GCFD 31044—21/10/2003—KRISHNAMURTHI—85848

International Journal of Computational Fluid Dynamics, 2003 Vol. not known (not known), pp. 1–10

Taylor & Francis
Taylor & Francis Group

Barron Special Issue

Adaptive Mesh Refinement of Supersonic Channel Flows on Unstructured Meshes*

R.C. RIPLEY[a,†], F.-S. LIEN[b] and M.M. YOVANOVICH[b]

[a]*Martec Ltd.—Combustion Dynamics Group, 400-1888 Brunswick Street, Halifax, NS Canada B3J 3J8;* [b]*Department of Mechanical Engineering, University of Waterloo, 200 University Avenue W., Waterloo Ont., Canada N2L 3G1*

(Received ?????)

An isotropic local mesh refinement technique for unstructured meshes is presented. The present work pertains to inviscid, steady, supersonic internal flows contained in a channel configuration. A density-based mesh refinement sensor function is used to identify regions in the mesh for refinement. A face-cell finite-volume method, permitting dynamic changes to mesh connectivity, is employed in the mesh refinement strategy. The AUSM+ convection scheme is used to compute the interface numerical flux using Mach number and pressure splitting functions. The present method is capable of recursive multi-level mesh adaption. Local mesh refinement equivalent to 64 and 256 times the coarse mesh resolution can be obtained using four and five levels of refinement, respectively. Results for two popular supersonic channel test cases are presented: Mach 1.4 flow over a 4% thick circular arc bump, and Mach 2.0 flow over a 10° compression ramp. The effort associated with the mesh refinement of the bump case accounts for only 4.6% of the total simulation time. For the ramp case, a factor of 4.2 for memory storage requirements and 7.7 for simulation time is required to obtain an equivalent uniform fine mesh solution.

Keywords: Unstructured grids; Compressible flow; Shock capturing; Mesh adaption; Euler equation; Inviscid supersonic flow

NOMENCLATURE

a	speed of sound	p	pressure
A	area of a cell	$\mathscr{P}$	AUSM+ pressure splitting function
CFL	Courant–Friedrichs–Lewy stability number	$\mathbf{Q}$	vector of conservative variables
E	total energy per unit mass		
$\mathbf{F}$	inviscid flux vector	S	face length
K	iteration number	t	time
L, R	left and right states	u, v	Cartesian velocity components
m	cell neighbour index	$\vec{V}$	velocity vector
M	Mach number	V_n	convective velocity normal to cell face
$\mathscr{M}$	AUSM+ Mach number splitting function	x, y	Cartesian coordinates
$\vec{n}$	unit vector normal to cell face	ρ	density

*A shorter version of this article was presented in the Proceedings of the 10th Annual CFD Society of Canada, CFD2002 Conference, Windsor, Ontario, 11–12 June, 2002.

†Corresponding author. Tel.: +1-902-425-5101. E-mail: rcripley@martec.com

ISSN 1061-8562 print/ISSN 1029-0257 online
DOI: 10.1080/10618560310001634168

INTRODUCTION

Local mesh refinement involves the addition of smaller cells in regions of high gradients in the solution. This technique is used to locally improve the spatial accuracy of a numerical solution. For compressible flows, the target of local mesh refinement is typically a shock wave, or other discontinuity. As the refinement of cells is based on the solution itself, these methods are called solution-adaptive, or adaptive mesh refinement (AMR) techniques. The method developed in the present work is a multi-level recursive algorithm applicable to unstructured triangular meshes.

Supersonic channel flows typically contain complex shock patterns which are governed by changes in cross-sectional area such as converging/diverging walls, bumps, ramps and sharp corners. These geometries are useful for studying inviscid high-speed flows, and are ideal for testing local mesh refinement strategies because of the complex shock reflections that divide regions of smooth gradients. In this work, mesh adaption techniques are presented for 2D unstructured triangular meshes. Channel configurations for a compression/expansion ramp and a circular arc bump are computed. The bump flow is used to demonstrate the AMR method and refinement time distribution; the ramp test case is used to validate the results as compared to an equivalent uniform fine mesh solution.

GOVERNING EQUATIONS

The governing equations used to model an inviscid compressible flow are the time-dependent Euler equations, which represent a system of conservation laws for mass, momentum and energy. In vector divergence form, the Euler equation is:

$$\frac{\partial \mathbf{Q}}{\partial t} + \nabla \cdot [\mathbf{F}(\mathbf{Q})] = 0 \tag{1}$$

The variable $\mathbf{Q}$ is a column vector of conservative flow variables and $\mathbf{F}$ is the inviscid flux vector:

$$\mathbf{Q} = \begin{bmatrix} \rho \\ \rho u \\ \rho v \\ \rho E \end{bmatrix}, \quad \mathbf{F} = V_n \begin{bmatrix} \rho \\ \rho u \\ \rho v \\ \rho E + p \end{bmatrix} + p \begin{bmatrix} 0 \\ n_x \\ n_y \\ 0 \end{bmatrix} \tag{2}$$

where ρ is the fluid density, u and v are the Cartesian velocity components, E is the total energy per unit mass and p is the pressure. For unstructured meshes, the normal vector is frequently used: n_x and n_y are the components of the outward unit normal vector, and V_n is the convective velocity normal to the face. The equation of state for an ideal gas is used to close the system of equations.

DATA STRUCTURE AND MESH

Two-dimensional unstructured triangular meshes have been used in the present work. A method of generating the initial mesh for solution-adaptive mesh refinement is a prerequisite. In addition, a flexible computational data structure, capable of describing the cell connectivity and which permits the dynamic addition of refined cells, is required.

Initial Mesh

The initial triangular mesh for the ramp test case was generated using Mesh2D (Niceno, 2001) and the coarse mesh for the bump test case was created using EasyMesh (Karamete, 2001). The Mesh2D and EasyMesh programs are available on the Internet for academic use. These grid generators produce high quality triangular meshes with the Delaunay property. The initial meshes are somewhat coarse, and are uniform in the region where mesh refinement is anticipated.

Flexible Data Structure

The components of the mesh and their connectivity are read from a mesh input file and are stored using several different techniques. The vertices are stored in a vector which is the symbolic equivalent of an array. The vector has the benefit of allowing direct access to the vertex data (coordinates, identification number, etc.) through the vector index number. Unlike an array which requires the size to be predefined and must address a contiguous piece of memory, the vector can be resized and appended dynamically.

The faces and cells in the mesh are stored in a linked-list data structure. Since the connectivity of the vertices, faces and cells is intertwined, the use of the linked-list permits cross-referencing using the memory address of individual objects. For example, each face has memory pointers to its two vertices and its two neighbour cells. The linked-list is also a suitable choice for the faces and cells since the solver loops through these objects during each iteration.

Each cell contains an array of pointers to its vertices, faces, and neighbours. These arrays are variable in length such that they describe cells with two forming points, as in ghost cells used on boundaries, three forming points for triangular cells, or more at mesh refinement interfaces. The freedom to dynamically increase the number of neighbouring cells allows for adaptive mesh refinement where a cell can be subdivided into smaller cells to increase the resolution of the solution.

The face-based solution method first loops through each face in the linked-list and computes the numerical flux based on the left and right states. The flux is stored at the cell centres of the left and right cells. The Gauss-Siedel point solver then loops through the linked-list of cells and explicitly solves for the conservative variables based on

the sum of the fluxes entering and exiting through its bounding faces. Iteration is performed until the residuals have been sufficiently decreased and the solution has converged. The numerical method for computing the fluxes is described in detail in the next section.

NUMERICAL METHOD

The conservative flow variables are stored at the cell centroids, and are assumed to be piece-wise constant within the cell. The flux vector is split using the AUSM+ flux vector splitting scheme (Liou, 1996; Liou and Singh, 1999), outlined as follows.

First, a is the interface numerical speed of sound which is taken as the arithmetic average of neighbouring left (L) and right (R) states:

$$a = \frac{a_L + a_R}{2} \tag{3}$$

The face-normal Mach numbers for the left and right cells are:

$$M_{nL} = \frac{\vec{V}_L \cdot \vec{n}}{a}, \quad M_{nR} = \frac{\vec{V}_L \cdot \vec{n}}{a} \tag{4}$$

The interface convective Mach number is determined by using Mach number splitting functions ($\mathscr{M}^{\pm}$) based on neighbouring Mach numbers,

$$M_n = \mathscr{M}^{+}(M_{nL}) + \mathscr{M}^{-}(M_{nR}) \tag{5}$$

The Mach number splitting functions (Liou, 1996) are:

$$\mathscr{M}^{\pm}(M) = \begin{cases} \frac{1}{2}(M \pm |M|), & \text{if } |M| > 1 \\ \pm\frac{1}{4}(M \pm 1)^2 \pm \frac{1}{8}(M^2 - 1)^2, & \text{otherwise} \end{cases} \tag{6}$$

The interface pressure is obtained with contributions from the left and right states using pressure splitting functions ($\mathscr{P}^{\pm}$) based on the neighbouring Mach numbers,

$$p = \mathscr{P}^{+}(M_{nL}) \cdot p_L + \mathscr{P}^{-}(M_{nR}) \cdot p_R \tag{7}$$

The pressure splitting functions (Liou, 1996) are:

$$\mathscr{P}^{\pm}(M) = \begin{cases} \frac{1}{2}(1 \pm \text{sign}(M)), & \text{if } |M| \geq 1 \\ \frac{1}{4}(M \pm 1)^2(2 \mp M) \pm \frac{3}{16}(M^2 - 1)^2, & \text{otherwise} \end{cases} \tag{8}$$

Full upwinding of both the pressure and Mach number is achieved for supersonic flow; a polynomial blending of the upstream and downstream contributions results for subsonic conditions. The inviscid numerical flux normal to the cell interface is then assembled using Eqs. (5) and (7).

$$\mathbf{F}_n = \frac{1}{2} a \left[M_n(\mathbf{Q}_L + \mathbf{Q}_R) - |M_n|(\mathbf{Q}_R - \mathbf{Q}_L) \right] + p \cdot [0 \; n_x \; n_y \; 0]^T \tag{9}$$

The fluxes are applied to each cell in the mesh and the solution to the conservative variables is advanced using an explicit Gauss–Siedel point solver. For the mesh refinement test cases, a first-order implementation of the fluxes was used,

$$\mathbf{Q}^{k+1} = \mathbf{Q}^k - \frac{\Delta t}{A} \sum_{m=1}^{3} \mathbf{F}_{n,m} S_m \tag{10}$$

where S_m is the face length of face m. The timestep used to advance the solution in Eq. (10) is computed using

$$\Delta t = \frac{\text{CFL} \cdot S_{\min}}{|\sqrt{u^2 + v^2} + a|} \tag{11}$$

where $S_{\min}$ is the characteristic length scale which is taken as the face length for equilateral or low-aspect-ratio triangular cells, and an appropriate choice of CFL number is used to ensure stability. The CFL number is the ratio of timestep to characteristic convection time, and must be less than unity for stability.

LOCAL MESH REFINEMENT STRATEGY

Solution-adaptive mesh refinement requires a test criterion to determine regions in the mesh to refine, and an algorithm to subdivide the mesh to improve the local resolution. An optimal mesh criterion is not required, as multiple levels of mesh refinement are applied until the maximum number of mesh refinement levels have been achieved.

Cell Division Options

Figure 1 shows three simple cell division options whereby a triangular control volume is subdivided into two, three and four smaller triangles, referred to as bilateral, trilateral and quadrilateral cell division, respectively.

Both the bilateral and trilateral cell division options produce smaller triangular cells which have an increased aspect ratio from that of the original cell. These cell division options are therefore anisotropic. Under successive cell division, anisotropic refinement methods can produce low quality cells that may only have a marginal improvement in solution resolution.

Isotropic refinement of a triangular cell is achieved using quadrilateral cell division. Each triangle is divided into four

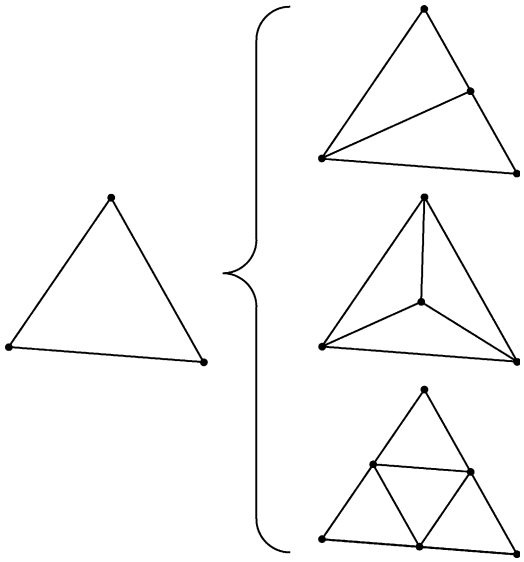

FIGURE 1 Options for refinement of a triangular cell (from top to bottom): bilateral cell division, trilateral cell division and quadrilateral cell division.

congruent triangular cells, each having exactly one quarter of the initial cell volume. The fine cells are created by joining the bisection point of each face in the parent cell, resulting in an inscribed cell and three corner cells, as shown in Fig. 1. For the present isotropic mesh refinement investigation, quadrilateral cell division is used exclusively.

Solution redistribution from the coarse to the fine cells is achieved by simply inheriting the flow variables from the parent cell.

Test Criterion

The sensor function used herein was adopted from Rivara (1989) and Chen *et al.* (1997) as implemented by Lien (2000). The sensor function is based on density using an undivided difference or delta. For compressible flow, a pressure-based sensor function is not suitable as it will not detect contact discontinuities (slip lines). For each cell in the mesh, a mesh refinement sensor function (Lien, 2000) was computed based on neighbouring cells,

$$\text{func}(\phi)_{\phi=\rho} = \sum_{m=1}^{3} |\rho_o - \rho_m| \tag{12}$$

where o is the cell under consideration and m is the cell neighbour index. Cells in the mesh should be tagged for local refinement only if the following criterion is met:

$$\text{func}(\phi) > \text{func}(\phi)_{\min} + \text{ratio}\left[\text{func}(\phi)_{\max} - \text{func}(\phi)_{\min}\right] \tag{13}$$

The adjustable ratio in Eq. (13), or mesh refinement threshold, was determined empirically by trial and error, and generally ranges from 5 to 10%. The marking of cells for refinement requires two sweeps of the mesh—one to determine the refinement criterion, and the second to test each cell against the threshold value.

Refinement along Curved Geometry

In addition to improving spatial resolution in the mesh volume, an increase in resolution of the geometry is necessary. Figure 2 shows the refinement of a cell adjacent to a curved boundary, where a simple adjustment to the vertex location is employed. This procedure is applicable when the radius of curvature is much greater than the face length of a cell. Examples of curved geometries include circular arcs, ellipses, and airfoil cross sections.

Mesh Refinement Rules

Multiple levels of mesh refinement are possible with a recursive algorithm. A numbering system is therefore required to identify the current level of refinement and to make reference to parent or children cells on different levels. Further, several mesh refinement rules are required to minimize weighting errors between coarse and fine cells:

1. neighbouring cells cannot differ in refinement level by more than one from the coarser cell; and,
2. a coarse cell cannot be surrounded by more than one refined cell.

These rules prevent a number of undesirable features in the mesh. Cell interfaces differing in refinement level by more than one will result in numerous flux contributions through relatively small faces, which will lead to local solution degradation by diffusion of the solution at the interface. Further, cells surrounded by more than one

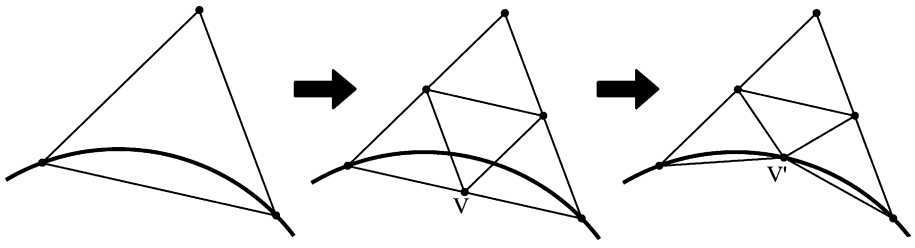

FIGURE 2 Example of cell refinement along a curved boundary. Vertex **V** is moved to a point **V′** which is interpolated along the curved boundary.

refined cell, or cells differing in refinement level by more than one, will create coarse cells formed by more than four faces. This can complicate post-processing since most readily available plotting utilities are designed exclusively for 2D finite-volume data consisting of three- and four-sided (triangular and quadrilateral) cells.

Mesh Refinement Procedure

The recursive mesh refinement procedure used for steady flows is described. The mesh refinement process for a given level is applied each time the RMS residual is decreased below 10^{-7} until the desired maximum number of refinement levels is achieved. The governing equations are iterated and partially converged for each level of mesh refinement before applying a subsequent level of finer cells.

The calculation of fluxes at mesh refinement interfaces between fine and coarse levels is straightforward using the present face-based method. Each coarse cell along the interface is modified such that the face adjacent to a refined cell is bisected into two new faces corresponding to the two neighbouring refined cells. As shown in Fig. 3, there are four fluxes contributing to the change in conservative variables in cell *C*. The unstructured-grid methods used in the code permit such modifications to the cell and face connectivity data—a process that would be difficult for some structured-grid methods with rigid data structures.

Local timestepping was employed whereby each cell in the mesh was advanced using the same timestep. A fixed CFL number of 0.35 was used to calculate the local timestep and the solution was then advanced using the minimum timestep in the domain. This ensures that stability is met at each timestep; however, as the cells ar locally refined, the global timestep is reduced by a factor of about two for each successive mesh refinement level.

Multi-level Mesh Refinement

Multiple levels of mesh refinement are possible, using so-called recursive or nested cell subdivisions. The parent/child analogy is used to reference between refinement levels. The relative number of cells on any given level can be determined using the following relationship:

$$N_{\text{cells}} = 4^{\text{Level}-1} \tag{14}$$

For example, the fifth level mesh refinement contains 256 times the number of cells within a single triangular cell on the coarsest mesh.

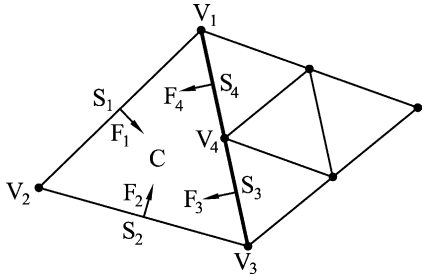

FIGURE 3 Details of interface between coarse and refined cells (**F**, flux; **S**, face length; **V**, vertex; *C*, cell).

MESH REFINEMENT RESULTS

Supersonic Bump Test Case

The bump test case features a supersonic channel flow with a circular arc bump placed along the lower wall. The inlet Mach number is 1.4 and the circular bump is 4% thick.

Figure 4 shows the convergence behaviour for the supersonic bump simulation. Spikes in the residuals are clearly evident, which correspond to refinement of the mesh. The effect on the residuals by the addition of refined cells indicates that the present method employs a non-conservative interpolation between mesh refinement levels. This convergence behaviour is consistent with the results of de Zeeuw and Powell (1993), who used Cartesian methods, and does not impact the steady-state solution. Sun (1998) demonstrated that a conservative interpolation method is possible—this would allow the current mesh adaption strategy to be extended to unsteady flows.

Distribution of Time Spent on Mesh Refinement Tasks

The distribution of mesh refinement tasks is shown in Fig. 5. Mesh refinement tasks accounted for only 4.6% of the total simulation time. Convergence was achieved on the finest mesh in 1975 s using a Pentium III—500 MHz CPU.

Refinement of cells involves cell division and modifications to the connectivity data. The Grow Mesh algorithm employs an isotropic advancing front method, proposed by Lien (2000), that sequentially expands the local refinement into the coarse mesh region. Growing the mesh increases the refinement region size, but can give rise to a problem called hanging cells, which are coarse cells along the refinement interface that are bounded by two or more refined cells. These hanging cells are recursively refined until they are completely removed. Miscellaneous tasks include: application of refinement sensor function; setting face normal vectors, face length, cell area, etc., for refined cells; interpolation of fine mesh solution for parent (coarse) cells; modification of boundary conditions; computing primitive variables; and, estimating gradients in the refined cells.

Figure 6 shows the intermediate steps during one level of mesh refinement. A diagonalized quadrilateral mesh was chosen for illustrative purposes. The solution is not computed on the intermediate meshes.

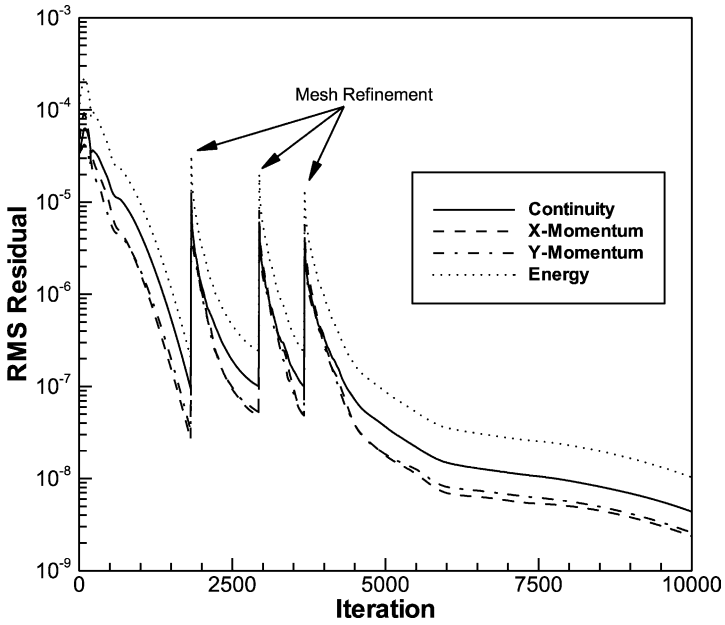

FIGURE 4 Convergence history for the 4-level mesh refinement of the Mach 1.4 flow past a 4% thick circular arc bump.

Figure 7 shows the initial and adapted meshes for the bump test case, and the resulting pressure contours for the coarse and level 4 meshes. The mesh adaption successfully captured the shock reflections, as well as the oblique shock from the trailing edge of the bump. The pressure contour plot illustrates very sharp resolution of the shock waves in comparison to the coarse mesh solution. The present results are in agreement with those of Lien (2000).

Supersonic Ramp Test Case

The supersonic ramp test case has an inlet Mach number of 2.0. The ramp geometry consists of a 10° compression ramp followed by a 10° expansion corner along the lower channel wall.

The pressure coefficient plot in Fig. 8 shows that the level 5 solution is nearly grid independent (compare levels 4 and 5) and there is little benefit to further mesh refinement considering the additional computational effort. Refinement to level 6 would exceed the available memory and was not attempted. Further, the increase from level 4 to level 5 increased the simulation time from a matter of minutes to the order of hours.

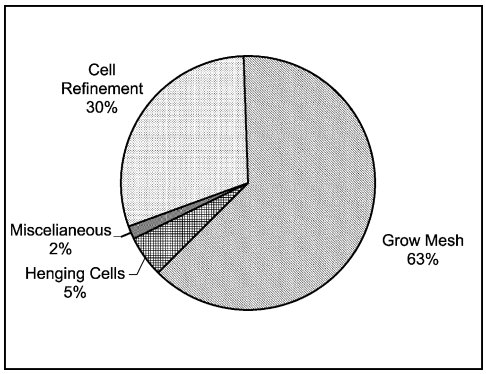

FIGURE 5 Distribution of time spent on mesh refinement tasks.

The execution time increases for each subsequent level of refinement because the local timestep is reduced by a factor of two since the characteristic cell size is reduced by two, and the number of cells is increased locally by a factor of four when using quadrilateral cell division. The net effect is an increase in computational effort by a factor of 8 for each level of mesh refinement. Local mesh refinement somewhat alleviates this burden by selective application of refined cells only to regions of interest, as compared to uniform mesh refinement, which subdivides every cell in the mesh.

Comparison to Uniform Fine Mesh

Table I summarizes the differences between simulations with the 5-level local mesh refinement and a uniform mesh equivalent to the level 5 mesh resolution. The level 5 resolution is equivalent to 256 times that of the coarse mesh, which contains 1094 cells. The time and number of iterations correspond to the time that convergence to machine zero is achieved. The ramp test case was executed on a 200 MHz Sun UltraSparcII CPU.

Figure 9 shows that the uniform mesh and the local mesh refinement produce identical solutions for lower wall pressure coefficient. Even the overshoot and oscillations at $x = 1$ are reproduced when using local mesh refinement. Pressure contour plots are also very similar.

Convergence histories (Fig. 10) show that the uniform mesh required considerably fewer iterations to converge to machine zero; however, each iteration was more time consuming for the uniform mesh due to the additional

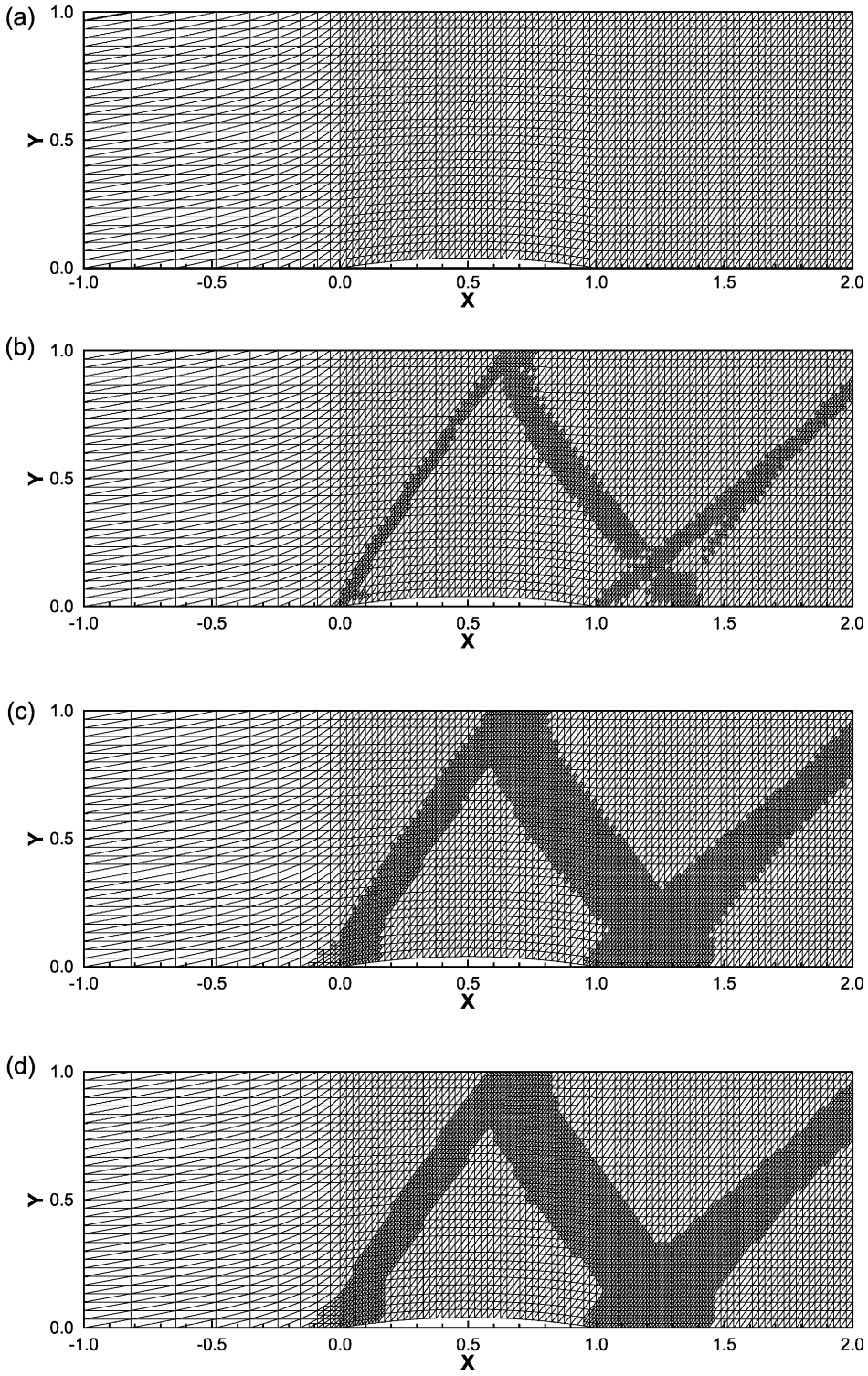

FIGURE 6 Illustration of intermediate mesh refinement steps. (a) Original unstructured mesh (5400 cells), (b) Cells targeted by sensor function have been refined (7611 cells), (c) Mesh following isotropic expansion of the refinement region (10,158 cells) and (d) Final mesh after smoothing the interface to remove hanging cells (10,314 cells).

cells. CPU time and memory requirements shown in Table I demonstrate that the local mesh refinement technique is superior.

Figure 11 shows the adapted mesh and pressure contours for the ramp test case. For level 2, the entire mesh above the ramp was refined because the coarse mesh solution was excessively diffuse and was targeted by a sufficiently relaxed sensor function. This is desirable since it provides a refined region for the solution to evolve where discontinuities can be differentiated from low gradient regions. For level 5, the oblique shock reflection pattern is clearly captured by the mesh refinement sensor

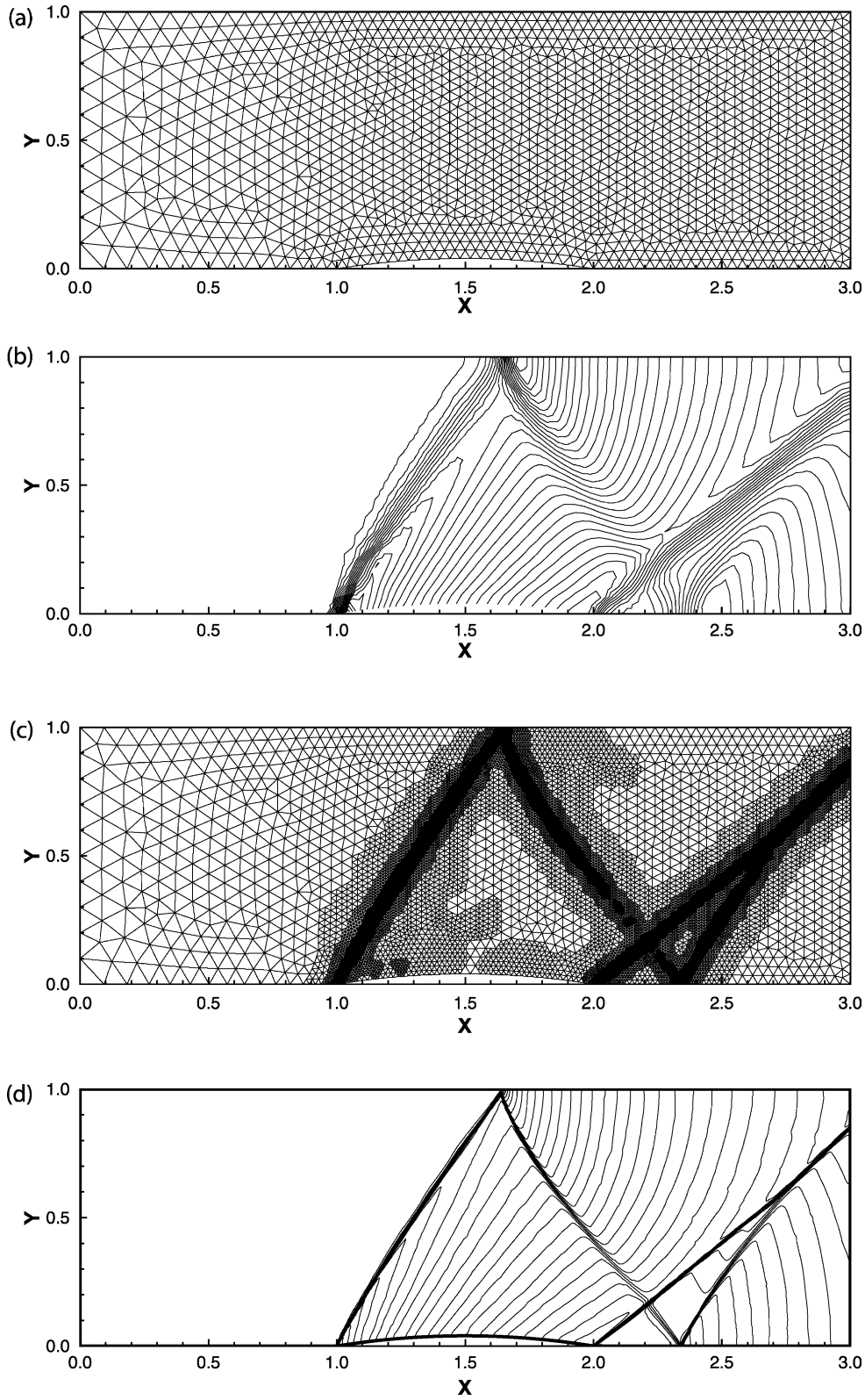

FIGURE 7 Mach 1.4 flow past a 4% thick circular arc bump on a channel wall illustrating meshes and isobar contour solutions (30 isobars for $0.458 < p < 1.478$ with $\Delta p = 0.033$). (a) Level 1 coarse mesh (3318 cells), (b) Level 1 initial isobar contour solution, (c) Level 4 adapted mesh (32,948 cells) and (d) Level 4 converged isobar contour solution.

TABLE I Performance comparison of AMR and uniform meshes

Measure	AMR	Uniform	Factor
Cells	50,837	280,064	5.5
Memory	76 MB	319 MB	4.2
Time	3.4 h	26.3 h	7.7
Iterations	19,140	12,230	0.64

function. Further, the expansion fan at $x = 2$ was also targeted by the mesh refinement algorithm. The isobar contours demonstrate crisp shock waves in the channel, and a substantial increase in resolution over the level 2 results.

Comparison to Analytical Results

Analytical solutions to oblique shock waves, shock reflections and Prandtl–Meyer expansion fans are tabulated and graphed in many gas dynamics textbooks, e.g. (John, 1984). The states computed in the supersonic ramp test case were validated against analytical results up to the point where the expansion fan interacts with the reflected shock, after which the analytical solutions are no longer valid. States 1–4, as indicated in Figure 11d, are compared for Mach number, which is representative of all the flow variables. In addition, the level 5 results were identical to the uniform mesh results and are listed as the numerical results. The comparison of analytical results to the present numerical results is given in Table II.

The Mach numbers at the four state locations are in excellent agreement with the analytical solutions. Further, from the oblique shock relationships (John, 1984), the oblique shock angle for Mach 2.0 flow past a 10° incline is 39°. The present numerical results, which predicted an oblique shock angle of 39.4°, are also in close agreement with the tabulated value. Note that the resolution of the tabulated results is in Mach 0.05 intervals and angles are interpolated to the nearest degree. Overall, the present numerical results are within 1% of the analytical predictions.

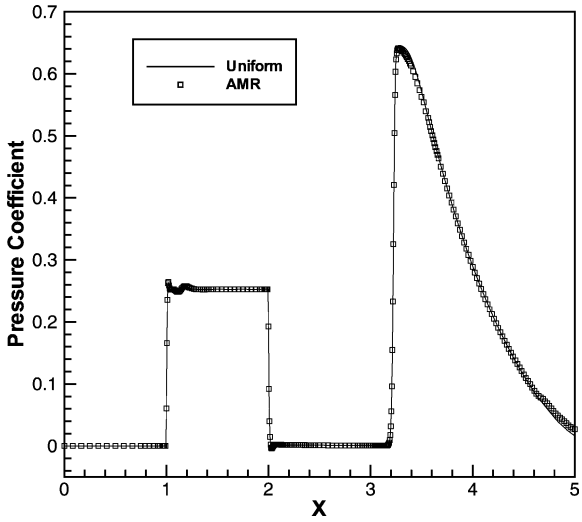

FIGURE 9 Comparison of lower wall pressure coefficient for adaptive (259 points) and uniform (964 points) meshes.

CONCLUSIONS

A multi-level recursive local mesh refinement strategy for unstructured triangular meshes has been developed for high-speed channel flows involving shock waves. The method uses flux-vector splitting techniques with a face-based solver. A mesh refinement sensor based on the density solution successfully targeted shock wave reflections and an expansion fan for local refinement. Application of the present technique for steady flows in three dimensions is possible within the present framework.

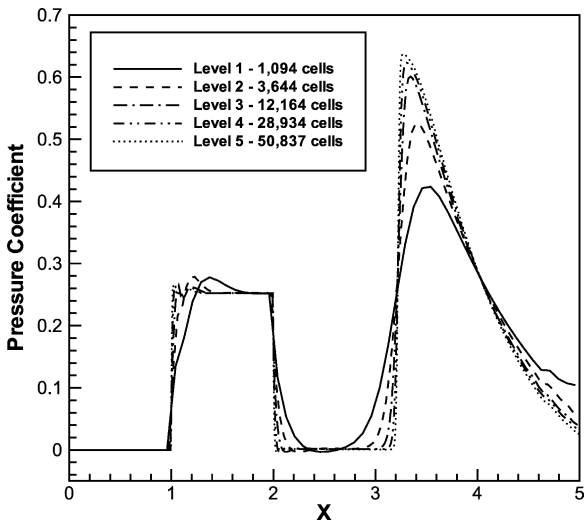

FIGURE 8 Pressure coefficient along lower ramp wall for each mesh refinement level.

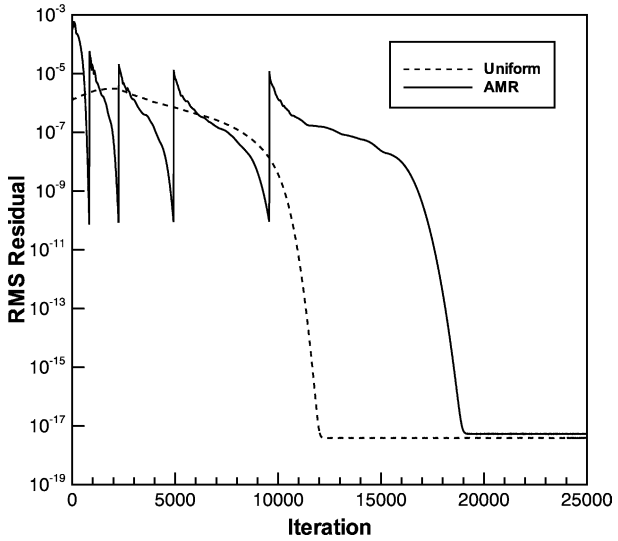

FIGURE 10 Comparison of convergence histories (RMS residual for continuity equation) for adaptive and uniform meshes.

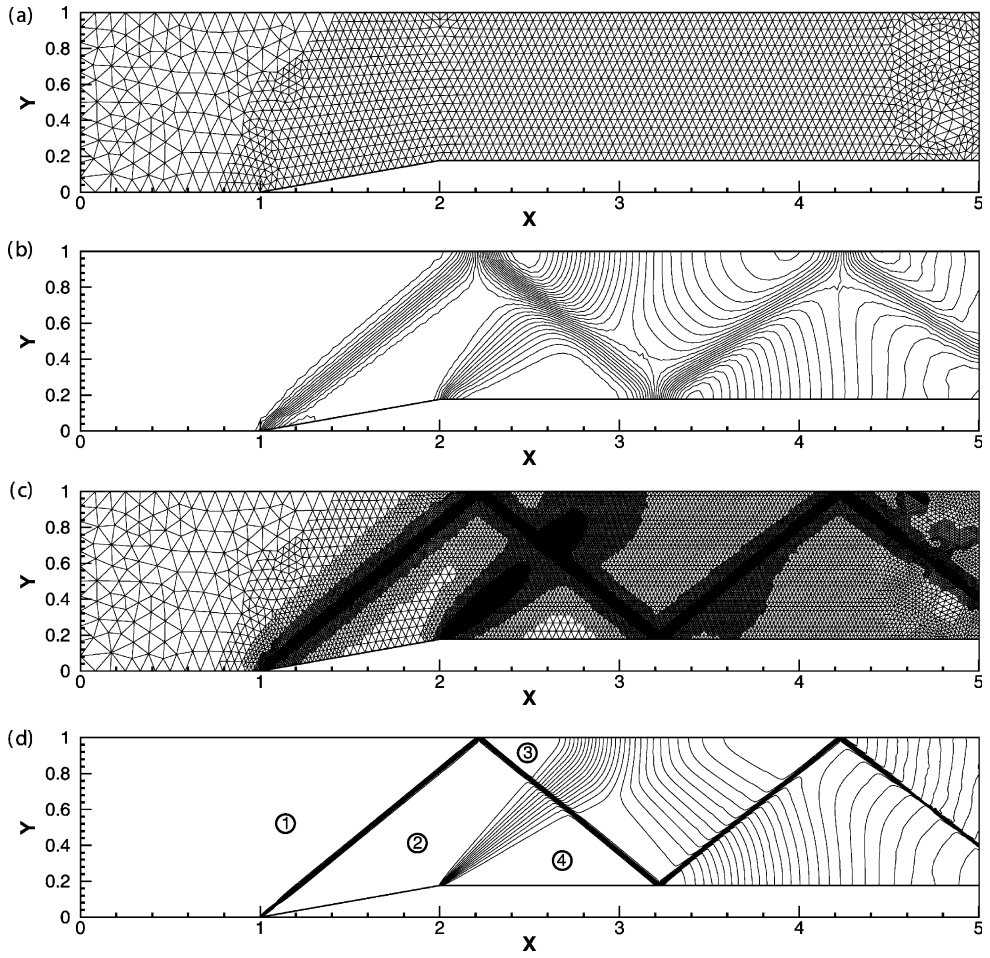

FIGURE 11 Mach 2.0 flow over a 10° ramp in a duct illustrating adapted meshes and resulting isobar contour solutions (30 isobars for $0.706 < p < 2.005$ with $\Delta p = 0.042$). (a) Level 2 intermediate mesh (3644 cells), (b) Level 2 intermediate isobar contour solution, (c) Level 5 adapted mesh (50,837 cells) and (d) Level 5 converged isobar contour solution.

TABLE II Comparison of numerical and analytical results

State point	Analytical Mach number	Numerical Mach number	Percent difference (%)
1	2.00	2.000	0.0
2	1.65	1.639	0.7
3	1.30	1.287	1.0
4	2.00	1.985	0.8

Although only internal channel flows were considered here, the current method is fully capable of computing steady inviscid flows about external geometries. The present method requires minor modifications, including the addition of a de-refinement algorithm and conservative interpolation, for extension to transient simulations.

Acknowledgements

The authors are grateful to the Natural Sciences and Engineering Research Council of Canada (NSERC) for the financial support of this work, and to Dr Meng-Sing Liou of the NASA Glenn Research Center for his helpful discussions of the AUSM scheme.

References

Chen, W.L., Lien, F.-S. and Leschziner, M.A. (1997) "Local mesh refinement within a multi-block structured grid scheme for general flows", *Comput. Methods Appl. Mech. Eng.* **144**, 327–369.

John, J.E.A. (1984) Gas Dynamics, 2nd Ed. (Allyn and Bacon, Boston, MA).

Karamete, B.K. (2001) User Manual of 2D Constrained Delaunay Mesh Generator MESH2D [www: http://www.andrew.cmu.edu/user/sowen/software/mesh2d.html].

Lien, F.-S. (2000) "A pressure-based unstructured grid method for all-speed flows", *Int. J. Numer. Methods Fluids* **33**, 355–374.

Liou, M.-S. (1996) "A sequel to AUSM: AUSM+", *J. Comp. Phys.* **129**, 364–382.

Liou, M.-S. and Singh, K.P. (1999) "Unstructured-grid solutions of turbulent flows using AUSM+", *Comput. Fluid Dyn. J.* **2**(8), 195–207.

Niceno, B. (2001) EasyMesh Version 1.4: A Two-dimensional Quality Mesh Generator [www: http://www.hta-bi.bfh.ch/sha/pwf/fem/EasyMesh/html/easymesh.html].

Rivara, M-.C. (1989) "Selective refinement/derefinement algorithms for sequences of nested triangulations", *Int. J. Numer. Methods Eng.* **28**, 2889–2906.

Sun, M., (1989) "Numerical and experimental studies of shock wave interactions with bodies" Ph.D. Thesis, Tohoku University (Japan).

de Zeeuw, D. and Powell, K.G. (1993) "An adaptively refined Cartesian mesh solver for the Euler equations", *J. Comp. Phys.* **104**, 56–68.