Commun. Comput. Phys.
doi: 10.4208/cicp.101009.120210a

Vol. **9**, No. 1, pp. 68-88
January 2011

Quadrilateral Cell-Based Anisotropic Adaptive Solution for the Euler Equations

H. W. Zheng[1], N. Qin[1,*], F. C. G. A. Nicolleau[1] and C. Shu[2]

[1] *Department of Mechanical Engineering, University of Sheffield, Mappin Street, Sheffield S1 3JD, UK.*
[2] *Department of Mechanical Engineering, National University of Singapore, 10 Kent Ridge Crescent, Singapore 119260.*

Received 10 October 2009; Accepted (in revised version) 12 February 2010

Available online 5 August 2010

Abstract. An anisotropic solution adaptive method based on unstructured quadrilateral meshes for inviscid compressible flows is proposed. The data structure, the directional refinement and coarsening, including the method for initializing the refined new cells, for the anisotropic adaptive method are described. It provides efficient high resolution of flow features, which are aligned with the original quadrilateral mesh structures. Five different cases are provided to show that it could be used to resolve the anisotropic flow features and be applied to model the complex geometry as well as to keep a relative high order of accuracy on an efficient anisotropic mesh.

AMS subject classifications: 51E12, 65M08, 68U05, 68U20

Key words: Anisotropic adaptive mesh refinement, quadrilateral mesh, finite volume, face-based algorithm, HLLC, Euler equation.

1 Introduction

Fluid phenomena contain many different length scales and flow features, such as flow separation, shock waves, interfaces between different fluids and viscous boundary layers. To resolve all these scales and features, the mesh adaption is a natural and economical way for accurate and efficient solutions. As a result, a large number of adaptive algorithms have been proposed in the literature.

Among them, the Cartesian adaptive method [1–14] is a quite popular method in which the nested hierarchical data structure is often employed. The main feature of this kind of methods is that the global index of a cell could be calculated by the level and the

*Corresponding author. *Email addresses:* `H.W.Zheng@sheffield.ac.uk` (H. W. Zheng), `n.qin@sheffield.ac.uk` (N. Qin), `F.Nicolleau@Sheffield.ac.uk` (F. C. G. A. Nicolleau), `mpeshuc@nus.edu.sg` (C. Shu) local index. This reduces the memory requirement. Most of them are isotropic Cartesian method which uses the tree data structures such as the quad-tree and the oct-tree. These isotropic Cartesian grid methods could not be efficient for the flow problems where highly anisotropic meshes aligned with the flow direction are most appropriate. Hence, several anisotropic Cartesian adaptive grid methods [11–14] are developed. Wang et al. [11, 12] proposed an anisotropic grid generation by adopting a 2^N tree data structure, which is a natural extension of the quad-tree or oct-tree. Ham et al. [13] proposed another anisotropic adaptive method. Like the isotropic Cartesian grid methods, the global index of a cell for this anisotropic Cartesian method [13] could be obtained by the level in each direction and the local index.

One of the main drawbacks of all these adaptive Cartesian grid methods is their difficulty when they are applied to complex geometry with non-Cartesian boundaries. To tackle this problem, some researchers [9] adopted the cut-cell or hybrid grid technique. These will increase the complexity and reduce the main advantage of the Cartesian method.

Another type of anisotropic adaptive method to treat the complex geometry is the unstructured mesh based adaptive methods [15–18]. Most of them are the triangular or tetrahedral cell based adaptive method. The adaptive grid based on triangular mesh (or tetra-hedras in 3D) is often results in highly skewed cells for anisotropic flow features such as boundary layers, which makes it difficult to maintain high accuracy. Besides, successive refinements in one direction of the triangle based adaptive method create triangle cell with some very small angles between faces. This resulted in large mesh skewness which can cause severe degradation in both solution accuracy and convergence.

In order to address the issues of mesh efficiency and solution convergence for adaptive solutions, Qin and Liu [19] proposed a feature-aligned mesh adaptation method, which combined the flexibility of unstructured meshes for complicated geometries and structured feature aligned mesh blocks to achieve efficient and accurate mesh adaptation. The refinement only needs to be carried out normal to the flow features, leaving relatively large mesh spacing along the features, such as shock waves and boundary layers. The overall solution efficiency is substantially improved by (1) the flow feature aligned anisotropic adaptive mesh and (2) the better behavior of the solver's convergence on such meshes. However, the extraction of the flow features and the insertion of the structured blocks for complicated flow problems are key issues to be fully addressed and automated.

As a step towards a fully automated and robust feature aligned mesh adaptation, this paper proposes an unstructured quadrilateral-cell based anisotropic mesh adaptive method to resolve the anisotropic flow features, given a background initial body conforming mesh. As compared to the Cartesian anisotropic adaptive method [11–14], the fully unstructured quadrilateral-cell based anisotropic adaptive meshes are body fitted rather than the cut-cell method or the hybrid method [9]. Similar to the isotropic unstructured quadrilateral based adaption proposed by Sun and Takayama [17] and Zheng et al. [18], the neighboring cells of each edge and the sub-edges of each edge are stored. However, the neighboring cells for the mother edge may not include the cell that the edge belongs to. This requires some different special treatments to be presented in the paper. Besides, it also adopts the idea of two different levels for each direction of the Cartesian adaptive method [13, 14] rather than one unique level number of most anisotropic adaptive methods. To directionally refine or coarse the target cell properly, the directional refinement criteria is also proposed.

The paper is organized as follows. The data structure, the directional refinement and coarsening, including how to initialize the solution on the new refined cells, for the anisotropic adaptive method, are proposed in the second section. The numerical method for compressible flows based on this anisotropic adaptive mesh is given in the third section. Five validation cases are provided in the fourth section, in order to demonstrate the behavior of the proposed method for a variety of transient compressible flows involving typical anisotropic flow features.

2 Anisotropic adaptive method on quadrilateral mesh

In this section, the anisotropic adaptive grid generator based on the unstructured quadrilateral cell mesh technique is addressed. In the following, the data structure, error detector, the directional refinement and directional coarsening are described.

2.1 Data structure

As indicated in the previous section, the present adaption algorithm is based on the quadrilateral cells instead of other types of cells such as the triangle cells or more general polygon cells. However, the mesh is not necessarily structured as the unstructured quadrilateral meshes offer geometrical flexibility yet to maintain the accuracy normally associated with quadrilateral cells. Therefore, an unstructured data management is required. As described by Zheng et al. [18], each cell contains an array of pointers to its nodes and edges and each edge stores an array of pointers to its nodes and sub-edges.

For the non-directional adaption [17], there is only one level number that is independent of the direction (Fig. 1(a)). Since the level number between the target cell and the neighboring cells cannot be larger than one [17], the cells in block (Fig. 1(a)) cannot be further refined if the neighboring cells is not refined in advance. Unlike the non-directional adaptation [17], there are two different level numbers for the two directions of a cell in the (anisotropic) directional adaption (Fig. 1(b)). Thus, the cell in block can be further refined in vertical direction (Fig. 1(b)). This shows the necessarity of the (anisotropic) directional adaption. Here, these two directions are denoted as 0 and 1 respectively. The level numbers in each direction are stored in two indices separately, namely, l_0 and l_1. Besides, as compared to the isotropic or anisotropic Cartesian adaption methods [13, 14] which usually employ tree data structure, the neighboring indices of the cells for anisotropic unstructured mesh adaptive solver could not be evaluated in a general formula. Instead, these indices are obtained by accessing its four edges which store the indices or pointers to the two neighboring cells. In order to avoid the recursive calculation, the indices of

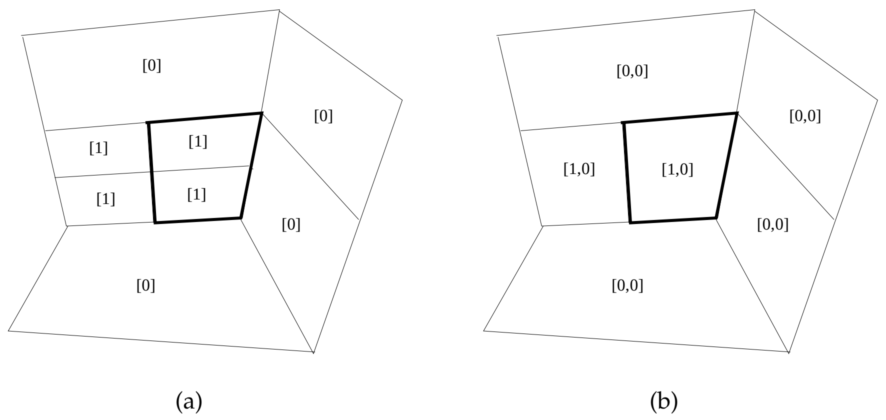


Figure 1: The example that the non-directional (a) and directional (b) adaption (the number in the square bracket is the level number).

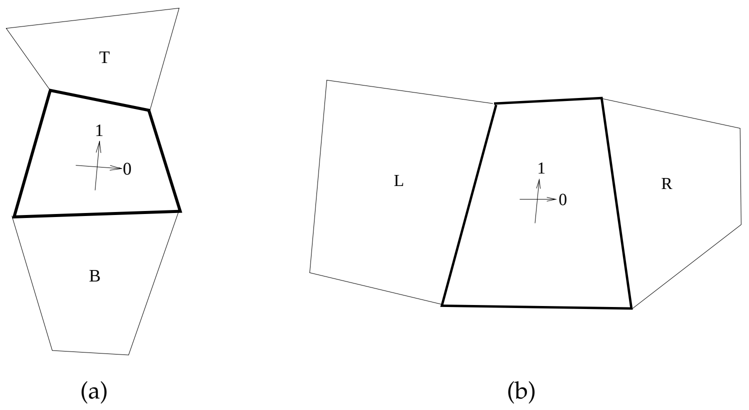


Figure 2: Stencils in the direction 0 (a) and 1 (b) for the refinement and coarsening criteria.

each cell in the present method are further separately stored in different lists according to their levels (Fig. 3(a)). Thus, the solution can be obtained in a level by level manner so that the cells of high level are calculated after the cells of low level. This makes the algorithm very efficient. Moreover, for those methods [17] where the information of leaf cells and non-leaf cells are stored in two different lists, the cells need to frequently move from one list to the other during the process of refinement and coarsening. In contrast, the current method does not need to do so. This shows that the present method also improves the efficiency especially when the mesh on the finest resolution level is large. In order to reduce the memory, only one directional edge is stored since the two opposite edges of the cell have the same direction. Hence, the two neighboring cells can share the same directional edge. Note that, unlike the isotropic adaption in [18], the neighboring cells for each edge may not be the cell itself for anisotropic adaption. As stated above,the indices

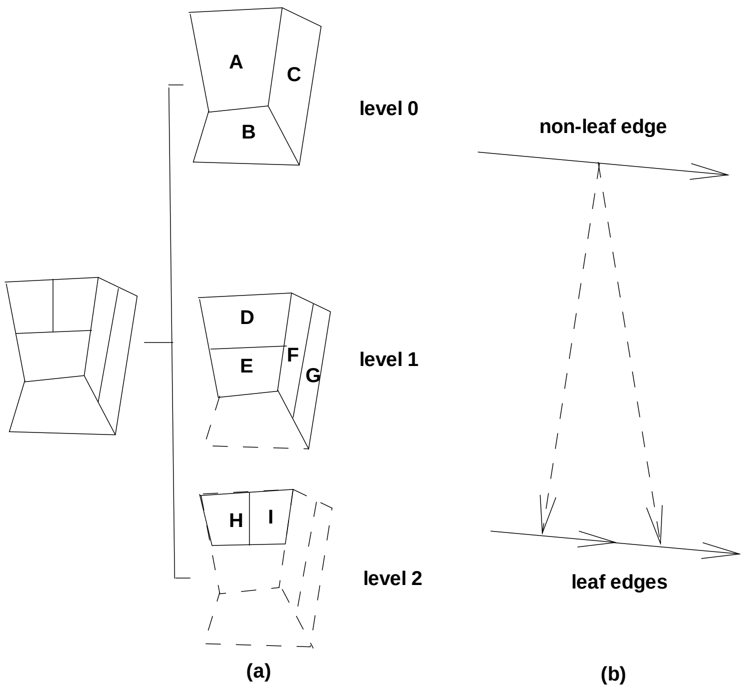


Figure 3: Arrangements of cells and edges.

of the two sub-edges and the indices of the two nodes for each edge are also stored. Thus, adding or deleting a node is made more efficient as the process does not need to access the neighboring cells.

Similar to Zheng et al. [18], in order to develop an face-based solution adaptive solver, the edges are further organized into two separated lists which are used to store the leaf edges and mother edges respectively (Fig. 3(b)). Therefore, the numerical flux calculation can be applied easily to the leaf edges. The numerical fluxes of the mother edges are not calculated. Besides, these numerical fluxes of the leaf edges are not stored and only used to update the residues of the left and right cells. As compared to the cell-based methods [17] which store the fluxes of the edges, the present method stores the residues at the cells, reducing the memory requirement.

2.2 Error detector for refinement and coarsening

To automatically perform the grid adaption, a refinement criterion is needed. In the literature, there are a number of criteria for anisotropic adaption [7,9,13,14]. For example, Keats and Lien [14] devised the following error detector,

$$\Delta x^* = \left[\frac{2\tau(\phi_y+\alpha)}{(\phi_x+\alpha)^2}\right]^{1/3}, \quad \Delta y^* = \left[\frac{2\tau(\phi_x+\alpha)}{(\phi_y+\alpha)^2}\right]^{1/3}. \tag{2.1}$$

However, it is only suitable to Cartesian mesh. In this work, similar to the isotropic criteria [17, 18], the error indicator of a cell is the maximum of the second-order Taylor truncation error over the first-order Taylor truncation error of density in the stencil (Fig. 3) of the desired direction. The difference from it is that the present criteria are direction related for the anisotropic adaptation. Here, we use L and R to denote the two stencils in the direction 0. Similarly, B and T refer to the two stencils in the direction 1. For example, the error detector for a cell is,

$$\varepsilon_0 = \max\left(\frac{\left|(\nabla_{l_n}\rho)_C - (\nabla_{l_n}\rho)_L\right|}{\alpha\rho_C/dl + \left|(\nabla_{l_n}\rho)_L\right|}, \frac{\left|(\nabla_{l_n}\rho)_C - (\nabla_{l_n}\rho)_R\right|}{\alpha\rho_C/dl + \left|(\nabla_{l_n}\rho)_R\right|}\right) \tag{2.2}$$

in the direction 0, and

$$\varepsilon_1 = \max\left(\frac{\left|(\nabla_{l_n}\rho)_C - (\nabla_{l_n}\rho)_T\right|}{\alpha\rho_C/dl + \left|(\nabla_{l_n}\rho)_T\right|}, \frac{\left|(\nabla_{l_n}\rho)_C - (\nabla_{l_n}\rho)_B\right|}{\alpha\rho_C/dl + \left|(\nabla_{l_n}\rho)_B\right|}\right) \tag{2.3}$$

in the direction 1. Here, dl is the distance of the two stencil cell centroids.

By calculating these error detectors of a cell, the refinement of a cell could be determined by comparing the detector in each direction with the pre-defined refining threshold ε_{refine}. In this work, the refinement is done in the following way,

(1) refine the cell in the direction 0 when $\varepsilon_0 < \varepsilon_{refine}$ and $\varepsilon_1 > \varepsilon_{refine}$ are satisfied;

(2) refine the cell in the direction 1 when $\varepsilon_0 > \varepsilon_{refine}$ and $\varepsilon_1 < \varepsilon_{refine}$ are satisfied;

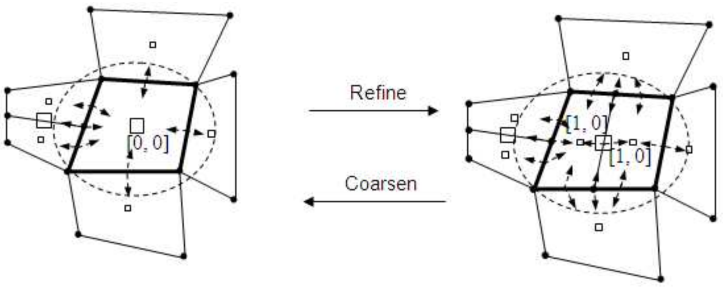
(3) refine the cell in both directions when $\varepsilon_0 < \varepsilon_{refine}$ and $\varepsilon_1 < \varepsilon_{refine}$ are satisfied.

The directional coarsening of a cell is carried out in a similar way. The refinement and coarsening thresholds are user defined and a compromise between the accuracy and memory and efficiency. In this paper, they are set as two constants, 0.06 and 0.0598 respectively.

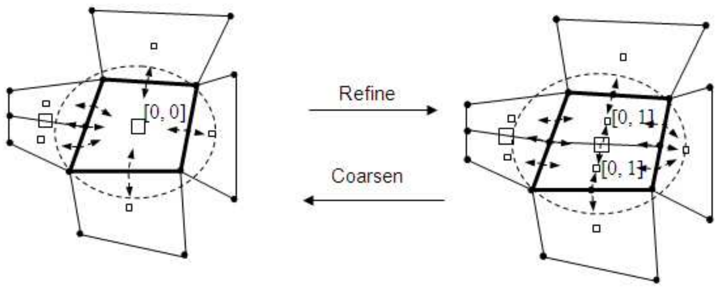
2.3 Directional refinement and coarsening

In the directional refinement, each cell is divided into two new quadrilateral sub-cells (Fig. 3). There are two constraints for this process. One is that the cell has no children (sub-cells) and the other is the one level difference rule in the stencil cells for the directional refinement in a direction. That is, the maximum difference of the levels between the current cell and the stencil cells in that direction must be less than two. This is different from the isotropic adaptive Cartesian methods where the adaptive process is independent of the direction. For example, as shown in Fig. 1, the level difference in the direction 1 for the target cell (denoted as block in the figure) is 0. However, the level difference in the other direction is 1. Thus, it could only be refined in the direction 1 but not in the direction 0.

If these two constraints in a direction are satisfied, the refinement of this cell in that direction could be performed. One needs to pay attention to set the new level number



(a)



(b)

Figure 4: Example of refinement and coarsening procedure in the direction 0 (a) and 1 (b).

in each direction. If a cell with the level $[l_0,l_1]_c$ is refined in the direction 0, the level number of the two sub-cells are set as $(l_0)_{c\to sub}=(l_0)_c+1,(l_1)_{c\to sub}=(l_1)_c$. Similarly, the level numbers of the two sub-cells are set as $(l_0)_{c\to sub}=(l_0)_c,(l_1)_{c\to sub}=(l_1)_c+1$ if the refinement is performed in the direction 1. Unlike the anisotropic Cartesian method in Keats and Lien [14] where the refined cell is modified to be the one of the new cells, the refined cell is still preserved without any change. Thus, there is no brick problem as shown in their paper. Besides, the addition of edges and nodes needs to be carefully checked to ensure that no redundant nodes and edges are created (Fig. 4). If one edge has no sub-edges, a new node is created. Otherwise, the existing center node of edge is used. Similarly, the two sub-edges of the desired pair of directional edges are created if the edge has no sub-edges. In addition, the neighboring cells of each edge should be re-assigned. Note that, unlike the isotropic adaption in Zheng et al. [18], the neighboring cells for each edge may not include the cell which the edge belongs to for the anisotropic adaption. For example, as shown in Fig. 4 (a), the left neighbor cell of the right edge of the target cell is pointed to the right child cell of the cell after the refinement in the direction 0.

As a reverse process of the refinement, the coarsening process removes the unnecessary cells from the mesh. Similarly, there are two constraints for this. The first one is that the cell must have children cells (sub-cells). The second is that the maximum one level difference between the current cell and the stencil cells is allowed. If all these constraints are satisfied, the cell is removed by calling the coarsening procedure. In this coarsening process, one also needs to change the neighboring cell information of the cell. This is simply implemented by reassigning the two pointers (indices) of the neighboring cells of the four edges of the cell.

3 Numerical method

Having the anisotropic adaptive mesh, the numerical method to solving the governing equations needs to be coupled with this adaptive mesh. In this paper, the inviscid compressible flows are the main interest. Thus, the time-dependent Euler equations are solved,

$$\partial_t U + \nabla \cdot F = 0, \tag{3.1}$$

where U is the state vector, and F is flux vector,

$$U = \begin{pmatrix} \rho \\ \rho\vec{u} \\ E \end{pmatrix}, \quad F = \begin{pmatrix} \rho\vec{u} \\ \rho\vec{u}\otimes\vec{u} + P[I] \\ (E+P)\vec{u} \end{pmatrix}. \tag{3.2}$$

Here, ρ is the density, $\vec{u}$ is the velocity, P is pressure, $[I]$ is the identity tensor, E is the total energy,

$$E = \rho e + \frac{1}{2}\rho\vec{u}^2. \tag{3.3}$$

3.1 Discretization on anisotropic adaptive mesh

After the refinement and coarsening of the cells, the numerical discretization of Eq. (3.1) is performed on the quadrilateral-cell based anisotropic adaptive mesh. Since the cells are separately stored in different lists according to their levels (Fig. 5 (a)), the solution can be obtained in a level by level manner so that the cells of high level are calculated after the cells of the low level. The set of equation (3.1) is only discretized at each leaf cell c by the multiple stage Runge-Kutta schemes,

$$U_c^{(0)} = U_c^n, \tag{3.4}$$

$$U_c^{(m)} = \alpha_{m-1}U_c^{(0)} + \beta_{m-1}U_c^{(m-1)} + \gamma_{m-1}\frac{\Delta t}{V_i}\mathrm{Res}\left(U_c^{(m-1)}\right), \quad m = 1,\cdots,k, \tag{3.5}$$

$$U_c^{n+1} = U_c^{(k)}, \tag{3.6}$$

where

$$\mathrm{Res}_c = \sum_f \Phi_f\left(U^-, U^+, \vec{n}\right)\cdot\Delta l_f. \tag{3.7}$$

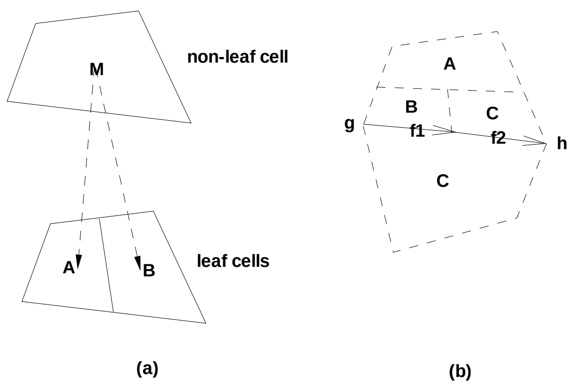


Figure 5: Evolution in a cell (a) and calculation of flux of edges (b).

Here, α_{m-1}, β_{m-1} and γ_{m-1} are the coefficients, Δl_f denotes the length of the edge f and $\Phi_f(U^-,U^+,\vec{n})$ is the numerical flux for an edge f.

For the non-leaf cells, the states are calculated by a simple summation of the states of their sub-cells,

$$U_c^{(0)} = \sum_i U_{c\to sub_i}. \tag{3.8}$$

Since the edges are separately stored into two lists for the leaf edges and non-leaf edges respectively (Fig. 3(b)), the residual could be calculated in an face-based manner. Thus, in the real calculation, Eq. (3.7) is replaced by updating the residue of all leaf edges only. That is, the residue ($Res_{f1\to L}$) at the left cell centroid of a leaf edge $f1$ (the left cell is denoted as $f1\to L$ or cell B in Fig. 5) is updated in the following way,

$$\text{Res}_{f1\to L} = \text{Res}_{f1\to L} + \Phi_{f1}\left(U^-,U^+,\vec{n}_{f1}\right)\cdot \Delta l_{f1}, \tag{3.9}$$

and the residual ($Res_{f1\to R}$) at the right cell centroid ($f1\to R$ or cell C in Fig. 5) of this leaf edge $f1$ is updated by

$$\text{Res}_{f1\to R} = \text{Res}_{f1\to R} - \Phi_{f1}\left(U^-,U^+,\vec{n}_{f1}\right)\cdot \Delta l_{f1}, \tag{3.10}$$

where Δl_{f1} denotes the length of the edge $f1$.

Note that the numerical flux Φ_f is calculated once for each non-leaf edge which has no sub-edges before the updating of the residual (Eqs. (3.7), (3.9), and (3.10)). There is no need to calculate the numerical flux for a non-leaf edge. For example, the right neighboring cell of non-leaf edge $\overrightarrow{gh}$ is the same as the right neighboring cell of leaf edges $f1$ and $f2$, that is, $f1\to R = f2\to R = \overrightarrow{gh}\to R = C$. Thus, the residual at the right leaf cell centroid of this non-leaf edge $\overrightarrow{gh}$ is automatically updated by the operations (Eqs. (3.9) and (3.10)) of its two sub-edges $f1$ and $f2$. This reduces the calculation of the numerical flux for the non-leaf edges as well as keeps the conservation of the numerical flux.

It is clear that the calculation of the numerical flux of an edge only requires the left state U^- and right state U^+ of the edge. To increase the solution accuracy in space, the piecewise linear reconstruction is employed to calculate these two states. In order to obtain a monotonic and stable solution, the reconstruction is based on primitive variables $W=(\rho,\, \vec{u},\, p)^T$ and slope limiters are enforced in the reconstruction,

$$W^- = W_L + \frac{1}{2}\psi_\beta\left(2\vec{\nabla}W_L\cdot d\vec{r} - \Delta W, \Delta W\right), \tag{3.11}$$

$$W^+ = W_R - \frac{1}{2}\psi_\beta\left(\Delta W, 2\vec{\nabla}W_R\cdot d\vec{r} - \Delta W\right), \tag{3.12}$$

$$\Delta W = W_R - W_L, \tag{3.13}$$

where $d\vec{r}$ is the distance vector between the two neighboring cell centroids of the edge, $\psi_\beta(a,b)$ is a superbee-like limiter,

$$\psi_\beta(a,b) = \frac{sign(a)+sign(b)}{2}\max[\min(\beta|a|,|b|),\min(|a|,\beta|b|)], \tag{3.14}$$

where β is a parameter which is in the range of [1,2]. It could be easily observed that it will be reduced to minmod limiter if β is set as one.

3.2 Flux calculation

In this work, the Harten, Lax and van Leer approximate Riemann solver (with the contact wave restored) (HLLC) scheme [20, 21] is adopted. Hence, the numerical flux in Eqs. (3.7), (3.9), (3.10) is calculated by,

$$\Phi_f\left(U^-, U^+, \vec{n}_f\right) = \frac{1}{2}\left[\phi^- + \phi^+ - sign(k_m)\left(U^+ - U^-\right)\right], \tag{3.15}$$

with

$$\phi^j = F\left(U^j\right)\cdot\vec{n}_f + k_j^\#\left(U^{j^*} - U^j\right), \quad j=-,+, \tag{3.16}$$

where the intermediate velocity U^{j^*} is calculated by,

$$U^{*j} = \frac{k_j - u_n^j}{k_j - s_m}\begin{pmatrix} \rho^j \\ \rho^j\left[\vec{u}^j - \left(u_n^j - s_m\right)\vec{n}\right] \\ E^j \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ \left(P^{*j}s_m - P^j u_n^j\right)/\left(k_j - u_n^j\right) \end{pmatrix}, \quad j=-,+, \tag{3.17}$$

with

$$P^{*j} = \rho^j\left(u_n^j - k_j\right)\left(u_n^j - k_m\right) + P^j, \quad j=-,+. \tag{3.18}$$

Here, u_n^j is the normal velocity ($u_n^j = \vec{u}^j \cdot \vec{n}$), $k_j^\#$ and k_m are the intermediate signal speeds which are calculated by,

$$k_-^\# = \min(k_-, 0), \quad k_+^\# = \max(k_+, 0) \tag{3.19}$$

and

$$k_m = \frac{\rho^+ u_n^+ (k_+ - u_n^+) - \rho^- u_n^- (k_- - u_n^-) + P^- - P^+}{\rho^+ (k_+ - u_n^+) - \rho^- (k_- - u_n^-)}, \tag{3.20}$$

with

$$k_j = \min\left(u_n^j - s^j, \tilde{q} - \tilde{s}\right), \quad j = -, +. \tag{3.21}$$

Here, s^j is the sound speed. $\tilde{q}$ and $\tilde{s}$ are the Roe average normal velocity and the sound speed.

3.3 Initialization of solution for a newly refined cell

During the refinement, new cells are generated and deleted continually as the solution evolves. It is very critical to give an accurate value of dependent variables at the newly-generated cells, especially for the time-dependent compressible flow problems. Similar to the reconstruction step, the slope limiters are used to obtain a smooth initial value. For example, the initial state of a new cell of the mother cell c is evaluated as,

$$U_{c \to sub_i} = U_c + \psi \vec{\nabla} U_c \cdot d\vec{r}_{c, c \to sub_i},$$

where $d\vec{r}_{c,c \to sub_i}$ is the distance vector between the new cell centroid and the mother cell centroid and ψ is a limiter. It is clear that this initialization also satisfies the conservation law in the mother cell.

4 Results and discussion

To demonstrate the features of the present anisotropic solver, five cases are tested in this section. They are the shock tube problem, the circular shock tube problem, the shock diffraction over back-step, the shock wave reflection over wedges, and the two dimensional Riemann problem.

4.1 Sod's shock tube problem

In this section, the Sod's shock-tube problem is used to demonstrate the anisotropic property. It is a standard case that two gases are separated by a diaphragm at the center position in a tube with the initial values (Luo et al. [22])

$$\rho_1 = 1 kg/m^3, \quad u_1 = 0.125 m/s, \quad v_1 = 0 m/s, \quad p_1 = 1 Pa, \quad \text{for } x < 0.5, \tag{4.1}$$

$$\rho_2 = 0.125 kg/m^3, \quad u_2 = 0 m/s, \quad v_2 = 0 m/s, \quad p_2 = 0.1 Pa, \quad \text{else}. \tag{4.2}$$

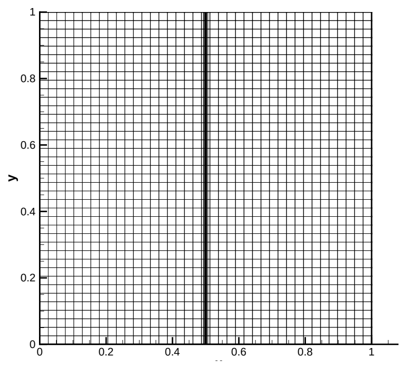
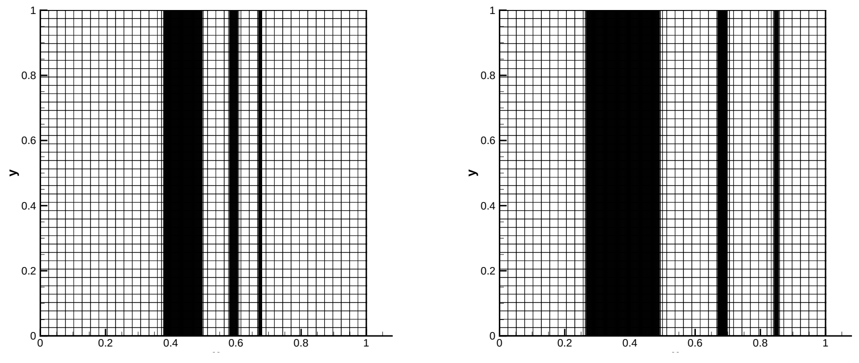


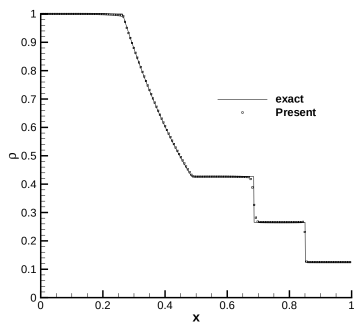
Figure 6: Initial mesh for shock tube problem.



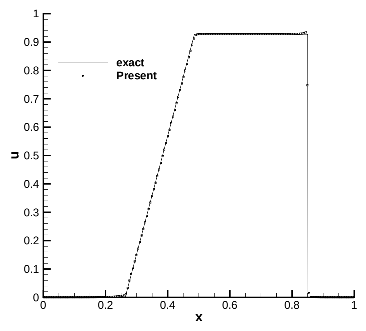
(a) (b)

Figure 7: mesh at $t=0.1$s (a) and mesh at $t=0.2$s (b).

For this problem, it is known that the exact solution consists of a right-going shock wave, a left-going rarefaction and a contact discontinuity in between. The initial mesh is adaptively generated by a uniform background mesh of 40×40 (level 0) with the maximum level of 4 (Fig. 6). The mesh at time 0.1s and 0.2s are shown in Fig. 7. From the observation of the mesh distributions (Fig. 7(b)), the shock wave and contact discontinuity appear to be very well captured. This shows that the present solver can capture the unsteady anisotropic flow feature, although this is actually a one-dimensional problem. From Fig. 7, it is clear that both the area of the shock and contact discontinuity is refined only normal to the waves (flow features in this problem) and the other smooth area is coarsened. As can be seen from Fig. 7, as the time evolves, the meshes change with the solution so that the mesh around the discontinuity is refined and the other smooth area is recovered to the background mesh. This relatively simple case shows that the present anisotropic solution-adaptive algorithm can reflect the important features of the unsteady flow. Furthermore, the density, u velocity and pressure cross-center profiles along the center line of y are plotted in Figs. 8-9. Good agreement of the results as compared to the exact solutions, showing the accuracy of the adaptive method.



(a)



(b)

Figure 8: Density (a) and u-velocity (b) cross-section profile along center line of y.

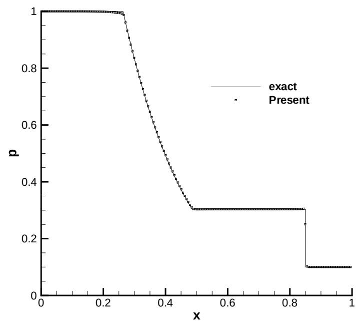


Figure 9: Pressure cross-section profile along center line of y.

4.2 Circular shock tube problem

To further demonstrate the advantage of the present anisotropic solver, the two-dimensional circular shock-tube problem is tested. Initially, the two ideal gases with the same initial conditions (Eqs. (4.1)-(4.2)) as those in the previous shock tube case are separated by a diaphragm in an annulus domain where the inner and outer radius is 0.01 and 1.01 respectively (Fig. 10).

Similar to the one dimensional shock tube problem, the circular shock tube problem has its exact solution which consists of a circular shock wave travelling away from the center, a circular contact surface travelling in the same direction and a circular rarefaction travelling towards the inner circle. It can be easily observed from Fig. 10(b) that the two outer circles are the positions where the shock wave and the contact surface are located and the inner area is the positions for the rarefaction waves. It is clear that they are refined only in the radial direction but not in the azimuthal direction. The density, u-velocity and pressure cross-sectional profiles along x-direction as a function of distance

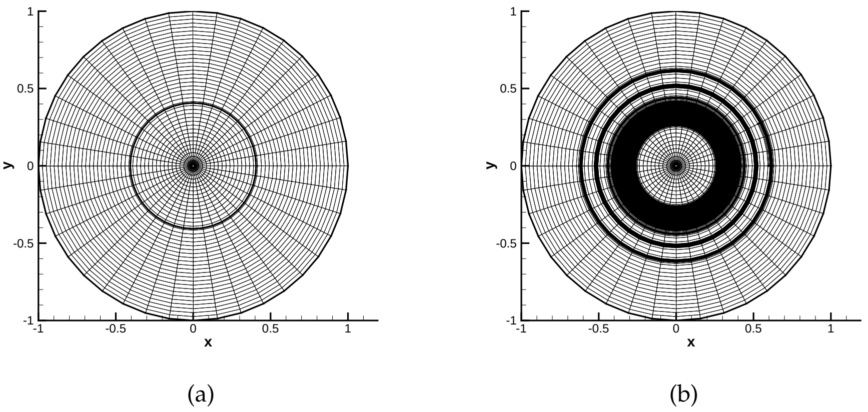


Figure 10: Mesh at $t=0.1$s (a) and mesh at $t=0.2$s (b).

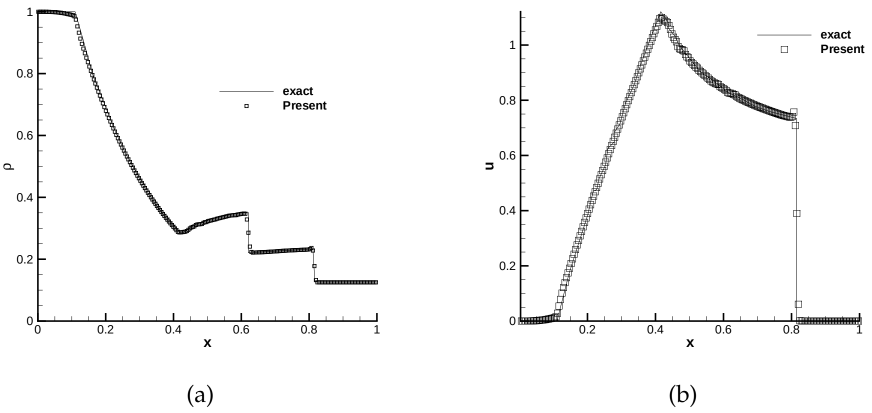


Figure 11: Density (a) and u-velocity (b) cross-section profile along center line of y.

from the inner boundary at 0.25 second are plotted in Figs. 11-12. They are compared with the exact solutions by the random choice method (RCM) method [21]. Good agreement of the results is clearly demonstrated in Figs. 11-12. It shows that the current adaptive criteria could be applied to the problem where the anisotropic flow feature is not aligned to the coordinate axis.

4.3 Shock diffraction over a back-step

In this section, the shock diffraction over a back-step is considered. This problem was studied extensively by experiment and numerical test in [17, 23]. Here, we choose the same geometry and conditions as those in [17]. Initially a right-moving Mach 1.3 shock is positioned at $x=0.5$ from the left boundary. The initial values after the position 0.5 are

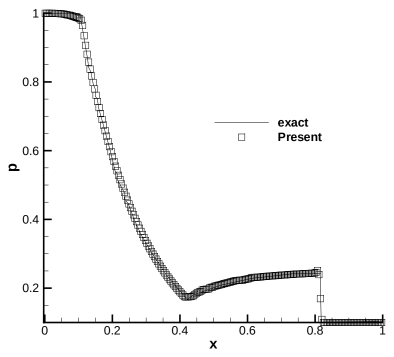


Figure 12: Pressure cross-section profile along center line of y.

set as,

$$\rho_1 = 1.3kg/m^3, \quad u_1 = 0m/s, \quad v_1 = 0m/s, \quad p_1 = 1.0\cdot 10^5 Pa. \tag{4.3}$$

It is conducted on the background mesh which consists of a uniform mesh with three blocks (32×32 in each block (level 0)) and the maximum level as 3. As the incident shock wave propagates to the right, a reflected expansion wave propagates backward and a diffracted shock travels down the step. It is known [17, 23] that the flow consists of expansion waves, a slipstream and the shock wave. From Fig. 13(a), it can be observed that the main shock wave near the top boundary is resolved by the present anisotropic solver as expected and only the vertical direction is refined. Besides, the foot of the diffracted shock wave is perpendicular to the lower wall. Thus, the mesh around the foot of the diffracted shock wave is refined only in the horizontal direction. As the time evolves, the spiral vortex (in Fig. 13) is formed by the rolling up of the slipstream [17]. The mesh for the vortex in our anisotropic solver clearly reflects the vortex structure which could not be clearly observed in the isotropic solver [17] as shown in Fig. 14. Note that Fig. 14 may not be the result at the same time as that of the present result since the physical time is not indicated in [17]. These show a better performance in capturing the anisotropic flow feature of the present anisotropic solver over the isotropic solver. In addition, the total number of cells used in the isotropic result [17] is 11382 while the anisotropic adaptive algorithm requires a total number of cells of 9686. Note that the performance of the present anisotropic adaptation depends on the alignment of the initial background mesh to the potential flow features in the problem. The more the flow features are aligned with the background mesh, the more efficient the anisotropic flow solution. Partial alignment can also improve the efficiency as compared with isotropic refinement.

4.4 Shock wave reflection over wedges

To show the ability of the present method on dealing with more complex problem, the shock wave reflection over a wedge is conducted. It is another benchmark which has

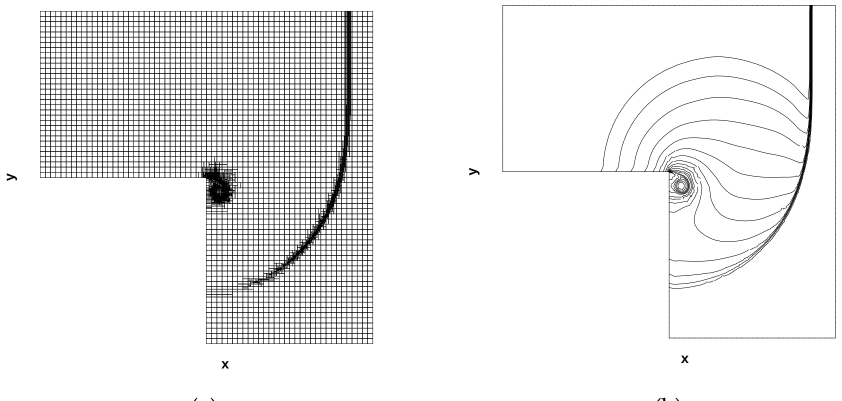


Figure 13: The anisotropic mesh (a) and density contour (b) for the shock diffraction problem at time 0.002s.

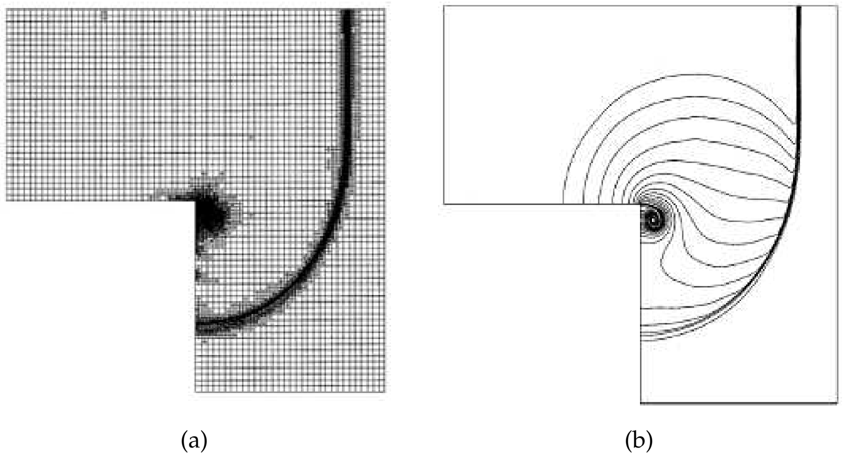


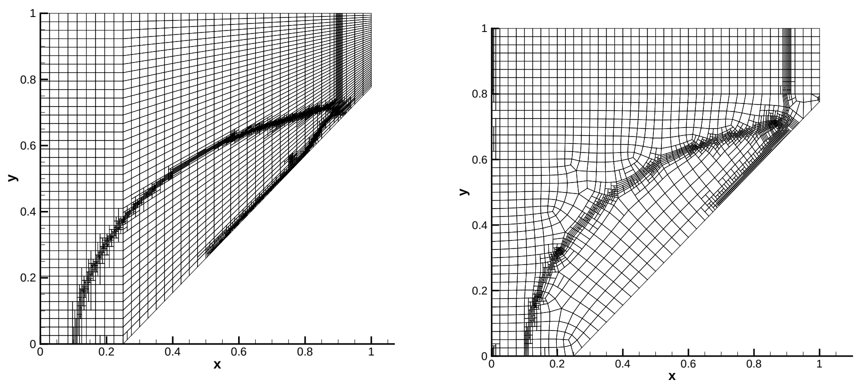
Figure 14: The isotropic mesh (a) and the density contour (b) taken from Sun and Takayama (1999).

been studied extensively in [24]. The wedge corner with the wedge angle is put at 0.25 in a unit square domain. Initially a right-moving Mach 2 shock positioned at $x=0.1$ in the trapezoid domain is specified. The values for the primitive variables before the shock are set as,

$$\rho_1=1.4kg/m^3,\quad u_1=0m/s,\quad v_1=0m/s,\quad p_1=1Pa,\quad \text{for } x<0.1.$$

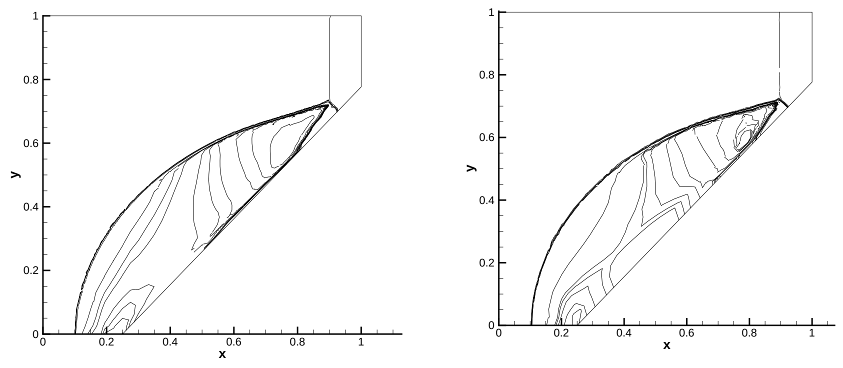
The wall boundary conditions are employed at the top and the two wedge surfaces, while the inlet and trans-missive boundary conditions are employed at the left and right boundary respectively. It is performed on both the structured and unstructured background mesh with a maximum level of 2 (both refinement and coarsening) (Fig. 15). It is clear that most of boundary cells in the unstructured background mesh are orthogonal to the boundary.

At the beginning, the incident shock wave hits the wedge surface and a reflected



(a) (b)

Figure 15: Final anisotropic mesh for the reflection over wedge problem on (a) structured and (b) unstructured background mesh.



(a) (b)

Figure 16: Present density contours on (a) structured and (b) unstructured background mesh.

shock wave is formed. As time evolves, the solution will consist of an incident shock, a reflected shock, a Mach stem, and a slipstream which is closed to the wedge surface. From the observation of the mesh distributions (Fig. 15), all these waves are captured by the present adaptive solver. It is clear that the region around the right-moving incident shock is refined only in the vertical direction in these two meshes which are based on the structure- and unstructured-based background meshes respectively. Besides, some parts of reflected shock wave are refined in the direction aligned with grid. However, there are some differences in the results on the two meshes. From Fig. 16, it is clear that the flow pattern on unstructured-based background meshes matches the pattern of the experiment (Fig. 17) especially in the regions that are close to the wedge. This is due to the fact that the mesh close to the wedge is refined in the direction aligned with the local flow features as can be seen from Fig. 15(b). These show that the present anisotropic solver can generate the anisotropic mesh to reflect the anisotropic flow features. Besides,

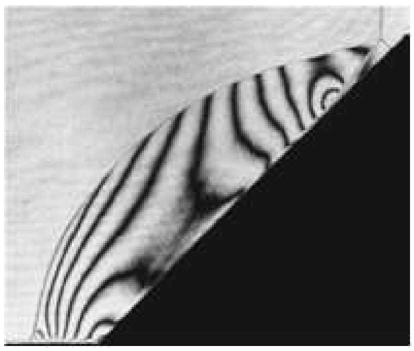


Figure 17: Experiment result.

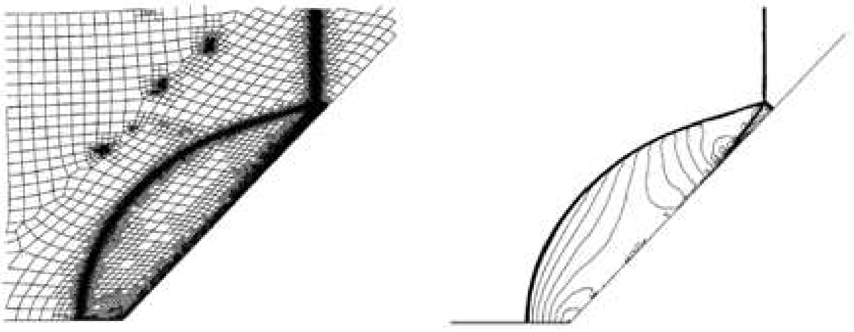


Figure 18: The isotropic adaptive mesh and density contour taken from Sun and Takayama (1999).

the isotropic mesh and results (Fig. 18) which are taken from Sun and Takayama [17] are included for comparison. As seen in Fig. 18(a), their mesh around the incident shock wave is not aligned with the shock wave.

4.5 Two-dimensional Riemann problem

In this section, a two-dimensional Riemann problem is investigated [25]. Initially, four gases are separated by four lines in the four quadrants in the domain. The initial data in the four quadrants are

$$\begin{aligned} &\rho_1=\rho_0, \quad u_1=u_0, \quad v_1=u_0, \quad p_1=p_0,\\ &\rho_2=\rho_0', \quad u_2=v_0, \quad v_2=u_0, \quad p_2=p_0',\\ &\rho_3=\rho_0, \quad u_3=v_0, \quad v_3=v_0, \quad p_3=p_0,\\ &\rho_4=\rho_0', \quad u_4=u_0, \quad v_4=v_0, \quad p_4=p_0', \end{aligned}$$

with

$$\rho_0=1.1kg/m^3, \quad u_0=0m/s, \quad v_0=0.8939m/s, \quad p_0=1.1Pa,$$

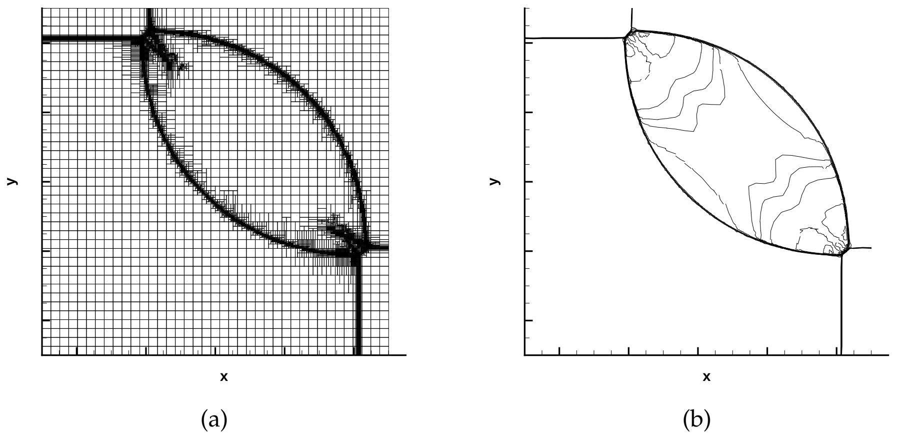


Figure 19: The anisotropic mesh (a) and density contour (b) for the two dimensional Riemann problem at $t=0.25$s.

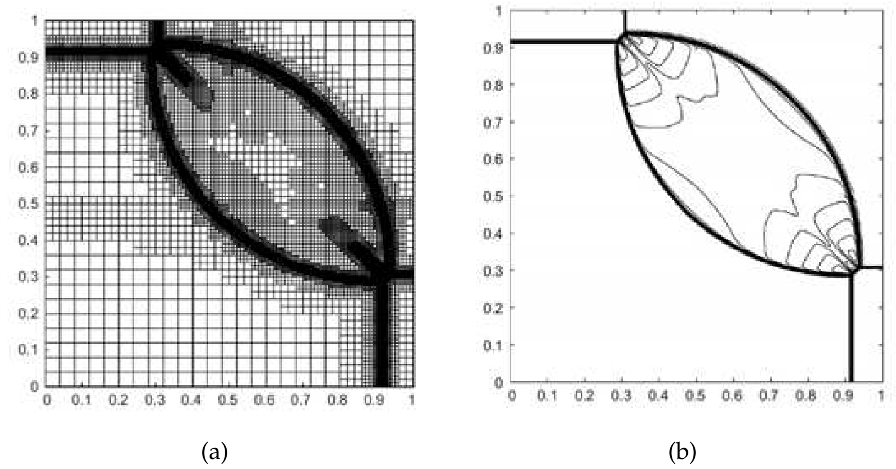


Figure 20: The adaptive mesh (a) and the density contour (b) taken from Tang and Song (2008).

and

$$\rho_0' = 0.5065 kg/m^3, \quad p_0' = 0.35 Pa.$$

It is easily observed that there are four shockwaves with the discontinuities in the density, velocity and pressure at these interfaces between each pair of quadrants. Thus, after breaking the membranes, the shock front between quadrant 1 and quadrant 2 will move from right to left because of the left-going shock wave. Similarly, the shock front between quadrant 2 and quadrant 3 will move upward.

This could be verified by the calculation with the initial mesh that is adaptively generated by a 40×40 background mesh (level 0) with the maximum level as 4. The mesh and density contour at time 0.25 second are shown in Fig. 19(a-b). From the observation of the mesh distributions (Fig. 19), it is clear that the region around the shock front is well refined. This shows that the shock waves are captured well. Besides, we can also observe the cone structure in Fig. 19 which agrees with the results (Fig. 20) in Tang et al. [25]. This confirms our previous expected solution based on the four shocks. From Figs. 19 and 20, it is also clear that the cell numbers used in the present anisotropic solver is much less than those in [25].

5 Conclusion

An anisotropic solution adaptive solver for two-dimensional compressible flows is presented in this paper. It is based on quadrilateral-cell based background mesh (structured or unstructured). The data structure management makes it feasible to apply the face-based algorithm to solve the Euler equations. Five examples have been carried out to examine the performance of the present solver for compressible flows. Numerical results show that the present solver can adaptively and accurately capture the transient anisotropic flow features such as shock waves, contact discontinuities and vortices etc. Substantial computational saving can be achieved in comparison with isotropic mesh refinement/coarsening, which depends on the alignment of the background mesh with the flow features in the problem.

References

[1] Berger M.J. and Oliger J. Adaptive mesh refinement for hyperbolic partial differential equations. J. Comput. Phys. 1984; 53(3):484-512.
[2] Hornung R. D., and Trangenstein J. A. Adaptive mesh refinement and multilevel iteration for flow in porous media. J. Comput. Phys. 1997; 136(2):522-545.
[3] Popinet S. Gerris: a tree-based adaptive solver for the incompressible Euler equations in complex geometries. J. Comput. Phys. 2003; 190(2):572-600.
[4] Liang Q., Borthwick A.G.L., and Stelling, G.. Simulation of Dam and Dyke-break Hydrodynamics on Dynamically Adaptive Quad-tree Grids. Int. J. for Num. Meth. in Fluids 2004; 46(2):127-162.
[5] Charlton E.F., and Powell, K.G. An octree solution to conservation-laws over arbitrary regions (OSCAR). AIAA Paper 1997; 97-0198.
[6] Berger M.J. and LeVeque R.. Adaptive mesh refinement for two-dimensional hyperbolic systems and the AMRCLAW software. SIAM J. Numer. Anal. 1998; 35:2298-2316.
[7] Zeeuw D. De and Powell, K. G. An Adaptively Refined Cartesian Mesh Solver for the Euler Equations. J. Comput. Phys. 1993; 104(1):56-68.
[8] Schmidt G. H. and Jacobs, F. J. Adaptive local grid refinement and multi-grid in numerical reservoir simulation. J. Comput. Phys. 1988; 77(1):140-165.
[9] Wang Z.J. A Quadtree-based adaptive Cartesian/Quad grid flow solver for Navier-Stokes equations. Computers and fluids 1998; 27(44):529-549.
[10] Wang Z.J. and Srinivasan K. An Adaptive Cartesian Grid Generation Method for 'Dirty' Geometry. Int. J. for Num. Meth. in Fluids 2002; 39(8):703-717.

[11] Wang Z.J., Chen R.F., Hariharan N., Przekwas A.J. and Grove D. A 2^N Tree Based Automated Viscous Cartesian Grid Methodology for Feature Capturing. AIAA Paper 1999; 99-3300.

[12] Wang Z.J. and Chen. R.F. Anisotropic Solution-Adaptive Viscous Cartesian Grid Method for Turbulent Flow Simulation. AIAA Journal 2002; 40(10):1969-1978.

[13] Ham F.E., Lien F.S., and Strong A.B. A Cartesian Grid Method with Transient Anisotropic Adaptation. J. Comput. Phys. 2002; 179(2):469-494.

[14] Keats W.A. and Lien F.S. Two-Dimensional Anisotropic Cartesian Mesh Adaptation for the Compressible Euler Equations. Int. J. for Num. Meth. in Fluids 2004; 46(11):1099-1125.

[15] Habashi WG., Dompierre J., Bourgault Y., Ait-Ali-Yahia D., Fortin M., Vallet MG. Anisotropic mesh adaptation: towards user-independent, mesh-independent and solver-independent CFD. Part II: Structured meshes. Int. J. for Num. Meth. in Fluids 2002; 39(8):657-673.

[16] Habashi WG., Dompierre J., Bourgault Y., Ait-Ali-Yahia D., Fortin M., Vallet MG. Anisotropic mesh adaptation: towards user-independent, mesh-independent and solver-independent CFD. Part III: Unstructured meshes. Int. J. for Num. Meth. in Fluids 2002; 39(8):675-702.

[17] Sun M., and Takayama K. Conservative smoothing on an adaptive quadrilateral grid. J. Comput. Phys. 1999; 150(1):143-180.

[18] Zheng H. W., Shu C. and Chew Y.T. An Object-Oriented and Quadrilateral-mesh based Solution Adaptive Algorithm for Compressible Multi-fluid Flows. J. Comput. Phys. 2008; 227(14):6895-6921.

[19] Qin N. and Liu X. Flow feature aligned grid adaptation. Int. J. for Num. Meth. in Engineering 2006; 67(6):787-814.

[20] Toro EF, Spruce M., and Speares W. Restoration of the contact surface in the HLL Riemann solver. Shock Waves 1994; 4(1):25-34.

[21] Toro EF. Riemann Solvers and Numerical Methods for Fluid Dynamics (2nd edn)Springer: Berlin, 1999.

[22] Luo H., Baum J.D., and Lohner R. A hybrid Cartesian grid and gridless method for compressible flows. J. Comput. Phys. 2006; 214(2):618-632.

[23] Takayama K. and Inoue O. Shock wave diffraction over a 90 degree sharp corner, Posters presented at 18th ISSW. Shock Waves 1991; 1(4):301-312.

[24] Takayama K. and Jiang Z. Shock wave reflection over wedges: A benchmark test for CFD and experiments. Shock Waves 1997; 7(4):191-203.

[25] Tang L., and Song H. A multiresolution finite volume scheme for two-dimensional hyperbolic conservation laws. Journal of Computational and Applied Mathematics 2008; 214(2):583-595.