INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN FLUIDS
Int. J. Numer. Meth. Fluids 2006; **52**:455–471
Published online 10 February 2006 in Wiley InterScience (www.interscience.wiley.com). DOI: 10.1002/fld.1191

Adaptive mesh refinement techniques for high-order shock capturing schemes for multi-dimensional hydrodynamic simulations

Antonio Baeza[‡] and Pep Mulet[*,†]

Dpt. Matemàtica Aplicada, Universitat de València, Spain

SUMMARY

The numerical simulation of physical phenomena represented by non-linear hyperbolic systems of conservation laws presents specific difficulties mainly due to the presence of discontinuities in the solution. State of the art methods for the solution of such equations involve high resolution shock capturing schemes, which are able to produce sharp profiles at the discontinuities and high accuracy in smooth regions, together with some kind of grid adaption, which reduces the computational cost by using finer grids near the discontinuities and coarser grids in smooth regions. The combination of both techniques presents intrinsic numerical and programming difficulties. In this work we present a method obtained by the combination of a high-order shock capturing scheme, built from Shu–Osher's conservative formulation (*J. Comput. Phys.* 1988; **77**:439–471; 1989; **83**:32–78), a fifth-order weighted essentially non-oscillatory (WENO) interpolatory technique (*J. Comput. Phys.* 1996; **126**:202–228) and Donat–Marquina's flux-splitting method (*J. Comput. Phys.* 1996; **125**:42–58), with the adaptive mesh refinement (AMR) technique of Berger and collaborators (Adaptive mesh refinement for hyperbolic partial differential equations. *Ph.D. Thesis*, Computer Science Department, Stanford University, 1982; *J. Comput. Phys.* 1989; **82**:64–84; 1984; **53**:484–512). Copyright © 2006 John Wiley & Sons, Ltd.

KEY WORDS: hyperbolic systems of conservation laws; adaptive mesh refinement; Shu–Osher conservative scheme

1. INTRODUCTION

It is well known that the solutions of hyperbolic systems of conservation laws can develop discontinuities even if the initial data are smooth. Since the differential equations that represent the physical conservation laws have been derived from integral forms under smoothness assumptions, the PDE's will not hold in the classical sense at all points.

*Correspondence to: P. Mulet, Dpt. Matemàtica Aplicada, Universitat de València, Spain.
†E-mail: mulet@uv.es
‡E-mail: Antonio.Baeza@uv.es

Contract/grant sponsor: EUCO; contract/grant numbers: HPRN-CT-2002-00282, HPRN-CT-2002-00286

Received 21 June 2005
Revised 25 November 2005
Accepted 26 December 2005

To address these difficulties the so-called high resolution shock capturing (HRSC) schemes arose. These methods provide an accurate (second or higher order) approximation to the solution in smooth regions and sharp profiles of the discontinuities with no significant oscillations.

The accuracy of the numerical simulations is, however, limited by the mesh size, since no method can resolve phenomena whose physical scale is smaller than the mesh size. To obtain solutions that resolve the small-scale features of the solution it is necessary to apply HRSC schemes to very fine computational grids. The computational cost of such calculations is very high, and is usually measured in terms of days or months. Even with several computers working in parallel a lot of problems is nowadays unapproachable.

Since fine grids are needed only in the parts of the solution which have non-smooth structure, some techniques have been developed to reduce the computational cost, mainly based on the use of non-uniform grids [1, 2]. These algorithms use a grid with cells of variable size, trying to use cells of small size in some regions of interest, maintaining cells of bigger size in smooth regions. These grids are often difficult to manipulate in more than one space dimension, because the solution at a cell depends on the solution at some neighbourhood around it. The use of cells of mixed size renders difficult the computation of the solution at the next time step.

Another difficulty of such approaches is stability. The Courant–Friedrichs–Lax (CFL) condition imposes a limit on the size of the time step, which depends on the size of the smallest cell. Small time steps have to be used for the whole computational domain, even if in smooth regions the CFL condition could be relaxed if the grids were uniform. See nevertheless Reference [3] for recent developments in schemes for moving meshes and local time-stepping.

Adaptive mesh refinement (AMR) [4–7] adds a new feature to this pool: temporal refinement. The goal of the AMR procedure is to perform as few cell updates as possible, instead of reducing the number of cells. The idea is to use a hierarchical set of Cartesian, uniform meshes that live at different resolution levels. At the coarser level there is a set of coarse mesh patches covering the whole domain. Mesh patches at the next resolution level are obtained by the sub-division of groups of coarse cells. By repeating this sub-division procedure one can cover the regions of interest with mesh patches so that the non-smooth structure of the solution can be resolved with the desired resolution. The grids at different resolution levels coexist, and some mesh connectivity information is needed to connect the solutions at different resolution levels. Provided the connectivity information, each mesh patch can be viewed in isolation and can be integrated independently. The presence of discontinuities at a small part of the domain does not restrict the time step that can be used at the coarse grid.

In this work, we present a numerical method to solve hyperbolic systems of conservation laws obtained by the combination of a high-order shock capturing scheme, built from Shu–Osher's conservative formulation [8, 9], a fifth-order weighted essentially non-oscillatory (WENO) interpolatory technique [10] and Donat–Marquina's flux-splitting method [11], combined with the AMR technique developed by Berger *et al.* [4–6]. The integration algorithm has been successfully tested on fixed grids in Reference [12].

The paper is organized as follows. Section 2 describes the AMR algorithm and the implementation of the whole method. The numerical method used to integrate the equations is briefly reviewed in Section 3. One- and two-dimensional simulations of a 1.22 Mach shock wave impinging a Helium bubble [13] are used in Section 4 to validate the algorithm. Finally, in Section 5 we point out the conclusions.

2. ADAPTIVE MESH REFINEMENT

The AMR algorithm is a general purpose framework for the efficient numerical integration of hyperbolic systems of equations. The algorithm was first developed by Berger in Reference [4] and in the joint works with Oliger [5] and Colella [6]. In this section, we will mainly follow the simplified approach of Quirk [7]. We refer to the cited references for a more complete description.

The AMR algorithm tries to fit the grid's resolution to the needs of the numerical solution by refining the grid only in regions where the solution has non-smooth structure. Under favourable circumstances the algorithm requires only a part of the computational power needed by an equivalent fine grid of uniform size.

This computational efficiency comes from the refinement in space provided by the hierarchical grid system, together with the refinement in time. Moreover, no special restrictions are imposed on the basic numerical method used to integrate the equations.

On the basis of the AMR algorithm, lies the fact that the solution of an hyperbolic system of conservation laws is composed by waves moving through regions in which the solution is smooth. The main difficulty of HRSC schemes consists of catching and resolving these waves, since the regions of smoothness can be solved with no major difficulties.

It is well known that the movement of the waves is governed by the eigenstructure of the Jacobian matrix of the system. In particular, the speed of the waves is controlled by the eigenvalues of the Jacobian matrix.

The main idea is that the AMR algorithm can ensure that, if the waves are initially covered by a fine grid, they will always be, by adapting the grids before the waves can escape.

2.1. Hierarchical grid system

The AMR algorithm uses a hierarchical grid system composed by a set of Cartesian coarse mesh patches, which constitutes the level 0 of the hierarchy and defines the computational domain. These patches can be refined locally by defining finer mesh patches that form the level 1 of the hierarchy. The finer patches are obtained by the sub-division of groups of coarse cells that have been marked for refinement. This process can be repeated to obtain even finer patches at level 2. Grids of the desired resolution can be obtained by iterating this process. Following Quirk's terminology [7], the term grid will refer to the set of mesh patches that belong to the same level, while a mesh (or mesh patch) refers to a single patch, which is a Cartesian product of intervals, i.e.

$$G_{\ell,k} = \prod_{i=1}^{d} [a_i, b_i]$$

where d is the number of space dimensions.

A grid hierarchy G with L levels of refinement is composed by L grids $\{G_\ell\}_{\ell=0}^{L-1}$, each of them formed by the union of some mesh patches

$$G_\ell = \bigcup_{k=1}^{N_\ell} G_{\ell,k}$$

where N_ℓ indicates the number of mesh patches at level ℓ.

The patches at different levels coexist, and are defined in a way such that the grid at a certain level ℓ is contained in the coarser grid at level $\ell - 1$. This nestedness property is essential in the AMR algorithm.

We denote by $r_\ell^1, \ldots, r_\ell^d$ the refinement factors corresponding to level ℓ in the grid hierarchy, where r_ℓ^k represents the number of sub-divisions made in the kth Cartesian direction in the adaption process. Cells at level $\ell - 1$ are divided into $\prod_{k=1}^{d} r_{\ell-1}^k$ cells. For simplicity we will suppose that the refinement factors are the same in each direction, i.e. $r_\ell^1 = \cdots = r_\ell^d$, so we represent them all by r_ℓ. Usual values of r_ℓ are 2 and 4.

2.2. The adaption process

Given a grid G_ℓ the adaption process obtains a set of mesh patches that will compose the grid $G_{\ell+1}$. This adaption procedure is composed by three major processes: first, a process to decide which cells of the grid G_ℓ have to be refined is needed; second, a clustering procedure groups the marked cells into Cartesian patches; finally, the newly created mesh patches are filled with a solution. The only restriction is the nestedness property.

2.2.1. Flagging for refinement. To decide which cells of a given grid have to be marked for refinement Berger [4] proposed to use an estimation of the local truncation error based on Richardson extrapolation, marking those cells for which the estimation of the local truncation error is above a prescribed tolerance. Despite Berger's approach being more accurate, we will use more pragmatic approaches as proposed by Quirk [7], which are actually cheaper to implement and seem to work properly.

Since the main three kind of singularities that appear in the solutions of hyperbolic systems of conservation laws constitute variations in the solution or in the gradient of the solution it could be enough to use the gradient of the solution as an indicator of the presence of such discontinuities. If the change in some seminorm of the gradient (e.g. the absolute value of the density component of the gradient for Euler equations or some norm of the gradient for general equations) of the solution between two adjacent cells is above a given tolerance R_{tol}, then both cells are flagged for refinement.

Once the coarse grid has been flagged we add a certain number of safety flags to ensure that the cells adjacent to a singularity are refined. The safety flags will avoid singularities to escape from the fine grid during one coarse time step.

2.2.2. The clustering process. Once the desired coarse cells have been flagged for refinement, the clustering process creates mesh patches at the next refinement level that contain every flagged cell and possibly some non-flagged cells. For each coarse mesh patch a simple and effective clustering procedure consists of the following steps. First the minimum Cartesian patch that contains all the flagged cells is found. If the ratio between the number of flagged cells and the total number of cells in the patch is above a prescribed tolerance C_{tol} then the patch is accepted and the process ends for the current patch. Otherwise the patch is sub-divided into smaller sub-patches and the same criterion is applied to each sub-patch. The procedure continues until no flagged cells remain un-flagged. The procedure is then applied to the next coarse patch, until every patch in the coarse grid had been clustered. A new fine grid is then generated using the accepted coarse patches. For each coarse patch we create a fine patch by the sub-division of each coarse cell into $(r_\ell)^d$ sub-cells.

For computational efficiency reasons we have implemented some procedures to avoid the formation of mesh patches that have a very high aspect ratio, or patches that are very small.

2.2.3. Transfer of solution. The new fine grid obtained by the clustering process needs to be filled with a numerical solution. To this aim the AMR algorithm uses the following steps: first, as the nestedness property ensures that the new grid is wholly contained in the coarser grid, a numerical solution is interpolated from the underlying coarser grid. Second, the numerical solution is copied from the grid of the same level that existed before the adaption process, in the regions in which both grids overlap, overwriting the interpolated solution. Finally, boundary conditions are applied wherever the patch boundary overlaps the domain boundary.

2.3. Flow integration and time refinement

One major advantage of the AMR algorithm is that each single mesh can be integrated in isolation. To achieve such an abstraction we have to supply some boundary information of the mesh. Since the boundary of an arbitrary patch does not necessarily coincide with the domain boundary, the boundary information can come from the underlying coarse grid, from another grid of the same level or from the domain boundary. A simple way to provide this boundary information is to augment each mesh patch with some dummy cells surrounding it, similarly to the usual approach used with a fixed grid covering the whole domain to implement the boundary conditions. These dummy cells are filled with data before the patch is integrated.

The AMR algorithm interleaves the integration of grids at different levels in a way such that each grid uses its own time step, according to its grid size. In one space dimension, if a grid G_ℓ, with grid size Δx_ℓ, uses a time step Δt_ℓ and its refinement factor is r_ℓ and we choose

$$\Delta t_{\ell+1} = \frac{\Delta t_\ell}{r_\ell} \tag{1}$$

the Courant number $\Delta t_\ell / \Delta x_\ell$ remains constant independently of ℓ. To evolve both grids from time t to time $t + \Delta t_\ell$ we have to perform one time step for the grid G_ℓ and r_ℓ time steps for the grid $G_{\ell+1}$. Similar procedures are used in more than one space dimension to ensure that the CFL condition is verified in each grid separately.

For a generic grid hierarchy the grids are integrated from coarse to fine, ensuring that each grid is always back in time with respect to coarser grids. Once a grid G_ℓ is integrated from time t to time $t + \Delta t_\ell$ it will not be integrated again until all the finer grids at levels $\ell + 1$ to $L - 1$ had been integrated until time $t + \Delta t_\ell$.

2.3.1. Treatment of patch boundaries. When a patch of a grid $G_{\ell+1}$ is integrated from time t to time $t + (\Delta t_\ell / r_\ell)$, its dummy cells have to be filled with data, but the grid G_ℓ is not available at time $t + (\Delta t_\ell / r_\ell)$. In this case linear interpolation in time is used to obtain a numerical solution at the intermediate times

$$t + \frac{\Delta t_\ell}{r_\ell}, t + \frac{2\Delta t_\ell}{r_\ell} \cdots, t + \frac{(r_\ell - 1)\Delta t_\ell}{r_\ell}$$

in which the grid $G_{\ell+1}$ is going to be integrated. The interpolation in time is performed using the known solutions in G_ℓ at times t and $t + \Delta t_\ell$. Then interpolation in space is applied to find approximated values to the numerical solution with which to prime the dummy cells.

2.3.2. Projection of solution. Once all grids have been evolved one coarse time step we have several numerical solutions corresponding to the grids at different resolution levels. As we shall use a conservative numerical method to integrate the equations it seems natural to enforce some kind of inter-grid conservation. Since changes in the conserved variables are due to the fluxes crossing the cell boundaries, a way to enforce inter-grid conservation is to ensure that the fluxes that cross a cell interface during one time step remain the same for all grids, following a purely finite-volume approach. The idea is to project to the coarse grid G_ℓ the arithmetic mean of the numerical fluxes computed in the r_ℓ integrations performed on the fine grid $G_{\ell+1}$, and then re-integrate the coarse grid using the projected numerical fluxes. This process amounts to a correction that can be applied to the updated coarse solution without actually re-integrating the coarse grid, see References [6, 7].

Our numerical tests show no inconsistencies due to the lack of inter-grid conservation if the above fix-up is not applied, but there is another reason for its application: the updated coarse solution with numerical fluxes coming from the finer grid provides more resolution of the shock waves and contact discontinuities. If some data is then interpolated from coarse to fine grids the results are more accurate if the conservative fix-up has been applied. This feedback mechanism provides a sharper coarse numerical solution which is visually distinguishable after some time steps from the coarse solution without the conservative fix-up. A sharper coarse solution needs less refinement, thus improving the performance of the algorithm.

2.4. The AMR algorithm

The main issue that has to be taken into account to adapt the grid hierarchy is to ensure that, if a discontinuity is covered by a grid G_ℓ it will always be covered by that grid, i.e. we have to adapt the grid G_ℓ before the discontinuity can escape. In Reference [7] Quirk shows that, for the linear advection equation, it is enough to adapt a grid G_ℓ after $r_{\ell-1}$ integrations (i.e. after an integration of the grid $G_{\ell-1}$) to prevent discontinuities to escape from the grid G_ℓ, provided the Courant numbers of each grid are given by (1), the typical CFL condition $(\Delta t_0/\Delta x_0)\leqslant 1$ is satisfied and at least one safety flag is added in the refinement process. This leads us to perform the adaption process of the grid G_ℓ after every $r_{\ell-1}$ integrations of that grid. The integration and adaption processes can be organized in a way such that the adaption process follows the integration process in the correct order, see Reference [7].

3. SHU–OSHER'S SCHEME WITH DONAT-MARQUINA'S FLUX SPLITTING

We briefly describe here the numerical method used to integrate the equations on each mesh patch, its building blocks being: Shu–Osher's flux formulation [8, 9], Donat–Marquina's flux-splitting formula [11], the WENO5 reconstruction procedure [10] and a third-order TVD Runge–Kutta ODE solver [9]. Our choice of Donat–Marquina's flux splitting, which is based on a double linearization, instead of other flux splittings such as Roe's, is motivated by the good results obtained in experiments where one linearization gives poorer results (shock reflections, carbuncle phenomena, etc.) [11, 14].

The hyperbolic system of conservation laws to be solved are:

$$\begin{aligned} U_t + \sum_{i=1}^{d} F_i(U)_{x_i} &= 0 \\ U(x,0) &= U_0(x) \end{aligned} \tag{2}$$

where $U = (U_1, \ldots, U_m)^{\mathrm{T}}, x = x_1, \ldots, x_d)$, $U_i: \mathbb{R}^d \longrightarrow \mathbb{R}$ and $F_i: \mathbb{R}^m \longrightarrow \mathbb{R}^m$, although, for the sake of clarity and given that schemes based on the Shu–Osher's formulation described below work in a dimension by dimension fashion, we will restrict this exposition to the one-dimensional case, i.e. $d = 1$ in (2). System (2) will be thus written as

$$\begin{aligned} U_t + F(U)_x &= 0 \\ U(x,0) &= U_0(x) \end{aligned} \tag{3}$$

We will denote by U_j^n the computed approximation to the exact solution $U(x_j, t^n)$ of (3), where $x_j = (j + \frac{1}{2})\Delta x, t^n = n\Delta t$. The vector $\{U_j^n\}$ is denoted by U^n.

We follow Shu–Osher's finite-difference flux formulation, which is better explained for a one-dimensional ($m = 1$) conservation law.

Let $\phi_{\Delta x}$ be the function that verifies

$$\frac{1}{\Delta x} \int_{x-\frac{\Delta x}{2}}^{x+\frac{\Delta x}{2}} \phi_{\Delta x}(s)\,\mathrm{d}s = F(U(x)) \tag{4}$$

then

$$F(U(x))_x = \frac{\phi_{\Delta x}(x + (\Delta x/2)) - \phi_{\Delta x}(x - (\Delta x/2))}{\Delta x} \tag{5}$$

and the conservation law $U_t + F(U(x))_x = 0$ is equivalent to

$$U_t + \frac{\phi_{\Delta x}(x + (\Delta x/2)) - \phi_{\Delta x}(x - (\Delta x/2))}{\Delta x} = 0 \tag{6}$$

Therefore, we can use the Euler scheme on this equation to obtain a conservative scheme with high-order space accuracy (the high-order time accuracy is obtained by the third-order TVD Runge–Kutta ODE solver of Reference [9]) if we accurately approximate $\phi_{\Delta x}(x_j + (\Delta x/2))$ by some numerical flux. This numerical flux may be obtained by the *reconstruction via primitive* approach, which essentially amounts to evaluating at the cell interface $x_{j+1/2}$ a high order essentially non-oscillatory interpolator (ENO, WENO, ...), based on the (known) cell averages $F(U(x_i))$ of the unknown $\phi_{\Delta x}$ in the cell $[x_i - (\Delta x/2), x_i + (\Delta x/2)]$.

Our extension to systems is achieved by the use of local characteristic variables and fluxes, following Donat–Marquina's flux-splitting formula, which is based on a double linearization of the Jacobian matrix $F'(U^n)$ at each cell interface $x_{j+1/2}$, avoiding the use of average matrices.

At a given cell interface $x_{j+1/2}$, we compute two sided interpolation $U_{j+1/2}^{L,R}$ of the conserved variables or the variables the eigenstructure of $F'(U)$ depends on. The values coming from the left, $U_{j+1/2}^L$, and from the right, $U_{j+1/2}^R$, are computed using high order essentially non-oscillatory interpolation procedures with upwind biased stencils that contain the points x_j and x_{j+1}, respectively. For this purpose, we have used a fourth-order WENO procedure based on

point values, similar in conception to the WENO5 reconstruction procedure of Jiang and Shu [10], originally developed by Liu *et al.* [15].

At each point x_k belonging to some upwind biased stencil that contains the given cell interface $x_{j+1/2}$, these interpolated quantities are used to define two sets of characteristic variables $w^{L,R}_{p,k} = \mathbf{l}_p(U^{L,R}_{j+1/2}) \cdot U_k$ and characteristic fluxes $F^{L,R}_{p,k} = \mathbf{l}_p(U^{L,R}_{j+1/2}) \cdot F(U_k)$, where $\mathbf{l}_p(U)$, $\mathbf{r}_p(U)$ stand for normalized left and right eigenvectors of the Jacobian matrix $F'(U)$ corresponding to the eigenvalue (characteristic speed) $\lambda_p(U)$. Upwind characteristic fluxes are then computed according to the characteristic speeds at both sides, except at sonic points, at which a local Lax-Friedrichs splitting (see Reference [9]) is applied.

Let $R(g_{-s_1}, \ldots, g_{s_2}, x)$ denote the evaluation at x of a reconstruction based on cell-averages g_j of a function at some $s_1 + s_2 + 1$ adjacent cells (we use here the WENO5 procedure [10]). The algorithm to compute the numerical fluxes at $x_{j+1/2}$ is as follows:

Algorithm 1

if $\lambda_p(U)$ does not change sign in a path in phase space connecting U_j and U_{j+1}

  if $\lambda_p(U_j) > 0$

    $\psi^L_{p,j} = R(F^L_{p,j-s_1}, \ldots F^L_{p,j+s_2}, x_{j+\frac{1}{2}})$

    $\psi^R_{p,j} = 0$

  else

    $\psi^L_{p,j} = 0$

    $\psi^R_{p,j} = R(F^R_{p,j-s_1+1}, \ldots F^R_{p,j+s_2+1}, x_{j+\frac{1}{2}})$

else

  $\alpha_{j+\frac{1}{2}} = \max(|\lambda_p(U_j)|, |\lambda_p(U_{j+1})|)$

  $\psi^L_{p,j} = R(\frac{1}{2}(F^L_{p,j-s_1} + \alpha_{j+\frac{1}{2}} w^L_{p,j-s_1}), \ldots \frac{1}{2}(F^L_{p,j+s_2} + \alpha_{j+\frac{1}{2}} w^L_{p,j+s_2}), x_{j+\frac{1}{2}})$

  $\psi^R_{p,j} = R(\frac{1}{2}(F^R_{p,j-s_1+1} - \alpha_{j+\frac{1}{2}} w^R_{p,j-s_1+1}), \ldots \frac{1}{2}(F^R_{p,j+s_2+1} - \alpha_{j+\frac{1}{2}} w^R_{p,j+s_2+1}), x_{j+\frac{1}{2}})$

With the numerical flux defined as

$$\hat{F}_{j+\frac{1}{2}} = \sum_p \psi^L_{p,j} \mathbf{r}_p(U^L_{j+\frac{1}{2}}) + \psi^R_{p,j} \mathbf{r}_p(U^R_{j+\frac{1}{2}}) \tag{7}$$

the system of ODEs

$$\partial_t U_j + \frac{\hat{F}_{j+\frac{1}{2}} - \hat{F}_{j-\frac{1}{2}}}{\Delta x} = 0 \tag{8}$$

is a highly accurate spatial semi-discretization of (3). A third-order TVD Runge–Kutta ODE solver [9] is applied to obtain third-order time accuracy as well.

For higher space dimensions, this scheme admits a straightforward tensorial (dimension by dimension) extension. A more detailed description of the overall algorithm can be found in Reference [12].

4. NUMERICAL RESULTS

In this section, we analyse the computational performance of the numerical scheme for a test problem consisting of a 1.22 Mach shock wave impinging a Helium bubble, studied by Haas and Sturtevant [13] and numerically simulated by Quirk and Karni [16]. The experiment has been successfully simulated by Marquina and Mulet [12] using the integration algorithm described here on a fixed grid.

An extension of the Euler equations for a fluid composed by the mixture of two perfect gasses in thermal equilibrium consists of adding a new equation which models the conservation of one of the gasses, which implies the conservation of both gasses due to the conservation of mass. We add a new variable ϕ which represents the mass fraction of one of the gasses. Consequently, the quantity $1-\phi$ represents the mass fraction of the other gas. The model can thus be expressed in terms of the conservation of total mass, momentum, total energy and mass of the first component.

4.1. One-dimensional experiment

The equation of conservation of the mass of the first gas is

$$(\rho\phi)_t + (\rho\phi u)_x = 0$$

The new system of equations extending the Euler equations of gas dynamics in one space dimension becomes:

$$\begin{pmatrix} \rho \\ \rho u \\ E \\ \rho\phi \end{pmatrix}_t + \begin{pmatrix} \rho u \\ \rho u^2 + p \\ u(E+p) \\ \rho\phi u \end{pmatrix}_x = 0 \tag{9}$$

where ρ represents the density of mass, u is the velocity, E is the total energy per unit volume and p is the pressure.

System (9) is hyperbolic. Its Jacobian matrix can be written as

$$\begin{pmatrix} 0 & 1 & 0 & 0 \\ \dfrac{\gamma-3}{2}u^2 - \phi\gamma'(\phi)\varepsilon & (3-\gamma)u & \gamma-1 & \gamma'(\phi)\varepsilon \\ \dfrac{\gamma-1}{2}u^3 - uH - u\phi\gamma'(\phi)\varepsilon & H-(\gamma-1)u^2 & \gamma u & u\gamma'(\phi)\varepsilon \\ -\phi u & \phi & 0 & u \end{pmatrix}$$

where ε is the specific internal energy, $\varepsilon = (E/\rho) - \frac{1}{2}u^2$ and H is the enthalpy, $H = [c^2/(\gamma(\phi)-1)] + \frac{1}{2}u^2$. The ratio of specific heats $\gamma = \gamma(\phi)$ depends on the composition of the mixture through the relation:

$$\gamma = \frac{C_{p_1}\phi + C_{p_2}(1-\phi)}{C_{v_1}\phi + C_{v_2}(1-\phi)} \tag{10}$$

where C_{p_i} (resp. C_{v_i}) are the specific heats at constant pressure (resp. volume) of fluid i. The pressure is related to the other variables through the equation:

$$p = (\gamma - 1)\rho\varepsilon \tag{11}$$

The eigenstructure of the Jacobian matrix depends only on three variables, the sound speed $c = \sqrt{\gamma p/\rho}$, the velocity u and the mass fraction ϕ.

Its eigenvalues are

$$\lambda_1 = u - c, \quad \lambda_2 = \lambda_3 = u, \quad \lambda_4 = u + c$$

the corresponding right eigenvectors are:

$$\mathbf{r}_1 = [1, u - c, H - uc, \phi]^{\mathrm{T}}$$

$$\mathbf{r}_2 = [1, u, \tfrac{1}{2}u^2, \phi]^{\mathrm{T}}$$

$$\mathbf{r}_3 = \left[0, 0, -\frac{\gamma'(\phi)c^2}{\gamma(\gamma-1)^2}, 1\right]^{\mathrm{T}}$$

$$\mathbf{r}_4 = [1, u + c, H + uc, \phi]^{\mathrm{T}}$$

and the (normalized) left eigenvectors are:

$$\mathbf{l}_1 = \left[\beta_2 + \frac{u}{2c} - \phi\beta_3, -\beta_1 u - \frac{1}{2c}, \beta_1, \beta_3\right]$$

$$\mathbf{l}_2 = [1 - 2\beta_2 + 2\phi\beta_3, 2\beta_1 u, -2\beta_1, -2\beta_3]$$

$$\mathbf{l}_3 = [-\phi, 0, 0, 1]$$

$$\mathbf{l}_4 = \left[\beta_2 - \frac{u}{2c} - \phi\beta_3, -\beta_1 u + \frac{1}{2c}, \beta_1, \beta_3\right]$$

where

$$\beta_1 = \frac{\gamma - 1}{2c^2}$$

$$\beta_2 = \beta_1 \frac{u^2}{2}$$

$$\beta_3 = \frac{\gamma'(\phi)}{2\gamma(\gamma - 1)}$$

The computational domain is the interval $[0, 0.356]$. The initial data represents a one-dimensional Helium "bubble", located in the interval $[0.15, 0.20]$ surrounded by air.

A left-travelling Mach 1.22 shock wave is located at $x = 0.225$:

$$U(x,0) = U_0(x) = \begin{cases} U_A & \text{if } 0 \leqslant x < 0.15 \text{ or } 0.2 < x < 0.225 \\ U_B & \text{if } 0.15 \leqslant x \leqslant 0.2 \\ U_S & \text{if } 0.225 \leqslant x \leqslant 0.356 \end{cases} \tag{12}$$

where

$$\begin{aligned} U_A &= (\rho_A, u_A, p_A, \phi_A) = (1, 0, 1, 1) \\ U_B &= (\rho_B, u_B, p_B, \phi_B) = (0.1819, 0, 1, 0) \\ U_S &= (\rho_S, u_S, p_S, \phi_S) = (1.3764, 0.3947, 1.5698, 1) \end{aligned} \tag{13}$$

Quantities U_A represents quiescent air, U_B quiescent Helium and U_S is the state connected with U_A by a Mach 1.22 shock travelling to the left. This is the same setup used in Reference [12] for the one-dimensional experiment.

The density profile of the numerical solution at time $t = 0.2$ obtained with the AMR algorithm with a grid hierarchy of four levels equivalent to fixed grids of $100, 200, 400$ and 800 points, respectively, is shown in Figure 1, compared with a reference solution computed with 8000 points. The diamonds represent the numerical solution obtained with the AMR algorithm, while the numerical solution obtained with a fixed grid of 800 points has been represented with dots. The solid line represents the reference solution. Figure 2 shows zoomed regions of the density profile of Figure 1. The parameters used in this simulation are $C_{\text{tol}} = 0.7$ and $R_{\text{tol}} = 0.5$. A CFL condition equal to 0.5 has been used.

The differences in density between the solution of the AMR algorithm and the solution obtained with an equivalent fixed grid are shown in Table I.

We observe that all the relevant features of the solution have been captured by the AMR algorithm, and that the percentage of integrations is closely related to the percentage of execution time. The big part of the computational time is used to integrate the equations, and

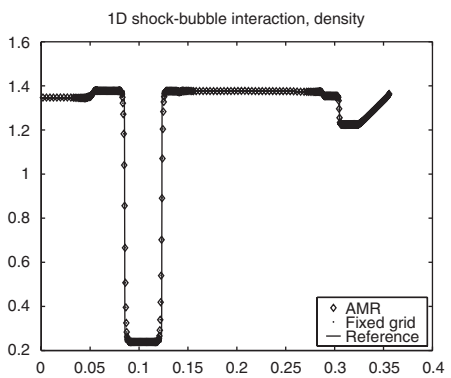

Figure 1. 1D shock–bubble, density.

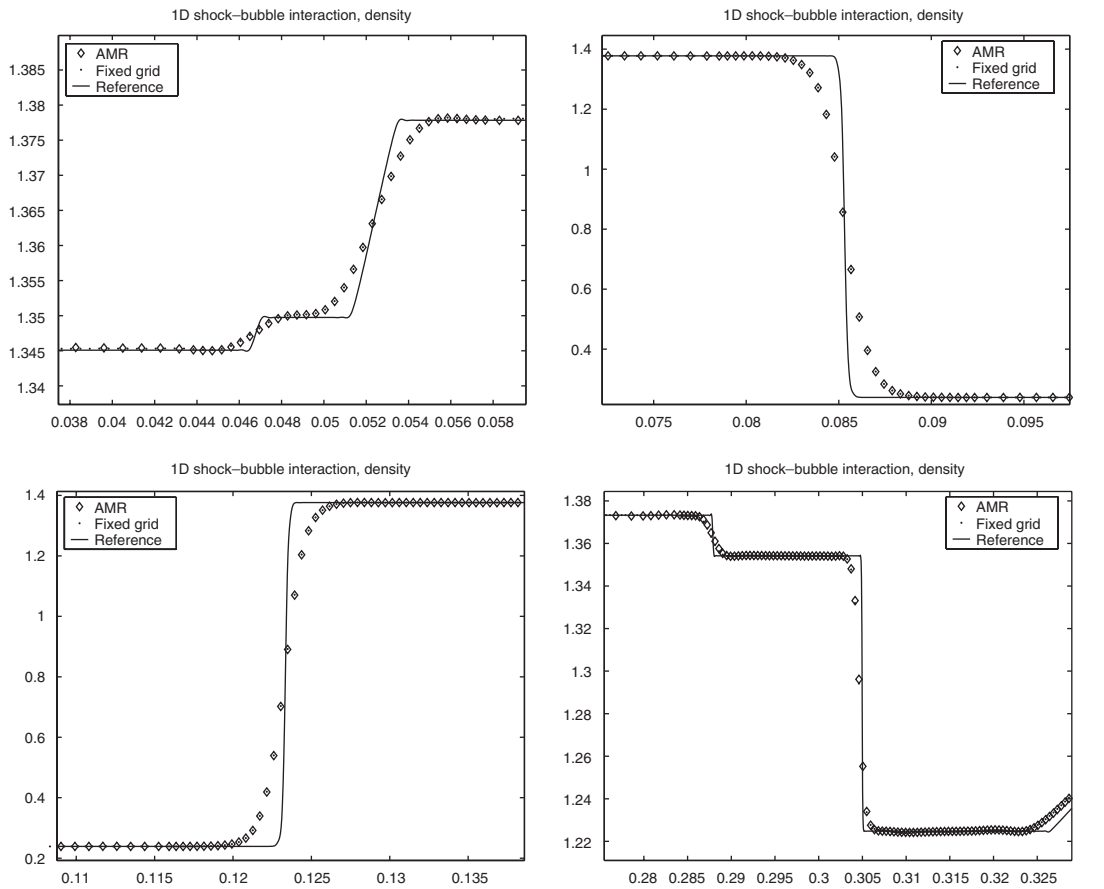

Figure 2. 1D shock–bubble interaction, zoomed regions of the density profile of Figure 1.

Table I. Comparison between the AMR algorithm and an equivalent fixed grid algorithm for the shock–bubble for the two-component 1D Euler equations (9) with initial data (12).

$\dim_0$	Levels	dim fix	$\|\rho_{\mathrm{FIX}} - \rho_{\mathrm{AMR}}\|_\infty$	% time	% integr.
100	4	800	$2.7275 \cdot 10^{-3}$	79.49	63.64
200	4	1600	$3.0694 \cdot 10^{-3}$	49.91	42.76
500	4	4000	$5.1692 \cdot 10^{-3}$	23.82	27.38
1000	4	8000	$3.6760 \cdot 10^{-3}$	24.67	20.54

The errors refer to the density field.

only a small percentage of the computational time is devoted to manage the grid hierarchy and to inter- and intra-grid communications, except when the number of integrations is very small.

Table II. Comparison of efficiency and quality of AMR approximations for different parameter sets.

R_{tol}	C_{tol}	% integrations	% time	$\|\rho_{\text{AMR}} - \rho_{\text{fix}}\|_1$	$\|\rho_{\text{AMR}} - \rho_{\text{fix}}\|_\infty$
1.0	0.7	18.53	19.21	3.31e − 04	1.08e − 02
1.0	0.8	17.10	17.74	3.31e − 04	1.08e − 02
1.0	0.9	15.22	16.99	3.18e − 04	5.32e − 03
4.0	0.7	11.77	12.18	3.64e − 04	1.44e − 02
4.0	0.8	11.17	11.69	3.65e − 04	1.46e − 02
4.0	0.9	10.65	11.38	3.55e − 04	1.48e − 02
8.0	0.7	11.29	11.63	4.71e − 04	2.85e − 02
8.0	0.8	10.19	10.75	4.72e − 04	2.88e − 02
8.0	0.9	10.03	10.51	4.55e − 04	2.29e − 02

The next experiment analyses the dependency of the approximation obtained by the method, and its efficiency, on the parameters C_{tol} and R_{tol}. A 100 coarse grid with four refinement levels (corresponding to a 800 fine grid), a Courant number of 0.45 and a final time 0.2 is used in the experiments, where R_{tol} vary in $\{1, 4, 8\}$ and C_{tol} vary in $\{0.7, 0.8, 0.9\}$. In Table II the results of these computations are shown. As expected, one can see that the percentage of CPU time with respect to an equivalent fix grid computation decreases when R_{tol} or C_{tol} increase. The density difference between the AMR computation and the fine grid computation (denoted in the table by $\rho_{\text{AMR}} - \rho_{\text{fix}}$), either measured in 1 or ∞ norm, does not vary significantly with the change in parameters, but a speedup of almost 2 can be achieved when appropriately choosing the tolerance parameters.

4.2. Two-dimensional experiment

In this section, we address the shock–bubble interaction problem in its two dimensional version. The governing equations for this problem are the Euler equations of gas dynamics augmented with an additional equation that models the conservation of one of the gasses:

$$(\rho\phi)_t + (\rho\phi u)_x + (\rho\phi v)_y = 0$$

The system can be written in the form $U_t + F(U)_x + G(U)_y = 0$:

$$\begin{pmatrix} \rho \\ \rho u \\ \rho v \\ E \\ \rho\phi \end{pmatrix}_t + \begin{pmatrix} \rho u \\ \rho u^2 + p \\ \rho uv \\ u(E+p) \\ \rho\phi u \end{pmatrix}_x + \begin{pmatrix} \rho v \\ \rho uv \\ \rho v^2 + p \\ v(E+p) \\ \rho\phi v \end{pmatrix}_y = 0 \tag{14}$$

where u and v are the velocity components of the fluid in the x and y directions, respectively, $E = (p/\gamma(\phi)) + \frac{1}{2}\rho(u^2 + v^2)$ is the total energy and the internal energy is given by the EOS (10) and (11).

The analysis made in the one-dimensional case for the Jacobian matrices can be repeated here for $F'(U)$ and $G'(U)$. We simply state here the eigenstructure of $F'(U)$. By interchanging u and v and the second and third components of each left and right eigenvector the eigenstructure of $G'(U)$ is obtained. See Reference [12] and references therein for further details.

The eigenvalues of $F'(U)$ are $\lambda_1 = u - c$, $\lambda_{2,3,4} = u$, $\lambda_5 = u + c$ and the corresponding right eigenvectors $\mathbf{r}_i$ and left eigenvectors $\mathbf{l}_i$, normalized so that $\mathbf{r}_i \cdot \mathbf{l}_j = \delta_{ij}$, are

$$
\begin{aligned}
\mathbf{r}_1 &= [1 \quad u-c \quad v \quad H-uc \quad \phi]^{\mathrm{T}} \\
\mathbf{r}_2 &= \left[1 \quad u \quad v \quad \frac{u^2+v^2}{2} \quad \phi\right]^{\mathrm{T}} \\
\mathbf{r}_3 &= [0 \quad 0 \quad 1 \quad v \quad 0]^{\mathrm{T}} \\
\mathbf{r}_4 &= \left[0 \quad 0 \quad 0 \quad -\frac{\gamma'(\phi)c^2}{\gamma(\gamma-1)^2} \quad 1\right]^{\mathrm{T}} \\
\mathbf{r}_5 &= [1 \quad u+c \quad v \quad H+uc \quad \phi]^{\mathrm{T}} \\
\mathbf{l}_1 &= \left[\beta_2 + \frac{u}{2c} - \phi\beta_3 \quad -\beta_1 u - \frac{1}{2c} \quad \beta_1 v \quad \beta_1 \quad \beta_3\right] \\
\mathbf{l}_2 &= [1 - 2\beta_2 + 2\phi\beta_3 \quad 2\beta_1 u \quad 2\beta_1 v \quad -2\beta_1 \quad 2\beta_3] \\
\mathbf{l}_3 &= [-v \quad 0 \quad 1 \quad 0 \quad 0] \\
\mathbf{l}_4 &= [-\phi \quad 0 \quad 0 \quad 0 \quad 1] \\
\mathbf{l}_5 &= \left[\beta_2 - \frac{u}{2c} - \phi\beta_3 \quad -\beta_1 u + \frac{1}{2c} \quad \beta_1 v \quad \beta_1 \quad \beta_3\right]
\end{aligned}
\tag{15}
$$

where

$$
\beta_1 = \frac{\gamma - 1}{2c^2}
$$

$$
\beta_2 = \beta_1 \frac{u^2 + v^2}{2}
$$

$$
\beta_3 = \frac{\gamma'(\phi)}{2\gamma(\gamma - 1)}
$$

and H is the enthalpy, $H = [c^2/(\gamma(\phi) - 1)] + \frac{1}{2}(u^2 + v^2)$.

The computational domain consists of the square $[0, 0.890] \times [0, 0.089]$. The Helium bubble is a circle with centre $(0.42, 0.0445)$ and radius $r = 0.025$. A vertical 1.22 Mach shock wave, initially located at $x = 0.6675$ is moving left through air, see Figure 3.

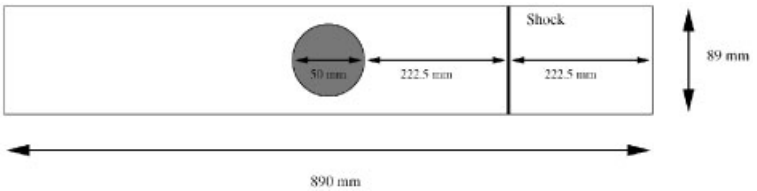

Figure 3. Computational domain of the two-dimensional experiment (not to scale).

The initial data are the following:

$$U(x,0)=U_0(x)=\begin{cases} U_A & \text{if } 0\leqslant x<0.6775 \text{ and } x\notin B \\ U_B & \text{if } x\in B \\ U_S & \text{if } 0.6675\leqslant x\leqslant 0.890 \end{cases}$$

where $B=\{(x,y)\in\mathbb{R}^2/(x-0.42)^2+(y-0.0445)^2\leqslant 0.025^2\}$ is the circle of centre $(0.42, 0.0445)$ and radius $r=0.05$ that represents the Helium bubble. The value of U_A, represents quiescent air, U_B represents Helium contaminated with a 28% of air, in equilibrium with the surrounding air, and U_S is the state connected with quiescent air by a vertical left moving 1.22 Mach shock wave. The respective values of U_A, U_B and U_S are:

$$U_A=(\rho_A,u_A,v_A,E_A,\rho\phi_A)=(1.225,0,0,253312.5,1.225)$$

$$U_B=(\rho_B,u_B,v_B,E_B,\rho\phi_B)=(0.2228,0,0,149007.35,0)$$

$$U_S=(\rho_A,u_S,v_S,E_S,\rho\phi_S)=(1.6861,-156.26,0,265521.1,1.6861)$$

We have used a coarse mesh of 100×5 cells to discretize the upper part of the computational domain. To obtain the lower part by symmetry we impose artificial reflecting boundary conditions, following the same approach of Quirk and Karni [16] and Marquina and Mulet [12]. Six levels of refinement with all refinement factors set to 2 have been used to obtain a resolution equivalent to a fixed grid of 3200×160 cells. In this experiment we have used the following parameters: the CFL condition has been set to 0.45, the refinement parameter is $R_{\text{tol}}=5.0$ and the clustering parameter is $C_{\text{tol}}=0.8$. In Figure 4 we display Schlieren images of the bubble at time $t=511\,\mu$s after the shock arrives at the Helium bubble, as computed with a fixed grid of 3200×160 cells and with the AMR algorithm. In this preliminary result we can see that the AMR algorithm has been able to resolve all the structure of the interface with the same quality as with the fixed grid algorithm. With this setup the AMR algorithm performs a 10.72% of integrations with respect to the fixed grid algorithm and requires a 11.72% of computational time.

In Figure 5 we overlay Schlieren images on the grid to compare the results of this simulation with two parameter sets: (a) $R_{\text{tol}}=3, C_{\text{tol}}=0.7$; (b) $R_{\text{tol}}=5, C_{\text{tol}}=0.9$. As can be observed in the graphics, the patches at the different levels of the grid hierarchy in case (a) are larger (but fewer in number) and cover more space than in case (b). The ratio of integrations with

Figure 4. Schlieren image at time $t = 511\,\mu s$ computed with the AMR algorithm (top) and with a fixed grid algorithm (bottom).

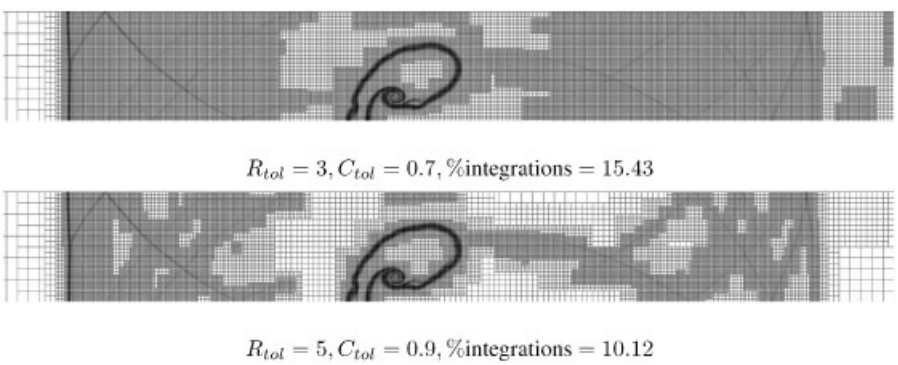

Figure 5. Comparison of AMR results for two different parameter sets, with only upper half part shown.

respect to the fixed grid algorithm is 15.43% in case (a) and 10.12% in case (b). Therefore, we recommend a clustering tolerance $C_{\text{tol}} \approx 0.8, 0.9$.

Unfortunately, we cannot give a general recommendation for the flagging tolerance R_{tol}, for it depends heavily on the problem: if this parameter is set too small, we will get lots of flagged cells, thus an inefficient execution, whereas if set too large, some regions where high resolution is needed would not be refined. We nevertheless feel that there is room for improvement if

we base, as intended in our future research, the flagging procedure on multiresolution analysis (see Reference [17] and references therein).

5. CONCLUSIONS

We have presented a numerical method for the solution of hyperbolic systems of conservation laws, obtained by the combination of a fifth-order high resolution shock capturing scheme, built from Shu–Osher's conservative formulation [8, 9], a fifth-order WENO interpolatory technique [10] and Donat–Marquina's flux-splitting method [11], with the adaptive mesh refinement technique of Berger *et al.* [4–6], in the simplified form proposed by Quirk [7]. The scheme inherits the robustness of Donat–Marquina's basic scheme and has shown to be able to resolve the structure of the numerical solution with an accuracy comparable to the computations made with fixed grids, with a significant reduction of the computational cost.

REFERENCES

1. Harten A, Hyman JM. Self-adjusting grid methods for one-dimensional hyperbolic conservation laws. *Journal of Computational Physics* 1983; **50**(2):235–269.
2. Osher S, Sanders R. Numerical approximations to nonlinear conservation laws with locally varying time and space grids. *Mathematics of Computation* 1983; **41**(164):321–336.
3. Tan Z, Zhang Z, Huang Y, Tang T. Moving mesh methods with locally varying time steps. *Journal of Computational Physics* 2004; **200**(1):347–367.
4. Berger MJ. Adaptive mesh refinement for hyperbolic partial differential equations. *Ph.D. Thesis*, Computer Science Department, Stanford University, 1982.
5. Berger MJ, Oliger J. Adaptive mesh refinement for hyperbolic partial differential equations. *Journal of Computational Physics* 1984; **53**:484–512.
6. Berger MJ, Colella P. Local adaptive mesh refinement for shock hydrodynamics. *Journal of Computational Physics* 1989; **82**:64–84.
7. Quirk JJ. An adaptive grid algorithm for computational shock hydrodynamics. *Ph.D. Thesis*, Cranfield Institute of Technology, 1991.
8. Shu C-W, Osher S. Efficient implementation of essentially non-oscillatory shock-capturing schemes. *Journal of Computational Physics* 1988; **77**(2):439–471.
9. Shu C-W, Osher S. Efficient implementation of essentially non-oscillatory shock-capturing schemes, II. *Journal of Computational Physics* 1989; **83**(1):32–78.
10. Jiang G-S, Shu C-W. Efficient implementation of weighted ENO schemes. *Journal of Computational Physics* 1996; **126**:202–228.
11. Donat R, Marquina A. Capturing shock reflections: an improved flux formula. *Journal of Computational Physics* 1996; **125**:42–58.
12. Marquina A, Mulet P. A flux-split algorithm applied to conservative models for multicomponent compressible flows. *Journal of Computational Physics* 2003; **185**(1):120–138.
13. Haas J-F, Sturtevant B. Interaction of weak shock waves with cylindrical and spherical inhomogeneities. *Journal of Fluid Mechanics* 1987; **181**:41–76.
14. Donat R, Font J, Ibáñez J-M, Marquina A. A flux-split algorithm applied to relativistic flows. *Journal of Computational Physics* 1998; **146**:58–81.
15. Liu X-D, Osher S, Chan T. Weighted essentially non-oscillatory schemes. *Journal of Computational Physics* 1994; **115**:200–212.
16. Quirk JJ, Karni S. On the dynamics of a shock–bubble interaction. *Journal of Fluid Mechanics* 1996; **318**: 129–163.
17. Chiavassa G, Donat R. Point value multiscale algorithms for 2D compressible flows. *SIAM Journal on Scientific Computing* 2001; **23**:805–823 (electronic).

