge–Kutta formulas with step size control for systems of first- and second-order differential equations. *Zeitschrift für Angewandte Mathematik und Mechanik* 1964; **44**:T17–T29.

Copyright © 2008 John Wiley & Sons, Ltd.

DOI: 10.1002/nme

18. Fehlberg E. Klassische Runge–Kutta Formeln fünfter und siebenter Ordnung mit Schrittweiten–Kontrolle. *Computing* 1969; **4**(2):93–106.
19. Stoer J, Bulirsch R. *Introduction to Numerical Analysis* (2nd edn). Text in Applied Mathematics, vol. 12. Springer: Berlin, 1991.
20. Ferm L, Löstedt P. Adaptive error control for steady state solutions of inviscid flow. *Journal of Scientific Computing* 2002; **23**:1777–1798.
21. Basdevant C, Deville M, Haldenwang P, Lacroix JM, Ouazzani J, Peyret R, Orlandi P, Patera AT. Spectral and finite difference solutions of Burgers equation. *Computers and Fluids* 1986; **14**:23–41.
22. Ronney PD. Near-limit flame structures at low Lewis number. *Combustion and Flame* 1990; **82**:1–14.
23. Bockhorn H, Fröhlich J, Schneider J. An adaptive two-dimensional wavelet-vaguelette algorithm for the computation of flame balls. *Combustion Theory and Modelling* 1999; **3**:1–22.
24. Domingues MO, Gomes SM, Roussel O, Schneider K. An adaptive multiresolution scheme with local time stepping for evolutionary PDEs. *Journal of Computational Physics* 2008; **227**:3758–3780.
25. Domingues MO, Gomes SM, Roussel O, Schneider K. Space–time adaptive multiresolution methods for hyperbolic conservation laws: applications to compressible Euler equations. *Applied Numerical Mathematics* 2008; accepted.
26. Farge M, Schneider K. Coherent vortex simulation (CVS), a semi-deterministic turbulence model using wavelets. *Flow*, *Turbulence and Combustion* 2001; **66**:393–426.

Copyright © 2008 John Wiley & Sons, Ltd.

DOI: 10.1002/nme