JOURNAL OF COMPUTATIONAL PHYSICS **102**, 98–113 (1992)

Global and Local Remeshing Algorithms for Compressible Flows

C. J. HWANG AND S. J. WU

Institute of Aeronautics and Astronautics, National Cheng Kung University, Tainan, Taiwan, Republic of China

Received September 18, 1990; revised April 24, 1991

A new adaptive remeshing approach for unstructured meshes, which includes the error indicator, global and local mesh regeneration techniques, has been developed in this paper. In this approach, nodes are first distributed according to the remeshing parameters, and those nodes are connected into a complete mesh. The concepts of side-based and vertex-based fronts are introduced to achieve the triangulation. According to the CPU time and the versatility, the vertex-based triangulation technique is proved to be more efficient. By using vertex-based triangulation approach, a local remeshing method, which regenerates only some regions of the flow domain, is presented. To demonstrate the reliability and availability of the proposed procedure, several compressible flow problems are investigated. The regular/stretched triangles and the mixed type of triangular and quadrilateral stretched elements are used. In this work, the Euler equations are solved by the multi-step Runge–Kutta Galerkin finite element methods. From the numerical results, the approaches, which employ the directionally stretched elements, are effective and suitable for treating the flow problems with one-dimensional features. The development of the local remeshing algorithm for unsteady flows is worthwhile and important.

1. INTRODUCTION

In the recent years, unstructured meshes have been widely used in computational fluid dynamics. Several mesh generation approaches have been developed. Given a set of points, Delaunay triangulation [1–3] and front technique [4] were respectively used to form triangles by properly connecting these nodes. Lo [5] generalized the notion of Delaunay triangulation to non-convex planar domains by integrating the method of advancing front and the Delaunay triangulation algorithm. On the other hand, one advancing front method, which generates nodes and elements at the same time, was developed by Peraire *et al.* [6].

In order to have more accurate numerical solutions, various grid adaptation methods have been used to solve problems with high gradient features, such as shock in compressible flow. Three typical grid adaptation techniques, mesh movement [7], mesh refinement [7–11] and mesh regeneration [6], have been developed. For the mesh movement, nodes are moved to new positions without increasing total number of nodes, so that the resolution of the final computation is limited by the initial grid [6]. Mesh refinement was devised to increase the number of nodes in the regions with high solution gradients. Though high-resolution results may be obtained, the refinement technique does not efficiently treat the areas with one-dimensional flow features [6]. Another method developed by Peraire *et al.* [6] shows a direction in adaptive grid generation, in which the new mesh on the overall domain is created according to solutions on the previous mesh. The main advantage of this method is that it allows efficient simulation of one-dimensional flow features by generating elements which are directionally stretched. As mentioned in Ref. [6], even though a full analysis of influence of stretching on the accuracy of solution has not been performed, the numerical computations indicate that good accuracy can still be maintained. Thornton *et al.* [12, 13] developed a remeshing approach which uses quadrilateral elements where possible, and triangles are introduced as needed. As mentioned in Ref. [12], this remeshing technique is suitable for boundary layers and takes considerably less computer storage in 3D applications. For the transient problems, Lohner [14] discussed the features of adaptive remeshing and refinement techniques. A combination of adaptive remeshing and *h*-refinement was introduced to investigate strongly unsteady flows. For moving boundary problems, Probert *et al.* [15] employed and extended a remeshing technique [6] to achieve the local regeneration of the grid. As indicated in Ref. [14], partial remeshing is an area that deserves further study, particularly if shocks change direction.

It is known that error indicators play an important role in the solution-adaptive procedures. Depending on the second derivatives of a key variable or the second derivatives of gradients of the variable, several kinds of error indicators for directional remeshing were used [6, 13, 14]. In the present paper, a new error indicator, which simply uses first derivatives, is developed to decide the directional remeshing parameters: node spacing, ratio

0021-9991/92 $5.00

and direction of stretching. In order to achieve the global or local remeshing, a straightforward regeneration algorithm is also presented. Two independent steps, distribution of nodes and triangulation, are employed in this work. For the placement of nodes, two types of node-clustering procedures are introduced to determine the positions of interior points for the regular and stretched elements, respectively. With regard to the formation of triangles, the concepts of side-based and vertex-based fronts are presented. From the numerical experiments, the vertex-based triangulation technique is more efficient. Combining the adaptive mesh generation technique mentioned above with a recovering procedure, the local remeshing method, which regenerates only some part of the flow domain, is successfully developed. Because of this progress, the coupling of time-varying meshes with flow solver becomes a possible and efficient way to treat the unsteady flow problems. To illustrate the reliability and availability of the present procedure, several compressible flow problems, which include supersonic flow over a compression corner, shock reflection at a wall, supersonic flow passing through a channel with a 4% circular are bump, transonic flow around a two-element airfoil, and shock propagation in a channel, are investigated. The regular/stretched triangles and the mixed type of triangular and quadrilateral stretched elements are used. In this paper, the Euler equations are solved by the multi-step Runge–Kutta Galerkin finite element methods.

2. ERROR INDICATOR

In the solution-adaptive approach, an indicator is required to evaluate the local "error" of the numerical solution. The major difference of error indicators for the directional and non-directional grid adaptation is that stretching parameters are needed for the former approach. For the directional mesh generation, three types of error indicators are available. They are described as follows:

1. An indicator by the second derivatives of a key variable [6].

2. The multiple indicators which are obtained by the second derivatives of a key variable and the second derivatives of gradients of the key variable [13].

3. A modified error indicator based on scaled second derivatives [14].

All the above error indicators are essentially based on the second derivatives. As mentioned in Ref. [13], the first indicator will result in larger elements in the center of a shock, while the second error indicator overcomes this problem. With regard to the multiple indicators, it is required to compute two sets of parameters and select the preferred one. For the third type of error indicator, the second derivatives are scaled by nodal mean values and gradients of solution. It has the advantages that "eating-up" effect [14] in the presence of a very strong shock is avoided and the wiggles or ripples can be filtered. As mentioned in Ref. [14], this type of error indicator cannot meet the requirement that the generated grids do not exhibit an element size much smaller than the prescribed minimum element size. Superior grids can be achieved by smoothing and limiting the initial distribution of remeshing parameters [14]. In this paper, a direct and simple approach is devised. Three parameters are used for the construction of a new mesh: node spacing δ, direction of stretching $\boldsymbol{\alpha}$, and stretching ratio S. The unit vector $\boldsymbol{\alpha}$ represents the direction along which an element is to be stretched. δ and δS are the spacings normal and tangential to the vector $\boldsymbol{\alpha}$, respectively. The following procedures are proposed to find the remeshing parameters.

A. *Node Spacing*

Node spacing must be smaller in regions of large change in properties and may be evaluated by several kinds of error indicators that have been used in non-directional grid adaptation methods [7–11]. For simplicity, node spacing is evaluated by the expression

$$\delta_i^2 \, |\nabla \varphi|_i = \text{const}, \tag{2.1}$$

where

φ = the selected key variable

δ_i = the spacing at node i

$|\nabla\varphi|_i$ = absolute value of the gradient of φ at node i.

For the computations of compressible flows, the density is selected as the key variable. The constant in Eq. (2.1) is determined by the product of a specified minimum node spacing and the absolute value of the maximum solution gradient over the whole mesh.

B. *Direction of Stretching*

The direction of solution gradient is the direction of maximum change in the solution, and it is expected that the nodal direction of stretching is normal to the direction of nodal gradients. Thus the stretching direction is easily obtained when the nodal solution gradients are computed. Let $\mathbf{n}_i$ be the unit vector of nodal solution gradient and $\boldsymbol{\alpha}_i$ be the unit vector which is normal to $\mathbf{n}_i$, i.e.,

$$\mathbf{n}_i = \frac{(\nabla\varphi)_i}{|\nabla\varphi|_i} \tag{2.2}$$

and

$$\mathbf{n}_i \cdot \boldsymbol{\alpha}_i = 0. \tag{2.3}$$

The direction of stretching, $\boldsymbol{\alpha}_i$, is expressed in a one-parameter form by the angle ϕ_i:

$$\boldsymbol{\alpha}_i = (\cos \phi_i)\mathbf{i} + (\sin \phi_i)\mathbf{j}. \tag{2.4}$$

Since either $\boldsymbol{\alpha}_i$ or $-\boldsymbol{\alpha}_i$ can be regarded as the direction of stretching, the value of the angle ϕ_i is defined in the range of 0 and 180°.

C. *Stretching Ratio*

The stretching ratio is expected to be large in regions where the flow is nearly one-dimensional. The flow structure in these regions can be evaluated by the variation of stretching directions. In order to find out the stretching ratio, the change of stretching directions must be defined. Let $\boldsymbol{\alpha}_i$, $\boldsymbol{\alpha}_j$, and $\boldsymbol{\alpha}_k$ be the stretching directions of the three nodes i, j, and k of an element e, and ϕ_i, ϕ_j, and ϕ_k are the corresponding angles defined by Eq. (2.4). The configuration is shown in Fig. 1. According to this figure, the variation of stretching directions between nodes i and j, Δ_{ij}, is obtained by the expression

$$\Delta_{ij} = \text{Min}(\theta_1, \theta_2), \tag{2.5}$$

where

$$\theta_1 = |\phi_i - \phi_j| \tag{2.6}$$

$$\theta_2 = |180° - \theta_1|. \tag{2.7}$$

Similarly, Δ_{jk} and Δ_{ki} are evaluated. The change of stretching direction of the element e is found by the average of Δ_{ij}, Δ_{jk}, and Δ_{ki}, i.e.,

$$\Delta_e = (\Delta_{ij} + \Delta_{jk} + \Delta_{ki})/3. \tag{2.8}$$

After Δ_e is computed for all elements, the variation of the angle ϕ, $(\Delta\phi)_i$, is calculated for all nodes by the area-weighted average:

$$(\Delta\phi)_i = \frac{\sum_{e=1}^{\text{nbe}} A_e \cdot \Delta_e}{\sum_{e=1}^{\text{nbe}} A_e}. \tag{2.9}$$

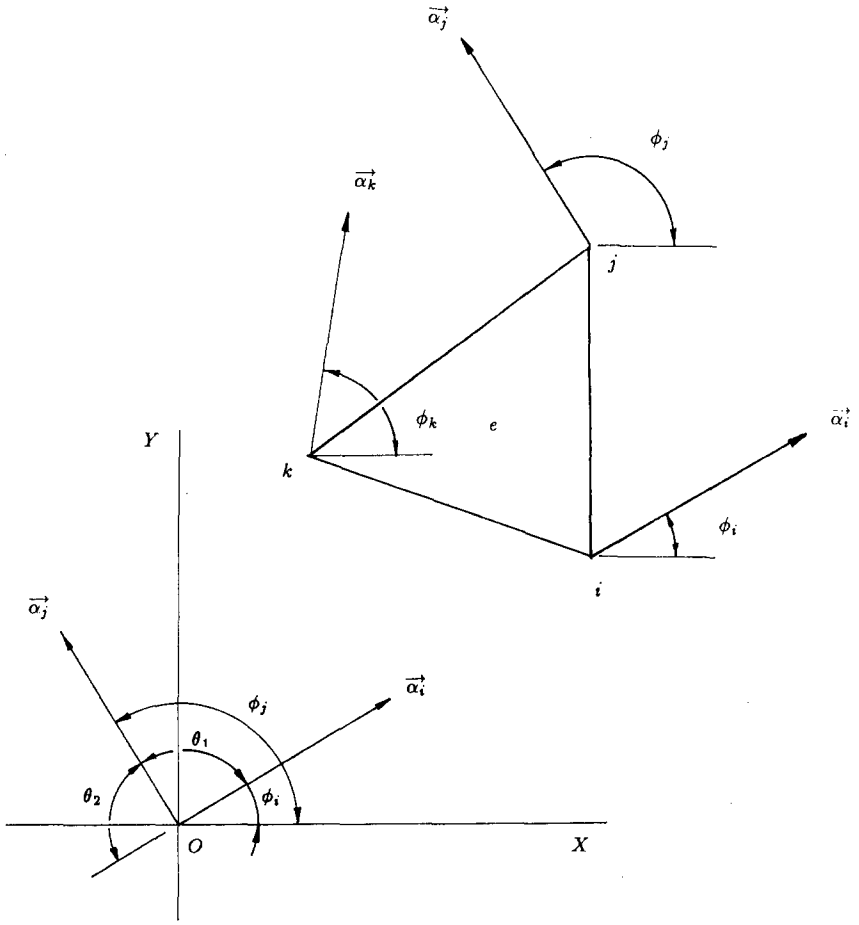


FIG. 1. Geometrical definitions of $\boldsymbol{\alpha}_i$, $\boldsymbol{\alpha}_j$, $\boldsymbol{\alpha}_k$, ϕ_i, ϕ_j, ϕ_k, θ_1, and θ_2.

Where A_e is the area of element e, and nbe represents the number of elements which surround the node i. Similar to the node spacing in Eq. (2.1), the stretching ratio at node i, S_i, is obtained by the relations

$$S_i^2(\Delta\phi)_i = \text{const} \tag{2.10}$$

and

$$S_i = \begin{cases} S_{\min} & \text{if } S_i < S_{\min}, \\ S_i & \text{if } S_{\min} \leqslant S_i \leqslant S_{\max}, \\ S_{\max} & \text{if } S_i > S_{\max}, \end{cases} \tag{2.11}$$

where the constant in Eq. (2.10) is taken as $S_{\min}^2(\Delta\phi)_{\text{avg}}$, and $(\Delta\phi)_{\text{avg}}$ is the average value of $(\Delta\phi)_i$ over the whole mesh. $S_{\min}$ and $S_{\max}$ are the specified values of minimum and maximum stretching ratios. In this paper, $S_{\min}$ is chosen as 1.0.

3. MESH GENERATION

The present mesh generation method is composed of three steps: construction of background grid, generation of boundary and interior nodes, and formation of triangles by front concept. For the distribution of interior points, two kinds of node-clustering techniques are suggested respectively for the regular and stretched elements. Concerning the triangulation, two methods, which utilize the concepts of side-based and vertex-based fronts, are presented. The first method is an extension of Lo's approach [4], so that both the regular (stretching ratio equal to one) and stretched (stretching ratio greater than one) triangles can be formed. In the second method, the front is represented by the vertices instead of the sides. Basically, the first method forms one triangle in one process, but the second method may generate more than one triangle in one process. Unlike the usual mesh generation methods, no smoothing process is operated in this work after the step of triangulation is accomplished. After finishing the triangulation, the mixed type of meshes, which include triangles and quadrilaterals, can be achieved by combining two triangles into a quadrilateral. The details of the mesh generation technique are described as follows:

3.1. *Background Grid*

The construction of the background grid has been introduced in Ref. [6]. This grid, which is required to cover the solution domain completely, is used to provide a piecewise linear spatial distribution of the three important remeshing parameters δ, α, and S. The parameters must be specified or computed from the solution for each node on the background grid.

3.2. *Generation of Nodes*

The computational domain is defined by several boundaries which are composed of a series of points. Those points

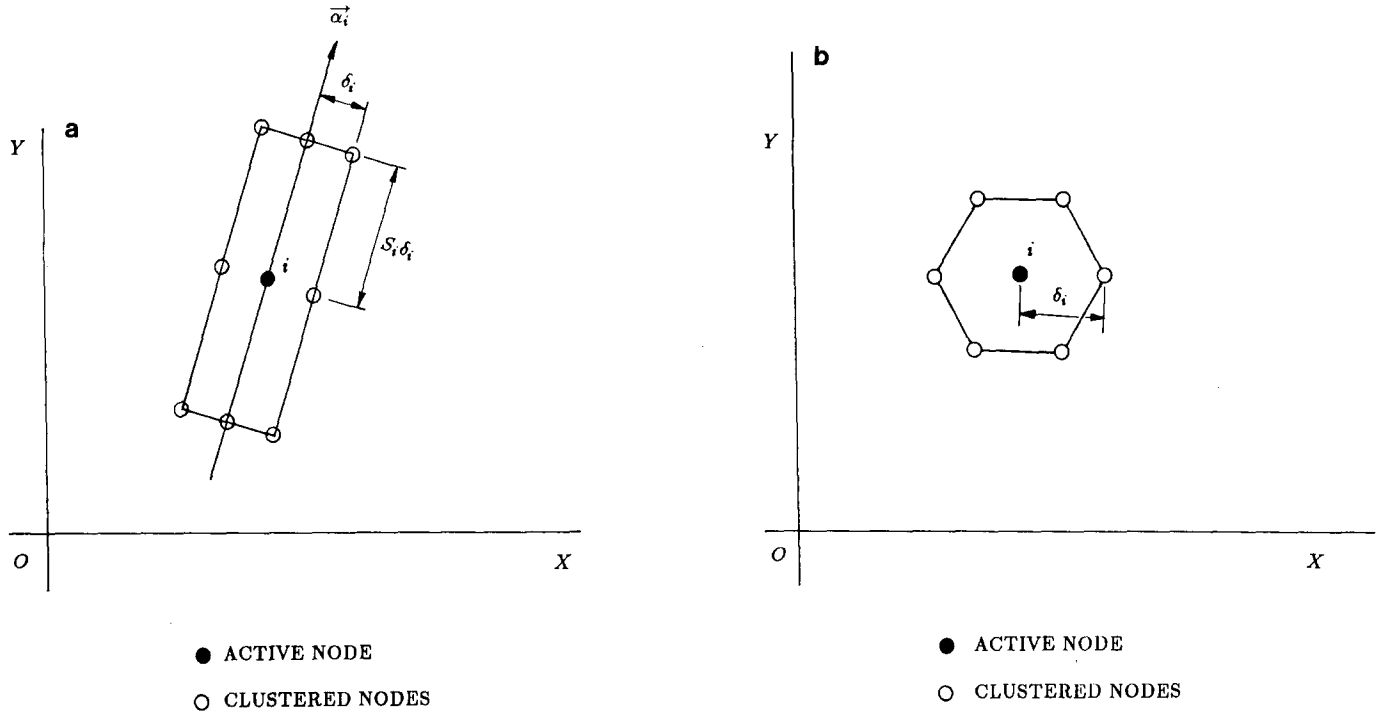


FIG. 2. Generation of clustering nodes around the active node i: (a) stretched mesh; (b) regular mesh.

are specified counterclockwise for external boundaries and clockwise for internal boundaries. Boundary nodes are added according to the information provided by the remeshing parameters on the background grid. For curved boundaries, the end points are specified and the database with fine-enough spacing (for example, 500 or 1000 points for upper surface of an airfoil) is used to generate the new nodes.

Starting from boundary nodes, interior points are generated by clustering new nodes around the existing ones. At the beginning of the process, all existing nodes are termed active. It means that those nodes are available for clustering new grid points. The following steps are involved in the distribution of interior points:

(a) Choose an active node i as a base. For obtaining the better distribution of grids, it is suggested that the node with smallest spacing is processed first.

(b) To create the stretched elements, eight nodes are clustered around the node i by the way shown in Fig. 2a. For generation of regular mesh, another approach shown in Fig. 2b is used. The rectangle with eight nodes is replaced by the hexagon with six nodes. If any of those clustered nodes is too close to any existing nodes or locates itself out of the domain, the node is deleted from the list of clustered nodes. The remaining clustered nodes are taken as the new nodes.

(c) Add those new nodes into the list of active nodes. Based on the values on the background grid, the remeshing parameters (δ_j^{int}, S_j^{int}, and $\boldsymbol{\alpha}_j^{\text{int}}$) for each new node are obtained by linear interpolation. If a smooth transition of grids in areas with an abrupt change of node spacing is needed, the values of δ_j^{int} at the clustered node j is evaluated by

$$\delta_j^{\text{new}} = w\delta_j^{\text{int}} + (1-w)\,\delta_i \tag{3.1}$$

where δ_i is the spacing of the node i, and δ_j^{new} is the new value of the parameter at node j. The factor w is less than 1.0 and the value is decided by the need of smoothness of the resulting mesh.

When the above clustering procedures are finished, the node i is removed from the list of active nodes. The creation of interior nodes is completely achieved as the number of active nodes is reduced to zero.

3.3 *Triangulation by Side-Based Front*

The procedures of triangulation are basically based on Lo's approach [4], and some extensions have been made to form triangles that are directionally stretched. Initially, the front is defined by the collection of boundary segments. Any side on the front is called the "active side," which is available for forming triangles. Starting from the initial front, triangular elements are generated in the following steps:

(a) Select an active side, which has the smallest spacing, as the acting side. Let the end nodes be denoted by N_a and N_b, and the remeshing parameters of this acting side are δ_s, $\boldsymbol{\alpha}_s$, and S_s. Make the local coordinates $(\tilde{x}, \tilde{y})$ at the midpoint of the side, such that $\tilde{y}$ coordinate is aligned with $\boldsymbol{\alpha}_s$ and scaled by a factor S_s.

(b) From the regions enclosed by the current front, find candidate nodes which lie at the left of the acting side. Sort candidate nodes by their distances to the midpoint in the local coordinates.

(c) From the sorted candidate nodes, find the best node N_c by Lo's formula [4] in the local coordinates. Form the triangle $N_aN_bN_c$ such that it does not contain any other candidate node and the line segments N_aN_c and N_bN_c do not intersect any existing sides. After forming the triangle,

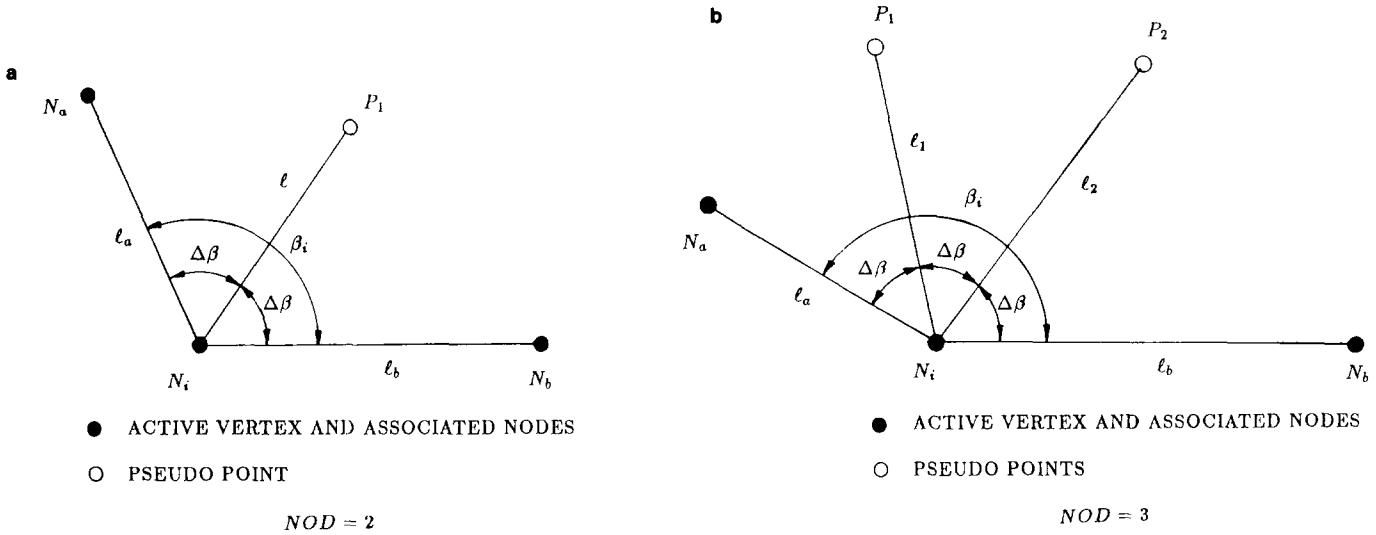


FIG. 3. Generation of pseudo points for the active vertex N_i: (a) $NOD = 2$; (b) $NOD = 3$.

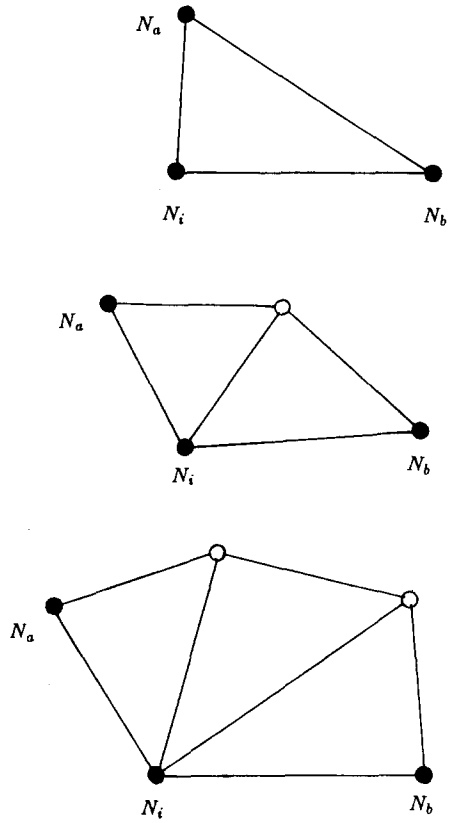


FIG. 4. Typical cases of triangulation for a vertex N_i.

the acting side N_aN_b is deleted from the list of active sides and the current front is updated.

The triangulation process stops if there is not any active side. Though this triangulation method is quite straightforward, it is still required to check the intersection of sides and make a transformation of the coordinates. In this paper, another approach, which avoids these cumbersome and inefficient actions, is developed. The details are described in the following part.

3.4. *Triangulation by Vertex-Based Front*

Instead of using boundary sides, the initial front is composed of boundary vertices. A vertex is defined by a common node of two consecutive segments which are located on the front. Initially, all vertices on the front are termed "active vertices", which are available for triangulation. The steps are described as follows:

(a) Choose an active vertex N_i which has the smallest vertex angle. As shown in Fig. 3, the associated nodes are denoted by N_a and N_b. The parameters β_i, l_a, and l_b represent the vertex angle, length of line segments N_aN_i, and N_iN_b.

(b) Define the parameter NOD (number of division) as the nearest integer of $\beta_i/(\pi/3)$. The $(NOD-1)$ "pseudo points" are located in the ways shown in Fig. 3. Positions of the pseudo points depend on the values of NOD, l_a, and l_b. If the value of NOD is not greater than one, no pseudo point is needed. When NOD is equal to two, one pseudo point is located with $l = w_l(l_a l_b)^{1/2}$ and $\Delta\beta = \beta_i/2$. For the case of $NOD = 3$, two pseudo points are generated with $l_1 = w_l(l_a^2 l_b)^{1/3}$, $l_2 = w_l(l_a l_b^2)^{1/3}$, and $\Delta\beta = \beta_i/3$ (see Fig. 3b). The formulas to generate the pseudo points are similar to those of Ref. [16]. The factor w_l, which is typically less than 1.0, is used to control the positions of the pseudo points.

(c) For each pseudo point, the node which is the nearest to that point is selected. Triangles can be formed by using those selected nodes. Some typical cases are shown in Fig. 4. If any of the newly formed triangular elements contains any nodes, it is decoupled into several triangles, such as the cases shown in Fig. 5. The front is updated by deleting the current vertex and adding new ones to the list of active vertices.

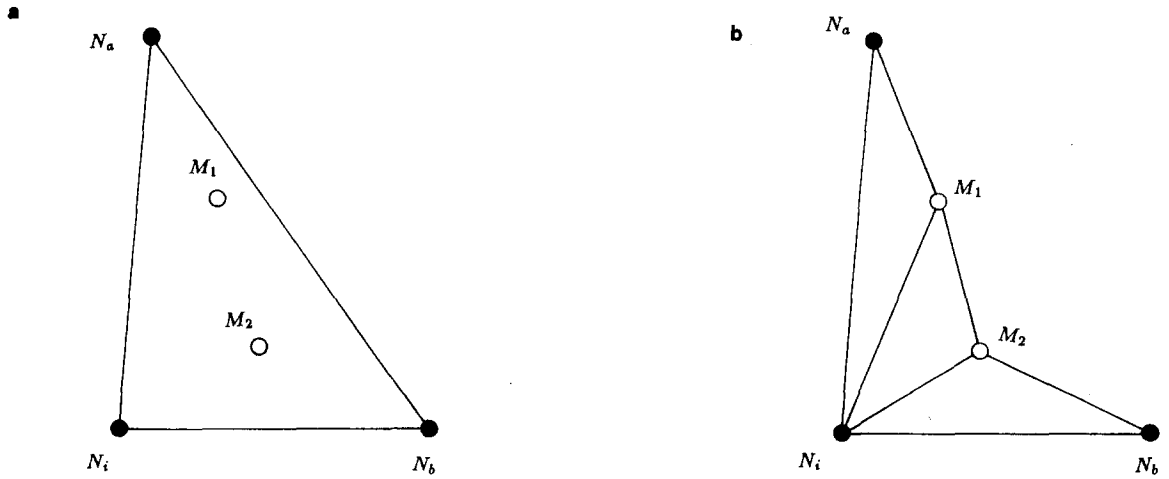


FIG. 5. Reformation of the triangle which includes nodes: (a) Original triangle; (b) new configuration of triangles.

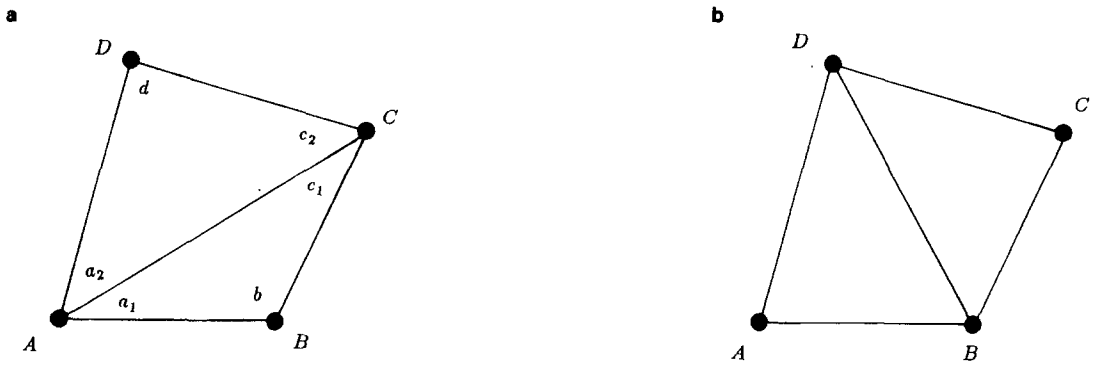


FIG. 6. Reformation of those triangles which are badly formed: (a) original configuration; (b) new configuration.

The triangulation stops when the number of vertices on the front is reduced to zero. If any of triangles is badly formed, a post-processor is activated. As shown in Fig. 6, the triangles ABC and ACD are reformed into triangles ABD and BCD if

$$\text{Max}(b, d) > (a_1 + a_2)$$

and

$$\text{Max}(b, d) > (c_1 + c_2),$$

where a_1, a_2, b, c_1, c_2, and d are the angles of triangular elements ABC and ACD. Unlike the side-based triangulation technique, the vertex-based approach can be used to generate triangles without local coordinate transformation. According to the numerical experiments, it is not necessary to check the intersection of sides if w_l is less than 0.5. Based on the CPU time, a primary comparison of those two triangulation approaches is shown in Section 6. The vertex-based front technique is much more efficient.

3.5. *Mixed Type of Triangular and Quadrilateral Meshes*

If the common side of any two triangles is the longest side of those two triangles, they are combined into a new quadrilateral. After the combination, only data about the elements are changed and the nodal data remain the same.

4. IMPLEMENTATION OF LOCAL REMESHING

All remeshing procedures try to redistribute grid points in the optimum way when numerical solutions on the present grid are obtained. If the steady-state problems are solved, the regeneration of mesh is done only a few times, so that speed of the grid adaptation is not a criticial factor. For the unsteady flows, meshes are changed with time to simulate the unsteady phenomena, so that the mesh regeneration process has to be efficient. It seems too redundant to regenerate the whole mesh if the flow properties change in limited regions. Lohner [14] combined the global remeshing with conventional h-refinement for strongly unsteady flows, and he suggested that partial remeshing, particularly if the shocks change direction, is an area which deserves further study.

The global remeshing starts to form triangles from the initial front composed of boundary segments. In the proposed local remeshing, the concept of the vertex-based front is employed. Some elements are recovered from the background grid and the front is advancing at the same time. Due to this recovering process, the flow region to be regenerated is reduced. Details of the recovering procedure are described as follows:

(a) By using the mesh generation procedure, a new mesh, and the associated parameters, δ_i^o, ϕ_i^o, and S_i^o, are obtained from the previous mesh. On the current mesh, a solution is computed. Based on this solution, the remeshing parameters, δ_i^n, ϕ_i^n, and S_i^n, are evaluated. It should be mentioned that those parameters with superscripts o and n represent old and new values on the same mesh.

(b) The elements with node i must be regenerated if any one of the relations

$$\text{Min}(|\phi_i^n - \phi_i^o|, 180 - |\phi_i^n - \phi_i^o|) > \phi_{\text{crit}} \tag{4.1}$$

$$\frac{S_i^n}{S_i^o} < K_1 \quad \text{or} \quad \frac{S_i^n}{S_i^o} > K_2 \tag{4.2}$$

$$\frac{\delta_i^n}{\delta_i^o} < K_1 \quad \text{or} \quad \frac{\delta_i^n}{\delta_i^o} > K_2, \tag{4.3}$$

is satisfied, where ϕ_{crit} is the specified tolerance of variation in stretching direction, and K_1 and K_2 are constants. In this work, the values of ϕ_{crit}, K_1 and K_2 are taken as 10°, 0.9 and 1.5. If the element is not to be regenerated, it is put into the recovering list.

(c) Starting from a node on the initial front, those elements, which contain this node and belong to the recovering list, are directly picked up as the new ones

without changing the original configurations. This process is continued node by node, and the front is updated untill all the elements in the recovering list have been checked.

After the recovered front is obtained, the procedures of global remeshing are applied to regions enclosed by this front.

Since the time required for recovering elements and nodes is much less than that of the other procedures, more CPU time can be reduced if more elements are recovered. To indicate the performance of the local remeshing, the recovery rate of nodes (RR) is defined and expressed by the following form:

$$RR = \frac{\text{No. of recovered nodes}}{\text{Total no. of nodes}}. \tag{4.4}$$

5. SOLUTION ALGORITHM

In this study, the conservative form of two-dimensional Euler equations is solved. About the spatial discretization, the Galerkin approach is employed. Linear and bilinear shape functions are used for triangular and quadrilateral elements, respectively. To suppress the numerical oscillations, the second- and fourth-order dissipation terms [3] are introduced. For steady-state problems, a four-step Runge–Kutta time integration method with local time stepping [17] is used to obtain a fast convergence. For shock propagation problem, two-step Runge–Kutta time integration method [18] is applied to obtain solutions with second-order time accuracy.

6. RESULTS AND DISCUSSION

In order to demonstrate the performance of the proposed grid adaptation algorithms, some numerical examples, which include steady, unsteady, external, and internal flow problems, are presented.

6.1. *Comparison of Two Triangulation Approaches*

To compare the performance of the proposed triangulation procedures, two regular meshes which have the same nodal distributions in the square domain are respectively generated by the side-based and vertex-based fronts. From those numerical results in Fig. 7, the CPU time (run on VAX 8600) of side-based triangulation grows much faster than that of the vertex-based triangulation, when the number of elements increases.

6.2. *Supersonic Flow over a Compression Corner*

The first problem investigated in this paper is a supersonic flow ($M_\infty = 3$) over a compression corner with 15°.

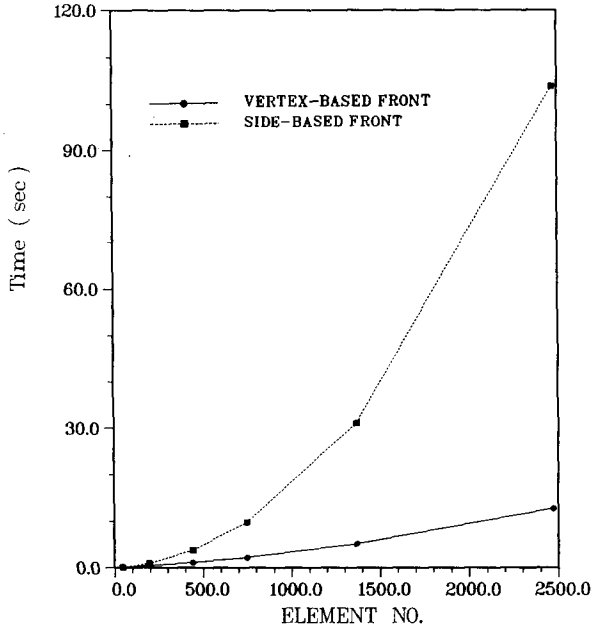


FIG. 7. Comparison of CPU time for side-based and vertex-based triangulations.

To confirm the availability of the present adaptive mesh generation technique and understand the effect of stretching meshes on the numerical solutions, the global remeshing about the regular triangles, stretched triangles, and mixed type of stretched triangles and quadrilaterals is employed to compare with conventional h-refinement. By using the triangular elements, the initial mesh and corresponding pressure contours are shown in Fig. 8. About the isobaric lines in Figs. 8–12, the maximum and minimum values of pressure are 0.7 and 2.0, respectively, and the increment is 0.05. By dividing one triangle into four small ones in high gradient regions, the final grid system and the pressure contours, which are obtained after three mesh refinements, are presented in Fig. 9. It is obvious that a large number of elements are added in the shock region. To obtain the pressure contours which are similar to the results on the final mesh in Fig. 9, two successive remeshings are operated. The sequence of meshes with regular triangles and the corresponding contours are shown in Fig. 10. It is apparent that the adaptive mesh regeneration technique can provide reasonable grid systems and is more flexible than the conventional mesh refinement approach. By introducing the stretched triangles and mixed type of elements, the meshes and related results are given in Figs. 11 and 12. The comparison of results in Figs. 10 and 11 indicates that the node numbers of stretched meshes are about half of those of regular meshes. As shown in Figs. 11 and 12, the meshes in the shock regions are directionally stretched and the grid lines are almost aligned with the shock. This fact demonstrates that the present error indicator and mesh

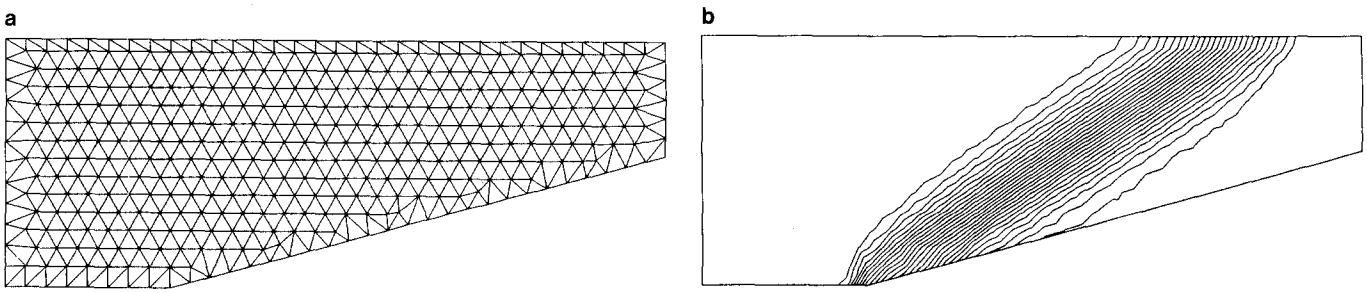


FIG. 8. Supersonic flow over a compression corner (triangular elements) : (a) initial mesh; (b) pressure contours (minimum = 0.7, maximum = 2.0, increment = 0.05).

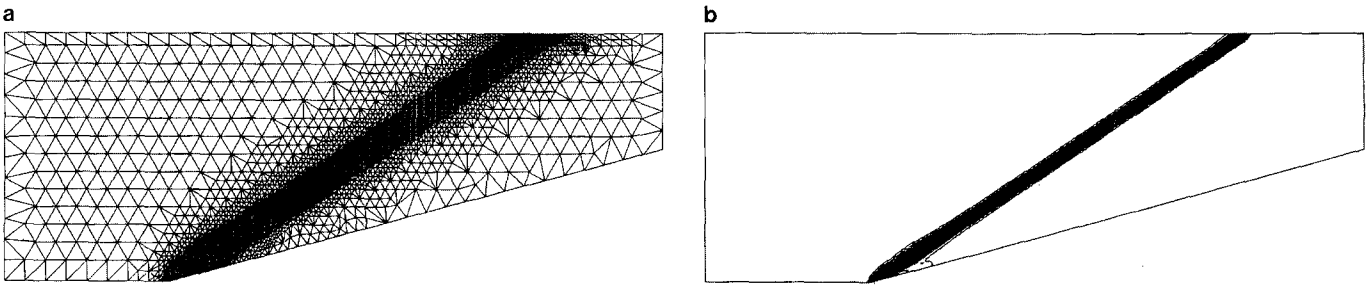


FIG. 9. Supersonic flow over a compression corner (adaptive refinement): (a) final mesh; (b) pressure contours (minimum = 0.7, maximum = 2.0, increment = 0.05).

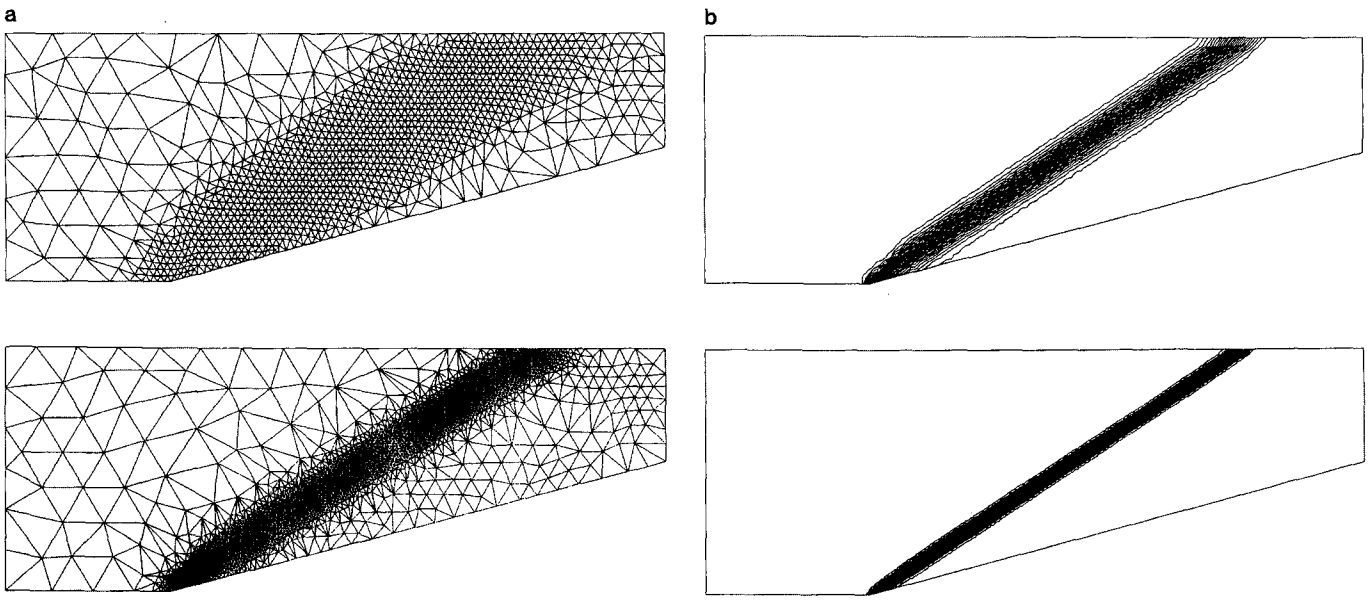
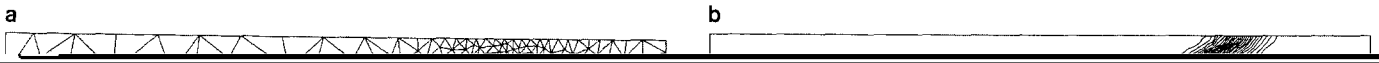


FIG. 10. Supersonic flow over a compression corner (adaptive remeshing, regular triangles): (a) the sequence of meshes; (b) pressure contours (minimum = 0.7, maximum = 2.0, increment = 0.05).


a
b

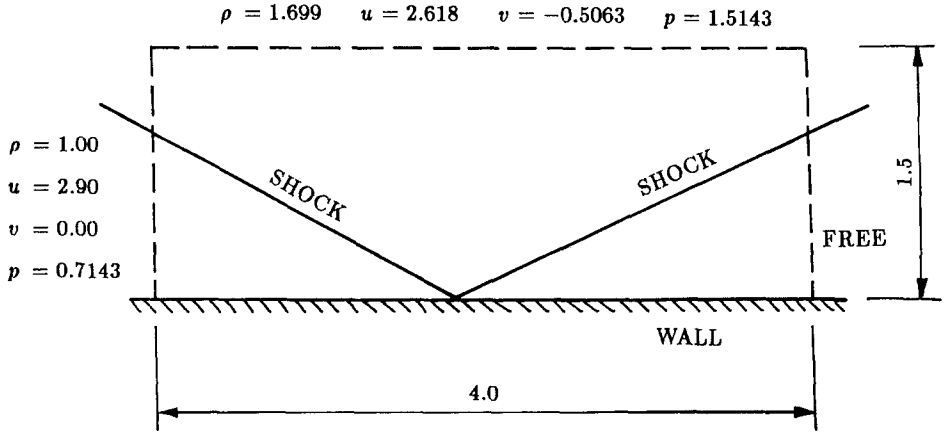


FIG. 13. Shock reflection at a wall: The problem definition and the computational domain.

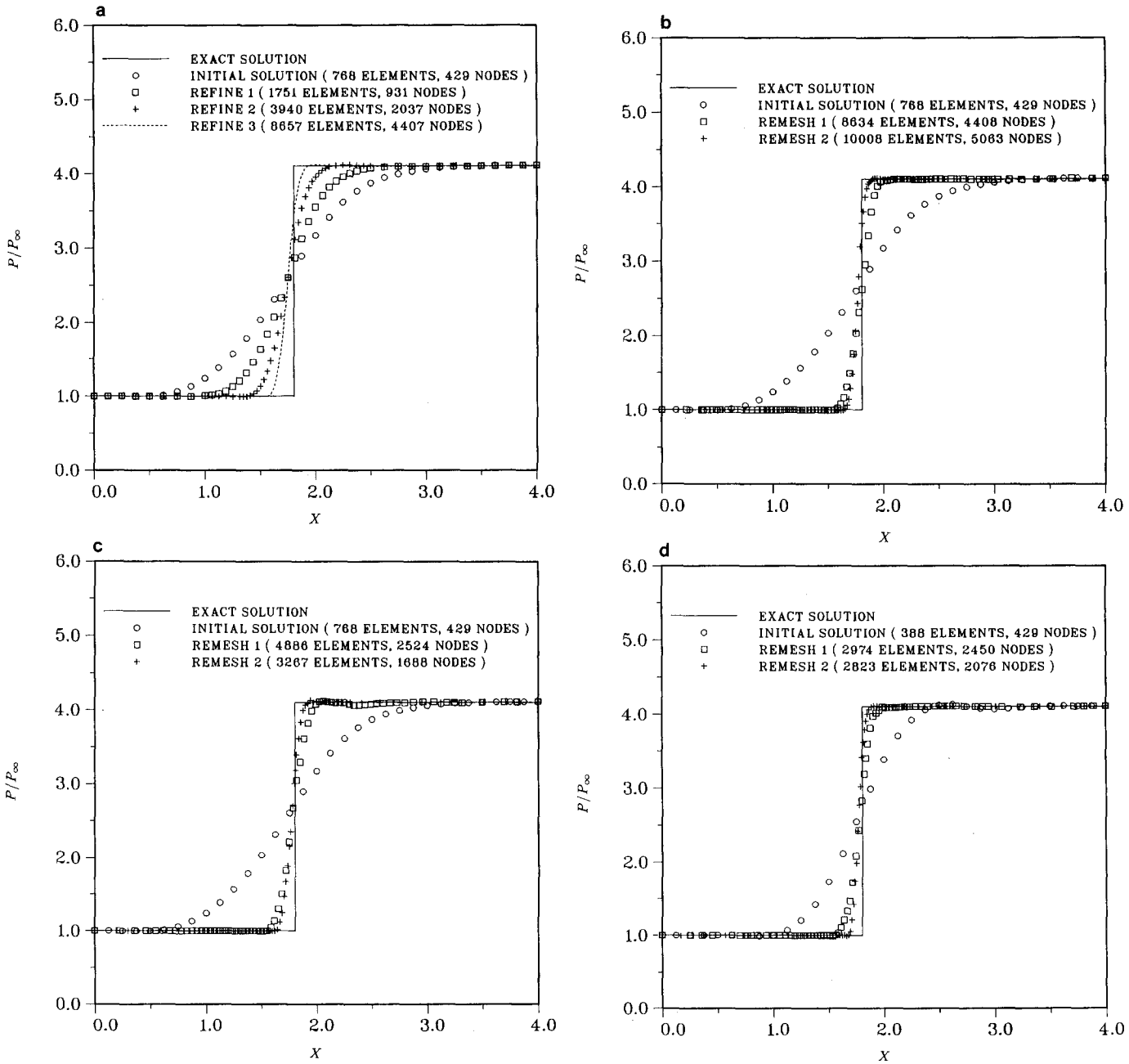


FIG. 14. Wall pressure distributions of shock reflection at a wall: (a) refinement; (b) remeshing (regular triangles); (c) remeshing (stretched triangles); (d) remeshing (mixed type of stretched elements).

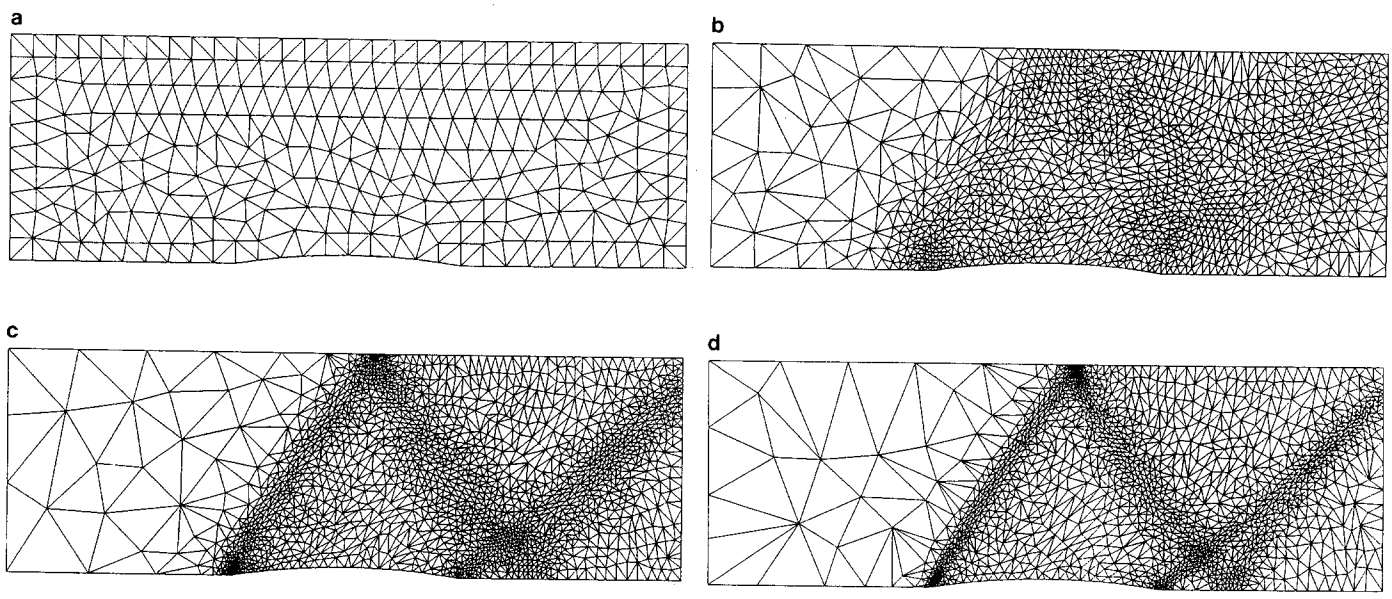


FIG. 15. The sequence of meshes for the supersonic flow ($M_\infty = 1.4$) passing through a channel with a 4% circular arc bump: (a) 526 elements, 304 nodes; (b) 2523 elements, 1332 nodes; (c) 3422 elements, 1786 nodes; (d) 2941 elements, 1539 nodes.

generation technique are reliable and suitable for treating the flow problems with one-dimensional features.

6.3. *Shock Reflection at a Wall*

To further evaluate the remeshing procedure, the second problem, an oblique shock reflection at a wall with inlet Mach number 2.9 is studied. The problem definition and computational domain are depicted in Fig. 13. By using three successive refinements, the resulting pressure distributions along the wall surface are given in Fig. 14a. When the global remeshing technique is applied to obtain regular triangles, stretched triangles, and mixed type of stretched elements respectively, the corresponding pressure distributions are plotted in Figs. 14b–d. Comparing with the

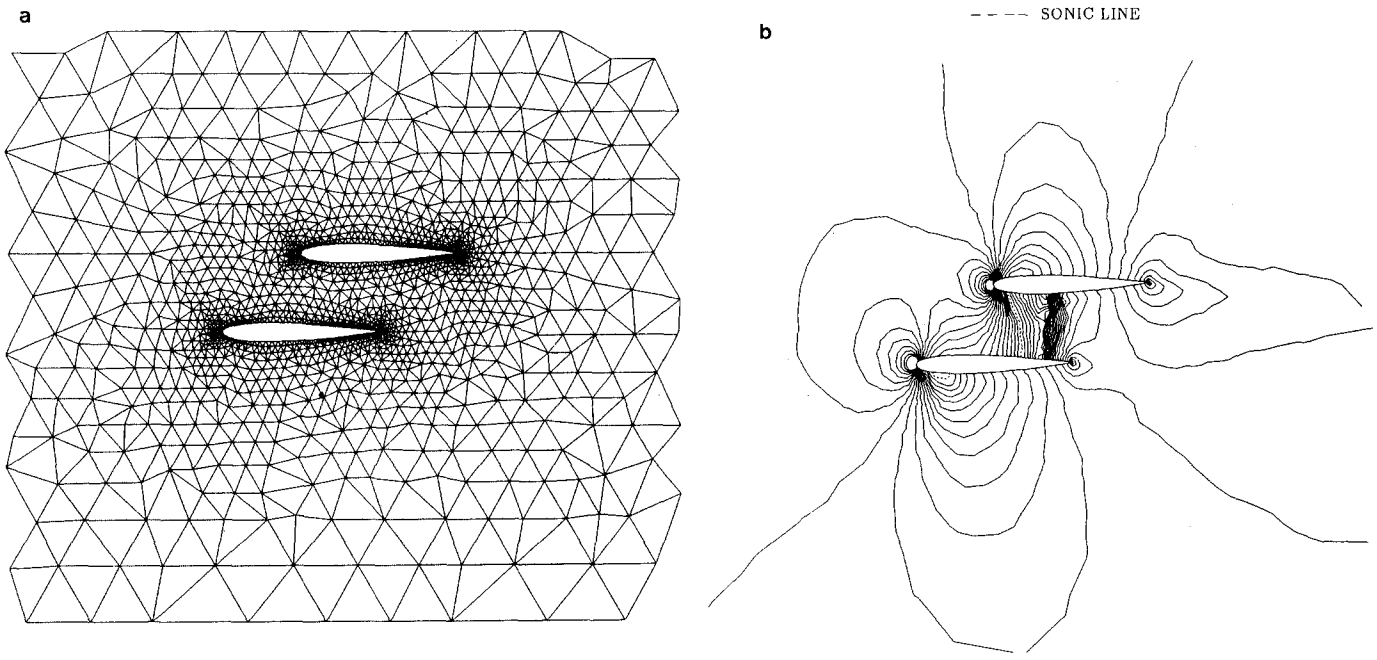


FIG. 17. (a) Initial mesh; (b) the corresponding Mach number contours (increment = 0.025) for the transonic flow ($M_\infty = 0.7$) around a two-element airfoil.

and 3544 elements, and the corresponding Mach number contours are shown in Fig. 17. The results after three global remeshings are plotted in Fig. 18. From this final mesh (2495 nodes and 4813 elements) and the Mach number contours, the resolution of the results around the shock is improved.

6.6. *Shock Propagation in a Channel*

The availability of the present local remeshing method in solving transient problems is evidenced by this example. The shock with a shock Mach number 1.4 moves to right side of a channel from the position $X = 0.0$ at $T = 0.0$.

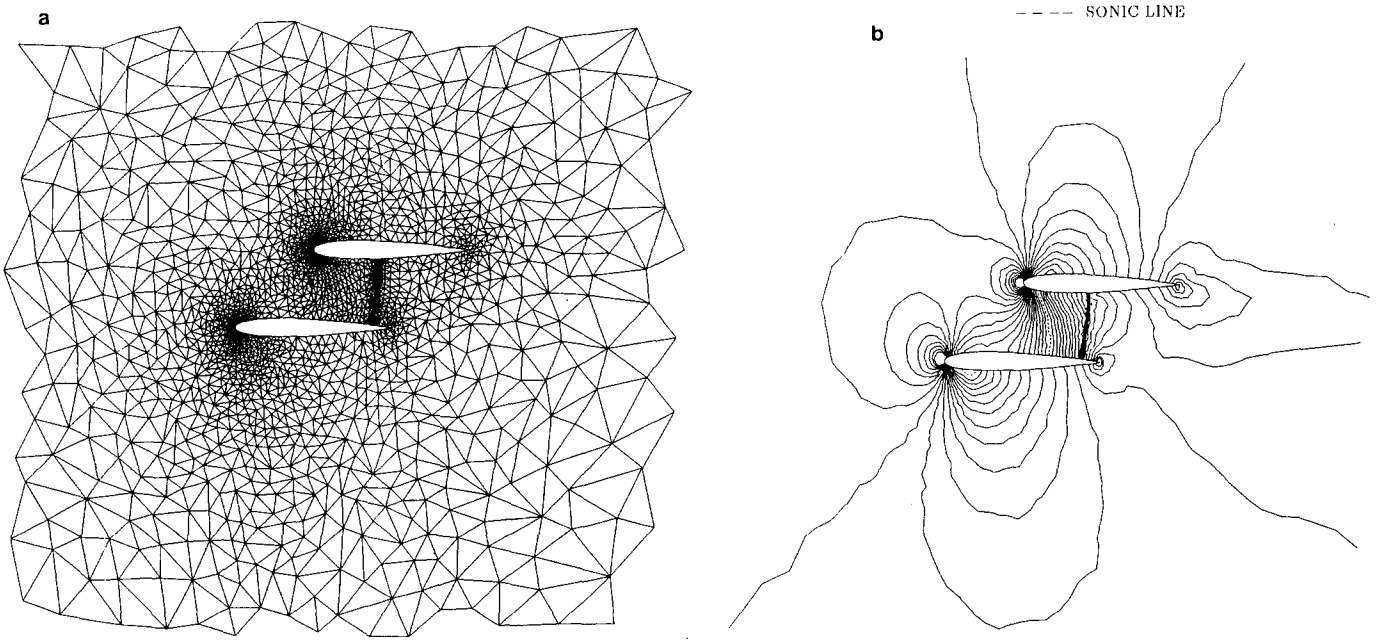


FIG. 18. (a) Final mesh; (b) the corresponding Mach number contours (increment = 0.025) for the transonic flow ($M_\infty = 0.7$) around a two-element airfoil.

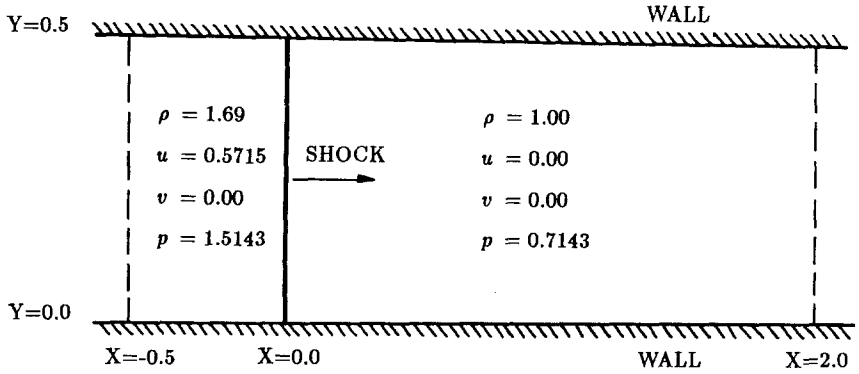


FIG. 19. Shock propagation in a channel: The problem definition and the computational domain.

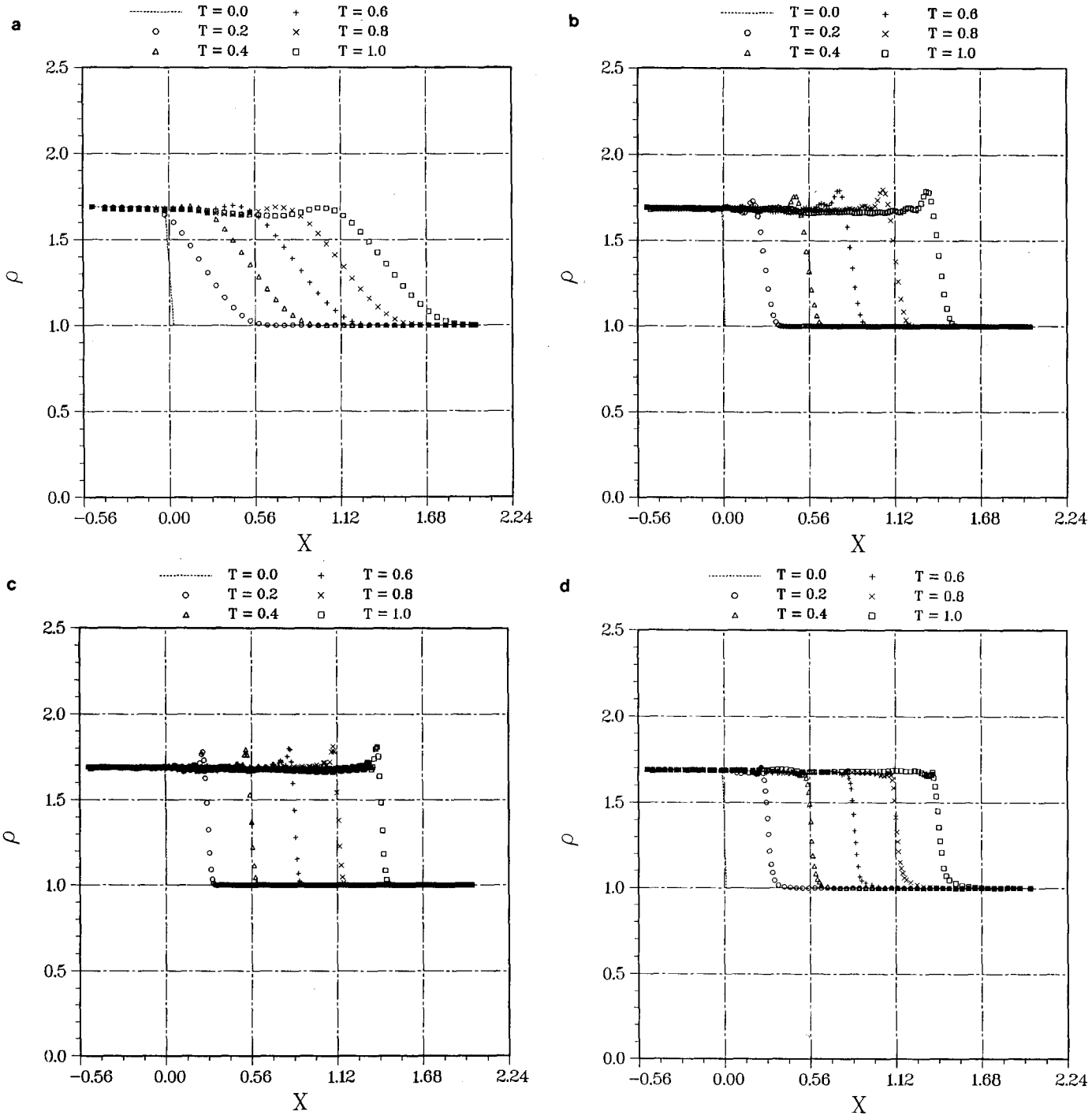


FIG. 20. Density distributions along the lower wall surface for the shock propagation in a channel on the fixed meshes with (a) 810, (b) 9000, (c) 32,490 regular elements, and (d) adaptive meshes with 2073 elements (averaged).

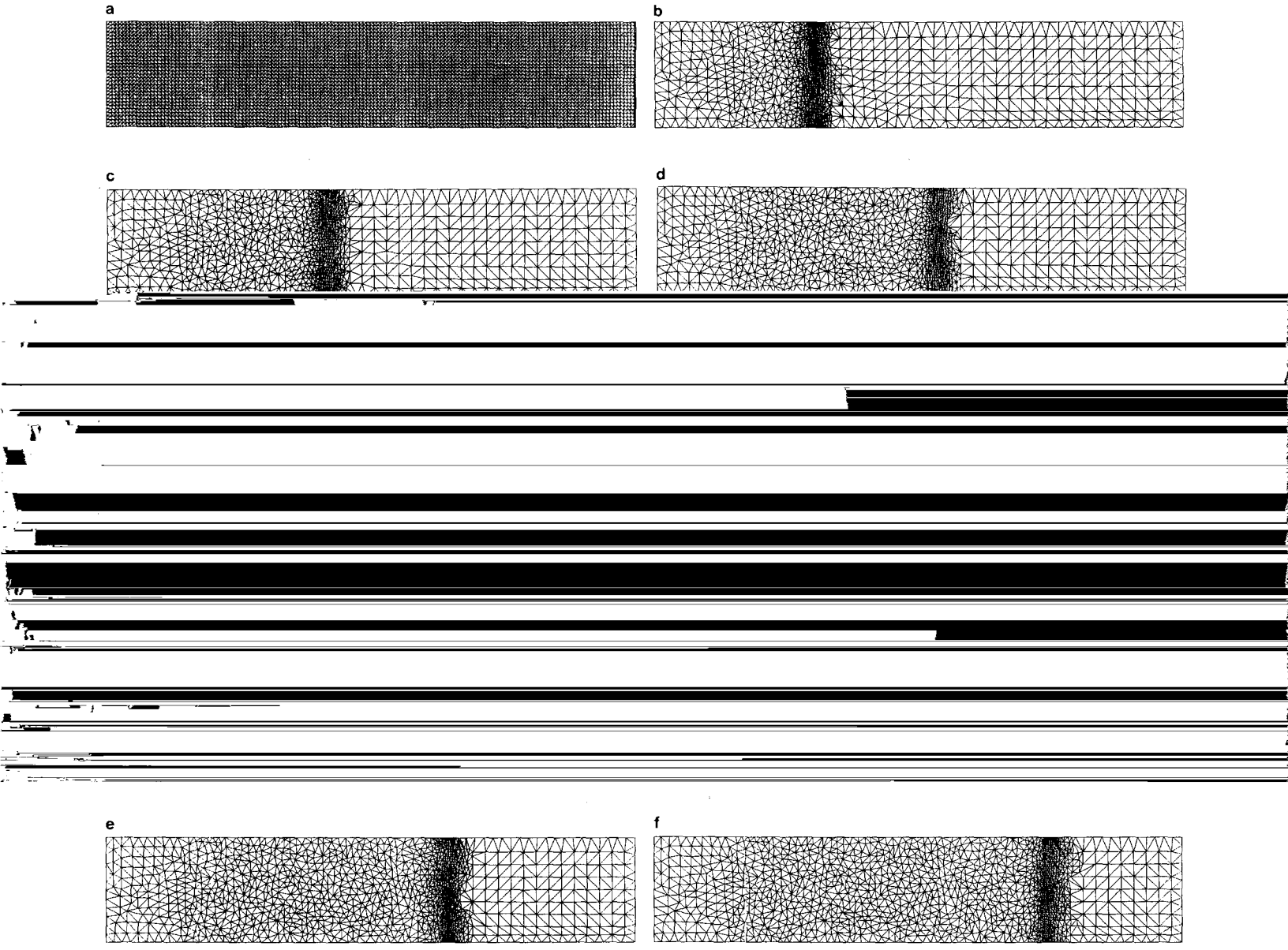
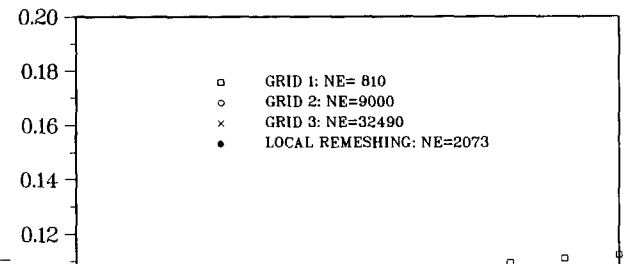


FIG. 21. Locally regenerated meshes for the shock propagation in a channel at (a) $T=0.0$, (b) $T=0.2$, (c) $T=0.4$, (d) $T=0.6$, (e) $T=0.8$, and (f) $T=1.0$.

Figure 19 shows the definition and computational domain of this problem. To understand the characteristics of local remeshing approach, the time-varying meshes, and three fixed meshes which contain 810, 9000, and 32,490 elements, respectively, are used. In this example, the time-varying meshes are locally regenerated every 10 time-steps. The minimum and maximum nodal spacings are equal to those used in the finest and coarsest fixed meshes, respectively.



The solutions on all meshes are achieved by advancing at the constant time step corresponding to the Courant number of 4.0 based on the minimum nodal spacing. For the fixed and adaptive grid systems, the density distributions along lower wall at $T=0.0$, 0.2, 0.4, 0.6, 0.8, and 1.0 are plotted in Fig. 20, and the corresponding time-varying meshes are shown in Fig. 21. The average number of elements of the adaptive meshes is 2073, and the average recovery rate is 65.66%. Comparing with the exact solutions, the numerical oscillations are larger and increase with time if the meshes are fixed. To quantitatively evaluate the accuracy of present solutions, the "error" of the numerical results is defined by the expression [20],

$$|\mathrm{err}| = \sqrt{\sum_{i=1}^{NP} (\hat{\rho}_i - p_i)^2 / NP}, \tag{6.1}$$

where ρ_i and $\hat{\rho}_i$ are the exact and numerical solutions

Fig. 21f, the CPU time for recovering 1028 nodes and 1729 elements is 3.07 s on the VAX 8600 system, and it takes 10.27 s to generate 322 new nodes and 783 elements. From the above discussion, the present local remeshing technique is reliable, and it provides an important and efficient way to study the unsteady problems.

7. CONCLUSIONS

A new adaptive remeshing approach for unstructured meshes, which includes the error indicator, global and local mesh regeneration techniques, has been presented. Based on the first derivatives of a key variable, an error indicator is developed to decide the remeshing parameters. In this approach, nodes are first distributed according to the remeshing parameters, and those nodes are connected into a complete mesh. Distribution of nodes are successfully achieved by the proposed two kinds of node-clustering techniques. The side-based and vertex-based fronts are introduced for triangulation, and the vertex-based triangulation technique is shown to be more efficient. By using vertex-based triangulation approach, a local remeshing method for shock propagation in a channel is presented. Several compressible flow problems are investigated to demonstrate the reliability of the proposed procedure. The regular/stretched triangles and the mixed type of triangular and quadrilateral stretched elements are used. From the numerical results, the approaches, which employ the directionally stretched elements, are effective and suitable for treating the flow problems with one-dimensional features. The development of a local remeshing algorithm, which provides a possible and efficient way to couple the time-varying meshes with flow solver for unsteady flows, is worthwhile and important. For unsteady flows, further study on the interpolation, error indicator, and criterion of elements recovery is useful to enhance the performance of the local remeshing algorithm.

REFERENCES

1. P. J. Green and R. Sibson, *Comput. J.* **21**, 168 (1978).
2. N. P. Weatherill, *Int. J. Numer. Methods Fluids* **8**, 181 (1988).
3. D. J. Mavriplis, *AIAA J.* **28** (2), 213 (1990).
4. S. H. Lo, *Int. J. Numer. Methods Eng.* **21**, 1403 (1985).
5. S. H. Lo, *Int. J. Numer. Methods Eng.* **28**, 2695 (1989).
6. J. Peraire, M. Vahdati, K. Morgan, and O. C. Zienkiewicz, *J. Comput. Phys.* **72**, 449 (1987).
7. R. Lohner, K. Morgan, and O. C. Zienkiewicz, "Adaptive Grid Refinement for the Compressible Euler Equations," in *Accuracy Estimates and Adaptive Refinements in Finite Element Computations*, edited by I. Babuska *et al.* (Wiley, New York, 1986), p. 281.
8. J. T. Oden, T. Strouboulis, and P. Devloo, *Comput. Methods Appl. Mech. Eng.* **59**, 327 (1986).
9. R. Lohner, *Comput. Methods Appl. Mech. Eng.* **61**, 323 (1987).
10. J. F. Dannenhoffer and J. R. Baron, AIAA Paper 85-0484, 1985 (unpublished).
11. R. Lohner, K. Morgan, and O. C. Zienkiewicz, *Comput. Methods Appl. Mech. Eng.* **51**, 441 (1985).
12. E. A. Thornton and G. R. Vemaganti, AIAA Paper 88-2662, 1988 (unpublished).
13. G. Vemaganti, E. Thornton, and A. Wieting, AIAA Paper 90-0401, 1990 (unpublished).
14. R. Lohner, *Comput. Methods Appl. Mech. Eng.* **75**, 195 (1989) (unpublished).
15. J. Probert, O. Hassan, J. Peraire, and K. Morgan, Proceedings of the Fifth Symposium on Numerical Methods in Engineering, Lausanne, Switzerland, 1989, pp. 801–808.
16. E. A. Sadek, *Int. J. Numer. Methods Eng.* **15**, 1813 (1980).
17. J. Peraire, K. Morgan, J. Peiro, and O. C. Zienkiewicz, AIAA Paper 87-0558, 1987 (unpublished).
18. C. W. Shu, *SIAM J. Numer. Anal.* **9** (6), 1073 (1988).
19. R. H. Ni, *AIAA J.* **20** (11), 1565 (1982).
20. M. J. Berger and J. Oliger, *J. Comput. Phys.* **53**, 484 (1984).