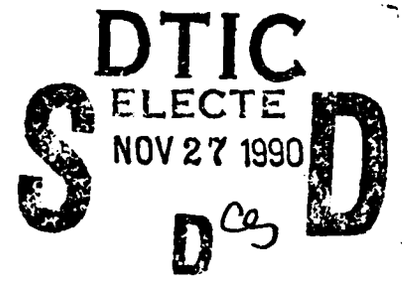


BRA-91-W001R

Development and Application of Numerical Models for Reactive Flows

Berkeley Research Associates, Inc.
P.O. Box 241
Berkeley, California 94701

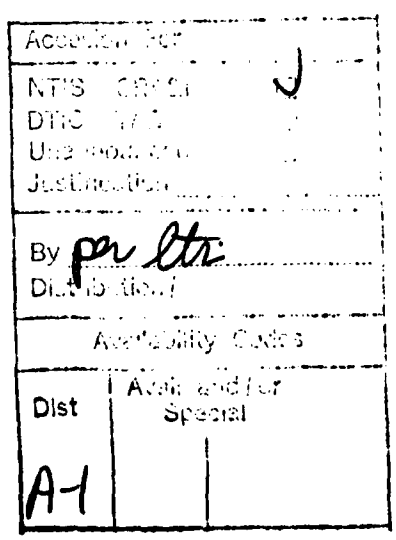
15 August 1990

Final Report

Contract No. N00014-86-C-2021

Prepared for
Laboratory for Computational Physics
Naval Research Laboratory
Washington, DC 20375-5000

TABLE OF CONTENTS



EXECUTIVE SUMMARY

This Final Report covers research performed under Contract No. N00014-86-C-2021 during the contract period 28 October 1985 to 30 September 1989. Berkeley Research Associates' efforts, in collaboration with the Laboratory for Computational Physics (LCP), have focused on developing mathematical and computational models which accurately and efficiently describe the behavior of complex physical systems such as: chemically reacting flows; transition from laminar to turbulent flow in reacting gas mixtures; laser-induced shock waves in an explosive medium; initiation, propagation, extinguishment and structure of laminar flames. The models, numerics and algorithms developed through this work are described in this document in response to task orders 2.1 thru 2.6.

In response to Task 2.1, Appendices A and B describe work carried out in extending the Flux-Corrected Transport (FCT) flow models to three dimensions and to generalized meshes. The extension to three dimensions required the development of surface definition software and sophisticated data structures to avoid excessive CPU-time overheads for the search operations involved. The hybrid method developed in combining the Finite-Element Method with the Flux-Corrected Transport method (FEM-FCT) is useful for solving problems involving steady and unsteday transonic and supersonic flow in irregular geometries.

The VOYEUR Graphics System is described in Appendix C in response to Task 2.2. It is used in conjunction with two instruments, namely a color hardcopy device (a Tektronix 4115 compatible printer/plotter, VOYEUR being implemented on a Tektronix 4115) or a DICOMED graphic output device. It is a menu driven system which builds up from a simple set-up to more complicated plots compiled from this initial information.

For Task 2.3, Appendices D thru H describe our work carried out in collaboration with LCP on transverse and axisymmetric jets, mixing enhancement in supersonic shear layers, unforced spatially-developing mixing layers and subgrid closures of premixed turbulent combustion. In Appendix D, the effect of small, random pressure fluctuations at the nozzle orifice on the growth of the mixing layer is examined. Also, the results from numerical simulations of the evolution of the Kelvin-Helmholtz instability for unforced, subsonic, compressible, axisymmetric, spatially-evolving shear layer are presented; these particular results are in response to Task 2.1. The numerical simulations of transverse jets into supersonic flows are described in Appendix E and are conducted by solving the conservation equations of mass, momentum, energy, and species densities using a fourth order FCT algorithm. Some of the fundamental questions concerning the definition of a supersonic shear layer and the effect of convective Mach number on the mixing and on the structure of the shear layer are presented and examined in Appendix F. It is shown that the convective Mach number describes the intrinsic character of the instability of a shear

layer. Mixing is enhanced when the convective Mach number is reduced. In response to both Tasks 2.1 and 2.3, in Appendix G two-dimensional numerical simulations of the Kelvin-Helmholtz instability in compressible, spatially-evolving, unforced, planar shear layers are used to investigate the reinitiation of unstable vortex roll-up near the trailing edge of the splitter plate. The process involved in roll-up reinitiation is examined. It is shown that spreading of the mixing layer through vortex merging depends strongly on the pressure field induced by the downstream events. Finally, in Appendix H, a review of the extensive literature on subgrid closures and other aspects of a LES of premixed turbulent combustion is presented.

In response to Task 2.4, Appendix I describes to use of time-dependent two-dimensional numerical simulations to study the effect of the rate of energy release on the regularity of the cellular structure of detonations in liquid nitromethane. Chemical decomposition of nitromethane is described by a two-step model composed of an induction time followed by energy release. The same expression is used for the induction time throughout the calculations. The energy release rate and its dependence on temperature were varied. A simplified equation of state based on the Walsh and Christian technique for condensed phases and the BKW equation of state for gas phases is used. When mixtures of both phases are present, pressure and temperature equilibrium is assumed. In the model, the walls of the detonation chamber are assumed to heavily confine the liquid explosive. Boundary layer effects are neglected. The solution thus simulates the detonation structure near the center of a wide channel. The simulations show that the energy release process controls whether the detonation dies, becomes one-dimensional, or becomes multidimensional. Once the multidimensional structure is established, its regularity is mainly controlled by the temperature dependence of the induction time, and to a lesser extent by the energy release rate. Increasing the energy release time and decreasing the activation energy generally improve the regularity. The simulations also show a correlation between the regularity of the cellular structure and both the change in induction zone thickness across the transverse waves and the shock front curvature. When the change in the width of the induction zone is larger, the shock has more curvature and the structure is more regular.

Task 2.5 deals with the development of implicit versions of FCT and the application of these algorithms to the detailed study of flames. Appendices J thru R deal extensively with this development. The importance of an implicit correction of the FCT implementation is that it removes the stringent limit on the timestep which explicit methods impose via the Courant-Friedrichs-Levy (CFL) condition. Appendix J deals with the barely implicit correction itself. Appendix K describes the detailed, two-dimensional model for flames, FLIC. It combines algorithms for subsonic convective transport with buoyancy, detailed chemical reaction processes, and diffusive transport processes such as molecular diffusion, thermal conduction, and viscosity. The study of the development of cellular structures in rich and lean hydrogen flames is presented in Appendix L. The model includes detailed hydrogen-

oxygen combustion with twenty-four elementary reactions of eight reactive species and a nitrogen diluent, molecular diffusion of all species, thermal conduction, and convection. In Appendices M and Q, the effect of gravity on flame instabilities is investigated. It is shown that the effects of gravity become more important as burning velocity is decreased which occurs as the lean flammability limit is approached. Gravitational effects are shown to be insignificant in the nonreacting jet. However, it the reacting jet, gravity produces the relatively low-frequency instabilities (eg. Kelvin-Helmholtz) typically associated with flame flicker. Appendix N is concerned with the time-dependent two-dimensional numerical simulations of a shock propagating through a compressible vortex. The scenarios presented are those of a strong shock (i.e. the fluid velocity behind the shock front is approximately the same as the maximum velocity in the vortex) and of a weak shock. Appendices O thru R deal with the numerical study of diffusion flames.

Finally, Task 2.6 entails the developement of algorithms for triangular meshes. The algorithm is described in Appendix S. In Appendix T, an adaptive finite element scheme for transient problems is presented. Examples involving shock-shock interactions and the impact of shocks on structures demonstate the performance of the method. Considerable savings in CPU-time and storage can be realized even for strongly unsteady flows. This is due to the method and the high degree of vectorizability that has been achieved on the CRAY-XMP-12 at the Naval Research Laboratory for this code. A high resolution finite element method for the solution of problems involving high speed compressible flows is presented in Appendix U. The method uses the concepts of FCT and is presented in a form which is suitable for implementation on completely unstructured triangular or tetrahedral meshes. Transient and steady state examples are solved to illustrate the performance of the algorithm. Appendix V gives one Berkeley Research Associates employee's view of the future direction of the use of finite elements in computational fluid dynamics. Appendices W and X describe the use of high resolution schemes for unstructured grids (FEM-FCT) and the importance in accurately modeling complicated geometries. Appendices X and Y also describes the use of the adaptive refinement method for triangular and tetrahedral meshes and give several numerical examples to prove its ability to accurately and efficiently handle difficult problems. Appendix Z describes the adaptive refinement method's application to transient problems.

APPENDIX A.

Generation of Three-Dimensional Unstructured Grids by the Advancing-Front Method (AIAA-88-0515)

GENERATION OF THREE-DIMENSIONAL UNSTRUCTURED GRIDS BY THE ADVANCING-FRONT METHOD

Rainald Löhner

Berkeley Research Associates
Springfield, VA 22150, USA
and
Laboratory for Computational Physics and Fluid Dynamics
Naval Research Laboratory
Washington, D.C. 20375, USA

and

Paresh Parikh

Vigyan Research Associates
Hampton, VA 23666, USA

Abstract

The generation of three-dimensional unstructured grids using the advancing-front technique is described. While this generation technique has been shown to be effective for the generation of unstructured grids in two dimensions, its extension to three-dimensional regions required the development of surface definition software and sophisticated data structures to avoid excessive CPU-time overheads for the search operations involved. After obtaining an initial triangulation of the surfaces, tetrahedrons are generated by successively deleting faces from the generation front. Details of the mesh generation algorithm are given, together with examples and timings.

1. Introduction

In recent years a wide variety of algorithms has been devised for the generation of unstructured grids around bodies of complex geometrical shapes. Among the different techniques we mention Watson's algorithm for Voronoi tesselations [1-6], the modified octree method [7] and the advancing front technique [8-10]. Baker's implementation and optimization of the Voronoi algorithm [6] has shown that fast and reliable grid generators for tetrahedral meshes can be produced. We currently believe that the advancing front technique is the best approach, because it can easily be used for grid regeneration with directional refinement [10]. The incorporation of directional refinement in the Voronoi algorithm appears difficult unless the reconnection of points based on the purely geometrical Delauney criterion [6] is substituted by some other criterion that incorporates directionality into the triangulation. Directional refinement is an essential ingredient in any optimal 3-D algorithm for compressible flows.

2. Algorithmic Steps of Advancing-Front Generators

The advancing-front grid generation technique [8-10] consists of the following steps:

F.1 Define the boundaries (surfaces) of the domain to be gridded.

F.2 Set up a background grid to define the spatial variation of the size, the stretching, and the stretching direction of the elements to be generated. The background grid consists of tetrahedrons. At the nodes we define the desired element size, stretching and stretching direction. This background grid must completely cover the domain to be gridded.

F.3 Using the information stored on the background grid, set up faces on all these boundaries. This yields the initial front. At the same time, find the generation parameters (element size, stretching and stretching direction) for the new faces from the background grid.

F.4 Select the next face to be deleted from the front; in order to avoid large elements crossing over regions of small elements, the face forming the smallest new element is selected as the next face to be deleted from the list of faces.

F.5 For the face to be deleted:

F.5.1 Select a 'best point' position for the introduction of a new point IPNEW.

F.5.2 Determine whether a point exists in the already generated grid that should be used in lieu of the new point. If there is such a point, set this point to IPNEW and continue searching (go to F.5.2).

F.5.3 Determine whether the element formed with the selected point IPNEW does not cross any given faces. If it does, select a new point as IPNEW and try again (go to F.5.3).

F.6 Add the new element, point, and faces to their respective lists.

F.7 Find the generation parameters for the new faces from the background grid.

F.8 Delete the known faces from the list of faces.

F.9 If there are any faces left in the front, go to F.4.

3. Definition of Surfaces

Several approaches have been proposed in the literature to define the surfaces of objects [11]. The two most common are: a) Boolean operations on surface patches, and b) Boolean operations on Solids. Both have advantages and disadvantages. In order to be compatible with the surface definition currently in use in the aerospace and car manufacturing industry, we adopted the first approach. Moreover, in order to define surfaces or solid objects, we attempt to minimize the manual input that is necessary as much as possible. To that end, we have adopted a hierarchical surface definition data structure. Three levels of data are allowed: points, lines and surfaces. Lines are obtained by joining points, and surfaces by joining lines.

The line-types currently available to the user are:

- straight line segment (defined by 2 points);
- parabolic line segment (defined by 3 points);
- cubic spline segment (defined by 4 or more points).

The surface-types currently available to the user are:

- plane (assumes all line segments lie in one plane);
- triangular isoparametric parabolic surface [12] (defined by 3 parabolic line segments);
- rectangular isoparametric serendipity surface [12] (defined by 4 parabolic line segments);
- triangular Barnhill-Gregory-Nielson patch [13] (defined by 3 arbitrary line segments);
- bilinear transfinite Coon's patch [13] (defined by 4 arbitrary line segments).

Observe that so far we have not implemented C^1- or C^2-continuous surface patches. The program was written in an 'open-end'-mode, so that the addition of further line segment-types or surface-types will be carried out as needed.

4. Background Grid

The background grid consists of tetrahedrons, and is used to define the desired spatial variation of element size, stretching and stretching direction. The background grid typically consists of only a few elements, so that manual interactive input with mouse presents no significant burden. If the user desires to construct a uniform mesh, then the background grid consists of only one element that covers the domain to be gridded completely. If a rapid and significant change of element size is desired, the grid generator itself may be used to generate first a background grid. At the nodes of this generated grid we then define the desired element size, stretching and stretching direction, and proceed to generate the final mesh. Within an adaptive refinement process, the current grid and flow solver solution are used as the background grid to generate a new, better grid for the flow problem under consideration.

5. Generation of the Initial Front

The generation of the initial front or surface triangulation is carried out in two main steps. a) the line segments are divided into straight line segments, called sides, and b) the surface segments are triangulated, starting from the sides of the corresponding line segments.

5.1 Generation of sides

Taking the background grid into consideration, each line segment is subdivided into straight line segments, called sides. The length of each side is determined by interpolation from the background grid, and can vary arbitrarily along the line.

5.2 Triangulation of surface segments

Each surface segment is triangulated independently using a 2-D version of the advancing front grid generator. The substeps are the same as outlined above (see section 2). The initial front consists of the sides corresponding to the lines that connect the current surface segment to other neighboring surface segments. The triangle size and stretching are computed from the 3-D background grid by interpolation. As the triangulator resides in a 2-D world, we need to reproduce as faithfully as possible the 3-D surface in a 2-D domain. To that end, mappings between the 2-D and 3-D worlds are used that maintain approximately the shape and size of the 3-D surface triangles in 2-D and vice-versa. The mappings used are:

- 3-D surface segment to unit 2-D triangle or square $(x, y, z \rightarrow \xi, \eta)$, and
- stretching and shearing of unit 2-D triangle or square to approximate 3-D surface segment in a 2-D domain $(\xi, \eta \rightarrow \xi'', \eta'')$.

This mapping process is illustrated in Figure 1. When computing the element size during the triangulation, at each stage we proceed as follows:

T.1 Transform the current 2-D position to the 3-D surface segment via the unit triangle or square: $\xi'', \eta'' \rightarrow \xi, \eta \rightarrow x, y, z$.

T.2 Determine from the background grid the desired element size and shape.

T.3 Transform back the desired element size and shape to the 2-D domain.

Transformation T.1 is also used to transform back the new points that have been added due to the triangulation of the surface segment.

The assembly of all triangles obtained from the surface triangulator yields the initial front for the 3-D advancing front grid generator. As an example, we show in Figure 2 the surface triangulation of an F-18. The surface definition used, shown in Figure 2a, is given by 325 points, 73 lines and 32 surfaces. The background grid, shown in Figure 2b, consists of 1635 tetrahedrons and 390 points. The resulting surface triangulation, depicted in Figures 2c,d has 9,962 triangles and 5,122 points.

6 Checking the Intersection of Faces

One of most important ingredients of the advancing front generator is a reliable and fast algorithm for checking whether two faces intersect each other. We have found that even slight changes in this portion of the generator greatly influence the final mesh. As with so many other problems in computational geometry, checking whether two faces intersect each other seems trivial for the eye, but is complicated to code. The problem is shown in Figure 3. We base our checking algorithm on the following observation: two triangular faces do not intersect if no side of either face intersects the other face. The idea then is to build all possible side-face combinations between any two faces and check them in turn. If no intersection is found, then the faces do not cross. With the notation defined in Figure 4, the intersection point is found as

$$\underline{x}_f + \alpha^1 \underline{g}_1 + \alpha^2 \underline{g}_2 = \underline{x}_s + \alpha^3 \underline{g}_3 \quad , \qquad (1)$$

where we have used the $\underline{g}_i$-vectors as a covariant basis. Using the contravariant basis $\underline{g}^i$ defined by

$$\underline{g}^i \cdot \underline{g}_j = \delta^i_j \quad , \qquad (2)$$

where δ^i_j denotes the Kronecker-delta, we obtain the α^i as

$$\begin{aligned} \alpha^1 &= (\underline{x}_s - \underline{x}_f) \cdot \underline{g}^1 \quad , \\ \alpha^2 &= (\underline{x}_s - \underline{x}_f) \cdot \underline{g}^2 \quad , \\ \alpha^3 &= (\underline{x}_f - \underline{x}_s) \cdot \underline{g}^3 \quad . \end{aligned} \qquad (3)$$

Because we are only interested in a triangular surface for the $\underline{g}_1, \underline{g}_2$ - plane, we define another quantity similar to the third shape function for a linear triangle:

$$\alpha^4 = 1 - \alpha^1 - \alpha^2 \quad . \qquad (4)$$

Using the α^i, two faces can be considered as 'crossed' if they only come close together. Then, in order for the side not to cross the face, at least one of the α^i has to satisfy

$$t > max(-\alpha^i, \alpha^i - 1) \quad , i = 1,4 \quad , \qquad (5)$$

where t is a predefined tolerance. By projecting the $\underline{g}_i$ onto their respective unit contravariant vectors, we can obtain the actual distance between a face and a side. The criterion given by Eqn.(5) would then be replaced by (see Figure 5):

$$d > \frac{1}{|\underline{g}^i|} max(-\alpha^i, \alpha^i - 1) \quad , i = 1,4 \quad . \qquad (6)$$

The first form (Eqn.(5)) produces acceptable grids. If the face and the side have points in common, then the α^i will all be either 1 or 0. As both Eqn.(5) and Eqn.(6) will not be satisfied, we need to make special provision for these cases. For each two faces, six side-face combinations are possible. Considering that on average about 40 close faces need to be checked, this way of checking the crossing of faces is very CPU-intensive. When it was first implemented, this portion of the grid generation code took more than 80% of the CPU time required. In order to reduce the work load, a three-layered approach was subsequently adopted:

a) Min/Max-search: The idea here is to disregard all face-face combinations where the distance between faces exceeds some prescribed minimum distance. This can be accomplished by checking the maximum amd minimum value for the coordinates of each face. Faces can not possibly cross each other if at least for one of the dimensions $i = 1,2,3$ they satisfy one of the following inequalities

$$max_{face1}(x^i_A, x^i_B, x^i_C) < min_{face2}(x^i_A, x^i_B, x^i_C) - d \quad , \qquad (7a)$$

$$min_{face1}(x^i_A, x^i_B, x^i_C) > max_{face2}(x^i_A, x^i_B, x^i_C) + d \quad , \qquad (7b)$$

where A, B, C denote the corner points of each face.

b) Local element coordinates: The purpose of checking for face-crossings is to determine whether the newly formed tetrahedron breaks already given faces. The idea is to extend the previous Min/Max-criterion with shape functions of the new tetrahedron. If all the points of a given face have shape-function values N^i that have the same sign and lie outside the $[-t,\ 1+t]$ interval, then the tetrahedron cannot possibly cross the face. We therefore disregard this face.

c) In-depth analysis of side-face combinations: All the faces remaining after the filtering process of steps a) and b) are analyzed using side-face combinations as explained above.

Each of these three filters requires about an order of magnitude more CPU-time than the preceding one. When implemented in this way, the face-crossing check required only 25% of the total grid generation time. When operating on a vector machine, we perform loops over all the possible combinations, building the $\underline{g}_i, \underline{g}^i, \alpha^i$, etc. in vector mode. Although the vector lengths are rather short, the chaining that results from the lengthy mathematical operations involved results in acceptable megaflop-rates on the CRAY-XMP.

7. Data Structures to Minimize Search Overheads

The operations that could potentially reduce the efficiency of the algorithm to $O(N^{1.5})$ or even $O(N^2)$ are (see section 2):

a) Finding the next face to be deleted (step F.4).

b) Finding the closest given points to a new point (step F.5.2).

c) Finding the faces adjacent to a given point (step F.5.3).

d) Finding for any given location the values of generation parameters from the background grid (steps F.3 and F.7). This is an interpolation problem on unstructured grids.

The verb 'to find' appears in all of these operations. The main task is to design the best data structures for performing the search operations a)-d) as efficiently as possible.

7.1 Heap List for the Face-Search

Heap lists are well-known binary tree data structures in computer science [14,15]. The ordering of the tree is accomplished by requiring that the key of any father (root) be smaller than the keys of the two sons (branches). An example of a tree ordered in this manner is given in Figure 6, where a possible tree for the letters of the word 'example' is shown. The letters have been arranged according to their place in the alphabet. We must now devise ways to add or delete entries from such an ordered tree without altering the ordering. Because we have to add faces as entries in the tree in this case, we replace 'entry' by 'face'. The ideas that follow use the heap-sort and heap-search algorithms [14,15] to determine quickly which face should be deleted next from the front.

Adding a new face: The idea is to add the new face at the end of the tree. If necessary, the internal order of the tree is re-established by comparing father and son pairs. Thus, we start at the bottom of the tree and work our way upwards. In this way, the face with the smallest associated key RFACE(IFACE) remains at the top of the list in position LHEAP(1). The process is illustrated in Figure 6, where the letters of the word 'example' have been inserted sequentially into the heap list.

Removing the face at the top of the heap list: The idea is to take out the face at the top of the heap list, and replace it by the face at the bottom of the heap list. If necessary, the internal order of the tree is re-established by comparing pairs of father and sons. Thus, we start at the top of the tree and work our way downwards. In this way, the face with the smallest associated key will again remain at the top of the list in position LHEAP(1). This process is illustrated in Figure 7, which shows the successive removal of the smallest element (alphabetically) from the previously constructed heap list.

It can be proved that both the insertion and the deletion of a face into the heap list will take $O(\log_2(\texttt{NHEAP}))$ operations [14,15] on the average.

7.2 Quad/Octrees for the Point-Search

Quadtrees and Octrees are used extensively to speed up the nearest neighbor searches in graphics algorithms [11], battle management [16], particle simulations [17], and to define solids for grid generators [7]. Samet [18] gives an extensive survey of the field. Their main role here is to provide an $O(\log N)$ search algorithm for arbitrary point distributions. We describe the main ideas behind their application for 2-D regions, from which the extension to 3-D regions is straightforward. Define an array LQUAD(1:7,MQUAD) to store the points, where MQUAD denotes the maximum number of quads allowed. For each quad IQ store in LQUAD(1:7,IQ) the following information:

LQUAD(7,IQ) :	< 0 :	the quad is full
	$= 0$:	the quad is empty
	> 0 :	the number of points stored in the quad
LQUAD(6,IQ) :	> 0 :	the quad the present quad came from
LQUAD(5,IQ) :	> 0 :	the position in the quad the present quad came from
LQUAD(1:4,IQ) :	for LQUAD(7,IQ) > 0 : the points stored in this quad for LQUAD(7,IQ) < 0 : the quads into which the present quad was subdivided	

We store at most four points per quad. If a fifth point falls into the quad, the quad is subdivided into four, and the old points are relocated in their respective quads. Then the fifth point is introduced to the new quad into which it falls. If the quad is full again, the subdivision process continues, until a quad with vacant storage space is found. This process is illustrated in Figure 8. The newly introduced point E falls into the quad IQ. As IQ already contains the four points A,B,C and D, the quad is subdivided into four. Points A,B,C and D are relocated to the new quads, and point E is added to the new quad MQUAD+2. Figure 8 also shows the entries in the LQUAD-array, as well as the associated tree-structure. In 3-D, we store eight points per octant, and subdivide an octant into eight suboctants if a ninth point falls into it.

In order to find points that lie inside a search region, we go down the levels of the octree, eliminating at the highest possible level the octants that lie outside the search region. In order to find the point closest to a given point, we again go down the levels of the octree. If the octant into which the point falls is empty, we collect into a list all the points lying in the octants that emanated from the subdivision giving rise to the present octant. The closest point is then chosen according to geometric arguments from this list.

With the octree, it takes $O(\log_8 N)$ operations to locate all points inside a search region or to find the point closest to a given point.

7.3 A Linked List for the Face/Point Search

In the present case, we need to develop a storage scheme that helps us answer the question: which are the faces adjacent to a given point? Since the number of faces surrounding a point varies from point to point, but usually fluctuates within certain bounds, we use the following scheme. Define an array LPOIN(1:NPOIN) over the points and another array LFAPO(1-3,MFAPO), where NPOIN denotes the number of points and MFAPO the maximum number of storage locations. Then store in:

LPOIN(IPOIN) :	the place IFAPO in LFAPO where the storage of the faces surrounding point IPOIN starts.	
LFAPO(3,IFAPO) :	> 0 :	the number of stored faces

	< 0 :	the place JFAPO in LFAPO where the storage of the faces surrounding point IPOIN is continued
LFAPO(1:2,IFAPO) :	$= 0$:	an empty location
	> 0 :	a face surrounding IPOIN

We allowed only two storage locations per entry in LFAPO because in 2-D there are typically two faces adjacent to a point. In 3-D we use nine or ten storage locations. Once this storage scheme has been set up, we can store and find the faces surrounding points. The process of adding a face to the linked list LPOIN/LFAPO is shown diagramatically in Figure 9. Changing the lists from 'face' to 'element,' we can also store in this way the elements that surround each point.

7.4 Interpolation on Unstructured Grids

Interpolating information from one unstructured grid onto another has been an outstanding problem for several years. There are at least two major attempts to solve this problem [19,20], none of which guarantees an optimal $O(N \log N)$ algorithmic process. With the data structures described above, it is possible to guarantee interpolation in $O(N \log N)$ operations by proceeding as follows [21]:

- Given an arbitrary point, find the closest points on the grid from which the information is to be interpolated by using the octree.
- Find the elements surrounding these points from a linked list that stores the elements surrounding points.
- Find from this list of elements, the element into which the point to be interpolated falls.
- Interpolate.

The region of close elements may have to be enlarged by layers of elements using the linked list, because for badly deformed elements, the closest points on the background grid may not belong to the element into which the point to be interpolated falls. As a result of this, many elements could be tested. However, this slight disadvantage is offset by the guarantee that the overall procedure requires only $O(N \log N)$ operations, regardless of grid stretching, holes, corners, and other topological features that deteriorate the performance of the other available methods. Moreover, practical experience indicates that in more than 90% of the cases, the element into which the point to be interpolated falls is found in the first layer.

8. Some Examples and Timings

8.1 Pathfinder Configuration in a Windtunnel: Figure 10 shows the experimental pathfinder configuration in a windtunnel. The surface definition was given by 147 points, 44 lines and 12 surface segments. The background grid had 96 points and 315 elements. Figures 10a,b show the surface triangulation of the model, where the mirroring capability of the code was invoked. Figure 10c shows the model in the windtunnel. The generated grid had 21,811 points and 119,861 tetrahedrons. Figure 10d shows the grid along the wall of the tunnel, and one can observe the point clusterings around the nose, the wing-roots and the tail.

8.2 Missile launcher: Figure 11 shows a generic missile launcher model. Again, only half the model is required for computational purposes. The surface definition was given by 39 points, 44 lines and 19 surface segments. The background grid had 48 points and 132 elements. Figures 10a shows the complete surface triangulation. The generated grid contains 14,508 points and 75,894 tetrahedrons. Figure 11b shows the grid along the axis of symmetry, and one can again observe finer grid zonings close to the missile and the launcher.

The time needed to generate a new element depends heavily on the amount of faces that are checked for possible crossing (see above, section 2). We find that for larger grids, the number of faces that are checked decreases on the average, as the distance between 'colliding fronts' is larger. For the grids shown, the rate at which new tetrahedrons were generated varied between 480-490 tetrahedrons per second on the CRAY-XMP-24 at NRL (using one processor).

9. Conclusions

This paper describes a mesh-generation procedure for three-dimensional regions. Input requirements to define objects or surfaces were minimized by adopting a hierarchical structure consisting of points, lines and surfaces. In order to reduce CPU-requirements, several optimal search algorithms were adapted into the present context. The described developments are by no means limited to advancing-front grid generators, but should also be useful for the construction of Voronoi tesselations and Delaunay triangulations or tetrahedrizations. Future developments will center on

- better surface definition,
- easier input of background grids,
- better criteria for the introduction of points,
- coupling of the grid generator with flow solvers, and
- adaptive remeshing.

10. Acknowledgements

This work was partially funded by the Office of Naval Reseach through the Naval Research Laboratory, the Applied and Computational Mathematics Program of DARPA, the NASA Johnson Space Center and SBIR-Grant-No. NAS1-18419.

We would also like to acknowledge the support of C. Gumbert (TAB, NASA LRC), who provided a number of very useful graphics interfaces for the IRIS, as well as Drs. M.D. Salas (TAB, NASA LRC) and B.

Westin (AMB, NASA LRC), through which we had access to the aerodynamic configurations shown in the paper.

11. References

[1] D.F. Watson - Computing the N- Dimensional Delaunay Tesselation with Application to Voronoi Polytopes; *The Computer Journal* 24, 2, 167-172 (1981).

[2] A. Bowyer - Computing Dirichlet Tesselations; *The Computer Journal* 24, 2, 162-167 (1981).

[3] S.W. Sloan and G.T. Houlsby - An Implementation of Watson's Algorithm for Computing 2-Dimensional Delaunay Triangulations; *Adv. Eng. Software* 6, 4, 192-197 (1984).

[4] M. Tanemura, T. Ogawa and N. Ogita - A New Algorithm for Three-Dimensional Voronoi Tesselation; *J. Comp. Phys.* 51, 191-207 (1983).

[5] A. Jameson, T.J. Baker and N.P. Weatherhill - Calculation of Inviscid Transonic Flow over a Complete Aircraft; AIAA-86-0103 (1986).

[6] T.J. Baker - Three Dimensional Mesh Generation by Triangulation of Arbitrary Point Sets; AIAA-87-1124-CP (1987).

[7] M.A. Yerry and M.S. Shepard - Automatic Three-Dimensional Mesh Generation by the Modified-Octree Technique; *Int. J. Num. Meth. Eng.* 20, 1965-1990 (1984).

[8] N. van Phai - Automatic Mesh generation with Tetrahedron Elements; *Int. J. Num. Meth. Eng.* 18, 237-289 (1982).

[9] S.H. Lo - A New Mesh Generation Scheme for Arbitrary Planar Domains; *Int. J. Num. Meth. Eng.* 21, 1403-1426 (1985).

[10] J. Peraire, M. Vahdati, K. Morgan and O.C. Zienkiewicz - Adaptive Remeshing for Compressible Flow Computations; *J. Comp. Phys.* 72, 449-466 (1987).

[11] J. Woodwark - *Computing Shape*; Butterworths (1986).

[12] O.C. Zienkiewicz and K. Morgan - *Finite Elements and Approximation*; J. Wiley & Sons (1983).

[13] R.E. Barnhill - Representation and Approximation of Surfaces; pp. 69-120 in *Mathematical Software III* (J.R. Rice ed.), Academic Press (1977).

[14] D.N. Knuth - *The Art of Computer Programming* , Vol. 3; Addison-Wesley (1973).

[15] R. Sedgewick - *Algorithms* ; Addison-Wesley (1983).

[16] A.J. Broder - An Algorithm for Incremental Nearest Neighbor Search in k-Dimensional Space; Rep. MP-86W00026, The MITRE Corp. (1986).

[17] L. Greengard and V. Rokhlin - A Fast Algorithm for Particle Simulations; *J. Comp. Phys.* 73, 325-348 (1987).

[18] H. Samet - The Quadtree and Related Hierarchical Data Structures; *Computing Surveys* 16, 2, 187-285 (1984).

[19] R. Löhner and K. Morgan - An Unstructured Multigrid Method for Elliptic Problems; *Int. J. Num. Meth. Eng.* 24, 101-115 (1987).

[20] D. Mavriplis and A. Jameson - Multigrid Solution of the Two-Dimensional Euler Equations on Unstructured Triangular Meshes; AIAA-87-0353 (1987).

[21] R. Löhner - Some Useful Data Structures for the Generation of Unstructured Grids; *Comm. Appl. Num. Meth.* , to appear (1988).

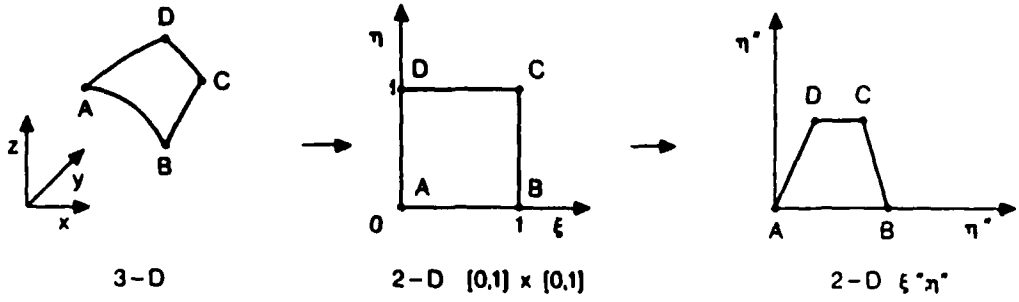


Fig. 1 Mapping of Surface Patch to Unit Square with Subsequent Stretching and Shearing

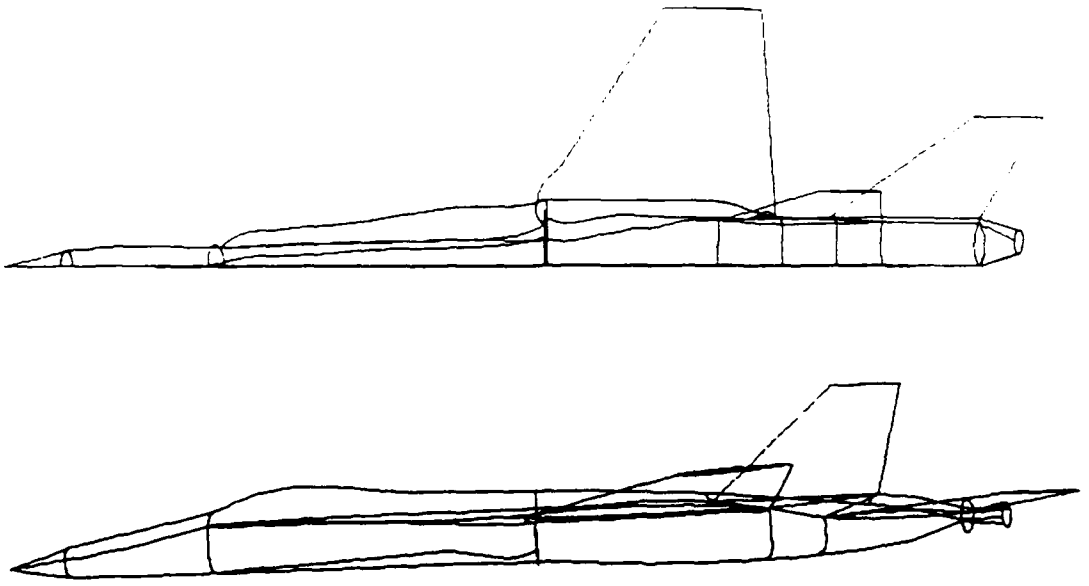


Fig. 2a F-18: Surface Definition

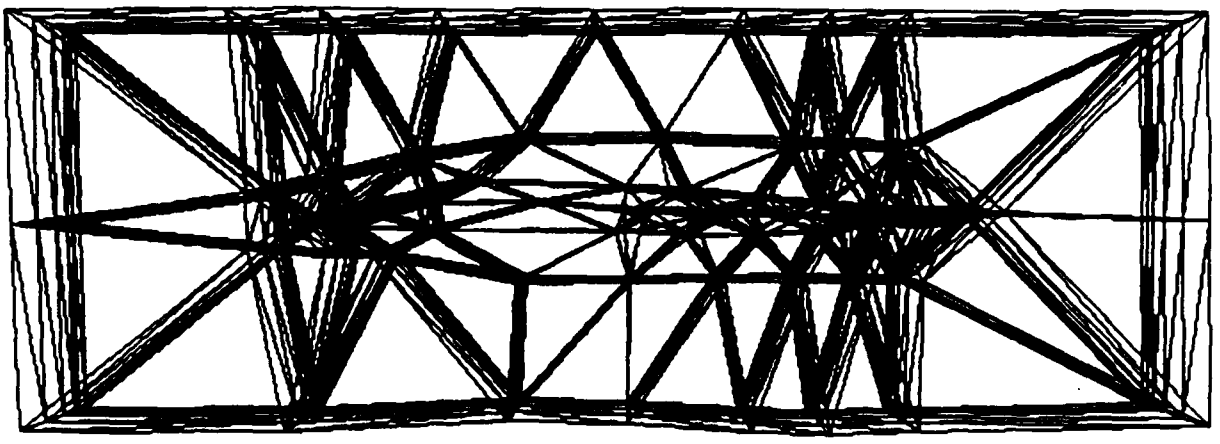


Fig. 2b F-18: Background Grid

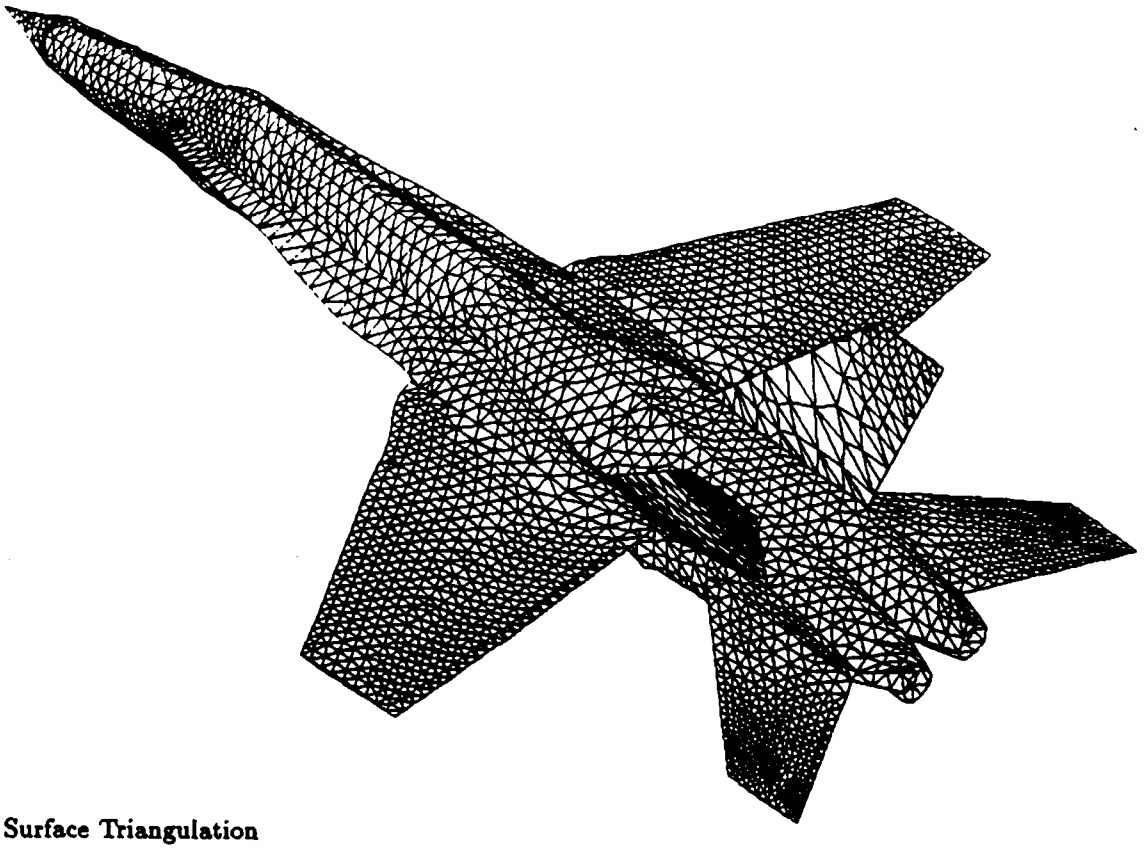


Fig. 2c F-18: Surface Triangulation

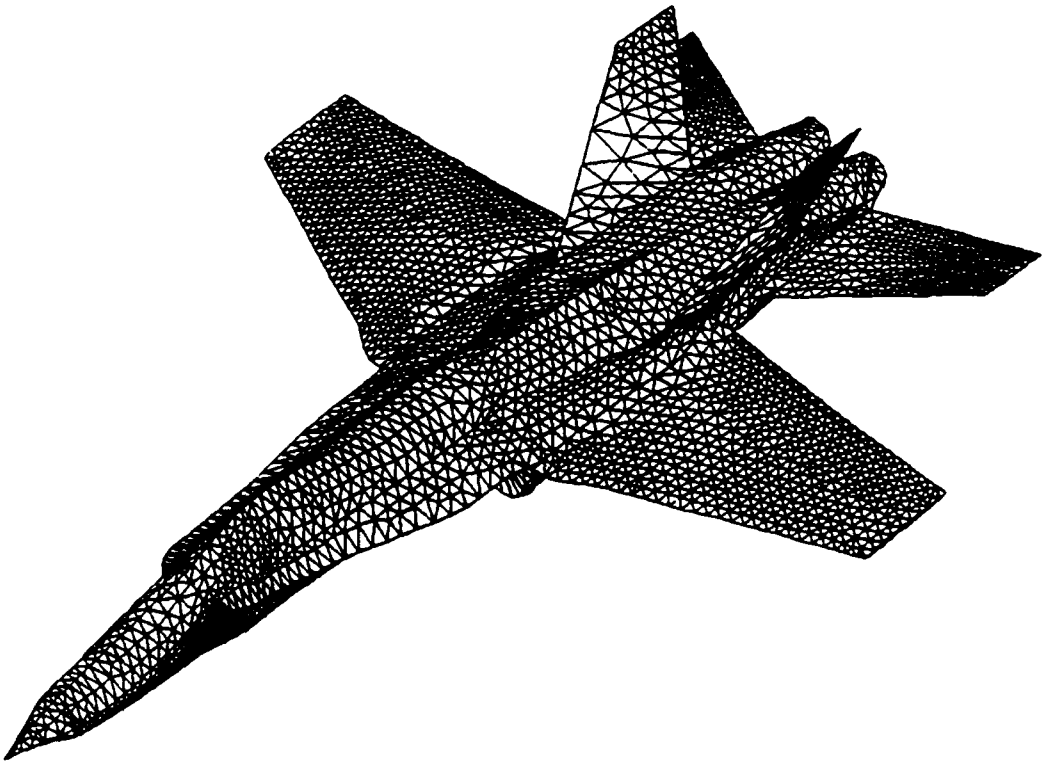


Fig. 2d F-18: Surface Triangulation

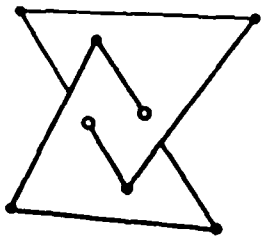


Fig. 3 Crossing of Faces

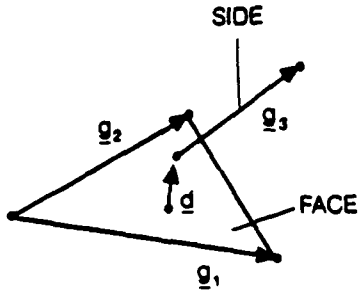


Fig. 4 Face-Side Combination

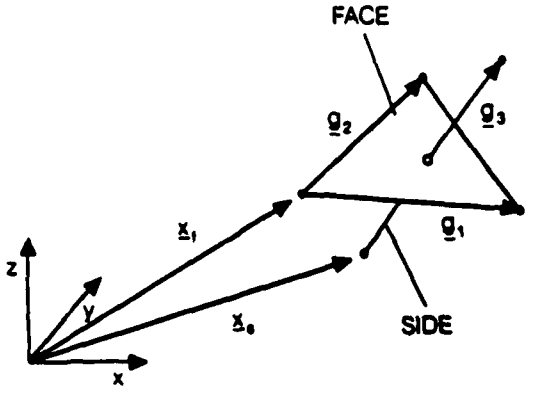


Fig. 5 Distance Between Face and Side

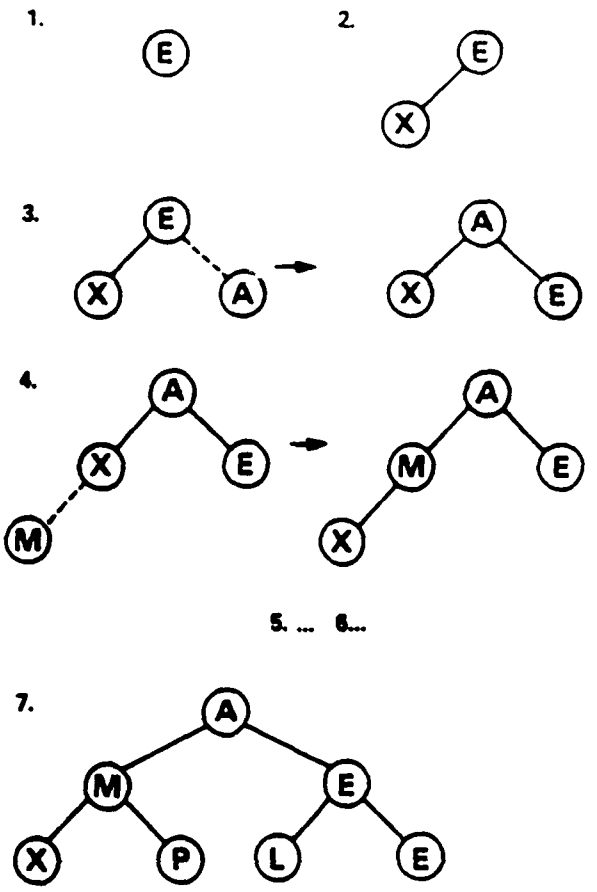


Fig. 6 Insertion of New Items into the Heap List

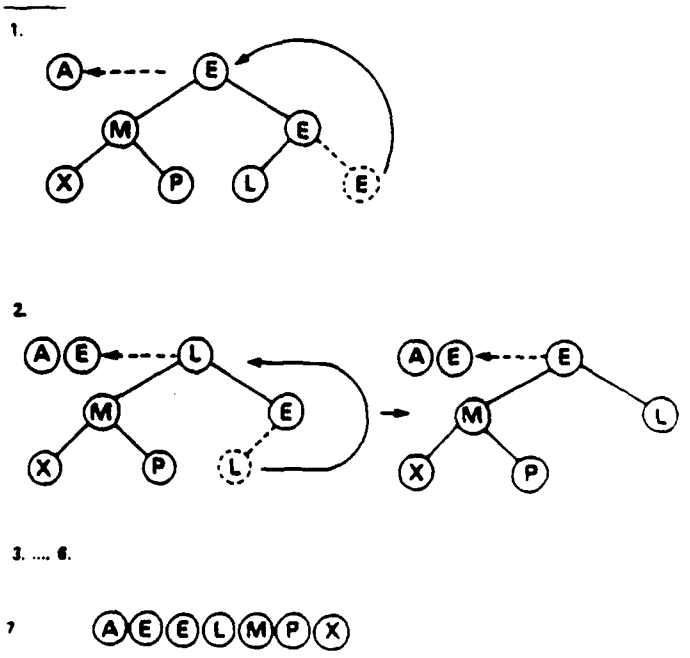


Fig. 7 Successive Deletion of the Smallest Item from the Heap List

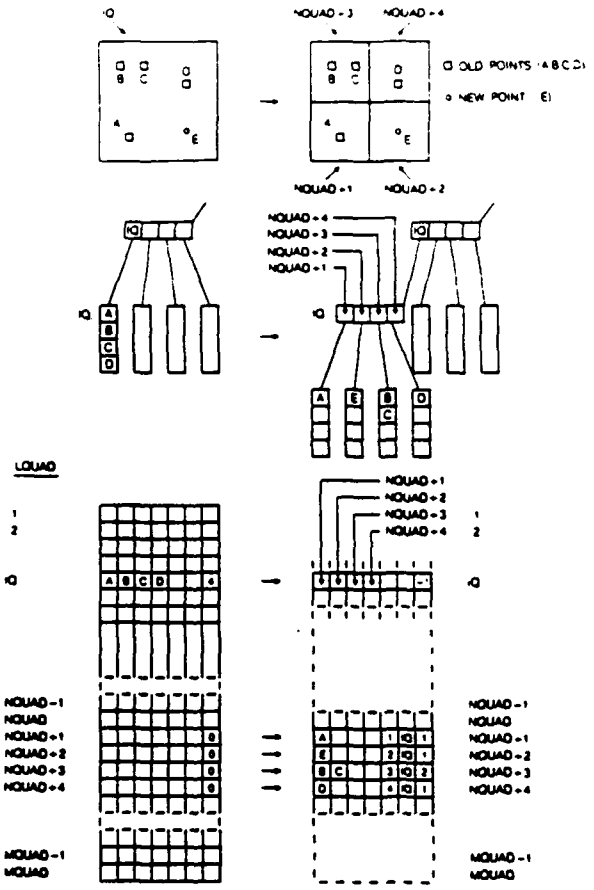


Fig. 8 Introduction of a New Point (E) into a Full Quad (IQ)

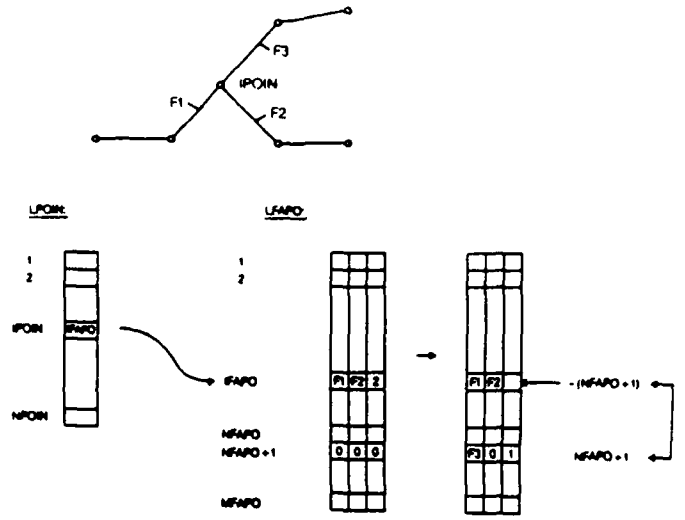


Fig. 9 Introduction of a New Face (E) into the Linked List

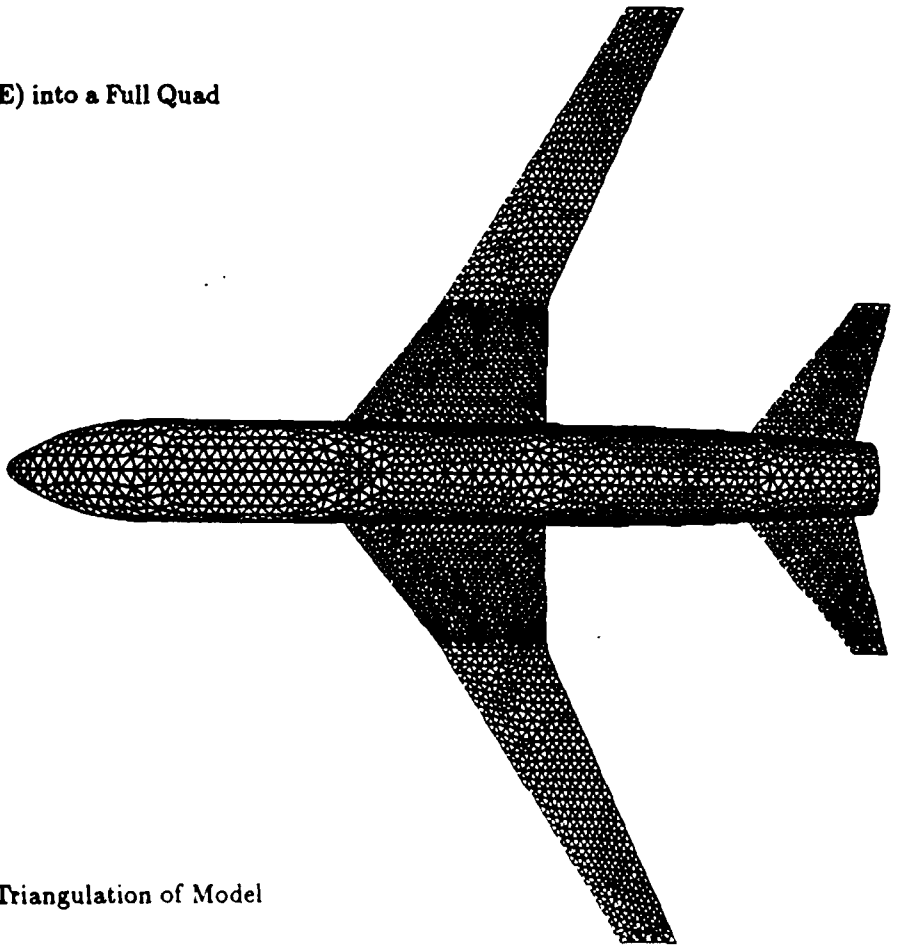


Fig. 10a Pathfinder: Surface Triangulation of Model

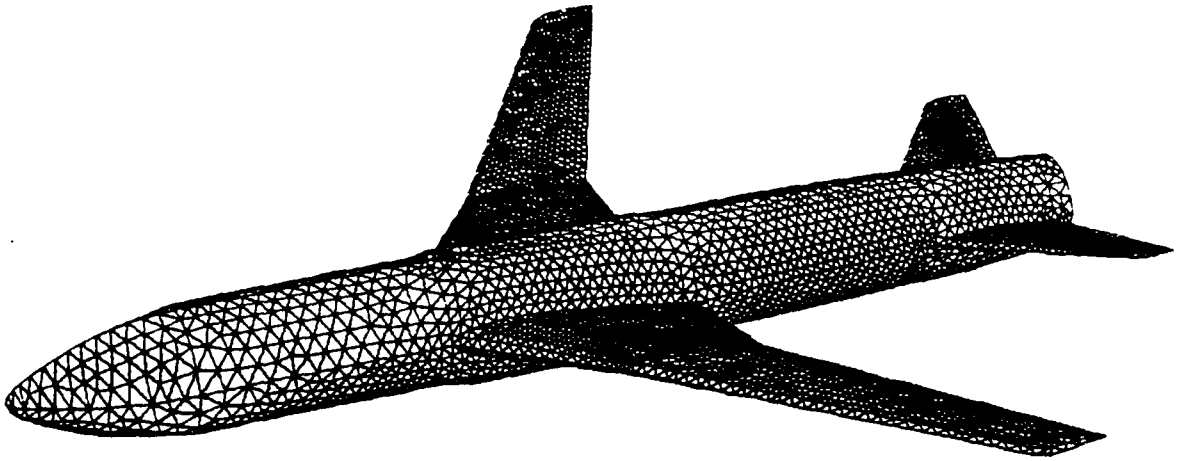


Fig. 10b Pathfinder: Surface Triangulation of Model

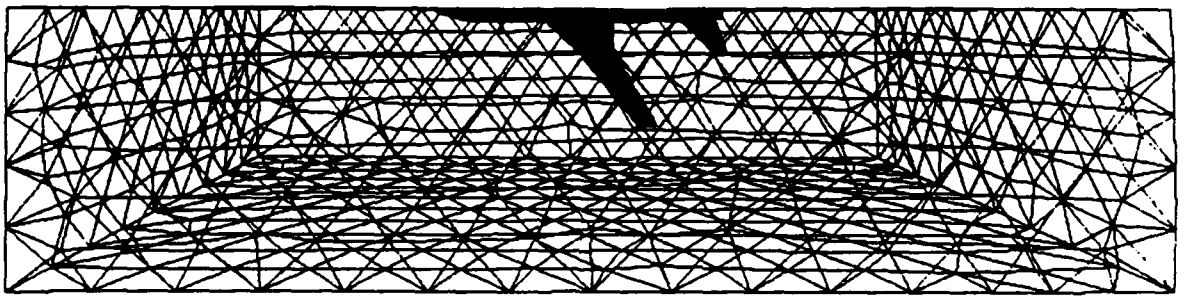


Fig. 10c Pathfinder: Complete Surface Triangulation

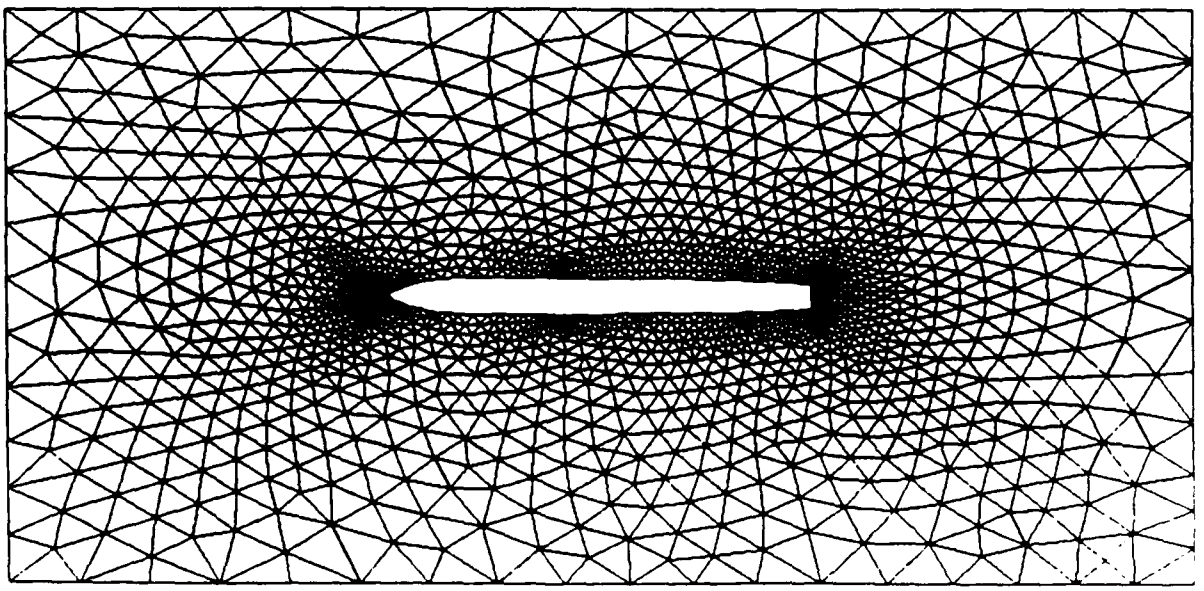


Fig. 10d Pathfinder: Grid in Plane of Symmetry

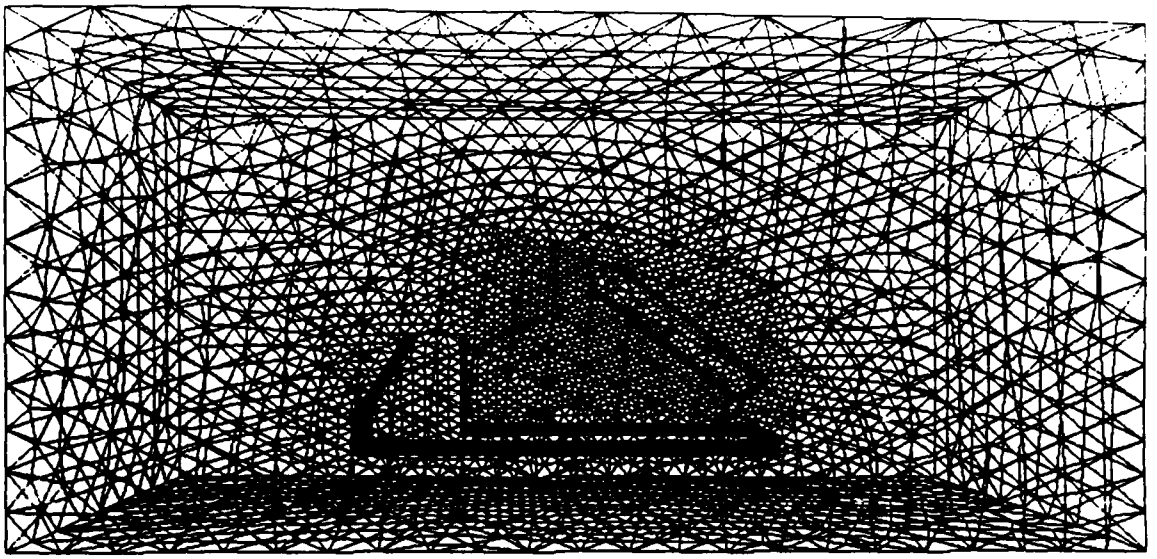


Fig. 11a Missile Launcher: Surface Triangulation

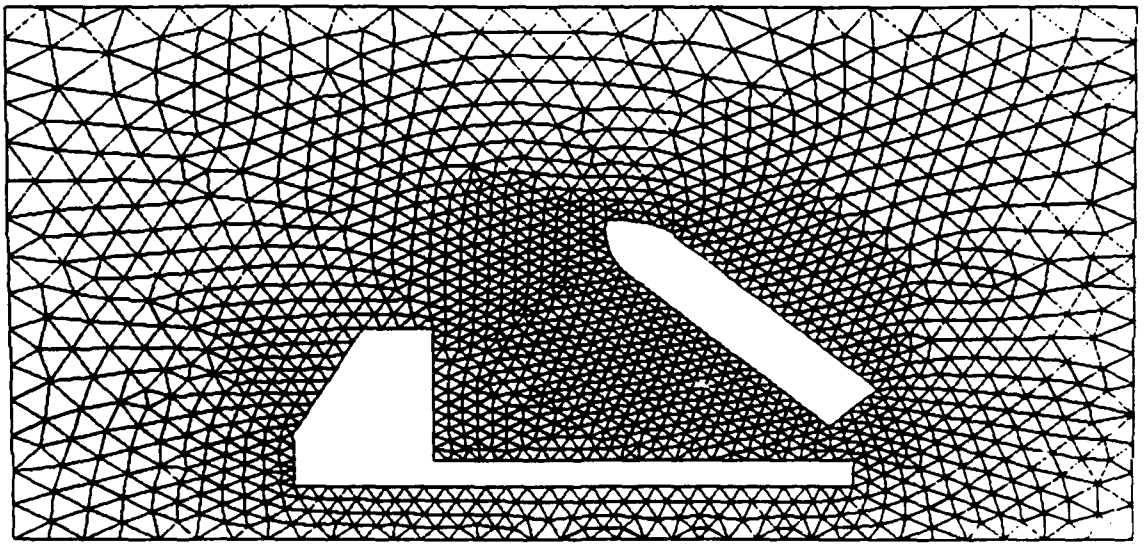


Fig. 11b Missile Launcher: Grid in Plane of Symmetry

APPENDIX B.

Applications of the Method of Flux-Corrected Transport to Generalized Meshes

APPLICATIONS OF THE METHOD OF FLUX-CORRECTED TRANSPORT TO GENERALIZED MESHES

Rainald Löhner and Gopal Patnaik
Berkeley Research Associates
Springfield, VA 22150

Jay P. Boris, Elaine S. Oran, and David L. Book
Laboratory for Computational Physics
Naval Research Laboratory, Washington, DC 20375

FCT on Unstructured Grids

A new technique for numerical solution of fluid equations has been developed by combining the Finite-Element Method with the method of Flux-Corrected Transport. The resulting hybrid method, called FEM-FCT, is useful for problems involving steady and unsteady transonic and supersonic flow in irregular geometries. The main computational advance grows out of the need to find a prescription for limiting fluxes through the sides of a triangular or other nonquadrilateral zone when arbitrarily many zones may meet at a given vertex (Löhner *et al.*, 1986).

The technique described here employs a version of Zalesak's (1979) fully multidimensional FCT on a triangular grid. The underlying transport scheme can be any standard high-order finite-element or finite-volume method, and need not be based on triangular zones. In the present work a Taylor-Galerkin scheme (Löhner *et al.*, 1985) has been used for this purpose. The low-order scheme employed in Zalesak's formulation is obtained by adding a numerical diffusion term to the high-order scheme. Antidiffusion removes this added diffusion except where the flux-limiting process modifies the antidiffusive fluxes to prevent unphysical extrema from forming. For each fluid equation all the high-order fluxes which tend to increase the value in a particular zone, and separately all the fluxes which tend to decrease the value there, are considered together. If in either case a maximum or minimum is formed anywhere which is not produced by the low-order scheme, all the participating fluxes are reduced by the same factor until the extraneous peak or valley is eliminated. The result for the transported quantity is a value which is intermediate between the high- and the low-order results, with just enough of the latter present to guarantee that *no* extrema form (except those produced physically).

For the test problems used in this paper to illustrate the method, the temporal discretization is a two-step (predictor-corrector) Lax-Wendroff-type method and the spatial discretization is done with triangular finite elements (Löhner *et al.*, 1985). A Lapidus diffusion term of the form

$$h^2 \frac{\partial}{\partial l}\left\{ \max\left[0, \left|\frac{\partial(\mathbf{l}\cdot\mathbf{v})}{\partial l}\right|\right] \frac{\partial \mathbf{u}}{\partial l} \right\},$$

where Δt is the timestep, h is the zone size, $\mathbf{l} = \nabla v/|\nabla v|$ is the unit vector in the direction of the gradient of $v = |\mathbf{v}|$, and $\mathbf{u}$ is the vector of dependent variables, is applied in expansion regions to prevent the formation of "terraces." Running times on a one-pipe Cray X-MP were 42–58 μs per

grid point per timestep. These are about a factor of three slower than efficient FCT algorithms on rectilinear grids, e.g., JPBFCT (Boris, 1981). The penalty results from the need to include gather-scatter operations in the algorithm because physically contiguous quantities need not be logically contiguous.

After optimization of the flux-limiting technique through tests on one-dimensional problems, the method was applied to a variety of unsteady flows, including spherical blast waves and shock waves diffracting around hemispheres and half-cylinders. Because the algorithm is not coordinate-timesplit the simulations maintained symmetry perfectly, so that, e.g., the projection of a spherical blast wave on the z-axis is identical to that on the r-axis. Tests were also carried out on the problem of a planar shock impinging on a complex structure composed of two irregular objects (Fig. 1). A strong ($M_s = 10$) shock hits the two structures shown, producing a bow shock and several rarefactions and contact discontinuities, as well as several reflected secondary shocks (e.g., below the structure on the left). Note the high resolution of the shocks and contact discontinuities; the jumps are resolved over 1–2 zones, in contrast with the number (4–6) needed in conventional finite-element schemes. Note also that those contours which are supposed to be straight remain straight, i.e., they essentially ignore the orientation of the underlying grid. Figure 2 shows the results of another calculation, in which a strong ($M_s = 25$) shock interacts with a channel aligned parallel to the shock front. At the time shown ($t = 1.2$ in normalized units) the incident (unreflected) shock has passed beyond the edge of the frame to the right. At the bottom it has reflected off the right wall of the channel and is seen propagating back towards the left. Focusing occurs because of the geometry, resulting in the formation of the "eye" just to the right of the middle of the channel. Note the very strong rarefaction fan attached to the top of the left wall of the channel; features like this in hydrocode calculations are ordinarily very susceptible to dispersive errors or (in the case of FCT algorithms) terracing.

One of the advantages of triangular gridding [Fig. 1(a) and Fig. 2(a)] is that it is easy to refine the mesh in regions where improvements in resolution are needed. The strategy we have followed is one of enrichment, not redistribution. That is, we introduce new triangles in the region of interest, rather than moving triangles from elsewhere. We have implemented an automatic adaptive mesh refinement routine in FEM-FCT (Löhner, 1986). The algorithm can be switched on locally whenever a predefined feature of interest can be identified. The switch is activated by estimating the local error and refining wherever it exceeds a prescribed limit. Since as a rule only small regions require refinement, the overhead involved in mesh refinement is essentially negligible. The timestep, however, is set by requiring that the maximum Courant number be less than unity, so the running time increases inversely with the minimum zone size.

If the physical feature requiring enhanced resolution disappears, the algorithm automatically does away with the extra triangles which had been introduced. Because it "remembers" the original triangulation, it is able to restore the grid to its exact form prior to the refinement. This type of algorithm lends itself readily to vectorization, and a mesh change (performed every 5–10 timesteps) requires only $\sim 100\ \mu s$ on the Cray X-MP-12.

The techniques described here have been extended to axisymmetric (r-z) systems. They have also been used to construct a three-dimensional code (based on tetrahedra), which has been used to solve several supersonic steady-state problems.

A Barely Implicit Correction to FCT for Nearly Incompressible Flow

For explicit finite-difference schemes the timestep Δt is restricted by the Courant condition $\max[(c+v)\Delta t/\Delta x] \leq 1$, where Δx is the zone size and the maximum is taken over the whole mesh. In many nearly incompressible fluid dynamics systems the flow velocity v is much less than c, the speed of sound. We are usually interested in phenomena which take place on the slow time scale, so it is of interest to develop implicit algorithms for which the limiting condition becomes $v\Delta t/\Delta x \leq 1$. Since positivity-preserving techniques are needed to maintain sharp concentration or other gradients, we are motivated to construct an implicit FCT algorithm (Patnaik, *et al.*, 1986).

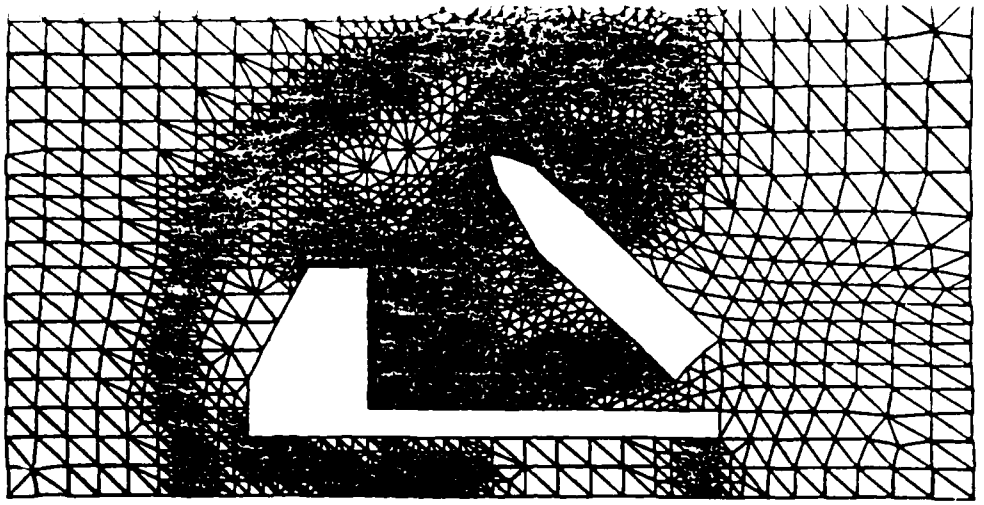
Analysis by Casulli and Greenspan (1984) has shown that the only quantities in the finite-difference form of the fluid equations which have to be differenced implicitly (i.e., defined on the advanced time level) are the pressure in the gradient term of the momentum equation and the velocity in the divergence term of the equation for the energy density E. The other terms of the equations can be differenced explicitly, in the present case by using a form of JPBFCT (Boris, 1981). The essential computational steps include an explicit prediction of density, momentum, and energy, determination of an implicit pressure correction by solving an elliptic equation, and corrections to the momentum and energy obtained from the pressure correction.

This procedure (called Barely Implicit Correction, or BIC) can be readily generalized to work with other explicit schemes, including FEM-FCT. To date, BIC has been implemented in one- and two-dimensional Cartesian explicit FCT routines. Figure 3 illustrates the ability of BIC-FCT to propagate a contact discontinuity without the introduction of additional diffusion. Figure 4 shows the damping of sound waves by BIC-FCT. Damping is seen to be negligible when the method is made semi-implicit, i.e., when the Crank-Nicholson parameter ω satisfies $\omega = 0.5$. Damping increases if the timestep or zone size is increased. In these and other problems to which it has been applied the Mach numbers were as low as 0.01, so that the timestep was up to an order of magnitude longer than would have been possible in an explicit code. The time required for one computational timestep compares very favorably to that required by the explicit two-step JPBFCT module.

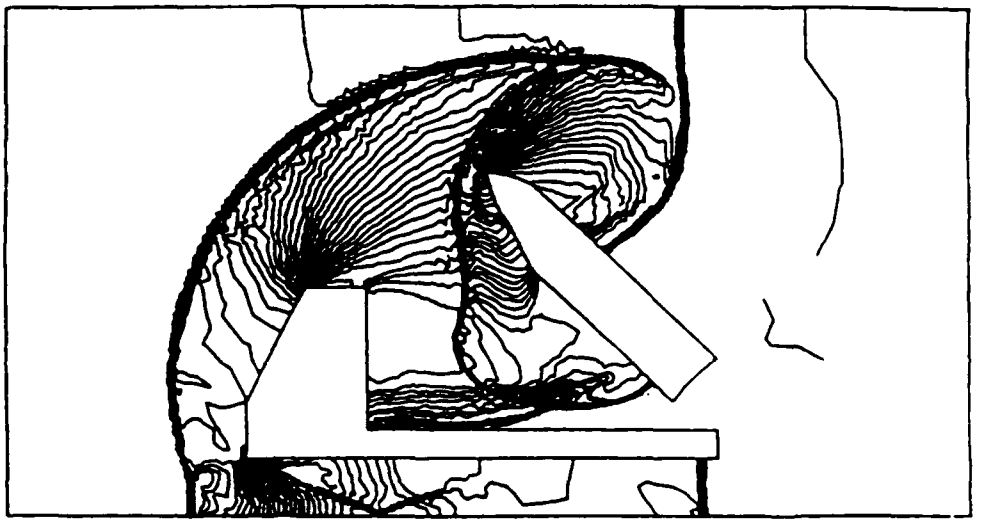
Applications have been made to two-dimensional flames ($v \sim 10$ m/s, $c \sim 300$ m/s) and to the transition to turbulence in jets. The decrease in running time permitted by barely implicit differencing makes it possible to include more detailed chemistry models in such simulations.

References

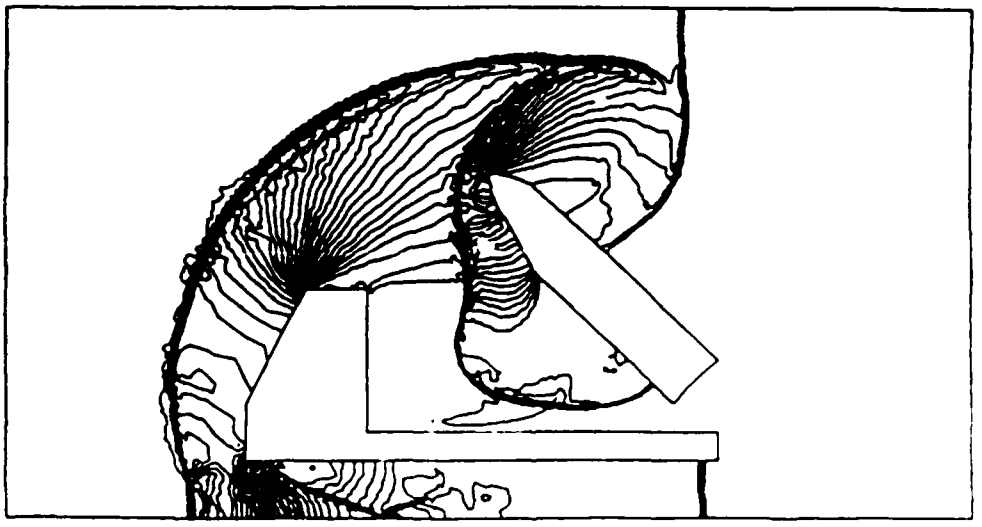
1. J. P. Boris, "ETBFCT: a fully vectorized FCT module," in *Finite-Difference Techniques for Vectorized Fluid Dynamics Calculations*, Ed. D. L. Book, Springer-Verlag, New York (1981).
2. V. Casulli and D. Greenspan, "Pressure gradient method for the numerical solution of transient compressible fluid flows," *Inter. J. Num. Methods in Fluids* **4**, 1001 (1984).
3. R. Löhner, "An Adaptive Finite-Element Scheme for Transient Problems in CFD," *Comp. Meth. Appl. Mech Eng.* (submitted 1986).
4. R. Löhner, K. Morgan, and O. C. Zienkiewicz, "An Adaptive Finite-Element Procedure for High-Speed Flows," *Comp. Meth. Appl. Mech. Eng.* **51**, 441 (1985).
5. R. Löhner, K. Morgan, M. Vahdati, J. P. Boris, and D. L. Book, "FEM-FCT: Combining unstructured grids with high resolution," *J. Comput. Phys.* (submitted 1986).
6. C. Patnaik, J. P. Boris, R. H. Guirguis, E. S.Oran, "A Barely Implicit Correction for Flux-Corrected Transport," *J. Comput. Phys.* (submitted 1986).
7. S. T. Zalesak, "Fully multidimensional Flux-Corrected Transport algorithms for fluids," *J. Comput. Phys.* **31**, 335 (1979).



(*a*) Grid

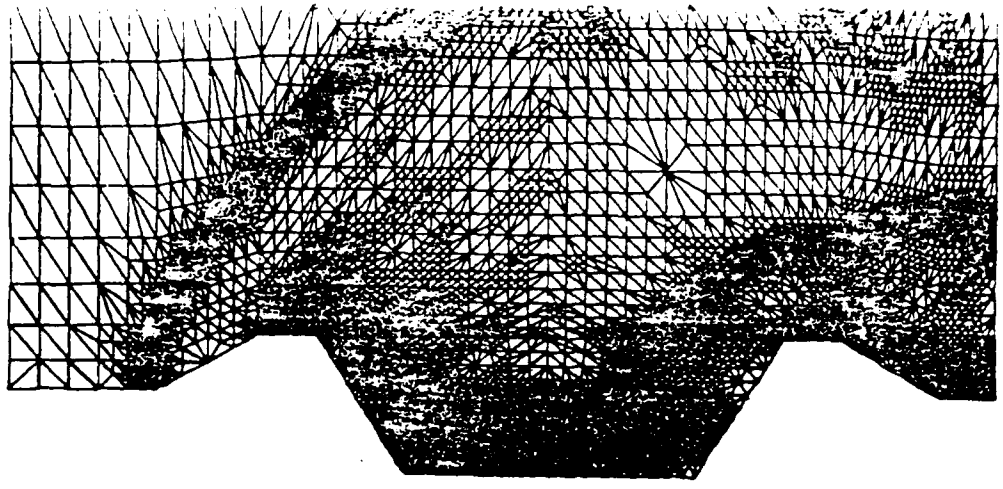


(*b*) Density

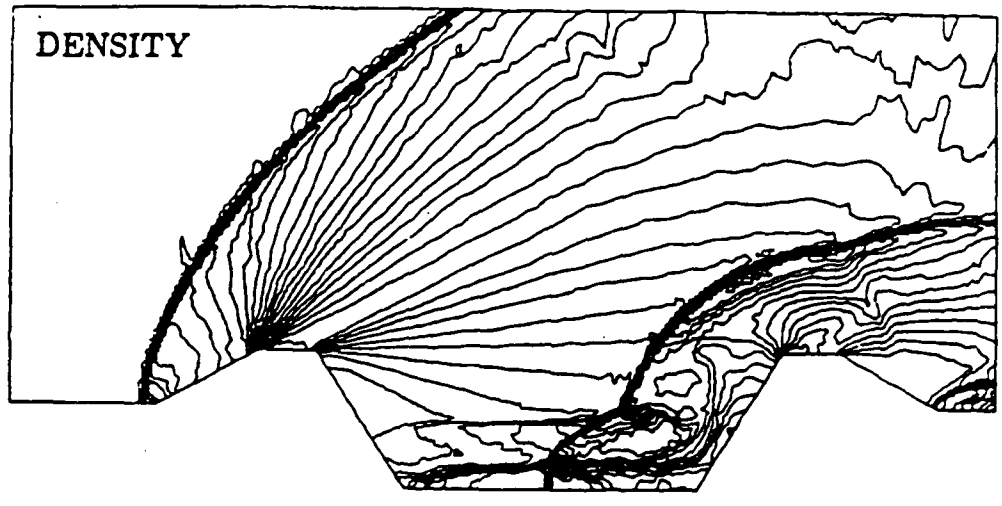


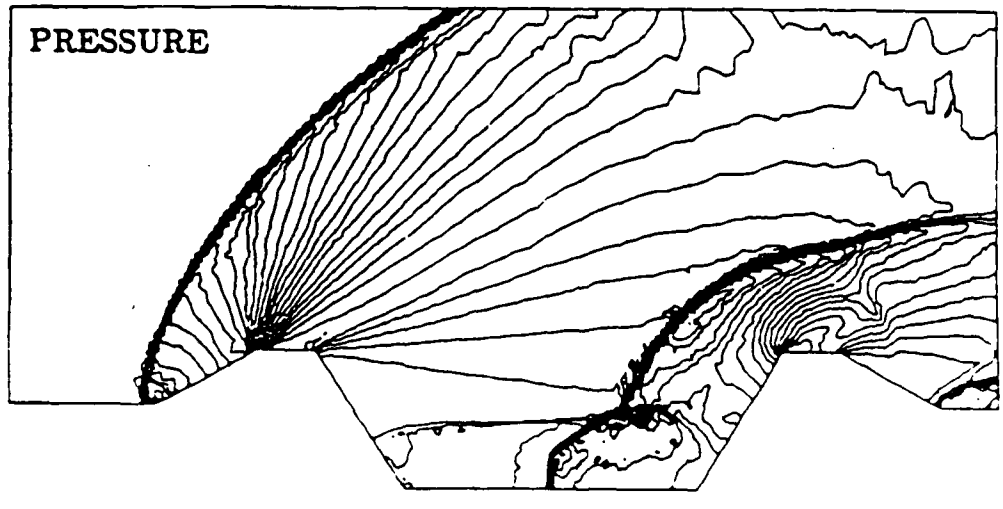
(*c*) Pressure

Figure 1



NELEM=13681, NPOIN=6984





T=0.12

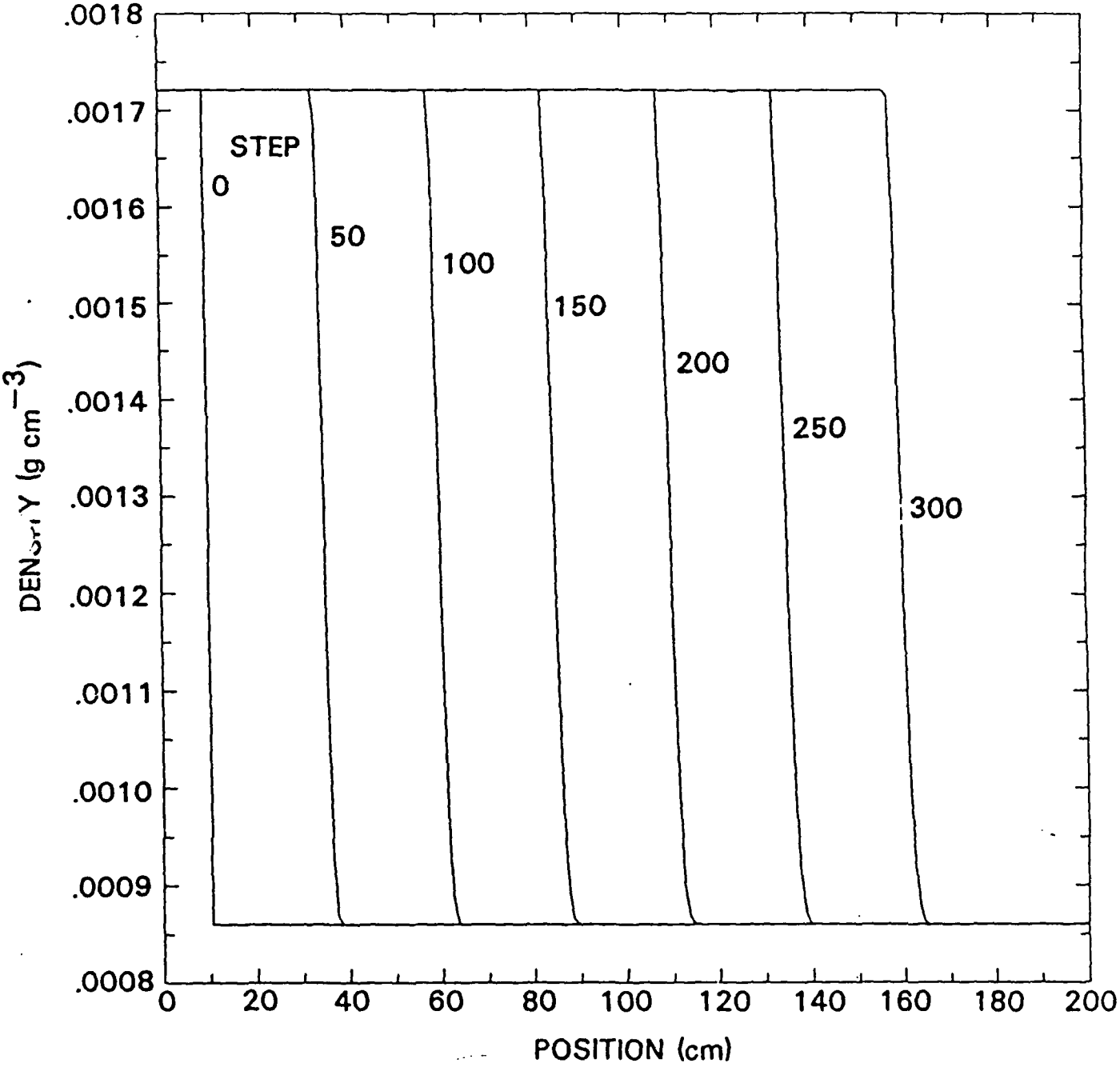

.0018
.0017
.0016
.0015
.0014
.0013
.0012
.0011
.0010
.0009
.0008
STEP
0
50
100
150
200
250
300
0
20
40
60
80
100
120
140
160
180
200
POSITION (cm)

Figure 3

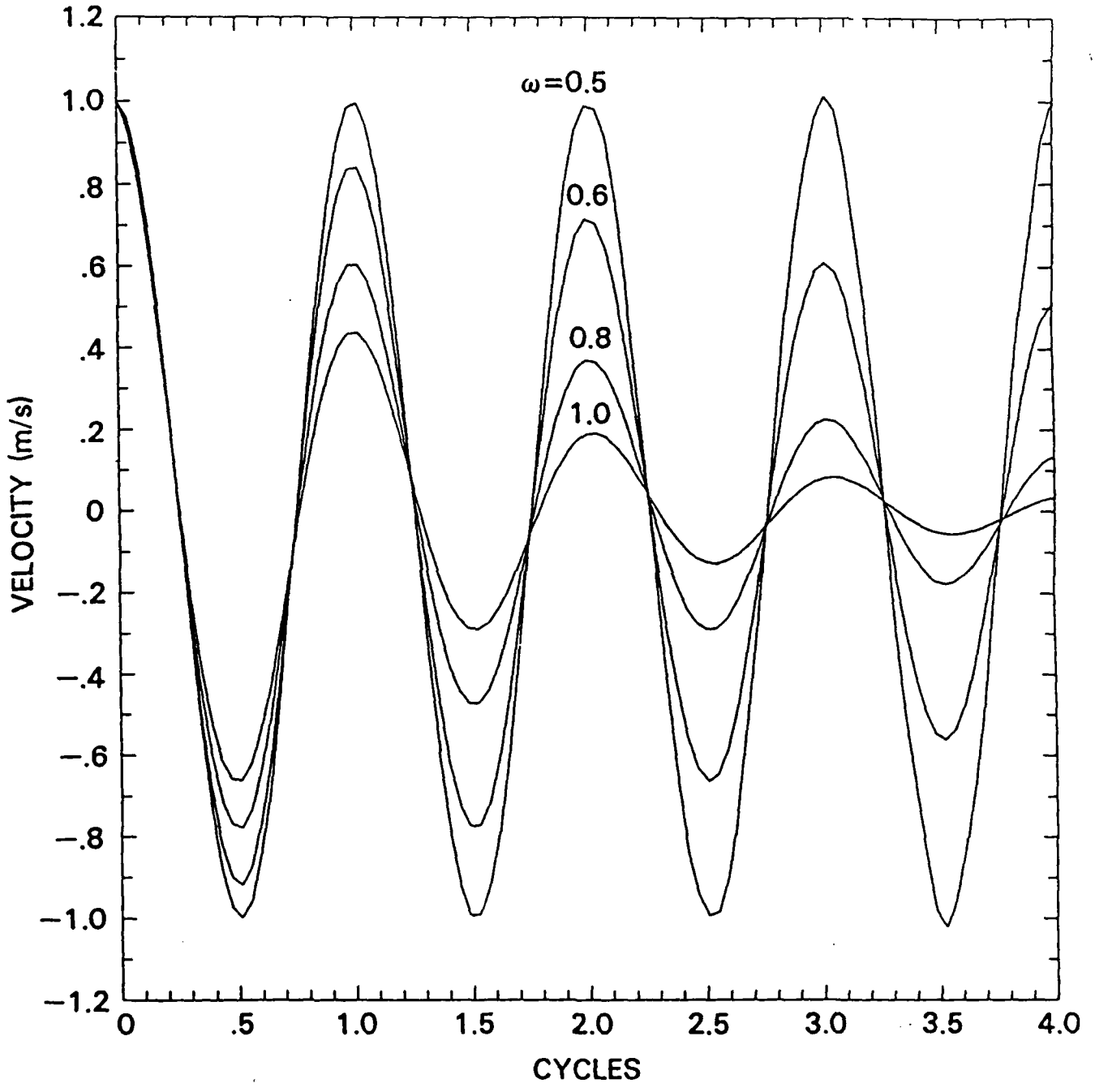

1.2
1.0
.8
.6
.4
.2
0
−.2
−.4
−.6
−.8
−1.0
−1.2
VELOCITY (m/s)
ω=0.5
0.6
0.8
1.0
0
.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
CYCLES

Figure 4

APPENDIX C.

Documentation of VOYEUR

Table of Contents

The following trademarks are used in this manual:

Tektronix 4115B is a trademark of Tektronix, Inc.
DICOMED is a trademark of Dicomed, Inc.
APTEC is a trademark of Aptec Computer Systems, Inc.

Introduction

VOYEUR is an interactive graphics system designed to allow users to "watch" their calculations as they progress. This gives the user a useful tool with which to visually inspect the correctness of his calculation. VOYEUR (on the Tektronix 4115) also allows the user the option of two types of permanent recording - color hardcopy (with a Tek 4115 compatible printer/plotter) or DICOMED graphic output. Its menu driven system of commands is designed to get the user up and running with a simple set-up in a short period of time and then allow him to build from there for more complicated plots.

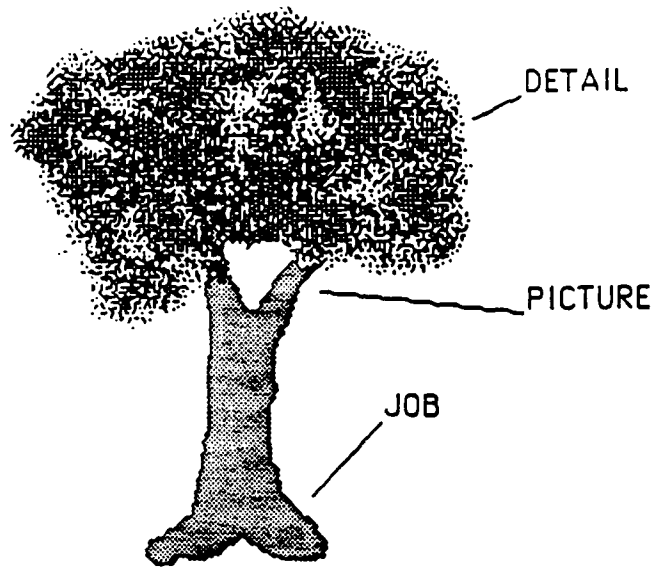
Within VOYEUR there are structures - think of them as objects - that form a three level hierarchy. A complete tree diagram of all the commands appears at the end of this manual.

The first structure is the JOB. The JOB isn't a real object in that it doesn't manifest itself directly on the screen. Rather, a JOB is an organizational tool allowing you to group related PICTURES and thence DETAILS together.

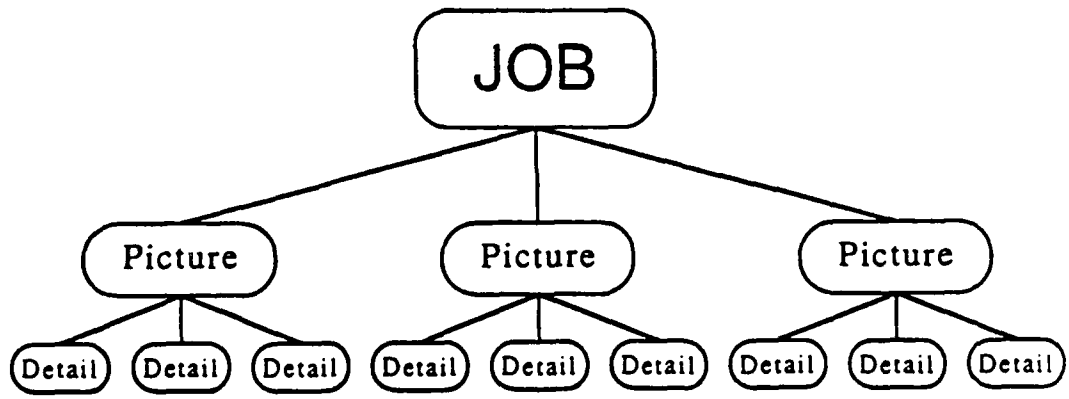
The second level to the hierarchy is the PICTURE which we will discuss in more detail in the PICTURES section of the manual. Each picture has a window attached to it which can display simulation results in one of four different formats. There are four kinds of PICTURES in VOYEUR. They are called PIXEL, VECTOR, GRAPH and SEGMENT. PICTURES from more than JOB can be displayed on the screen at the same time. This allows several different calculations, perhaps originating from different processors, to be watched concurrently.

The third and final level to the hierarchy is the DETAIL - it will be discussed in the DETAILS section. There are four kinds of DETAILS in VOYEUR - they are LABEL, DATA, AXIS and BORDER DETAILS. DETAILS are always attached to a PICTURE, although there can be many different DETAILS (or none) attached to a PICTURE.

If you were to use your imagination, you could think of this hierarchy as forming a tree shape. The JOB would be the root and trunk of the tree with the PICTURE being the branches and the DETAIL being the leaves. In fact, it might look something like this:



If, perhaps, you are more scientifically inclined, it might look more like this:



In this manual we will "climb" that tree. Starting at the root, we'll cover the commands that create and modify a JOB. Moving to the branches, we'll take a look at PICTUREs and how to create and change them. We'll have reached the leaves of the tree when we discuss DETAILS.

Afterwards, we'll come back to the ground and discuss some of the things that help hold this tree upright. We will look at the TV Guide and the Color Map and then take a peek at MACROs. By that time, you should have a very complete understanding of how to use VOYEUR. There will be some practice files available to help you on your way.

At the end of this manual, there are two helpful sections. The first is the VOYEUR Quick Reference - it lists all the commands in the VOYEUR command tree along with some brief explanatory text. The second is a command card - it's really a pictorial representation of the Quick Reference without the comments.

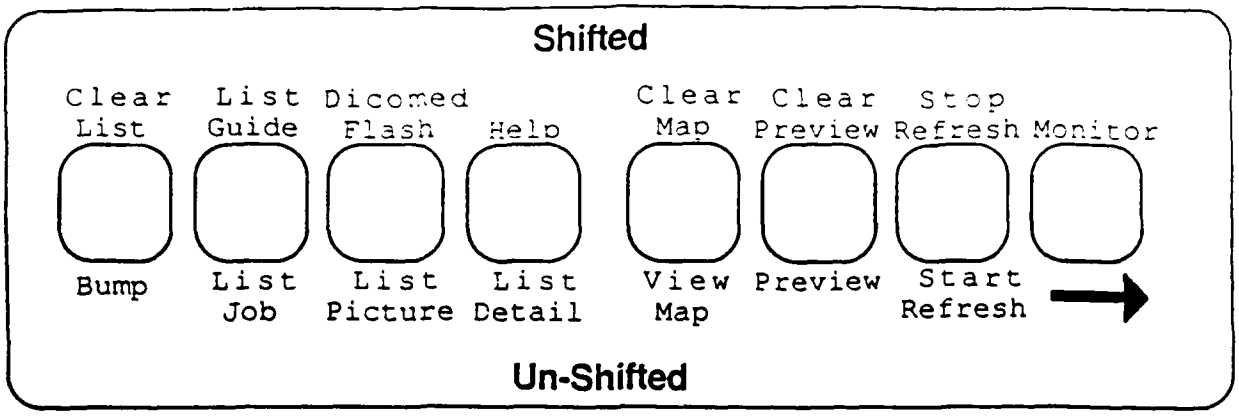
EXECUTING VOYEUR

VOYEUR is started from the DCL command line by typing in the keyword VOYEUR followed by an optional TV guide name (TV guides are explained in a later section). After starting VOYEUR, the top of the screen will contain a list of menu choices, one of which is highlighted. This highlighted selection is called the active command. The second line contains a statement describing the active command. The right most key in the second set of function keys at the top of the keyboard is marked with a large arrow pointing to the right. Depressing this key selects the command to the right of the active command, continuing to the right most command, then wrapping around to the beginning of the line. In this way, a simple help message can be examined for each available command.

Commands can be executed in one of two ways. Depressing the RETURN key executes the highlighted or active command. Another shorter mechanism is to simply touch the key corresponding to the first letter of any visible command. This is the mechanism used to describe each of the commands in this manual. The character entered to execute the command is in **Bold**, and the remaining portion is in *italics*. The first character of each command on a line is always unique.

Executing a command will either bring up a new set of commands on the top line, or it will perform an action. In order to "back up" to the previous command level, press the BUMP key, which is the left most key on the top of the keyboard. Successive depressions of the BUMP key will bring you back to the top of the command tree.

Commands which require more information to execute will normally inquire for the data on the second line. After entering the answer to the questions presented, it is necessary to press the RETURN key.



The picture at the top of the page is of the sixteen function keys that can be found at the top left of the Tek 4115B keyboard. VOYEUR gives each of these keys a definition. They represent commands that work similarly to the commands you are used to entering by using the first letter. The ones on the top are generated by shifting the key (holding down the shift key while pressing the appropriate function key).

Perhaps the most important of these is the Bump button. As you enter successive commands in VOYEUR, you go deeper into a tree. The Bump button moves back up one level closer to the top command line each time it is pressed. To go to the top command line from anywhere in the command tree should take no more than six touches of the Bump button.

The next most useful of these is the Monitor button. Anytime after you have put a PICTURE into the display list, pressing the Monitor button will cause VOYEUR to start displaying those selected PICTURE's. VOYEUR will keep refreshing the screen until it is told to stop.

You can start and stop the refreshing of the screen with the Start Refresh and Stop Refresh buttons. They work just as their name implies Start Refresh will cause VOYEUR to start refreshing the screen and Stop Refresh will cause VOYEUR to stop refreshing the screen. You should note that the Monitor button will not restart the refreshing of the screen after the Stop Refresh button has been pressed.

Two useful buttons are the View Map and Clear Map buttons. These will cause VOYEUR to either put a bar showing each of the colors at the bottom of the screen (View Map) or to remove it (Clear Map). The

location of the color bar cannot be changed. It always appears at the bottom of the screen.

There are two buttons that are very useful when you are deciding on the layout of your screen. They are the Preview and Clear Preview buttons. The Preview button works quite a bit like the Monitor button, but it doesn't cause VOYEUR to start displaying the actual PICTURE data. Instead, a filled box the size of each PICTURE's window is put on the screen along with DETAIL's. You can use this to set things up even when there is no data to display in the mass memory of the APTEC I/O Computer system. As its name implies, the Clear Preview button wipes the preview off the screen.

There are five buttons used to look at your definitions. They are the List Guide, List Job, List Picture, List Detail and Clear List buttons. As with the rest of the function keys, their purpose is readily deduced from their names. The list Job will display the current active Job definitions (name, system file etc.). The other Lists work similarly. To remove this list from the screen, simply press the Clear List button.

The final button is the Dicomed Flash button. Pressing this button will tell VOYEUR that you want a snapshot of the screen to be written into a Dicomed file. For more information on using the Dicomed features of VOYEUR, take a look at the "Getting Hardcopy" section of this manual. Note that the Help key has not yet been implemented.

JOBS

As we saw in the Introduction, the root of the hierarchy in VOYEUR is the JOB. A JOB represents a collection of related PICTURE's and thence DETAIL's. For example, you might group all your different temperature PICTURE's under one JOB. Or perhaps, you might put the temperature, density, and velocity PICTURE's for a certain kind of calculation under one JOB.

Creating a JOB

No matter what organization you wish to use, you will still create a new JOB the same way. From the root of the command tree, enter the following command sequence:

E*dit* G*enerate* J*ob*

Remember to enter "E", then "G", then "J", and that no Returns are necessary. Once you do this, VOYEUR will ask you for a name to identify this new JOB. You should think of an appropriate name (one that you will remember, because you will need to know this later) and enter it followed by pressing the RETURN key. One thing to remember is that JOB names (in fact ALL symbols and names within VOYEUR) are case sensitive. That means that the names 'foobar' and 'FOOBAR' are not the same.

The next thing that VOYEUR will ask you will be the name of the JOB's system file (if there is to be one). The system file is a mass memory file that contains parameters used by the simulations. These two items are all that VOYEUR needs to set-up a JOB. You can go from there to generate new PICTURE's and DETAIL.

Copying an existing JOB

Another way to create a new JOB is to copy it. When you copy a Job, everything that is contained by that job is copied also. The command sequence for copying Jobs is:

Edit Copy Job

VOYEUR will then prompt you for two things: the name of the Job to copy and what to call the new Job.

Changing an existing JOB's System File

If you decide to change the name of a JOB's system file, you can use another command sequence. It is :

Edit Modify Job

VOYEUR will ask a question that in effect means "Do you wish to modify this JOB?", and will give you the name of the current active JOB. If you do, then press RETURN, otherwise, type 'no' and then press RETURN – VOYEUR will ask you for a new JOB name. After VOYEUR has the right JOB, it will ask you for the name of the JOB's system file. You should give the full name of the new system file and press RETURN.

Deleting an old JOB

In VOYEUR, you can remove old JOB's very easily. All you need to do is enter this command sequence:

Edit Remove Job

VOYEUR will now ask you for the name of the JOB you wish to remove. Just type in the name and press RETURN. You should remember that names are case sensitive. BE CAREFUL! Once a JOB is removed, there is no way to instantly bring it back! It is always a good idea to save everything to a TV guide before deleting a JOB.

PICTURES

Pictures, it has been said, are worth a thousand words. Well, pictures are what VOYEUR is all about. The whole idea is to display your thousand words' worth of pictures with enough speed and style to make it all worthwhile. Of course, there are all kinds of pictures, but with VOYEUR, you need be concerned with only four kinds. VOYEUR refers to them as PIXELs, GRAPH's, SEGMENT's and VECTOR's. Each kind of Picture has different attributes and different associated commands. ~~They also share some common requirements.~~

All the Pictures that VOYEUR displays share some common requirements. For instance, the data describing all four Picture types must be stored in the mass memory of the APTEC I/O computer system. Furthermore, unlike most files, it must be contiguous. That is to say that the data cannot be stored as little pieces scattered throughout the memory. If it is, VOYEUR will give you some very strange results.

If your Picture's data file is being created as VOYEUR is running, it must be opened as a shared file so that both VOYEUR and your creating process can access it at the same time. Each type of PICTURE uses a different format for the data stored in mass memory.

PIXEL

The first type of Picture that we will discuss is the PIXEL Picture. In its simplest form, a PIXEL Picture is a collection of numbers representing the color of each pixel in the Picture. It is simply one byte value per location in the screen window stored in mass memory as a contiguous array of data. You create a new PIXEL Picture, after you have created a JOB to contain it, with this command sequence:

***E**dit* ***G**enerate* ***P**ixels*

You should remember that each command is entered by pressing its first letter with no use of the RETURN key. VOYEUR will ask you if this new Picture is to be attached to the current active JOB. If this is the case, simply press RETURN. If you want your new Picture attached to

another JOB, then say no and then give VOYEUR a new JOB name – the one this new Picture is supposed to be attached to. VOYEUR will now ask you what you want this new Picture to be named. Just think of a suitable name and enter it followed by pressing RETURN. Next, VOYEUR will ask you for the name of the mass memory file that is to contain the data describing your Picture. These are the only parameters that the operator must enter when generating a PICTURE. There are several characteristics that can be modified however and the next step gives you the opportunity to change them. What you will see now is a command line that looks like this:

N*ame* **O***ffset* **L***ocation* **W***idth* **H***eight* **F***ile-size* **P***ixel* **R***efresh*

This is identical to the command line presented while modifying the Picture using the **E***dit* **M***odify* **P***icture* command. You can change the mass memory file name where VOYEUR will get the PIXEL data with the Name command. This will cause VOYEUR to prompt you for the name of the mass memory file in which you are storing your data. This file must reside on the APTEC mass memory disk, and it must be contiguous. It need not exist when creating the Picture entry, but it must be present before attempting to display it or an error will occur. The very next thing that you will want to do is tell VOYEUR how wide and how tall your Picture is supposed to be. This is accomplished with the **W***idth* and **H***eight* commands. The range for widths is 0 to 1280 and the range for heights is 0 to 1024. PIXEL uses a different scale here than any other type of PICTURE or DETAIL. Each dimension is measured in absolute screen coordinates. The dimensions of the Tektronix 4115 is 1280 x 1024. In every other case in VOYEUR, dimensions are based on a scale of 4096 x 3277, what Tektronix calls terminal coordinates. This difference is required since the data in mass memory is mapped directly to the screen, one byte per pixel, and the user must be able to specify the exact dimensions of the window.

Every new Picture created by VOYEUR starts out in the lower left-hand corner of the screen. Most of the time, you will want to put your Picture someplace else on the screen. By entering the **L***ocation* command, you can tell VOYEUR where on the screen you want your Picture to be displayed. You select the location not by entering coordinates, but by rotating the thumb wheels on the keyboard until the outline of your Picture is where you want it to be. You should remember to set the width

and height before you try to change the location of your Picture because a Picture that is zero pixels by zero pixels is very hard to see!

Sometimes, the "real" data of any one Picture doesn't start at the beginning of a file. It may be the case that you are putting two Pictures into the same file one after the other. In this case, you need to tell VOYEUR some more information in order for it to find the right data. The **O***ffset* command can be used to tell VOYEUR how many bytes into the file specified by the **N***ame* command the data for this particular Picture is stored. Suppose that you had two different Pictures stored in the same mass memory file. To display both of them, you would generate two Pictures – the first would have an offset of 0 and the second would have an offset of n – where n is the number of bytes used in the first Picture. Or, you could use the offset to look at just one portion of the Picture – sort of a close-up.

A command that is used frequently in conjunction with the **O***ffset* command is the **F***ile-size* command. The **F***ile-size* command is used to tell VOYEUR how many bytes of information a Picture uses. By default, VOYEUR assumes that file size is the number specified with the **W***idth* command times the number specified with the **H***eight* command. For example, suppose you had a Picture that was 10 by 10. VOYEUR would at first assume that the file size was 100. But what if you only wanted to look at the first fifty bytes of information? You would use the **F***ile-size* command with a value of 50 to tell VOYEUR to read only the first fifty bytes. If you wanted to look at the last fifty bytes, you would first give VOYEUR an **O***ffset* of 50 and then a **F***ile-size* of 50. With the **F***ile-size* and **O***ffset* commands, you can selectively pick any consecutive series of bytes to be displayed as one Picture.

On most screens, a pixel is a very small dot whose size cannot be changed. With VOYEUR, you can have pixels that are any size you want (within reason). You can also change the way individual pixels are written to the screen. All the commands for these are in a sub-level that can be reached with the **P***ixel* command. After entering it, you will get a command like looking something like this:

W*idth* **H***eight* **A***xis* **M***ode* **T***ype*

You use the **W***idth* sub-command to tell VOYEUR how many units wide you want each pixel to be. This **W***idth* can be any integer in the range of -1280 to 1280. The **H***eight* sub-command is used similarly – with a range of -1024 to 1024. You should remember that changing the size of the pixels will change the size of the overall Picture.

Arbitrarily, VOYEUR writes pixels from left to right starting at the top of the screen and working down – it starts by stacking pixels horizontally. You can change to a vertical stacking with the **A***xis* sub-command. You may have noticed that pixel widths and heights can be negative. This doesn't mean that you will have pixels that collapse in on themselves, however. A negative pixel dimension will change the direction of pixel writing. This can be used in conjunction with the **A***xis* command to flip the Picture. For example, if you specify a negative pixel width (not changing the axis) the pixels will be drawn from right to left – performing a horizontal flip of the Picture. A negative pixel height will cause your pixels to be written from bottom to top – thus performing a vertical flip. Note that data is always read from mass memory from the beginning of a file in a continuous stream to higher addresses.

There are really two types of pixel data. The simplest to understand is known as raw pixel data. That is the one point one value method – no encoding of the data to make it smaller. The other type of pixel format is called run-length encoded data. VOYEUR can use both types of data – but they can't be interchanged in the same Picture. To tell VOYEUR which kind of pixel data you are using, you would use the T*ype* sub-command. VOYEUR will prompt you for the pixel type. You should give VOYEUR a value of 0 if you are using raw data or a value of 2 if you are using run-length encoded pixel data. There is a good discussion in the 4115 Option 3A manual (look under PX: format DMA transfers) that explains the difference. Run-length encodes pixel data uses 3 bytes per code group. The first two contain the repeat count, up 64K, and the third byte contains the pixel value to be repeated. It is useful when large portions of the screen are to be painted with the same color, but can take up a lot more space and be slower if a lot of DETAILS are being painted. T*ype* values of 1 and 3 mean that the pixel data is unnormalized. This is useless within VOYEUR, since there is no way to specify surfaces.

The final pixel sub-command is the **M***ode* sub-command. With it, you can change the pixel ALU writing mode. This specifies the way the pixel data in memory is combined with the new pixel data overwriting it. If A is the value currently in the pixel, and B is the new value being written, the following table describes the value resulting in the pixel position after writing with all of the mode values that are valid on the 4115.

0	no change
7	A XOR B
11	B (i.e. just overwrite the pixel)
12	A AND B
15	A OR B
17	A + B
18	A - B

The most common and default value is 11, which just results in the new data overwriting the old. Another useful value is 7, which results in a pixel value of 0 wherever the new pixel and the old pixel are the same, and a different color everywhere else. It could be used to determine where successive plots have changed for example. I have yet to find any earthly use for the rest of these modes, but they are there if needed. If anyone comes up with a snazzy use for the different ALU modes please let me know.

REFRESH

The process that is creating all this pixel data is presumably running the whole time and generating new frames. This means that the Picture you see on the screen needs to be updated at some specifiable interval. You can tell VOYEUR how you want your Picture refreshed with the commands that come under the **R***efresh* command. The refresh parameters specify conditions under which the picture can be delayed. That is, VOYEUR will hold up and do nothing until the particular refresh conditions are satisfied. Thus, if you have more than one picture on the screen, normally the first picture will have some additional conditions attached to it, while the rest will have the continuous attribute. The effect of this is to wait until the particular refresh conditions are satisfied, then draw all of the pictures together. The screen will then freeze until the next set of refresh conditions are valid.

Entering the **R***efresh* command will give you the following command line:

Continuous Time Synchronize Lockstep Value

If you want your Picture to be updated all the time, you should use the **C***ontinuous* sub-command. This means that there are no refresh conditions, and VOYEUR will start re-drawing the Picture the instant it has finished the previous one.

The **T***ime* option is used to re-draw a picture at intervals. As soon as a picture is started to be drawn, the current time is saved. When attempting to draw the picture again, VOYEUR will hold up until the previous time plus an interval has elapsed before re-drawing that picture. Hence, if it takes 5 seconds to re-draw the screen, putting a delta time of less than 5 seconds will have no effect. After entering the **T***ime* sub-command, VOYEUR will prompt you for a delta time between re-draws. This should be entered in the standard VAX delta time format:

dddd hh:mm:ss.cc

where d stands for day, h for hours, m for minutes, s for seconds and c for hundredths of a second. You can omit the values, but must always use the leading delimiters. This means that two minutes can be entered as either '0000 00:02:00.00' or '0 :02'. As another example, 5 seconds should be entered as '0 ::05'.

The third option for refresh parameters is to synchronize the re-drawing of the Picture with a parameter stored in the APTEC mass memory system file. Normally, this is used with a variable containing the time step. You tell VOYEUR the specifics with the **S***ynchronize* command. VOYEUR will ask you a series of three questions. The first thing it will ask for will be the offset, from the beginning of the containing JOB's system file, for the parameter. This should be a multiple of four as the values in the system file are always four bytes long. The second question will be the parameter increment. The third and final question will be whether or not you wish VOYEUR to hold the calculation while it is refreshing the Picture.

This is the only mechanism where the programs in the array processors handshake with VOYEUR. In order to use this mechanism, the array processor must be running a program capable of this handshake protocol. The array processors should read the value in mass memory and if it is negative, the array processor calculates a new set of pictures based on that value. When the calculation is complete, the array processor should write a 0 into the parameter, then wait, looking for a negative value again. VOYEUR, meanwhile, will write a new negative value into mass memory whenever the calculations should proceed, and wait until the mass memory parameter becomes 0 before displaying anything.

If the parameter value is positive, the array processor should produce new pictures continuously. The handshake mechanism will fail if a positive value is entered for the parameter increment value, but the operator could create a data detail with a reference to the same offset, and write a positive value into that location.

Normally, calculations of physical problems are performed interactively, each iteration being one step. This parameter usually is the number of steps between pixel dumps to the mass memory. For example, say that -10 is entered as an increment parameter. VOYEUR will write -10 to start the calculations, the array processor then performs 10 steps, dumps the results to mass memory as a PIXEL file, and halts, writing the memory parameter to 0. VOYEUR then detects the 0 and draws the pictures, and writes the parameter back to -10, thus starting the entire process over again.

The calculation can be held or not. If it is held, then VOYEUR will only write the memory parameter with 0 after all of the pictures have been drawn. Otherwise, VOYEUR will immediately write the parameter and the array processors can continue calculating while VOYEUR is refreshing the screen.

A fourth option is also to have the Picture refreshed with a step value. This is done with the ***Lockstep*** command. The array processors calculate continuously, producing PIXEL data during some interval number of steps. The lock step option asks for an increment. The current value of the memory parameter is read at the beginning of the monitor process. Thereafter, VOYEUR will wait until the value in that parameter is greater than or equal to the initial value plus the specified

increment. Normally, this is the iteration count, or time step. The questions asked are the same as the first two for the *Synchronize* option.

Finally, the fifth refresh option is to have the Picture re-draw be based on a specific value or flag in the mass memory. This is accomplished with the *Value* sub-command. VOYEUR will ask you for the value's offset in the system file and what value it should look for to signal a refresh. VOYEUR will simply wait until the array processors write that value into memory before refreshing. If the value is not changed, this becomes a single trigger, because the refresh condition is automatically satisfied from that point forward.

Graph

The second type of Picture we will discuss is referred to as the Graph. Graph data is stored in mass memory contiguously just like PIXEL's data, except that the values are strings of valid 4115 escape sequence commands. They must be filled out to the nearest 512 byte boundary, unless you explicitly change the file size. Use carriage return, space, or line feed characters to null fill, since they are interpreted by the 4115 as NOP's. Any escape sequences can be specified, but be careful specifying segment numbers below about 1024, since VOYEUR stores all of its DETAIL as separate segments. It is quite possible to destroy the screen data, and/or get all kinds of invalid parameter errors by redefining or deleting these segments. Normally, it is not necessary to set up the window or viewport, since VOYEUR lets you do it from the command line.

To get started making a VOYEUR Graph, you use the following command sequence (again from the top command line):

Edit Generate Graph

VOYEUR will ask you if this new Picture is to be attached to the current active JOB. If this is the case, simply press RETURN. If you want your new Picture attached to another JOB, then say no and then give VOYEUR a new JOB name – the one this new Picture is supposed to be attached to. VOYEUR will now ask you what you want this new Picture to be named. Just think of a suitable name (perhaps the hardest thing to do in VOYEUR!) and enter it followed by RETURN. Next, VOYEUR will ask you for the name of the mass memory file that contains the data for you

Picture. You should enter the full name of the file to avoid any possible ambiguities that might arise.

Now, after all this, you will get a command line that is similar to the Pixels command line:

Name Offset Location File-size Refresh Size Window

The first five commands should look familiar to you. They each work the same way that their Pixels counterparts do. The two new commands are ***Size*** and ***Window***.

Remember the Pixels commands of ***Width*** and ***Height***? Well, the ***Size*** command is really telling VOYEUR similar information except that this command will not affect the ***File-size*** value (which now gets its information from the file system on the APTEC). When you enter the ***Size*** command VOYEUR will prompt you for the window size for your Graph Picture. This ***Size*** is a pair of integers (width, height) with width in the range 0 to 4096 and height in the range 0 to 3276 .This command only establishes the screen size for your Graph and should be used before the ***Location*** command so that there is something of enough size to see to move!

The ***Window*** command is really a path to several sub-commands that affect the characteristics of your Graph's screen window. This window defines the viewport that is used to show all of the escape sequence commands. After entering this command, you will see this new command line:

Border Wipe Origin Extent

The ***Border*** sub-command is a way for you to create and/or alter the border around your Graph. This border surrounds the viewport and can be turned off and on with the BORDER key on the lower right side of the keyboard. You can use the ***Index*** sub-sub-command to tell VOYEUR what color you want the border to be. The ***Visibility*** sub-sub-command is use to tell VOYEUR if the border is to be seen or not. Pressing the RETURN key after entering this command will toggle the border visibility.

You can use the **W***ipe* sub-command to specify the index used when the window is 'erased' prior to re-drawing. This becomes the background color used for drawing your data, so you should choose an index that mixes well with the colors selected for displaying your data. In other words, a red background would make red vectors difficult to see.

The **O***rigin* sub-command is used to specify what numerical coordinates correspond to the origin of the window. This is used in conjunction with the **E***xtent* sub-command which tells VOYEUR what numerical coordinates correspond to the point maxwidth, maxheight in your window. Notice that if the origin is 0,0, and the extent is 32767, 32767 (the default), then a vector drawn from 0,0 to 32767, 32767 will go from the lower left corner to the upper right corner. Vectors will be clipped if they attempt to draw outside the window.

Segment

The third type of Picture in VOYEUR is called the Segment. Segment pictures use the format described under the SG: parameter in Chapter 3 of the 4115 Option 3A manual. The basic layout contains a segment number followed by picture processor commands. Picture processor commands are described later. This PICTURE is created with the command sequence:

***E**dit **G**enerate **S**egment*

VOYEUR will ask you if this new Picture is to be attached to the current active JOB. If this is the case, simply press RETURN. If you want your new Picture attached to another JOB, then say no and give VOYEUR a new JOB name – the one this new Picture is supposed to be attached to. VOYEUR will now ask you what you want this new Picture to be named. Just think of a suitable name (perhaps the hardest thing to do in VOYEUR!) and enter it followed by RETURN. Next, VOYEUR will ask you for the name of the mass memory file containing the data for you new Picture. Give VOYEUR the full file name and press RETURN – you will now get a very familiar command line:

***N**ame **O**ffset **L**ocation **F**ile-size **R**efresh **S**ize **W**indow*

In fact, all these commands work just like their Graph counterparts. The only difference between a Graph and Segment the kind of data and the way that data is stored in the mass memory file.

Vector

The fourth and final type of Picture is called a Vector. This type of picture corresponds to the DS: parameter described in the 4115 Option 3A manual. It is identical with the Segment PICTURE, except that the picture processor commands are not headed by a segment specification.

Picture Processor Commands

Picture processor commands are binary sequences used to instruct the 4115 to do various things. They are fully documented in Chapter 11 of the 4115 Option 3A manual. There are several VAX FORTRAN callable subroutines that are available to write picture commands to a file in a format for display. They can be linked into a program on the VAX by including the following file specification in the link command line:

user2:[reusser.voyeur]vmove.obj

The following functions are supplied:

fno = vopen(name)

Opens a file for putting commands into of the specified name. The name is a character descriptor. fno is an integer returned which is then used for writing to the specified file. At this time only one file can be open at a time, although this will later be rectified.

fno = vopena(name)

Same as vopen, except the name is a null terminated ascii string rather than a character descriptor.

vmove(fno, i, j)

fno is the open file number returned from vopen or vopena, i is the x direction, and j is the y direction. This is an absolute location and is relative to 0,0. Notice that changing the window origin in the Vector command will offset this location. This is a 16-bit move command.

vdraw(fno, i, j)

Draw a vector line from the previous position to the specified i,j coordinate pair. Note that i,j is relative from the previous location. This is a 16 bit draw command.

vcolor(fno, icolor)

Changes the color of the succeeding vectors and/or markers. The color can be viewed by executing a view color map command.

vstyle(fno, istyle)

Changes the line style of the succeeding vector commands. The various line styles are described in the 4110/4120 series command reference manual under the Set Line Style command.

vmarker(fno, itype)

Draws a marker at the current position of the specified type. Marker types are described in the 4110/4120 series command reference manual under the Set Marker Type command.

Changing Existing Pictures

Perhaps you don't want to create a new Picture, but only to modify an existing one. This is a very simple thing to do. From the root of the command tree, just enter this sequence:

E*dit* **M***odify* **P***icture*

VOYEUR will now put up a line containing the name of the current active Picture and ask you if that is the Picture you want to modify. If you say yes (by pressing RETURN) you will then get the same command line that you did when you created that Picture. If you say no, VOYEUR will then ask you if you want to modify a Picture attached to the current active JOB. If you say yes to this new question, you will then be asked for the name of the Picture you wish to modify and then get that Picture type's command line. If you say no, VOYEUR will ask you for the

Picture type's command line. If you say no, VOYEUR will ask you for the name of the JOB that contains the Picture you wish to modify and then ask you for the name of that Picture. Once again, you will get the same command line as when you created the Picture.

All the commands will have the same functions that they did when you created the Picture. The command line that you will get will depend on what type of Picture you select to modify.

Copying an Existing Picture

One quick way to generate a new Picture is to copy an old one. When you copy a Picture, the Picture type and the attached DETAILS come along with it. The command sequence for copying a Picture is similar to the one used to copy a Job:

Edit Copy Picture

Voyeur will then ask you for the new Picture name and the name of the Job that is to contain this new Picture. Next, it asks you for the name of the old Picture. You can modify this new Picture just like any other Picture.

Deleting a Picture

PICTUREs are deleted with the command sequence

Edit Remove Picture

VOYEUR will ask you if the Picture you wish to delete is in the current active JOB. If it is, just press RETURN and VOYEUR will ask you for the name of the Picture to delete. If the Picture you wish to delete is under another JOB, then say "no" to the current JOB question. Now VOYEUR will ask you for the name of the JOB that contains the Picture you wish to delete. Given this, VOYEUR will ask you for the name of the Picture to delete. Give VOYEUR the Picture name, and then you are done – the Picture is deleted from the memory of the machine.

DETAILS

DETAILS are parts of the Picture used to make the display more presentable. Detail can also be used to interact with the simulation(s) being performed. There are four kinds of Detail: LABEL, DATA, AXIS, and BORDER

DETAILS are not independent entities. Instead, they are attached to Pictures - much the same way that Pictures are attached to Jobs. The Detail's position is considered to be relative to that of its owning Picture with the location (0,0) being the lower left-hand corner of the Picture. This means that (-100,-100) is below and to the left and (100,100) is above and to the right and so forth. Normally the user will not have to worry about the position, since the thumb wheels are used to move DETAILS around. Moving a picture will automatically move all of the DETAILS attached to it.

Label

Labels are used to put a character string in the display. To generate a Label Detail is quite simple. From the root of the command tree, simply enter the following sequence:

E*dit* **G***enerate* **L***abel*

Remember that you only use the first letter of each command and that you don't use the RETURN key. Once you have entered the sequence, you will see a command line that looks like this.

C*haracter-size* **L***ocation* **S***lant* **R***otation* **I***ndex* **B***old* **T***ext*

The first thing that you will want to do is enter the text of the label. This is accomplished with the **T***ext* command. At the prompt, give Voyeur the text of your label, which you should enter as a single line terminated with RETURN.

The next thing to do is select the location of your label. To accomplish that, use the **L***ocation* command. You can use the thumb wheels to move the label to the desired location and then strike any key

to terminate the move and return to the label command line. Should you decide that you really did not wish to move your label, just press the Bump button and the move will be aborted. This will work only of you haven't struck a key - telling VOYEUR that you were finished moving your label. If the Bold parameter is not set to one, then the text will appear very strange as you move it, although it will be okay when you stop.

If you are trying to move a label and it looks like you are adjusting the location of a bowl of spaghetti, this is the problem. Simply select the **B***old* command and enter 1. The mechanism used for moving is to place the screen in XOR mode and write the data once to make it appear and then writing it in the same location to erase it. Bolding works by drawing the text in one location, then moving up and to the right and drawing it again, then down and to the left, etc. Thus moving bolded text will cause it to destructively interfere with itself. Re-select the proper bolding value after the detail has been moved.

The remaining commands at this level deal with the way in which the label is put on the screen. At first, your new label is displayed in what one might call "plain text." You can use these remaining commands to change the color of your text, its rotation, how italicized it is and how thickly it's drawn (using the **B***old* command discussed above). Lets say that you like big labels. At first, your label is written in size 13 characters, which results in about 80 characters filling one line across the screen. You can tell Voyeur to change the size of the characters with the **C***har-size* command. You give Voyeur an integer in the range 0 to 50 or so and terminate it with RETURN. If you use a size of 0, then you will have invisible characters This value is actually the number of pixels separating each character. Each character is 3*n wide and 4*n high where n is the size. Remember that this is on a scale of 4096 x 3277 for the entire screen.

Next, you may want to add a little pizazz to your label by italicizing it. This can be done with the **S***lant* command. You specify the slant in degrees from the vertical - a positive value is a slant to the right and a negative value is a slant to the left. Slants that are odd multiples of 90° will result in a label that is a straight line and should be avoided.

There will be times that you don't want your label horizontally on the screen. You can use the **R***otation* command to turn the label about the lower left-hand corner of the first character. VOYEUR will ask you for a number of degrees to rotate the label. This number can be either positive, meaning rotate clockwise, or negative, meaning rotate counterclockwise. There is no problem with rotations that are multiples of 90° as there is with the **S***lant* command. The **R***otation* command, combined with the **S***lant* command can be used to perform flips of your label Detail. The **R***otation* command is useful for creating label for the left or right side of a Picture.

You change the color of your label with the **I***ndex* command. Voyeur will ask you for the index of the color you wish to have your label drawn in. Indices are integers from 0 to n and correspond to one color in the current color map and n corresponds to the number of colors available in your system. You should remember that changing the color map will change the color that any one index represents.

Data

The second type of Detail is the Data Detail. The Data Detail is basically a Label Detail with an additional feature. You can read a value from the mass memory and attach it to a Data Detail. The data in mass memory is an actual value instead of a vector position or pixel color. This allows the addition of "important" numbers such as maxima or time steps to your pictures.

You generate a Data Detail in much the same way that you generate a Label Detail except the initial sequence is slightly different:

***E**dit **G**enerate **D**ata*

At this point, you will see a command line that looks similar to the Label command line.

***C**har-size **L**ocation **S**lant **R**otation **I**ndex **T**ext **B**old **O**ffset **W**rite*

The **C***haracter size*, **L***ocation*, **S***lant*, **R***otation* , **I***ndex* and **B***old* commands work the same as they do for Labels. The **T***ext* command is slightly different, however. Instead of entering plain text, you are really

entering the C version of a format statement. The general syntax of a C format line is :

text %<width.precision>**conversion_character** text

where the items in bold type are required. The percent sign indicates the beginning of a numeric definition. The width is the minimum size for an integer field and precision is the number of decimal places. The conversion character specifies the type of number that is to be displayed. The valid conversion characters for VOYEUR are:

d	convert to decimal format
o	convert to octal format (unsigned)
x/X	convert to hex format (unsigned) x specifies lowercase digits a-f X specifies uppercase digits A-F
e/E	convert to "E" format m..m . nnnnnE ± xx or m..m . nnnnne ± xx with case selection the same as for x-format and n's specified by precision (default = 6)
f	convert to floating point format m..m . nnnnnn with n's specified by precision (default = 6)
g	convert to d, e or f depending on which is shorter

One thing that you should remember is that there should be only one number field associated with any one Data Detail.

Several examples of the different formats are:

1. Text = %d

The value is written in decimal format. The number of characters used to display the value is exactly those required, without leading blanks. For example if the value of the data was 156, the above format would display:

Text = 156

2. Text = %e

The value is written in exponential format. For example if the value of the data was 156, the above format would display:

Text = 1.56xe02

3. Text = %g

This is the most common format. The value is interpreted as a floating point value and the format which results in the fewest characters output is used, of %d, %f, and %e. That is, if the value is an integer, it is displayed without a floating point sign. If it is a very large or very small number it is displayed using exponential format, and if it is intermediate in value, it is displayed using the %f format.

There is an additional command on the Data command line. It is the **W***rite* command, and with it you can write a value into the Aptec mass memory while VOYEUR is running. This is handy when you want to be able to "talk" to one of your processes. The **W***rite* command will ask you for a value and will also specify the type is expected. It is perfectly okay to enter integer data when floating point format is expected, since the decimal point is automatically appended.

Perhaps the most important command in this line is the **O***ffset* command. This is how you tell VOYEUR where to find the number to be displayed. In essence, it is the distance in bytes from the beginning of the system file specified when the containing JOB was created. Numbers associated with Data DETAILS are to be stored as four byte floating point numbers, so the second Data element in a file would have an offset of 4, the third 8 and so on. If you make a mistake in your offset, you will get unpredictable results.

Border

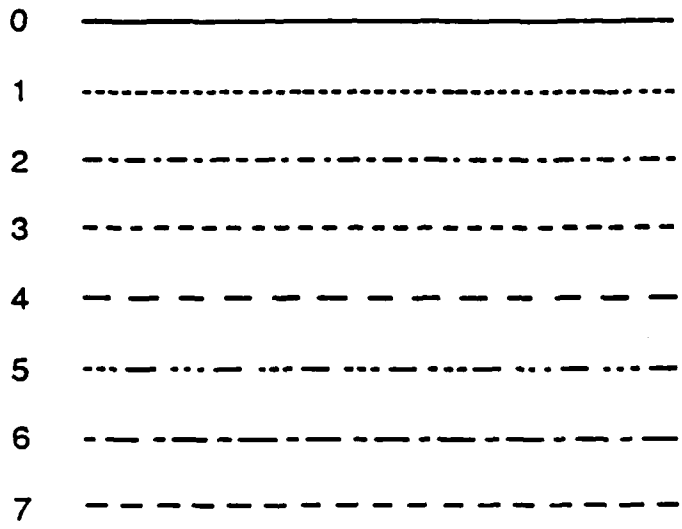
The third type of Detail that you will encounter in VOYEUR is the Border Detail. It's use is rather self explanatory, it generates a border around a picture. It is invoked with this sequence

E*dit* **G***enerate* **B***order*

and it gives you a command line that looks like this:

I*ndex* **S***tyle* **E***nable* **D***isable*

The sub-commands associated with the Border Detail control the style, color, and existence of borders. **I***ndex* works the same way here as it has in other places - it selects a color for the border. The **S***tyle* command is used to define the type of line used to create the border. The different styles of borders and their number codes are:



Finally, one can either **E***nable* or **D***isable* a border. **E***nable* turns a border on, and **D***isable* turns it off. When you enable or disable borders, you specify which of the four sides you are changing. They are associated with the sub-commands **T***op*, **B***ottom*, **L***eft*, and **R***ight*.

AXIS

The fourth type of detail is the Axis Detail. It is used to define axis or grids for a picture. Like the other DETAILS, it is invoked with the sequence:

E*dit* **G***enerate* **A***xis*

and it will give you a new command line that looks like this:

Label **M***arks* **A***xis-xy-loc* **S***tyle* **R***otation* **I***ndex* **K***ind* **T***otal length*

The **S***tyle* command is the same as in the **B***order* command, and the **I***ndex* has the same effect as in the previous detail, that is, it changes the color. You specify the length of the Axis with the **T***otal-length* command. As far as an Axis is concerned, the size of the screen is 4096 X 3256, so an Axis that is half as wide as the screen would have a length of 2048.

The **A***xis-xy-loc* command is used to move the axis. Just position your axis with the thumb wheels and touch any key. As usual, the Bump button will abort the move if you happen to change your mind.

To tell VOYEUR that you want a different type of axis, use the **K***inds* command. You will be prompted by VOYEUR to enter a code number corresponding to the type of axis you want. At this time, the only supported type is linear.

Axis are marked off in discrete sections with something called a tick mark. You can specify the characteristics of the tick marks on your axis from within the **M***arks* command. It will give you a new command line that looks like this:

C*ount* **L***ength* **P***er-value*

You change the number of tick marks with the **C***ount* sub-command. Be careful not to ask for too many tick marks or your axis will become quite crowded.

The size of the tick marks is specified with the **L***ength* sub-command. Here you give two values - one for the length of the major tick marks and one for the minor. Lengths can be either positive or negative. A negative length causes a tick mark to be drawn on the "other side" of the axis. Very long lengths can be specified to draw a grid across the display.

The **P***er-Value* command is used to tell VOYEUR the number of minor tick marks between major tick marks. You should not use a negative number here as it could cause problems. Otherwise, just about

anything else is okay. Normally the count is a multiple of this value, since major tick marks are usually desired on each end of the axis.

Major tick marks are also where the labels are placed. So the **C***ount* and **P***er-value* commands also determine where the labels occur, and how many of them.

The **L***abel* command is used to determine the format and layout of the major tick mark labels. Like the **M***arks* command, **L***abel* will give you a new command line. It should look something like this:

***F**ormat* ***C**har size* ***D**istance* ***R**otation* ***S**lant* ***I**ndex* ***V**alues*

The ***C**har-size*, ***R**otation*, ***S**lant* and ***I**ndex* commands work in just the same way as all the other commands by the same name. The only unfamiliar commands here are the ***F**ormat* and ***V**alues* commands.

The Values command is used to determine the starting and ending values for the axis After entering it, VOYEUR will prompt you for the two values. They should be separated by either a comma or a space.

Closely related to the ***V**alues* command is the ***F**ormat* command. ***F**ormat* works the same way that the Text command works for the Data Detail. After entering this command, VOYEUR will prompt you for a C style format line to place at each of the major tick marks. The values for the numeric fields are calculated automatically from the minimum and maximum specified with the ***V**alues* command. For example, suppose that you had an axis that represented time in whole seconds. A typical label format for such an axis would be something like this:

'Time (Seconds) = %d'

Normally, the entire format string is '%g', which results in a single number that is the smallest possible representation for the label value.

Finally, VOYEUR needs to know how far from the axis it should put your label. You can specify this with the ***D**istance* command.

VOYEUR will prompt you for a new distance which you should enter as an integer within a reasonable range, such as -50 to +50. A negative distance will place the label on the 'other' side of the axis as opposed to a positive distance.

Changing Existing DETAILS

The same commands apply to changing DETAILS as well. The only difference is that **G***enerate* creates new ones and **M***odify* alters existing ones. When you want to modify a Detail, you use the sequence (as always from the top command line)

E*dit* **M***odify* **D***etail*

VOYEUR will then ask you to either use the thumb wheels to place the pick cursor over the Detail you wish to modify and strike a key or to press the Bump button and enter the Detail's number. After either method, you will be given the same command line that you had when you created the selected detail. The commands on that line will do the same things as they did before. Note that the detail number is used for very crowded regions where the pick cursor doesn't have the resolution to pick a specific detail.. The numbers are displayed by touching the List Picture or List Job function keys.

The detail will start to blink when selected to indicate which detail was actually picked. To make it stop blinking, simply touch the L*ocation* command, and then press BUMP.

Copying a Detail

You can copy DETAILS just as easily as you can copy Jobs and Pictures. The command sequence is even similar:

E*dit* **C***opy* **D***etail*

VOYEUR will now ask you to select a Detail in the same manner as you did to select a Detail to modify. It then makes a copy of that Detail. Note that the detail is overlaid on top of the copied detail, so you must pick it and move it to another location before being able to see it.

Deleting old DETAIL's

When you decide that you really didn't want that DETAIL, you can delete it from the memory of the machine. To do this, enter the following command sequence – from the root of the command tree.

Edit Remove Detail

VOYEUR will now show you the screen and all the DETAILS that you have defined. You should use the thumb wheels to position the cursor over the DETAIL that you wish to remove and then press any key. You can use the BUMP button to select the Detail by number. REMEMBER! once a DETAIL is deleted, it cannot be easily resurrected!

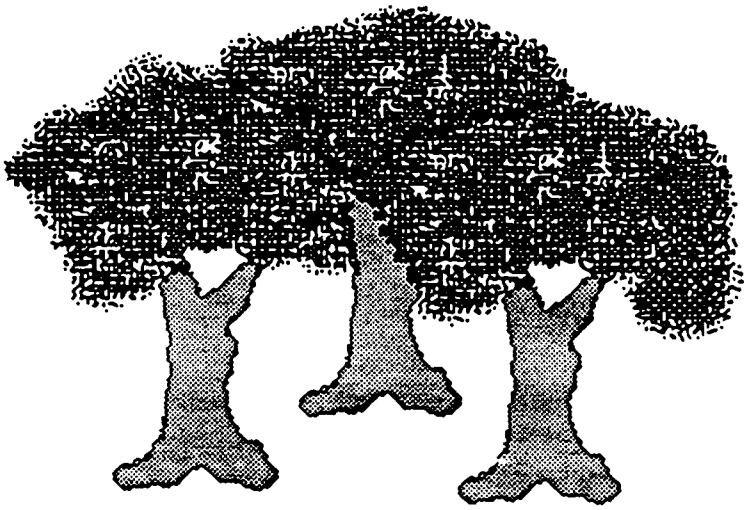
Suggestions

The easiest way to learn about VOYEUR, and particularly about axis detail is to sit down at the 4115 and generate some. For the Axis on the left hand side of the picture use a label slant of 270 degrees, and an axis rotation of 90 degrees. Experiment using tick mark lengths equal to the height or width of the picture, giving grid marks. Also try various combinations of count and per-value amounts.

In Data details, the most common format used is %g. For example, if the format is "STEP NUMBER = %g", the step number will be picked up in standard DEC floating point format from mass memory and displayed as an integer if there is no fractional part. Since it is usually much easier to generate floating point values in array processors than integers, this is preferred.

The TV Guide

Earlier, we saw that the JOB, PICTURE and DETAIL formed what could be thought of as a tree. Well, there are a few more elements to the picture. The first is called the TV Guide. Each time that you generate or remove a JOB, PICTURE or DETAIL, that information is placed in or removed from the internal TV Guide. Think of the TV Guide as a script that controls your session with VOYEUR. A TV Guide can contain several different JOBs, so if you want a pictorial representation, try thinking of a grove of trees:



Saving a TV Guide

Generally, there are so many items in the TV Guide that to redefine them each time you wanted to use VOYEUR would be time consuming. VOYEUR gives you the ability to store your TV Guide in a standard text file so that you can read the information in again at a later time. To save your TV Guide you would enter this command sequence from the root of the command tree:

Write TV Guide

At this point, VOYEUR will ask you for the name of the file that is to contain the TV Guide. You should give VOYEUR a standard file name that includes any necessary directory and device information.

Reading A TV Guide

The TV Guide file wouldn't do much good if it could only be written. That's why there is another command sequence to read a TV Guide. From the root of the command tree, enter this sequence:

Read TV Guide

Now, VOYEUR will ask you if you want to remove the current internal TV Guide from the memory. Most of the time you will want to say yes. There may be times, however when you wish to add two previously separate TV Guides together. In this case, say no. Next, VOYEUR will ask for the name of the disk file containing the TV Guide to be read. You should enter a standard file name that includes any needed device or directory information.

Changing the TV Guide

There are two ways to modify an existing TV Guide. The first is to use the commands of VOYEUR to change the internal TV Guide and then write it out to a file. This is perhaps the easiest, but if you prefer, you can use one of the system editors to modify the TV Guide disk file. You should be careful to maintaining the syntax of the TV Guide or it will give errors when you try and read it, or display very strange things on the screen.

The Color Map

The other supporting element that we will discuss here is the Color Map. The Color Map holds all the color definitions for you session with VOYEUR. Like the TV Guide, the Color Map has an internal (in memory) and a disk representation. When you first start, VOYEUR gives you a default internal Color Map, but you may wish to edit some of the colors to more closely suit your needs.

All the commands to alter colors in VOYEUR can be found under the main command called **C***olor.* After you enter this command, you will see the following command line:

D*efault* **M***odify* **S***tretch*

You can use the **D***efault* sub-command to reload the default Color Map if you decide that you've messed up the colors beyond all hope of repair. Remember that you only need to press the first character of any command and that the RETURN key isn't needed.

When you wish to change a color, you use the **M***odify* sub-command. VOYEUR will put up a grid and allow you to move the thumb wheel to select the color to modify. It will then put up a box shaded with that color that displays the values for four things: Index, Hue, Lightness and Saturation. You can alter the last three with the **H***ue*, **L***ightness* and **S***aturation* commands respectively. Each of these three will give you the same new command line:

U*p* **D***own*

The **U***p* command will cause the value being altered to go up and the **D***own* command will cause it to go down. You should experiment to find the colors that suit you best.

Suppose that you wanted your colors to progress in an even spectrum from one color to another. You can accomplish this with the **S***tretch* sub-command. VOYEUR will ask you for the two indices between which you want the stretch to be made. This command is especially useful if you want your picture to sort of "melt" from one level to

another. Usually, you must have the saturation at 100%, and the lightness of both beginning and ending indices at 50% for best results.

Saving and Reading Color Maps

Just as it would be a time consuming to redefine the internal TV Guide each time you ran VOYEUR, it would be a pain to redefine all your colors. That is the reason behind the idea of the Color Map file. The commands to read or write this file are very much similar to their TV Guide counterparts. You can use the sequence **R*ead C*olor *map*** to read in a color map and the sequence **W*rite C*olor** map to save a color map of your own creation. Each sequence will cause VOYEUR to ask you for the *name of a color map file.*

The Attach command allows you to specify a color map name to be added to the next TV guide write without actually reading it in. The color map name is expanded by VOYEUR so that the directory and device is added to the name before loading it into the TV guide. Thus, if the TV guide moves to a different directory, the same Color map file will be referenced.

Getting Output

As we said in the Introduction, there are currently two kinds of permanent output available with VOYEUR. The first is color hardcopy from a Tektronix T4115B compatible color plotter. The second is Dicomed film – such as 35mm slides or viewgraphs. The methods for obtaining each of these are rather simple. But first, let's talk about how to get VOYEUR to display something on the screen.

Screen Output

To control the refreshing of the screen, VOYEUR uses something called a *Display List.* When VOYEUR goes to refresh the screen, it goes down this Display List one item at a time until it is at the end of the list. In order to display a Picture, you must first add it to the Display List, and there are two ways to do this. The first is to put all the Pictures defined in the internal TV Guide into the list. This is accomplished with the command sequence **D***isplay* **A***ll* (this sequence should be entered from the top command line). If you don't want to look at all your pictures, you can put them in one at a time with the sequence **D***isplay* **S***elect.* (this, again is from the top line) VOYEUR will then ask you for the name of the Picture to add. You should enter the full name of the Picture and press RETURN. If you want to remove Pictures from the Display List, you again have two options. The first is a global removal with the command sequence **D***isplay* **R***eset* from the top line. This will remove all Pictures from the Display List. For a one by one removal, you should use the sequence **D***isplay* **U***nselect.* VOYEUR will then ask you for the name of the Picture to remove from the Display List.

Once you have put all the Pictures you want to see in the Display List, you can tell VOYEUR to start displaying them with the command sequence **D***isplay* **M***onitor* (or you can use the Monitor Function Key). VOYEUR will then start refreshing the screen and continue to do so until you either press the Bump button or the Stop Refresh button.

Color Hardcopy

Built into the T4115B is a feature that allows the user to generate a copy of the current screen. This feature is made available to

users of VOYEUR. The procedure is quite simple. First, make sure that the plotter is connected to the T4115B and that it is turned on and on line. Second, press the Stop Refresh button in the upper left-hand corner of the keyboard. You should wait a few moments after this to give VOYEUR time to finish processing the Display List. This way, the screen will not be changed while you are plotting. Next, press the Hardcopy key – you should see the little red light in the key go on, this means that a screen copy is in progress. The Hardcopy button is located on the right-hand side of the keyboard, near the top. Hardcopy takes anywhere from five to as much as ten minutes to complete the screen copy. Afterwards, you can press the Start Refresh button and VOYEUR will proceed with its processing of the Display List.

Dicomed Output

The Dicomed is a high resolution graphic film recorder that lets you create 35mm slides, viewgraphs, and 16mm movies. Built into VOYEUR is a facility to create files containing Dicomed commands. This means that you can get a slide of what you see on the screen or make a movie. Since you can see the movie/slide progress on the screen, you can know instantly if there are any problems – and fix them.

All the commands that pertain to the creation of Dicomed files are accessed through the **D*isplay* Dicomed** top line sequence. This gives you the following command line:

R*un* F*lash* N*one* S*eparate* B*uffer-size*

You use the **R*un*** command to tell VOYEUR that you want a Dicomed frame after every screen refresh while you Monitor. The first time you enter this command, VOYEUR will ask you for the name of the file that is to contain the Dicomed information. The second and any subsequent time you enter this command, VOYEUR will ask you if you wish to continue writing to the previous file name. If you say no, the VOYEUR will ask you for a new name. If you say yes, VOYEUR will ask you if you want to append to that old file or create a new version of that file.

You can use the Flash command to tell VOYEUR that you want a Dicomed frame each time you press the Dicomed Flash function key. This is useful for making slides of selected frames. The first time you enter

this command, VOYEUR will ask you for the name of the file that is to contain the Dicomed slide information. The second and any subsequent time you enter this command, VOYEUR will ask you if you wish to continue writing frames to the previous file name. If you say no, VOYEUR will ask you for a new name. If you say yes, VOYEUR will ask you if you want to append to that old file or create a new version of that file.

To stop generating Dicomed information and close the Dicomed file, you use the **N***one* command. Further Dicomed frames will not be made until you use either the **R***un* or **F***lash* command and the contents of the buffer will be flushed to disk.

Dicomed frames in VOYEUR require a large amount of information. Writing all this to the disk each time would slow you down considerably. To help avoid this problem, VOYEUR buffers all its Dicomed information in mass memory. You can specify the size of this buffer with the **B***uffer-size* command. Theoretically, you could specify a **B***uffer-size* of up to 12 megabytes, but this isn't very practical. For most applications, the default buffer size will be enough.

The final option with Dicomed output is the **S***eparate* option. You use this to tell VOYEUR to separate the colors and generate three frames for each screen – one for the Red, one for Green and one for Blue. You shouldn't need to use this option very often.

MACROS

Macros are used to redefine the meanings of keys. A string, special VOYEUR commands, even other macros can be assigned to any key, so that when the operator touches the defined key, the assigned string is used instead.

Defining a Macro

The following command sequence permits you to define macros:

M*acro* D*efine*

After entering this command the key to assign the macro to will be requested. The key values are listed in the next section. Note that all terminals with a lower number than 4120 cannot assign key values -2 to -178. Defining a key value of -1 undefines all macros, so that is probably not useful either. 4115 terminals will give an error if a negative value is used.

After the key number is entered, a request for the string to assign to the key is made. You generate this string from a combination of special macro symbols and characters representing VOYEUR commands. Hitting RETURN terminates the string.

The special commands are used in VOYEUR to direct it to perform actions not normally available from the command string. All of these symbols can be upper or lower case. They are:

Normal Display Commands

\\	single backslash
\n	terminate a command string request
\b	bump up one level in the command tree
\s	suppress command display and echoing
\q	turn on command display and echoing
\t	bump to top of entire command tree

Monitor Mode Commands

\1	exit monitor mode
\2	Dicomed flash
\4	stop refresh
\5	start refresh
\6	view color map
\7	remove color map (if being displayed)

The Normal Display commands work during graphics editing sessions only. The Monitor mode commands work only while the display is being displayed in monitor mode.

An example and explanation of the use of each backslash command follows:

1. wtDISPLAY.TVG\n

The wt selects the Write TV guide command. The DISPLAY.TVG names the TV guide to write to, and the \n signals the command line processor that the end of the line is reached. Notice here that the \n solves the problem of how to terminate the requests VOYEUR makes inside a macro, since a RETURN will terminate the macro definition. The upper case is for emphasis only, and has no other significance.

2. \twtDISPLAY.TVG\n\t

This corrects a problem in macro example #1 above. The flaw in that macro was that it assumed that anytime a 'W' key was touched, the Write command would be executed. That is only true from the top level command line. This example uses a \t special command to insure that before the 'W' is executed, the top of the tree is found. This makes this command always work, regardless where in the command tree you are located (unless actually executing another command). The terminating \t leaves you at the top of the tree after the command has completed.

3. \s\t wtDISPLAY.TVG\n \q\t

When executing macro commands, the effect is precisely the same as if the keys are entered by hand from the keyboard. As a result, the command line flickers and the help line is displayed for each command. The \s special command will turn off the display and disable all writes to the screen. The \q turns it back on, so that further commands can be entered. Otherwise, the macro here has the same effect as macro example #2. Notice that there are spaces in the macro definition. In general (except after commands requiring string input), tabs and spaces have no effect within macro definitions.

4. \1 \t wtDISPLAY.TVG\n \tDAM

This macro would be used only while the screen is being written in monitor mode. The \1 bumps out of monitor mode, then the wtDISPLAY.TVG\n writes a TV guide, then the DAM commands restart the monitor.

Creating a Macro File

Macro files are disk files which contain macro definitions. They are executed in one of two ways, either with the **Macro Read** command, or by including a MACRO statement in the TV guide (done automatically when writing a TV guide after reading a macro file). Macro files are generally used to assign macros to keys, and to execute a predetermined script of VOYEUR commands automatically when VOYEUR is started, although they can be used to execute any number of VOYEUR commands directly. Macro files can be nested, that is, they can contain other Macro files, up to 16 levels deep.

Macro files are commented by including comment text between /* and */ markers. Text between these two character string is ignored (comments cannot be nested). Similarly, spaces, tabs, and carriage returns are ignored (except inside quoted strings, and inside arguments which are themselves strings). Uncommented macro files are obtuse and indecipherable Please comment them.

All characters enclosed in double quotes (") in a macro file are transmitted exactly. This includes comments, spaces, tabs, and carriage returns. This is useful when assigning macro strings to keys, since

special commands in macro assignment strings conflict with the same commands in the file.

Macro files contain backslash command sequences which have the same effect as within a macro, except that the effect is immediate, rather than being delayed until a key is touched. The following commands have the same effect in a macro file as inside a macro definition:

\n	end of string for commands
\b	bump to next higher level
\t	bump to top of tree
\s	suppress command display
\q	turn on command display

After a macro file is built, to include it in the TV guide just read it in or attach it to the TV Guide using the **M*acro* A*ttach*** command, then write the new TV guide. The macro file will automatically be executed whenever the TV guide is read.

The following program defines two keys. The Control Q key will terminate Voyeur whenever it is used, and the Control X key will write the TV guide file to DISPLAY.TVG and then exit.

```
/*
*           Example Macro command file
*
*/

/*
*  First, suppress all display, so that the screen will not flicker
*  while the macros are being defined.  Also go to the top of the
*  command tree.
*/

/s/t

/*
*  Now define control Q as a macro.  From the table go down the left
*  hand side and find Q.  Then go across horizontally until the
* column marked CTRL is found.  The value is 17.  This is the key
```

```
* value.  Note that the assignment string is quoted, since the
* special commands inside the assignment string are supposed to
* be executed only when the Control Q key is touched.
*/

     \t MR 17\n "\s\t Q"\n

/*
*  Next, define the Control X key.
*/

     \t MR 24\n "\s\t WTdisplay.tvg\n \t Q" \n
/*
*  Finally, turn on command display and get to the top of the
*  command tree.
*/

 \q\t
```

Caution

Macros are a very powerful and hence potentially destructive tool. You should use them carefully. One important thing to remember is to make frequent use of the '\T' to insure that you know from what level in the command tree VOYEUR starts. 'E' at one level could have a different effect from 'E' at another!

Key	Un Shifted	Caps Lock	Shifted	Ctrl	Ctrl Shift
{ [	91	91	123	27	27
! 1	49	49	33	49	33
@ 2	50	50	64	50	0
# 3	51	51	35	51	35
$ 4	52	52	36	52	36
% 5	53	53	37	53	37
^ 6	54	54	94	54	30
& 7	55	55	38	55	38
* 8	56	56	42	56	42
(9	57	57	40	57	40
) 0	48	48	41	48	41
_ -	45	45	95	45	31
+ =	61	61	43	61	43
}]	93	93	125	29	29
RUB OUT	127	127	-34	-35	-36
ESC	27	27	-37	-38	-39
~ `	124	124	126	28	30
Q	113	81	81	17	17
W	119	87	87	23	23
E	101	69	69	5	5
R	114	82	82	18	18
T	116	84	84	20	20
Y	121	89	89	25	25

Key	Un Shifted	Caps Lock	Shifted	Ctrl	Ctrl Shift
U	117	85	85	21	21
I	105	73	73	9	9
O	111	79	79	15	15
P	112	80	80	16	16
`	92	92	96	28	0
\					
BACK SPACE	8	8	-40	-41	-42
LINE FEED	10	10	-43	-44	-45
TAB	9	9	-46	-47	-48
A	97	65	65	1	1
S	115	83	83	19	19
D	100	68	68	4	4
F	102	70	70	6	6
G	103	71	71	7	7
H	104	72	72	8	8
J	106	74	74	10	10
K	107	75	75	11	11
L	108	76	76	12	12
: ;	59	59	58	59	58
" '	39	39	34	39	34
RET	13	13	-49	-50	-51
Z	122	90	90	26	26
X	120	88	88	24	24
C	99	67	67	3	3
V	118	86	86	22	22
B	98	66	66	2	2
N	110	78	78	14	14
M	109	77	77	13	13
< ,	44	44	60	44	60
> .	46	46	62	46	62
? /	47	47	63	47	63
SPACE	32	32	-52	-53	54

Key	Un Shifted	Caps Lock	Shifted	Ctrl	Ctrl Shift
F1	128	128	136	-2	-10
F2	129	129	137	-3	-11
F3	130	130	138	-4	-12
F4	131	131	139	-5	-13
F5	132	132	140	-6	-14
F6	133	133	141	-7	-15
F7	134	134	142	-8	-16
F8	135	135	143	-9	-17
DIALOG	-111	-111	-117	-123	-129
SETUP	-112	-112	-118	-124	-130
LOCAL	-113	-113	-119	-125	-131
COPY	-114	-114	-120	-126	-132
PAGE	-115	-115	-121	-127	-133
BREAK	-116	-116	-122	-128	-134
ZOOM	-18	-18	-22	-26	-30
PAN	-19	-19	-23	-27	-31
NEXT VIEW	-20	-20	-24	-28	-32
VIEW	-21	-21	-25	-29	-33

VOYEUR QUICK REFERENCE

The VOYEUR system employs a menu-driven system of commands. These commands are organized what can be thought of as a tree (see the Command Card for a pictorial view). This section gives a brief description of each of the commands in VOYEUR and is intended to be used by those who are more familiar with VOYEUR. If you are just starting with VOYEUR, you should consider reading the tutorial first. This section uses the convention that the further indented a command is, the further down the tree it is.

Display - Alter the contents of the display list

- ***Select*** - Select a picture to add to the display list
- ***All*** - Select all pictures for display
- ***Unselect*** - Remove a picture from the display list
- ***Reset*** - Empty the display list
- ***Dicomed*** - Alter the settings for Dicomed Slide generation
  - ***Run*** - Enable a new Dicomed frame after every display refresh
  - ***Flash*** - Write a Dicomed frame when the DICOMED FLASH key is pressed
  - ***None*** - Disable all Dicomed frames and close any current file
  - ***Separate*** - Separate or combine the Red Green and Blue
  - ***Buffer-Size*** - Change the size of the mass memory buffers used for Dicomed
- ***Monitor*** - Start the VOYEUR monitor

Edit - Edit the internal TV-Guide structure

- ***Copy*** - Copy a JOB, PICTURE, or DETAIL
  - ***Job*** - Copy a JOB structure
  - ***Picture*** - Copy a PICTURE structure
  - ***Detail*** - Copy a DETAIL structure
- ***Modify*** -Modify the current internal TV-Guide
  - ***Job*** - Modify a JOB specification
  - ***Picture*** - Modify internal PICTURE specification
    See commands under the Generate section
  - ***Detail*** - Modify internal DETAIL specification
    See commands under the Generate section
  - ***Set-re-draw*** - Set one or more RE-DRAW flags
  - ***Clear-re-draw*** - Clear one or more RE-DRAW flags
  - ***Index*** - Modify the index for all detail loaded

Generate - Generate a new component for the internal TV-Guide
- Job - Generate a new JOB structure
- Pixels - Generate a new PIXELS picture specification
  - Name - Modify mass memory file name where PIXEL data is located
  - Offset - Modify offset from beginning of PIXEL data file
  - Location - Modify PIXEL picture location
  - Width - Modify PIXEL picture width
  - Height - Modify PIXEL picture height
  - File-size - Modify PIXEL file size
  - Pixel - Modify pixel dimensions or axis
    - Width - Modify pixel width
    - Height - Modify pixel height
    - Axis - Modify pixel major axis
    - Mode - Modify pixel ALU writing mode
    - Type - Modify pixel write type
  - Refresh - Modify the picture refresh parameters
    - Continuous - Refresh picture continuously
    - Time - Refresh picture after specified time delay
    - Synchronize - Refresh picture on a mass memory flag
    - Lockstep - Refresh picture, synchronizing with a STEP value
    - Value - Refresh picture based on a specific value in memory
- Graph - Generate a new GRAPH picture specification
  - Name - Modify mass memory file name where GRAPH data is located
  - Offset - Modify offset from beginning of file of GRAPH data
  - Location - Modify GRAPH picture location
  - File-size - Modify GRAPH file size
  - Refresh - Modify refresh parameters
    - Continuous - Refresh picture continuously
    - Time - Refresh picture after specified time delay
    - Synchronize - Refresh picture on a mass memory flag
    - Lockstep - Refresh picture, synchronizing with a STEP value
    - Value - Refresh picture based on a specific value in memory
  - Size - Modify picture size
  - Window - Modify window parameters
    - Border - Change border index or visibility
      - Index - Change the border index
      - Visibility - Change the border visibility
    - Wipe - Change the wipe index
    - Origin - Change the window origin
    - Extent - Change the window extent
- Segment - Generate a new SEGMENT picture specification
  - Name - Modify mass memory file name where SEGMENT data is located
  - Offset - Modify offset from beginning of file of SEGMENT data
  - Location - Modify SEGMENT picture location
  - File-size - Modify SEGMENT file size
  - Refresh - Modify refresh parameters
    - Continuous - Refresh picture continuously
    - Renew - Renew viewpoint before each display
    - Time - Refresh picture after specified time delay
    - Synchronize - Refresh picture on a mass memory flag

- *Lockstep* - Refresh picture, synchronizing with a STEP value
- *Value* - Refresh picture based on a specific value in memory
- *Size* - Modify picture size
- *Window* - Modify window parameters
  - *Border* - Change border index or visibility
    - *Index* - Change the border index
    - *Visibility* - Change the border visibility
  - *Wipe* - Change the wipe index
  - *Origin* - Change the window origin
  - *Extent* - Change the window extent

Vector - Generate a new VECTOR picture specification

- *Name* - Modify mass memory file name where VECTOR data is located
- *Offset* - Modify offset from beginning of file of VECTORdata
- *Location* - Modify VECTOR picture location
- *File-size* - Modify VECTOR file size
- *Refresh* - Modify refresh parameters
  - *Continuous* - Refresh picture continuously
  - *Renew* - Renew viewport before each display
  - *Time* - Refresh picture after specified time delay
  - *Synchronize* - Refresh picture on a mass memory flag
  - *Lockstep* - Refresh picture, synchronizing with a STEP value
  - *Value* - Refresh picture based on a specific value in memory
- *Size* - Modify picture size
- *Window* - Modify window parameters
  - *Border* - Change border index or visibility
    - *Index* - Change the border index
    - *Visibility* - Change the border visibility
  - *Wipe* - Change the wipe index
  - *Origin* - Change the window origin
  - *Extent* - Change the window extent

Axis - Generate a new AXIS detail structure

- *Label* - Modify parameters associated with attached label strings
  - Format - Change text and format of label string
  - Char size - Change size of the label string
  - Distance - Change distance of labels from the Axis
  - Rotation - Change label rotation
  - Slant - Change label slant
  - Index - Change the color of the label
  - Values - Change the values of the end points
- *Marks* - Modify parameters associated with the tick marks
  - *Count* - Modify the number of tick marks used
  - *Length* - Modify the length of the tick marks
  - *Per-value* - Modify the number of tick marks per value
- *Axis-xy-loc* - Modify AXIS location
- *Style* - Modify the line style used to draw the AXIS
- *Rotation* - Change overall rotation of the entire AXIS
- *Index* - Modify index used for lines in AXIS
- *Kind* - Modify the kind of AXIS used
- *Total-length* - Modify the total length of AXIS

Label - Generate a new LABEL detail specification

- *Character-size* - Modify the size of text characters

- Location - Modify the LABEL location
- Slant - Modify the LABEL slant
- Rotation - Modify the LABEL rotation
- Index - Modify the LABEL index
- Bold - Modify the number of times the string is written
- Text - Modify the text associated with the LABEL

Data - Generate a new DATA detail specification
- Character-size - Modify the size of text characters
- Location - Modify the DATA location
- Slant - Modify the DATA slant
- Rotation - Modify the DATA rotation
- Index - Modify the DATA index
- Text - Modify the text associated with the DATA
- Bold - Modify the number of times the string is written
- Offset - Modify the file offset for the data variable
- Write - Write a new value into mass memory at the designated offset

Border - Generate a new BORDER detail specification
- Index - Modify the index used for the BORDER
- Style - Modify the line style used for the BORDER
- Disable - Disable BORDER
  - Top - Disable drawing BORDER on top
  - Bottom - Disable drawing BORDER on bottom
  - Right - Disable drawing BORDER on right
  - Left - Disable drawing BORDER on left
- Enable - Enable BORDER
  - Top - Enable drawing BORDER on top
  - Bottom - Enable drawing BORDER on bottom
  - Right - Enable drawing BORDER on right
  - Left - Enable drawing BORDER on left

Remove - Remove a JOB, PICTURE, or DETAIL from the internal TV-Guide
- All - Clear the entire TV-Guide from memory
- Job - Remove a JOB from the internal TV-Guide
- Picture - Remove a PICTURE from the internal TV-Guide
- Detail - Remove a DETAIL from the internal TV-Guide

Defaults - Change the defaults associated with various things
- Picture - Change the picture fill index
- Aperture - Change the pick aperture
- Line - Change the default line index
- Bold - Change the default bold for DETAILS
- Dialog - Change one or more of the dialog area indexes
  - Alternate - Change alternate index (for commands)
  - Background - Change the dialog background index
  - Character - Change the character index used for help line

Color - Modify the internal color map

Attach - Attach a Color map name to the TV Guide

Default - Load the default color map

Modify - Modify the internal color map
- Hue - Alter the Hue
  - Up - Increase the Hue
  - Down - Decrease the Hue
- Lightness - Alter the Lightness
  - Up - Increase the Lightness
  - Down - Decrease the Lightness
- Saturation - Alter the Saturation
  - Up - Increase the Saturation
  - Down - Decrease the Saturation

Stretch - Stretch and even spectrum between two colors

Read - Read a VOYEUR file

- **Color-Map** - Read a new color map
- **TV Guide** - Read a new TV-Guide

Write - Write a VOYEUR file

- **Color-Map** - Write the current color map
- **TV Guide** - Write the current TV-Guide

Select - Select a new JOB or PICTURE to be worked on

- **Job** - Select a new JOB to work on
- **Picture** - Select a new PICTURE to work on

Open - Reopen all mass memory files

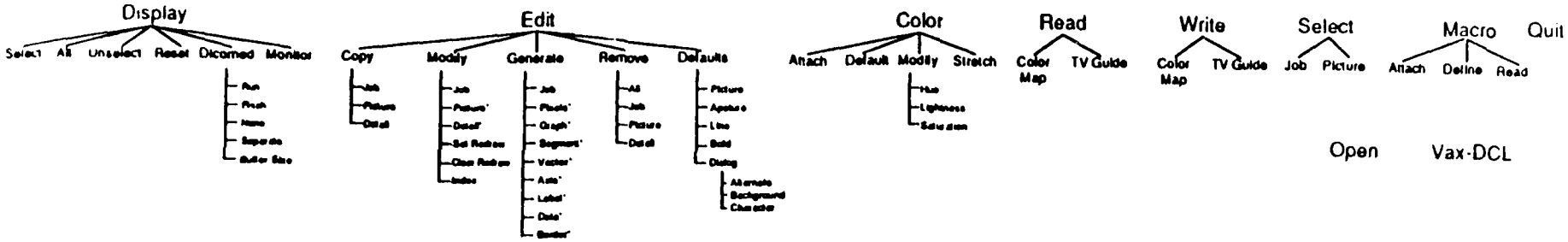
Macro - Manipulate VOYEUR macros
- **Define** - Define a macro
- **Read** - Read a macro file

Quit - Exit VOYEUR

Vax-DCL - Spawn a VAX DCL sub process (press LO to return)

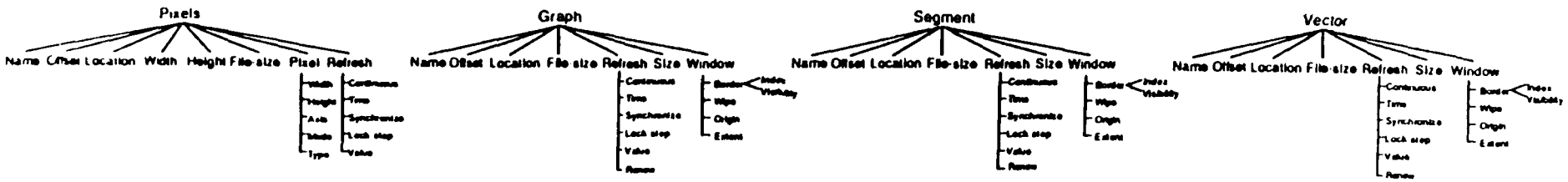
Voyeur Commands - Tree Structure

Top Line

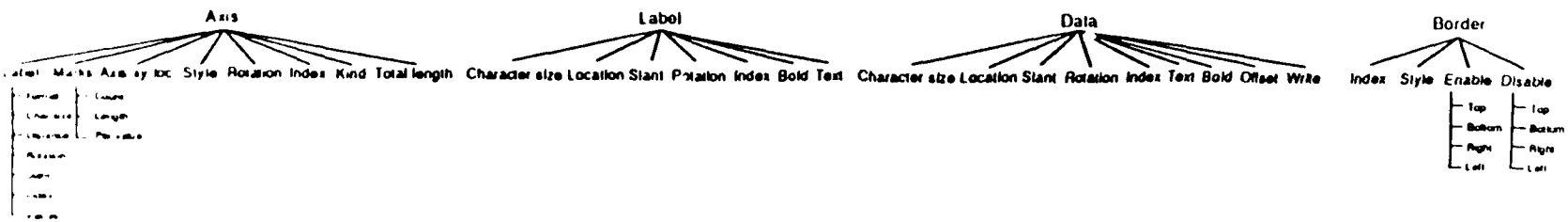


* Edit Generate & Edit Modify
Sub-Commands

Picture



Detail



APPENDIX D.

Direct Numerical Simulation of Axisymmetric Jets (AIAA-86-0039)

DIRECT NUMERICAL SIMULATION OF AXISYMMETRIC JETS

F.F.Grinstein,*, E.S.Oran,† and J.P.Boris,‡
Laboratory for Computational Physics
Naval Research Laboratory
Washington, D.C. 20375

Abstract

We present results from numerical simulations of the evolution of the Kelvin-Helmholtz instability for an unforced, subsonic, compressible, axisymmetric, spatially-evolving shear layer. In addition, we study the effect of small, random pressure fluctuations at the nozzle orifice on the growth of the mixing layer. These fluctuations model inflow perturbations in experimental flows arising from turbulence and boundary layers in the nozzle. The finite-difference numerical model used to perform the simulations solves the two-dimensional time-dependent conservation equations for an ideal fluid using the Flux-Corrected Transport algorithm and timestep-splitting techniques. No subgrid turbulence model has been included. In the absence of perturbations, the calculations indicate that the large scale development of the unforced jet shear layer has an underlying degree of organization. This is the result of a feedback mechanism in which the shear layer ahead of the nozzle edge is modulated by the far field induced by the mergings downstream, near the end of the potential core of the jet. The studies with random high frequency perturbations on the inflow velocity show that they effectively tend to break the temporal correlations between the structures.

Introduction

Experimental investigations in the last decade have shown that large spanwise coherent structures dominate the entrainment and mixing processes in shear layers [1]. Recently, it has become possible to study these structures by direct numerical simulation of their large scale features. These simulations are an important alternative and supplementary tool in the basic research of the properties of fluid flow transitioning to turbulence. Since a simulation can calculate values for all the primary flow field properties as a function of time, statistical information can be obtained about the system through spatial and temporal averages. In addition, parameters of the flow can be easily varied. While the experimental conditions may not be fully controllable in the laboratory, the simulation conditions are.

Numerical studies of coherent structures have used spectral [2], vortex dynamics [3], and finite-difference [4–6] techniques. Numerical studies of the evolution of flows similar to those seen in the laboratory experiments have considered both two-dimensional planar and axisymmetric shear layers. Previous finite-difference calculations have modeled either temporally-developing mixing layers [4], where it is assumed that the relevant vortex dynamics takes place in relatively compact space regions, or spatially developing layers [5–7], which represent the more realistic situations occurring in the laboratory.

In previous work we have performed finite-difference, compressible, spatially-developing simulations of planar shear flows, with the objective of investigating asymmetries in mixing [6] and the basic mechanisms involved in the reinitiation of the instabilities [7]. Here we describe finite-difference calculations of the evolution of the Kelvin-Helmholtz instability for a spatially-evolving compressible axisymmetric jet emerging into a quiescent background. The instabilities sustain themselves through a feedback mechanism in the flow. The evolution and merging of the downstream structures affect the inflowing material upstream, thus triggering the growth and shedding of new vortices [7]. In addition, we study the effect of small, random pressure fluctuations at the nozzle orifice on the growth of the mixing layer. These fluctuations model inflow perturbations in experimental flows arising from turbulence and boundary layers in the nozzle.

The Numerical Model

The numerical model used to perform the simulations solves the two-dimensional time-dependent conservation equations for mass, momentum and energy for an ideal gas

$$\frac{\partial \rho}{\partial t} = -\nabla \cdot \rho \mathbf{V}, \tag{1}$$

$$\frac{\partial (\rho \mathbf{V})}{\partial t} = -\nabla \cdot \rho \mathbf{V}\mathbf{V} - \nabla P, \tag{2}$$

$$\frac{\partial \epsilon}{\partial t} = -\nabla \cdot \epsilon \mathbf{V} - \nabla \cdot P\mathbf{V}, \tag{3}$$

where $\epsilon = P/(\gamma - 1) + (1/2)\rho V^2$, is the internal energy, and $\mathbf{V}$, P, ρ, and γ, are the velocity, pressure, mass density, and the ratio of specific heats. The equations are solved using the Flux-Corrected Transport (FCT) algorithm [8] and timestep-splitting techniques. FCT is an explicit, fourth-order, finite-difference algorithm, which ensures that all conserved quantities remain monotonic and positive. FCT modifies the linear properties of a high order algorithm by adding diffusion during convective transport to prevent dispersive ripples from arising. The added diffusion is subtracted out appropriately where not needed in an anti-diffusion phase of the integration cycle to maintain high order accuracy. With this approach, no artificial viscosity is required to stabilize the algorithm.

The model uses inflow and outflow boundary conditions which have been developed and tested for these types of multidimensional FCT calculations [6,9]. The conditions ensure the proper behavior of the fluid near the boundaries. The inflow boundary conditions specify the inflow density and velocity, and use a zero slope condition on the pressure at the inflow boundary to define the energy at the guard cells:

* Research Physicist, Berkeley Research Associates Inc., Springfield, VA
† Senior Scientist, LCP, Member AIAA
‡ Chief Scientist, LCP

$$\rho_g = \rho_{inflow}, \tag{4a}$$

$$v_g = v_{inflow}, \tag{4b}$$

$$P_g = P_1, \tag{4c}$$

where P_1 is the pressure at the first (inflow) cell. This allows pressure differences between the jet and the surrounding fluid to generate transverse flows. In addition, a short inflow plenum is modelled by including a portion of the nozzle within the computational domain. The outflow boundary conditions define the density and velocity at the guard cells by means of extrapolations from the last two cells:

$$\rho_g = \rho_n, \tag{5a}$$

$$v_g = 2v_n - v_{n-1}. \tag{5b}$$

The guard cell pressure, in turn, is defined by interpolating between the boundary pressure value and ambient pressure (assumed at infinite distance from the trailing edge of the nozzle):

$$P_g = P_n + \frac{(Y_g - Y_n)}{(Y_g - Y_j)}(P_{amb} - P_n), \tag{5c}$$

where Y is either the radial or axial coordinate, and the subscript j refers to the trailing edge of the nozzle. In this way, the outflow boundary conditions impose a slow relaxation of the pressure towards the known ambient value. These boundary conditions allow feedback to occur between the downstream vortices and the inflowing material. This physically realistic approach avoids the need to constantly drive the instability, allowing for it to evolve naturally in the calculation. No subgrid turbulence model has been included. The simulations are expected to be adequate in describing large scale features of gas phase flow for large Reynolds numbers.

The computational grid was set up initially and held fixed in time. The timesteps were chosen to satisfy the Courant condition. The basic finite-difference grid used 120 cells in the cross-stream (radial) direction and 220 in the streamwise (axial) direction, with the mesh spacings varying in the ranges $0.05 \leq \Delta Z \leq 0.52$ cm and $0.02 \leq \Delta R \leq 0.067$ cm. Figure 1 shows a schematic diagram of the grid. The cells are closely spaced in the radial (R) direction across the shear layer, where the large structures form, and they become farther apart as the distance from the shear layer increases for $R > R_o$. The cell separations in the streamwise direction (Z) also increase in size as we move downstream and upstream from the trailing edge of the nozzle, located at $R = R_o = 0.9$ cm, $Z = Z_o \approx 1.67$ cm. This takes advantage of the fact that the structures merge and grow downstream, so that fewer cells are necessary to keep the same relative resolution as obtained near the nozzle. Convergence of the results was verified by checking their consistency with results obtained on grids with 440×120 and 220×240 computational cells, which doubled the resolution in the axial and radial directions, respectively, relative to the 220×120 grid.

The Unforced Axisymmetric Jet

The system studied is a high speed jet containing a mixture of molecular hydrogen and nitrogen emerging into a quiescent background mixture of molecular oxygen and nitrogen, with a jet to background density ratio of 0.67. The system was initialized with a step function axial velocity profile at a uniform temperature (298 K) and pressure (1 atm). The

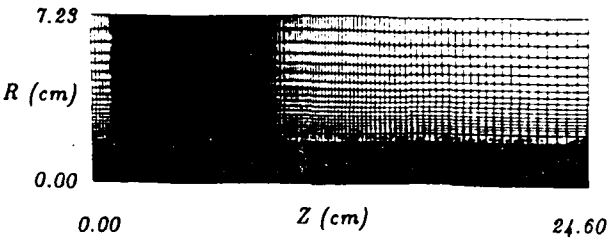


Fig. 1: Schematic diagram of the 220×120 *computational grid.*

jet was subsonic, with Mach number 0.57, and corresponding velocity 2.0×10^4 cm/s. Figure 2 is a schematic diagram of the flow configuration. The perturbation, which initiated the instabilities and occurred only at the very beginning of the calculation, is a very small cross-stream pressure gradient generating vorticity at the shear layer just ahead of the nozzle edge. This disturbance moves downstream as the integration proceeds, generating the transverse flows which trigger the Kelvin-Helmholtz instability. Previously, initial sinusoidal perturbations along the shear interface were considered [9]. The current approach to initiating the instability was used in the simulation of planar shear flows [6,7], and is closely analogous to using a delta function perturbation at the center in an idealized periodic simulation involving two equal and opposite streams.

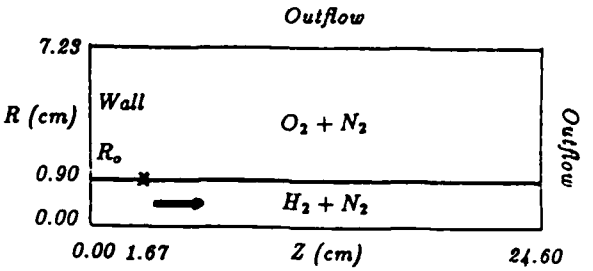


Fig. 2: Flow configuration for the axisymmetric jet simulation. The 'x' indicates the location of the trailing edge of the nozzle.

Typical features of the flow at the early stages are shown in the sequence of isovorticity contours in Fig. 3. Vortex rings develop because of the non-linear growth of the instability. This occurs at an essentially fixed distance from the nozzle edge, somewhat less than one diameter $D = 2R_o$, at $z = (Z - Z_o)/D \approx 0.4$. The newly formed structures move along the interface, interact with each other and thereby spread the vorticity until the center, potential core region disappears, at approximately $z = 3$. The structures are displaced vertically by a low-frequency modulation of the shear layer. When appropriately phased with the structures, the modulation tends to favor the merging of three structures in half of the modulation cycle, where they will coalesce into larger structures. Since the jet is unforced, the low-frequency modulation can only be due to the pressure field induced by the larger structures downstream, as they pass near the end of the potential core of the jet. As the effect of the downstream events on the inflowing fluid becomes important, there is an interaction between the basic instability mechanism at the shear layer and the feedback mechanism. The flow pattern becomes subsequently different from that

at the initial stages in the flow development, which is dominated by the shear layer instability. A striking feature, on the last panel of Fig. 3, is the spatial coherence between the structures. This is characteristic of the feedback phenomena between the downstream events and the inflowing fluid in jets [10]. The distance between the second and the third merging locations increases, relative to the distance between the second and the first mergings, by a factor of the order of 2 - 3, in good agreement with the results of jet experiments with low-level acoustical excitation [12].

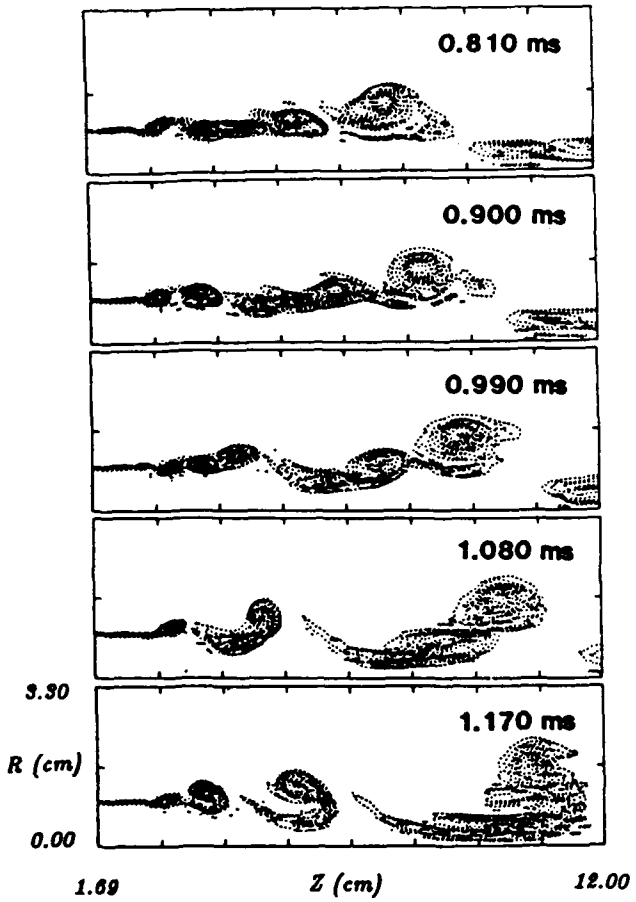


Fig. 3: Initial stages in the flow development.

Frames at much later times in the development of the flow can be seen in Fig. 4. A noticeable feature of this sequence of frames is a regular repeating spatial pattern in the evolution of the flow. We have performed a spectral analysis of the axial (streamwise) velocity fluctuations, based on the results of the calculations for these later times. In the sequence of panels in Fig. 5, we show the results of the analysis at various relevant axial locations on the center of the shear layer ($R = R_o$), and in terms of the normalized (Strouhal) frequency $St = 2\pi f \theta_o / v_o$, where θ_o and v_o are the initial momentum thickness of the shear layer and the jet velocity, respectively. The latter thickness was taken to be effectively defined by $\theta_o = \Delta R = 0.02$ cm, where ΔR is the size of a radial cell in the region of the shear layer (within which the step of the initial velocity profile is defined).

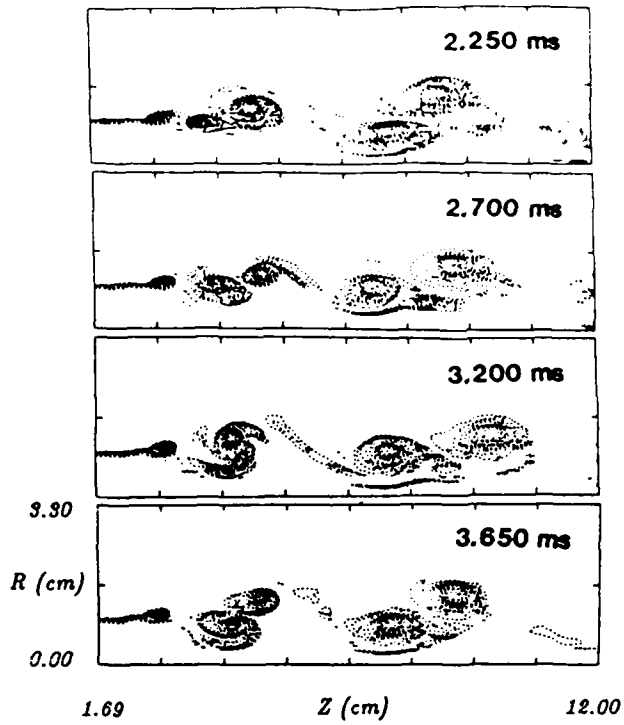


Fig. 4: Regularity and spatial coherence of the flow development for the unforced jet.

Figure 5a, at $z = 0.4$, corresponding to a location near the nozzle exit, shows peaks located at the frequencies $St_o \approx 0.100$, $St_1 \approx 0.050$, $St_2 \approx 0.025$, $St_+ \approx 0.125$, and $St_- \approx 0.075$. Here, St_o can be associated with the shear layer instability frequency, in good quantitative agreement with the predictions of linear inviscid instability theory [13]. In addition, the modulation of the initial shear layer with $St_1 \approx St_o/2$ and $St_2 \approx St_o/4$ is associated with the first and second subharmonic, while $St_\pm \approx (St_o \pm St_2)$ result from the non-linear interaction between the shear layer instability and the feedback process. Moreover, we note that the low frequency peak is located at non-dimensional frequency $St_{jet} = fD/v_o \approx 0.36$. This is within the range $0.2 \leq St_{jet} \leq 0.5$, where the frequency of the preferred jet mode (characteristic of the largest scales on an unforced subsonic jet) is known to lie [12]. As we move downstream, the amplitudes for the high frequencies tend to diminish, as can be expected. In particular, Figs. 5b,c indicate locations of the beginning and concluding stages, respectively, of a first merging (associated with St_1). Similarly, Figs. 5d and 5e correspond to locations where second (associated with St_2) and third (associated with $St_3 \approx St_o/8$) mergings take place.

The detailed flow visualization through the sequence of panels in Fig. 6 allows to determine the approximate pairing locations, as defined by the locations where the vertical alignment of the vortices takes place. In this way the locations of the second and third pairings can be estimated from panels c) and d), respectively, at $Z \approx 4.8$ cm, and $Z \approx 9.7$ cm. The beginning of the first pairing can be also observed in panel d), at $Z \approx 3.3$ cm, where the vortices are seen in the process of rolling around each other. There is

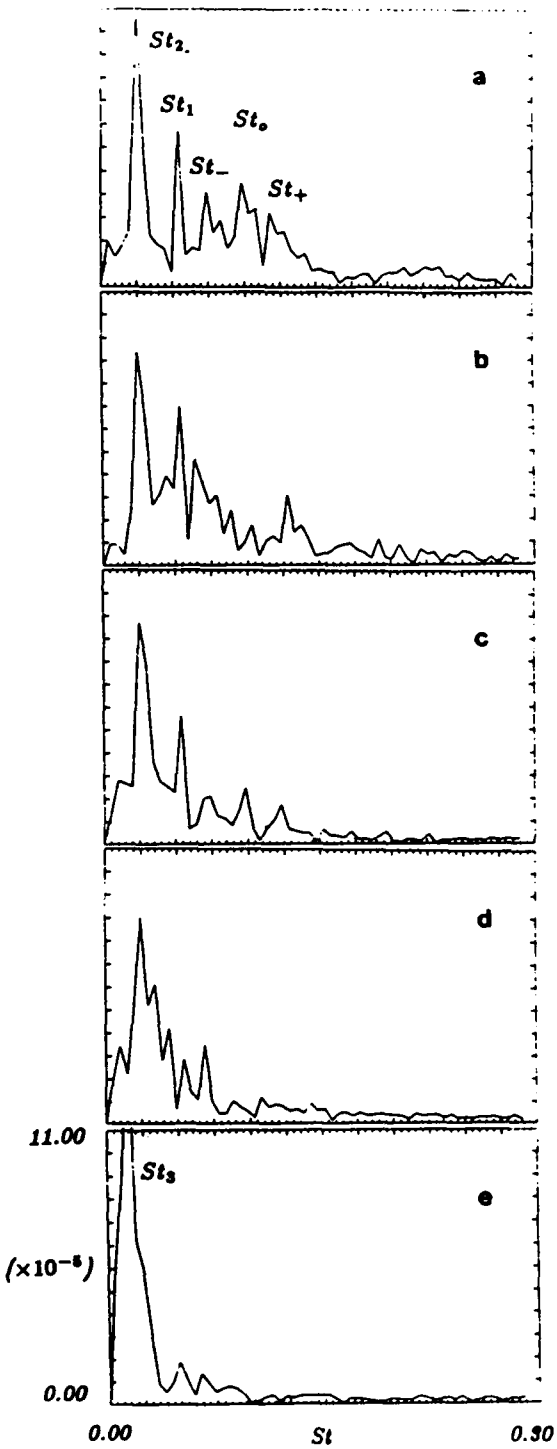


Fig. 5: Spectral amplitude at $R = D/2$, and various axial locations, as a function of Strouhal frequency St. a) $Z = 2.39$ cm ($z = 0.40$); b) $Z = 2.94$ cm ($z = 0.71$); c) $Z = 3.54$ cm ($z = 1.04$); d) $Z = 5.24$ cm ($z = 1.98$); e) $Z = 9.44$ cm ($z = 4.32$).

a close agreement between these merging locations obtained by direct flow visualization, and the results of the spectral analysis above. Note, in particular, that the third merging actually occurs downstream of the end of the potential core of the jet. Because of this, the effect of this merging on the shear layer just ahead of the trailing edge of the nozzle is negligible, as indicated by the spectral distribution in Fig. 5a. This is evidence of the role of the jet diameter, and hence of the location of the end of the potential core, in the selection of the dominant low frequency mode of the jet.

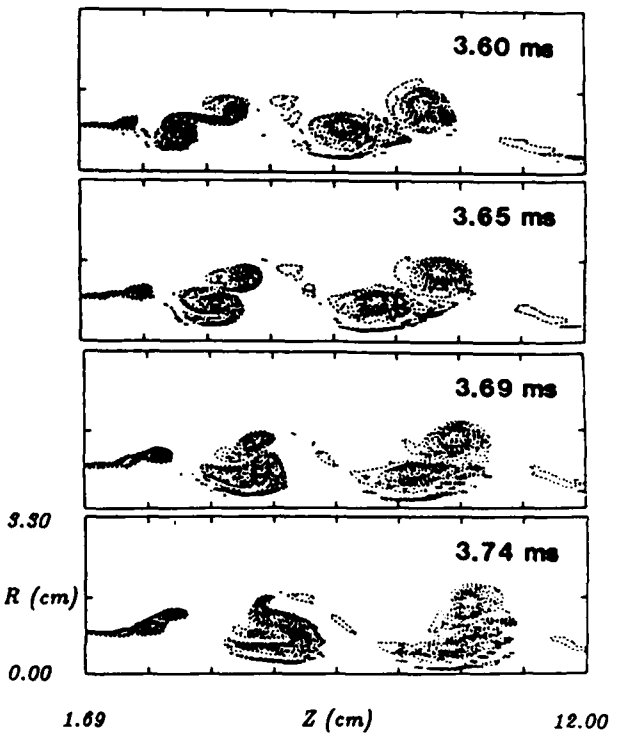


Fig. 6: Flow visualization showing detailed vortex merginge.

Effect of Random Inflow Perturbations

We have also simulated the effects of high frequency random perturbations in the inflowing jet stream. Such fluctuations are typically present in the experimental conditions in the laboratory, and are due to turbulence and boundary layers in the nozzle. In the calculations this is done by replacing the streamwise velocity U at the inflow (i.e., for $Z = 0.0$ cm, $R \leq R_o$) with $U(1+p)$, where $p = p(R,t)$ is the perturbating term. This term is defined by a sine Fourier series in the variable R, with R_o the largest wavelength:

$$p(R,t) = \frac{1}{m} \sum_{m=1}^{M} p_m^o \delta_m(t) sin(\frac{2\pi m R}{R_o} + \phi_m), \qquad (6)$$

where $\delta_m(t)$ is a time-dependent amplitude defined by

$$\delta_m(t) = \frac{2(t - t_{om})}{\delta t_m}, \quad \text{if } t_{om} \leq t \leq \frac{(t_{om} + \delta t_m)}{2},$$

$$\delta_m(t) = \frac{2(t_{om} - t)}{\delta t_m} + 2, \quad \text{if } \frac{(t_{om} + \delta t_m)}{2} \leq t \leq t_{om} + \delta t_m,$$

and by

$$\delta_m(t) = 0,$$

otherwise. In this way, each added perturbation term varies between 0.0 and a maximum value of no more than a fraction p'_m/m of the inflow jet velocity. The duration δt_m, and maximum value p'_m of the amplitude, as well as the intrinsic phases $\phi_m (0 \leq \phi_m \leq 2\pi)$ of the terms in the Fourier series are randomly generated numbers.

A calculation was performed which restarted the program at time $t = 2.52$ ms of the previous unperturbed calculation, but now including inflow fluctuations. Here, the series in eq. (6) was truncated to its first four terms (M=4), including only fluctuations of wavelength R_o, $R_o/2$, $R_o/3$, and $R_o/4$, with durations δ_m in the range of $5 - 12.5$ μs ($20 - 50$ time steps). Due to the changes in the frequency content of the velocity field, the previous regularity in the flow pattern was lost in spite of the relatively small amplitude of the fluctuations. This can be seen by comparison of the spectral distributions obtained in this case with those for the unperturbed case discussed above. We compare Figs. 7a-c, for mean perturbation levels of 0.1, 1 and 5 %, respectively, with Fig. 5a. This indicates that the peaks become broader, while new high frequency peaks are now present. As the fluctuation level increases, the calculated spectral distribution

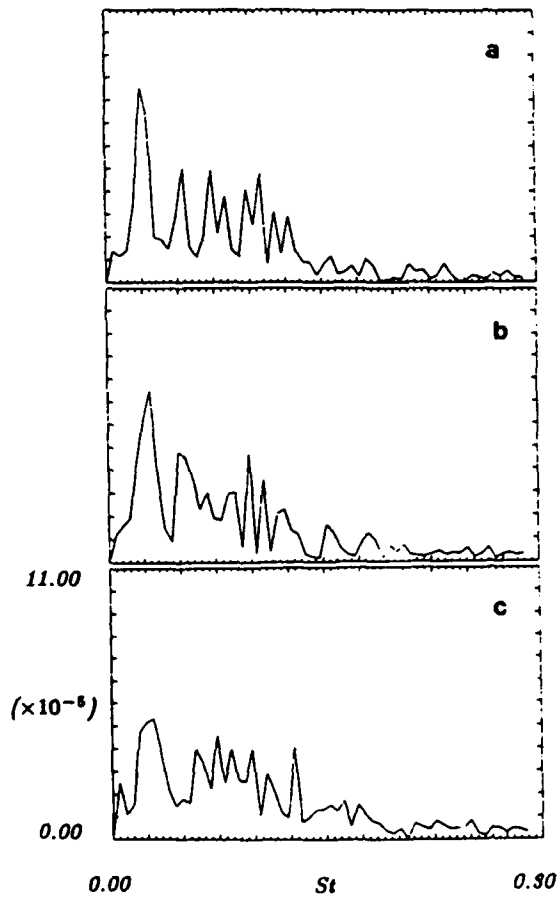


Fig. 7: Spectral amplitude at $R = D/2$ and $Z = 2.39$ cm ($z = 0.40$), as a function of the Strouhal frequency, for the randomly perturbed jet with, a) 0.1 %, b) 1 %, and c) 5 % fluctuation levels.

begins to look like the distributions obtained in the analysis of near field pressure fluctuations in experiments with non-excited-jets [12]. The experimentally obtained distributions consist of a broad peak tilted towards the low frequency side, and centered on $St_{jet} \approx 0.37$, in agreement with the value of St_{jet} found in the present numerical simulations.

Summary and Conclusions

We have presented results from numerical simulations of the evolution of the Kelvin-Helmholtz instability for a subsonic, compressible, axisymmetric, spatially-evolving shear layer. In the absence of boundary layers and small-scale inflow turbulence, the calculations indicate that the large-scale development of the unforced jet shear layer has an underlying degree of organization. This is the result of a feedback mechanism in which the shear layer ahead of the nozzle edge is modulated by the far field induced by the mergings downstream, near the end of the potential core of the jet. This is in agreement with the experimental observations [12].

The dominant frequencies in the unforced flow depend on the two length-scales of the jet, namely, the initial shear layer thickness and the jet diameter. These scales are associated with the two basic mechanisms in the jet, the jet shear layer instability and the jet column instability, which determine the high and low frequency natural modes of the jet, respectively. The spectral analysis of the axial velocity fluctuations at the center of the shear layer, just ahead of the nozzle edge, shows a sequence of peaks separated by a regular interval approximately equal to $St_o/4$. These peaks correspond to the subharmonics of the jet shear layer instability frequency St_o and to the non-linear interactions between them. The non-dimensional frequency of the preferred jet mode, $St_{jet} = (f_o/4)D/v_o \approx 0.36$, is in good agreement with the experimental values for subsonic jets. Moreover, the value for St_o was consistent with the predictions of linear inviscid instability theory. In addition, the distribution of the merging locations was in qualitative agreement with that observed in jet-experiments with low-level forcing.

The studies with random high frequency perturbations on the inflow velocity show that such perturbations tend to break the organized jet shear layer, and hence the temporal coherence between the structures. By generating more incoherent mergings the fluctuations tend to destroy the regularity of the idealized, unforced two-dimensional case.

Acknowledgements

We would like to acknowledge useful discussions with Dr. K. Kailasanath. This work was sponsored by the Mechanics Division of the Office of Naval Research and by the Naval Research Laboratory.

References

[1] Winant, C. D. and Browand, F. K., *J. Fluid Mech.* **63**, 237 (1974). Brown, G. and Roshko, A., *J. Fluid. Mech.* **64**, 775 (1974). Browand, F.K. and Ho, C. M., *Journal de Mécanique Théorique et Apliquée*, Numéro spécial, pp. 99-120 (1983), and references therein.

[2] Leonard, A., *J. Comp. Phys.* **37**, 289 (1980).

[3] Riley, J. J. and Metcalfe, R. W., in *Proc. of the Seventh Int. Conf. on Num. Meth. in Fluid Mech.*, ed. by W. C. Reynolds and R. W. MacCormack, Springer, New

York, 1981, p. 279.

[4] Corcos, G. M. and Sherman, F. S., *J. Fluid Mech.* 139, 29 (1984).

[5] Davis, R. W. and Moore, E. F., *Phys. Fluids* 28, 1626 (1985).

[6] Grinstein, F. F., Oran, E. S. and Boris, J. P., "Direct Simulation of Asymmetric Mixing in Planar Shear Flows", to appear in *J. Fluid Mech* (1986); see also NRL Memorandum Report 5621 (1985), Naval Research Laboratory, Washington, DC.

[7] Grinstein, F. F., Oran, E. S. and Boris, J. P., "Kelvin-Helmholtz Instability in unforced Spatially-Developing Mixing Layers", to be submitted to *J. Fluid Mech* (1985).

[8] Boris, J. P. and Book, D., in *Meth. in Comp. Phys.*, Vol. 16, Ch. 11, Academic Press, 1976; see also Boris, J. P., NRL Memorandum Report 327 (1976), Naval Research Laboratory, Washington, DC.

[9] Boris, J. P., Oran, E. S., Gardner, J. H., Grinstein, F. F. and Oswald C. E., in *Ninth Int. Conf. on Num. Meth. in Fluid. Mech.*, ed. by Soubbaramayer and J. P.Boujot, Springer, NY, 1985, pp. 98-102.

[10] Laufer, J., in *Transition and Turbulence*, ed. by R.E. Meyer, pp. 63-76, Academic Press, N.Y. (1981).

[11] Laufer, J. and Monkewitz, P.A., *AIAA paper* 80-0962, Hartford.

[12] Kibens, V., *AIAA J.* 18, 434 (1979).

[13] Michalke, A., "Instability of compressible circular free jet with consideration of the influence of the jet boundary layer thickness", *NASA Tech. Memo* 75190 (1977); Michalke, A. and Hermann, G., *J. Fluid Mech.* 114, 343 (1982).

APPENDIX E.

A Numerical Study of Transverse Jets into Supersonic Flows and Influence on Pressure Waves

A NUMERICAL STUDY OF TRANSVERSE JETS INTO SUPERSONIC FLOWS AND INFLUENCE OF PRESSURE WAVES

Extended Abstract
(Submitted to the AIAA 28th Aerospace Meeting)

C. Li* and K. Kailasanath

Center for Reactive Flow and Dynamical Systems
Laboratory for Computational Physics and Fluid Dynamics
Naval Research Laboratory
Washington, D.C. 20375

Address correspondence to:
C Li
Code 4410
Laboratory for Computational Physics and Fluid Dynamics
Naval Research Laboratory
Washington, D.C. 20375-5000

---

* Berkeley Research Associates, Springfield, VA 22150

Note: This is an extended abstract containing preliminary figures and a brief analysis of the results. The final figures and detailed analysis of the results will be submitted in the final draft of the paper.

Abstract

Numerical simulations of transverse jets into supersonic flows are conducted by solving the conservation equations of mass, momentum, energy, and species densities using a fourth order flux-corrected transport algorithm. These simulations show that the mixing between the jet and the main flow depends strongly on the shocks and expansion waves induced by the transverse jet. Therefore, a systematic study of the influence of these pressure waves on mixing is also being performed. Preliminary results highlight the role of expansion waves in the enhancement of mixing.

I. Introduction

Mixing and subsequent combustion between fuel and oxidizer in supersonic and hypersonic flows are crucial issues for air-breathing engines working in the high mach number regimes. However, these issues still remain largely unresolved. Further studies on mixing mechanisms and combustion in different flow configurations are needed in order to provide the necessary understanding for developing efficient and safe supersonic and hypersonic engines.

Molecular diffusion is a rather weak process and in general, it is not sufficient to fully mix the fuel and oxidizer for complete combustion in high speed flows. Therefore, vorticity production at the fuel-oxidizer interface and the generation of other time-dependent flow-structures may play a dominant role in the mixing enhancement by increasing the interface area substantially.

Since the experiments of Brown and Roshko [1], the mixing enhancement by large-scale flow structures in shear flows has received increased attention. The presence and significance of these large scale structures in supersonic flows have been the subject of a number of numerical studies [2-5]. Furthermore, there have been efforts to find different methods of creating and enhancing the large-scale structures. Among the methods, the possibility of using transverse injection has been studied [6][7]. It was qualitatively shown that mixing might be increased by carefully arranging the positions of several transverse jets. It was also shown that the transverse (i.e. normal to the main stream) momentum carried by the jet may perturb the mixing layer and create instability and, subsequently, large-scale vortical structures. However, more study is

needed to provide detailed information and understanding of the penetration of transverse jets into the supersonic main flow and their mixing with the main stream. In this paper, numerical simulations of transverse jets of different injection pressure will be presented. These numerical computations provide important information on the penetration of the jet and the impact of shocks and expansion waves created by the transverse jet on the mixing process.

II. Physical Assumptions and Numerical Method

The following physical assumptions are made in this numerical study: (1) The flow is two dimensional. (2) All diffusive transport processes are neglected. (3) The flow consists of ideal gas of constant ratio of specific heat (k=1.4).

The conservation equations of mass, momentum, energy, and species densities are solved using a fourth order flux-corrected transport algorithm[8]. The flux-corrected transport algorithm is a monotonic, conservative, positivity- preserving algorithm. In this algorithm, accuracy, robustness, and stability are achieved by introducing a diffusive flux and later correcting the calculated results by an antidiffusive flux with the flux limiter assuring the positivity and monotonicity.

III. Numerical Simulations of transverse Jets

Numerical simulation of a transverse jet of different pressures into the supersonic main flow are conducted in a 30cm x 5cm computational domain. The left and right boundaries of this computational domain are open. The top and bottom boundaries represent solid walls. These computations provide information on the overall development of the jet and its mixing with the main stream. The main stream inlet pressure is 101.3kPa, density is 0.5883kg/m^3, velocity in the x-direction is 2455 m/sec, and mach number is 5.0. A sonic transverse jet is injected from the bottom boundary. The jet is located between 4 and 5cm from the inlet. Two pessures are used: 202.6kPa (twice of the inlet pressure) and 1519kPa (fifteen times of the inlet pressure). The jet densities are also two and fifteen times of the inlet values, respectively. The density, pressure, air-fraction, and x-velocity fields from the calculated results are shown in Figs. 1 and 2. In these figures, a curved shock generated by the jet is also shown. The strength of the

shock depends on the injection pressure and other jet properties. The upper part of this curved shock is similar to an oblique shock which is reflected back to the central field by the top wall and later, intersects the interface between the jet flow and the main flow. The upper part of the flow field turns upward after the initial-curved shock and then, readjusts to the direction parallel to the upper wall after the reflected oblique shock. The lower part of the shock is similar to a normal shock and therefore, the pressure near the jet (after the shock) is close to the pressure after the normal shock at the inlet mach number and much higher than the downstream pressure in the two calculated cases. Therefore, a set of expansion waves emerges to allow the pressure after the jet to gradually reduce to the downstream pressure. Since the lower part of the flow field after the shock is subsonic, the flow almost turns 90 degrees to the jet direction and then, turns back to the direction parallel to the bottom wall through the set of expansion waves.

It is apparent from Figs. 1 and 2, that the initial penetration of the jet depends strongly on the injection pressure. However the final mixing is dominated by the large scale structures generated in the flow field although there exists a large difference in the penetration depth caused by the significantly different jet-pressures. In both cases, instability of the interface develops in the expansion region after the initial shock. In Fig. 1, since the initial shock is relatively weak, the reflected shock does not intersect the interface in the computational domain and the instability develops into rather large-scale structures in this long expansion region. However, in Fig. 2, the instability begins to develop in the expansion region after the initial shock. Then, the interface is pushed to the bottom wall by the reflected shock and the instability developed in the expansion region after the initial shock is suppressed by the reflected shock. After the interface passes through the reflected shock, instability develops again in the expansion region after this reflected shock.

In order to further study the initial penetration of the jet and the initial development of the instability, a 5cm x 2.5cm computational domain is employed. The left, right,and top boundaries are open and the bottom boundary simulates a solid wall. The main flow inlet conditions are the same as those used in the above calculations in the 30cm x 5cm domain. The jet properties are also the same except the jet pressures are 1013kPa (ten times the inlet pressure) and 1519kPa (fifteen times the inlet pressure) and the jet densities are ten and fifteen

times the inlet values, respectively. The jet is situated between 1 and 2cm from the inlet. Although the difference in the jet pressures is moderate and flow fields are similar in these two cases, strong dependency of the initial penetration of the jet on the injection pressure is again shown in Figs. 3 and 4. The initial development of the interface instability is also observed in the expansion region after the initial shock. The stronger instability observed in Fig. 4 may be attributed to the stronger density gradient. The results obtained in this 5cm x 2.5cm domain and the previous 30cm x 5cm domain show a strong velocity gradient in the interface region between the jet flow and the main stream. Also, the expansion fan after the initial shock generates strong pressure and density gradients. These gradients are likely to be the key-factors in the generation of intability and subsequently, large-scale structures. Furthermore expansion waves after the initial and reflected shocks seem to play a favorable role in the generation of instability and large-scale structures which greatly enhance the mixing. In order to fully understand the roles played by pressure waves in the generation of instability, production of large-scale structures, and enhancement of mixing between the jet and the main flow, it is helpful to conduct studies on interaction between mixing layers and shocks or expansion waves under more controlled conditions.

IV. Numerical Simulation of the Influence of Pressure Waves on Mixing

In supersonic flows, shocks and expansion waves are basic features of the flow field. As an interface of two different streams crosses these pressure waves, strong impact is expected on the structure and mixing character of the interface. It would be desirable to use these waves to create instability in the interface region and subsequently, the large-scale flow structures. Unfortunately, the influence of shock and expansion waves, i.e., which type of waves under what conditions will produce more favorable effects, has not yet been clearly understood. In [2], Numerical simulations were conducted for supersonic shear layers between two streams of different pressures. In those simulations, it was observed that large-scale vortices were indeed generated by reflected shock and expansion waves in some cases of underexpanded and overexpanded supersonic mixing layers. Similar results were also obtained in other simulations where small obstacles were used to generate shock and expansion waves[7][9]. Although the overall impact is visible, because of the complexity of the flow field and co-existence of different fami-

lies of the original and reflected shock and expansion waves, it is extremely difficult to clearly determine effects of different type of pressure waves on the generation of large-scale vortices and enhancement of mixing in those flow configurations. Furthermore, since the supersonic flows of very high speed were dealt with in those numerical simulations, the significant amount of momentum normal to the shear layer induced by those inserted bodies may also play an important role in vortex generation.

Although the earlier simulations have shown that large scale structures can be produced and subsequently, mixing be enhanced by pressure waves, generalization and application of these results will be very much case dependent unless more fundamental understanding of shock-interface and expansion-wave-interface interactions are obtained. Therefore, a systematic study of the interactions of shocks and expansion waves with supersonic mixing layer has been undertaken.

In order to study interactions between different pressure waves and supersonic mixing layers and clearly identify the role played by each type of those waves, some simple flow configurations need to be selected. The computational domain used in this study is 25cm in the x-direction and 2.5cm in the y-direction. A hydrogen stream and an air stream enter the left boundary of the computational domain parallel to each other. The upper, left, and right boundaries are open and the bottom boundary simulates a solid wall. Three cases are considered: (a) both streams are parallel to the x-direction, namely, parallel to the solid bottom wall; (b) the two streams have a pitch angle of 10 degrees counter-clockwise; (c) the pitch angle is 10 degrees clockwise. These three configurations are shown in schematics at the bottom of Figs. 5-7. The upper (hydrogen) stream: pressure=101.3kPa, density=0.244kg/m^3, and mach number=1.5; the bottom (air) flow: pressure=101.3kPa,density=3.53kg/m^3, and mach number=8.

It is shown in Fig. 6 that large scale structures are generated when the mixing layer passes through a family of expansion waves while no structures are generated in Fig. 5 where no strong shocks or expansion waves exist. In Fig. 7, the mixing layer passes through an oblique shock and the shock has no visible effects on the interface immediately after the shock. However, instability begins to develop further down stream in small scales but no large-scale structures are observed. In this set of results, the favorable role played by expansion waves on large scale vortex generation, which was suggested in the previous section, has been further confirmed

under the flow conditions used in these simulations.

V. Preliminary Conclusions

The penetration depth of a transverse jet into a supersonic flow depends on the injection pressure. However, the final mixing seems dominated by the large scale structures generated in the flow field.

The pressure waves (shocks and expansion waves) play an important role in generation and enhancement of large-scale vortices. The expansion waves apparently have more favorable effects. However, more numerical, analytical, and experimental studies are required to yield more definitive and general conclusions. Currently, more detailed numerical experiments are being carried out to further understand the role of pressure waves in supersonic mixing layers.

Acknowledgement

This work was supported by the Naval Research Laboratory through the Office of Naval Research. The support and encouragement of Drs. J. Boris and E. Oran is gratefully acknowledged. Several preliminary calculations were performed on the Graphical Array Processing System (GAPS) at the Laboratory of Computational Physics and Fluid Dynamics.

References

1. Brown, G.L. and Roshko, A., On Density Effects and Large Structure in Turbulent Mixing-Layers, J. Fluid Mech., 64:775,1974.

2. Guirguis, R.H., Grinstein, F.F.,Young, T.R., Oran, E.S., Kailasanath, K., Boris, J.P., Mixing Enhancement in Supersonic Shear Layers, AIAA-87-0369

3. Papamoschou, D. and Roshko, A., Observation of Supersonic Free Shear layers, Sadhana, 12:1,1988

4. Lele, S.K., Direct Numerical Simulation of Compressible Free Shear Flows, AIAA-89-0374.

5. Tang, W., Sankar,L.N., and Komerath, N., Mixing Enhancement in Supersonic Free Shear Layers, AIAA-89-0981.

6. Weidner, E.H. and Drummond, J.P., Numerical Study of Staged Fuel Injection for Supersonic Combustion, J. AIAA, 20:1426,1982

7. Drummond, J.P. and Mukunda, H.S., A Numerical Study of Mixing Enhancement in Supersonic Reacting Flow Fields, Presented at the 3rd. Int. Conf. on Numerical Combustion, Antibes, France, 1989.

8. Boris, J.P. and Book, D.L., Flux-Corrected Transport I: SHASTA, A Fluid Transport Algorithm that Works, J. Comp. Phy.,11:38,1973

9. Guirguis, R.H. Mixing Enhancement in Supersonic Shear Layers: III. Effect of Convective Mach Number, AIAA-88-0701.

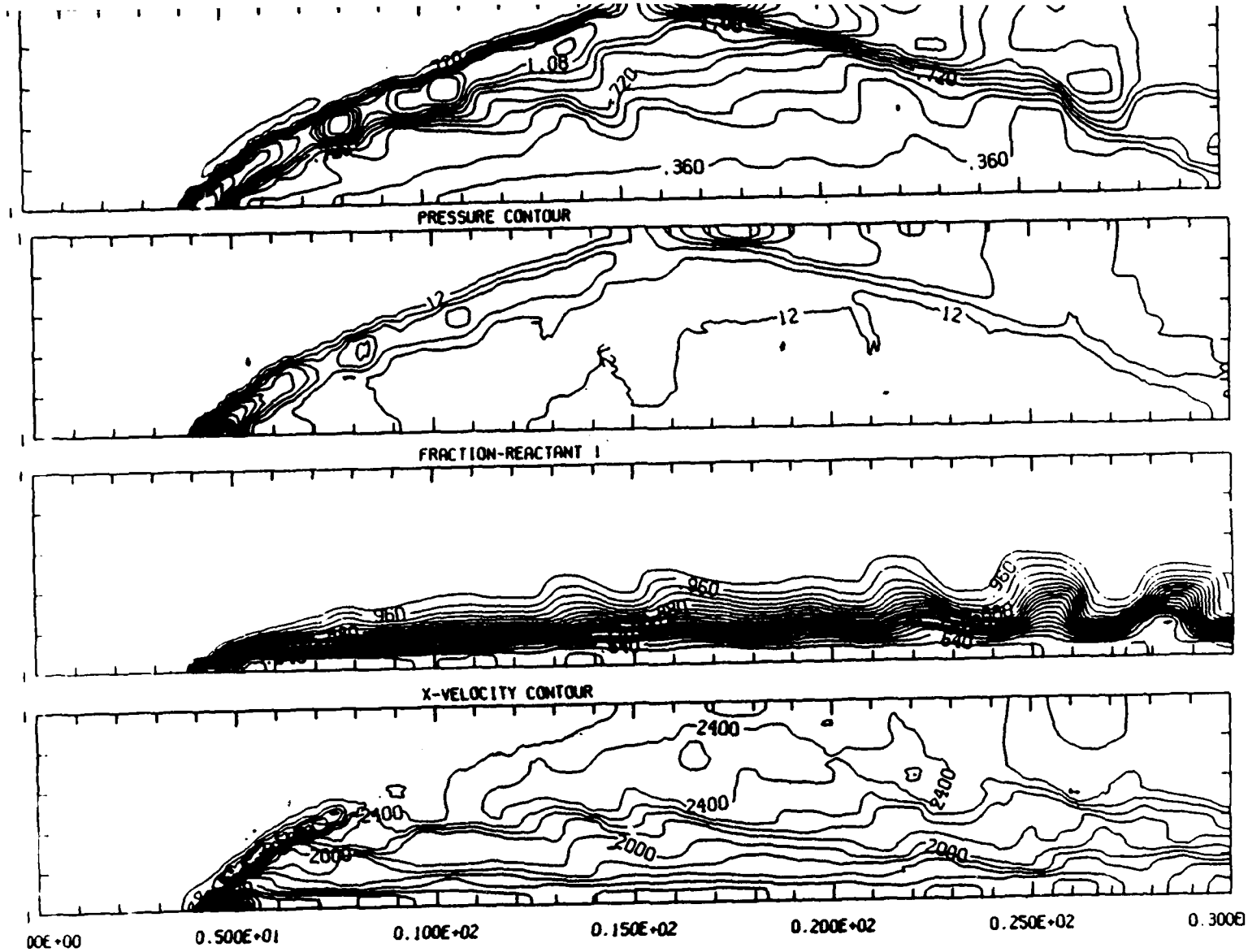


Figure 1. Computed Results for a Transverse Jet into a Supersonic Flow in a 30cm x 5cm Domain. The Main Flow: Density=0.5883kg/m^3, Pressure=101.3kPa, and Mach Number=5.0; and the Sonic Transverse Jet: Density=1.177kg/m^3, Pressure=202.6kPa, and Mach Number=1.0. The Jet Is Located between 4 and 5cm from the Inlet.

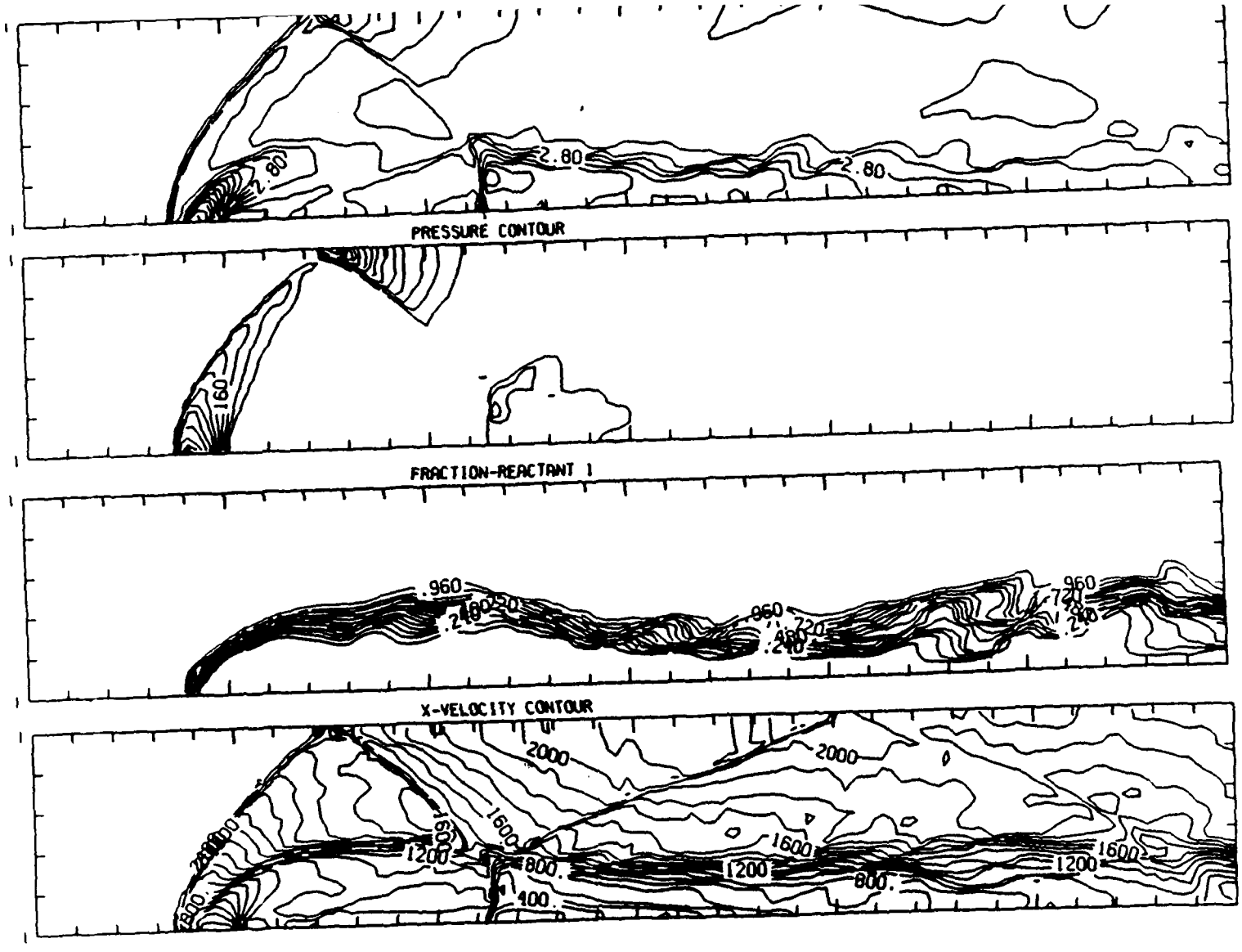


Figure 2. Computed Results for a Transverse Jet into a Supersonic Flow in a 30cm x 5cm Domain. All flow conditions are the Same as in Fig.1 except for the transverse jet density=8.824kg/m^3 and pressure=1519kPa.

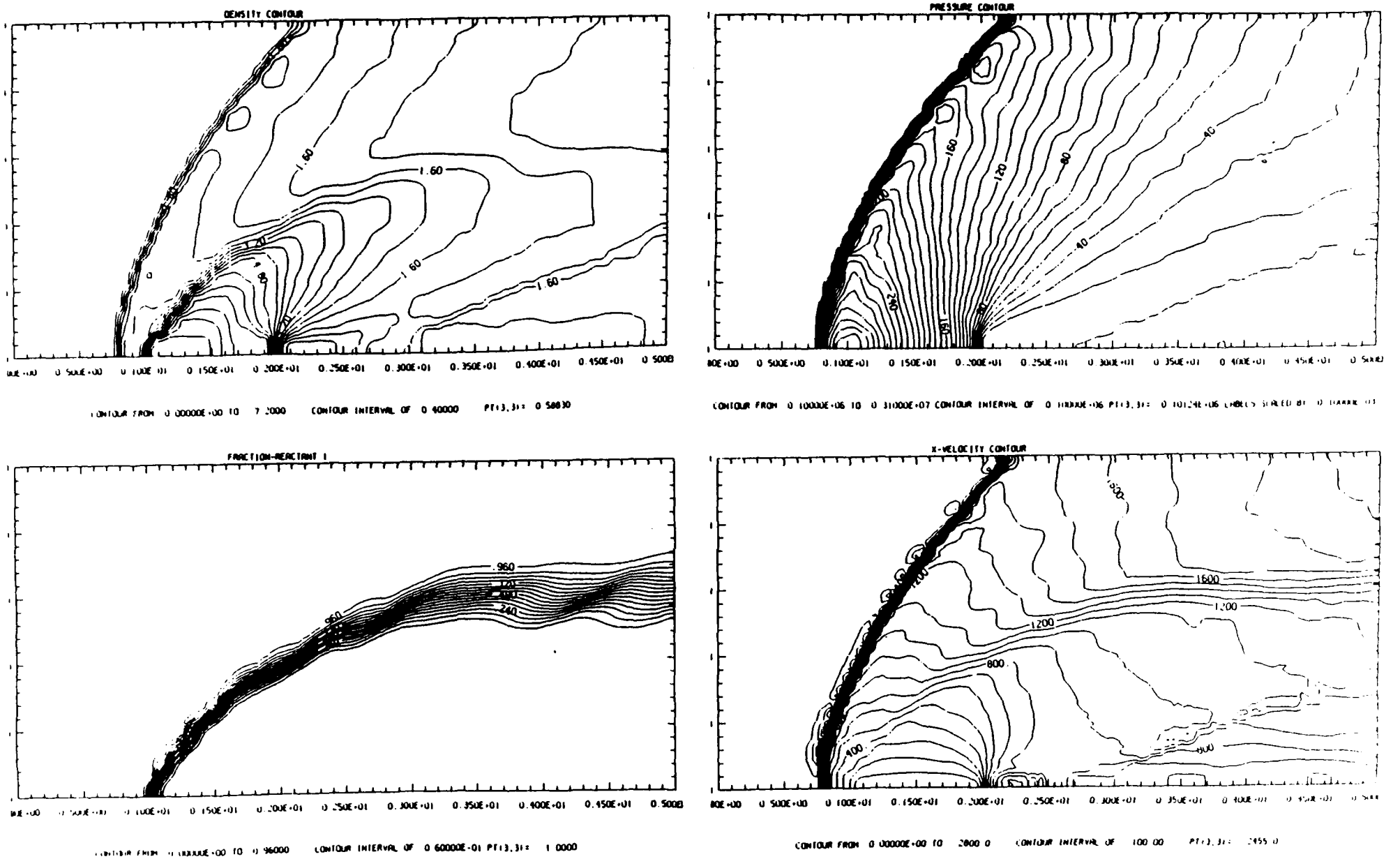


Figure 3. Computed Results for a Transverse Jet into a Supersonic Flow in a 5cm x 2.5cm Domain. The Main Flow conditions are the same in Fig. 1; and the Sonic Transverse Jet: Density=5.883kg/m^3, Pressure=1013kPa. The Jet Is Located between 1 and 2cm from the Inlet.

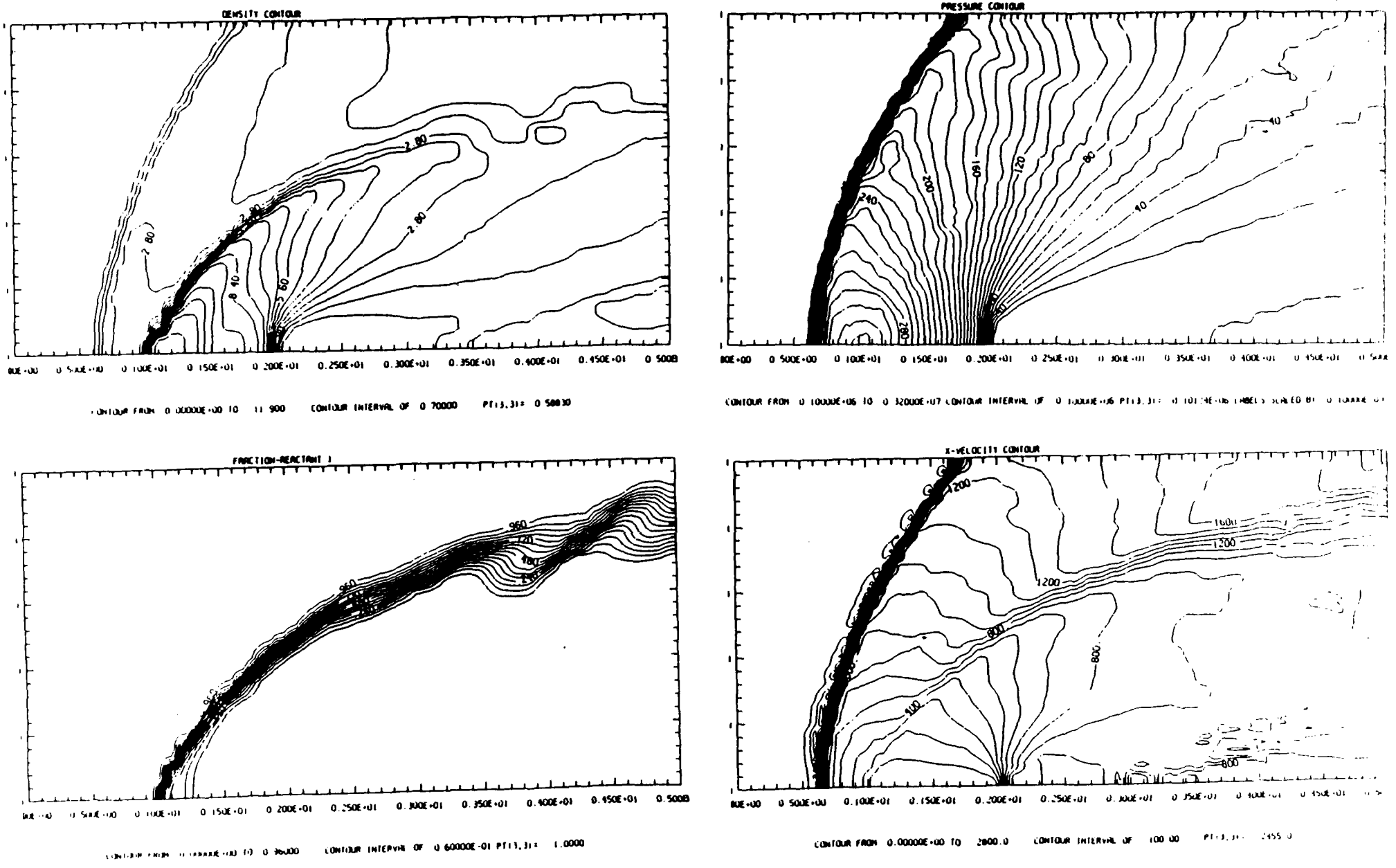


Figure 4. Computed Results for a Transverse Jet into a Supersonic Flow in a 5cm x 2.5cm Domain. All Flow Conditions Are the Same as in Fig. 3 except the jet density=8.824kg/m^3 and pressure=1519kPa.

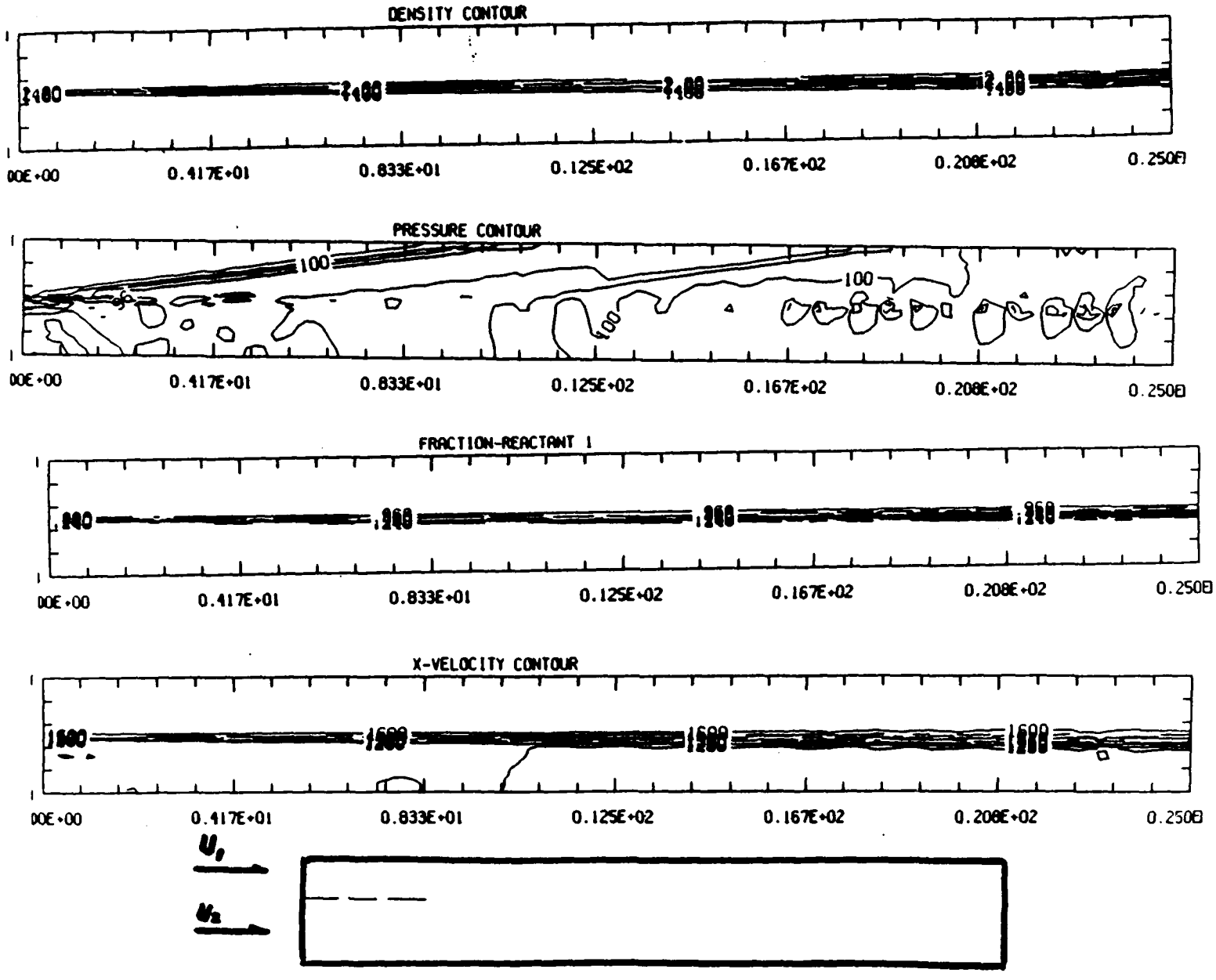


Figure 5. Computed Results for a Supersonic Mixing Layer in 2.5cm x 25cm Domain. The Bottom Flow: Density=3.530kg/m^3, Pressure=101.3kPa, and Mach Number=8. The Upper Flow: Density=0.244kg/m^3, Pressure=101.3kPa, and Mach Number=1.5. The Flows Are Parallel to the Bottom boundary.

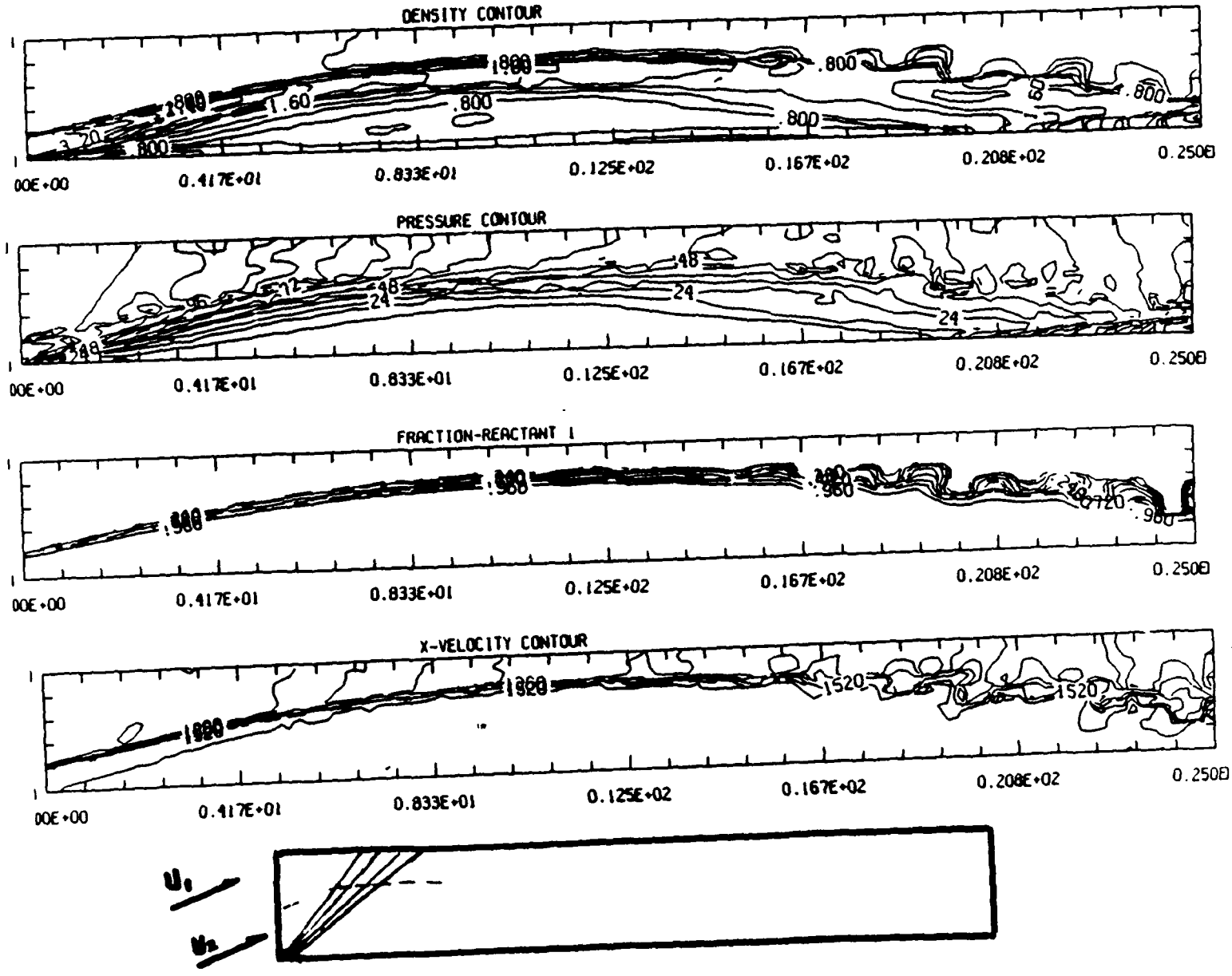


Figure 6. Computed Results for a Supersonic Mixing Layer in 2.5cm x 25cm Domain. The Flow Conditions Are the Same as in Fig. 5 except the Flows Are 10 Degrees from the Bottom Boundary.

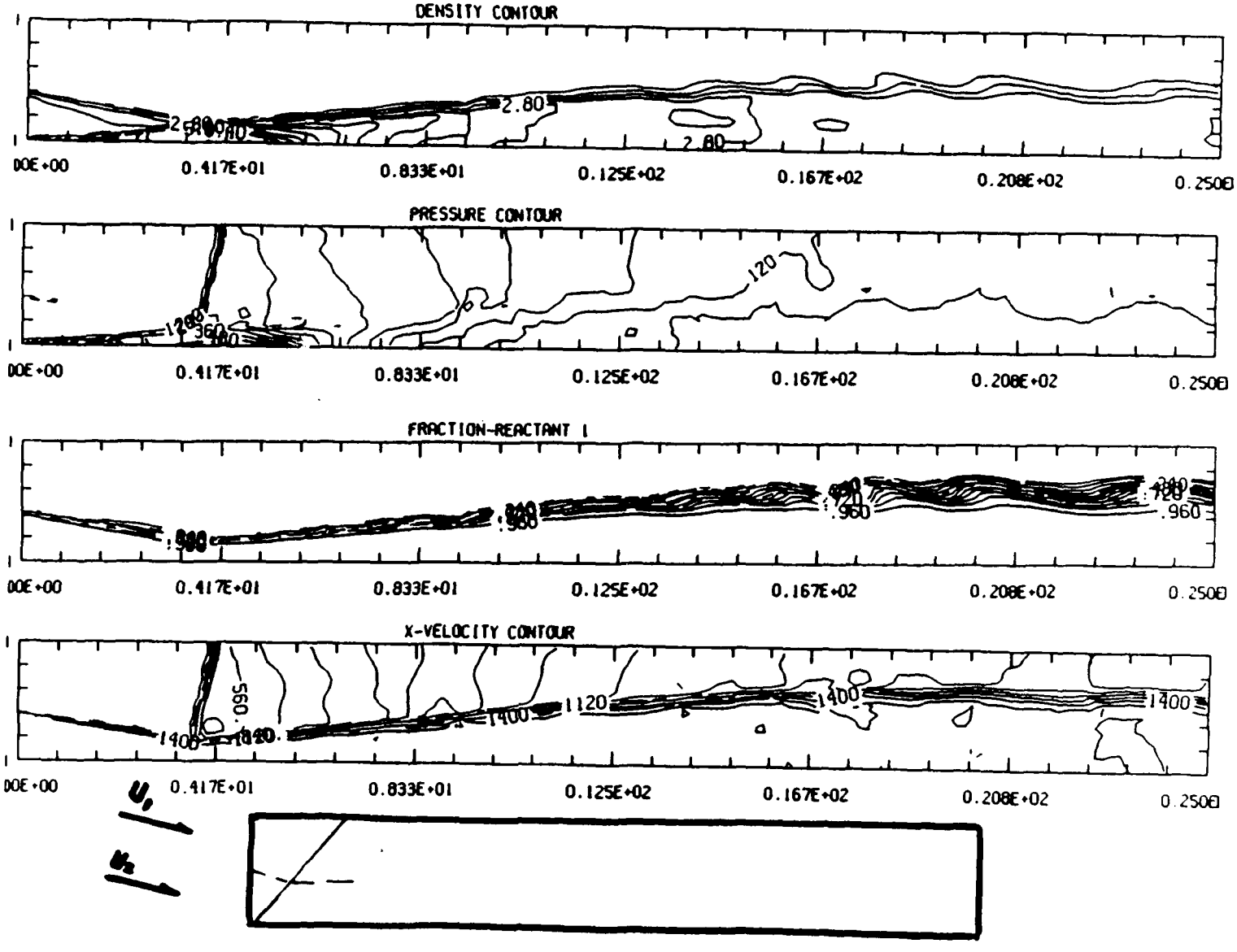


Figure 7. Computed Results for a Supersonic Mixing Layer in 2.5cm x 25cm Domain. The Flow Conditions Are the Same as in Fig. 5 except the Flows Are -10 Degrees from the Bottom Boundary.

Figure Captions

Figure 1. Computed Results for a Transverse Jet into a Supersonic Flow in a 30cm x 5cm Domain. The Main Flow: Density=0.5883kg/m^3, Pressure=101.3kPa, and Mach Number=5.0; and the Sonic Transverse Jet: Density=1.177kg/m^3, Pressure=202.6kPa, and Mach Number=1.0. The Jet Is Located between 4 and 5cm from the Inlet.

Figure 2. Computed Results for a Transverse Jet into a Supersonic Flow in a 30cm x 5cm Domain. All flow conditions are the Same as in Fig.1 except for the transverse jet density=8.824kg/m^3 and pressure=1519kPa.

Figure 3. Computed Results for a Transverse Jet into a Supersonic Flow in a 5cm x 2.5cm Domain. The Main Flow conditions are the same in Fig. 1; and the Sonic Transverse Jet: Density=5.883kg/m^3, Pressure=1013kPa. The Jet Is Located between 1 and 2cm from the Inlet.

Figure 4. Computed Results for a Transverse Jet into a Supersonic Flow in a 5cm x 2.5cm Domain. All Flow Conditions Are the Same as in Fig. 3 except the jet density=8.824kg/m^3 and pressure=1519kPa.

Figure 5. Computed Results for a Supersonic Mixing Layer in 2.5cm x 25cm Domain. The Bottom Flow: Density=3.530kg/m^3, Pressure=101.3kPa, and Mach Number=8. The Upper Flow: Density=0.244kg/m^3, Pressure=101.3kPa, and Mach Number=1.5. The Flows Are Parallel to the Bottom boundary.

Figure 6. Computed Results for a Supersonic Mixing Layer in 2.5cm x 25cm Domain. The Flow Conditions Are the Same as in Fig. 5 except the Flows Are 10 Degrees from the Bottom Boundary.

Figure 7. Computed Results for a Supersonic Mixing Layer in 2.5cm x 25cm Domain. The Flow Conditions Are the Same as in Fig. 5 except the Flows Are -10 Degrees from the Bottom Boundary.

APPENDIX F.

Mixing Enhancement in Supersonic Shear Layers: III. Effect of Convective Mach Number (AIAA-88-0701)

Mixing Enhancement in Supersonic Shear Layers: III. Effect of Convective Mach number

Raafat H. Guirguis

Abstract

This paper addresses some of the fundamental questions concerning the definition of a supersonic shear layer and the effect of convective Mach number on the mixing and on the structure of the shear layer. The study shows that the convective Mach number indeed describes the intrinsic character of the instability of a shear layer. A supersonic shear layer is defined as having a supersonic convective Mach number. Mixing is enhanced when the convective Mach number is reduced.

Introduction

With the growing interest in supersonic combustion, there is renewed interest in supersonic shear layers[1-7]. The work done in the 1960's on supersonic jets focused on the study of the wake and the noise caused by a supersonic jet discharging into a stagnant background. Now we are concerned with mixing in supersonic shear layers as a first step in understanding combustion at supersonic speeds.

Supersonic combustion means burning the fuel into a supersonic stream of the oxidizer, without decelerating the flow to subsonic speeds. In the combustor, the high speed of the flow makes it difficult to mix the fuel and the oxidizer in the very short time it takes the flow to pass through the combustor. Traditionally, mixing is accomplished by injecting a jet of fuel into the oxidizer stream. Jets and shear layers are naturally unstable and usually lead to large scale mixing. Unfortunately, at high Mach numbers, jets and shear layers become more stable, which reduces the amount of mixing in a given channel length.

In a previous paper[5], we used time-dependent two-dimensional numerical simulations to study the effect of underexpansion and overexpansion on the mixing process in confined supersonic shear layers. The results of the simulations indicated that mixing in confined supersonic shear layers is enhanced when the pressures of the two streams are different. We also studied the structure of supersonic shear layers and how it is different from the structure of subsonic shear layers. We showed that large vortex structures are formed. However, these structures are not as coherent as those structures observed in subsonic shear layers. We also showed that pressure plumes, resulting from the interaction between the flow and the large vortices, tend to inhibit vortex merging. The goal of this paper is to resolve some of the fundamental questions concerning the definition of a supersonic layer, the sources of noise in numerical simulations of supersonic shear layers, and the effect of convective Mach number on the mixing and on the structure of the shear layer.

Model

We have made the following assumptions in the model:

1. The two streams are ideal gases with constant specific heats.
2. All diffusive transport processes are neglected. Thus only inviscid or convective mixing is considered.
3. The flow is two-dimensional.
4. At the inlet, all flow variables are specified and remain constant with time.

Definition of a supersonic shear layer

A basic question that has to be addressed is what makes a shear layer supersonic. As will become obvious later in this paper, a supersonic faster stream is not sufficient to produce a supersonic shear layer, even if the slower stream is quiscent. In his thesis, Papamoschou[4] gave a comprehensive review of the concept of convective Mach number. It is the Mach number in a frame of reference convecting with the large structures or the dominant waves. The concept of a convective Mach number was originally derived in connection with studies of linear stability of infinitesimally thin vortex sheets. In turbulent finite-thickness shear layers, Bogdanoff[1] and Papamoschou[4] used heuristic arguments to derive an expression of the convective Mach number in terms of the Mach number of the faster stream and the velocity and density ratios. Denoting the properties of the faster and slower streams by subscripts 1 and 2, respectively, the convective Mach numbers are defined as

$$M_{c1} = \frac{u_1 - u_c}{a_1}$$

and

$$M_{c2} = \frac{u_c - u_2}{a_2}.$$

The derivation by Papamoschou assumes that in a system of coordinates moving with the large structures, there exist stagnation points common to both streams. The assumption is justified if the interface rolls up because the flow structure has to allow some periodicity in the streamwise direction. Whether supersonic shear layers roll up or not, and how essential is the roll up to the derivation of the expression for the convective Mach number is an open question. The results described below should increase the degree of confidence in the expression of the convective Mach number. Assuming $\gamma_1 = \gamma_2$,

$$M_{c1} = M_{c2} = M_c = M_1 \frac{1 - \frac{1}{r_s}}{1 + r_\rho},$$

where $r_u = u_1/u_2$ and $r_\rho = \sqrt{\rho_1/\rho_2}$. Experimental data[1-4] of the growth rate of a compressible shear layer when normalized by the growth rate of an incompressible shear layer at the same velocity and density ratios appear to collapse on a single curve if plotted against the convective Mach number. The data also indicates that the growth rate start decreasing rapidly near $M_c = 0.4 - 0.5$.

Since subsonic shear layers roll up in a coherent manner, it is natural to ask ourselves if a shear layer, with at least the faster stream coming in at supersonic speed, can be made to behave as a subsonic shear layer and roll up by changing the flow variables in such a way as to reduce M_c below 1. Here, we have two classes of problems:

1. Both streams are supersonic

In this case, min $M_2 = 1$. From the definition of Mach number, $r_u = M_1/r_\rho$. This dependency between r_ρ and M_1 imposes a lower bound on M_c. Since M_c is proportional to M_1, the minimum value M_{cmin} occurs at $M_1 = 1$, hence

$$M_{cmin} = \frac{1 - \frac{1}{r_u}}{1 + \frac{1}{r_u}}.$$

In the table below, M_{cmin} is given in terms of r_u. Noticing that the spreading rate of mixing layers increases with the velocity ratio, it becomes clear that trying to reduce M_c below 0.5, while keeping $M_2 \geq 1$, requires reducing the velocity ratio below 3, in which case the shear layer is thin, even if it is incompressible. What we need is to identify a class of cases for which r_ρ and M_1 are independent.

r_u	10	5	3	2	1
M_c	0.81	0.67	0.5	0.33	0

2. Only the faster stream is supersonic

Although $M_2 < 1$ reduces M_c further, a subsonic stream 2 is unfavored because it allows the sound waves to propagate upstream the disturbances generated downstream. Allowing any disturbances to propagate upstream of the splitter plate changes the character of the instability because the slower stream will enter the domain of solution already in an excited state. Instead, we elect to impose on the definition of a supersonic shear layer the condition that disturbances are not allowed to propagate upstream of the tip of the splitter plate. This definition, excludes the cases $0 < M_2 < 1$. $M_2 = 0$ is included, however, by assuming that a wall confines the stagnant lower half of the flow at the inlet (left boundary). The wall should prevent the disturbances created downstream from propagating upstream. The main advatage of $u_2 = 0$ is that it causes r_ρ and M_1 to become independent, hence

$$M_c = \frac{M_1}{1 + r_\rho}.$$

In this case, we can reduce M_c, while keeping M_1 constant, by increasing the density ratio r_ρ.

Source of noise

Another question that has to be addressed is what triggers the instability in the a numerical simulation of a supersonic shear layer. Mechanical vibrations, flow diturbances generated in a compressor, bends in a duct, turbulent boundary layers upstream, are but few of the many sources of random noise in an experiment. But in a numerical simulation, the velocity, density, and energy at the inlet are specified. The intensity of turbulence at the inlet is zero by definition. Even if the initial conditions were not uniform, the disturbances will be swept out of the computational domain since the flow at the inlet is supersonic and sound waves cannot propagate these disturbances upstream.

Roundoff is a source of relatively small errors, depending on the accuracy (number of bits per word) of the computer used. Truncation is another source of errors. However, these errors are not truly random and cannot be considered noise. In other words, if we start from the same value, the errors committed are the same. The question to ask then, is not what is the source of noise, but what makes the flow parameters at one grid point different from the flow parameters at another point. Once different, the errors introduced in each will be different, which creates a disturbance. If the numerical diffusion inherent in the algorithm used is not excessive to the extent of damping the disturbances, and if the flow conditions are in an unstable regime, these disturbances will be amplified.

In the spatial shear layers described below, the transfer of x-momentum from the faster to the slower stream is what causes the flow parameters at point 1a in Fig. 1, to become different from the flow parameters at point 1b, and different from 1c, etc. On the slower stream side, 2a becomes different from 2b, and different from 2c, etc. The transfer of x-momentum is due to numerical diffusion inherent in finite-difference algorithms. After the first timestep, $u_{1a} < u_1$ and $u_{2a} > u_2$. Since the x-velocity of the flow at the inlet is specified, in the next timestep, the pressure at grid point 1a increases while that at point 2a decreases. The disturbance created at the tip of the splitter plate then convects with the flow and is eventually amplified if the flow conditions happens to fall in an unstable regime.

Numerical Shadowgraphs

In a previous paper[5], we introduced an equivalent of a shadowgraph which was produced by photocopying the contours of mass fraction for an underexpanded supersonic shear layer using an out-of-focus

lens. In a shadowgraph of a shear layer, optical visualization is possible if the two streams have different indices of refraction. Since the largest gradient in the index of refraction occurs at the interface, the shadowgraph highlights the interface between the two streams. It is difficult to experimentally produce a purely two-dimensional flow. Optical effects depicted on a sensitive film are usually the result of integrating the change in some optical property along the third dimension.

In a numerical simulation , the composition at a given point is represented by the mass fraction. If the indices of refraction of the two streams are different, the regions where the gradient of the index of refraction is large are the regions where the gradient of composition or mass fraction is large, i.e. the regions where the contours are concentrated. The contours of mass fraction contains all the information in a shadowgraph, except that the large gradients have to be highlighted. For that I use a regular Xerox machine and hold the original a few centimeters from the document glass. Out-of-focus copying not only highlights the dense regions, but also diffuses the concentration of the contours, thus enhancing the resemblence between the resulting interface and that depicted in shadowgraghs where high gradients are diffused in all, including the third, dimension.

Results and Discussion

In the calculations described below we used a two-dimensional computational domain 20 cm long and 2.4 cm high. We used a 200x80 uniformly spaced grid, hence $\delta x = 0.1$ cm and $\delta y = 0.03$ cm. The details of the numerical model are given in reference 5. Figure 2, reproduced from reference 5, compares the contours of mass fraction for an underexpanded confined, equal-pressure unconfined, and overexpanded confined supersonic shear layers. When the pressures of the two streams are different, the high pressure stream expands through a rarefaction fan centered at the tip of the splitter plate while the low pressure stream is compressed to the same pressure through an oblique shock wave also attached to the tip of the splitter plate. The actual flow parameters affecting the properties of the shear layer are the values ahead (downstream) of these waves. The flow parameters behind (upstream) the splitter plate are never realized by the shear layer.

In order to isolate the effects of underexpansion and overexpansion while keeping all other parameters fixed, the inlet conditions were selected such that the actual flow parameters are the same in all three cases: $p_1 = p_2 = 2.452$ atm, $M_1 = 4.903, M_2 = 1.355$, $T_1 = 521.6$K and $T_2 = 636.2$K, yielding $u_1/u_2 = 3.276, \rho_1/\rho_2 = 1.220$, and a convective Mach number $M_c = 1.618 > 1$. Both the underexpanded and the overexpanded shear layers show large scale mixing. In the equal-pressure unconfined case (b), the shear layer is much more stable and no large scale mixing is observed. The slight spread in the shear layer is caused by the small residual numerical diffusion in the algorithm. This spread is also a measure of the mixing by numerical diffusion that is superimposed on the large scale convective mixing in the confined cases. In the early stages of the calculation the thin vortex sheet spread uniformly in the y direction. The first large scale structure to form was usually observed at a point nearly 2/3 of the way downstream of the channel length. In other words, the initial disturbance that was created at the tip of the splitter plate at the start did not form a large structure until it was convected 2/3 of the length of the channel. When $M_c > 1$, both streams are moving at supersonic speeds relative to the large structures. Thus, it was not surprising to observe a system of oblique shock waves attached to the large structures[5].

In order to investigate the effect of reducing the convective Mach number below 1 in the unconfined case in Fig. 2b, I reduced the velocity ratio and increased the density ratio by changing the temperature ratio ($T_1 = 300$K, $T_2 = 1200$K), keeping the same Mach numbers and pressures, yielding $u_1/u_2 = 1.809, \rho_1/\rho_2 = 4.0$, and $M_c = 0.731$. The resulting mass fraction contours are shown in Fig. 3. Due to the decrease in the velocity ratio the shear layer remained stable. The instability observed near $x = 17$ cm is transient. It appeared always near the same location, was swept out of the domain and then reappeared again, etc.

In order to reduce M_c without reducing u_1/u_2, I then used a supersonic stream, $M_1 = 2$, in contact with a quiscent back ground, $u_2 = 0$, confined by a wall at the inlet and assumed a large density ratio, $\rho_1/\rho_2 = 16$ ($T_1 = 300$K, $T_2 = 4800$K), yielding $M_c = 0.4$. The mass fraction contours of Fig. 4, illustrate the time evolution of the shear layer. Although the faster stream is supersonic, the instability and breakup patterns of the shear layer are closely resembling those patterns observed in subsonic shear layers. The mixing accomplished within the length of the system is also enhanced. During the calculation, the disturbance that was created at the tip of the splitter plate at the start caused the interface at the tip to start rolling immediately. As time progressed, new vortex structures appeared at the tip while those structures formed earlier convected downstream and grew in size. Downstream, we can still observe a stable shear layer. It remains stable, until the first disturbance created at the tip rteaches it.

The advantages of using the technique described above to produce a shadowgraph, are illustrated in Figs. 5 and 6. In Fig. 5, we compare between the mass fraction contours before and after it was copied out-of-focus. The procedure highlights the interface. Large structures 1 and 2 are diffused and appear as a wiggle in the shear layer. Near the splitter plate, the large strctures are too small to be resolved and they appear as wiggles. Downstream, where the large vortices are well resolved, we can make observations of the rollup of the interface and of the coherence of the large structures. Figure 6 is a comparison of three shadowgraphs. Case (a), reproduced from reference

8, is a shadowgraph of the mixing layer between two subsonic streams. Case (b) is an equivalent shadowgraph of a supersonic stream into a quiscent background but at a convective Mach number $M_c < 1$. Case (c), the underexpanded case in Fig. 2a, illustrates the incoherent character of the large structures that can be observed in supersonic shear layers, i.e. when $M_c > 1$.

Conclusions

1. The convective Mach number indeed describes the intrinsic character of the instability of the shear layer, as predicted by the studies of Bogdanoff[1], Papamoschou[4], and Papamoschou and Roshko[2].
2. If $M_c < 1$ but at least one stream is supersonic, the flow structure (pressure, density) may look different from the flow structure when the convective Mach number is the same but both streams are subsonic. However, the intrinsic character of the instability (vorticity) should be the same.
3. Supersonic shear layer exhibit well defined large structures, but they are not as coherent as those structures observed in subsonic shear layers.
4. Mixing is enhanced when the convective Mach number is reduced.
5. The transfer of x-momentum in the vicinity of the tip of the splitter plate from the faster to the slower stream, due to numerical diffusion in the y-direction, is identified as the mechanism which triggers the shear layer instability in numerical simulations.

ACKNOWLEDGEMENT

Paper AIAA 88-0701 was originally submitted to AIAA under the title: *Mixing Enhancement in Supersonic Shear Layers: II. Effect of Bluff Center Bodies*, by R. H. Guirguis, T. R. Young, F. F. Gristein, E. S. Oran, and J. P. Boris, from the Laboratory for Computational Physics and Fluid Dynamics (LCP&FD), at Naval Research Laboratory (NRL), Washington, D.C 20375. Since I am no longer affiliated with NRL, some of the coauthors expressed to me their concern about my right to present the paper because the quality of my presentation can indirectly reflect on the quality of the work done at LCP&FD. The work done on the effect of bluff bodies was carried out when I was working at NRL and the coauthors contributed to the work. Therefore, I can't but honor their request of not presenting the paper on the effect of bluff center bodies and hope that this acknowledgement releases them from any responsibility.

However, since their request to withdraw the paper was addressed to me near the middle of December, and since AIAA requires a confirmation on the intent to present papers by November 5, and I did confirm it at the time, I felt that it would be improper to withdraw it at such a late date. I believe that my confirmation was a promiss to the meeting attendees who are interested in the subject of supersonic combustion to a least make available the extended abstract that was accepted by the reviewers and that was cleared for publication at the time it was submitted. This abstract is included as an appendix.

Based on the above, and to make up for not presenting the full paper, I decided to add the work in this paper, which is a new look on the peculiarities of supersonic shear layers, using some of my old published results[5], and a theoretical investigation of the effects of convective Mach using the results of 2 old computer runs in order to test the conclusions of the theoretical analysis. These 2 computer runs were initiated and were carried out by myself on the GAPS[9] system of LCP&FD when I was working at NRL, and although the results of these 2 runs are not published yet, they were presented by myself at an earlier meeting.

Finally I would like to express my appreciation to Jay P. Boris for making the GAPS system available to me when I was part of LCP, and would like to give him credit for starting the simulations of subsonic shear layers and flows on wide splitter plates which brought my attention the possibility of using bluff bodies to enhance the mixing in supersonic shear layers. I would also like to express my appreciation to Theodore Young for supplying me with the original fluid dynamics GAPS codes and for taking the time to teach me the architecture of multi-processor systems in order to modify the codes for my applications. I would also like to thank Fernando Grinstein for many helpful discussions and appologize to him for not presenting the full paper in which he is a coauthor.

References

1. Bogdanoff, D.W., Compressibility Effects in Turbulent Shear Layers, *AIAA J.*, 21, 926, 1983.
2. Papamoschou, D. and Roshko, A., Observations of Supersonic Free Shear Layers, *AIAA* 86-0162, 1986.
3. Chinzei, N., Masuya, G., Komuro, T., Murakami, A., and Kudou, K., Spreading of Two-Stream Supersonic Turbulent Mixing Layers, *Phys. Fluids*, 29, 1345, 1986.
4. Papamoschou, D., Experimental Investigation of Heterogeneous Compressible Shear Layers, Ph.D. thesis, Cal. Tech., Pasadena, California, 1987.
5. Guirguis, R.H., Grinstein, F.F., Young, T.R., Oran, E.S., Kailasanath, K., and Boris, J.P., Mixing Enhancement in Supersonic Shear Layers, *AIAA* 87-0373, 1987.

6. Drummond, J.P. and Hussaini, M.Y., Numerical Simulation of a Supersonic Reacting Mixing Layer, *AIAA* 87-1325, 1987.
7. Ragab, S.A. and Wu, J. L., Linear Instability Waves in Supersonic Turbulent Mixing Layers, *AIAA* 87-1418, 1987.
8. Roshko, A., Structure of Turbulent Shear Flows: A New Look, *AIAA J.*, 14, 1349, 1976.
9. Boris, J.P., Reusser, E., and Young, T.R., The Graphical and Array Processing System (GAPS), to appear as NRL memo. rep., Washington, D.C. 20375, 1987.

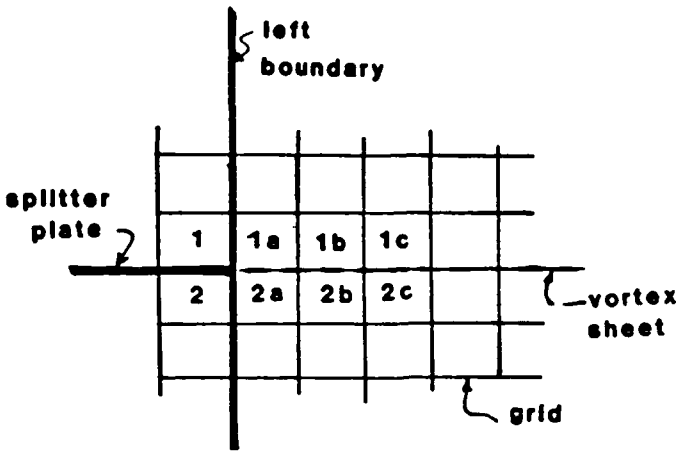


Fig. 1 Schematic representation of the computational grid in the vicinity of the splitter plate.

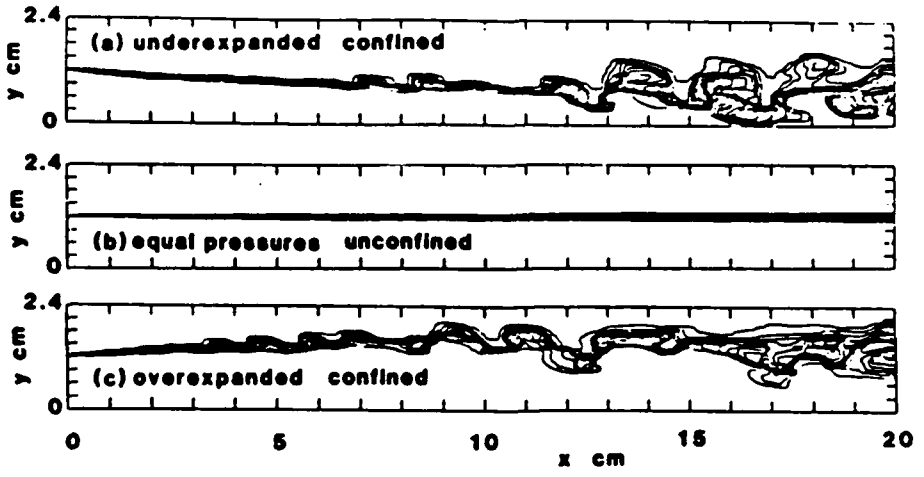


Fig. 2 Comparison of the mixing contours for (a) underexpanded confined, (b) equal-pressure unconfined, and (c) overexpanded confined supersonic shear layers. The velocity ratio, density ratio,and the convective Mach number are the same in all three cases: $u_1/u_2 = 3.276, \rho_1/\rho_2 = 1.220, M_c = 1.618$.

Fig. 3 Mixing contours for an unconfined shear layer. Both streams are supersonic; $M_1 = 4.903, M_2 = 1.355$; $u_1/u_2 = 1.809, \rho_1/\rho_2 = 4.0, M_c = 0.731$.

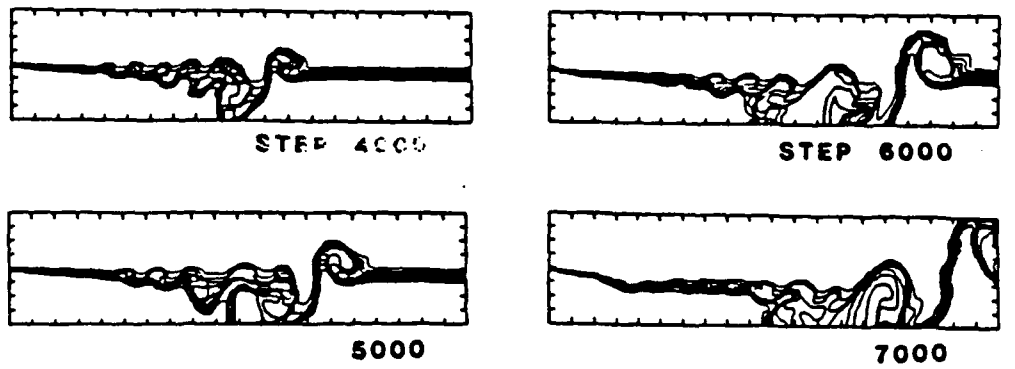


Fig. 4 Time evolution of an unconfined shear layer between a supersonic stream and a stagnant back ground; $M_1 = 2, \rho_1/\rho_2 = 16, M_c = 0, 4$.

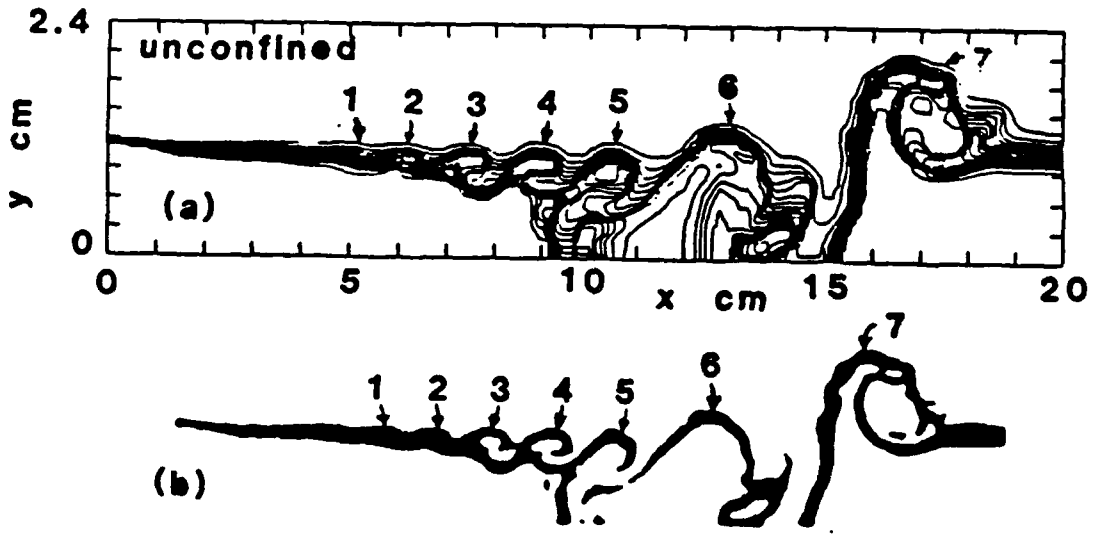


Fig. 5 Producing the equivalent of a shadowgraph by photocopying the contours of mass fraction using an out-of-focus lens. The procedure highlights the interface.

Fig. 6 Comparison of (a) shadowgraph of a mixing layer between two subsonic streams (reproduced from A. Roshko, *AIAA J.*, 14, 1349, 1976), and (b,c) numerically produced shadowgraphs of shear layers in which at least one stream is supersonic. The flow parameters are: (b) $M_1 = 2, M_2 = 0, \rho_1/\rho_2 = 16, M_c = 0,4$; (c) $u_1/u_2 = 3.276, \rho_1/\rho_2 = 1.220, M_c = 1.618$.

APPENDIX G.

Numerical Simulation of Unforced Spatially-Developing Mixing Layers

NUMERICAL SIMULATION OF UNFORCED SPATIALLY-DEVELOPING MIXING LAYERS

F.F. Grinstein*, E.S. Oran, and J.P. Boris

Laboratory for Computational Physics
Naval Research Laboratory
Washington, D.C., 20375

ABSTRACT

Two-dimensional numerical simulations of the Kelvin-Helmholtz instability in compressible, spatially-evolving, unforced, planar shear layers are used to investigate the reinitiation of unstable vortex roll-up near the trailing edge of the splitter plate. The calculated flow pattern of coherent structures shows that the process by which the instability is reinitiated, and thus the self-sustaining nature of the vortex roll-ups in the flow, depends strongly on the events in the nearby flow field. This observation led to a simplified flow model which gives insight into the process involved in the roll-up reinitiation process. The calculations also show that spreading of the mixing layer through vortex merging depends strongly on the the pressure field induced by the downstream events. The dynamic fluid motions downstream generate subharmonic perturbations which induce vortex interactions and subsequent vortex mergings upstream. Spatial coherence between the first few roll-ups is observed, thus indicating the presence of an underlying degree of self-organization in the unforced mixing layer. The results are evidence of important feedback between the fluid accelerations in the vortex mergings downstream and the unstable vortex roll-ups occuring upstream.

I - INTRODUCTION

The discovery of large, spatially-evolving, spanwise coherent structures in turbulent flows (Brown & Roshko 1974; Winant & Browand 1974) has led to a shift in the emphasis from the statistical to the deterministic aspects of transitional and turbulent flows. Shear flows generated by splitter-plate partitions (e.g., Brown & Roshko 1974; Browand & Weidman 1976; Ho & Huerre 1984) exhibit most of the intricacies associated with the transition to turbulence and also have are important for many practical

* Berkeley Research Associates, P.O.Box 852, Springfield, VA 22152

applications. Large scale vortical structures are intrinsic features of the mixing layers at high enough Reynolds numbers, and their sequential mergings are the primary mechanism for spreading the layer in the downstream direction (Winant & Browand 1974). Numerical simulations have indicated that the vortex interactions leading to the pairing process depend in a crucial way on the presence in the shear layer of an appropriately phased subharmonic of the most unstable linear mode (Patnaik et al. 1976; Riley & Metcalfe 1980). In addition, experiments (Ho & Huang 1982) have shown that the spreading rate of the mixing layers can be changed considerably by perturbing the flow at a very low forcing level near a subharmonic. In particular, Ho & Huang (1982) proposed a feedback mechanism to explain the observed evolution of the structures and the pattern of streamwise vortex merging locations.

There are three processes that can influence the reinitiation of vortex roll-up in the shear flow: 1) perturbations in the flow coming from upstream, 2) perturbations in the flow coming from the splitter plate or nozzle (e.g., due to boundary layers, wakes or small recirculation zones), and 3) perturbations in the flow coming from downstream. In laboratory experiments this last mechanism is difficult to study quantitatively because turbulence in the inflow and boundary layers off the splitter plate or nozzle cannot be eliminated. In numerical simulations of the spatially-evolving mixing layer, various algorithms for the inflow and outflow boundary conditions potentially modify or even mask this downstream feedback. The advantage of numerical simulations with physically realistic boundary conditions is that inflow perturbations can be eliminated. Then the downstream feedback remains the only process that can reinitiate the instability.

The idea of a feedback mechanism through which the downstream events influence the upstream flow was first proposed by Dimotakis & Brown (1976) to explain unusually long autocorrelation times of the streamwise velocity fluctuations observed in their planar shear flow experiments. Subsequently, Laufer & Monkewitz (1980) observed that the unstable shear layer, close to the nozzle of a jet, was modulated with a low frequency corresponding to the passage frequency of the large scale structures at the end of the potential core. This suggested the presence of such a feedback mechanism for flows in a circular jet.

The feedback mechanism discussed by Ho & Huang (1982) was very similar in principle to that in the inpinging jet problem as studied by Ho & Nosseir (1981), and

discussed by Laufer (1981) for free and forced jets. They conjectured that the cross-stream perturbations required for the pairing process may be provided by the pressure field induced by the downstream (already paired) vortices. The induced field perturbs the thin shear layer near the trailing edge of the splitter plate or the nozzle. This generates new Kelvin-Helmoltz waves from which new coherent structures develop and then subsequently merge. The feedback loop from the trailing edge to the merging location involves the unstable vortex roll-ups propagating downstream and pressure waves from these changing, predominantly rotational flows, propagating upstream.

Numerical studies of coherent structures in shear layers have used spectral methods, vortex dynamics, and finite difference techniques. Many simulations have been performed to study temporally-evolving shear layers, which are defined by periodic boundary conditions at the inflow and outflow (e.g., Patnaik et al. 1976; Riley & Metcalfe 1980). However, simulations of the evolution of spatially-evolving flows similar to those seen in laboratory experiments require the use of realistic inflow and outflow boundary conditions. Spatially-evolving two-dimensional shear layers have been simulated using vortex techniques (Ashurst 1979; Aref & Siggia 1980; Mansour & Barr 1985; Ng & Ghoniem 1985) and finite-difference methods (Davis & Moore 1985; Grinstein et al. 1985, 1986).

This paper discusses basic physical mechanisms for the reinitiation of vortex roll-up and feedback phenomena in planar shear flows. We first describe two-dimensional finite-difference numerical simulations of the evolution of the Kelvin-Helmholtz instability of a spatially-evolving, unforced planar shear layer. This provides a picture of the large scale flow dynamics which is examined through isovorticity contours. The calculated flow patterns show that the reinitiation process depends strongly on the events in the nearby flow field. This led us to formulate a simplified analytical flow model which gives insight into the mechanisms involved in the reinitiation process. In addition, the flow visualization shows the effect of the long range pressure field induced by the downstream fluid accelerations, which results in spatial coherence between the first few roll-ups. The results give direct evidence of a feedback mechanism in unforced mixing layers.

II – NUMERICAL SIMULATION OF AN UNFORCED MIXING LAYER

A. The Numerical Model

The numerical model solves the time-dependent, inviscid conservation equations for mass, momentum and energy in two dimensions. It uses the Flux-Corrected Transport (FCT) algorithm (Boris & Book 1976) and timestep-splitting techniques. This is an explicit, conservative, monotone, fourth-order finite-difference algorithm, with no artificial viscosity for stabilization. The algorithm is designed to incorporate variable and moving grids. Inflow and outflow boundary conditions have been developed and tested for these compressible multidimensional FCT calculations (Boris et al. 1985; Grinstein et al. 1985). The inflow boundary conditions allow pressure fluctuations from the downstream vortices to influence the inflowing material. Such a physically realistic inflow boundary condition avoids the need to constantly drive the instability, allowing for its natural evolution in the calculation. The pressure values used at the outflow guard cells are defined by interpolating between the pressure values at the outflow boundary cells and ambient (background) pressure. This boundary condition imposes a slow relaxation of the pressure towards the known ambient value. Applications and tests of the model for the simulation of two-dimensional mixing layers have been described in our previous studies (Grinstein et al. 1985, 1986). The calculations are inviscid and no subgrid turbulence model beyond the natural FCT filtering has been included. The simulations are expected to be adequate for describing large scale features of high Reynolds number flows.

Figure 1 shows the two-dimensional flow configuration. Two coflowing laminar streams of air are initially separated by a thin splitter plate and then enter a long chamber confined between two walls. The values in the Y direction, indicated between parenthesis in the figure, indicate alternative locations of the walls. This variation in chamber width was used to evaluate the influence of the separation between the walls on the phenomena being studied. The trailing edge of the splitter plate is located on the centerline at approximately $X = 2.0$ cm, as shown in the figure.

The simulations are initialized by assuming that both gas streams have the same initial pressure (1 atm) and temperature (298 K), and have different (fast and slow) free stream velocities V_f and V_s. The free stream velocity ratio,

$$R = \frac{V_f - V_s}{V_f + V_s} ,$$

was varied in the range 0.5 – 1.0 and the mean Mach number was in the range 0.17 – 0.40. The flows were subsonic and virtually incompressible. Since the FCT algorithm used to describe the convection is fully compressible, sound waves are well resolved. This means that acoustic delay times for pressure waves are properly included.

The computational grid, discussed in Grinstein et al. (1985), was set up initially and held fixed in time. Typical finite difference grids used 120 – 240 cells in the cross-stream direction and 300 – 600 in the streamwise direction. The cells are closely spaced in the cross-stream direction (Y) near the shear layer, and they become more widely spaced as the distance from the shear layer increases. The grid in the streamwise direction (X) was either uniform (600 cells), or increasingly coarser (300 cells) as the distance downstream from the splitter plate increases. A non-uniform grid takes advantage of the fact that the expected vortical structures merge and grow so that fewer cells are necessary to keep the same relative resolution to that obtained near the splitter plate. The results presented in this paper were obtained on grids of 300 x 120 and 300 x 160 cells, for the smaller and larger computational domains, respectively. The cell sizes were 0.012 cm $\leq \Delta Y \leq$ 0.196 cm, and 0.030 cm $\leq \Delta X \leq$ 0.231 cm. The computed results were consistent with those obtained in more resolved calculations with 300 x 240 and 600 x 120 cells. These higher resolution tests were used to check the convergence of the calculations. The timesteps were chosen to satisfy the Courant condition.

B. Reinitiation of the Unstable Vortex Roll-ups

The initial shear layer is Kelvin-Helmholtz unstable. The small perturbation initiating the instabilities and subsequent vortex roll-ups occurs at the very beginning of the calculations. This initiation is analogous to using a delta function perturbation at the center in a periodic simulation of two equal and opposite streams. The perturbation generates small cross-stream pressure gradients at the shear layer, near the edge of the splitter plate. These perturbations induce the transverse flows triggering the instability initially.

For a two-dimensional inviscid, incompressible flow, the transport of vorticity, ω, is described by the Helmholtz equation

$$\frac{d\omega}{dt} = 0 \, . \tag{1}$$

Because the vorticity is fixed to the fluid, the lines of constant vorticity act as fluid markers much like the dye used in the laboratory experiments. Since the flows consid-

ered here are inviscid and virtually incompressible, the time evolution of the flow can be visualized through sequences of isovorticity contours.

Figure 2 shows the initial development of the flow for the case in which $R = 0.82$. As the instability grows, the vorticity along the centerline is distorted from the initially uniform distribution and becomes concentrated locally in two regions. The layer breaks and forms a pair of vortices behind the trailing edge.

A consistency test between the simulations and linear inviscid stability theory can be made by evaluating the Strouhal number for the initial instability region. Linear theory predicts that the Strouhal number,

$$St = \frac{f\theta}{\tilde{u}} \tag{2}$$

of the most amplified frequency, i.e., the natural frequency of the mixing layer, is $St_n = 0.032$, changing only within 5 % between $R = 0$ and $R = 1$ (Ho & Huerre 1984). To calculate the nondimensional flow parameter St, the frequency f is scaled by θ, the initial thickness of the shear layer, and the mean free stream velocity $\tilde{u}$ given by

$$\tilde{u} = \frac{(V_s + V_f)}{2} .$$

We define a Strouhal number relative to an initial shear layer thickness $\theta_o = \Delta Y$, where $\Delta Y = 0.012$ cm is the length of a cross-stream cell at the centerline. This is appropriate for our initial step function velocity profile. Then, we can estimate the wavelength of the most amplified mode from the approximate size of the vortical structures in the bottom panel in figure 2. We obtain $St \approx 0.040$, in reasonable agreement with St_n.

Figure 3 shows the subsequent evolution of the instabilities. The first two panels show the beginning of the build-up of new centers of vorticity at the ends of the undisturbed layers, near the developed vortices. In the last two panels the vorticity layer breaks up further and new vortices are shed. The new vortices roll around neighboring vortices and, as before, new vorticity centers build up at the ends of the undisturbed vortex strips. Figure 4 shows vortex mergings and new vortex generation. Thus we see a pattern evolving in which finite-sized vortices develop at the ends of the undisturbed layers, grow, separate, and then merge with the neighboring structures.

Figures 5 and 6 show streamlines of the velocity field in a reference frame moving with the fluid at velocity $\tilde{u}$, corresponding to $R = 1$ and $R = 0.82$, respectively.

These figures again show the reinitiation pattern through a different flow visualization. Results for two different velocity ratios are shown to emphasize that the reinitiation pattern is independent of R. The top panel in figure 5 (for $R = 1$, i.e., for $V_s = 0$) shows the reinitiation of the vortex roll-up (left side) after the formation of the initial vortex pair (right side). In time, the new vortex grows and begins its pairing with the neighboring vortex. In the last panel reinitiation begins again at approximately the same location as it did earlier in the top panel. Figure 6, the later evolution of the flow for $R = 0.82$, shows the first vortex merging. Although the reinitiation pattern for smaller R is the same, the pattern is shifted downstream. This is expected, since the growth rates of the Kelvin-Helmholtz instability increase with R (see, e.g., Ho & Huerre 1984). The process is shown at later times in figure 7.

The calculations were initiated with a step-function velocity profile along the whole streamwise extent of the computational domain. This procedure initially masks the spatially-developing nature of the flow. Figures 2 – 4 are reminiscent of a temporally-developing mixing layer with upstream-downstream symmetry. Essentially the same reinitiation pattern is observed on both sides of the leading pair of vortices. This suggests that the reinitiation of the vortex roll-up is mainly due to the effects in the nearby flow field, i.e., to the interaction of the edge of the vorticity layer with the immediately neighboring vortex. There is a strongly nonlinear interaction between the edge of the vorticity layer and the vortex. The result of this interaction is that a vorticity clump appears in the layer, near the edge, while the layer becomes progressively thinner upstream of the edge. This eventually leads to vortex shedding.

C. Feedback Phenomena

At later times the flow visualization exhibits the same basic pattern of reinitiation near the splitter plate. New vortices are formed farther and farther from the leading roll-up, which has moved downstream. One effect of the vortex pairings is the generation of subharmonics of the mode that was originally most unstable. The panels in figures 8 and 9 show that the region upstream of the pairing is modulated with a wavelength of the order of the dimensions of the resulting merged structure. The modulation perturbs the vortices by displacing them in the cross-stream direction. This induces pairings when the perturbation has the proper phase relationship with the vortices, as indicated by previous numerical simulations (Riley & Metcalfe, 1980).

It is important to test the calculations to ensure that the results are inherent in the equations and do not arise from the numerical solution procedures. For example, a stringent test of convergence is to change the computational cell spacing and do the calculations again. We performed additional calculations using 600 $\times$ 120 computational cells in a mesh uniformly spaced in the streamwise direction. Since these calculations showed no significant differences from our original ones, we felt that the original calculations had converged and the results are independent of the gridding in the streamwise direction. In addition, we note that the outflow boundary condition extrapolation used here (Grinstein et al. 1985) depends explicitly on the size of the last cell in the X direction, which differs by a factor of eight from the smaller to the larger meshes used. Thus this test also ensured the basic insensitivity of the results to changes in both the location at which the outflow boundary condition is implemented and its particular expression. The independence of the results relative to the gridding in the cross-stream direction was checked by comparing calculations on a 300 $\times$ 240 computational grid to the standard 300 $\times$ 120 results.

The panels in figure 9 show a steady pattern near the trailing edge. Shedding and initial pairing of vortices occur regularly at approximately fixed distances from the trailing edge. The largest structure on the right goes through succesive pairing processes, by which it grows as it moves further from the splitter plate. Mergings which involve more than two vortices become possible upstream as the wavelength of the modulation increases. The third panel, at 0.69 ms, shows a wavy shear layer inducing an interaction among three vortices. This three-vortex interaction is shown with greater temporal resolution in figure 10. The vortices are drawn together in one part of the period of the modulated shear layer, as they coalesce into larger structures. After the three-vortex interaction, mergings with the leading vortex structure are delayed because the smaller vortices have merged among themselves. Then the large structure appears somewhat decoupled from the upstream shear layer. Figure 11, taken from a calculation for $R = 0.67$, shows two such mergings where three vortices roll up into larger ones as a result of these collective interactions involving more than two vortices. The flow visualizations are reminiscent of the multiple-vortex mergings observed by Ho & Huang (1982) in their experimental studies of forced mixing layers. As the larger structure on the right side of the panels reaches the upper wall of the chamber (e.g., at $t \sim 0.9$ ms, for $R = 0.82$), further development of the downstream flow is inhibited.

As mentioned in the introduction, there are several indications of feedback phenomena in the experimental studies of mixing layers and jets. In the case of forced shear layers, a feedback loop has been postulated to exist between a wave convecting the instabilities downstream and an acoustic wave propagating pressure pulses upstream (Ho 1981; Laufer 1981). Laufer & Monkewitz (1980) have shown that the locations of the mergings observed in the forced-jet experiments of Kibens (1980) satisfy the feedback equation (Ho & Nosseir 1981)

$$\frac{X_i}{\bar{u}} + \frac{X_i}{a} = \frac{N}{f_i} \,, \tag{3}$$

where X_i, a, f_i, and N, are, respectively, the location of the i-th merging, the acoustic speed, the merging frequency, and an integer which is fixed for a given feedback loop. The merging frequency is reduced by a factor of two at each pairing. In general, $f_i = f_o/2^i$, where f_o is the frequency of the most unstable mode at the initiation of the shear layer. Equation 3 is obtained by requiring that the phase difference between the two wavetrains at any point be $2\pi N$, and by assuming that the downstream phase velocity is constant along the paths. The vortex merging induces a perturbation at the trailing edge which is then convected back to the merging location. The merging location is such that this process takes place over an average time that is an integer multiple of the pairing period. This average time is approximately the sum of the acoustical upstream propagation time plus the convection time downstream. The feedback model predicts that the distance between the virtual origin of the flow where the roll-ups initiate and the first pairing event is equal to the distance between the first and second pairings.

In the case of free, unforced mixing layers, the pure tone resonances seen in the forced experiments are unlikely. Fluctuations, such as perturbations in the inflowing streams arising from turbulence or boundary layers, are always present. These fluctuations introduce changes in parts of the feedback loop, and these changes cause time variations in the magnitude of X_i. Our simulations of idealized, unforced mixing layers have minimal externally introduced fluctuations, and do, however, show evidence of spatial coherence. A first pairing occurs regularly at an approximately fixed location (X_1) from the trailing edge, as can be seen in figure 9. In addition, the distance ($X_2 - X_1$) between the second and the first pairing location is approximately equal to the distance ($X_1 - X_o$) between the first pairing location and the location where the roll-ups initiate (see, e.g., the bottom panels of figures 9 or 10).

We have also tested the effect of the separation between the side walls on the development of the flow. A characteristic time in the chamber is the sonic transit time between the walls or, equivalently, the time for a pressure pulse to propagate from the centerline to a wall, reflect, and return. For the cases discussed above, in which the smallest separation betweeen the walls was used (see figure 1), this characteristic time was $T_1 \approx 0.18\ ms$. For the larger domain, the characteristic time was $T_2 \approx 0.40\ ms$.

Figure 12 shows a sequence of panels with results from a calculation on the larger domain for $R = 0.82$. The panels are for the same times as those in figure 8 for the smaller domain. The top panels of the figures are essentially identical, indicating that the basic process of reinitiation and evolution of the instabilities at the earlier stages is relatively independent of the separation between the walls. This should be true as long as the time elapsed since the initiation of the instabilities (at $t = t_o \approx 0.1$ ms) is smaller than the characteristic time T_i. At later stages, shear layer modulations upstream of the larger structure can be observed in both figures, with approximately the same wavelength, but different phases. Because of these phase differences, the locations of the induced vortex pairings are different. The phase differences we observe between the larger and smaller domain calculations result from the differences in the pressure pulses which have propagated upstream from the large structure. In the large domain calculation, these pulses have not yet reflected from the walls, since $(t - t_o) < T_2$. The later time, larger domain panels for $(t - t_o) > T_2$ show the same general flow features as seen in the case of the smaller domain. Figure 13 shows a sequence with three vortices merging. This sequence is very similar to that shown in the first four panels of figure 10.

III - A MODEL FOR THE REINITIATION OF VORTEX ROLL-UP

A. The Model

As discussed in Section II, the simulations show a pattern indicating that the reinitiation and the self-sustaining nature of unstable vortex roll-up depend strongly on the nearby flow field. The results suggest the simplified flow model below, which gives insight into the mechanisms involved in the reinitiation process.

As an initial flow, consider an unperturbed two-dimensional vortex sheet extending from $-\infty$ to $+\infty$ in X at $Y = 0$. This sheet separates two equal and opposite semi-infinite streams in a frame of reference moving with the mean free-stream velocity $\tilde{u}$. As in our simulations, variations in the Z direction are neglected but the center of our coordinate system moves downstream tracking the element of fluid which was initially at the trailing edge of the splitter plate. The unperturbed velocity field caused by the thin shear layer can be assumed to have a profile described by the hyperbolic tangent function (see, e.g., Ho & Huerre 1984),

$$\mathbf{u}^{(o)} = -U_o \tanh\left(\frac{2Y}{\theta_o}\right) \hat{\mathbf{i}}\,, \tag{4}$$

where

$$U_o = \frac{(V_f - V_s)}{2} = \tilde{u}R\,.$$

This thin vortex layer has a thickness of θ_o, which we take to be very small in what follows. There is a corresponding vorticity profile which varies as $[\,\cosh(2Y/\theta_o)\,]^{-2}\,\hat{\mathbf{k}}$. As θ_o will be small, the quantity of importance is the integrated vorticity across the layer, i.e., $2U_o$, the circulation per unit length across the layer. Figure 2 shows that the initial roll-up of the shear layer appears as a growing finite width strip. The vorticity in this strip clumps into two finite-strength spanwise vortices placed symmetrically above and below the center of the initially uniform shear layer. Figure 14 shows this situation schematically. In our analytical idealization the circulation which was spread through the gap, from $-X_o$ to X_o, forms two vortices. Each vortex has circulation $2U_oX_o$, located at distance r_o and angles γ_o and $\gamma_o + \pi$ relative to the origin of our moving coordinate system. The vortices grow by increasing the width of the gap and incorporating the vorticity which must be conserved as the gap widens.

The total velocity field is thus composed of three terms,

$$\mathbf{u} = \mathbf{u}^{(o)} + \mathbf{u}^{(s)} + \mathbf{u}^{(v)}, \tag{7}$$

where $\mathbf{u}^{(o)}$ is the velocity of the initial shear flow given by equation 4, $\mathbf{u}^{(s)}$ is the velocity of the flow associated with the missing strip of positive circulation between $-X_o$ and X_o, with circulation per unit length $-2U_oX_o$, and the third term, $\mathbf{u}^{(v)}$, is the flow associated with the two vortices. Since circulation is conserved, the overall flow far away from the disturbed region about the origin reduces to $\pm U_o\,\hat{\mathbf{i}}$ as required.

The expression for the velocity due to a specified vorticity distribution ω is given by (Batchelor 1981)

$$\mathbf{u}_\omega(\mathbf{x}) = -\frac{1}{4\pi}\int \frac{\mathbf{s}\times\omega(\mathbf{x}')}{s^3}d^3x', \tag{8}$$

where $\mathbf{s} = \mathbf{x} - \mathbf{x}'$. For the substracted strip we obtain

$$\mathbf{u}^{(s)} = \frac{U_o}{\pi}\int_{-X_o}^{X_o} \frac{[\,(X-X')\,\hat{\mathbf{i}} + Y\,\hat{\mathbf{j}}\,]\times\hat{\mathbf{k}}}{(X-X')^2+Y^2}dX'. \tag{10}$$

These integrals can be reduced to quadratures, giving expressions for $u_x^{(s)}$ and $u_y^{(s)}$

$$u_x^{(s)}(X,Y) = \frac{U_o}{\pi}\,\tan^{-1}\left[\,\frac{2X_oY}{Y^2+X^2-X_o^2}\,\right], \tag{11a}$$

$$u_y^{(s)}(X,Y) = \frac{U_o}{2\pi}\,\log\left[\,\frac{(X-X_o)^2+Y^2}{(X+X_o)^2+Y^2}\,\right]. \tag{11b}$$

Figure 15a shows streamlines of the flow determined by the sum $\mathbf{u}^{(os)} = \mathbf{u}^{(o)} + \mathbf{u}^{(s)}$. As seen in these consistent velocity fields, a gap in a vortex sheet widens as the edges of the vortex sheet roll-up.

Finally, we introduce the contribution to the velocity field due to the two vortices with centers at (X_v, Y_v) (upper vortex on the left) and $(-X_v, -Y_v)$ (lower vortex on the right). Here, $X_v = r_o\cos\gamma_o$ and $Y_v = r_o\sin\gamma_o$. Rather than using point vortices, we use a more realistic model with finite cores. This avoids introducing singularities in the model. The core radius ρ_o is chosen such that $\rho_o \ll r_o$. The expression for the contribution to $\mathbf{u}$ from these vortices is:

$$\mathbf{u}^{(v)}(X,Y) = \frac{X_oU_o}{\pi}\,(f_+\mathbf{r}_+ + f_-\mathbf{r}_-)\,, \tag{12}$$

where

$$\mathbf{r}_\pm = -(Y \pm Y_v)\,\hat{\mathbf{i}} + (X \pm X_v)\,\hat{\mathbf{j}}\,,$$

$$f_\pm = \begin{cases} 1/|\mathbf{r}_\pm|^2, & \text{if } |\mathbf{r}_\pm| > \rho_o\,, \\ 1/\rho_o^2, & \text{if } |\mathbf{r}_\pm| \le \rho_o\,. \end{cases}$$

Typical streamlines for the flow field due to the vortices are shown in figure 15b. The total velocity field is obtained by substituting the contributions from equations 4, 11 and 12, in equation 7.

The model parameters are U_o, ρ_o/X_o, γ_o, and the ratio r_o/X_o. The quantity U_o fixes the velocity scale in the reference frame moving with the vortex pair. Since we are interested in the flow field outside the cores of the vortices, the results presented below are actually independent of ρ_o/X_o, which we have chosen to be

$$\rho_o/X_o < 0.05 \ .$$

In addition, we can determine the relevant ranges of the other parameters from the simulations discussed in Section II. These are

$$160^\circ < \gamma_o < 175^\circ,$$

$$0.5 < \frac{r_o}{X_o} < 0.7 \ ,$$

and

$$\theta_o \approx 0.012 \text{ cm.}$$

A general trend observed in the simulations was for γ_o to approach 175° and r_o/X_o to approach 0.5, as R approaches 1.0. The limiting situation, $R = 1$, corresponds to having the lower stream emerging into a quiescent background, i.e., $V_s = 0$.

Figure 15c, shows streamlines for the total velocity field u given by equation 7. Comparing this to the top panel of figure 5, or the bottom panel of figure 2, we note the similarity in the flow patterns described by the velocity field u and the flow patterns in the numerical simulations.

B. Asymptotic Behavior of the Velocity and the Reinitiation of the Vortex Roll-up

We now derive an expression for the asymptotic behavior of the velocity close to the line of the undisturbed layer, $Y = 0$, for $X \ll -X_o$. This expression is important for understanding how upstream vortex roll-ups are reinitiated.

We can find the first order corrections to the streamwise velocity $u^{(o)}$, due to the redistribution of vorticity, from equations 11a and 12. Using the expansion

$$\arctan(\alpha) \approx \alpha + O(\alpha^3) \ ,$$

for $|\alpha| < 1$, with

$$\alpha = \frac{2X_oY}{(Y^2 + X^2 - X_o^2)^2}$$
$$\approx \frac{2X_oY}{X^2}\left[1 + O\left(\frac{Y^2 - X_o^2}{X^2}\right)\right],$$

for $|X| \gg |Y| + X_o$ we obtain,

$$u_x^{(s)}(X,Y) = \frac{2X_oU_o}{\pi}\frac{Y}{X^2} + O\left[\left(\frac{X_o}{X}\right)^4\right]. \qquad (13)$$

In the same limit, equation 12 gives

$$u_x^{(v)}(X,Y) = -\frac{2X_oU_o}{\pi}\frac{Y}{X^2} + \frac{4X_oU_oX_vY_v}{\pi}\frac{1}{X^3} + O\left[\left(\frac{X_o}{X}\right)^4\right]. \qquad (14)$$

The leading terms in equations 13 and 14 cancel, as we add $u_x^{(s)}$ and $u_x^{(v)}$ to the expression for $u_x^{(o)}$ given by equation 4. Hence, for the total streamwise velocity field, we have

$$u_x(X,Y) = -U_o\tanh\left(\frac{2Y}{\theta_o}\right) + \frac{4X_oU_oX_vY_v}{\pi X^3} + O\left[\left(\frac{X_o}{X}\right)^4\right]$$
$$= -U_o\tanh\left(\frac{2Y}{\theta_o}\right) + \frac{2U_o}{\pi}\left(\frac{r_o}{X_o}\right)^2\sin(2\gamma_o)\left(\frac{X_o}{X}\right)^3 + O\left[\left(\frac{X_o}{X}\right)^4\right]. \qquad (15)$$

In an analogous way, we can start from equation 11b and use the standard expansion (for $|\beta| < 1$)

$$\log\left(\frac{1+\beta}{1-\beta}\right) \sim 2\beta + \left(\frac{2}{3}\right)\beta^3 + O\ (\beta^5),$$

with

$$\beta = -\frac{2XX_o}{(X^2 + X_o^2 + Y^2)},$$

to obtain for $|X| \gg \max(|Y|; X_o)$

$$u_y^{(s)} = -\frac{2U_o}{\pi}\frac{X_o}{X}\left[1 - \left(\frac{Y}{X}\right)^2 - \frac{2}{3}\left(\frac{X_o}{X}\right)^2 + O\left[\left(\frac{X_o}{X}\right)^4\right]\right], \qquad (16a)$$

$$u_y^{(v)} = \frac{2U_o}{\pi}\frac{X_o}{X}\left[1 - \left(\frac{Y}{X}\right)^2 + \frac{X_v^2 - Y_v^2}{X^2} + \frac{8X_vY_vY}{X^3} + O\left[\left(\frac{X_o}{X}\right)^4\right]\right]. \qquad (16b)$$

Combining equations 16a and 16b, we find

$$u_y(X,Y) = \frac{2U_o}{\pi}\left(\frac{r_o}{X_o}\right)^2\left(\frac{X_o}{X}\right)^3\left[\cos(2\gamma_o) + \frac{2}{3}\left(\frac{X_o}{r_o}\right)^2 + 4\sin(2\gamma_o)\left(\frac{Y}{X_o}\right)\left(\frac{X_o}{X}\right)\right] + O\left[\left(\frac{X_o}{X}\right)^5\right] . \quad (17)$$

The negative vorticity strip and the pair of vortices add a small positive term to the streamwise velocity $u_x^{(o)}$. The lowest order correction term is independent of Y and is orders of magnitude smaller than $u_x^{(o)}$,. In the case of the cross-stream direction, a nonzero velocity component is now present. Equations 16 indicate that the asymptotic contributions from the strip and the pair have opposite signs. Thus the leading terms cancel. The remaining terms reinforce each other for $(3/4)\pi < \gamma_o < \pi$. The resulting negative component, that has contributions from both the vorticity strip and the vortex pair, tends to deflect the shear layer downwards and along the centerline for $X < -X_o$. This is due to the counter-clockwise circulation associated with the redistributed positive vorticity from the vortex sheet and becomes more important closer to the left vortex. The first correction term, of order $(X_o/X)^4$, depends on Y. This term either enhances or reduces the leading cross-stream velocity term, depending on whether Y is positive or negative, respectively.

C. Spatial Development of the Model.

We next examine the spatially developing problem, which is more closely connected to the simulations. We define a time-dependent velocity field which represents the velocity field during the period of growth of the initial vortex pair observed in the simulations. The pair of vortices is assumed to grow linearly, with fixed $(r_o/X_o), (\rho_o/X_o)$, and γ_o. This growth starts at time t_1 at $x_1 = x(t_1)$, the point in the shear layer at which a perturbation is first perceived in the flow visualizations. We follow the linear growth up to time t_2 when the center of the vortex pair is located at $x_2 = x(t_2)$ and the vorticity concentration at the edges of the undisturbed vorticity layers is noticeable. For example, for the case $R = 0.82$ discussed above, this growth period extends from some time before the first panel in figure 2 to approximately the time of the top panel in figure 3. The parameters of the model were chosen to fit the flow visualization at $t = t_2$. Linear growth was assumed for X_o between $X_o = X_o(t_1) = 0.0$ and $X_o = X_o(t_2)$.

Since the velocity of the flow is known, we can calculate the motion of any particle in the flow and visualize the spatial development of the flow model by streakline plots. Streaklines are produced by injecting a set of passive markers into the flowfield at fixed time intervals. They are injected at the location $x = x_s$, where the trailing edge of the splitter plate is assumed to be located, and for various values of Y above and below the centerline. The convection of the markers is studied by integrating the equations

$$\frac{d\mathbf{r}_i(t)}{dt} = \mathbf{u}_{lab}\Big(x_i(t), y_i(t), t\Big) \ , \tag{18a}$$

between t_1 and t_2 with the initial conditions

$$\mathbf{r}_i(t_1) = (x_s, y_i) \ , \tag{18b}$$

where the subscript refers to the i-th marker. The velocity $\mathbf{u}_{lab}$ at $\mathbf{r}$ in the laboratory reference frame is

$$\mathbf{u}_{lab}(x_i, y_i, t) = \tilde{u}\ \hat{\mathbf{i}} + \mathbf{u}(x_i - x_c(t), y_i, t) \ , \tag{19}$$

where $\mathbf{u}$ is given by equation 7, and $x_c(t) = \tilde{u}(t - t_1) + x_1$, is the location of the center of the vortex pair at time t.

Figure 16a shows a typical streakline plot corresponding to $R = 0.67$. Markers with the "−" and "+" symbol are injected above and below of the interface between the

two fluids, respectively. The solid rectangles indicate both the locations of x_c, the edges of the undisturbed layers, and the centers of the vortices. In this figure the vertical scale is uniformly stretched relative to the horizontal scale by a factor of two. Figure 16b shows isovorticity contours from the numerical simulations at $t = t_2$. A noticeable feature in the streaklines is the beginning of a roll-up in the flow near the edges of the vorticity layers. This can be understood in terms of the superposition of the velocity fields $\mathbf{u}^{(os)}$ and $\mathbf{u}^{(v)}$. Inspection of figures 15a and 15b shows that the cross-stream velocity components of the fields have opposite signs outside of the region between the vorticity layers, where they nearly cancel out (see, e.g., equation 16). Because the axis of the vortex pair is not parallel to the centerline, there is no cancellation of the cross-stream component in the neighborhood of the edges of the layers at $X = \pm X_o$. Approaching the neighborhood ahead of $X = -X_o$ from below, for example, $u_y^{(os)}$ is greater than $u_y^{(v)}$ up to a location above the centerline where they are equal. Above this, $u_y^{(v)}$ becomes more important.

In the limit when $\gamma_o \to 175°$ and $r_o/X_o \to 0.5$, as $R \to 1.0$ the coefficients of the leading term in equation 17 tend to increase. Thus as $R \to 1.0$, the downwards deflection of the shear layer for $X < -X_o$ due to the vortex pair is more pronounced and we can expect that the trend towards the reinitiation of unstable vortex roll-up becomes more important.

The results shown above for the breakup of the vorticity layer and subsequent vorticity roll-up and redistribution indicate that the model proposed contains the basic mechanism for triggering the reinitiation of the vortex roll-up present in the numerical simulations. The subsequent reinitiation of the vortex roll-up can be studied by extending the same procedure to model the further redistribution of the vorticity at one side of the initial vortex pair. The roll-ups near the edges of the vorticity layers correspond to the centers of vorticity observed in the numerical simulation (see, e.g.,figure 16a). The roll-ups are mainly the result of the interaction between the edges of the undisturbed vorticity layers and the immediately neighboring vortices. The thinning of the shear layer (especially from above) observed upstream from the vorticity center at the left edge in the simulations is consistent with the asymptotic expression obtained above for u_y given in equation 17. There is an induced velocity distribution associated with the accumulation of positive vorticity at the edge of the shear layer which tends to carry fluid around the edge in a counter-clockwise direction. This in turn increases

the concentration of vorticity at the edge. The fluid tends to move more slowly above the shear layer behind the edge, thus increasing the pressure in that region due to the relative accumulation of fluid. This will induce further thinning behind the edge and finally lead to new vortex shedding.

IV - SUMMARY AND CONCLUSIONS

We have presented results of the numerical simulation of the evolution of the Kelvin-Helmholtz instability and the subsequent formation of large scale structures in a planar, unforced, spatially-evolving mixing layer. The focus of this work has been the investigation of the feedback mechanisms involved in the reinitiation of the vortex roll-up behind of the trailing edge of the splitter plate. We have also studied the spreading of the mixing layer and the role of the feedback from downstream on the growth of the mixing layer.

The results indicate that the reinitiation of vortex roll-up, and hence the self-sustaining nature of the flow instabilities, depends strongly on the nearby flow field. An analytic flow model was presented and used to gain insight into the reinitiation process. The process of reinitiation starts with the concentration of vorticity at the end of the relatively undisturbed shear layer facing the most recently shed vortex structure. This is followed by vortex shedding. Newly generated vorticity is concentrated at the end of the mixing layer, and this self-induced vorticity redistribution leads to new vortex shedding. This is due mainly to the near field interaction of the newly shed vortex with the vorticity layer.

Spreading of the mixing layer through vortex merging depends on the pressure field induced by the downstream fluid accelerations. Because of the particular way in which our simulations were initiated, i.e., with a perturbation behind the trailing edge of the splitter plate, it was possible to observe the effect of the downstream events on the shear layer upstream. The shear layer is modulated with a wavelength of the order of the dimensions of the larger structure downstream. When this modulation is appropriately phased with the structures it produces vertical displacements of the vortices upstream, which induce mergings. Collective interactions involving mergings of three vortices, were observed when the wavelength of the modulation became sufficiently large. These collective interactions were similar to those observed in the experiments with (low frequency) forced mixing layers by Ho & Huang (1982).

A noticeable feature of the simulated flow was the temporal coherence between the first few roll-ups. This indicated an underlying degree of organization in the unforced flow. The coherence is due to feedback between the downstream events and the incoming fluid. Generally, this organization is not expected because of the unavoidable

presence of fluctuations in the flow field due to boundary layers in the splitter plate and turbulence in the inflowing streams. The present numerical simulations model the large scale features of an idealized system in which these potential perturbations have been minimized. In this way, we have been able to isolate the features of the flow dynamics in the absence of such fluctuations. This coherence in the flow pattern was also observed in numerical studies of unforced axisymmetric jets (Grinstein et al. 1986). In these calculations it was shown that the presence of a low level of random inflow perturbations weakened the organization of the shear layer, and hence there was less temporal coherence between the vortex structures.

ACKNOWLEDGEMENTS

This work was performed under Office of Naval Research project # RR024-03-01 and Naval Research Laboratory project # RR011-09-43.

REFERENCES

Ashurst, W.T. 1979, Numerical Simulation of Turbulent Mixing Layers Via Vortex Dynamics, *Turbulent Shear Flows I*, ed. F. Durst et al., pp. 402-13, Springer Verlag, New York.

Aref, H. & Siggia, E.D. 1980, Vortex Dynamics of the Two-Dimensional Turbulent Shear Layer, J. Fluid Mech. 100, 705.

Batchelor, G.K. 1981, *An Introduction to Fluid Dynamics*, p. 87, Cambridge University Press, Cambridge.

Boris, J.P. & Book, D.L. 1976, Solution of Continuity Equations by the Method of Flux Corrected Transport, *Methods in Computational Physics*, Vol. 16, pp. 85-129, Academic Press, New York.

Boris, J.P., Oran, E.S., Gardner, J.H., Grinstein, F.F. & Oswald, C.E. 1985, Direct Simulations of Spatially Evolving Compressible Turbulence – Techniques and Results, *Ninth International Conference on Numerical Methods in Fluid Dynamics*, ed. Soubbaramayer and J.P. Boujot, pp. 98-102, Springer Verlag, New York.

Browand, F.K. & Weidman, P.D. 1976, Large Scales in the Developing Mixing-Layer, J. Fluid Mech. 76, 127.

Brown, G. & Roshko, A. 1974, On Density Effects and Large Structure in Turbulent Mixing Layers, J. Fluid Mech. 64, 775.

Davis, R.W. & Moore, E.F. 1985, A Numerical Study of Vortex Merging in Mixing Layers, Phys. Fluids 28, 1626.

Dimotakis, P.E. & Brown, G.L. 1976, The Mixing Layer at High Reynolds Number: Large Structure Dynamics and Entrainment, J. Fluid Mech. 78, 535.

Grinstein, F.F., Oran, E.S. & Boris, J.P. 1985, Numerical Simulations of Asymmetric Mixing in Planar Shear Flows, NRL Memo. Report 5621, August 1985; to appear in J. Fluid Mech., 1986.

Grinstein, F.F., Oran, E.S. & Boris, J.P. 1986, Direct Numerical Simulation of Axisymmetric Jets, AIAA paper 86-0039, Reno.

Ho, C.M. 1981, Local and Global Dynamics of Free Shear Layers, *Numerical and Physical Aspects of Aerodynamic Flows*, pp. 521-533, ed. by T. Cebeci, Springer Verlag, 1981.

Ho, C.M. & Huang, L.S. 1982, Subharmonics and Vortex Merging in Mixing Layers, J. Fluid Mech. 119, 443.

Ho, C.M., & Huerre, P. 1984, Perturbed Free Shear Layers, Ann. Rev. Fluid Mech. 16, 365.

Ho, C.M. & Nosseir, N.S. 1981, Dynamics of an impinging jet. Part 1. The Feedback Phenomenon, J. Fluid Mech. 105, 119.

Kibens, V. 1980, Discrete Noise Spectrum Generated by an acoustically Excited Jet, AIAA J., 18, 434.

Laufer, J. & Monkewitz, P.A. 1980, On turbulent jet flows : A new perspective, AIAA paper 80-0962, Hartford.

Laufer, J. 1981, Instability and Turbulence in Jets, *Transition and Turbulence*, pp. 63 - 76, ed. by R.E. Meyer, Academic Press, New York.

Mansour, N.N. & Barr, P.K. 1985, Simulation of a Turbulent Mixing Layer, *Proceedings of the Fifth Symposium on Turbulent Shear Flows*, Ithaca, pp. 3.33 - 3.44.

Ng, K.K. & Ghoniem, A.F. 1985, Numerical Simulation of a Confined Shear Layer, *Proceedings of the Tenth International Colloquium on Dynamics of Explosions and Reactive Systems*, Berkeley.

Riley, J.J. & Metcalfe, R.W. 1980, Direct Numerical Simulation of a Perturbed Turbulent Mixing Layer, AIAA paper 80-0274, Pasadena.

Patnaik, P.C., Sherman, F.S. & Corcos, G.M. 1976, A numerical simulation of Kelvin-Helmholtz waves of finite amplitude, J. Fluid Mech. 73, 215.

Winant, C.D. & Browand, F.K. 1974, Vortex Pairing: The Mechanism of Turbulent Mixing Layer Growth at Moderate Reynolds Number, J. Fluid Mech. 63, 237.

FIGURE CAPTIONS

Figure 1 Schematic diagram of the flow configuration showing the initial and boundary conditions for the domains used in the splitter-plate simulation. The values between parenthesis correspond to the larger domain, used to study the effect of the separation between the walls.

Figure 2 Sequence of isovorticity contours for the initial stages in the development of the Kelvin-Helmholtz unstable flow for the splitter plate problem for calculations in the smaller domain in the case $R = 0.82$. The trailing edge of the splitter plate is located at $X = 2.03$. The contour levels are equally spaced, with an interval of $1.0 \times 10^4\ s^{-1}$. The vorticity at the outermost contour is $-1.0 \times 10^4\ s^{-1}$, decreasing to a minimum of $-3.0 \times 10^5\ s^{-1}$ near the tip of the splitter plate.

Figure 3 Isovorticity contours, as in figure 2, for later stages in the development of the flow.

Figure 4 Isovorticity contours, as in figure 2, for later stages in the development of the flow.

Figure 5 Streamlines of the velocity field in a reference frame moving with the mean free velocity, for the case $R = 1.0$.

Figure 6 Streamlines of the velocity field, as in figure 5, at later stages, for the case $R = 0.82$.

Figure 7 Isovorticity contours, as in figure 2, for later stages in the development of the flow.

Figure 8 Isovorticity contours, as in figure 2, for later stages in the development of the flow. Contour levels are equally spaced, with an interval of $6.0 \times 10^3\ s^{-1}$. The vorticity at the outermost contour is $-2.0 \times 10^3\ s^{-1}$, decreasing to a minimum of $-9.8 \times 10^4\ s^{-1}$ near the tip of the splitter plate.

Figure 9 Isovorticity contours, as in figure 8, for later stages in the development of the flow. X_o, X_1, and X_2 indicate, respectively, the approximate virtual origin of the flow (where the roll-ups initiate), and the first and second merging locations.

Figure 10 Isovorticity contours, as in figure 9, with a greater temporal resolution

Figure 11 Isovorticity contours, showing three-vortex mergings, as in figure 9, for the case $R = 0.67$. Contour levels are equally spaced, with an interval of $6.0 \times 10^3\ s^{-1}$.

The vorticity at the outermost contour is -2.0×10^3 s^{-1}, decreasing to a minimum of -9.8×10^4 s^{-1} near the tip of the splitter plate.

Figure 12 Isovorticity contours, as in figure 8, for the larger domain.

Figure 13 Isovorticity contours, as in figure 12, at later times.

Figure 14 Schematic diagram of the flow configuration for the definition of the vortex pair model.

Figure 15 Typical streamlines for the velocity fields in the vortex pair model, a) $\mathbf{u}^{(os)}$, b) $\mathbf{u}^{(v)}$, c) $\mathbf{u}$.

Figure 16 Spatial-development of the flow, for $R = 0.67$. The fixed parameters of the model are $r_o/X_o = 0.62$, $\rho_o/X_o = 0.02$, and $\gamma_o = 165°$. a) Calculated streaklines using the vortex pair model; b) isovorticity contours from the numerical simulations.

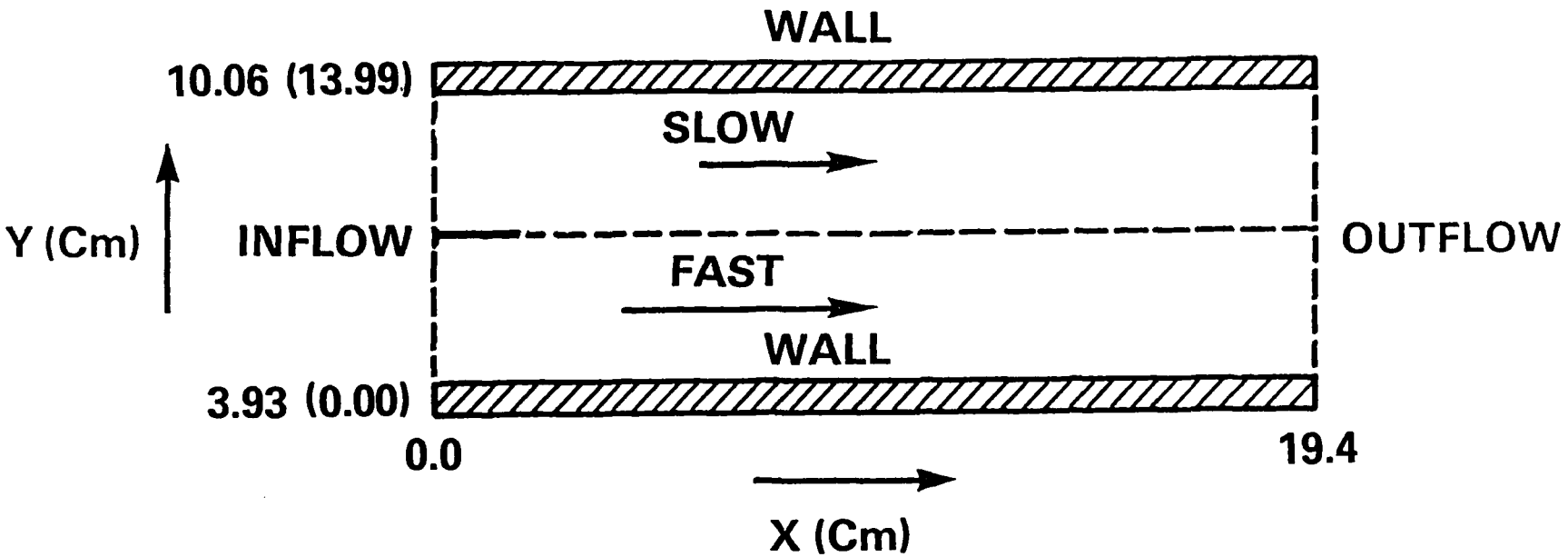

WALL
10.06 (13.99)
SLOW
Y (Cm)
INFLOW
OUTFLOW
FAST
WALL
3.93 (0.00)
0.0
19.4
X (Cm)

Figure 1

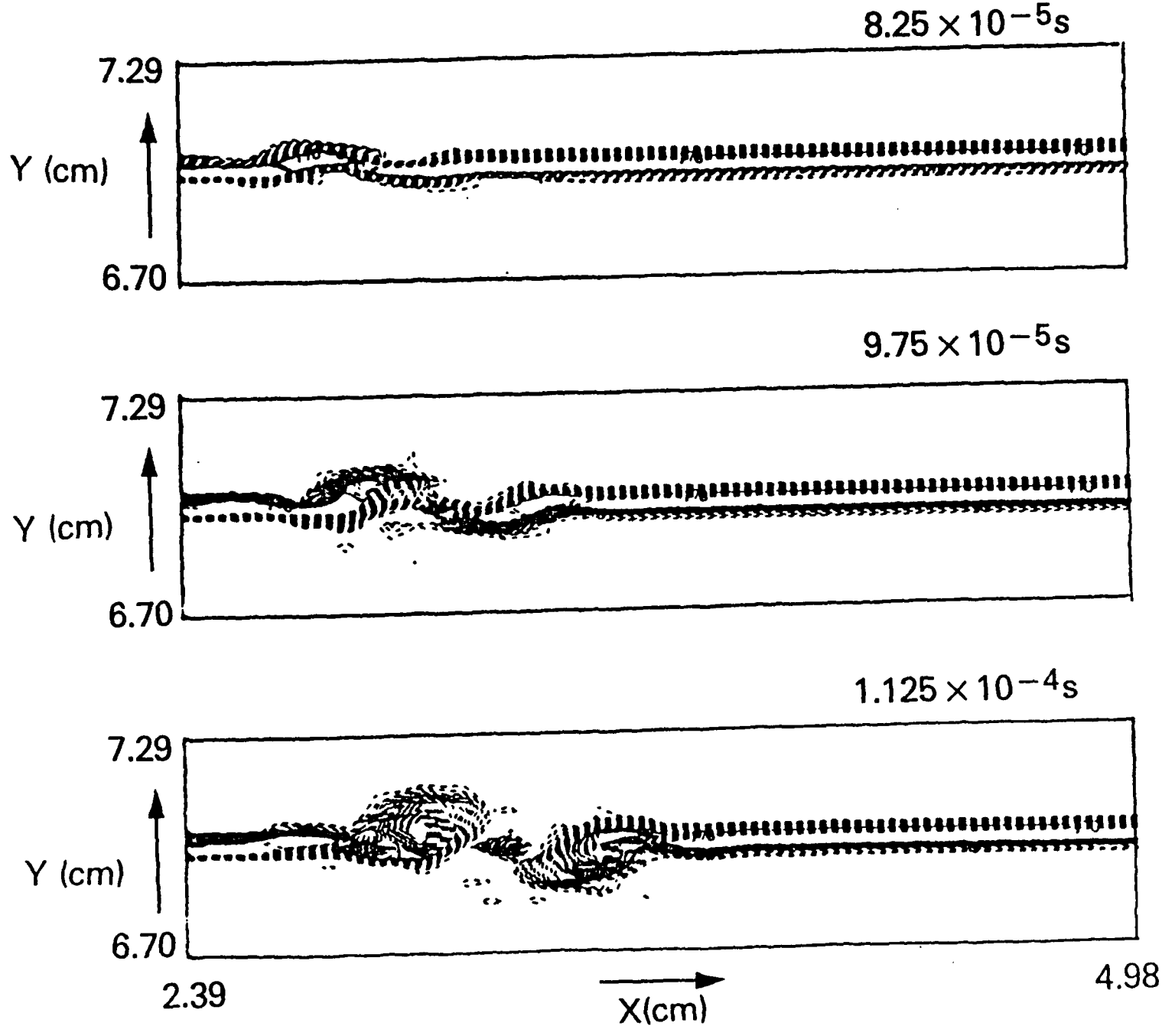

8.25 × 10⁻⁵s
7.29
Y (cm)
6.70
9.75 × 10⁻⁵s
7.29
Y (cm)
6.70
1.125 × 10⁻⁴s
7.29
Y (cm)
6.70
2.39
X(cm)
4.98

Figure 2

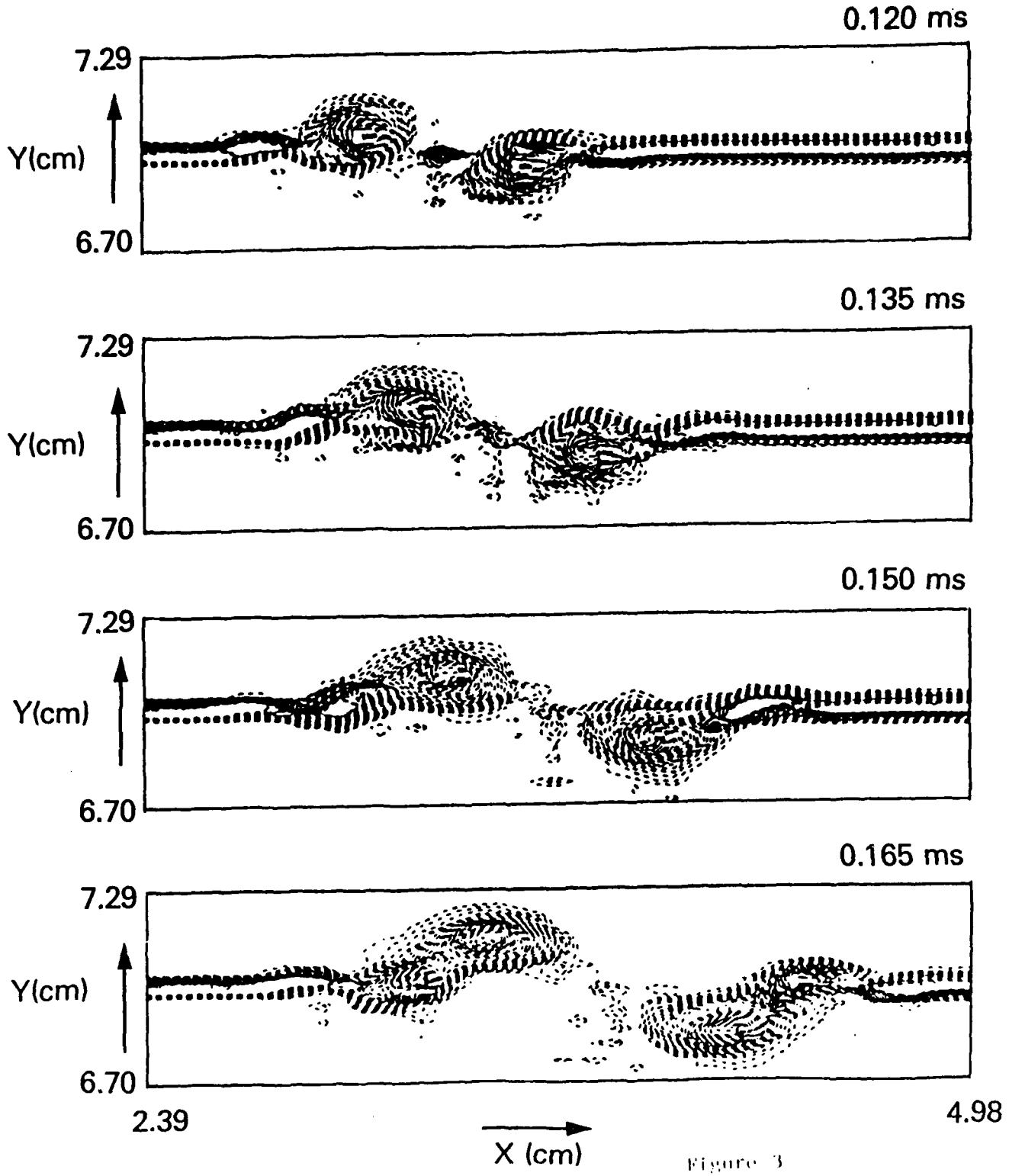

0.120 ms
7.29
Y(cm)
6.70
0.135 ms
7.29
Y(cm)
6.70
0.150 ms
7.29
Y(cm)
6.70
0.165 ms
7.29
Y(cm)
6.70
2.39
4.98
X (cm)

Figure 3

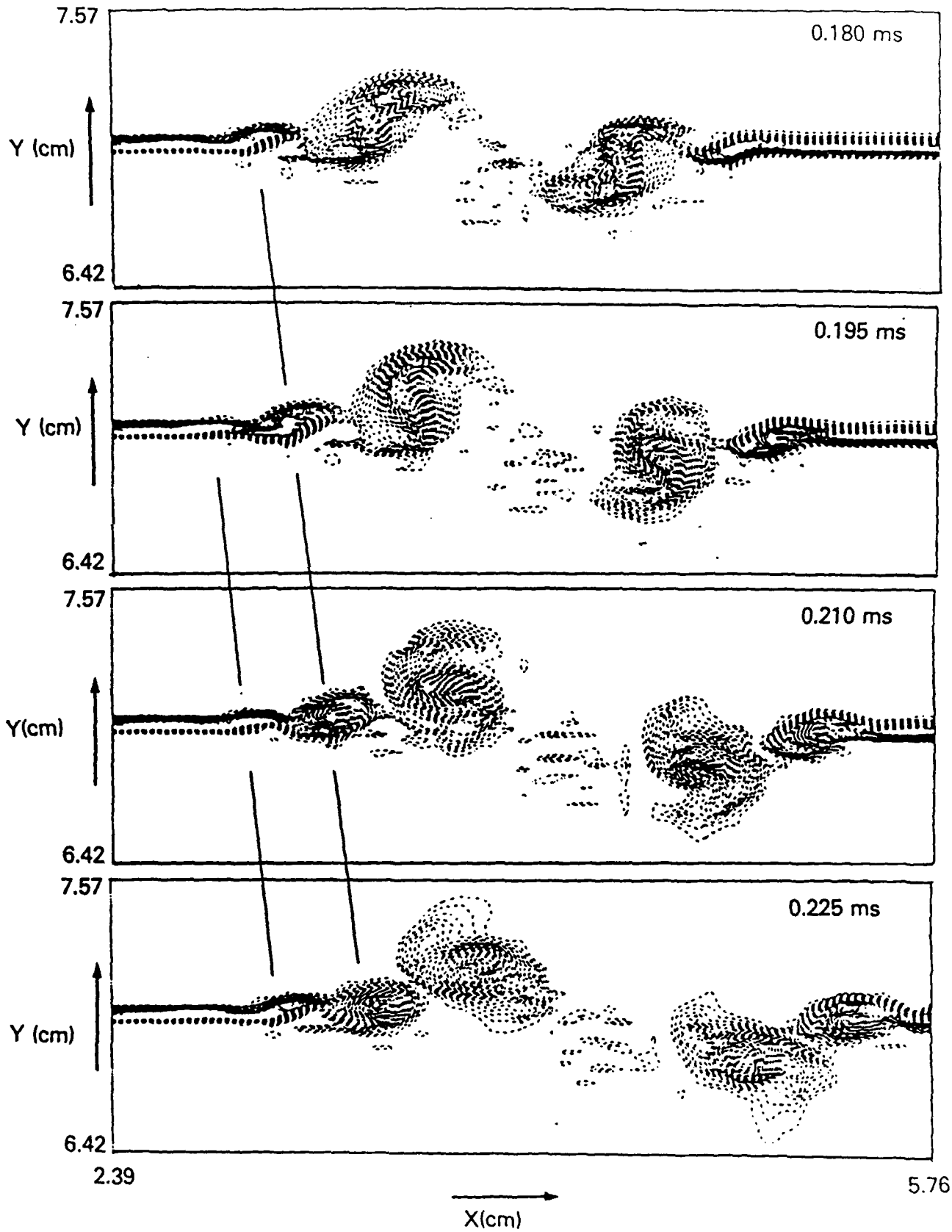

7.57
0.180 ms
Y (cm)
6.42
7.57
0.195 ms
Y (cm)
6.42
7.57
0.210 ms
Y(cm)
6.42
7.57
0.225 ms
Y (cm)
6.42
2.39
5.76
X(cm)

Figure 4

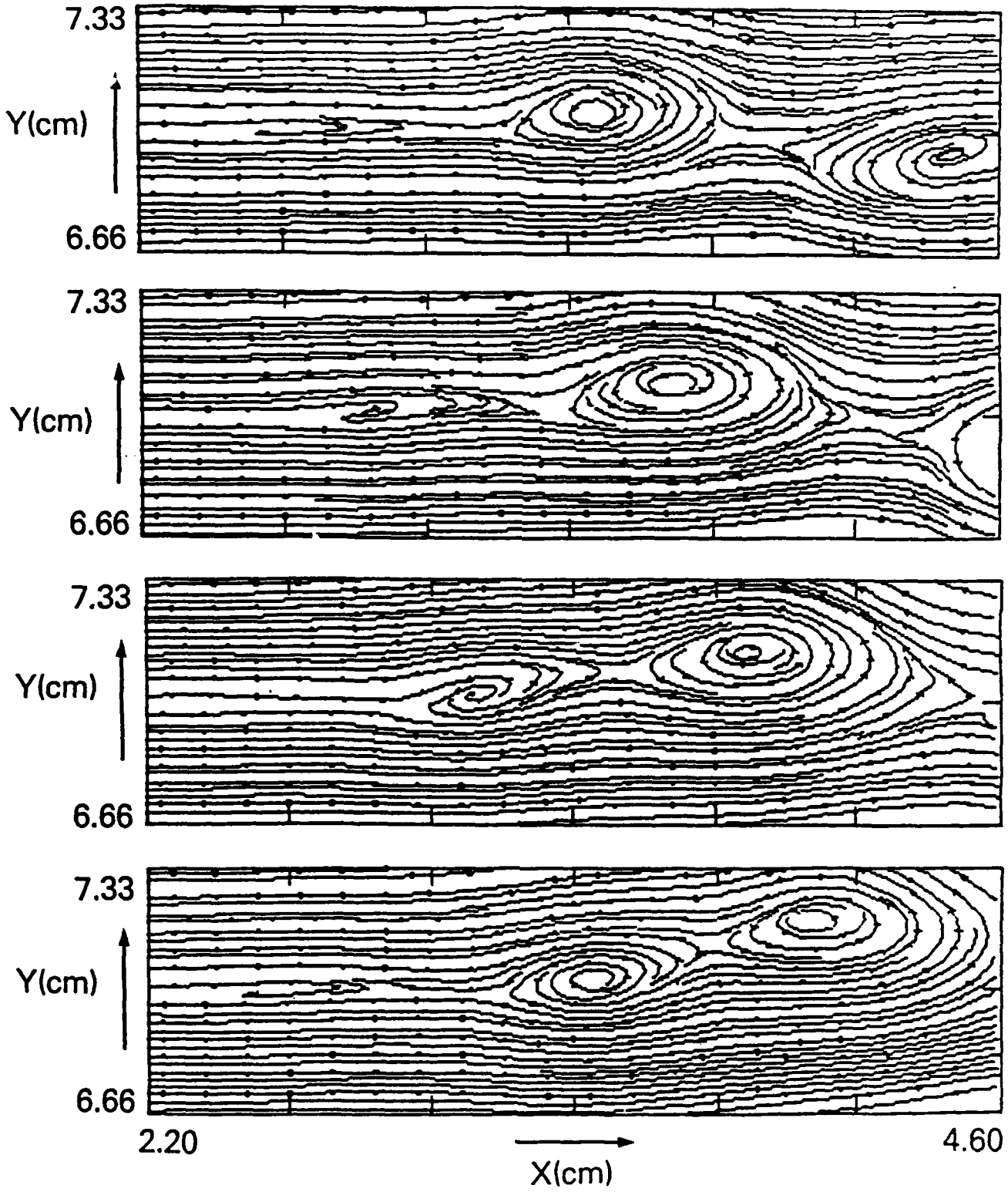


Figure 5

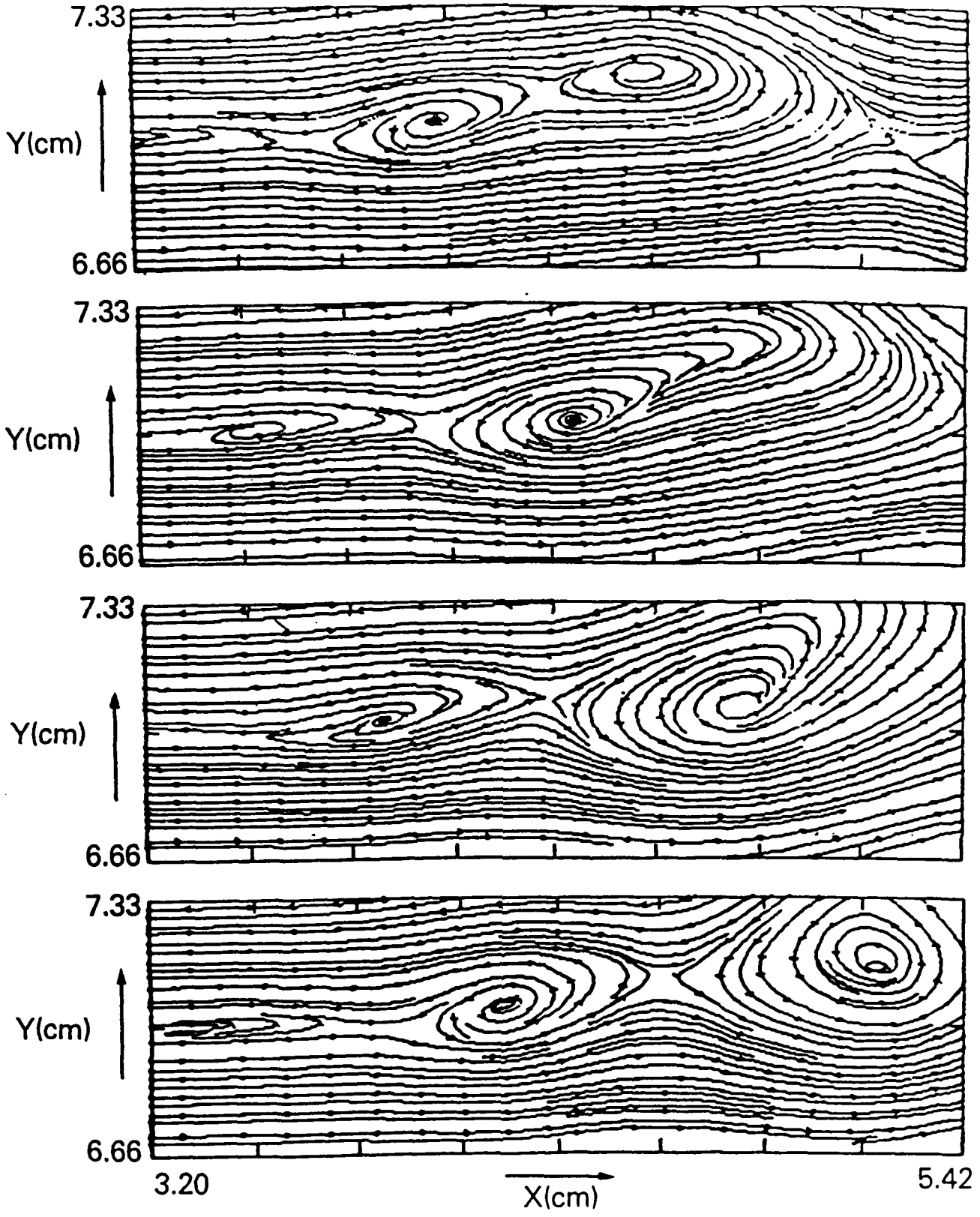


Figure 6

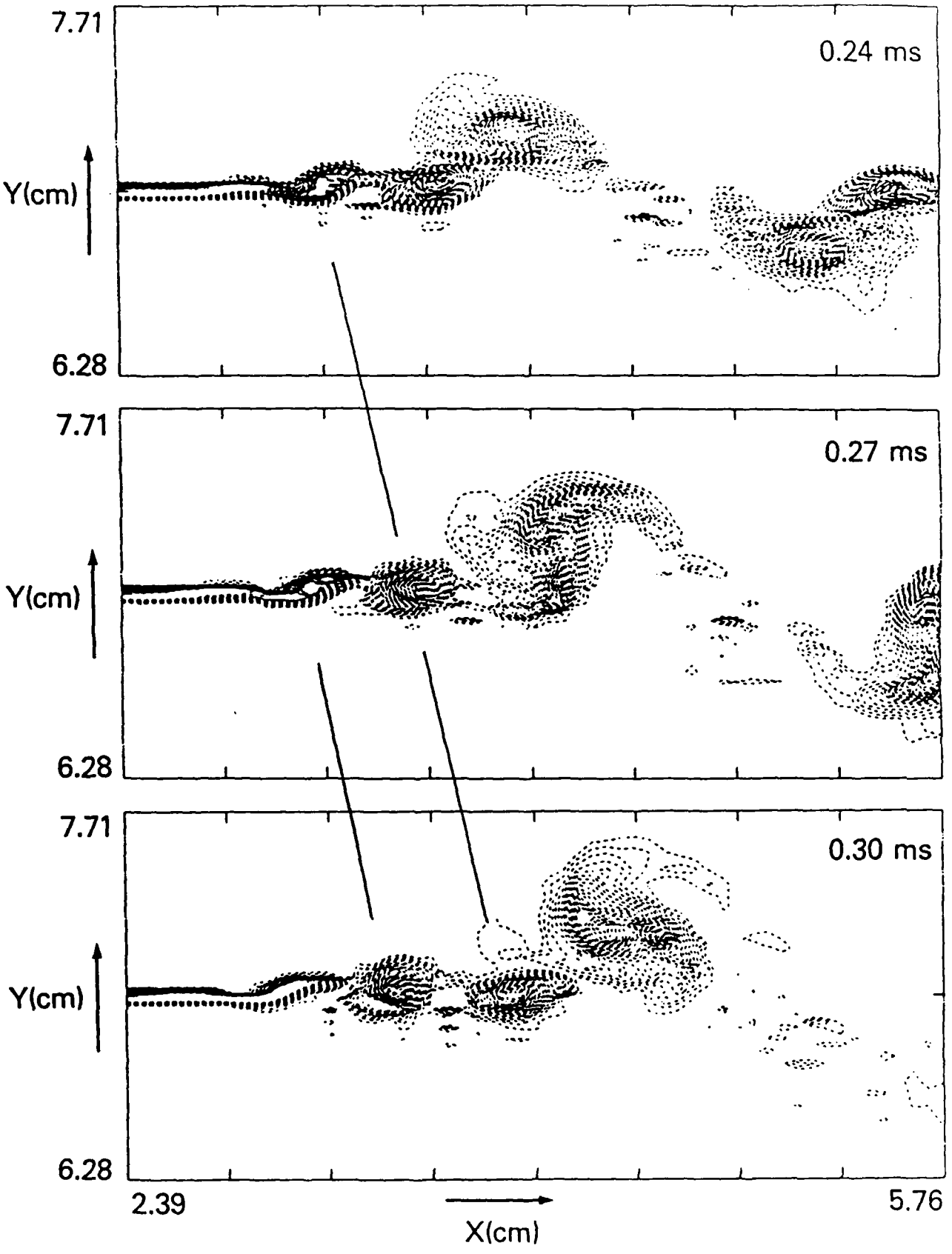

7.71
0.24 ms
Y(cm)
6.28
7.71
0.27 ms
Y(cm)
6.28
7.71
0.30 ms
Y(cm)
6.28
2.39
5.76
X(cm)

Figure 7

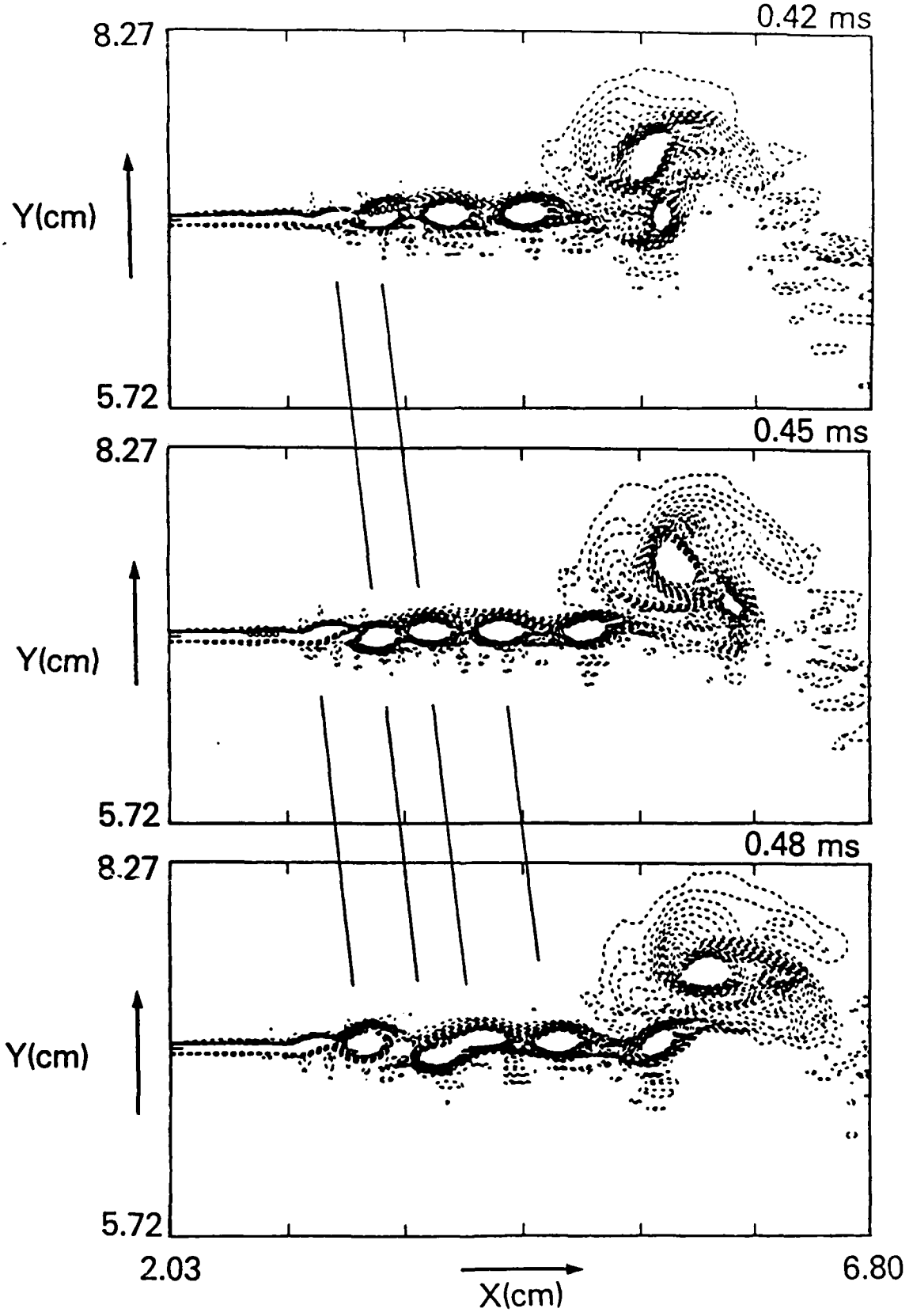

0.42 ms
8.27
Y(cm)
5.72
0.45 ms
8.27
Y(cm)
5.72
0.48 ms
8.27
Y(cm)
5.72
2.03
X(cm)
6.80

Figure 8

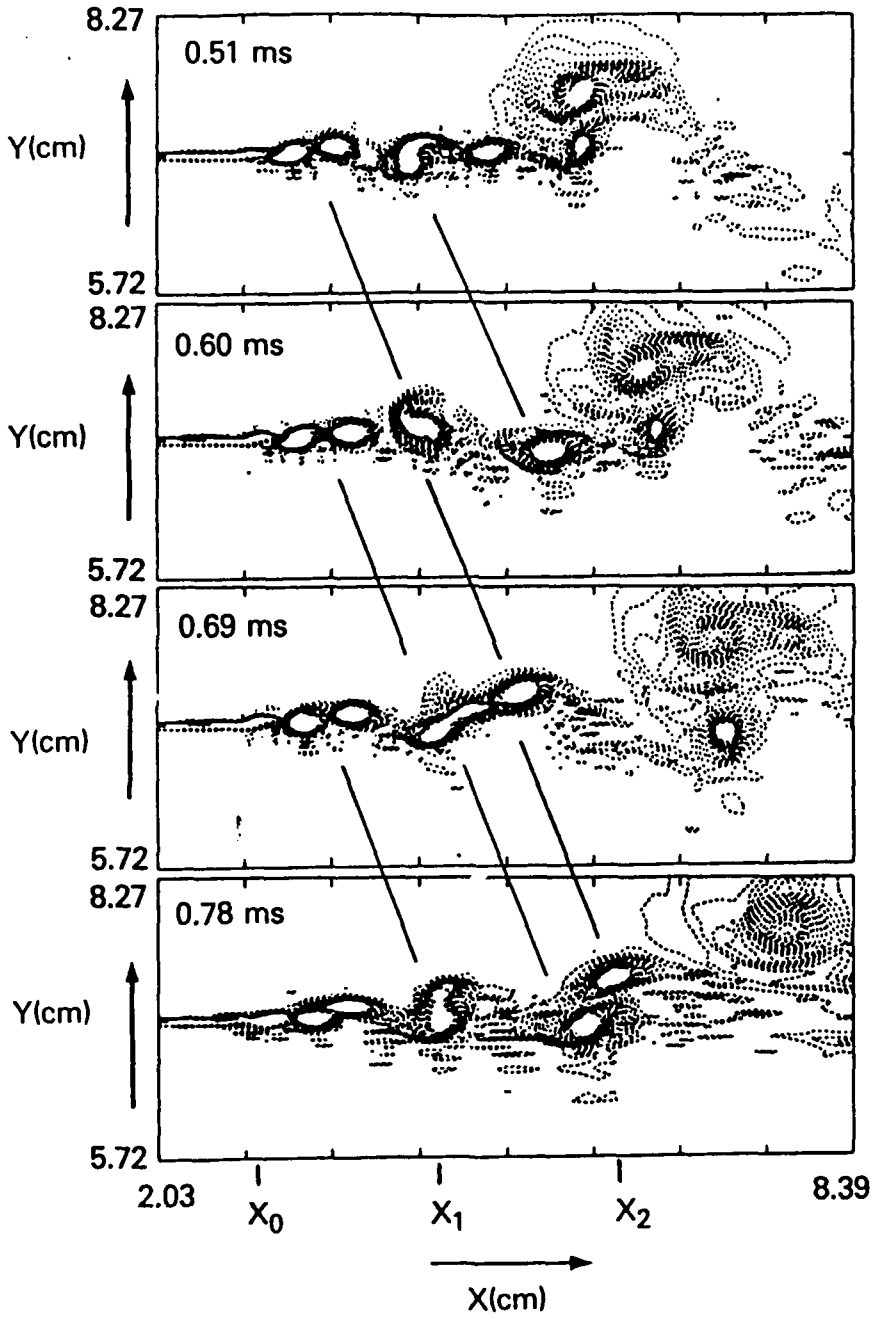

8.27
0.51 ms
Y(cm)
5.72
8.27
0.60 ms
Y(cm)
5.72
8.27
0.69 ms
Y(cm)
5.72
8.27
0.78 ms
Y(cm)
5.72
2.03
X_0
X_1
X_2
8.39
X(cm)

Figure 9

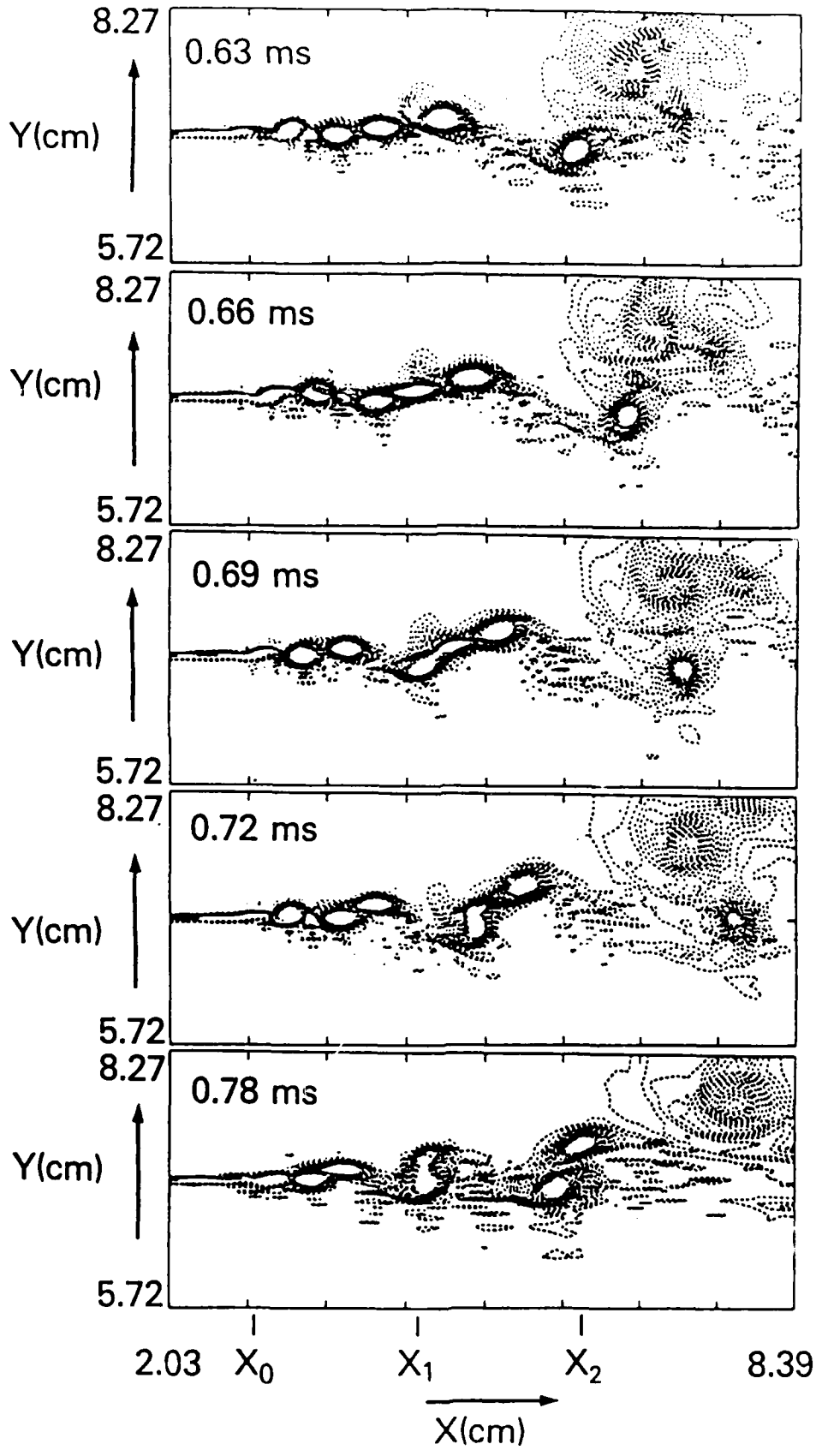

8.27
0.63 ms
Y(cm)
5.72
8.27
0.66 ms
Y(cm)
5.72
8.27
0.69 ms
Y(cm)
5.72
8.27
0.72 ms
Y(cm)
5.72
8.27
0.78 ms
Y(cm)
5.72
2.03
X_0
X_1
X_2
8.39
X(cm)

Figure 10

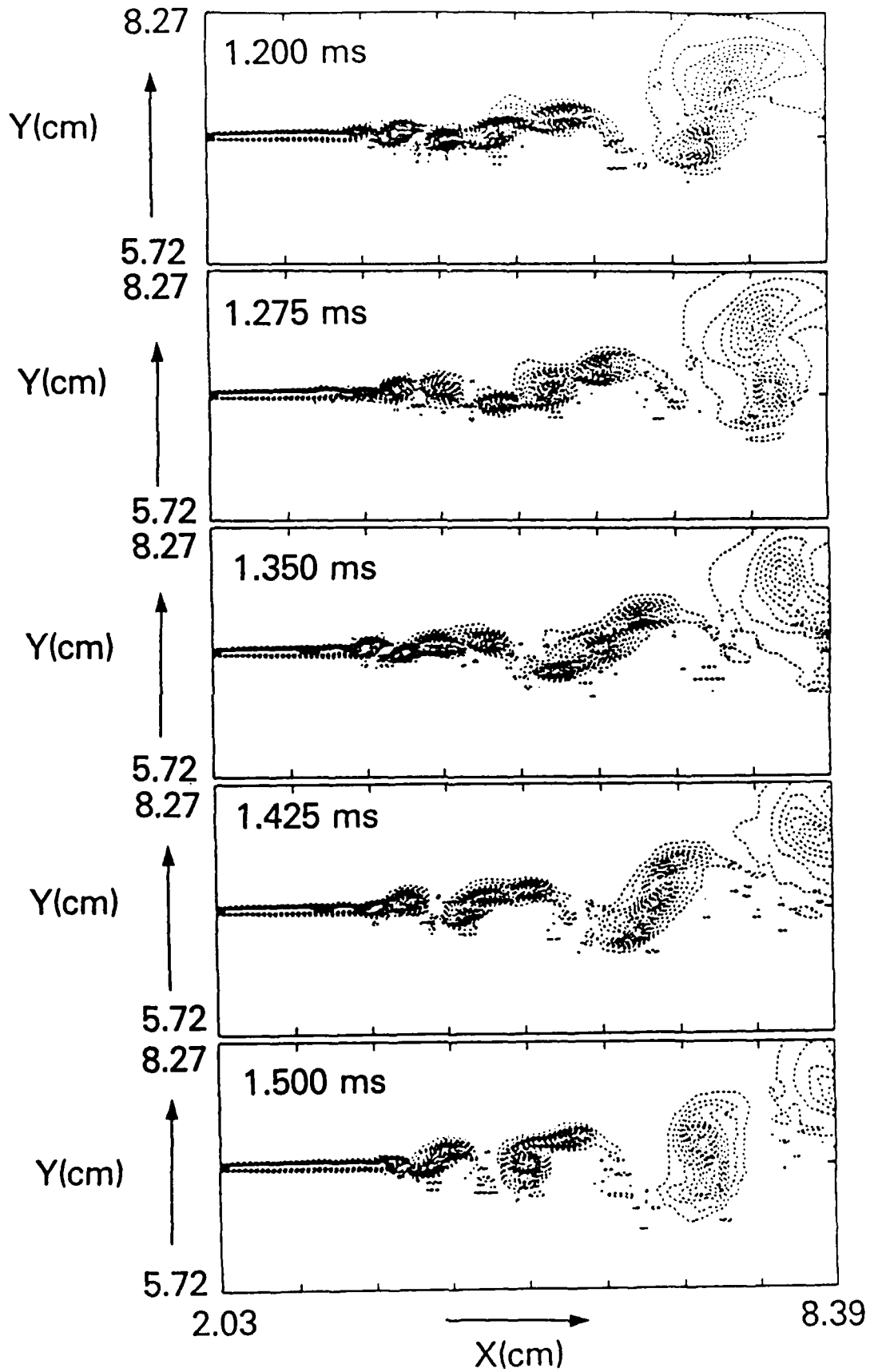

8.27
1.200 ms
Y(cm)
5.72
8.27
1.275 ms
Y(cm)
5.72
8.27
1.350 ms
Y(cm)
5.72
8.27
1.425 ms
Y(cm)
5.72
8.27
1.500 ms
Y(cm)
5.72
2.03
8.39
X(cm)

Figure 11

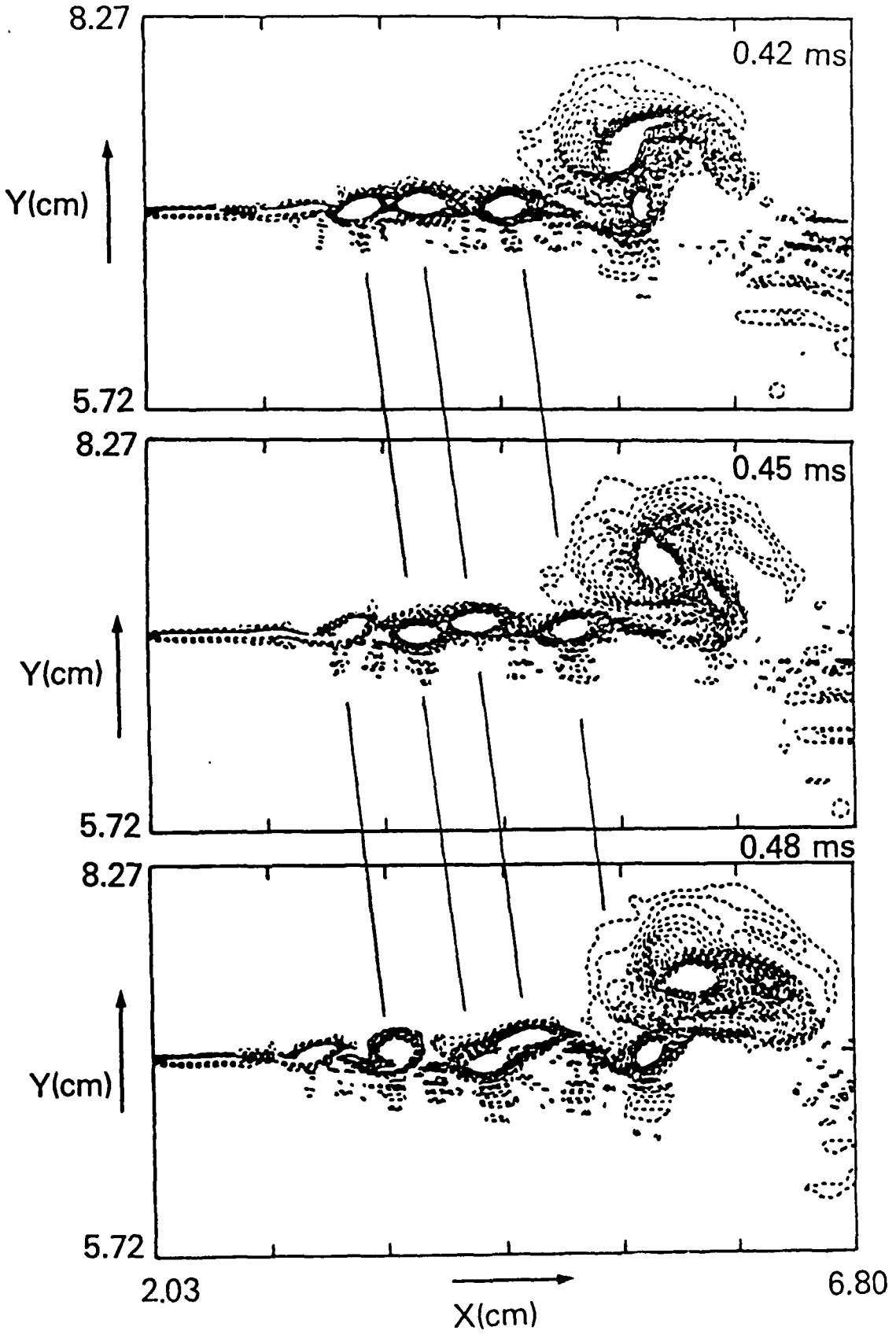

8.27
0.42 ms
Y(cm)
5.72
8.27
0.45 ms
Y(cm)
5.72
0.48 ms
8.27
Y(cm)
5.72
2.03
6.80
X(cm)

Figure 12

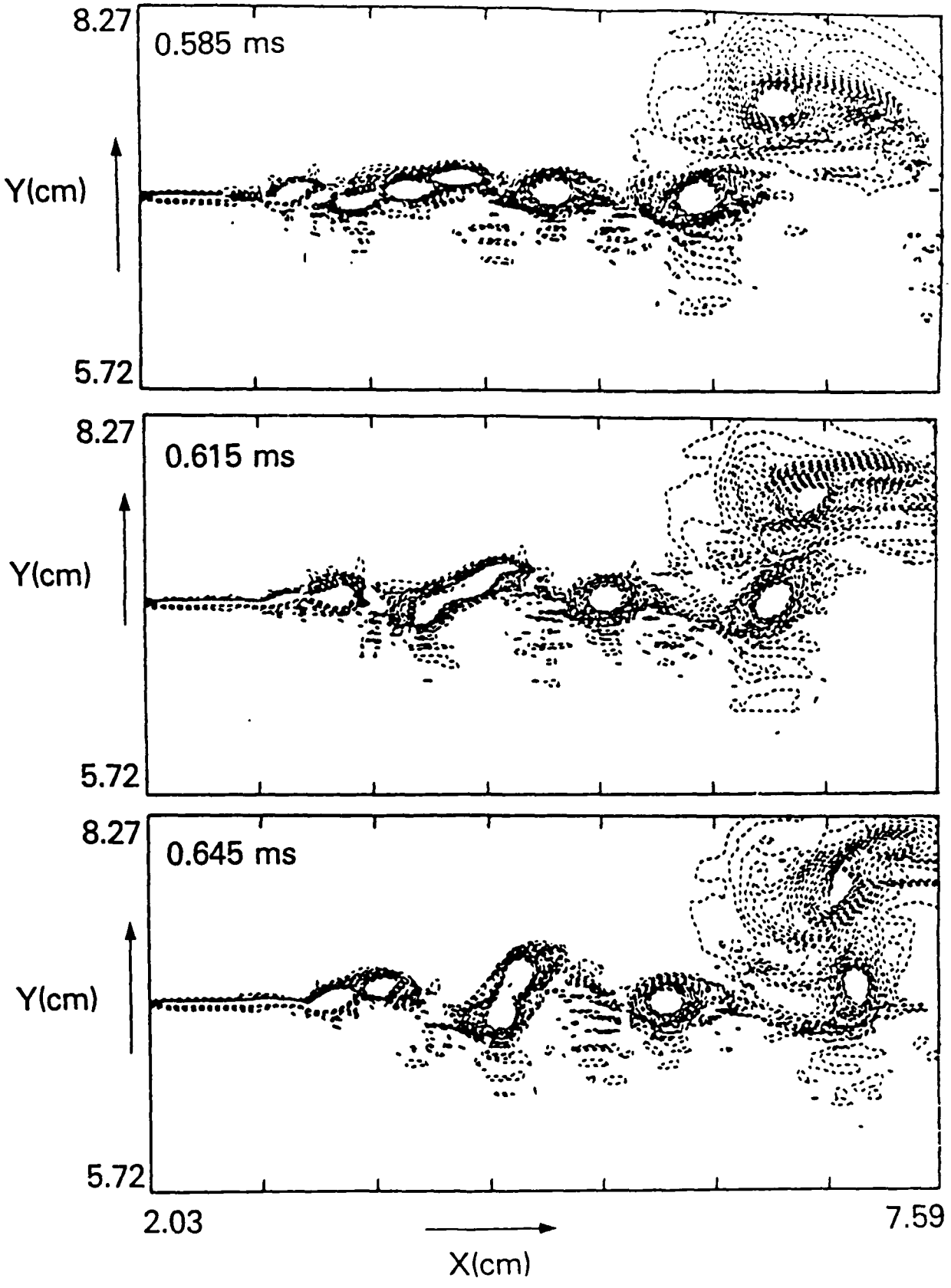

8.27
0.585 ms
Y(cm)
5.72
8.27
0.615 ms
Y(cm)
5.72
8.27
0.645 ms
Y(cm)
5.72
2.03
7.59
X(cm)

Figure 13

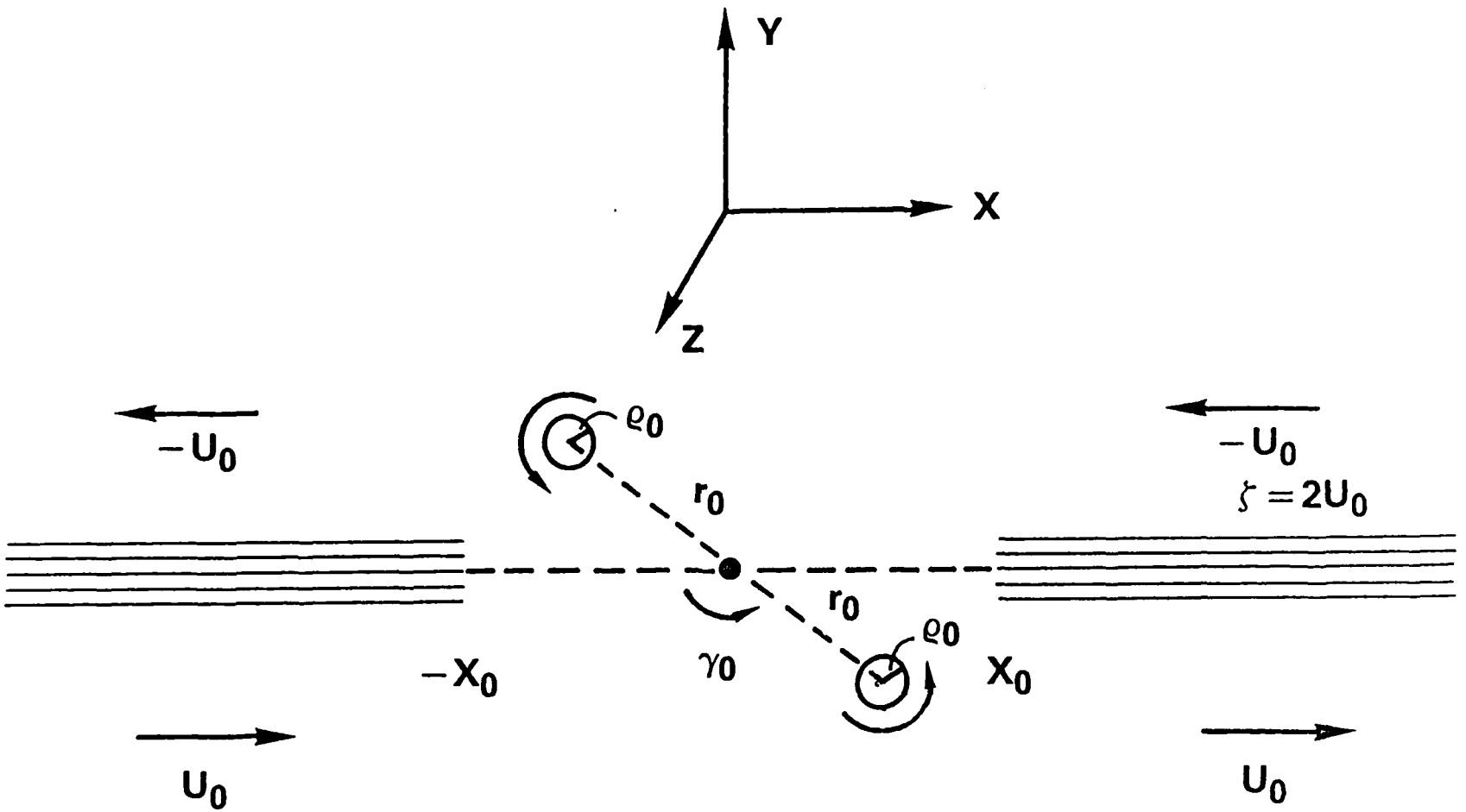


Figure 14

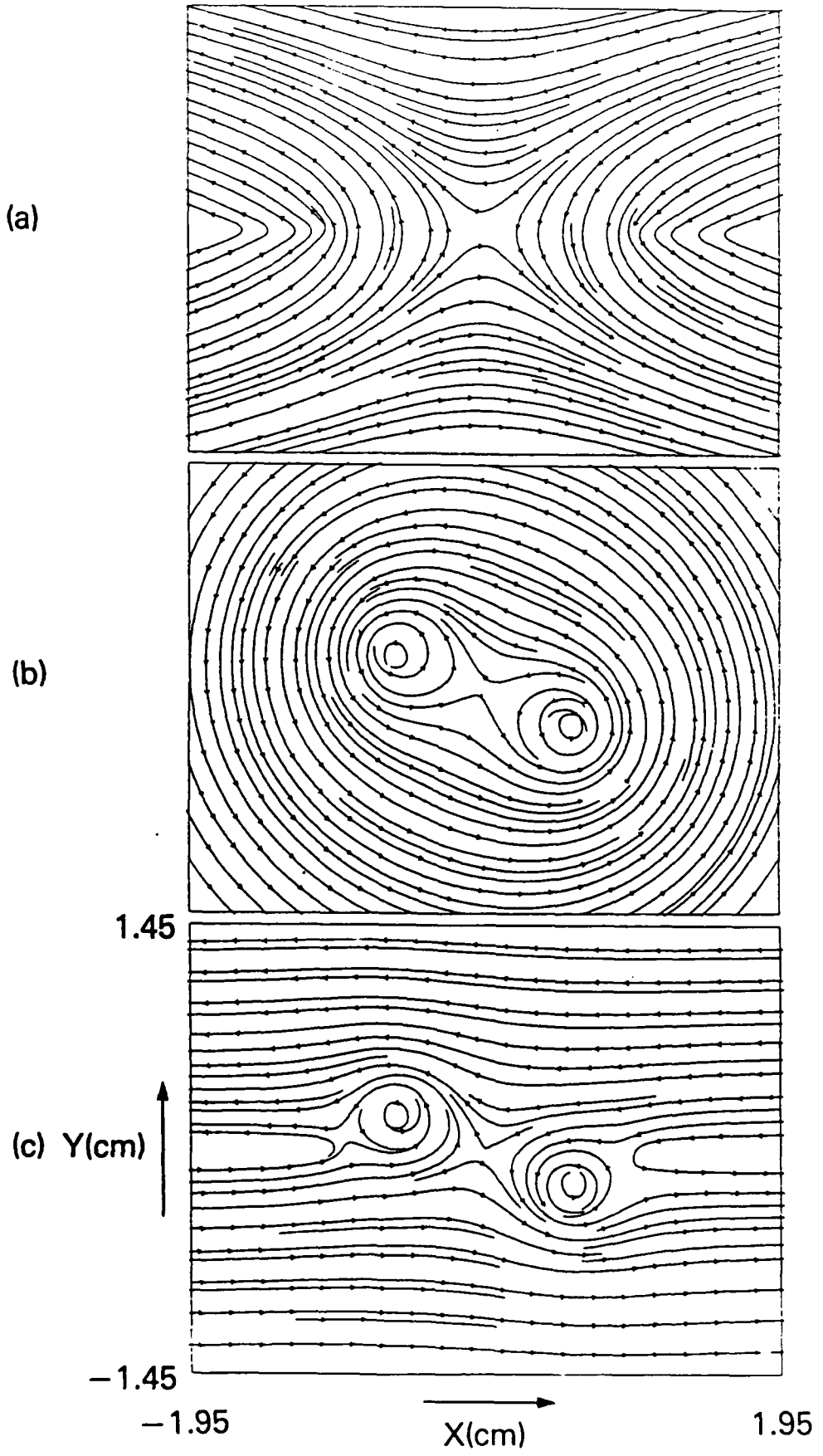
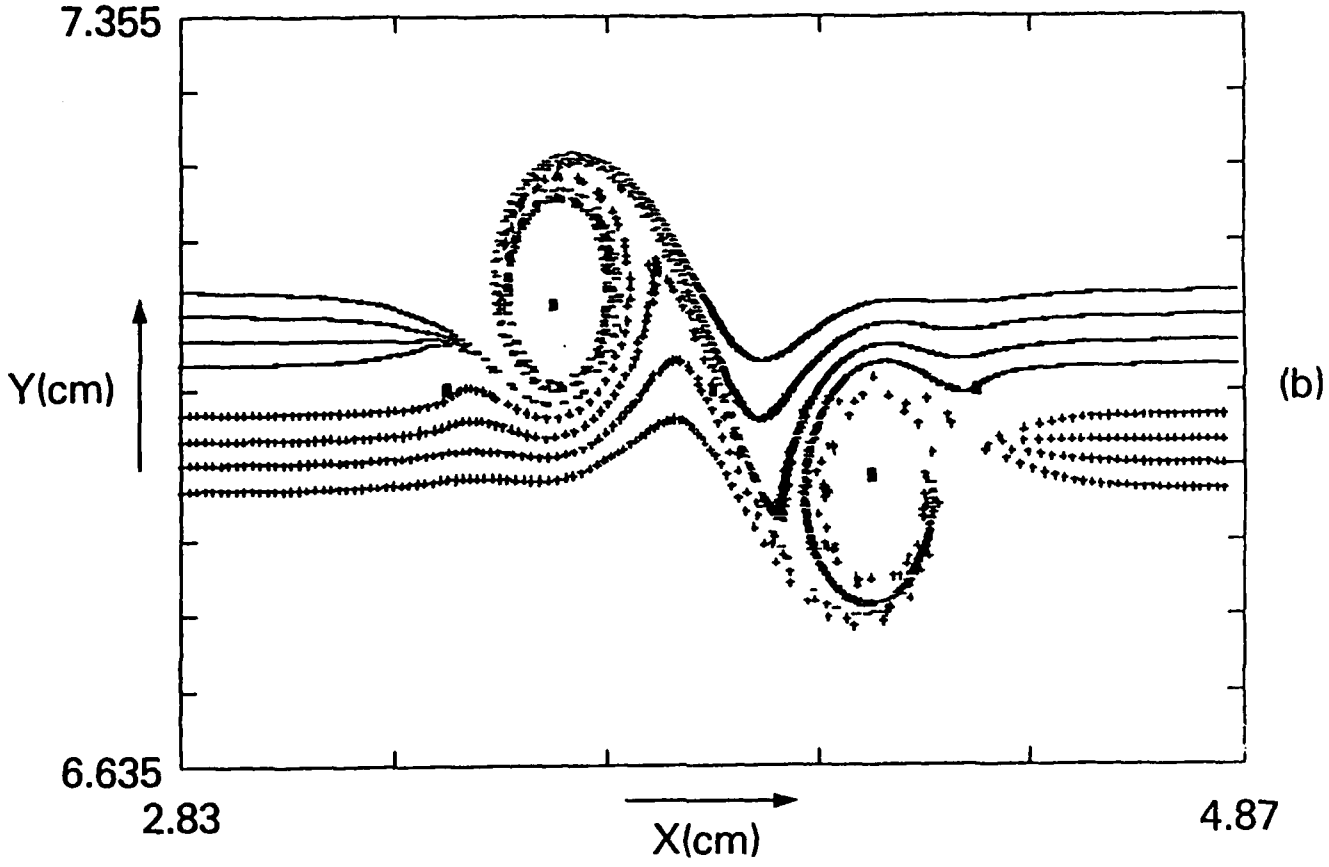


Figure 15



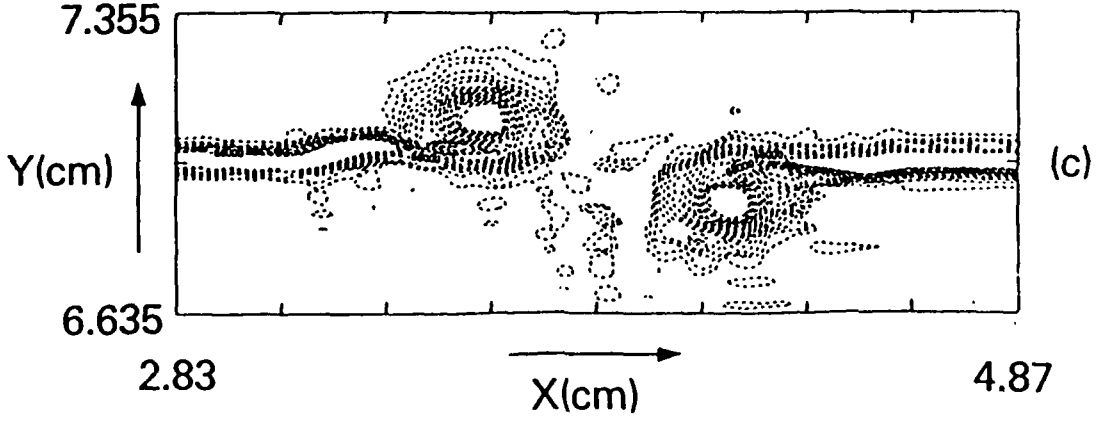


Figure 16

APPENDIX H.

Review of Subgrid Closures and Other Aspects of a LES of Premixed Turbulent Cumbustion

Memorandum: Review of subgrid closures and other aspects of a LES of premixed turbulent combustion

From : Paul A. Libby, Consultant*

Berkeley Research Associates

Date: July 28, 1986 June 20, 1986

Abstract: A LES of premixed turbulent combustion requires models for the turbulent transport of momentum and species due to fluctuations at the subgrid scales. The extensive literature relative to such models for flows with constant fluid properties is reviewed in order to rationalize to the extent possible models for application to turbulent combustion. Other aspects of such a LES application are also discussed.

* Also Professor of Fluid Mechanics, University of California San Diego, La Jolla, California 92093

INTRODUCTION

In [1] and [2] we discuss the LES of premixed turbulent combustion involving the idealization associated with the Bray-Moss model of the thermochemistry of such combustion. That model incorporates assumptions commonly accepted in combustion theory and when applied to a LES reduces the describing equations to those for conservation of mass, momentum and a progress variable $c(x,t)$ which determines the entire thermochemical state of the gas. With averaging as called for in a LES the nonlinearity of the partial differential equations leads to flux terms which represent turbulent transport due to scales smaller than the grid size, i.e., to the subgrid scales. In [1] and [2] we are rather casual regarding the models to be applied to remove these terms and thereby to achieve a closed set of equations. It is the purpose of the present memorandum to reconsider such modeling by first distilling the extensive literature on subgrid models for flows with constant properties and by then setting forth the most promising models for a LES of premixed turbulent combustion. In the course of carrying out this literature review we uncovered other aspects of a LES of int aspects of a LES of significance and we therefore comment on them.

An interesting historical perspective on subgrid scale modeling is given by Herring [3] who notes that Lilly [4] in 1967 "...appears to be the first to make (explicit) the separation of computational grid scales from subgrid scales by introducing volume averaging over the computational grid. Such averaging is implicit in earlier work but explicit use of grid averaging did not occur." The notions of volume averaging and filtering are now standard in the vocabulary of LES. It is also interesting to note that Lilly [4] in the same reference states:"It is totally inconceivable that a computer could

resolve both the energy containing and dissipative scales in a high Reynolds number regime but it is not at all unlikely that the limits of resolution could extend from the largest energy containing scale into the inertial subrange." This perspective continues to motivate interest in LES despite the significant increase in computing power which has occured since 1967.

ANALYSIS

We start our considerations by writing in Cartesian tensor notation the fundamental partial differential equations describing the system to be discussed, namely

$$\frac{\partial \rho}{\partial t} + \frac{\partial}{\partial x_k}(\rho u_k) = 0 \tag{1}$$

$$\frac{\partial}{\partial t}(\rho u_i) + \frac{\partial}{\partial x_k}(\rho u_k u_i) = -\frac{\partial p}{\partial x_i} \tag{2}$$

$$\frac{\partial}{\partial t}(\rho c) + \frac{\partial}{\partial x_k}(\rho u_k c) = w \tag{3}$$

where we neglect molecular transport.* In doing so we implicitly assume either that applications are confined to free shear flows or that the viscous sublayers are not resolved in flows involving walls.**

These five equations for the velocity components $u_i(x,t)$, the pressure $p(x,t)$ and the progress variable $c(x,t)$ are completed by a model for the chemical reaction term w and by the algebraic relation for $\rho(c)$ given by the Bray-Moss model, namely

$$\rho = \frac{\rho_r}{1 + \tau c} \tag{4}$$

where τ is a heat release parameter and ρ_r is the density within reactants.

* The indices k and l are reserved for summation, i.e., they appear only in pairs. Other indices which might be repeated do not imply summation.

** Since there appears to be some uncertainity regarding the boundary counditions applicable at solid surfaces (cf., e.g., Deardorff [5]), we discuss later this matter.

In [2] we propose a model for w.

It is worth noting that with the chemical reaction term suppressed Eqs. (1) - (4) describe low speed mixing of a binary system. In this case c is the mass fraction of one species and τ is a molecular weight parameter. For example, if helium mixes with air, then $\tau = 6.2$. The implication of the notation $\rho(c(x,t))$ applies to binary mixing. The only aspect of the following discussion which does not apply to this nonreactive case pertains to the existence of flamelets and to regions of constant density separated thereby. In binary mixing the density can take on all values between the limiting values associated with the two pure species. However, the filtering, modeling and numerical analysis can be applied without change to binary mixing.

In a LES Eqs. (1) - (4) are volume averaged by application of a filter function $G(x' - x)$ with a characteristic dimension Δ, roughly equal to, but in general different from, the spacing of the finite difference grid, so that filtered values of the dependent variables are calculated. We suggest in [1] that for the variable density flow of interest in the present context Favre averaging is appropriate so that the filtered velocity components are given by

$$\overline{\rho(x,t)\ \tilde{u}_i(x,t)} = \int dx'\ G(x' - x)\ \rho(c(x',t))\ u_i(x',t)$$

$$= \overline{\rho u_i}(x,t) \qquad (5)$$

where the volume averaged density is

$$\overline{\rho(x,t)} = \int dx'\ G(x' - x)\ \rho(c(x',t)) \qquad (6)$$

Thus tilded quantities are mass averaged, barred quantities are conventionally averaged. Similar expressions prevail for the filtered progress variable $\tilde{c}(x,t)$ and the averaged pressure $p(x,t)$. Note that if molecular transport is included, Favre averaging complicates the molecular terms but within the context of the present discussion this feature can only be important when a turbulent flame impinges on a wall. In binary mixing attention must be devoted to the entire wall region but as noted earlier this is the case whether Favre or conventional averaging is employed.

It is worth noting that the application of fintie differencing is equiivalent to a form of filtering. Consider the frequently adopted central diference representation of a first derivative and its interpretation from this perspective, namely

$$\left|\frac{\partial u}{\partial x}\right|_n \simeq \frac{u_{n+1} - u_{n-1}}{2h} = \int_{x-h}^{x+h} \frac{1}{2h} \frac{\partial u}{\partial x'}\, dx'$$

In this case the filter function is two valued, 1/2 h for $x - h < x' < x + h$ and zero elsewhere. This is the filter employed by Deardorff [5]. Finite difference representations of higher order derivatives can be synthesized by combinations of first order derivatives and can thus be interpreted in terms of sums and differences of filtered first derivatives. From this perspective the filtering used to develop the equations solved in a LES should be considered preliminary to the differencing subsequently used to solve numerically the resulting partial differential equations. This is the separation of computational and subgrid scales noted earlier.

In many premixed reacting flows of applied interest reactants and

products are separated by surfaces, laminar flamelets, whose thickness is estimated to be less than a Kolmogoroff length and thus to be at an unresolved, subgrid scale. Accordingly, only at grid points several multiples of the filter length Δ from the instantaneous location of the flamelet does the density differ from either ρ_r or $\rho_p = \rho_r/(1+\tau)$. Thus for most grid points the averaging of Eq. (5) reduces to that conventionally applied in LES. The implication is that the subgrid models carefully developed for constant density flows apply directly to premixed turbulent combustion within the context of the Bray-Moss model except in the neighborhood of the flamelets where we find that explicit variable density effects enter the subgrid modeling.

It is appropriate to note in this regard that the analysis of [1] is based on tracking the flamelet location, i.e., on flamelet dynamics. An alternative which may be numerically more convenient is discussed in [2] and assumed to apply here; it is based on the calculation of $\tilde{c}(x,t)$ by solving the filtered form of Eq. (3) such that $\tilde{c}$ is zero or unity at all grid points except those within several multiples of Δ from the instantaneous flamelet location.

We identify the difference between a grid point value of a fluid mechanical variable and its volume averaged counterpart as a fluctuation; thus as in the usual turbulence phenomenology involving Favre averaging

$$u_i(x,t) = \tilde{u}_i(x,t) + u_i''(x,t) \tag{7}$$

where

$$\overline{\rho u_i''} = 0, \quad \overline{u_i''} \neq 0$$

Balance equations for statistical quantities involving the fluctuations can be developed and closely resemble those used in second moment methods and indeed in one subgrid closure method for LES the transport coefficient involves the "turbulent kinetic energy ", i.e., $\tilde{k} = \frac{1}{2} \rho u_k'' u_k'' / \rho$, and thus a balance equation for $\tilde{k}(x,t)$ is incorporated into the system of describing equations (cf. Lilly [4] and Grotzbach and Schumann [6]). However, more generally second moment quantities involving fluctuations are expressed in terms of prime dependent variables, i.e., closure is achieved at the first moment level.

The Filtered Equations and the Subgrid Terms

A review of the existing literature on LES indicates that applications to date are limited to constant density flows and that for such flows a variety of subgrid closures are employed, a situation indicative of active development of the method. We have found two references [6] and [7] describing applications involving passive scalars, either small temperatures or small concentrations of a contaminant, and thus of interest in the treatment of Eq. (3). Among the extensive literature on subgrid modeling we find the presentation of Leslie and Quarini [8] relative to the decomposition of the several contributions to the subgrid terms clear and direct and we therefore tailor our discussion after their work; they in turn credit Leonard [9] with guiding their exposition. The review article by Rogallo and Moin [10] is also an excellent source for many aspects of LES including an extensive bibliography circa 1984.

When volume averaging is applied to Eqs. (1) - (3) we obtain

$$\frac{\partial \bar{\rho}}{\partial t} + \frac{\partial}{\partial x_k} (\bar{\rho} \tilde{u}_k) = 0 \qquad (8)$$

$$\frac{\partial}{\partial t}(\bar{\rho}\,\tilde{u}_i) + \frac{\partial}{\partial x_k}(\bar{\rho}\,\tilde{u}_k\,\tilde{u}_i) = -\frac{\partial \bar{p}}{\partial x_i} + \frac{\partial}{\partial x_k}(\mathcal{L}_{ik} + T_{ik}) \tag{9}$$

$$\frac{\partial}{\partial t}(\bar{\rho}\,\tilde{c}) + \frac{\partial}{\partial x_k}(\bar{\rho}\,\tilde{u}_k\,\tilde{c}) = \bar{w} + \frac{\partial}{\partial x_k}(\mathcal{L}_k + T_k) \tag{10}$$

where the last terms on the right side of Eqs. (9) and (10) describe the influence of fluctuations at the subgrid scale on the filtered variables. Specifically, the terms L_{ij} and L_i represent the Leonard stresses and Leonard fluxes* defined by

$$L_{ij} = \overline{\rho\tilde{u}_i\tilde{u}_j} - \bar{\rho}\,\tilde{u}_i\tilde{u}_j \tag{11}$$

$$L_i = \overline{\rho\tilde{u}_i\tilde{c}} - \bar{\rho}\,\tilde{u}_i\tilde{c} \tag{12}$$

while the terms T_{ij} and T_i represent the true subgrid stresses and fluxes and are given by

$$T_{ij} = \overline{\rho u_i''\tilde{u}_j} + \overline{\rho u_j''\tilde{u}_i} + \overline{\rho u_i''u_j''} \tag{13}$$

$$T_i = \overline{\rho u_i''\tilde{c}} + \overline{\rho c''\tilde{u}_i} + \overline{\rho u_j''c''} \tag{14}$$

The Leonard Terms

In [9] the Leonard stresses are evaluated by expanding the individual filtered velocities at x' in a Taylor series about x and by carrying out the requisite volume averaging. When applied to variable density flows subject to Favre averaging, this proceedure leads to the intermediate result

* This identification is new but involves a direct extension of the notion of Leonard stresses to the turbulent diffusion of a scalar.

$$L_{ij} = \frac{\partial}{\partial x_k}(\bar{u}_i\bar{u}_j) \int dx' \; G(x' - x) \; (x_k' - x_k) \; \rho(c(x',t))$$

$$+ \frac{\partial \bar{u}_i}{\partial x_k}\frac{\partial \bar{u}_j}{\partial x_l} \int dx' \; G(x' - x) \; (x_k' - x_k) \; (x'_l - x_l) \; \rho(c(x',t)) \qquad (15)$$

If the density is constant, the first term on the right is zero for filter functions which are symmetric in each coordinate direction and the second term is non-zero only if k = 1. Thus we recover Leonard's result. The implication of Eq. (15) is that in the neighborhood of the laminar flamelets there is an alteration of the Leonard stresses.

Realistic and proper evaluation of the integrals in Eq. (15) requires knowledge of the location of the flamelet within the filter volume since moments about x of the partial volumes occupied by the reactants with $\rho = \rho_r$ and products with $\rho = \rho_p$ within the filter volume are involved. In our present approach the location of the flamelet is estimated from knowledge of the filtered progress variable $\tilde{c}(x,t)$ with the consequence that only crude evaluations of the integrals are possible. Numerical experimentation and comparison with data are needed to establish the adequacy of such an evaluation.*

To proceed we provisionally assume that within the effective filter volume we can approximate the density as follows:

* It is worth noting in this regard that if $\tilde{c}(x,t)$ is known, then the fraction of the filter volume occupied by product at each grid point is likewise known from $V_p = (1 + \tau)\,\tilde{c}/(1 + \tau\,\tilde{c})$ which is exactly β in the Bray-Moss description of the thermochemistry of premixed turbulent combustion. We discuss later the means of assessing subgrid models.

$$\rho(c(x',t)) \simeq \rho(\tilde{c}(x,t)) + \left(\frac{d\rho}{d\tilde{c}}\right)(x,t)\left(\frac{\partial\tilde{c}}{\partial x_k}\right)(x,t)(x_k' - x_k) \tag{16}$$

where from Eq. (4)

$$\frac{d\rho}{d\tilde{c}}(x,t) = -\frac{\tau\rho_r}{(1+\tau\tilde{c}(x,t))^2}$$

Equation (16) has the correct limiting behavior; if $\tilde{c} = 0,1$, then $\rho(c(x',t))$ becomes either of the two limiting values of ρ. If the flame sheet is located at x, the gradient $(\partial\tilde{c}/\partial x_i)(x,t)$ is maximum and the maximum gradients of ρ within the filter volume are obtained. Finally, the series expansion of the density via Eq. (16) is consistent with locating the flame sheet in terms of the distribution $\tilde{c}(x,t)$. Thus Eq. 16 provides a plausible, although perhaps provisional, means for evaluating the integrals in Eq. (15). Note that in an application to binary mixing Eq. (16) is consistent with the Taylor series expansion used for the the velocity components.

With this approximation Eq. (15) becomes

$$L_{ij} \simeq \gamma_k \left| -\frac{\tau\rho_r}{(1+\tau\tilde{c})^2}\frac{\partial\tilde{c}}{\partial x_k}\frac{\partial}{\partial x_k}(\tilde{u}_i\tilde{u}_j) + \bar{\rho}\frac{\partial\tilde{u}_i}{\partial x_k}\frac{\partial\tilde{u}_j}{\partial x_k}\right| \tag{17}$$

where

$$\gamma_i \equiv \int_{-\infty}^{\infty} dx'\, G(x'-x)\,(x_i' - x_i)^2$$

is a constant depending only on the filter function. For a symmetric Gaussian filter

$$G(x' - x) = \left|\frac{6}{\pi\Delta^2}\right|^{3/2} \exp\left|-\frac{6}{\Delta^2}(x_k' - x_k)(x_k' - x_k)\right| \tag{18}$$

the coefficient γ_i is independent of the index and is found to be

$$\gamma_i = \gamma = \frac{\Delta^2}{6} \tag{19}$$

At a location fully within either reactants or products so that $\partial\tilde{c}/\partial x_i = 0$ Eq. (17) reduces properly to one of the two expressions for L_{ij} given in [9].

Note that the Leonard stresses are proportional to Δ^2. Thus their modeling and inclusion are indicated only if the finite differencing of the filtered equations is second order accurate, i.e., involves errors of $O(h^3)$. Many applications of LES employ fourth order differencing schemes (cf., e.g., [11]).

Leonard [9] provides an alternative form for L_{ij} on the basis of expanding the product $u_i u_j(x',t)$ about x,t (See also Eq. (5) of [10]); there results a model possessing the unappealing feature of raising the order of the filtered momentum equations, i.e., of involving $\partial^2 u_i u_j/\partial x_k \partial x_k$ so that $\partial L_{ik}/\partial x_k$ is third order. Although this feature is apparently recognized by research workers involved in LES, it is not seriously considered.* In our extension to variable density flows we suggest employing Eqs. (17) and (19).

A similar calculation can be carried out for L_i; in this case we approximate $\tilde{c}(x',t)$ by expanding about x and obtain the following intermediate result

* Private communication with Professor A. Leonard. In [7] a detailed discussion of the differencing scheme used to treat the filtered equations contains no mention of the third-order Leonard terms.

analogous to Eq. (15):

$$L_i = \frac{\partial}{\partial x_k}(\bar{u}_i\bar{c}) \int dx'\, G(x' - x)\,(x_k' - x_k)\,\rho(c(x',t)) +$$

$$\frac{\partial \bar{u}_i}{\partial x_k}\frac{\partial \bar{c}}{\partial x_l}\int dx'\, G(x' - x)\,(x_k' - x_k)(s_l' - x_l)\,\rho(c(x',t)) \qquad (20)$$

If we again use the approximation of Eq. (16) and Eq. (18), this becomes

$$L_i = \gamma\left|-\frac{\tau\rho_r}{(1+\tau\bar{c})^2}\frac{\partial\bar{c}}{\partial x_k}\frac{\partial}{\partial x_k}(\bar{u}_i\bar{c}) + \bar{\rho}\frac{\partial\bar{u}_i}{\partial x_k}\frac{\partial\bar{c}}{\partial x_k}\right| \qquad (21)$$

In regions of the flow where $\bar{c} = 0,1$ Eq. (21) yields, as it should, $L_i = 0$.

The Subgrid Stresses and Fluxes

There is an extensive literature on the modeling of the subgrid stresses T_{ij} and a few references to the subgrid fluxes. For constant density flows the most widely used and thoroughly studied is an eddy viscosity model. Its generalization to the variable density flows of interest to us would appear to be*

$$T_{ij} = -2\mu_T\left(\frac{\partial\bar{u}_i}{\partial x_j} + \frac{\partial\bar{u}_j}{\partial x_i} - \frac{2}{3}\delta_{ij}\frac{\partial\bar{u}_k}{\partial x_k}\right) + \frac{2}{3}\bar{\rho}\,\bar{k}\,\delta_{ij} \qquad (22)$$

where $\bar{k} = \frac{1}{2}\overline{\rho u_k'' u_k''}/\bar{\rho} = T_{kk}/2\bar{\rho}$ is the kinetic energy of the subgrid velocity fluctuations. Equation (22) with $i = j$ and the definition of $\bar{k}$ are interpreted as four equations for T_{ii} and $\bar{k}$ given a model for the exchange

* The form given in [1] does not contract properly and should be replaced by Eq. (22).

coefficient μ_T and the gradients of the mean velocity $\partial\bar{u}_i/\partial x_j$. In addition Eq. (22) reduces properly upon contraction to an identity and for constant density flows to the usual form of T_{ij}.

Various models for the turbulent exchange coefficient appear in the literature of LES. A modification of the widely used Smagorinsky model is

$$\mu_T = (c_\mu \Delta^2 \bar{\rho} \Omega \sigma_{kl} \sigma_{kl})^{\frac{1}{2}} \tag{23}$$

where

$$\sigma_{ij} = \frac{1}{2}\left(\frac{\partial\bar{u}_i}{\partial x_j} + \frac{\partial\bar{u}_j}{\partial x_i}\right)$$

is the mean rate of strain tensor based on the resolved velocities and where c_μ is an empirical constant. Values of c_μ in the range 0.1 to 0.23 are found to be required in constant density flows on the basis of a variety of applications.

Several examples illustrate the uncertainities involved; in decaying isotropic turbulence the requirement that the computed energy decay rate match experimental data leads to c_μ between 0.19 and 0.24. In channel flow Deardorff [5] finds that $c_\mu = 0.1$. There appears to be some indication (cf. Ragollo and Moin [10]) that the uncertainities in c_μ can be reduced if the mean rate of strain is removed from the model for c_μ, i.e., if Eq. (23) is replaced by

$$\mu_T = (c_\mu \Delta^2 \bar{\rho} \Omega (\bar{\sigma}_{kl} - \langle\bar{\sigma}_{kl}\rangle)(\bar{\sigma}_{kl} - \langle\bar{\sigma}_{kl}\rangle))^{\frac{1}{2}} \tag{24}$$

where $\langle\bar{\sigma}_{ij}\rangle$ is an average of the mean rate of strain. In a statistically stationary flow this mean quantity can be computed as a running average at each

grid point or over several neighboring points while in a transient flow averaging over some suitable plane or volume is indicated. On the basis of these results we must anticipate that within the context of premixed turbulent combustion c_μ must be adjusted to achieve agreement with experimental data and that a range of numerical experiments must be undertaken. The selection of c_μ is part of the assessment of subgrid models discussed later.

We turn now to the subgrid flux $T_i(x,t)$; Antonopoulos-Domis [7] introduces a second empirical constant denoted c_θ which "...can be thought of an an eddy Prandtl number" so that

$$T_i = - c_\theta \mu_T \bar{\rho} \frac{\partial \tilde{c}}{\partial x_i} \qquad (25)$$

By comparing the predictions of a LES with the experimental data of Yeh and van Atta [11] on the decay of velocity and temperature intensities in a heated grid flow Antonopoulos-Domis concludes that $c_\mu = 0.23$, a value at the edge of the observed range, and $c_\theta = 2.0$. There is considerably less attention devoted to assessing Eq. (25) thatn to the constant density counterpart of Eq. (22).

Evaluation of Subgrid Models

The question arises as to the means for evaluating the accuracy of the models of the Leonard stresses L_{ij}, the Leonard fluxes L_i, the subgrid stresses T_{ij} and the subgrid fluxes T_i. We discuss this matter within the context of constant density flows with the notion that similar means apply to variable density and reacting flows at least in principal. Generally such an evaluation is based on comparisons of the predictions of some filtered variables measured

in a related experiment, i.e., the distributions of the velocity components u_i and the pressure p. If satisfactory agreement is achieved, it is assumed that the subgrid modeling used to make the prediction is satisfactory at leat in the limited sense implied by such an agreement. It is recognized that this evaluation is indirect and that more rigorous methods are desireable. In fact this is the need addressed by Clark et al [12] who carry out a direct numerical simulation of isotropic decaying turbulence in a cube of side L simulating grid flow so that the velocity and pressure on a 64^3 grid for fifty time steps provide a basis for evaluating subgrid models directly.

This proceedure is initiated by overlaying the 64^3 grid with a coarser 8^3 grid representing that used in a LES. By filtering the 64^3-data with a top hat filter possessing a width L/8 the filtered velocity components and pressure at each point in the 64^3 grid are calculated. Subsequently the fluctuations in these velocity components and pressure are calculated at these same points.* The data on the fluctuations are then used to compute the subgrid stresses $u'_iu'_j$ and the contributions to the rate of strain tensor for the filtered velocity field, e.g., $\partial u_i/\partial x_j$, at each point of the 8^3 grid. Thus a direct comparison of true subgrid stresses given by the direct numerical simulation and various subgrid models is possible.

Before discussing the details of the calculations in [12] it is worth noting an interesting and illuminating by-product of this proceedure as indicated in Fig. 1 taken therefrom. Shown is the distribution of the u-velocity

* Although these calculations can be carried out for each of the fifty time steps, it appears that the detailed assessment of the subgrid models in [12] is based on the data at one time. Note that since periodic boundary conditions are imposed on the edges of the cube the filtering can be applied to all point including those on the edges.

component along a line segment of the 64^3- grid at a particular time as given by the direct numerical simulation and by filtering as in a LES. We thus see graphically for this one realization the variables u, $\bar{u}$ and u'. Clearly the filtered velocity exhibits the large scaale variations but not the small scale fluctuations whose effect is presumably described by the subgrid stresses. Such behavior is the central feature of the LES proceedure.

With a top hot filter the filtered velocity components are calculated from the relation

$$\bar{u}_r(x_{1i},x_{2j},x_{3m},t) = \frac{1}{17^3}\sum\sum\sum u_r(x'\text{subbl},x'_2,x'_3,t) \qquad (26)$$

where the sums extend over points such that $x_{1i}-\triangle \le x'_1 \le x_{1i}+\triangle$ etc. With the filtered velocities known the velocity fluctuations at each point of the 64^3 grid can be readily determined. The subgrid stresses are computed at each point of the 8^3 grid from the equation

$$\overline{u'_r u'_s}(x_{1i},x_{2j},x_{3m},t) = \frac{1}{17^3}\sum\sum\sum u'_r(\quad)\,u'_s(\quad)$$

where the summations extend over the ranges indicated in connection with Eq. (26) Finally, the contributions to the rates of strain associated with the filtered velocities are computed at each point of the 8^3 grid by differencing the filtered velocity components. Here there should be employed the same differencing scheme as is used, or intended to be used, in the LES employing the subgrid models being assessed. A similar proceedure could be carried out for the evaluation of the subgrid fluxes provided the appropriate direct numerical simulation were available.

In Clark et al [12] models for the subgrid Reynolds streses $\overline{u'_i u'_j}$ are assessed at various levels: at the tensor level, i.e., $\overline{u'_i u'_j}$ directly; at the

vector level, i.e., $\partial u'_i u'_j / \partial x_m$; and at the scalar level, i.e., comparing the true and modeled energy dissipation as given by the quantity $u_k \partial u'_k u'_l / \partial x_l$. Their assessments are expressed in terms of correlation coefficients obtained by averaging over all points of the 8^3 grid at one particular time so that 512 entries are involved.

We need give only a brief summary of the findings in [12]. Generally the Smagorinsky model which we have used as a basis of our suggestions for the LES application to premixed turbulent combustion does as well as any other model at all three levels, although at the tensor level the correlation coefficient is only about one third. The following quote is illuminating: "The modeling of the a subgrid scale Reynolds stress is not so good as to eliminate the need for improvements but neither is it so bad as to cause one to reject it out of hand; we are unable to find any model more accurate than Smagorinsky's."

An interesting aspect of the results in [12] is the finding that frequent, significant misalignments of the stress and rate of strain tensors is the principal source of errors, i.e., the cause of reductions in the correlation coefficient. The implication is that refined models for the exchange coefficient connecting these two tensors are not indicated, i.e., for example, the use of second moment averaging to improve the subgrid models should not be expected to lead to significant improvements.

The Wall Layer

Many applications of LES relate to homogeneous flows with periodic boundary conditions on the edges of cubes. Relatively little attention is devoted to wall-bounded flows with the exception of channel flows studied by Deardorff[5], Grotzbach and Schumann [6] and Kim and Moin [11]. Although Ragolla

and Moin [10] state that "(t)he specification of boundary conditions at smooth solid boundaries does not pose any difficulty," there does indeed appear to be a fundamental difficulty connected with such conditions as is clearly indicated by their subsequent discussion. Our concern can be expressed as follows: We know that the thickness of the viscous sublayer, the thin region adjacent to the wall in which molecular and turbulent transport interchange dominant roles, is several Kolmogoroff lengths in extent and is therefore not resolved in a LES. Although finer grid spacing in the neighborhood of a wall is indicated, full resolution of the viscous sublayer would imply either the assignment of excessive computing capability thereto or the restriction to low turbulent Reynolds numbers. Either alternative is inconsistent with the basic notion of a LES. Although no slip conditions are generally imposed at the wall, this practice is difficult to rationalize.

Deardorff [5] states: "In the absence of any known, rigorous formulation which would hold on each time step and at each grid point, the following boundary conditions have been found to work satisfactorily." Those conditions involve no slip in the crosswise velocity component and derivative conditions at points half grid distances from the lower and upper walls. Grotzbach and Schumann [6] use a similar ad hoc boundary condition by requiring the instantaneous wall shear to be perfectly correlated with the streamwise velocity component one mesh cell from the wall. Rogallo and Moin [10] discuss this specification and conclude that "...measurements ... support this assumption very close to the wall pvovided that a (sizable) time delay between these two quantities is introduced." The ad hoc nature of this specification is clearly evident.

Two relatively recent publications explicitly address this issue. Chapman and Kuhn [14] report on a computer experiment involving a reduced form of the Navier-Stokes equations appropriate for the description of the thin viscous sublayer subjected to periodic, two dimensional fluctuations at its outer edge. These forcing fluctuations are chosen on the basis of experimental results. It is concluded that "(t)he results ... reveal a good potential for constructing quantitative models of viscous sublayer turbulence founded on experimental observations ... " Robinson [15] describes a detailed experimental investigation of the near-wall boundary layer utilizing a skin friction meter and an array of hot wires in order to obtain instantaneous velocity profiles scaled in terms of wall values. Although interesting data, probably of future value, are obtained, they do not provide a definitive guide to the proper boundary conditions in a LES.

The Filter Scale

We find surprising the lack of specific discussion in the literature related to LES concerning the filter scales and the turbulence Reynolds numbers. In [4] it is shown that if $\triangle$ lies within the inertial subrange of the flow being studied, then the constant c_μ in the Smagorinsky subgrid model is fixed by the universal spectral constant at 0.185 which is within the range of values found to be appropriate for this constant by various other means. Even in the otherwise excellent review article by Rogallo and Moin [10] no discussion of the criteria for the selection of desireable values of $\triangle$ is given. The implication from this situation might be that the filter scale is taken to be as small as possible within the available computing capabilities. Clearly the smaller the value of $\triangle$ the smaller the reliance on the subgrid modeling and

the more nearly the LES corresponds to a direct numerical simulation.

The following considerations may be suggestive: Define u ' as a measure of the turbulent velocity fluctuations in the region of the flow with the highest turbulence Reynolds number and L as the measure of the global length scale of the flow. Then turbulence theory indicates that

$$\frac{u' L}{\nu} = \left(\frac{u' l_K}{\nu}\right)^4$$

where l_K is the Kolmogoroff length. Now the maximum dissipation occurs at length scales roughly a decade larger than l_K so that it is convenient to take $\Delta = \alpha l_K$ where α is on the order of $10 - 10^2$, the smaller values implying less reliance on the subgrid modeling. Accordingly, it is easy to show that

$$\frac{L}{\Delta} = \frac{1}{\alpha} \left(\frac{u' L}{\nu}\right)^{3/4} \qquad (27)$$

For $\alpha = 10$ this relation yields for $u' L/\mu$ of 10^3 and 10^4 values of L/Δ of 18 and 100 respectively. These values are consistent with present day computing capabilites and with current LES calculations. For example, Deardorff [5] used an ungraded mesh of 20x24x14 points. The implication of Eq. (27) is that graded meshes and adaptive griding may be effective in minimizing the errors due to subgrid closures for a given number of points. These values indicate the need for significant grading of grid size so as to concentrate points in regions of high turbulence Reynolds numbers and possibly for adaptive griding.

REFERENCES

1. Libby, P.A., Large eddy simulation of premixed turbulent combustion, Berkeley Research Associates Memorandum, 1 April 1985.

2. Libby, P.A., The equation for the progress variable in a LES of premixed turbulent combustion, Berkeley Research Associates Memorandum, 6 December 1985.

3. Herring, J.R., Subgrid scale modeling - an introduction and overview, 347-352. Eds. F. Durst et al, Turbulent Shear Flows I, Springer-Verlag, Berlin (1979).

4. Lilly, D.K., The representation of small-scale turbulence in numerical simulation experiments, in Proceedings of IBM Scientific Computing Symposium on Environmental Science (Thomas J.Watson Research Center, Yorktown Heights, 1967) 195-210.

5. Deardorff, J.W., Numerical study of turbulent channel flow, J. Fluid Mech. 41:453-480 (1970).

6. Grotzbach, G. and Schumann, U., Direct numerical simulation of turbulent velocity, pressure and temperature fields in channel flows, 370-385, Eds. F. Durst et al, Turbulent Shear Flows I, Springer-Verlag, Berlin (1979).

7. Antonopoulos-Domis, N., Large-eddy simulation of a passive scalar in isotropic turbulence, J. Fluid Mech. 104:55-79 (1981).

8. Leslie, D.C. and Quarini, G.L., The application of turbulence theory to the formulation of subgrid modelling proceedures, J. Fluid Mech. 91:65-91 (1979).

9. Leonard, A., Energy cascade in large-eddy simulations of turbulent flows. Advances in Geophysics, Academic Press, New York 237-248 (1974).

10. Rogallo, R.S. and Moin, P., Numerical simulation of turbulent flows. Annual Reviews of Fluid Mechanics, 16:99-137, Annual Reviews, Palo Alto (1984).

11. Yeh, T.T. and van Atta, C.W., Spectral transfer of scalar and velocity fields in heated grid turbulence. J. Fluid Mech. 58: 233-261 (1973).

12. Clark, R.A., Ferziger, J.H. and Reynolds, W.C., Evaluation of subgrid models using an accurately simulated turbulent flow. JFM 91:1-16 (1979).

13. Kim, J. and Moin, P., Large eddy simulation of turbulent channel flow. AGARD Symposium on Turbulent Boundary Layer - Experiment, Theory and Modeling. The Hague, Netherlands, Sept. 24-27, 1979.

14. Chapman, D.R. and Kuhn, G.D., Two-component Navier-Stokes computational model of viscous sublayer turbulence. AIAA 81-1024 (1981).

15. Robinson, S.K., An experimental search for near wall boundary conditions for large eddy simulation. AIAA 82-0963 (1982).

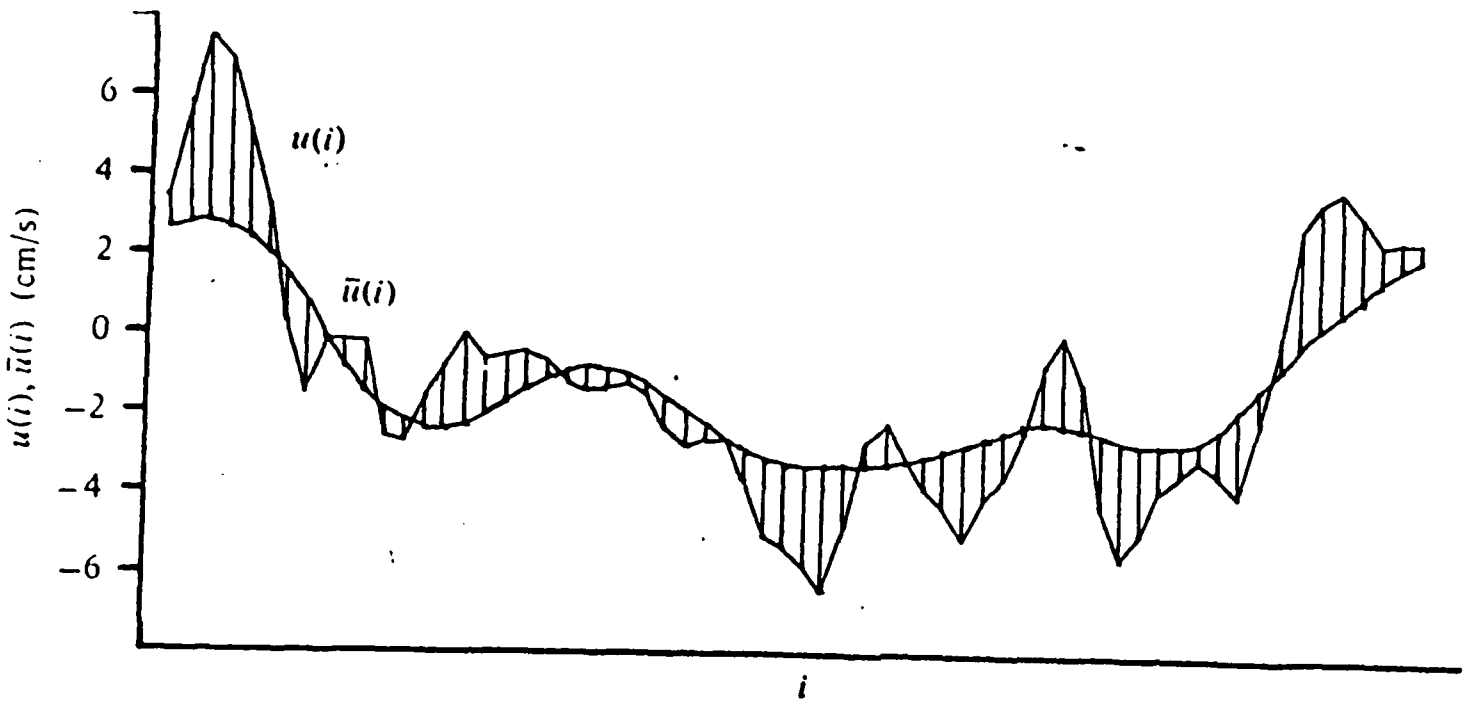


Figure 1. An unfiltered and filtered velocity component.
From reference 12.

APPENDIX I.

The Effect of Energy Release on the Regularity of Detonation Cells in Liquid Nitromethane (21st Comb. Symp., Aug. 86, Munich)

THE EFFECT OF ENERGY RELEASE ON THE REGULARITY OF DETONATION CELLS IN LIQUID NITROMETHANE

R. Guirguis
Berkeley Research Associates
Springfield, VA 22150

E.S. Oran and K. Kailasanath
Laboratory for Computational Physics
Naval Research Laboratory, Washington D.C. 20375

Subject Heading: Detonations – 4

Please send communications to:
Dr. Raafat Guirguis
Code 4040
Naval Research Laboratory
Washington, D.C. 20375

INTRODUCTION

Unsteady three-dimensional structures in detonation waves had been observed for some time before Shchelkin[1] made the first theoretical effort to relate this behavior to the instability of one-dimensional detonations. Many studies of the stability of detonations followed. Of these, Erpenbeck[2-4] presented a formal mathematical analysis that showed that one-dimensional detonations are unstable to both longitudinal and transverse perturbations. Barthel and Strehlow[5] showed that high-frequency acoustic waves behind the detonation front can produce multiple shock fronts. Abouseif and Toong[6] attributed the instability of the detonation front to the interaction between the irreversible temperature fluctuations produced by the shock and the finite reaction zone. All of these studies used a global one-step chemical reaction model. They all concluded that detonations become more stable as the degree of overdrive increases and the activation energy decreases. When the detonation is unstable, the one-dimensional structure evolves into a dynamically stable configuration: a repeatable multidimensional unsteady structure, that conforms to the geometry of the boundaries.

However, the multidimensional structure of the detonation wave is not always stable. The regularity of the cellular structures produced is usually poor, changing size from one cell to the next. The problem of irregularity has been discussed as early as 1968 by Strehlow[7]. Yet there have been no rigorous attempts to explain why detonations produce regular cellular structures in some materials and irregular in others. Attempts to relate trends in cellular regularity to the activation energy and to the sensitivity of induction time to variations in Mach number are not definitive, because they make comparisons between mixtures with different specific heats and different sound speeds[8]. Nevertheless, they conclude that regularity improves as the activation energy decreases and when the induction time is less sensitive to variations in Mach number.

In a recent paper[9], we used two-dimensional time-dependent simulations to study the effect of induction time parameters on the regularity of the cellular structure in liquid nitromethane. The chemical decomposition of nitromethane was described by a two-step model composed of an induction time followed by energy release. The

experimentally determined expression for the induction time is[10] $\tau_i^o = A_i^o \exp(E_i^o/RT)$, where $A_i^o = 2.3 \times 10^{-6}$ μs and $E_i^o = 29.1$ Kcal. The expression we used for the energy release time was[11] $\tau_r^o = A_r^o \exp(E_r^o/RT)$, where $A_r^o = 2.5 \times 10^{-9}$ μs and $E_r^o = 53.6$ Kcal.

When these values of τ_i^o and τ_r^o were used in the numerical simulations, they produced a cellular structure with poor regularity. Figure 1a, reproduced from Guirguis et al.[9], shows the resulting structure for a tube 0.05 mm wide. We then did a parametric study in which we varied A_i^o and E_i^o such that the Arrhenius line, $\log \tau_i^o$ vs $1/T$, was rotated about 2700 K. This is the temperature behind the one-dimensional detonation. It was chosen in order to keep the same detonation cell size. This is in analogy to gas phase detonations in which a correlation was found to exist between the cell size and the width of the induction zone behind the one-dimensional detonation. The expression for τ_r^o was not changed throughout the calculations.

The results of these studies showed that increasing the slope of the Arrhenius line produced more regular structures. However, it also produced large pockets of unreacted material. The calculations showed that there is a correlation between the change in induction zone thickness across the transverse waves and the regularity. The larger the change, the more regular the cellular pattern. The calculations also showed that regularity is improved when the the curvature of the shock front increases. The regular structure shown in Fig. 1b was obtained by increasing by 50% the slope of the Arrhenius line, yielding $\tau_i^o = 1.55 \times 10^{-7} \exp(43,650/RT)$ μs. It is not nearly as repeatable, however, as those patterns observed and calculated for H_2–O_2 mixtures highly diluted in argon[12].

In this paper, we study the effect of the rate of energy release and its dependence on temperature on the regularity of the cellular structure of detonations in liquid nitromethane. Here, we do not vary the expression for the induction time, but we change the energy release rate. The other input parameters in the calculations are kept constant.

PHYSICAL MODEL

The model we use has been discussed in detail by Guirguis et al.[9] We solve

the Euler equations for compressible flow in two dimensions. The walls are assumed to heavily confine the liquid explosive and boundary layer effects are neglected. The solution thus simulates the detonation structure near the center of a wide channel. The transformation of liquid fuel to gaseous products is described by a two-step model. Step 1 is an energy-neutral step, the rate of which is expressed in terms of an induction time, $\tau_i^o(T)$. Step 2 starts only after this induction time is elapsed. If f denotes the fraction of induction time elapsed at time t, and w denotes the fraction of fuel mass in a mixture of fuel and products,

$$\frac{d(fw)}{dt} = \frac{w}{\tau_i^o} , \tag{1}$$

where $f(0) = 0$, and

$$\frac{dw}{dt} = \begin{cases} 0 & \text{if } f < 1 \\ -\frac{w}{\tau_r^o} & \text{if } f \geq 1 . \end{cases} \tag{2}$$

The HOM equations of state, described by Mader[11], are used for both condensed fuel and gaseous products. The equation of state for the condensed phase is based on the method of Christian and Walsh[13]. The equation of state of the gaseous products is constructed using the BKW equation of state for the final products. In those regions which contain a mixture of phases, pressure and temperature equilibrium are assumed.

TEMPERATURE DEPENDENCE OF THE ENERGY RELEASE RATE

The different stages of reaction of a system initially at temperature T_o are illustrated in Fig. 2. At the end of the induction time, τ_i^o, the temperature increases as energy release begins. As the fuel is consumed, the increase in temperature causes an increase in the rate of energy release. Meanwhile, w decreases, causing a decrease in the rate of energy release. These two competing effects produce the inflection point on the curve, where the rate of energy release reaches a maximum. The time it takes to reach this maximum can be interpreted as an effective induction time, τ_i. If fuel consumption proceeds at a constant rate independent of w and equal to the rate at the initial temperature, T_o, the reaction follows the dashed line in Fig. 2. The time required to consume the fuel is then τ_r^o.

In all the calculations presented below, $\tau_i^o = 4.57 \times 10^{-7} \exp(37,830/RT)$ μs. This is the expression that gave the most regular structure in our previous study[9]. Figure 3 summarizes the various energy release functions, τ_r^o, we have studied. The line (0) corresponds to $\tau_r^o = 2.5 \times 10^{-9} \exp(53,600/RT)$ μs, the expression for the energy release time taken from Mader[11] and used throughout our previous study. At 2700 K, the energy release time, τ_r^o, is only about one tenth of the induction time, τ_i^o. When τ_r^o is represented by any of the dashed lines, (2), (5), or (6), the rate of fuel consumption is constant, independent of temperature and mass of fuel. In this case, the energy release process is described by the dashed curve in Fig. 2.

NUMERICAL SOLUTION

The details of the method of numerical solution have been described extensively by Guirguis et al.[9] The procedure is based on timestep splitting the fluid dynamics and the chemical terms[14]. The fluid dynamics is advanced using a fourth-order Flux-Corrected Transport (FCT) algorithm[15]. This algorithm introduces a minimal amount of numerical diffusion, thus allowing acoustic perturbations to grow naturally if the system is unstable to the wavelengths present. The smallest acoustic wavelength that can develop is limited, however, by the grid spacing. Because transverse waves develop whenever acoustic perturbations of wavelength smaller than the size of the reaction zone are introduced[5], the computational grid is designed to have at all times at least five grid points within the induction zone.

To estimate the resolution required, one-dimensional calculations were used as a guide. Such calculations showed that for a one-dimensional detonation, the induction zone is 2×10^{-4} cm wide. In our two-dimensional $x - y$ domain, we use 200×125 computational cells, initially with a uniform spacing of 4×10^{-5} cm, representing a channel 0.05 mm wide. The grid in the y-direction is kept fixed. The right boundary of the grid extends with the wave front along the x-direction, such that the grid spacing around the shock and reaction zones is always a uniform 4×10^{-5} cm. The timestep is limited to 1/4 the Courant condition, giving an average timestep of 0.8×10^{-5} μs.

RESULTS

The calculations were performed in a two-dimensional domain 0.05 mm wide representing a channel closed at one end. They were initialized with a one-dimensional overdriven detonation which decays as it propagates. Before the detonation reaches Chapman-Jouguet conditions, energy is deposited into a rectangular pocket centered on the axis behind the shock front. When the shock wave generated from the expanding pocket interacts with the detonation front, two transverse waves are formed. After a few collisions with the channel walls, the transverse waves establish the detonation front structure[9,16]. Both the energy deposited in the system to initiate the one-dimensional detonation and the energy deposited in the pocket to produce the first transverse waves are the same in all of the calculations.

We first considered the "square-wave" case in which the energy is released instantaneously at the end of the induction period. In this calculation, although two transverse waves formed as a result of the interaction between the detonation front and the shock wave generated from the pocket, the detonation front became one-dimensional early in the calculation. Erpenbeck,[17] Fickett,[18] and Abouseif and Toong[6] used perturbation analysis to study the stability of the square-wave detonation. Erpenbeck and Fickett concluded that such a model results in an infinite set of unstable modes, with amplification rates increasing with frequency. Abouseif and Toong, on the other hand, concluded that the degree of instability is reduced rapidly as the frequency is increased, so that the one-dimensional detonation is stable at high frequencies. The results of our numerical study support the conclusion of Abouseif and Toong.

We then investiaged the effects of decreasing the activation energy, E_r^o, using line (0) in Fig. 3 as the base case. Lines (1) and (2) have the same value for the energy release time at 2700 K, $\tau_r^o = 5 \times 10^{-5}$ μs, as line (0), but different activation energies. For line (1), $E_r^o = 26.8$ Kcal. Line (2) corresponds to a constant rate of energy release, $E_r^o = 0$.

Figure 4 compares the pressure and temperature contours resulting from using lines (0), (1), and (2). A heavy concentration of pressure contours occurs at both shock

and reaction fronts. The reaction front is particularly well defined on the temperature contours. When the temperature behind the shock front is below the selected minimum contour level, the shock front does not show on the temperature contours. This appears in the figures as gaps ahead of the reaction front. The region confined between the shock and reaction front is the induction zone. At a triple point, there is a change in induction zone thickness and a slip line originates.

We notice several trends as we decrease the activation energy. First, as the activation energy is reduced from 53.6 to 26.8 Kcal, going from line (0) to line (1), the change in induction zone thickness at the triple points increases. Larger unreacted pockets form. The detonation front becomes more curved and the detonation structure becomes more regular.

Now we reduce the activation energy to zero, in which case τ_r^o becomes independent of temperature, corresponding to line (2). The front is nearly one-dimensional and very little change is noticed in the size of the induction zone at the triple points. Of the three cases, line (1) leads to the most regular cellular structure.

We have also investigated the effect of changing the magnitude of the energy release rate by repeating the calculations using energy release times five times shorter and five times longer. These are shown in Fig. 3 as lines (3) and (4), respectively. For the shorter energy release times, there is less change in induction zone thickness at the triple points as shown in Fig. 5. At step 3000, a new triple point starts forming at the lower wall. This new triple point disappears in steps 3200 and 3400, but its temporary formation demonstrates a decline in the cell structure regularity.

The longer energy release times, line (4) in Fig. 3, caused large changes in the size of the induction zone at the triple points. The result is that very large pockets formed. The front was highly curved. At later stages, the large pockets of unreacted material caused the detonation to weaken to the point where the shock front decoupled from the reaction zone, and the detonation wave eventually died out.

For line (4) in Fig. 3, τ_r^o is longer than τ_i^o at temperatures lower than 2175 K. The resulting detonation wave died out. To investigate the effect of the relative magnitude

of induction and energy release times, we repeated the calculations for two temperature-independent energy release rates, described by lines (5) and (6). For line (5), τ_r^o is half the induction time at 2700 K. Reducing the activation energy to zero, going from line (4) to (5) showed the same trend described above. The change in the width of the induction zone across the transverse waves was slightly reduced. Smaller unreacted pockets were formed. The shock front was slightly less curved. Although these conditions are usually associated with a less regular structure, the smaller pockets did not cause the detonation wave to die out.

The pressure and temperature contours of the detonation wave corresponding to line (5) are shown in Fig. 6. These contours indicate a wider induction zone and a more curved shock front than those in Fig. 4. The pressure contours at steps 2000 and 2800 are similar except for an upward or downward displacement equal to half the width of the channel, showing that the detonation wave has gone through one half of a cell length. At step 2800, the two tansverse waves are about to interact near the center of the channel. The interaction produces a new Mach stem and a region of high temperature behind it. Due to the corresponding decrease in induction time, a new reaction front is initiated near the center that isolates the pocket of unreacted material shown by the temperature contours at step 3000. Other possible explanations for the formation of unreacted pockets were given by Oran et al.[19]

Finally, for line (6), the energy release time is longer than the induction time at all temperatures higher than 2700 K. Again, large unreacted pockets formed that led the reaction front to eventually decouple from the leading shock, and the detonation to die out.

DISCUSSION AND CONCLUSIONS

In this paper, we have presented the results of time-dependent two-dimensional numerical studies of the effect of the rate of energy release on the regularity of the detonation cell structure in liquid nitromethane. In a previous work[9], we investigated the effect of the induction time on the regularity of the detonation structure. In the work presented in this paper, the same expression describes the induction time for all

calculations, but we vary the rate of energy release.

The regularity of the detonation cells is generally classified as excellent, good, poor, or irregular, depending on the uniformity of the pattern inscribed by the triple points[20,21]. In terms of this classification, the structures we see in our calculations showed either good or poor regularity. When the curvature of the shock was well defined throughout the front and changed only at the two triple points, and when the detonation structure was reasonably symmetrical and repeatable from cell to cell, we call the regularity good. This was the case for the calculations shown in Fig. 4a using lines (0) and (1), and for those in Fig. 6 using line (5). Weakly curved fronts, on the other hand, showed local changes in curvature that were of the same size or even larger than the change at the two main triple points. This is the case for the calculations in Fig. 4 which used line (2), and in Fig. 5 which used line (3). We call the regularity of these structures poor.

In the calculations presented above, the detonation wave either died out, became one-dimensional, or inscribed a generally regular cellular structure. We conclude that the rate of energy release controls whether the detonation wave stays one-dimensional or becomes multidimensional. But when the detonation is multidimensional, the regularity of the cellular structure is controlled mainly by the temperature dependence of the induction time. The rate of energy release can affect the regularity, but to a lesser extent. Longer energy release times produce more regular structures. However, if the energy release times become too long, the detonation dies out. Decreasing the activation energy also leads to more regular structures. But reducing it below a certain value makes the structure less regular again.

One way in which changes in the activation energy can affect the regularity is by changing the effective induction time τ_i, defined in Fig. 2. To investigate this, we solved Eq. (2) for a constant volume system for which $\tau_i^o = 0$, using two activation energies, 50 and 25 Kcal. We found that for temperatures larger than about 2600 K, the slope of $\log \tau_i$ vs $1/T$ for 25 Kcal is larger than the slope of the corresponding curve for 50 Kcal. For temperatures less than 2600 K, 50 Kcal produces larger slopes. The

crossover temperature, 2600 K, at which the two slopes are equal, depends on the values of the two activation energies. We have shown[9] that increasing the slope of $\log \tau_i^o$ vs $1/T$ produces more regular structures. Thus, reducing the activation energy up to a limit improves the regularity. If the activation energy is further reduced, the crossover temperature becomes so high that the slope of the effective induction time is lower throughout the whole range of temperatures behind the shock front of the detonation wave. The cellular structure then becomes less regular.

When the energy release time is much shorter than the induction time, changes in the rate of energy release can affect the regularity of the cell structure. However, when the energy release and induction times are comparable, the effective induction time, τ_i, becomes significantly different from the steady induction time, τ_i^o. In this case, the detonation cell size significantly deviates from its nominal value and the slow energy release can even cause the detonation to die out.

The results of the calculations are consistent with the conclusions of our study of the effect of induction time on regularity[9]. The stability of the multidimensional detonation depends on the difference between the thermodynamic properties of the induction zones behind the Mach stem and the incident shock sections of the front. If these two zones are not very distinct, small disturbances affect the system and irregular structures result. The difference between the thermodynamic properties of these two zones can be increased by increasing the temperature dependence of the induction time, increasing the energy release time, or decreasing the activation energy. Each of these changes, or any combinations of them, yields a more curved shock front. The triple points are then sharply defined at all times, and the structure becomes more regular. However, as the difference between the properties of the two zones is increased, large unreacted pockets form. These cause a deficit in the energy released behind the shock front, which is the energy that supports the detonation wave. If this deficit becomes too large, the reaction front decouples from the shock and the detonation wave eventually dies out.

ACKNOWLEDGEMENT

This work was supported by the Naval Research Laboratory, Special Focus Program on Controlled Energy Release, through the Office of Naval Research. The authors would like to acknowledge the help of E.S. Gold in preparing the figures.

REFERENCES

1. Shchelhin, K.I.: Dokl. Akad. Nauk S.S.S.R. 160, 1144 (1965)
2. Erpenbeck, J.J.: Phys. Fluids 7, 684 (1964).
3. Erpenbeck, J.J.: Phys. Fluids 8, 1192 (1965).
4. Erpenbeck, J.J.: Phys. Fluids 9, 1293 (1966).
5. Barthel, H.O. and Strehlow, R.A.: Phys. Fluids 9, 1896 (1966).
6. Abouseif, G.E. and Toong, T.Y.: Comb. Flame 45, 67 (1982).
7. Strehlow, R.A.: Comb. Flame 12, 81 (1968).
8. Moen, I.O., Sulmistras, A., Thomas, G.O., Bjerketvedt, D., and Thibault, P.A.: *The Influence of Cellular Regularity on the Behavior of Gaseous Detonations.* Paper presented at the tenth ICDERS, Berkeley, California, Aug. (1985).
9. Guirguis, R.H., Oran, E.S., and Kailasanath, K.: *Numerical Simulations of the Cellular Structure of Detonations in Liquid Nitromethane — Regularity of the Cell Structure.* To appear in Comb. Flame (1986).
10. Chaiken, R.F.: *Correlation of Shock Pressure, Shock Temperature, and Detonation Induction Time in Nitromethane, Proceedings of the High Dynamic Pressure Symposium,* Commissariat d'Energie Atomique, Paris, France (1978).
11. Mader, C.L.: Numerical Modeling of Detonations, University of California Press, 1979.
12. Kailasanath, K., Oran, E.S., Boris, J.P., and Young, T.R.: Comb. Flame 61, 199 (1985).
13. Walsh, J.M. and Christian, R.H.: Phys. Rev. 97, 1544 (1955).
14. Oran, E.S. and Boris, J.P.: Prog. Ener. Combust. Sci. 7, 1, (1982).

15. Boris, J.P.: *Flux-Corrected Transport Modules for Solving Generalized Continuity Equations*, Naval Research Laboratory Memo. Rep. 3237, 1976.

16. Kailasanath, K., Oran, E.S., Boris, J.P., and Young, T.R.: *A Computational Method for Determining Detonation Cell size*, AIAA paper 85-0236, AIAA 23rd Aerospace Sciences Meeting, Jan., 1985.

17. Erpenbeck, J.J.: *Nineth Symposium (International) on Combustion*, p. 442, The Combustion Institute, 1963.

18. Fickett, W.: Physica D, 16, 358 (1985).

19. Oran, E.S., Young, T.R., Boris, J.P., Picone, J.M., and Edwards, D. H.: *Nineth Symposium (International) on Combustion*, p. 573, The Combustion Institute, 1982.

20. Strehlow, R.A.: Fundamentals of Combustion, Krieger Publishing Co., 1979; also Combustion Fundamentals, McGraw Hill, 1984.

21. Fickett, W. and Davis, W.C.: Detonation, University of Califfornia Press, 1979.

FIGURE CAPTIONS

Fig. 1. Pressure contours of a detonation wave propagating in a channel 0.05 mm wide. Solid traces are loci of main triple points (points at which shock front sharply changes curvature). Dashed traces are loci of secondary triple points (minor change in front curvature, recognized by associated transverse waves). Collection of solid and dashed traces define a cellular structure with (a) poor regularity, and (b) good regularity.

Fig. 2. Time evolution of temperature for a typical chemical reaction. Dashed line corresponds to a case in which the rate of energy release is constant.

Fig. 3. Energy release functions, τ_r^o vs $1/T$, used in the numerical simulations. Dashed lines represent constant energy release rates.

Fig. 4 Pressure (a) and temperature (b) contours from simulations using lines (0), (1), and (2) in Fig. 3, for a channel 0.05 mm wide.

Fig. 5. Pressure contours from simulations using line (3) in Fig. 3, for a channel 0.05 mm wide.

Fig. 6. Pressure and temperature contours from simulations using line (5) in Fig. 3, for a channel 0.05 mm wide.

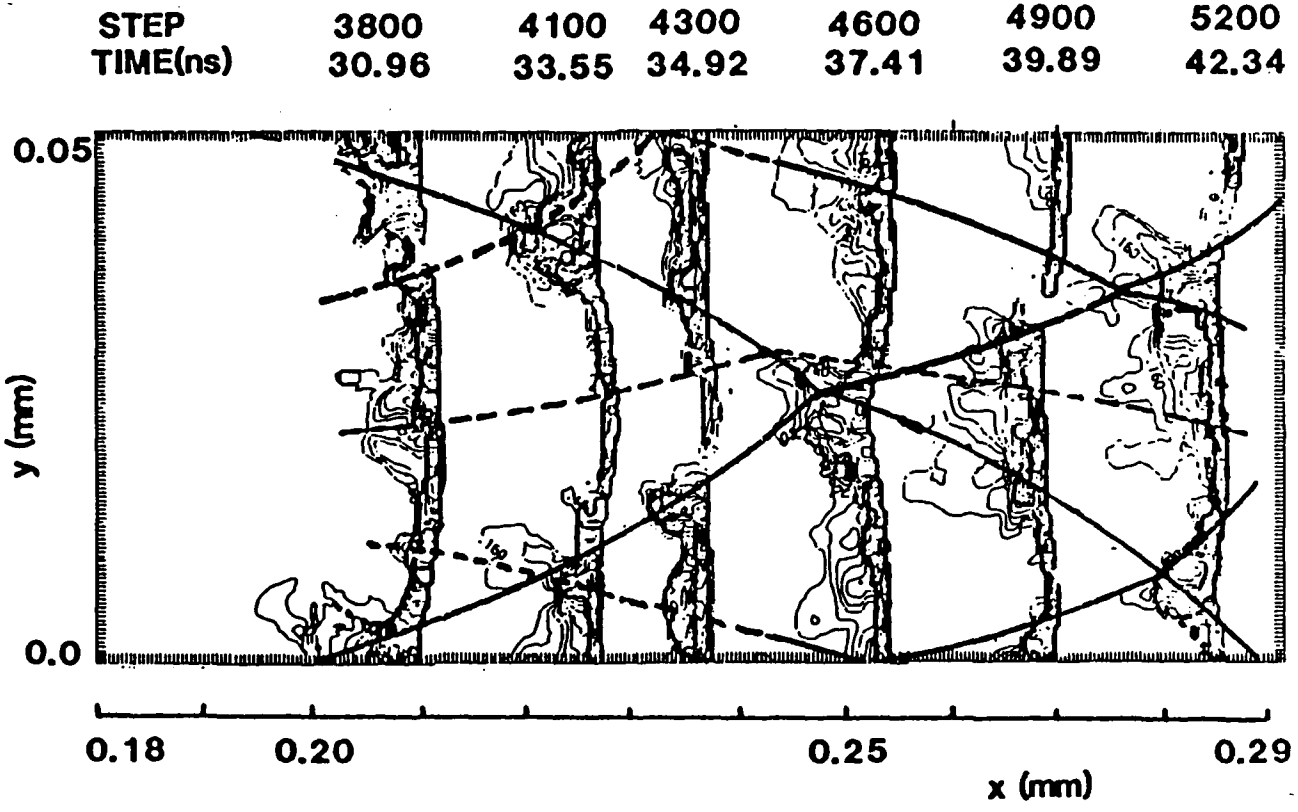

STEP 3800 4100 4300 4600 4900 5200
TIME(ns) 30.96 33.55 34.92 37.41 39.89 42.34
0.05
0.0
y (mm)
0.18
0.20
0.25
0.29
x (mm)

Fig. 1a

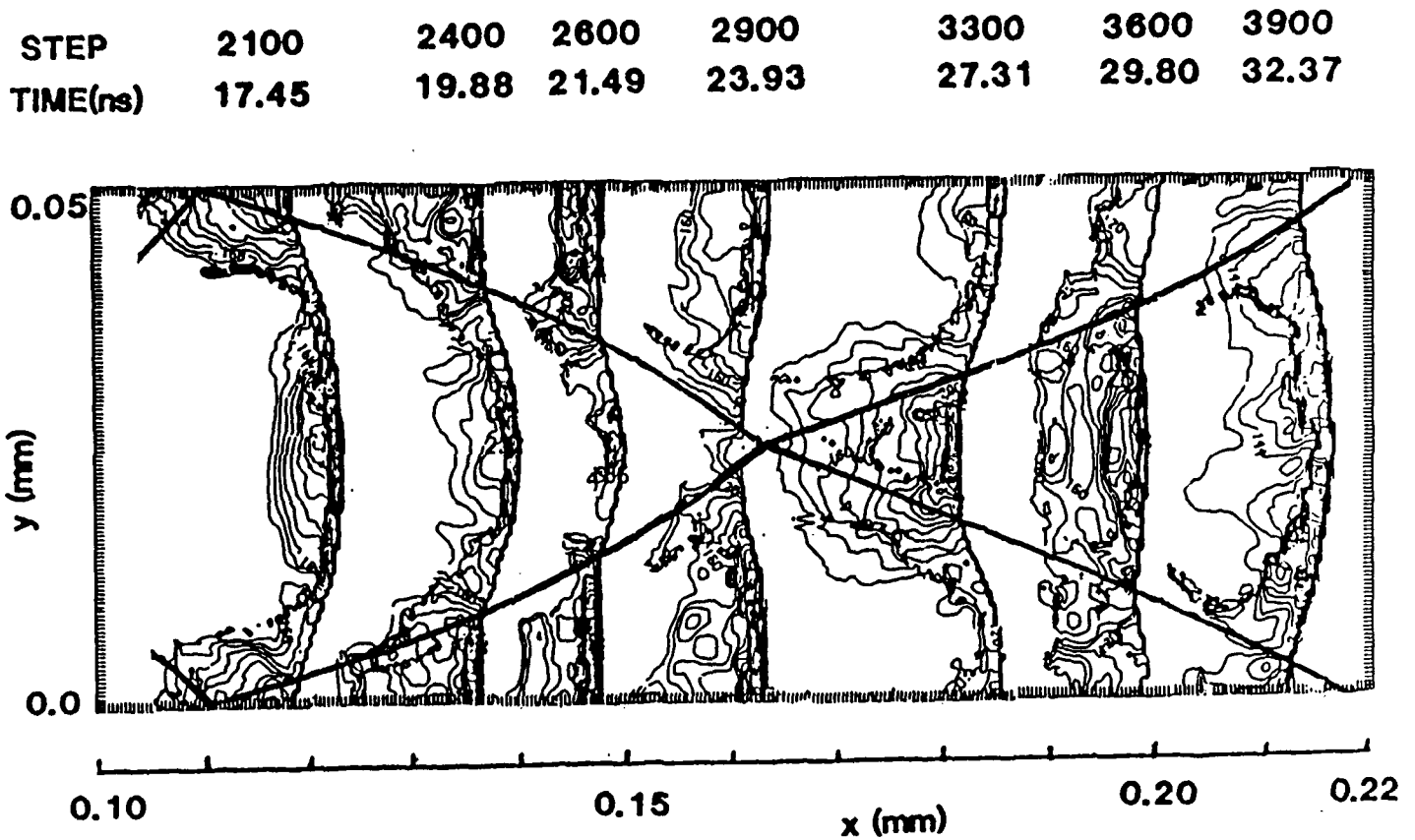

STEP 2100 2400 2600 2900 3300 3600 3900
TIME(ns) 17.45 19.88 21.49 23.93 27.31 29.80 32.37
0.05
0.0
y (mm)
0.10
0.15
0.20
0.22
x (mm)

Fig. 1b

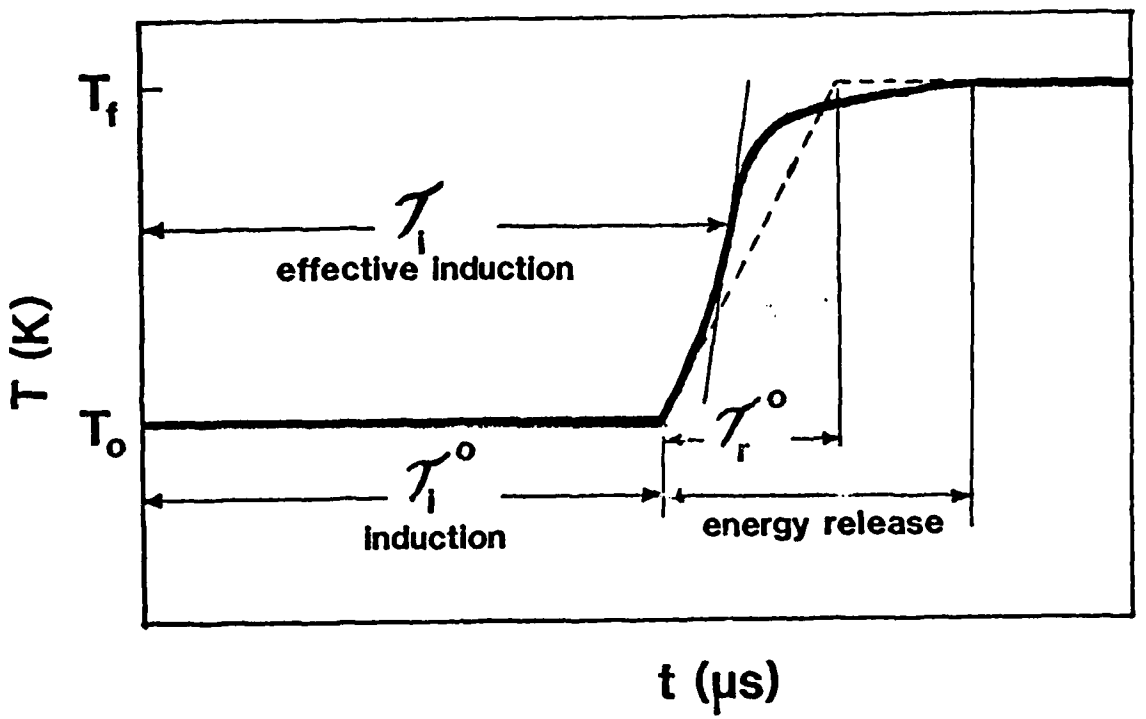

T_f
T_o
T (K)
τ_i
effective induction
τ_i^o
induction
τ_r^o
energy release
t (μs)

Fig. 2

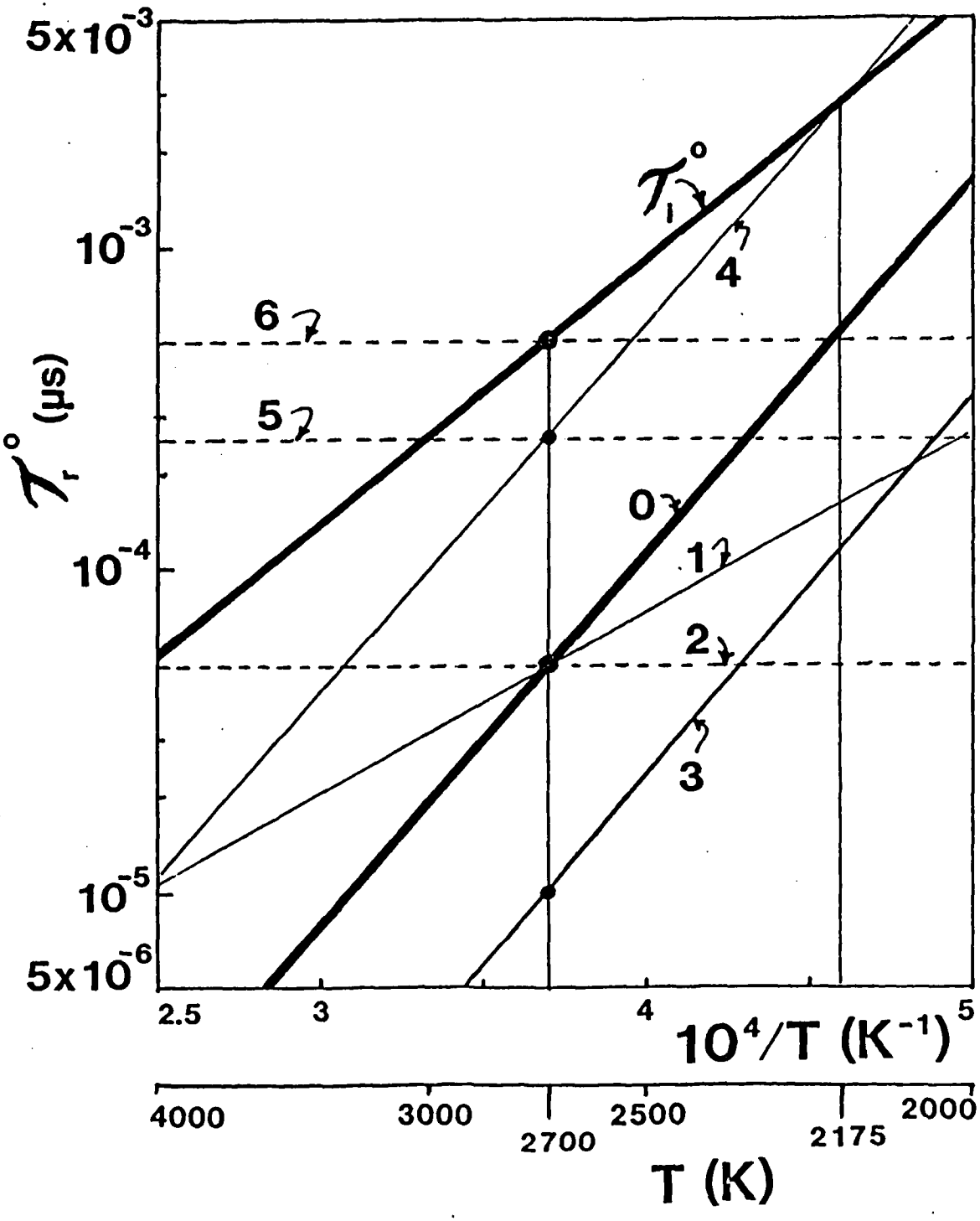

5x10^-3
10^-3
10^-4
10^-5
5x10^-6
τ° (μs)
τ1°
6
5
4
0
1
2
3
2.5
3
4
5
10^4/T (K^-1)
4000
3000
2700
2500
2175
2000
T (K)

Fig. 3

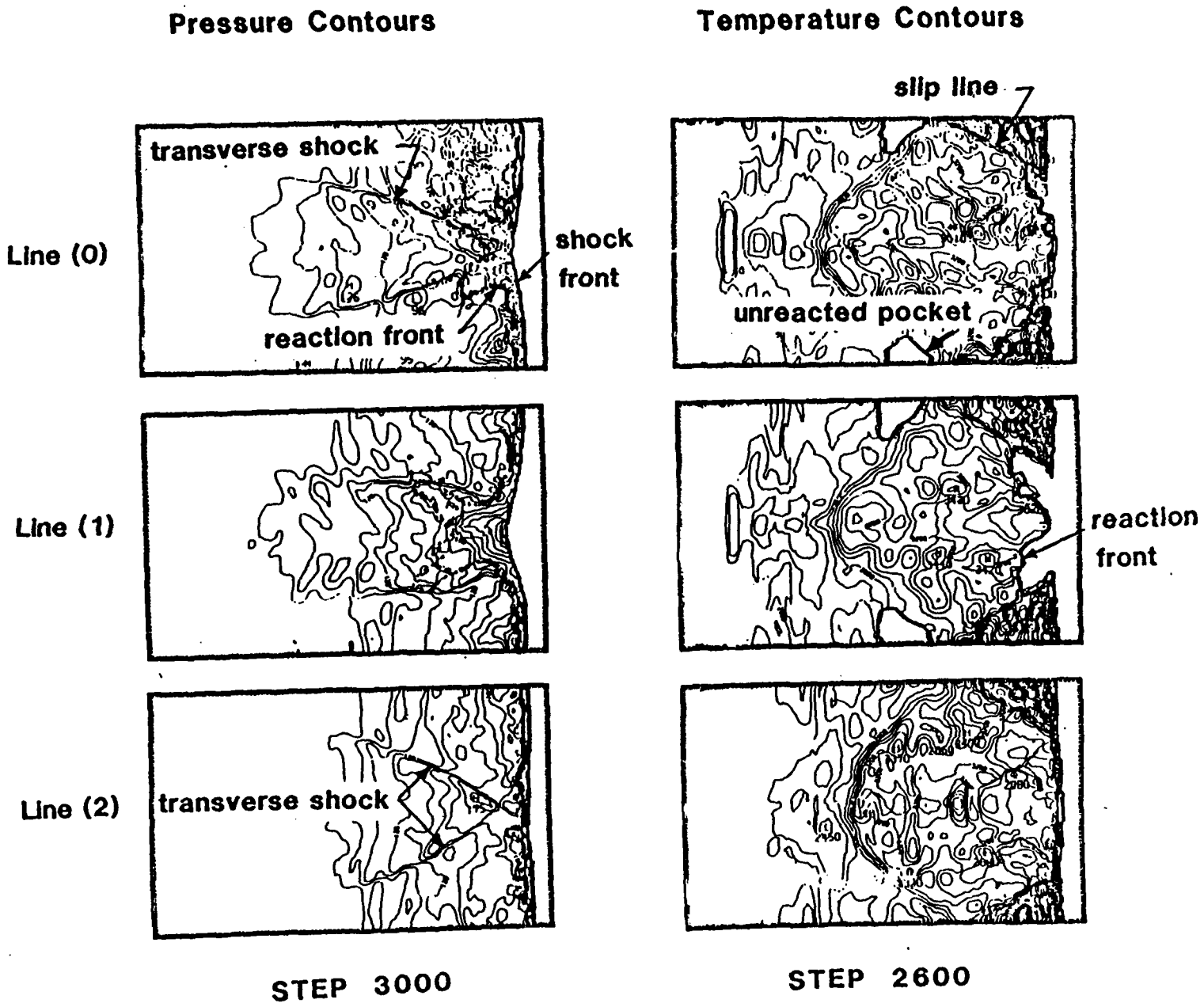
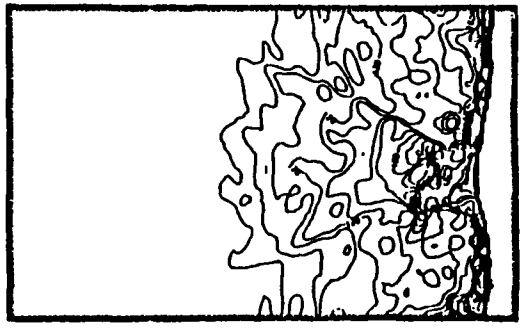

Pressure Contours
Temperature Contours
slip line
transverse shock
shock
front
Line (0)
reaction front
unreacted pocket
Line (1)
reaction
front
Line (2)
transverse shock
STEP 3000
STEP 2600

Fig. 4

Pressure Contours

Line (3)

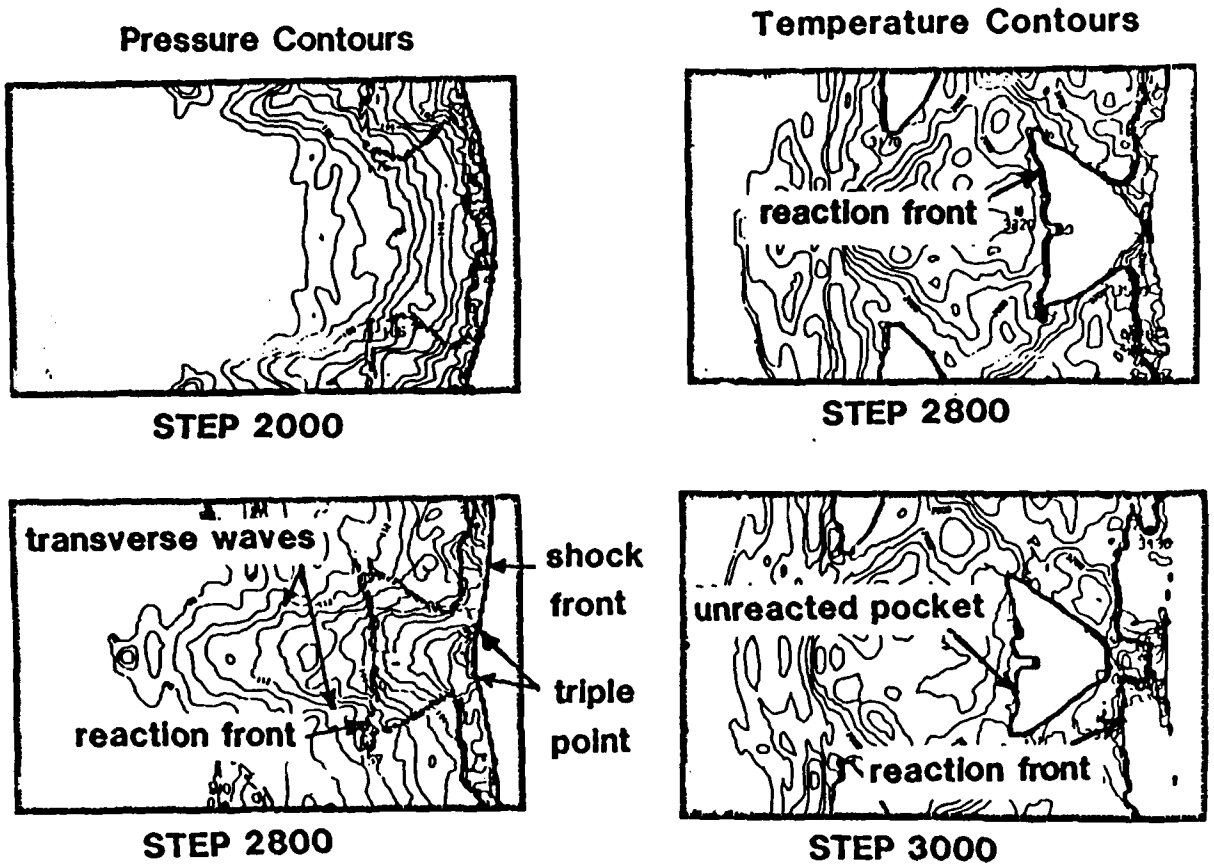


STEP 3000

STEP 3200

STEP 3400

Fig. 5


Pressure Contours
Temperature Contours
reaction front
STEP 2000
STEP 2800
transverse waves
shock
front
triple
point
reaction front
unreacted pocket
reaction front
STEP 2800
STEP 3000

Line (5)

Fig. 6

APPENDIX J.

A Barely Implicit Correction for Flux-Corrected Transport

Reprinted from JOURNAL OF COMPUTATIONAL PHYSICS
All Rights Reserved by Academic Press, New York and London

Vol. 71, No. 1, July 1987
Printed in Belgium

A Barely Implicit Correction for Flux-Corrected Transport

G. PATNAIK*, R. H. GUIRGUIS*, J. P. BORIS, AND E. S. ORAN

Laboratory for Computational Physics, Naval Research Laboratory, Washington, D. C. 20375

Received April 11, 1986; revised August 12, 1986

The barely implicit correction (BIC) removes the stringent limit on the timestep imposed by the sound speed in explicit methods. This is done by adding one elliptic equation which has to be solved implicitly. BIC is combined with the flux-corrected transport algorithm in order to represent sharp gradients in subsonic flows accurately. The resultant conservative algorithm costs about the same per timestep as a single explicit timestep calculated using an optimized FCT module. Several examples show the technique's ability to solve nearly incompressible flows very economically.

I. INTRODUCTION

The solution of time-dependent compressible flow problems is complicated by conflicting requirements of mathematical accuracy, nonlinearity, physical conservation, and positivity. This is especially true near discontinuities where "accurate" high-order algorithms produce ripples while linear monotonic (i.e., positivity-preserving) schemes are highly diffusive. After Godunov [1] showed that a linear algorithm ensures positivity only if it is first order, the next logical step was to look at nonlinear methods to develop effectively higher order, more accurate monotonic schemes. The first high-order monotone algorithm (Boris [2]) was designed to maintain local positivity near steep gradients while keeping a high order of accuracy elsewhere. The major principles of the monotone high-order algorithms are that they maintain positivity through a procedure that uses a nonlinear combination of diffusive and antidiffusive fluxes. The flux-corrected transport (FCT) algorithm that we use in this paper [3, 4] is made fourth order by the appropriate subtraction of corrected fluxes. Other monotone methods have been reviewed by Woodward and Collela [5] and Baer [6]. In this paper we confine our discussions to a barely implicit correction (BIC) to FCT. BIC is also extendable to other monotone methods.

Positivity-preserving monotone FCT methods were developed to calculate shocks accurately. Even for subsonic flows with discontinuities, their high accuracy produced much better solutions than standard finite difference techniques. The early

* Berkeley Research Associates, Springfield, VA 22150.

0021-9991/87 $3.00

FCT methods were explicit. No serious limitation arose from this explicitness in supersonic flows because the major features of interest in the flow move at about the sound speed. Using these methods for subsonic flows, however, is economical only if the characteristic velocities in the flow field are a reasonable fraction of the speed of sound [7, 8] or if the fast sound waves are mathematically removed from the system of equations.

The barely implicit correction described in this paper was motivated by the need to calculate subsonic flows accurately in which the velocities of the important flow structures are much lower than the speed of sound. In typical cases, we are interested in flow velocities from centimeters to tens of meters per second. These flow velocities are encountered, for example, in laminar flames and low-speed fuel injection in engines. Our objectives are to remove the timestep limit imposed by the speed of sound, retain the accuracy required to resolve the detailed features of the flow, and reduce the computational costs.

The obvious way to beat the sound-speed limit on the timestep is to make the calculation implicit. This has been done successfully for many linear methods, such as the MacCormack method [9], the Beam and Warming method [10], and the semi-implicit ICE method [11–13]. In addition, recent developments have been reported for implicit, nonlinear PPM [14] and TVD [15] methods. A major problem with these methods is that they are relatively expensive, even though they can be made relatively accurate.

Another approach is the asymptotic methods. Examples of these are the methods developed by Jones and Boris [16], Rehm and Baum [17], and Paolucci [18]. In these methods, the only effects of compression that are allowed are the changes in density due to heating or cooling. Pressure fluctuations are filtered out, thus removing the timestep limit imposed by the sound speed. However, other effects from sound waves are removed in this process.

As a useful approach was given by Casulli and Greenspan [19]. Their analysis indicated that it is not necessary to treat all of the terms in the gas dynamic equations implicitly to be able to use longer timesteps than those dictated by explicit stability limits. Only those explicit terms which force this limit need to be treated implicitly. This approach results in a single elliptic equation for pressure. Because of the choice of terms, the algorithm produced is stable. Note that the ICE method also results in a single elliptic equation. However, the elliptic equation in ICE is different, and does not completely eliminate the sound-speed restriction.

The conservative algorithm presented in this paper has two steps. The first step is explicit. It is performed at a large timestep governed by a CFL condition on the fluid velocity. This step should be done with an accurate nonlinear monotone method, and we have used FCT in the examples given. The second step is an implicit correction step requiring the solution of one elliptic equation for the pressure correction. The term barely implicit correction emphasizes our use of the idea of Casulli and Greenspan, that only certain terms must be treated implicitly.

The total cost per timestep of BIC–FCT is about the same as for a full explicit FCT step. Thus the cost of a complete calculation is one or two orders of

magnitude below that required if a very slow flow were treated explicitly. Since only one elliptic equation is solved, the method is considerably faster than many implicit methods commonly used. In addition, using a nonlinear monotone method for the explicit step ensures high accuracy.

II. Method of Solution

Derivation of the Barely Implicit Correction

We are solving the compressible gas dynamics conservation equations for density ρ, momentum density $\rho\mathbf{v}$, and total internal energy, E,

$$\frac{\partial \rho}{\partial t} = -\nabla \cdot \rho \mathbf{v}, \tag{1}$$

$$\frac{\partial \rho \mathbf{v}}{\partial t} = -\nabla \cdot \rho \mathbf{v}\mathbf{v} - \nabla P, \tag{2}$$

$$\frac{\partial E}{\partial t} = -\nabla \cdot (E + P)\,\mathbf{v}, \tag{3}$$

where the total energy density E is

$$E = \varepsilon + \tfrac{1}{2}\rho \mathbf{v}^2. \tag{4}$$

The equation of state relating pressure and internal energy is

$$P = (\gamma - 1)\,\varepsilon. \tag{5}$$

In their recent paper, Casulli and Greenspan [19] showed that it is not necessary to treat every term in a finite-difference algorithm implicitly to avoid the timestep constraint imposed by the Courant condition. Further, they showed that only the pressure in Eq. (2) and the velocity in Eq. (3) must be treated implicitly. Their paper provides the starting concepts for the work we present. In addition, we have extended their analysis to include an implicitness parameter, ω, that can be used to vary the degree of implicitness of the algorithm. In general, we can have $0.5 \leqslant \omega \leqslant 1$, where the implicit terms are centered in time for $\omega = 0.5$. For $\omega < 0.5$, the method is found to be unstable for sufficiently large timesteps.

There are two stages to the algorithm. One stage is an explicit predictor that determines $\tilde{\rho}$ and the provisional value $\tilde{\mathbf{v}}$,

$$\frac{\tilde{\rho} - \rho^{\circ}}{\Delta t} = -\nabla \cdot \rho^{\circ}\mathbf{v}^{\circ}, \tag{6}$$

$$\frac{\tilde{\rho}\tilde{\mathbf{v}} - \rho^{\circ}\mathbf{v}^{\circ}}{\Delta t} = -\nabla \cdot \rho^{\circ}\mathbf{v}^{\circ}\mathbf{v}^{\circ} - \nabla P^{\circ}, \tag{7}$$

The tilde denotes predictor values at the new time, and the superscripts o and n are used to denote the old time and new time, respectively. So far only time has been differenced, not space. The implicit forms of Eqs. (2) and (3) are

$$\frac{\rho^n \mathbf{v}^n - \rho^o \mathbf{v}^o}{\Delta t} = -\nabla \cdot \rho^o \mathbf{v}^o \mathbf{v}^o - \nabla[\omega P^n + (1-\omega) P^o], \tag{8}$$

$$\frac{E^n - E^o}{\Delta t} = -\nabla \cdot (E^{\cdot} \quad {}^{P^o})[\omega \mathbf{v}^n + (1-\omega) \mathbf{v}^o], \tag{9}$$

where ω is the implicitness parameter discussed above. When $\omega = 1$, the algorithm is completely implicit and reverts to the original equations analyzed by Casulli and Greenspan.

We can reduce this implicit system to only one equation by eliminating $\mathbf{v}^n$ between Eq. (8) and (9). To do this, we first define the change in pressure, δP, as

$$\delta P \equiv \omega(P^n - P^o). \tag{10}$$

Then the correction equation for momentum can be obtained in terms of δP by subtracting Eq. (7) from Eq. (8),

$$\frac{\rho^n \mathbf{v}^n - \tilde{\rho}\tilde{\mathbf{v}}}{\Delta t} \approx -\nabla\omega(P^n - P^o) = -\nabla\delta P. \tag{11}$$

We obtain the new velocity by rearranging Eq. (11) and letting $\rho^n = \tilde{\rho}$ because the density is treated explicitly. Then

$$\mathbf{v}^n = -\frac{\Delta t}{\tilde{\rho}} \nabla \delta P + \tilde{\mathbf{v}}. \tag{12}$$

We obtain a correction equation for energy using the equation of state with γ constant, Eq. (4),

$$\varepsilon^n = \frac{\delta P}{(\gamma - 1)\omega} + \varepsilon^o, \tag{13}$$

where the ω factor appears from the definition of δP. We find δP by substituting Eqs. (12) and (13) into Eq. (9),

$$\begin{aligned} \frac{\tilde{\rho}\tilde{\mathbf{v}}^2 - \rho^o \mathbf{v}^{o2}}{2\Delta t} + \frac{\delta P}{(\gamma - 1)\omega \Delta t} = {} & \omega \Delta t \nabla \cdot \left(\frac{E^o + P^o}{\tilde{\rho}}\right) \nabla \delta P \\ & - \omega \nabla \cdot (E^o + P^o)\tilde{\mathbf{v}} \\ & - (1-\omega)\nabla \cdot (E^o + P^o)\mathbf{v}^o. \end{aligned} \tag{14}$$

Note that the kinetic energy change is included explicitly. For convenience, we define the quantity $\bar{E}$,

$$\frac{\bar{E}-E^{\circ}}{\Delta t} \equiv -\nabla\cdot(E^{\circ}+P^{\circ})[\omega\tilde{\mathbf{v}}+(1-\omega)\,\mathbf{v}^{\circ}]. \tag{15}$$

This allows us to rewrite Eq. (14),

$$\frac{\delta P}{(\gamma-1)\,\omega\Delta t}-\omega\Delta t\nabla\cdot\left(\frac{E^{\circ}+P^{\circ}}{\tilde{\rho}}\right)\nabla\delta P=\frac{\bar{E}-E^{\circ}}{\Delta t}-\frac{\tilde{\rho}\tilde{\mathbf{v}}^{2}-\rho^{\circ}\mathbf{v}^{\circ 2}}{2\Delta t}, \tag{16}$$

which provides us with an elliptic equation for δP. The right-hand side of Eq. (16) is evaluated explicitly using Eq. (15). After the elliptic equation is solved for δP, momentum and energy are corrected by Eqs. (11) and (13). Note that we started with two equations with implicit terms, and now we have reduced it to one equation, Eq. (16).

The barely implicit correction is carried out in three stages. In the first, Eqs. (6), (7), and (15) are integrated with any one-step explicit method. The pressure correction equation, Eq. (16) is solved by an elliptic solver in the second stage. The last stage requires the use of Eqs. (11) and (13) to obtain the final values of momentum and energy at the new timestep.

Solution Procedure

The derivation given above does not involve any specific choice of method for differencing the spatial derivatives. The only restriction so far is that the spatial derivatives must be evaluated at the appropriate time levels indicated by the superscripts. This allows great flexibility in the choice of the differencing scheme for these terms. Thus we can integrate the explicit predictor equations, Eqs. (6), (7), and (15) with FCT. This gives us the benefits of using a high-order monotone method. We have given the name BIC–FCT to this particular combination of BIC and FCT. Tests, such as those presented below, indicate that it has the same accuracy and flexibility as FCT.

At each timestep, the solution procedure we have implemented is divided into the three states:

(1) Explicit predictor stage. The density and momentum are advanced explicitly as specified by Eqs. (6) and (7) using FCT. This produces the intermediate quantities, $\tilde{\rho}$ and $\tilde{\rho}\tilde{\mathbf{v}}$. The $\tilde{\mathbf{v}}$ is found from $\tilde{\rho}\tilde{\mathbf{v}}/\tilde{\rho}$. Then $\tilde{\mathbf{v}}$ is used to obtain $\bar{E}$ given by Eq. (15). FCT is also used to obtain $\bar{E}$.

(2) Solution of Eq. (16) for δP. In one dimension, the solution to the difference form of Eq. (16) requires the solution of a system of linear equations by a tridiagonal matrix solver. In two dimensions, the solution requires an elliptic solver. For the two-dimensional calculations shown below, we used a multigrid method [20]. A substantial part of the computer time required in this stage is in setting up the coefficients for an elliptic equation solver.

(3). Momentum and energy corrections. These corrections are obtained from the pressure change δP using Eqs. (11) and (13), respectively. These corrected values and the density obtained explicitly in the first stage are the starting conditions at the new timestep.

These three stages are carried out at every timestep. The derivatives involving pressure in the pressure difference equation, Eq. (16), are approximated by central differences. All physical quantities are calculated at cell centers, and those values needed at cell interfaces are obtained by averaging.

This technique can be implemented in one, two, or three dimensions. In one and two dimensions, several different geometries are possible. For example, we have implemented two-dimensional planar and axisymmetric geometries, and one-dimensional Cartesian, cylindrical, and spherical. Any boundary conditions that are commonly used with the standard FCT modules can be used with this algorithm [7].

Boundary conditions for the elliptic pressure correction equation are needed. Symmetry or outflow boundaries can be simulated by a Neumann condition on the pressure correction. At an inflow, the pressure is related to the internal energy by the equation of state. Thus the boundary condition for the pressure correction can be derived from the boundary condition on energy. If the internal energy is fixed at a boundary, the pressure there is a constant and thus δP is zero.

III. Tests of the Method

Advection of a One-Dimensional Contact Discontinuity

The problem we consider first is the flow of air through a duct in one dimension. The duct is initially filled with air at standard temperature and pressure. Then cold air with twice the density flows into the duct. There is a contact discontinuity at the location where the cold, dense air and normal air meet. In the absence of diffusive processes, the contact discontinuity should move at the velocity of the incoming air. This numerical test shows the ability of BIC–FCT to propagate a contact discontinuity with the same accuracy as FCT.

The computational domain was divided into 200 evenly spaced cells of 1 cm. Initially, the discontinuity was 0.1 m from the inlet. The flow velocity of air in the duct was 10 m/s. The inlet conditions corresponding to the cold air are held fixed throughout the calculation.

The timestep used in this calculation is 0.5 ms, which is the time required for the fluid to cross half a cell. This should be compared to the Courant limit of 24 μs. Typical explicit methods are forced to employ a timestep of less than half of the Courant limit to control the growth of perturbations in pressure and velocity. In this example, a factor of forty to fifty is gained over the explicit timestep.

Figure 1 shows the density profiles at intervals of 50 steps. The discontinuity, initially across one cell, spreads to three or four cells as it moves across the system. Most important, however, is that the discontinuity spreads no further throughout

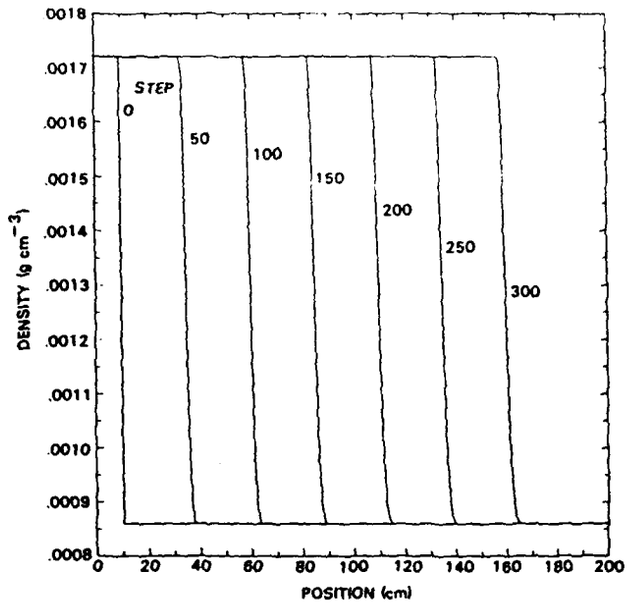


FIG. 1. Density profiles of propagating contact discontinuity at 50 step intervals.

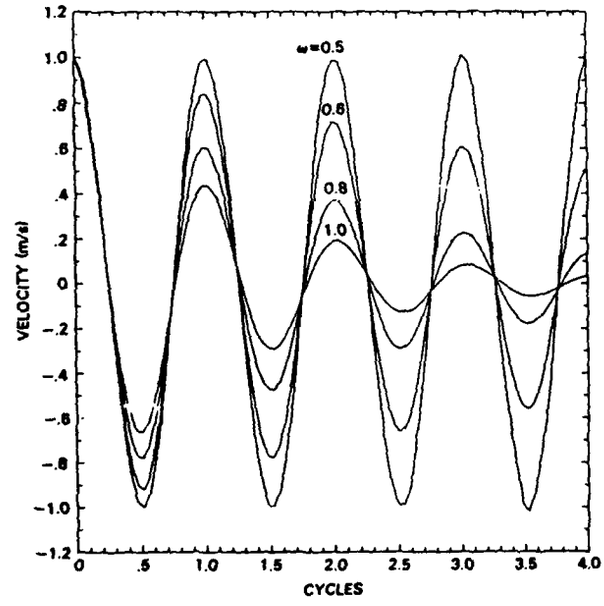


FIG. 2. Effect of ω on the damping of a sound wave.

the course of the solution and there are no ripples in the solutions. Both of these features are in the underlying explicit FCT algorithm. The results presented in the figure were obtained with $\omega = 1$. The influence of sound waves in this problem are negligible, so that any stable value of ω gives the same result.

Sound Wave Damping

Sound waves, especially high-frequency sound waves, are attenuated by most finite-difference methods. Implicit methods, however, tend to damp all frequencies, with the lower frequencies damped least. The problem we now present tests the sound-wave damping in BIC–FCT.

We consider a closed, one-dimensional pipe 1 m long in which the fluid velocity was initialized with a sinusoidal variation. The maximum amplitude of the variation was 1 m/s at the center of the pipe. Effectively, the initial conditions correspond to a sound wave in the pipe with a wavelength of 2 m.

Each curve in Fig. 2 shows the fluid velocity at the center of the pipe as a function of the number of cycles for a different value of ω. The damping is greatest when $\omega = 1$, which is when the method is completely implicit. The damping decreases as ω is reduced, and it becomes negligible when $\omega = 0.5$. Any further reduction in ω leads to instability of the numerical method. We conclude that the amount of damping is a strong function of implicitness parameter. The results shown in Fig. 2 were for a sound wave with a cell size of 2.5 cm using a timestep of 0.1 ms.

The dispersion relation, obtained directly from the calculations, is shown in Fig. 3a for CFL = 0.5. The CFL number is defined as

$$\text{CFL} = \frac{\text{sound speed} \times \text{time step}}{\text{cell size}}.$$

These calculations were made by varying the timestep as well as the number of cells in the 1 m pipe. The product of the wave number, k, and the cell size Δx is inversely related to the accuracy of representation of the wave. The number of cells per wavelength is given by

$$N = \frac{2\pi}{k\Delta x}.$$

On the vertical axis, we show ω_d and ω_t, the observed and theoretical frequencies of the wave. A totally dispersion free algorithm, in which $\omega_d = \omega_t$, would yield the 45° line shown. Curves for different values of the implicitness parameter are presented. For comparison purposes, the results for the explicit, predictor–corrector JPBFCT method [4] are included. JPBFCT requires two applications of the FCT algorithm at each timestep. This two-step process makes JPBFCT second order in time.

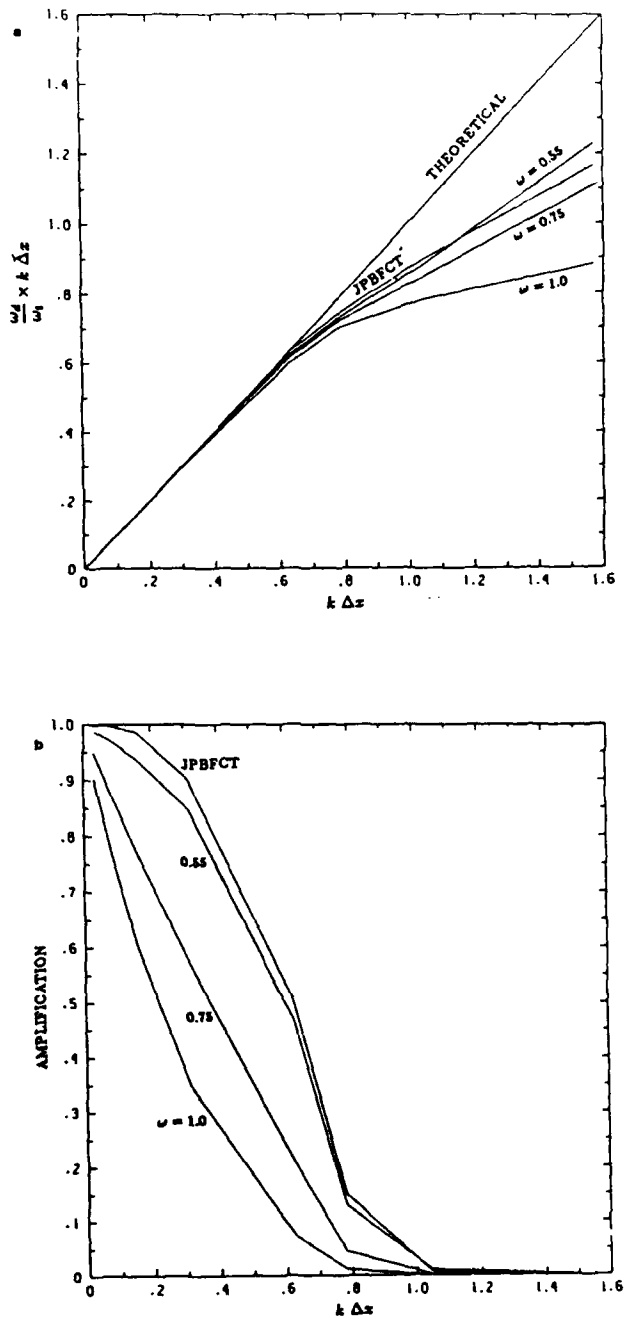


FIG. 3. Dispersion and damping of sound waves by BIC, CFL = 0.5, (a) dispersion relation, (b) damping rate.

Figure 3b indicates the change in amplitude of the wave in one period. The amplification is always less than unity, indicating that the wave is damped. If the amplification were greater than unity, the calculation would be unstable. As expected, damping increases when the method is more implicit. Poorly resolved wavelengths are damped, even by the fully explicit, time-centered, JPBFCT method. It should be noted that BIC–FCT with $\omega = 0.55$ performs nearly as well as the explicit JBPFCT. Values of ω nearer 0.5 brings results of the two methods even closer.

Figures 4a, b, and 5a, b give the dispersion relation and damping for CFL = 2 and CFL = 10, respectively. Representation of the sound wave deteriorates more rapidly as resolution is lost for these CFLs. Curves for CFL = 10 are shorter than those for lower CFL because the timestep becomes too large to resolve the oscillatory nature of the wave.

This example points out the need for caution when attempting to resolve sound waves at high CFLs. Poorly resolved wavelengths are strongly damped and cannot be adequately represented. However, if only long wavelengths are of interest, BIC–FCT can provide a substantial gain over an explicit method.

A Two-Dimensional Problem

When BIC–FCT is applied in two dimensions, the same basic three-step procedure is used. In addition, we use time splitting in the two spatial dimensions to implement the explicit FCT predictor step. However, for the method to work, the elliptic pressure change equation must be solved in two dimensions.

The solution of the elliptic pressure change equation is a substantial part of the computional effort at each timestep. In one dimension, the finite-difference form of the pressure difference equation can be solved efficiently in $O(N)$ operations, where N is the number of grid points, using standard tridiagonal methods (e.g., see Roache [21]). In two dimensions, it is important to have an efficient elliptic solver, and preferably one that is not limited to specific types of problems with specific boundary conditions. In the calculation presented below we use a multigrid method, MGRID [20], which is very fast and requires $O(N \log N)$ operations. This method is suitable for the parallel processing in pipelined, parallel, and vector computers. It is straightforward to use any other suitable elliptic solver.

The two-dimensional Cartesian test problem was selected to demonstrate the ability of BIC–FCT to treat nearly incompressible swirling flows. Calculating this type of flow is difficult for most Eulerian methods, and thus it provides a very stringent test of our method. A potential vortex with a central core was used as an initial condition in a square 10 m × 10 m region. The initial conditions correspond to the analytic solution of a line vortex with diffusion which is of the form [22]

$$V_{\text{tangential}} = \frac{c}{r}\left[1 - e^{-r^2/4\nu t}\right],$$

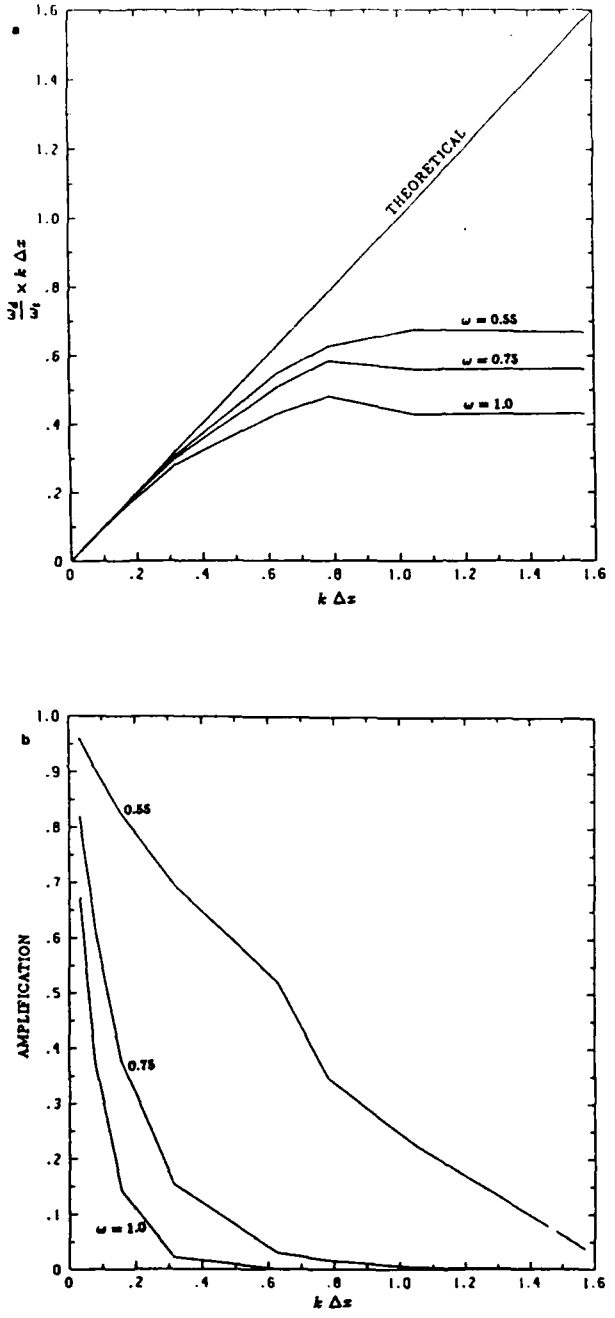


FIG. 4. Dispersion and damping of sound waves by BIC, CFL = 2.0, (a) dispersion relation, (b) damping rate.

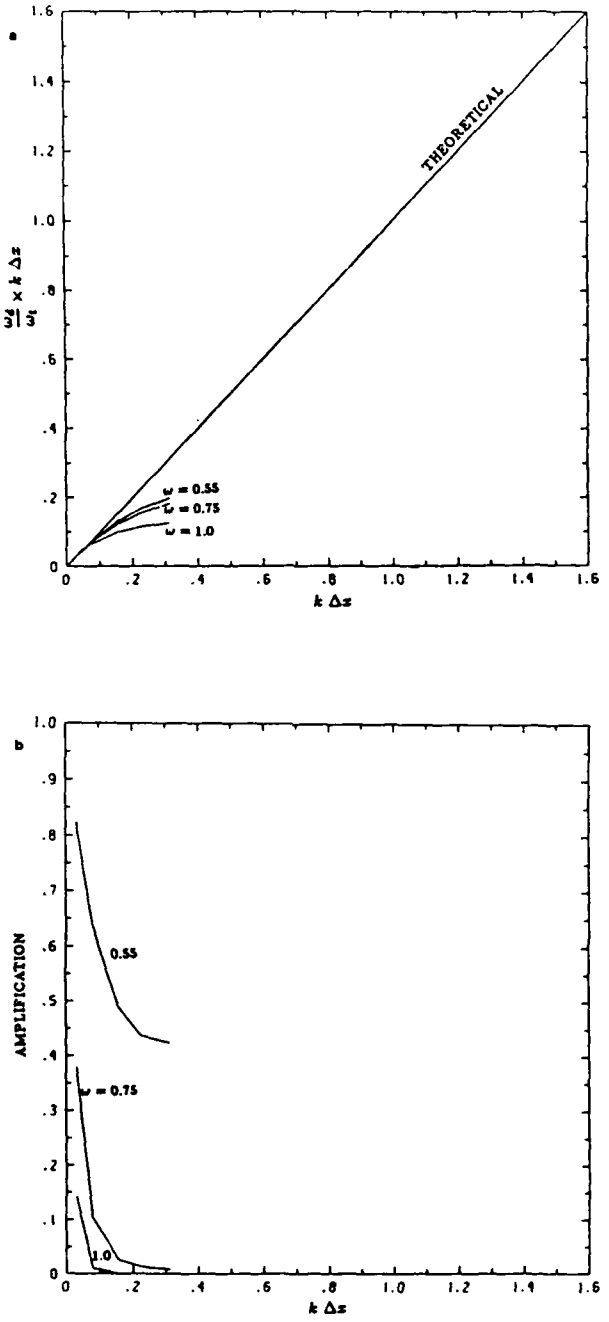


FIG. 5. Dispersion and damping of sound waves by BIC, CFL = 10.0, (a) dispersion relation, (b) damping rate.

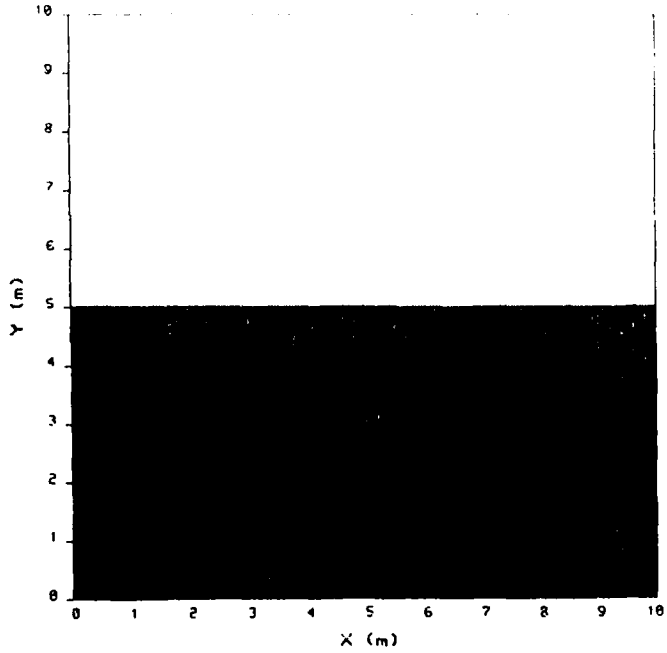


FIG. 6. Flow visualization of two-dimensional vortex flow, initial condition.

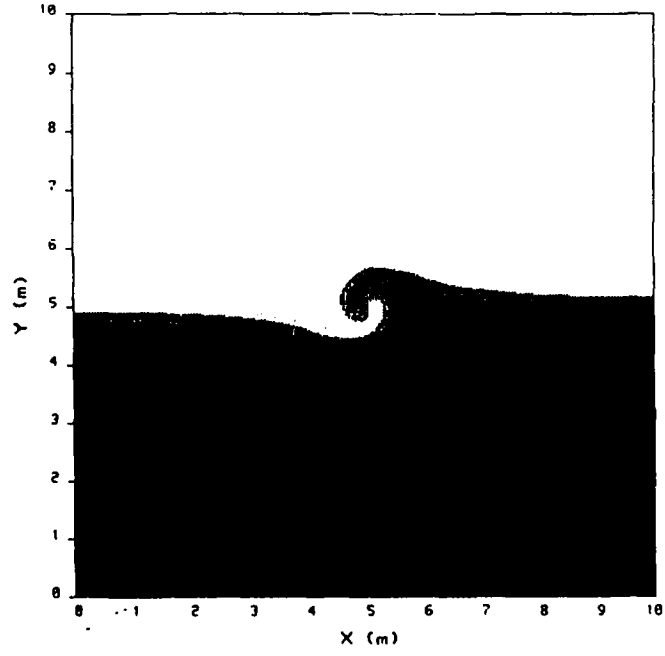


FIG. 7. Flow visualization of two-dimensional vortex flow, 50th step.

where c and ν are constants. The flow very rapidly adjusts to the presence of the walls, but this does not affect the flow close to the vortex center. In this test, a stretched 40×40 grid was used with the smallest cells 10 cm in size placed at the center of the vortex. The maximum velocity, at the start of the calculation, was 30 m/s. A conservative timestep of 1 ms was used. This should be contrasted to the 60 to 120 μs timesteps required for stability in a fully explicit method. In this nearly steady-state problem, the effects of pressure fluctuations are expected to be negligible. Therefore, we could use $\omega = 1$, the fully implicit method.

For flow visualization purposes, the lower half of the fluid has been marked and appears as the dark area in Fig. 6. In the absence of diffusion processes, either physical or numerical, this interface remains sharp as the fluid rotates at a constant velocity. Figures 7 and 8 show the position of the interface after 50 and 200 timesteps, respectively. The interface between the marked and unmarked fluid is no longer sharp, due to numerical diffusion. The interface remains fairly sharp outside the core region.

The velocity decay is given in a more quantitative manner by the scatter plots shown in the next set of figures. Tangential and radial components of velocity are plotted as a function of distance from the vortex center. Crosses denote the velocity at each grid point actually obtained from the program and the solid line provides a least squares fit of the data to the form of the analytic solution of the vortex with diffusion [22]. The initial condition is shown in Figs. 9a and b. Figures 10a, b, and

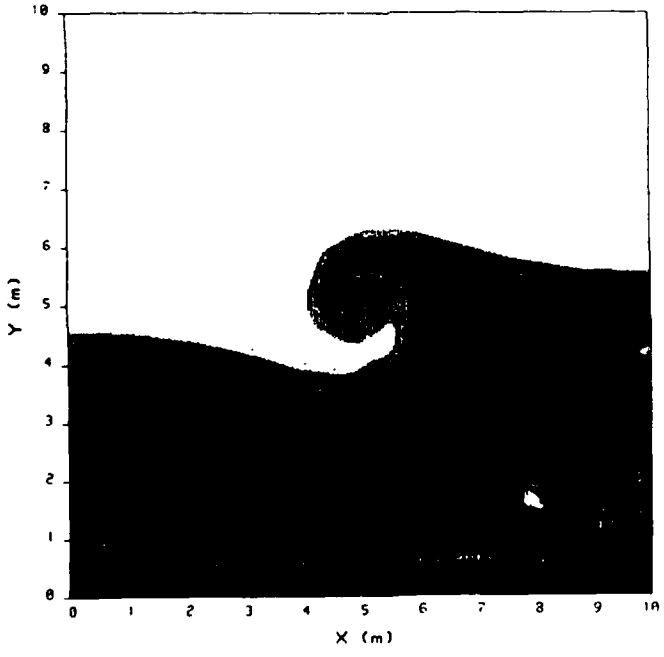


FIG. 8. Flow visualization of two-dimensional vortex flow, 200th step.

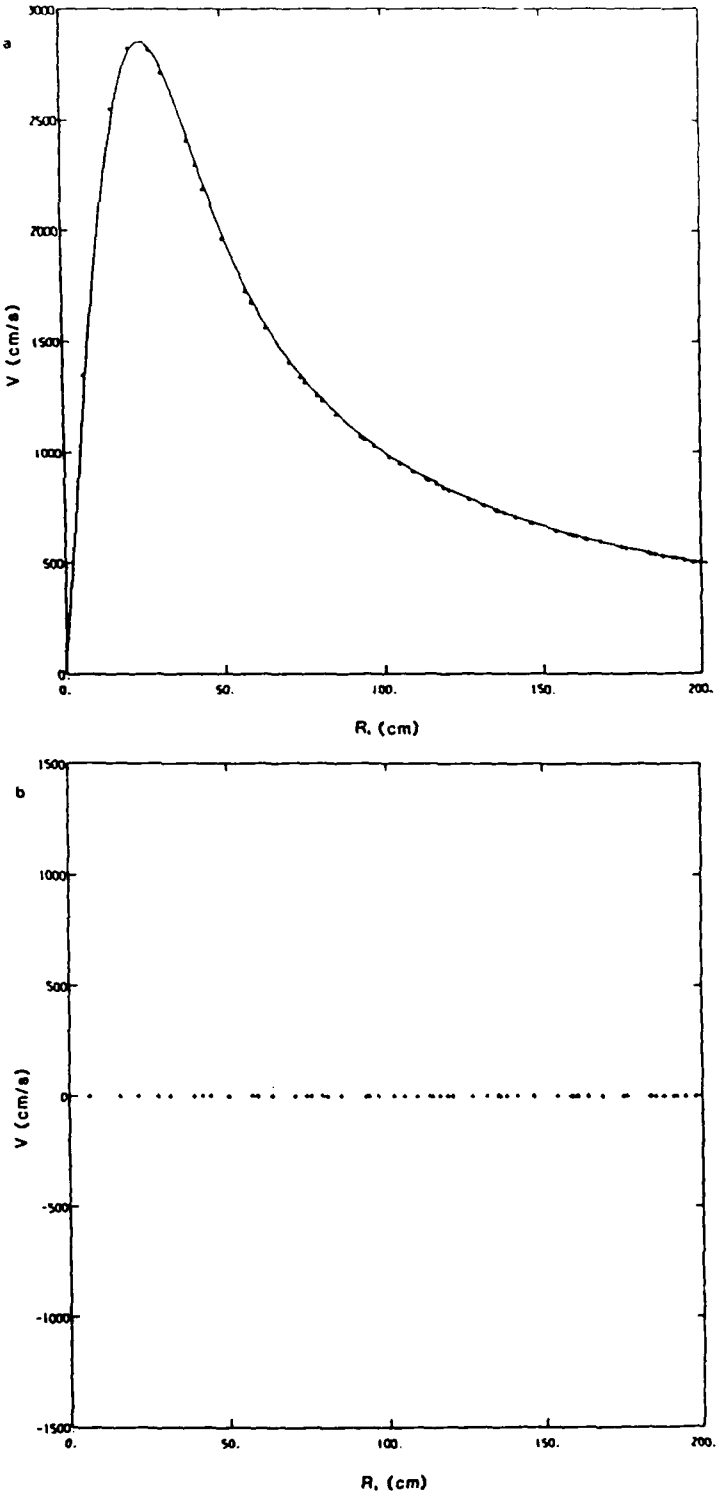


FIG. 9. Scatter plot of velocity in vortex flow, initial condition, (a) tangential component of velocity, (b) radial component of velocity.

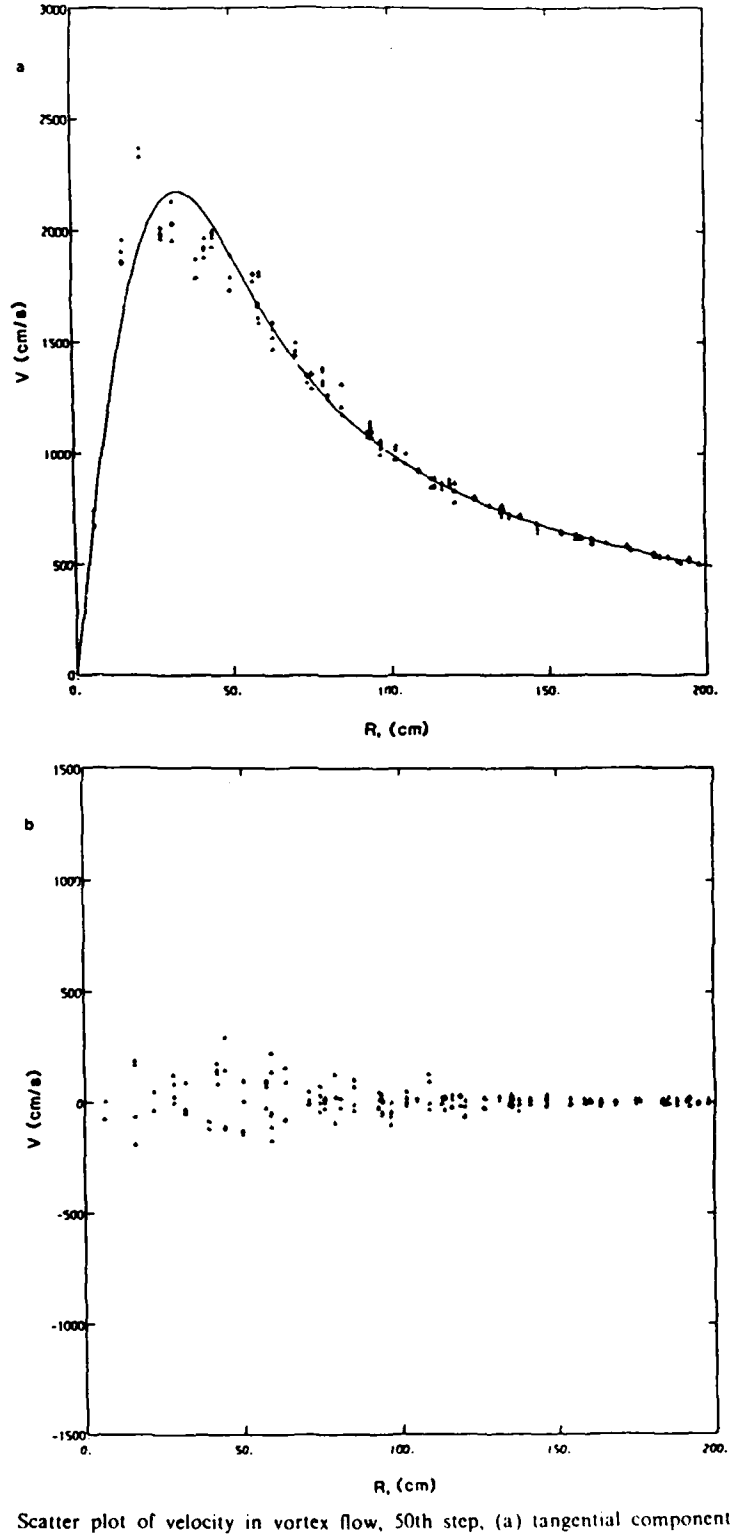


FIG. 10. Scatter plot of velocity in vortex flow, 50th step, (a) tangential component of velocity, (b) radial component of velocity.

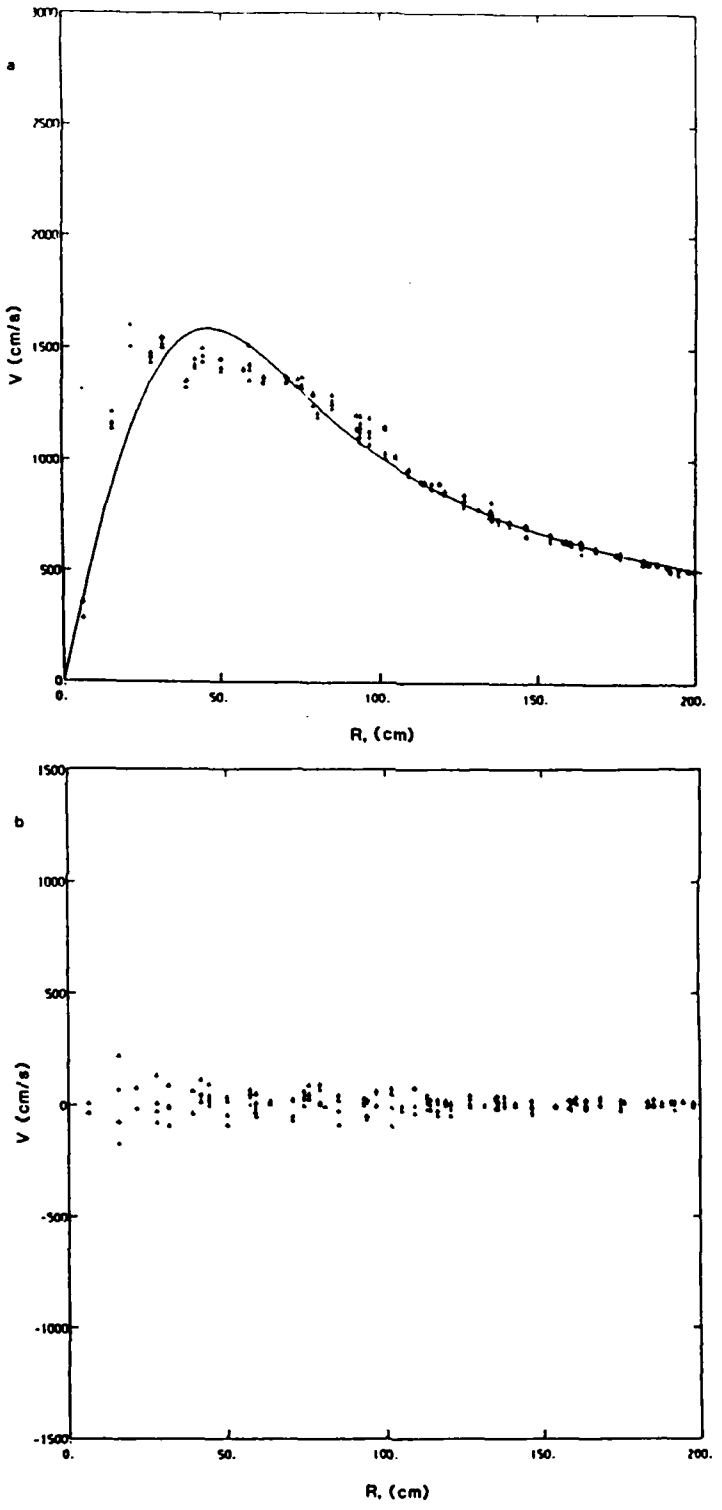


FIG. 11. Scatter plot of velocity in vortex flow, 200th step, (a) tangential component of velocity, (b) radial component of velocity.

TABLE I

Timings[a] per Step of BIC-FCT and JPBFCT

	20 × 20	40 × 40	80 × 80
BIC-FCT			
Explicit	6.8 ms	17.0 ms	54.1 ms
Elliptic	3.8	8.4	22.5
Other	2.7	5.9	17.1
Total	13.3	31.3	93.7
Per point	33.3 μs	19.6 μs	14.6 μs
JPBFCT	13.6 ms	33.9 ms	108.1 ms
Per point	34.0 μs	21.2 μs	16.9 μs

[a] On CRAY XMP-12.

11a, b show the velocity after 50 and 200 timesteps, respectively. The peak tangential velocity decreases due to numerical diffusion. However, the effective diffusion coefficient is not a constant either in space or time which leads to an imperfect fit of the data to the analytic solution. Scatter in the tangential velocity at the same location is due to the nonuniform retardation caused by varying amounts of numerical diffusion. Since the flow is essentially incompressible, nonzero radial velocities are generated.

We now examine the time it takes to do one computational timestep. Table I shows a timing comparison between BIC–FCT and the standard module, JPBFCT, very similar to that described by Boris [4]. In fact, the explicit FCT predictor in BIC–FCT is similar to the corrector step of JPBFCT. The table shows that the computational time required per timestep compares extremely favorable to that for the explicit method, especially at the larger grid sizes.

IV. Summary and Discussion

In this paper we have described the barely implicit correction method (BIC) for calculating subsonic flows. As pointed out by Casulli and Greenspan, only the pressure and velocity terms in the momentum and energy equations, respectively, have to be treated implicitly. This is sufficient to remove the sound-speed limit on the timestep. We then manipulated the equations to yield a single implicit equation, which is solved for a correction to an explicit predictor step. BIC can be used with any spatial differencing scheme. BIC–FCT provides the accuracy of the high-order monotone flux-corrected transport method but allows the large timesteps possible with an implicit method.

A number of test problems showed that BIC–FCT maintains the desirable high-order monotone characteristics of the explicit FCT algorithm. First, we showed that it could propagate a contact discontinuity as well as the two-step JPBFCT. We also presented a two-dimensional example of a swirling flow.

The implicitness parameter, ω, plays an important role in BIC–FCT whenever sound waves and pressure oscillations are important in the solution. Damping is negligible for long wavelengths and timesteps when $\omega = 0.5$. When sound waves are not important, ω can be set to unity.

The major gain is that the timestep is no longer restricted by the sound speed. This improvement is achieved at little or no additional cost per timestep. The cost of solving the elliptic equation is recovered by the elimination of the half-step calculations in explicit FCT.

In two-dimensional problems, an efficient method of solution of the elliptic pressure equation is essential. The multigrid technique MGRID used here is among the fastest. However, the application of this technique to even modestly complicated geometries is not straightforward. Unstructured multigrid methods [23] should provide the necessary flexibility.

Equations (6), (8), (9), and (16) are in conservative form. FCT has been shown to be conservative and Eq. (16), the pressure correction equation, is differenced in a conservative manner. Thus BIC–FCT is a conservative scheme. Since the pressure correction only appears as a gradient in the velocity correction, vorticity generation and transport are unchanged by BIC. Thus vorticity also stays a local, convected quantity.

BIC–FCT is not restricted to ideal gases. The equation of state can be generalized to the form

$$P^n = P^{n-1} + \frac{\partial P}{\partial e}(\varepsilon^n - \varepsilon^{n-1})$$

where $\partial P/\partial \varepsilon$ can vary in time and space. However, incompressible flows cannot be handled by BIC–FCT in its current form because FCT is not a divergence-free algorithm.

In summary, BIC–FCT has opened up the possibility of doing accurate, very slow flow calculations in which compression is important. Future computational directions include extensions to finite elements [24] and to addition of other physical processes such as gravity, viscosity, and chemical reactions to simulate premixed flames, diffusion flames, and turbulent jets.

ACKNOWLEDGMENTS

This work was sponsored by NASA in the Microgravity Science Program and by the Naval Research Laboratory through the Office of Naval Research. The authors would like to thank Rainald Löhner, Rick deVore, and Paul Woodward for their helpful suggestions.

REFERENCES

1. S. K. GODUNOV, *Mat. Sb.* 47, 271 (1959).
2. J. P. BORIS, in *Computing as a Language of Physics* (International Atomic Energy Agency, Vienna, 1971), p. 171.

3. J. P. Boris and D. L. Book, *Meth. Comput. Phys.* **16**, 85 (1976).
4. J. P. Boris, Naval Research Laboratory Memorandum Report 3237, 1976 (unpublished).
5. P. Woodward and P. Colella, *J. Comput. Phys.* **54**, 115 (1984).
6. M. R. Baer and R. J. Gross, Sandia National Laboratories Report SAND 85-0613, 1986 (unpublished).
7. F. F. Grinstein, E. S. Oran, and J. P. Boris, *J. Fluid Mech.* **165**, 201 (1986).
8. K. Kailasanath, J. H. Gardner, J. P. Boris, and E. S. Oran, in *Proceedings of the 22nd JANNAF Combustion Meeting, Pasadena, California*, 1985, p. 341.
9. R. W. MacCormack, AIAA Paper No. 81-0110, AIAA, New York, 1981.
10. R. M. Beam and R. F. Warming, *AIAA J.* **16**, 393 (1978).
11. F. H. Harlow and A. A. Amsden, *J. Comput. Phys.* **3**, 80 (1968).
12. F. H. Harlow and A. A. Amsden, *J. Comput. Phys.* **8**, 197 (1971).
13. C. W. Hirt, A. A. Amsden, and J. I. Cook, *J. Comput. Phys.* **14**, 227 (1974).
14. B. A. Fryxell, P. R. Woodward, P. Colella, and K-H. Winkler, "An Implicit-Explicit Hybrid Method for Lagrangian Hydrodynamics," *J. Comput. Phys.* **63**, 283–310 (1986).
15. H. C. Yee and A. Harten, in *AIAA 7th Computational Fluid Dynamics Conference, Cincinnati, Ohio*, 1985, p. 228.
16. W. W. Jones and J. P. Boris, *J. Phys. Chem.* **81**, 2532 (1978).
17. R. G. Rehm and H. R. Baum, *J. Res. Nat. Bur. Stand.* **83**, 297 (1978).
18. S. Paolucci, Sandia National Laboratories Report SAND 82-8257, 1982 (unpublished).
19. V. Casulli and D. Greenspan, *Int. J. Num. Methods Fluids* **4**, 1001 (1984).
20. C. R. DeVore, Naval Research Laboratory Memorandum Report 5504, 1984 (unpublished).
21. P. J. Roache, *Computational Fluid Mechanics* (Hermosa, Albuquerque, New Mexico, 1972), p. 345.
22. G. K. Batchelor, *An Introduction to Fluid Mechanics* (Cambridge Univ. Press, Cambridge, 1967), p. 204.
23. R. Löhner and K. Morgan, in *Proceedings of the Third International Conference on Numerical Methods in Thermal Problems*, edited by R. W. Lewis *et al.* (Pineridge, Swqansea, 1985), p. 132.
24. R. Löhner, K. Morgan, M. Vahdati, J. P. Boris, and D. L. Book, *J. Comput. Phys.*, (1986), in press.

Printed by the St. Catherine Press Ltd., Tempelhof 41, Bruges, Belgium

APPENDIX K.

FLIC - A Detailed, Two-Dimensional Flame Model (NRL Memorandum Report 6555)

Contents

FLIC — A DETAILED, TWO-DIMENSIONAL
FLAME MODEL

1. Introduction

This report describes the new computer code ***FL***ame with *I*mplicit *C*onvection (***FLIC***), a two-dimensional time-dependent program developed specifically to compute and study the behavior of flames and other subsonic chemically reactive flows. In order to describe a flame in enough detail to simulate its initiation, propagation, and extinction, ***FLIC*** combines algorithms for subsonic convective transport with buoyancy, detailed chemical reaction processes, and diffusive transport processes such as molecular diffusion, thermal conduction, and viscosity. Currently, we have not included algorithms for radiation transport or thermal diffusion, although these important processes can, in principle, be added in the same modular fashion as those physical processes that have been included.

FLIC solves the reactive-flow conservation equations for density, ρ, momentum, $\rho\vec{V}$, energy, E, and number densities of the individual species, n_k, $k = 1, ..., n_{sp}$, *according to:*

$$\frac{\partial \rho}{\partial t} + \nabla \cdot \left(\rho \vec{\mathbf{V}}\right) = 0, \tag{1.1}$$

$$\frac{\partial \rho \vec{V}}{\partial t} + \nabla \cdot \left(\rho \vec{V} \vec{\mathbf{V}}\right) = -\nabla P + \vec{F} - \nabla \times \mu \nabla \times \vec{V} + \nabla \left(\frac{4}{3} \mu \nabla \cdot \vec{V}\right), \tag{1.2}$$

$$\frac{\partial E}{\partial t} + \nabla \cdot \left(E \vec{\mathbf{V}}\right) = -\nabla \cdot \left(P \vec{\mathbf{V}}\right) + \nabla \cdot (\kappa \nabla T) - \sum_{k=1}^{n_{sp}} \nabla \cdot \left(n_k h_k \vec{V}_k\right) + \sum_{r=1}^{n_r} Q_r, \tag{1.3}$$

$$\frac{\partial n_k}{\partial t} + \nabla \cdot \left(n_k \vec{\mathbf{V}}\right) = -\nabla \cdot \left(n_k \vec{V}_k\right) + w_k. \tag{1.4}$$

Here $\vec{V}$ is the fluid velocity, P is the pressure, μ is the coefficient of shear viscosity, $\vec{F}$ is a body force, κ is the thermal conductivity of the mixture of gases, h_k is the enthalpy of species k, $\vec{V}_k$ is the diffusion velocity of species k, Q_r is the heat

Manuscript approved July 26, 1989.

released from reaction r, and w_k is production of species k by chemical reaction. These equations are solved assuming that the individual species are ideal gases obeying the thermal equation of state,

$$P_k = n_k kT\,, \tag{1.5}$$

and that the differential relation between internal energy u and pressure P is given by

$$\delta u = \frac{\delta P}{\gamma - 1}\,, \tag{1.6}$$

where γ, the ratio of specific heats of the mixture, is a function of its temperature and composition.

FLIC solves these equations in two dimensions for low-velocity compressible flow such that the Mach number is less than 0.1. Depending on specified initial and boundary conditions, the flames modeled can be either premixed or diffusion flames in either planar or axisymmetric geometries.

To date, ***FLIC*** has been applied to the study of the cellular instability near the flammability limits of a premixed flame in a gas containing hydrogen, oxygen, and nitrogen flame [1]. Cellular flames are formed when the mass diffusion of the deficient reactant overwhelms the stabilizing influence of heat conduction [2,3]. For hydrogen flames, this behavior is seen close to the extinction limit [4] when the flame is thick and temperatures are low. The fluid velocity is usually low, typically less than 20 cm/s in the burned region. Radiation effects are not important because the flame is not luminous and absorbing species, such as CO_2 found in hydrocarbon flames, are absent. A brief summary of this work is given in section 6.

1.1 Algorithm Development and Implementation

Producing a code that describes low-speed flames required the development of several new numerical methods as well as finding new ways to implement existing algorithms. For example, a new multidimensional, low-speed convection algorithm had to be developed. An important requirement of any convection algorithm is that the numerical diffusion not be larger than the physical diffusion processes that must be resolved. The FCT method [5,6] meets this criterion, but it is inherently

an explicit method; so that the small timesteps it requires makes it inefficient for low-speed flows. The BIC-FCT method [7] was developed specifically to combine the accuracy of FCT with the efficiency of an implicit method for low-speed flows. BIC-FCT allows timesteps up to a hundred times larger than FCT and yet the calculation time of one BIC-FCT timestep is approximately equal to the calculation time of one timestep in the standard FCT module. Section 2 describes BIC-FCT in detail.

To solve the detailed combustion equations for hydrogen-oxygen chemistry involves solving coupled nonlinear ordinary differential equations for eight species and 48 chemical reactions representing the conversion of chemical species and chemical energy release into the system. This is the most expensive part of a reactive flow calculation because it requires integrating the set of equations at each computational cell for each timestep. The characteristic times of these differential equations vary by orders of magnitude, resulting in a set of very "stiff" equations. Then, because the cost of the calculation is approximately linear with the number of computational cells, the computational cost can be extreme in multidimensional computations. ***FLIC*** handles the cost of integrating ODEs in two ways: one, by not integrating the chemical reaction equations where there is nothing or essentially nothing happening, the other is by optimizing the integration procedure. We are using the CHEMEQ [8,9] method to solve stiff sets of ordinary differential equations, but in the TBA implementation that is fully optimized for the CRAY X-MP computer. TBA, described in section 3, allows speeds of up to 50 percent over VSAIM. the multidimensional implementation of CHEMEQ used to date.

Thermo-physical properties of the individual species and the mixture are required throughout the computation. These properties are modelled with high-order curve fits to values derived from more accurate calculations. The individual properties are combined where needed to obtain mixture properties through mixing rules. This simplified method is highy efficient yet sufficiently accurate. Section 4 describes the numerical solution of the diffusive transport processes.

The submodels representing the various physical processes are in independent modules that are coupled together. Several modifications have been made to the usual timestep splitting method in order to increase the stability limits and improve the efficiency of ***FLIC***. Section 5 describes the details of these improvements.

1.2 Comparison with FLAME1D

FLAME1D is a one-dimensional program used extensively to study the properties of the ignition and extinction of hydrogen flames [10]. It is useful to compare ***FLIC*** to FLAME1D because some FLAME1D algorithms are not obviously extendable to multidimensions and others are simply too expensive. The more important differences between these codes are described here.

1. ***FLIC*** uses an Eulerian representation of the convective transport instead of a Lagrangian representation. Specifically, ADINC, the one-dimensional Lagrangian algorithm [11] used in FLAME1D, is replaced by BIC-FCT [7], a very accurate multidimensional implicit Eulerian algorithm. A Lagrangian formulation, though preferable, is exceedingly difficult in multidimensions. However, unlike ADINC, BIC-FCT can be readily used to describe two-dimensional or three-dimensional flows.

2. Although the basic CHEMEQ algorithm is used in both codes, the VSAIM implementation used in FLAME1D was replaced by the TBA implementation. This algorithm is optimized for the CRAY X-MP, and can be retrofitted into CRAY versions of FLAME1D.

3. Another expensive part of the flame calculation is determining the amount that individual species diffuse. In FLAME1D, we used an iterative matrix expansion algorithm [12] that produces the diffusion flux of a species to arbitrary order, although we generally used it only up to second order. In ***FLIC***, we use a simpler technique which first evaluates a Fickian flux and then makes a correction. This approach is equivalent to the first order of the matrix expansion, cannot be made higher order, but it is computationally less intensive and certainly adequate for the flame problems treated to date.

4. In FLAME1D, thermal conduction is computed directly from expressions for the individual thermal conductivities of the species derived from molecular theory. In ***FLIC***, we use curve fits and mixture rules which has been benchmarked against the more exact calculations used in FLAME1D.

5. Whereas FLAME1D did not include algorithms for either viscosity or gravity, both of these effects are included in ***FLIC***.

6. At the moment, FLAME1D has a more flexible gridding algorithm. Because the basic convection algorithm in FLAME1D was Lagrangian, relatively non-diffusive cell splitting and merging routines are used to refine or coarsen the grid. The result is a rather general gridding capability. The general approach to adaptive gridding must be different in a multidimensional Eulerian code, and this is now being developed for ***FLIC***.

2. Convection

In this chapter, we describe the fluid convection algorithm, BIC-FCT, how it is used in ***FLIC***, and how the gravitational acceleration term is included. In detailed flame simulations that must resolve the individual species diffusion, the numerical diffusion that results from solving the convection equations numerically must be small enough so that we can resolve the physical diffusion processes. For high-speed flows, the Eulerian explicit monotone methods such as FCT [13] or most of the TVD [14] methods achieve this goal, and some even allow variable-order accuracy. Unfortunately, the timestep required by explicit methods must be small enough to resolve the sound waves in the system, otherwise the numerical method is unstable. There is little or no penalty paid for this small timestep in high-speed flow in which the physical phenomena evolve fast, but for low-speed flows, explicit methods are very inefficient and expensive. For example, resolving a microsecond of physical time with a timestep of 10^{-8} s requires 100 timesteps, but resolving one second requires 10^8 timesteps. For many low-speed flows this temporal accuracy is unnecessary; we need only to be able to resolve a millisecond of physical time with 100 timesteps. For this reason, implicit methods that allow large timesteps are usually used for calculations of low-speed flows.

One often-considered approach to eliminating numerical diffusion is to use Lagrangian methods, in which diffusion is totally absent by definition. We have found that this approach works well in one dimension, but there are a number of serious problems in multi-dimensions [15]. In complex flows, the multidimensional Lagrangian grid becomes distorted to the point where nearest neighbors are no longer connected by grid lines, a situation that leads to extremely inaccurate calculations. Eventually grid lines can even cross, which makes the solution unstable. Such problems are often avoided by a regridding procedure that actually adds diffusion to

the solution, or by using dynamically restructuring grid such as in SPLISH [16], which adds complexity. Except in one dimension, it has not yet been shown that Lagrangian methods are to be preferred because of the geometric complexities that arise.

Most common methods for solving convection problems use algorithms that produce ripples near steep gradients such as in a flame or shock front. The first high-order, nonlinear, monotone algorithm, Flux-Corrected Transport (FCT) [13], was designed to prevent these ripples by maintaining local positivity near steep gradients while keeping a high order of accuracy elsewhere. Other nonlinear methods have been reviewed by Woodward and Collela [14]. Although these methods are explicit, there are recent reports on implicit, nonlinear methods [17,18]. A major problem with applying these implicit methods to low-speed flows is that they are expensive even though they can be very accurate.

The Barely Implicit Correction to Flux-Corrected Transport, BIC-FCT [7], was designed to overcome the problem of numerical diffusion in low-velocity implicit methods. BIC-FCT combines an explicit high-order, nonlinear FCT method [5,6] with an implicit correction process. This method removes the timestep limit imposed by the speed of sound on explicit methods, retains the accuracy required to resolve the detailed features of the flow, and keeps the computational cost as low as possible.

2.1 The BIC-FCT Algorithm

BIC-FCT is based on an approach suggested by Casulli and Greenspan [19], who showed that it is not necessary to treat all of the terms in the gas-dynamic equations implicitly to be able to use longer timesteps than those dictated by explicit stability limits. Only those explicit terms which force a timestep limit due to the sound speed have to be treated implicitly. In a pure convection problem, the timestep is still limited by the fluid velocity, but for the low Mach number flow in flames, this results in a hundredfold increase in the timestep. The term "Barely Implicit Correction" emphasizes that only the minimal number of terms in the conservation equations are treated implicitly.

BIC-FCT solves the convective portion of the Navier-Stokes equations:

$$\frac{\partial \rho}{\partial t} + \nabla \cdot \left(\rho \vec{\mathbf{V}}\right) = 0 , \tag{2.1}$$

$$\frac{\partial \rho \vec{V}}{\partial t} + \nabla \cdot \left(\rho \vec{V} \vec{\mathbf{V}}\right) = -\nabla P + \vec{F} , \tag{2.2}$$

$$\frac{\partial E}{\partial t} + \nabla \cdot \left(E \vec{\mathbf{V}}\right) = -\nabla \cdot \left(P \vec{\mathbf{V}}\right) + S , \tag{2.3}$$

$$\frac{\partial n_k}{\partial t} + \nabla \cdot \left(n_k \vec{\mathbf{V}}\right) = 0 . \tag{2.4}$$

where $\vec{F}$ is a body force, usually gravity, and S is used to couple in the contributions to the change in energy due to other processes as discussed in chapter 5. In addition, note that convection of species is done at the same time.

BIC-FCT takes a predictor-corrector approach. The first explicit step uses FCT and a large timestep governed by a CFL condition on the fluid velocity,

$$\frac{\rho' - \rho^o}{\Delta t} = -\nabla \cdot \left(\rho^o \vec{V}^o\right), \tag{2.5}$$

$$\frac{\rho' \vec{V}' - \rho^o \vec{V}^o}{\Delta t} = -\nabla \cdot \left(\rho^o \vec{V}^o \vec{V}^o\right) - \nabla P^o, \tag{2.6}$$

$$\frac{E' - E^o}{\Delta t} = -\nabla \cdot (E^o + P^o) \left[\omega \vec{V}' + (1-\omega)\vec{V}^o\right] + S, \tag{2.7}$$

where the implicitness parameter $0.5 \leq \omega \leq 1.0$. This produces the intermediate values denoted by primes.

The second step is an implicit correction requiring the solution of one elliptic equation for the pressure correction, $\delta P \equiv \omega(P^n - P^o)$:

$$\frac{\delta P}{(\gamma - 1)\omega \Delta t} - \omega \Delta t \nabla \cdot \left(\frac{E^o + P^o}{\rho'} \nabla \delta P\right) = \frac{E' - E^o}{\Delta t} - \frac{\rho' \vec{V}'^2 - \rho^o \vec{V}^{o\,2}}{2\Delta t} \tag{2.8}$$

The elliptic equation (Eq. (2.8)) is solved by the multigrid method, MGRID [20]. This method is O(N) in both number of computations as well as storage. It is vectorized and extremely efficient on the CRAY X-MP. MGRID requires that the number of grid points in each direction be factorizible by a large power of two. While this has not proven to be very restrictive for ***FLIC***, it has been a problem in other applications. MGRID also has only a limited set of boundary conditions. Fortunately, the Neumann boundary conditions used in solving Eq. (2.8) has been implemented.

The final step is the correction of the provisional values for momentum, energy, and pressure:

$$\rho^n \vec{V}^n = -\Delta t \nabla \delta P + \rho' \vec{V}', \tag{2.9}$$

$$E^n = \frac{\delta P}{(\gamma - 1)\omega} + E^o, \tag{2.10}$$

$$P^n = \delta P + P^o. \tag{2.11}$$

Because BIC-FCT uses FCT for the explicit step, it has the high-order monotone properties that accurately treat sharp gradients. This accuracy combined with the savings that result from removing the sound-speed limitation on the timestep makes BIC-FCT a very cost effective convection algorithm. BIC-FCT takes about 15μs per point per computational timestep on the CRAY X-MP computer. This is as fast as the explicit FCT code currently in use. BIC-FCT has opened up the possibility of doing accurate, multidimensional, slow-flow calculations in which fluid expansion is important. Because the cost of BIC-FCT is modest even in two dimensions, reasonably detailed chemistry models as well as other physical processes can be included. Premixed flames, diffusion flames, and turbulent jet flames are some of the applications for which BIC-FCT is well suited.

2.2 Gravity

Buoyancy effects due to the force of gravity have been incorporated in ***FLIC***. The body force term $\vec{F}$ in Eq. (2.6) is used for this purpose and is given by

$$\vec{F} = \vec{g}(\rho - \rho_\infty) \tag{2.12}$$

where ρ_∞ is a suitable reference density, usually the cold ambient density. As currently implemented, the direction of the gravity vector is aligned with the flow direction, but this restriction can be removed trivially. Indeed, the gravity vector can be made time-dependent to simulate g-jitter in microgravity.

If the ambient density cannot be used, the hydrostatic head has to be included in its place. This second approach is not as suitable as the method given by Eq. (2.12) because of the need for very high precision calculations.

3. Chemistry

This chapter describes the integration package TBA that is used to solve the ordinary differential equations (ODE's) representing the chemical reactions and energy release. TBA is a fully vectorized FORTRAN subroutine for the CRAY X-MP. It is designed to replace VSAIM, the older vectorized version of CHEMEQ [8,9], an algorithm that solves a system of ODE's in a single computational cell. Both VSAIM and TBA are based on CHEMEQ and are designed to make the calculation of a large number of sets of ODE's more efficient. TBA is faster than VSAIM because it is designed to make specific use of the Cray X-MP's gather/scatter hardware and other capabilities.

The ODE's solved by these routines are of the form

$$\frac{dn_i}{dt} = F_i = Q_i - L_i n_i, \tag{3.1}$$

where n_i is the number density of species i, Q_i is the rate of formation of species i, and $L_i n_i$ is the rate of destruction of species i. Sometimes these equations can be solved by classical algorithms, sometimes they are stiff and need special techniques. TBA uses a different algorithm for each type of equation and gains efficiency by gathering all equations of a given type together and integrating them by groups.

A detailed hydrogen-oxygen reaction scheme has been implemented in *FLIC*. This reaction scheme consists of forty-eight reversable reactions involving eight species. Nitrogen, acting as a diluent, is considered to be chemically inert. This reaction scheme, developed by Burks and Oran [21], has been used by Kailasanath *et al.* [22] in FLAME1D and is given in table 3.1. The total rate of formation and destruction of each species is obtained algebraically from the reaction rates of the individual chemical reactions and from the species concentrations. The reaction

rate for each chemical reaction r is assumed to follow the modified Arrhenius form:

$$k_r = AT^B e^{-C/T} \tag{3.2}$$

The computation of the rates in Eq. (3.2) is quite time consuming, mainly due to the calculation of the exponential term. Some time can be saved by computing the temperature dependence of the reaction rates only once per global timestep. The rates of formation and destruction of the species, which are also dependent on the species concentrations, is computed as often as needed by TBA, which updates them many times during each global timestep.

3.1 Numerical Method

TBA uses a second-order predictor-corrector method that is essentially the same numerical integration algorithm as CHEMEQ [9]. Normal ODE's are integrated using a simple classical scheme and stiff ODE's are integrated using an asymptotic method. Unlike CHEMEQ or VSAIM, TBA also recognizes equations that will approach equilibrium during the period of integration and handles them with a third scheme. TBA sorts the equations from every cell into one of three types — normal, stiff, or equilibrium, and then integrates all of the equations of each type together. A large number of cells are integrated simultaneously; as cells are completed, the results are returned to the control program and stored and data from new cells are read in and integrated.

The entire set of equations from each cell is integrated using the smallest timestep required by any equation in the set. The timestep is then increased or decreased based on the relative difference between the predictor and corrector stages. The predictor stages for the three types of equations are:

$$n_i' = n_i^o + \delta t F_i^o \,, \qquad \text{(Normal)} \tag{3.3}$$

$$n_i' = n_i^o + \frac{\delta t F_i^o}{1 + \delta t L_i^o} \,, \qquad \text{(Stiff)} \tag{3.4}$$

$$n_i' = Q_i / L_i \,. \qquad \text{(Equilibrium)} \tag{3.5}$$

The corrector stages are:

$$n_i^n = n_i^o + \frac{\delta t}{2} [F_i^o + F_i'] \,, \qquad \text{(Normal)} \tag{3.6}$$

PAGES 12, 13, 14, 15, 16, 17, 18,
ARE
MISSING
IN
ORIGINAL
DOCUMENT

4.2 Thermal Conduction

The effects of thermal conduction are expressed in the energy equation as

$$\frac{\partial E}{\partial t} = -\nabla \cdot (\kappa \nabla T) \ , \tag{4.12}$$

where κ is the mixture thermal conductivity. Explicit finite differencing introduces a stability limit

$$\max\left(\kappa \Delta t / \rho c_p \Delta x^2\right) < 1/2 \ ,$$

where $\kappa/\rho c_p$ is the thermal diffusivity coefficient. Subcycling is used here in the same manner as it is for mass diffusion. However, this stability condition is less stringent than that for mass diffusion and typically only two or three subcycles are needed.

The mixture thermal conductivity κ is obtained by combining the thermal conductivities of the individual gases $\{\kappa_k\}$ that are in the mixture. The $\{\kappa_k\}$ are estimated theoretically and are a function of temperature. We have used the third-order polynomial fit in temperature determined by Laskey [37] and is presented in Table 4.2. The mixture thermal conductivity is then calculated using (Mathur *et al.* [38])

$$\kappa = \frac{1}{2}\left[\sum_{k=1}^{n_{sp}} X_k \kappa_k + \frac{1}{\sum_{k=1}^{n_{sp}} \frac{X_k}{\kappa_k}}\right] . \tag{4.13}$$

4.3 Viscosity

The viscosity terms in the Navier-Stokes equations are included in *FLIC*. The stress-tensor term, which represents the effect of viscosity in the momentum conservation equations, is:

$$\frac{\partial \rho \vec{V}}{\partial t} = -\nabla \cdot \tau \ , \tag{4.14}$$

where

$$\tau = \left(\frac{2}{3}\mu - \lambda\right)\left(\nabla \cdot \vec{V}\right) \mathrm{I} - \mu\left[\left(\nabla \vec{V}\right) + \left(\nabla \vec{V}\right)^T\right] \ , \tag{4.15}$$

Species	A	B	C	D
H	4.710(3)	3.354(1)	-9.971(-3)	1.964(-6)
O	1.089(3)	1.038(1)	-3.739(-3)	8.251(-7)
H_2	6.306(3)	4.304(1)	-8.505(-3)	2.160(-6)
OH	1.679(3)	1.091(1)	-1.613(-3)	4.150(-7)
H_2O	-2.077(2)	1.603(1)	-7.932(-4)	1.530(-7)
O_2	3.862(2)	8.613(0)	-1.966(-3)	3.619(-7)
HO_2	1.576(2)	1.070(1)	-1.143(-3)	4.471(-8)
H_2O_2	-1.097(3)	9.895(0)	-1.779(-3)	1.396(-7)
N_2	7.024(2)	6.917(0)	-1.191(-3)	2.035(-7)

Table 4.2 Thermal conductivity of species $k, \kappa_k = A + BT + CT^2 + DT^3$, erg/cm-s-K. Exponentials to the base 10 are given in parentheses.

and μ is the dynamic viscosity coefficient. The quantity λ, the second coefficient of viscosity, is set to $-2/3\mu$. The stress tensor τ includes all the viscous terms which arise in the compressible Navier-Stokes equations.

Eq. (4.14) is solved explicitly in the same manner as the mass diffusion or thermal conduction equations. Thus there is a stability criterion given by

$$\max\left(\mu\Delta t/\rho\Delta x^2\right) < 1/2\ ,$$

where μ/ρ is the kinematic viscosity. Subcycling is used so that the overall computational timestep is set by the convection stability limit and not by the viscosity stability limit. If the number of subcycles required for stability exceeds some maximum value, the global timestep is reduced. The viscous diffusion algorithm was tested using two test problems: 1) the boundary-layer growth over a flat plate parallel to the flow, and 2) the boundary-layer thickness on a flat plate normal to the flow (stagnation point flow).

For the parallel-plate test, the velocity profile of the parallel flow was initially uniform along the plate with a typical velocity profile that is valid for a boundary-layer thickness greater than three computational cells. This particular initialization was chosen because a physical boundary layer less than three cells wide would be

swamped by numerical diffusion from the FCT algorithm that creates a boundary layer at least this thick. Setting up the problem this way simulates the growth of a boundary layer away from the leading edge of the plate and minimizes any numerical effects on the solution. The result of this test is a boundary layer whose growth matches the Blasius solution.

The stagnation-flow test was initialized with a boundary layer of constant thickness and uniform velocity along the plate. Again, the initial boundary layer thickness was more than three cells to minimize numerical effects. The results showed the development of a constant-thickness boundary layer whose velocity profile very closely matched that predicted by theory.

For a gas containing a single species k, the dynamic viscosity μ_k can be otained from the kinetic theory of gases [33]. Over a suitable range of temperature, this can be expressed as a third-order polynomial in temperature. Laskey [37] has compared this polynomial fit, presented in Table 4.3, to the calculations and to tabulated values for the viscosity and found good agreement. The mixture dynamic viscosity is calculated using the expression (Wilke [39])

$$\mu = \sum_{k=1}^{n_{sp}} \frac{X_k \mu_k}{\sum_{j=1}^{n_{sp}} X_j \Phi_{kj}} , \tag{4.16}$$

where

$$\Phi_{kj} = \frac{1}{\sqrt{8}} \left(1 + \frac{M_k}{M_j}\right)^{-\frac{1}{2}} \left(1 + \left(\frac{\mu_k}{\mu_j}\right)^{\frac{1}{2}} \left(\frac{M_j}{M_k}\right)^{\frac{1}{4}}\right)^2 . \tag{4.17}$$

This mixture viscosity, μ, is used in the stress tensor (Eq. (4.15)) to model the viscous portion of the Navier-Stokes equations.

Species	A	B	C	D
H	1.516(-5)	1.074(-7)	-3.178(-11)	6.255(-15)
O	4.504(-5)	5.355(-7)	-1.811(-10)	3.740(-14)
H_2	2.802(-5)	2.236(-7)	-6.958(-11)	1.399(-14)
OH	5.630(-5)	5.193(-7)	-1.678(-10)	3.413(-14)
H_2O	-8.597(-6)	6.608(-7)	-2.305(-10)	4.601(-14)
O_2	4.930(-5)	5.861(-7)	-1.983(-10)	4.093(-14)
HO_2	5.006(-5)	5.952(-7)	-2.013(-10)	4.156(-14)
H_2O_2	-9.109(-6)	3.966(-7)	-1.345(-10)	2.623(-14)
N_2	5.302(-5)	4.596(-7)	-1.464(-10)	2.969(-14)

Table 4.3 Viscosity of species $k, \mu_k = A + BT + CT^2 + DT^3$,dyne-s/cm^2. Exponentials to the base 10 are given in parentheses.

5. Model Integration

The conservation equations contain terms representing convection, buoyancy, thermal conduction, molecular diffusion, viscous diffusion, and chemical reactions. The approach that ***FLIC*** uses is to determine a global timestep, solve equations representing the individual physical processes separately for that timestep, and then couple the solutions. The coupling procedure is a variation of the standard timestep splitting method of Yanenko [40] and described for reactive flows by Oran and Boris [15]Chapters 4 and 13.

The usual explicit timestep-splitting approach assumes that in some predetermined global timestep, the effect of all the physical processes can be evaluated as a running sum of the effects of individual processes. Each physical process is integrated independently using the results of the previous process as initial conditions. This method is correct in the limit of small timesteps and works well in a practical sense when the changes in the variables during the global timestep are small. Using this appproach, the global timestep is often limited to the smallest timestep required by the stability limits of the integration algorithms for the various processes. This is the approach we have used in a number of programs in which the convection is solved by an explicit integration procedure. The global timestep is usually determined by the CFL condition on the sum of the sound speed and the fluid velocity. The chemistry integration is subcycled in the global timestep. However, subcycling can sometimes be used for individual processes if changes in variables due to that particular process are not too large.

Figure 5.1 summarizes the integration process in a typical ***FLIC*** timestep. The global timestep is first estimated based on the stability requirements of the convection algorithm, BIC-FCT. Then it is compared with the stability requirements of the particular physical processes which are allowed to subcycle if necesary to

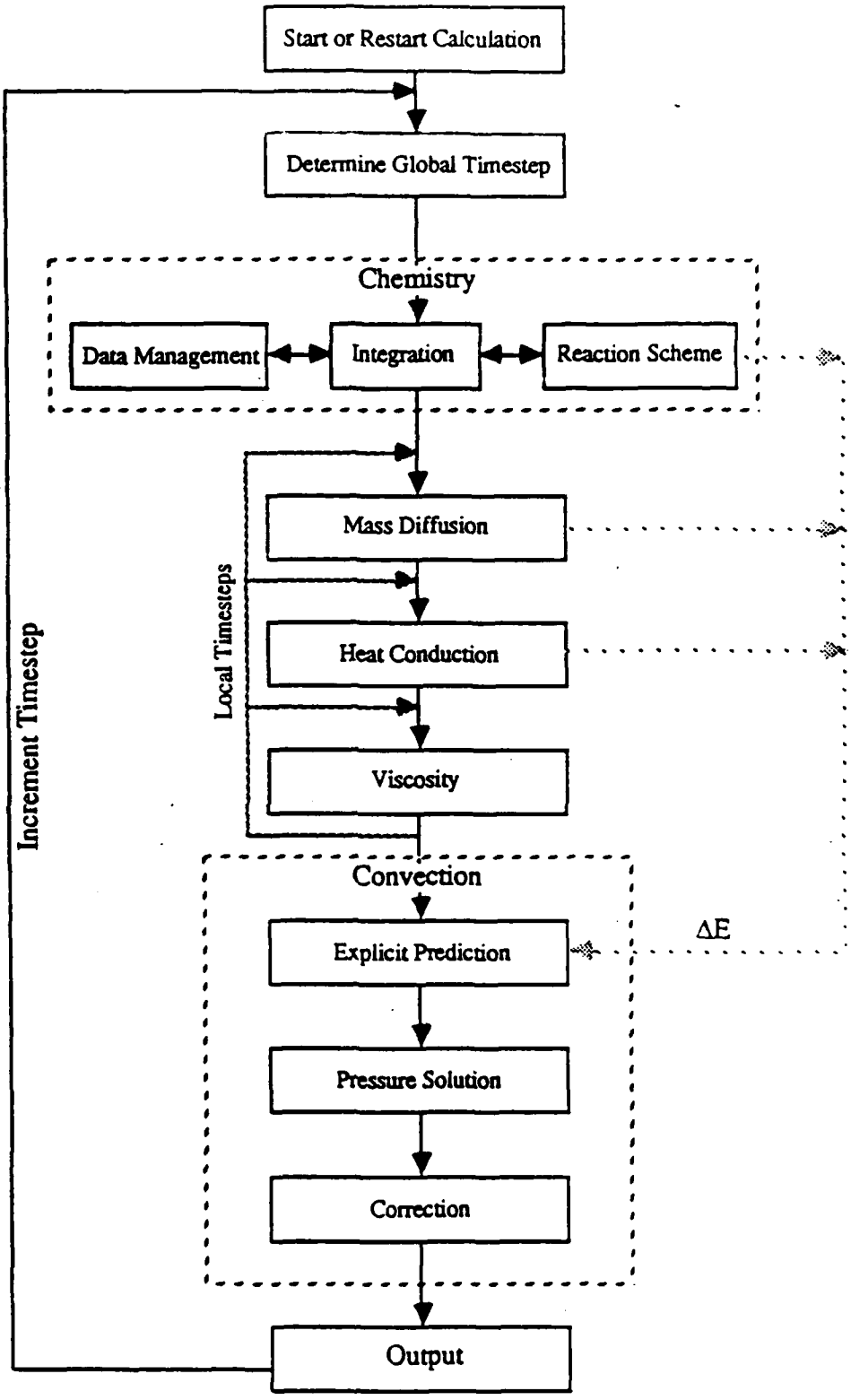


Figure 5.1 Flowchart of *FLIC*, indicating interprocess communication.

ensure stability. The overall timestep must be decreased sometimes to ensure that a physical variable does not change too much during a timestep. The most obvious difference between the scheme shown in Fig. 5.1 and the standard timestep splitting approach is that changes in the internal energy caused by each process are evaluated and added up at the end of a global timestep. This energy change is then used by the BIC-FCT convection algorithm (see Eq. (2.7)). This approach is quite similar to that used in FLAME1D [22], however, now changes in the internal energy are accumulated instead of changes in the pressure. The ADINC method [11], used in FLAME1D, does not solve an energy equation, and thus the only way other processes can be coupled is through the pressure. The BIC-FCT scheme used by ***FLIC*** does not require that energy changes be applied only during the convection step; this is merely a convenience that allows for larger global timesteps due to tighter coupling.

Some of the individual process integrations in ***FLIC*** are subcycled within a global timestep, including the ordinary differential equations representing the chemical reactions and the diffusive terms such as molecular diffusion and thermal conduction. For example, the timestep limit imposed by some chemical reactions may be orders of magnitude lower than that required by other physical processes, and so the chemistry integration is subcycled. Subcycling is built into the chemistry integration in an extremely sophisticated manner [9], so that the maximum allowable timestep at each computational cell is used, completely independent of the timestep in other cells. The chemistry is integrated up to the overall timestep before it is coupled to the other processes. However, if the energy release in an overall timestep due to chemistry is too large (typically greater than 10%), then the overall timestep must be decreased. Mass diffusion and thermal conduction are also subcycled, but only up to five times. The accuracy of the solution is generally tested by performing a separate calculation with a smaller timestep and noting whether the solution has converged.

All dependent variables, except for internal energy, are updated after each process integration, but dependent variables are not updated during subcycling of a process. This leads to a considerable savings, especially during the chemistry integration, because the evaluation of the temperature exponentials in Arrhenius expressions is done only once per global timestep. On the other hand, the global timestep may have to be decreased if the dependent variables change too much.

One consequence of a source term in the energy equation is the possibility of "ripples" in the pressure, which then manifest themselves in other variables as well. These ripples arise if a strong source is localized to a region only a few cells wide. These ripples are insignificant in ***FLIC*** where the flame zone is well resolved, but can be significant in other applications [37]. One way to avoid the ripples is to use a high-frequency filter such as

$$P^{filtered} = P + \alpha \nabla^4 P,$$

where α is a small constant. Other filters, including another FCT step, can also be used.

The particular advantages to using timestep splitting are that we can write very modular programs in which the integration of each physical process can be carried out with an optimum method, debugging is simpler, and explicit subcycling can be used to keep the costs down. Implicit methods can be used to avoid subcycling, but are more expensive when compared to explicit methods subcycled only a few times. In ***FLIC***, only five subcycles are allowed for each of the diffusion processes. Chemistry, on the other hand, subcycles thousands of times. At this point, it is not entirely clear if the extreme simplicity of the CHEMEQ scheme [9] results in faster integration of the chemistry equations than a more complicated implicit scheme. Disadvantages of timestep splitting are that the coupling process can be complicated, the algorithms and the overall program are less stable, and the timestep must be carefully controlled. However, we have found that the benefits of modular and fast programs outweigh potential disadvantages. This is discussed in some detail by Oran and Boris [15], pages 131–133.

6. Applications

FLIC has been written in a general manner so that it can be applied to solve a variety of slowly evolving problems involving chemistry and diffusive transport processes. It has already been applied to the study of multidimensional flames in premixed gases [1,41,42] and co-flowing diffusion flames [37,43]. A specific application of the code to the study of cellular flames in hydrogen-oxygen-nitrogen mixtures is described below to show what is required to perform a calculation with the code and to interpret the output from the code. Then other applications of the code are briefly discussed to bring out the generality of ***FLIC***.

6.1 A Sample Calculation

Flames in lean hydrogen-oxygen-nitrogen mixtures are known to exhibit a multidimensional cellular structure. The cellular structure is the result of a thermo-diffusive instability of a planar flame in the same mixture. Below we demonstrate how FLIC can be used to simulate the transition from a planar flame to a multidimensional cellular flame in a zero-gravity environment.

We need as input to the model:

A chemical reaction scheme involving all the species of interest (table 3.1),

Molecular diffusion coefficients for each pair of species (table 4.1),

For each species:

Molecular weights,

Thermal conductivity (Table 4.2),

Viscosity (Table 4.3),

Heats of formation [22],

Enthalpy coefficients [22].

To complete the specification of the problem, we also need the initial and boundary conditions such as composition, pressure, and temperature, in addition to the physical and chemical parameters given above. Figure 6.1 describes the configuration studied and gives the boundary conditions of the computational domain. Unburnt gas flows in from the left, and reaction products of the flame front flow out at the right. If the inlet velocity equals the burning velocity of the flame, the flame is fixed in space yielding a steady solution. Thus, transient effects from ignition can be eliminated. The initial conditions for the two-dimensional calculations were taken from one-dimensional calculations that gave the conditions for steady, propagating flames. The two-dimensional computational domain for this simulation was 2.0 cm $\times$ 4.5 cm, resolved by a 56 $\times$ 96 variably spaced grid. Fine zones, 0.36 mm $\times$ 0.15 mm, were clustered around the flame front.

The initial conditions specify a planar flame in a fuel-lean hydrogen-oxygen mixture diluted with nitrogen, H_2:O_2:N_2/1.5:1:10, a flame that showed multidimensional structure in the experiments by Mitani and Williams [4]. In order to study the evolution to cellular structure, the initial conditions were perturbed by displacing the center portion of the planar flame in the direction of the flow. The evolution to cellular structure is obtained by studying the output from the simulations.

The output from the calculation consists of the spatial and temporal distribution of all the species involved as well as the fluid density, temperature, pressure, internal energy, and momenta. Display of the data is not done by ***FLIC***, but is instead done as a post-processing operation. This cuts down the length and complexity of the ***FLIC*** code. A very useful diagnostic is contour plots showing isotherms and species distributions. Diagnostics, such as color-flood plots, allow visual interpretation of the data.

Figure 6.2 shows a sequence of isotherms from the calculation. The isotherms show that the temperature increases in the center portion of the flame, convex to the flow, and decreases in the the two adjacent concave regions, indicating more vigorous reaction in the convex region. The atomic hydrogen concentration increases in the convex and decreases in the concave regions. Also, the burning velocity in the

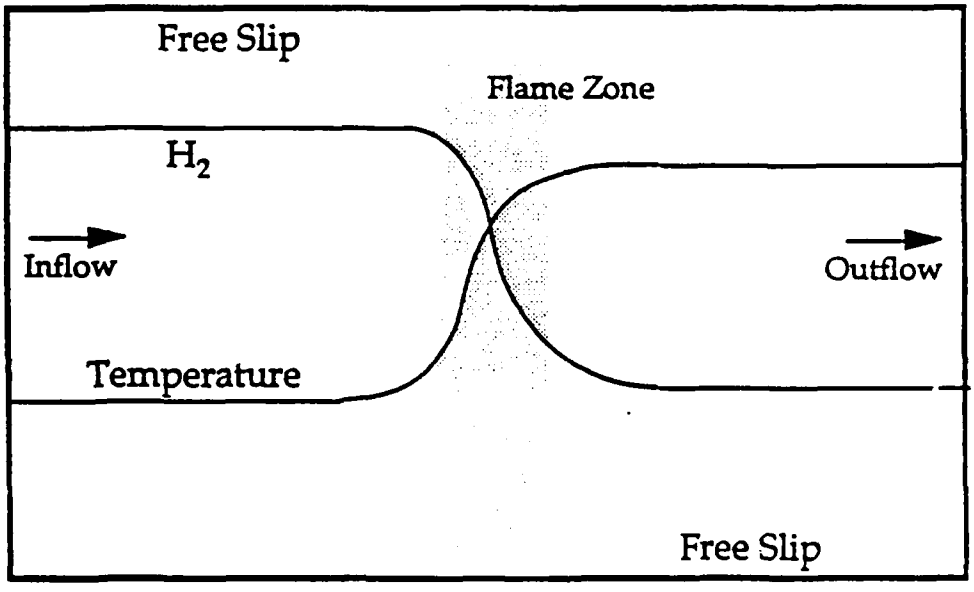


Figure 6.1 Initial and boundary conditions for the two-dimensional flame calculations.

convex region appears slightly higher than the burning velocity in a planar region, and the burning velocity in the concave regions were noticeably lower. Thus, this calculation shows that the planar lean one-dimensional hydrogen flame is unstable and evolves into a multidimensional flame having a cellular structure.

6.2 Applications of *FLIC*

The ***FLIC*** code has been used extensively to study the detailed evolution of cellular flames in premixed gases [1], to investigate the mechanisms which can lead to cellular structure [42], effects of gravity on flame structure and instabilities [41,42]. These studies have helped strengthen the prevailing theory of cellular instability and cast serious doubt on another theory which was also under consideration [1] ***FLIC*** has been geared primarily to these types of applications and several other related applications to premixed flames are planned for the near future.

FLIC can be converted to the study of diffusion flames extremely easily, and is begging for a suitable application in this area. A low-speed diffusion flame code [37,43] which has been used to study jet flames uses the same transport and diffusive packages as ***FLIC***. This code uses simplified chemistry, which was first calibrated by comparison to a detailed calculation with ***FLIC***.

6.3 Future Applications and Code Development

Calculations such as the one described earlier are time consuming (5 hours of CRAY X-MP time). There is interest to carry out these calculations in a larger domain for longer times. The cellular structure exhibited by flames is actually three-dimensional, so if these instabilities are to be studied fully, a three-dimensional version of ***FLIC*** is required. The computer time for these studies can quickly become intolerable; it is proposed to alleviate this by taking advantage of the multiprocessor architechure of the CRAY X-MP and the new Y-MP with its 16 processors. This will require some restructuring of the code and its algorithms to utilize microtasking appropriately. This restructuring may be simplified with the new "autotasking"

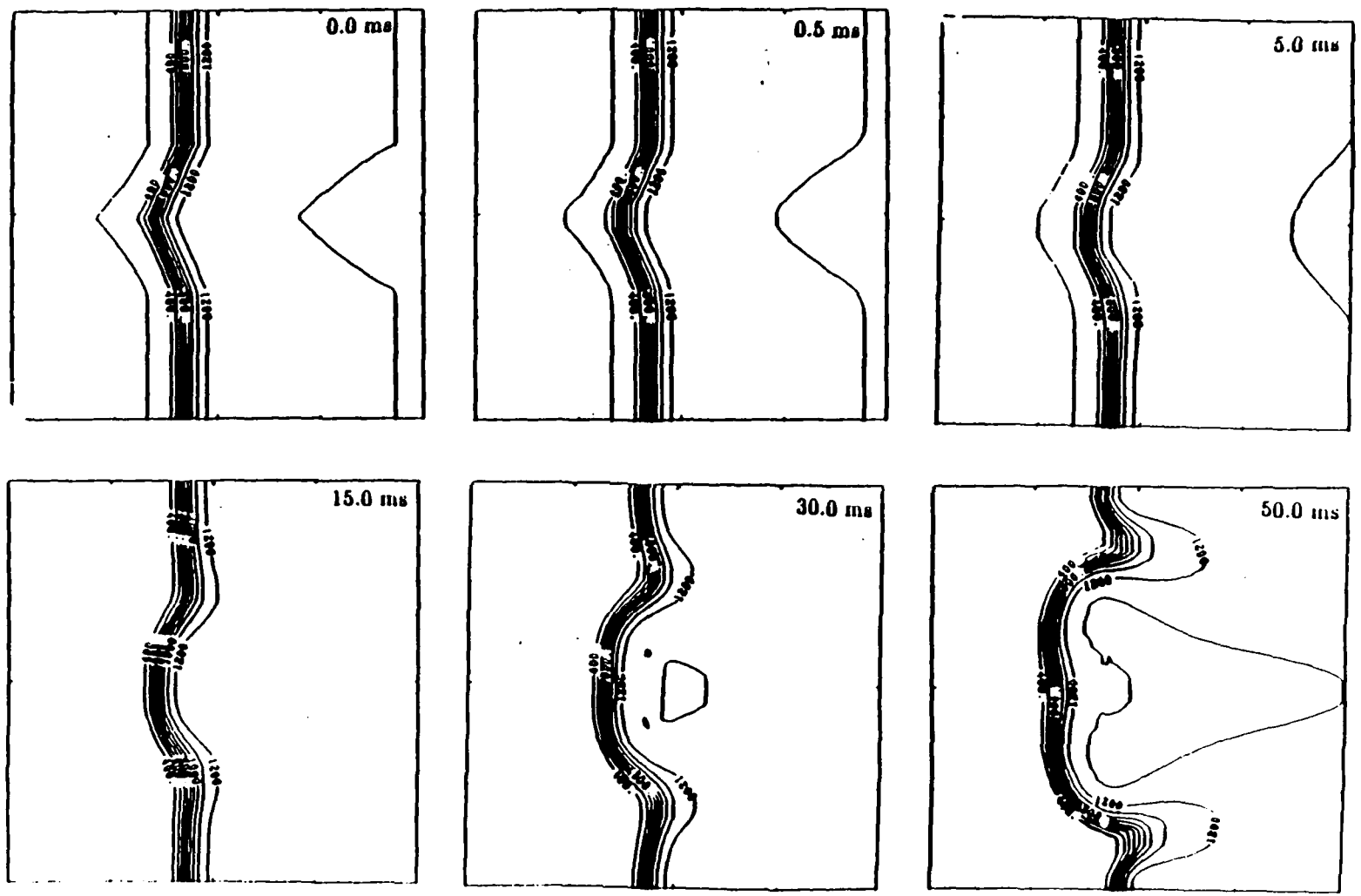


Figure 6.2 Evolution of isotherms in a fuel-lean hydrogen flame.

facility on the CRAY Y-MP.

One area of interest is the investigation of the behavior of flames near extinguishment. In particular, the prediction of flammability limits of flames in a flammability tube will provide a means for quantitative comparison with experiment. This complex task will require the addition of mechanisms which represent losses, including losses to the wall. The physics of the loss mechanisms of radicals to the walls is not yet fully understood. These additional mechanisms will need to be incorporated into ***FLIC***.

Detailed calculations will need to be performed for other more complex fuels of practical interest. The primary difficulty is in coping with the large number of species and chemical reactions that will be required for even the simplest hydrocarbon fuels. While the precise scaling of computer time with chemical complexity is not known, it is expected that the increase in the number of species will have the most dramatic effect. Thus, there is a need for reliable simplified chemistry models which have been first validated against a full model. It is anticipated that suitable simplified models for methane will become available soon. Radiation can not be neglected in hydrocarbon combustion. Thus a gas phase radiation model will have to be incorporated. Consideration of soot will have to wait until reliable models are available, which will not be in the near future.

All future applications are in areas that will require enormous amounts of computer time. One way to cut down on costs is to perform calculations only where they are strictly required. Calculations can be skipped in regions of low gradients, be it spatial or temporal gradients. This will require dynamic regridding or, more generally, re-discretization of the governing equations. A suitably efficient algorithm for this is essential, and its development is underway. Limitations and peculiarities of the target computer architecture will play a large role in determining the shape of these future versions of ***FLIC***.

Acknowledgments

This work was sponsored by NASA in the Microgravity Sciences Program and the Office of Naval Research through the Naval Research Laboratory. We also acknowledge some computational support from the Pittsburgh Supercomputing Center.

References

[1] G. Patnaik, K. Kailasanath, K. Laskey, and E. S. Oran. Detailed Numerical Simulations of Cellular Flames. In *Proceedings of the 22nd Symposium (International) on Combustion*, Pittsburgh, PA, 1989. The Combustion Institute.

[2] G. I. Barenblatt, Y. B. Zeldovich, and A. G. Istratov. On Diffusional Thermal Stability of Laminar Flame. *Zh. Prikl. Mekh. Tekh. Fiz.*, 4:21–26, 1962.

[3] G. I. Sivashinsky. Diffusional-Thermal Theory of Cellular Flames. *Combust. Sci. Tech.*, 15:137–146, 1977.

[4] T. Mitani and F. A. Williams. Studies of Cellular Flames in Hydrogen-Oxygen-Nitrogen Mixtures. *Combust. Flame.*, 39:169–190, 1980.

[5] J. P. Boris and D. L. Book. Solution of Convective Equations by the Method of Flux-Corrected Transport. In *Methods in Computational Physics*, volume 16, chapter 7, page 85. Academic Press, New York, 1976.

[6] J. P. Boris. Flux-Corrected Transport Modules for Solving Generalized Continuity Equations. Memorandum Report 3237, Naval Research Laboratory, 1976.

[7] G. Patnaik, R. H. Guirguis, J. P. Boris, and E. S. Oran. A Barely Implicit Correction for Flux-Corrected Transport. *J. Comput. Phys.*, 71:1–20, 1987.

[8] T. R. Young and J. P. Boris. A Numerical Technique For Solving Stiff Ordinary Differential Equations Associated with the Chemical Kinetics of Reactive Flow Problems. *J. Phys. Chem.*, 81:2424, 1977.

[9] T. R. Young. CHEMEQ — A Subroutine for Solving Stiff Ordinary Equations. Memorandum Report 4091, Naval Research Laboratory, 1980.

[10] K. Kailasanath and E. S. Oran. Time-Dependent Simulations of Laminar Flames in Hydrogen-Air Mixtures. Memorandum Report 5965, Naval Research Laboratory, 1987.

[11] J. P. Boris. ADINC: An Implicit Lagrangian Hydrodynamics Code. Memorandum Report 4022, Naval Research Laboratory, 1979.

[12] W. W. Jones and J. P. Boris. An Algorithm for Multispecies Diffusion Fluxes. *Comp. Chem.*, 5:139–146, 1981.

[13] J. P. Boris. A Fluid Transport Algorithm That Works. In *Computing as a Language of Physics*, pages 171–189. International Atomic Energy Agency, Vienna, 1971.

[14] P. Woodward and P. Colella. The Numerical Simulation of Two-Dimensional Fluid Flow with Strong Shocks. *J. Comp. Phys.*, 54:115–173, 1984.

[15] E. S. Oran and J. P. Boris. *Numerical Simulation of Reactive Flow.* Elsevier, New York, 1987.

[16] M. J. Fritts and J. P. Boris. The Lagrangian Solution of Transient Problems in Hydrodynamics Using a Triangular Mesh. *J. Comp. Phys.*, 31:173–215, 1979.

[17] B. A. Fryxell, P. R. Woodward, P. Colella, and K-H Winkler. An Implicit-Explicit Hybrid Method for Lagrangian Hydrodynamics. *J. Comput. Phys.*, 62, 1986.

[18] H. C. Yee and A. Harten. Implicit TVD Schemes for Hyperbolic Conservation Laws in Curvilinear Coordinates. In *AIAA 7th Computational Fluid Dynamics Conference*, pages 228–241, Washington, DC, 1985. AIAA.

[19] V. Casulli and D. Greenspan. Pressure Method for the Numerical Solution of Transient, Compressible Fluid Flows. *Int. J. Num. Methods Fluids*, 4:1001–1012, 1984.

[20] C. R. DeVore. Vectorization and Implementation of an Efficient Multigrid Algorithm for the Solution of Elliptic Partial Differential Equations. Memorandum Report 5504, Naval Research Laboratory, 1984.

[21] T. L. Burks and E. S. Oran. A Computational Study of the Chemical Kinetics of Hydrogen Combustion. Memorandum Report 4446, Naval Research Laboratory, 1981.

[22] K. Kailasanath, E. S. Oran, and J. P. Boris. A One-Dimensional Time-Dependent Model for Flame Initiation, Propagation, and Quenching. Memorandum Report 4910, Naval Research Laboratory, 1982.

[23] D. L. Baluch, D. D. Drysdale, D. G. Horne, and A. C. Lloyd. *Evaluated Kinetic Data for High Temperature Reactions*, volume 1. Butterworths, London, 1972.

[24] R. F. Hampson and D. Garvin. Chemical Kinetic and Photochemical Data for Modelling of Atmospheric Chemistry. NBS Technical Note 866, National Bureau of Standards, Washington, 1975.

[25] N. Cohen and K. R.. Westberg. Data Sheets. Technical report, The Aerospace Corporation, P. O. Box 92957, Los Angeles, CA, 1979.

[26] D. B. Olson and W. C. Gardiner Jr. An Evaluation of Methane Combustion Mechanisms. *J. Phy. Chem.*, 81:2514, 1977.

[27] A. C. Lloyd. Evaluated and Estimated Kinetic Data for Phase Reactions of the Hydroperoxyl Radical. *Int. J. Chem. Kinetics*, 6:169, 1974.

[28] G.. S. Bahn. *Reaction Rate Compilation for H-O-N System.* Gordon and Breach, New York, 1968.

[29] T. A. Brun, T.R Young, and G. Patnaik. TBA — An Optimized Stiff ODE Solution Scheme. In preparation, 1989.

[30] P. J. Sydow. *Optimization Guide.* CRAY Research, Mendota Heights, 1983. CRAY Computer Systems Technical Note.

[31] S. Chapman and T. G. Cowling. *The Mathematical Theory of Non-Uniform Gases.* Cambridge University Press, Cambridge, 1970.

[32] T. P. Coffee and J. M. Heimerl. Transport Algorithms for Premixed Laminar Steady-State Flames. *Combust. Flame*, 43:273–289, 1981.

[33] J. O. Hirschfelder, C. F. Curtiss, and R. B. Bird. *Molecular Theory of Gases and Liquids.* John Wiley, New York, 1954.

[34] T. R. Marrero and E. A. Mason. Gaseous Diffusion Coefficients. *J. Phy. Chem. Ref. Data*, 1:3–118, 1972.

[35] C. F. Curtis and J. O. HIrschfelder. Transport Properties of Multicomponent Gas Mixtures. *J. Chem. Phys.*, 17:550, 1949.

[36] R. B. Bird, W. E. Stewart, and E. N. Lightfoot. *Transport Phenomena.* John Wiley, New York, 1960.

[37] K. J. Laskey. *Numerical Study of Diffusion and Premixed Flames.* PhD thesis, Department of Mechanical Engineering, Carnegie-Mellon University, Pittsburgh, PA, 1989.

[38] S. Mathur, P. K. Tandon, and S. C. Saxena. Thermal Conductivity of Binary, Ternary, and Quaternary Mixtures of Real Gases. *Mol. Phys.*, 12:569, 1967.

[39] C. R. Wilke. A Viscosity Equation for GAs Mixtures. *J. Chem. Phys.*, 18:517, 1950.

[40] N. N. Yanenko. *The Method of Fractional Steps.* Springer-Verlag, Ney York, 1971.

[41] K. Kailasanath, G. Patnaik, and E. S. Oran. Effect of Gravity on Multi-Dimensional Laminar Premixed Flame Structure. IAF Paper 88-354, IAF, Paris, 1988.

[42] G. Patnaik, K. Kailasanath, and E. S. Oran. Effect of Gravity on Flame Instabilities in Premixed Gases. AIAA Paper 89-0502, AIAA, Washington, DC, 1989.

[43] J. L. Ellzey, K. J. Laskey, and E. S. Oran. Study of Low-Speed Diffusion Flames. In preparation, 1988.

APPENDIX L.

Detailed Numerical Simulations of Cellular Flames

Twenty-Second Symposium (International) on Combustion/The Combustion Institute, 1988/pp 1517–1526

DETAILED NUMERICAL SIMULATIONS OF CELLULAR FLAMES

G. PATNAIK
Berkeley Research Associates, Springfield, VA 22150

K. KAILASANATH AND E. S. ORAN
Center for Reactive Flow and Dynamical Systems
Laboratory for Computational Physics and Fluid Dynamics
Naval Research Laboratory
Washington, D.C. 20375

K. J. LASKEY
Carnegie-Mellon University, Pittsburgh, PA 15213

Time-dependent, two-dimensional simulations of perturbed premixed laminar flames have been used to study the development of cellular structures in rich and lean hydrogen flames. The model includes detailed hydrogen-oxygen combustion with 24 elementary reactions of eight reactive species and a nitrogen diluent, molecular diffusion of all species, thermal conduction, and convection. Calculations of perturbed lean hydrogen flame evolution showed that the flame was unstable at the front, and the structure that evolved resembled cellular structures observed in experiments. The same perturbation applied to a rich hydrogen flame showed that the perturbation died out and cellular structures did not appear. Binary diffusion coefficients were varied to test the role of molecular diffusion in the development of cellular structure. When the coefficient of molecular hydrogen was set equal to that of molecular oxygen, the perturbation died out; when the coefficient of molecular oxygen was set equal to that of molecular hydrogen, the instability persisted. The results of the simulations support the diffusional-thermal theory of flame instability.

Introduction

Previous one-dimensional, time-dependent numerical simulations of laminar premixed flames have given physical insights and quantitative information on the effects of parameters such as stretch, curvature, and dilution on flame initiation and propagation, as well as qualitative information on flammability limits.[1,2] The structure of flames, especially near flammability limits, is usually multidimensional. To investigate such multidimensional effects, we have developed a time-dependent two-dimensional model that can simulate either premixed or diffusion flames. In this paper, we present results of calculations using this model to study premixed laminar flames in rich and lean hydrogen-oxygen mixtures diluted with nitrogen. These results are discussed in terms of current theories of flame stability and cellular structure formation, and the validity of two prominent theories is investigated.

Early experimental observations showed that cellular flames occur both in lean hydrogen-air and rich hydrocarbon-air mixtures such as propane and ethane.[3,4] Cellular flames are generally observed in lean mixtures when the fuel is lighter than air, and in rich mixtures when it is heavier than air.[4,5] These observations suggest that the relative diffusivities of fuel and air are crucial in determining the sensitivity to cellular instability, and lead to the formulation of the preferential diffusion theory.[5] By this theory, preferential diffusion of the lighter reactant creates regions of varying stoichiometry in front of a non-planar flame. This in turn causes portions of the flame to move at different speeds, making the flame front either less or more planar. This theory, however, does not explain all observations of cellular flames. For example, ethane and ethylene have molecular weights slightly less than that of oxygen, yet show cellular structure in rich mixtures.

A second theory, the diffusional-thermal theory[6,7] assumes an abundance of the excess component, so that the reaction is controlled only by the deficient component. For a simple one-step reaction in what is effectively a single reactant system, this theory predicts cellular instability when the thermal diffusivity of the mixture is sufficiently smaller than the mass diffusivity of the reactant. For lean hydrogen-air mixtures, hydrogen is the limiting reactant and its mass diffusivity significantly exceeds the thermal diffusivity of the mixture. In rich hydro-

gen-air mixtures, oxygen is the limiting reactant and its mass diffusivity is nearly the same as the mixture thermal diffusivity. Hence this theory agrees with early experimental observations of unstable lean hydrogen-air mixtures and stable rich mixtures.

A limitation of the diffusional-thermal theory is that it assumes constant fluid density, and hence neglects the effects of thermal expansion. Furthermore, in this theory, the cell size increases indefinitely as the Lewis number goes to its critical value. More recent experiments[8,9] have shown cellular instabilities in some rich and near-stoichiometric hydrogen-air mixtures, contradicting both theories. The diffusional-thermal theory is strictly valid only for strongly non-stoichiometric mixtures, although it has been extended[10,11] to near-stoichiometric mixtures by considering both the deficient and abundant components. According to this two-reactant theory, stability depends on the stoichiometry of the mixture, the difference between the Lewis numbers of the two components, and the reaction orders of each reactant. This suggests that in multicomponent systems, the Lewis numbers of many components and detailed chemical kinetics might play a role. The modified diffusional-thermal theory still neglects thermal expansion of combustion products.

The simulations of multidimensional flame structure presented here include a multi-reaction mechanism for hydrogen combustion, molecular diffusion between the reactants, intermediates, and products, thermal conduction, and convection. Such a detailed model should allow us to investigate the tendency to show cellular instability, evaluating the contributions of various physical processes, and the predictions of these theories. For example, previous explanations have considered the fuel and oxidizer, but the instability may also depend on the diffusivities of intermediate radicals. The effects of the fluid dynamics (and therefore the variable gas density) can also be evaluated. Below we describe the numerical model, show its results in simulating hydrogen-oxygen flames with different stoichiometries, and relate these results to the predictions of the two theories.

Multidimensional Flame Model

A detailed model of a flame must accurately represent convective, diffusive, and chemical processes. The importance of each process varies from rich to lean flames, and is especially notable near the flammability limits,[2] where the exact behavior depends on a delicate balance among them. In this section we briefly describe the algorithms used to model and couple the effects.

The fluid convection algorithm must maintain the sharp gradients present in flames. This means that numerical diffusion must be considerably less than any physical diffusion effect. The Barely Implicit Correction Flux-Corrected Transport (BIC-FCT) algorithm[12] solves the coupled continuity equations for a low-speed variable density flow. BIC-FCT combines an explicit nonlinear FCT method[13,14] with an implicit correction process. This combination maintains high-order accuracy, yet removes the timestep limit imposed by the speed of sound. Using FCT for the explicit step, BIC-FCT can accurately compute sharp gradients without overshoots and undershoots. Thus, unphysical chemical reactions from spurious oscillations do not occur. The computational cost per timestep of BIC-FCT is about the same as the standard explicit FCT method.[12]

Thermal conductivity of individual species is modeled by a polynomial fit in temperature to experimental data. Individual conductivities are then averaged using a mixture rule[15,16] to get the coefficient for the mixture. A similar process yields the mixture viscosity from individual viscosities. Heat and momentum diffusion are then calculated explicitly using these coefficients.

Mass diffusion also strongly effects the properties of laminar flames. Binary diffusion coefficients are represented by exponential fits to experimental data, and each species coefficient is obtained by mixture rules.[15] Individual species diffusion velocities are solved explicitly by Fick's law, with a correction to ensure zero net flux.[16] This procedure is equivalent to the algorithm DFLUX[17] which solves multi-component diffusion equations exactly. This method uses no matrix inversions.

Chemistry of the hydrogen-oxygen flame is modeled by 24 reversible reactions describing the interaction of eight reacting species, H_2, O_2, H, O, OH, HO_2, H_2O_2, H_2O, and N_2 as a non-reacting dilutent.[18] These are solved each timestep with TBA, a vectorized version of CHEMEQ, an integrator for stiff ordinary differential equations.[19] Because of the complexity of the reaction scheme, and the large number of cells in a two-dimensional calculation, the solution of the chemical rate equations takes a large fraction of the total computational time.

All chemical and physical processes are solved sequentially, then coupled asymptotically by timestep splitting.[20] This modularity greatly simplifies the model, and makes it easier to test and change. Individual modules were tested against known analytic and numerical solutions. One-dimensional predictions of the complete model were compared to those of the Lagrangian model FLAME1D, which has been benchmarked extensively against theory and experiment.[15] Exact values of input data for a hydrogen-oxygen flame, e.g. chemical reaction rates, thermodynamics parameters, and molecular diffusion coefficients, have been described in detail previously,[15] and hence are only referenced here.

Tests of Mechanisms of Laminar Flame Instability

Initial conditions for 2D calculations were taken from 1D calculations giving conditions for steady, propagating flames. Figure 1 describes the configuration studied, and gives the boundary conditions of the computational domain. Unburnt gas flows in from the left, and reaction products of the flame front flow out at the right. If the inlet velocity equals the burning velocity of the flame, the flame is fixed in space yielding a steady solution. Thus, transient effects from ignition can be eliminated, and the one-dimensional solution provides initial conditions for the two-dimensional calculation. The two-dimensional computational domain was 2.0 cm × 4.5 cm, resolved by a 56 × 96 variably spaced grid. Fine zones, 0.36 mm × 0.15 mm, were clustered around the flame front.

Flames in a Hydrogen-Lean Mixture:

The first calculation is of a flame in a fuel-lean hydrogen-oxygen mixture diluted with nitrogen, $H_2:O_2:N_2/1.5:1:10$, a flame that was clearly multidimensional in the experiments by Mitani and Williams.[9] The initial condition described by Fig. 1 is perturbed by displacing the center portion of the flame in the direction of the flow. Figure 2 shows isotherms just after the perturbation, and their subsequent evolution in time. The isotherms indicate temperature increases in the center portion of the flame, convex to the flow, and decreases in the two adjacent concave regions. This indicates more vigorous reaction in the convex region. The atomic hydrogen concentration increases in the convex and decreases in the concave regions. Also, the burning velocity in the convex region appears slightly higher than the burning velocity in a planar region, and in the concave regions noticeably lower. We conclude that this lean one-dimensional flame is unstable; a pattern resembling a cellular structure appears by the time the calculation is terminated.

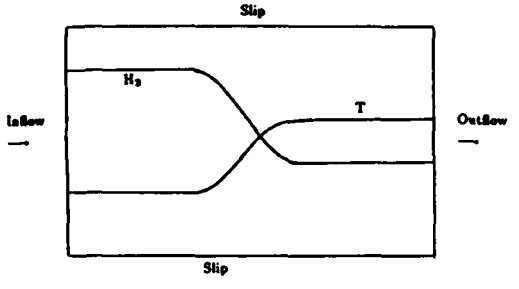


FIG. 1. Initial and boundary conditions for the two-dimensional flame calculations.

Flames in a Hydrogen-Rich Mixture:

Case two considers a disturbance evolving in a fuel-rich hydrogen-oxygen mixture, $H_2:O_2:N_2/3.0:1:16$. This mixture is stable in both the experiments of Mitani and Williams,[9] and our calculations. The initial disturbance quickly damps, restoring the flame profile to its one-dimensional shape. Figure 3 shows the evolution of isotherms for the rich flame. The perturbed center section rapidly straightens, indicating lower flame speed than in the planar sections. A lower number density of H in the center indicates a weaker reaction, hence a lower burning velocity. This is opposite to the behavior observed in lean flame calculations.

These results agree with experiments, and with both the preferential diffusion and diffusional-thermal theories. In the next two studies, the diffusion coefficients of the reactants (H_2 and O_2) were changed systematically to determine the role of mass diffusion. All other variables were held the same as in the fuel-lean case ($H_2:O_2:N_2/1.5:1:10$).

The Role of Mass Diffusion:

Equating the diffusion coefficient of hydrogen to that of oxygen causes the light and heavy reactants to diffuse at the same speed. This change stabilized the one-dimensional flame. Figure 4 shows that the isotherms straighten out and slowly return to the unperturbed planar condition. A lower H concentration was found in the center of the flame front, which is consistent with the observed flame stability.

The stability of the modified lean flame described above supports the preferential diffusion theory. Because the diffusion coefficients of hydrogen and oxygen are equal, preferential diffusion of hydrogen cannot cause instability. Stabilizing factors, such as heat conduction, tend to restore the flame to its unperturbed position. However, these results also agree with the diffusional-thermal theory. Since the Lewis number of the reactants in the modified lean flame is close to unity, the diffusional-thermal instability is greatly reduced, and stabilizing factors overcome any destabilizing tendencies.

A further test was needed to distinguish between these two mechanisms. The diffusion coefficients of O_2 were set to those of H_2, yielding two fast species. All other conditions were unchanged. This case again eliminates preferential diffusion; if preferential diffusion is indeed the mechanism, this flame should be stable. The diffusional-thermal theory, on the other hand, predicts that because the mass diffusion of the deficient species, H_2, remains high, this flame will be unstable. As Fig. 5 indicates, the flame is unstable and closely resembles the stan-

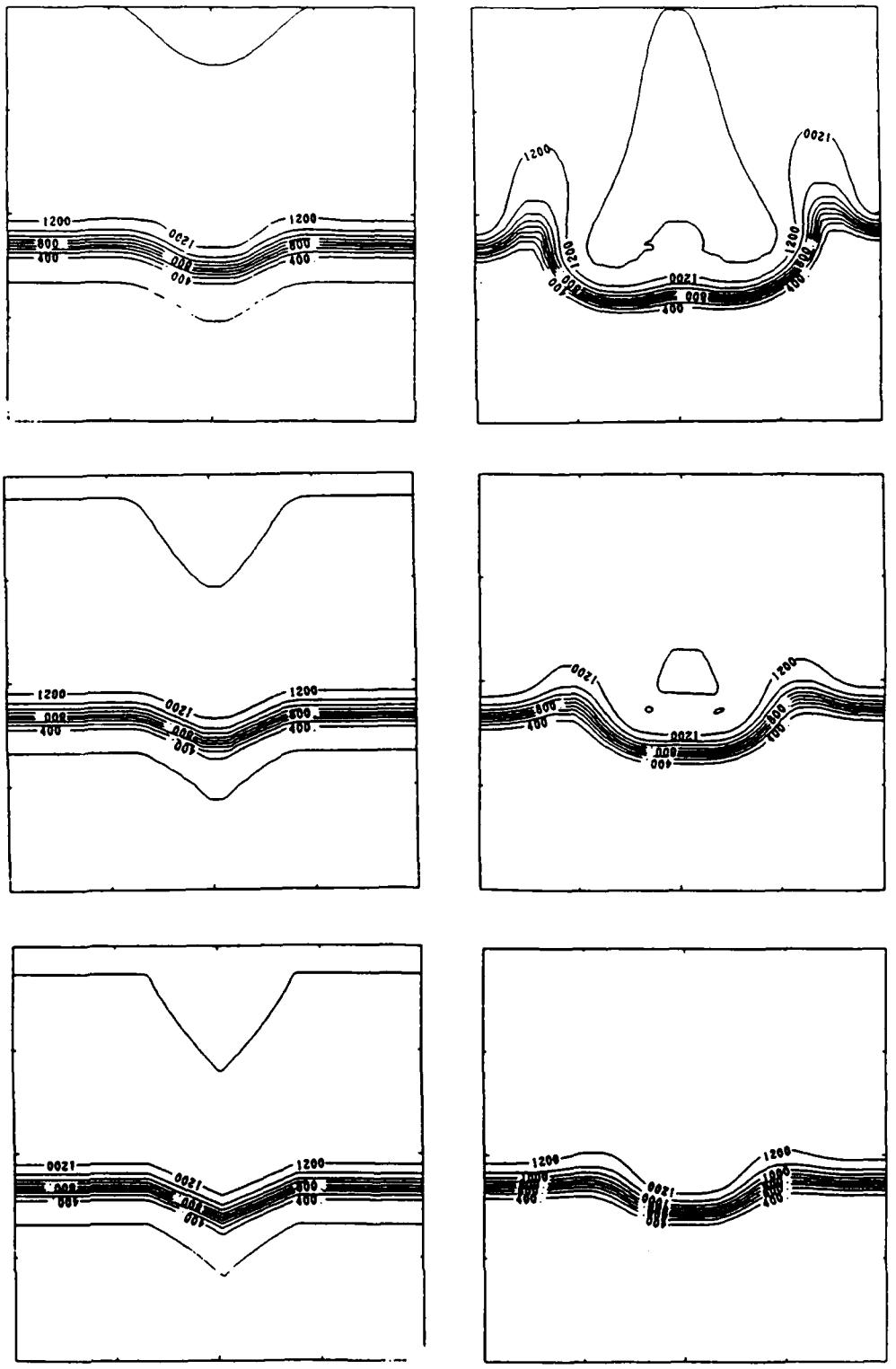


FIG. 2. Evolution of isotherms in a fuel-lean hydrogen flame, at 0.0, 0.5, 5.0, 15.0, 30.0, 50.0 msecs.

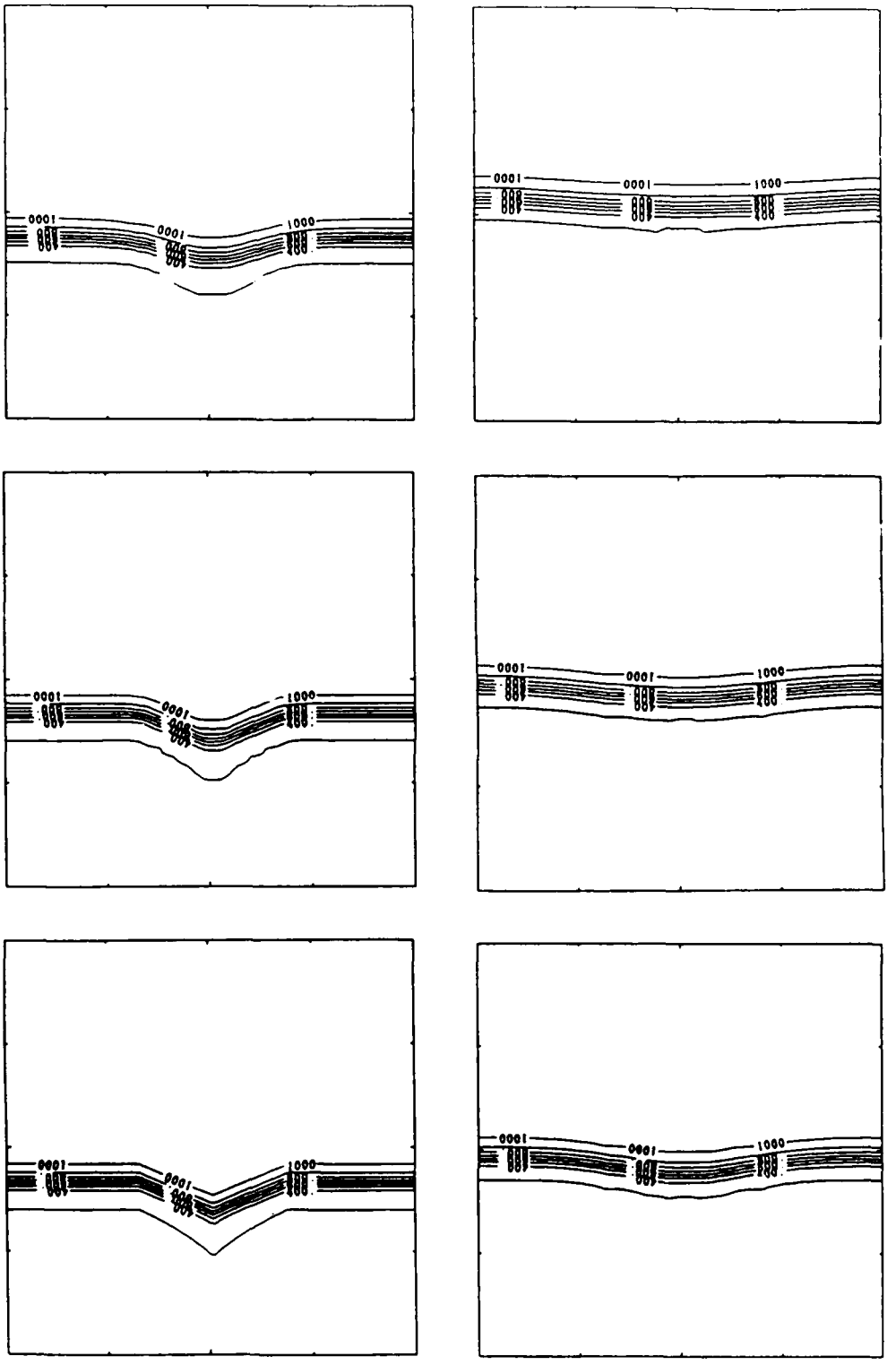


FIG. 3. Evolution of isotherms in a fuel-rich hydrogen flame.

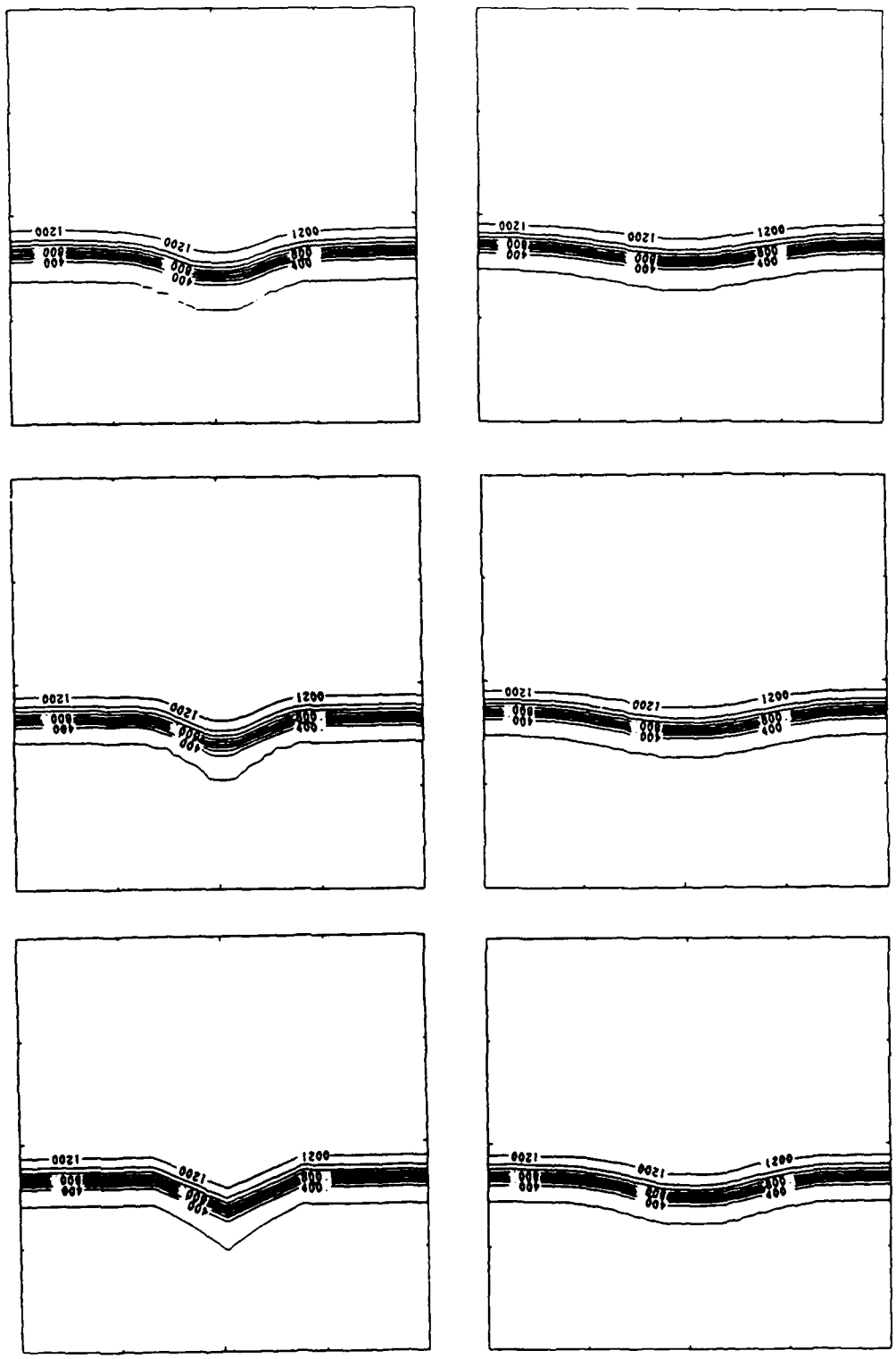


FIG. 4. Evolution of isotherms, diffusion coefficients of H_2 equal to O_2.

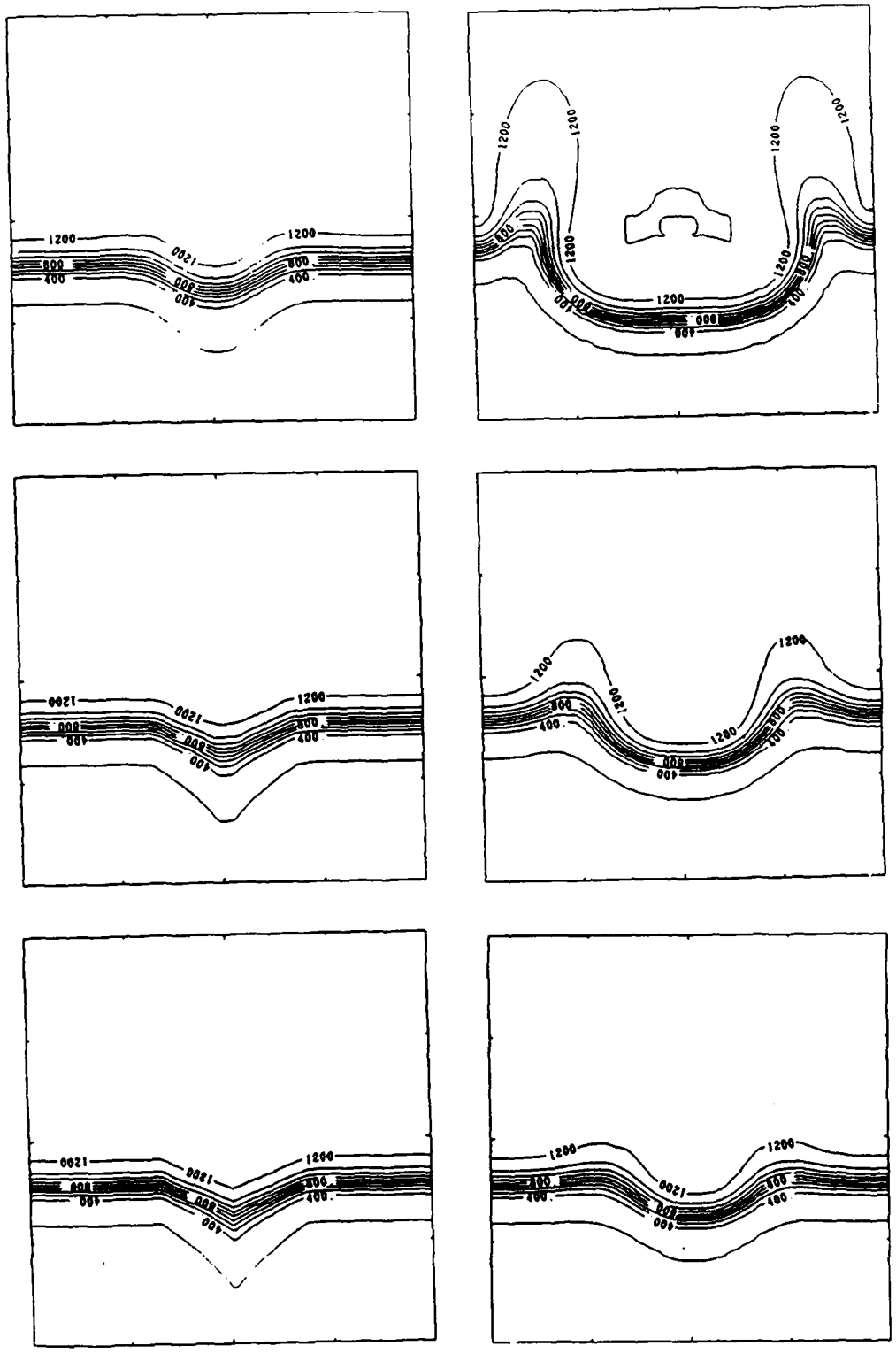


FIG. 5. Evolution of isotherms, diffusion coefficients of O_2 equal to H_2.

dard lean-flame result in Fig. 2. Thus, the simulations presented here support the diffusional-thermal theory of cellular instability.

Conclusion

A new time-dependent two-dimensional model was used to study the stability of rich and lean premixed laminar hydrogen-oxygen flames with nitrogen. The model included a detailed chemical reaction scheme consisting of 24 reversible reactions among eight species, thermal conductivity, and the individual species diffusion velocities. Four tests showed the evolution of perturbed flames:

1) A lean flame, $H_2:O_2:N_2/1.5:1:10$,
2) A rich flame, $H_2:O_2:N_2/3.0:1:16$,
3) The lean flame with the H_2 diffusion coefficients set equal to the O_2 diffusion coefficients, and
4) The lean flame with the O_2 diffusion coefficients set equal to the H_2 diffusion coefficients.

The simulations show that one-dimensional lean flame is unstable, and evolves into a pattern resembling a cellular flame. The rich flame, however, is stable; the perturbation dies in time. Both of these results agree with the experiments of Mitani and Williams[9] for mixtures with the same stoichiometry.

The third and fourth calculations attempt to determine the mechanism of the instability, by focusing on the mass diffusion of the reactants. In the third calculation, the mass diffusivity of hydrogen was set equal to oxygen. This eliminates preferential diffusion, and means that the light fuel moves no faster than the other reactant. If the instability mechanism were preferential diffusion, the lean flame would be stable. This is in fact the result: the flame was stable.

This result supports both the preferential diffusion and the diffusional-thermal theories. A further calculation sets the mass diffusivity of oxygen equal to that of hydrogen. In this case, the flame was unstable, disagreeing with the prediction preferential diffusion, but agreeing with those of diffusional-thermal theory. Thus, these hydrogen flame studies support the diffusional-thermal theory proposed by Barenblatt[6] and Sivashinsky.[7]

This series of calculations raised several computational issues. It would be interesting to eliminate certain limitations on the model. We would like the computational domain to be larger with the same or better resolution. A larger domain would allow central parts of the calculation to be less influenced by the boundary conditions; we could see if more cells form, and how they change in time. It would be useful to vary the perturbation at the front; for example, to alter its wavelength. This would address issues such as the necessary width of the perturbation relative to the flame thickness to destabilize lean flames, and the effects of long versus short wavelengths on the growth and type of instabilities.

The computations were very expensive. Each one cost between four and five hours of CRAY X-MP 2/4 time, which translates to 0.5 ms per point-step. Because of the expense, calculations were terminated even though it would have been interesting to follow them longer. We would like to inquire if the structure in Fig. 2 will split into more cells. The computations were bounded not by computer memory, but by CPU, with the most time spent in integrating the extensive chemical reactions. One possible approach, now that we have been able to reproduce experimentally observed results, is to consider a model with greatly simplified reactions. This appears justified in studying cellular instability, since chemistry does not appear to play a major role. The numerical model could be used to test theoretical concepts much less expensively, and perhaps less ambiguously.

There are, however, advantages to having the full simulation model. Other phenomena we wish to investigate, that occur near flammability limits, might be very sensitive to the details of the chemical reactions. The mass diffusivities of light intermediate species, such as atomic hydrogen, might also help determine stability. Now that the model exists, we can move in either direction.

Acknowledgments

This work was sponsored by NASA in the Microgravity Science Program, the Office of Naval Research through the Naval Research Laboratory, and the Pittsburgh Supercomputing Center. The authors would like to acknowledge a useful and interesting discussion with Professor C. K. Law. The précis was done by Todd Brun; he also optimized TBA.

REFERENCES

1. KAILASANATH, K. AND ORAN, E. S.: Prog. Astro. Aero. 105, Part 1, 167 (1986).
2. KAILASANATH, K. AND ORAN, E. S.: Time-Dependent Simulations of Laminar Flames in Hydrogen-Air Mixtures, Naval Research Laboratory Memorandum Report 5965, 1987.
3. MARKSTEIN, G. H.: J. Aero. Sci. 3, 18 (1951).
4. MARKSTEIN, G. H.: Non-Steady Flame Propagation, p. 78, Pergamon, 1964.
5. STREHLOW, R. A.: Fundamentals of Combustion, p. 221, R. E. Kreiger, 1979.

6. BARENBLATT, G. I., ZELDOVICH, Y. B. AND ISTRATOV, A. G.: Zh. Prikl. Mekh. Tekh. Fiz. 4, 21 (1962).
7. SIVASHINSKY, G. I.: Comb. Sci. Tech. 15, 137 (1977).
8. BREGON, B., GORDON, A. S. AND WILLIAMS, F. A.: Comb. Flame 33, 33 (1978).
9. MITANI, T. AND WILLIAMS, F. A.: Comb. Flame 39, 169 (1980).
10. JOULAN, G. AND MITANI, T.: Comb. Flame 40, 235 (1980).
11. SIVASHINSKY, G. I.: SIAM J. Appl. Math. 39, 67 (1980).
12. PATNAIK, G., GUIRGUIS, R. H., BORIS, J. P. AND ORAN, E. S.: J. Comput. Phys. 71, 1 (1987).
13. BORIS, J. P. AND BOOK, D. L.: Meth. Comput. Phys. 16, 85 (1976).
14. BORIS, J. P.: Flux-Corrected Transport Modules for Solving Generalized Continuity Equations, Naval Research Laboratory Memorandum Report 3237, 1976.
15. KAILASANATH, K., ORAN, E. S. AND BORIS, J. P.: A One-Dimensional Time-Dependent Model for Flame Initiation, Propagation, and Quenching, Naval Research Laboratory Memorandum Report 4910, 1982.
16. KEE, R. J., DIXON-LEWIS, G., WARNATZ, J., COLTRIN, M. E. AND MILLER, J. A.: A FORTRAN Computer Code Package for the Evaluation of Gas-Phase Multi-Component Transport Properties, Sandia National Laboratories Report SAND86-8246, 1986.
17. JONES, W. W. AND BORIS, J. P.: Comp. Chem. 5, 139 (1981).
18. BURKS, T. L. AND ORAN, E. S.: A Computational Study of the Chemical Kinetics of Hydrogen Combustion, Naval Research Laboratory Memorandum Report 4446, 1981.
19. YOUNG, T. R. AND BORIS, J. P.: J. Phys. Chem. 81, 2424 (1977).
20. ORAN, E. S. AND BORIS, J. P.: Numerical Simulation of Reactive Flow, p. 130, Elsevier, 1987.

COMMENTS

G. Joulin, CNRS, France. You didn't say anything about the flow field corresponding to the cellular flames; could you comment about that?

Author's Reply. The full equations for the compressible Naiver-Stokes flow were solved. There was some vorticity production at the curved flame front; however, the flow field was not greatly disturbed.

•

A. L. Kuhl, RDA, USA. You have presented results for a single spatial perturbation with a wavelength equal to the flame thickness. What would be the effect of multiple wavelengths, and thus are your conclusions sensitive to the initial perturbation assumed?

Your calculations have only investigated the early-time stability characteristics. Perhaps a more interesting problem would be the late-time cellular flame structure, and whether your 2D simulations agree with experiments. Multiple perturbation wavelengths would be needed for that case, and you should demonstrate that such late-time structure is independent of the perturbation assumed.

Author's Reply. We have not yet investigated the effect of varying the initial perturbation, we would like to do so. Such investigations will be useful.

The simulation of the lean flame do show the tendency of cells to split, as observed in experiments. However, we had insufficient resources to carry out the simulations further.

•

N. Peters, RWTH Aachen, Fed. Rep. of Germany. In varying the diffusivities of the fuel and oxygen you have expressed some skepticism in the one-step reaction asymptotic results of Barenblatt, Zeldowich, and Sivachinsky. Recently B. Rogg and myself have extended the asymptotic description to more realistic chemistry for methane flames and find very similar results as for one-step kinetics. This also seems to confirm the validity of the thermo-diffusional theory as you have demonstrated.

Author's Reply. We have not yet found an adequate one step reaction mechanism for hydrogen combustion. It would be interesting to compare such a simplified scheme to our complete reaction mechanism. This would shed light on the importance of the details of the mechanism on cellular instability.

•

F. A. Williams, Univ. of California, San Diego, USA. For your lean flame, I have felt that after some splittings, the structure will settle down to a number of nearly identical cells moving somewhat ir-

regularly and leaving unignited gas between (if lean enough). It would be interesting to carry your integration to longer times to see if this occurs. Is that possible?

Author's Reply. We do observe the initial spliting of the structure and the cooling down of the region between the cells. We need longer calculations to determine if the structures will settle down to a certain size. Our calculations were limited by budgetary considerations.

•

R. A. Strehlow, Univ. of Illinois, USA. First let me say that I take no credit for the preferential diffusion theory. I was only repeating an instant and reasonable explanation for the effect in my text.

Second, please try the case of a fuel lean mixture with $D_{H2} = D_{O2} > D_{N2}$ to see if it is unstable.

Author's Reply. Thank you, we will.

APPENDIX M.

Effect of Gravity on Flame Instabilities in Premixed Gases (AIAA-89-0502)

EFFECT OF GRAVITY ON FLAME INSTABILITIES IN PREMIXED GASES

G. Patnaik*, K. Kailasanath and E.S. Oran

Laboratory for Computational Physics and Fluid Dynamics
Naval Research Laboratory
Washington, D.C. 20375

*Berkeley Research Associates, Springfield, VA 22150

Abstract

Time-dependent, two-dimensional numerical simulations are used to investigate the effects of gravity on instabilities in laminar, premixed flames in lean H_2-O_2-N_2 mixtures. The calculations show that the effects of gravity become more important as burning velocity is decreased which occurs as the lean flammability limit is approached. In a 1.5:1:10 hydrogen-oxygen-nitrogen mixture with a burning velocity of 9.9 cm/s, gravity plays only a secondary role in determining the multidimensional structure of the flame. The stability and structure of the flame is controlled primarily by the thermo-diffusive instability mechanism. However, in a 1:1:10 hydrogen-oxygen-nitrogen mixture, in which the burning velocity is 2.0 cm/s, gravity is more important. Here the upward-propagating flame is highly curved and evolves into a bubble rising upwards in the tube; the zero-gravity flame shows a cellular structure; the structure of the downward-propagating flame oscillates between structures with concave and convex curvatures towards the unburnt mixture. These observations are explained on the basis of an interaction between the buoyancy-induced Rayleigh-Taylor instability and the thermo-diffusive instability. A theoretical dispersion relation describing the growth rate of a disturbance was evaluated at the conditions of the numerical simulations. Some of the trends it predicted were correct, but no quantitative agreement was found. However, the idealized nature of the theory makes comparison of its predictions to numerical results difficult.

I. Introduction

Instabilities are often observed in propagating flames, especially near the flammability limits. In this paper, we discuss the various types of instabilities that may arise in such premixed flames and use numerical simulations to isolate and study their properties. The emphasis of this paper is on the effects of gravity on flame instabilities in gases of premixed hydrogen, oxygen, and nitrogen.

Linear stability analyses can provide information on the roles of various processes at the onset of instability. However, the prediction of the growth of this instability to the final form is beyond the scope of these analyses. Numerical calculations can be used to help understand both the onset of the instability and the evolutionary process that produces the multidimensional structure. The numerical simulations of flames presented here include as input a multireaction mechanism for hydrogen combustion, molecular diffusion between the reactants, intermediates, and products, thermal conduction, convection, and gravity. Such a detailed model allows us to investigate the multidimensional structure of flames and to evaluate the importance of various contributing physical processes. We use the numerical simulations to evaluate the relative importance of these processes in normal earth gravity and zero-gravity conditions.

The three major instabilities that might occur in premixed flames are: the hydrodynamic instability, independently proposed by Landau and Darrieus[1,2]; the thermo-diffusive instability investigated by Barenblatt and Zeldovich[3,4]; and the buoyancy-induced instability generally called the Rayleigh-Taylor instability[5]. Because our numerical model of the premixed hydrogen flame includes all the physical mechanisms that lead to these three types of instabilities, it provides an ideal test-bed to study these instabilities and their interactions.

Hydrodynamic instability was the first kind of flame instability studied theoretically. The analyses of Darrieus[2] (1938) and Landau[1] (1944) showed that a planar flame, considered as a density discontinuity that propagates normal to itself at a constant speed, is unstable to wavelengths of all sizes with the smaller wavelengths growing faster. Physically, hydrodynamic instabilities are due to the fluid expansion and can be expected to occur in all flames in the absence of stabilizing mechanisms. The actual existence of this mode of instability has never been conclusively shown because of the difficulty in isolating and studying such an idealized flame experimentally. This difficulty in observing hydrodynamically unstable flames suggests that stabilizing mechanisms are important in real flames. For very small wavelengths that are of the order of the flame thickness, the hypothesis that a flame can be treated as a discontinuity is invalid. In fact, the phenomenological analysis of Markstein[6] (1951) has shown that introducing a characteristic length of the order of the flame thickness has a stabilizing effect on the short wavelength modes. It might still be possible to see hydrodynamic instabilities for wavelengths much greater

than the flame thickness, but stable flames have been observed in systems with the characteristic dimensions even a hundred times greater than the flame thickness. This suggests that other effects such as buoyancy, stretch, and heat losses play an important role in the stability of real flames.

The thermo-diffusive instability mechanism proposed by Zeldovich[4], Barenblatt et al.[3], and Sivashinsky[7,8] invloves a competition between mass diffusion of the deficient reactant and diffusion of heat in the mixture This instability can lead to the formation of cellular flames. This mechanism assumes an abundance of the excess component, so that the extent of the reaction is controlled only by the deficient component. For a simple one-step reaction in what is effectively a single reactant system, this mechanism predicts the formation of cellular structure whenever the thermal diffusivity of the mixture is sufficiently smaller than the mass diffusivity of the reactant. For lean hydrogen-air mixtures, hydrogen is the limiting reactant and its mass diffusivity significantly exceeds the thermal diffusivity of the mixture. In rich hydrogen-air mixtures, oxygen is the limiting reactant and its mass diffusivity is nearly the same as the mixture thermal diffusivity. Hence this theory agrees with early experimental observations[9,10] of unstable lean hydrogen-air mixtures and stable rich mixtures. More recent experiments[11,12] have shown cellular flames in some rich and near-stoichiometric hydrogen-air mixtures, contradicting this simple theory. The theory has been extended[13,14] to near-stoichiometric mixtures by considering both the deficient and abundant components.

The Rayleigh-Taylor instability occurs when a heavier fluid is accelerated into a lighter fluid[2]. On earth, this acceleration is provided by gravitational attraction. In an upward-propagating flame, the light, hot burned material is on the bottom, and the dense, cold unburned material is on the top, resulting in instability. In a downward-propagating flame. the light material is on the top, and the Rayleigh-Taylor mechanism stabilizes the system. The physical mechanisms causing the thermo-diffusive instability and this buoyancy-induced instability can be important simultaneously so that under certain conditions, this interaction appears to suppress the formation of cellular structure[15]. Dimensional arguments[16] and theoretical analysis[17] indicate that the importance of the buoyancy-induced instability increases as the flame speed decreases, and hence is more important when the mixture is near its flammability limits.

There are real mixtures near the flammability limit for which the thermo-diffusive mechanism and the buoyancy-induced mechanism compete. Cellular structures in flames have been observed in the microgravity experiments in the NASA drop tower[18] and Learjets[19]. These flames in microgravity are free from any buoyancy-induced instability, thus making it possible to examine experimentally only the thermo-diffusive instability. Another approach to isolate these instabilities is by the detailed numerical modelling of flames.

Our numerical simulations of premixed flames contain detailed models of the physical processes that cause the various instabilities. Previously we have used simulations to show that, in the absence of gravity, cellular structure is caused by the thermo-diffusive mechanism[20]. We have shown that this mechanism is correct for mixtures sufficiently far from stoichiometric. The goal of this paper is to investigate the effect of gravity on flame instability and, in particular, to study the interaction between the thermo-diffusive instability and the buoyancy-induced Rayleigh-Taylor instability.

II. Multidimensional Flame Model

A detailed model of a flame must contain accurate representations of the convective, diffusive, and chemical processes. The individual importance of these processes varies from rich to lean flames, and is especially notable near the flammability limits[21] where the exact behavior of these flames depends on a delicate balance among the processes. In this section we briefly describe the algorithms and input data used to model and couple the effects.

The fluid convection algorithm must be able to maintain the sharp gradients present in flames. Numerically this means that any important numerical diffusion in the calculation must be considerably less than any physical diffusion effect. Many explicit algorithms now exist that treat sharp discontinuities in flow variables accurately, but these methods are extremely inefficient at the very low velocities associated with laminar flames. The Barely Implicit Correction Flux-Corrected Transport (BIC-FCT) algorithm[22] was developed specifically to solve low-speed flow problems with high accuracy. BIC-FCT combines an explicit high-order, nonlinear FCT method[23,24] with an implicit correction process. This combination maintains high-order accuracy and yet removes the timestep limit imposed by the speed of sound. By using FCT for the explicit step, BIC-FCT is accurate enough to compute with sharp gradients without overshoots and undershoots. Thus spurious numerical oscillations that would lead to unphysical chemical reactions do not occur.

Thermal conductivity of the individual species is modeled by a polynomial fit in temperature to existing experimental data. Individual conductivities are then averaged using a mixture rule[25,26] to get the thermal conductivity coefficient of the gas mixture. A similar

process is used to obtain the mixture viscosity from individual viscosities. Heat and momentum diffusion are then calculated explicitly using these coefficients. In the problem considered in this paper, the timestep restriction of an explicit method for the diffusion terms does not cause any loss in efficiency.

Mass diffusion also plays a major role in determining the properties of laminar flames. Binary mass diffusion coefficients are represented by an exponential fit to experimental data, and the individual species diffusion coefficients are obtained by applying mixture rules[25]. The individual species diffusion velocities are solved for explicitly by applying Fick's law followed by a correction procedure to ensure zero net flux[26]. This procedure is equivalent to using the iterative algorithm DFLUX[27] to second order. This method is substantially faster than one that uses matrix inversions and is well suited for a vector computer. This algorithm is also explicit, but because the effective Lewis number of the mixture is close to one, the timestep suitable for heat conduction is adequate for mass diffusion as well.

Chemistry of the hydrogen–oxygen flame is modelled by a set of 24 reversible reaction rates describing the interaction of eight species, H_2, O_2, H, O, OH, HO_2, H_2O_2, H_2O, and N_2 is considered a nonreacting diluent[28]. This reaction set is solved at each timestep with a vectorized version of CHEMEQ, an integrator for stiff ordinary differential equations[29]. Because of the complexity of the reaction scheme and the large number of cells in a two-dimensional calculation, the solution of the chemical rate equations takes a large fraction of the total computational time. A special version of CHEMEQ called TBA was developed to exploit the special hardware features of the CRAY X-MP vector computer.

All of the chemical and physical processes are solved sequentially and then are coupled asymptotically by timestep splitting[30]. This modular approach greatly simplifies the model and makes it easier to test and change the model. Individual modules were tested against known analytic and other previously verified numerical solutions. One-dimensional predictions of the complete model were compared to those from the Lagrangian model FLAME1D which has been benchmarked extensively against theory and experiment[25].

III. Results

Initial conditions for the two-dimensional calculations were obtained by performing a one-dimensional calculation to provide the conditions for steady, propagating flames. Figure 1 describes schematically the configuration under study and gives the boundary conditions of the computational domain. Fresh unburnt gas flows in from the left, and the products of chemical reaction at the flame front flow out at the right. If the inlet velocity is set to the burning velocity of the flame, the flame zone is fixed in space and there is a steady, propagating flame. Thus, the transient effects arising from the ignition process can be eliminated and the one-dimensional solution provides a relaxed initial condition for the two-dimensional calculation. This steady, one-dimensional solution was compared to and found in agreement with a similar calculation with FLAME1D[25]. The one-dimensional solution was used to provide the initial conditions for the two-dimensional calculation. The two-dimensional computational domain was 2 0 cm × 4.5 cm, which was resolved by a 56 × 96 variably spaced grid. Fine zones, 0.36 mm × 0.15 mm, were clustered around the flame front.

Flames in a H_2:O_2:N_2 / 1.5:1:10 Mixture

The first calculation is of a zero-gravity flame in a fuel-lean mixture of hydrogen and oxygen diluted with nitrogen, H_2:O_2:N_2 / 1.5:1:10, a flame that was clearly multidimensional in the experiments in earth gravity by Mitani and Williams[12]. The initial condition described by Fig. 1 is perturbed by displacing the center portion of the flame in the direction of the flow. The frames in the top half of Fig. 2 show isotherms just after the perturbation and then their evolution at subsequent times. The isotherms indicate that the temperature increases in the center portion of the flame which is convex to the flow and decreases in the the two adjacent concave regions. This indicates that the reaction progresses more vigorously in the convex region, a conclusion corroborated by the intermediate species (OH) number-density contours shown in the bottom half of Fig. 2. Figure 2 also shows that the concentration of OH increases in the convex region and decreases in the concave regions. Also, the burning velocity in the convex region appears to be slightly highe. than in a planar region, while in the concave regions the burning velocity has noticeably decreased. We conclude that in this lean mixture, a planar flame is unstable and a pattern resembling a cellular structure has appeared by the time the calculation was terminated.

Figure 3 compares this zero-gravity flame to flames propagating upward and downward in this same mixture, so that the flames are propagating opposite to and in the direction of gravity. Initially (up to 20 ms) all of these flames are similar; they are clearly cellular with noticeable though not very significant differences. The thickness of the flame is affected by gravity, with the upward-propagating flame being thicker than the downward-propagating flame. The figure also indicates that the upward-propagating flame transitions to a cellular flame more rapidly than the downward-propagating flame, and the zero-gravity results are intermediate. These observations suggest that buoyancy-induced instability enhances the growth of the initial

disturbance in the upward-propagating flame and retards the growth in the downward-propagating flame.

The fact that a planar flame is unstable in all three cases and evolves to a cellular structure is consistent with our previous results that show that cellular structure is due to a thermo-diffusive instability mechanism. Our current calculations show that in this mixture, the effect of buoyancy through the Rayleigh-Taylor instability is not substantial. In the case of downward propagation, if the mixture were not prone to cellular instability, we would expect the buoyancy-induced instability to damp the initial disturbance and return the flame to a planar configuration.

Flames in a H_2:O_2:N_2 / 1:1:10 Mixture

A similar comparison as in Fig. 3 has been made in Fig. 4 for flames propagating in a H_2:O_2:N_2 / 1:1:10 mixture. For this mixture, the differences among the three cases are more dramatic. The zero-gravity case again shows a cellular structure but with a larger cell size than in the previous mixture. At 20 ms, the shape and size of the three flames are comparable. The zero-gravity flame exhibits a stable cellular structure with little change from 20 to 100 ms. The upward-propagating flame becomes more and more curved and the central portion of the flame moves more rapidly than the sides. In this case, the buoyancy-induced and the thermo-diffusive mechanisms are both destabilizing. At 100 ms, the upward-propagating flame exhibits a bubble-like appearance characteristic of near-limit upward-propagating flames. In downward propagation, buoyancy tends to stabilize and return a perturbed flame to its initial planar configuration. However, at later times, the flame front begins to oscillate around its planar configuration. The fact that early in the calculations, at 20 ms, all three flames look alike suggests that the thermo-diffusive instability is growing more rapidly than thr buoyancy-induced instability. By 100 ms, the buoyancy-induced instability has had enough time to strongly interact with the thermo-diffusive instability and in the case of the downward-propagating flame it essentially nullifies the effects of the thermo-diffusive instability.

Upward-Propagating Flames. In Fig. 5, temperature and OH concentration profiles are shown at four late times in the evolution of the upward-propagating flame. The reaction zone, as indicated by OH profiles, is more vigorous in the center of the flame than near the walls. With time, this difference in the reaction rates increases and the flame appears to be stretched as the central portion moves more rapidly than the sides. The temperature contours substantiate this observation because they show that the overall thickness of the flame is greater at the walls than in the center. For flames in lean premixed hydrogen-oxygen-nitrogen mixtures, a larger flame thickness corresponds to a slower flame[31]. With further dilution and perhaps for later times even at this dilution, these results suggest that the central portion of the flame might break away and rise as a bubble. Experimental observations of such a phenomenon has been observed near the lean flammability limit[32]. The effects of heat and radical losses to the walls might also play a role in the actual extinction of these flames.

Downward-Propagating Flames. The long-term evolution of the downward-propagating flame is compared to that of the zero-gravity flame in Fig. 6. The zero-gravity flame is cellular and remains so with time. The downward-propagating flame is nearly planar at 60 ms, develops a concave front towards the unburnt mixture by 100 ms, and appears to show a cellular structure again by 200 ms. At an intermediate time (160 ms), it goes through a nearly planar stage. The structure changes from 160 ms to 220 ms because a planar flame in this mixture is unstable to the thermo-diffusive instability. However, as discussed earlier, the buoyancy-induced instability tends to suppress a multidimensional structure for downward-propagating flames. The flame fronts at later times than shown here would probably show the curvature decreasing and the flame becoming nearly planar, then the front becoming concave and the cycle repeating itself. Downward-propagating flames in earth gravity sometimes oscillate near the flammability limits[33], a behavior characteristic of waves in fluids stabilized by buoyancy.

IV. Discussion

The dimensional arguments presented by Williams[16] and the theoretical analysis of Clavin[17] indicate that the buoyancy-induced instability becomes more important as the flame speed decreases, a result that is clear from our numerical results as well. The calculations for the 1.5:1:10 lean hydrogen flame, with a burning velocity of 9.9 cm/s, indicate that gravity plays only a secondary role in the evolution of cellular structure and that the stability of the flame is controlled primarily by the thermo-diffusive mechanism. Some differences in the flame structure can be seen 20 ms after the initial perturbation between upward-propagating, zero-gravity, and downward-propagating flames, though later the structures again appear similar.

The 1.5:1:10 mixture, though quite lean, is still far from the flammability limit. Therefore, we have performed computations for a 1:1:10 mixture with a still lower burning velocity. In this mixture the effect of gravity was more significant. An upward-propagating flame is highly curved and evolves into a bubble rising upwards in the tube. The zero-gravity flame shows a cellular flame structure. The structure of

the downward-propagating flame changes in time, exhibiting both concave and convex curvatures towards the unburnt mixture. These structures result from the interplay of the processes controlling the buoyancy-induced instability and the thermo-diffusive instability.

In every instance, the effect of an initial perturbation on a planar front is first to produce a cellular structure. Then for later times, buoyancy-induced instability tends to make the upward-propagating flame more curved and the downward-propagating flame less curved. These observations and the observation that a cellular structure characteristic of the thermo-diffusive instability appears cyclically in the downward propagating case indicates that this instability grows more rapidly than the buoyancy-induced instability.

Comparision with Linear Stability Analysis

A theoretical analysis combining the effects of the hydrodynamic, thermo-diffusive, and the buoyancy-induced intstability has been carried out by Pelecé and Clavin[34,17]. This analysis gives a dispersion relation for the initial growth rate of a disturbance as a function of its wave number. In an attempt to correlate the theory's predictions to our numerical results, we have evaluated the growth rate for a disturbance with a wavelength of 1 cm at the conditions found in our simulations (see Appendix). The result is that the theory is in qualitiative agreement with certain but not all of the numerical observations. For example, in the 1:1:10 mixture, the theory correctly predicts the oscillatory behavior found in downward propagation and also indicates that the effect of gravity would be stronger. The theory incorrectly predicts that downward propagation in the 1.5:1:10 mixture exhibits damped oscillations.

However, the intepretation of the theory's results is not straightforward. We have only investigated the growth rate for a 1 cm disturbance, though our initial disturbance in the numerical simulations was not purely sinusiodal. In addition, we discovered that the growth rate is sensitive to the input parameters, though the results presented in the Appendix are evaluated at the conditions specified by the theory. The theory make several assumptions which are unrealistic. For example, this theory considers only a single chemical reaction which can be represented by only one activation energy, which is a poor representation of the hydrogen-oxygen chemistry. The results of the theoretical calculations are thus presented in the Appendix for a representative range of activation energies.

V. Summary and Conclusions

A detailed two-dimensional numerical simulation of flame instabilities in lean H_2-O_2-N_2 mixtures has been carried out for two mixtures near the flammability limit with mixture ratios of 1.5:1:10 and 1:1:10 with flame speed of 9.9 cm/s and 2.0 cm/s respectively. Physical processes included in the model are: fluid convection, detailed hydrogen-oxygen chemistry, species diffusion, thermal conduction, viscosity, and gravity. The effects of buoyancy-induced instability was found to be small in the 1.5:1:10 mixture, though its effect on the 1:1:10 mixture was more dramatic. In this leaner mixture the upward propagating flame had the characteristic bubble shape observed experimentally and the downward propagating flame had oscillations characteristic of thr Rayleigh-Taylor instability. These results agree with the theory[17] that indicates that the influence of gravity is greater for lower flame speeds.

A comparison of the theoretical dispersion relation which predicts the growth rate of instabilities was carried out at the conditions in our numerical calculations. Some qualitative agreement was observed: the theory predicts the oscillatory behavior in downward propagating flames and that the effect of gravity is more marked in the leaner mixture. However, some other theoretical predictions were found to be incorrect. The discrepancies may arise from the sensitivity to input parameters or from an unrealistic assumption in the theory . Even though the comparison of the theory to the numerical calculations is not exact, this combined approach is quite promising. Because of the extreme flexibilty of numerical simulations, it is possible to carry out a range of experiments which cannot be performed in the laboratory. The numerical simulations provide the necessary details to verify and improve the theory. A two-way interaction between theory and numerical simulation can lead to a step-up in the pace of our understanding of flame instabilities.

Calculations for leaner mixtures are needed to address the actual extinction behavior of upward and downward-propagating flames. Loss mechanism such as heat and radical losses to the walls as well as radiation might also play a role in determining the detailed extinction behavior of these flames. These effects will be systematically considered in further calculations.

Acknowledgements

This work was sponsored by NASA in the Microgravity Science Program and by the Office of Naval Research through the Naval Research Laboratory. Thanks to D. L. Book from the Naval Research Laboratory for his help with the interpretation of the dispersion relation.

References

1. Landau, L., *Acta Physicochim. URSS* 19 77 (1944).
2. Darrieus, G., Propagation d'un front de flamme, unpublished works presented at La Technique Moderne in (1938) and at Congrès de Méchanique Appliquée Paris (1945).

3. Barenblatt, G. I., Zeldovich, Y. B. and Istratov, A. G., *Zh. Prikl. Mekh. Tekh. Fiz.* **4** 21 (1962).
4. Zeldovich, Y. B., The Theory of Combustion and Detonation, Academy of Sciences, USSR, 1944. Also, Zeldovich, Y.B., and Drosdov, N.P., *J. Experi. Theoret. Physics.* **17** 134 (1943).
5. Lord Rayleigh, Scientific Papers, Dover, New York, 1964.
6. Markstein, G.H., *J. Aero. Sci.* **18** 199 (1951).
7. Sivashinsky, G.I., *Combust. Sci. Tech.* **15** 137 (1977).
8. Sivashinsky, G.I., *Ann. Rev. Fluid Mech.* **15** 179 (1983).
9. Markstein, G.H., Non Steady Flame Propagation, p. 78, Pergamon, New York, 1964.
10. Strehlow, R. A.,Fundamentals of Combustion, p. 221, R.E. Kreiger, New York, 1979.
11. Bregon, B., Gordon, A. S. and Williams, F.A., *Combust. Flame.* **33** 33 (1978).
12. Mitani, T. and Williams, F. A., *Combust. Flame* **39** 169 (1980).
13. Joulan, G. and Mitani, T., *Combust. Flame* **40** 235 (1980).
14. Sivashinsky, G. I., *SIAM J. Appl. Math.* **39** 67 (1980).
15. Von Lavante, E., and Strehlow, R.A., *Combust. Flame* **49** 123 (1983).
16. Williams, F. A., Combustion Theory, Benjamin/Cummings, Menlo Park, 1985.
17. Clavin, P., Prog. Energy Combust. Sci. **11** 1 (1895).
18. Ronney, P.D., *Combust. Flame.* **62** 121 (1985).
19. Strehlow, R.A., Work presented at the NASA Microgravity Combustion Science Working Group Meeting, NASA-LRC, Cleveland, Ohio, Oct. 1984. also Strehlow, R.A., Noe, K.A., and Wherley, B.L., The Effect of Gravity on Premixed Flame Propagation and Extinction in a Vertical Standard Flammability Tube, *Proceedings of the 21st Symposium (International) on Combustion*, The Combustion Institute, Pittsburgh, PA., 1988.
20. Patnaik, G., Kailasanath, K., Laskey, K.J., and Oran, E.S., Detailed Numerical Simulations of Cellular Flames, to appear in the *Proceedings of the 22th Symposium (International) on Combustion*, The Combustion Institute, Pittsburgh, 1988.
21. Kailasanath, K., Oran, E.S., Time-dependent Simulations of Laminar Flames in Hydrogen-Air Mixtures, Proceedings of 2nd Workshop on Modelling of Chemical Reaction Systems, Springer-Verlag, Heidelberg, 1987.
22. Patnaik, G., Guirguis, R.H., Boris, J.P., and Oran, E.S., *J. Comput. Phys.* **71** 1 (1987).
23. Boris, J.P., and Book, D.L., *Meth. Comput. Phys.* **16** 85 (1976).
24. Boris, J.P., *Flux-Corrected Transport Modules for Solving Generalized Continuity Equations*, Naval Research Laboratory Memorandum Report 3237, 1976.
25. Kailasanath, K., Oran, E.S., and Boris, J.P., *A One-Dimensional Time-Dependent Model for Flame Initiation, Propagation and Quenching*, Naval Research Laboratory Memorandum Report 4910, 1982.
26. Kee, R.J., Dixon-Lewis, G., Warnatz, J., Coltrin, M.E., and Miller, J.A., *A Fortran Computer Code Package for the Evaluation of Gas-Phase Multi-Component Transport Properties*, Sandia National Laboratories Report SAND86-8246, 1986.
27. Jones, W.W., and Boris, J.P., *Comp. Chem.* **5** 139 (1981).
28. Burks, T.L., and Oran, E.S., *A Computational Study of the Chemical Kinetics of Hydrogen Combustion*, Naval Research Laboratory Memorandum Report 4446, 1981.
29. Young, T.R., and Boris, J.P., *J. Phys. Chem.* **81** 2424 (1977).
30. Oran, E.S., and Boris, J.P., *Numerical Simulation of Reactive Flow*, Elsevier, New York, 1987.
31. Kailasanath, K., and Oran, E.S., *Prog. Aero. and Astro.* **105** Part 1 167 (1986).
32. Levy, A., *Proc. Roy. Soc. London* **A 283** 134 (1965).
33. Jarosinski, J., *Prog. Energy Combust. Sci.* **12** 81 (1986).
34. Pelecé, P., and Clavin, P., *J. Fluid Mech.* **124** 219 (1982).

Appendix

We have evaluated the dispersion relation for the combined effects of the hydrodynamic, thermo-diffusive, and buoyancy-induced Rayleigh-Taylor instabilities of flame structure[17] at conditions in our numerical simulations. The relation, expressing the dependence of the growth rate σ to the wave number k of the disturbance, is:

$$\mathbf{A}(k)\sigma^2 + \mathbf{B}(k)\sigma + \mathbf{C}(k) = 0,$$

where

$$\mathbf{A}(k) = (2-\gamma) + \gamma\left(\frac{\mathcal{L}}{d} - \frac{1}{\gamma}\log\frac{1}{1-\gamma}\right)\frac{\rho_b D_{th}}{\rho_u u_L}k,$$

$$\mathbf{B}(k) = 2u_L k + \frac{2}{1-\gamma}\left(\frac{\mathcal{L}}{d} - \log\frac{1}{1-\gamma}\right)\frac{\rho_b D_{th}}{\rho_u}k^2,$$

$$\mathbf{C}(k) = \frac{\gamma}{1-\gamma}u_L^2 k\left(\frac{1}{2}k_c - k\left(\delta - \frac{k}{k^*}\right)\right),$$

and

$$\delta = 1 + \frac{gd}{u_L^2}(1-\gamma)\left(\frac{\mathcal{L}}{d} - \frac{1}{\gamma}\log\frac{1}{1-\gamma}\right),$$

$$k^* = \left(1 + \frac{2+\gamma}{\gamma}\frac{\mathcal{L}}{d} - \frac{2}{\gamma}\log\frac{1}{1-\gamma}\right)^{-1}\frac{\rho_u u_L}{\rho_b D_{th}},$$

$$k_c = \frac{2g(1-\gamma)}{u_L^2}.$$

The Markstein length $\mathcal{L}$ is given in terms of the flame thickness d by

$$\mathcal{L} = \left(Ze \frac{(Le-1)}{2} + 1 \right) d,$$

where Ze is the Zeldovich number, and Le the Lewis number.

The following numerical values were used for the physical properties: $D_{th} = 0.293$; $Le = 0.378$; with $\rho_b/\rho_u = 0.265$, $\gamma = 0.739$, $d = 0.55$ cm, $u_L = 2.0$ cm/s for the 1:1:10 mixture. The values for $\rho_b/\rho_u, \gamma$, d, and u_L for the 1.5:1:10 mixture are $0.240, 0.760, 0.35$ cm, and 9.9 cm/s respectively. Three values of the Ze corresponding to activation energy E of 10, 20, 30 kcal/mole were used. This is the only number for which there is no exact correspondence to the numerical model, which includes a full chemical kinetics mechanism.

The following table gives the value obtained for σ for $k = 2\pi$, which corresponds to a wavelength of 1 cm. Negative values of σ have not been presented, because this mode is expected to decay rapidly.

Growth Rates for $k = 2\pi$

E	Upward	Zero G	Downward
H_2:O_2:N_2 / 1:1:10			
10	103.0	15.8	$0.9 \pm 100.0i$
20	90.1	15.6	$2.2 \pm 85.8i$
30	85.2	15.5	$2.7 \pm 80.5i$
H_2:O_2:N_2 / 1.5:1:10			
10	95.2	52.8	$-14.3 \pm 54.7i$
20	83.1	44.5	$-5.4 \pm 53.5i$
30	78.2	40.6	$2.4 \pm 52.7i$

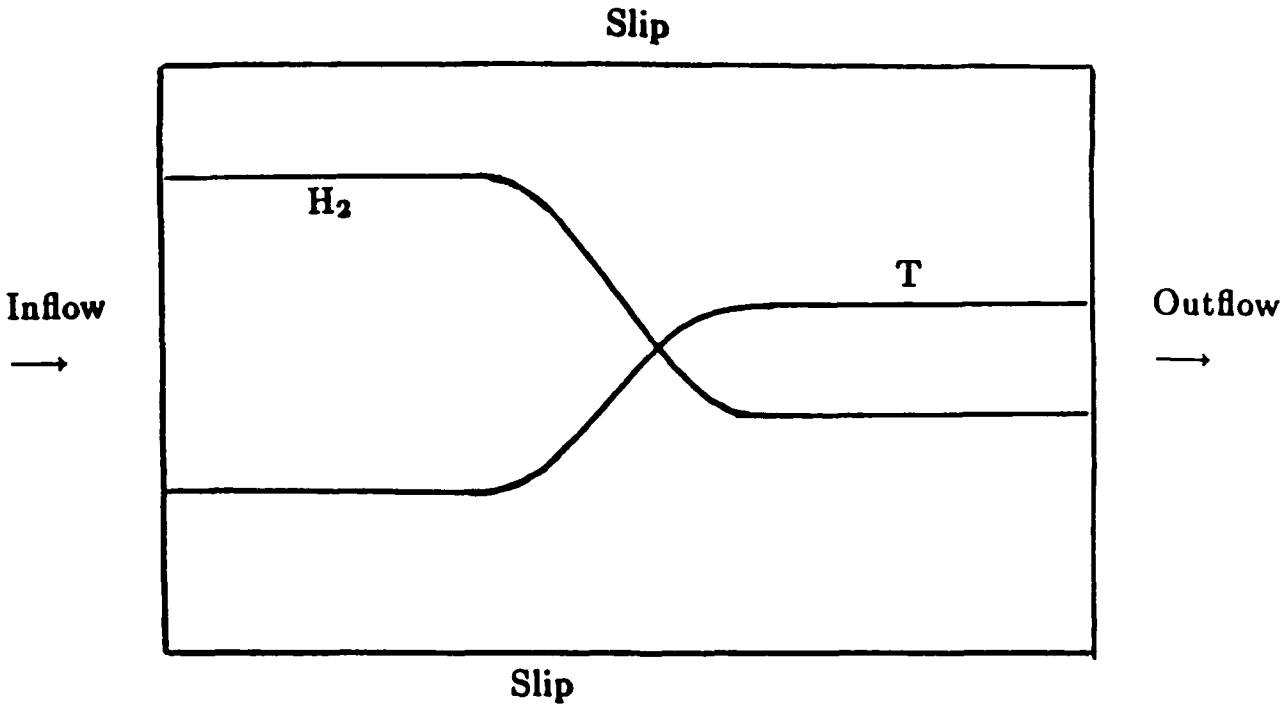


Figure 1. Initial and boundary conditions for the two-dimensional flame calculations.

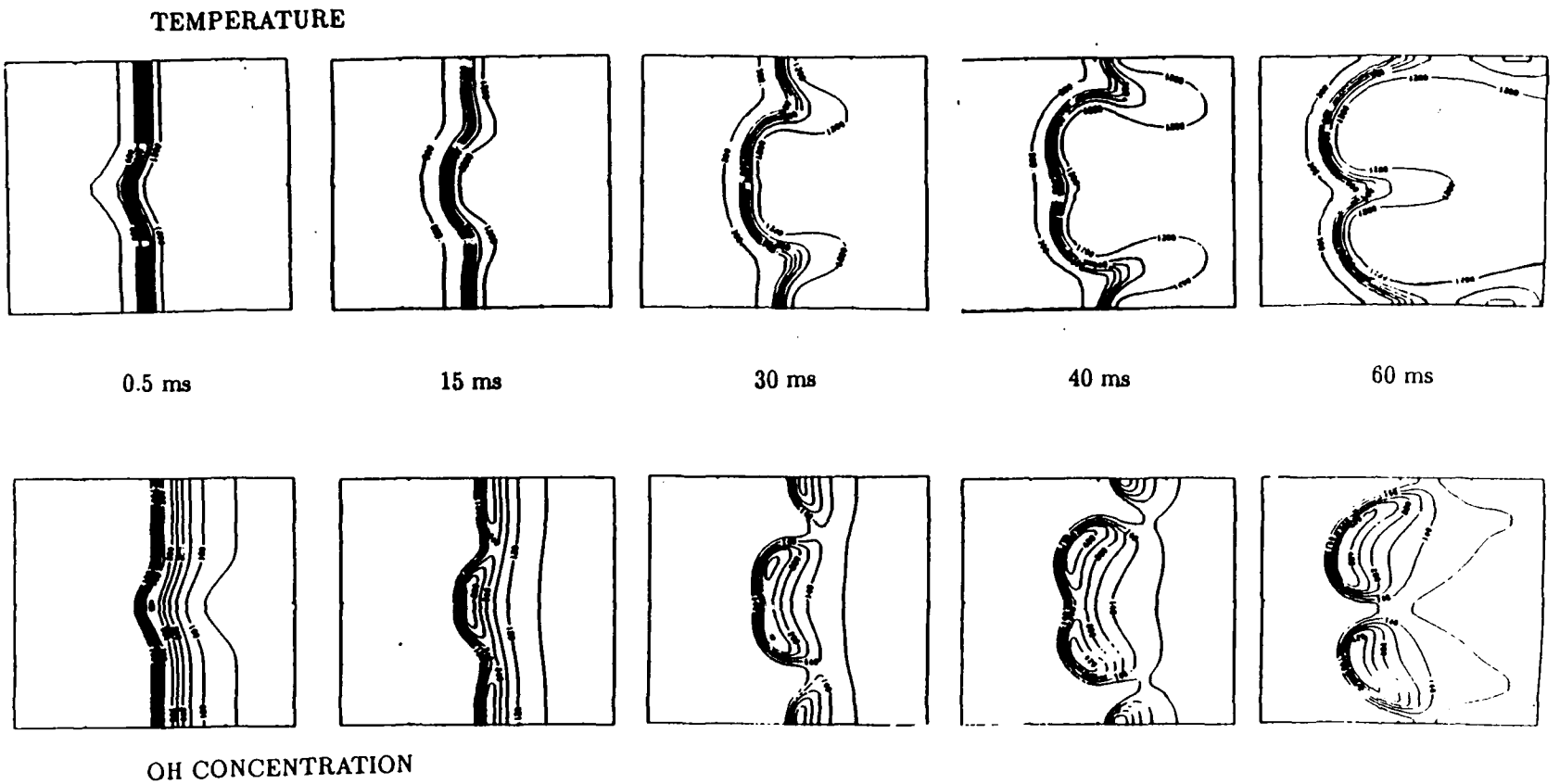


Figure 2. Time evolution of temperature and OH concentration contours showing the transition to cellular structure in zero gravity for a H_2:O_2:N_2 / 1.5:1:10 mixture.

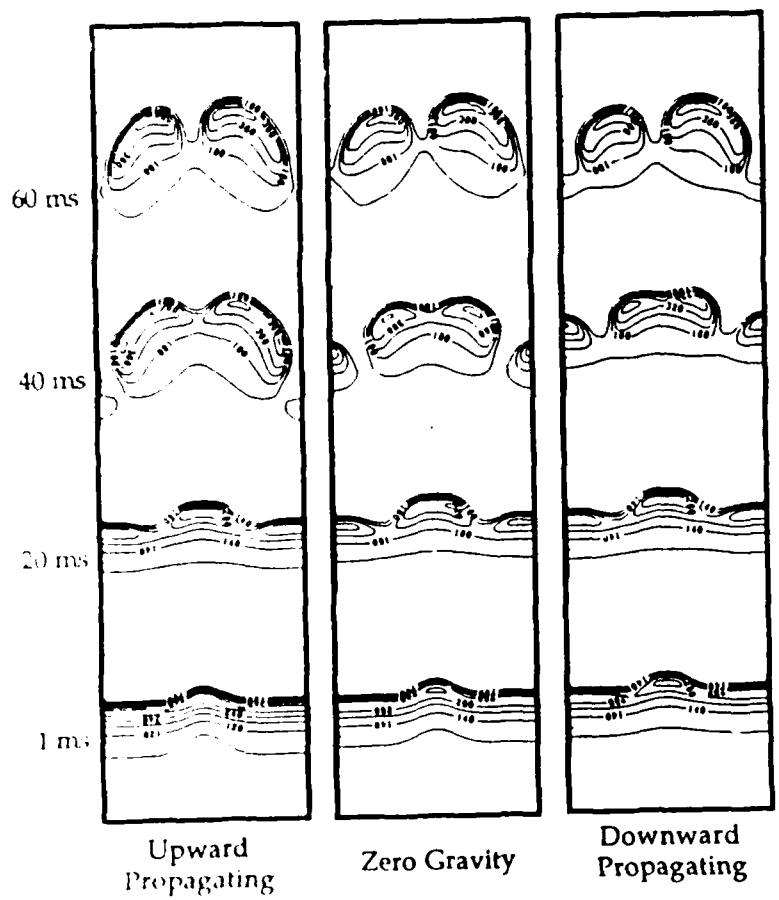


Figure 3. Concentration contours of OH showing the effects of gravity on flame structure in a H_2:O_2:N_2 / 1.5:1:10 mixture.

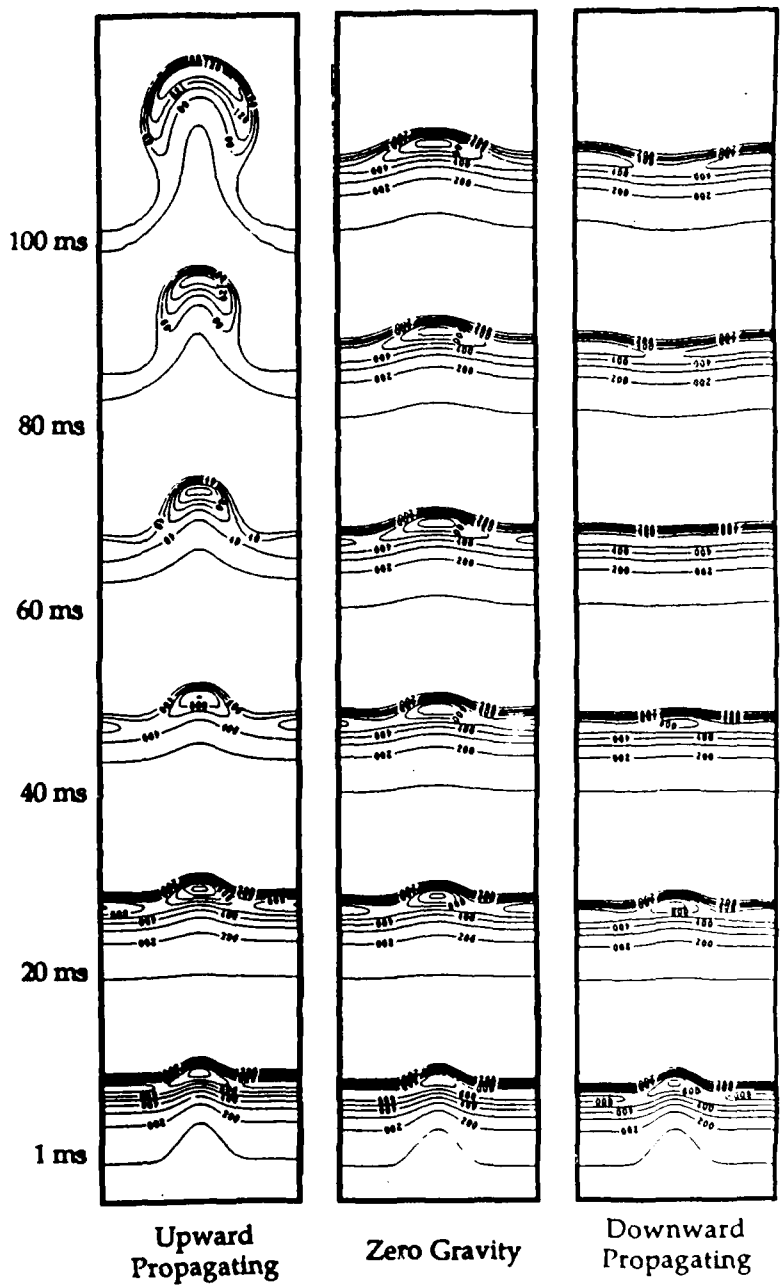


Figure 4. Concentration contours of OH showing the effects of gravity on flame structure in a H_2:O_2:N_2 / 1:1:10 mixture.

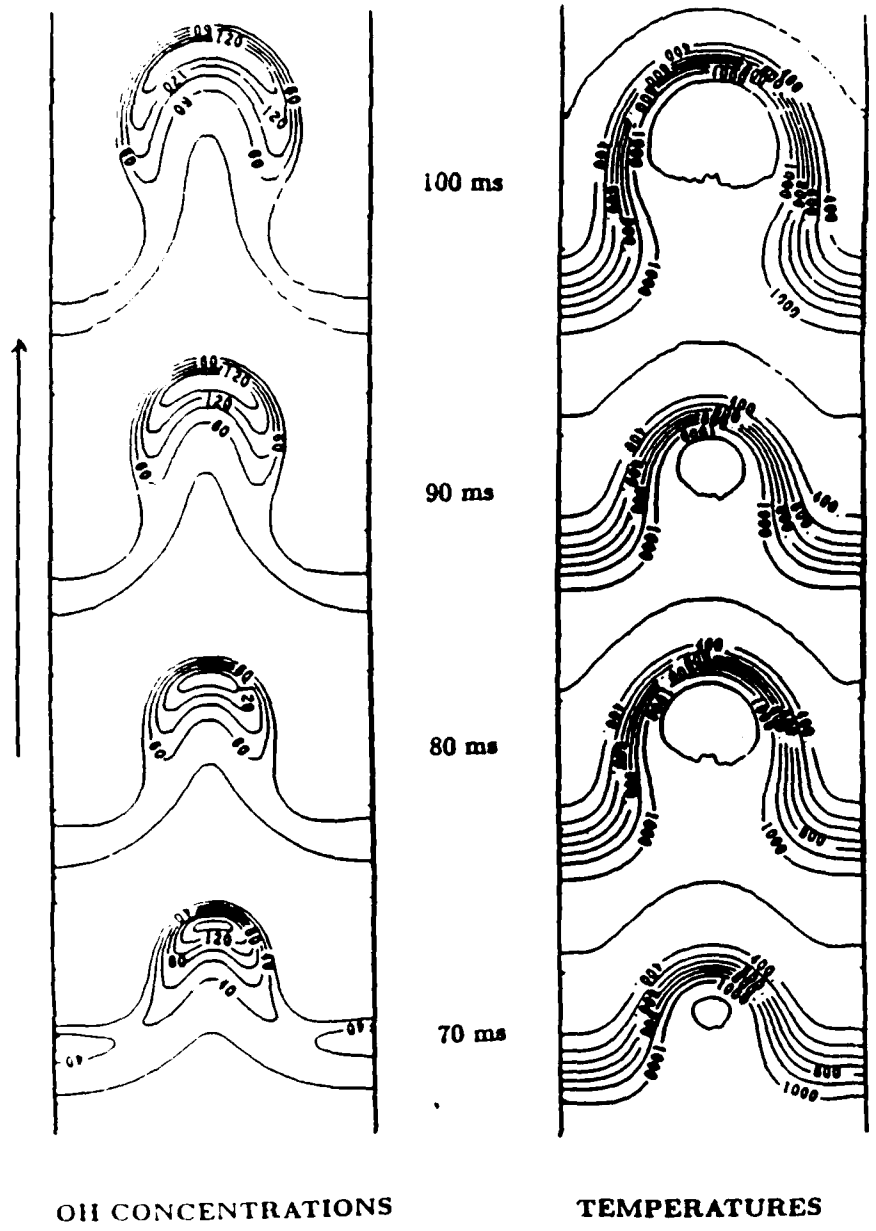


Figure 5. Time evolution of the detailed structure of an upward-propagating flame in a H_2:O_2:N_2/1:1:10 mixture.

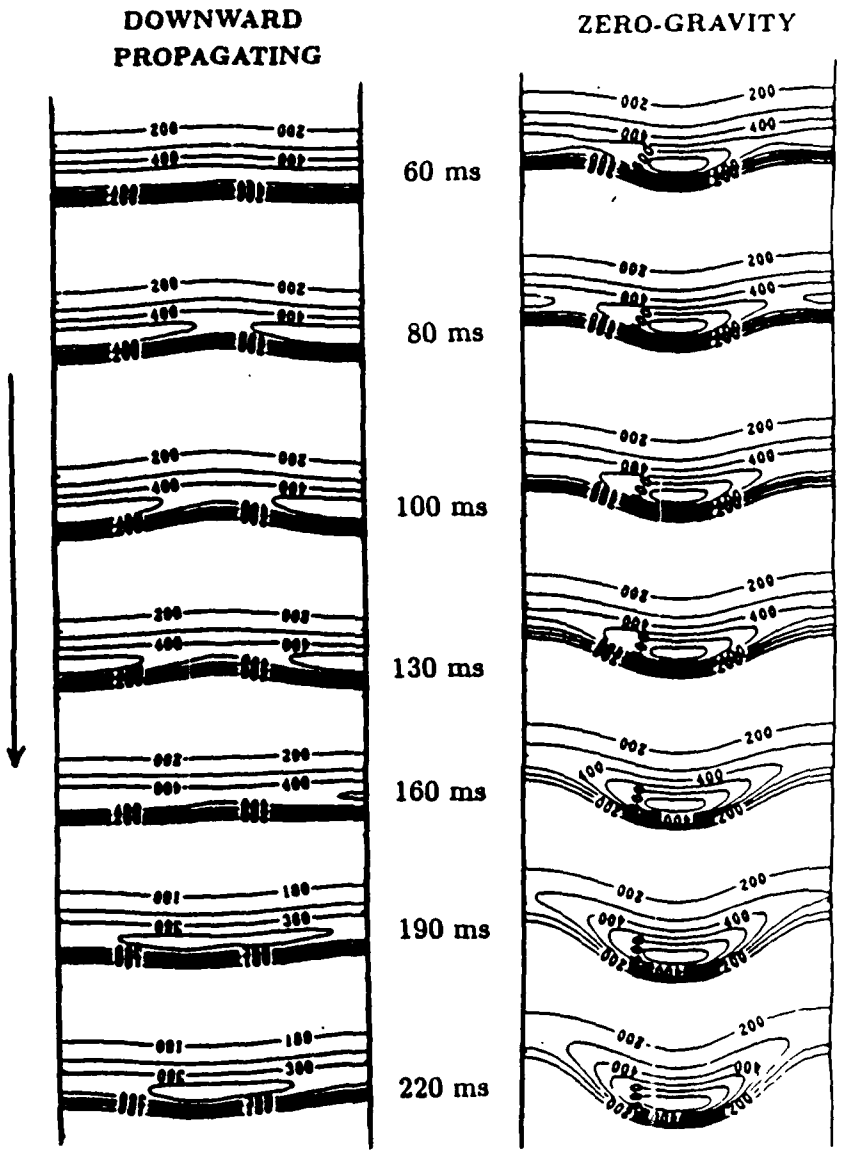


Figure 6. A detailed comparison of the late time evolution of the structure of a downward-propagating and zero-gravity flame in a H_2:O_2:N_2/1:1:10 mixture.

APPENDIX N.

Interaction of a Shock with a Compressible Vortex (Proc. of 17th Intern. Symp. on Shock Waves and Shock Tubes, Lehigh Univ., July, 1989)

To appear, Proceedings of the 17th International
Symposium on Shock Waves and Shock Tubes,
Lehigh University, July, 1989

INTERACTION OF A SHOCK WITH A COMPRESSIBLE VORTEX

Janet L. Ellzey*, Elaine S. Oran, and J. Michael Picone
Laboratory for Computational Physics and Fluid Dynamics
Naval Research Laboratory, Washington, D.C. 20375

ABSTRACT

A fundamental element of compressible turbulence is the interaction of a shock with a vortex existing in the background fluid through which the shock propagates. This paper presents time-dependent two-dimensional numerical simulations of a shock propagating through a compressible vortex. Two cases are discussed: a strong shock, in the sense that the fluid velocity behind the shock front is approximately the same as the maximum velocity in the vortex; and a weak shock, in the sense that the fluid velocity behind the shock front is very much less than the maximum velocity in the vortex. In general, the vortex breaks the initial planar shock into four curved shocks which appear to merge in time. Meanwhile, the vortex itself is compressed and distorted by the shock, and the degree to which this happens depends on the relative strength of the shock and vortex.

INTRODUCTION

A central issue in compressible turbulence is the interaction among shocks and rotational structures in the flow. Shocks and vortices coexist and interact in supersonic propulsion systems, aerodynamic systems, and many laser driven and explosive systems. In some cases, the effects of these interactions on mixing are of crucial importance. In others, the distortions in the pressure and flow are important. Sometimes the interactions are short term, such as those when a shock passes through a vortex, and sometimes there is a continuous interaction between the shock structure and the generation of vortices in the flow.

When a shock interacts with a vortex, both the shock and the vortex may be distorted. In this paper, we investigate the changes in the shock structure as it passes through and interacts with a compressible vortex. In particular, we consider the interaction of a shock with a single, compressible vortex. Isolating this simple interaction allows us to investigate a single element of the more complex processes that occur in high-speed compressible flows. This is an extension of previous work[1] that focused primarily on the effects that the shock produced on the vortex, essentially ignoring the later evolution of the shock. The current study is performed from the macroscopic point of view, using the equations of compressible fluid dynamics. In this sense, we are confining the study to those effects that can be resolved macroscopically with a monotone, numerical algorithm with minimal numerical diffusion.

BACKGROUND

Previous analytical and experimental research on the interaction of a shock with large-scale structures have shown that both the the shock and the structures

* Berkeley Research Associates, Springfield, VA

themselves are distorted in various ways. For example, when a shock passes over a an inhomogeneity in the flow such as a low-density bubble or a high-density drop, the vorticity generated at the boundary of the inhomogeneity becomes a vortex filament pair[2,3]. This is consistent with experimental observations of a shock wave interacting with a flame[4]. In a highly compressible turbulent flow, individual vortices may be flattened by the shocks, vortices may be formed behind the shocks, or vortices may be formed in strong colliding shocks[5].

Experiments by Weeks and Dosanjh[6] have shown that when a shock propogates through an opposing jet, the shock is distorted into a shape that is very similar to that of the mean fluid velocity profile of the jet. The turbulence has only a secondary effect on the shock front. In supersonic mixing layers, large-scale vortical structures coexist with weak and strong shocks, and the interactions are strongly coupled[7,8] to the point where it is difficult to decouple the effects in the interaction.

In addition to the observed changes in the shock front when it becomes curved, the vortex changes due to its interaction with the shock. In the experiment by Dosanjh and Weeks[9], a shock was passed over an inclined airfoil and the flow behind the shock generated a spiral vortex. When the shock reflected and passed over the vortex, the vortex was compressed into an ellipse with the major axis equal to the original diameter of the vortex. In addition, the density at the center of the vortex decreased abruptly. Ting[10] constructed an analytical solution to the governing equations for the transmission of a weak point vortex through a shock wave. His results give the disturbance pressure behind the shock induced by a point vortex as a function of location, time, the vortex circulation, and the shock strength.

NUMERICAL METHODS

To study the shock-vortex interaction, we solve the time-dependent equations for conservation of mass, momentum, and energy in two dimensions,

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{v}) = 0 , \tag{1}$$

$$\frac{\partial \rho \mathbf{v}}{\partial t} + \nabla \cdot (\rho \mathbf{v}\mathbf{v}) = -\nabla P , \tag{2}$$

$$\frac{\partial E}{\partial t} + \nabla \cdot (E\mathbf{v}) = -\nabla \cdot P\mathbf{v} , \tag{3}$$

where ρ is the mass density, $\mathbf{v}$ is the fluid velocity, E is the energy per volume, and P is the pressure. The equation of state is

$$P = \rho R T , \tag{5}$$

and the relation between internal energy per unit volume, e, and pressure, P, is given by

$$e = \frac{P}{\gamma - 1} , \tag{6}$$

where R is the gas constant, T is the temperature, and γ is the ratio of specific heats.

This set of equations is rewritten in terms of finite-difference approximations on an Eulerian grid. The mass density, momentum, and total energy are convected using the latest version of the monotone Flux-Corrected Transport (FCT) algorithm[11], LCPFCT[12,13]. This is an explicit, finite-difference algorithm with fourth-order phase accuracy, and it has been used extensively for computations of time-dependent flows in which it is crucial to maintain accuracy at steep gradients. In particular, the algorithm has been used for simulating the behavior of a number of flows similar to those described in this paper: shocks and shock-shock interactions, vortex behavior in subsonic, compressible flows, and for complex shock-vortex interactions in supersonic flows.

One property of the algorithm which is particularly useful in the calculations presented below is the high-frequency filter inherent in all monotone algorithms. This has the effect of damping all frequencies that describe features that cannot be resolved on the computational grid. The filter acts as an effective viscosity, dissipating the highest frequencies. But because the filter is nonlinear, there is thus minimal effect on wavelengths greater than five computational cells, and no effect on wavelengths greater than ten cells. The property greatly increases the accuracy of the computation for the objects that are resolved. Shocks, however, are not truly resolved. Their width is determined by the numerical viscosity of the computation, and because they are physically so thin compared to the computational grid, in a complex shock-interaction computation, they are at best resolved with a few computational cells.

The computational domain, shown in Figure 1, is a 20 cm by 10 cm region resolved by 480 × 240 cells. The grid is uniform and stays fixed throughout the course of the computation. The typical timesteps used in the calculation are on the order of 3×10^{-7} s.

At the beginning of the computation, a shock approaches a vortex from the left. Thus the inflow conditions on the left of the computational domain are determined from normal shock relations for an ideal, planar shock. Two different shock strengths are investigated in this paper: the strong shock has a Mach number of 1.5, a pressure ratio between the shocked and background gas of 2.45, and density ratio of 1.86; the weak shock has a Mach number of 1.05, a pressure ratio of 1.1196, and a density ratio of 1.0839. In both cases, the ambient, background pressure and density are $P_o = 1.01 \times 10^6$ dynes/cm^2 and $\rho_o = 1.14 \times 10^{-3}$ g/cm^3, respectively.

There is a single vortex rotating counterclockwise centered at $x = 7.75$ cm, $y = 5$ cm in front of the shock. The velocity field of the vortex consists of two regions: an inner region where the velocity is described by solid-body rotation of the form

$$v(r) = \frac{v_{max}}{r_1} r , \tag{7}$$

and an outer region where the velocity decays to zero according to

$$v(r) = Ar + \frac{B}{r} . \tag{8}$$

where v_{max} is the maximum velocity, occuring at $r = r_1$, and approximately equals 230 m/s for the all cases presented in this paper. The constants A and B are chosen such that the velocity matches that determined by Eq. (7) at $r = r_1$ and

decays to zero at $r = r_2$. The inner radius, r_1, is 0.75 cm and the outer radius, r_2, is 1.75 cm. Tests in which the inner and outer radii were varied did not show qualitatively different behavior in the structure of the shock and vortex from the results presented in this paper. The top, bottom, and right boundaries are reflecting walls, representing a perfectly smooth, reflecting chamber with no losses to the walls.

RESULTS OF THE COMPUTATIONS

The computations performed with different shock strengths represent two classes of shock-vortex interactions: weak-shock behavior and strong-shock behavior. In the results described below, the velocity behind the weak shock is much less than the maximum velocity in the vortex, and the velocity behind a strong shock is about the same as the maximum velocity in the vortex.

Figure 2 shows a time sequence of contours of the pressure difference, $P - P_o$ for a computation of the weak shock with Mach number 1.05. Contours cover a time span from the beginning of the computation, step 0, time 0.0 s, to a time well after the shock has passed through the vortex, step 800 at 0.34 ms. The maximum velocity of the vortex is approximately 230 m/s and the fluid velocity behind the shock is 27 m/s.

At step 200, the shock is diffracted around the vortex. Emerging from the top of the vortex is a shock that is curved at the top of the vortex, but straightens out quickly in the ambient gas. Emerging from the bottom of the vortex is a shock that is also curved, but which also straightens out as it propagates through the background gas. This is consistent with the fact the background velocity in the top half of the vortex is in the opposite direction from the direction the shock is propagating, it is in the direction of propagation in the bottom half of the vortex. Therefore, the background fluid impedes the shock in the top half of the vortex and advances the shock in the bottom half. At step 250, the two curved, diffracted fronts meet at a point at the top of the vortex and a pressure peak forms. The diffracted shocks are then reflected and propagate outward.

Meanwhile, the shock itself continues to propagate at an overall velocity approximately equal to its original velocity. The shock is still not a uniform planar shock at the end of the computation, but is still slightly curved. At step 500, the upper trailing wave reflects from the top boundary and at approximately step 700, the lower trailing wave reflects from the lower boundary. The vortex is circular except for the brief time that the trailing shocks are interacting at the vortex edge. At this time, the vortices are deformed slightly but they recover their shape by time step 450. The vortex itself is not significantly affected by having felt the shock, whereas the shock remains affected by the vortex for sometime.

Figure 3 shows a time sequence of pressure-difference contours for a strong shock, Mach number 1.5, propagating over a vortex. The major difference now is that the vortex is significantly affected by its interaction with the strong shock because the fluid velocity behind the shock is on the order of that in the vortex, approximately 230 m/s. Immediately after the shock has passed over the vortex pair, the vortex is still essentially circular, and the pressure contours have the same general shape they did before the shock but the interaction with the trailing diffraction shock steepens the pressure gradient on one side of the vortex. In a steady-state axisymmetric vortex, the pressure gradient, dp/dr, balances the

centripetal force, $\rho v_\theta^2/r$. When the local pressure field of the vortex is distorted, this balance is perturbed and the velocity, density, and pressure field readjust to a new steady state.

In this strong-shock case, the maximum velocity increases from an initial value of approximately 230 m/s to 280 m/s at time step 250 when the shock has just passed over the vortex pair. This acceleration of the vortex occurs because its radius is compressed by the shock and since angular momentum is conserved, the velocity increases. It is not, however, stable in this new configuration. Examining the density shows that the density gradient from the center to the outer edge of the vortex is steepened as the shock passes.

There are, however, fewer long-term effects of the vortex on the shock. The initial structure that the shock has as it leaves the vortex quickly decays, leaving an elliptical vortex behind. The shock structure is similar in Figures 2 and 3 between steps 300 and 500, but because the shock is stronger, it more quickly approaches its initial planar configuration and the strength of the reflected shocks dies.

DISCUSSION AND CONCLUSIONS

The interaction of a shock with a vortex existing in the background fluid through which the shock propagates is a fundamental element of compressible turbulence. In this paper, we have shown the results of time-dependent two-dimensional numerical simulations of a shock propagating through a compressible vortex for two limiting cases: a strong shock, in the sense that the fluid velocity behind the shock front is approximately the same as the the maximum velocity in the vortex; and a weak shock, in the sense that the fluid velocity behind the shock front is very much less than the maximum velocity in the vortex.

In general, the vortex breaks the shock up into four shocks which appear to merge in time and return to the original planar form. Meanwhile, the vortex itself is compressed and distorted by the shock, and the degree to which this happens depends on the relative strength of the shock and vortex. When the shock is strong, it recovers quickly from the interaction with the vortex. The vortex itself is highly compressed and distorted into an elliptical shape, and its rotation rate consequently increases. When the shock is weak, the shock takes longer to recover from the interaction. The vortex is initially distorted by the shock, but quickly regains its initial spherical shape and there is little change in its rotation rate.

The change in shape and velocity of the shock can be understood based on local interaction of the vortical fluid and the shock. When the vortical fluid is moving in the direction of the shock propagation, the shock velocity increases; when the vortical fluid is moving in the opposite direction from that of the shock propagation, the shock slows down. This almost obvious result is seen both in our simulations and in experiments[9].

ACKNOWLEDGMENTS

This work was sponsored by the Naval Research Laboratory through the Office of Naval Research and by the Defense Applied Research Projects Agency in the Applied and Computational Mathematics Program. The authors would like to acknowledge extremely helpful discussions with John H. Gardner in constructing the LCPFCT module.

REFERENCES

1. J.L. Ellzey and E.S. Oran, Simulation of Shock and Vortex Interactions, to appear, *Proceedings of the International Workshop on Compressible Turbulent Mixing*, Springer, 1989.
2. J.M. Picone, *J. Fluid Mech.* 189, 23–51, 1988.
fem 3. J.F. Haas, and B. Sturtevant, *J. Fluid Mech.* 181, 41–76, 1987.
4. G.H. Markstein, Experimental Studies of Flame-Front Instability, *Nonsteady Flame Propagation*, AGARD Report No. 75, Permagon Press, Oxford, 75–100, 1964.
5. T. Passot, and A. Pouquet, *J. Fluid Mech.* 181,441–466, 1987.
6. T.M. Weeks and D.S. Dosanjh, *AIAA J.*, 1, 1527–1533, 1963.
7. R. Guirguis, F.F. Grinstein, K. Kailasanath, E.S. Oran, J.P. Boris, and T.R. Young, *Mixing Enhancement in Supersonic Shear Layers*, AIAA 25th Aerospace Sciences Meeting, Paper No. AIAA-87-0373, AIAA, NY, 1987.
8. J.P. Boris, E.S. Oran, J.H. Gardner, K. Kailasanath, and T.R. Young, *Computational Studies of a Localized Supersonic Shear Layer*, AIAA Paper No. 89-0125, AIAA, Washington, DC, 1989.
9. D.S. Dosanjh, and T.M. Weeks, *AIAA J.*, 3, 216–223, 1965.
10. L. Ting, *Phys. Fluids*, 17, 1518–1526, 1974.
11. J.P. Boris, and D.L. Book, *Methods in Computational Physics*, 16, 85-129, 1976.
12. Oran, E.S., and Boris, J.P., *Numerical Simulation of Reactive Flow*, Elsevier, New York, 1987.
13. J.P. Boris, J.H. Gardner, E.S. Oran, S. Zalesak, J. Ellzey, G. Patnaik, D.L. Book, R.H. Guirguis, *LCPFCT – A Monotone Algorithm for Solving Continuity Equations*, to appear, Naval Research Laboratory Memorandum Report, 1989.

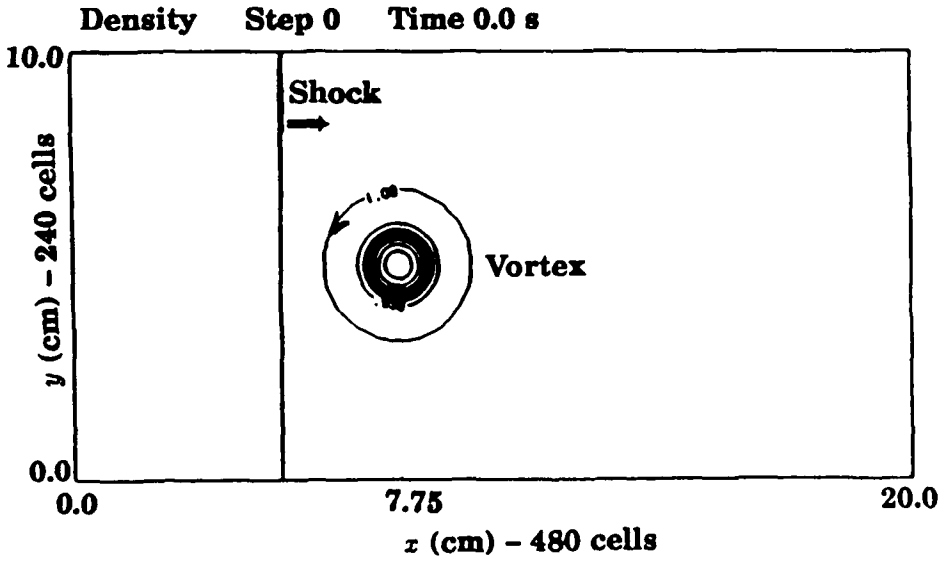


Figure 1. Computational domain and initial conditions of simulations of shock-vortex interaction.

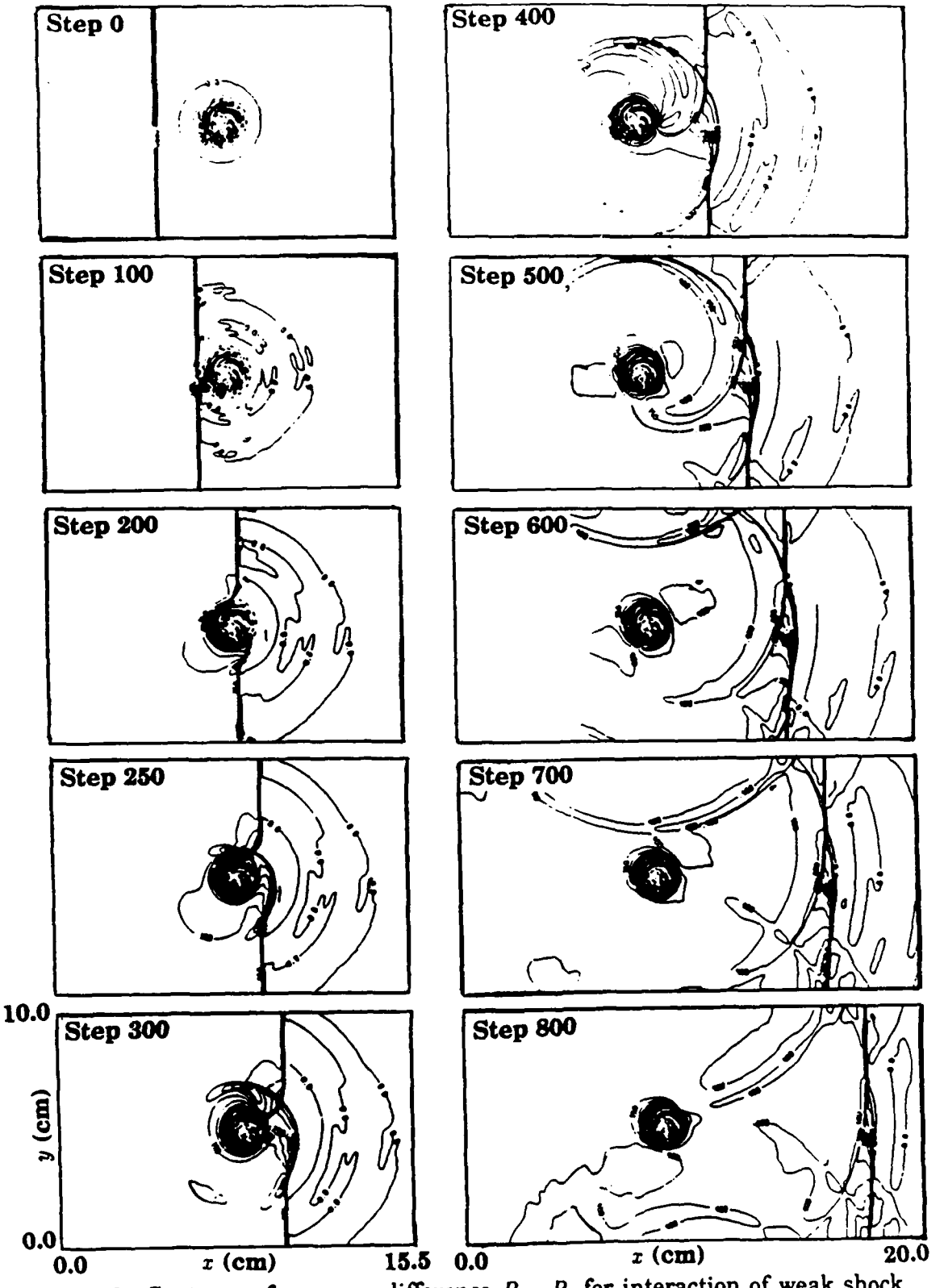


Figure 2. Contours of pressure difference $P - P_o$ for interaction of weak shock and vortex.

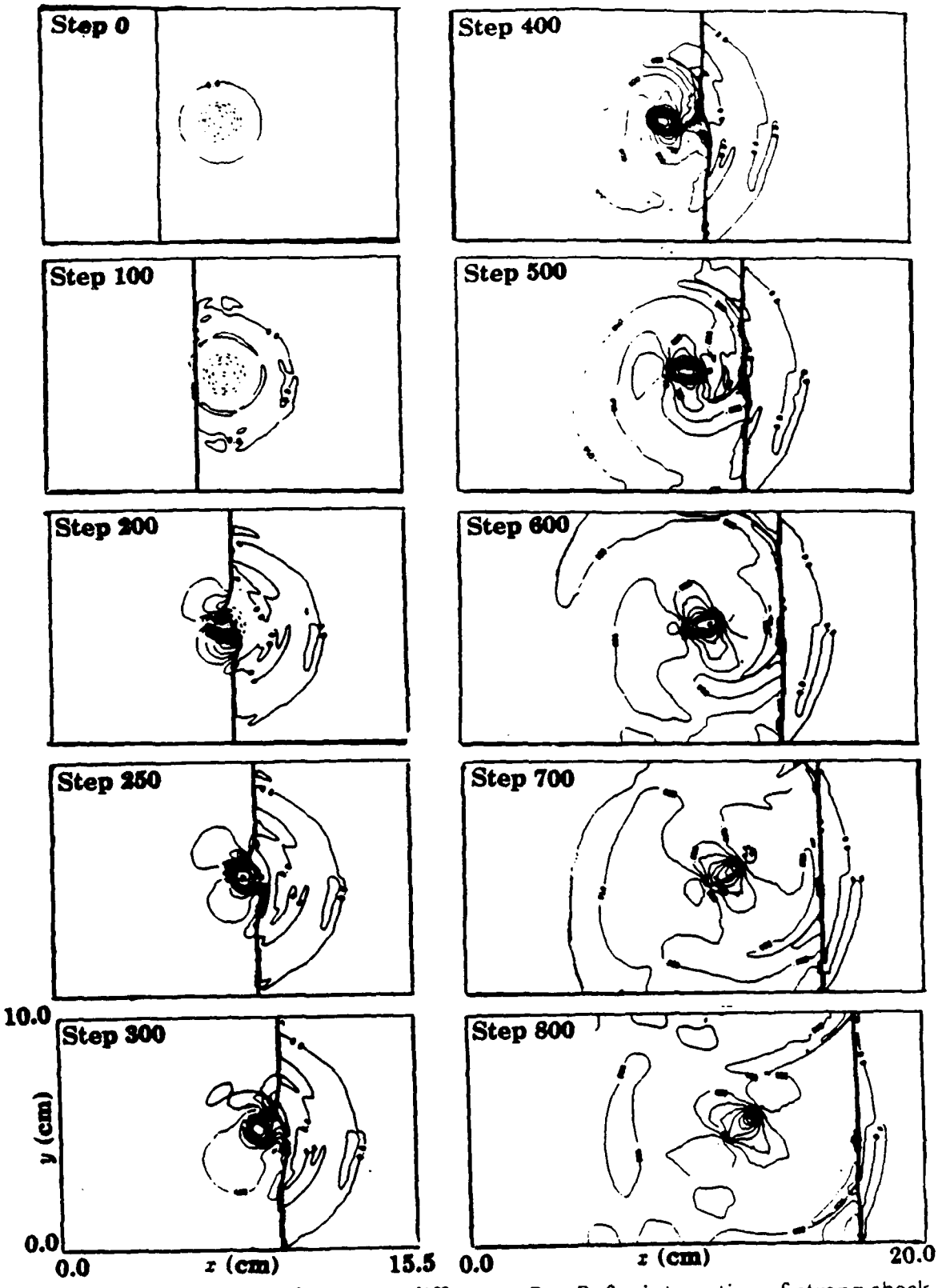


Figure 3. Contours of pressure difference $P - P_o$ for interaction of strong shock and vortex.

Reaction Rate	A[a]	B	C[a]	Source
$H + HO \rightleftharpoons O + H_2$	1.40(−14)	1.00	3.50(+03)	[23]
	3.00(−14)	1.00	4.48(+03)	[23]
$H + HO_2 \rightleftharpoons H_2 + O_2$	4.20(−11)	0.00	3.50(+02)	[23]
	9.10(−11)	0.00	2.91(+04)	[23]
$H + HO_2 \rightleftharpoons HO + HO$	4.20(−10)	0.00	9.50(+02)	[23]
	2.00(−11)	0.00	2.02(+04)	[23]
$H + HO_2 \rightleftharpoons O + H_2O$	8.30(−11)	0.00	5.00(+02)	[24]
	1.75(−12)	0.45	2.84(+04)	$k_r = k_f/K_c$
$H + H_2O_2 \rightleftharpoons HO_2 + H_2$	2.80(−12)	0.00	1.90(+03)	[23]
	1.20(−12)	0.00	9.40(+03)	[23]
$H + H_2O_2 \rightleftharpoons HO + H_2O$	5.28(−10)	0.00	4.50(+03)	[23]
	3.99(−10)	0.00	4.05(+04)	$k_r = k_f/K_c$
$HO + H_2 \rightleftharpoons H + H_2O$	1.83(−15)	1.30	1.84(+03)	[25]
	1.79(−14)	1.20	9.61(+03)	[25]
$HO + HO \rightleftharpoons H_2 + O_2$	1.09(−13)	0.26	1.47(+04)	$k_f = k_r K_c$
	2.82(−11)	0.00	2.42(+04)	[26]
$HO + HO \rightleftharpoons O + H_2O$	1.00(−16)	1.30	0.00(+00)	[25]
	3.20(−15)	1.16	8.77(+03)	$k_r = k_f/K_c$
$HO + HO_2 \rightleftharpoons H_2O + O_2$	8.30(−11)	0.00	5.03(+02)	[27]
	2.38(−10)	0.17	3.69(+04)	$k_r = k_f/K_c$
$HO + H_2O_2 \rightleftharpoons HO_2 + H_2$	1.70(−11)	0.00	9.10(+02)	[23]
	4.70(−11)	0.00	1.65(+04)	[23]
$HO_2 + H_2 \rightleftharpoons HO + H_2O$	1.20(−12)	0.00	9.41(+03)	[26]
	1.33(−14)	0.43	3.62(+04)	$k_r = k_f/K_c$

Table 3.1 Chemical Reaction Rates for H_2-O_2 Combustion: $k_i = AT^B \exp(-C/T)$[b]

[a] Exponentials to the base 10 are given in parentheses: $1.00(-10) = 1.00 \times 10^{-10}$.

[b] Bimolecular reaction rate constants are in units of cm^3/ (molecule s).

Reaction Rate	A[a]	B	C[a]	Source
$HO_2 + HO_2 \rightleftharpoons H_2O_2 + O_2$	3.00(−11)	0.00	5.00(+02)	[24]
	1.57(−09)	−0.38	2.20(+04)	$k_r = k_f/K_c$
$O + HO \rightleftharpoons H + O_2$	2.72(−12)	0.28	−8.10(+01)	$k_f = k_r K_c$
	3.70(−10)	0.00	8.45(+03)	[23]
$O + HO_2 \rightleftharpoons HO + O_2$	8.32(−11)	0.00	5.03(+02)	[27]
	2.20(−11)	0.18	2.82(+04)	$k_r = k_f/K_c$
$O + H_2O_2 \rightleftharpoons H_2O + O_2$	1.40(−12)	0.00	2.12(+03)	[24]
	5.70(−14)	0.52	4.48(+04)	$k_r = k_f/K_c$
$O + H_2O_2 \rightleftharpoons HO + HO_2$	1.40(−12)	0.00	2.13(+03)	[24]
	2.07(−15)	0.64	8.23(+03)	$k_r = k_f/K_c$
$H + H + M \rightleftharpoons H_2 + M$	1.80(−30)	−1.00	0.00(+00)	[23]
	3.70(−10)	0.00	4.83(−04)	[23]
$H + HO + M \rightleftharpoons H_2O + M$	6.20(−26)	−2.00	0.00(+00)	[23]
	5.80(−09)	0.00	5.29(+04)	[23]
$H + O_2 + M \rightleftharpoons HO_2 + M$	4.14(−33)	0.00	−5.00(+02)	[23]
	3.50(−09)	0.00	2.30(+04)	[23]
$HO + HO + M \rightleftharpoons H_2O_2 + M$	2.50(−33)	0.00	−2.55(+03)	[23]
	2.00(−07)	0.00	2.29(+04)	[23]
$O + H + M \rightleftharpoons HO + M$	8.28(−29)	−1.00	0.00(+00)	[28]
	2.33(−10)	0.21	5.10(+04)	$k_r = k_f/K_c$
$O + HO + M \rightleftharpoons HO_2 + M$	2.80(−31)	0.00	0.00(+00)	[28]
	1.10(−04)	−0.43	3.22(+04)	$k_r = k_f/K_c$
$O + O + M \rightleftharpoons O_2 + M$	5.20(−35)	0.00	−9.00(+02)	[23]
	3.00(−06)	−1.00	5.94(+04)	[23]

Table 3.1 Continued Chemical Reaction Rates for H_2–O_2 Combustion: $k_i = AT^B \exp(-C/T)$[b]

[a] Exponentials to the base 10 are given in parentheses: $1.00(-10) = 1.00 \times 10^{-10}$.

[b] Bimolecular reaction rate constants are in units of cm^3/ (molecule s).

$$n_i^n = n_i^o + \frac{2\delta t\,[Q_i' - L_i^o n_i^o + F_i^o]}{4 + \delta t\,[L_i' + L_i^o]}\,, \qquad \text{(Stiff)} \qquad (3.7)$$

$$n_i^n = \frac{2Q_i'}{L_i' + L_i^o}\,. \qquad \text{(Equilibrium)} \qquad (3.8)$$

The original CHEMEQ report [9] describes the details of the timestep selection algorithm, stiffness criterion, and other important details. A detailed report on TBA [29] is currently under preparation.

3.2 Data-Handling Algorithm

TBA is designed to handle a large number of cells, each with an independent timestep. Thus, each cell is integrated with a different number of subcycles. Cells have to be constantly shuffled in and out of the integration routine. Whenever the integration of a cell is complete, it is put onto a list. At the end of the integration loop, this list is passed to another routine, CDATA, which stores the results and inserts new cells to be integrated into the places left by the completed cells. Towards the end of the integration procedure there are no additional cells to integrate and the arrays in the integrator will be only partially filled. When this occurs, we sort and move all completed cells to the ends of the arrays where they will not be integrated further. When the integration of the remaining cells is complete, all the data is written out in one operation. Thus we avoid both unnecessary shuffling and unnecessary calls to CDATA. The sort algorithm developed specifically for this application is $O(N)$ and makes the minimum number of swaps. The sort is optimized by the use of CRAY assembly routines that make bitwise comparisons.

In the original VSAIM routine, all equations passed through one integration loop where they were tested for stiffness by `if` statements and either the stiff or normal calculations were performed. The CRAY X-MP vectorizes `if` statements by performing the calculations for both possibilities and then throwing away the results for the false condition, so this approach is wasteful. TBA creates index arrays for different types of equations and sets up a separate integration loop for each of the three types. This approach succeeds due to the speed of the CRAY X-MP gather/scatter hardware.

3.3 Programming Strategies

Many strategies or tricks were used to enhance the speed of TBA, most of which are in what is considered extremely poor programming style, but reflect certain idiosyncracies of CRAY programming in general. Some are documented in CRAY manuals, especially the *Optimization Guide* [30].

For example, many of the arrays are equivalenced as both one- and two-dimensional arrays. The CRAY will only vectorize the innermost nested loop, so wherever possible we looped over the one-dimensional equivalent array, to avoid an outer loop. Working space for TBA is supplied in a common block, as this proved substantially faster than passing it through the argument list in the calling sequence. Memory access is the bottleneck on the CRAY so scalar temporaries are used to reduce memory traffic. A full 50% speed up was achieved through their use. A power of two as the number of species results in memory conflicts that slow the program considerably. There are eight (2^3) chemical species in ***FLIC*** which would result in very poor performance. Thus a "dummy" species was put in, removing the memory conflict.

The following code fragment illustrates the importance of large vector lengths. Note that if the order of the loops were switched, the `if` could be taken out of the inner loop and the memory access made contiguous. However, a much shorter inner loop results, and the execution time is actually greater.

```
      do 250 i=1,numeqns
         do 240 j=1,numcells
            if (convchk(j)) then concent(i,j)=corr2d(i,j)
240      continue
250   continue
```

These sorts of tricks are higly specific to the CRAY X-MP computer and must be used to gain full advantage of its speed. Though TBA can be transported to other machines (it was developed on a VAX), it was written specifically for the CRAY X-MP with its gather/scatter hardware in mind.

4. Diffusive Transport Processes

The diffusive transport processes currently included in ***FLIC*** are molecular diffusion including the Dufour effect, thermal conduction, and viscosity, all processes that are crucial for describing flame structure. As yet, we have have not included the thermal effects on mass diffusion (thermal diffusion or the Soret effect), thermal dissipation due to viscosity, or radiation transport. Although it is a second-order effect, thermal diffusion may become important for near-limit flames and so should be included in the future. Thermal dissipation is negligible at the extremely low Mach numbers in the flows under consideration. Radiation effects are important in many flames, particularly hydrocarbon flames that form soot, but is not particularly important in hydrogen flames.

The differential equations describing these diffusive processes have been solved with a two-dimensional, explicit Eulerian scheme. Spatial derivatives have been approximated by central differences and a simple forward-Euler time marching scheme has been used for time advancement. This explicit method has a timestep limit roughly equal to the timestep required by BIC-FCT for the fluid convection. However, each diffusive transport process has its own stability condition, and on occasion it can require a timestep up to five times smaller. When this is the case, the approach we have taken is to subcycle the integration for that process. The alternate approach would be to use an implicit algorithm for selected terms, but this is generally more expensive than subcycling when fewer than ten subcycles are required.

4.1 Mass Diffusion

The part of the species conservation equations for species number density that describes mass diffusion are

$$\frac{\partial n_k}{\partial t} = -\nabla \cdot \left(n_k \vec{V}_k \right) \ , \quad n_k \ = \ 1, ... n_{sp} \ , \tag{4.1}$$

subject to the constraint

$$\sum_{k=1}^{n_{sp}} Y_k \vec{V}_k = 0 \ . \tag{4.2}$$

The effects of molecular diffusion in the energy equation (Dufour effect) appear as

$$\frac{\partial E}{\partial t} = -\nabla \cdot \left(\sum_{k=1}^{n_{sp}} n_k h_k \vec{V}_k \right) \ , \tag{4.3}$$

where h_k is the temperature-dependent enthalpy for each of the species. The values of $\{V_k\}$ are found by solving [31]

$$S_k = \sum_{\substack{j=1 \\ j \neq k}}^{n_{sp}} \frac{X_j X_k}{D_{kj}} (V_j - V_k) = \nabla X_k, \tag{4.4}$$

subject to

$$\sum_{k=1}^{n_{sp}} S_k = 0, \tag{4.5}$$

and Eq. (4.2), where X_k and Y_k are the mole and mass fractions of species k. The exact solution of this equation for the set $\{V_k\}$ can be obtained by solving the matrix equation implied by Eq. (4.4).

To avoid solving the full matrix equation at each location at each timestep, we find an approximation to the $\{V_k\}$ using Fick's Law and then correct this by the procedure described by Coffee and Heimerl [32] to ensure that the constraint Eq. (4.2) is met. The diffusion velocity according to Fick's law is

$$\widehat{V_k} = -\frac{1}{X_k} D_{km} \nabla X_k \ , \tag{4.6}$$

where D_{km} is the diffusion coefficient of species k in the mixture of gases. Equation 4.6 determines the diffusion velocities to within a constant. We then assume that a

constant $\vec{V}_c$ is added to all the raw diffusion velocities $\widehat{V}_k$ and require that the sum of the diffusion fluxes equal zero. Thus,

$$\sum_{k=1}^{n_{sp}} Y_k \vec{V}_k = \sum_{k=1}^{n_{sp}} Y_k \left(\widehat{V}_k + \vec{V}_c \right) = 0 \tag{4.7}$$

leads to

$$\vec{V}_c = - \sum_{k=1}^{n_{sp}} Y_k \widehat{V}_k \ . \tag{4.8}$$

The component of the corrected diffusion velocity defined by

$$\vec{V}_k = \widehat{V}_k + \vec{V}_c \tag{4.9}$$

is then used in Eq. (4.1).

The set of $\{V_k\}$ found in this way is algebraically equivalent to the first iteration of the DFLUX algorithm [12], an approach based on a matrix expansion that converges to arbitrary order. Values of $\{V_k\}$ obtained by the procedure described above were compared to the results of DFLUX to check their accuracy. The explicit finite differencing used to solve Eq. (4.1) has the numerical stability limit,

$$\max \left(D_{km} \Delta t / \Delta x^2 \right) < 1/2 \ .$$

Generally the mass-diffusion algorithm is subcycled within an overall timestep determined by the convection algorithm. However, the code is designed to decrease the overall timestep to below that required by the mass-diffusion stability limit if the number of subcycles required exceeds a specified maximum value. Subcycling becomes necessary when the temperature of the reacting flow becomes high and the diffusion coefficients increase accordingly.

Determining the diffusion velocities by Eq. (4.6) requires as input the set of diffusion coefficients of species k into the mixture, $\{D_{km}\}$. However, these quantities are difficult to obtain from first principles and are usually found by applying a mixture rule to the individual binary diffusion coefficients. Binary diffusion coefficients can be estimated theoretically [33] and sometimes measured experimentally [33,34]. Here we use the same approach as in FLAME1D (Kailasanath *et al.* [22]). Binary diffusion coefficients are expressed in the form

$$D_{kl} = \frac{A_{kl}}{n_{tot}} T^{B_{kl}} \tag{4.10}$$

	O	H_2	OH	H_2O	O_2	HO_2	H_2O_2	N_2
H	6.30(17)	8.29(17)	6.30(17)	6.70(17)	6.70(17)	6.70(17)	4.43(17)	6.10(17)
	7.28(-1)	7.28(-1)	7.28(-1)	7.28(-1)	7.32(-1)	7.32(-1)	7.28(-1)	7.32(-1)
O		3.61(17)	1.22(17)	2.73(17)	9.69(16)	9.69(16)	1.57(17)	9.69(16)
		7.32(-1)	7.74(-1)	6.32(-1)	7.74(-1)	7.74(-1)	6.32(-1)	7.74(-1)
H_2			3.49(17)	6.41(17)	3.06(17)	3.06(17)	4.02(17)	2.84(17)
			7.32(-1)	6.32(-1)	7.32(-1)	7.32(-1)	6.32(-1)	7.38(-1)
OH				2.73(17)	1.16(17)	9.69(16)	1.57(17)	9.69(16)
				6.32(-1)	7.24(-1)	7.74(-1)	6.32(-1)	7.74(-1)
H_2O					2.04(17)	2.04(17)	1.57(17)	1.89(17)
					6.32(-1)	6.32(-1)	6.32(-1)	6.32(-1)
O_2						8.74(17)	1.14(17)	8.29(16)
						7.24(-1)	6.32(-1)	7.24(-1)
HO_2							1.14(17)	8.85(16)
							6.32(-1)	7.74(-1)
H_2O_2								1.14(17)
								6.32(-1)

Table 4.1 Binary diffusion coefficients expressed in the form: $D_{jk} = A_{jk}\frac{T^{B_{jk}}}{N}$. For each pair of species, the upper term is A_{jk} and the lower term is B_{jk}, exponentials to the base 10 are given in parentheses.

where the sets $\{A_{kl}\}$ and $\{B_{kl}\}$ are tabulated [22] for each pair of species in the hydrogen–oxygen reaction system, and are given in table 4.1. These are then combined by a mixture rule [35,36]

$$D_{km} = \frac{1 - Y_k}{\sum_{\substack{l=1 \\ l \neq k}}^{n_{sp}} \frac{X_l}{D_{kl}}} , \tag{4.11}$$

which provides values of D_{km} to use in Eq. (4.6).

APPENDIX O.

Dynamics of an Unsteady Diffusion Flame: Effects of Heat Release and Viscosity (accepted by AIAA Progress in Astronautics and Aeronautics)

accepted by AIAA Progress in Astronautics & Aeronautics

DYNAMICS OF AN UNSTEADY DIFFUSION FLAME: EFFECTS OF HEAT RELEASE AND VISCOSITY

J.L. Ellzey* , **K.J. Laskey**** and **E.S. Oran**

Laboratory for Computational Physics and Fluid Dynamics
Naval Research Laboratory
Washington, D.C. 20375

Abstract

Experiments in jet diffusion flames have shown that the flow field consists of two types of instabilities. Low frequency instabilities form in the outer region of the flow field and high frequency structures form at the interface between the high and low velocity fluid. In this paper, we investigate the effects of heat release and viscosity on the development of the high frequency structures. In our computations, the conservation equations for mass, momentum, energy, and species are solved using Flux-Corrected Transport (FCT), which is an explicit, finite-difference technique for solving generalized conservation equations. The system of equations is closed with an ideal gas equation of state. Recently, the FCT algorithm has been extended to lower speed flows by including an implicit correction step which allows a much larger time step. The viscous diffusion, mass diffusion, and conduction terms are solved independently, and a simplified reaction model is used to represent the reaction of fuel and oxider. We simulate an axisymmetric H_2-N_2 fuel jet which is surrounded by coflowing air. The jet velocity is 10 m/s and the air velocity is 15 cm/s. The results indicate that heat release delays the formation of large-scale instabilities. Once formed, the structures are weaker than in the nonreacting case. When viscosity is included in the calculation of the reacting jet the fluctuations in the flow field are reduced substantially and the flame appears almost laminar.

Introduction

Visualizations of jet diffusion flames show that both low frequency and high frequency structures develop in the flow field (1,2). The low frequency (10-15 Hz)

* Berkeley Research Associates, Springfield, VA
** Grumman Space Station Program Support Division, Reston, VA

oscillations appear to be buoyancy-driven and form in the low-speed fluid outside the mixing region due to the temperature gradients associated with heat release. These structures, which have been observed in both experiments (1-3) and computations (4) are only weakly affected by changes in jet velocity or fuel composition.

The high frequency structures are Kelvin-Helmholtz instabilities which form at the interface between the high and low velocity fluid with typical frequencies of a few hundred Hertz. Previous research suggests that these structures are affected by heat release. Mahalingam, et al. (5) used an inviscid, linearized stability analysis to show that heat release in a low speed diffusion flame shifts the most amplified mode to lower frequency and reduces the growth rate of the mixing layer. Experiments on a chemically reacting mixing layer (6) indicate that heat release reduces the vorticity thickness.

In this paper, we present time-dependent simulations of an axisymmetric $H_2 - N_2$ jet with coflowing air. Results for reacting and nonreacting jets with and without viscosity are presented. In these calculations, gravity is not included and the outer buoyancy-driven structures do not develop. Instead, we focus on the effects of heat release and viscosity on the development of the high frequency structures.

Numerical Technique

The computer code developed for this study is based on that developed by Laskey (7) to simulate jet flames and by Patnaik, et al. (8) to simulate low-speed premixed flames. The code solves the following conservation equations for mass, momentum, energy, and species:

$$\frac{\partial \rho}{\partial t} + \nabla \cdot \left(\rho \vec{V}\right) = 0 \tag{1}$$

$$\frac{\partial \rho \vec{V}}{\partial t} + \nabla \cdot \left(\rho \vec{V} \vec{V}\right) = -\nabla P - \nabla \cdot \tau \tag{2}$$

$$\frac{\partial E}{\partial t} + \nabla \cdot \left(E \vec{V}\right) = -\nabla \cdot P\vec{V} + \nabla \cdot (\kappa \nabla T) - \nabla \cdot \sum_{k=1}^{n_{sp}} n_k \vec{V}_k h_k + Q \tag{3}$$

$$\frac{\partial n_k}{\partial t} + \nabla \cdot \left(n_k \vec{V}\right) = -\nabla \cdot \left(n_k \vec{V}_k\right) + w_k \ , \tag{4}$$

where ρ is the density, $\vec{V}$ is the velocity vector, t is time, P is pressure, E is the total energy density, κ is thermal conductivity, T is temperature, n_k and h_k are the number of molecules and enthalpy of species k per unit volume, $\vec{V}_k$ is the diffusional velocity vector of species k, Q is the energy released with reaction, and w_k is number of molecules of species k per unit volume per unit time produced with reaction.

This system of equations is closed by the equation of state

$$P_k = n_k kT \tag{5}$$

and the relationship between internal energy e and pressure is given by

$$de = dP/(\gamma - 1) \tag{6}$$

where γ is the ratio of specific heats.

The equations are rewritten in terms of finite-difference approximations on an Eulerian mesh and then solved numerically for specified boundary conditions. The accuracy of the solution is determined by the finite difference algorithm, the spatial resolution set by the computational grid, and the temporal resolution set by the timestep. In reacting flow problems, there is a wide range of important spatial and temporal scales. Because it is not possible to resolve phenomena on all of these scales, the smallest scales must usually be modeled. The basic assumption in most flow modeling of diffusion flames is that mixing of fuel and oxidizer takes place much more slowly than the chemical reactions, and therefore, a global reaction mechanism may be used. However, because a simulation that includes a detailed set of elementary reaction rates will be possible in the next few years, this program is designed in a modular form such that more detailed chemistry can be substituted later.

The conservation equations contain terms representing convection, conduction, species diffusion, chemical reaction, and viscous effects. In our algorithm, the separate processes are solved independently. The individual algorithms for the various processes and the coupling mechanism are described in the next section. A more detailed description of the method is presented by Laskey (7).

Convection

The solution to equations (1) - (4) is obtained using the high-order implicit method, BIC-FCT. The Flux-Corrected Transport (FCT) itself is an explicit, finite-difference algorithm with fourth-order phase accuracy. Recently, Patnaik et al. (8) developed the Barely Implicit Correction for Flux-Corrected Transport, BIC-FCT, for subsonic flows by including an implicit correction step. This technique is based on the idea proposed by Casulli and Greenspan (9) that only the terms containing the pressure in the momentum equation and the velocity in the energy equation must be treated implicitly in order to avoid the sound-speed limitation on the timestep. BIC-FCT has three steps. In the first step, the conservation equations are solved explicitly with FCT using a relatively large timestep governed by the Courant condition on the fluid velocity. In the second step, the energy and momentum equations are rewritten in terms of a pressure correction, δp. These equations can be manipulated such that only one elliptic equation for δp must be solved. In the third step, final values of momentum and energy are obtained by adding the pressure correction terms. Patnaik et al. (10) have incorporated this algorithm in a two-dimensional flame program to investigate laminar instabilities in premixed flames.

Molecular diffusion

An algorithm for molecular diffusion has been formulated to estimate the molecular diffusion fluxes without having to solve a full matrix problem. The change in species concentration for each species k due to molecular diffusion only is

$$\frac{\partial n_k}{\partial t} = -\nabla \cdot n_k \vec{V}_k \tag{7}$$

where $\vec{V}_k$ is the diffusion velocity calculated using Fick's Law and then corrected by a procedure described by Kee et al. (11) to satisfy the requirement that the sum of the diffusion fluxes is zero. This method is algebraically equivalent to the first iteration of the DFLUX algorithm (12), an iterative approach that solves for diffusion velocities.

The change in total energy density due to molecular diffusion alone is

$$\frac{\partial E}{\partial t} = -\nabla \cdot \sum_{k=1}^{n_{sp}} n_k h_k \vec{V}_k. \tag{8}$$

This energy term is calculated during the diffusion algorithm but is added to the total energy at the end of the time step.

The explicit finite-differencing procedure applied to this term introduces a numerical stability condition, $D_{km}\Delta t/(\frac{1}{\Delta x^2}+\frac{1}{\Delta y^2}) < 1/2$, where D_{km} is the diffusion coefficient for species k diffusing into a mixture. To maintain stability, this condition can require a timestep smaller than that required by the convection, which could add substantially to the cost of the calculation. To avoid this problem, the diffusion term is evaluated several times during a convection timestep. This is especially important if the elevated temperature of the reacting flow results in higher diffusion coefficients.

Binary diffusion coefficients are calculated from kinetic theory and are in the following form

$$D_{kl} = \frac{A_{kl}}{n} T^{B_{kl}} , \tag{9}$$

where A_{kl} and B_{kl} are dependent on species k and l. Values for A_{kl} and B_{kl} have been tabulated by Kailasanath et al. (13). The diffusion coefficient of species k in a mixture of n_{sp} species is calculated according to

$$D_{km} = \frac{1 - Y_k}{\sum\limits_{\substack{kk=1 \\ kk \neq k}}^{n_{sp}} \frac{X_{kk}}{D_{k,kk}}} \tag{10}$$

where Y_k is the mass fraction of species k, X_k is the mole fraction of species k, and $D_{k,kk}$ is the diffusion coefficient of species k diffusing into species kk.

Thermal conduction

A two-dimensional model has also been formulated to simulate thermal conduction. Restricting our attention only to the Fourier conduction term, the energy equation appears as

$$\frac{\partial E}{\partial t} = -\nabla \cdot (\kappa \nabla T) . \tag{11}$$

As with the molecular diffusion algorithm, the use of explicit finite differencing introduces a stability limit for the thermal conduction calculation, $\kappa\Delta t/\rho c_p(\frac{1}{\Delta x^2}+\frac{1}{\Delta y^2}) < 1/2$, where $\kappa/\rho c_p$ is the thermal diffusivity. The thermal conduction term is evaluated several times during each convection time step in the same manner as

the molecular diffusion term. Thermal conductivities, κ_k, for the individual species were calculated from kinetic theory over the temperature range 300K to 3300K, and these values have been fit to a third-order polynomial. The mixture thermal conductivity is then calculated using the expression from Kee et. al (11)

$$\kappa = \frac{1}{2}\left[\sum_{k=1}^{n_{sp}} X_k \kappa_k + \frac{1}{\sum_{k=1}^{n_{sp}} \frac{X_k}{\kappa_k}}\right] . \tag{14}$$

Viscous stress

An algorithm which calculates the viscous terms in the momentum equation is included in the program. The momentum transport associated with viscous diffusion only is

$$\frac{\partial \rho \vec{V}}{\partial t} = -\nabla \cdot \tau \tag{15}$$

where

$$\tau = \left(\frac{2}{3} - \zeta\right)\left(\nabla \cdot \vec{V}\right)\mathbf{I} - \mu\left[\left(\nabla \vec{V}\right) + \left(\nabla \vec{V}\right)^T\right] \tag{16}$$

and ζ is the second coefficient of viscosity and is assumed to equal zero. Equation 15 is rewritten in terms of an explicit finite difference approximation which introduces a numerical stability condition, $\mu \Delta t / \rho(\frac{1}{\Delta x^2} + \frac{1}{\Delta y^2}) < 1/2$.

The values for μ_k were calculated from kinetic theory over the temperature range 300 to 3000K and fit to a third order polynomial. The mixture viscosity was calculated from

$$\mu = \sum_{k=1}^{n_{sp}} \frac{X_k \mu_k}{\sum_{j=1}^{n_{sp}} X_j \Phi_{kj}} \tag{17}$$

where Φ_{kj} is a weighting factor

$$\Phi_{kj} = \frac{1}{\sqrt{8}}\left(1 + \frac{M_k}{M_j}\right)^{-\frac{1}{2}}\left(1 + \left(\frac{\mu_k}{\mu_j}\right)^{\frac{1}{2}}\left(\frac{M_j}{M_k}\right)^{\frac{1}{4}}\right)^2 \tag{18}$$

and M_k is the molecular weight of species k (14).

Model for chemical reaction

Ideally, we would like to simulate the chemical reaction by including a detailed set of elementary reactions to describe the production of the individual species and the energy release in the flame. However, the cost of computer time and memory makes this prohibitive for problems in which the flows are complex.

In the original flame sheet model proposed by Burke and Schumann (15), the fuel and oxidizer react completely and are not permitted to coexist. Thus, the flame is an infinitesimal interface between regions of fuel and oxidizer. In the PDR model developed by Laskey (7), a single global reaction mechanism is used but the fuel and oxidizer do not react instantaneously. Instead, the reaction occurs over a finite time interval. The fraction of fuel which reacts in a particular time step is determined from a separate simulation for a laminar flame. The reaction rate for the laminar flame was adjusted such the the maximum temperature was the adiabatic flame temperature.

Coupling

A complete solution to the governing equations requires solving the terms for individual processes as well as accounting for the interaction among the processes. In the calculations presented below, we use timestep splitting which assumes that the net effect of all the processes is the sum of the solutions to individual processes. This technique is valid if the changes in the dependent variables during a timestep are small.

In these computations, the changes in internal energy resulting from the individual processes are not incorporated into the solution as soon as they are computed, but instead are accumulated. The entire change in internal energy is then added to the energy equation in the fluid convection step. The coupling technique has been described by Oran and Boris (16), and a modification by Patnaik, et al. (176) has been shown to allow for a greater addition of energy per timestep while maintaining numerical stability.

Application of the Model to Jet Diffusion Flames

The initial computational domain and initial conditions for the jet diffusion flame calculations are shown in Figure 1. The domain is 10×67 cm and consists of 128×192 cells. Cells of approximately 0.02 cm are concentrated around the jet exit. Beginning at $r = 1$ cm, the size of each cell is increased by 0.03% over the size of its neighboring cell for both simulations of a nonreacting jet and for the simulation of a reacting jet without viscosity. For the simulation of a reacting jet with viscosity, the cell size is stretched by 0.05% to increase the size of the domain to 27×67 cm. The cells in the axial direction for all simulations are stretched by 0.03% starting at $z = 1$ cm.

A fuel mixture consisting of 78% H_2 and 22% N_2 flows through a jet of radius 0.5 cm at 10 m/s at the lower boundary. Air flows through the outer annular region between $r = 0.5$ and $r = 10.0$ at 15 cm/s. The outer boundary at $r = 10.0$ is a free-slip wall. The inner boundary at $r = 0.0$ is the jet centerline. An outflow boundary is specified at $z = 67$ cm.

Results

Nonreacting Jet without Viscosity

Figure 2 shows the instantaneous contours for radial velocity, axial velocity, and local equivalence ratio. The instability forms approximately 1.5 diameters downstream of the jet The radial velocity is a maximum (500 cm/s) at this point and decreases downstream. The structures appear fairly symmetrical throughout their development. Although they are not evident in this plot, there are weak instabilites at the top of the figure. The contours of local equivalence ratio indicate that the concentration field is significantly affected by the rotating structures.

Nonreacting Viscous Jet

Figure 3 shows that the viscosity does not qualitatively alter the flow. The instability forms approximately 1.5 diameters downstream of the jet and weakens rapidly in the downstream direction. The maximum radial velocity is 300 cm/s and occurs in the center of the first recognizable structure. The axial velocity field and local equivalence ratio are similar to the case without viscosity.

Reacting Jet without Viscosity

When the fuel mixture reacts, the nature of the flow field changes significantly. The radial velocity contours indicate that the initial instablity still forms close to the jet, approximately 3 diameters downstream, but it is very weak at this point. The radial velocity increases downstream and is a maximum (80 cm/s) in the center of the structure at about 10 jet diameters. In addition, the structures are elongated due to the acceleration at the flame interface.

The lower radial velocity decreases the amount of fluid entraine .d results in a lower growth rate of the mixing layer. The axial velocity contours show a narrower mixing layer than in the nonreacting case. The structures have not merged along the centerline and a large potential core region still exists. The contours of equivalence ratio show that the stoichiometric surface is shifted radially outward and is outside the region of intense mixing.

The maximum temperature occurs along the reaction zone and is approximately 2100 K. A large region around the flame is heated due to conduction. The temperature field is distorted slightly at the base of the flame. The reaction in this region is quite intense. As the burned gases are accelerated downstream, coflowing air is entrained. In this simulation, the outside boundary is a free-slip wall and the entrainment establishes a weak recirculation zone at the bottom of the domain.

Reacting Viscous Jet

Figure 5 shows that viscosity has a significant effect on the reacting jet. The magnitude of the radial velocity have been reduced by viscosity. Weak instabilities are still evident at approximately 5 diameters downstream with a maximum velocity of 10 cm/s. Further downstream the flowfield appears laminar. There is a large region of negative radial velocity indicated by dotted lines. The axial velocity contours show that the mixing region is shifted radially outward by the addition of viscosity. A potential core of undisturbed fluid is evident along the jet centerline. The maximum temperature is approximately 2100 K and occurs along the stoichiometric contour ($\phi = 1$).

Discussion

In these calculations, the Reynolds numbers for the jets without viscosity (Figs.

2 and 4) are poorly defined because there is only numerical viscosity which is dependent on the local cell size. The numerical viscosity is clearly lower than the physical viscosity because the velocity field changes when the viscous terms are included. This implies that the effective Reynolds number for the calculations without viscosity is higher than that for the calculations with viscosity.

The Reynolds number of the nonreacting jet (Fig. 3) is 2100 based on jet diameter and the properties of the fuel jet. At this Reynolds number, viscous effects are relatively small. The instabilities are primarily inviscid and viscosity reduces the maximum radial velocity in the flow field without altering the nature of the instability. In the reacting jet (Fig. 5), however, the elevated temperature significantly increases the viscosity which reduces the Reynolds number. At 1000 K, the Reynolds number is 300 and viscous effects are significant.

When reaction is included in the calculations without viscosity (Fig. 4), the heat release decreases the strength of the vortices and delays their formation. This agrees with the inviscid analysis by Mahalingam, et al. (5), which shows that heat release stabilizes the mixing layer. Their work also indicates that the stabilizing effect of heat release is greater as the mixing layer spreads. This suggests that the layer becomes more stable further downstream. In our simulations (Fig. 4), the layer is more stable close to the jet and less stable further downstream. The analysis cannot, however, be compared directly because the heat release is not uniform in the axial direction in the calculations. The most intense reaction occurs close to the jet where pure reactants mix. Further downstream where the reactants are mixed with products, the heat release decreases significantly.

The calculations which do not include viscosity (Fig. 2 and 4) also agree with experimental data which show that heat release decreases the layer thickness (6) in high Reynolds number flows. When viscosity is included, the results are difficult to compare, because the Reynolds number for the calculations is much less than that in the experiments.

The calculations in this paper show that heat release stabilizes the flow field of a $H_2 - N_2$ diffusion flame. When viscosity is added to the calculation for a reacting jet, the flow becomes almost laminar. In future work, we will calculate mean quantities in order to compare to experimental data.

Acknowledgements

This work was sponsored by the Naval Research Laboratory through the Office of Naval Research. The authors would like to thank Professor Norman Chigier from Carnegie-Mellon University for introducing us to the many interesting problems in diffusion flames. In addition, we thank Dr. W.M. Roquemore from the Air Force Wright-Patterson Aeronautics Laboratory for his support and suggestions.

References

1. Yule, A.J., Chigier, N.A., Ralph, S., Boulderstone, R., and Ventura, J., "Combustion-Transition Interaction in a Jet Flame," *AIAA J.*, vol. 19, no. 6, 752-760, 1981.
2. Chen, L.D., Seaba, J.P., Roquemore, W.M., and Goss, L.P., "Buoyant Diffusion Flames," Proceedings of *Twenty-Second Symposium (International) on Combustion*, Seattle, WA, Aug. 14-19, 677-684, 1988.
3. Chamberlain, D.S. and Rose, A., "The Flicker of Luminous Flames," Proceedings of *First Symposium on Combustion*, Swampscott, MA, Sept. 10-14, 27-35, 1928.
4. Laskey, K.J., Ellzey, J.L., and Oran, E.S., "A Numerical Study of an Unsteady Diffusion Flame," AIAA paper 89-0572 presented at the 27th Aerospace Sciences Meeting, Jan. 9-12, Reno, NV, 1989.
5. Mahalingam, S., Cantwell, B., and Ferziger, J., "Effects of Heat Release on the Structure and Stability of a Coflowing, Chemically Reacting Jet," AIAA paper 89-0661, presented at the 27th Aerospace Sciences Meeting, Jan. 9-12, Reno, NV, 1989.
6. Hermanson, J.C., Mungal, M.G., and Dimotakis, P.E., "Heat Release Effects on Shear Layer Growth and Entrainment," AIAA paper 85-0142, 23rd AIAA Aerospace Sciences Meeting, Reno, NV, 1985.
7. Laskey, K. J., *Numerical Study of Diffusion and Premixed Jet Flames*, Ph.D. dissertation, Department of Mechanical Engineering, Carnegie-Melon University, Pittsburgh, PA, 1988.
8. Patnaik, G., Boris, J.P., Guirguis, R.H., and Oran, E.S., "A Barely Implicit Correction for Flux-Corrected Transport", *J. Computational Physics*, 71, 1-20, 1987.
9. Casulli, V., and Greenspan, D., "Pressure Method for the Numerical Simulation of Transient, Compressible Fluid Flows", *Int. J. Num. Methods Fluids* , 4, 1001, 1984.
10. Patnaik, G., Kailasanath, K., Laskey, K.J., and Oran, E.S., "Detailed Numerical Simulations of Cellular Flames," *Twenty-Second Symposium (International) on Combustion*, Seattle, WA, Aug. 14-19, 1988.
11. Kee, R.J., Dixon-Lewis, G., Warnatz, J., Coltrin, M.E., and Miller, J.A., *A Fortran Computer Code Package for the Evaluation of Gas-Phase Multicomponent Transport Properties*, SAND86-8246, Sandia National Laboratory, 1986.
12. Jones, W.W.and Boris, J.P., "An Algorithm for Multispecies Diffusion Fluxes," *Comp. and Chem.*, vol. 5, 139-146, 1981.
13. Kailasanath, K., Oran, E.S., and Boris, J.P., "A One-Dimensional Time-Dependent

Model for Flame Initiation, Propagation, and Quenching", NRL Memorandum Report 4910, Naval Research Laboratory, 1982.

14. Wilke, C.R., "A Viscosity Equation for Gas Mixtures," *J. Chem. Phys.*, vol. 18, 578-579, 1950.
15. Burke, S.P., and Schumann, T.E.W., *Indust. Eng. Chem.* 20, (10), 998-1004, 1928.
16. Oran, E.S., Boris, J.P., *Numerical Simulation of Reactive Flow*, Elsevier, New York, 1987.
17. Patnaik, G., Laskey, K.J., Kailasanath, K., Oran, E.S., and Brun, T.A., "FLIC–A Detailed, Two-Dimensional Flame Model," NRL Memorandum Report (in preparation), Naval Research Laboratory, 1988.

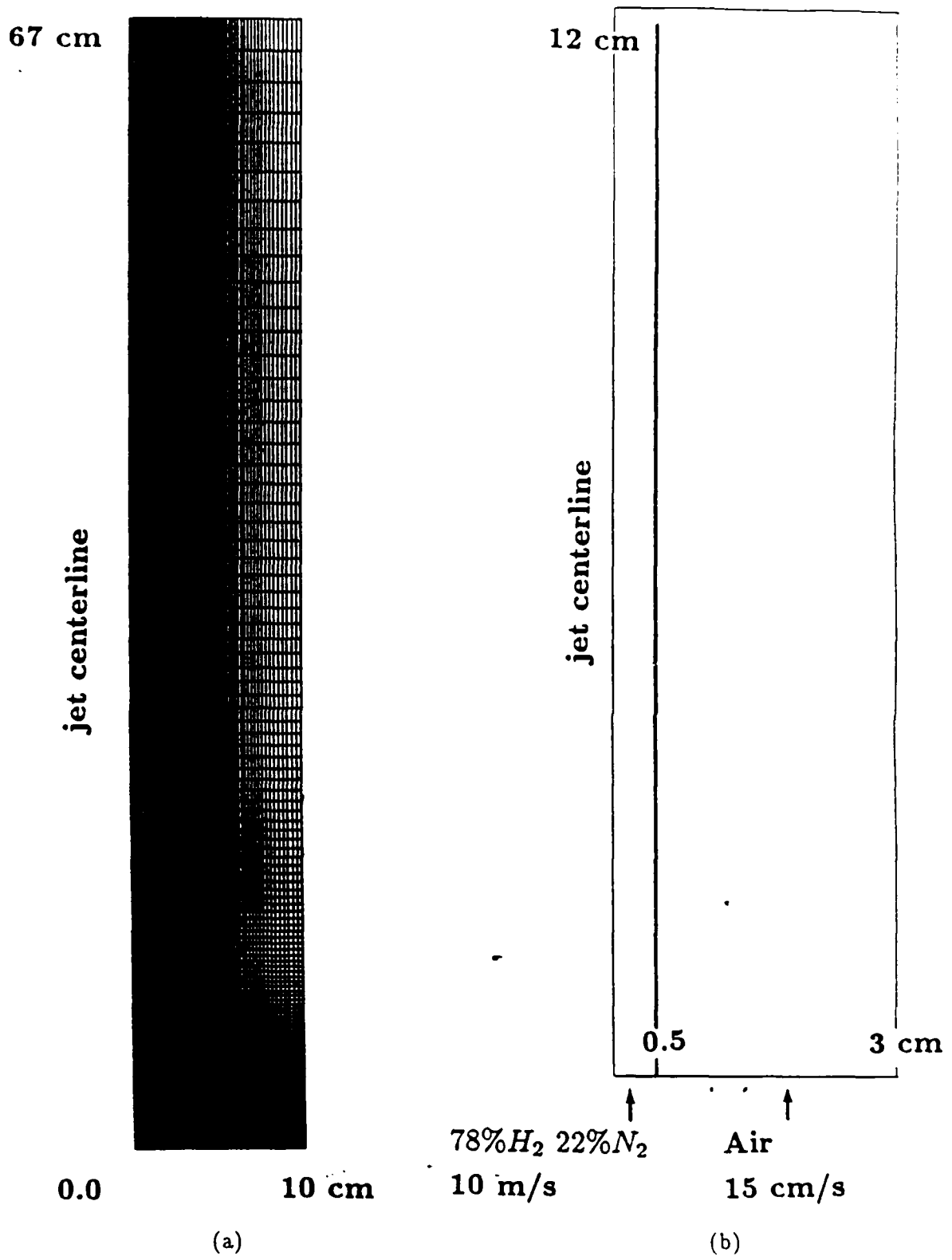


Figure 1. (a) Computational domain and (b) initial conditions for $H_2 - N_2$ jet calculations. Figure 1b shows only part of the full computational domain.

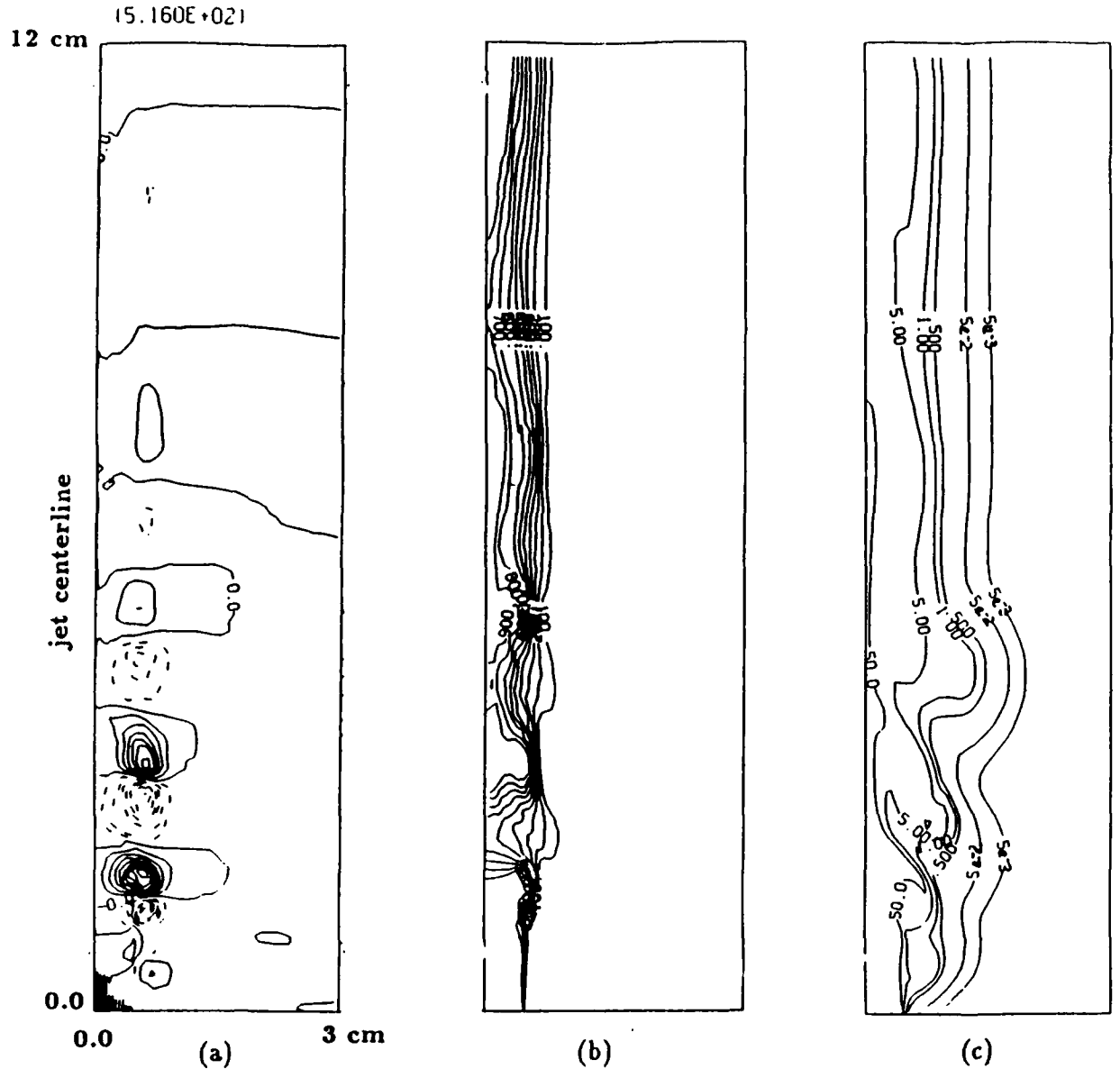


Figure 2. Contours of (a) radial velocity (b) axial velocity (cm/s) (c) local equivalence ratio for nonreacting $H_2 - N_2$ jet without viscosity. Radial velocities are non-dimensionalized by value (cm/s) given at top of plot and contour values range from -1.0 to 1.0 in 0.1 increments. Dotted lines indicate negative values. Solid lines indicate positive values. Lowest contour value for axial velocity is 100 cm/s; highest is 900 cm/s.

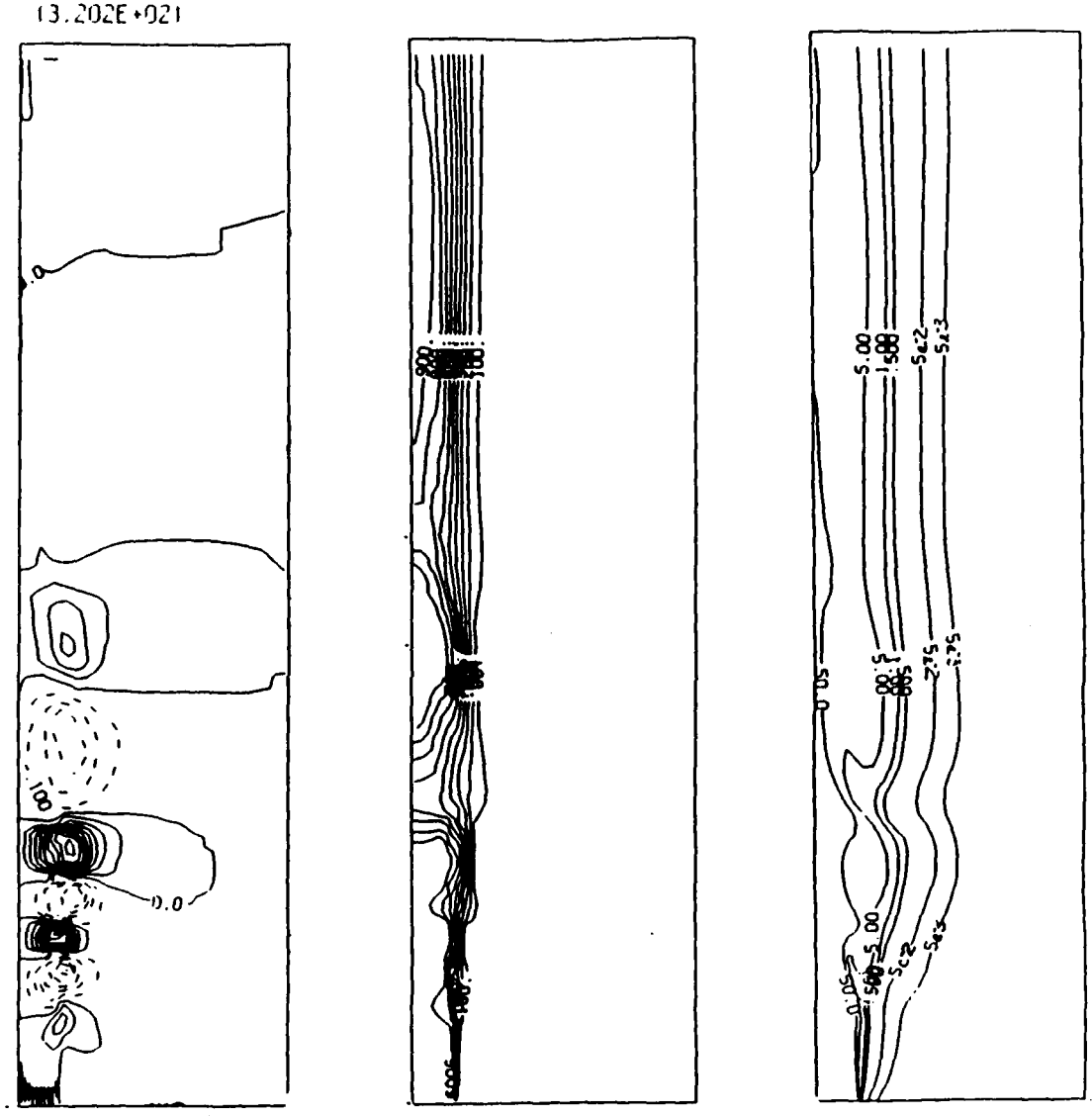


Figure 3. Contours of (a) radial velocity (b) axial velocity (c) local equivalence ratio for nonreacting $H_2 - N_2$ jet with viscosity.

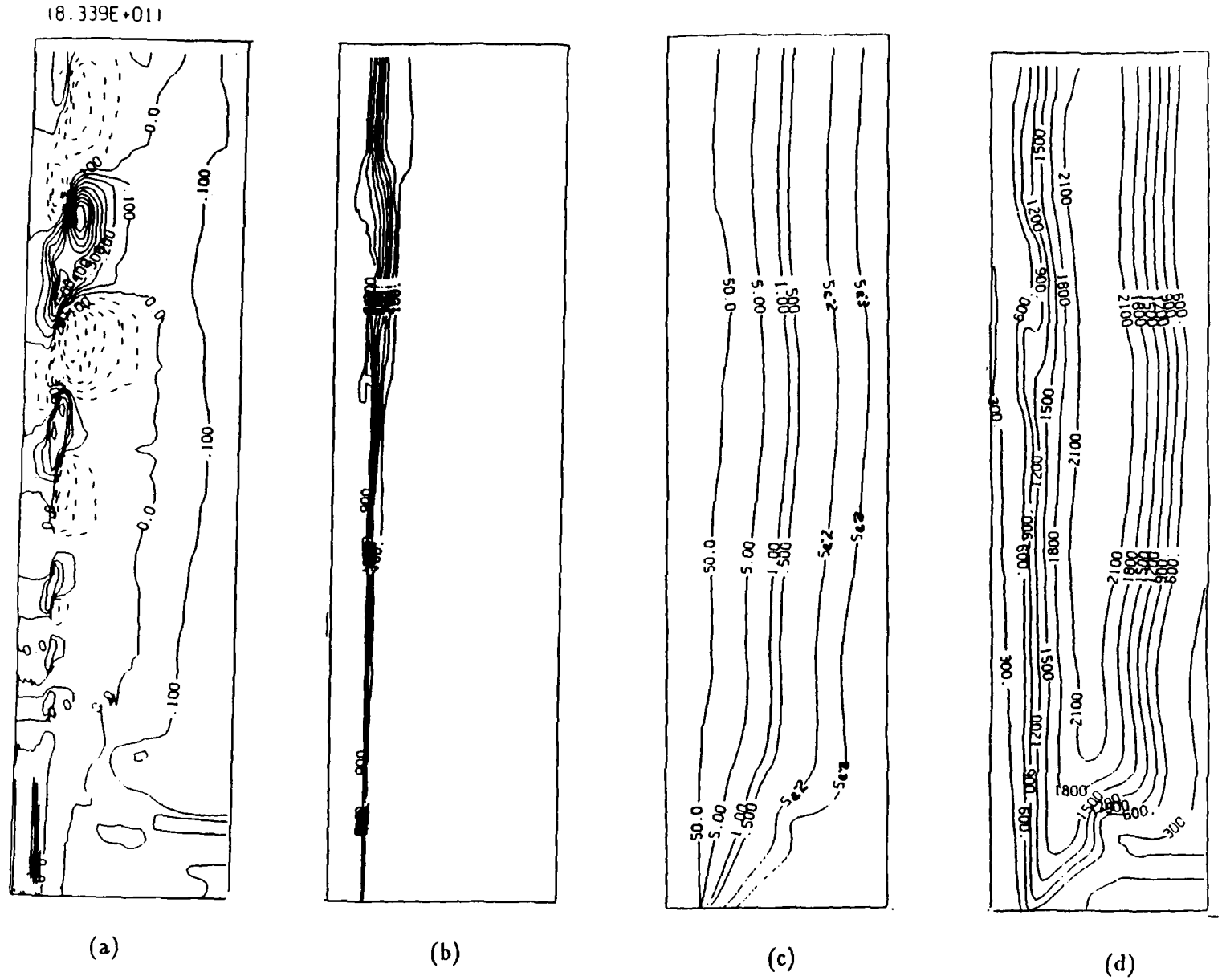


Figure 4. Contours of (a) radial velocity (b) axial velocity (c) local equivalence ratio (d) temperature for reacting $H_2 - N_2$ jet without viscosity.

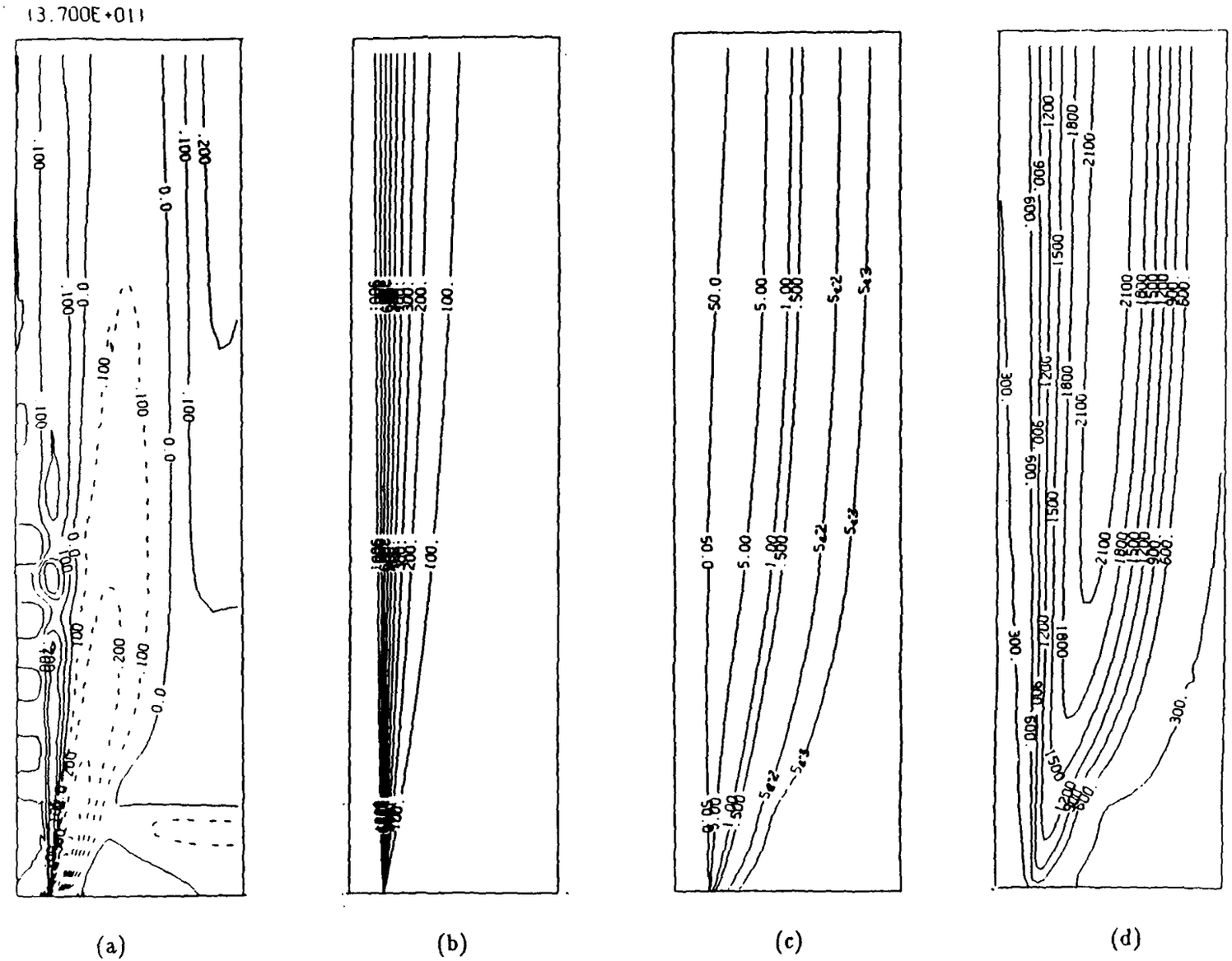


Figure 5. Contours of (a) radial velocity (b) axial velocity (c) local equivalence ratio (d) temperature for reacting $H_2 - N_2$ jet with viscosity.

APPENDIX P.

A Study of Confined Diffusion Flames (submitted to Combustion and Flame)

Submitted to Combustion & Flame

A Study of Confined Diffusion Flames

J.L. Ellzey*, K.J. Laskey**, and E.S. Oran
Laboratory for Computational Physics and Fluid Dynamics
Naval Research Laboratory
Washington, D.C. 20375

Abstract

A numerical simulation of an axisymmetric confined diffusion flame formed between a H_2-N_2 jet and coflowing air at 30 cm/s is presented in this paper. For the initial computations, the restrictions of the Burke-Schumann theory are imposed and the results of the computation are compared with the analytical solution for flame location. For both the underventilated and overventilated flames, the results of the computations are in excellent agreement with the analytical solution. However, the flame behavior becomes more complex as the restrictions are relaxed. When variable diffusion coefficients and densities are included in the calculation, small radial velocities are induced and the flame interface is slightly distorted. When heat release is included, the flame is shorter and an unsteady mixing region forms at the fuel-oxidizer interface. The instabilities are damped when viscous effects are included. Large-scale instabilities form in the oxidizer region with a frequency of approximately 15-20 Hz when gravity is included in the calculation.

I. Introduction

The analytical work of Burke and Schumann (1) has formed many of our fundamental ideas about laminar diffusion flames. In the original Burke-Schumann problem, axisymmetric coflowing streams of fuel and oxidizer flow through a confined duct, and the velocities, densities, and diffusion coefficients of the fuel and oxidizer are equal. Reaction is instantaneous, resulting

* Berkeley Research Associates, Springfield, VA
** Grumman Space Station Program Support Division, Reston, VA

in a flame sheet of infinitesimal thickness in which the reaction rate is effectively controlled by the diffusion rate. The solution to this problem is an equation which can be solved for the location of the flame front. In the original analysis, Burke and Schumann chose the diffusion coefficients in order to obtain good agreement with experiments. Later work (2) extended the theory to describe the flame interface for unequal velocities and diffusion coefficients. Further extensions of the theory predict the behavior of multiple, coupled diffusion flames (3,4).

Many of the restrictive assumptions in the Burke-Schumann analysis can be removed by a numerical solution of the reactive flow equations. In particular, there are a number of steady-state numerical solutions that simulate laminar diffusion flames. Gosman et al. (5) solved the two-dimensional steady-state equations for a case in which all of the diffusion coefficients were the same and the Lewis number was unity. Mitchell et al. (6) numerically simulated steady-state flames with nonunity Lewis numbers but kept the basic idea of the flame sheet model.

This paper describes a numerical model which is designed specifically to simulate the nonsteady behavior of axisymmetric diffusion flames. It contains submodels for finite-rate chemistry, viscosity, thermal conduction, and the temperature dependence of material properties such as specific heats and diffusion coefficients. As one of the first uses of the diffusion flame model, we simulate a Burke-Schumann flame and remove the restrictions individually. We present results for a classic Burke-Schumann flame with all of the restrictions included in the analysis and compare to the anlytical solution. Then we include the following sequentially:

1. Variable density and diffusion coefficients, radial convection, and axial diffusion,
2. Heat release,
3. Viscous effects,
4. Gravitational effects.

This set of computations is a benchmark of the model that is currently being applied to more complex transitional diffusion flames. Although practical examples of Burke-Schumann flames are not as abundant as turbulent or unsteady flames, they represent an important class of problems which can be studied through theory, computation, and experiment.

Besides the applications of the model to the Burke-Schumann problem, this paper also describes the numerical model in detail. Specific notable features of the model are its time dependence, the finite-rate chemical model, the temperature dependence of the transport coefficients, and the nonunity Lewis number. The new elements of the numerical model that make such computations possible are the BIC-FCT algorithm used to compute the convection and the parametric diffusion-reaction (PDR) model used for the finite-rate chemistry. These features and algorithms are described in some detail in the next section of this paper.

II. Numerical Methods and the Model Structure

The numerical model used for this study is based on those originally developed by Patnaik et al. (7) to simulate low-speed premixed flames and by Laskey (8) to simulate jet flames . The program solves the equations for conservation of mass density, momentum, energy, and individual species number densities:

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{v}) = 0 \tag{1}$$

$$\frac{\partial \rho \mathbf{v}}{\partial t} + \nabla \cdot (\rho \mathbf{v}\mathbf{v}) = -\nabla P + \rho \mathbf{G} - \nabla \cdot \tau \tag{2}$$

$$\frac{\partial E}{\partial t} + \nabla \cdot (E\mathbf{v}) = -\nabla \cdot P\mathbf{v} + \nabla \cdot (\kappa \nabla T) - \nabla \cdot \sum_{k=1}^{n_{sp}} n_k \mathbf{v}_k h_k + Q \tag{3}$$

$$\frac{\partial n_k}{\partial t} + \nabla \cdot (n_k \mathbf{v}) = -\nabla \cdot (n_k \mathbf{v}_k) + w_k \ , \tag{4}$$

with the additional relations

$$P = nkT \ , \tag{5}$$

and

$$d\epsilon = \rho c_v \, dT \ . \tag{6}$$

The various quantites used in these equations and throughout this article are defined in the accompanying Nomenclature.

Equations (1) – (4) contain terms representing convection, thermal conduction, species diffusion, chemical reactions, and viscosity. These equations are then rewritten in terms of finite-difference approximations on an Eulerian mesh and solved numerically for specified boundary and initial conditions. The accuracy of the solution is determined by the specific finite-difference algorithm, the spatial resolution set by the computational grid, and the temporal resolution set by the timestep. There is a wide range of important spatial and temporal scales in reacting flow problems. Because it is not usually possible to resolve phenomena on all of these scales, the smallest scales must be modeled phenomenologically.

However, as computational capabilities and model inputs improve, it should be possible to replace certain submodels by more accurate or faster submodels. For example, assuming that the rate of diffusion is much less than the reaction rate helps justify using a global reaction mechanism. Using such a global reaction is, however, approximate and we believe that within a few years it will be possible to include a detailed set of elementary reaction rates. Therefore, the computer program is designed in a modular form so that particular submodels can be updated in a relatively straightforward way. In the computer program, algorithms representing different physical processes are solved separately and then the results are combined, as summarized below. More detailed descriptions are presented by Laskey (8).

Convection

The solution to the convective terms in Eqs. (1) – (4) is obtained using the new algorithm, Barely Implicit Correction to Flux-Corrected Transport (BIC-FCT), that was developed to solve the convection equations for

low-velocity flows (9). The Flux-Corrected Transport algorithm itself is an explicit, finite-difference algorithm that is constructed to have fourth-order phase accuracy (10). Through a two-step predictor-corrector algorithm, FCT ensures that all conserved quantities remain monotone and positive. The FCT procedure is to first modify the properties of a high-order algorithm by adding diffusion during a convection step and then to subtract out the diffusion in an antidiffusion phase. In addition, fluxes are limited to ensure that no new unphysical maxima or minima are added during the convection process.

However, because FCT is an explicit algorithm, the numerical timestep required for accuracy and stability is limited by the velocity of sound according to the Courant-Friedrichs-Lewy condition, $\Delta t < min(\Delta x/c_s)$. To avoid this restriction, which would make computations of slowly evolving flows prohibitively expensive, the convection equations are usually solved implicitly. This filters out the sound waves from the equation and, therefore, removes the sound-speed condition. Patnaik et al. (7) developed BIC-FCT so that the timestep would be limited by the fluid velocity and not the sound speed. This implementation has great advantages for computations of slow flows because one BIC-FCT timestep costs the same as one regular FCT explicit timestep, but the size of the timestep might be a factor of 50 to 100 times greater.

BIC-FCT is based on the idea proposed by Casulli and Greenspan (11) that only the terms containing the pressure in the momentum equation and the velocity in the energy equation must be treated implicitly in order to avoid the sound-speed limitation on the timestep. BIC-FCT has three steps. In the first step, the conservation equations are solved explicitly with FCT using a relatively large timestep governed by fluid velocity. In the second step, the energy and momentum equations are rewritten in terms of a pressure correction, δP. These equations can be manipulated such that only one elliptic equation for δP must be solved. In the third step, final val-

ues of momenta and energy are obtained by adding the pressure correction terms. Patnaik et al. (7) have used this algorithm in a two-dimensional flame program to investigate laminar instabilities in premixed flames.

The form used for the relation between the change in internal energy and the change in the pressure, required in the second step of BIC-FCT, can be derived from Eq. 6 and simplifies under certain assumptions (8). For example, if all of the constituents have the same temperature dependence, we can write

$$d\epsilon = \frac{\delta P}{\gamma - 1} + \left(\epsilon - \frac{P}{\gamma - 1}\right) d(\ln n) , \tag{7}$$

where γ is a mixture quantity. The second term in Equation (7) is dependent on the change in the local species concentration and is important only in the reaction zone. In our simplified reaction submodel, we do not include the intermediate species. Consequently, this term cannot be represented accurately and we do not include it explicitly. In the reaction zone, we are primarily interested in representing a finite flame thickness and reproducing the correct maximum temperature. We calibrate the reaction submodel to predict a realistic flame temperature and flame thickness without this term included.

Molecular Diffusion

An algorithm for molecular diffusion has been formulated to estimate the molecular diffusion fluxes without having to solve a full matrix problem. The change in species concentration for each species k due to molecular diffusion is

$$\frac{\partial n_k}{\partial t} = -\nabla \cdot n_k \mathbf{v}_k \tag{8}$$

where the diffusion velocity, $\mathbf{v}_k$, is calculated from Fick's Law,

$$\mathbf{v}_k = -\frac{1}{X_k} D_{km} \nabla X_k , \tag{9}$$

and then corrected by a procedure described by Kee et al. (12) to satisfy the requirement that the sum of the diffusion fluxes is zero. This method is

algebraically equivalent to the first iteration of the DFLUX algorithm (13), an iterative approach that solves for diffusion velocities to optimal accuracy.

The change in total energy density due to molecular diffusion alone is

$$\frac{\partial E}{\partial t} = -\nabla \cdot \sum_{k=1}^{n_{sp}} n_k h_k \mathbf{v}_k \ . \tag{10}$$

This energy term is calculated within the diffusion algorithm but is added to the total energy at the end of the time step.

The explicit finite-differencing procedure applied to this term introduces a numerical stability condition,

$$D_{km} \Delta t \left(\frac{1}{\Delta x^2} + \frac{1}{\Delta y^2} \right) < \frac{1}{2} \ , \tag{11}$$

where D_{km} is the diffusion coefficient for species k diffusing into a mixture. To maintain stability, this condition may require a timestep smaller than that required by the convection condition, which adds substantially to the cost of the calculation. To avoid this problem, the diffusion term is evaluated several times during a convection timestep. This is especially important if the elevated temperature of the reacting flow results in higher diffusion coefficients.

Binary diffusion coefficients were calculated from kinetic theory and are in the following form

$$D_{kl} = \frac{A_{kl}}{n} T^{B_{kl}} \ , \tag{12}$$

where A_{kl} and B_{kl} depend on species k and l. Values for A_{kl} and B_{kl} have been tabulated by Kailasanath et al. (14). The diffusion coefficient of species k in a mixture of n_{sp} species is calculated according to

$$D_{km} = \frac{1 - Y_k}{\sum\limits_{\substack{kk=1 \\ kk \neq k}}^{n_{sp}} \frac{X_{kk}}{D_{k,l}}} \tag{13}$$

where Y_k is the mass fraction of species k, X_k is the mole fraction of species k, and $D_{k,l}$ is the diffusion coefficient of species k diffusing into species l (15).

Thermal Conduction

A two-dimensional model has also been formulated to simulate thermal conduction. Restricting our attention only to the Fourier conduction term, the energy equation appears as

$$\frac{\partial E}{\partial t} = -\nabla \cdot (\kappa \nabla T) \; . \tag{14}$$

As with the molecular diffusion algorithm, the use of explicit finite differencing introduces a stability limit for the thermal conduction calculation,

$$\frac{\kappa \Delta t}{\rho c_p} \left(\frac{1}{\Delta x^2} + \frac{1}{\Delta y^2} \right) < \frac{1}{2} \; , \tag{15}$$

where $\kappa / \rho c_p$ is the thermal diffusivity. The thermal conduction term is evaluated several times during each convection time step in the same manner as the molecular diffusion term. Thermal conductivities, κ_k, for the individual species were calculated from kinetic theory over the temperature range 300 K to 3300 K, and these values were fit to a third-order polynomial. The mixture thermal conductivity is then calculated using the expression from Kee et al. (12)

$$\kappa = \frac{1}{2} \left[\sum_{k=1}^{n_{sp}} X_k \kappa_k + \frac{1}{\sum\limits_{k=1}^{n_{sp}} \frac{X_k}{\kappa_k}} \right] \; . \tag{16}$$

Viscous Stress

The momentum transport associated with viscous diffusion only is

$$\frac{\partial \rho \mathbf{v}}{\partial t} = -\nabla \cdot \tau \tag{17}$$

where

$$\tau = \left(\frac{2}{3} \mu - \zeta \right) (\nabla \cdot \mathbf{v}) \mathbf{I} - \mu \left[(\nabla \mathbf{v}) + (\nabla \mathbf{v})^T \right] \tag{18}$$

and ζ is the second coefficient of viscosity and is assumed to equal zero. Equation 17 is rewritten in terms of an explicit finite difference approximation which introduces a numerical stability condition,

$$\frac{\mu \Delta t}{\rho}\left(\frac{1}{\Delta x^2}+\frac{1}{\Delta y^2}\right) < \frac{1}{2}\,. \tag{19}$$

The values for μ_k were calculated from kinetic theory over the temperature range 300 K to 3000 K and fit to a third-order polynomial. The mixture viscosity is calculated from

$$\mu = \sum_{k=1}^{n_{sp}} \frac{X_k \mu_k}{\sum_{j=1}^{n_{sp}} X_j \Phi_{kj}} \tag{20}$$

where Φ_{kj} is a weighting factor

$$\Phi_{kj} = \frac{1}{\sqrt{8}}\left(1+\frac{M_k}{M_j}\right)^{-\frac{1}{2}}\left(1+\left(\frac{\mu_k}{\mu_j}\right)^{\frac{1}{2}}\left(\frac{M_j}{M_k}\right)^{\frac{1}{4}}\right)^2 \tag{21}$$

and M_k is the molecular weight of species k (16).

Model for Chemical Reactions

Ideally, we would like to simulate the chemical reaction by including a detailed set of elementary reactions to describe the production of the individual species and the energy release in the flame. However, the cost of computer time and memory makes this prohibitive for simulations of complex flows. We have, therefore, used the parametric diffusion-reaction (PDR) model that reflects many of the major characteristics of a more detailed calculation.

In the original flame-sheet model proposed by Burke and Schumann (1), the fuel and oxidizer react completely and are not permitted to coexist. Thus, the flame is an interface of infinitesimal thickness between regions of fuel and oxidizer. In the PDR model developed by Laskey (8), a single global reaction mechanism, $2H_2 + O_2 \rightarrow 2H_2O$, is used but the fuel and oxidizer do not react instantaneously. Instead, the reaction occurs over several time steps in the simulation. The heat release rate, Q, is determined from

$$Q = -C q_f \frac{dn_f}{dt} \tag{22}$$

where q_f is the heat of combustion, dn_f/dt is the reaction rate of the fuel, and C is a calibration constant.

In a real flame, the increase in temperature in the reaction zone is a result of the change in internal energy of the local mixture which includes reactants, product, and intermediates. In our simulation, we do not include the intermediates and, as a result, the specific heat of the gas is not represented accurately in the flame zone. Even if we obtain a realistic overall rate for the global reaction, the temperature in the flame zone is too high unless we compensate for the inaccuracy in the specific heat.

The two important physical characteristics of the flame zone are the maximum temperature and thickness. We determined the constant reaction rate in a one-dimensional transient diffusion flame such that a thin reaction zone developed. In addition, we adjusted the calibration constant in equation (22) such that the maximum temperture was approximately the adiabatic flame temperature for a stoichiometric mixture of fuel and oxidizer.

Coupling

A complete solution to the governing equations requires solving the terms for individual processes as well as accounting for the interaction among the processes. In the calculations presented below, we use timestep splitting, which assumes that the net effect of all the processes is the sum of the solutions to individual processes. This technique is valid if the changes in the dependent variables during a timestep are small. Table 1 is an outline that shows the order of the computations during one timestep in the computer code.

Table 1. Outline of Diffusion Flame Code

Initialize Variables
* Increment time
1. Thermal Conduction
    Integrate from t to $t + \Delta t$:
    Calculate $\Delta\epsilon_1$
    Do not update any variables
    (Subcycle as necessary)
2. Ordinary Diffusion
    Integrate from t to $t + \Delta t$:
    Only update $\{n_i(x)\}$
    Calculate $\Delta\epsilon_2$
    (Subcycle as necessary)
3. Viscosity
    Integrate from t to $t + \Delta t$:
    Only update $\rho\mathbf{v}$
4. Chemical Reactions
    Integrate from t to $t + \Delta t$:
    Only update $\{n_i(x)\}$
    Calculate $\Delta\epsilon_3$
5. Convective Transport
    Integrate from t to $t + \Delta t$:
    x direction transport
        Update $\rho, \rho\mathbf{v}, E, n_i$
    y direction transport
        Update $\rho, \rho\mathbf{v}, E, n_i$
    Implicit correction– update p, ϵ, and E
Start New Timestep (go to * above)

Due to different requirements of accuracy and stability, the type of coupling used for lower-velocity implicit calculations is different from that needed for higher-velocity explicit calculations. General information on these approaches is described in some detail in Reference (10), Chapter 13. In these implicit computations, the changes in pressure or internal energy resulting from the individual processes should not be added into the solution as soon as they are computed, but instead should be accumulated over the timestep.

The entire change in internal energy is then included in the fluid convection step. The specific coupling technique used in this program (17) allows larger changes in variables per timestep while maintaining numerical stability.

III. Application of the Algorithm to a Confined Diffusion Flame

The numerical procedure described above was used to simulate several confined diffusion flames. The geometry used in the calculations (Figure 1) consists of an inner jet of radius a and an outer annular region between the jet and the walls at radius b. Typically, fuel flows through the inner jet and oxidizer flows through the outer annular region. The appropriate boundary conditions for this geometry are

1. $r = 0$ is a line of symmetry,
2. $r = b$ is a solid, adiabatic, free-slip wall,
3. $z = 0$ is an inflow boundary where the concentrations, velocities, and temperatures of the fuel, oxidizer, and inert are specified,
4. $z = z_1$ is an outflow boundary where the pressure is adjusted to equal 1 atmosphere.

For the Burke-Schumann flame, $a = 1$, $b = 2$, $z_1 = 10$ cm. The computational domain consists of a 32×88 grid. The grid spacing is uniform in the radial and the axial directions. Calculations on finer grids, such as 64×176, resulted in smoother flame interfaces but did not change the result significantly. The computational time step is 1 ms.

For the simulations of a confined diffusion flame without all of the Burke-Schumann restrictions, $a = 0.5$, $b = 2.5$, $z_1 = 10$ cm. The grid consisted of 64×88 cells with fine cells concentrated around the jet exit. In the radial direction, the grid spacing is approximately 0.02 cm from the centerline to $r = 0.7$ cm and then expands gradually to a grid spacing of 0.12 at $r = 2.5$ cm. In the axial direction, the grid spacing is uniform through the domain and equals 0.11 cm. A typical timestep is 10 μs.

Computational Time Requirements

A two-dimensional simulation, which includes convection, chemical reaction, molecular diffusion, viscous diffusion, and conduction, requires approximately 25 μs per grid point per timestep on a Cray Y-MP. The convection algorithm requires approximately twice the cpu time of either the molecular diffusion or the viscous diffusion algorithms and four times that of the conduction algorithm. The parametric diffusion-reaction flame model requires insignificant cpu time.

IV. The Burke-Schumann Flame

Burke and Schumann (1) found a solution for a set of equations that give the location of the flame interface for a laminar diffusion flame under a certain set of limiting conditions. The Burke-Schumann analysis of the laminar diffusion flame and a comparison of their analysis to the computatuional results are presented in this section.

In order to solve the equations, Burke and Schumann invoked a number of simplifying assumptions:

1. The velocities of the fuel and oxidizer are equal and uniform everywhere,
2. The radial velocity is zero,
3. The densities and diffusion coefficients are equal for all components,
4. Radial diffusion is much greater than axial diffusion,
5. Reaction takes place at an infinitesimal flame sheet.

With these assumptions, the conservations equations can be reduced to a single species equation which is solved with the following boundary conditions:

1. $r = b$ is a solid wall,
2. $r = 0$ is a line of symmetry,
3. at $z = 0$, the compositions of the fuel and oxidizer streams are specified.

The analytic solution to the equation yields the location of the flame surface as a function of a/b and the initial concentrations of fuel and oxidizer.

These assumptions enforce various unrealistic restrictions on the flow field. If the velocity is uniform across the radius of the tube, then the no-slip condition cannot be imposed at the wall boundary and, as a result, a parabolic velocity profile typical of confined flows cannot develop. The assumption of equal densities requires that one mole of fuel reacts with s moles of oxidizer to form $1 + s$ moles of product. Finally, in a real flame, the volumetric expansion associated with heat release distorts the one-dimensional flow resulting in a radial component to the velocity and a nonuniform density field. Consequently, the heat release and expansion are not considered in the Burke-Schumann analysis.

Figure 1 shows two general cases that can be solved with the Burke-Schumann approach: the underventilated flame, which has insufficient oxidizer for complete burning, and the overventilated flame, which has excess oxidizer. In the overventilated case, the fuel is completely consumed in the reaction and the flame surface is closed at the centerline of the jet. In the underventilated case, the flame surface bends outward and is attached to the outer wall. The two different cases may be obtained by changing either the ratio a/b or the composition of the fuel or oxidizer stream.

While the details of most flames cannot be represented realistically by the results of the Burke-Schumann analysis, the predictions of the analysis are surprisingly good for steady, laminar flames. In addition, it provides an analytical result against which to test the numerical model.

Simulation of the Burke-Schumann Flame

For the first cases considered in this study, the ratio of the radii, a/b, is 0.5 ($a = 1$ cm, $b = 2$ cm), and the fuel flows in the inside jet and oxidizer flows in the outside annular region. The velocities of the fuel and oxidizer streams are uniform and equal to 10 cm/s. The densities of the two streams are equal but the composition was varied by diluting the fuel or oxidizer stream with an inert gas. All diffusion coefficients were equal to an equivalent mixture of H_2 and N_2 diffusing into O_2.

Figure 2 shows a sequence of fuel and oxidizer concentration contours for the numerical simulation of the Burke-Schumann flame as it evolves from the initial condition to a steady state. At $t = 0$, pure fuel exists at $r < 1$ cm and an oxidizer mixture, consisting of one part oxidizer and one part inert by volume, exists at $r > 1$ cm. Since there is an overall excess of oxidizer, the fuel and oxidizer interface moves inward during time steps 0 to 1000 because the fuel is completely consumed in the reaction at the flame surface. In all cases, fuel and oxidizer do not coexist because the Burke-Schumann flame sheet model assumes that the reaction goes to completion. At time step 1000 which equals 1 second of physical time, the concentration field has reached a steady state.

In Figure 3a, the computed contours for 1% of the inlet fuel concentration and 1% of the inlet oxidizer concentration are superimposed on the Burke-Schumann solution for the flame front. The analytic solution is the interface between the fuel and oxidizer regions and should correspond to the zeroth contour for either the fuel or the oxidizer, and it should occur between the two 1% contours. Figure 3b shows a similar calculation for a fuel mixture of one part fuel and three parts by volume of inert reacting with pure oxidizer. In both cases, the computed flame front is within one cell of the analytical solution.

A similar computation was conducted for an underventilated Burke-Schumann flame. In this case, the inlet jet was pure fuel and the oxidizer consisted of one part of oxidizer to four parts inert. The steady state fuel contours are shown in Figure 3c with the analytic solution superimposed. In this case, the flame bends outward and attaches to the outer wall. Again, the flame shape and height are within the accuracy of the calculation.

V. Elimination of the Burke-Schumann Restrictions

The restrictions on the Burke-Schumann analysis prevent its application to a wide range of problems. In this section, we describe how the computed results are affected by eliminating some of the restrictions in the analysi.

In these computations, the ratio of the inner to the outer radii is 0.2 (a = 0.5, b = 2.5) and the inlet velocity is 30 cm/s. The fuel consists of 3.41 parts of H_2 to 1 part N_2 by volume and the oxidizer is air.

Introduction of correct stoichiometry, densities, and diffusion coefficients, but maintaining the conditions of uniform inlet velocity and isothermal reaction, required that radial gradients be included. The full system of equations (1) - (4) were solved for this case and the results are shown in Figure 4 after 0.4 s. Figures 4a and 4b show the radial and axial velocity contours and Figures 4c and 4d show the contours of fuel and oxidizer mole fractions. Small radial velocities are induced in the reaction zone even though heat release is not included. The maximum axial velocity of approximately 35 cm/s occurs about 0.25-0.50 cm from the jet centerline. There is a region of lower velocity within the fuel-rich zone close to the centerline which initially consisted of pure fuel mixture. There is only a small region very close to the inlet which is still pure fuel mixture because product has diffused across the jet. This diffusion of heavier gases increases the density in the fuel zone. Because momentum is conserved, the velocity decreases. The flame interface lies between the lowest fuel and oxidizer contours. The flame shape is slightly distorted due to the fluctuations in the radial velocity.

So far, heat release from the chemical reaction has not been included, i.e. Q in Equation (3) is zero. The effects of including heat release are shown in Figure 5 after approximately 0.3 seconds. The volumetric expansion associated with the heat release accelerates the flow resulting in maximum axial velocities of 120 cm/s. The hot gases are accelerated outward in the radial direction with a maximum velocity of about 9 cm/s (Fig. 5b). The radial velocity contours show that the flame is not steady but has vortices which form at the fuel/oxidizer interface. This flame is shorter than that predicted when heat release is not included in the calculation. The increased radial velocity provides an additional mechanism for mixing so the reaction zone is wider and the flame is shorter.

When viscous effects are included in the calculation, the flow field changes again and these results are shown in Figure 6. The gradients in the axial velocity are reduced and the vortical structures are damped. The concentration and temperature fields are similar to those without viscosity. This flame is slightly longer than the flame without viscosity because the radial mixing has been reduced.

In the final simulation, gravitational effects were included. This simulation includes all effects discussed in the equations (1) - (4). This case was started from the flame in Figure 6 and the results after 0.25 seconds are shown in Figure 7 for an upward-facing jet. The radial velocity field shows large structures which form in the high temperature region near the jet. These structures convect downstream and change the local concentration field. The H_2 concentration field is not affected by gravity but the O_2 concentration field has changed. Gravity has a significant effect because there is a large density difference between the burnt and unburnt gases in this flow.

Figure 8 shows a time sequence of O_2 mole fraction contours. In the first frame, a bulge appears approximately in the middle of the computational domain and convects upward in frames 2 and 3. By frame 4, the O_2 field is very distorted as the structure convects upward. In the final two frames, the structure continues to roll up as it convects out the computational domain. We estimate the frequency to be 15-20 Hz. These structures are similar to those observed experimentally in unconfined diffusion flames (20-21) and are often attributed to the effect of buoyancy.

VI. Conclusions

The Burke-Schumann analysis has formed many of our ideas about laminar diffusion flames. This simplified approach describes the global nature of a confined laminar flame but ignores many of the physical phenomena in real flames. In this paper, we described a new computer program which includes these effects. The simulations show details of the flames which cannot be observed from the analytical solution.

Introducing variable density and diffusion coefficients for a $H_2 - N_2$ fuel jet with coflowing air results in small radial velocities. In a flame where the inlet fuel and oxidizer velocities are equal, heat release accelerates the gases and produces a mixing region characterized by large-scale instabilities which are damped by viscosity. The effects of heat release and viscosity are not included in the Burke-Schumann analysis but they appear to counter-act each other.

Gravity produces a significant change in the flow field of a confined diffusion flame. The flame fluctuates in time as the buoyancy-driven structures convect upward. These low frequency fluctuations obviously are not represented in the steady state analysis of Burke and Schumann, but these fluctuations do not change the flame location significantly.

Thus, as the Burke-Schumann restrictions are eliminated, the flame characteristics change. Realistic stoichiometry, diffusion coefficients, and densities for a $H_2 - N_2$ flame with coflowing air results in a laminar flame with only small radial velocities. When heat release is included, vortices form in the reaction zone but these structures are damped by viscosity. Finally, gravity induces large-scale structures to form in the region outside of the reaction zone.

Nomenclature

Symbol	Definition
c_s	Speed of sound (cm/s)
c_v	Specific heat (erg/g-K)
D_{ik}	Binary diffusion coefficient between species i and k (cm^2/s)
E	Total energy density:$\epsilon + \frac{1}{2}\rho v^2$ (erg/cm^3)
G	Gravitational acceleration (980.67 cm/s^2)
h	Enthalpy per molecule (erg/molecule)
$\mathbf{I}$	Unit tensor (nondimensional)
k	Boltzmann constant (1.3805×10^{-16} erg/K)
n	Number density ($molecules/cm^3$)
P	Pressure ($dyne/cm^2$)
q_f	Heat of combustion (erg/molecule fuel)
Q	Energy release rate (erg/cm^3-s^1)
T	Temperature (K)
$\mathbf{v}$	Velocity (cm/s)
w	Production rate of species ($molecules/cm^3s^1$)
x	Spatial coordinate (cm)
X	Mole fraction
y	Spatial coordinate (cm)
Y	Mass fraction
Greek	
ϵ	Specific internal energy (erg/cm^3)
γ	Ratio of specific heats, c_p/c_v
κ	Thermal conductivity coefficient (erg/s-K-cm)
μ	Coefficient of shear viscosity (poise, g/cm-s)
ρ	Mass density (g/cm^3)
τ	Viscous stress tensor ($dynes/cm^2$)
Superscripts	
T	Transpose operation on a matrix
Subscripts	
m	Mixture of species
i, j, k, or l	Individual species

Acknowledgments

This work was sponsored by the Naval Research Laboratory through the Office of Naval Research. The authors would like to thank Dr. Gopal Patnaik for his many helpful suggestions and comments on this work. In addition, we thank the referees for their thorough reviews of this paper.

References

1. Burke, S.P., and Schumann, T.E.W., *Indust. Eng. Chem.* 20: 998-1004 (1928).
2. Penner, S.S., Bahadori, M.Y. and Kennedy, E.M. *Dynamics of Flames and Reactive System: AIAA Progress in Astronautics and Aeronautics*, J.R. Bowen, N. Manson, A.K. Oppenheim, and R.I. Soloukin (ed.), 95: 261-292 (1984).
3. Bahadori, M. Y., Li, C. -P., and Penner, S.S. in *Dynamics of Flames and Reactive System: AIAA Progress in Astronautics and Aeronautics* (J.R. Bowen, J.-C. Leyer, and R.I. Soloukin, Ed.), 105: 192-206 (1986).
4. Li, C.-P., Wiesenhahn, D., and Penner, S.S., *Combustion and Flame*, 65: 215-225 (1986).
5. Gosman, A.D., Pun, W.M., Runchal, A.K., Spalding, D.B., and Wolfshteim, M., *Heat and Mass Transfer in Recirculating Flows*, Academic Press, London (1969).
6. Mitchell, R.E., Sarofim, A.F., and Clomburg, L.A., *Combustion and Flame* 37: 227-244 (1980).
7. Patnaik, G., Kailasanath, K., Laskey, K.J., and Oran, E.S., *Twenty-Second Symposium (International) on Combustion*, Seattle, WA, Aug. 14-19, 1988, The Combustion Institute, Pittsburgh 1989 p. 1517.
8. Laskey, K. J., *Numerical Study of Diffusion Jet Flames*, Ph.D. dissertation, Department of Mechanical Engineering, Carnegie-Mellon University, Pittsburgh, PA (1988).
9. Patnaik, G., Boris, J.P., Guirguis, R.H., and Oran, E.S., *J. Computational Physics*, 71: 1-20 (1987).
10. Oran, E.S., Boris, J.P., *Numerical Simulation of Reactive Flow*, Elsevier, New York, 1987.
11. Casulli, V., and Greenspan, D., *Int. J. Num. Methods Fluids* , 4:1001-1012 (1984).
12. Kee, R.J., Dixon-Lewis, G., Warnatz, J., Coltrin, M.E., and Miller, J.A., *A Fortran Computer Code Package for the Evaluation of Gas-Phase Multicomponent Transport Properties*, SAND86-8246, Sandia National Laboratory (1986).
13. Jones, W.W.and Boris, J.P., *Comp. and Chem.*, 5: 139-146 (1981).
14. Kailasanath, K., Oran, E.S., and Boris, J.P., "A One-Dimensional Time-Dependent Model for Flame Initiation, Propagation, and Quenching", NRL Memorandum Report 4910, Naval Research Laboratory (1982).
15. Bird, R.B., Stewart, W.E., and Lightfoot, E.N., *Transport Phenomema*, John Wiley and Sons, N.Y. (1960).

16. Wilke, C.R., *J. Chem. Phys.*, 18: 578-579 (1950).

17. Patnaik, G., Laskey, K.J., Kailasanath, K., Oran, E.S., and Brun, T.A., "FLIC–A Detailed, Two-Dimensional Flame Model," NRL Memorandum Report 6555, Naval Research Laboratory (1989).

18. Chamberlin, D.S., and Rose, A., *First Symposium on Combustion*, Sept. 10-14, 1928, Swampscott, MA, The Combustion Institute, Pittsburgh 1965 p. 27.

19. Kimura, I., *Tenth Symposium (International) on Combustion*, Aug. 17-21, 1964, Cambridge, England, The Combustion Institute, Pittsburgh 1965 p. 1295.

20. Ballantine, A. and Bray, K.N.C., *Sixteenth Symposium (International) on Combustion*, Aug. 15-20, 1976, Cambridge, MA, The Combustion Institute, Pittsburgh 1977 p. 777.

21. Chen, L.-D., Seaba, J.P., Roquemore, W.M., and Goss, L.P., *Twenty-Second Symposium (International) on Combustion*, Aug. 14-19, Seattle, WA, 1988, The Combustion Institute, Pittsburgh 1989 p. 677.

Figure Captions

1. Geometry used for Burke-Schumann calculations showing flame location for typical underventilated or overventilated flame.

2a. Contours of fuel concentration normalized by inlet fuel concentration for overventilated Burke-Schumann at times (a) 0.0 (b) 0.1 sec (c) 0.4 sec (d) 0.7 (e) 1.0 seconds.

2b. Contours of oxidizer concentration normalized by inlet oxidizer concentration for overventilated Burke-Schumann at times (a) 0.0 (b) 0.1 sec (c) 0.4 sec (d) 0.7 (e) 1.0 seconds.

3. Comparison of analytical and computed solutions for flame location for three different Burke-Schumann flames. (a) Pure fuel reacting with 1 part oxidizer + 1 part inert. (b) One part fuel + 3 parts inert reacting with pure oxidizer. (c) Pure fuel reacting with 1 part oxidizer + 4 parts inert.

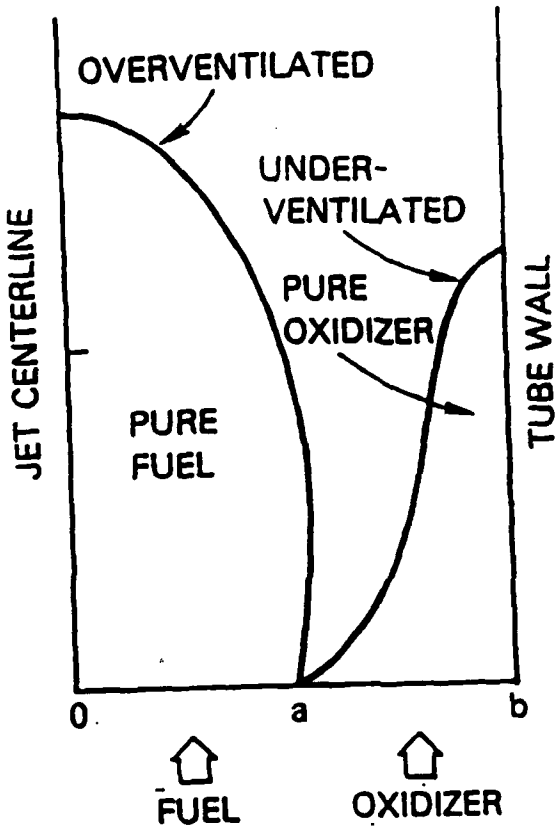
4. Contours of (a) radial velocity (b) axial velocity (c) mole fraction H_2 (d) mole fraction O_2 for $H_2 - N_2$ diffusion flame without heat release. Dimensions are in cm, velocities are in cm/s. Dotted lines indicate negative values.

5. Contours of (a) radial velocity (b) axial velocity (c) mole fraction H_2 (d) mole fraction O_2 (e) temperature for $H_2 - N_2$ diffusion flame with heat release. Dimensions are in cm, velocities are in cm/s, temperature is in K. Dotted lines indicate negative values.

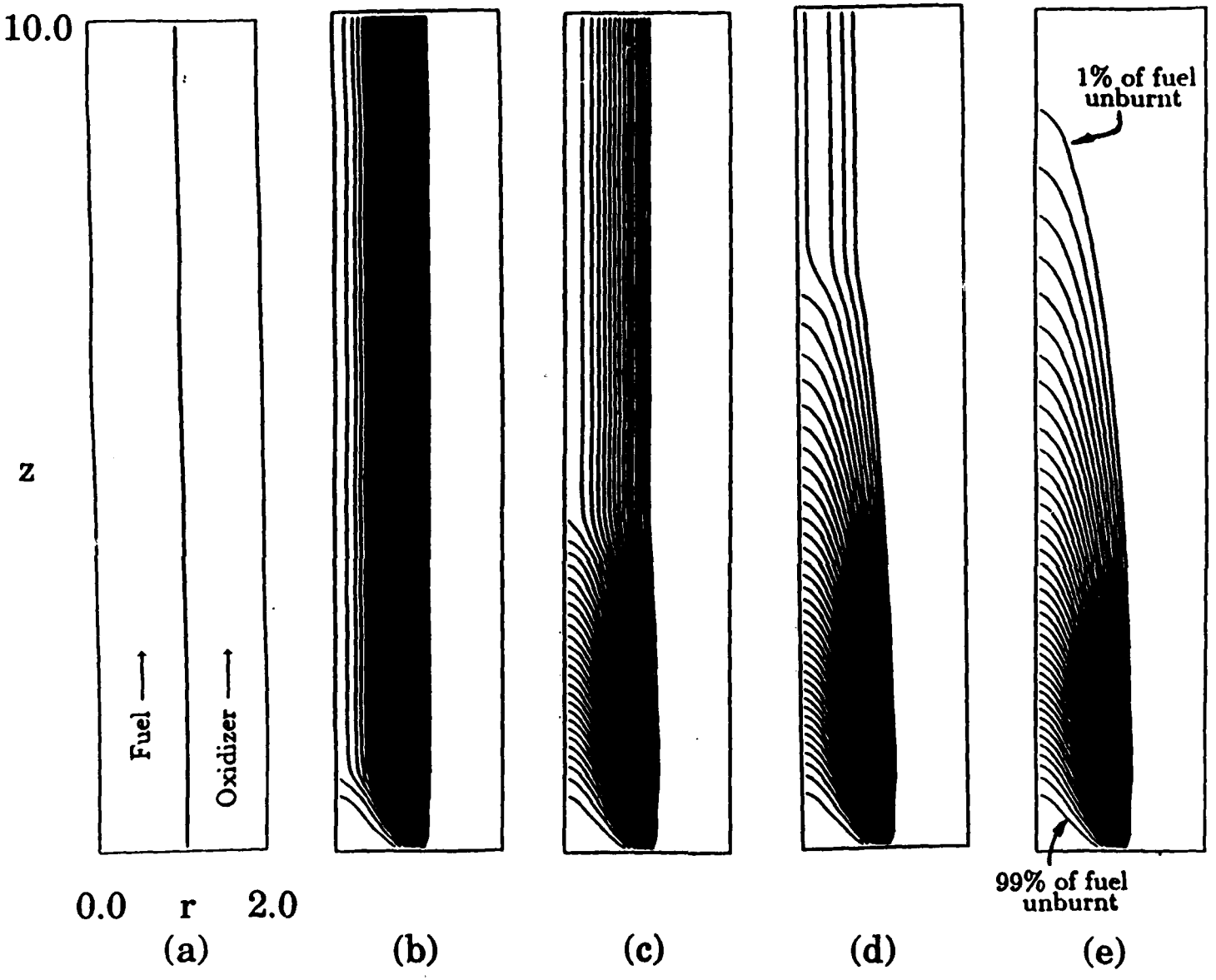
6. Contours of (a) radial velocity (b) axial velocity (c) mole fraction H_2 (d) mole fraction O_2 (e) temperature for $H_2 - N_2$ diffusion flame with heat release and viscosity. Dimensions are in cm, velocities are in cm/s, temperature in in K. Dotted lines indicate negative values.

7. Contours of (a) radial velocity (b) axial velocity (c) mole fraction H_2 (d) mole fraction O_2 (e) temperature for $H_2 - N_2$ diffusion flame with heat release, viscosity, and gravity for an upward-facing jet. Dimensions are in cm, velocities are in cm/s, temperature is in K. Dotted lines indicate negative values.

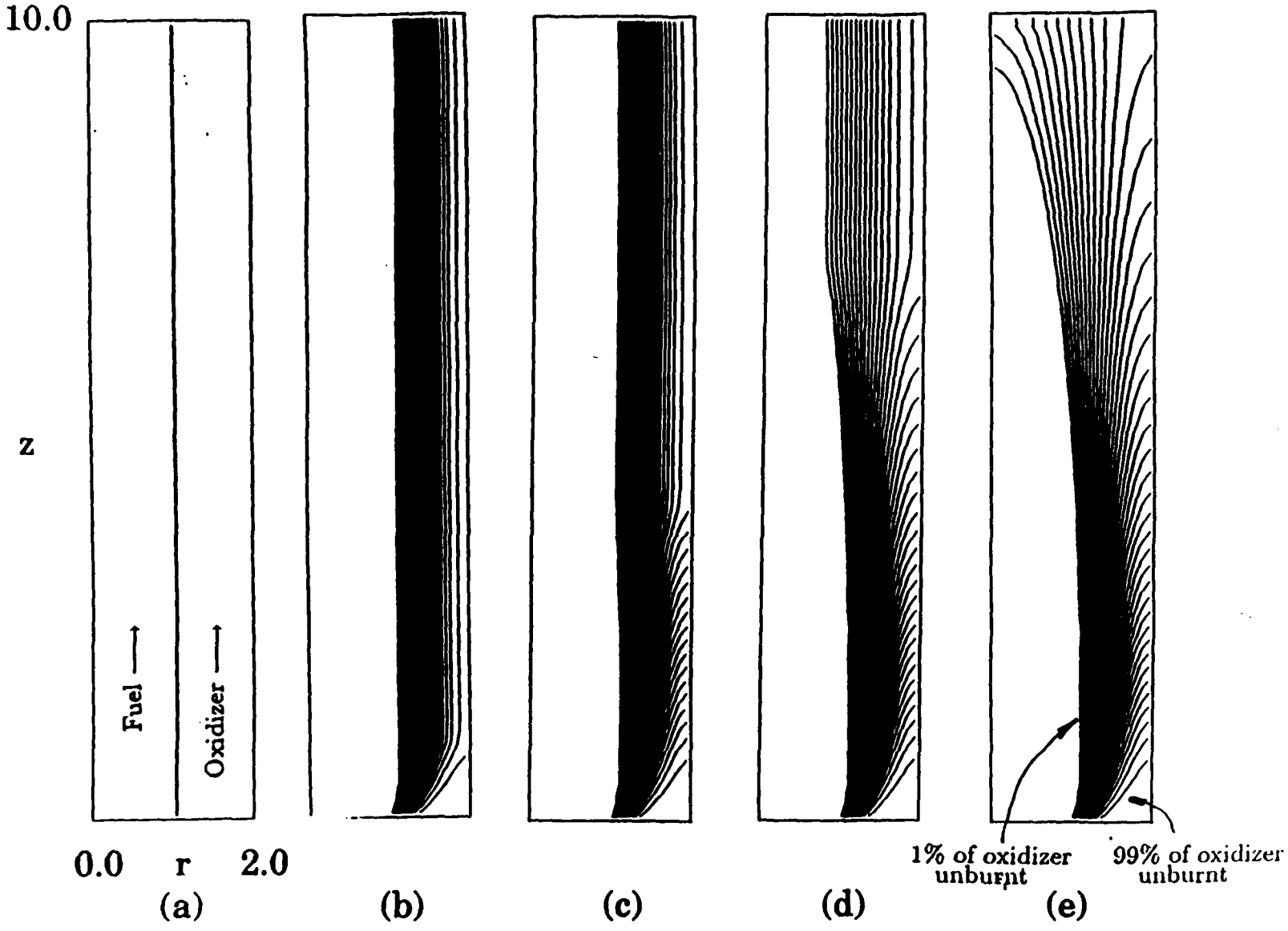
8. Time sequence for contours of O_2 mole fraction showing the formation of large structure. The time interval between each frame is 10 ms.



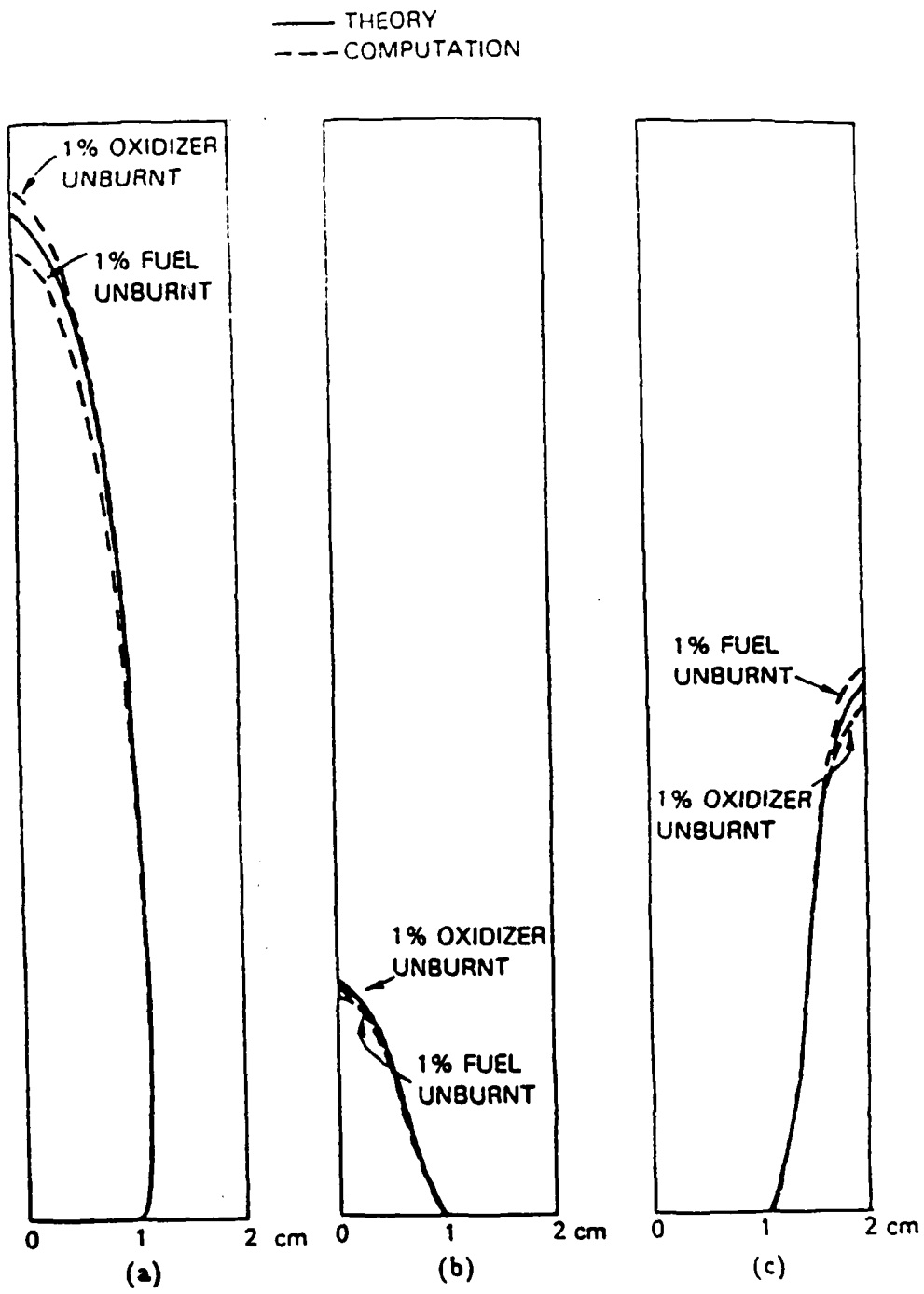
1. Geometry used for Burke-Schumann calculations showing flame location for typical underventilated or overventilated flame.



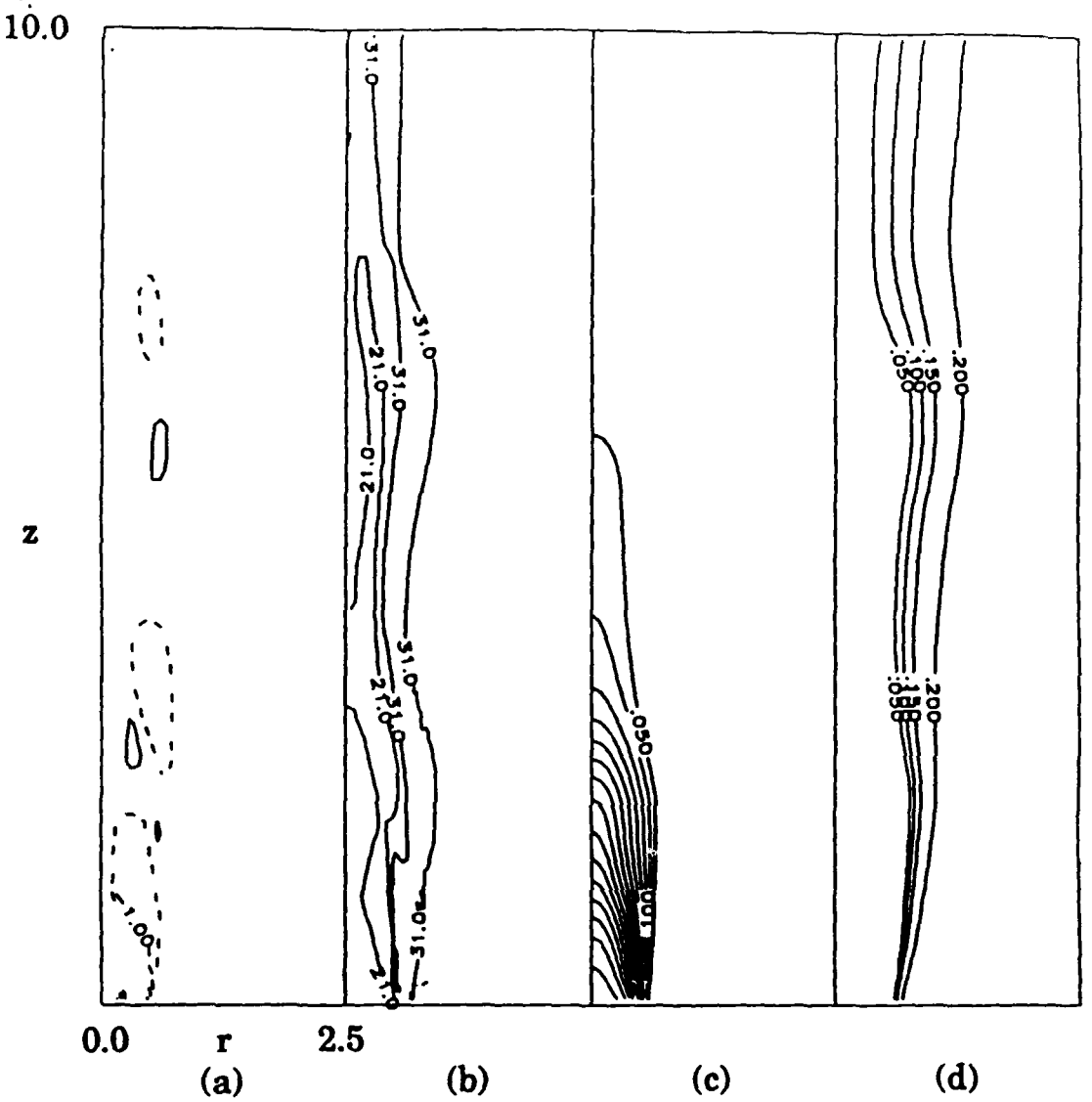
2a. Contours of fuel concentration normalized by inlet fuel concentration for overventilated Burke-Schumann at times (a) 0.0 (b) 0.1 sec (c) 0.4 sec (d) 0.7 (e) 1.0 seconds.



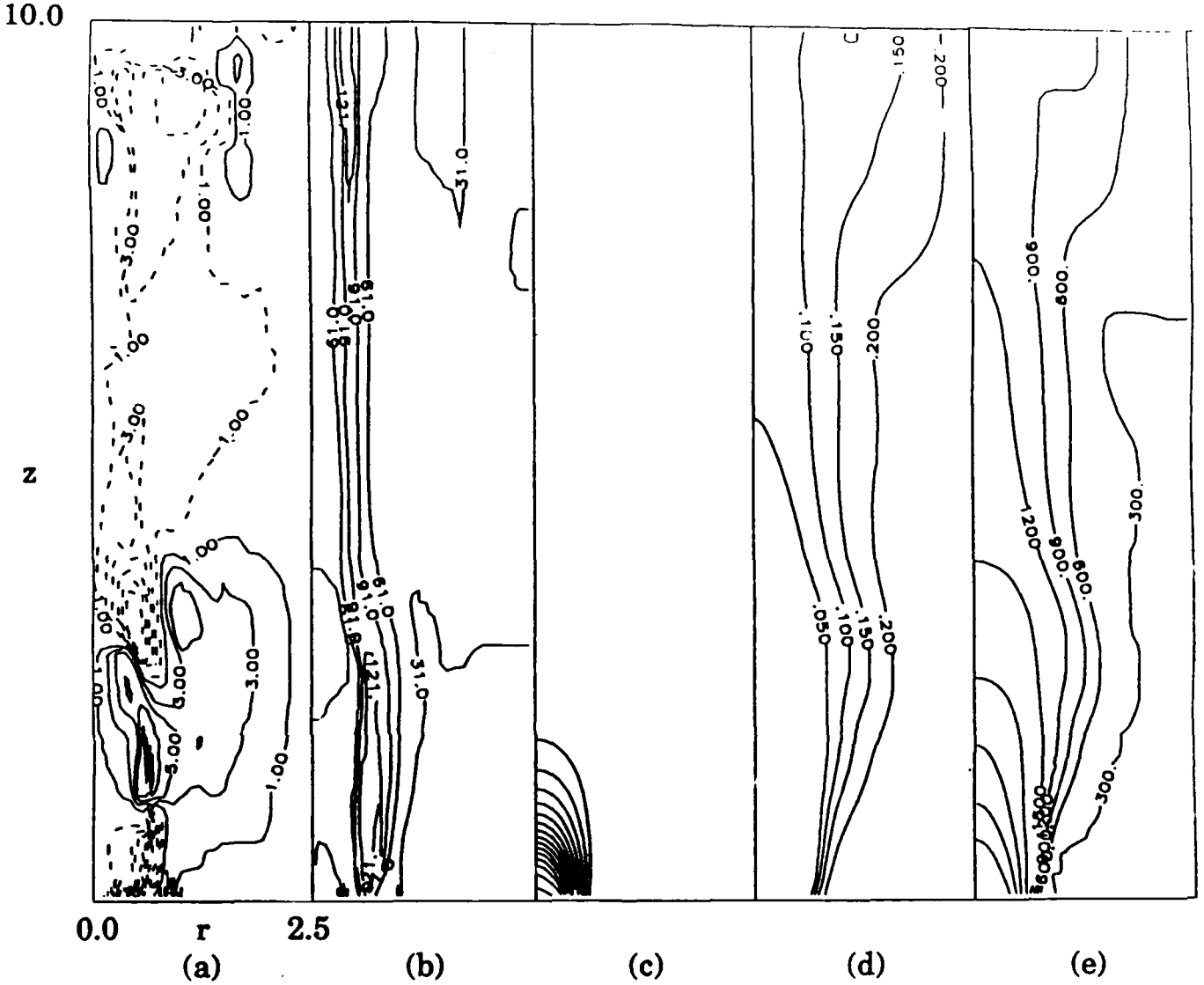
2b. Contours of oxidizer concentration normalized by inlet oxidizer concentration for over-ventilated Burke-Schumann at times (a) 0.0 (b) 0.1 sec (c) 0.4 sec (d) 0.7 (e) 1.0 seconds.



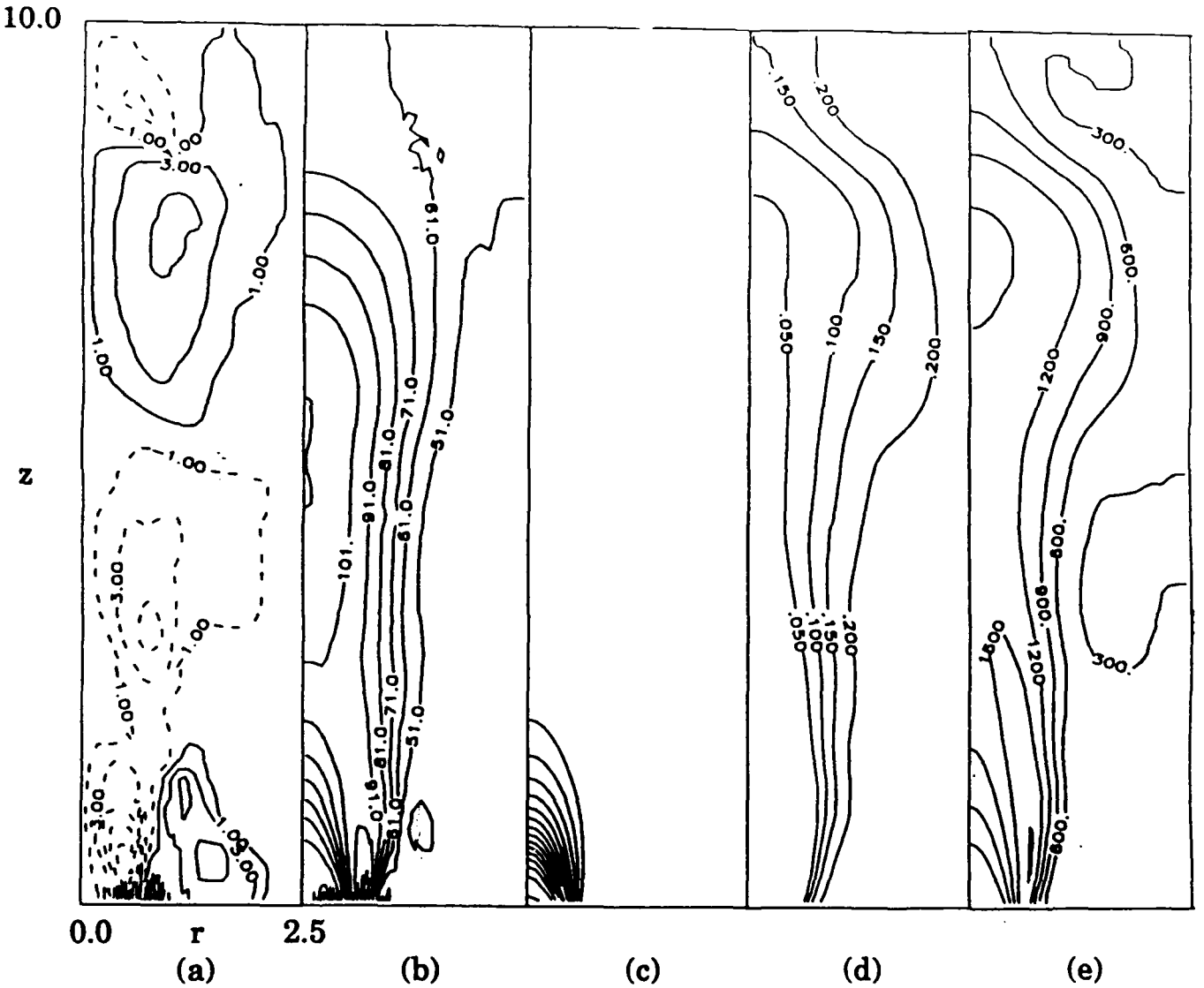
3. Comparison of analytical and computed solutions for flame location for three different Burke-Schumann flames. (a) Pure fuel reacting with 1 part oxidizer + 1 part inert. (b) One part fuel + 3 parts inert reacting with pure oxidizer. (c) Pure fuel reacting with 1 part oxidizer + 4 parts inert.



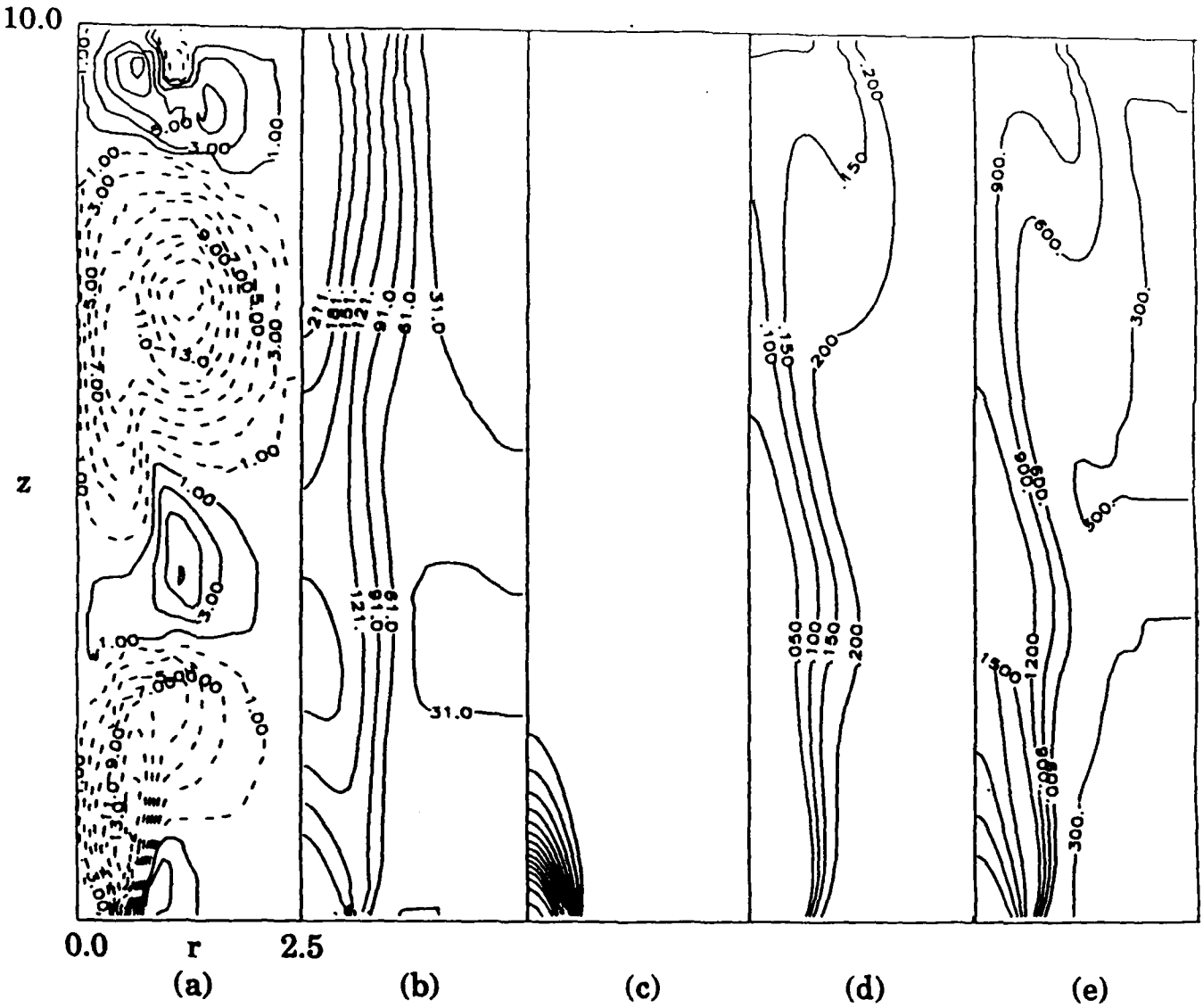
4. **Contours of (a) radial velocity (b) axial velocity (c) mole fraction H_2 (d) mole fraction O_2 for $H_2 - N_2$ diffusion flame without heat release. Dimensions are in cm, velocities are in cm/s.**



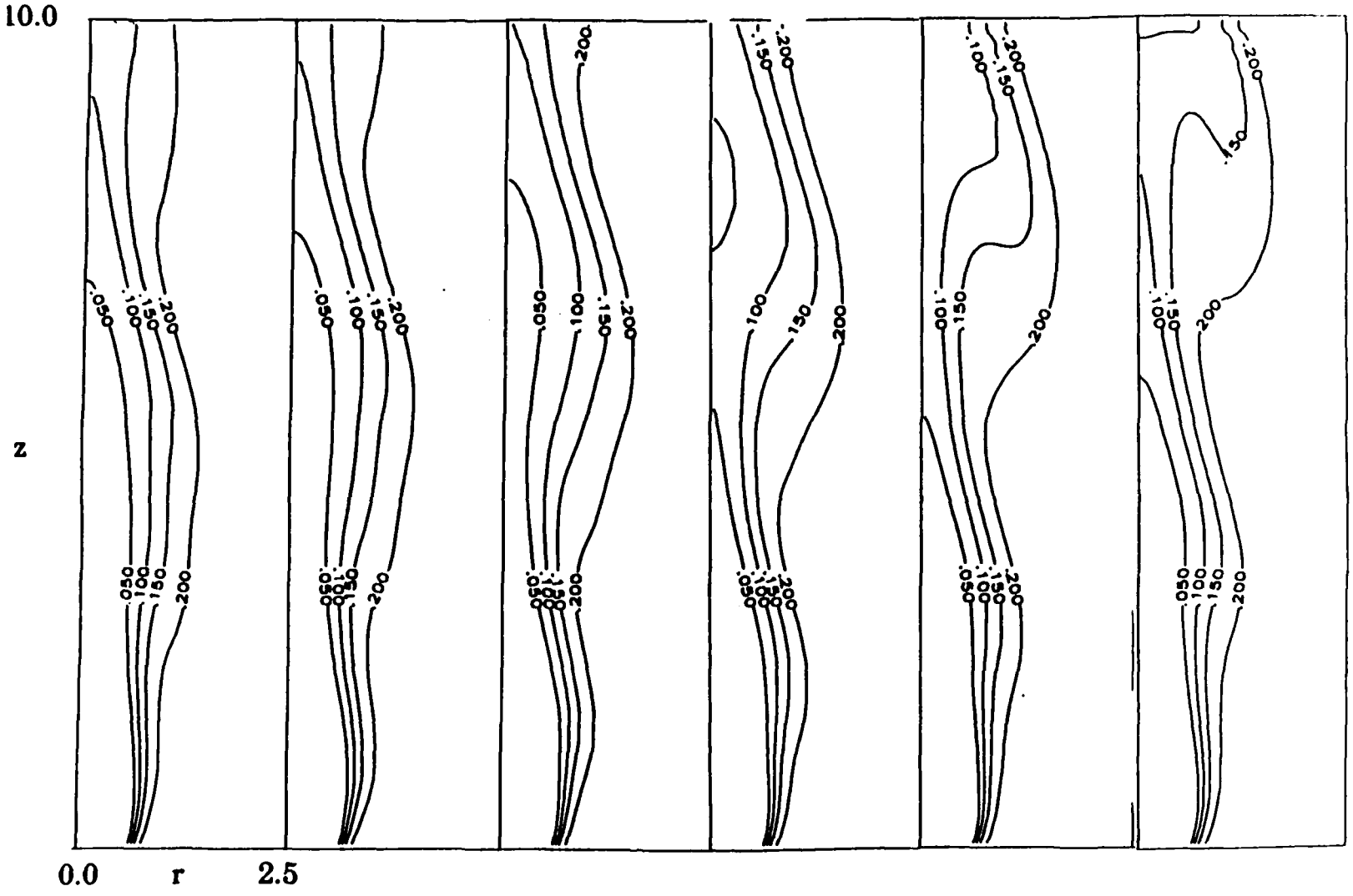
5. Contours of (a) radial velocity (b) axial velocity (c) mole fraction H_2 (d) mole fraction O_2 (e) temperature for $H_2 - N_2$ diffusion flame with heat release. Dimensions are in cm, velocities are in cm/s, temperature is in K.



6. Contours of (a) radial velocity (b) axial velocity (c) mole fraction H_2 (d) mole fraction O_2 (e) temperature for $H_2 - N_2$ diffusion flame with heat release and viscosity. Dimensions are in cm, velocities are in cm/s, temperature in in K.



7. **Contours of (a) radial velocity (b) axial velocity (c) mole fraction H_2 (d) mole fraction O_2 (e) temperature for $H_2 - N_2$ diffusion flame with heat release, viscosity, and gravity. Dimensions are in cm, velocities are in cm/s, temperature is in K.**



8. Time sequence for contours of O_2 mole fraction showing the formation of large structure. The time interval between each frame is 10 ms.

APPENDIX Q.

Effect of Heat Release and Gravity on an Unsteady Diffusion Flame
(accepted for publication in the Proc. of the 23rd Intern. Symp. on Comb.)

accepted for publication in the Proceedings of the 23rd Symposium (International) on Combustion

EFFECTS OF HEAT RELEASE AND GRAVITY ON AN UNSTEADY DIFFUSION FLAME

J.L. Ellzey
Berkeley Research Associates
Springfield, VA

E.S. Oran
Laboratory for Computational Physics and Fluid Dynamics
Naval Research Laboratory, Code 4404
Washington, D.C. 20375

Subject Headings:
(17) Instability
(10) Flames
(19) Modeling and Simulation

Corresponding Author:
Dr. Janet L. Ellzey
Code 4410
Naval Research Laboratory
Washington, DC 20375
phone: 202-767-3615

EFFECTS OF HEAT RELEASE AND GRAVITY ON AN UNSTEADY DIFFUSION FLAME

J.L. Ellzey* and E.S. Oran
Laboratory for Computational Physics and Fluid Dynamics
Naval Research Laboratory
Washington, D.C. 20375

ABSTRACT

This paper presents time-dependent axisymmetric numerical simulations of an unsteady diffusion flame formed between a $H_2 - N_2$ jet and a coflowing air stream. The computations include the effects of convection, molecular diffusion, thermal conduction, viscosity, gravitational forces, and chemical reactions with energy release. Previous work has shown that viscous effects are important in these flames and, therfore, all of the viscous terms in the compressible Navier-Stokes equations are included. In addition, the resolution is increased so that the large, vortical structures in the coflowing gas are resolved and the boundary conditions are improved so that the velocity field near the jet is more realistic. Computations with and without chemical reactions and heat release, and with and without gravity, are compared. Gravitational effects are insignificant in the nonreacting jet but in the reacting jet gravity produces the relatively low-freqency instabilities typically associated with flame flicker. Kelvin-Helmholtz instabilities develop in the region between the high-velocity and low-velocity fluid when there are no chemical reactions, but heat release dampens these instabilities to produce a mixing region which is almost steady in time.

* Berkeley Research Associates, Springfield, VA

INTRODUCTION

Experiments on laminar diffusion flames have shown that gravity affects the flame length and width as well as its extinction characteristics (1-4). These studies have been conducted in drop towers and have focused on fuel jets with very low velocities of less than 50 cm/s. Although these experiments have increased our basic understanding of laminar diffusion flames by emphasizing the importance of bouyancy, it is not clear how to apply these results to higher-velocity flames which are unsteady or fluctuating. Studying higher-velocity fuel jets from larger nozzles is more difficult experimentally because the flames can be quite long and the instabilities may not have time to evolve during a single experiment. Through numerical simulations, we can examine an unsteady flame with and without gravity in the kind of detail that is not practical in an experiment.

Two types of instabilities are observed in low-speed diffusion flames (5,6). The high-frequency structures grow from Kelvin-Helmholtz instabilities at the interface between the high-velocity and low-velocity fluid and typically have frequencies of a few hundred Hertz. The low-frequency structures form in the region outside the flame zone with typical frequencies of 10–20 Hertz.

This paper examines the effect of heat release and gravity on the formation and evolution of these two types of instabilities by presenting a series of time-dependent, two-dimensional simulations of an axisymmetric H_2–N_2 jet in a coflowing air stream. The calculations include convection, thermal conduction, molecular diffusion, viscosity, chemical reactions with energy release, and gravitational forces. This model is based on the one developed by Laskey (7), which includes a new algorithm for convective transport developed by Patnaik et al. (8). Previously, Laskey (9) presented computations of diffusion flames of the type presented here and Patnaik et al. (10) used a similar model to study the stability properties of very low-speed premixed flames. All of these efforts have tested the various parts of the model and have given credibility to its overall validity.

These computations are different from previous ones reported by Laskey et al. (9) and Ellzey et al. (11) for several reasons. Greater resolution and improved boundary conditions now allow correct zero-gravity computations. The energy-release model now properly limits the final temperatures allowed and no longer produces a strong recirculation zone at the jet exit. Viscosity has been shown to be very important in diffusion flames at these velocities (11) and is, therefore, included in all of the calculations presented in this paper.

NUMERICAL METHOD

The numerical model consists of separate algorithms for the various processes, and these algorithms are coupled by timestep splitting methods. Table I is an outline of one computatonal timestep. Given a set of initial values for the basic variables, an approriate computational timestep is estimated based on accuracy and stability criteria. Then the effects of thermal conduction are evaluated using a two-dimensional explicit finite-difference model (7). Thermal conductivities, for the individual species were calculated from kinetic theory over the temperature range 300 to 3300 K, these values were fit to a third-order polynomial, and then are used to calculate the mixture thermal conductivity (13). Molecular diffusion is included using an explicit finite-difference formulation. First, the diffusion velocities are calculated according to Fick's law and then corrected (13) to satisfy the requirement that the sum of the diffusion fluxes is zero. Binary diffusion coefficients, calculated from kinetic theory (14), are used to compute the diffusion coefficients for a particular species in a mixture (13). The viscosity coefficients μ_k, calculated from kinetic theory over the temperature range 300 to 3000K and fit to a third order polynomial, were used to compute the mixture viscosity (15). The model for chemical reactions and heat release is an extension of the Parametric Diffusion Reaction Model (7,12), which is designed to replace the integration of the full, detailed set of ordinary differential equations representing the chemical kinetics. A single, global reaction is used but the reaction is not instantaneous. Instead, the finite reaction rate is

Table I. One Timestep in the Diffusion-Flame Model

Given Initial Variables
1. Determine Timestep
2. Thermal Conduction
Integrate from t to $t + \Delta t$:
Calculate $\Delta\epsilon_1$. Do not update any variables. Subcycle as necessary.
3. Ordinary Diffusion
Integrate from t to $t + \Delta t$:
Only update $\{n_i(x)\}$. Calculate $\Delta\epsilon_2$. Subcycle as necessary.
4. Viscosity
Integrate from t to $t + \Delta t$:
Only update $\rho\vec{v}$. Calculate $\Delta\epsilon_3$.
5. Chemical Reactions
Integrate from t to $t + \Delta t$:
Only update $\{n_i(x)\}$. Calculate $\Delta\epsilon_4$.
6. Convective Transport
Integrate from t to $t + \Delta t$:
x direction transport, then update $\rho, \rho\vec{v}, E, n_i$.
y direction transport, then u update $\rho, \rho\vec{v}, E, n_i$.
Implicit correction, then update p, ϵ, and E.
7. Increment Time and go to 1.

determined such that the maximum temperature in a one-dimensional transient diffusion flame is the adiabatic flame temperature for a stoichiometric mixture of the fuel and oxidizer. The transport of density, momentum, energy, and individual species density is accomplished through the high-order implicit method, BIC-FCT (8). This involves an explicit step, based on the standard FCT algorithm (16), and then an implicit correction.

The general timestep splitting approach for coupling the various physical processes was developed for slow-flow implicit calculations. In these computations, the change in internal energy resulting from each individual process is not incorporated into the solution as soon as it is computed, but instead is accumulated, as indicated by the $\{\epsilon_i\}$ in Table 1. The entire change in internal energy is then added to the

energy equation in the fluid convection step 6. The coupling technique has been described by Oran and Boris (16), and a modification by Patnaik et al. (17) has been shown to allow for a greater addition of energy per timestep while maintaining numerical stability.

In essence, the model solves the time-dependent two-dimensional conservation equations for mass density, ρ, momentum, $\rho\mathbf{v}$, and total energy, E and these are coupled to models for chemical reactions among the species $\{n_i\}$ with subsequent heat release, molecular diffusion, thermal conduction, viscosity, and gravitatonal forces. Additional equations include the perfect gas equation of state and a relation between the internal energy and the pressure. The specific set of equations and more detailed discussions of the numerical methods are given in References (7) and (12).

The computations described in this paper, using the enlarged computational grid and including all of the physical processes, require 0.7 s/computational timestep on a Cray YMP. This means that a typical calculation, about 50,000 timesteps, requires about 10 hours of computer time.

APPLICATION TO UNSTEADY DIFFUSION FLAMES

The computational grid for the region near the jet and the initial conditions are shown in Figure 1. The full domain is 10 cm × 172 cm and consists of 128 × 224 cells. Cells of approximately 0.02 cm are concentrated around the jet exit. Beginning at $r = 1$ cm, the size of each cell is increased by 0.03% over the size of its neighboring cell for all simulations. The cells in the axial direction for all simulations are stretched by 0.03% starting at $z = 1$ cm. A fuel mixture consisting of 78% H_2 and 22% N_2 flows through a jet of radius 0.5 cm at 10 m/s at the lower boundary. Air flows through the outer annular region between $r = 0.5$ and $r = 10.0$ at 30 cm/s. The outer boundary at $r = 10.0$ is a free-slip wall. The inner boundary at $r = 0.0$ is the jet centerline. An outflow boundary is specified at $z = 172$ cm where the pressure is adjusted to atmospheric.

RESULTS

Nonreacting Jet

Figure 2 shows the instantaneous contours of axial and radial velocity and mole fraction late in the simulation of zero-gravity nonreacting jet. Kelvin-Helmholtz instabilities occur near the jet exit leading to vortical structures that then convect downstream. These structures, which transport fuel and oxidizer radially and broaden the mixing zone, weaken substantially in the first ten jet diameters. Small radial velocities, not evident in the contours, still exist at this point. Figure 3 shows the mean and rms velocity for the nonreacting jet at three axial locations. At $z = 0.5$ cm, the mixing region is narrow with only small fluctuations of a few cm/s. At $z = 1.0$ cm, the instabilities result in large fluctuations across the entire jet core. By $z = 10$ cm, there are small fluctuations across the entire jet region. The results for the nonreacting jet with gravity are not distinguishable from those for the same jet in zero gravity, and so are not shown here.

Reacting Jet

Instantaneous contours late in the calculation of the reacting jet in zero gravity, Figure 4, show that the volumetric expansion and the change in temperature have a significant effect on the flow. The radial velocities arise from the expansion at the flame front but are relatively uniform. The axial velocity and concentration fields are steady in time. Figure 5, the mean and rms velocity for this case, show that the mixing region is wider due to the expansion. Fluctuations are insignificant and not visible on the plot.

Figure 6 shows that gravity changes the flow significantly. Instabilities form outside the reaction zone in the region with large temperature and density gradients. The maximum radial velocity is approximately 30 cm/s and occurs at the center of the structure. The concentration and temperature fields are distorted as these instabilities convect downstream. The flame front lies at the fuel-oxidizer interface in the region of maximum temperature and fluctuates in time. Figure 7 shows a

time sequence of the H_2O mole-fraction contours. In the first frame, a bulge is developing on the outside of the H_2O contours. In subsequent frames, it rolls up and moves downstream. In the final frame, it is moving out of the domain shown as the next instability forms below it. These outer, slower-moving vortical structures occur at approximately 15 Hz.

DISCUSSION AND CONCLUSIONS

Comparisons of the four computations of the 10 m/s $H_2 - N_2$ jet into the 30 m/s coflowing air background shows that gravity and heat release interact substantially to change the flow. Without chemical reactions and subsequent heat release, gravity does not noticeably change the velocity or concentation fields. Even though there are significant density gradients between the $H_2 - N_2$ fuel jet and the co-flowing air, these gradients occur in a region of relatively high velocity where momentum effects dominate.

In the reacting jet, there are significant density gradients in the coflow region where the velocity is low. These gradients are due to the conduction of heat away from the reaction zone. In this region, the bouyant forces dominate and large instabilities form. These have been observed in experiments for many years (6, 18–20) and are considered to be responsible for flame flicker.

Volumetric expansion and the effects of changing temperature stabilize the mixing region of the reacting jet. The increase in viscosity with temperature accounts for part of the stabilization but analytical results show that inviscid instabilities are also damped by heat release (21). Preliminary computations with constant viscosity indicate that the stabilization effect due to the change in viscosity with temperature may be insignificant compared to the effect of heat release. Previous calculations (11) show that even without including viscosity, heat release reduces the strength of the Kelvin-Helmholtz instability.

Future computations are proceeding in several different directions. First, we are considering the downward-propagating diffusion flame and how this differs from

upward and zero-gravity flames. Second, we are reducing the coflow velocity so that the computations have the same parameters as recent experiments at the Air Force Wright Aeronautical Laboratory. At that point, detailed comparisons will be made between the computations and experimental results. We are continuing to develop the energy-release model so that the energy release as a function of temperature is better represented. Finally, we have been investigating new types of computers that might allow full-chemistry or three-dimensional computations of such flames.

ACKNOWLEDGMENTS

This work was sponsored by the Naval Research Laboratory through the Office of Naval Research. We thank Dr. W.M. Roquemore from the Air Force Wright Aeronautical Laboratory for his support and suggestions. This work is based on an earlier computer code, Axisymmetric, Low-speed Jet Flame (ALJF) code, written by Dr. Kenneth Laskey. In addition, the authors would like to acknowledge the help and advice of Drs. Gopal Patnaik and Kenneth Laskey. The computations were performed at the NAS computer facility at NASA Ames Research Center.

REFERENCES

1. Cochrane, T.H. and Mascia, W.J., "Effects of Gravity on Laminar Gas Jet Diffusion Flames," NASA TN D-5872, June, 1970.
2. Cochrane, T.H. and Mascia, W.J., Proceedings of *Thirteenth Symposium (International) on Combustion*, pp. 821-829, The Combustion Institute, Pittsburgh, PA, 1970.
3. Haggard, J.B. and Cochrane, T.H., *Combust. Sci. Tech.*, 5, 291–298, 1972.
4. Haggard, J.B. and Cochrane, T.H., "Hydrogen and Hydrocarbon Diffusion Flames in a Weightless Environment," NASA TN D-7165, February, 1972.
5. Yule, A.J., Chigier, N.A., Ralph, S., Boulderstone, R., and Ventura, J., *AIAA J.*, 19, 752–760, 1981.
6. Chen, L.D., Seaba, J.P., Roquemore, W.M., and Goss, L.P., *Twenty-Second Symposium (International) on Combustion*, 677–684, The Combustion Insitute, Pittsburgh, PA, 1988.
7. Laskey, K. J., *Numerical Study of Diffusion and Premixed Jet Flames*, Ph.D. dissertation, Department of Mechanical Engineering, Carnegie-Melon University, Pittsburgh, PA, 1988.
8. Patnaik, G., Boris, J.P., Guirguis, R.H., and Oran, E.S., *J. Comput. Phys.*, 71, 1–20, 1987.
9. Laskey, K.J., Ellzey, J.L., and Oran, E.S., "A Numerical Study of an Unsteady Diffusion Flame," AIAA Paper 89-0572, AIAA, Washington, DC, 1989.
10. Patnaik, G., Kailasanath, K., Laskey, K.J., and Oran, E.S., *Twenty-Second Symposium (International) on Combustion*, 1517–1526, The Combustion Institute, Pittsburgh, PA, 1988.
11. Ellzey, J.L., Laskey, K.J., and Oran, E.S., "Dynamics of an Unsteady Diffusion Flame: Effects of Heat Release and Viscosity," submitted to *AIAA Progress in Astronautics and Aeronautics*, 1989.
12. Ellzey, , J.L., Laskey, K.J., and Oran, E.S., "A Study of Confined Diffusion

Flames," submitted to *Combust. Flame*, 1989.

13. Kee, R.J., Dixon-Lewis, G., Warnatz, J., Coltrin, M.E., and Miller, J.A., *A Fortran Computer Code Package for the Evaluation of Gas-Phase Multicomponent Transport Properties*, SAND86-8246, Sandia National Laboratory, 1986.
14. Kailasanath, K., Oran, E.S., and Boris, J.P., *A One-Dimensional Time-Dependent Model for Flame Initiation, Propagation, and Quenching*, NRL Memorandum Report 4910, Naval Research Laboratory, Washington, DC, 1982.
15. Wilke, C.R., *J. Chem. Phys.*, 18, 578–579, 1950.
16. Oran, E.S., Boris, J.P., *Numerical Simulation of Reactive Flow*, Elsevier, New York, 1987.
17. Patnaik, G., Laskey, K.J., Kailasanath, K., Oran, E.S., and Brun, T.A., *FLIC – A Detailed, Two-Dimensional Flame Model*, NRL Memorandum Report, Naval Research Laboratory, Washington, DC, 1989.
18. Chamberlin, D.S., and Rose, A., *First Symposium on Combustion*, p. 27, The Combustion Institute, Pittsburgh PA, 1965.
19. Kimura, I., *Tenth Symposium (International) on Combustion*, p. 1295, The Combustion Institute, Pittsburgh, PA, 1965.
20. Ballantine, A. and Bray, K.N.C., *Sixteenth Symposium (International) on Combustion*, p. 777, The Combustion Institute, Pittsburgh, PA, 1977.
21. Mahalingam, S., Cantwell, B., and Ferziger, J., "Effects of Heat Release on the Structure and Stability of a Coflowing, Chemically Reacting Jet," AIAA Paper 89-0661, AIAA, Washington, DC, 1989.

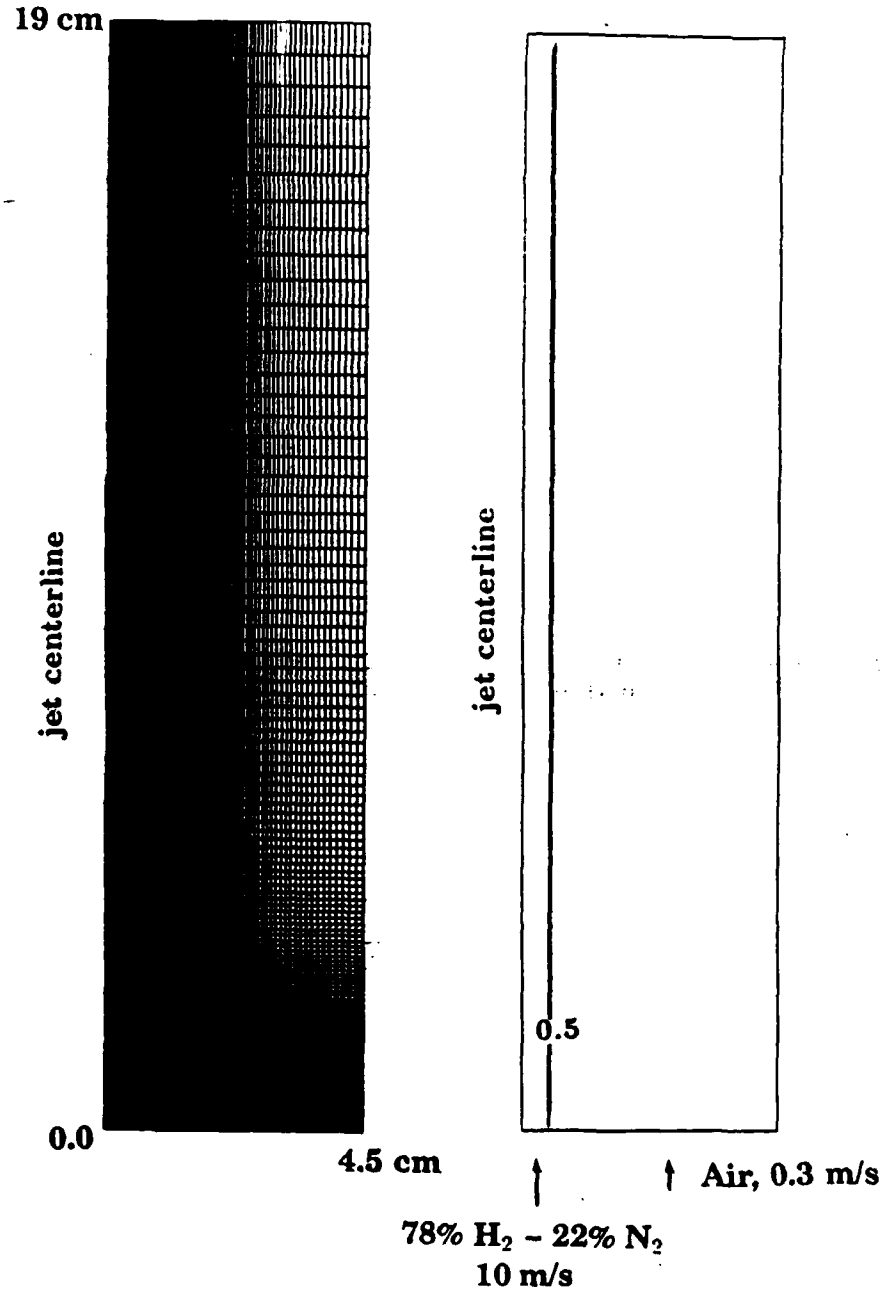


Figure 1. Computational domain and initial conditions for the computations of a $H_2 - N_2$ jet into coflowing air. Note that the figures only show the part of the full domain with the high resolution.

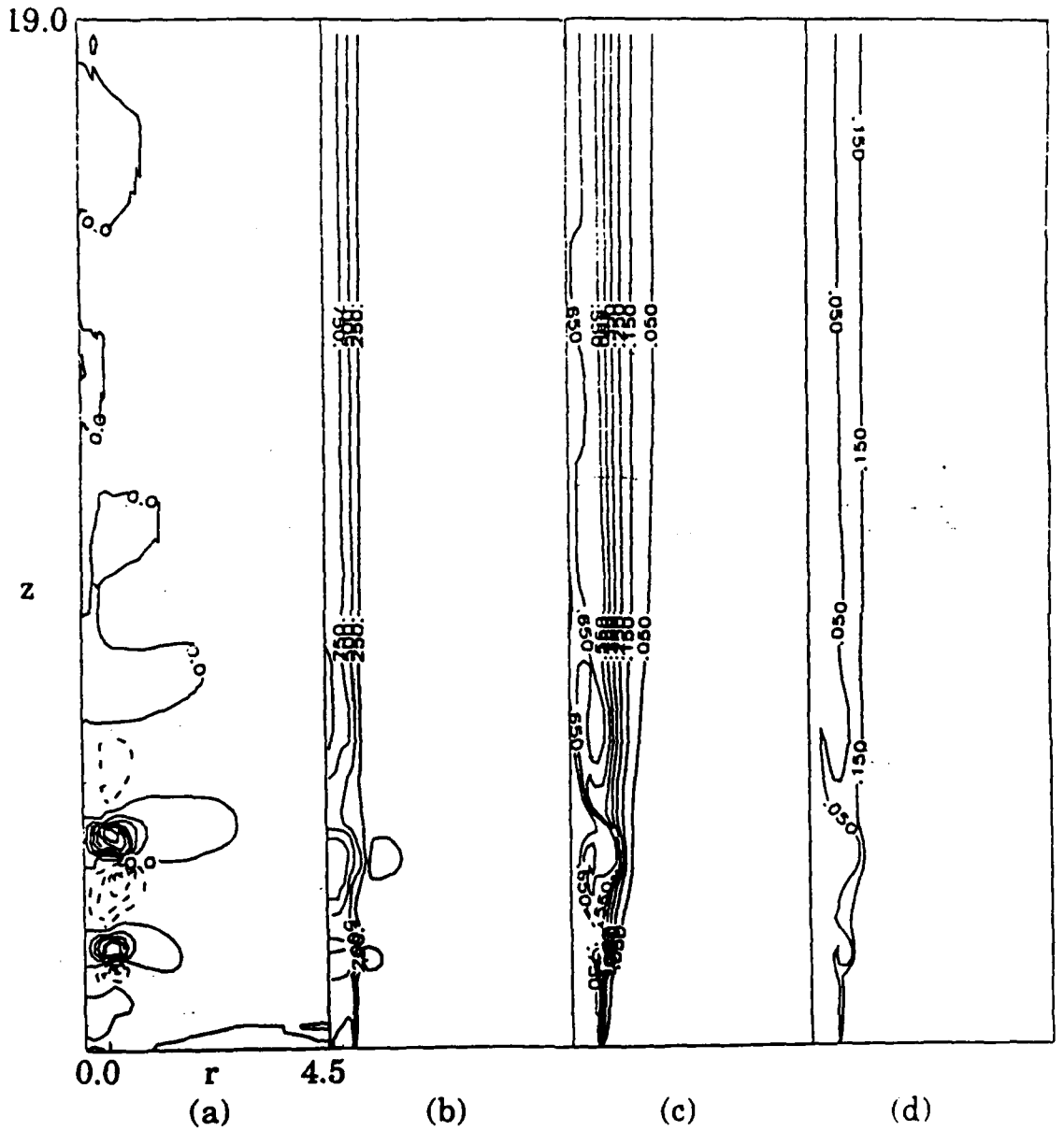


Figure 2. Contours of (a) radial velocity, (b) axial velocity, (c) mole fraction H_2, (d) mole fraction O_2 for a nonreacting, zero-gravity jet of $H_2 - N_2$ into coflowing air. Dimensions are in cm, velocities are in cm/s.

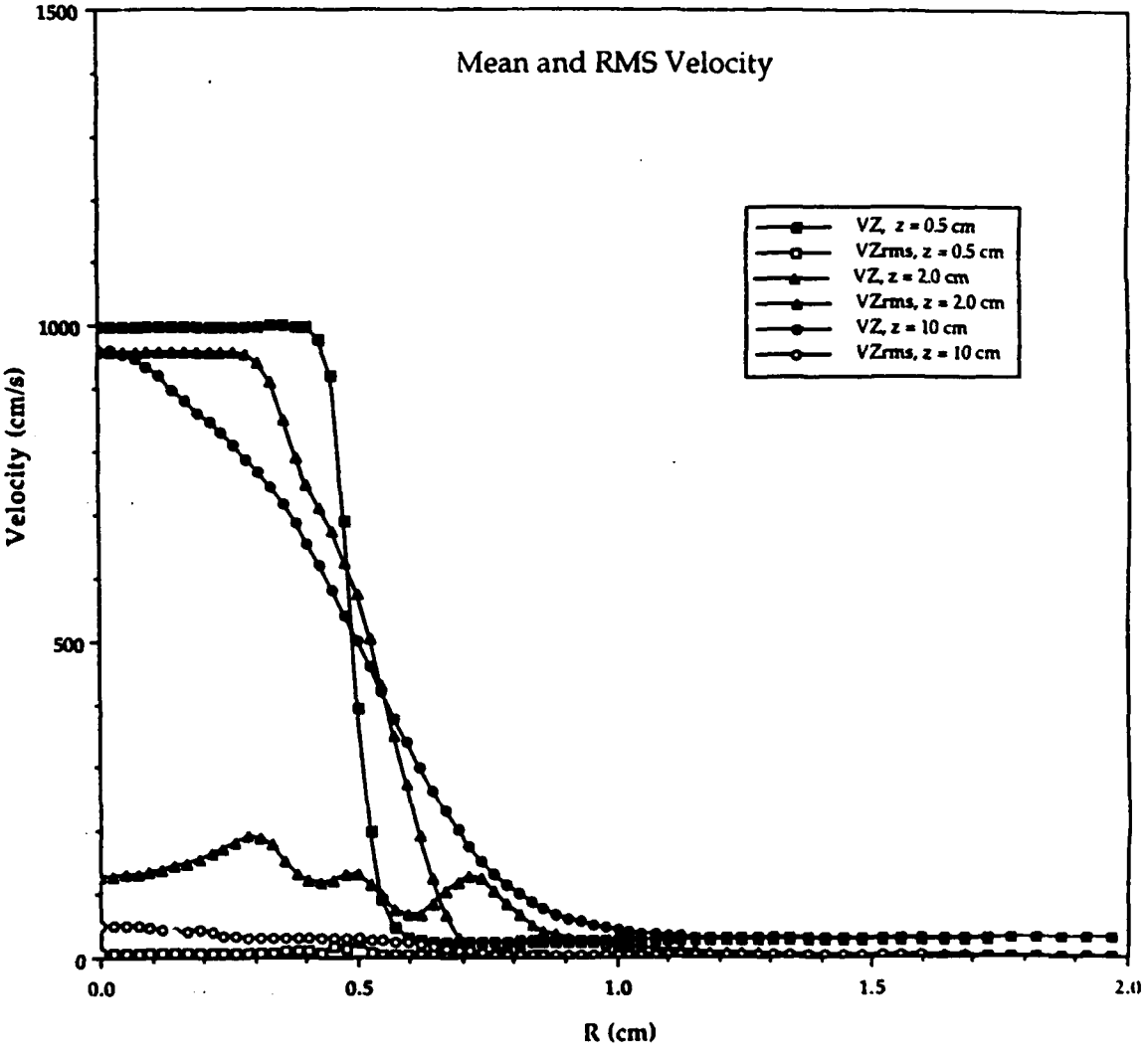


Figure 3. Mean and rms velocity for the zero-gravity nonreacting jet at three axial locations.

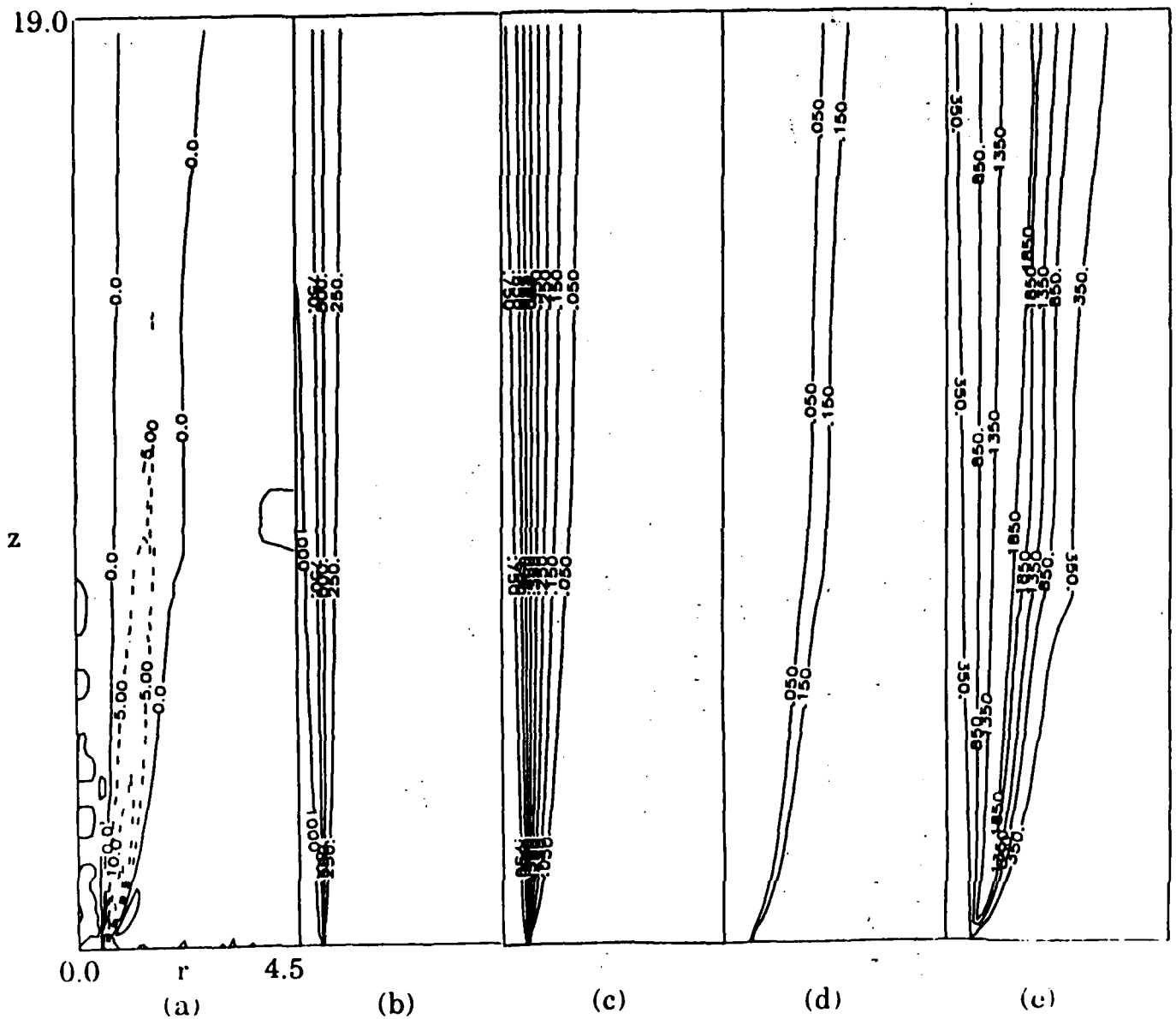


Figure 4. Contours of (a) radial velocity, (b) axial velocity, (c) mole fraction H_2, (d) mole fraction O_2, (d) temperature for a zero gravity diffusion flame formed between a $H_2 - N_2$ jet and coflowing air. Dimensions are in cm, velocities are in cm/s, temperature is in K.

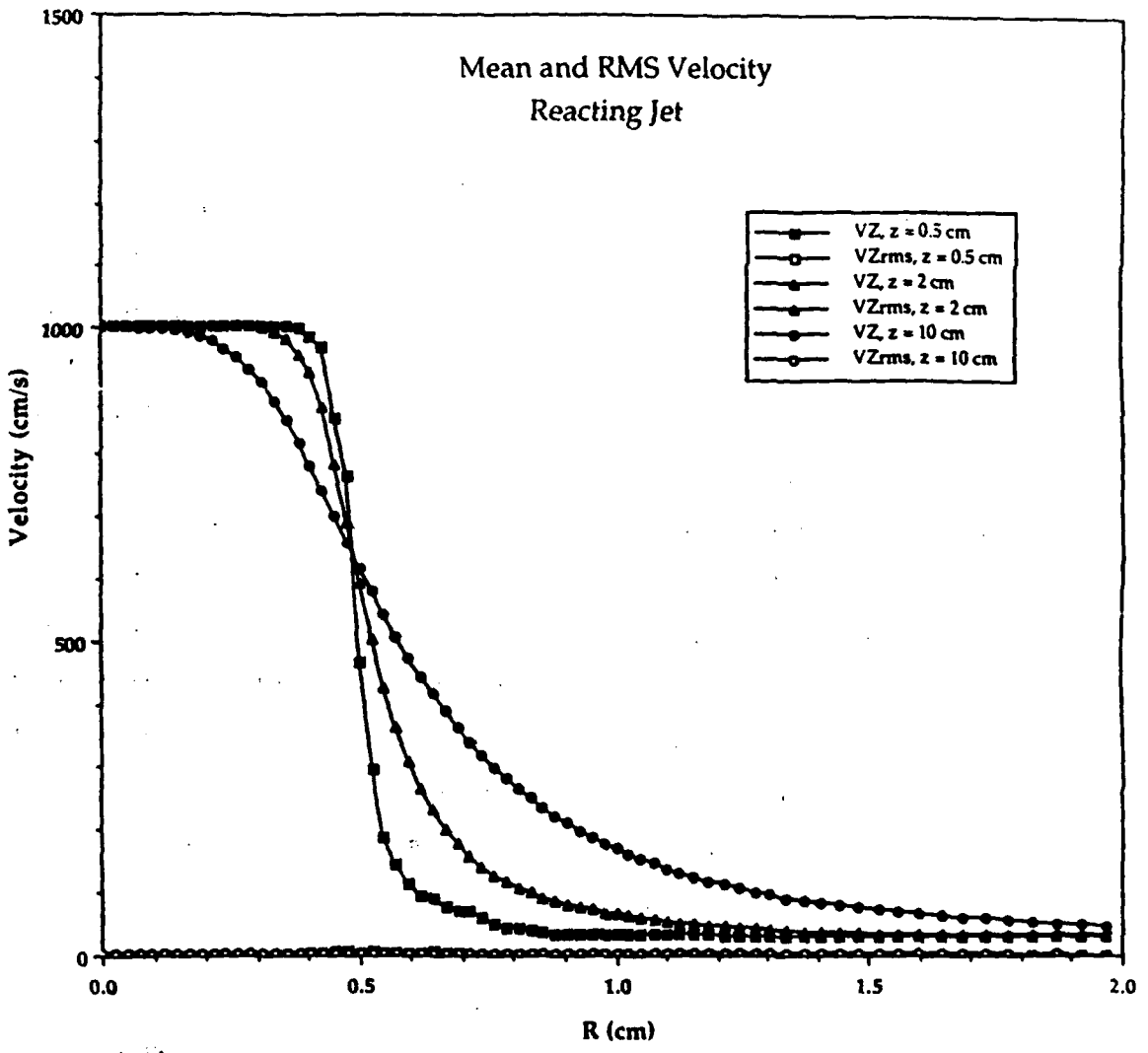


Figure 5. Mean and rms velocity for the zero-gravity diffusion flame at three axial locations.

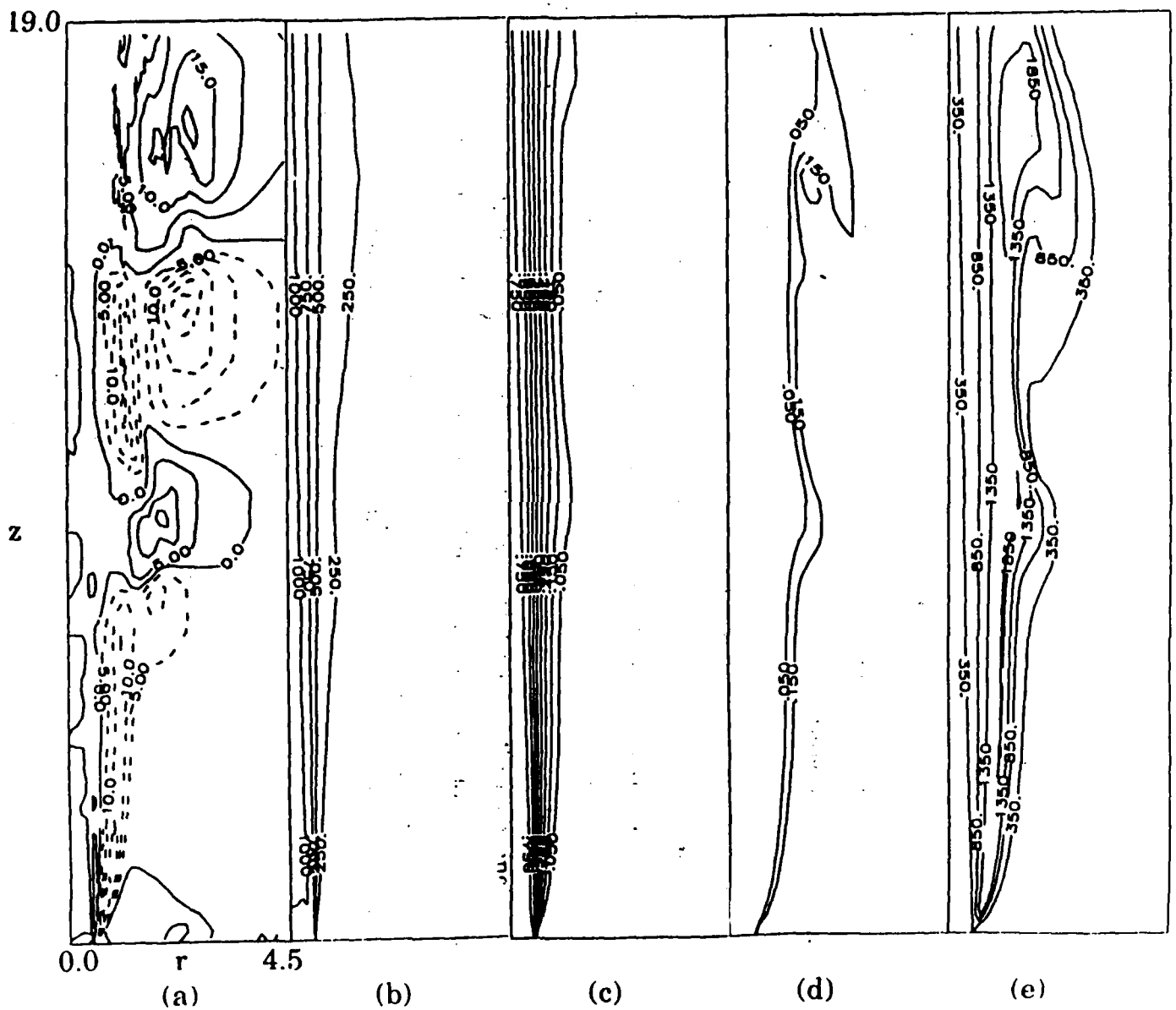


Figure 6. Contours of (a) radial velocity, (b) axial velocity, (c) mole fraction H_2, (d) mole fraction O_2, (d) temperature for normal-gravity diffusion flame formed between a $H_2 - N_2$ jet and coflowing air. Dimensions are in cm, velocities are in cm/s, temperature is in K.

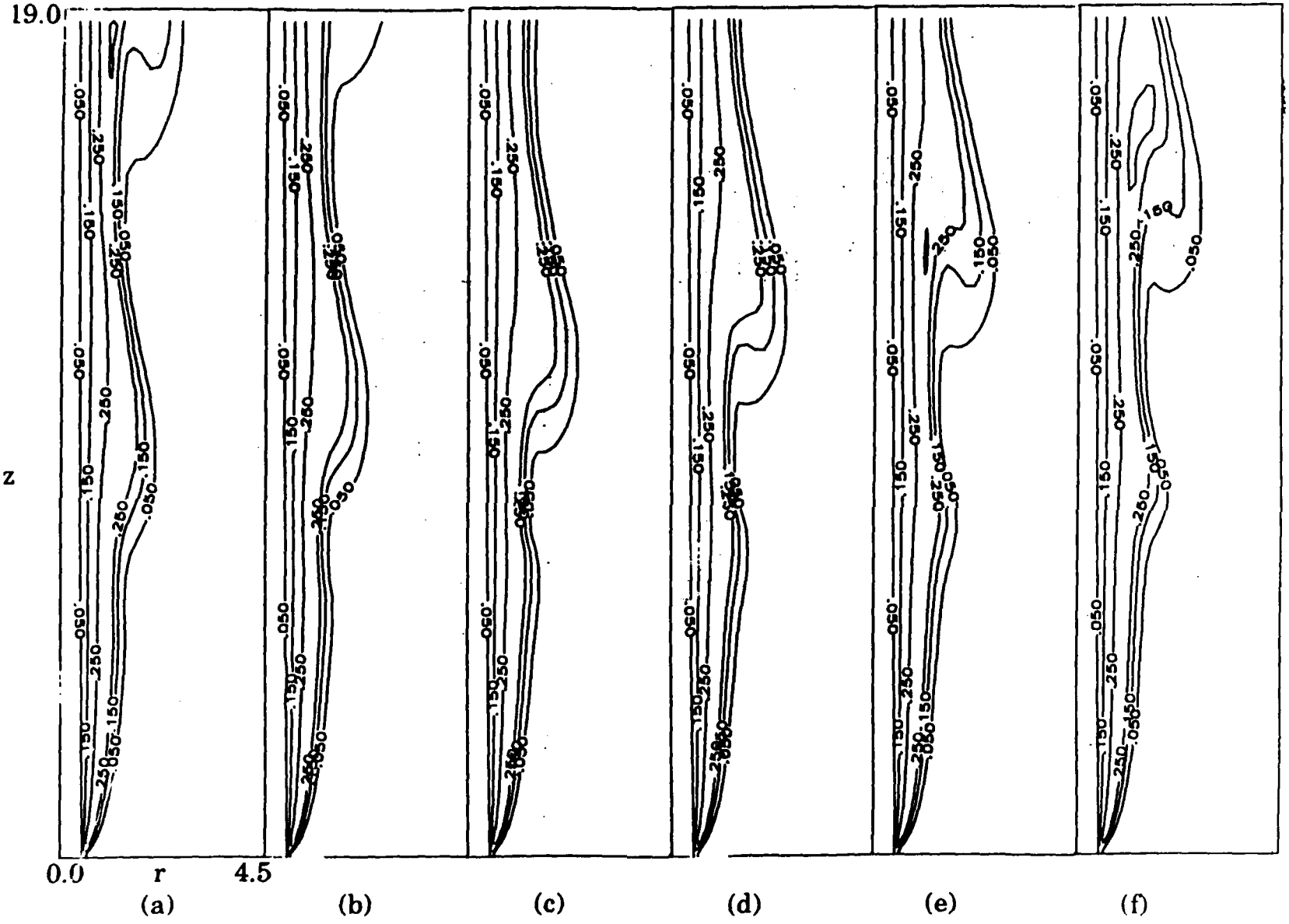


Figure 7. Sequence of contours of H_2O mole fraction showing the formation of the large vortical structures in the coflow region. The time interval between each frame is 11.25 ms.

APPENDIX R.

A Numerical Study of an Unsteady Diffusion Flame (AIAA-89-0572)

A NUMERICAL STUDY OF AN UNSTEADY DIFFUSION FLAME

K.J. Laskey*, J.L. Ellzey**, and E.S. Oran

Laboratory for Computational Physics and Fluid Dynamics
Naval Research Laboratory
Washington, D.C. 20375

Introduction

Experiments in diffusion flames have shown that the mixing of fuel and oxidizer is strongly influenced by both large and small scale structures. In a steady laminar diffusion flame where the flame interface is constant in space and time, this mixing occurs only through diffusion of the reactants into the flame zone. At higher velocities, the flames are unsteady or fluctuating and the mixing process is more complex. Typically, the unburnt fuel and oxidizer are entrained by large-scale structures and are then convected into a high temperature region where the length scales are reduced further. Ultimately, diffusive processes mix the reactants on a molecular level where chemical reactions can occur.

The existence of the large-scale structures has been extensively documented for non-reacting mixing layers and jets [1-4]. In jet diffusion flames, Roquemore et al. [5] have shown that two types of instabilites, which have very different temporal and spatial scales, exist in certain flow regimes. The smaller, inner structures, which occur closer to the jet centerline, result from Kelvin-Helmholtz instabilities in shear layers. The larger, outer structures, which form in the low-speed flow outside the mixing region, are believed to be buoyancy-driven and account for the flickering of diffusion flames. In this paper, we will simulate the region relatively near the jet of a diffusion flame and investigate the effect of these two types of structures on the flow field.

Numerical Technique

The Axisymmetric, Low-speed Jet Flame (ALJF) code [6] used for these simulations solves the following conservation equations for mass, momentum, energy, and species number density:

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{V}) = 0 \tag{1}$$

$$\frac{\partial \rho \mathbf{V}}{\partial t} + \nabla \cdot (\rho \mathbf{V}\mathbf{V}) = -\nabla P - \rho \mathbf{G} \tag{2}$$

$$\frac{\partial E}{\partial t} + \nabla \cdot (E\mathbf{V}) = -\nabla \cdot P\mathbf{V} + \nabla \cdot (\kappa \nabla t) - \nabla \cdot \sum_{k=1}^{n_{sp}} n_k \mathbf{V}_k h_k + \dot{Q} \tag{3}$$

$$\frac{\partial n_k}{\partial t} + \nabla \cdot (n_k \mathbf{V}) = -\nabla \cdot (n_k \mathbf{V}) + w_k \tag{4}$$

where ρ is the density, $\mathbf{V}$ is the velocity vector, t is time, P is pressure, $\mathbf{G}$ is the gravity vector, E is total energy density, κ is thermal conductivity, T is temperature, n_k and h_k are the number density and enthalpy of species k, and $\dot{Q}$ and w_k are the energy and species produced with reaction. The ALJF code includes a subroutine which solves all the viscous terms in the compressible Navier-Stokes equations and couples these to the results of Eqs. (1)–(4). In the results presented here, however, the viscous terms were not included.

In addition to Eqs. (1)–(4), we also need thermal and caloric equations of state. The species included in the analysis are assumed to be ideal gases obeying the thermal equation of state,

$$P = nkT . \tag{5}$$

The internal energy, e, for an ideal gas is a function of temperature only. For a multicomponent mixture of ideal gases, the differential form of the caloric equation of state is

$$de = \frac{P}{\gamma - 1}\left(\frac{dP}{P} - \frac{dn}{n}\right) + \sum_{k=1}^{n_{sp}} \bar{e}_k dn_k . \tag{6}$$

where $\bar{e}_k$ is the internal energy per mole of species k. If all of the species have a similar relationship for their internal energy, $\bar{e}_k = \bar{e}(T)$, Eq. (6) reduces to

$$de = \frac{dP}{\gamma - 1} + \left(e - \frac{P}{\gamma - 1}\right) d(\ln n) . \tag{7}$$

The set of equations is rewritten in terms of finite-difference approximations on an Eulerian mesh and then solved numerically. The accuracy of the solution is determined by the finite-difference algorithm, the spatial resolution set by the computational

* Carnegie-Mellon University, Pittsburgh, PA
Current address: Grumman Space Station Program Support Division, P.O. Box 4650, Reston, VA 22090
** Berkeley Research Associates, Springfield, VA

grid, and the temporal resolution set by the timestep. Our approach has been to model the individual processes separately and then combine the results using timestep-splitting techniques. The individual models are described in detail by Laskey [6] and discussed briefly below.

Convection

The solution to the convection of mass, momentum, and energy is obtained with the Flux-Corrected Transport (FCT) algorithm, an explicit, finite-difference algorithm with fourth-order phase accuracy. FCT has been used extensively in supersonic flows and has produced results that are in excellent agreement with theory and experiment [7]. Recently, Patnaik et al. [8] developed the Barely Implicit Correction for Flux-Corrected Transport (BIC-FCT) and this algorithm has been extended by Laskey [6]. BIC is based on the idea proposed by Casulli and Greenspan [9] that only the terms containing the pressure in the momentum equation and the velocity in the energy equation must be treated implicitly in order to avoid the sound-speed limitation on the timestep. BIC-FCT has three steps. In the first step, the conservation equations are solved explicitly with FCT using a relatively large timestep governed by the Courant condition on the fluid velocity. In the second step, the energy and momentum equations are rewritten in terms of a pressure correction, δP. These equations can be manipulated such that only one elliptic equation for δP must be solved. In the third step, final values of momentum and energy are obtained by adding the pressure correction terms. Accumulated energy fluxes are added to the calculation during the solution of δP. Patnaik et al. [10] have incorporated this algorithm in a two-dimensional flame program to investigate laminar instabilities in premixed flames.

Molecular Diffusion

A molecular diffusion algorithm has been formulated to estimate the molecular diffusion fluxes without having to solve a full matrix problem. The diffusion velocities are solved individually using Fick's Law and then corrected using a procedure described by Kee et al. [11] to assure that the sum of the diffusion fluxes is zero. This method is algebraically equivalent to the first iteration of the DFLUX algorithm [12], an iterative approach that solves for diffusion velocities. The energy flux which results from molecular diffusion is calculated during the diffusion step and is added to the total energy during the calculation of the implicit correction.

The molecular diffusion algorithm is fully two-dimensional and explicit subject to the stability constraint

$$D_{km}\Delta t\left(\frac{1}{\Delta x^2}+\frac{1}{\Delta y^2}\right)<\frac{1}{2}\,, \qquad (8)$$

where D_{km} is the diffusion coefficient for species k diffusing into a mixture and Δx and Δy are the local dimensions of a computational cell. If the limiting timestep from Eq. (8) is smaller than that required by the convection, a subcycling procedure is used to evaluate the diffusion term several times during a global convection timestep. Subcycling becomes necessary when diffusion coefficients increase due to the elevated temperature of the reacting flow.

Thermal Conduction

The conduction algorithm is also two-dimensional and explicit subject to the stability condition

$$\frac{\kappa\Delta t}{\rho c_p}\left(\frac{1}{\Delta x^2}+\frac{1}{\Delta y^2}\right)<\frac{1}{2}\,. \qquad (9)$$

where $\kappa/\rho c_p$ is the thermal diffusivity. The conduction heat flux is added to the total energy in the implicit correction step. Subcycling is implemented for thermal conduction in the same manner as it is for molecular diffusion.

Model for Chemical Reaction

The ALJF code models a single, global reaction using the Parametric Diffusion Reaction (PDR) model [6]. In the original flame sheet model proposed by Burke and Schumann [13], the fuel and oxidizer react completely and are not permitted to coexist. Thus, the flame is an infinitesimal interface between a region of pure fuel and pure oxidizer. In the PDR model, the reaction between the fuel and oxidizer does not go to theoretical completion and it occurs over a finite volume and time interval. The fraction of reaction completion during a specified time interval is calibrated against a simulation which includes the detailed chemical reactions. In the PDR model, the amount of fuel and oxidizer which react in a time step does not exceed that amount determined from the detailed chemistry simulation.

Coupling

A complete solution to the governing equations requires solving the terms for individual processes as well as accounting for the interaction between the processes. In this simulation, we use timestep splitting which assumes that the net effect of all the processes can be represented as a sum of the solutions to individual processes. This technique is valid if the changes in the dependent variables during a timestep are small.

In these computations, the changes in internal energy resulting from the individual processes are not incorporated into the solution as soon as they are computed, but instead are accumulated until the end of a convection time step. The entire change in internal energy for that time step is then added to the

energy equation in the fluid convection step. The coupling technique is described by Oran and Boris [7] and modified by Patnaik et al. [14] who have shown that this technique allows for a greater addition of energy per timestep while maintaining numerical stability.

Description of the System Modeled

In this paper, we model the geometry, composition, and flow rates used in the jet flame experiments discussed in [16] and [17]. A fuel jet (see Figure 1), whose molar composition is 78% H_2 and 22% N_2, flows vertically upward from a 0.5 cm radius contoured nozzle. The fuel jet velocity in [16] and [17] is 3.26 cm/s; the present study examines fuel jet velocities of 3.26 and 20 m/s. In both the experiments and the calculations, the jet is surrounded by a coflowing air stream whose average velocity is 0.155 m/s. The Reynolds number of the physical flame resulting from the 3.26 cm/s jet is 1000 based on physical viscosity at 300K, and drops to 135 for viscosity at 1000K. For the 20 m/s jet, the Reynolds number at 1000K rises to only about 800. Thus, the physical flames modeled are not fully turbulent, but are transitional flows in which instabilities lead to unsteady behavior.

The initial temperature and pressure in the domain are 300K and 1.013×10^6 dynes/cm^2 (1 atm), respectively. The initial number density is calculated from the ideal gas law (Eq. (5)) and this density is apportioned 78% H_2, 22% N_2 for radial distances less than 0.5 cm from the jet centerline and 80% N_2, 20% O_2 (i.e., approximately air) for radial distances greater than 0.5 cm. Total mass density is calculated from the individual number densities. The axial velocity of the hydrogen-nitrogen region is initialized to either 3.26 or 20 m/s and the axial velocity of the air region is initialized to 0.155 m/s. All radial velocities are zero at the start of the simulations. Energy is initialized using data from the JANAF tables [15].

The computational domain shown in Figure 2 is 6.4×44.7 cm and consists of 72×128 computational cells. The cell size is uniform near the nozzle exit and stretches toward the top and right boundaries. The left vertical boundary is the jet centerline of the axisymmetric flow and the lower boundary is the exit plane of the jet nozzle and the inflow boundary for the calculation. The timestep is adjusted throughout the calculation to ensure that the Courant stability limit based on flow velocity is satisfied. For the 3.26 m/s simulation, the timestep is approximately 30 μs; for the 20 m/s simulation, it is approximately 8 μs. Both simulations spanned approximately 0.5 s of physical time and required 6 to 8 hours on a Cray-XMP.

Results of Simulations

Here we use contours of important physical quantities to display the results of the calculations. Relative species concentrations are shown by figures labeled R_f, R_o, and R_p, where

$$R_f = \frac{f}{f+o+p} \tag{10}$$

$$R_o = \frac{o}{f+o+p} \tag{11}$$

$$R_p = \frac{p}{f+o+p} \tag{12}$$

and f, o, and p are the fuel (H_2), oxygen (O_2), and product (H_2O) number densities, respectively. Each frame in the figures is labeled with the computational step number and the corresponding physical time in seconds. In addition, we present contours of radial velocity normalized to the maximum velocity magnitude (value in parentheses) in the domain shown. The contours are shown at intervals of 0.1, where solid lines indicate zero and positive values and dashed lines indicate negative values.

The results of the 3.26 m/s simulation, beginning with Figure 3, show a domain which extends 2.41 cm from the jet centerline in the radial direction and 7.38 cm from the inlet boundary in the axial. Both the calculations and the experiments show periodic growth and shedding of vortical structures outside the flame surface, but no structures at the jet shear layer internal to the flame. A typical pattern of the growth and shedding of the outer structure is shown by contours of R_f, R_o, and R_p in Figures 3, 4, and 5. The flame surface in these figures is located between the 0.100 R_f contour (rightmost in each R_f plot) and the 0.100 R_o contour (left most in each R_o plot). The fuel (R_f) contours are pinched and bulge or ripple periodically as the structure passes, but the structure itself is not visible in the R_f contours because it is outside the fuel region. It appears in the oxygen (R_o) and product (R_p) contours that show the vortical structure and the entrainment of oxygen and product. The radial velocity contours in Figure 6 also show the evolution of this large outer structure. The vortices begin to grow near the axial location above the exit plane where the rapid expansion of the flame surface is complete. This structure is commonly attributed to instability of the shear layer between the flame and the surrounding air, but the frequency and size of the structure do not correlate with empirical predicitions of shear layer instabilities. The initiation of the vortex may be the result of the flow field adjusting to the sudden change in the flow area created by the expanding flame. The frequency of the growth and shedding cycle is approximately 12 Hz and this is

consistent with the flicker frequency observed in [16] and [17].

Instantaneous temperature profiles are shown in Figures 7a and b at two different axial locations, 0.92 cm and 4.09 cm above the nozzle exit plane. These temperature profiles were taken at the same time as the contours in Figures 3 – 6 At the lower axial position shown in Figure 7a, the temperature profiles remain relatively constant in time, varying little in magnitude or location. In contrast, peak temperatures at the higher axial position, an example of which is shown in Figure 7b, vary by as much as 230K and the radial location of the peak shifts in position by 0.3 to 0.6 cm. In Figure 7b, the peak temperature ranges between 2500K and 2730K and the radial location of the peak varies from 0.8 cm to 1.3 cm. The minimum peak temperature corresponds to densely packed R_p contours in Figure 5. The dense packing indicates that the reaction zone is thinned and, consequently, the temperature gradient is steepened. This enhances thermal conduction and contributes to a lower peak temperature. In addition, the large outer structures strongly affect the transport of oxygen to the flame surface, an effect that could lead either to lower temperatures due to mixing with cooler gases or higher temperatures because of exhanced mixing which encourages more reaction. The complicated interactions between the flame and the coflow region are directly related to the peak magnitude and movement of the high-temperature region. The location, size, and frequency of changes in the temperature peak generally agree with experimental results presented in [16] and [17]. However, the maximum calculated peak temperature is on the order of 2730K and this exceeds the experimental maximum by approximately 400K.

The source of the temperature discrepancy may indicate the need for further calibration of the PDR model. The concentration of the product species in the calibration are a function of the flame temperature and the present PDR model does not account for varying flame temperature when specifying the fraction of allowable reaction, x. If x is too large, the higher resulting temperature leads to larger diffusion coefficients, more fuel and oxygen in the reaction zone, and yet higher temperatures. For example, using a traditional flame sheet model ($x = 1$) produces maximum temperatures in excess of 3000K. A small change in x reflecting the oscillating temperature of the flame may be sufficient to reduce the peak temperature to a more correct value. The higher temperatures in the calculations may also occur because there is no nozzle lip in the computational domain and the reaction model does not include losses that occur because of the nozzle. Effects such as heat transfer to the nozzle and the absence of chemical reactions in the immediate vicinity of the nozzle would reduce the local temperature which will feed back through the diffusion coefficients to further reduce the temperature.

The preliminary experimental results for the 20 m/s jet not only show periodic growth and shedding of a vortex structure outside the flame surface, but also vortices forming inside the flame surface along the shear layer between the fuel jet and the slower, coflowing product generated by the reaction. The computer simulation, especially the R_f contours in Figure 8 and the radial velocity contours in Figure 9 show similar structures being generated, growing, and merging along the shear layer. The domain shown in these figures extends 1.79 cm from the jet centerline in the radial direction and 14.2 cm from the nozzle exit in the axial. The time elapsed between frames in each figure is approximately 0.164 ms.

From the simulation of nonreacting axisymmetric shear layers, Grinstein et al. [18] note that the natural instability frequency of the shear layer is characterized by the Strouhal number based on shear layer thickness, $St_{\theta_o} = f\theta_0/U_0$, where f is the instability frequency, U_0 is the velocity of the jet and θ_0 the initial shear layer thickness . When the nozzle is not included in the computational domain, θ_0 is effectively equal to the radial spacing of the computational grid. The value of St_{θ_o} which predicts the frequency is typically about 0.014, giving a predicted frequency for the present jet of approximately 1200 Hz. Note that the inner structure has a much higher frequency than the structure outside the flame surface. If the instability wavelength, λ_0, is given by $\lambda_0 = v_p/f$, where v_p is the typical longitudinal phase velocity equal to $0.6U_0$ [18], then the expected wavelength for the present jet would be approximately 1 cm. The predicted wavelength is approximately the same as the distance measured between successive initiations of the vortex as seen from Figure 9, and the predicted frequency is within a factor of two of the frequency estimated from the figure. Thus, although a shear layer in the simulation is analagous to the physical shear layer, the length and time scales of the calculated shear layer are governed by the effective momentum thickness introduced by the spacing of the computational grid.

The results of the 20 m/s jet simulation also show vortices growing and being shed outside the flame surface. A typical cycle is shown by contours of R_f, R_o, and R_p in Figures 10–12. These figures show a domain which extends 3.88 cm from the jet centerline in the radial direction and 14.2 cm from the inlet boundary in the axial.

As in the lower-velocity jet, the R_f contours in Figure 10 bulge and pinch periodically, but the outer structure is not visible. The structure does appear in the oxygen and product contours in Figures 11 and

12, but the vortices are not as well formed as those generated in the lower-velocity simulation. Figure 13, the radial velocity contours, shows one possible reason for this. The domain shown in this figure extends to 6.07 cm in the radial direction and 14.2 cm in the axial direction, and the velocity is contoured over the range -2 to 2 m/s in intervals of 0.2 m/s. Whereas the outer structure in Figure 6 grew and moved independently of the outside flow, a small structure in Figure 13 appears to break off the end of a larger, structure which moves along with it. The larger, structure loses form as it moves out of the axially well-resolved region, but a similar structure is periodically generated approximately 4 cm above the nozzle exit. The frequency of this cycle is approximately 8 Hz.

The temperature trends shown in Figures 14a and b for the higher-velocity jet mirror those discussed above for the lower-velocity case. Below the height where the outer structure grows (for example, 2.52 cm from the jet inlet), the temperature profile is fairly constant in time. At a height of 5.81 cm from the jet, the flow is affected by the growth and passage of the outside structure and the location and magnitude of the temperature peaks reflect the denser packing of the R_p product contours shown in Figure 12. For the higher-velocity jet, however, the peak temperatures have increased, varying from a minimum of 2720K to a maximum of 2880K. This is approximately 150K higher than lower-velocity peak temperatures. As with the 3.26 cm/s jet, the interaction between the flame surface and the structures in the coflow region include a complex combination of convection and heat transfer effects. But, in addition, the high frequency structures at the jet shear layer may enhance mixing between the fuel and product, leading to more fuel being available at the flame surface. This could account for the higher peak temperatures.

Conclusions

In this paper, we present the results of numerical simulations of two unsteady $H_2 - N_2$ diffusion flames with jet velocities of 3.26 m/s and 20 m/s and investigate the instabilities which occur in these flows. At the lower velocity, the flow field is smooth except for the existence of large structures in the slowly moving coflow region. These structures have a frequency of approximately 12 Hz and appear to be linked to flickering observed in diffusion flames. At the higher velocity, we see not only low-frequency structure but also a higher-frequency instability which forms in the mixing layer between the high and low velocity gases. The main effect of the high-frequency structure may be to enhance the transfer of fuel to the flame surface. The larger structures in the coflow, however, affect the overall location of the flame surface, the local irregularities of the surface, and the convective and heat transport processes interacting with the flame. This is most apparent in the variation of the location and magnitude of the peak temperature at a given axial location. The results shown in this paper qualitatively agree with experimental results presented by Roquemore et al. [5] that show two types of instabilities.

The flows shown in this paper result from a complex interplay between the jet and the coflowing air, and this interaction leads to the structures and patterns of vortex initiation and growth. Thus, it is important to assess the effect of the computational grid on these outer coflow patterns. Each frame of the radial velocity contours in Figure 6 shows a domain which extends 4.38 cm from the jet centerline in the radial direction and the same 7.38 cm in the axial. Because of limited resolution and grid stretching in the cells approaching the right boundary, the flow field is distorted and the results are degraded. The intent of the initial grid design was to provide a region which could act as a cushion between the structures forming near the flame surface and the edge of the computational domain. It was not intended to model a region which may contain complicated structure of its own. Similarly, a large portion of the axial extent of the domain consists of highly stretched cells that act as a cushion and damp spurious reflections from the top boundary. This means that there is good resolution only in a domain approximately 1.5 cm from the jet centerline in the radial direction and 5.0 cm from the jet inlet in the axial. The influence of the flow patterns from radial distances greater than 1.5 cm may obscure the details of structure growth and the interactions between the generated vortices may not be represented accurately. Similarly, the stretched grid at axial distances greater than 5.0 cm probably affects the ability of vortices to maintain their structure further downstream. It should be noted that of the 72 computational cells in the radial direction of the full domain, 55 are in the 1.5 cm radial distance and 90 of the 128 cells in the axial direction are in the 5.0 cm axial distance. The high-temperature region is contained in the well-resolved radial portion of the grid, and the temperature of the coarsely resolved region is expected to be near ambient.

These results point to a number of future calculations that would enhance of understanding of unsteady jet diffusion flames. For example, the variations in temperature implies that the larger structures affect the reaction of fuel and oxidizer, but the convective process in these flows is quite complex. For example, the large structures may either increase the amount of reaction which occurs by more efficiently convecting unburnt reactants to the flame zone, or they may quench the reaction by supplying an excess of cold gases to the flame. In addition, although

the computational time requirements for these calculations were substantial, simulations with finer grids and larger domains are necessary to obtain detailed statistical information about passing frequency and to correlate these results to the energy release in the flame. Finally, we must improve the chemical reaction model to reflect changes in local temperature.

Acknowledgements

The authors would like to acknowledge the support, encouragement, and extremely useful discussions with Dr. W.M. Roquemore from the Air Force Wright Aeronautics Laboratory, and the support from the the Naval Research Laboratory through the Office of Naval Research. In addition, we would like to thank Dr. Gopal Patnaik who developed many of the algorithms used in the simulations and provided many helpful suggestions. The computer simulations were performed on the Cray X-MP at the Pittsburgh Supercomputing Center under proposal PSCA 194.

References

1. C.-H. Ho and P. Huerre, Perturbed Free Shear Layers,*Ann. Rev. Fluid Mech.*, 16, 365–424, 1984.
2. C.-H. Ho and L.S. Huang, Subharmonics and Vortex Merging in Mixing Layers, *J. Fluid Mech.*, 119, 443–73, 1982.
3. F.K. Browand, An Experimental Investigation of the Instability of an Incompressible Separated Shear Layer, *J. Fluid Mech.*, 26, 281–307, 1966.
4. W.J.A. Dahm and P.E. Dimotakis, Measurements of Entrainment and Mixing in Turbulent Jets, *AAIA J.*, 25 (9), 1216–1223, 1987.
5. W.M. Roquemore, L.-D. Chen, L.P. Gross, and W.F. Lynn, The Structure of Jet Diffusion Flames, to appear in the Proceedings of the United States – France Joint Workshop on Turbulent Reactive Flows, Rouen, France, 1987.
6. K.J. Laskey, *Numerical Study of Jet Diffusion Flames*, Ph.D. Dissertation, Department of Mechanical Engineering, Carnegie-Mellon University, 1989.
7. E.S. Oran and J.P. Boris, *Numerical Simulation of Reactive Flow*, Elsevier, New York, 1987.
8. G. Patnaik, R.H. Guirguis, J.P. Boris, and E.S. Oran, A Barely Implicit Correction For Flux-Corrected Transport, *J. Comput. Phys.*, 71, 1–20, 1987.
9. V. Casulli and D. Greenspan, Pressure Method for the Numerical Solution of Transient, Compressible Fluid Flows, *Int. J. Num. Methods Fluids*, 4, 1001–1012, 1984.
10. G. Patnaik, K. Kailasanath, K. J. Laskey, and E. S. Oran, Detailed Numerical Simulations of Cellular Flames, to appear in the *Proceedings of the Twenty-Second Symposium (International) on Combustion*, The Combustion Institute, Pittsburgh, PA, 1988.
11. R.J. Kee, G. Dixon-Lewis, J. Warnatz, M.E. Coltrin, and J.A. Miller, A Fortran Computer Code Package for the Evaluation of Gas-Phase Multicomponent Transport Properties, SAND86-8246, Sandia National Laboratory, 1986.
12. W.W. Jones and J.P. Boris, An Algorithm for Multispecies Diffusion Fluxes, *Comp. Chem.*, 5,139–146, 1981.
13. S.P. Burke and T.E.W. Schumann, Diffusion Flames, *Indust. Eng. Chem.*, 20, 998–1004, 1928.
14. G. Patnaik, K.J. Laskey, K. Kailasanath, E.S. Oran, and T.A. Brun, *FLIC – A Detailed, Two-Dimensional Flame Model*, NRL Memorandum Report (in preparation), Naval Research Laboratory, Washington, DC, 1988.
15. *JANAF Thermochemical Tables*, second edition, National Bureau of Standards, 1971.
16. W.F. Lynn, L.P. Gross, T.H. Chen, and D.D. Trump, Two-Dimensional Velocity Measurements on an Axially Symmetric H_2-N_2 Jet Diffusion Flame, presented at the 1988 Spring Technical Meeting of the Combustion Institute, Central States Section, Indianapolis, IN.
17. L.P. Gross, V. Vilimpoc, W.F. Lynn, and B. Sarka, Simultaneous TFP and RMS Measurements on a H_2 Jet Diffusion Flame, presented at the Twenty-Second Symposium (International) on Combustion, Seattle, WA, 1988.
18. Direct Numerical Simulation of Axisymmetric Jets, F.F. Grinstein, E.S. Oran, and J.P. Boris, *AIAA J.*, 92-98, 1987; F.F. Grinstein, E.S. Oran, and J.P. Boris, Numerical Simulation of the Transitional Region of an Axisymmetric Jet, AIAA 25th Aerospace Sciences Meeting, Paper No. AIAA-87-0052, AIAA, NY, 1987.

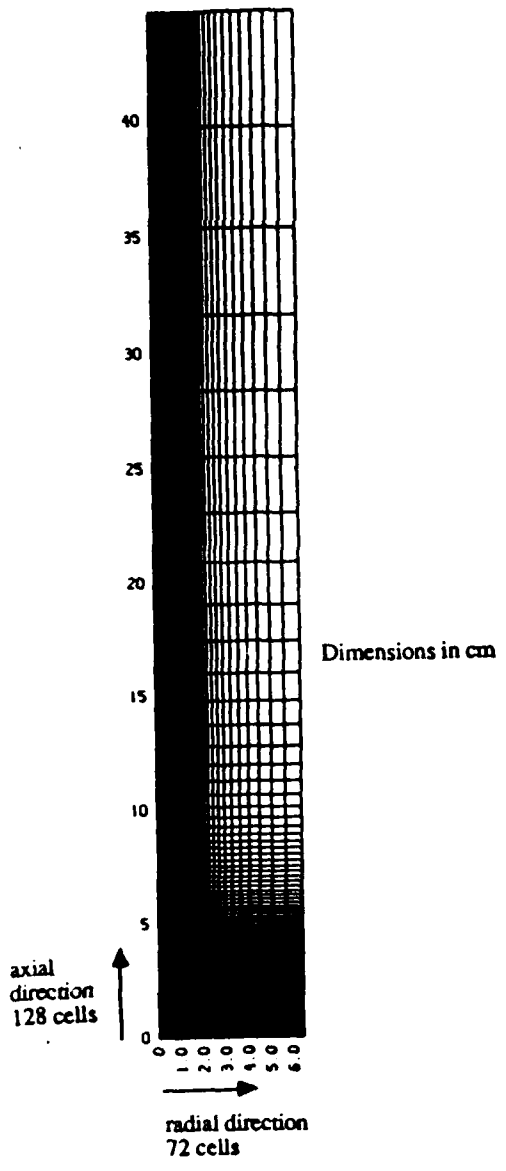


Figure 1. Full computational domain.

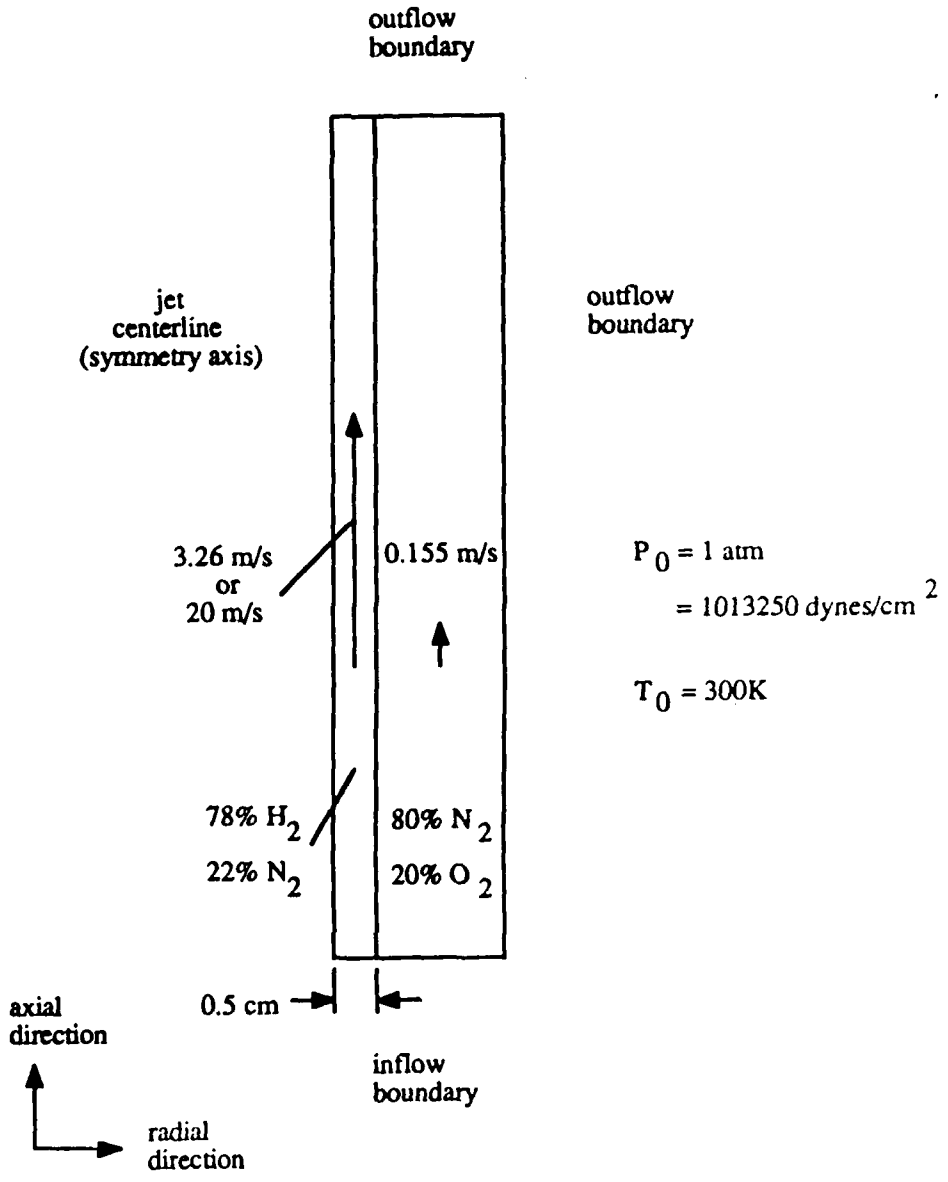


Figure 2. Calculation initialization

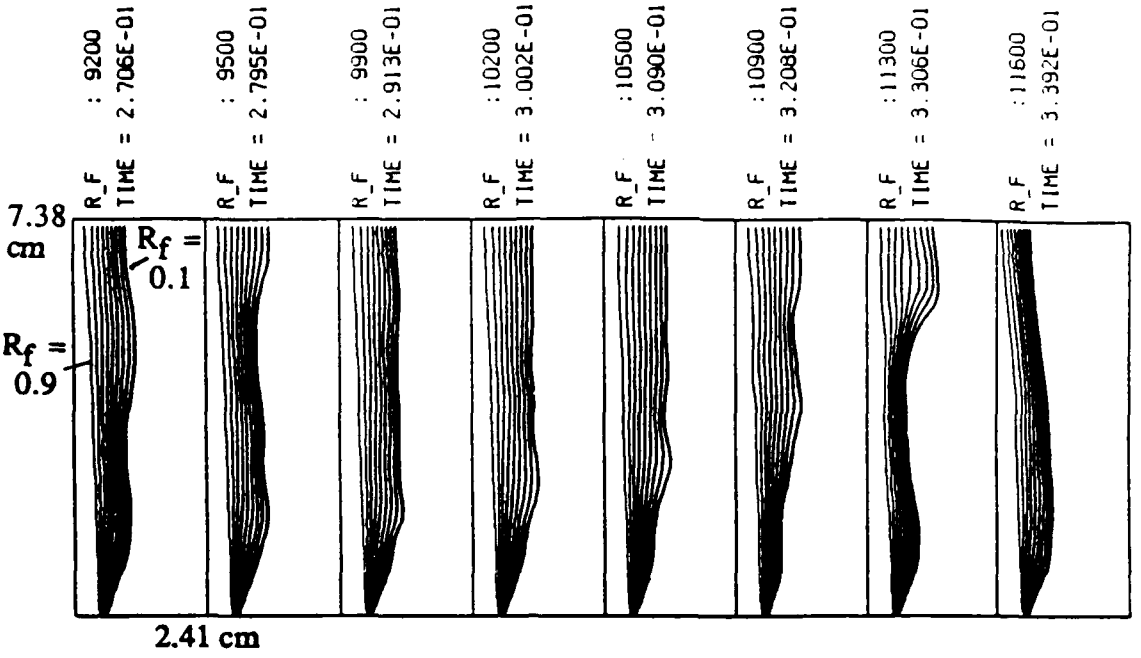


Figure 3. Relative fuel concentration countours for one cycle of outer structure growth.

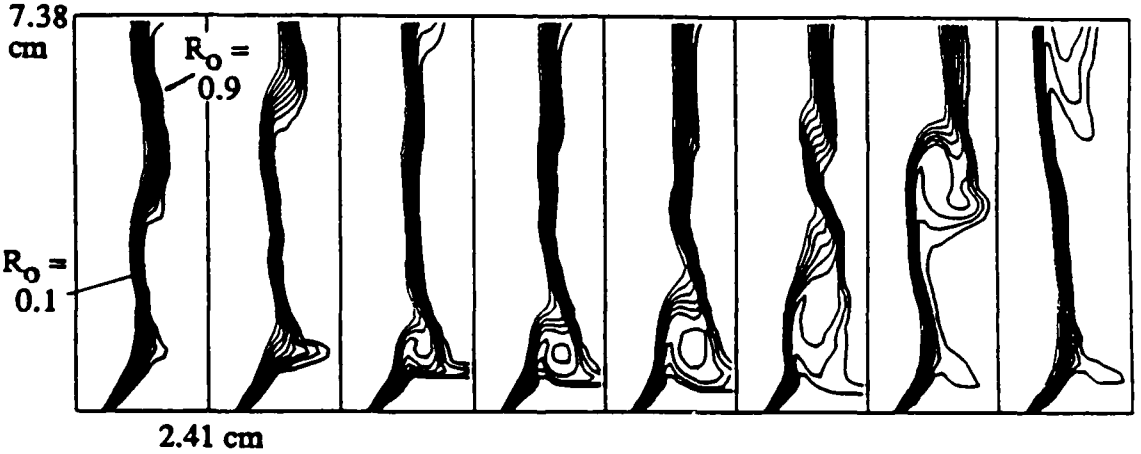


Figure 4. Relative oxygen concentration countours for one cycle of outer structure growth.

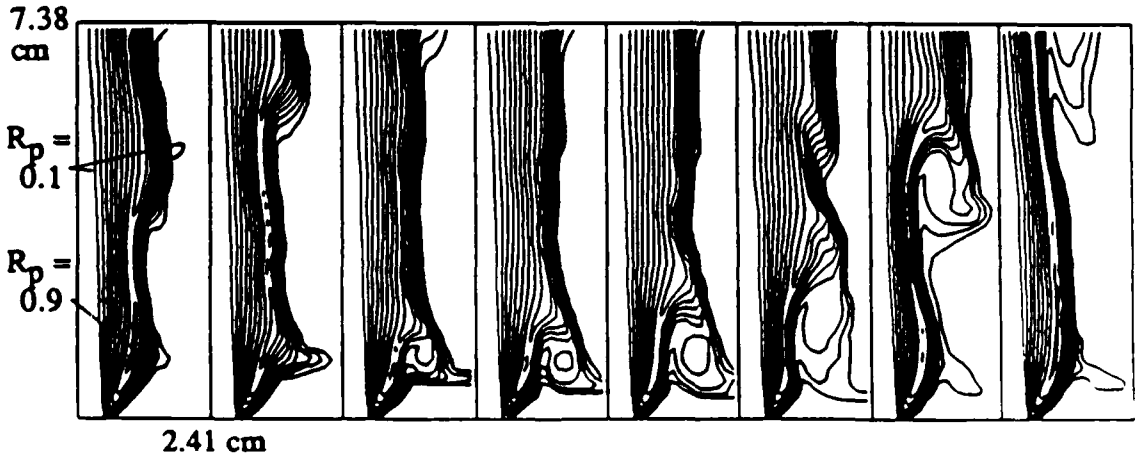


Figure 5. Relative product concentration countours for one cycle of outer structure growth

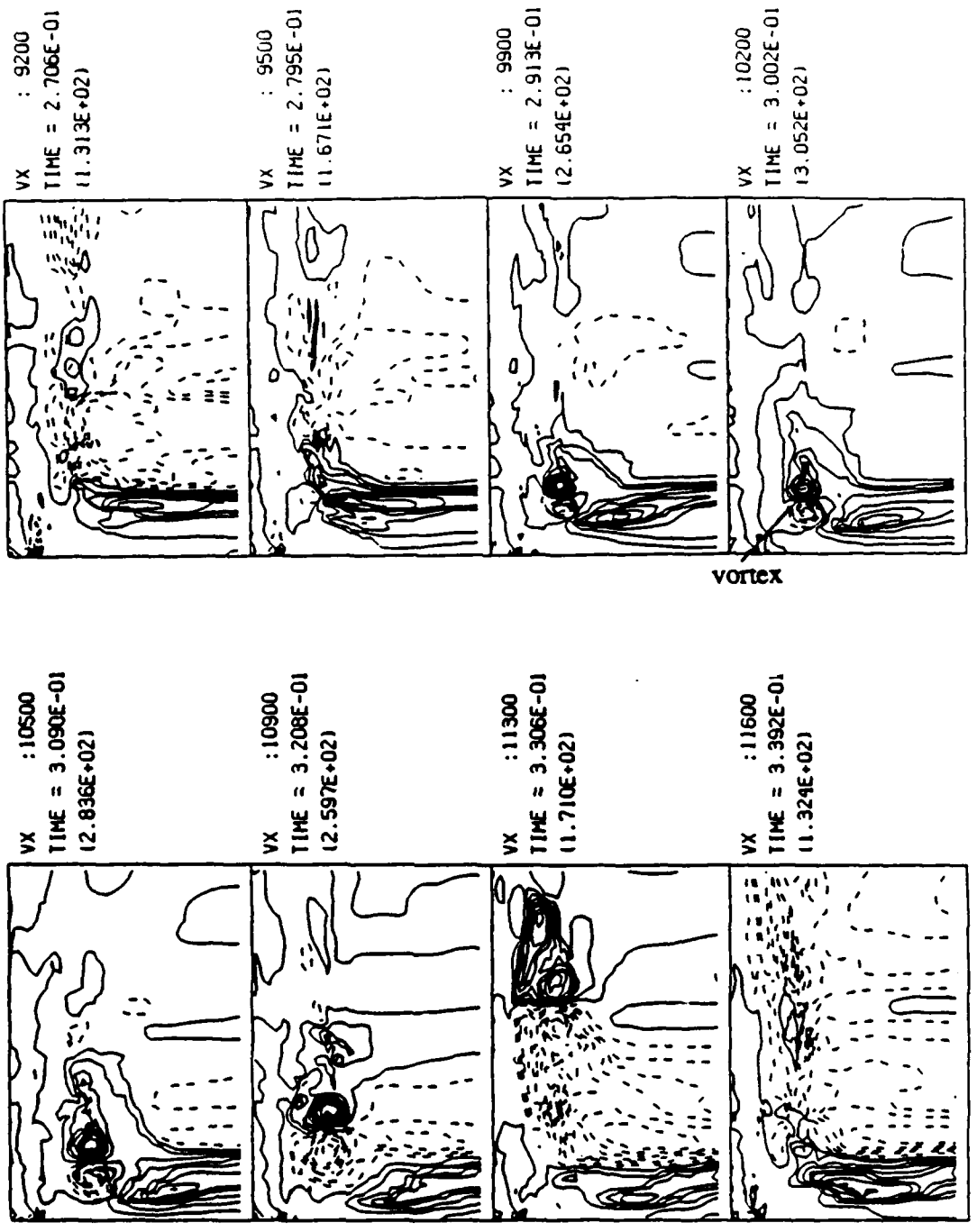


Figure 6. Radial velocity contours showing outer structures.

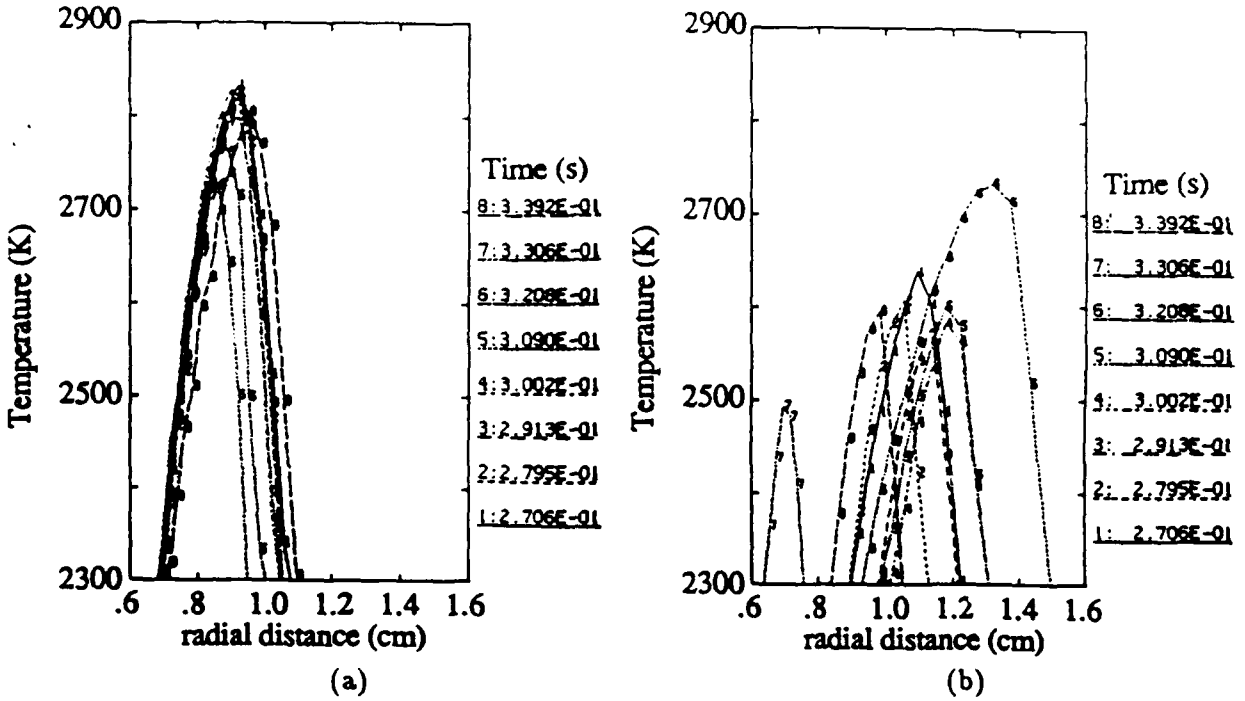


Figure 7. Instantaneous temperature profiles at an axial height of (a) 0.92 cm (b) 4.09 cm.

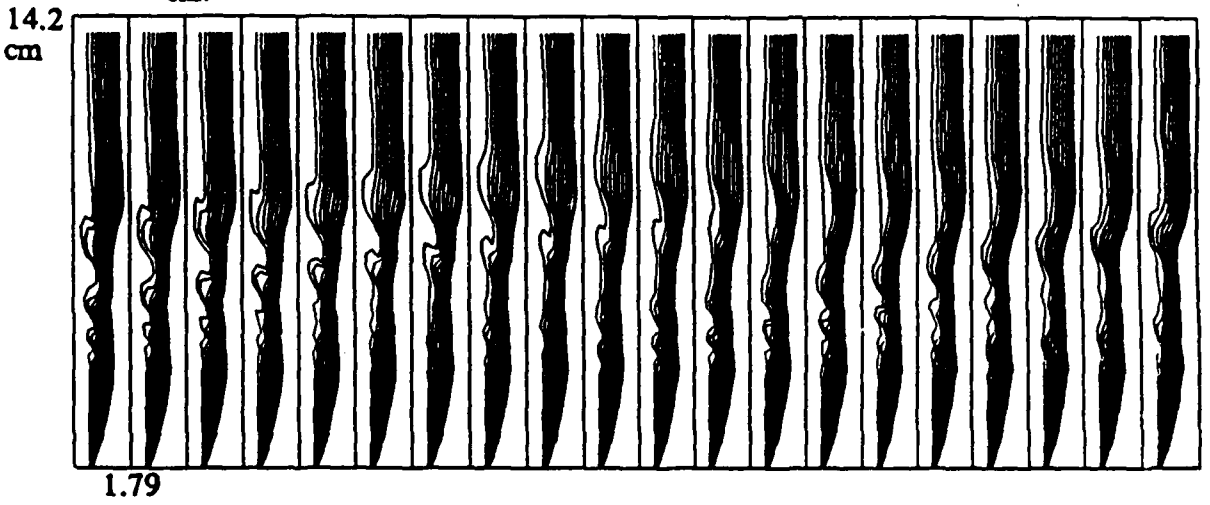


Figure 8. Relative fuel concentration showing inner structures.

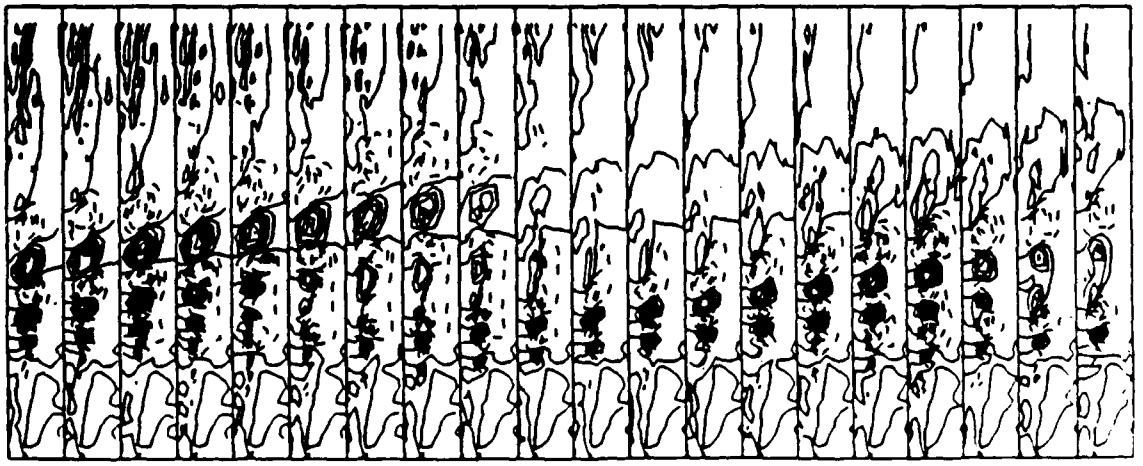


Figure 9. Radial velocity contours showing inner structures.

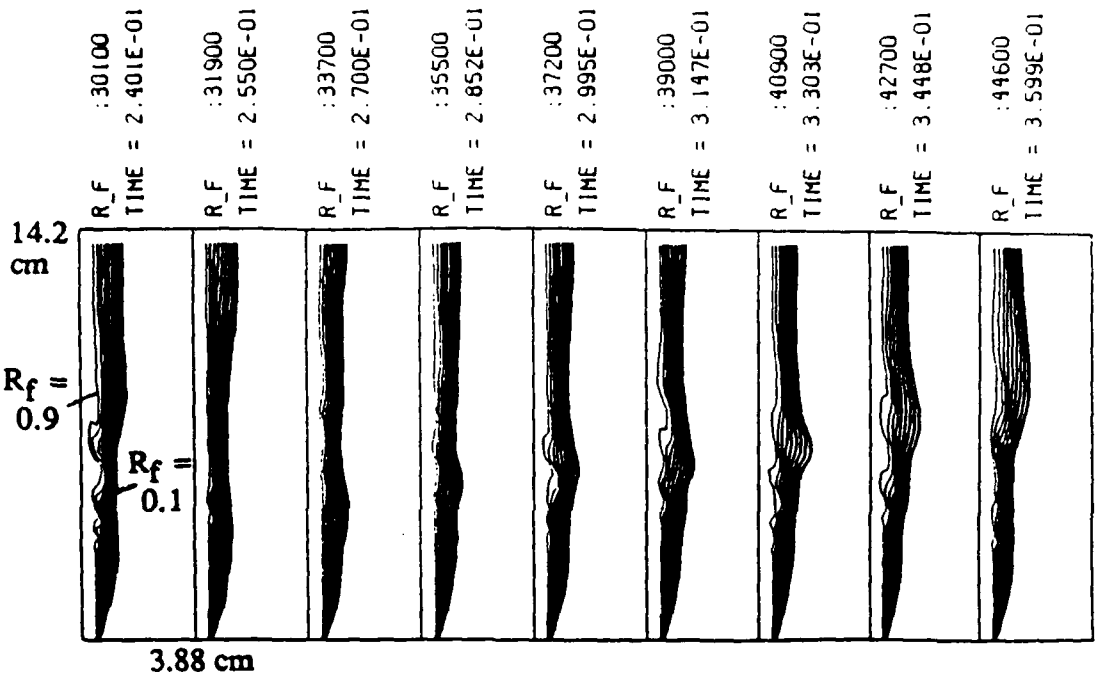


Figure 10. Relative fuel concentration countours for one cycle of outer structure growth for the 20 m/s jet.

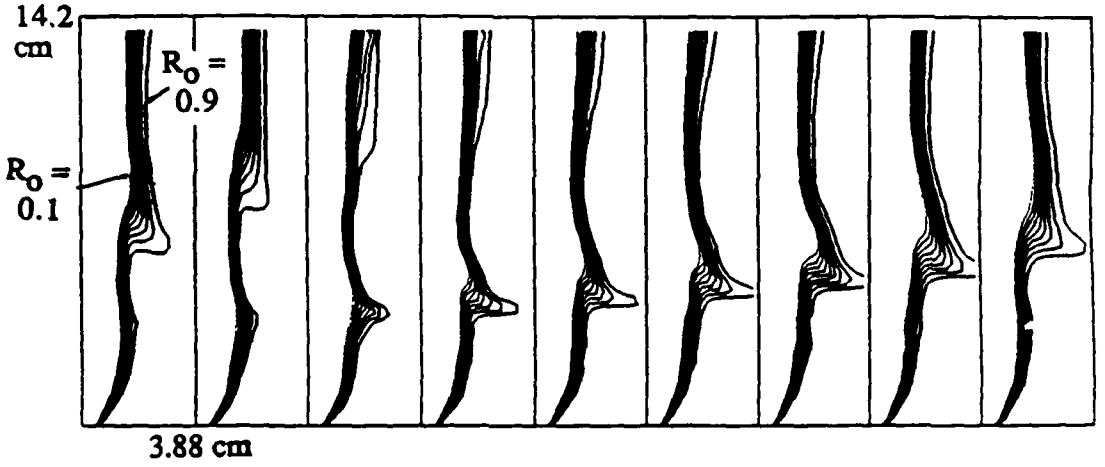


Figure 11. Relative oxygen concentration countours for one cycle of outer structure growth for the 20 m/s jet.

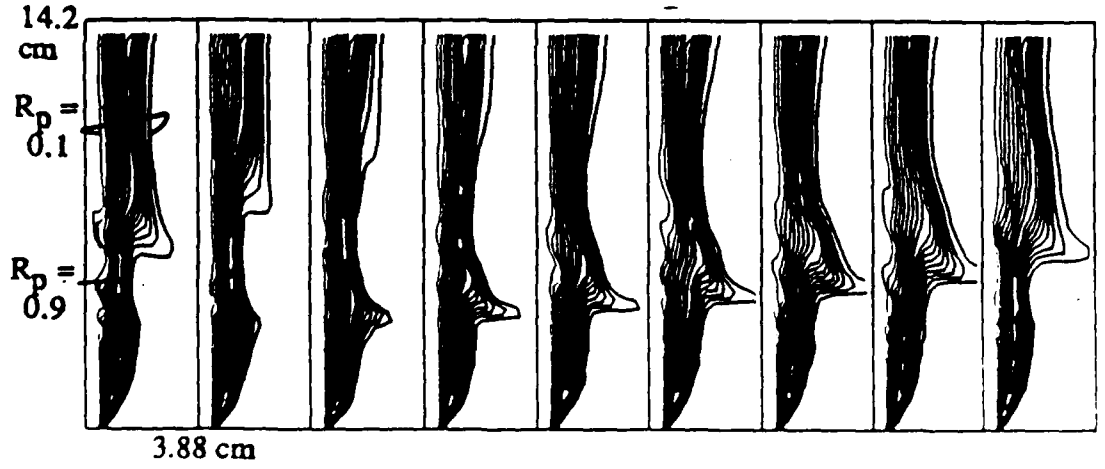


Figure 12. Relative product concentration countours for one cycle of outer structure growth for the 20 m/s jet.

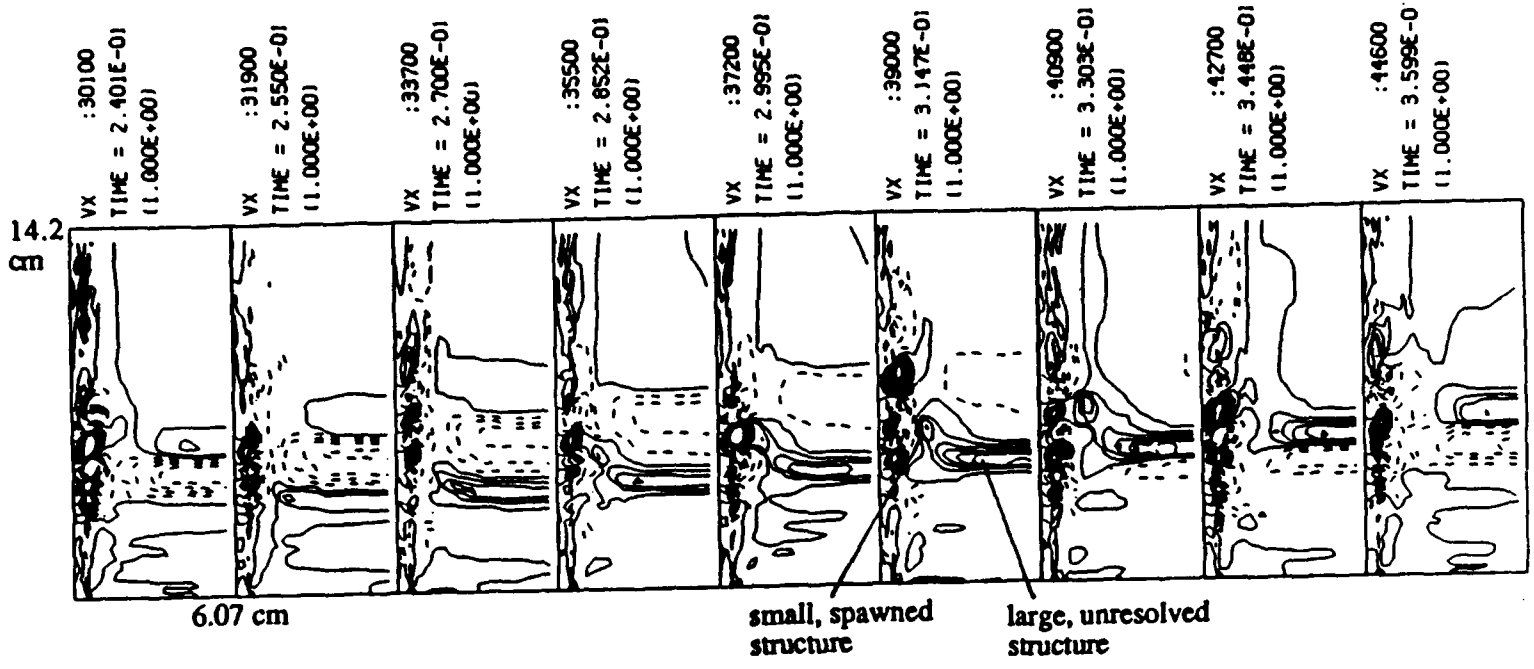


Figure 13. Radial velocity contours showing outer structures for 20 m/s jet.

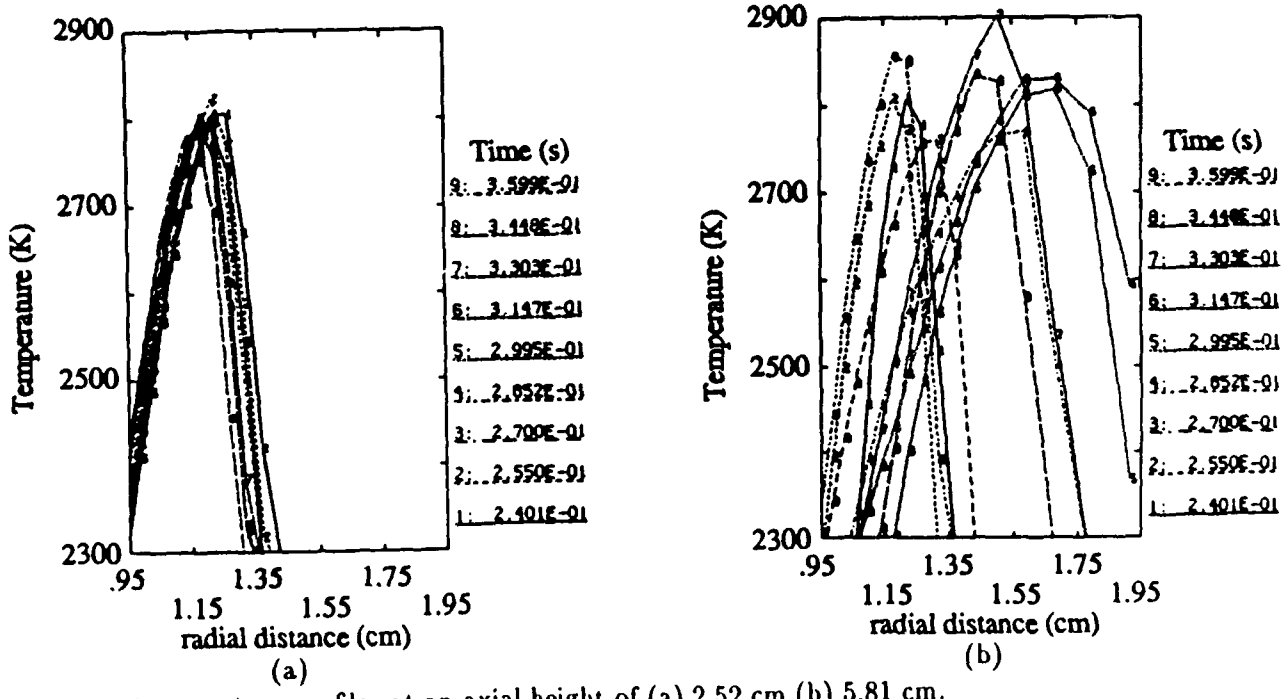


Figure 14. Instantaneous temperature profiles at an axial height of (a) 2.52 cm (b) 5.81 cm.

APPENDIX S.

Adaptive Finite Element Flux Corrected Transport Techniques for CFD (to appear in Springer Lecture Notes)

to appear in Springer Lecture Notes

ADAPTIVE FINITE ELEMENT FLUX CORRECTED TRANSPORT TECHNIQUES FOR CFD

K. Morgan, J. Peraire and R. Löhner

I.N.M.E.
University of Wales
Swansea SA2 8PP
United Kingdom

Code 4040
Naval Research Laboratory
Washington, DC 20370
U.S.A.

INTRODUCTION

In previous papers [1,2] we have described an explicit finite element solution procedure for the compressible Euler and Navier-Stokes equations. The approach was a finite element equivalent of a two-step Lax-Wendroff scheme and was implemented on unstructured triangular or tetrahedral grids. An important feature of the work was the use of adaptive mesh refinement methods for the solution of steady state problems in 2D, using error indicators based upon interpolation theory.

In this paper, some recent developments in the extension of this approach are considered. We will describe how the basic solution procedure can be modified in a straightforward manner to produce a high resolution scheme on unstructured grids. This is accomplished by utilizing, in a finite element context, Zalesak's [3] multidimensional extension of the flux corrected transport (FCT) ideas of Boris and Book [4]. The problem of triangular mesh generation will be addressed and the adaptive mesh approach will be widened to handle 2D problems involving strongly transient phenomena. This will be implemented by allowing adaptive refinement and derefinement of the mesh as the solution proceeds. Finally, it will be demonstrated how directional refinement procedures can be incorporated for the efficient computation of steady 2D flows involving significant 1D features.

BASIC ALGORITHM

The basic solution algorithm will be briefly described for the two dimensional Navier-Stokes equations written in the conservative form

$$\frac{\partial \underline{U}}{\partial t} + \frac{\partial \underline{F}_j}{\partial x_j} = \frac{\partial \underline{G}_j}{\partial x_j} \qquad (j=1,2) \qquad (1)$$

where $\underline{U}$ is the vector of conservation variables and $\underline{F}_j$ and $\underline{G}_j$ denote the advective and viscous flux vectors respectively. A time-stepping scheme for this

equation can be developed, in an operator-split fashion, by treating the diffusion terms in an explicit manner and the advective terms in the Lax-Wendroff fashion [5]. The result is that

$$\underline{U}^{m+1} = \underline{U}^{m} - \Delta t \frac{\partial \underline{F}_j^m}{\partial x_j} + \frac{\Delta t^2}{2} \frac{\partial}{\partial x_j} \left[\underline{A}_j^m \frac{\partial \underline{F}_k^m}{\partial x_k} \right] + \Delta t \frac{\partial \underline{G}_j^m}{\partial x_j} \qquad (2)$$

where a superscript m denotes an evaluation at time $t = t_m$, $t_{m+1} = t_m + \Delta t$ and

$$\underline{A}_j = \frac{d\underline{F}_j}{d\underline{U}} \qquad (3)$$

The spatial domain, Ω, is discretized using 3-noded linear triangular elements and a weighted residual [6] form of equation (2) is considered. The resulting integrals are evaluated exactly (in a 2-step fashion by firstly calculating an element level approximation to $\underline{U}^{m+1/2}$ [7]), leading to an equation

$$\underline{M}\ \underline{\delta U^H} = \underline{f}^m \qquad (4)$$

where $\underline{M}$ is the consistent mass matrix, $\underline{\delta U^H}$ is the vector of changes in the nodal values of $\underline{U}$ over the timestep and the superscript H is introduced for use later. For the simulation of transient flows, this equation system can be solved iteratively and explicitly [8] and the method coupled with a domain splitting technique [9] to produce an efficient computational procedure. For steady flows, local time steps are employed and equation (4) is replaced by the explicit scheme

$$\underline{M}_\ell\ \underline{\delta U^H} = \underline{f}^m \qquad (5)$$

where $\underline{M}_\ell$ is the lumped diagonal mass matrix. It should be noted that , in 1D equation (5) reduces to the well-known finite difference scheme of Burstein [10].

For problems involving strong shocks, the solution $\underline{U}^{m+1}$ is smoothed, by the application of artificial viscosity [11], before proceeding to the next time step.

FCT EXTENSION

A more robust solution scheme, giving better resolution of flow discontinuities, can be produced by adding an FCT procedure to the above process. Erlebacher [12] and Parrott and Christie [13] have demonstrated how Zalesak's [3] multidimensional extension of the FCT algorithm of Boris and Book [4] can be implemented on triangular grids. The idea is to combine a high order scheme with a low order scheme in such a way that the high order scheme is employed in regions where the flow variables vary smoothly, whereas the low order scheme is favored in those regions where the variables vary abruptly. The low order schere

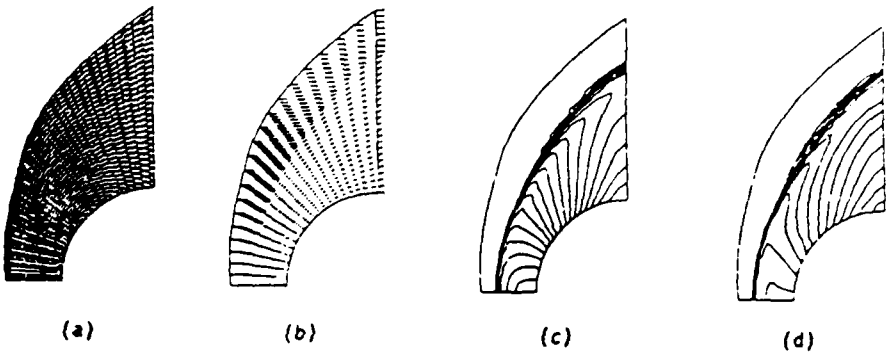


Figure 1

Mach 8 flow past a cylinder. (a) Mesh (b) Velocity vectors (c) Pressure contours (d) Density contours.

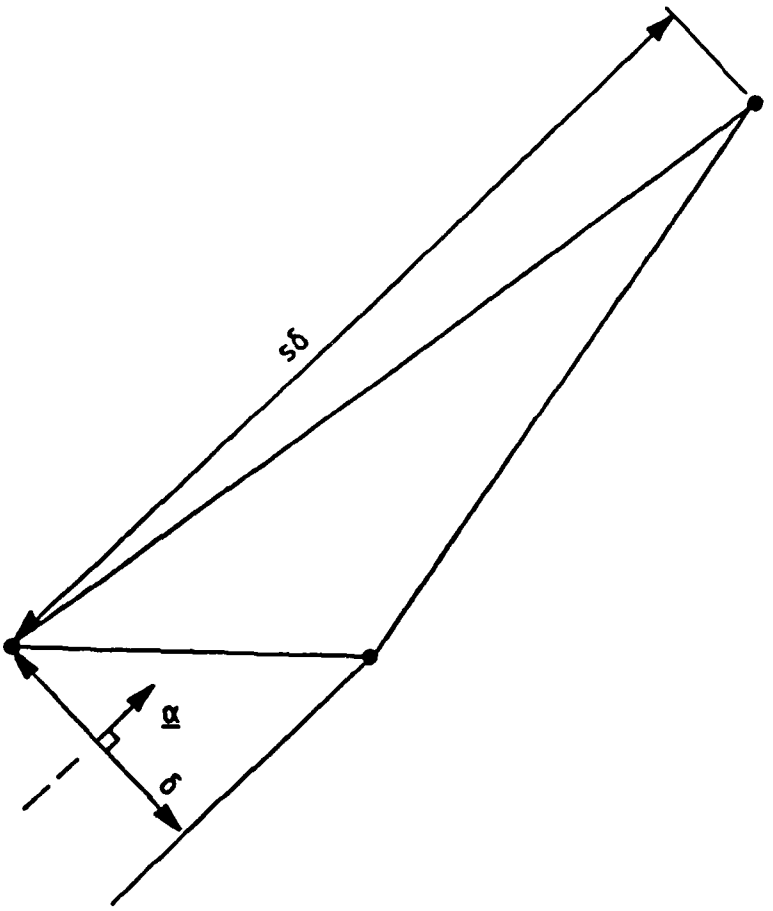


Figure 2

Definition of the mesh parameters δ, s and α.

should give monotonic results for the problem of interest.

The solution method of equation (4) will be used as a high order scheme and a high order solution, $\underline{U}^H$, after an time step can be defined by

$$\underline{U}^H = \underline{U}^m + \underline{\delta U}^H \tag{6}$$

Similarly, a low order solution, $\underline{U}^L$, is defined by

$$\underline{U}^L = \underline{U}^m + \underline{\delta U}^L \tag{7}$$

where the low order increment is calculated as

$$\underline{\delta U}^L = \underline{\delta U}^H + \underline{D} \tag{8}$$

and the smoothing term $\underline{D}$ is given by

$$\underline{D} = C^L \underline{M}_\ell^{-1} (\underline{M} - \underline{M}_\ell) \underline{U}^m \tag{9}$$

where C^L is a constant. This form for the smoothing is suggested by the fact that at node i on a uniform grid in 1D

$$[\underline{M}_\ell^{-1} (\underline{M} - \underline{M}_\ell) \underline{U}^m]_i = (U^m_{i+1} - 2\underline{U}^m_i + \underline{U}^m_{i-1})/6 \tag{10}$$

It should be noted that equations (6-8) can be re-arranged to give

$$\underline{U}^H = \underline{U}^L + \underline{D} \tag{11}$$

The new solution is computed according to

$$\underline{U}^{m+1} = \underline{U}^L + \underline{D}^* \tag{12}$$

where $\underline{D}^*$ is obtained by writing the element contributions to $\underline{D}$, in such a way as to attempt to ensure that $\underline{U}^{m+1}$ is free from extrema not found in $\underline{U}^m$ or $\underline{U}^L$ [14, 15].

The numerical performance of this FCT scheme is illustrated in Figure 1 which shows the solutions obtained for the problem of Mach 8 flow past a cylinder.

Further generalizations of FCT are possible which offer interesting possibilities for future investigation [16].

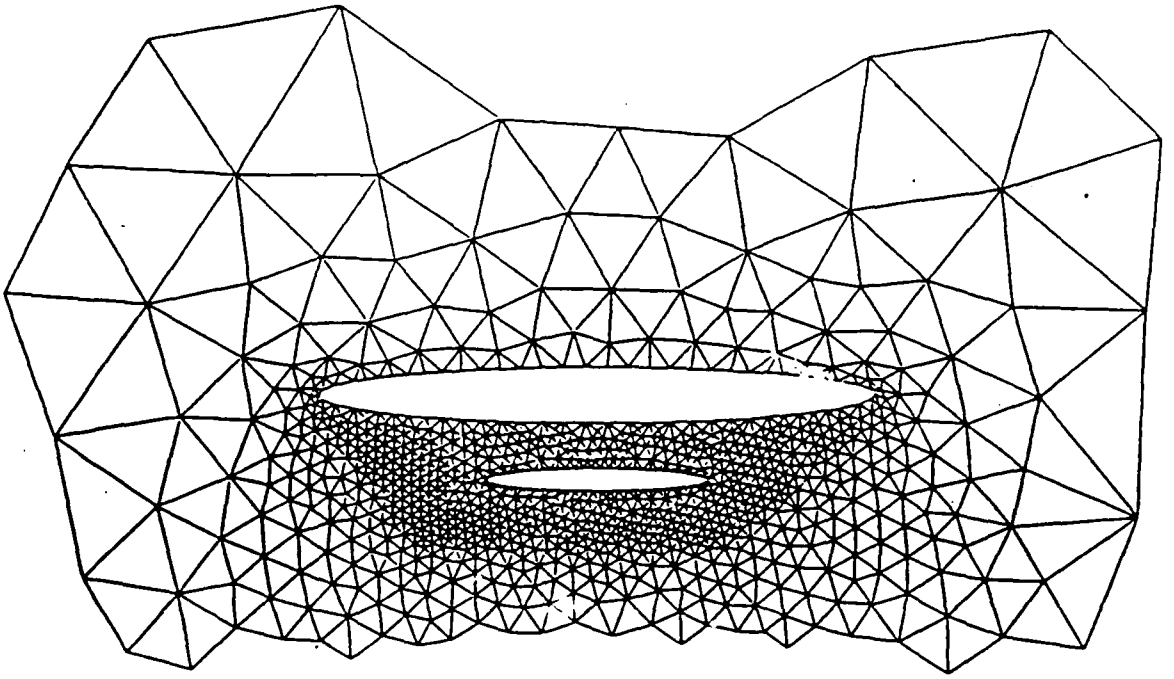


Figure 3

Detail of the initial mesh produced for the analysis of a store separation problem.

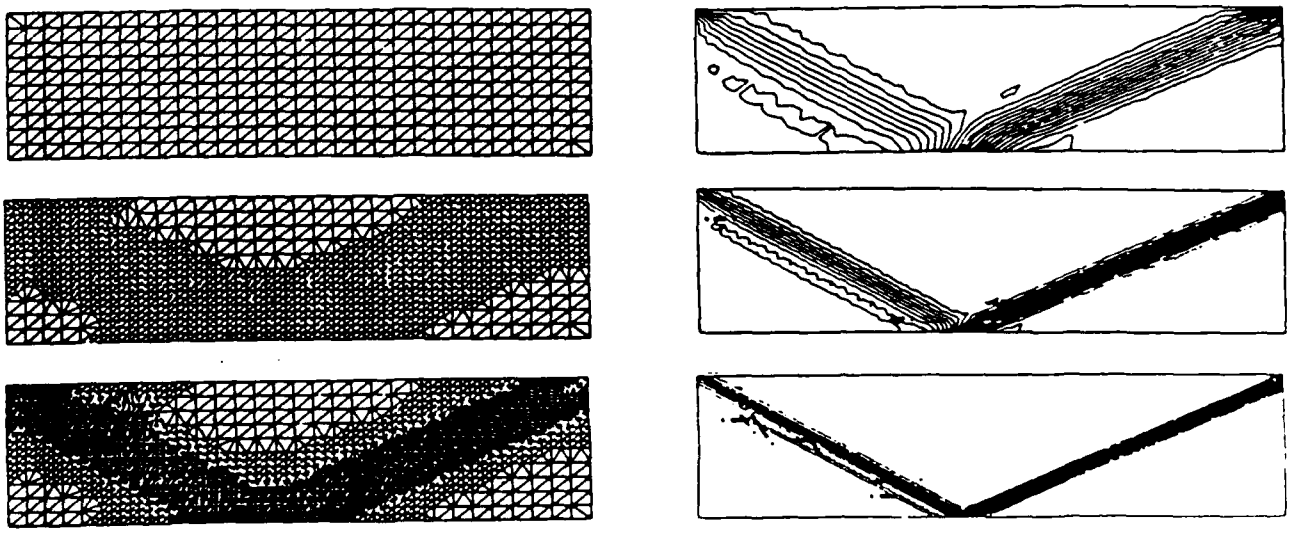


Figure 4

Regular shock reflection at a wall. Sequence of meshes produced using mesh enrichment and the corresponding pressure contours.

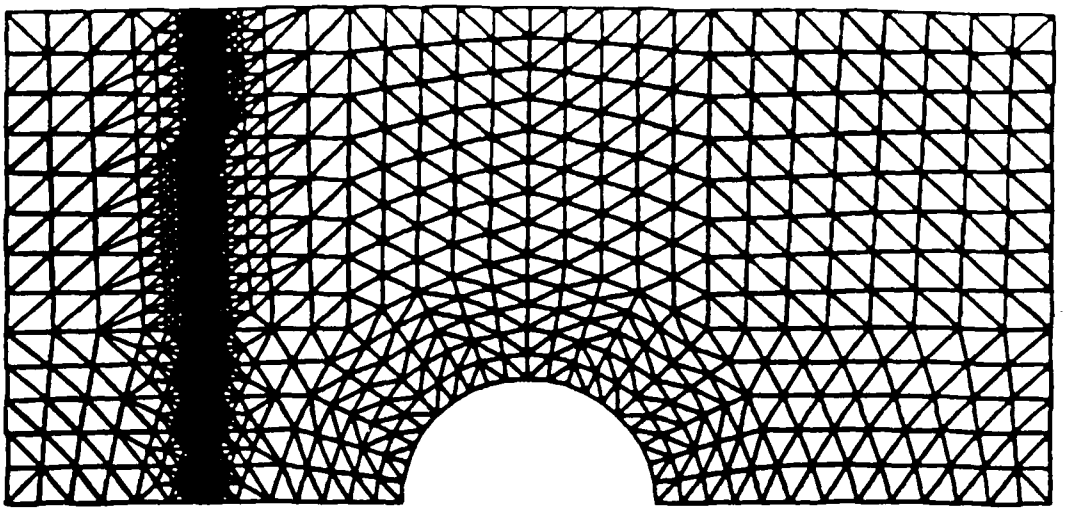
MESH GENERATION

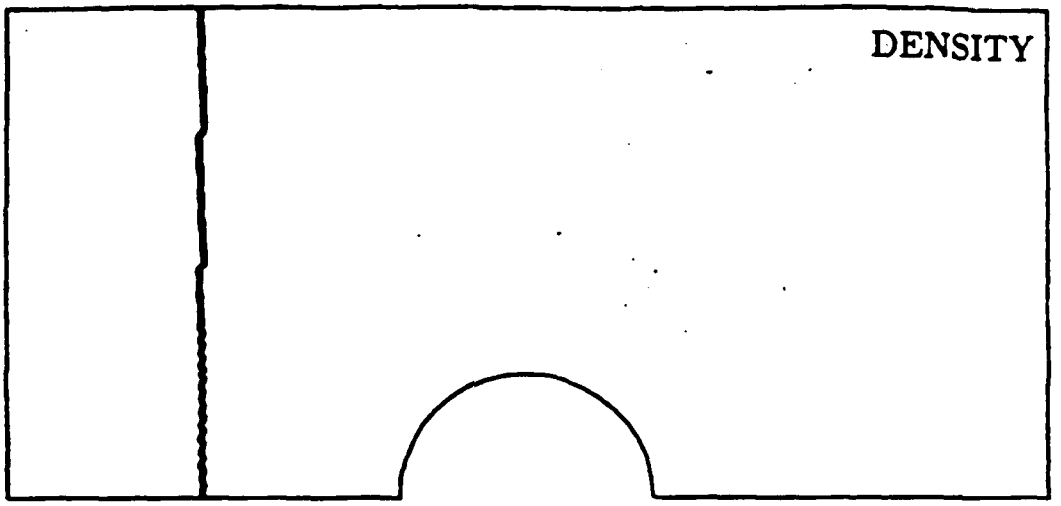
The use of triangular elements in 2D means that computational domains of complex geometrical shape can be readily modelled and a variety of triangular mesh generation algorithms for planar domains are available [17, 18]. The approach to mesh generation to be outlined here begins by defining the boundaries of the solution domain in terms of Bezier polynomials and then covering this domain with a coarse 'background' grid of 3 noded linear triangles. This grid is normally constructed by hand and the only geometrical requirement imposed is that the solution domain should be completely covered by this grid i.e. the background grid is not required to approximate the geometry. At each node on the background grid, we specify the values of mesh parameters δ, s, $\underline{\alpha}$. During the mesh generation process, the local values of these parameters for the mesh being generated will be obtained by interpolation over the background grid. For the exact definition of these mesh parameters, it is useful to refer to Figure 2 which shows a typical generated triangle. The triangle has length $s\delta$ in the direction of $\underline{\alpha}$ and length δ in the direction at right angles to $\underline{\alpha}$. We call δ the local node spacing, s the local degree of stretching and $\underline{\alpha}$ the local direction of stretching. The full flexibility of the mesh generator need not be used to construct an initial mesh for a given problem, but it will be used in an adaptive mesh process to be described later. In particular, if a uniform distribution of δ is required, with no stretching, the background grid need only consist of a single element. The mesh generation process begins with the placing of the boundary nodes. The lines joining successive boundary nodes form the initial generation 'front', which is the collection of sides available to construct triangles. One side in the front is chosen, and a triangle is constructed with values of δ, s and $\underline{\alpha}$ interpolated from the background grid. The front is updated and the process is repeated until the front is empty, at which stage the whole solution domain has been discretized. Full details of the mesh generation process can be found elsewhere [19].

The performance of the mesh generator is demonstrated in Figure 3, which shows a detail of the initial mesh produced for the analysis of a store separation problem. This problem has been used to demonstrate the full power of the generator by directly coupling it to the transient solution procedure and using it to locally regenerate the mesh as the store moves through the flow field [20].

ADAPTIVE MESH STRATEGIES

Adaptive mesh strategies have a major role to play in the development of efficient solution techniques for large problems in CFD. The ultimate objective is the ability to solve a given problem to a prescribed accuracy with the optimum number of grid points and, although this goal has not yet been met, major steps





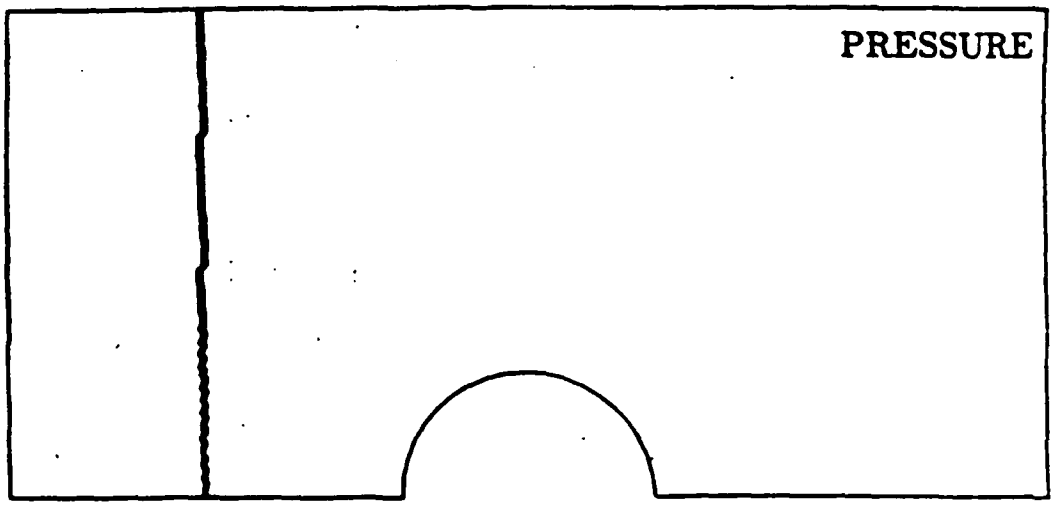
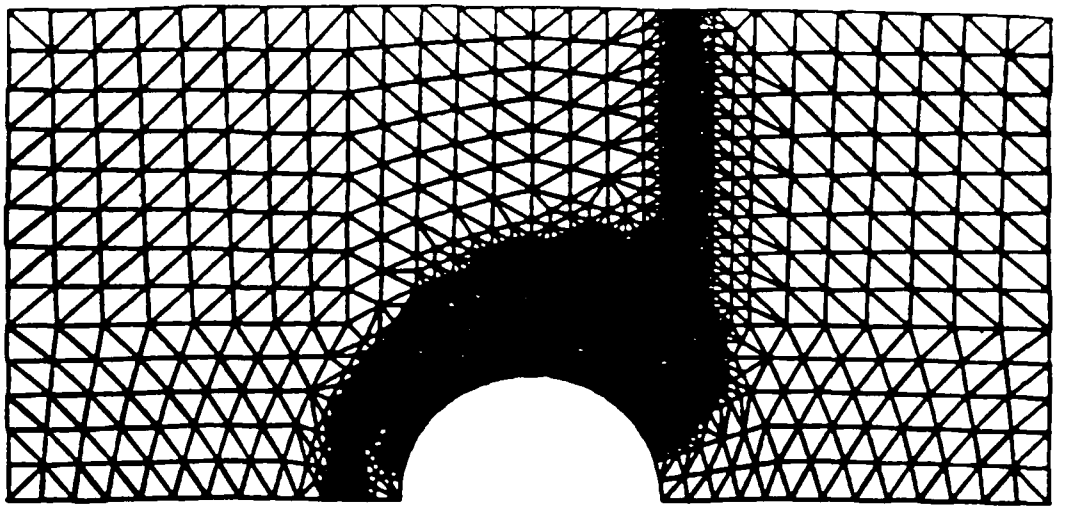
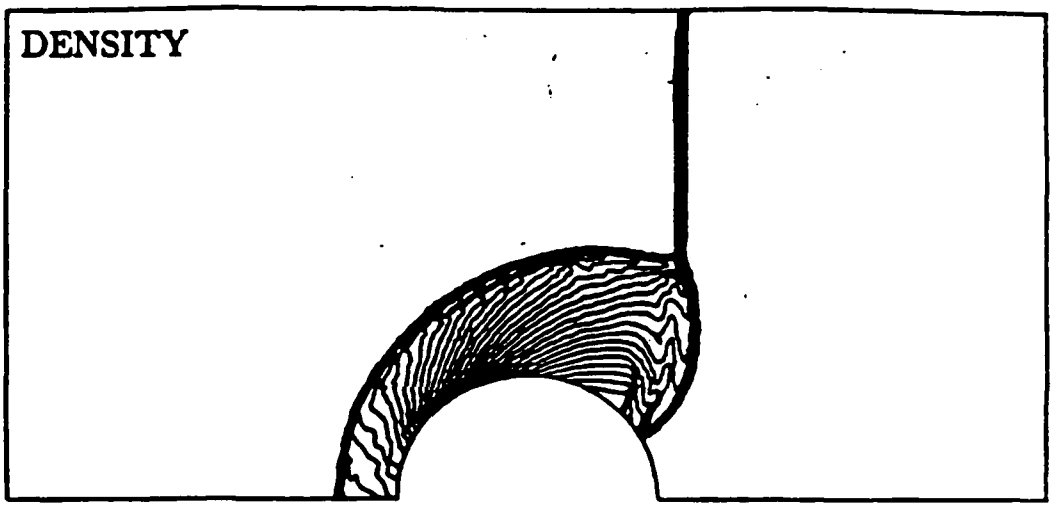


Figure 5

Shock impinging on a half-cylinder
(a) t=0, 2554 elements, 1335 points





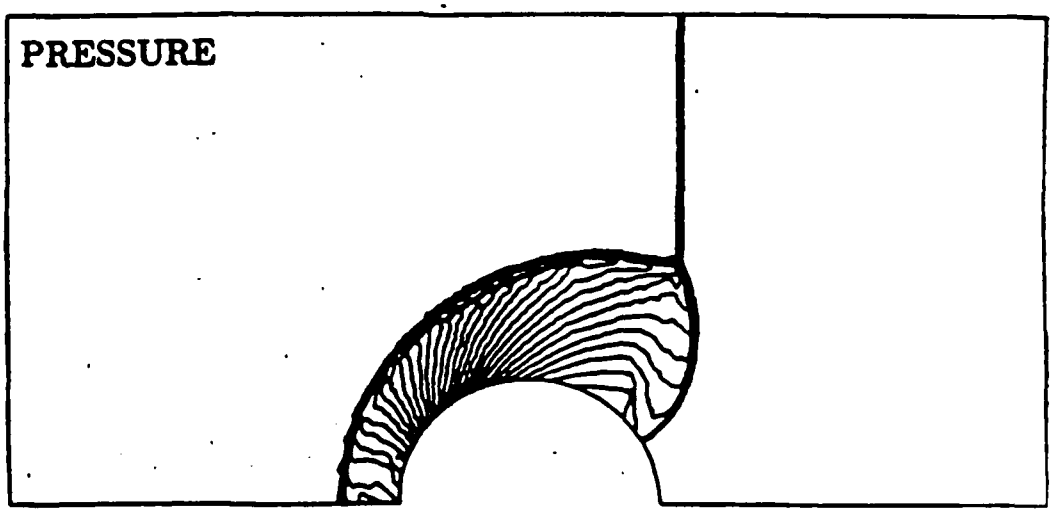
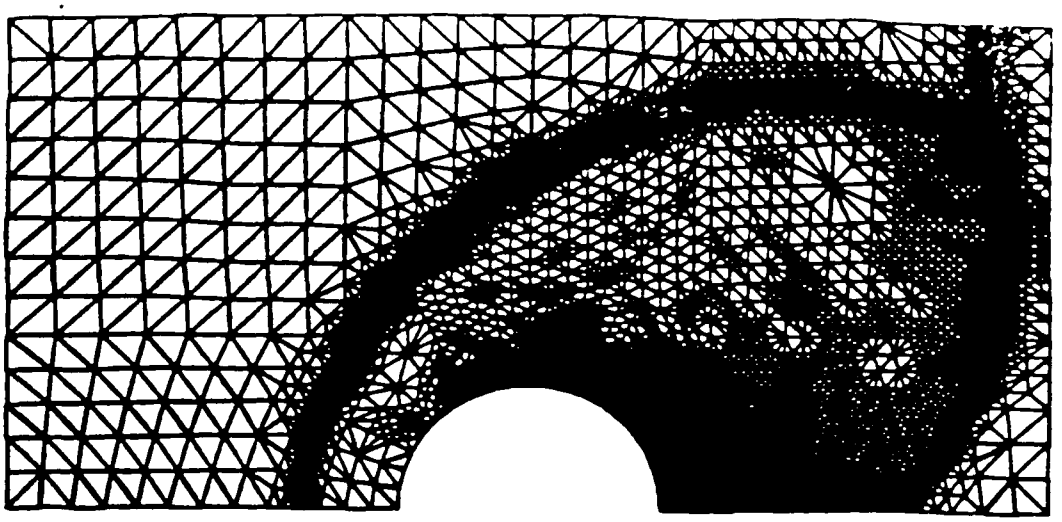
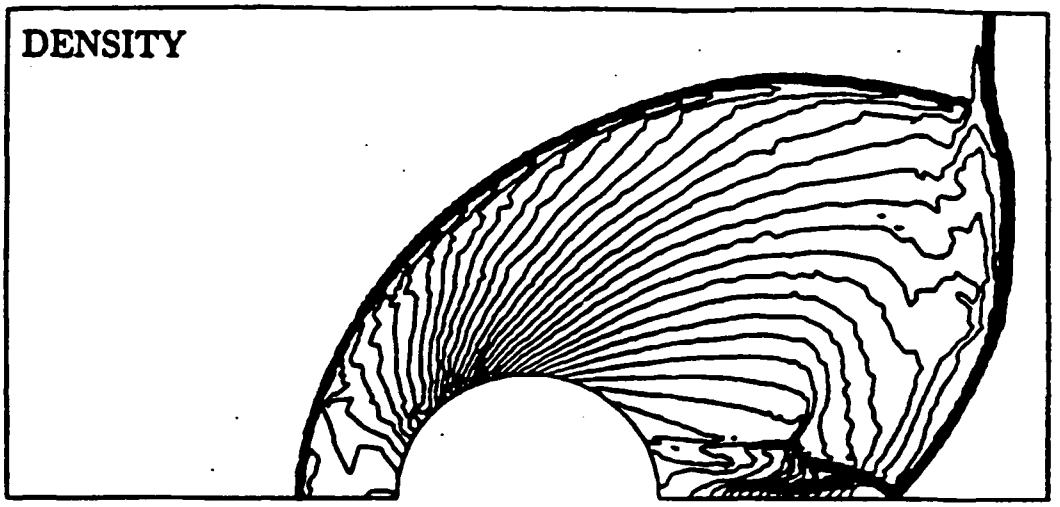


Figure 5 (cont)

(b) t=0.3, 9057 elements, 4626 points





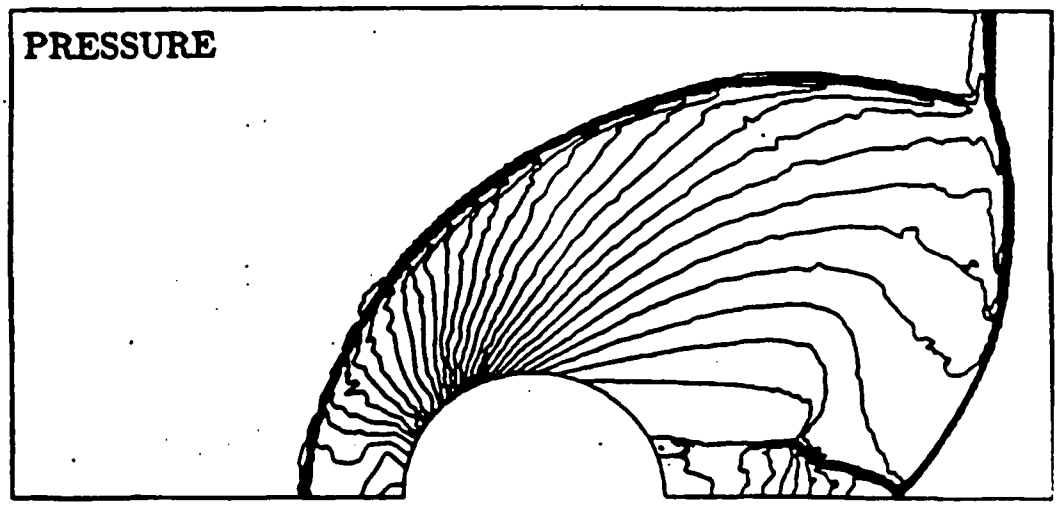


Figure 5 (cont)

(c) t=0.5, 12020 elements, 6129 points

Mesh movement techniques can be contemplated [21, 22] but sufrer from the drawback that the accuracy of the final computation can be limited by the structure and resolution of the initial grid. Mesh enrichment algorithms for steady problems [23, 24] generally advance the solution towards steady state on an initial coarse grid and then obtain an estimation of the error in each computational cell by using an error indicator. For the Euler or Navier-Stokes equation systems, the error indication is normally based upon a key-variable eg. the density is a popular choice for the Euler equations. Indicators based upon interpolation theory can be used, with equi-distribution of the error being the object of the refinement process [25]. The cells exhibiting largest error are automatically subdivided and the computation proceeds, with this process being repeated until the analyst is satisfied with the solution quality.

The mesh enrichment approach works well in practice [7] and Figure 4 shows the solution of problem of regular shock reflection at a wall which has been solved in this manner. This solution was produced using the basic solution algorithm described above.

The extension of the mesh enrichment concepts to the solution of transient problems has been demonstrated recently [26]. Now, as the flow features of interest are moving through the solution domain, for economy of computation mesh enrichment has to be combined with the capability of derefining the mesh in regions where the error indication is small. This work has produced a highly vectorizable algorithm, with low storage requirements, and with the ability to recover the original grid when the flow feature of interest has passed. The performance of the algorithm is shown in Figure 5 which displays the computed FCT solution for the problem of a Mach 10 shock impinging upon a half-cylinder. The solutions are depicted at three selected times during the transient.

A drawback of the mesh enrichment approach is that it provides a uniform local mesh refinement, whereas many flow features of interest are essentially one-dimensional in character. This suggests that directional refinement techniques could be computationally more efficient and work in this area has already begun. If element error indicators are replaced by indicators along element sides [27], directional refinement for steady problems can be achieved, along with derefinement. This is illustrated in Figure 6 which again shows the solution obtained for the problem of regular shock reflection at a wall using the basic solution algorithm. An alternative approach, is to use the mesh generator described earlier to regenerate the mesh based upon information provided by the computed solution on the current mesh [19]. Figure 7 shows the problem of Mach 25 flow past a blunt body at an angle of attack of 20° which has been solved using FCT in this manner with a sequence of three grids.

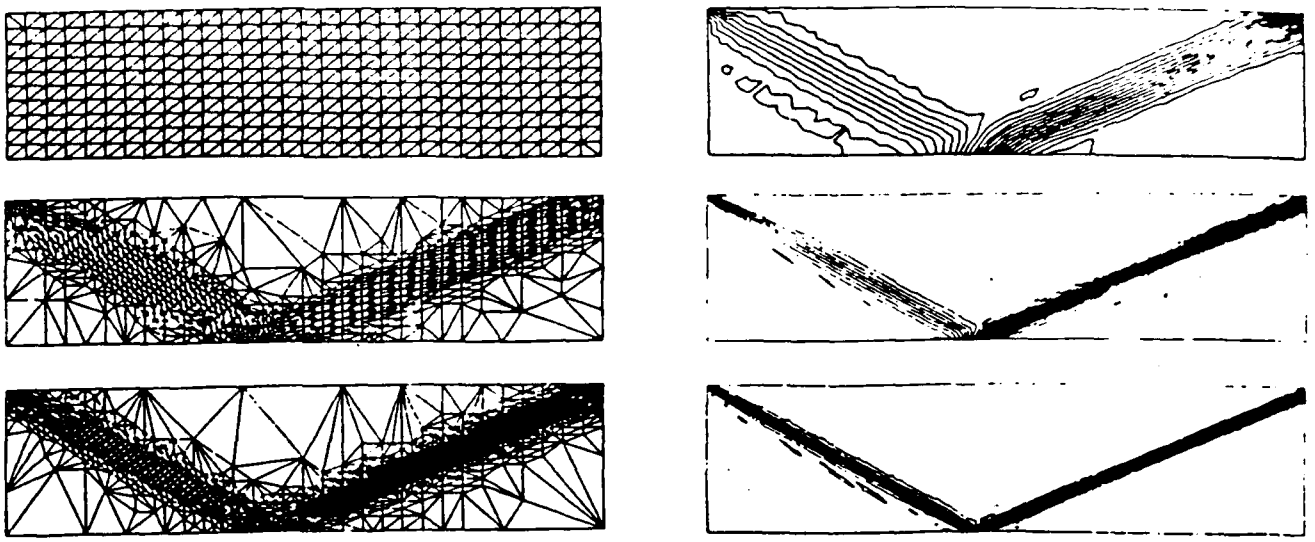


Figure 6

Regular shock reflection at a wall. Sequence of meshes produced using directional refinement and the corresponding pressure contours.

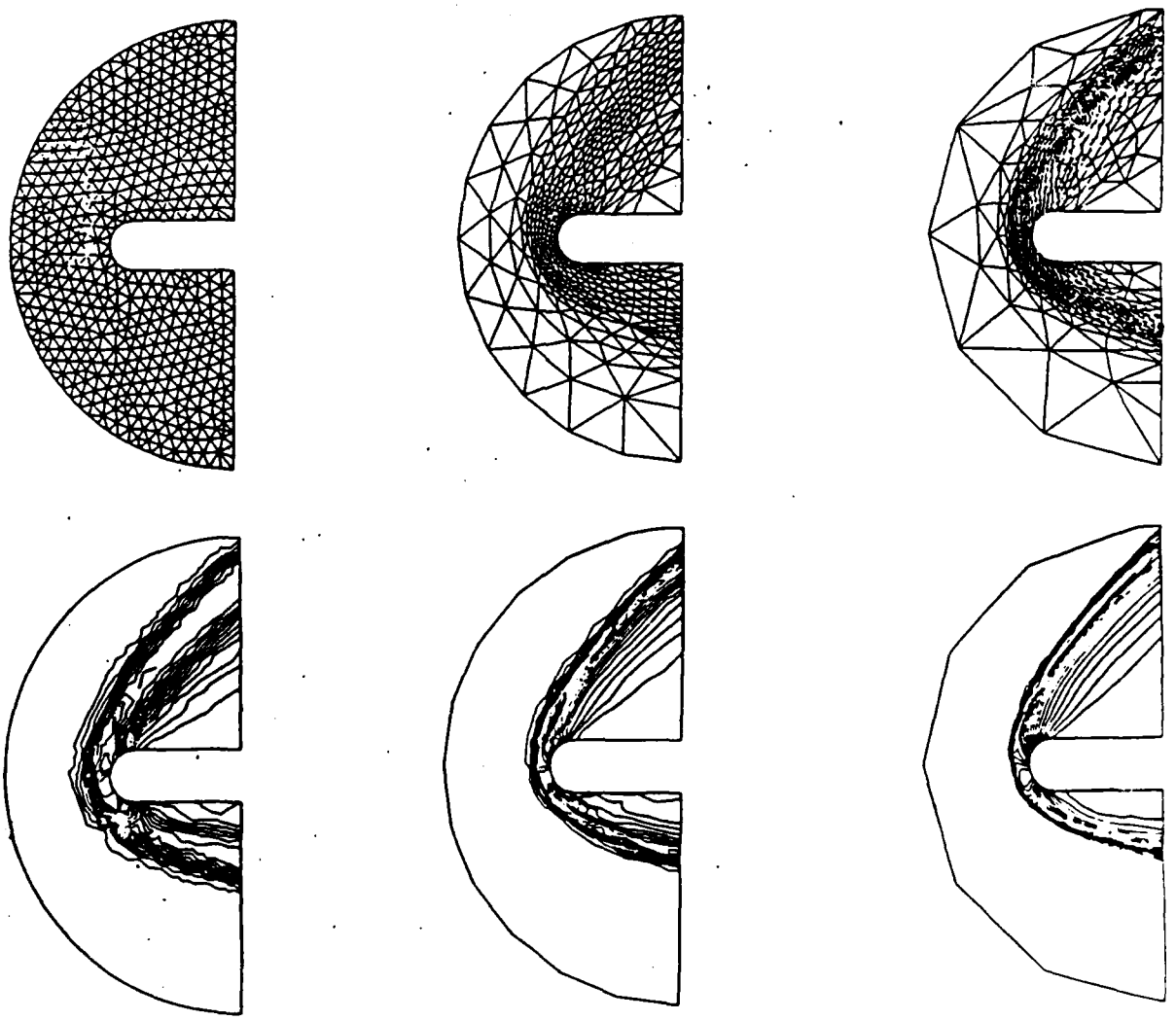


Figure 7

Mach 25 flow past a blunt body at 20° angle of attack. Sequence of meshes produced by adaptive mesh regeneration and the corresponding density

ACKNOWLEDGEMENTS

The authors would like to thank the Aerothermal Loads Branch of the NASA Langley Research Center for partial support of this work under Grant No. NAGW-478 and especially A. R. Wieting and K. S. Bey for their continued interest and encouragement. We would also like to acknowledge fruitful discussions with O. C. Zienkiewicz, J. P. Boris, D. L. Book and S. T. Zalesak.

REFERENCES

1. R. Löhner, K. Morgan, J. Peraire and O. C. Zienkiewicz, "Finite Element Methods for High Speed Flows," AIAA-85-1531-CP, 1985.

2. R. Löhner, K, Morgan, J. Peraire, O. C. Zienkiewicz and L. Kong, "Finite Element Methods for Compressible Flow," in Numerical Methods for Fluid Dynamics (Edited by K. W. Morton and M. J. Baines), Oxford University Press, 1986.

3. S. T. Zalesak, "Fully Multidimensional Flux Corrected Transport Algorithm for Fluids," J. Comp. Phys., 31, 335-362, 1979.

4. J. P. Boris and D. L. Book, "Flux Corrected Transport I: SHASTA, A Fluid Transport Algorithm that Works," J. Comp. Phys., 11, 38-69, 1973.

5. J. Donea, "A Taylor-Galerkin Method for Convective Transport Problems," Int. J. Num. Meth. Engng., 20, 101-119, 1984.

6. O. C. Zienkiewicz and K. Morgan, "Finite Elements and Approximation," Wiley, New York, 1983.

7. R. Löhner, K. Morgan and O. C. Zienkiewicz, "An Adaptive Finite Element Procedure for Compressible High Speed Flows," Comp. Method Appl. Mech. Engng., 51, 441-464, 1985.

8. J. Donea and S. Giuliani, "A Simple Method to Generate High-Order Accurate Convection Operators for Explicit Schemes Based on Linear Finite Elements," Int. J. Num. Meth. Fluids, 1, 63-79, 1981.

9. R. Löhner, K Morgan and O. C. Zienkiewicz, "The Use of Domain Splitting With An Explicit Hyperbolic Solver," Comp. Meth. Appl. Mech. Engng., 45, 313-329, 1984.

10. S. Z. Burstein, "Finite Difference Calculations of Hydrodynamic Flows Containing Discontinuities," J. Comp. Phys., 2, 198-222, 1967.

11. R. Löhner, K. Morgan and J. Peraire, "A Simple Extension to Multidimensional Problems of the Artificial Viscosity Due to Lapidus," Comm. Appl. Num. Methods 1, 141-147, 1985.

12. G. Erlebacher, "Solution Adaptive Triangular Meshes with Application to the Simulation of Plasma Equilibrium," Ph.D. Thesis, Columbia University, 1984.

13. A. K. Parrott and M. K. Christie, "FCT Applied to the 2D Finite Element Solution of Tracer Transport by Single Phase Flow in a Porous Medium," in Numerical Methods for Fluid Dynamics (Edited by K. W. Morton and M. J. Baines), Oxford University Press, 1986.

14. R. Löhner, K. Morgan, M. Vahdati, J. P. Boris and D. L. Book, "FEM-FCT: Combining Unstructed Grids with High Resolution," University College of Swansea, Report C/R/539/86, 1986.

15. K. Morgan, R. Löhner, J. R. Jones, J. Peraire and M. Vahdati, "Finite Element FCT for the Euler and Navier-Stokes Equations," in the Proceedings of the 6th International Symposium on Finite Element Methods in Flow Problems, INRIA, Paris, 89-95, 1986.

16. R. Löhner, K. Morgan and J. Peraire, "Further Generalizations of FCT," to be Submitted to Comm. Appl. Num. Meth., 1986.

17. J. C. Cavendish, "Automatic Triangulation of Arbitrary Planar Domains for the Finite Element Method," Int. J. Num. Meth. Engng., 8, 679-696, 1974.

18. S. H. Lo, "A New Mesh Generation Scheme for Arbitrary Planar Domains," Int. J. Num. Meth. Engng., 21, 1403-1426, 1985.

19. J. Peraire, M. Vahdati, K. Morgan and O. C. Zienkiewicz, "Adaptive Remeshing for Compressible Flow Computations," University College of Swansea, Report CR/R/544/86.

20. L. Formaggia, J. Peraire and K. Morgan, "Simulation of a Store Separation Using the Finite Element Method," Submitted to Comp. Meth. Appl. Mech. Engng., 1986.

21. P. A. Gnoffo, "A Finite Volume Adaptive Grid Algorithm Applied to Planetary Entry Flowfields," AIAA J., 21, 1249-1254, 1983.

22. R. Löhner, K. Morgan and O. C. Zienkiewicz, "Adaptive Grid Refinement for the Compressible Euler Equations," in Accuracy Estimates and Adaptivity for Finite Elements (Edited by I. Babuska et al), J. Wiley and Sons, 1986.

23. J. F. Dannenhoffer and J. R. Baron, "Grid Adaptation for the 2D Euler Equations," AIAA Paper 85-0484, 1985.

24. F. Angrand, V. Billey, A. Dervieux, J. Periaux, C. Pouletty and B. Stouflet, "2D and 3D Euler Flow Calculations with Second Order Accurate Galerkin Finite Element Method," AIAA Paper 85-1706, 1985.

25. J. T. Oden, "Notes on Grid Optimization and Adaptive Methods for Finite Elements," University of Texas at Austin, TICOM Report, 1983.

26. R. Löhner, "An Adaptive Finite Element Scheme for Transient Problems in CFD," Submitted to Comp. Meth. Appl. Mech. Engng., 1986.

27. R. Löhner and K. Morgan, "Improved Adaptive Refinement Strategies for Finite Element Aerodynamic Computations," AIAA-86-0499, 1986.

APPENDIX T.

An Adpative Finite Element Scheme for Transient Problems in CFD (submitted to Comp. Meth. Appl. Mech. Eng.)

AN ADAPTIVE FINITE ELEMENT SCHEME

FOR TRANSIENT PROBLEMS IN CFD

R. Löhner
Berkeley Research Associates
Springfield, VA 22150, USA

1. Abstract

An adaptive finite element scheme for transient problems is presented. The classic h-enrichment/coarsening is employed in conjunction with a triangular finite element discretization in two dimensions. A mesh change is performed every n timesteps, depending on the Courant-number employed and the number of 'protective layers' added ahead of the refined region. In order to simplify the refinement/coarsening logic and to be as fast as possible, only one level of refinement/coarsening is allowed per mesh change. A high degree of vectorizability has been achieved on the CRAY-XMP-12 at NRL. Several examples involving shock-shock interactions and the impact of shocks on structures demonstrate the performance of the method, indicating that considerable savings in CPU-time and storage can be realized even for strongly unsteady flows.

2. Introduction

The solution of large-scale transient problems around complex geometries is a common problem to many fields of computational fluid dynamics. Among the many applications one can envision we mention the impact of shock waves on structures [1], shock-shock interactions [2], detonations [3], the computation of advancing flame fronts [4], the secondary or tertiary recovery in petroleum engineering [5] and bore formations in estuaries [6]. All of these flow fields share a feature common to all advection dominated problems: regions of rapid change are embedded in regions where the flow variables vary smoothly. However, the inaccurate representation of these subregions in the numerical solution may influence large regions of the solution domain, which in turn may deteriorate the overall accuracy obtained. As these subdomains which require a finer gridding are usually not known a priori and/or change position, either a fine grid has to be employed over the whole domain (which for many problems means an excessive overhead in CPU and storage), or adaptive refinement techniques have to be invoked.

For the reasons just mentioned, adaptive refinement techniques are currently receiving increased attention in the literature [7-20]. It is fair to say that for steady state problems very robust and elaborate schemes are available (see, for example [13]). As the refinement is performed only as the solution reaches steady state, CPU and storage requirements for the mesh adaptation in this case do not play important roles. This in turn means that one has the freedom to develop more elaborate mesh refinement schemes, e.g. those that refine the grid in only one direction where needed [16].

On the other hand, if transient problems are considered, the following requirements must be met:

- As the grid adaptation has to be performed many times, the adaptation algorithm must be fast , and therefore must lend itself to vectorization/parallelization.

- As the grid adaptation process becomes an integral part of any code, the algorithm should not be storage intensive.

- As the feature that has been refined may pass again (e.g. a shock reflection), the original grid should be recovered after the feature has passed.

This in turn implies that:

- No directional refinement [16] can be used, as these schemes appear as too storage and CPU-intensive.

- Classic h-enrichment/coarsening must be employed, as it does not require a major storage overhead and due to its simplicity lends itself easily to vectorization.

- Only one level of refinement/coarsening is employed per 'mesh change' in order to minimize the logic involved and thus CPU-requirements.

- For triangles, successive subdivision of a triangle into two (see Figure 1) has to be avoided. This in turn reduces the number of refinement cases considerably.

3. An Error Indicator

Many possible error indicators have been suggested in the literature (see [7-20] and many other publications), and numerical experience indicates that all perform similarly well. However, in the present context, the following requirements must be met:

- The error indicator must be fast.

- As the feature may move only very slowly or come to a standstill (e.g. a shock entering a very dense region), the error indicator must also be reliable for steady state applications.

- As systems of equations are solved, and more than one 'key-variable' [11] may be employed, the error indicator should be dimensionless.

- In order to be applicable to a large class of problems, the error estimator should be bounded (independently of the solution), so that preset refinement/coarsening tolerances can be employed.

In order to meet these requirements, a modified form of the classic interpolation estimates [19] used for steady state computations [14-18,20] has been adopted. These estimators make use of an appropriate seminorm for the detection of those regions which need further refinement or coarsening, e.g. the H2-seminorm [15,19,20]

$$\|u - u^h\|_0 \leq c \cdot h^2 \cdot |u|_2, \qquad (1)$$

where u denotes the exact and u^h the approximate solution, c is a mesh-size-independent constant, h is the characteristic mesh size, and

$$|u|_2 = \sqrt{\int_\Omega \sum_{i,j} \left[\frac{\partial^2 u}{\partial x^i \partial x^j}\right]^2 d\Omega} \tag{2}$$

Second derivatives are justified here because the shape functions used in the finite element discretization are linear. Numerically, we first evaluate the second derivatives at the nodes via a variational statement [22], and then approximate the integral (2) 'conservatively' by taking at element level the maximum second derivative at the nodes. For linear elements of constant length h in 1-D, at the nodes one obtains:

$$e_i = h^{-2} \cdot |U_{i+1} - 2 \cdot U_i + U_{i-1}|. \tag{3}$$

The modified error indicator is given by:

$$E_i = \frac{|U_{i+1} - 2 \cdot U_i + U_{i-1}|}{|U_{i+1} - U_i| + |U_i - U_{i-1}| + \epsilon\,[|U_{i+1}| + 2 \cdot |U_i| + |U_{i-1}|]} \tag{4}$$

We remark the following properties of this modified error indicator:

- By dividing the second derivative by the 'jumps' (gradients) the 'eating-up' effect in the presence of a very strong shock is avoided (i.e. only the value of the normalized H2-seminorm is of importance, not the magnitude of the H2-seminorm as such).

- Normalizing in this way also has the advantage that the error indicator becomes dimensionless, so that more than one 'key-variable' can be used without encountering dimensioning problems.

- Moreover, the modified error indicator is now bounded ($0 \leq E_i < 1$), so that preset tolerances can be employed (this is of particular importance for transient problems).

- The terms following ϵ are added as a 'noise' filter in order not to refine 'wiggles' or 'ripples' which may appear due to loss of monotonicity. The value for ϵ thus depends on the algorithm chosen to solve the PDEs describing the physical process at hand.

The generalization of this error estimator to multidimensional situations is as follows:

$$E^I = \sqrt{\frac{\sum_{k,l} \left(\int_\Omega N^I_{,k} N^J_{,l}\, d\Omega \cdot U_J\right)^2}{\sum_{k,l} \left(\int_\Omega |N^I_{,k}| \left[|N^J_{,l} U_J| + \epsilon \left(|N^J_{,l}||U_J|\right)\right] d\Omega\right)^2}}, \tag{5}$$

where N^I denotes the shape-function of node I.

After having determined the values of the error indicators in the elements, all elements lying above a preset threshold value CTORE are refined, while all elements lying below a preset threshold value CTODE are coarsened. Protective layers of elements are

added to surround the elements to be refined, so that the 'feature' (e.g. a shock) always travels into an already refined region. The number of protective layers that are added depends on the Courant-number employed and the number of timesteps taken between grid modifications. Usually the refinement is performed every 5-10 timesteps, so that for a Courant-number of $C = 0.2 - 0.4$ one to three protective layers are sufficient.

4. Grid Logic

As described above, we limit the number of refinement/coarsening levels per mesh change to one. Moreover, we only allow refinement of a triangle into two or four and avoid the successive refinement of a triangle into two. This implies that there exist only six possible cases for refinement and three for coarsening. These cases are shown in Figures 1,2.

In order to identify the 'parent' and 'son' elements of any element, six integer locations per element were employed in the present situation. The first three integers store the new three neighbor elements ('sons') of an element that has been subdivided into four (the center element of the four is kept as 'parent'). In the fouth integer the element from which the present element originated (the 'parent' element) is stored, while the fifth integer denotes the side of the 'parent' element this element came from. Finally, in the sixth integer location the refinement level is remembered. These six integer locations per element are sufficient to construct further refinements or to reconstruct the original grid. Observe that no classical tree-structure is employed.

The introduction of further nodes (refinement) is performed by first identifying the sides that require refinement, and then labelling these sides with the new node numbers. By doing this, the introduction of coordinates, values for the unknowns and boundary conditions at the new nodes can be performed independently of the introduction of new elements. In principle, these operations could be performed in parallel.

The whole mesh refinement/coarsening process described can be vectorized to a large extent, particularly if the machine available vectorizes GATHER/SCATTER loops efficiently. On the CRAY-XMP-12 at NRL the present algorithm performs more than 99% of all operations necessary for a mesh change inside long (≥ 50) vectorizable loops. In fact, the CPU-time spent in non-vectorizable loops is of the order of 1%, indicating a high degree of efficiency. The processing rate for the grid adaptation alone is of the order of $100\mu sec$ per grid point per mesh change. Although this figure seems high, one has to bear in mind that the adaptation is performed only every 5-10 timesteps, which means that effectively it is much lower.

5. Numerical Examples

We confine the numerical examples to the computation of highly unsteady compressible flows with shock waves. Other fields of applications are obvious. As the basic hydrodynamics-solver we employ the FEM-FCT code of Löhner et.al. [21], which is capable of reproducing moving and stationary shocks over two elements without loss of

monotonicity. For this class of problems and the algorithm employed it was found that the following choice of refinement/coarsening parameters produced acceptable results:

- refinement tolerance: CTORE=0.3
- coarsening tolerance: CTODE=0.1
- noise parameter : ϵ =0.2
- key variable : density

We also tested other key-variables as error indicators, but it was found that for the class of problems under consideration, i.e. compressible Euler equations, the density gave the best results (pressure is not a good error indicator at contact discontinuities).

Unless otherwise stated this set of parameters was employed for all the examples shown below.

5.1 Circular Blast-wave : The problem statement, as well as the solutions obtained are shown in Figure 3. A quadrant of a cylinder in the lower left hand corner was given a density of 10.0 and a pressure of 40.0, while the rest of the computational region was filled with density 1.0 and pressure 1.0. Because all grid points inside radius 5.1 were disturbed and all gridpoints outside were not, the surface of the cylinder on the finite element grid is not completely circular. The number of refinement levels allowed in this case was NREMX=5 which would correspond to a regular grid of 160*160=25600 points, and the grid was modified every 10 timesteps with two protective layers, while the hydrodynamics code was run at a Courant number of $C = 0.2$. This case was run to test the symmetry or 'circularity' of the numerical solution. Although the numerical solution obtained is not completely symmetric with respect to the 45°-line (the symmetry is lost after fifteen mesh changes), the difference between the residuals of the x and y-momenta is only discernable in the fouth significant digit.

The evolution of the main shock is clearly visible in Figures 3a-3b, and the solution at time T=9.3 is shown in Figure 3c,d. Observe that at T=9.3, the number of gridpoints needed (2400) when adapting the mesh as compared to a uniformly refined grid (25600) is more than a factor of 11. Admittedly, this is a simple example, but it clearly demonstrates the great potential that adaptive refinement offers for a large class of problems. The density for all points in the domain is shown in Figure 3e plotted versus the radial distance from the origin, indicating a nearly perfect symmetry.

5.2 Shock impinging on a half-cylinder: The problem under consideration is shown in Figure 4a. A strong shock ($M_s = 10$) runs onto a half-cylinder, is reflected in part, forms a Mach-stem and produces an expansion behind the cylinder.The number of refinement levels allowed in this case was NREMX=3 (which would correspond, on a uniformly refined grid to NPOIN=28032 points), and the grid was modified every 5 timesteps with only one protective layer, while the hydrodynamics code was again run at a Courant number of $C = 0.2$.

The solutions obtained at different times during the transient are depicted in Figures 4a-4f. The maximum number of gridpoints needed for this computation was

NPOIN=6129, which implies a savings-factor of more than four over a uniformly refined grid.

5.3 Shock impinging on two obstacles: This problem is essentially the same as the previous one, only that the geometry is considerably more involved. It is reproduced here in order to demonstrate not only the effectiveness of the adaptive grid scheme presented, but also of unstructured grids in general, when realistic problems need to be simulated. The two obstacles, as well as the solution at time T=0.0 (a strong shock ($M_s = 10$) coming from the left) are shown in Figure 5a. The number of refinement levels allowed in this case was NREMX=3. Had the grid been refined uniformly, this would correspond to NPOIN=38080 points. The grid was modified every 5 timesteps with one protective layer. The Courant number was again set to $C = 0.2$.

The solutions obtained at different stages during the transient are depicted in Figures 5a-d. Observe the amount of detail that the adaptive refinement procedure is capable of reproducing. The maximum number of points required (corresponding to time T=0.8) was NPOIN=8101, again less than four times what a uniform refinement would have required. During most of the computation, the number of gridpoints is of course much lower (the savings much bigger). As the physics become more involved, more grid points are required, and correspondingly larger portions of the domain are refined.

6. Conclusions

An adaptive finite element scheme for transient problems has been presented. The classic h-enrichment/coarsening is employed in conjunction with a triangular finite element discretization in two dimensions. The grid is adapted every n timesteps, depending on the Courant-number employed and the number of 'protective layers' added ahead of the refined region. Particular emphasis was placed on speed and low storage requirements from the outset. Therefore, only one level of refinement/coarsening was allowed per mesh change, and the subsequent subdivision of a triangle into two was avoided. It has been demonstrated that with these restrictions in mind a high degree of vectorizability can be achieved on modern supercomputers.

An obvious extension of the present work is the development of a similar scheme in three dimensions involving tetrahedrons, and the combination of different types of elements (quads,triangles) in one single code.

7. Acknowledgements

I would like to acknowledge fruitful discussions with Drs. K. Morgan, M. Fritts and J.P. Boris during the development of the schemes presented.

8. References

[1] H.M. Glaz, P. Colella, I.I. Glass and R.L. Deschambault - A Numerical Study of Oblique Shock-Wave Reflections with Experimental Comparisons; Proc.Roy.Soc. Lond. A 398, 117-140 (1985).

[2] M. Fry, J. Tittsworth, A. Kuhl, D.L. Book, J.P. Boris and M. Picone- Transport Algorithms with Adaptive Gridding; Proc. 13th Int.Symp. on Shock Tubes and Waves (C.E. Treanor and J.G. Hall eds.), 376-384 (1982).

[3] K. Kailasanath, E.S. Oran, J.P. Boris and T.R. Young - Determination of Detonation Cell Size and the Role of Transverse Waves in Two-Dimensional Detonations; Combustion and Flame 61, 199-209 (1985).

[4] N. Peters and J. Warnatz (eds.) - Numerical Methods in Laminar Flame Propagation; Vieweg Notes on Numerical Fluid Mechanics Vol. 6, Vieweg (1982).

[5] R.E. Ewing (guest ed.) - Comp.Meth.Appl.Mech.Eng. 47,1-2 (1984).

[6] O.C. Zienkiewicz - private communication.

[7] P. A. Gnoffo - A Finite-Volume, Adaptive Grid Algorithm Applied to Planetary Entry Flowfields; AIAA J.21, 1249-1254 (1983).

[8] K. Nakahashi and G. S. Deiwert - A Three-Dimensional Adaptive Grid Method; AIAA-85-0486 (1985).

[9] M.J. Berger and J. Oliger - Adaptive Mesh Refinement for Hyperbolic Partial Differential Equations; J.Comp.Phys.53,484-512 (1984).

[10] M.D. Smooke and M.L. Koszykowsky - Two-Dimensional Fully Adaptive Solutions of Solid-Solid Alloying Reactions; J.Comp.Phys.62,1-25 (1986).

[11] J.F. Dannenhoffer and J.R. Baron - Adaptive Procedure for Steady State Solution of Hyperbolic Equations; AIAA-84-0005 (1984).

[12] J.F. Dannenhoffer and J.R. Baron - Grid adaptation for the 2-D Euler Equations; AIAA-85-0484 (1985).

[13] J.F. Dannenhoffer and J.R. Baron - Robust Grid adaptation for Complex Transonic Flows; AIAA-86-0495 (1986).

[14] R. Löhner, K. Morgan and O.C. Zienkiewicz - An Adaptive Finite Element Procedure for High Speed Flows; Comp.Meth.Appl.Mech.Eng. 51, 441-465 (1985).

[15] R. Löhner, K. Morgan, and O.C. Zienkiewicz - Adaptive Grid Refinement for the Compressible Euler Equations; Chapter x in Accuracy Estimates and Adaptivity for Finite Elements (Babuska et.al. eds.), J. Wiley and Sons. To appear (1986).

[16] R. Löhner and K. Morgan - Improved Adaptive Refinement Strategies for Finite Element Aerodynamic Computations; AIAA-86-499 (1986).

[17] B. Palmerio, V. Billey, A. Dervieux and J. Periaux - Self-Adaptive Mesh Refinements and Finite Element Methods for Solving the Euler Equations; Proc. ICFD Conf. on Numerical Methods for Fluid Dynamics, Reading, U.K., March 1985.

[18] F. Angrand, V. Billey, A. Dervieux, J. Periaux, C. Pouletty and B. Stoufflet - 2-D and 3-D Euler Flow Calculations with a Second-Order Accurate Galerkin Finite Element Method; AIAA-85-1706 (1985).

[19] J.T. Oden - Notes on Grid Optimization and Adaptive Methods for Finite Element Methods; TICOM Rep.(1983).

[20] J.T. Oden, P. Devloo and T. Strouboulis - Adaptive Finite Element Methods for the Analysis of Inviscid Compressible Flow: I. Fast Refinement/Unrefinement and Moving Mesh Methods for Unstructured Meshes; preprint (1986).

[21] R. Löhner, K. Morgan, M. Vahdati, J.P. Boris and D.L. Book - FEM-FCT: Combining High Resolution with Unstructured Grids; Submitted to J.Comp.Phys. (1986).

[22] O.C. Zienkiewicz and K. Morgan - Finite Elements and Approximation; J. Wiley & Sons (1983).

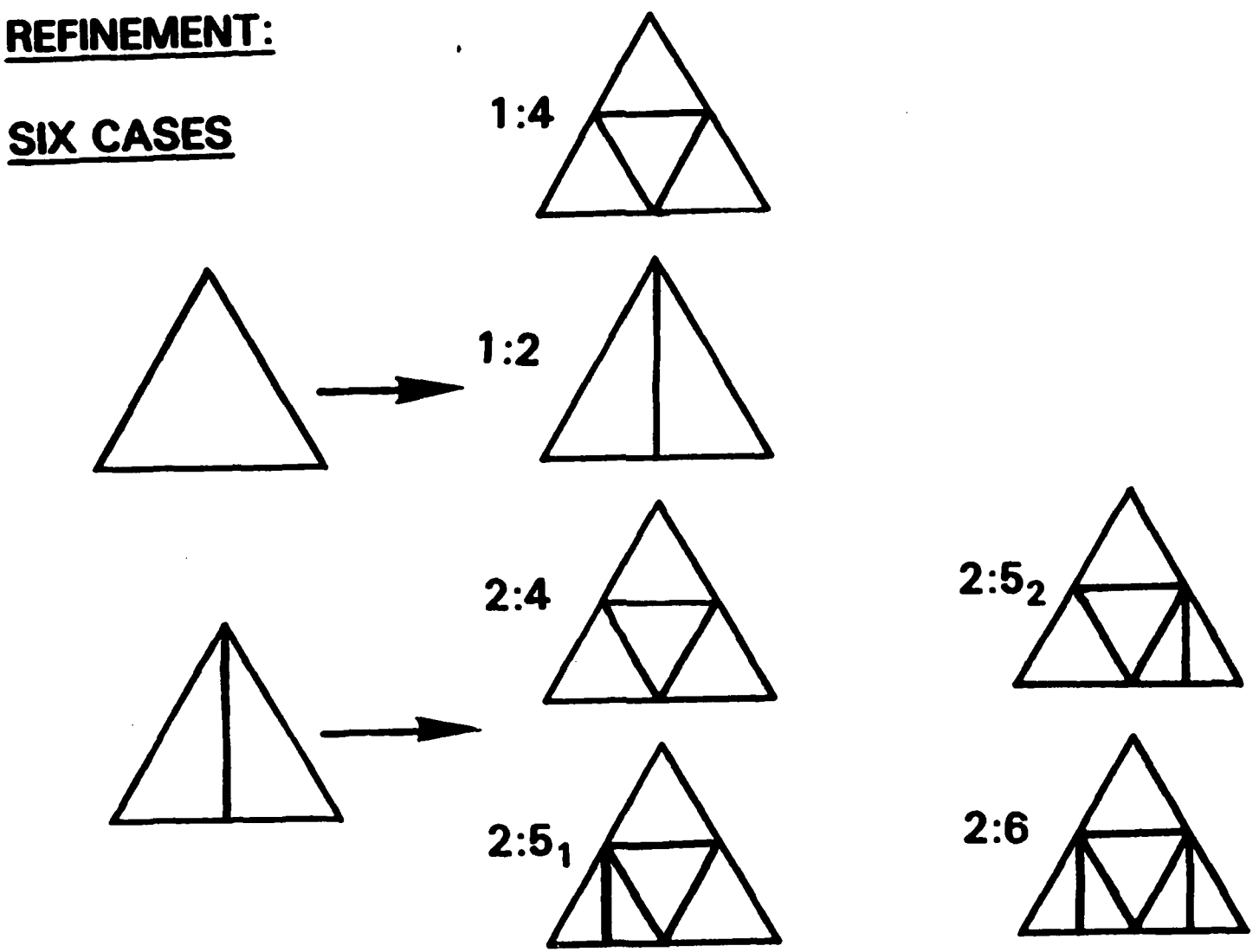


Figure 1: Allowable refinement cases.

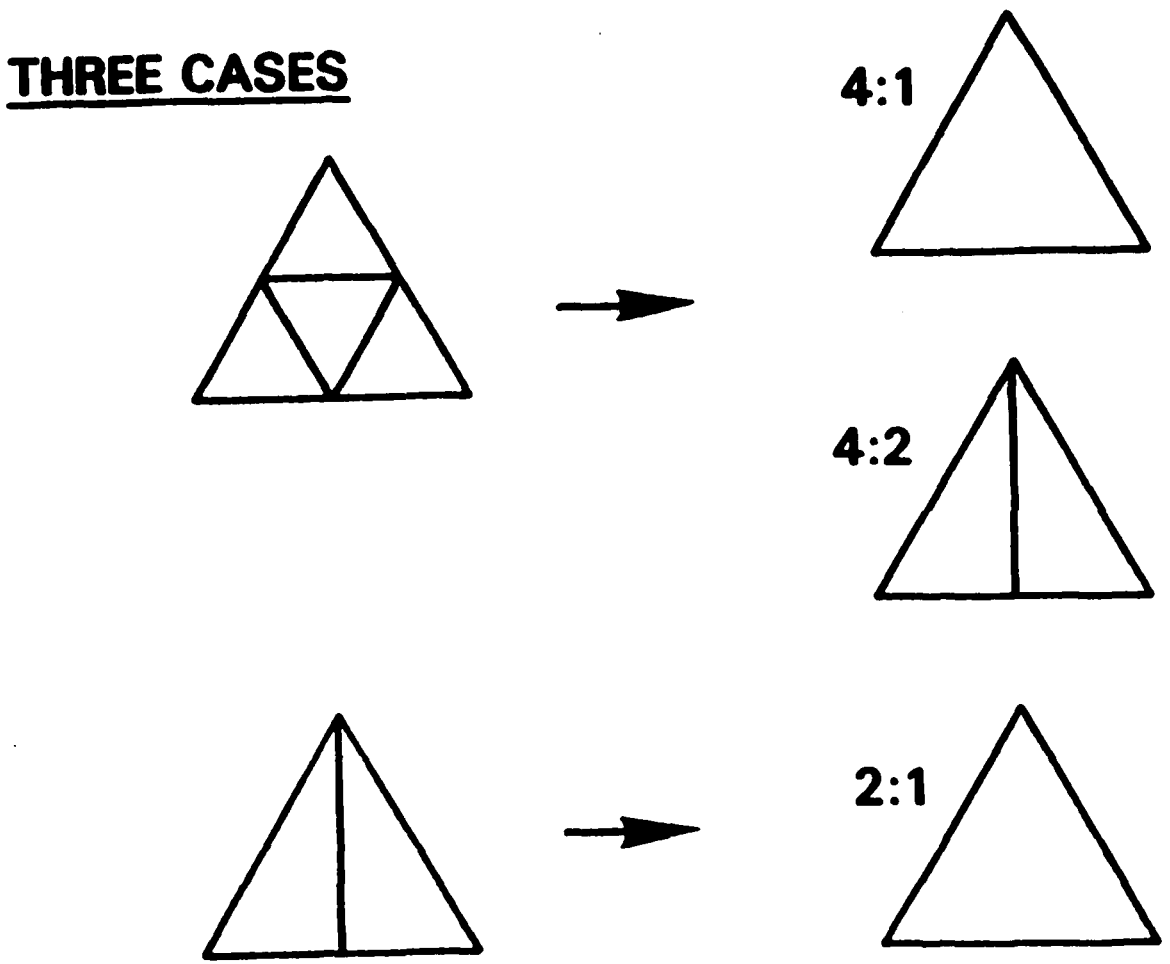
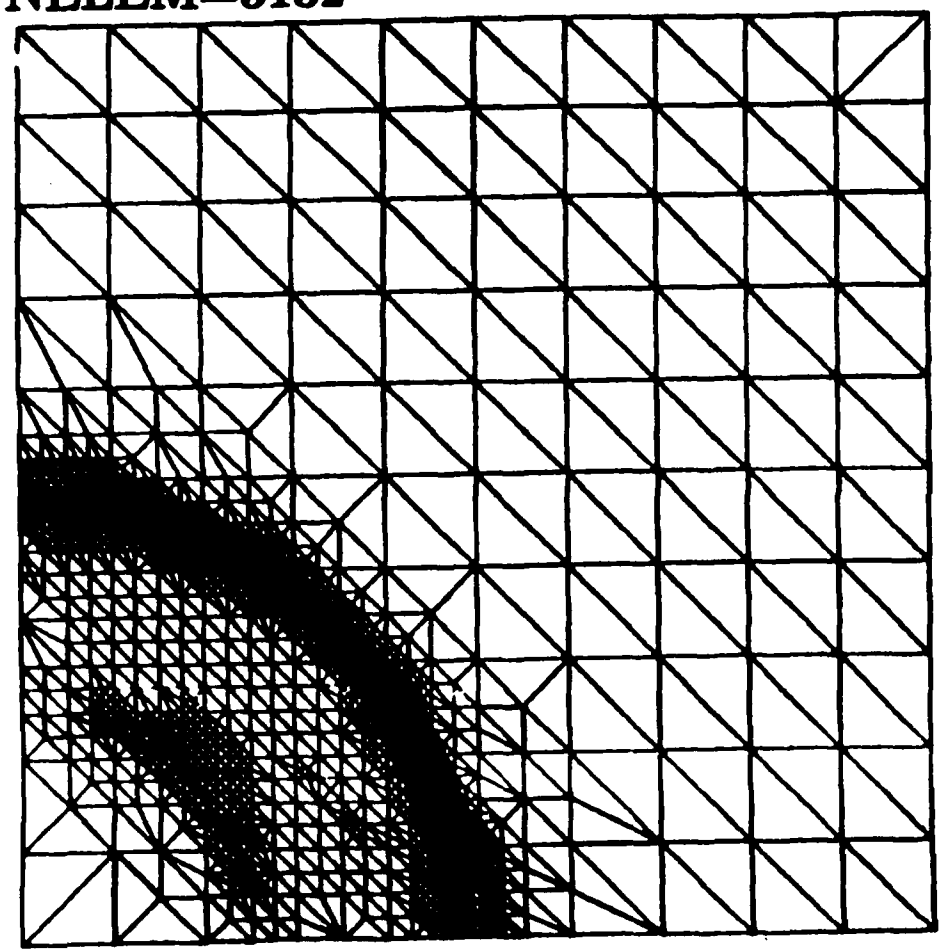


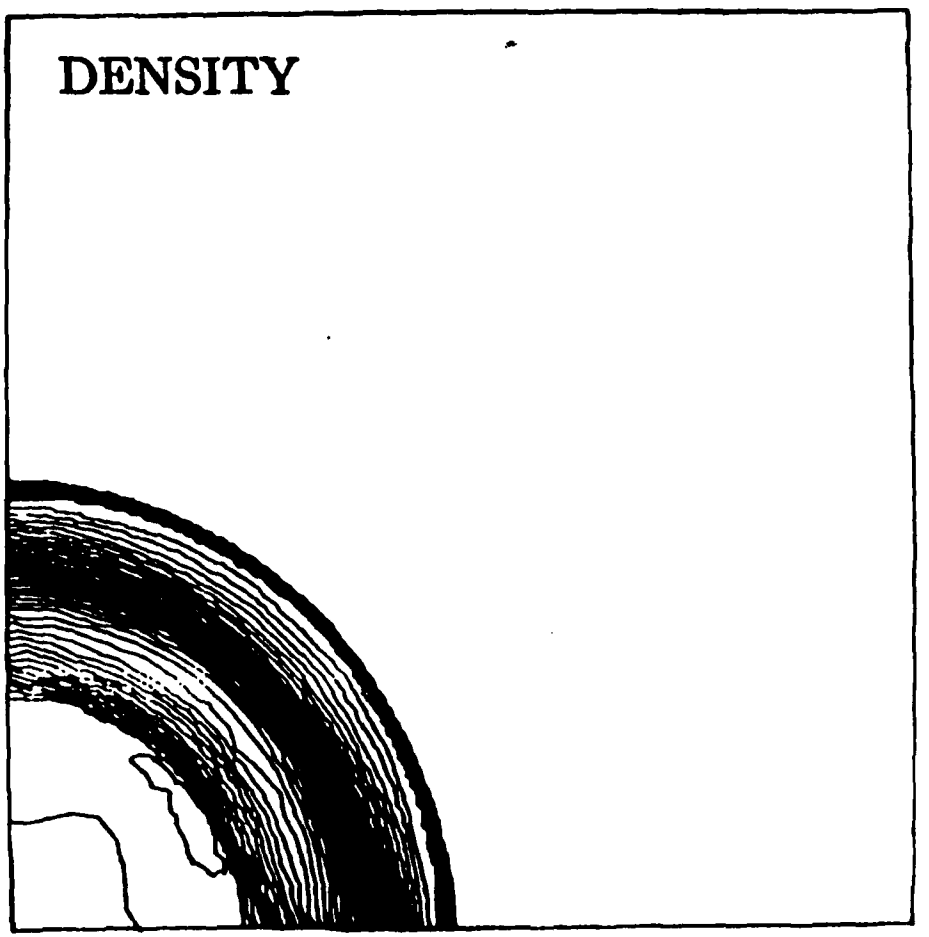
Figure 2: Allowable coarsening cases.

t=5.13

NPOIN=1599

NELEM=3132





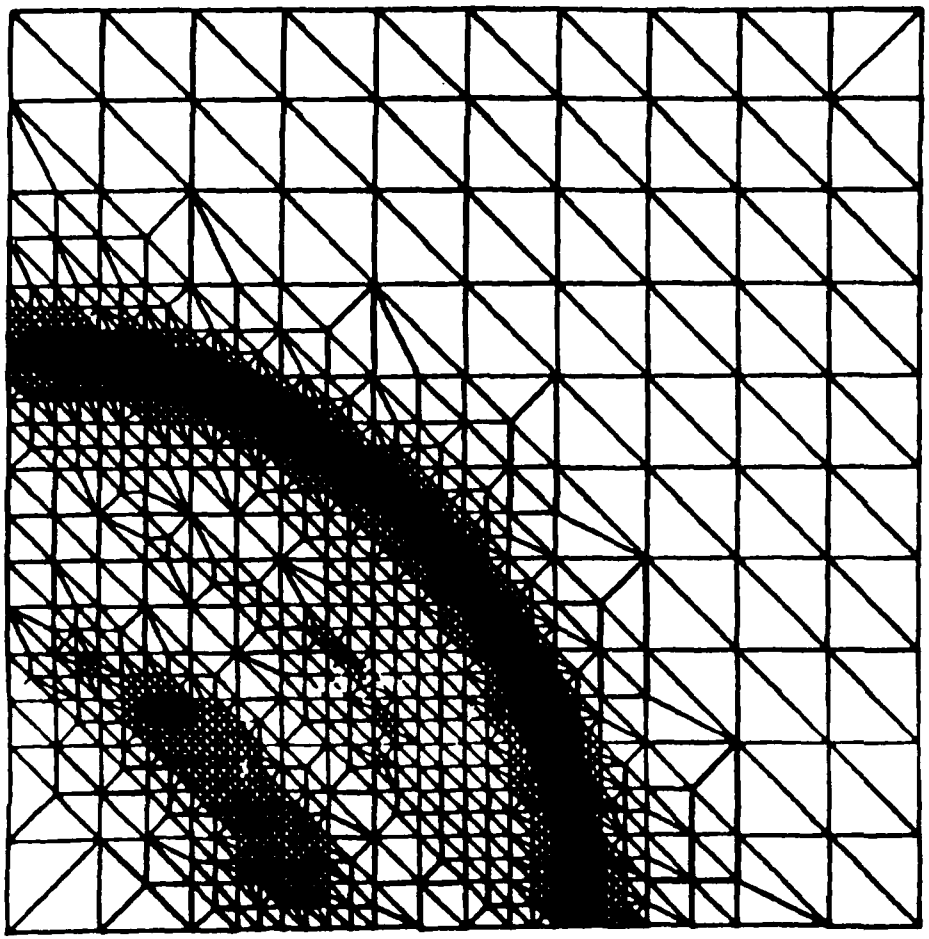
a)

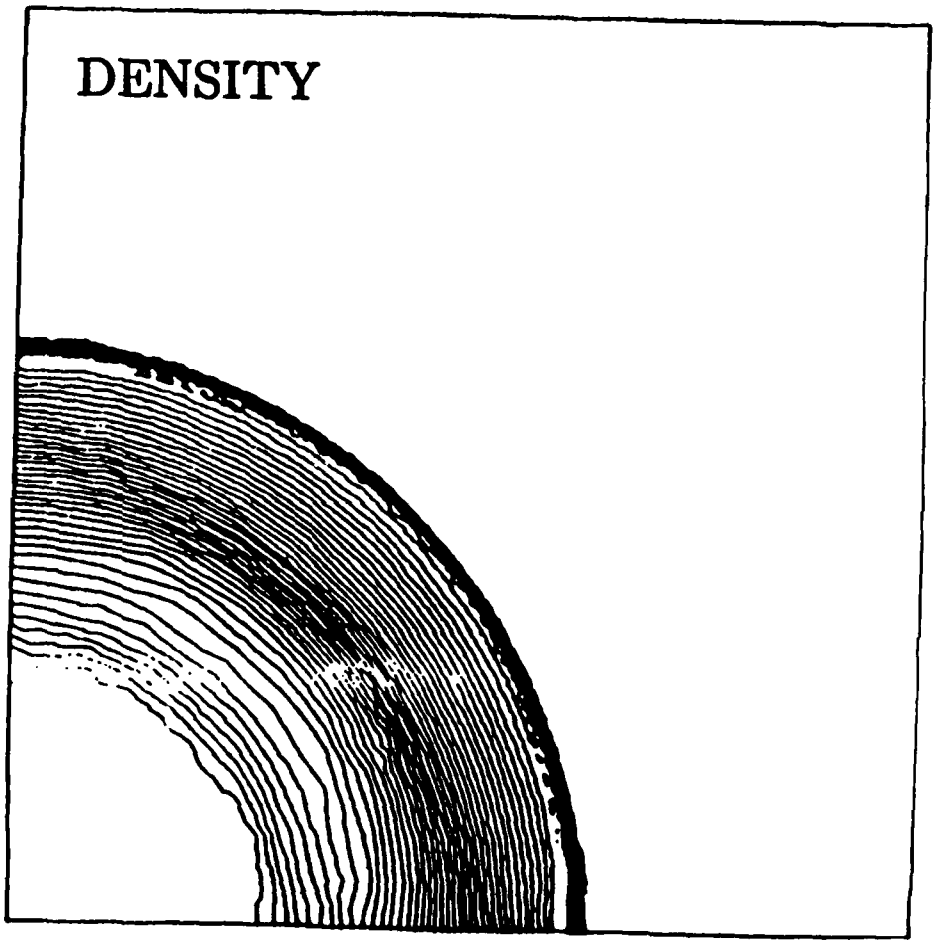
Figure 3: Blast wave propagation.

t=7.68

NPOIN=2267

NELEM=4459





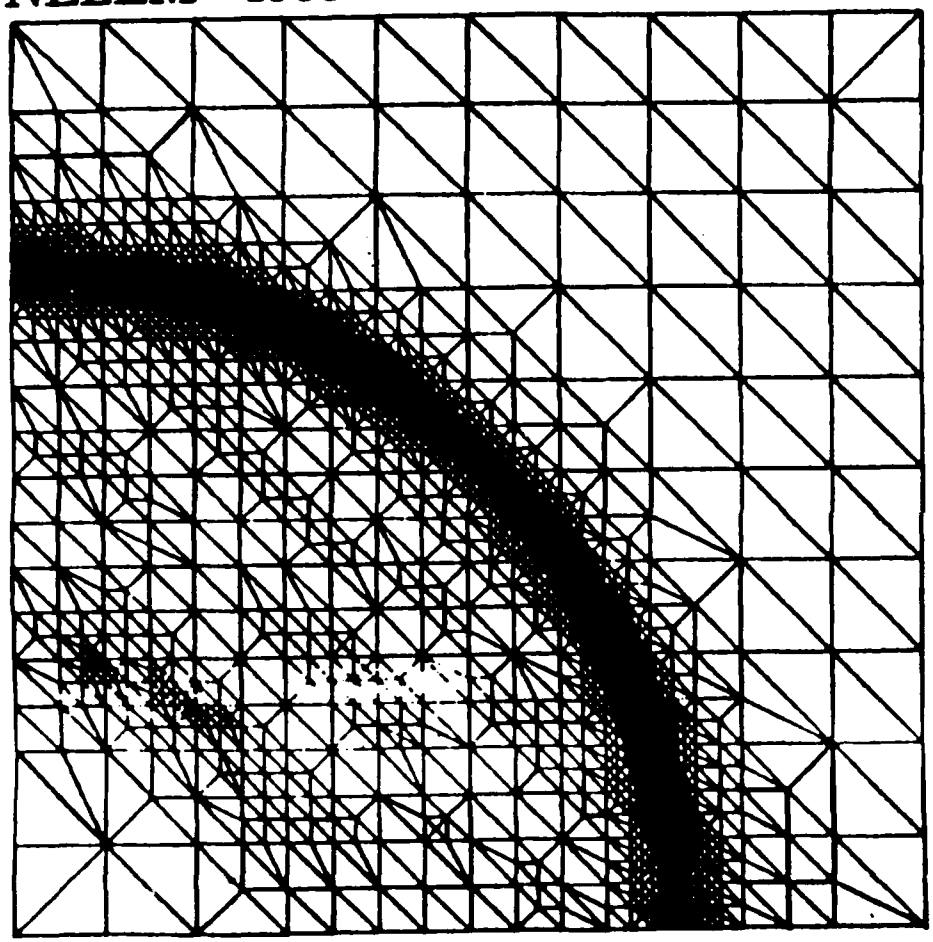
b)

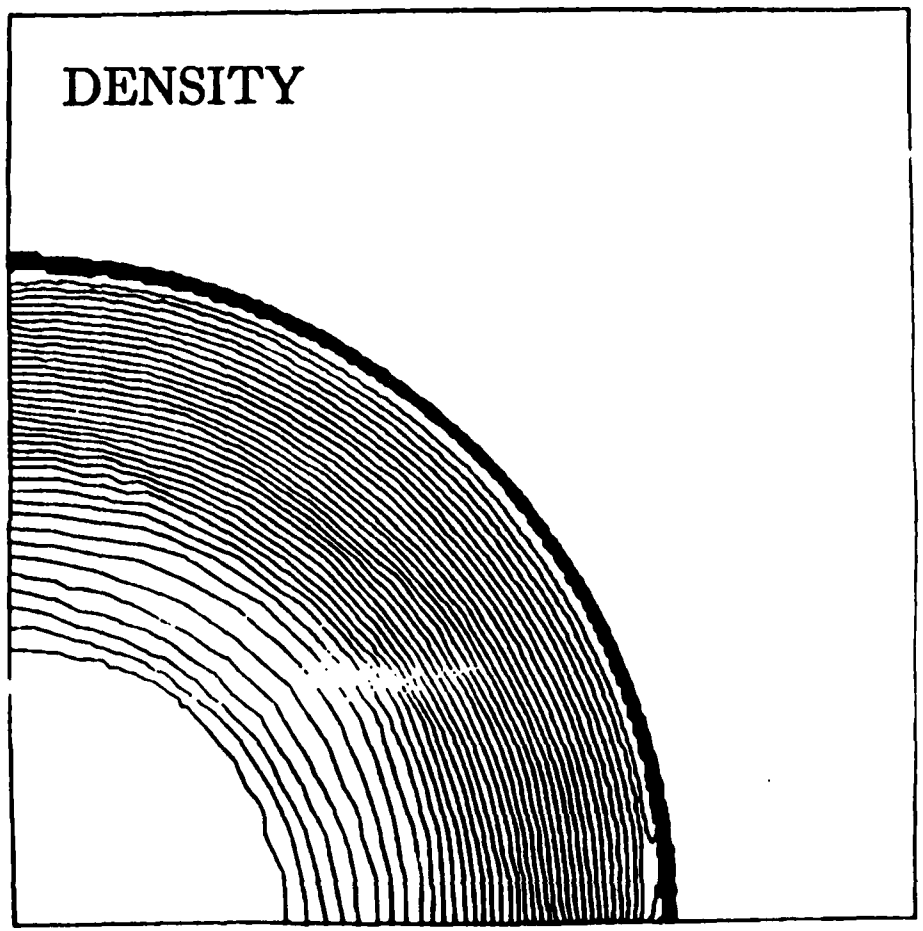
Figure 3: (cont.)

t=9.3

NPOIN=2290

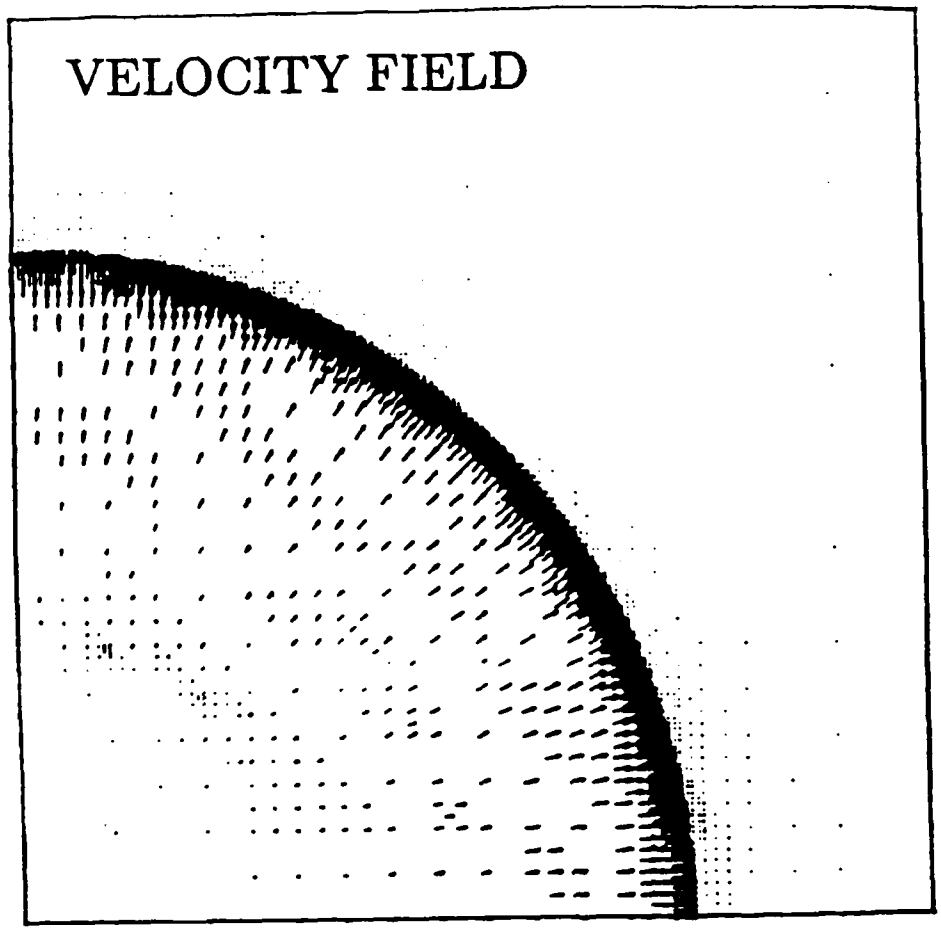
NELEM=4508

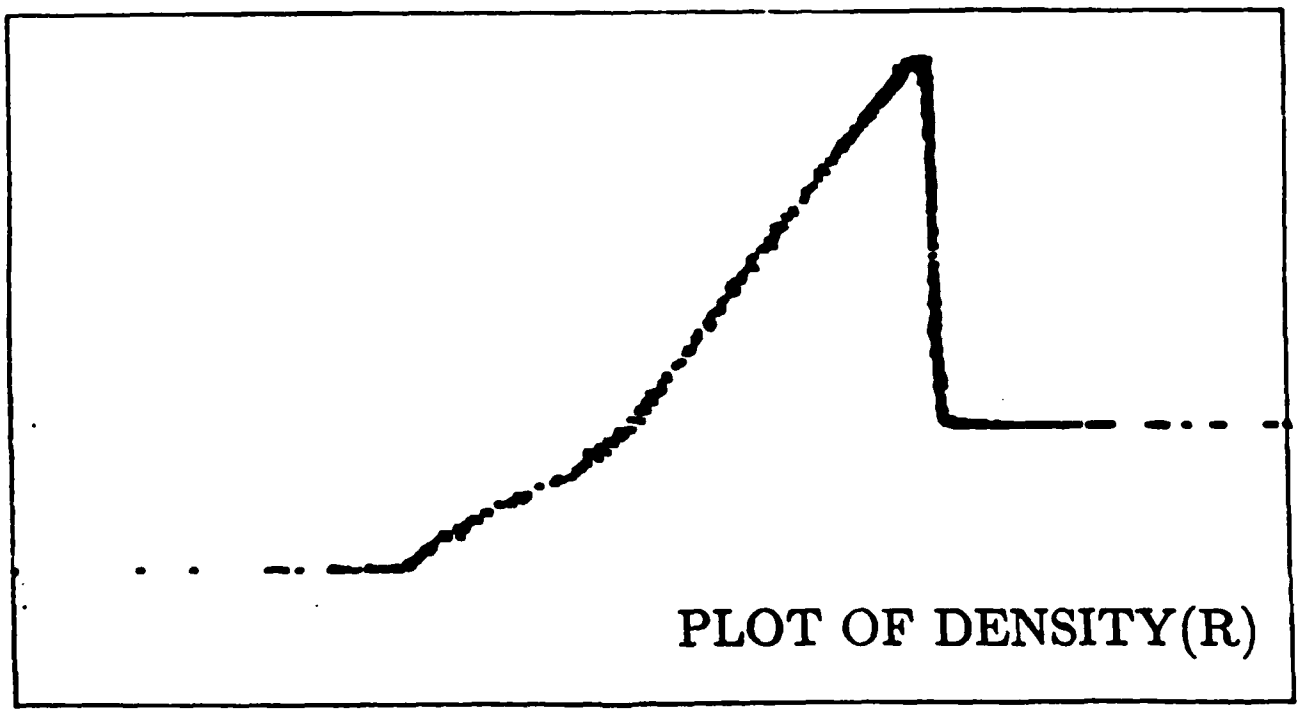




c)

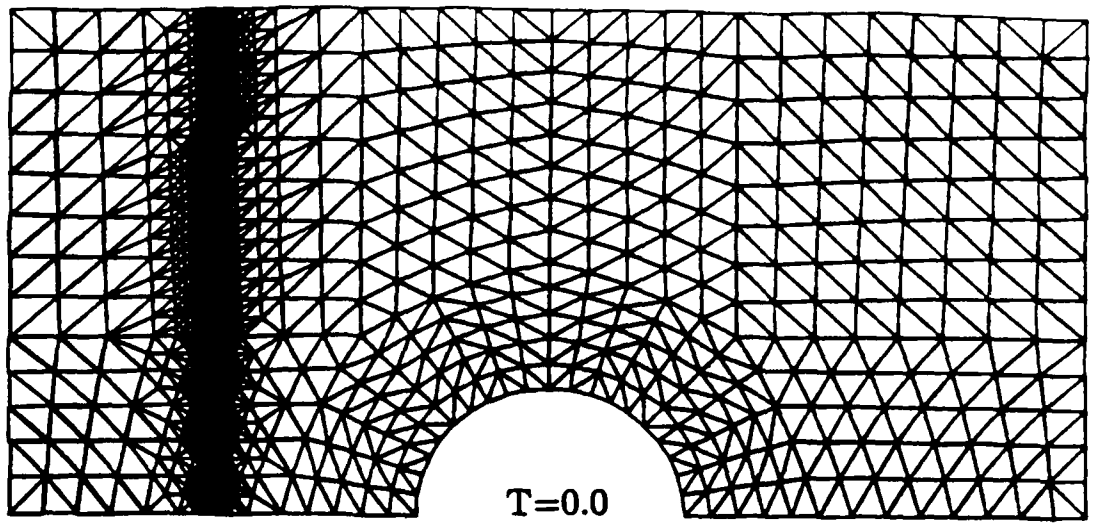
Figure 3: (cont.)

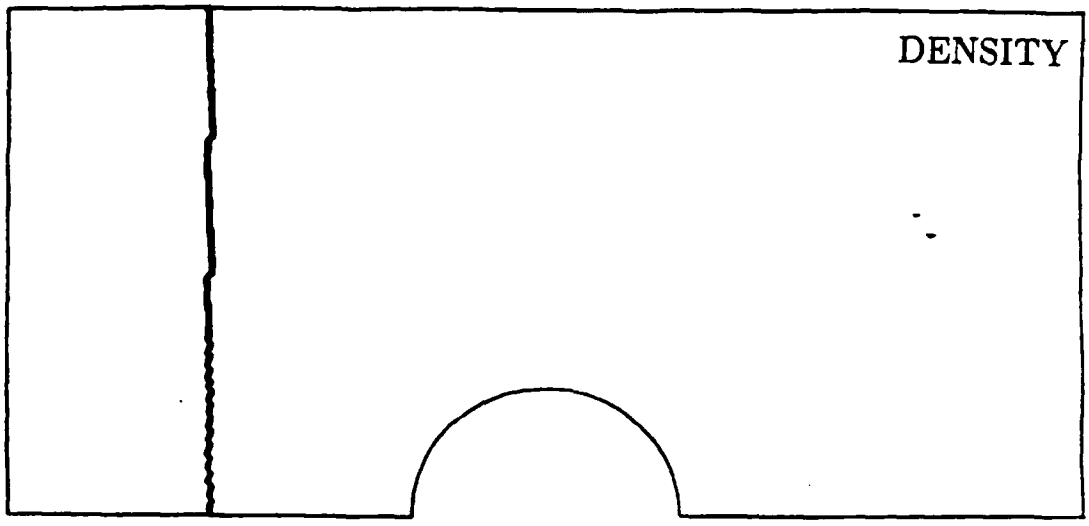




d)

Figure 3: (cont.)





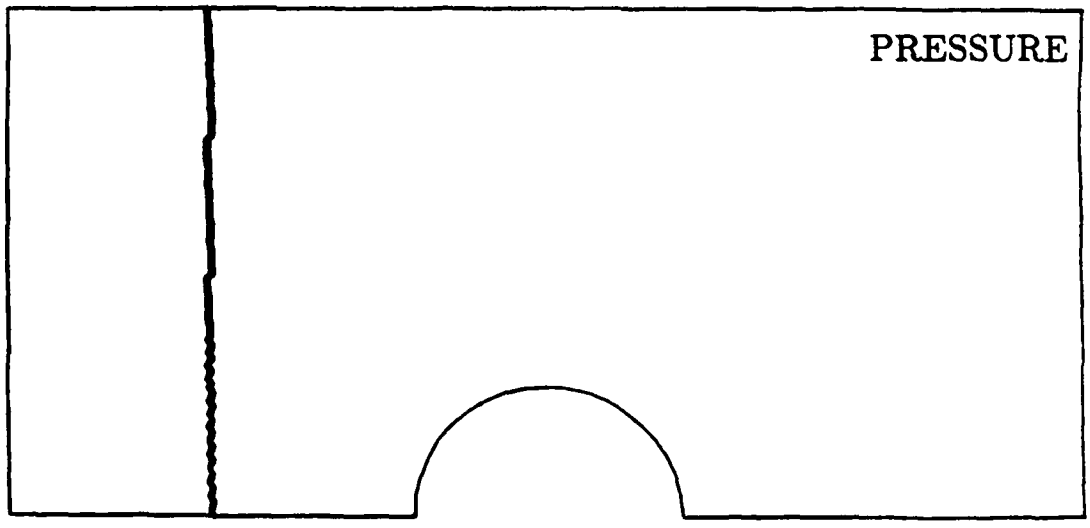
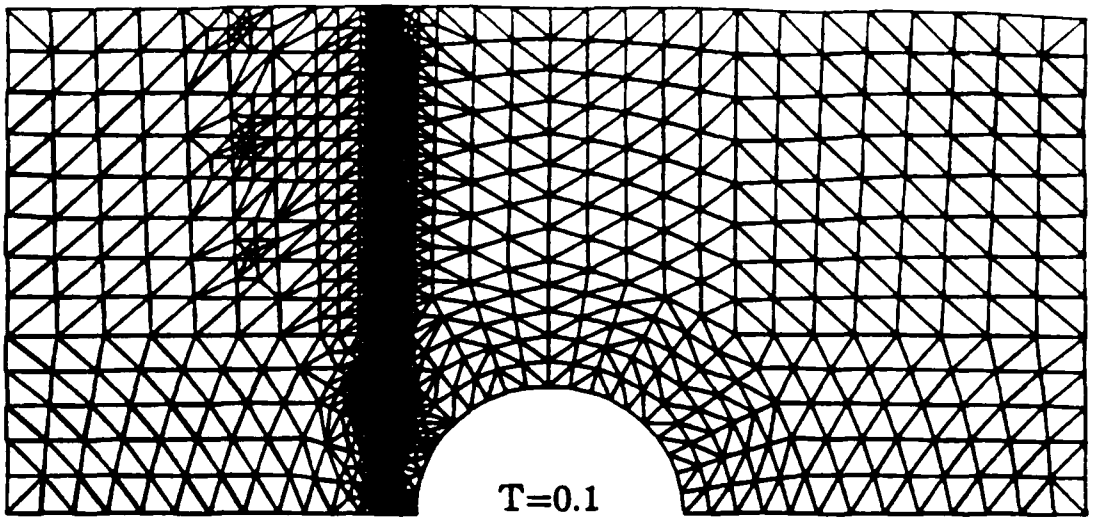
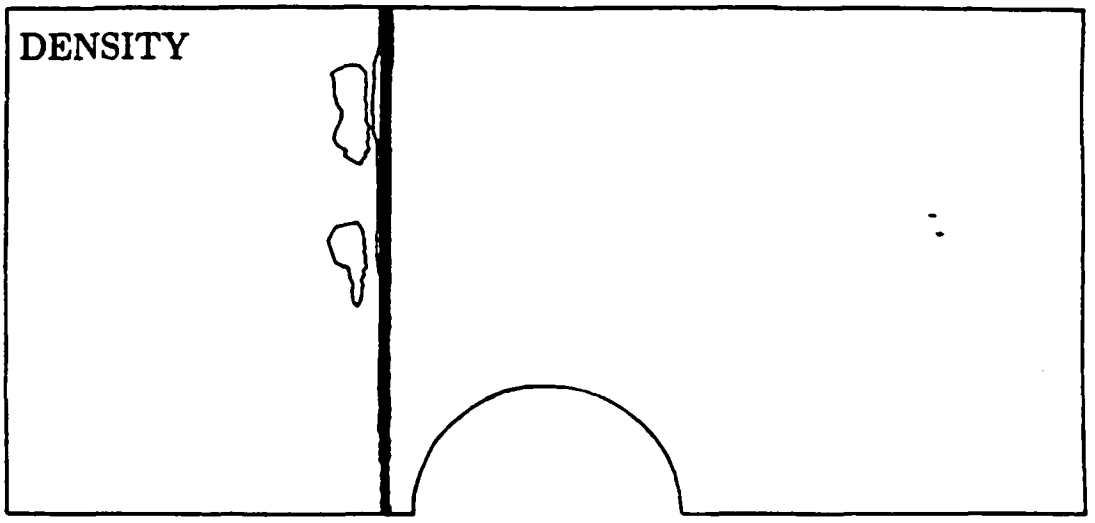


Figure 4: Shock impinging on a half-cylinder.

a) NELEM=2554, NPOIN=1335





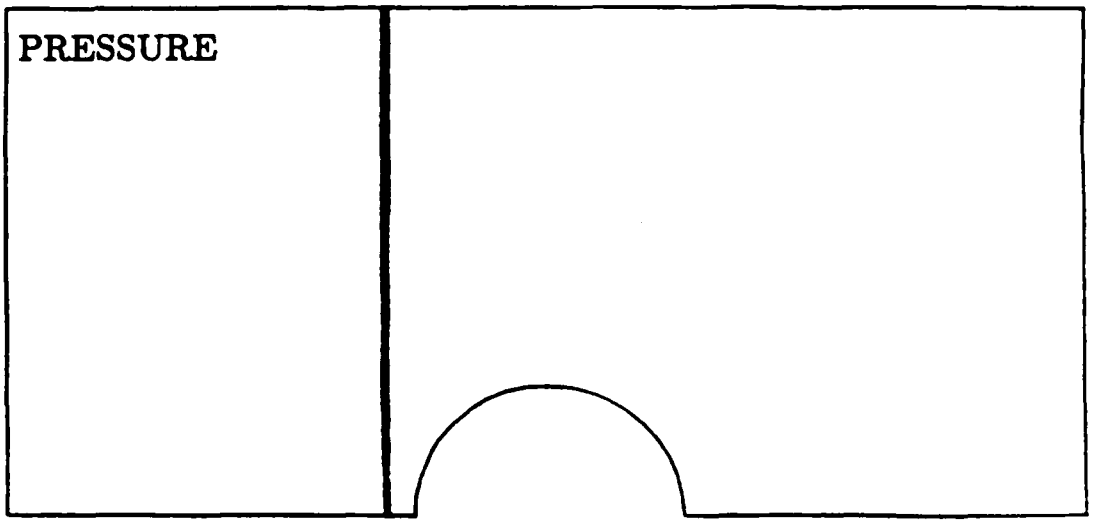
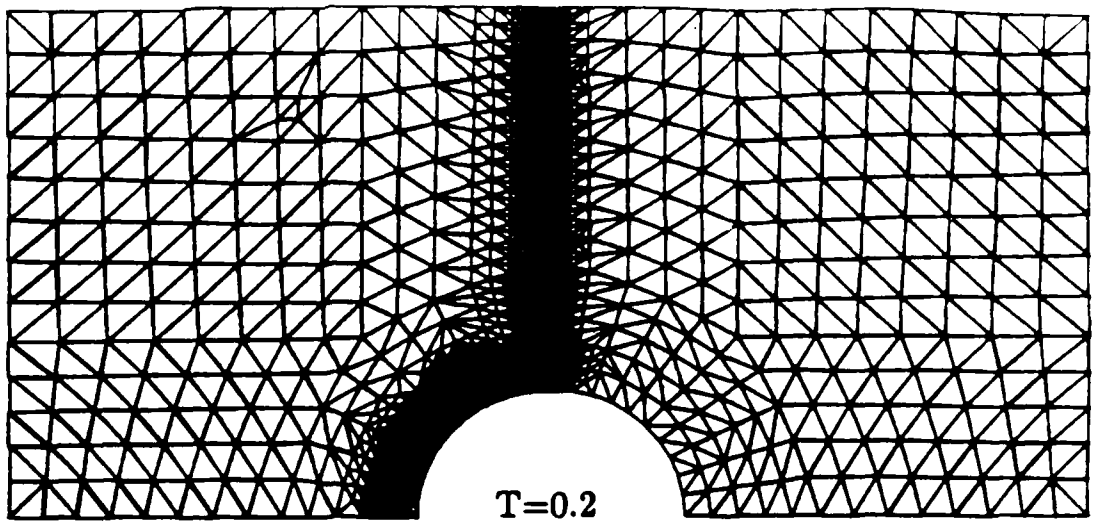
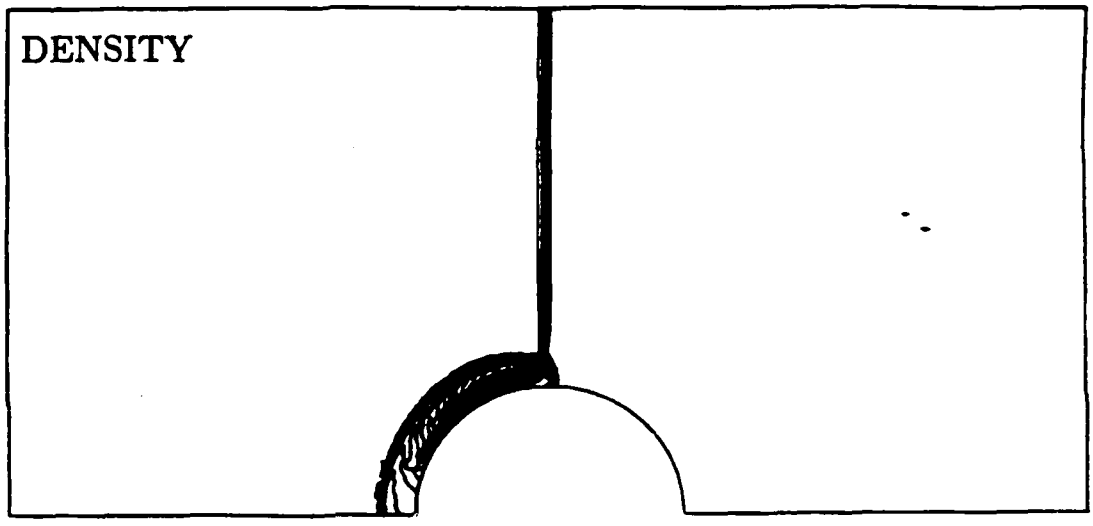


Figure 4: (cont.) NELEM=3096, NPOIN=1608

b)





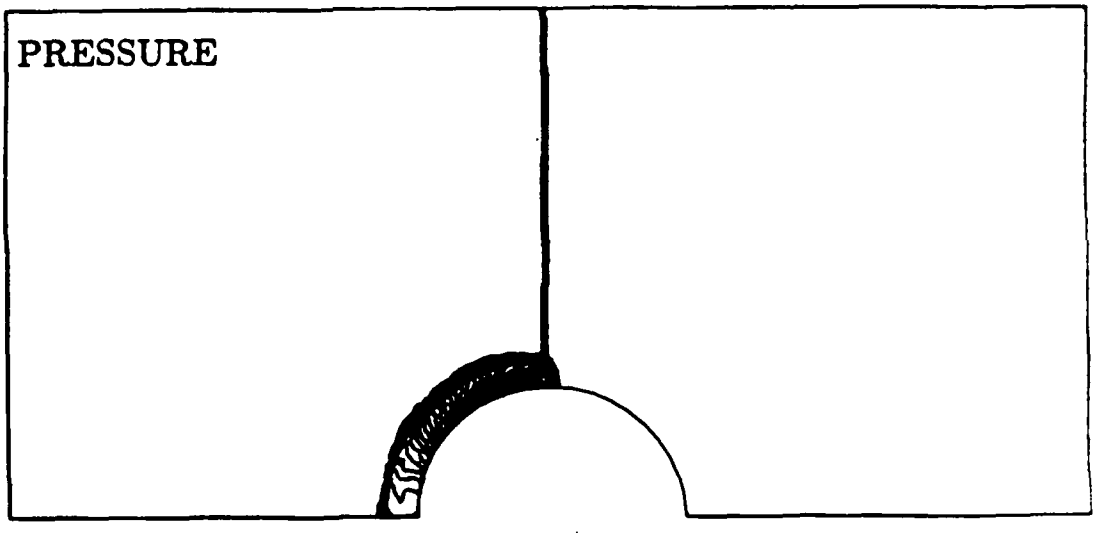
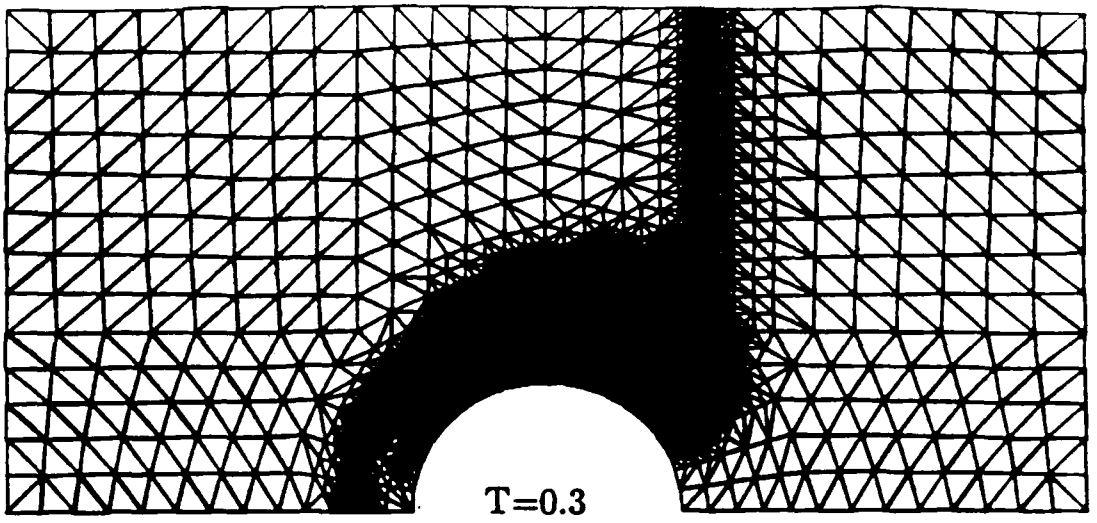
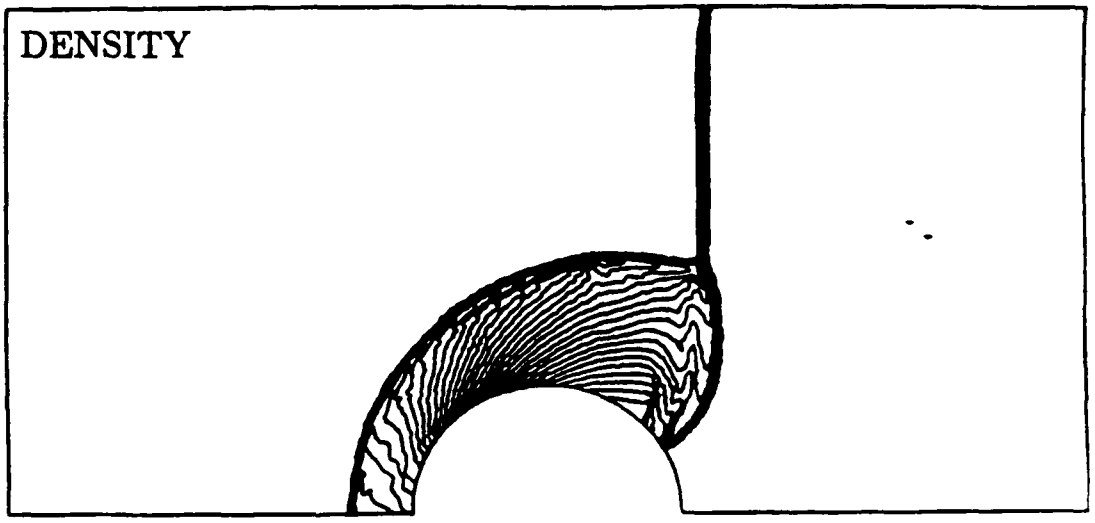


Figure 4: (cont.) NELEM=4628, NPOIN=2394

c)





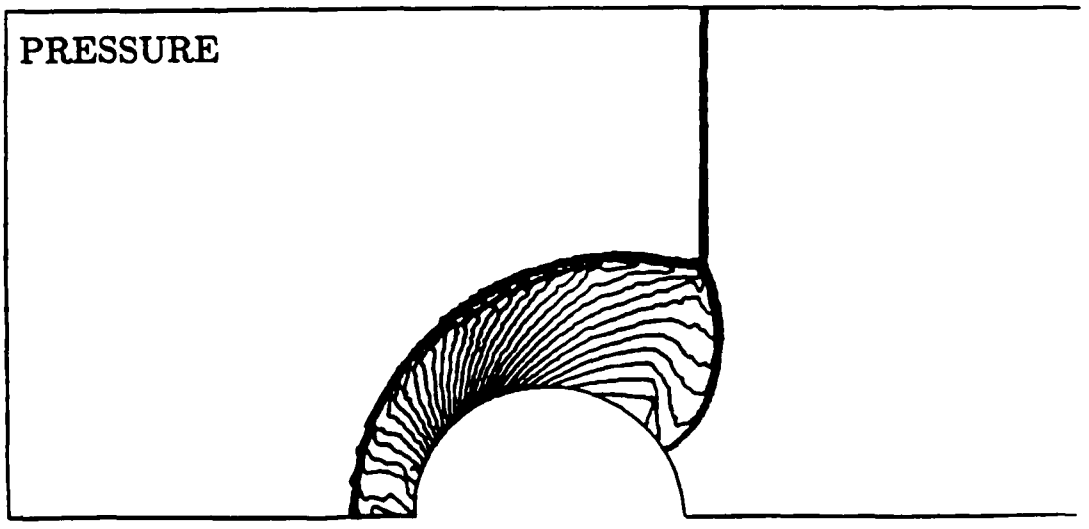
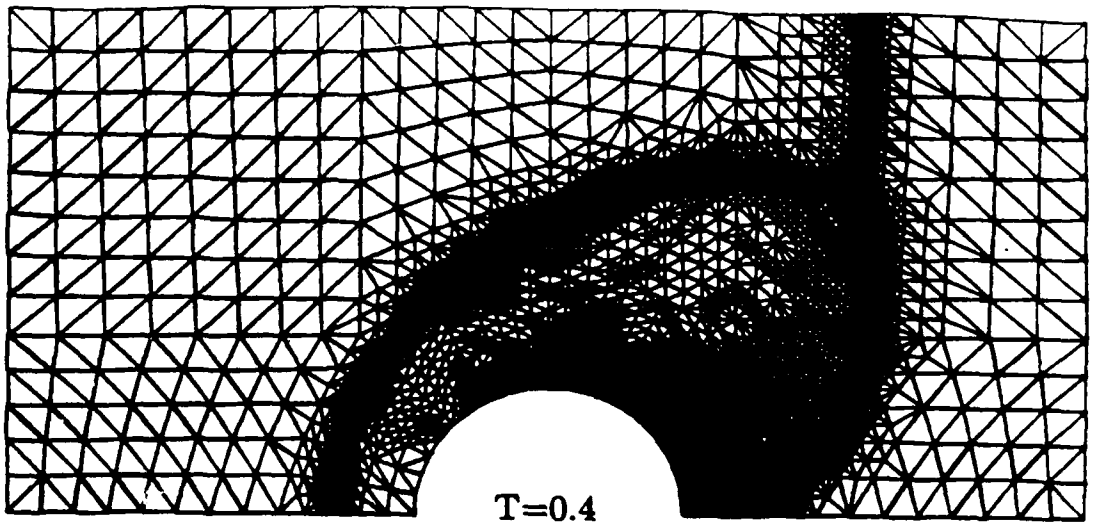
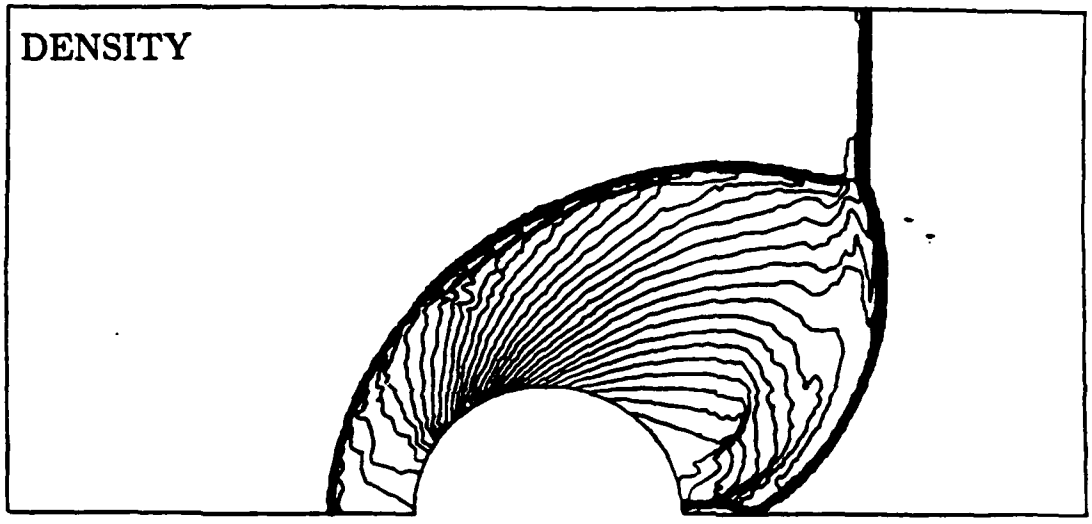


Figure 4: (cont.) NELEM=9057, NPOIN=4626

d)





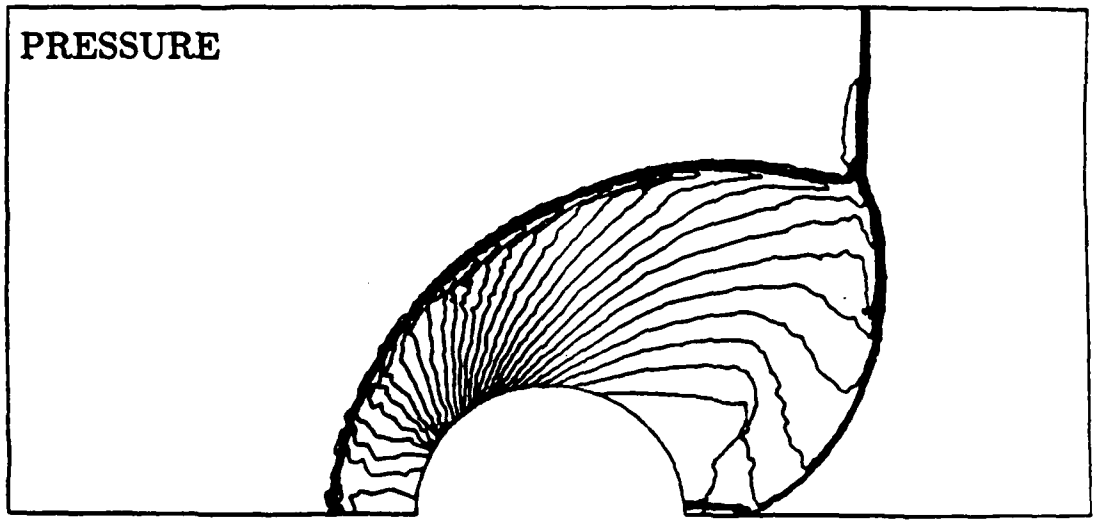
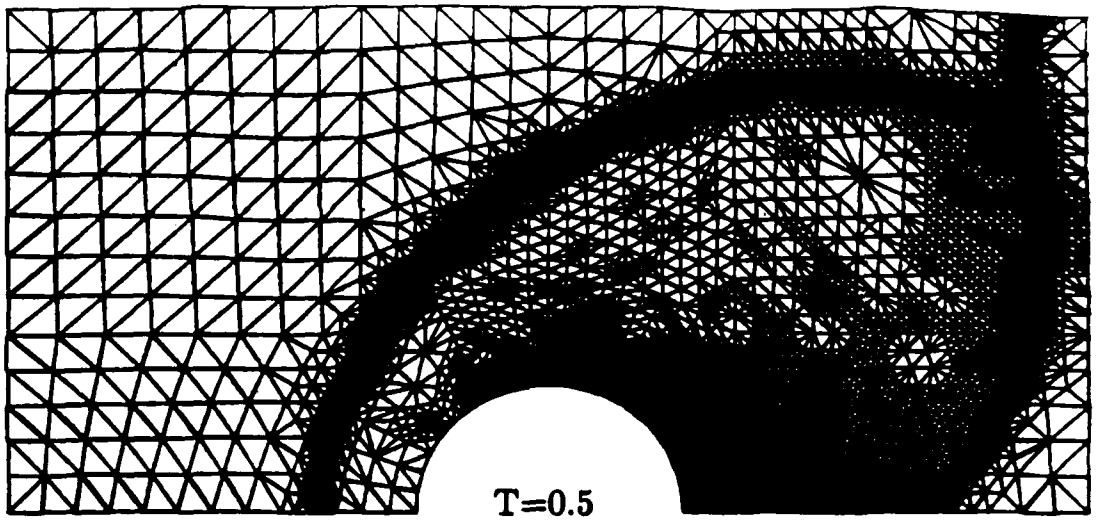
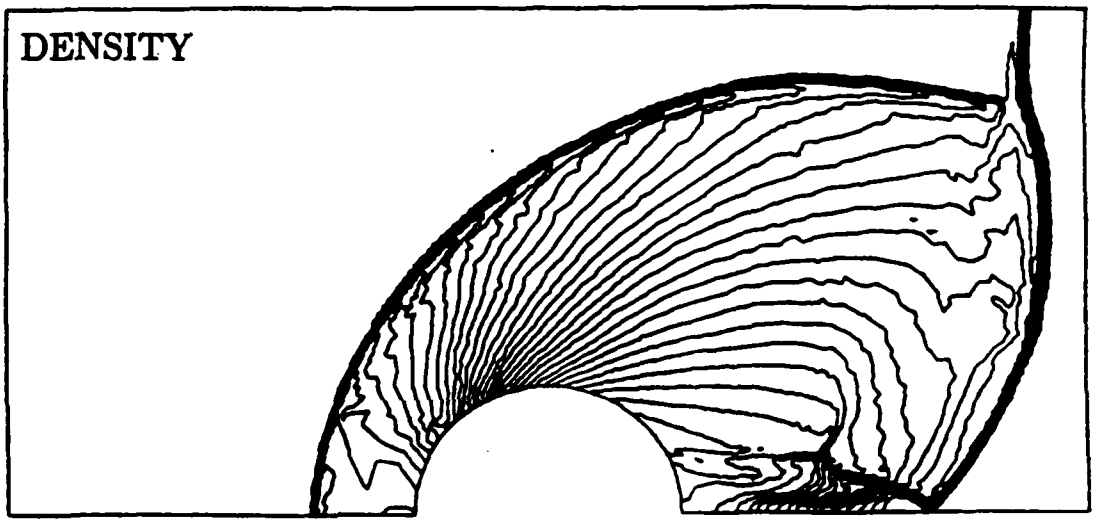


Figure 4: (cont.) NELEM=10071, NPOIN=5142

e)





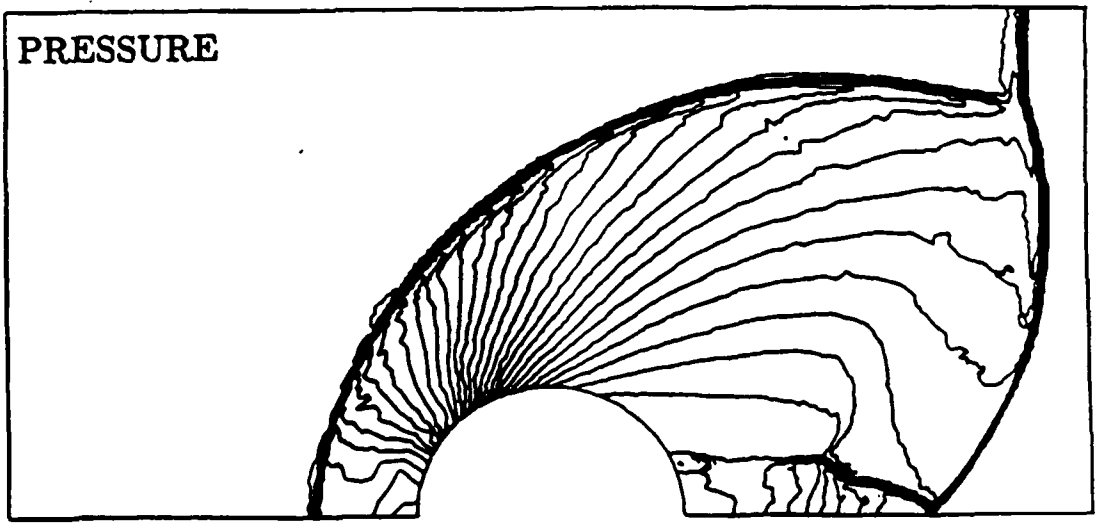
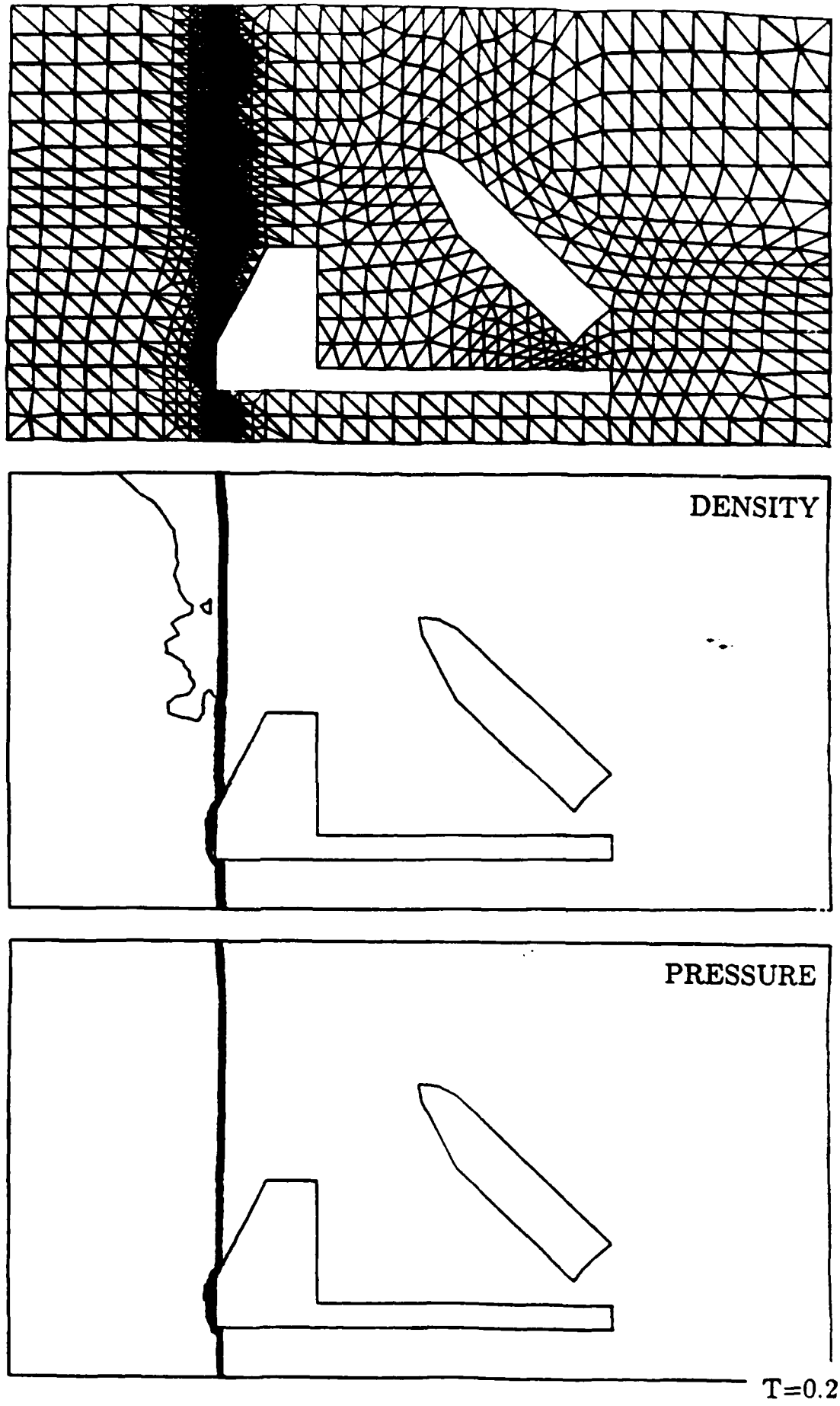


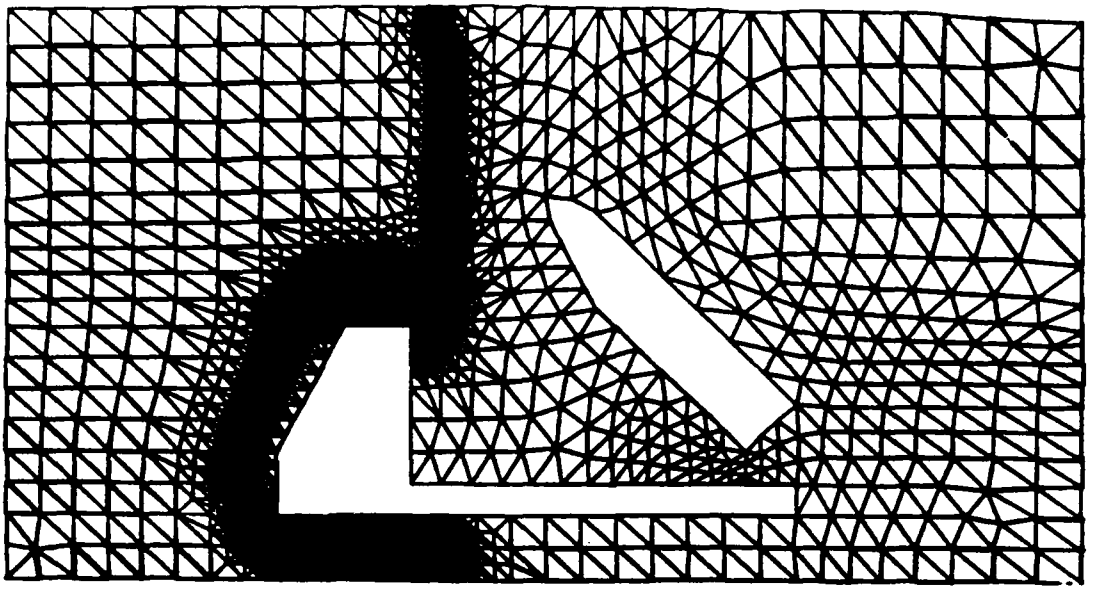
Figure 4: (cont.) NELEM=12020, NPOIN=6129

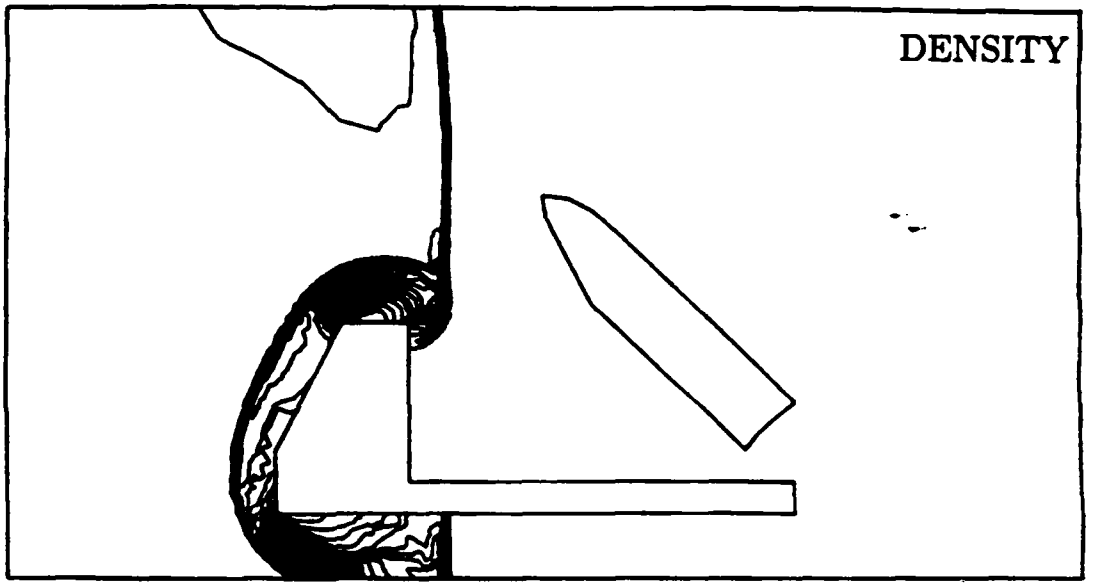
f)

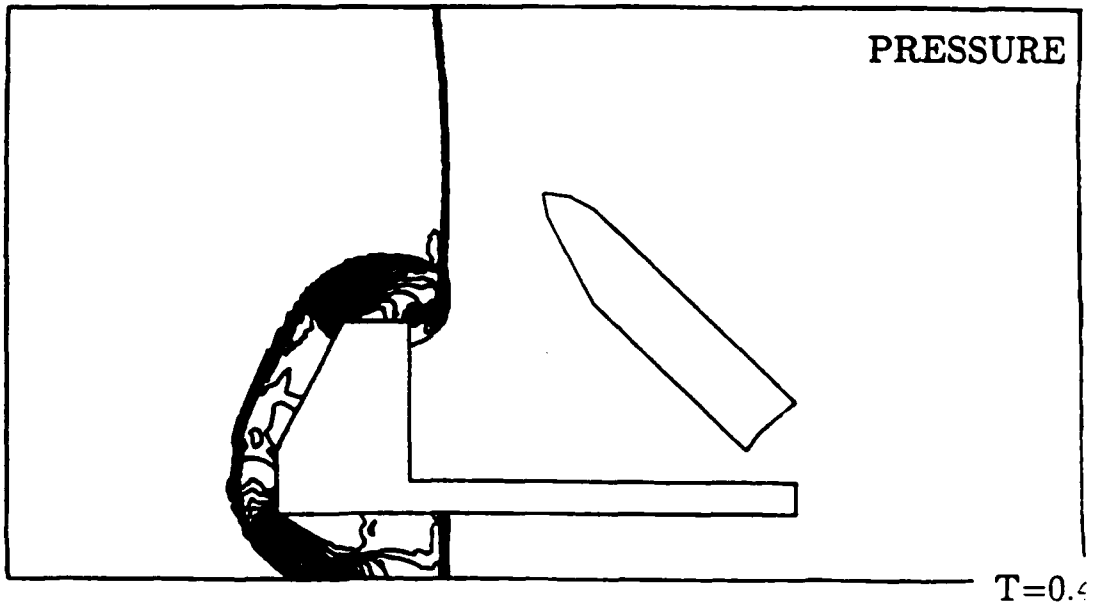


a)

Figure 5: Shock impinging on two obstacles.

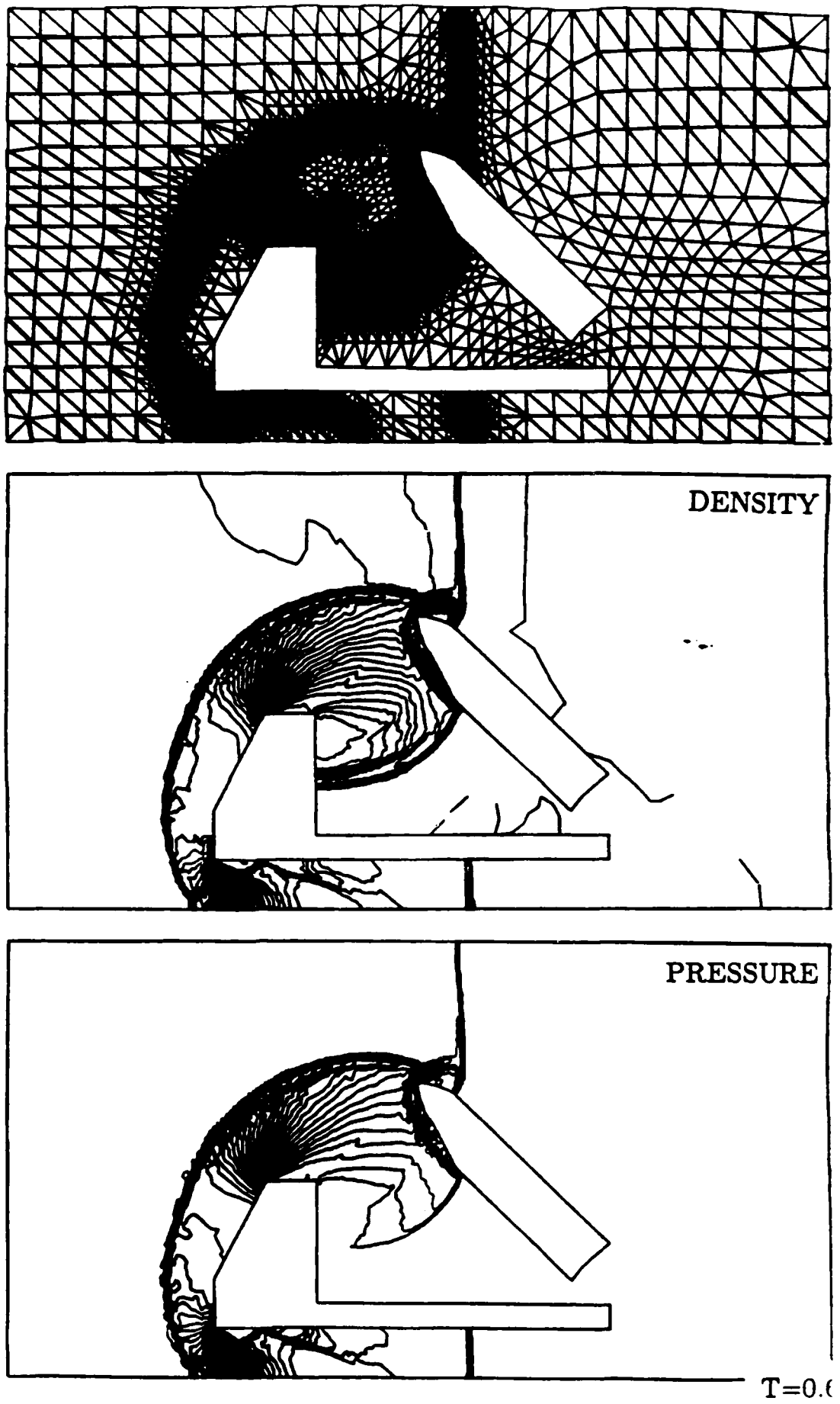




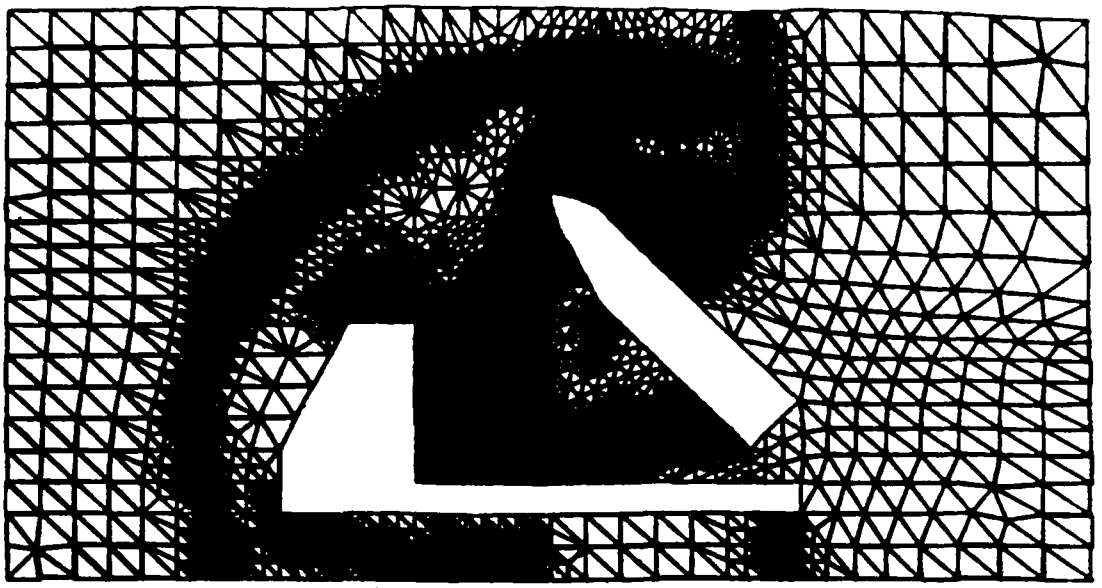


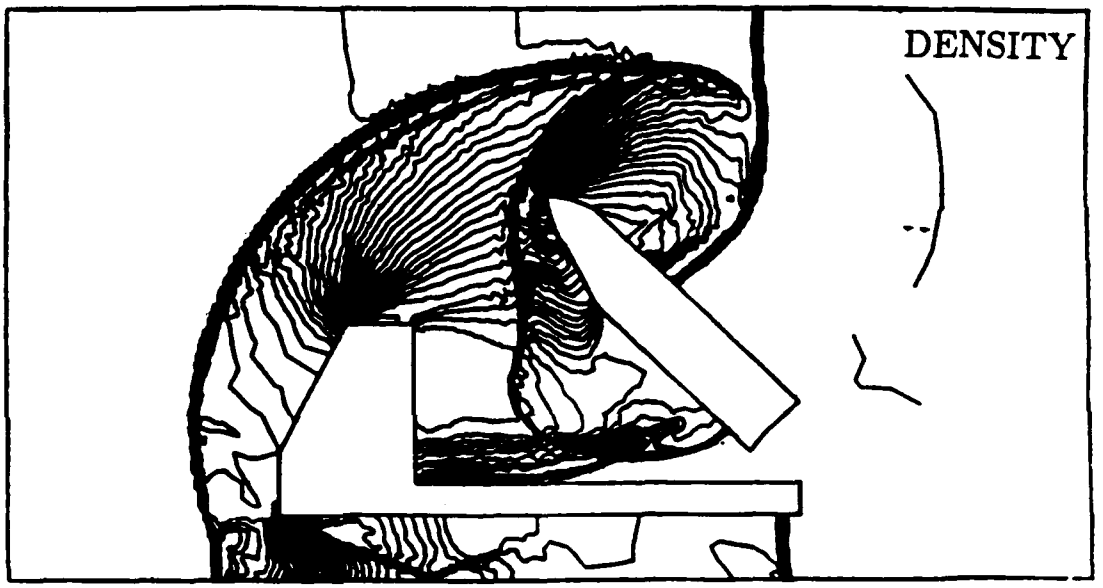
b)

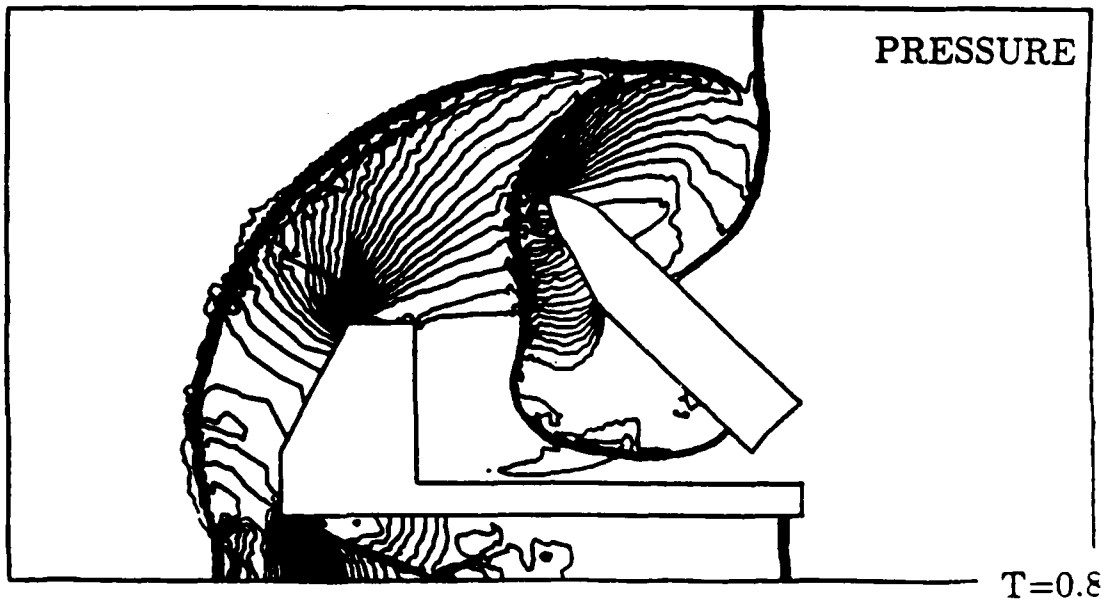
Figure 5: (cont.) NELEM=6038, NPOIN=3153



c) Figure 5: (cont.) NELEM=10713, NPOIN=5526







d) **Figure 5:** (cont.) NELEM=15798, NPOIN=8101

APPENDIX U.

Finite Element Flux-Corrected Transport (FEM-FCT) for the Euler and Navier-Stokes Equations

FINITE ELEMENT FLUX-CORRECTED TRANSPORT (FEM-FCT)

FOR THE EULER AND NAVIER-STOKES EQUATIONS

Rainald Löhner

Berkeley Research Associates
Springfield, VA 22150, USA
and
Laboratory for Computational Physics and Fluid Dynamics
Naval Research Laboratory
Washington, D.C. 20375, USA

Ken Morgan, Jaime Peraire and Mehdi Vahdati

Institute for Numerical Methods in Engineering
University of Wales
Swansea SA2 8PP, Wales, U.K.

ABSTRACT

A high resolution finite element method for the solution of problems involving high speed compressible flows is presented. The method uses the concepts of flux-corrected transport and is presented in a form which is suitable for implementation on completely unstructured triangular or tetrahedral meshes. Transient and steady state examples are solved to illustrate the performance of the algorithm.

INTRODUCTION

Over the past few years, there has been an ongoing interest in the application of unstructured grid finite element methods to the solution of problems of high speed compressible flow. In this area, the authors [18-20] have proposed a two-step explicit implementation of a second order Taylor-Galerkin procedure [16,17] and have used this approach to solve successfully a variety of inviscid and viscous problems. The addition of artificial viscosity is required to stabilize this solution procedure when it is applied to the analysis of problems involving strong discontinuities, and this has the effect of spreading flow discontinuities over several computational cells.

Solution methods based upon high resolution schemes [1-6] give sharper definition of flow discontinuities and are supposedly more robust. In two and three dimensions, these methods are generally implemented by using operator splitting and applying one-dimensional concepts in each coordinate direction separately. The finite element practitioner, however, finds difficulty in operating in this same manner, as the use of unstructured grids makes this approach impractical. The one high resolution method which can be used directly on unstructured grids is Zalesak's [7] multidimensional generalization of the 1-D flux-corrected transport (FCT) ideas of Boris and Book [8-10]. This method employs a high-order scheme together with a low-order scheme and attempts to combine these in such a way that the high-order solution is used in smooth regions of the flow whereas the low-order solution is favored near discontinuities. The low-order scheme should produce monotonic results for the problem to be solved. Erlebacher [11] and Parrott and Christie [12] showed how FCT ideas could be interpreted in the finite element context for a single governing equation and implemented on triangular meshes. Our contribution is the extension of the technique to deal with the solution of a system of equations and the formulation of a scheme with high temporal accuracy, which is well-suited for the analysis of transient problems. The numerical examples presented to demonstrate the performance of the algorithm involve the solution of both steady and transient flows of inviscid and viscous fluids.

THE EQUATIONS OF COMPRESSIBLE FLOW

The governing equations of compressible flow can be written in the conservation form

$$\frac{\partial U}{\partial t} + \frac{\partial F_j^a}{\partial x_j} = \frac{\partial F_j^v}{\partial x_j}, \tag{1}$$

where the summation convention has been employed and

$$U = \begin{Bmatrix} \rho \\ \rho u_i \\ \rho e \end{Bmatrix} \quad , \quad F_j^a = \begin{Bmatrix} \rho u_j \\ \rho u_i u_j + p\delta_{ij} \\ u_j(\rho e + p) \end{Bmatrix} \quad , \quad F_j^v = \begin{Bmatrix} 0 \\ \sigma_{ij} \\ u_i\sigma_{ij} + k\dfrac{\partial T}{\partial x_j} \end{Bmatrix}. \tag{2}$$

Here ρ, p, e, T and k denote the density, pressure, specific total energy, temperature and thermal conductivity of the fluid respectively and u_i is the component of the fluid velocity in the direction x_i of a Cartesian coordinate system. The equation set is completed by the addition of the state equations

$$p = (\gamma - 1)\rho[e - \frac{1}{2}u_j u_j] \quad , \quad T = c_v[e - \frac{1}{2}u_j u_j] \quad , \tag{3}$$

which are valid for a perfect gas, where γ is the ratio of the specific heats and c_v is the specific heat at constant volume. The components of the viscous stress tensor σ_{ij} are given by

$$\sigma_{ij} = \mu(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}) + \lambda \frac{\partial u_k}{\partial x_k}\delta_{ij} \tag{4}$$

and it is assumed that λ and μ are related by

$$\lambda = -\frac{2\mu}{3} \tag{5}$$

THE FLOW SOLVER: FEM-FCT

As stated above, high resolution, monotonicity preserving schemes must be developed in order to be able to simulate the strong nonlinear discontinuities present in the flows under consideration. Although the pertinent literature abounds with high resolution schemes [1-6], only Zalesak's generalization [7] of the 1-D FCT schemes of Boris and Book [8-10] can be considered a truly multidimensional high resolution scheme. We remark here that the use of unstructured grids requires such truly multidimensional schemes, as the lack of lines or planes in the mesh inhibits the use of operator splitting.

Erlebacher [11], and Parrot and Christie [12] first analyzed FCT schemes in the context of finite element methods. We develop their ideas further to include the consistent mass, which yields high temporal accuracy, and to systems of equations.

The Concept of Flux-Corrected Transport (FCT)

We consider a set of conservation laws given by a system of partial differential equations of the form given in eqn.(1), and assume that the advective fluxes $F^a = F^a(U)$ play a dominant role over the viscous fluxes $F^v = F^v(U)$. For flows described by eqn.(1), discontinuities in the variables may arise (e.g. shocks or contact discontinuities). Any numerical scheme of order higher than one will produce overshoots or ripples at such discontinuities (the so-called 'Godunov theorem' [15]). Very often, particularly for mildly nonlinear systems, these overshoots can be tolerated. However, for the class of problems studied here, overshoots will eventually lead to numerical instability, and will therefore have to be suppressed.

The idea behind FCT is to combine a high-order scheme with a low-order scheme in such a way that in regions where the variables under consideration vary smoothly (so that a Taylor expansion makes sense) the high-order scheme is employed, whereas in those regions where the variables vary abruptly the schemes are combined, in a conservative manner, in an attempt to ensure a monotonic solution.

The temporal discretization of eqn.(1) yields

$$U^{n+1} = U^n + \Delta U, \tag{6}$$

where ΔU is the increment of the unknowns obtained for a given scheme at time $t = t^n$. Our aim is to obtain a ΔU of as high an order as possible without introducing overshoots. To this end, we re-write eqn.(6) as:

$$U^{n+1} = U^n + \Delta U^l + (\Delta U^h - \Delta U^l), \tag{7}$$

or

$$U^{n+1} = U^l + (\Delta U^h - \Delta U^l). \tag{8}$$

Here ΔU^h and ΔU^l denote the increments obtained by some high- and low-order scheme respectively, whereas U^l is the monotone, ripple-free solution at time $t = t^{n+1}$ of the low-order scheme. The idea behind FCT is to limit the second term on the right-hand side of eqn.(8):

$$U^{n+1} = U^l + lim(\Delta U^h - \Delta U^l), \tag{9}$$

in such a way that no new over/undershoots are created.

It is at this point that a further constraint, given by the conservation law (1) itself must be taken into account: strict conservation on the discrete level should be maintained. The simplest way to guarantee this for node-centered schemes (and we will only consider those here) is by constructing schemes for which the sum of the contributions of each individual element (cell) to its surrounding nodes vanishes ('all that comes in goes out'). This means that the limiting process (eqn.(9)) will have to be carried out in the elements (cells).

Algorithmic Implementation

We can now define FCT in a quantitative way. We follow Zalesak's exposition [7], but modify the term 'flux' by 'element contribution to a node'. Those more familiar with finite volume or finite difference schemes should replace 'element' by 'cell' in what follows.

FCT consists of the following six algorithmic steps:

1) Compute LEC: the 'low-order element contribution' from some low-order scheme guaranteed to give monotonic results for the problem at hand;

2) Compute HEC: the 'high-order element contribution', given by some high-order scheme;

3) Define AEC: the 'antidiffusive element contributions' :

$$AEC = HEC - LEC$$

4) Compute the updated low-order solution :

$$U^l = U^n + \sum_{el} LEC = U^n + \Delta U^l \tag{10}$$

5) Limit or 'correct' the AEC so that U^{n+1} as computed in step 6 below is free of extrema not also found in U^l or U^n :

$$AEC^c = Cel * AEC, \quad 0 \leq Cel \leq 1; \tag{11}$$

6) Apply the limited AEC :

$$U^{n+1} = U^l + \sum_{el} AEC^c. \tag{12}$$

The Limiting Procedure

Obviously, the whole approach depends critically on the all-important step 5 above. We define the following quantities:

a) $P_I^{\pm}$: the sum of all positive (negative) antidiffusive element contributions to node I

$$P_I^{\pm} = \sum_{el} \begin{Bmatrix} max \\ min \end{Bmatrix} (0, AEC_{el})$$

b) $Q_I^{\pm}$: the maximum (minimum) increment (decrement) node I is allowed to achieve in step 6 above

$$Q_I^{\pm} = U_I^{\substack{max \\ min}} - U^l$$

where $U_I^{\substack{max \\ min}}$ (defined below) represents the maximum (minimum) value the unknown U at node I is allowed to achieve in step 6 above.

c) $R^{\pm}$:

$$R^{\pm} := \begin{cases} min(1, Q^{\pm}/P^{\pm}) & if \quad P^+ > 0, \; P^- < 0 \\ 0 & if \quad P^{\pm} = 0 \end{cases}$$

Now take, for each element:

$$Cel = min(element\ nodes) \begin{cases} R^+ & if \quad AEC > 0, \\ R^- & if \quad AEC < 0. \end{cases} \tag{13}$$

Finally, we obtain $U_I^{max}_{min}$ in three steps :

a) maximum (minimum) nodal U of U^n and U^l :

$$U_I^* = \begin{Bmatrix} max \\ min \end{Bmatrix} (U_I^l, U_I^n) \ ,$$

b) maximum (minimum) nodal value of element :

$$U_{el}^* = \begin{Bmatrix} max \\ min \end{Bmatrix} (U_A^*, U_B^*, ..., U_C^*) \ ,$$

where $A, B, ..., C$ represent the nodes of element el.

c) maximum (minimum) U of all elements surrounding node I :

$$U_I^{max}_{min} = \begin{Bmatrix} max \\ min \end{Bmatrix} (U_1^*, U_2^*, ..., U_m^*) \ .$$

where $1, 2, ..., m$ represent the elements surrounding node I.

This completes the description of the limiting procedure. Up to this point we have been completely general in our description, so that eqns.(6)-(13) may be applied to any FEM-FCT scheme. In what follows, we restrict the exposition to the finite element schemes employed in the present work, describing the high and low-order schemes used.

The High-Order Scheme: Consistent-Mass Taylor Galerkin

As the high-order scheme, we employ a two-step form [18-20] of the one-step Taylor-Galerkin schemes described in [16,17]. These schemes belong to the Lax-Wendroff class, and could be substituted by any other high-order scheme which appears more convenient, including implicit schemes. Given the system of equations (1), we advance the solution from t^n to $t^{n+1} = t^n + \Delta t$ as follows:

a) First step (advective predictor):

$$U^{n+\frac{1}{2}} = U^n - \frac{\Delta t}{2} \cdot \left. \frac{\partial F_j^a}{\partial x_j} \right|^n \tag{14}$$

b) Second step :

$$\Delta U^n = U^{n+1} - U^n = -\Delta t \left. \frac{\partial F_j^a}{\partial x_j} \right|^{n+\frac{1}{2}} + \Delta t \left. \frac{\partial F_j^v}{\partial x_j} \right|^n \tag{15}$$

The spatial discretization of (14) and (15) is performed via the classic Galerkin weighted residual method [18-20], using linear elements, i.e. 3-noded triangles in 2-D and 4-noded tetrahedra in 3-D. For (15) the following system of equations is obtained:

$$M_C \cdot \Delta U^n = R^n, \tag{16}$$

where M_C denotes the consistent mass matrix [18-20], ΔU the vector of nodal increments and R the vector of added element contributions to the nodes. As M_C possesses an excellent condition number, eqn.(16) is never solved directly, but iteratively, requiring typically three passes [17]. We recast the converged solution of eqn.(16) into the following form, which will be of use later on :

$$M_L \cdot \Delta U^h = R + (M_L - M_C) \cdot \Delta U^h. \tag{17}$$

Here M_L denotes the diagonal, lumped mass-matrix (see [17]).

The Low-Order Scheme: Lumped-Mass Taylor Galerkin plus Diffusion

The requirement placed on the low-order scheme in any FCT-method is monotonicity. The low-order scheme must not produce any artificial, or numerical, 'ripples' or 'wiggles'. It is clear that the better the low-order scheme, the easier the resulting task of limiting will be. Therefore an obvious candidate for the low-order scheme is Godunov's method [15]. However, this scheme would be relatively expensive, and its extension to unstructured grids remains unclear.

We have so far added 'mass-diffusion' to the lumped-mass Taylor-Galerkin scheme in the context of FEM-FCT [13,14]. This simplest and least expensive form of diffusion is obtained by subtracting the lumped mass-matrix from the consistent mass-matrix for linear elements:

$$DIFF = c_d \cdot (M_C - M_L) \cdot U^n. \tag{18}$$

The element matrix thus obtained for 2-D triangles is of the form

$$c_d \cdot (M_C - M_L)_{el} = - \frac{c_d \cdot Vol_{el}}{12} \begin{Bmatrix} 2 & -1 & -1 \\ -1 & 2 & -1 \\ -1 & -1 & 2 \end{Bmatrix} \tag{19}$$

Observe that we cannot simply add this diffusion to the high-order scheme in order to obtain monotonic results, as a multipoint-coupling of the right-hand side occurs due to the consistent mass-matrix employed in the high-order scheme . The imposition of monotonicity can nevertheless be achieved by using a lumped mass-matrix instead. As the terms originating from the discretization of the fluxes F^i in (1) are the same as in (15), the low-order scheme is given by

$$M_L \cdot \Delta U^l = R + DIFF. \tag{20}$$

Resulting Antidiffusive Element Contributions

Subtracting (20) from (17) yields the equation

$$M_L \cdot (\Delta U^h - \Delta U^l) = R + (M_L - M_C) \cdot \Delta U^h - R - DIFF, \tag{21}$$

or, using eqn.(18)

$$\Delta U^h - \Delta U^l = M_l^{-1} \cdot (M_L - M_C) \cdot (c_d \cdot U^n + \Delta U^h). \tag{22}$$

Note that all terms arising from the discretization of the fluxes F^i in (1),(15),(20) have now disappeared. This is of particular importance if the antidiffusive element contributions must be recomputed (small core memory machines), and real gas effects are taken into account (table look-up times are considerable) or real viscosity effects have to be included (Navier-Stokes equations).

Limiting for Systems of Equations

The results available in the literature [8-10] and our own experience [13,14] have shown that with FCT results of excellent quality can be obtained for a single PDE. However, when trying to extend the limiting process to systems of PDEs, no immediately obvious or natural limiting procedure becomes apparent. Obviously, for 1-D problems one could advect each simple wave system separately, and then assemble the solution at the new time step. However, for multidimensional problems such a splitting is not possible, as the acoustic waves are circular. FDM-FCT-codes used for production runs [21,22] have so far limited each equation separately, invoking operator-splitting arguments. This approach does not always give very good results, as may be seen from Sod's comparison of schemes for the Riemann problem [23], and has been a point of continuing criticism by those who prefer to use the more costly Riemann-solver-based, essentially one-dimensional TVD-schemes [1-6]. It would therefore appear as attractive to introduce 'system character' for the limiter by combining the limiters for all equations of the system. Many variations are possible and can be implemented, giving different performance for different problems. We just list some of the possibilities here, commenting on them where empirical experience is available.

a) Independent treatment of each equation as in operator-split FCT: this is the least diffusive method, tending to produce an excessive amount of ripples in the non-conserved quantities (and ultimately also in the conserved quantities).

b) Use of the same limiter (C_{el}) for all equations: this produces much better results, seemingly because the phase errors for all equations are 'synchronized'. This was also observed by Harten and Zwaas [24] and Zhmakin and Fursenko [25] for a class of schemes very similar to FCT. We mention the following possibilities:

i) Use of a certain variable as 'indicator variable' (e.g. density, pressure, entropy).

ii) Use of the minimum of the limiters obtained for the density and the energy ($C_{el} = min(C_{el}(density), C_{el}(energy))$) : this produces acceptable results, although

some undershoots for very strong shocks are present. This option is currently our preferred choice for transient problems.

iii) Use of the minimum of the limiters obtained for the density and the pressure ($C_{el} = min(C_{el}(density), C_{el}(pressure))$) : this again produces acceptable results, particularly for steady-state problems.

NUMERICAL EXAMPLES

a) Shock over an indentation: The first problem considered simulates the transient flowfield produced by the interaction of a strong shock with an indentation in the ground. For this case, the shock Mach number was set to $M_s = 25$, which corresponds to a pressure-jump ratio of about 1:100. During the transient, pressure ratios as high as 1:1000 result. The problem statement, solution domain, spatial discretization and solutions obtained are shown in Figs.1a-1e. Note that an adaptive refinement scheme for transient problems [26] was used to reduce the overall storage and CPU requirements.

As the shock travels over the indentation, it produces a bow shock and a rarefaction (Figs.1a,1b). Then, it collides with the right wall of the indentation and bounces back, producing several shock/shock and shock/contact discontinuity interactions (Figs.1c,1d). Observe the level of physically relevant detail that the scheme is able to reproduce, e.g. the triple shock produced at T=0.12 (Figs.1d,1e). The velocity pattern generated by these interactions has been magnified in Fig.1e, and shows a large residual vortex, as well as the different shock fronts and other discontinuities. We remark that at all times the shocks are captured within 2 to 3 elements.

In the present case, we used as limiter for all equations the minimum of the limiters computed for the continuity and energy equations. It is found, that for the strong shocks present in such flowfields, even a pressure-undershoot of 0.1% will lead to negative pressures. Therefore, the pressure is additionally limited artificially in order to be positive (albeit small) at all times.

b) Steady supersonic flow past a circular cylinder: This problem involves inviscid Mach 3 flow past a circular cylinder. The solution has been obtained by relaxing, with local timesteps, the transient solution towards the final steady-state. During this iteration process, the grid was adapted three times to the solution by using an adaptive mesh regeneration technique [27]. The final grid is shown in Fig.2a. A detail of the pressure coefficient distribution is shown in Fig.2b, and the variation of pressure coefficient along the centre line and over the cylinder surface is given in Fig.2c.

c) Shock-bubble interaction: This problem is included here to demonstrate a new axisymmetric capability, and also to show that not only geometrically complex domains, but also physically complex problems can be handled economically by the methodologies developed. Initially, a weak shock ($M_s = 1.29$), coming from the left in Fig.3a, travels into a bubble of heavier material. In the present case, the outer medium was assumed to be air, while the bubble was assumed to consist of freon. Due to the higher density of freon, the shock speed inside the bubble decreases (Fig.3b). While the outer

shock bends over, the inner shock focuses at the right end of the bubble producing a significant overpressure (Fig.3c), and initiating a small, circular blast wave (Fig.3d).

d) Steady supersonic flow over a flat plate: The fourth problem considered is the steady state solution of supersonic viscous flow over a flat plate. The flow conditions correspond identically to one of the cases considered by Carter [28], using a finite difference scheme. The free stream Mach number is 3 and the Reynolds number based on the plate length is 1000. The temperature of the plate is assumed constant. The Sutherland viscosity law (see, e.g. Schlichting [29]) is used and the initial conditions are chosen to be appropriate to the case of a flat plate impulsively inserted into the free stream. The mesh used is displayed in Fig.4a, and the general features of the solution can be appreciated in the density contour plots shown in Fig.4b. The variation of the computed wall pressure distribution is given in Fig.4c.

CONCLUSIONS

It has been demonstrated how unstructured grids and high resolution schemes may be combined, yielding FEM-FCT. The numerical examples indicate that a high accuracy can be obtained economically for problems involving complex domains and/or adaptive mesh refinement. Furthermore, the 'equation-splitting' employed in classic FCT-codes [21,22] has been extended by coupling or 'synchronizing' the limiters of all the equations involved, without taking recourse to more costly Riemann-solver-based monotone schemes.

Extensions of the present work are under investigation and involve the development of better limiters for systems of equations in the context of FEM-FCT, the extension of FEM-FCT to implicit or semi-implicit time-stepping schemes [31], and the combination of FEM-FCT with unstructured multigrid methods [32] for the rapid solution of steady state problems.

ACKNOWLEDGEMENTS

It is a pleasure to acknowledge many fruitful and stimulating discussions during the course of the present work with Drs. J.P. Boris, D.L. Book and S.T Zalesak.

This work was funded by the Office of Naval Research through the Naval Research Laboratory, and by the Aerothermal Loads Branch of the NASA Langley Research Center.

REFERENCES

[1] P. Woodward and P. Colella - The Numerical Simulation of Two-Dimensional Fluid Flow with Strong Shocks; *J.Comp.Phys.* 54, 115-173 (1984).

[2] B. van Leer - Towards the Ultimate Conservative Scheme. II. Monotonicity and Conservation Combined in a Second order Scheme; *J.Comp.Phys.* 14, 361-370 (1974).

[3] P.L. Roe - Approximate Riemann Solvers, Parameter Vectors and Difference Schemes; *J.Comp.Phys.* 43, 357-372 (1981).

[4] S. Osher and F. Solomon - Upwind Difference Schemes for Hyperbolic Systems of Conservation Laws; *Math.Comp.* 38, 339-374 (1982).

[5] A. Harten - High Resolution Schemes for Hyperbolic Conservation Laws; *J.Comp.Phys.* 49, 357-393 (1983).

[6] P.K. Sweby - High Resolution Schemes Using Flux Limiters for Hyperbolic Conservation Laws; *SIAM J.Num.Anal.* 21, 995-1011 (1984).

[7] S.T. Zalesak - Fully Multidimensional Flux-Corrected Transport Algorithm for Fluids; *J.Comp.Phys.* 31, 335-362 (1979).

[8] J.P. Boris and D.L. Book - Flux-corrected Transport. I. SHASTA, a Transport Algorithm that works; *J.Comp.Phys.* 11, 38 (1973).

[9] D.L. Book, J.P. Boris and K. Hain - Flux-corrected Transport. II. Generalizations of the Method; *J.Comp.Phys.* 18, 248 (1975).

[10] J.P. Boris and D.L. Book - Flux-corrected Transport. III. Minimal-Error FCT Algorithms; *J.Comp.Phys.* 20, 397-431 (1976).

[11] G. Erlebacher - Solution Adaptive Triangular Meshes with Application to the Simulation of Plasma Equilibrium; *Ph.D. Thesis, Columbia University* (1984).

[12] A.K. Parrott and M.A. Christie - FCT Applied to the 2-D Finite Element Solution of Tracer Transport by Single Phase Flow in a Porous Medium; *Proceedings of the ICFD-Conf. on Numerical Methods in Fluid Dynamics*, Reading, Academic Press, 1986.

[13] R. Löhner, K. Morgan, M. Vahdati, J.P. Boris and D.L. Book - FEM-FCT: Combining High Resolution with Unstructured Grids; Submitted to *J.Comp.Phys.* (1986).

[14] K. Morgan, R. Löhner, J.R. Jones, J. Peraire and M. Vahdati - Finite Element FCT for the Euler and Navier-Stokes Equations; *Proc. 6th Int. Symp. Finite Element Methods in Flow Problems*, INRIA (1986).

[15] S.K. Godunov - *Mat. Sb.* 47, 271-306 (1959).

[16] J. Donea - A Taylor Galerkin Method for Convective Transport Problems: *Int.J.Num.Meth.Engng.* 20, 101-119 (1984).

[17] R. Löhner, K. Morgan and O.C. Zienkiewicz - The Solution of Nonlinear Systems of Hyperbolic Equations by the Finite Element Method; *Int.J.Num.Meth.Fluids* 4, 1043-1063 (1984).

[18] R. Löhner, K. Morgan and O.C. Zienkiewicz - An Adaptive Finite Element Procedure for High Speed Flows; *Comp.Meth.Appl.Mech.Eng.* 51, 441-465 (1985).

[19] R. Löhner, K. Morgan, J. Peraire and O.C. Zienkiewicz - Finite Element Methods for High Speed Flows; AIAA-85-1531-CP (1985).

[20] R. Löhner, K. Morgan, J. Peraire, O.C. Zienkiewicz and L. Kong - Finite Element Methods for Compressible Flow; pp. 28-53 in *Numerical Methods for Fluid Dynamics* (K.W. Morton and M.J. Baines eds.), Oxford University Press, (1986).

[21] M.A. Fry and D.L. Book - Adaptation of Flux-Corrected Transport Codes for Modelling Dusty Flows; *Proc. 14th Int.Symp. on Shock Tubes and Waves* (R.D. Archer and B.E. Milton eds.), New South Wales University Press (1983).

[22] D.E. Fyfe, J.H. Gardner, M. Picone and M.A. Fry - Fast Three-Dimensional Flux-Corrected Transport Code for Highly Resolved Compressible Flow Calculations: *Springer Lecture Notes in Physics* 218, 230-234, Springer Verlag (1985).

[23] G. Sod - *J.Comp.Phys.* 27, 1-31 (1978).

[24] A. Harten and Zwaas - Self-Adjusting Hybrid Schemes for Shock Computations: *J.Comp.Phys.* 6, 568-583 (1972).

[25] A.I. Zhmakin and A.A. Fursenko - A Class of Monotonic Shock-Capturing Difference Schemes; *NRL Memo. Rep.* 4567, (1981).

[26] R. Löhner - An Adaptive Finite Element Scheme for Transient Problems in CFD: to appear in *Comp.Meth.Appl.Mech.Eng.* (1987).

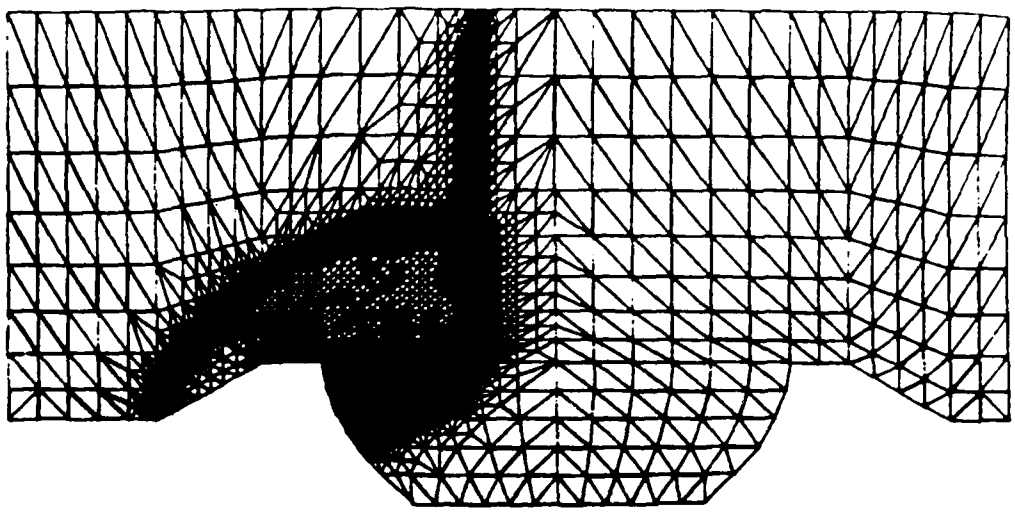
[27] J. Peraire, M. Vahdati, K. Morgan and O.C. Zienkiewicz - Adaptive Remeshing for Compressible Flow Computations; submitted to *J.Comp.Phys.* (1986).

[28] J.E. Carter - Numerical Solutions of the Navier-Stokes Equations for the Supersonic Laminar Flow Over a Two-Dimensional Compression Corner; *NASA Tech. Rep. R-385* (1972).

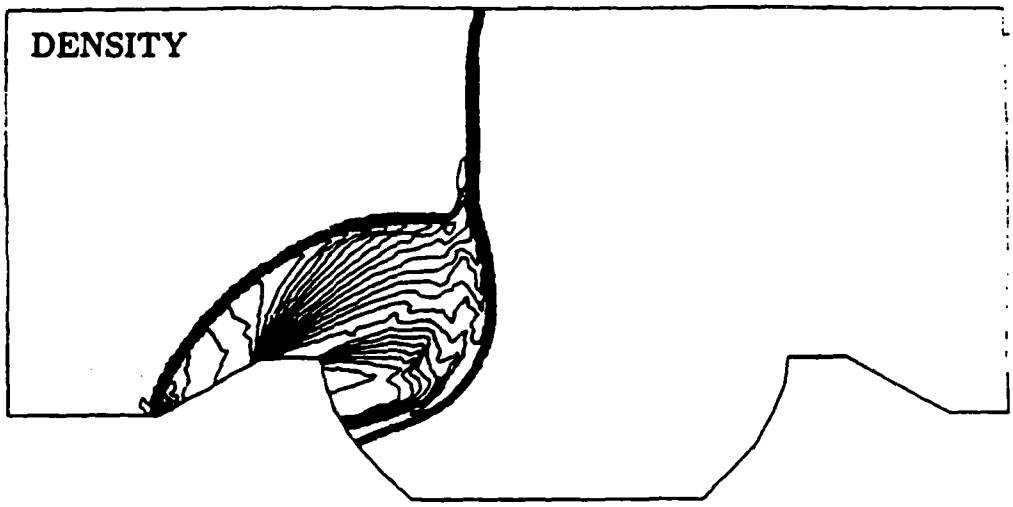
[29] H. Schlichting - *Boundary Layer Theory* ; Mc Graw Hill(1979).

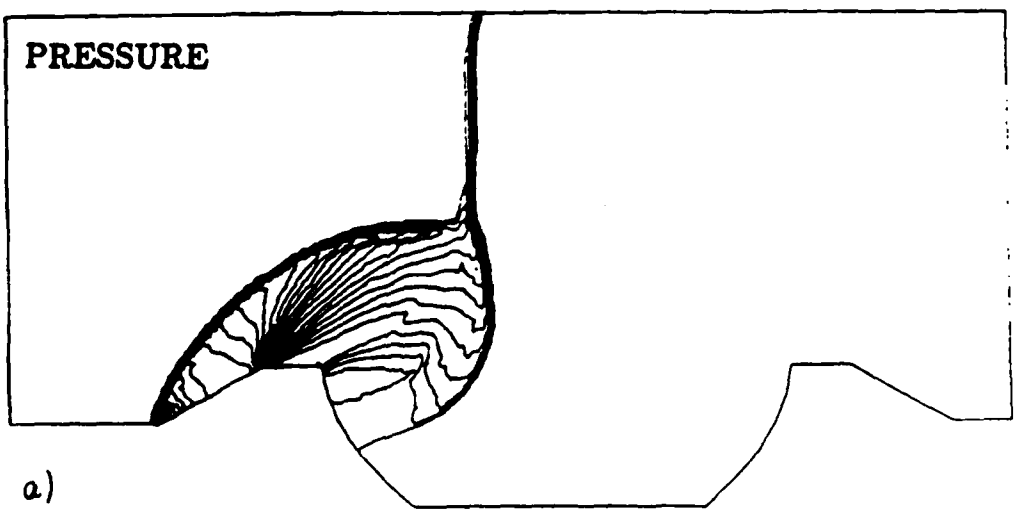
[30] G. Patnaik, R.H. Guirguis, J.P. Boris and E.S. Oran - A Barely Implicit Correction for Flux-Corrected Transport; to appear in *J.Comp.Phys.* (1987).

[31] R. Löhner and K. Morgan - An Unstructured Multigrid Method for Elliptic Problems; to appear in *Int.J.Num.Meth.Eng.* (1987).



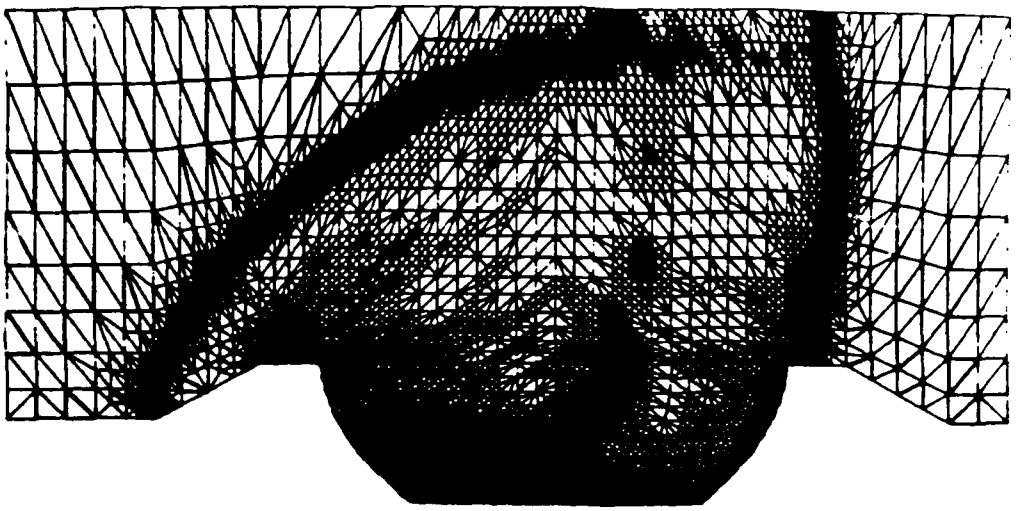
NELEM=5897, NPOIN=3021



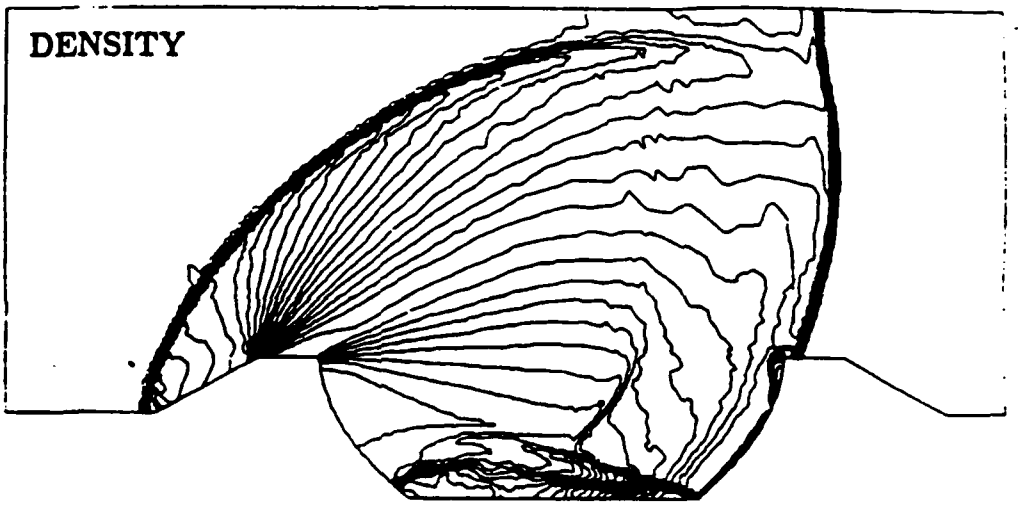


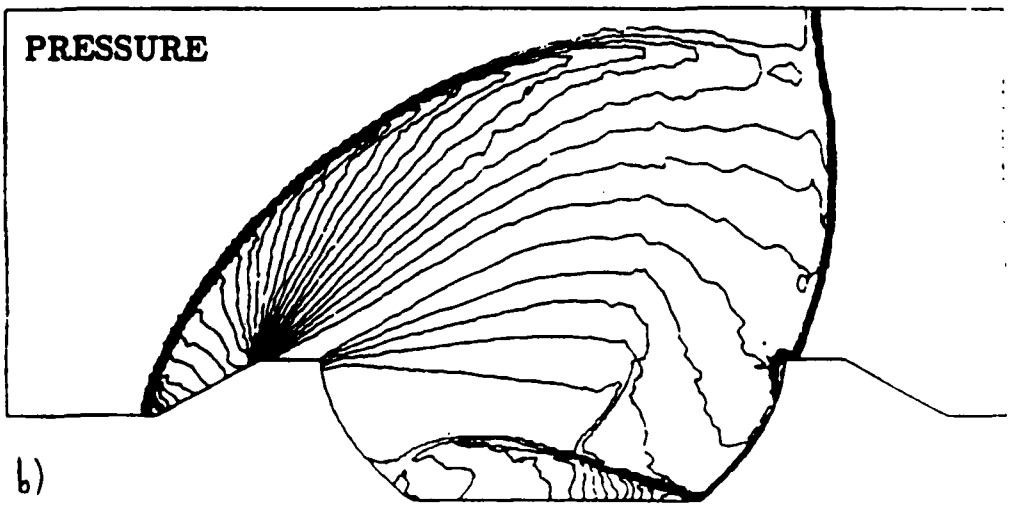
a)

T=0.04



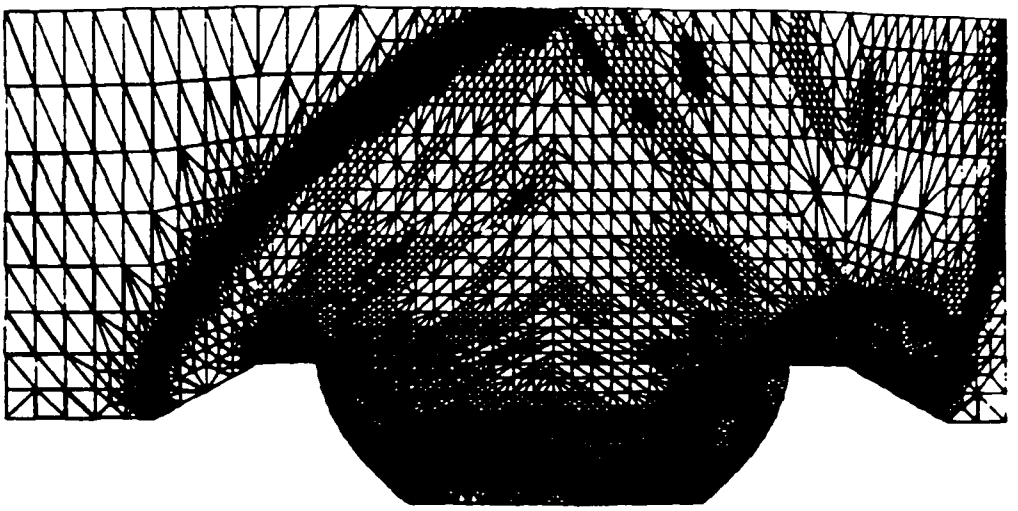
NELEM=11881, NPOIN=6073



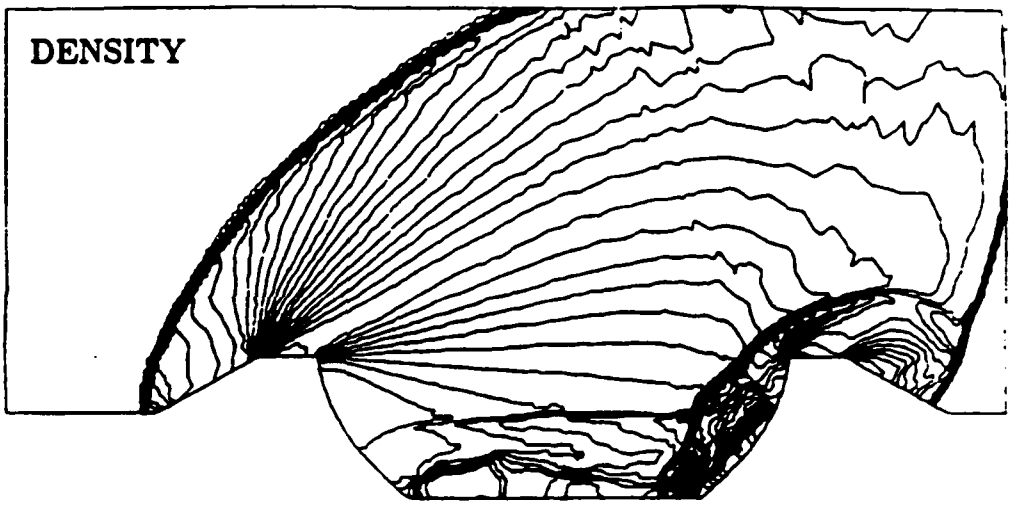


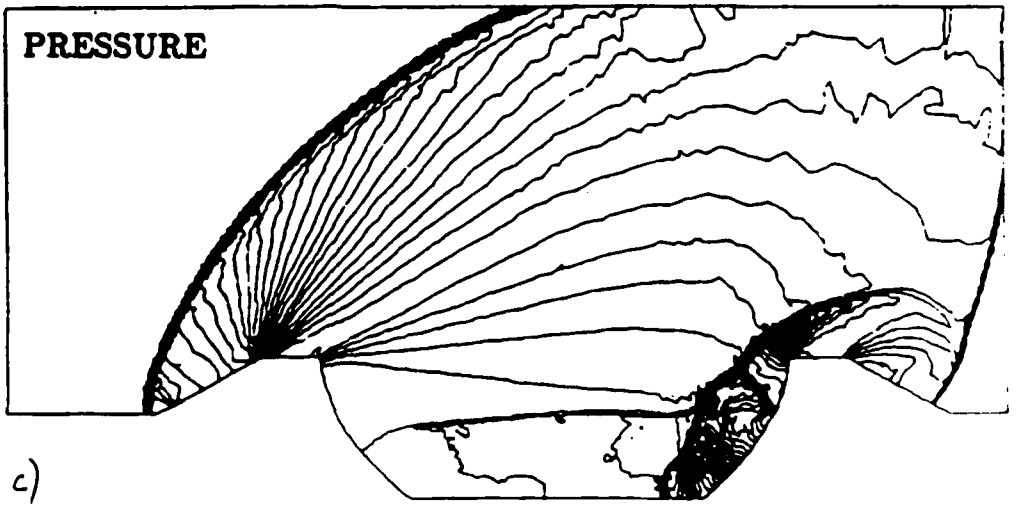
b)

T=0.08

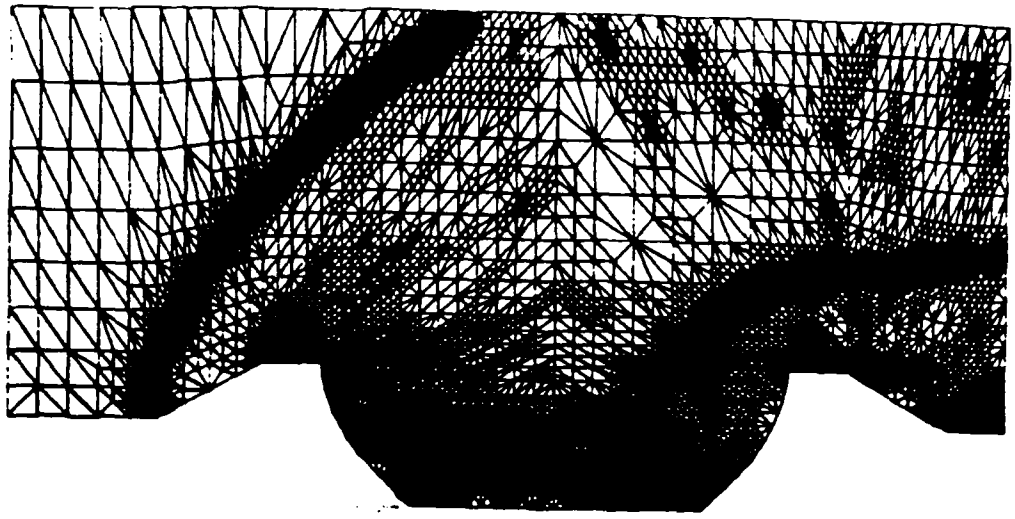


NELEM=13433, NPOIN=6862

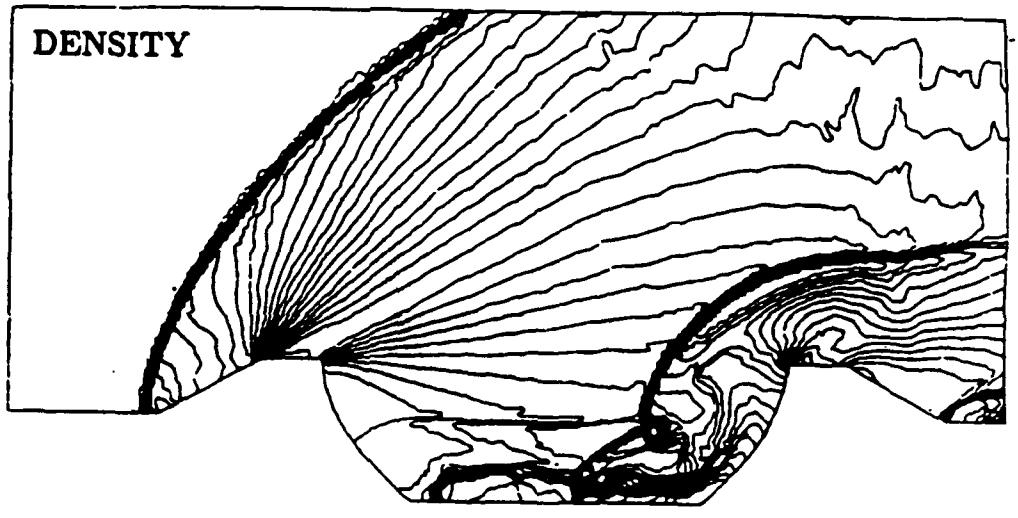


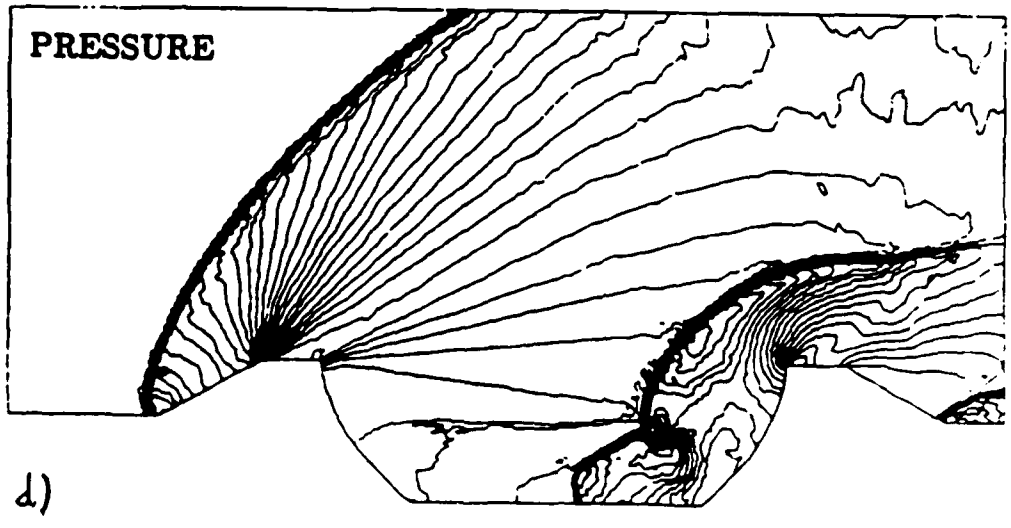


c)



NELEM=13852, NPOIN=7065





d)

T=0.12

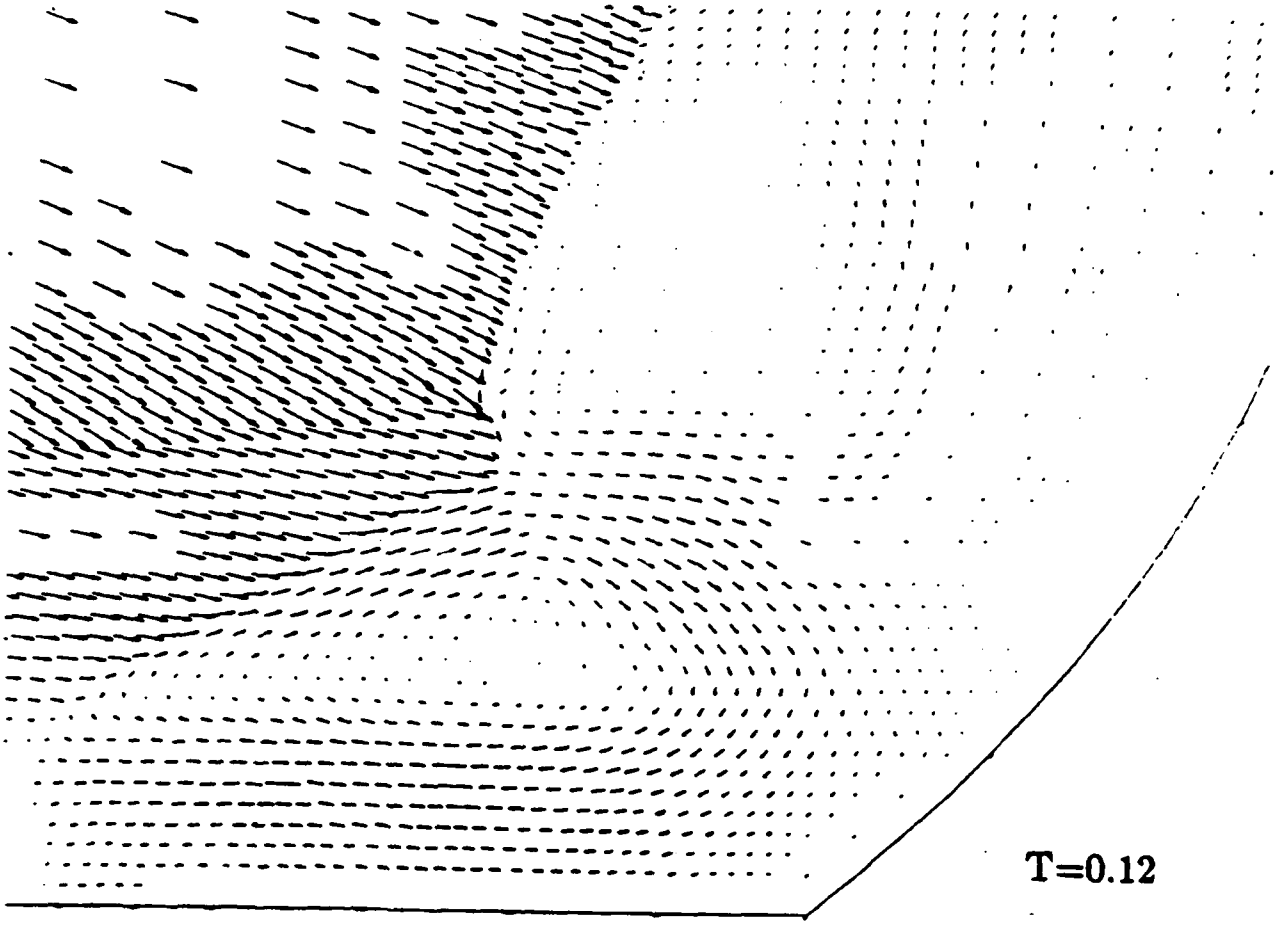


Figure 1e: Velocity vectors at T=0.12 (enlargement)

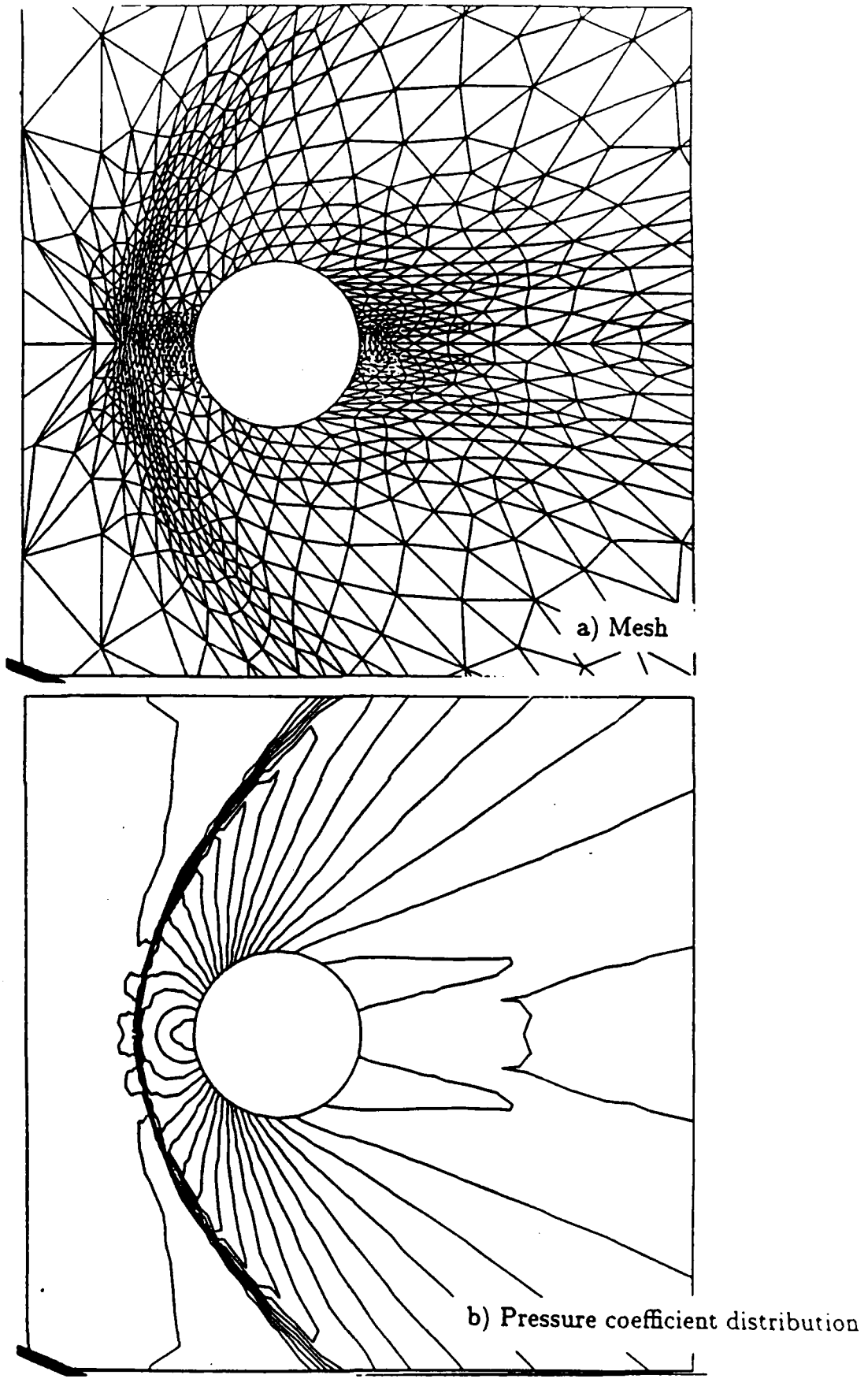


Figure 2: Steady supersonic flow past a cylinder

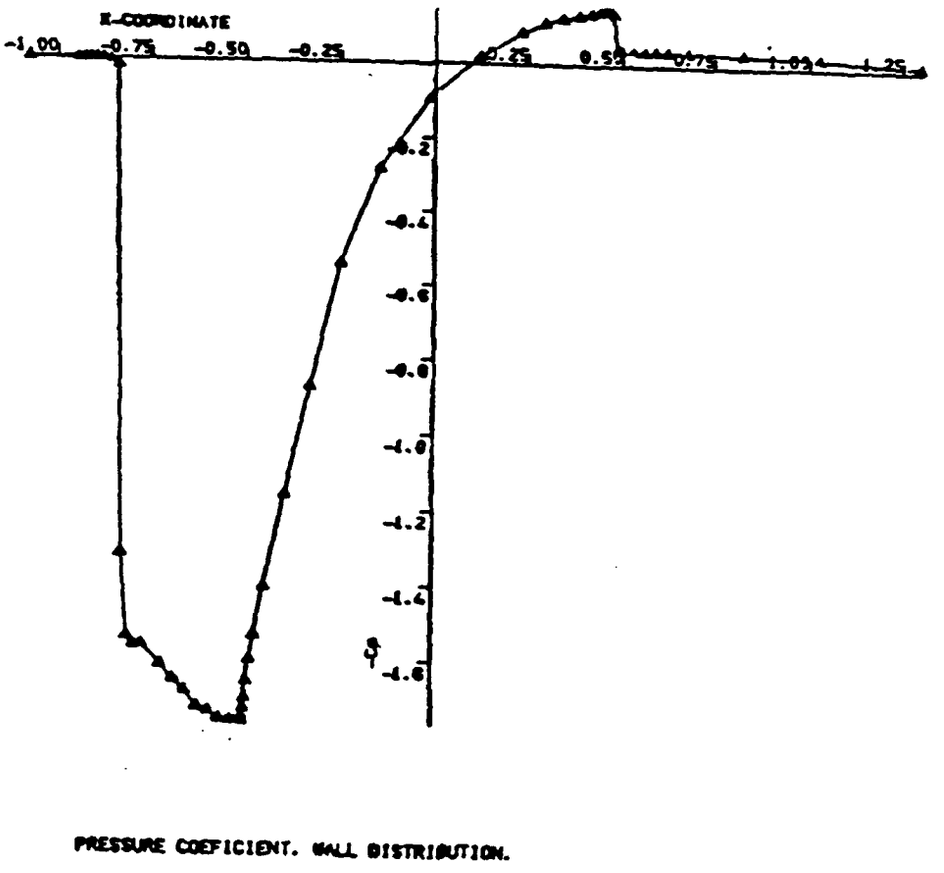
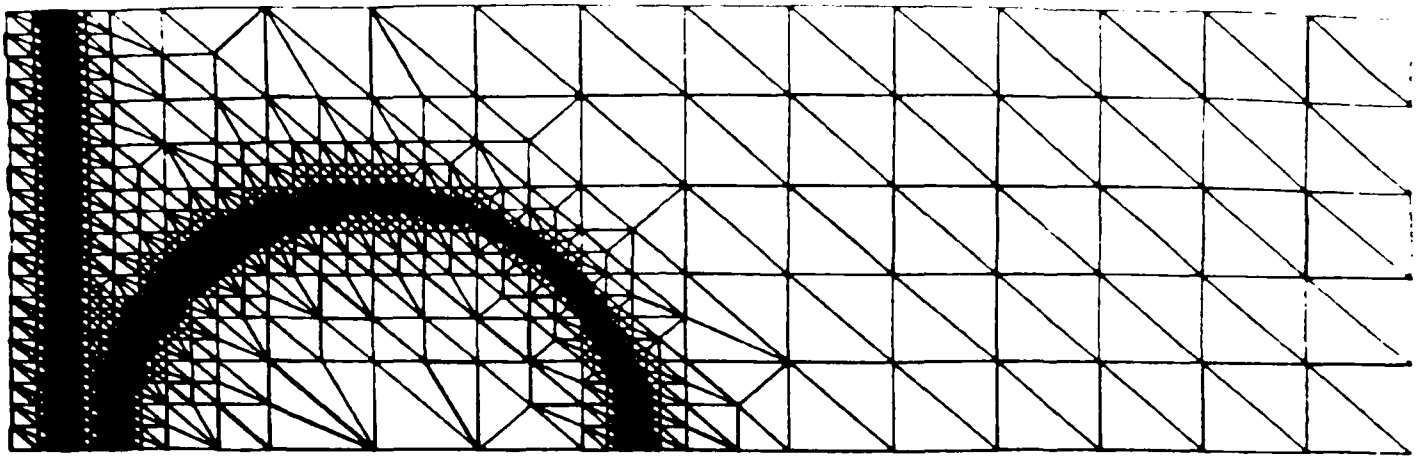


Figure 2: Steady supersonic flow past a cylinder

c) Variation of the pressure coefficient along the center line and over the cylinder surface



NELEM=6635, NPOIN=3374

$$\begin{bmatrix} \rho \\ u_1 \\ u_2 \\ e \end{bmatrix} = \begin{bmatrix} 1.3938 \\ 0.3475 \\ 0.0 \\ 2.1083 \end{bmatrix}$$

DENSITY

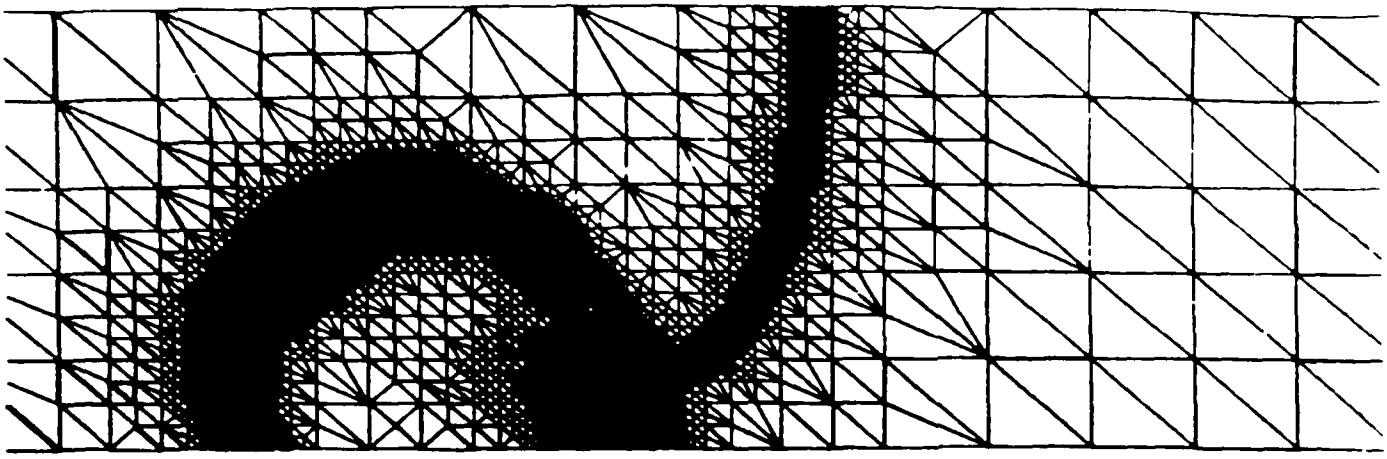
$$\begin{bmatrix} \rho \\ u_1 \\ u_2 \\ e \end{bmatrix} = \begin{bmatrix} 1.0 \\ 0.0 \\ 0.0 \\ 1.8758 \end{bmatrix}$$

$$\begin{bmatrix} \rho \\ u_1 \\ u_2 \\ e \end{bmatrix} = \begin{bmatrix} 2.8605 \\ 0.0 \\ 0.0 \\ 0.6243 \end{bmatrix}$$

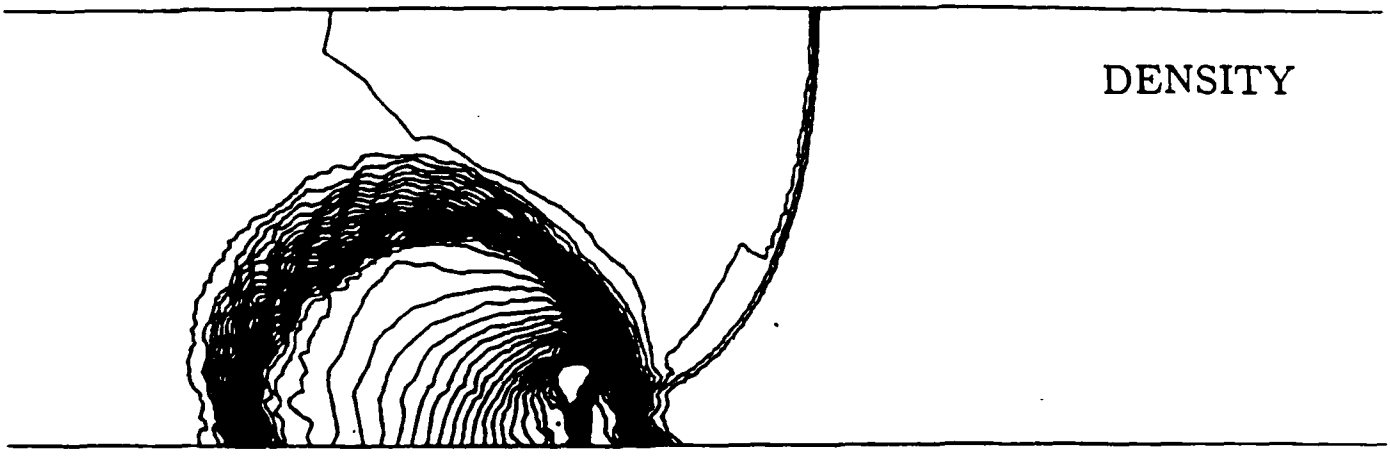
PRESSURE

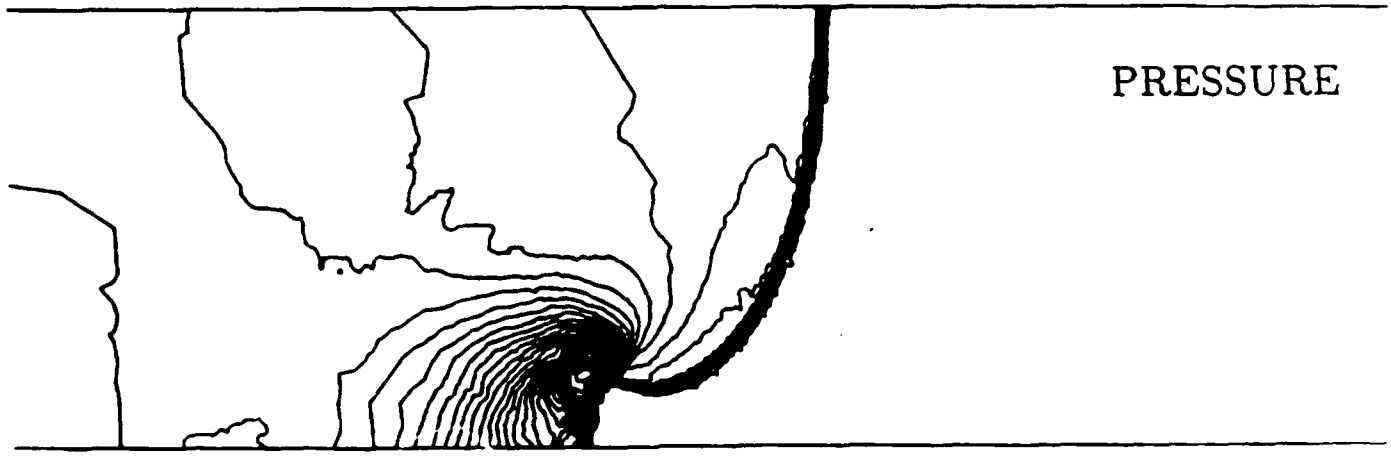
a) T=0.0

Figure 3: Shock-bubble interaction



NELEM=14528, NPOIN=7335

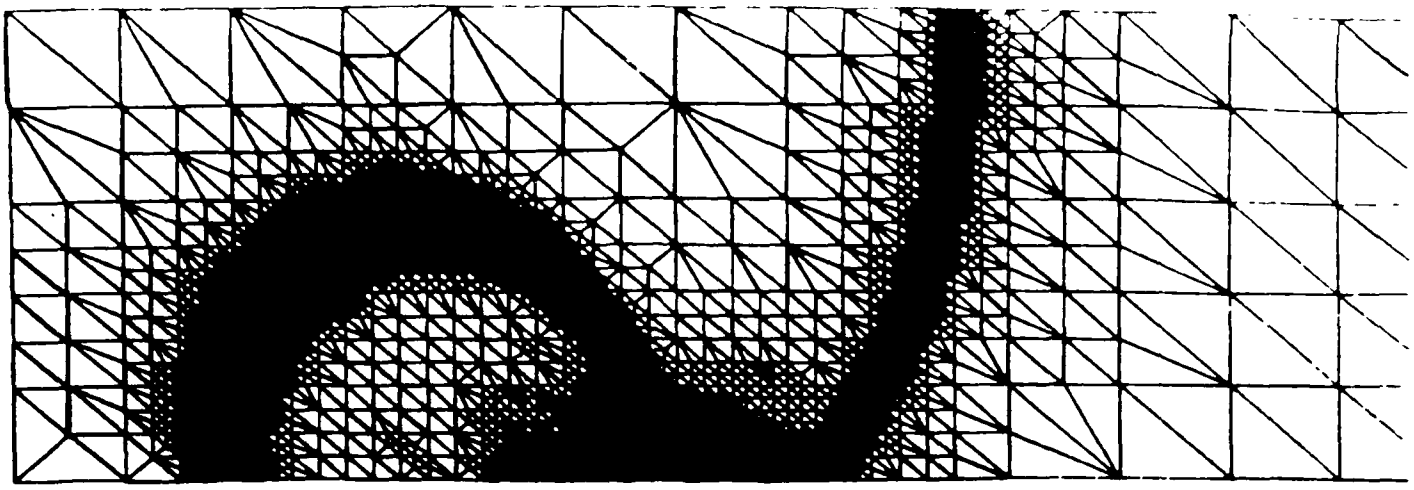




b)

T=0.6

Figure 3: Shock-bubble interaction (cont.)

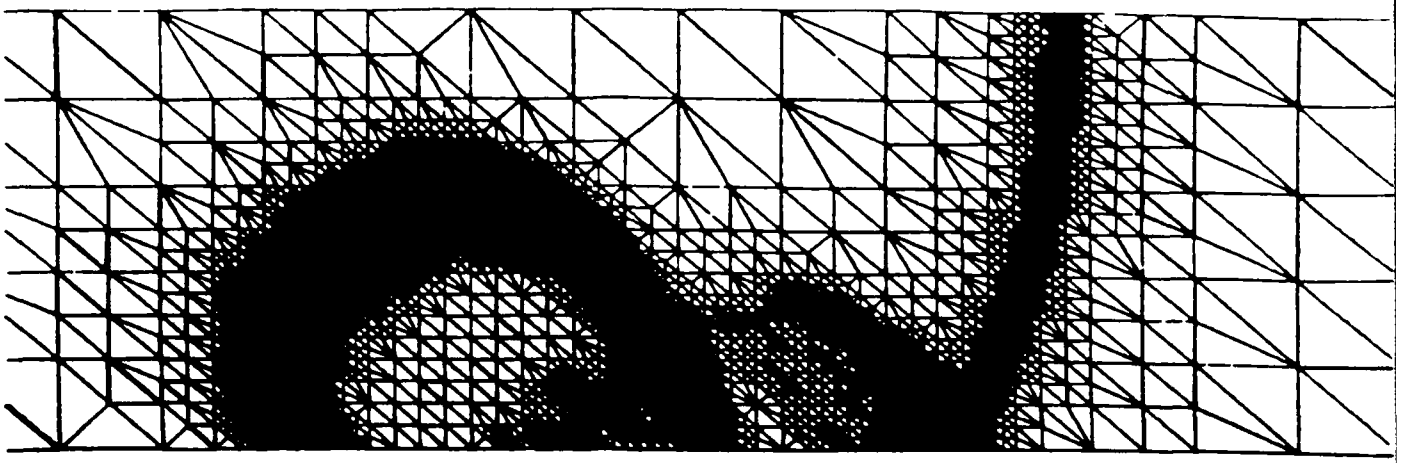


NELEM=16736, NPOIN=8464

c)

T=0.7

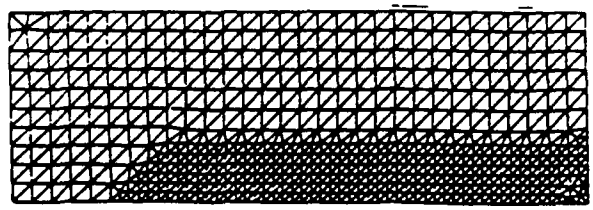
Figure 3: Shock-bubble interaction (cont.)



NELEM=16577, NPOIN=8386

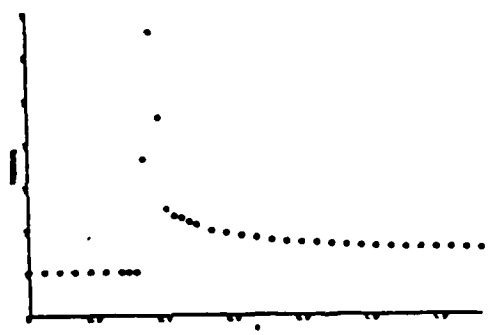
d) T=0.8

Figure 3: Shock-bubble interaction (cont.)



(a)

(b)



(c)

FIGURE 4 MACH 3 FLOW PAST A FLAT PLATE REYNOLDS No. 1000
(a) Mesh, (b) Density contours, (c) Pressure variation along the line of the plate

APPENDIX V.

Finite Elements in CFD: What Lies Ahead

FINITE ELEMENTS IN CFD: WHAT LIES AHEAD

R. Löhner

Berkeley Research Associates
Springfield, VA 22150, USA
and
Laboratory for Computational Physics and Fluid Dynamics
Naval Research Laboratory
Washington, D.C. 20375, USA

Invited paper presented at the
First World Congress on Computational Mechanics
Austin, Texas, September 22-26, 1986

1. Introduction

Computational fluid dynamics (CFD) has always been at the forefront of the development of numerical techniques for the simulation of physical phenomena. Several reasons have contributed to this leadership among the many disciplines that can be numbered under the heading of 'Computational Mechanics'. The first is the inherently nonlinear behavior of fluids (advection, turbulence) which must be accounted for. The second is the mixed hyperbolic/elliptic character of the partial differential equations describing fluid motion. This mixed hyperbolic/elliptic character implies an increased algorithmic complexity. The third reason is the need of engineers designing new aeroplanes to obtain much more accurate performance estimates (and therefore more accurate results for the simulations) than their colleagues designing bridges. The fourth reason is the size of typical problems and follows from the first three. A problem in structural mechanics is considered large if it exceeds $5 \cdot 10^3$ nodes, whereas 'large' in CFD means more than $5 \cdot 10^5$ gridpoints.

For 25 years CFD grew around Finite Difference Methods, as these were simple to understand and code, easy to vectorize, and the structured grids typically associated with them described appropriately the simple geometric complexity of the fields that were solved. However, as computers became bigger and faster, attempts were made to simulate more and more complex flow domains, and it soon became clear that structured grids where not flexible enough to describe these domains. It was at this point in time that unstructured grids, and Finite Element Techniques - a natural way of discretizing operators on them - entered the scene of CFD. Since then, unstructured grids have become a driving force within CFD, and many new developments could only become a reality by using them, such as domain splitting, adaptive refinement by enrichment within the same grid, directional refinement and the solution of the flowfield around a complete aircraft with engines.

However impressive the entry of Finite Elements into the field of CFD may have been, many more developments are still needed in order to transform what are now still rudimentary methods into efficient predictive engineering tools. The aim of this paper is to point out what appear currently as the main shortcomings of Finite Element algorithms, so as to concentrate the efforts to remove them. I cannot review the whole literature in this field and cite every relevant paper, and therefore apologize at the beginning to all those whose efforts may not seem to receive due credit. For other review papers see [1-3].

The rest of the paper is divided as follows: after some general remarks on unstructured grids, individual aspects which appear during the construction of Finite Element algorithms are treated separately. These are: improved flow solvers, implicit schemes, unstructured multigrid methods, better adaptive refinement schemes and grid generation in 3-D. Then, the accent shifts from algorithmic considerations to coding: graphics in 3-D, exploitation of supercomputer hardware and the reduction of memory requirements.

2. Some General Remarks on Unstructured Grids

The accurate representation of arbitrary domains represents perhaps one of the most challenging problems in CFD. The magnitude of this problem does not become apparent in two dimensions, because

a) only a few singular points usually appear in the field (and may be ignored), and

b) due to the computer capacity now available a gross overmeshing in certain regions of the domain can still be handled.

However, anyone trying to mesh complicated geometries in three dimensions with structured grids will encounter singular lines (even for such simple components as wings (see, e.g. [82])), and the unavoidable cost of overmeshing can no longer be ignored (the result being coarse grids).

It is by now generally accepted that only unstructured grids are capable of describing accurately complicated geometries in 3-D. Two different levels of unstructuring are possible:

a) macro-unstructuring, where blocks of structured grids are combined to form an overall unstructured grid (these are the so-called zonal methods, see [4-7]), or

b) micro-unstructuring, in which case the point and element distribution can in principle be random.

Although macro-unstructuring is being actively pursued by several groups (e.g. [5,7]), the inherent structure at the difference-level precludes simple mesh refinement by local enrichment (which destroys the grid structure). Major problems will also appear when the region to be refined/enriched crosses zone-boundaries. Micro-unstructuring does not have these inherent limitations, but also has its disadvantages:

- Programming complexity increases: if the central DO-loop of the simplest Finite Element code encompasses 25 FORTRAN statements, the central DO-loop of Mac Cormack's scheme [8] only requires 2 FORTRAN statements.

- Due to the heavy use of GATHER/SCATTER operations, the maximum megaflop-rate that can be achieved on vector-machines is lower than the one corresponding to a structured-grid scheme (i.e. CPU-times increase). GATHER/SCATTER operations require between 2.5-5.0 times as much CPU as simple DO-loops on machines like the CYBER-205 and CRAY-XMP.

- Storage requirements increase, as element-to-node correspondences have to be stored. Moreover, more temporary arrays are needed when gathering/scattering.

- As the grid possesses neither lines nor surfaces, no optimal implicit schemes (like ADI) or splittings can be invoked, so that approximate factorization , now so popular among Finite Difference and Finite Volume researchers [31-34] cannot be employed. This fact also precludes the use of line/zebra/surface relaxation [41-43] as efficient smoothing operators in the context of multigrid procedures.

- As the difference-stencils obtained on these grids are inherently multidimensional, no extension of 1-D concepts into 2-D/3-D via splitting is possible. This renders most of the work done for TVD-schemes and limitors over the past decade [11-18] useless for Finite Element Methods. However, attempts to employ 1-D concepts along element sides or faces have met with some success [18a].

However great these disadvantages of micro-unstructured grids may appear, the advantages these grids offer by far outweigh them:

- any geometry can be described by them [9,10],

- mesh refinement either by movement [61-64], enrichment [65-75] or remeshing [76] presents no problems, and

- domain splitting [108,109] for transient problems can easily be performed.

These facts were realized years ago by the NRL/LCP team for free surface hydrodynamics and multiphase flow problems [113,114], and the French research teams at INRIA and Dassault. Much pioneering FEM-CFD work, leading to the first computation of transonic flow past a complete aircraft with engines [9] was performed by the French teams. Recently, Jameson [10] also presented results using unstructured grids, and this alone may indicate a turning point in the development of CFD.

After these introductory remarks, I will now focus on particular areas where rapid progress is needed (and expected).

3. Improved Flow Solvers

The quest for the 'ultimate conservative scheme' is an old and tedious task, and many a valiant CFD-knight has perished on the way to this holy grail. As stated above, for unstructured grids the extension of schemes from 1-D to 2-D/3-D cannot be performed by operator splitting. This means that only schemes that are truly multi-dimensional in nature can be used. This renders many of the TVD-schemes developed over the last decade useless for unstructured grids. Only two families of high resolution schemes which can be used on these grids are known to the author. These are the Petrov-Galerkin methods developed by Hughes et.al. [27-30], and the FCT-algorithms

of Boris and Book [20-22] as first generalized to multidimensional problems by Zalesak [23], and later extended in the context of Finite Elements by Erlebacher [24], Parrott and Christie [24a] and Löhner et.al. [25,26]. I will only review the latter class, as I am much more familiar with it, but urge anyone interested in Finite Elements to study the work of Hughes and his group at Stanford.

3.1 FCT Defined

Whenever solving a set of conservation laws given by a system of partial differential equations of the form

$$\frac{\partial U}{\partial t} + \frac{\partial F_a^i}{\partial x^i} = \frac{\partial F_v^i}{\partial x^i}, \tag{1}$$

where the advective fluxes F_a play a dominant role over the viscous fluxes F_v, discontinuities in the variables throughout the field may arise (e.g. shocks or contact discontinuities). It is well known that any scheme of order higher than one will produce nonphysical overshoots or ripples at sharp gradients or discontinuities, even if the system of partial differential equations (1) is linear (so-called 'Godunov theorem'). Very often, particularly for mildly nonlinear systems, these nonphysical overshoots can be ignored. However, for most problems, overshoots will eventually lead to numerical instability, and therefore have to be suppressed.

The idea behind FCT is to combine a high-order scheme with a low-order scheme in such a way that in regions where the variables under consideration vary smoothly (so that a Taylor expansion makes sense) the high-order scheme is employed, whereas in those regions where the variables vary abruptly the low-order scheme is favored.

The temporal discretization of Eqn.(1) yields

$$U^{n+1} = U^n + \Delta U, \tag{2}$$

where ΔU is the increment of the unknowns at time $t = t^n$ obtained for a given scheme. Our aim is to obtain a ΔU of as high an order as possible without introducing overshoots. To this end, we re-write Eqn.(2) as:

$$U^{n+1} = U^n + \Delta U^l + (\Delta U^h - \Delta U^l), \tag{3}$$

or

$$U^{n+1} = U^l + (\Delta U^h - \Delta U^l). \tag{4}$$

Here ΔU^h and ΔU^l denote the increments obtained by some high- or low-order scheme respectively, whereas U^l is the (ripple-free) solution at time $t = t^{n+1}$ of the low-order scheme. The idea behind FCT is to limit the second term on the right-hand side of Eqn.(4):

$$U^{n+1} = U^l + lim(\Delta U^h - \Delta U^l), \tag{5}$$

in such a way that no nonphysical over/undershoots are created.

It is at this point that a further constraint, given by the conservation law (1) itself must be taken into account: strict conservation on the discrete level should be maintained. The simplest way to guarantee this for the node-centered schemes considered here is by constructing schemes for which the sum of the contributions of each individual element (cell) to its surrounding nodes vanishes, i.e. 'all that comes in goes out'. This means that the limiting will have to be carried out in the elements (cells).

It is beyond the scope of this paper to describe the limiting procedure in more detail. The interested reader may find it in [23-26]. However, I will point out some of the problems still inherent in the method at its present stage of development:

- For systems of equations, no obvious or 'natural' way of limiting has been identified yet. Several possibilities have been explored, among them (for the Euler equations) operator splitting (treating each equation independently), the use of averages of the limitors for each equation, limiting based on some 'key variable' (pressure, entropy), and others (see [26]). The author nevertheless feels that this weak point of FCT-schemes deserves further theoretical investigation, taking into account Riemann invariants or maybe even Flux-Vector or Flux-Difference concepts.

- Entropy is not always monotonic for FCT. This may be due to the low- order scheme employed. It is obvious that the 'ultimate low-order scheme' is Godunov's scheme [11-14], but this scheme is much more expensive than the simple smoothers currently in use [25,25a,26]. A simpler, more efficient version of Godunov's scheme for unstructured grids should be developed.

- Steepeners for contact discontinuities: as contact discontinuities are linear, any scheme that does not possess a steepener will eventually flatten these out. The current version of the author's FCT-code contains a very crude steepener, based on the argument that the antidiffusive element contributions should only steepen gradients. Note that no physical argument leads to a distinction of linear and nonlinear discontinuities. Some kind of detection mechanism may usefully be incorporated for contact discontinuities.

4. Implicit Schemes

Implicit methods are needed when the fastest time-scale present in the system of equations is not physically relevant for the problem at hand, and much lower modes carry the important physical information. A typical example is the viscous flow past an airfoil, where the speed of sound limitation in the cells covering the boundary layer would impose severe timestep-restrictions for an explicit scheme without achieving any further accuracy.

Among the many implicit schemes that have been developed over the years, only the following three will be discussed further:

a) Briley and MacDonald [31] (or Beam and Warming [32]) scheme: this very popular scheme, in its basic form, can be re-written for unstructured grids. However, the approximate factorization used for structured grids must be replaced by the solution

of a full matrix (as in [33]). The solution of full matrices can only be attacked via unstructured multigrid methods, which are discussed separately in the next paragraph.

b) Mac Cormack's implicit two-step procedure [34]: this scheme makes heavy use of upwind-differencing, thus always assuming a structured grid, and for this reason cannot be used in the present context.

c) The Barely Implicit Procedure [35]: unlike the two former methods, the barely implicit procedure treats only the sound waves implicitly by solving a modified Poisson equation for the pressure. This implies that this scheme is only conditionally stable. However, instead of solving five coupled equations in 3-D for the Euler equations, only one needs to be solved. The resulting Poisson equation is again solved via unstructured multigrid methods (see below). The Barely Implicit Procedure has already been shown to be very useful for all compressible flows with low Mach-numbers, such as external aerodynamics of cars, boundary layers and low-speed flames.

For the solution of chemically reacting flows, the time-scales associated with the chemistry are orders of magnitude faster than the wave-speed scales of the fluid. The recourse taken here is to treat point-implicitly [36-38] the 'chemistry', while solving the 'fluid flow' as before. For steady state problems, the inversion of the whole Jacobian at each point seems to be the preferred solution algorithm [36,37], whereas for transient problems the implicit-explicit approach descibed in [38] appears as more advantageous. Both concepts can easily be incorporated into the Finite Element framework.

5. Unstructured Multigrid Methods

Multigrid methods combine two very desirable properties: they require the least amount of operations to solve large problems ($O(Nlog(N))$ for a problem with N grid-points), and their storage requirements are also low (again $O(Nlog(N))$ for a problem with N gridpoints). In 1985, Löhner and Morgan [39,40] advanced the concept of unstructured multigrid methods. It became clear that as the finest grid had to be unstructured in order to accurately represent the domain, it could not be obtained by subdivision of some coarser grid. Instead, a set of unrelated coarsening grids had to be employed. The reason why multigrid methods should still work on sets of unrelated, unstructured grids, - the same argument on which all multigrid methods base the convergence rate estimates - is that if the residual is smooth, any coarser grid should be able to 'see' it. In all other aspects, the theory follows exactly the lines of traditional multigrid methods [41-43].

5.1 The Elliptic Case

The solution of elliptic PDEs via multigrid methods is by now well understood [41-43], and rigorous theoretical estimates for the expected convergence rates are available [42]. The main difficulty that can appear for unstructured grids lies in the construction of efficient smoothers, as neither line- nor plane-relaxation are possible. If Jacobi-type smoothers [43] are employed, the convergence rate of the highest modes can degrade

seriously for highly stretched elements or diffusion tensors in which one direction is dominant [44]. Three different smoothing schemes are known to avoid this problem:

a) Use of supersteps: here, the simple Jacobi-smoothing is over-and underrelaxed alternatively using a Chebyshev-series [39,40,44-46]. Although not advisable for highly stretched grids, e.g. stretchings beyond 1:100, this method is very simple to code and lends itself easily to vectorization.

b) Solution of local problems: Instead of a tighter coupling of modes via line- or plane-relaxation, groups of elements are relaxed, producing the desired effect [47]. This method is applicable in all cases, but may not be vectorizable and also requires some software- overhead.

c) Element by element preconditioning: although the transfer of information in the element by element iterative solver [48,49] is local in nature and therefore cannot compete with multigrid methods, this scheme may prove useful as a preconditioner. The compression of the eigenvalue spectrum is achieved by multiplying the system matrix with the (local) element-matrix inverses where appropriate. Vectorization of this type of method should also be investigated further.

5.2 The Hyperbolic Case

For the hyperbolic case, the theory of multigrid methods is still far from complete. Although Ni's method [50-52a,55] has been shown to work well in many cases, Jameson's multigrid solver [53,54] seems to emerge as the more reliable. This is to be expected, since the Runge-Kutta timestepping allows more possibilities for choosing appropriate 'damping-sequences' than the Lax-Wendroff schemes. Of course, hybrid schemes which make use of both approaches can be devised [52a]. The combination of unstructured multigrid methods with Runge-Kutta timestepping for the Euler equations may be found in [56,57].

6. Better Adaptive Refinement Schemes

In nearly all advection-dominated problems discontinuities or regions with sharp gradients appear. The regions in which the flow variables vary abruptly are usually small and are surrounded by large portions of the field in which the flow varies smoothly. It therefore seems attractive to locally and adaptively refine the mesh where needed, until a preset tolerance for the error has been achieved. Because of the obvious advantages of adaptive refinement, this field is currently receiving increased attention in the literature [58-59]. Any adaptive refinement scheme consists of three different stages: determining what we want to achieve by refining the grid, developing an error indicator/estimator to detect the regions to be refined, and a refinement strategy, such as movement, enrichment or remeshing.

Instead of reviewing all the approaches that have so far been devised, I will limit myself to the following remarks:

a) Typically, one aims to have an equidistribution of the 'error' throughout the grid (see [58-81]).

b) Whole families of error indicators, based on different concepts, have been shown to be useful. Among the most popular are those based on the change of some 'indicator-variable' (e.g. entropy [71] or Mach-number [64]), those based on interpolation theory estimates [62,65-70,80,81] and the indicators based on Richardson extrapolation [78] (but see [70] for a more complete review).

c) For the accurate resolution of the flowfields at or near discontinuities p-enrichment (whereby the polynomial order of the approximation is increased) does not seem to be attractive. Besides, p-enrichment implies a considerable increase in software complexity. However, it may prove useful for boundary layers, where an essentially smooth flowfield needs to be resolved.

6.1 More Reliable Error Indicators

In a recent paper, Löhner [81] formulated a modified interpolation indicator which was specifically intended for flowfields in which discontinuities of widely varying strength are present. For a typical error estimator based on interpolation theory one would estimate the second derivatives at points, for linear elements of constant length h in 1-D:

$$e_i = h^{-2}|U_{i+1} - 2U_i + U_{i-1}|. \tag{6}$$

If several discontinuities of widely varying strength are present in the domain, one observes that the strongest will 'swallow' all the added elements (in the case of enrichment) or 'draw' all elements to itself (in the case of movement). In order to avoid these undesired tendencies Eqn.(6) is modified as follows:

$$E_i = \frac{|U_{i+1} - 2U_i + U_{i-1}|}{|U_{i+1} - U_i| + |U_i - U_{i-1}| + \epsilon\,[|U_{i+1}| + 2|U_i| + |U_{i-1}|]} \tag{7}$$

Note the following properties of this modified error indicator:

- By dividing the second derivative by the 'jumps' (gradients) the 'eating-up' effect in the presence of a very strong shock is avoided because only the value of the normalized H2-seminorm is of importance, not the magnitude of the H2-seminorm itself. It is interesting to note that normalising the second differences by the first differences is, in essence, what is done in TVD schemes.

- Normalizing in this way also has the advantage that the error indicator becomes dimensionless, so that more than one 'key-variable' can be used without dimensional problems.

- Moreover, the modified error indicator is now bounded ($0 \leq E_i < 1$), so that preset tolerances can be employed (this is of particular importance for transient problems).

- The terms following ϵ are added as a 'noise' filter in order not to refine 'wiggles' or 'ripples' which may appear due to loss of monotonicity. The value for ϵ thus depends on the algorithm chosen to solve the PDEs describing the physical process at hand.

The generalization of this error indicator to multidimensional situations may be found in [81]. Although this modified error indicator has proved very reliable in practise, the author feels that more theoretical work is needed.

6.2 Directional Refinement

At the 24th AIAA Aerospace Sciences meeting Löhner and Morgan [75] introduced the concept of directional refinement. It is based on the observation that in most flowfields the regions that ought to be refined are of lower dimensionality than the physical space in which the solution is sought (e.g. shocks in 2-D/3-D). Therefore, if thin, elongated elements parallel to these discontinuities could be generated during the adaptive refinement process, considerable savings in CPU and storage would be realized without sacrificing accuracy.

The first algorithm devised for this purpose was based on mesh enrichment [75], and turned out to be storage, CPU and software intensive. Since then, Palmerio and Dervieux [64] have tried to incorporate this concept into a mesh movement framework, while Peraire et.al. [76] have advanced the concept of refinement by remeshing. This last concept represents a new and powerful refinement strategy, combining in a very elegant way the advantages of mesh enrichment (such as versatility by the introduction of points and a coarse initial grid for steady state problems) and mesh movement (which produces the desired element shapes near shocks) . If it proves useful in 3-D (the search-problem needs to be addressed here), it may completely replace movement and enrichment as refinement strategies. Another major area of applications for fast remeshing algorithms is given by 3-D Free-Lagrange Methods [77], where the restructuring of the grid currently represents a major problem. Remeshing would be an ideal solution in this case.

6.3 Adaptive Refinement for Transient Problems

When solving transient problems, in which only a few discontinuities appear, adaptive refinement can also be useful in reducing storage and CPU-requirements. However, in comparison to steady state problems, further constraints need to be placed on the refinement algorithms in order to realize significant savings:

a) As the grid adaptation has to be performed many times, the adaptation algorithm must be fast, and therefore must lend itself to vectorization/parallelization.

b) As the grid adaptation process becomes an integral part of any code, the algorithm should not be storage intensive.

c) As the feature that has been refined may pass again (e.g. a shock reflection), the original grid should be recovered after the feature has passed.

So far, the only successful, general-purpose algorithms based on unstructured grids meeting these requirements have employed the classic h-enrichment strategy (subdivision of elements into smaller ones without raising the polynomial order of the approximation) [80,81], allowing only one level of refinement/derefinement per grid modification.

Of course, directional remeshing or even movement could be incorporated for those cases in which the discontinuities are fairly straight (for curved shocks that interact with each other only classic h-enrichment will yield acceptable solutions), but the interpolation problem must be solved (otherwise the shock-speeds will be wrong). Another potential problem may arise due to the (apparent) non-vectorizability of the remeshing algorithm. Obviously, this whole topic of adaptive refinement for transient problems represents one of the most dynamic ones in CFD, and many further innovations are expected.

7. Grid Generation

The fast generation of grids for arbitrary domains in three dimensions has been the focus of much research in recent years. A variety of different approaches have been investigated. The most promising seem to be: the macro-element approach [84,84a], Watson's algorithm [86-91] for Voronoi tesselations combined with a point distribution obtained by superposition of local (structured) grids [10], modified octree [93], on an advancing front [76,85,92], and from a regular background grid [94]. All of them have advantages and disadvantages, and none has been fast and simple enough to generate efficiently grids of the size needed in 3-D aerodynamic simulations. Most of these schemes have been used to generate grids of up to a few hundred elements, where search-overheads and storage limitations don't become noticeable. Even the generation of a small mesh of only 12000 gridpoints, surely close to the minimum needed to describe a Boeing 747, requires half an hour of CRAY-time [10]. However, one has to bear in mind that the first algorithms for the solution of partial differential equations were developed nearly 25 years ago and since then have been improved by hundreds of researchers worldwide. Grid generation for unstructured grids in 3-D, on the other hand, has been pursued for no more than 3-4 years by only a few individuals.

Reviewing the literature, the following conclusions can be drawn:

- Partially unstructured grids (the macro-element approach [84,84a]) do not offer enough flexibility to serve as the basis of a general mesh generator for complex domains, unless some major breakthrough in interactive graphical display is achieved.

- Use of regular background grids [94] is not advisable for unbounded problems, as element stretching and point clustering are not accommodated easily. However, this technique may prove useful for internal flow problems.

- The generation of points via superposition of local grids (mappings) [10] seems to offer the greatest flexibility at minimal cost. The point distribution for each local grid is obtained algebraically and is therefore very fast. The grids are chosen from a menu of possible local grids.

- The tetrahedrization of the domain via Watson's algorithm [86,87], as employed and modified in [10,88-91], is not advisable, as this algorithm is

a) suboptimal, requiring $O(N^{1.5})$ operations, and

b) the treatment of voids (wings, fuselage, etc.) in the fluid domain becomes both grid logic and CPU-intensive.

- It seems attractive to pursue the element generation on an advancing front à la van Phai [92], combining it with a fast neighbor finder (see, for example [39,95,95a]).

8. Graphics in 3-D

Rapid, userfriendly, flexible graphical display of results is a basic tool during all stages of CFD: grid generation, debugging and evaluation of results are rendered impossible without it. Yet, up to now, software packages tailored to the specific needs of CFD have been developed only for structured grids (see, e.g. [96]). There are inherent difficulties to any display of 3-D field data:

a) only planes or surfaces can be displayed, but never the whole field (this is a fundamental difference from 2-D simulations),

b) if the user specifies a surface that happens not to be a particular plane of the grid, 3-D interpolation becomes unavoidable. This implies increased search-operations (CPU-times), particularly for unstructured grids. Fast search algorithms and plotting algorithms for arbitrary (curved) surfaces with random point distributions need to be developed.

Fortunately, in modern 'graphics engines' like the Evans & Sutherland, Iris and Sun- workstations, the main CPU-intensive operations such as translation, rotation and hidden-line removal are hardware-coded, so that animation becomes possible. The main bottleneck appears to be the low tranfer-rate between 'host-CPU' and 'graphics engine'. Experience with an Evans & Sutherland attached to a VAX-780 indicates that for large 3-D problems the wait times for reading in and processing the data can become restrictive. This is not surprising, as the machines on which the hydro-codes are run are between 200-1000 times faster than a VAX-780. Faster and bigger pre- and post-processors are needed. The ratio between the hydro-computer and the graphics-computer should come down to about ten to one.

9. Exploitation of Supercomputer Hardware

However good a method may be, if it does not lend itself to some form of parallelism, its future will always remain a dubious one. The speed-up ratio between a code that exploits the machine hardware and one that does not lies between 1:10 and 1:20 on today's vector machines. This performance ratio will go up drastically when massively parallel computing becomes available [97,100]. Fortunately, the bigger the problem to be solved, the easier it is to exploit some inherent parallelism of an algorithm.

9.1 Vector Machines

On vector machines the important factors that determine the performance of an algorithm are DO-loop length and contiguity in memory (even on a CRAY !). The typical Finite Element code will have 'element subroutines'. This means that for each element, the contributions to the right-hand side or the stiffness-matrix are computed in turn. The vector-length thus obtained is typically of order 8-10, far too short to

exploit vector-machine hardware. The only way to achieve acceptable vector-lengths is to perform the assembly process on groups of elements, possibly the whole grid at once. This means that one-element-type codes should be favored, in contrast to the more usual many-element-type code now in use in industry.

Three different types of DO-loops are most often encountered:

a) Loops over the same type of data: these are loops which only involve one type of data (either point or element data), and are the 'favorites' of vector machines.

b) GATHER-loops: this type of loop appears when point information needs to be processed at the element level (a point may be shared by several elements). Most vector machines have hardware-tailored GATHER devices, but this type of loop will nevertheless run between 2.5-5.0 times slower than type a).

c) SCATTER-ADD: these loops occur when assembling element contributions at points (e.g. formation of a right-hand-side vector). As a point may receive contributions from several elements in the same loop, the simple SCATTER operation is not sufficient (see [101-103] for a more complete description of this problem). Coloring schemes have to be devised, and the original loop (over all the elements) has to be broken up into groups of elements, so that the use of straight SCATTER-operations becomes possible. This type of loop will run at roughly the same speed as the GATHER-loop.

9.2 Parallel Computing

When speculating about parallel computing, one ought to distinguish between mildly parallel machines (up to 10 processors) and massively parallel machines. The author only has experience with the former type of machine, and therefore the latter one will not be discussed further.

Examples of mildly parallel machines are the CRAY-XMP-48 and the systems of array processors attached to a host machine. At NRL-LCP one such system, called GAPS (Graphical Array Processor System) is now in use for production runs. It consists of four array processors attached via an APTEC to a VAX-780 and a Tektronix screen. The runs are monitored online on the Tektronix via the VOYEUR-software package, which allows study and evaluation of active data from mass-memory while the code is running. The speed attained with the system as it stands now is roughly one third of a CRAY-XMP-12, so that 8 hours of CRAY-time a day can be achieved. This is typical for this kind of set-up (see [104-107] for a more detailed description of GAPS and further examples).

From a user's point of view, the main difficulties facing the development of codes on such a system are the poor FORTRAN capabilities and the very small local memory of the presently available array processors, as well as the very bad debugging software. It therefore only pays off to re-write codes which have been thoroughly tested and will undergo no major modifications for these machines. In an area as dynamic as CFD, few codes ever make it to an array processor. However, the LCP FCT models have, and with great success [104,105].

From a theoretical point of view, all algorithms which split the domain into sufficiently large 'subdomains' are suited for mildly parallel machines. Examples of this kind are simple operator splitting (e.g. line by line, as long as the line is big enough), or the growing family of domain splitting algorithms [108,109].

10. Reduction of Memory Requirements

As stated earlier, schemes operating efficiently on unstructured grids require more memory than their structured-grid counterparts. This should not surprise us, because:

a) Much more information needs to be stored for an unstructured grid (e.g. the gridpoints corresponding to each element).

b) The typical vector length extends over the whole grid, so that all temporary (scratch) arrays, which are needed for vectorization, are comparatively long.

c) For structured-grid codes operator splitting is usually invoked, which means that the typical vector lengths are of the order of one line or plane, so that the space allocated for temporary arrays is negligible.

d) The GATHER/SCATTER-operations associated with unstructured grids require additional temporary arrays.

Typical figures quoted in the literature lie between 310 [110] and 610 [10] words per gridpoint, while the author's 3-D FEM-FCT code requires around 525.

Although the current trend in supercomputing indicates that the memory of these machines is increasing at a faster rate than the CPU (the CRAY-2 memory has 256 megawords of memory but is still slower than a CRAY-XMP-48), experience in a multiuser environment indicates that for fast turnaround, it pays to sacrifice CPU for memory.

Among the possibilities that can be pursued in order to reduce memory requirements, we mention:

a) Careful coding: an obvious possibility, but one that is usually considered only after a code has been shown to work (i.e. at a stage where as little as possible should be changed). The author was able to reduce the memory requirements for his 3-D FEM-FCT code from 525 to 315 words per gridpoint, while incurring a CPU increase of only 20%.

b) Splitting into subdomains: the peak efficiency of vector-machines is achieved for vector lengths that are only a fraction of the total number of gridpoints typically encountered when solving 3-D problems. The idea is then to save as much storage on temporal arrays as possible, performing all algorithmic steps (formation of right-hand sides, limiting, etc.) on subdomains. Experience here indicates that memory requirements can be reduced by about 50%, at the expense of some additional CPU and considerable code complication (the number of statements is increased by about 100% [103]).

c) Encasement of the unstructured grid in a structured grid: here, unstructuredness is only allowed close to the body, where the highest geometrical/physical complexity

is expected. At wider distances from the body a structured grid is employed. This approach holds considerable promise of reducing memory requirements by more than an order of magnitude, but will require sophisticated programming and mesh generation capabilities. It is currently pursued by several groups [111-112].

11. Conclusions and Outlook

The present review indicates that the following developments need to take place, as they represent the greatest shortcomings of FEM-CFD:

Finite Element Algorithms:

- Optimal limiting for systems of equations in the context of FEM-FCT [25,25a,26].
- Improved low-order schemes for FEM-FCT.
- Steepeners for unstructured grids.
- Extension of block-implicit methods [31-33] to unstructured grids. This mainly requires efficient smoothers to drive the unstructured multigrid solvers [33].
- Extension of the Barely Implicit Correction [35] scheme to unstructured grids. Efficient smoothers to drive the unstructured multigrid solvers have already been developed for the resulting Poisson equation.
- Further theoretical validation of error indicators.
- Development of efficient search procedures for adaptive remeshing [76] in three dimensions.
- Extension of adaptive remeshing and other directional refinement methods to transient problems and Free-Lagrange codes. The placement of further points, and strict conservation if interpolation is required remain open questions.
- Combination of the 'advancing front' concept [92] and a fast neighbor finder [44,95,95a] for the efficient generation of grids in three dimensions.

Finite Element Codes:

- Unified input/output software for FEM-codes.
- Faster pre/post-processor hardware.
- Coding of FEM-algorithms for parallel machines.
- Reduction of memory requirements by encasement of the unstructured grid inside a structured (outer) grid, or the use of structured grids inside the boundary layer regions [111].

Many other small improvements, not mentioned in the list above, are of course possible. The author has only tried to identify the main trends in FEM-CFD.

Although in comparison to structured grids everything seems more complicated when done on unstructured grids, the rewards of all the mentioned possible improvements should be well worth the efforts. We can without speculation look forward to the day when the computation of any new configuration will be accomplished not in a

matter of weeks but of days. At this time we can only follow the poet in saying: the best of unstructured grids and Finite Elements in CFD is yet to come.

12. Acknowledgements

I would like to acknowledge many fruitful and stimulating discussions during the course of the years with Profs. O.C. Zienkiewicz and K. Morgan, as well as Drs. J.P. Boris, D.L. Book, S.T. Zalesak and M.D. Salas.

13. References

[1] E.S. Oran and J.P. Boris - Detailed Modelling of Combustion Systems; Prog.Energy Combust.Sci., Vol. 7, 1-72 (1981).

[2] E. Turkel - Progress in Computational Physics; Comp.Fluids 11, 2, 121-144 (1983).

[3] A. Jameson - The Evolution of Computational Methods in Aerodynamics; ASME J.Appl.Mech.50, 1052-1069 (1983).

[4] L.W. Spradley, J.F. Stalnaker and A.W. Ratliff - Solution of Three-Dimensional Navier-Stokes Equations on a Vector Processor; AIAA J.19, 1302-1308 (1980).

[5] T.L. Holst, U. Kaynak, K.L. Gundy, S.D. Thomas and J. Flores - Numerical Solutionof transonic Wing Flows Using an Euler/Navier-Stokes Zonal Approach; AIAA-85-1640 (1985).

[6] J.A. Benek, P.G. Buning and J.L. Steger - A 3-D Chimera Grid Embedding Technique; AIAA-CP-85-1523 (1985).

[7] M.M. Rai - A Conservative Treatment of Zonal Boundaries for the Euler Equations; J.Comp.Phys. 62, 472-503 (1986).

[8] R.W. MacCormack - The Effect of Viscosity in Hypervelocity Impact Cratering; AIAA 69-354 (1969).

[9] M.O. Bristeau, O. Pironneau, R. Glowinsky, J. Periaux, P. Perrier and G. Poirier - Application of Optimal Control and Finite Element Methods to the Calculation of Transonic Flows and Incompressible Viscous Flows; pp.203-312 in Proc. IMA Conf. on Num.Meth. in Appl. Fluid Mech. (B. Hunt ed.), Academic Press (1980).

[10] A. Jameson, T.J. Baker and N.P. Weatherhill - Calculation of Inviscid Transonic Flow over a Complete Aircraft; AIAA-86-0103 (1986).

[11] B. van Leer - Towards the Ultimate Conservative Scheme. II. Monotonicity and Conservation Combined in a Second Order Scheme; J.Comp.Phys.14, 361-370 (1974).

[12] P.L. Roe - Approximate Riemann Solvers, Parameter Vectors and Difference Schemes; J.Comp.Phys.43, 357-372 (1981).

[13] S. Osher and F. Solomon - Upwind Difference Schemes for Hyperbolic Systems of Conservation Laws; Math.Comp.38, 339-374 (1982).

[14] P. Colella - Multidimensional Upwind Methods for Hyperbolic Conservation Laws; LBL-17023, Preprint (1983).

[15] A. Harten - High Resolution Schemes for Hyperbolic Conservation Laws; J.Comp.Phys.49, 357-393 (1983).

[16] P. Woodward and P. Colella - The Numerical Simulation of Two-Dimensional Fluid Flow with Strong Shocks; J.Comp.Phys.54, 115-173 (1984).

[17] P.K. Sweby - High Resolution Schemes Using Flux Limiters for Hyperbolic Conservation Laws; SIAM J.Num.Anal.21, 995-1011 (1984).

[18] H.C. Yee, R.F. Warming and A. Harten - J.Comp.Phys.57, 327-360 (1985).

[18a] V. Billey, J. Periaux, P. Perrier and B. Stoufflet - 2D and 3D Euler Computations with Finite Element Methods in Aerodynamics; Proc. First Int. Congress on Hyperbolic Problems, St. Etienne (1986).

[19] R. Löhner, K. Morgan, J. Peraire and O.C. Zienkiewicz - Finite Element Methods for High Speed Flows; AIAA-85-1531-CP (1985).

[20] J.P. Boris and D.L. Book - Flux-corrected Transport. I. SHASTA, a Transport Algorithm that works; J.Comp.Phys.11, 38- (1973).

[21] D.L. Book, J.P. Boris and K. Hain - Flux-corrected Transport. II. Generalizations of the Method; J.Comp.Phys.18, 248- (1975).

[22] J.P. Boris and D.L. Book - Flux-corrected Transport. III. Minimal-Error FCT Algorithms; J.Comp.Phys.20, 397-431 (1976).

[23] S.T. Zalesak - Fully Multidimensional Flux-Corrected Transport Algorithm for Fluids; J.Comp.Phys.31, 335-362 (1979).

[24] G. Erlebacher - Solution Adaptive Triangular Meshes with Application to the Simulation of Plasma Equilibrium; Ph.D. Thesis, Columbia University (1984).

[24a] A.K. Parrott and M.A. Christie - FCT Applied to the 2-D Finite Element Solution of Tracer Transport by Single Phase Flow in a Porous Medium; pp. 609-619 in Numerical Methods for Fluid Dynamics II (K.W. Morton and M.J. Baines eds.), Clarendon Press, 1986.

[25] R. Löhner, K. Morgan, M. Vahdati, J.P. Boris and D.L. Book - FEM-FCT: Combining High Resolution with Unstructured Grids; Submitted to J.Comp.Phys. (1986).

[25a] R. Löhner, K. Morgan, J. Peraire and M. Vahdati - Finite Element Flux-Corrected Transport (FEM-FCT) for the Euler and Navier-Stokes Equations; Submitted to Int.J.Num.Meth. Fluids (1986).

[26] R. Löhner, K. Morgan, M. Vahdati and J. Peraire - Further Generalizations of FCT; in preparation (1986).

[27] T.J.R. Hughes, L. Franca and M. Mallet - A New Finite Element Formulation for Computational Fluid Dynamics: I. Symmetric Forms of the Compressible Euler and Navier-Stokes Equations and the Second Law of Thermodynamics; Comp.Meth.Appl.Mech.Eng.54, 223-234 (1986).

[28] T.J.R. Hughes, M. Mallet and A. Mizukami - A New Finite Element Formulation for Computational Fluid Dynamics: II. Beyond SUPG; Comp.Meth.Appl.Mech.Eng.54, 341-355 (1986).

[29] T.J.R. Hughes and M. Mallet - A New Finite Element Formulation for Computational Fluid Dynamics: III. The Generalized Steamline Operator for Multidimensional Advective-Diffusive Systems; to appear in Comp.Meth.Appl.Mech.Eng. (1986).

[30] T.J.R. Hughes and M. Mallet - A New Finite Element Formulation for Computational Fluid Dynamics: IV. A Discontinuity Capturing Operator for Multidimensional Advective-Diffusive Systems; to appear in Comp.Meth.Appl.Mech.Eng. (1986).

[31] W.R. Briley and H. McDonald - Solution of the Multi-Dimensional Compressible Navier-Stokes Equations by a Generalized Implicit Method; J.Comp.Phys. 21, 372-397 (1977).

[32] R.M. Beam and R.F. Warming - An Implicit Finite Difference Algorithm for Hyperbolic Systems in Conservation-Law Form; J.Comp.Phys.22, 87-110 (1978).

[33] W. Mulder - Multigrid Relaxation for the Euler Equations; J.Comp.Phys.60, 235-252 (1985).

[34] R.W. MacCormack - A Numerical Method for Solving the Equations of Compressible Viscous Flow; AIAA J. 20, 1275-1281 (1982).

[35] G. Patnaik, J.P. Boris, R.H. Guirguis and E. Oran - A Barely Implicit Correction for Flux-Corrected Transport; submitted to J.Comp.Phys. (1986).

[36] T.R.A. Bussing and E.A. Murman - A Finite Volume Method for the Calculation of Compressible Chemically Reacting Flows; AIAA-85-0331 (1985).

[37] D.R. Eklund, J.P. Drummond and H.A. Hassan - The Efficient Calculation of Chemically Reacting Flow; AIAA-86-0563 (1986).

[38] T.R. Young and J.P. Boris - A Numerical Technique for Solving Stiff Ordinary Differential Equations Associated with the Chemical Kinetics of Reactive-Flow Problems; J.Phys.Chem.81, 2424-2427 (1977).

[39] R. Löhner and K. Morgan - Unstructured Multigrid Methods: First Experiences; Proc. of the Third International Conference on Numerical Methods in Thermal Problems, (R.W. Lewis et.al. eds.), Pineridge Press, Swansea (1985).

[40] R. Löhner and K. Morgan - An Unstructured Multigrid Method for Elliptic Problems; Proc. of the Second European Multigrid Conf., Köln, W. Germany, October 1985.

[41] K. Brand - Multigrid Bibliography, 3rd Edition; GMD, Bonn, W. Germany (1983).

[42] W. Hackbusch and U. Trottenberg (eds.) - Multigrid Methods; Lecture Notes in Mathematics 960, Springer Verlag (1982).

[43] Journal of Computational Physics, Vol. 48 (1982).

[44] R. Löhner and K. Morgan - An Unstructured Multigrid Method for Elliptic Problems; To appear in Int.J.Num.Meth.Eng. (1987).

[45] W. Gentzsch and A. Schlüter - Über ein Einschrittverfahren mit zyklischer Schrittweitenänderung zur Lösung parabolischer Differentialgleichungen; ZAMM 58, T415-T416 (1978).

[46] W. Gentzsch - Über ein verbessertes explizites Einschrittverfahren Lösung parabolischer Differentialgleichungen; DFVLR-Report (1980).

[47] C. Börgers and C.S. Peskin - A Lagrangian Method Based on the Voronoi Diagram for the Incompressible Navier Stokes Equations on a Periodic Domain; pp.87-113 in Springer Lecture Notes in Physics 238 (M.J. Fritts, W.P. Crowley and H. Trease eds.), Springer Verlag (1985).

[48] T.J.R. Hughes, M. Levit and J. Winget - Element-by-Element Implicit Algorithms for Heat Conduction; J. of Eng.Mech. 109,2 (1983).

[49] T.J.R. Hughes, M. Levit and J. Winget - An Element-by-Element Solution Algorithm for Problems in Structural and Solid Mechanics; Comp. Meth. Appl. Mech. Eng. 36, 241-254 (1983).

[50] R.H. Ni - A Multiple Grid Scheme for Solving the Euler Equations; AIAA J. 20. 1565-1571 (1982).

[51] G.M. Johnson - Multiple-Grid Acceleration of Lax-Wendroff Schemes; NASA TM-82843 (1982).

[52] W.J. Usab and E.M. Murman - Embedded Mesh Solutions of the Euler Equations Using a Multiple-Grid Method; pp. 447-472 in Advances in Computational Transonics (W.G. Habashi ed.), Pineridge Press, Swansea (1985).

[52a] M.G. Hall - A Node Centered Multigrid Method for the Euler Equations ? pp. ... in Numerical Methods for Fluid Dynamics II (K.W. Morton and M.J. Baines eds.), Clarendon Press (1986).

[53] A. Jameson, W. Schmidt and E. Turkel - Numerical Solution of the Euler Equations by Finite Volume Methods using Runge-Kutta Time-Stepping Schemes; AIAA-Paper 81-1259 (1981).

[54] E. Perez - Finite Element and Multigrid Solution of the Two Dimensional Euler Equations on a Non Structured Mesh; INRIA Rep. 442 (1985).

[55] R. Löhner, K. Morgan and L. Kong - An Unstructured Multigrid Method for the Compressible Euler Equations; Proc. of the 6th GAMM-Conf. on Num.Meth.

in Fluid Mech. (D. Rues, W. Kordulla eds.), Vieweg Notes on Numerical Fluid Mechanics, Vol 10, Vieweg Verlag (1986).

[56] E. Perez, J. Periaux, J.P. Rosenblum, B. Stoufflet, A. Dervieux and M.H. Lallemand - Adaptive Full-Multigrid Finite Element Methods for Solving the Two-Dimensional Euler Equations; Proc. 10th Int.Conf.Num.Meth. Fluid Dynamics, Peking, June 1986. Springer Verlag (to appear).

[57] D. Mavriplis and A. Jameson - Multigrid Solution of the Two-Dimensional Euler Equations on Unstructured Triangular Meshes; Paper presented at the First World Congress of Computational Mechanics, Austin, Texas, September 22-26 (1986).

[58] I. Babuska, J. Chandra and J.E. Flaherty (eds.) - Adaptive Computational Methods for Partial Differential Equations; SIAM Philadelphia (1983).

[59] I. Babuska et.al. (eds.) - Accuracy Estimates and Adaptive Refinements in Finite Element Computations; J. Wiley and Sons (1986).

[60] J.T. Oden - Notes on Grid Optimization and Adaptive Methods for Finite Element Methods; TICOM Rep.(1983).

[61] P. A. Gnoffo - A Finite-Volume, Adaptive Grid Algorithm Applied to Planetary Entry Flowfields; AIAA J.21, 1249-1254 (1983).

[62] A.R. Diaz, N. Kikuchi and J.E. Taylor - A Method for Grid Optimization for the Finite Element Method; Comp.Meth.Appl.Mech.Eng. 41, 29-45 (1983).

[63] K. Nakahashi and G. S. Deiwert - A Three-Dimensional Adaptive Grid Method; AIAA-85-0486 (1985).

[64] B. Palmerio and A. Dervieux - Application of a FEM Moving Node Adaptive Method to Accurate Shock Capturing; Proc. First Int. Conf. on Numerical Grid Generation in CFD, Landshut, W. Germany, July 14-17 1986. Pineridge Press.

[65] W. Schönauer, K. Raith and K. Glotz - The Principle of Difference Quotients as a Key to the Self-Adaptive Solution of Nonlinear Partial Differential Equations; Comp.Meth.Appl.Mech.Eng. 28, 327-359 (1981)

[66] J.F. Dannenhoffer and J.R. Baron - Adaptive Procedure for Steady State Solution of Hyperbolic Equations; AIAA-84-0005 (1984).

[67] J.F. Dannenhoffer and J.R. Baron - Grid adaptation for the 2-D Euler Equations; AIAA-85-0484 (1985).

[68] J.F. Dannenhoffer and J.R. Baron - Robust Grid adaptation for Complex Transonic Flows; AIAA-86-0495 (1986).

[69] R. Löhner, K. Morgan and O.C. Zienkiewicz - An Adaptive Finite Element Procedure for High Speed Flows; Comp.Meth.Appl.Mech.Eng. 51, 441-465 (1985).

[70] R. Löhner, K. Morgan, and O.C. Zienkiewicz - Adaptive Grid Refinement for the Compressible Euler Equations; Chapter 15 in Accuracy Estimates and Adaptive

Refinements in Finite Element Computations (I. Babuska et.al. eds.); J. Wiley and Sons (1986).

[71] B. Palmerio, V. Billey, A. Dervieux and J. Periaux - Self-Adaptive Mesh Refinements and Finite Element Methods for Solving the Euler Equations; Proc. ICFD Conf. on Numerical Methods for Fluid Dynamics, Reading, U.K., March 1985.

[72] F. Angrand, V. Billey, A. Dervieux, J. Periaux, C. Pouletty and B. Stoufflet - 2-D and 3-D Euler Flow Calculations with a Second-Order Accurate Galerkin Finite Element Method; AIAA-85-1706 (1985).

[73] D.G. Holmes and S.C. Lamson - Compressible Flow Solutions on Adaptive Triangular Meshes; Open Forum AIAA - Reno'86 - Meeting (1986).

[74] O.C. Zienkiewicz, J.P. de S.R. Gago and D.W. Kelly - The Hierarchical Concept in Finite Element Analysis; Comp.Struct. 16, 53-65 (1983).

[75] R. Lõhner and K. Morgan - Improved Adaptive Refinement Strategies for Finite Element Aerodynamic Computations; AIAA-86-499 (1986).

[76] J. Peraire, M. Vahdati, K. Morgan and O.C. Zienkiewicz - Adaptive Remeshing for Compressible Flow Computations; submitted to J.Comp.Phys. (1986).

[77] M.J. Fritts, W.P. Crowley and H. Trease (eds.) - The Free-Lagrange Method; Sringer Lecture Notes in Physics 238, Springer Verlag (1985).

[78] M.J. Berger and J. Oliger - Adaptive Mesh Refinement for Hyperbolic Partial Differential Equations; J.Comp.Phys.53,484-512 (1984).

[79] M.D. Smooke and M.L. Koszykowsky - Two-Dimensional Fully Adaptive Solutions of Solid-Solid Alloying Reactions; J.Comp.Phys.62,1-25 (1986).

[80] J.T. Oden, P. Devloo and T. Strouboulis - Adaptive Finite Element Methods for the Analysis of Inviscid Compressible Flow: I. Fast Refinement/Unrefinement and Moving Mesh Methods for Unstructured Meshes; preprint (1986).

[81] R. Lõhner - An Adaptive Finite Element Scheme for Transient Problems in CFD; submitted to Comp.Meth.Appl.Mech.Eng. (1986).

[82] L.E. Eriksson - Practical Three-Dimensional Mesh Generation Using Transfinite Interpolation; CFD Lecture Series Notes 1983-04, von Karman Inst., Brussels (1983).

[83] W.C. Thacker - A Brief Review of Techniques for Generating Irregular Computational Grids; Int.J.Num.Meth.Eng.15, 1335-1341 (1980).

[84] O.C. Zienkiewicz and D.V. Phillips - An Automatic Mesh Generation Scheme for Plane and Curved Surfaces by Isoparametric Co-ordinates; Int.J.Num.Meth.Eng.3. 519-528 (1971).

[84a] A. Ecer, J. Spyropoulos and J.D. Maul - A Three-Dimensional, Block-Structured Finite Element Grid Generation Scheme; AIAA J. 23, 10, 1483-1490 (1985).

[85] S.H. Lo - A New Mesh Generation Scheme for Arbitrary Planar Domains: Int.J.Num.Meth.Eng.21, 1403-1426 (1985).

[86] A. Bowyer - Computing Dirichlet Tesselations; The Computer Journal 24,2, 162-167 (1981).

[87] D.F. Watson - Computing the N-Dimensional Delaunay Tesselation with Application to Voronoi Polytopes; The Computer Journal 24,2, 167-172 (1981).

[88] M. Tanemura, T. Ogawa and N. Ogita - A New Algorithm for Three- Dimensional Voronoi Tesselation; J.Comp.Phys.51, 191-207 (1983).

[89] R.C. Kirkpatrick - Nearest Neighbor Algorithm; pp.302-309 in Springer Lecture Notes in Physics 238 (M.J. Fritts, W.P. Crowley and H. Trease eds.), Springer Verlag (1985).

[90] D.N. Shenton and Z.J. Cendes - Three-Dimensional Finite Element Mesh Generation Using Delaunay Tesselation; IEEE Trans. on Magnetics, MAG-21, 2535-2538 (1985).

[91] J.L. Coulomb, Y. du Terrail and G. Meunier - FLUX3D, a Finite Element Package for Magnetic Computation; IEEE Trans. on Magnetics, MAG-21, 2499-2502 (1985).

[92] N. van Phai - Automatic Mesh Generation with Tetrahedron Elements; Int.J.Num.Meth.Eng. 18, 237-289 (1982).

[93] M.A. Yerry and M.S. Shepard - Automatic Three-Dimensional Mesh Generation by the Modified-Octree Technique; Int.J.Num.Meth.Eng. 20, 1965-1990 (1984).

[94] W.C. Thacker, A. Gonzalez and G.E. Putland - A Method for Automating the Construction of Irregular Computational Grids for Storm Surge Forecast Models; J.Comp.Phys.37, 371-387 (1980).

[95] J.P. Boris - A Vectorized 'Nearest Neighbors' Algorithm of Order N Using a Monotonic Logical Grid; J.Comp.Phys. 66, 1-20 (1986).

[95a] S.G. Lambrakos and J.P. Boris - Geometric Properties of the Monotonic Logical Grid Algorithm for Near Neighbor Calculations. J.Comp.Phys. (1986). To appear.

[96] P. Buning and J.L. Steger - Graphics and Flow Visualization in Computational Fluid Dynamics; AIAA-CP-85-1507 (1985).

[97] R.W. Hockney and C.R. Jesshope - Parallel Computers; Adam and Hilger (1981).

[98] D. Book (ed.) - Finite Difference Techniques for Vectorized Fluid Dynamics calculations; Springer Series in Computational Physics, Springer Verlag (1981).

[99] W. Gentzsch - Vectorization of Computer Programs with Applications to CFD; Vieweg Notes on Numerical Fluid Mechanics, Vol 8, Vieweg Verlag (1984).

[100] W. Schönauer and W. Gentzsch (eds.) - The Efficient Use of Vector Computers with Emphasis on CFD; Vieweg Notes on Numerical Fluid Mechanics, Vol 9. Vieweg Verlag (1984).

[101] R. Diekkämper - Vectorized Finite Element Analysis of Nonlinear Problems in Structural Dynamics; pp.293-298 in Proc. Parallel Computing '83 (M. Feilmeier, G. Joubert and U. Schendel eds.), North Holland (1984).

[102] R. Löhner and K. Morgan - Finite Element Methods on Supercomputers: The Scatter Problem; Proc. NUMETA'85 Conf. (J. Middleton and G. Pande eds.), 987-990, A.A. Balkema, Rotterdam, 1985.

[103] R. Löhner, K. Morgan and O.C. Zienkiewicz - Effective Programming of Finite Element Methods for CFD on Supercomputers; pp.117-125 in The Efficient Use of Vector Computers with Emphasis on CFD (W. Schönauer and W. Gentzsch eds.), Vieweg Notes on Numerical Fluid Mechanics, Vol 9, Vieweg Verlag (1985).

[104] J.P. Boris - Supercomputing at the U.S. Naval Research Laboratory; Chapter in 'Optical and Hybrid Computing' (H. Szu ed.), SPIE publication, Seattle, Wash. (1986).

[105] J.P. Boris, E. Reusser and T.R. Young - The Graphical and Array Processing System (GAPS); NRL-Memo-Rep. (1986).

[106] E. Clementi, G. Corongiu and J.R. Dietrich - Parallelism in Computations in Quantum and Statistical Mechanics; Comp.Phys.Comm.37, 287-294 (1985).

[107] W.T. Thompkins - Experience with a Personal Size Supercomputer - Implications for Algorithm Development; pp.31-52 in Progress and Supercomputing in Computational Fluid Dynamics (E.M. Murman and S.S. Abarbanel eds.), Birkhäuser (1985).

[108] R. Glowinsky, Q.V. Dinh and J. Periaux - Domain Decomposition Methods for Nonlinear Problems in Fluid Dynamics; Comp.Meth.Appl.Mech.Eng.40, 27-109 (1983).

[109] R. Löhner, K. Morgan and O.C. Zienkiewicz - The Use of Domain Splitting with an Explicit Hyperbolic Solver; Comp.Meth.Appl.Mech.Eng. 45, 313-329 (1984).

[110] E.A. Thornton, R. Ramakrishnan and P. Dechaumpai - A Finite Element Approach for the Solution of the Three-Dimensional Euler Equations; AIAA-86-0106.

[111] K. Nakahashi - FDM-FEM Zonal Approach for Computations of Compressible Viscous Flows; Proc. 10th Int.Conf.Num.Meth.Fluid Dynamics, Beijing, July 1986. To appear in the Springer Lecture Notes in Physics Series (1986).

[112] E.M. Murman - personal communication.

[113] M.J. Fritts and J.P. Boris - J.Comp.Phys. 31,173 (1979).

[114] D.E. Fyfe, E.S. Oran and M.J. Fritts - Surface Tension and Viscosity with Lagrangian Hydrodynamics on a Triangular Mesh; submitted to Journal of Computational Physics (1986).

APPENDIX W.

FEM-FCT: Combining Unstructured Grids with High Resolution (submitted to J. Comp. Phys.)

submitted to J. Comp Phys

FEM-FCT : COMBINING UNSTRUCTURED GRIDS WITH HIGH RESOLUTION

R. Löhner

Berkeley Research Associates
Springfield, VA 22150, USA

K. Morgan and M. Vahdati

Institute for Numerical Methods in Engineering
University of Wales, Swansea SA2 8PP, U.K.

J.P. Boris and D.L. Book

Laboratory for Computational Physics
Naval Research Laboratory
Washington, D.C. 20375, USA

1.Abstract

We present the extension of Flux-Corrected Transport Schemes (FCT) to unstructured grids. The spatial discretization is performed via finite elements. In particular we have chosen triangular elements in two dimensions. The limiting procedure is based on Zalesak's [18] extension to more than one dimension of the FCT-schemes of Boris and Book [13,19,20]. The resulting scheme, FEM-FCT, is capable of resolving moving and stationary shocks within two elements, and several examples are given that demonstrate the accuracy attainable, even for complicated geometries.

2. Introduction

The solution of partial differential equations with dominant first derivatives (advection dominated problems) on domains of complex geometrical shape is of great practical importance in many branches of science and engineering. The transport (passive advection) equation, the inviscid Burgers equation, the shallow water equations and the Euler and Navier-Stokes equations fall under this category.

Finite difference and finite volume methods for the solution of this class of problems have reached a high degree of sophistication and are currently in widespread use [1-5]. However, due to the structured grids typically associated with these methods, the treatment of complicated domains [6] and the use of adaptive mesh refinement [7] become inherently difficult. Finite element methods, on the other hand, are usually implemented on completely unstructured grids, with consequent reduction in the complexity of mesh generation [8,9] and a straightforward implementation of adaptive mesh refinement [10]. Moreover, finite element methods have now been shown [11,12] to solve

convective PDEs as accurately as conventional finite difference/ finite volume methods. Thus, it now seems appropriate to explore the possibility of high resolution monotone schemes for unstructured grids. One-dimensional splitting is inhibited by the irregularity of the grid, so the scheme employed must be truly multi-dimensional in nature. A variety of high-resolution schemes can be found in the literature [13-17], but only Zalesak's generalization [18] of the 1-D FCT-schemes of Boris and Book [13,19,20] can be considered a multi-dimensional high resolution scheme. It therefore appears natural to extend these ideas to a 'finite element methodology for flux-corrected transport': hence the name FEM-FCT.

In a recent paper, Parrott and Christie [25] have proposed such an approach and have shown how it may be implemented for the case of scalar advection in two dimensions. Here, we consider the extension to systems of equations and demonstrate the viability of the process by solving a number of examples.

3. FEM-FCT defined

As the present algorithm was to a large extent inspired by Zalesak's paper [18], we will follow his exposition closely. In any finite element scheme, the notion of 'flux' as used by Zalesak disappears, and must be replaced by 'element contribution to a node (EC)'. Thus, as may be seen from Fig. 1, element E1 contributes to nodes I,J,K, while node I in turn is affected by contributions from the elements E1-E5.

Any FCT-technique consists of the following six steps :

1) Compute LEC: the 'low-order element contributions' from some low order scheme guaranteed to give monotonic results for the problem at hand;

2) Compute HEC: the 'high-order element contributions', given by some high order scheme;

3) Define the 'antidiffusive element contributions' :

$$AEC = HEC - LEC$$

4) Compute, for each gridpoint, the updated low-order solution :

$$U_I^l = U_I^n + \sum_{el} LEC, \quad I = 1, ..., NPOIN, \tag{1}$$

where the summation extends over all elements surrounding node I (see Fig. 1), $NPOIN$ is the total number of points in the grid, and the superscript l was introduced to define the low-order solution.

5) Limit or 'correct' the AEC in a manner such that U^{n+1} as computed in step 6 below is free of extrema not also found in U^l or U^n :

$$AEC^c = C_{el} * AEC, \quad 0 \leq C_{el} \leq 1; \tag{2}$$

6) Apply the limited AEC to the previously calculated low order contributions :

$$U_I^{n+1} = U_I^l + \sum_{el} AEC^e. \tag{3}$$

to obtain the solution at the end of the time-step.

4. The limiting procedure

We wish to limit the element contributions in such a way that U^{n+1} does not exceed some maximum U^{max} nor falls below some minimum U^{min} . We leave the determination of U^{max}, U^{min} until later. Using Zalesak's notation, we define the six quantities :

$P_I^{\pm}$: the sum of all positive (negative) element contributions to node I :

$$P_I^{\pm} = \sum_{el} \begin{Bmatrix} max \\ min \end{Bmatrix} (0, EC_{el}), \tag{4}$$

where the summation extends over all elements surrounding node I.

$Q_I^{\pm}$: the maximum (minimum) increment node I is allowed to achieve in step 6 above :

$$Q_I^{\pm} = U_I^{\begin{smallmatrix} max \\ min \end{smallmatrix}} - U^l. \tag{5}$$

$$R_{\pm}^{+} := \begin{cases} min(1, Q_{\pm}^{+}/P_{\pm}^{+}) & if \quad P^+ > 0,\ P^- < 0 \\ 0 \quad if \quad P_{\pm}^{+} = 0 \end{cases} \tag{6}$$

The last equation ensures that no greater portion than that given by the high-order scheme is added, and could be removed if artificial steepening is desired. Assuming that $Q^+ \geq 0$ (it must be), all three of the above (+)-quantities are positive, and R_I^+ represents the least upper bound on a coefficient which must multiply all the positive element contributions to node I in order to guarantee no overshoot at node I. The same applies to R^- with respect to undershoots.

We have to take the most conservative estimate in order not to create any numerical over/undershoots arising from the high order scheme. Within each element, we compute

$$C_{el} = min(element\ nodes) \begin{cases} R^+ & if \quad EC > 0, \\ R^- & if \quad EC < 0. \end{cases} \tag{7}$$

Finally, we obtain $U_I^{\begin{smallmatrix} max \\ min \end{smallmatrix}}$ in three steps :

a) maximum (minimum) nodal U of U^n and U^l :

$$U_I^* = \begin{Bmatrix} max \\ min \end{Bmatrix} (U_I^l, U_I^n) \tag{8}$$

b) maximum (minimum) of element nodes :

$$U_{el}^* = \begin{Bmatrix} max \\ min \end{Bmatrix} (U_A^*, U_B^*, ..., U_C^*), \tag{9}$$

where the count extends over all nodes of element *el*;

c) maximum (minimum) U of all element surrounding node I :

$$U_I^{\substack{max \\ min}} = \begin{Bmatrix} max \\ min \end{Bmatrix} (U_1^*, U_2^*, ..., U_m^*), \tag{10}$$

where the count extends over all elements surrounding node I.

This completes the description of the limiting procedure. Up to this point we have been completely general in our description, so that eqns.(1)-(10) may be applied to any FEM-FCT scheme. In what follows, we specialize the exposition to the FEM-schemes employed in the present work, describing the high and low-order schemes used.

5. The high-order scheme : Consistent-Mass Two-step Taylor Galerkin

As the high order scheme, we employ a two-step form of the one-step Taylor-Galerkin schemes described in [21,22]. The method has been expounded thoroughly in [10]. We will only give a brief description of it here. Given the system of equations

$$\frac{\partial U}{\partial t} + \frac{\partial F^i}{\partial x^i} = 0, \tag{11}$$

we advance the solution from t^n to $t^{n+1} = t^n + \Delta t$ as follows :

a) First step :

$$U^{n+\frac{1}{2}} = U^n - \frac{\Delta t}{2} \cdot \frac{\partial F^i}{\partial x^i}|^n \tag{12}$$

b) Second step :

$$\Delta U^n = U^{n+1} - U^n = -\Delta t \cdot \frac{\partial F^i}{\partial x^i}|^{n+\frac{1}{2}}. \tag{13}$$

The spatial discretization of (9) and (10) is then performed via the classical Galerkin weighted residual method [23,24]. We remark that at $t^{n+\frac{1}{2}} = t^n + \frac{1}{2}\Delta t$ both U and F are interpolated using piecewise constant functions, while at t^n and t^{n+1}

linear shape functions are employed [10]. In this way the values of $U^{n+\frac{1}{2}}, F^{n+\frac{1}{2}}$ may be readily obtained at element centers (no matrix operations are involved). For (13) the following system of equations is obtained:

$$M_c \cdot \Delta U^n = R^n, \tag{14}$$

where M_c denotes the consistent mass matrix [23], ΔU the vector of nodal increments, and R the added element contributions to each node. As M_c posesses an excellent condition number, Eqn.(14) is never solved directly, but iteratively:

$$M_l \cdot (\Delta U^n_{i+1} - \Delta U^n_i) = R - M_c \cdot \Delta U^n_i, \quad i = 0, .., niter, \quad \Delta U^n_0 = 0. \tag{15}$$

Here M_l denotes the (diagonal) 'lumped' mass-matrix [23], obtained by adding all off-diagonal elements in a row of M_c to the diagonal (thus conserving mass). Typically, niter=3 is sufficient [22]. We finally recast the converged solution of Eqn.(15) into the following form, which will be of use later on :

$$M_l \cdot \Delta U^h = R - (M_l - M_c) \cdot \Delta U^h. \tag{16}$$

Here we have introduced the superscipt 'h' for high-order scheme.

6. The low-order scheme: Lumped-Mass Taylor Galerkin plus Diffusion

The requirement placed on the low order scheme in any FCT-method is monotonicity. The low order scheme must not produce any artificial, or numerical 'ripples' or 'wiggles'. In the original SHASTA-code of Boris and Book [13] this was achieved by adding a 'viscosity'-term in 1-D to a Lax-Wendroff-like scheme of the form

$$d = c_d \cdot \frac{\partial}{\partial x}(h^2 \frac{\partial U}{\partial x}) \quad . \tag{17}$$

This gives a contribution of the form

$$D_i = c_d \cdot (U_{i-1} - 2 \cdot U_i + U_{i+1}) \tag{18}$$

for a regular spacing of gridpoints with element length h. Zalesak employed the donor-cell scheme as the low-order scheme. In the present situation, particularly for grids with distorted elements in more than one dimension, equations (17),(18) must be re-interpreted as an 'averaging' operator, which may be obtained by adding a modified Laplace-smoothing of the form

$$d = c_d \cdot \sum_i \frac{\partial}{\partial x^i} \left[\left(\sum_K |\frac{\partial N^K}{\partial x^i}| \right)^{-2} \cdot \frac{\partial U}{\partial x^i} \right]. \tag{19}$$

N^K denotes the shape-function of node K. Observe that this 'diffusion' is purely numerical and also grid-dependent. Its function, as stated previously, is to ensure the monotonicity of the low-order solution.

The FCT technique adds to the high order scheme enough of this diffusion to obtain monotonic results for the problem at hand. However, as a consistent mass-matrix is employed in the high-order scheme (which implies a multipoint-coupling of the right-hand side), the imposition of monotonicity becomes impossible. Monotonicity can nevertheless be achieved by using a lumped mass-matrix instead. As the terms originating from the discretization of the fluxes F^i in (8) are the same as in (11), the low order scheme can then be written as

$$M_l \cdot \Delta U^l = R + D, \tag{20}$$

where we have introduced the superscript 'l' for 'low-order'-scheme.

7. Computation of the antidiffusive element contributions

Subtracting (20) from (16) yields the equation

$$M_l \cdot (\Delta U^h - \Delta U^l) = R + (M_l - M_c) \cdot \Delta U^h - R - D, \tag{21}$$

or

$$M_l \cdot (\Delta U^h - \Delta U^l) = (M_l - M_c) \cdot \Delta U^h - D \tag{22}$$

Note that all terms arising from the discretization of the fluxes F^i in (8),(16),(20) have now disappeared.

8. Numerical examples for a single PDE

8.1 Passive advection of a square wave

This is the same example as was used by Boris and Book [13] to demonstate the accuracy and monotonicity of their FCT-schemes. As the equation being solved (the transport equation) is linear, both amplification- and phase-errors can be identified easily. The wave extends over 20 gridpoints, and is convected with a Courant-number of C=0.2 for 800 steps. In Fig. 2 we compare the solutions obtained when using, in the high-order sheme, a lumped mass-matrix and a consistent mass-matrix. Observe that the consistent mass-matrix gives better phase-accuracy. After 800 steps the initial discontinuity is spread over only 5 gridpoints.

8.2 Shock propagation for the Burgers equation in 1-D

This nonlinear advection example is used to assess the performance of the scheme for those cases in which inherent steepening (overtaking characteristics) is present. Initially, the unknown is set to u=1.0 to the left of the shock, and to u=0.0 elsewhere. A uniform grid with gridsize h=1.0, and a timestep of $\Delta t = 0.2$ is employed. In Fig. 3 we compare the solutions obtained when using, as the high-order sheme, a lumped mass-matrix and a consistent mass-matrix. Observe that again the consistent mass-matrix gives better results, producing a shock over at most two gridpoints.

8.3 Passive advection in 2-D

This is the same example that Zalesak [18] used to test his FCT-algorithms. Again, as the equation solved is linear, both amplification and phase-errors can be identified easily. The problem statement may be found in [18]. The mesh used for this case, as well as the results obtained are depicted in Fig. 4.

8.4 Propagation of a linear discontinuity in 2-D

This steady state example, taken from [26], is used to test the 'spreading' due to inherent diffusion of a discontinuity being propagated in a constant velocity field. The problem statement, as well as the solutions obtained for different flow angles are shown in Fig. 5. Observe that no spreading is obtained for the algorithms presented here, indicating full second-order spatial accuracy.

9. Limiting for systems of equations

As evident from the results presented above, very good results can be obtained in one and two dimensions for a single PDE. However, when trying to extend the limiting process to systems of PDEs, no immediately obvious or natural limiting procedure becomes apparent. So far, the FCT-codes used for production runs [27,28] have limited each equation separately, invoking operator-splitting arguments. However, this approach does not always give very good results, as may be seen from Sod's comparison of schemes for the Riemann problem [29], and has been a point of continuing criticism by those who prefer to use the more costly Riemann-solver-based TVD-schemes [14-17]. It would therefore seem preferable to introduce 'system character' for the limiter by combining the limiters for all equations of the system. Many variations are possible and can be implemented, giving different performance for different problems. Our numerical experience for the compressible Euler equations indicates that

a) treating each equation independently as in operator-split FCT is the least diffusive method, tending to produce an excessive amount of ripples in the non-conserved quantities (and ultimately also in the conserved quantities);

b) using the same limiter (C_{el}) for all equations produces much better results, seemingly because the phase errors for all equations are 'synchronized'; for the Euler equations the following two limiters seem to perform best:

b1) just the limiter obtained for the density (continuity equation): this may produce some undershoot at shock-fronts in velocity and energy, but is very economical;

b2) taking as limiter for all equations the minimum of the limiters obtained for the density and the energy ($C_{el} = min(C_{el}(density), C_{el}(energy))$) : this produces better results than the previous limiter, but also requires more operations.

10. Numerical results for the Euler equations.

10.1 Sod-problem (1-D):

This classic example, taken from [29], solves the Riemann-problem for the compressible Euler equations in 1-D. The same grid as in [29] is employed, and the solutions are shown at the same time (t=14.75). In Fig. 6 the results obtained by taking as limiters the operator-split case, the density and the min(density,energy) are shown. As one can see, the last limiter achieves the best overall results.

10.2 Circular Blast-wave (2-D): The problem statement, as well as the solutions obtained are shown in Fig. 7. A quadrant of a cylinder in the lower left hand corner was given a density of 10.0 and a pressure of 40.0, while the rest of the computational region was filled with density 1.0 and pressure 1.0. Because all grid points inside radius 5.1 were disturbed and all gridpoints outside were not, the surface of the cylinder on the finite element grid is not completely circular. This case was run to test the symmetry or 'circularity' of the numerical solution. The density for all points in the domain is shown in Fig. 7d plotted versus the radial distance from the origin, indicating a better symmetry than that usually achieved by time-split codes.

10.3 Shock-reflection at a wall:

This problem has also been used extensively to asess the accuracy of schemes used for the solution of steady state problems [30,31]. The problem statement, as well as the pressure distributions obtained for the original Taylor-Galerkin scheme and FEM-FCT are shown in Fig. 8. Observe that the shocks are captured so sharply that the underlying grid structure becomes visible in the contour-plots. The steady state solution for this problem took 300 iterations, the residuals dropping 4 orders of magnitude. Fig. 8d depicts the variation of the density along the line y=0.5, and, as one can see, no over/undershoots are present.

10.4 Flow past an aerfoil in transonic flow

This example shows that acceptable solutions can be achieved with the present algorithm for the transonic flow regime. The case at hand is a NACA-0012 aerfoil, and the Mach number at infinity and angle of attack were set to $M_{\text{inf}} = 0.85$ and $\alpha = 0.0$. The grid-point distribution was taken from Jameson's FLO52-code [32], and corresponds to a 96 by 16 mesh. The pressure distributions at steady state are depicted in Fig. 9, and the c_p -distributions on the airfoil-surface obtained for FEM-FCT and the original two-step Taylor-Galerkin scheme presented in [10] is given in Fig.9c. Although the solution achieved by FEM-FCT is better than that of the ordinary Taylor-Galerkin scheme, for steady state aerodynamic applications, where shocks are only locally important, the additional cost of the high-resolution schemes does not make them attractive for production runs. Adaptive refinement [10] is a much more effective way of obtaining sharp shocks for steady flows.

11. Conclusions

It has been demonstrated how unstructured grids and high resolution schemes may be combined, yielding FEM-FCT. The numerical examples indicate that a high accuracy can be obtained economically for problems involving complex domains and/or

adaptive mesh refinement. Furthermore, the 'equation-splitting' employed in classic FCT-codes [13,18,19,20] has been extended by coupling or 'synchronizing' the limiters of all the equations involved, without taking recourse to more costly Riemann-solver-based monotone schemes. Points which still need to be addressed are the development of a better theory for flux limiters for systems of equations and the combination of FEM-FCT with unstructured multigrid methods [33] for the rapid solution of steady state problems.

12. Acknowledgements

The authors would like to acknowledge the many fruitful discussions with S. Zalesak during the course of this work.

The first three authors would like to thank the Aerothermal Loads Branch of the NASA Langley Research Center for partial support of this research under Grant No. NAGW-478.

13. References

[1] P. Woodward and P. Colella, *J.Comp.Phys.* **54** (1984), 115.

[2] P.K. Sweby, *SIAM J.Num.Anal.* **21** (1984), 995.

[3] R.M. Beam and R.F. Warming, *J.Comp.Phys.* **22** (1978), 393.

[4] J.C. South - Recent Advances in Computational Transonic Aerodynamics; AIAA-85-0366 (1985).

[5] A. Jameson, *ASME J.Appl.Mech.* **50** (1983), 1052.

[6] J.F. Thompson, Z.U.A. Warsi and C.W. Mastin - Numerical Grid Generation; Elsevier (1985).

[7] M.J. Berger and J. Oliger, *J.Comp.Phys.* **53** (1984), 484.

[8] M.A. Yerry and M.S. Shephard, *Int.J.Num.Meth.Eng.* **20** (1984), 1965.

[9] S.H. Lo, *Int.J.Num.Meth.Eng.* **21** (1985), 1403.

[10] R. Lõhner, K. Morgan and O.C. Zienkiewicz, *Comp.Meth.Appl.Mech. Eng.* **51** (1985), 441.

[11] F. Angrand, V. Billey, A. Dervieux, J. Periaux, C. Pouletty and B. Stoufflet -2-D and 3-D Euler Flow Calculations with a Second-Order Accurate Galerkin Finite Element Method; AIAA-85-1706 (1985).

[12] R. Lõhner, K. Morgan, J. Peraire and O.C. Zienkiewicz - Finite Element Methods for High Speed Flows; AIAA-85-1531-CP (1985).

[13] J.P. Boris and D.L. Book, *J.Comp.Phys.* **11** (1973), 38.

[14] P.L. Roe, *J.Comp.Phys.* **43** (1981), 357.

[15] B. van Leer, *J.Comp.Phys.* **14** (1976), 361. (1974).

[16] A. Harten, *J.Comp.Phys.* **49** (1983), 357.

[17] S. Osher and F. Solomon, *Math.Comp.* **38** (1982), 339.

[18] S.T. Zalesak, *J.Comp.Phys.* **31** (1979), 335.

[19] D.L. Book, J.P. Boris and K. Hain, *J.Comp.Phys.* **18** (1975), 248.

[20] J.P. Boris and D.L. Book, *J.Comp.Phys.* **20** (1976), 397.

[21] J. Donea, *Int.J.Num.Meth.Engng.* **20** (1984), 101.

[22] R. Löhner, K. Morgan and O.C. Zienkiewicz, *Int.J.Num.Meth.Fluids* **4** (1984), 1043.

[23] O.C. Zienkiewicz - The Finite Element Method; McGraw Hill (1982).

[24] O.C. Zienkiewicz and K. Morgan - Finite Elements and Approximation; J. Wiley and Sons (1983).

[25] A.K. Parrott and M.A. Christie - FCT Applied to the 2-D Finite Element Solution of Tracer Transport by Single Phase Flow in a Porous Medium; to appear in the proceedings of the ICFD-Conf. on Numerical Methods in Fluid Dynamics, Reading, Academic Press, 1986.

[26] T.J.R. Hughes, M. Mallet and A. Mizukami - A New Finite Element Formulation for Computational Fluid Dynamics: II. Beyond SUPG; to appear in *Comp.Meth.Appl.Mech.Eng.* (1986).

[27] M.A. Fry and D.L. Book, in Proc. 14th Int.Symp. on Shock Tubes and Waves (R.D. Archer and B.E. Milton eds.), New South Wales University Press 1983.

[28] D.E. Fyfe, J.H. Gardner, M. Picone and M.A. Fry, in Springer Lecture Notes in Physics 218, p.230, Springer Verlag, Berlin 1985.

[29] G. Sod, *J.Comp.Phys.* **27** (1978), 1.

[30] H.C. Yee, R.F. Warming and A. Harten, *J.Comp.Phys.* **57** (1985), 327.

[31] P. Colella - Multidimensional Upwind Methods for Hyperbolic Conservation Laws; LBL-17023, Preprint (1983).

[32] A. Jameson, W. Schmidt and E. Turkel - Numerical Solutions of the Euler Equations by Finite Volume Methods using Runge-Kutta Time-Stepping Schemes; AIAA-Paper 81-1259 (1981).

[33] R. Löhner and K. Morgan - An Unstructured Multigrid Method for Elliptic Problems; Proc. of the Second European Multigrid Conf., Köln, W. Germany, October 1985. To appear.

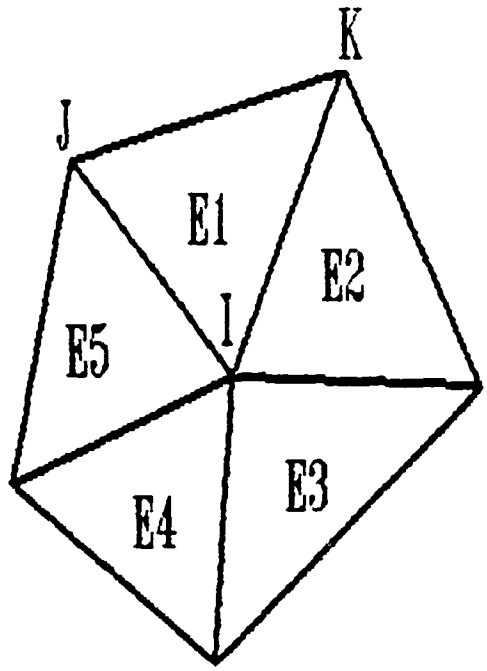
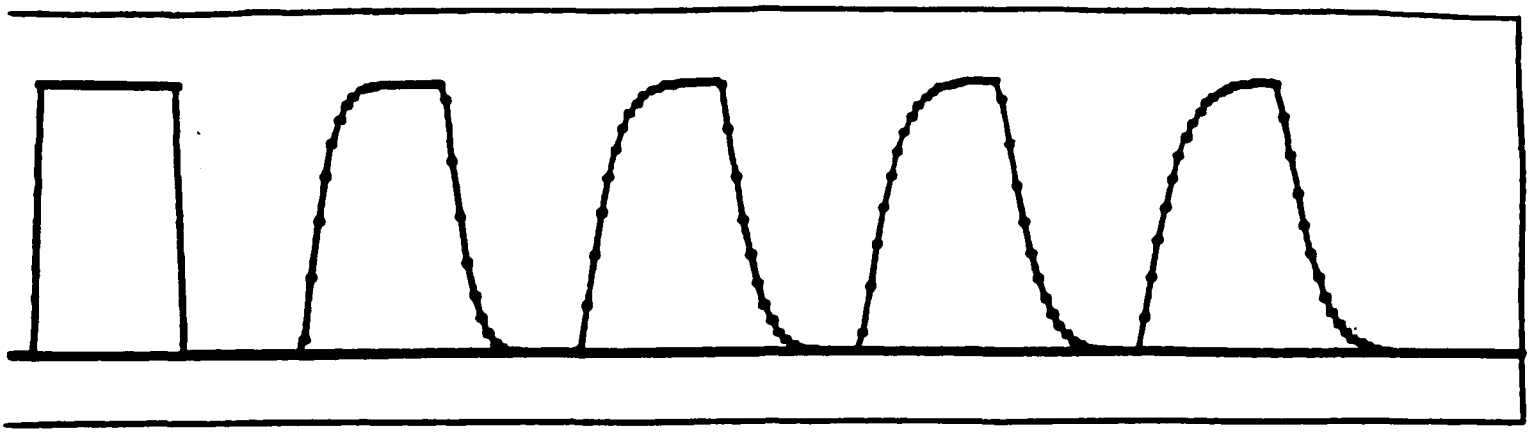
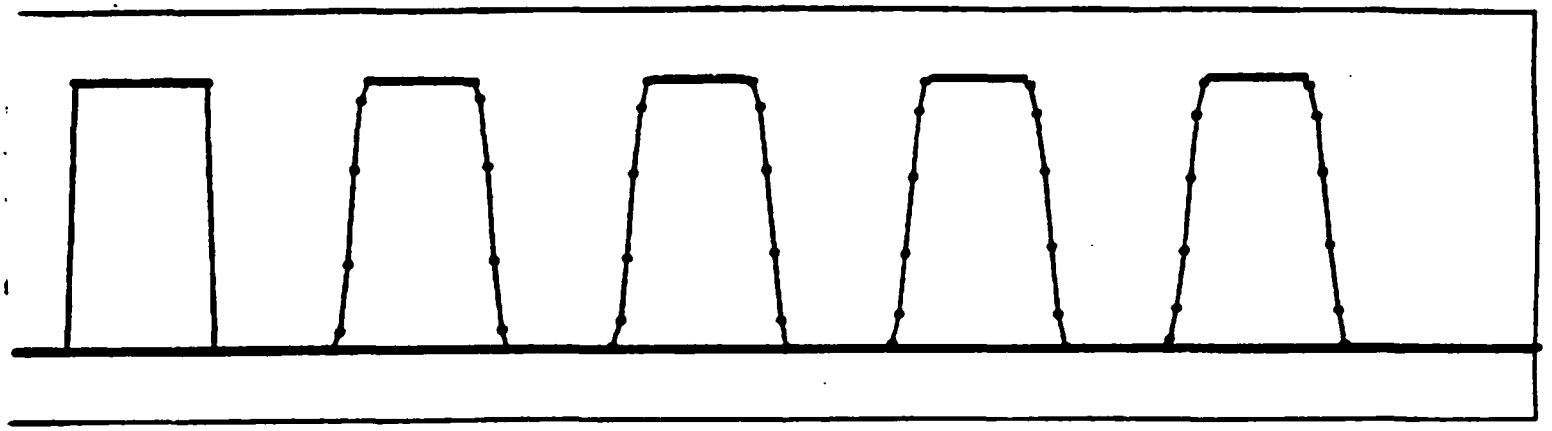


Figure 1: Definition of elements and nodes



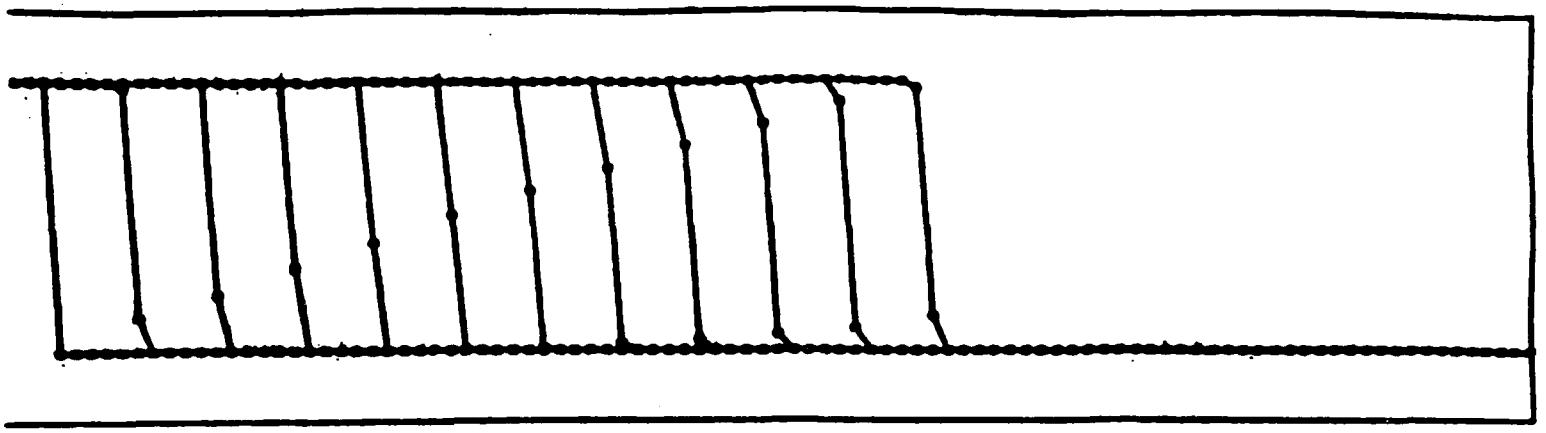
a) High-order scheme: Lumped-Mass Taylor Galerkin (niter=1)



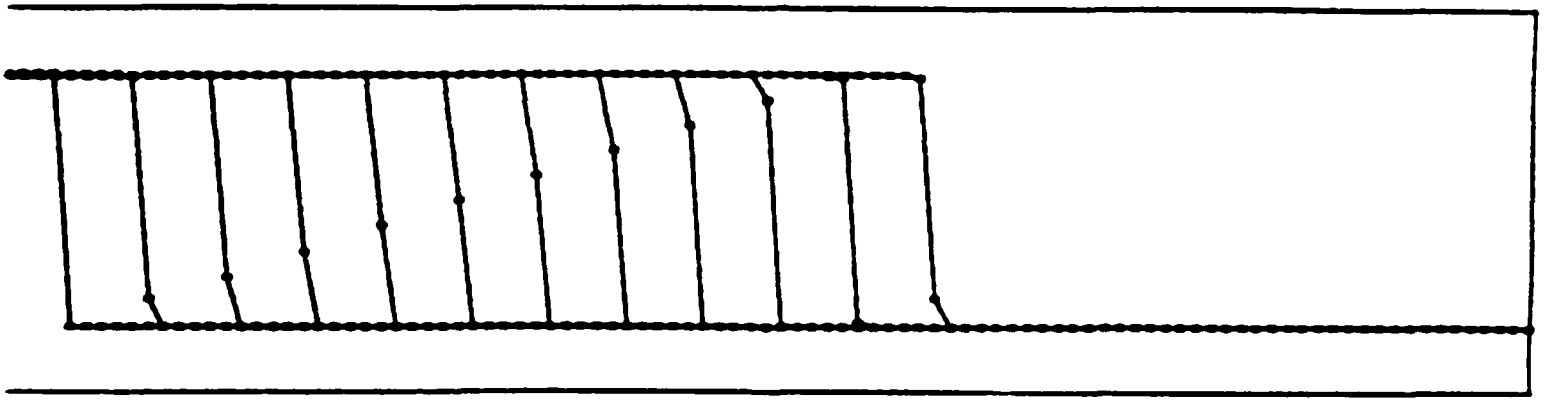
b) High-order scheme: Consistent-Mass Taylor Galerkin (niter=3)

Figure 2: Passive advection of a square wave (1-D)

C=0.2, plot every 200 steps



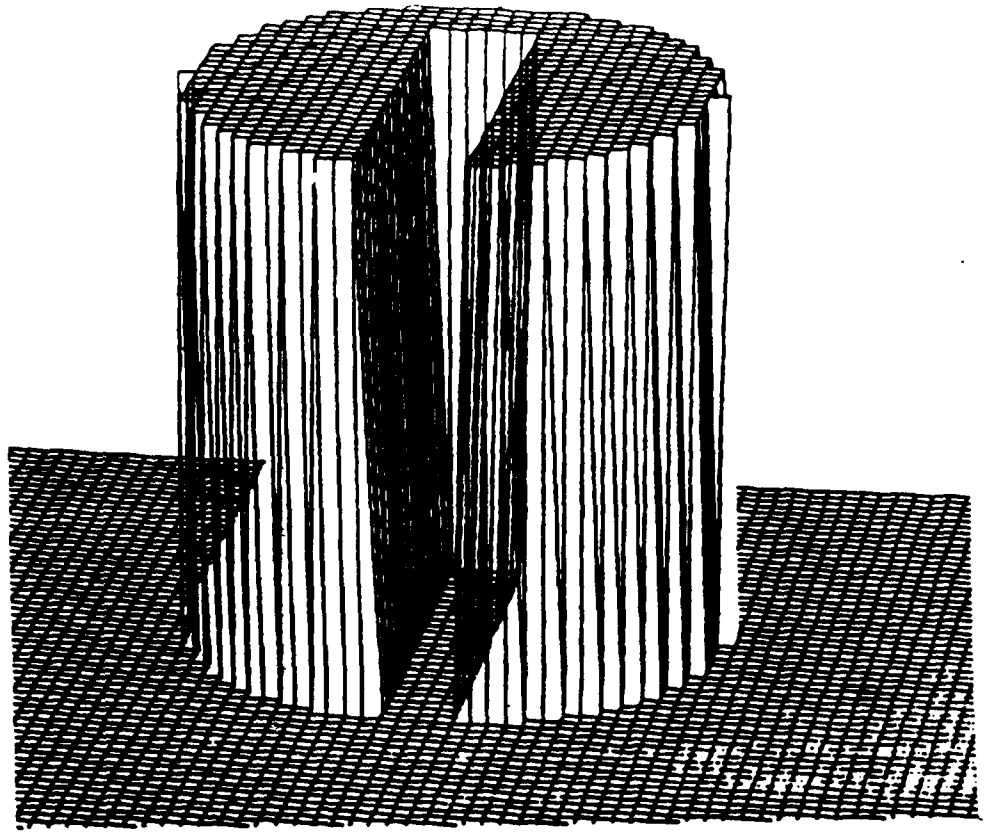
a) High-order scheme: Lumped-Mass Taylor Galerkin (niter=1)



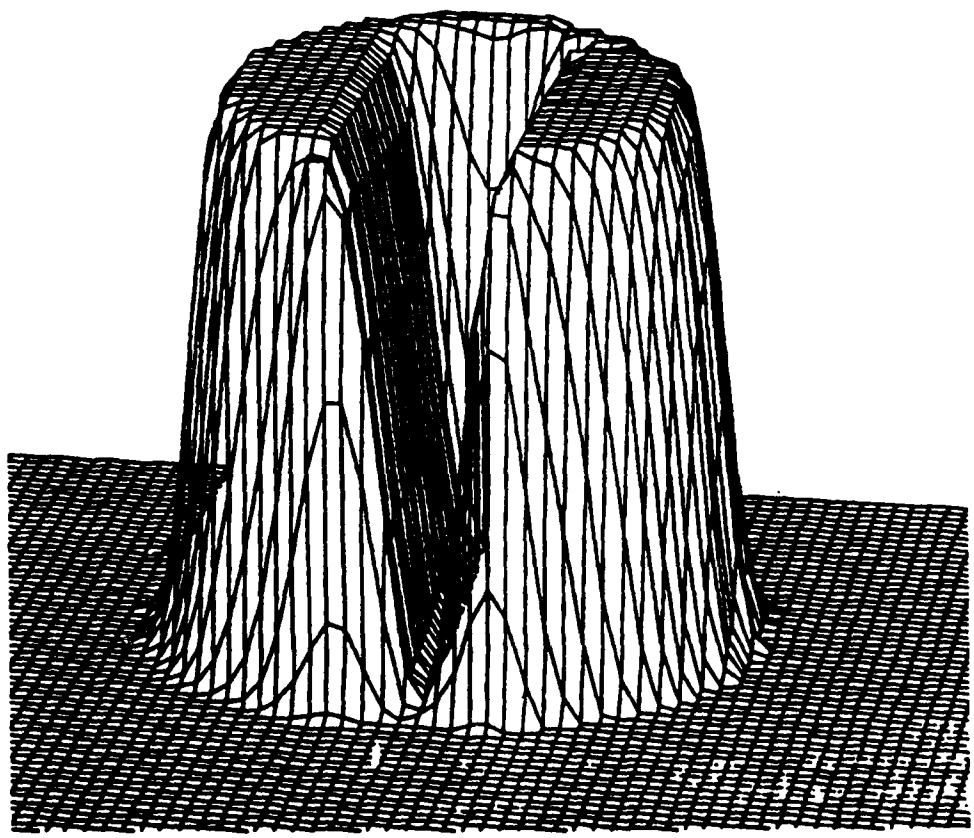
b) High-order scheme: Consistent-Mass Taylor Galerkin (niter=3)

Figure 3: Shock propagation for Burgers equation (1-D)

h=0.1, Δt=0.2, plot every 51 steps



a) Solution at time t=0.0



b) Solution after 628 iterations ($\frac{1}{4}$ revolution).

The perspective view has been rotated with the cylinder.

Figure 4: Passive Advection in 2-D: Zalesak's example [18]

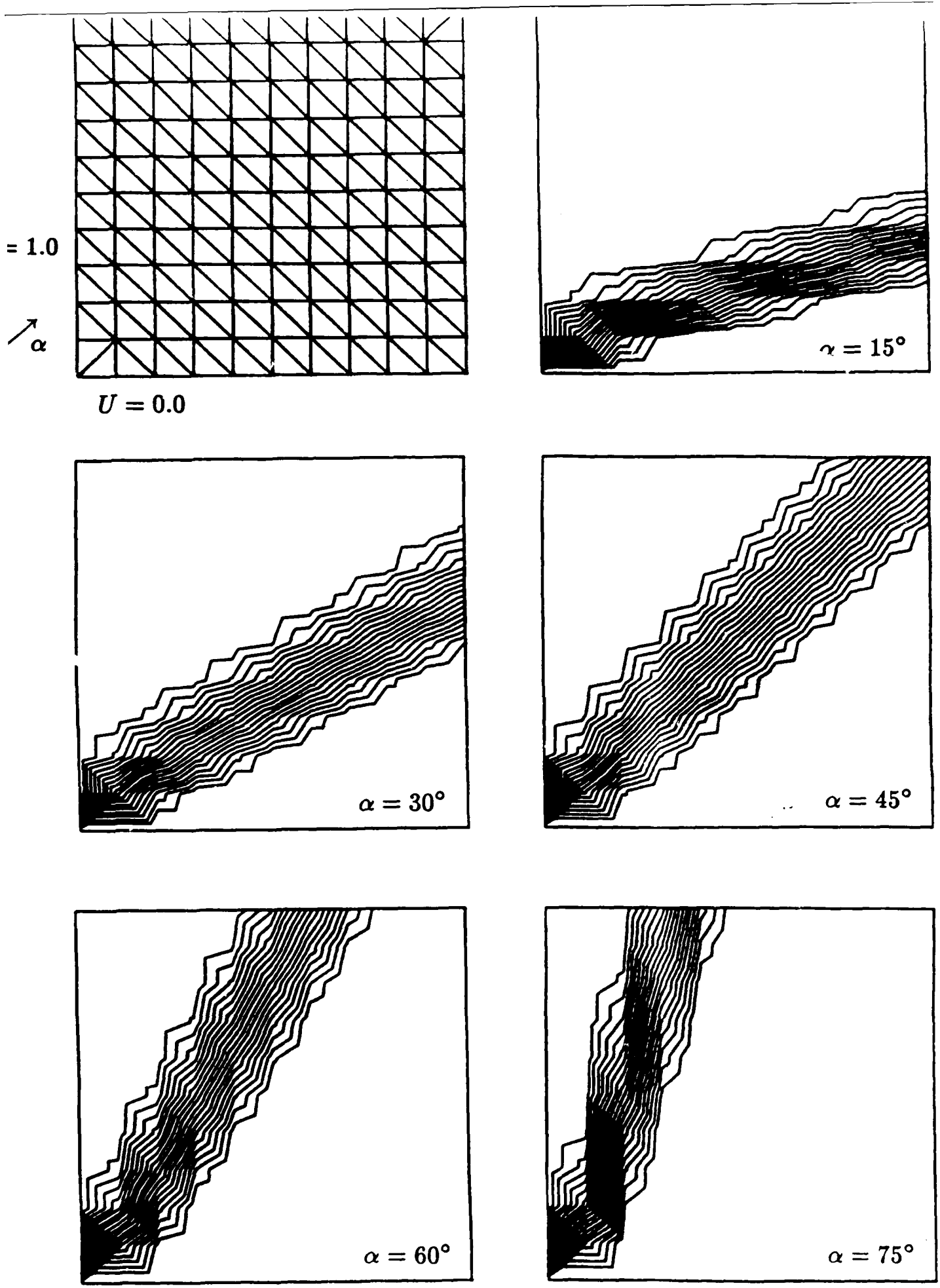
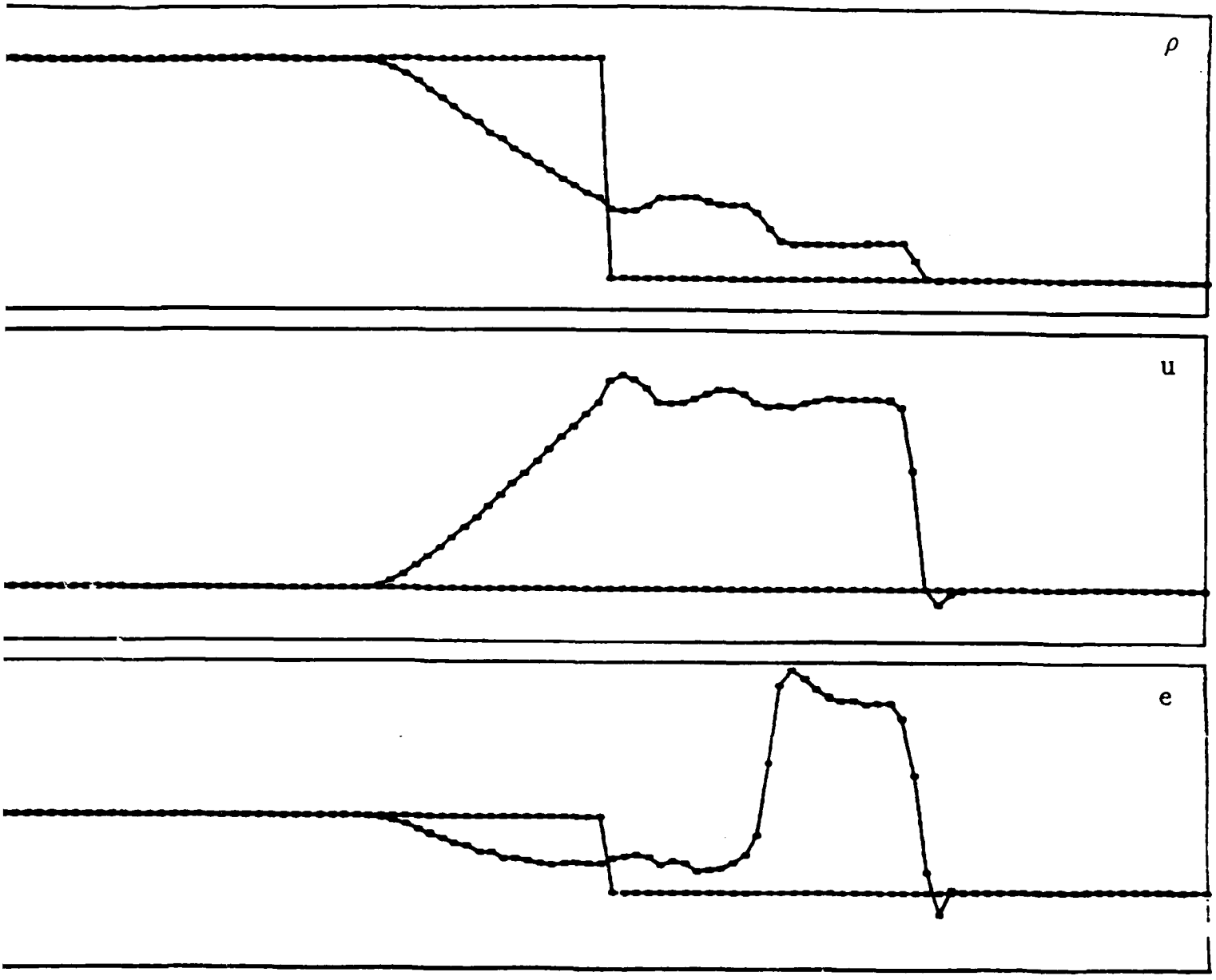
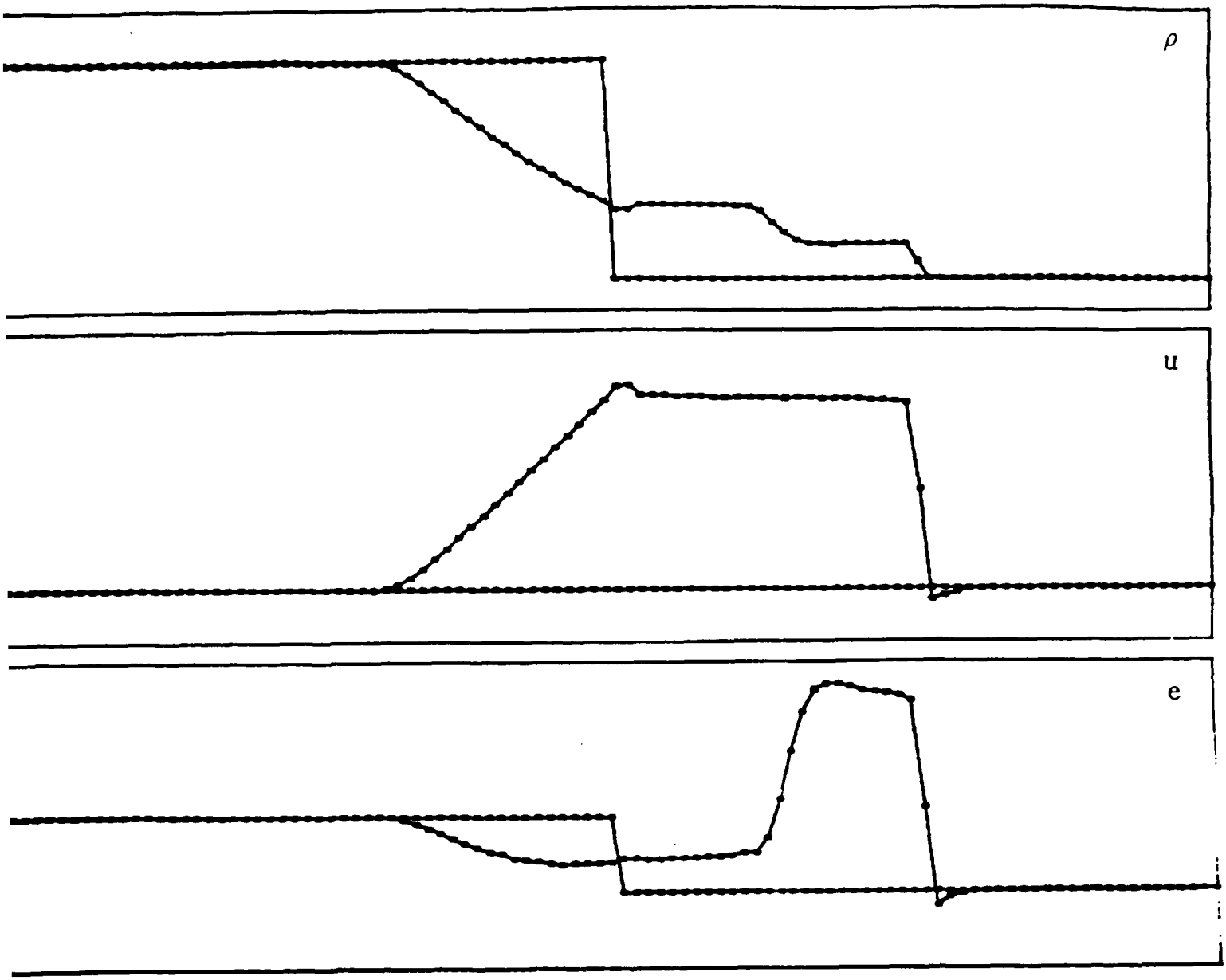


Figure 5: Advection skew to the mesh (steady state) (C.I.=0.05)



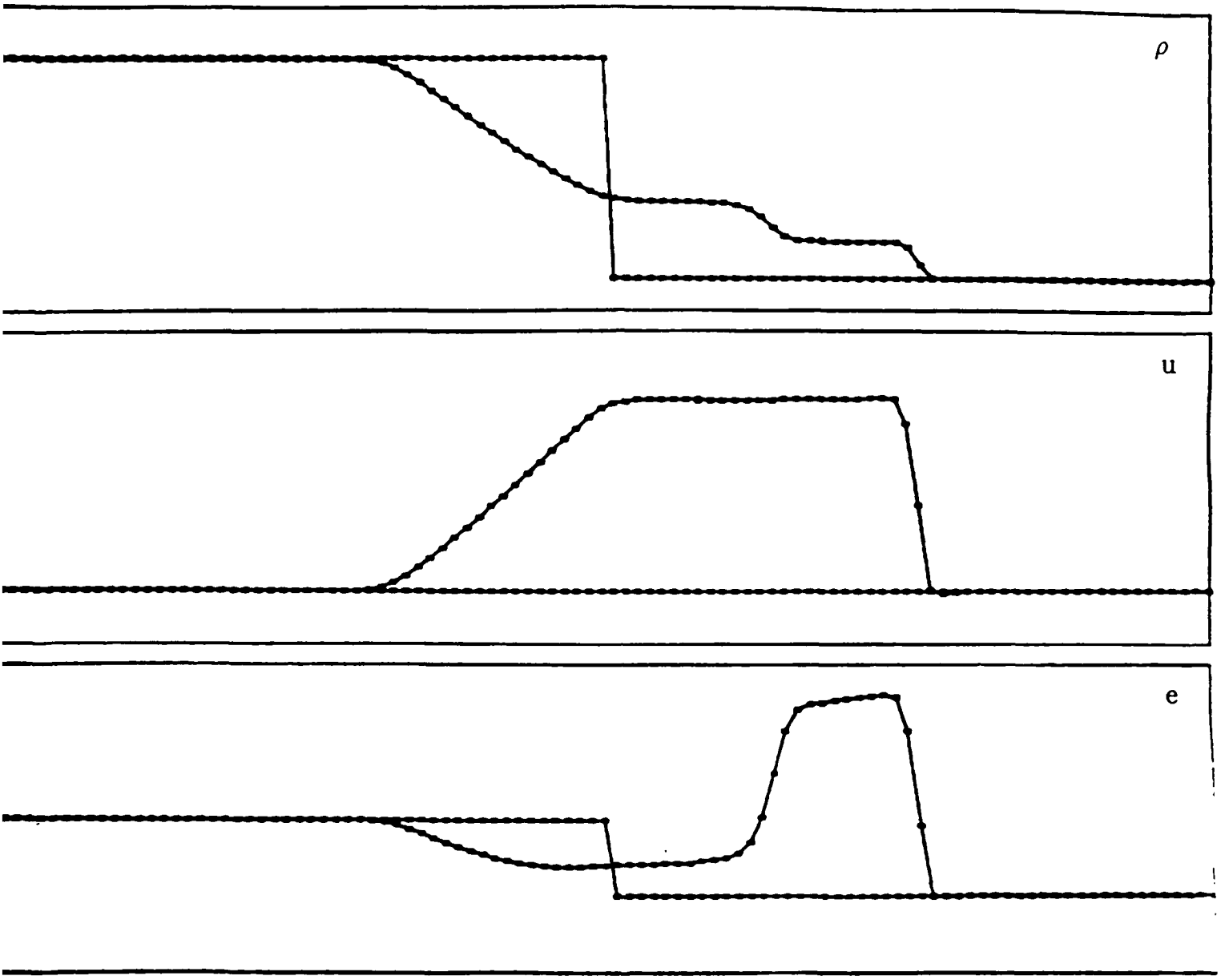
a) C_{el} independent for each equation

igure 6: Sod's Riemann-problem [29]



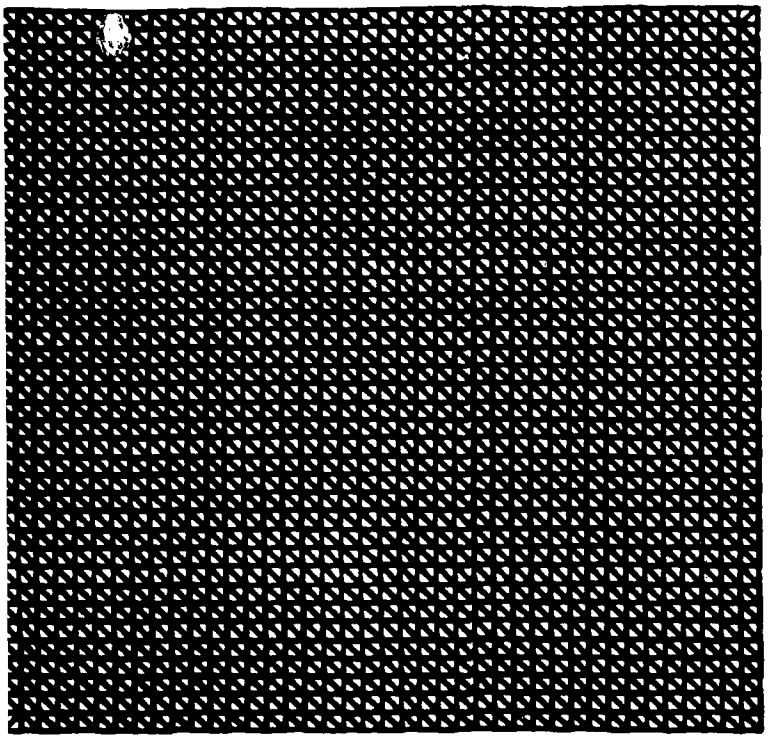
b) $C_{el} = C_{el}(density)$

'igure 6: Sod's Riemann-problem (cont.)

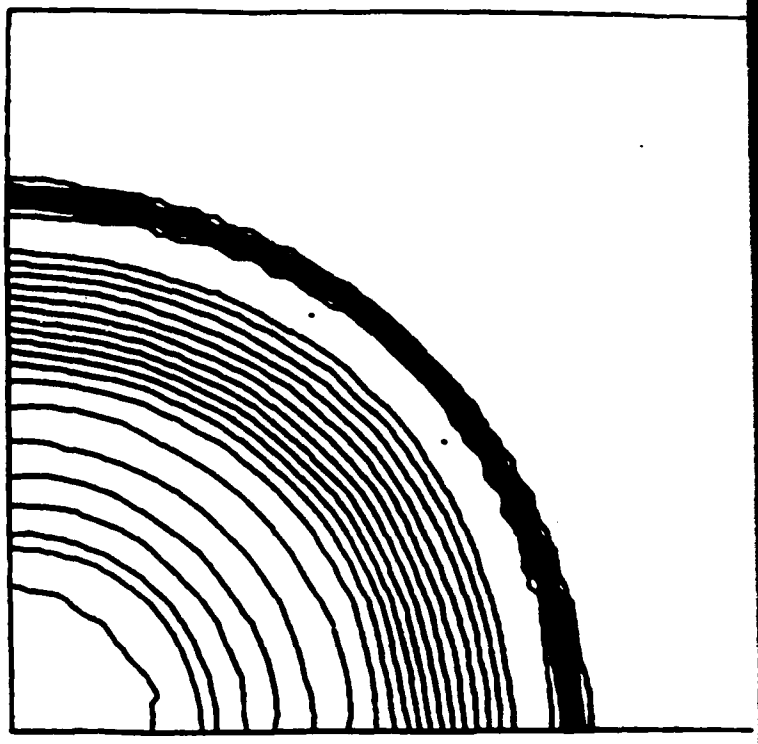


c) $C_{el} = min(C_{el}(density), C_{el}(energy))$

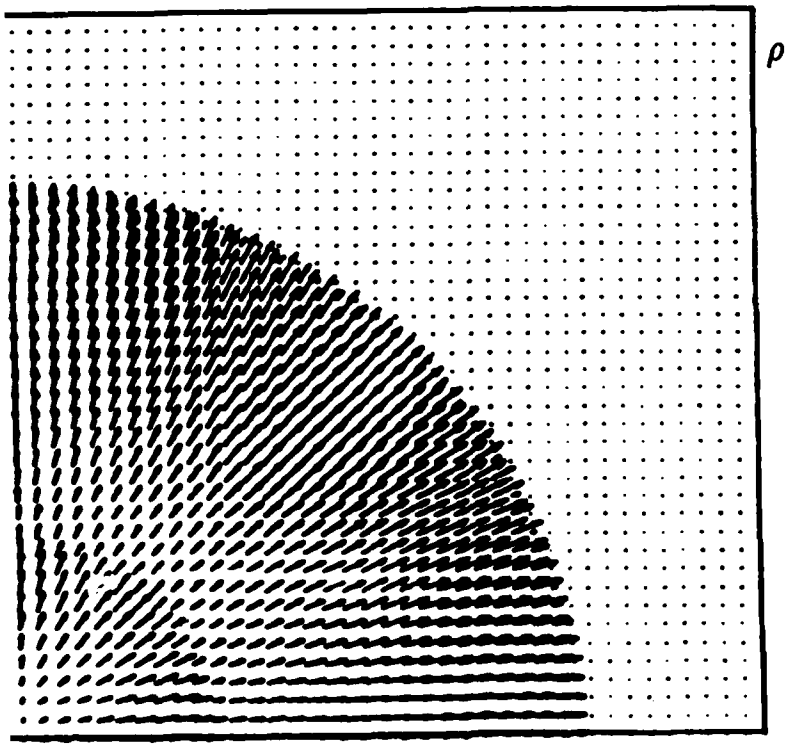
Figure 6: Sod's Riemann-problem (cont.)



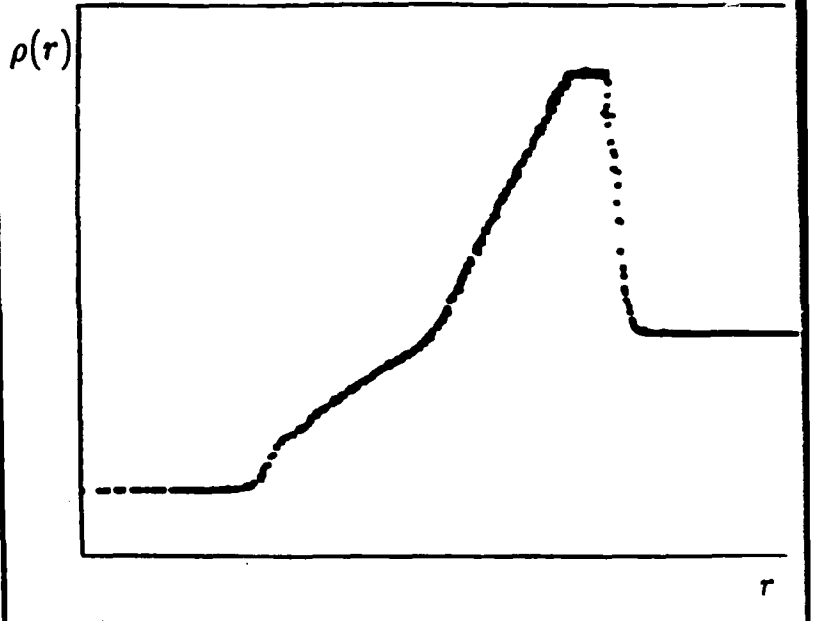
a) Grid



b) Density distribution at t=9.3 (C.I.=0.1)



:) Velocity distribution at t=9.3

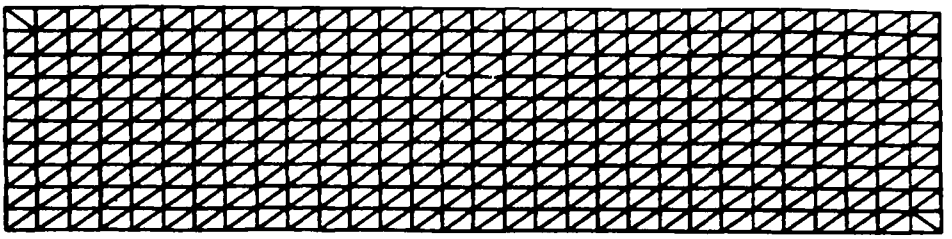


d) Density distribution as a function fron origin (t=9.3)

Figure 7: Cylindrical blast wave

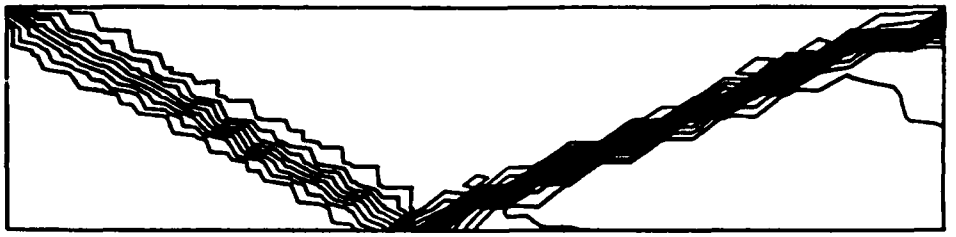
$\rho = 1.7, u = 2.6185, v = -0.5063, e = 5.806$

$\rho = 1.0$
$u = 2.9$
$v = 0.0$
$e = 5.991$

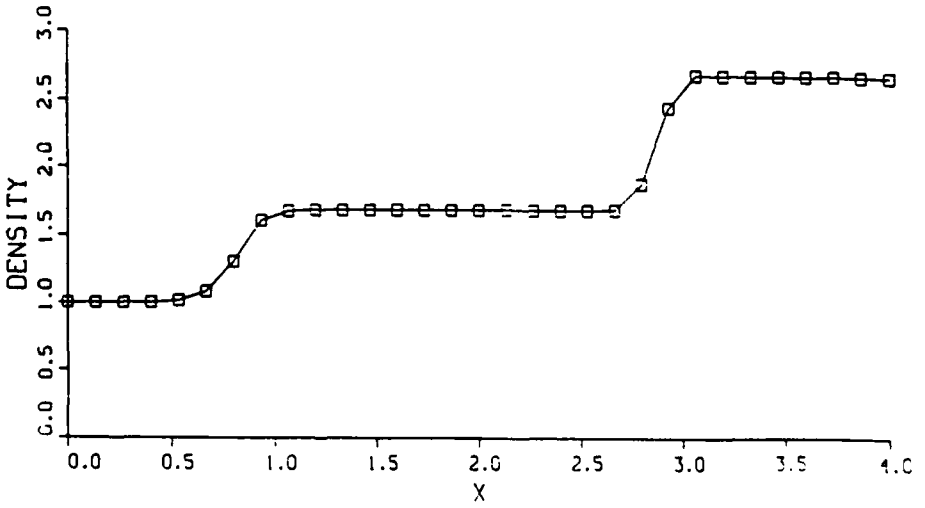


a) Grid

b) Pressure distribution obtained for Taylor-Galerkin scheme [10,12] (C.I.=0.1)

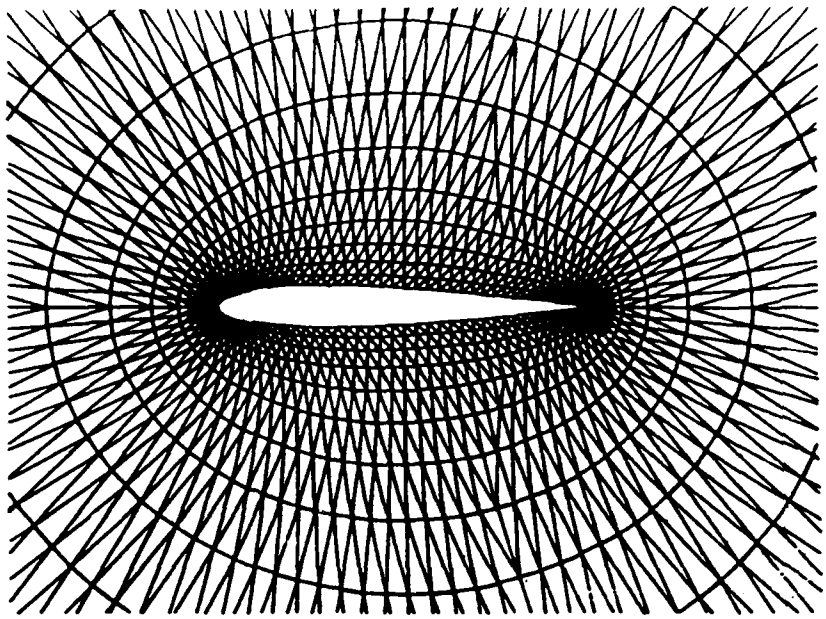


c) Pressure distribution obtained for FEM-FCT (C.I.=0.1)

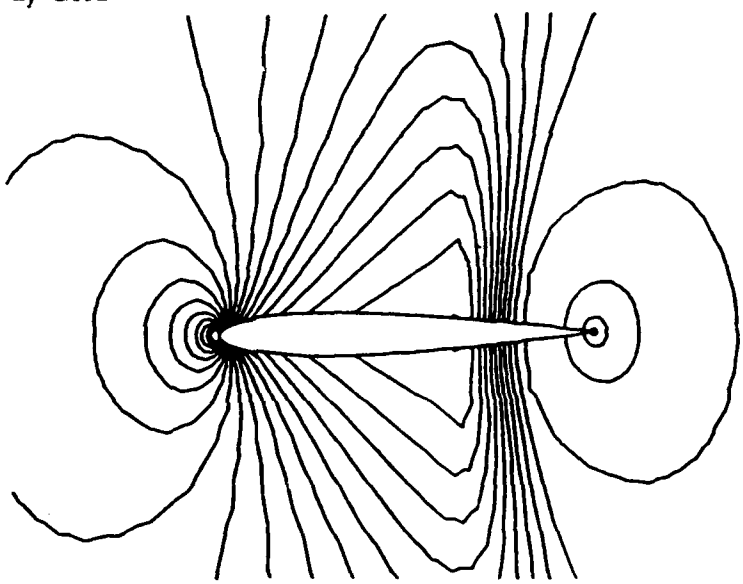


d) Density distribution along line y=0.5

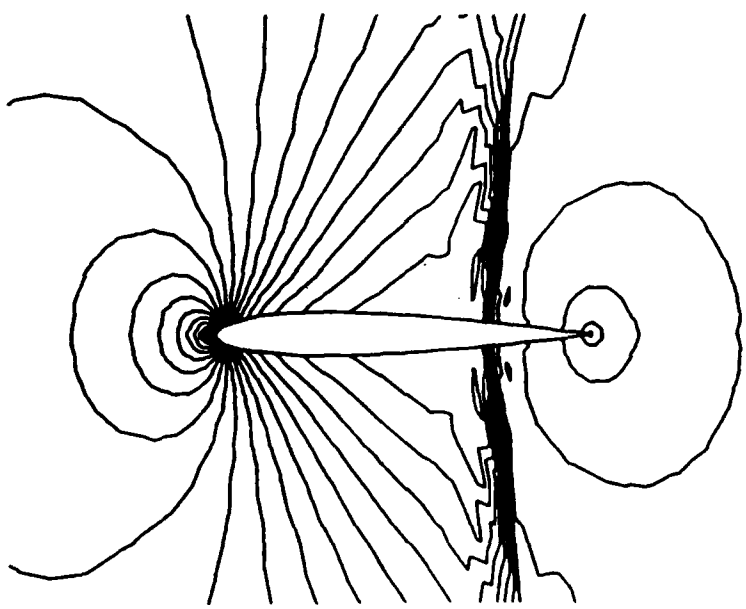
Figure 8: Shock reflexion at a wall (steady state)



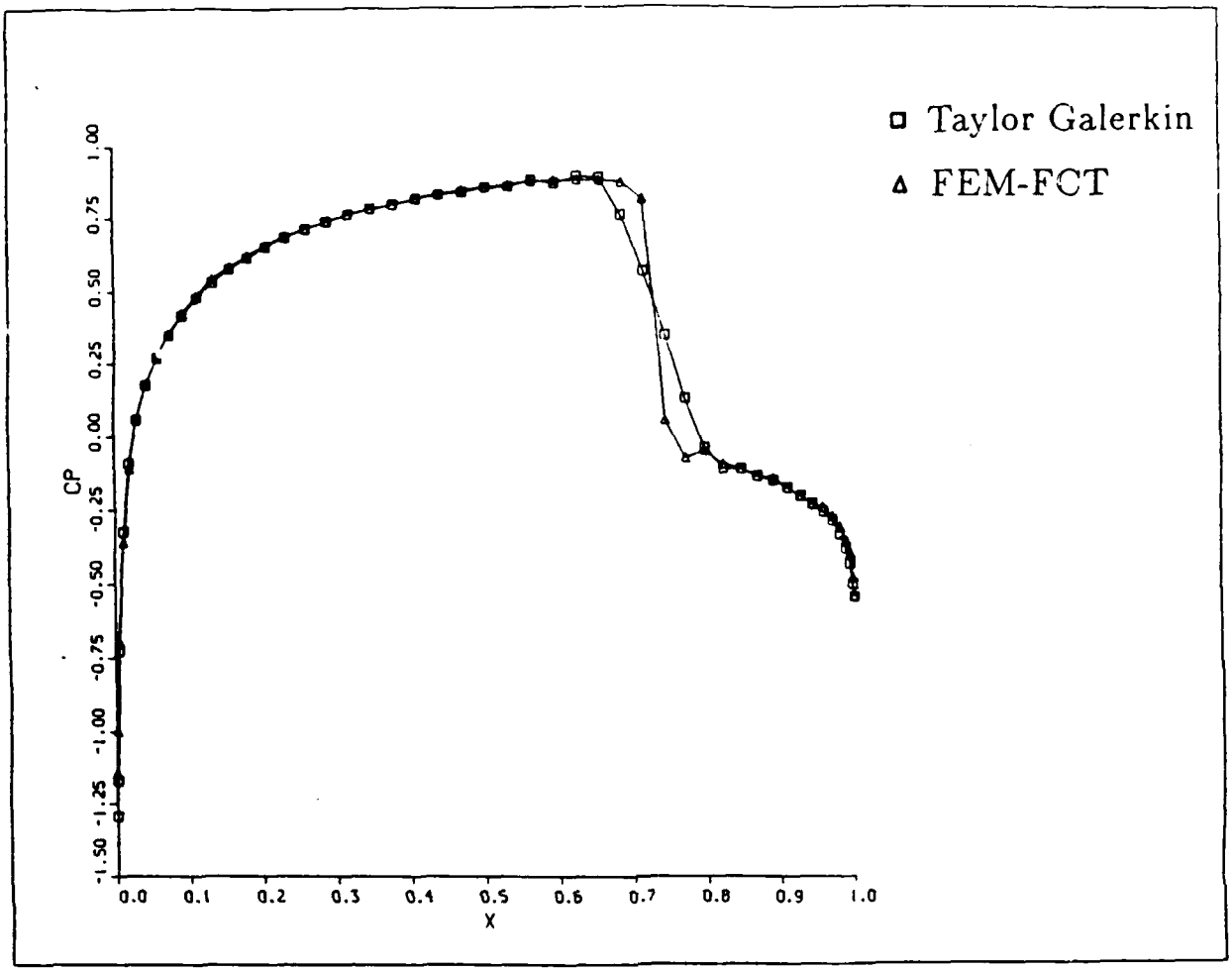
a) Grid



b) Pressure distribution obtained for Taylor-Galerkin scheme [10,12] (C.I.=0.05)



c) Pressure distribution obtained for FEM-FCT (C.I.=0.05)



d) Comparison of Cp-distribution on the surface

Figure 9: Transonic flow past an aerfoil (steady state)

$M_\infty = 0.85$, $\alpha = 0.0$

List of Symbols

∂	partial
ρ	rho
Δ	capital delta

APPENDIX X.

FEM-FCT and Adaptive Refinement Schemes for Strongly Unsteady Flows (Proc. ASME Winter Annual Meeting, Anaheim, CA, Dec. 1986)

Proc. ASME Winter Annual Meeting
Anaheim, California, December 1986

FEM-FCT AND ADAPTIVE REFINEMENT SCHEMES FOR STRONGLY UNSTEADY FLOWS

R. Löhner

Berkeley Research Associates
Springfield, VA 22150, USA
and
Laboratory for Computational Physics
Naval Research Laboratory
Washington, D.C. 20375, USA

ABSTRACT

We describe the combination of two effective computational techniques for the simulation of strongly unsteady compressible flows. These are a) a high resolution scheme for unstructured grids (FEM-FCT), and b) an adaptive refinement method for transient problems. The high resolution scheme is based on Zalesak's [11] generalization of the FCT-algorithms due to Boris and Book [12-14], and is able to capture moving and stationary shocks over two elements. For the adaptive refinement method the classic h-enrichment/coarsening is employed in conjunction with a spatial finite element discretization (triangles or tetrahedra). A modified interpolation theory error estimator is derived and shown to be effective for the problems at hand. Several numerical examples demonstrate that the schemes give results of excellent quality at minimal cost.

INTRODUCTION

The aim of the present research effort is to develop efficient computational techniques for the simulation of strongly unsteady flows past bodies of complex geometrical shape. Typical examples of physical problems falling under this category are shock-shock, shock-surface, and shock-structure interactions [1-3]. By incorporating chemistry models into the partial differential equations (PDEs) describing the flows under consideration, one would also be able to simulate detonation and flame propagation [4,5].

The problem statement resulted in three computational design decisions:

1) As the geometry of the bodies under consideration is complex unstructured grids in conjunction with Finite Element Methods were pursued from the start.

2) As strong shocks are present in the flowfields simulated, a high resolution, monotonicity-preserving algorithm for unstructured grids had to be developed. The method, called FEM-FCT, is based on Zalesak's [11] generalization of the Flux-Corrected Transport (FCT) algorithms of Boris and Book [12-14] to multidimensional problems.

3) As the flowfield is largely smooth, with a few regions where strong gradients appear, efficient adaptive refinement techniques for transient problems are required.

THE FLOW SOLVER: FEM-FCT

As stated above, high resolution, monotonicity preserving schemes must be developed in order to be able to simulate the strong nonlinear discontinuities present in the flows under consideration. Although the pertinent literature abounds with high resolution schemes [6-10], only Zalesak's generalization [11] of the 1-D FCT schemes of Boris and Book [12-14] can be considered a truly mutidimensional high resolution scheme. We remark here that the use of unstructured grids requires such truly multidimensional schemes, as the lack of lines or planes in the mesh inhibits the use of operator splitting.

Parrot and Christie [15] first analized FCT schemes in the context of Finite Element Methods, and Löhner et.al. [16,17] extended these ideas further to include the consistent mass which yields high temporal accuracy and to systems of equations.

FCT Defined

We consider a set of conservation laws given by a system of partial differential equations of the form

$$\frac{\partial U}{\partial t} + \frac{\partial F_a^i}{\partial x^i} = \frac{\partial F_v^i}{\partial x^i}, \tag{1}$$

where the advective fluxes $F_a = F_a(U)$ play a dominant role over the viscous fluxes $F_v = F_v(U)$. For flows described by eqn.(1), discontinuities in the variables may arise (e.g. shocks or contact discontinuities). Any numerical scheme of order higher than one will produce overshoots or ripples at such discontinuities (so-called 'Godunov theorem'). Very often, particularly for mildly nonlinear systems, these overshoots can be tolerated. However, for the class of problems studied here, overshoots will eventually lead to numerical instability, and will therefore have to be suppressed.

The idea behind FCT is to combine a high-order scheme with a low-order scheme in such a way that in regions where the variables under consideration vary smoothly (so that a Taylor expansion makes sense) the high-order scheme is employed, whereas in those regions where the variables vary abruptly the low-order scheme is favored.

The temporal discretization of eqn.(1) yields

$$U^{n+1} = U^n + \Delta U, \tag{2}$$

where ΔU is the increment of the unknowns obtained for a given scheme at time $t = t^n$. Our aim is to obtain a ΔU of as high an order as possible without introducing overshoots. To this end, we re-write eqn.(2) as:

$$U^{n+1} = U^n + \Delta U^l + (\Delta U^h - \Delta U^l), \tag{3}$$

or

$$U^{n+1} = U^l + (\Delta U^h - \Delta U^l). \tag{4}$$

Here ΔU^h and ΔU^l denote the increments obtained by some high- or low-order scheme, whereas U^l is the (ripple-free) solution at time $t = t^{n+1}$ of the low-order scheme. The idea behind FCT is to limit the second term on the right-hand side of eqn.(4):

$$U^{n+1} = U^l + lim(\Delta U^h - \Delta U^l), \tag{5}$$

in such a way that no new over/undershoots are created.

It is at this point that a further constraint, given by the conservation law (1) itself must be taken into account: strict conservation on the discrete level should be maintained. The simplest way to guarantee this for node-centered schemes (and we will only consider those here) is by constructing schemes for which the sum of the contributions of each individual element (cell) to its surrounding nodes vanishes ('all that comes in goes out'). This means that the limiting process (eqn.(5)) will have to be carried out in the elements (cells).

We can now define FCT in a quantitative way. We follow Zalesak's exposition [11], but modify the term 'flux' by 'element contribution to a node'. Those more familiar with Finite Volume or Finite Difference schemes should replace 'element' by 'cell' in what follows.

FCT consists of the following six algorithmic steps:

1) Compute LEC: the 'low-order element contribution' from some low-order scheme guaranteed to give monotonic results for the problem at hand;

2) Compute HEC: the 'high-order element contribution', given by some high-order scheme;

3) Define AEC: the 'antidiffusive element contributions' :

$$AEC = HEC - LEC$$

4) Compute, for each node I, the updated low-order solution :

$$U_I^l = U_I^n + \sum_{el} LEC = U_I^n + \Delta U_I^l, \quad I = 1, ..., NPOIN, \tag{6}$$

5) Limit or 'correct' the AEC so that U^{n+1} as computed in step 6 is free of extrema not also found in U^l or U^n :

$$AEC^c = Cel * AEC, \quad 0 \leq Cel \leq 1; \tag{7}$$

6) Apply the limited AEC :

$$U_I^{n+1} = U_I^l + \sum_{el} AEC^c. \tag{8}$$

The Limiting Procedure

Obviously, the whole approach depends critically on the all-important step 5 (see above). We define the following quantities:

a) $P_I^{\pm}$: the sum of all positive (negative) antidiffusive element contributions to node I

$$P_I^{\pm} = \sum_{el} \begin{Bmatrix} max \\ min \end{Bmatrix} (0, AEC_{el})$$

b) $Q_I^{\pm}$: the maximum (minimum) increment (decrement) node I is allowed to achieve in step 6 above

$$Q_I^{\pm} = U_I^{\overset{max}{min}} - U^l$$

where $U_I^{\overset{max}{min}}$ (defined below) represents the maximum (minimum) value the unknown U at node I is allowed to achieve in step 6 above.

c) $R^{\pm}$:

$$R^{\pm} := \begin{cases} min(1, Q^{\pm}/P^{\pm}) & IF \quad P^{+} > 0, \; P^{-} < 0 \\ 0 \quad IF \quad P^{\pm} = 0 \end{cases}.$$

Now take, for each element:

$$Cel = min(element\ nodes) \begin{cases} R^{+} \; if \; AEC > 0, \\ R^{-} \; if \; AEC < 0. \end{cases} \tag{9}$$

Finally, we obtain $U_I^{\overset{max}{min}}$ in three steps :

a) maximum (minimum) nodal U of U^n and U^l :

$$U_I^{*} = \begin{Bmatrix} max \\ min \end{Bmatrix} (U_I^l, U_I^n) \ ,$$

b) maximum (minimum) nodal value of element :

$$U^*_{el} = \begin{Bmatrix} max \\ min \end{Bmatrix} (U^*_A, U^*_B, ..., U^*_C) \ ,$$

where $A, B, ..., C$ represent the nodes of element el.

c) maximum (minimum) U of all elements surrounding node I :

$$U_I^{max} = \begin{Bmatrix} max \\ min \end{Bmatrix} (U^*_1, U^*_2, ..., U^*_m) \ .$$

where $1, 2, ..., m$ represent the elements surrounding node I.

This completes the description of the limiting procedure. Up to this point we have been completely general in our description, so that eqns.(1)-(9) may be applied to any FEM-FCT scheme. In what follows, we restrict the exposition to the FEM-schemes employed in the present work, describing the high and low-order schemes used.

The high-order scheme : Consistent-Mass Two-step Taylor Galerkin

As the high-order scheme, we employ a two-step form of the one-step Taylor-Galerkin schemes described in [18,19]. The method has been expounded thoroughly in [20,28]. We will only give a brief description of it here. Given the system of equations

$$\frac{\partial U}{\partial t} + \frac{\partial F^i}{\partial x^i} = 0, \tag{10}$$

we advance the solution from t^n to $t^{n+1} = t^n + \Delta t$ as follows :

a) First step :

$$U^{n+\frac{1}{2}} = U^n - \frac{\Delta t}{2} \cdot \frac{\partial F^i}{\partial x^i}|^n \tag{11}$$

b) Second step :

$$\Delta U^n = U^{n+1} - U^n = -\Delta t \cdot \frac{\partial F^i}{\partial x^i}|^{n+\frac{1}{2}}. \tag{12}$$

The spatial discretization of (11) and (12) is then performed via the classical Galerkin weighted residual method [20,28]. We remark that at $t^{n+\frac{1}{2}} = t^n + \frac{1}{2}\Delta t$ both U and F are interpolated using piecewise constant functions, while at t^n and t^{n+1} linear shape functions are employed. In this way the values of $U^{n+\frac{1}{2}}, F^{n+\frac{1}{2}}$ may be readily obtained at element centers (no matrix operations are involved). For (12) the following system of equations is obtained:

$$M_c \cdot \Delta U^n = R^n, \tag{13}$$

where M_c denotes the consistent mass matrix [19], ΔU the vector of nodal increments, and R the added element contributions to each node. As M_c possesses an excellent condition number, Eqn.(13) is never solved directly, but iteratively (typically, three passes are sufficient [19]). We finally recast the converged solution of Eqn.(13) into the following form, which will be of use later on :

$$M_l \cdot \Delta U^h = R + (M_l - M_c) \cdot \Delta U^h. \tag{14}$$

Here we have introduced the superscript 'h' for high-order scheme, and M_l denotes the diagonal, lumped mass-matrix (see [19]).

The low-order scheme: Lumped-Mass Taylor Galerkin plus Diffusion

The requirement placed on the low-order scheme in any FCT-method is monotonicity. The low-order scheme must not produce any artificial, c[illegible]numerical, 'ripples' or 'wiggles'. So far, we have simply added 'mass-diffusion' in the [illegible]ntext of FEM-FCT [16,17]. This simplest (and cheapest) form of diffusion is obtained by subtracting the lumped mass-matrix from the consistent mass-matrix for linear elements:

$$DIFF = c_d \cdot (M_c - M_l) \cdot U^n. \tag{15}$$

The element matrix thus obtained for 2-D triangles is of the form

$$(M_c - M_l)_{el} = -\frac{c_d \cdot A}{12} \begin{bmatrix} 2 & -1 & -1 \\ -1 & 2 & -1 \\ -1 & -1 & 2 \end{bmatrix}, \tag{16}$$

where A denotes the element area and c_d the diffusion coefficient employed. A similar expression can be derived for tetrahedra in 3-D.

Observe that we cannot simply add this diffusion to the high-order scheme in order to obtain monotonic results for the problems at hand, as a multipoint-coupling of the right-hand side occurs due to the consistent mass-matrix employed in the high-order scheme . The imposition of monotonicity can nevertheless be achieved by using a lumped mass-matrix instead. As the terms originating from the discretization of the fluxes F^i in (10) are the same as in (12), the low-order scheme can then be written as

$$M_l \cdot \Delta U^l = R + DIFF, \tag{17}$$

where we have introduced the superscript 'l' for low-order scheme.

Computation of the antidiffusive element contributions

Subtracting (17) from (14) yields the equation

$$M_l \cdot (\Delta U^h - \Delta U^l) = R + (M_l - M_c) \cdot \Delta U^h - R - DIFF, \tag{18}$$

or

$$M_l \cdot (\Delta U^h - \Delta U^l) = (M_l - M_c) \cdot \Delta U^h - DIFF \tag{19}$$

Note that all terms arising from the discretization of the fluxes F^i in (10),(12),(17) have now disappeared. This is of particular importance if the antidiffusive element contributions must be recomputed (small core memory machines), and real gas effects are taken into account (table look-up times are considerable) or real viscosity effects have to be included (Navier-Stokes equations).

Limiting for Systems of Equations

The results available in the literature [11-14] and our own experience [16] has shown that very good results can be obtained for a single PDE. However, when trying to extend the limiting process to systems of PDEs, no immediately obvious or natural limiting procedure becomes apparent. Obviously, for 1-D problems one could advect each simple wave system separately, and then assemble the new solution at the new time step. However, for multidimensional problems such a splitting is not possible, as the acoustic waves are circular. FDM-FCT-codes used for production runs [3,4] have so far limited each equation separately, invoking operator-splitting arguments. This approach does not always give very good results, as may be seen from Sod's comparison of schemes for the Riemann problem [21], and has been a point of continuing criticism by those who prefer to use the more costly Riemann-solver-based, essentially one-dimensional TVD-schemes [6-9]. It would therefore seem preferable to introduce 'system character' for the limiter by combining the limiters for all equations of the system. Many variations are possible and can be implemented, giving different performance for different problems. We just list some of the possibilities here, commenting on them where empirical experience is available.

a) Independent treatment of each equation as in operator-split FCT: this is the least diffusive method, tending to produce an excessive amount of ripples in the non-conserved quantities (and ultimately also in the conserved quantities).

b) Use of the same limiter (C_{el}) for all equations: this produces much better results, seemingly because the phase errors for all equations are 'synchronized'. This was also observed by Harten and Zwaas [22] for a class of schemes very similar to FCT.

c) Use of a certain variable as 'indicator variable' (e.g. density, pressure, entropy).

d) Use as the limiter for all equations the minimum of the limiters obtained for the density and the energy ($C_{el} = min(C_{el}(density), C_{el}(energy))$) : this produces acceptable results, although some undershoots for very strong shocks are present.

For the results presented here, this last option was employed.

TRANSIENT ADAPTIVE REFINEMENT

The advantages of adaptive refinement schemes for the efficient simulation of steady state flow fields have by now been demonstrated by several authors [23-31].

Storage and CPU requirements will typically be reduced by more than an order of magnitude (as compared to uniform refinement) for most of the problems falling into this class (steady state, advection dominated) when adaptive refinement schemes are employed.

Recently, several authors have studied adaptive schemes for transient problems [32-35]. Several additional restrictions have to be placed on the adaptive schemes employed here in order that they achieve similar savings in CPU and storage requirements:

- the refinement scheme/logic has to be fast, as the grid is modified many times during the computation;

- the refinement scheme should not be memory (storage) intensive, as it becomes an integral part of the code;

- the original mesh should be recovered after the feature has passed, in case that the feature returns (e.g. a shock reflection).

This in turn implies that:

- No directional refinement [30,31] can be used, as these schemes appear as too storage and CPU-intensive.

- Classic h-enrichment/coarsening must be employed, as it does not require a major storage overhead and due to its simplicity lends itself easily to vectorization.

- Only one level of refinement/coarsening is employed per 'mesh change' in order to minimize the logic involved and thus CPU requirements.

- For triangles, successive subdivision of a triangle into two (see Figure 1) has to be avoided. This in turn reduces the number of refinement cases considerably.

A Modified Interpolation Theory Error Indicator

Many possible error indicators have been suggested in the literature (see [23-35] and many other publications), and numerical experience indicates that all perform similarly well. However, in the present context, the following requirements must be met:

- The error indicator must be fast.

- As the feature may move only very slowly or come to a standstill (e.g. a shock entering a very dense region), the error indicator must also be reliable for steady state applications .

- As systems of equations are solved, and more than one 'key-variable' [27] may be employed, the error indicator should be dimensionless.

- In order to be applicable to a large class of problems, the error estimator should be bounded (independently of the solution), so that preset refinement/coarsening tolerances can be employed.

In order to meet these requirements, a modified form of the classic interpolation estimates [23,24] used for steady state computations [28-31] has been adopted. These estimators make use of an appropriate seminorm for the detection of those regions which need further refinement or coarsening, e.g. the H2-seminorm [23,24,28-31,34]

$$\|u - u^h\|_0 \leq c \cdot h^2 \cdot |u|_2, \tag{20}$$

where u denotes the exact and u^h the approximate solution, c is a mesh-size independent constant, h is the characteristic mesh size, and

$$|u|_2 = \sqrt{\int_\Omega \sum_{i,j} \left[\frac{\partial^2 u}{\partial x^i \partial x^j}\right]^2 d\Omega} \tag{21}$$

Second derivatives are justified here because the shape functions used in the finite element discretization are linear. Numerically, we first evaluate the second derivatives at the nodes via a variational statement [29], and then approximate the integral (21) 'conservatively' by taking for each element the maximum second derivative at the associated nodes. For linear elements of constant length h in 1-D, one obtains for the first step at the nodes :

$$e_I = h^{-2} \cdot |U_{I+1} - 2 \cdot U_I + U_{I-1}|. \tag{22}$$

The modified error indicator is given by:

$$E_I = \frac{|U_{I+1} - 2 \cdot U_I + U_{I-1}|}{|U_{I+1} - U_I| + |U_I - U_{I-1}| + \epsilon\,[|U_{I+1}| + 2 \cdot |U_I| + |U_{I-1}|]} \tag{23}$$

We remark the following properties of this modified error indicator:

- By dividing the second derivative by the 'jumps' (gradients), the 'eating-up' effect in the presence of a very strong shock is avoided (i.e. only the value of the normalized H2-seminorm is of importance, not the magnitude of the H2-seminorm as such).

- Normalizing in this way also has the advantage that the error indicator becomes dimensionless, so that more than one 'key-variable' can be used without encountering dimensioning problems.

- Moreover, the modified error indicator is now bounded ($0 \leq E_I < 1$), so that preset tolerances can be employed (this is of particular importance for transient problems).

- The terms following ϵ are added as a 'noise' filter in order not to refine 'wiggles' or 'ripples' which may appear due to loss of monotonicity. The value for ϵ thus depends on the algorithm chosen to solve the PDEs describing the physical process at hand.

The generalization of this error estimator to multidimensional situations is as follows:

$$E^I = \sqrt{\frac{\sum_{k,l} (\int_\Omega N^I_{,k} N^J_{,l} d\Omega \cdot U_J)^2}{\sum_{k,l} (\int_\Omega |N^I_{,k}| \left[|N^J_{,l} U_J| + \epsilon \left(|N^J_{,l}| |U_J|\right)\right] d\Omega)^2}}, \tag{24}$$

where N^I denotes the shape-function of node I.

After having determined the values of the error indicators in the elements, all elements lying above a preset threshold value CTORE are refined, while all elements lying below a preset threshold value CTODE are coarsened.

Grid Logic

As described above, we limit the number of refinement/coarsening levels per mesh change to one. Moreover, we allow only refinement of a triangle into two or four and avoid the successive refinement of a triangle into two. This implies that there exist only six possible cases for refinement and three for coarsening. These cases are shown in Figures 1,2.

In order to identify the 'parent' and 'son' elements of any element, six integer locations per element were employed in the present situation. The first three integers store the new three neighbor elements ('sons') of an element that has been subdivided into four (the center element of the four is kept as 'parent'). In the fourth integer the element from which the present element originated (the 'parent' element) is stored, while the fifth integer denotes the side of the 'parent' element this element came from. Finally, in the sixth integer location the refinement level is remembered. These six integer locations per element are sufficient to construct further refinements or to reconstruct the original grid. Observe that no classical tree-structure is employed.

The introduction of further nodes (refinement) is performed by first identifying the sides that require refinement, and then labelling these sides with the new node numbers. By doing this, the introduction of coordinates, values for the unknowns and boundary conditions at the new nodes can be performed independently of the introduction of new elements. In principle, these operations could be performed in parallel. For more details on the grid logic, see [35].

EXPLOITATION OF SUPERCOMPUTING HARDWARE

A key question which has to be accounted for when solving large-scale transient problems on today's supercomputers is whether the algorithms lend themselves to vectorization. Although this particular issue seems meaningless to many theoreticians, practical experience indicates that the difference in computing times between a fully vectorized code and one that does not lend itself to vectorization is more than a factor of 10 (1000% !). This figure is likely to increase even further when massively parallel computers become available. Therefore the inherent degree of parallelism of any new algorithm may decide more and more its competitiveness against other methods.

Both FEM-FCT and the described mesh refinement/coarsening process can be vectorized to a large extent, particularly if the machine available vectorizes GATHER/SCATTER loops efficiently. The only non-trivial operation that has to be re-coded with some care is the so-called SCATTER-ADD process which occurs when assembling a 'right-hand side vector'. By reordering the element numbering, this

semi-recursive operation can be rearranged into a set of SCATTER-loops, thus again operating in vector mode.

On the CRAY-XMP-12 at NRL, for typical runs (a mesh change every 5 steps), the CPU-time spent in non-vectorizable loops is of the order of 1%, indicating that more than 99% of all operations are performed in vector mode. The processing rate for the consistent mass FEM-FCT algorithm is of $58\mu sec$ per grid point per time step in 2-D, while the grid adaptation alone requires about $100\mu sec$ per grid point per mesh change. Although this last figure seems high, one has to bear in mind that the adaptation is performed only every 5-10 timesteps, which means that effectively it is much lower. From previous experience [36], one can expect these processing rates to be similar on a one-pipe CYBER-205.

NUMERICAL EXAMPLES

All numerical examples shown involve shock-structure and shock-shock interactions. It was found that for this class of problems and the algorithm employed the following choice of refinement/coarsening parameters produced acceptable results:

- refinement tolerance: CTORE=0.3
- coarsening tolerance: CTODE=0.1
- noise parameter : ϵ =0.2
- key variable for error indicator: density

We also tested other key-variables as error indicators, but it was found that for the class of problems under consideration, i.e. compressible Euler equations, the density gave the best results (pressure is not a good error indicator at contact discontinuities).

Unless otherwise stated this set of parameters was employed for all the examples shown below.

Mach Reflection on a Double Wedge

The problem statement, as well as the solutions obtained, are shown in Figures 3a-3d. The aim of this simulation was to reproduce the complex flow structures observed in experiments [37] (see Figure 3e). Initially, a weak shock ($M_s = 1.29$) travels towards the double wedge from the right (the angles were chosen so as to agree with those in case 5 of the shock tube experiments of Ben-Door and Dewey [37]). The shock then reflects off the the first wedge, forming a Mach stem and a contact discontinuity. Although the contact discontinuity is weak (see contours of density), the error indicator nevertheless senses it, accordingly increasing the grid resolution in this region. The shock then reflects off the second wedge, forming the complex flow features seen in Figures 3c,3d. Observe that the grid has been well adapted to the physical complexity, and that the simulation reproduces essentially all the gas-dynamic discontinuities seen with optical diagnostics in the experiment (Figure 3e). As the first reflected shock becomes weaker and weaker, in some parts the error indicator lies below the coarsening tolerance, therefore producing the 'ragged' grid seen in Figures 3c and 3d. The

discrepancy between experiment (Figures 3e) and computation (Figures 3a-3d) stems mainly from the fact that the length of the first wedge was larger in the experiment than in the computation. As we employed the ideal gas equation of state for this run, this could yield another possible explanation for the discrepancies.

Shock over an Indentation

The problem statement, as well as the solutions obtained at different timesteps during the numerical simulation are shown in Figures 4a-4e. A strong shock ($M_s = 25$) travels over an indentation, producing a bow shock and a rarefaction (Figures 4a,4b). It then collides with the right wall of the indentation and bounces back, producing several shock/shock and shock/contact discontinuity interactions (Figures 4c,4d). Observe the level of physically relevant detail that the scheme is able to reproduce, e.g. the triple shock produced at T=0.12 (see Figures 4d and 4e). The velocity pattern generated by these interactions has been magnified in Figure 4e, and shows a large residual vortex, as well as the different shock fronts and other discontinuities.

For the numerical examples shown, the savings in CPU and storage requirements due to adaptive refinement were of the order of 10, a figure usually quoted with the technological leap of one generation in supercomputing technology.

CONCLUSIONS

The combination of FEM-FCT and adaptive refinement by classic h-enrichment/coarsening has been presented. It was found, that the combination of these two techniques yields results of excellent quality at minimal cost for strongly unsteady compressible flows. Shocks are captured over two elements, and the savings in both CPU and storage requirements due to adaptive refinement are of the order of 10 (as compared to uniform refinement) for typical problems.

Further extensions of the present work include

- better limiters for systems of equations in the context of FEM-FCT;
- inclusion of chemistry and real viscosity into the current inviscid, nonreacting flow models;
- extension of the present techniques to 3-D problems;
- incorporation of directional refinement techniques for transient problems;
- combination of different element types (quads/triangles) in a single code.

Some of these extensions are already being implemented and examined by the author and several other groups worldwide.

ACKNOWLEDGEMENTS

It is a pleasure to acknowledge many fruitful and stimulating discussions during the course of the present work with Drs. K. Morgan, J.P. Boris, D.L. Book and S.T Zalesak.

REFERENCES

[1] P. Woodward and P. Colella - The Numerical Simulation of Two-Dimensional Fluid Flow with Strong Shocks; *J.Comp.Phys.* 54, 115-173 (1984).

[2] H.M. Glaz, P. Colella, I.I. Glass and R.L. Deschambault - A Numerical Study of Oblique Shock-Wave Reflections with Experimental Comparisons; *Proc.Roy.Soc.Lond.* A 398, 117-140 (1985).

[3] M. Fry, J. Tittsworth, A. Kuhl, D.L. Book, J.P. Boris and M. Picone- Transport Algorithms with Adaptive Gridding; *Proc. 13th Int.Symp. on Shock Tubes and Waves* (C.E. Treanor and J.G. Hall eds.), 376-384 (1982).

[4] K. Kailasanath, E.S. Oran, J.P. Boris and T.R. Young - Determination of Detonation Cell Size and the Role of Transverse Waves in Two-Dimensional Detonations; *Combustion and Flame* 61, 199-209 (1985).

[5] N. Peters and J. Warnatz (eds.) - *Numerical Methods in Laminar Flame Propagation*; Vieweg Notes on Numerical Fluid Mechanics Vol. 6, Vieweg (1982).

[6] B. van Leer - Towards the Ultimate Conservative Scheme. II. Monotonicity and Conservation Combined in a Second order Scheme; *J.Comp.Phys.* 14, 361-370 (1974).

[7] P.L. Roe - Approximate Riemann Solvers, Parameter Vectors and Difference Schemes; *J.Comp.Phys.* 43, 357-372 (1981).

[8] S. Osher and F. Solomon - Upwind Difference Schemes for Hyperbolic Systems of Conservation Laws; *Math.Comp.* 38, 339-374 (1982).

[9] A. Harten - High Resolution Schemes for Hyperbolic Conservation Laws; *J.Comp.Phys.* 49, 357-393 (1983).

[10] P.K. Sweby - High Resolution Schemes Using Flux Limiters for Hyperbolic Conservation Laws; *SIAM J.Num.Anal.* 21, 995-1011 (1984).

[11] S.T. Zalesak - Fully Multidimensional Flux-Corrected Transport Algorithm for Fluids; *J.Comp.Phys.* 31, 335-362 (1979).

[12] J.P. Boris and D.L. Book - Flux-corrected Transport. I. SHASTA, a Transport Algorithm that works; *J.Comp.Phys.* 11, 38 (1973).

[13] D.L. Book, J.P. Boris and K. Hain - Flux-corrected Transport. II. Generalizations of the Method; *J.Comp.Phys.* 18, 248 (1975).

[14] J.P. Boris and D.L. Book - Flux-corrected Transport. III. Minimal-Error FCT Algorithms; *J.Comp.Phys.* 20, 397-431 (1976).

[15] A.K. Parrott and M.A. Christie - FCT Applied to the 2-D Finite Element Solution of Tracer Transport by Single Phase Flow in a Porous Medium; *Proceedings of the ICFD-Conf. on Numerical Methods in Fluid Dynamics*, Reading, Academic Press, 1986.

[16] R. Löhner, K. Morgan, M. Vahdati, J.P. Boris and D.L. Book - FEM-FCT: Combining High Resolution with Unstructured Grids; Submitted to *J.Comp.Phys.* (1986).

[17] K. Morgan, R. Löhner, J.R. Jones, J. Peraire and M. Vahdati - Finite Element FCT for the Euler and Navier-Stokes Equations; *Proc. 6th Int. Symp. Finite Element Methods in Flow Problems*, INRIA (1986).

[18] J. Donea - A Taylor Galerkin Method for Convective Transport Problems; *Int.J.Num.Meth.Engng.* 20, 101-119 (1984).

[19] R. Löhner, K. Morgan and O.C. Zienkiewicz - The Solution of Nonlinear Systems of Hyperbolic Equations by the Finite Element Method; *Int.J.Num.Meth.Fluids* 4, 1043-1063 (1984).

[20] R. Löhner, K. Morgan, J. Peraire and O.C. Zienkiewicz - Finite Element Methods for High Speed Flows; AIAA-85-1531-CP (1985).

[21] G. Sod - *J.Comp.Phys.* 27, 1-31 (1978).

[22] A. Harten and Zwaas - Self-Adjusting Hybrid Schemes for Shock Computations; *J.Comp.Phys.* 6, 568-583 (1972).

[23] I. Babuska, J. Chandra and J.E. Flaherty (eds.) - *Adaptive Computational Methods for Partial Differential Equations*; SIAM Philadelphia (1983).

[29] I. Babuska et.al. (eds.) - *Accuracy Estimates and Adaptive Refinements in Finite Element Computations*; J. Wiley and Sons (1986).

[25] B. Palmerio and A. Dervieux - Application of a FEM Moving Node Adaptive Method to Accurate Shock Capturing; *Proc. First Int. Conf. on Numerical Grid Generation in CFD*, Landshut, W. Germany, July 14-17 1986. To appear.

[26] W. Schönauer, K. Raith and K. Glotz - The Principle of Difference Quotients as a Key to the Self-Adaptive Solution of Nonlinear Partial Differential Equations; *Comp.Meth.Appl.Mech.Eng.* 28, 327-359 (1981)

[27] J.F. Dannenhoffer and J.R. Baron - Robust Grid adaptation for Complex Transonic Flows; AIAA-86-0495 (1986).

[28] R. Löhner, K. Morgan and O.C. Zienkiewicz - An Adaptive Finite Element Procedure for High Speed Flows; *Comp.Meth.Appl.Mech.Eng.* 51, 441-465 (1985).

[29] R. Löhner, K. Morgan, and O.C. Zienkiewicz - Adaptive Grid Refinement for the Compressible Euler Equations; Chapter 15 in *Accuracy Estimates and Adaptive Refinements in Finite Element Computations* (I. Babuska et.al. eds.); J. Wiley and Sons (1986).

[30] R. Löhner and K. Morgan - Improved Adaptive Refinement Strategies for Finite Element Aerodynamic Computations; AIAA-86-499 (1986).

[31] J. Peraire, M. Vahdati, K. Morgan and O.C. Zienkiewicz - Adaptive Remeshing for Compressible Flow Computations; submitted to *J.Comp.Phys.* (1986).

[32] M.J. Berger and J. Oliger - Adaptive Mesh Refinement for Hyperbolic Partial Differential Equations; *J.Comp.Phys.* 53,484-512 (1984).

[33] M.D. Smooke and M.L. Koszykowsky - Two-Dimensional Fully Adaptive Solutions of Solid-Solid Alloying Reactions; *J.Comp.Phys.* 62,1-25 (1986).

[34] J.T. Oden, P. Devloo and T. Strouboulis - Adaptive Finite Element Methods for the Analysis of Inviscid Compressible Flow: I. Fast Refinement/Unrefinement and Moving Mesh Methods for Unstructured Meshes; preprint (1986).

[35] R. Löhner - An Adaptive Finite Element Scheme for Transient Problems in CFD; submitted to *Comp.Meth.Appl.Mech.Eng.* (1986).

[36] R. Löhner, K. Morgan and O.C. Zienkiewicz - Effective Programming of Finite Element Methods for CFD on Supercomputers; pp.117-125 in *The Efficient Use of Vector Computers with Emphasis on CFD* (W. Schönauer and W. Gentzsch eds.), Vieweg Notes on Numerical Fluid Mechanics, Vol 9, Vieweg Verlag (1985).

[37] G. Ben-Door and J.M. Dewey - The Theory and Experimental Investigation of Regular and Mach Reflection over a Double Wedge; *Proc. 9th Int. Symp. Military Appl. Blast Simulation,* Oxford, U.K. (1985).

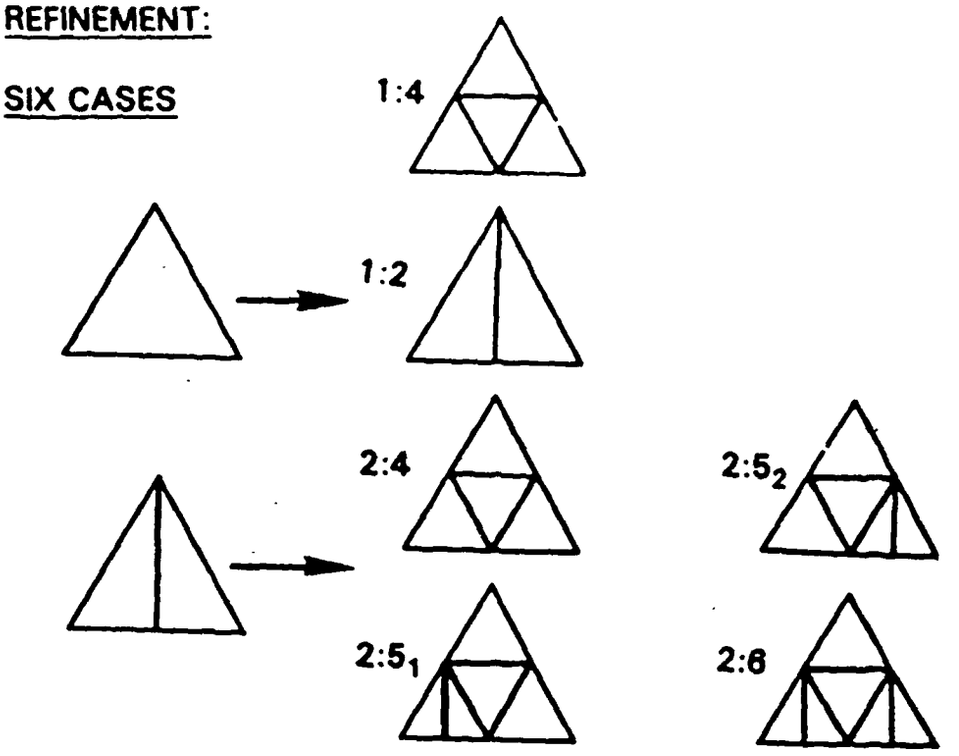


Figure 1: Allowable refinement cases

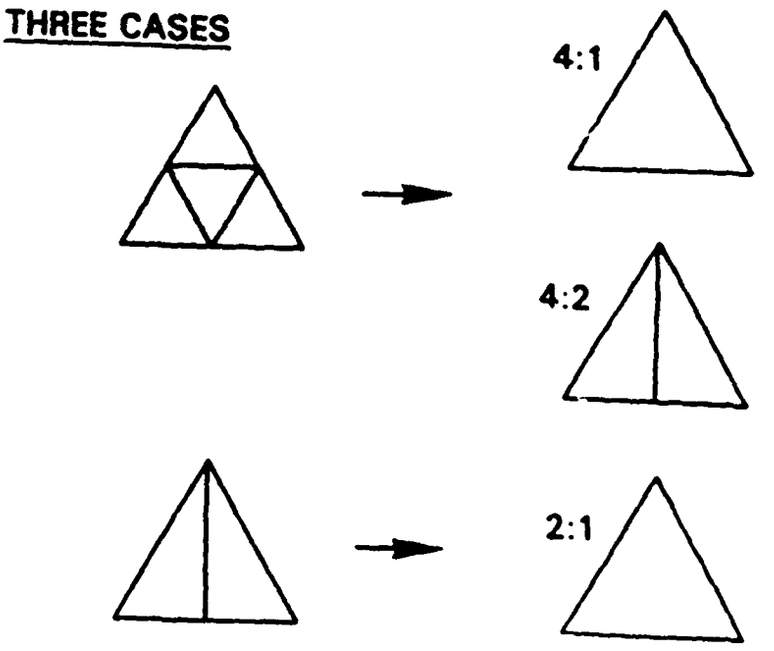


Figure 2: Allowable coarsening cases

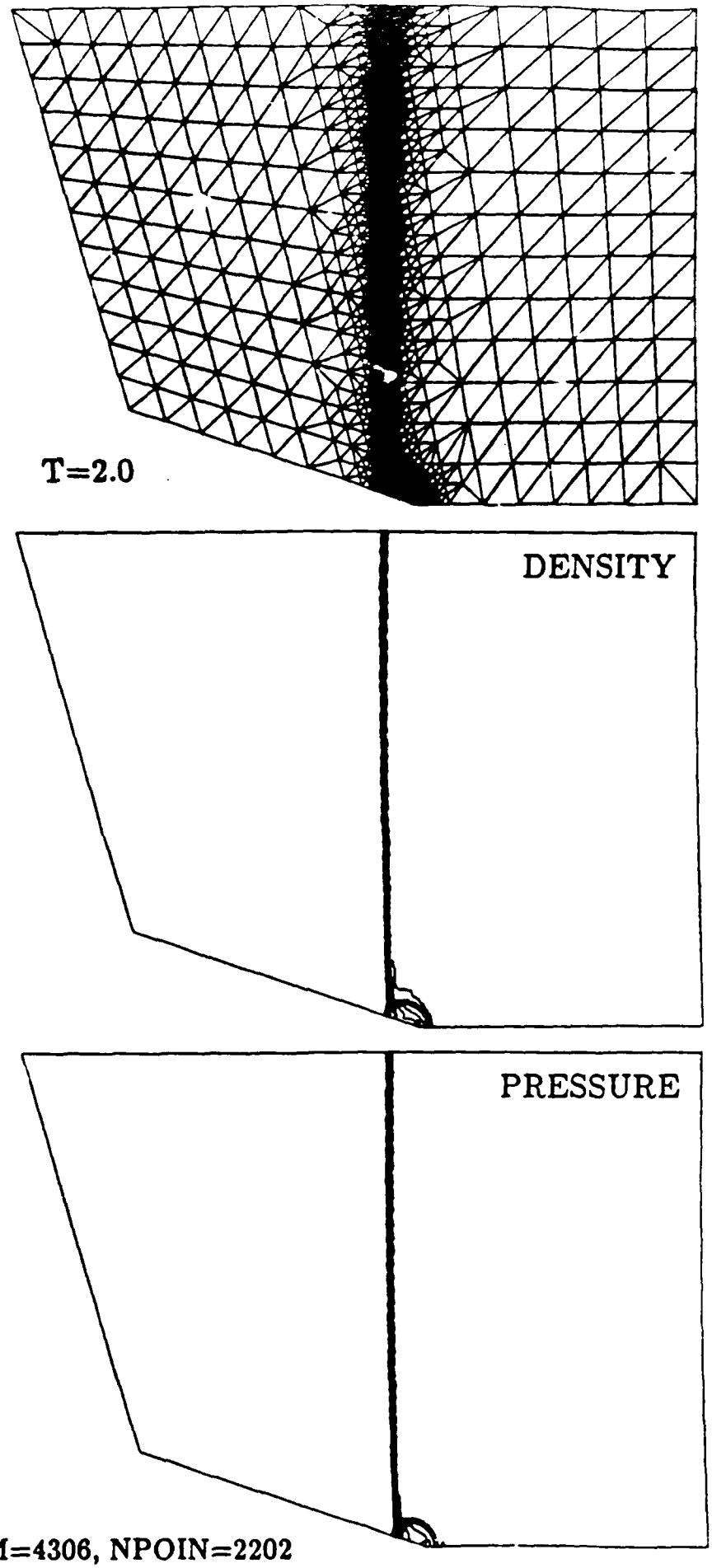


Figure 3a: Shock Reflection over Double Wedge

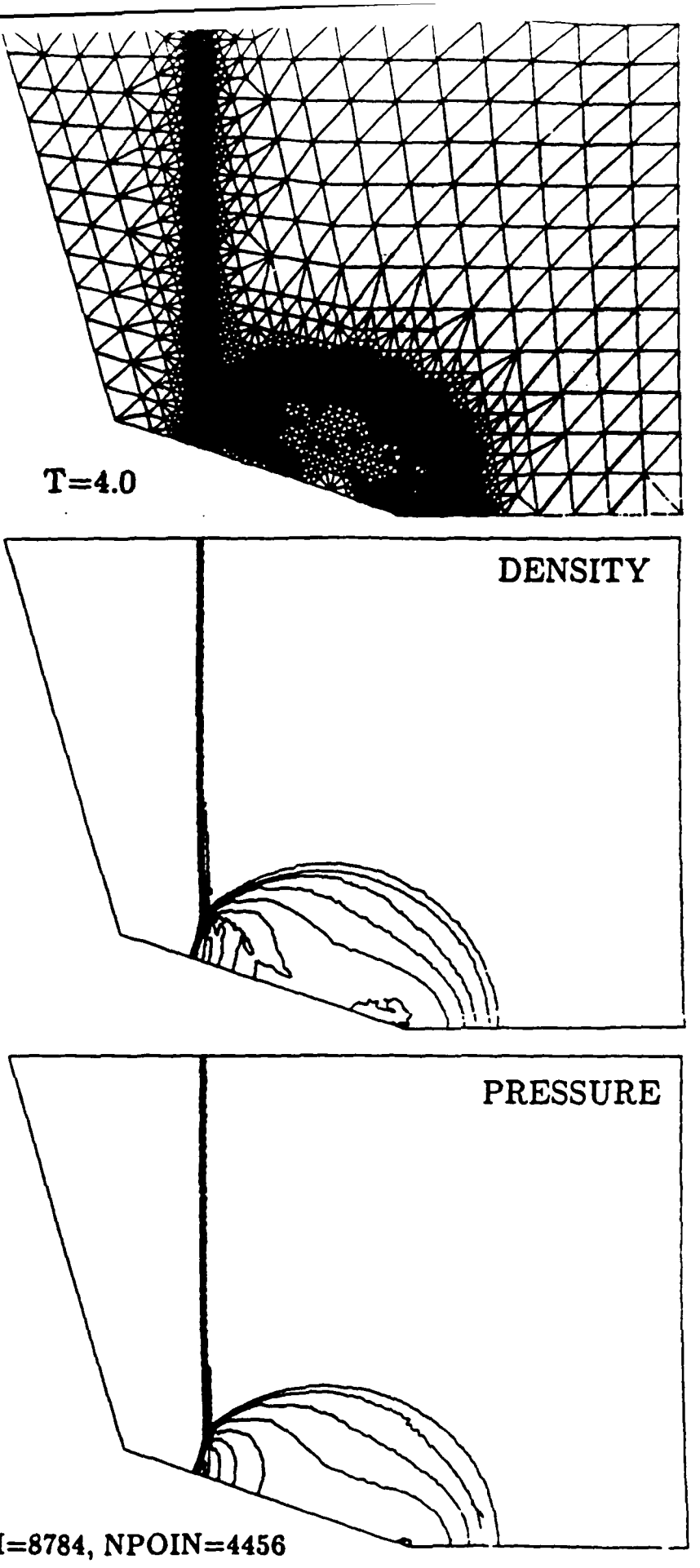


Figure 3b: Shock Reflection over Double Wedge (cont.)

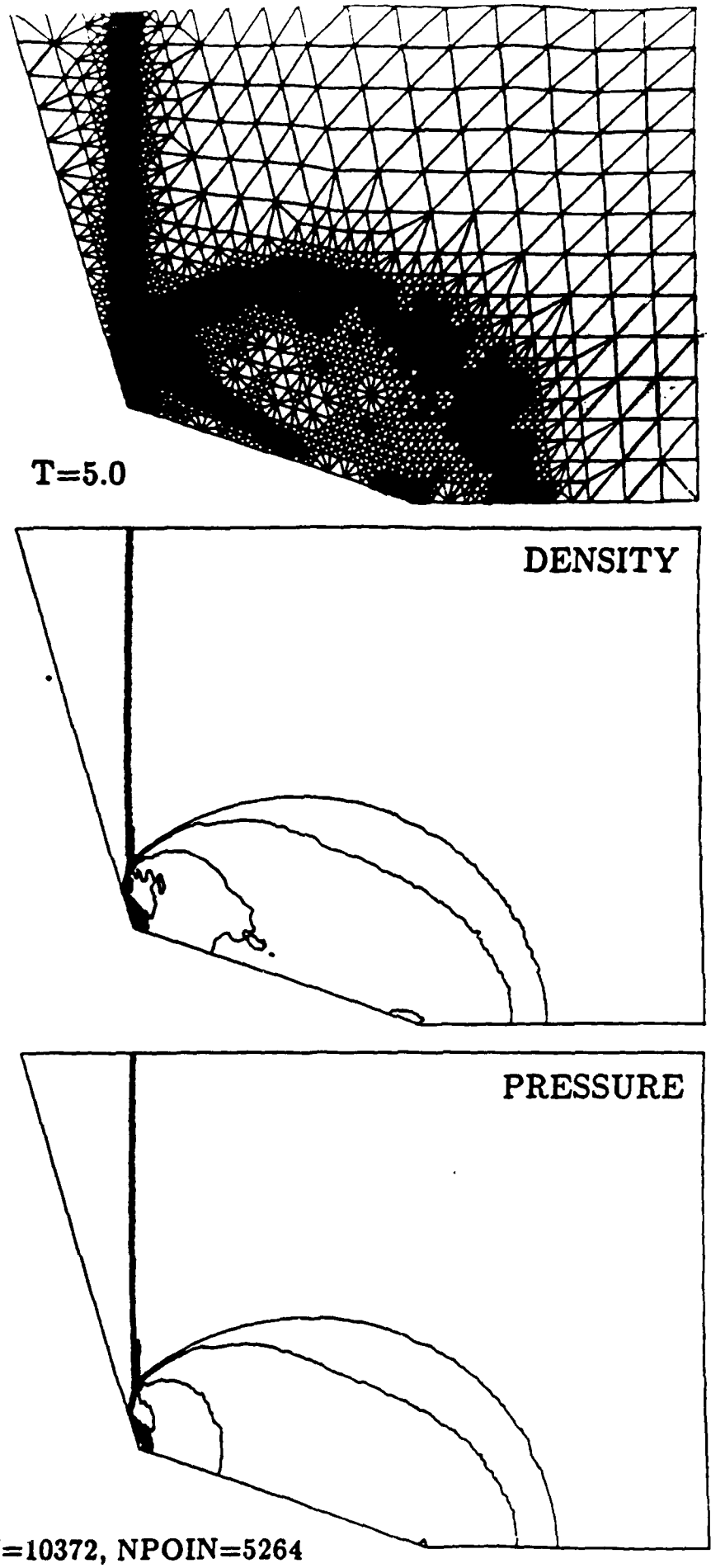


Figure 3c: Shock Reflection over Double Wedge (cont.)

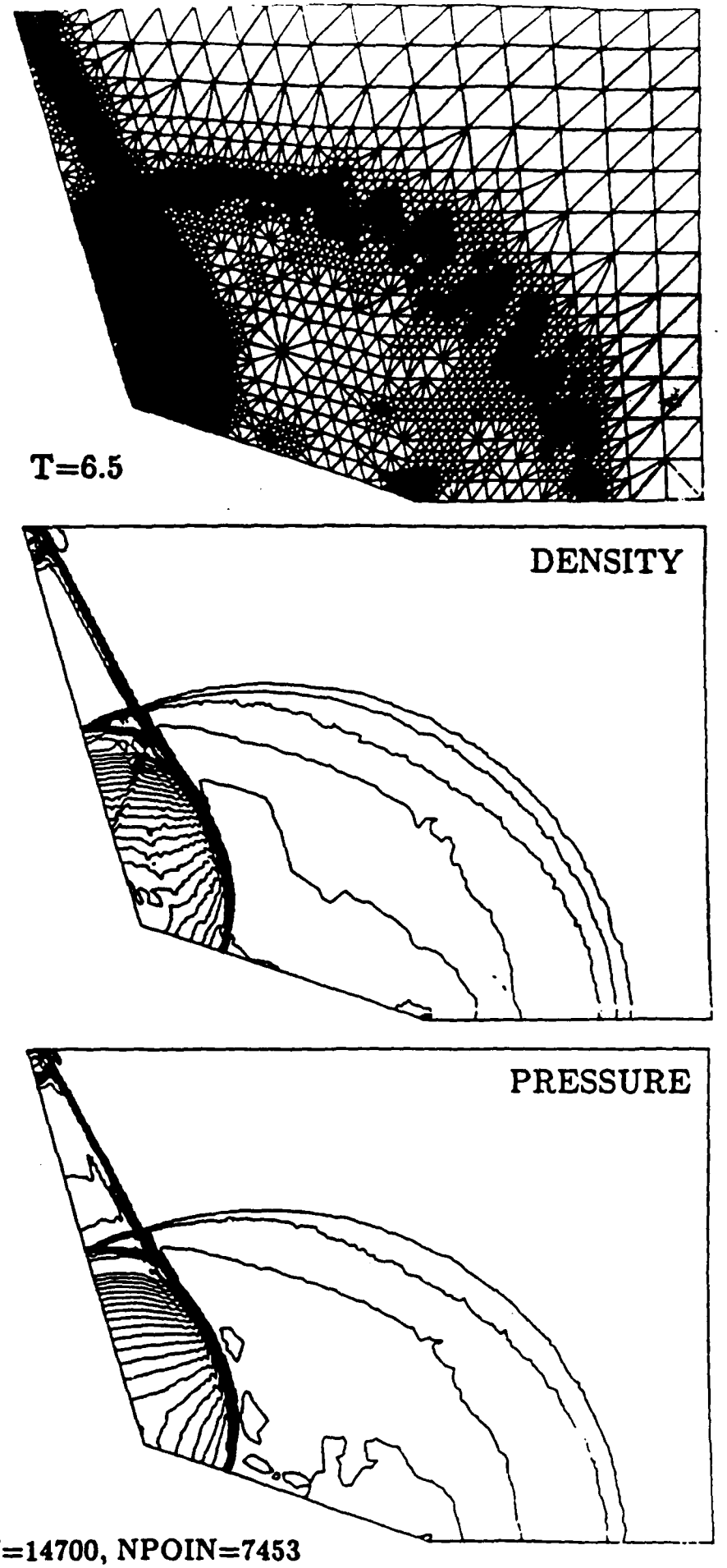


Figure 3d: Shock Reflection over Double Wedge (cont.)

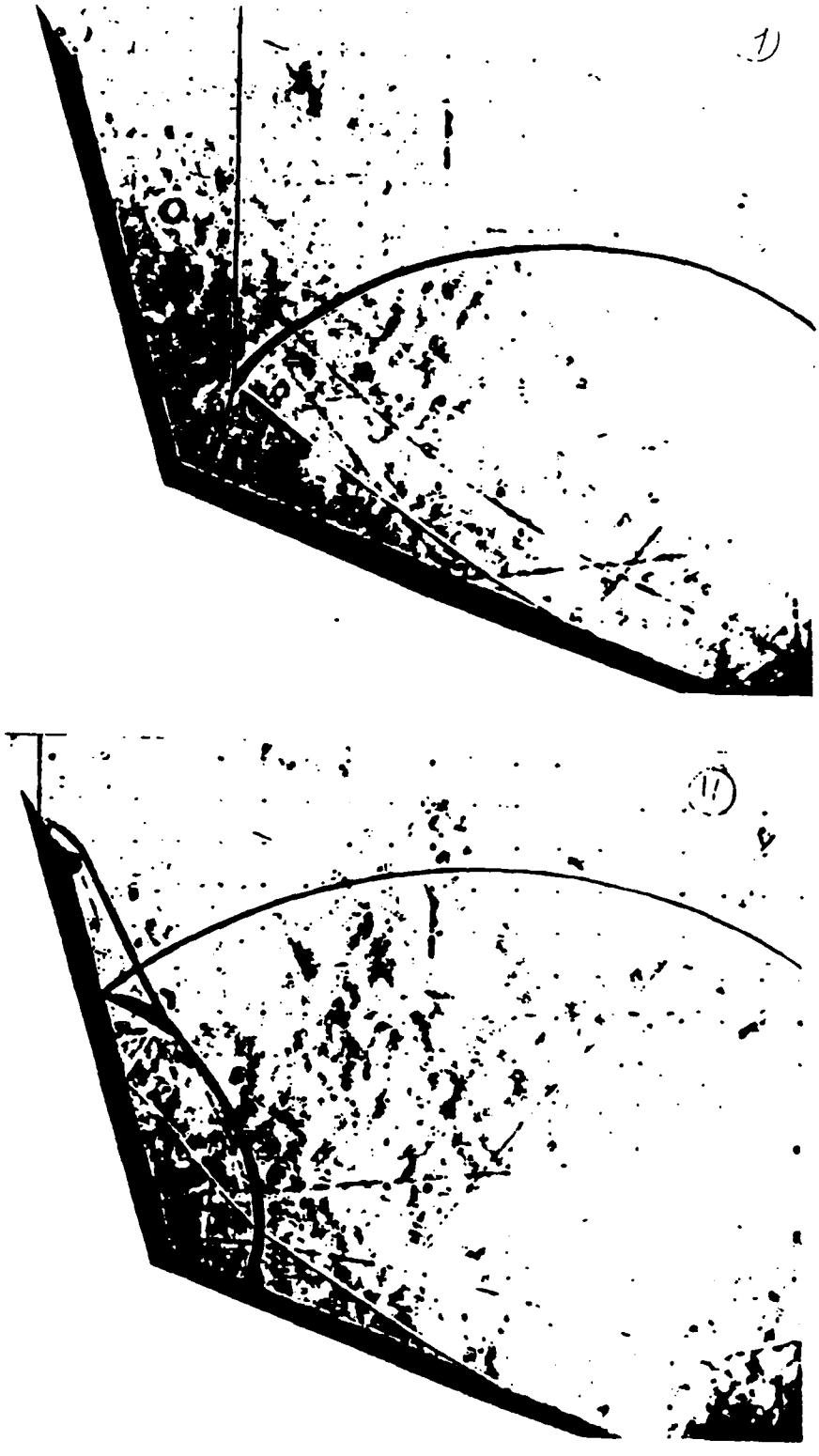
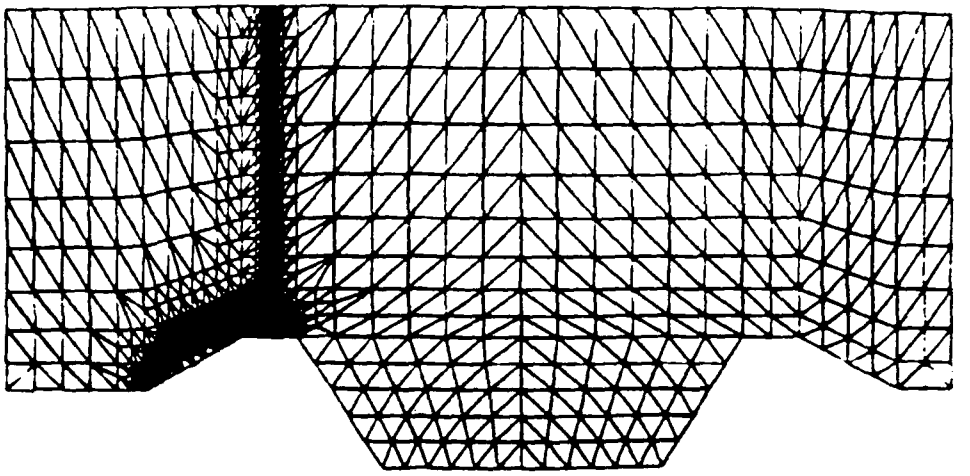
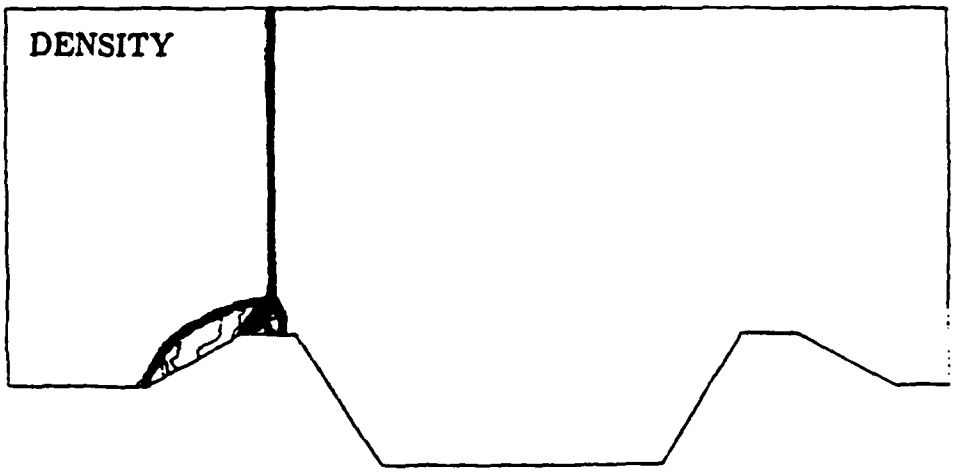
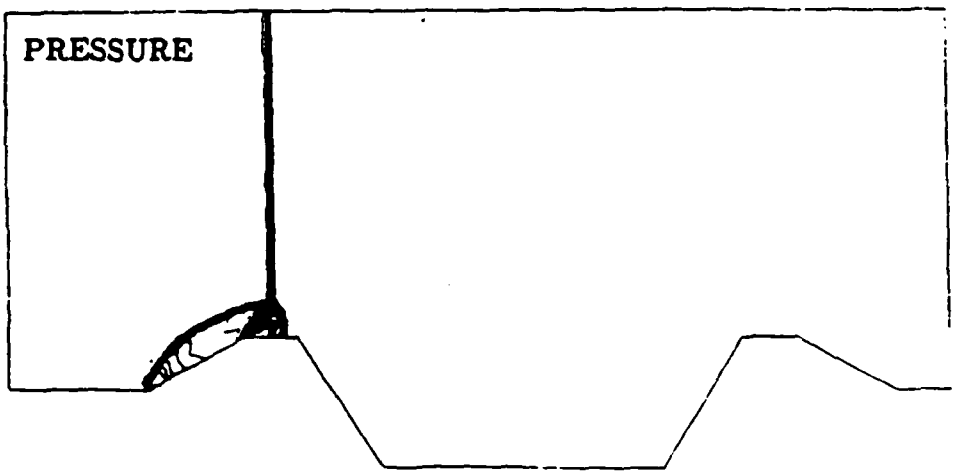


Figure 3e: Shock Reflection over Double Wedge (experimental data)



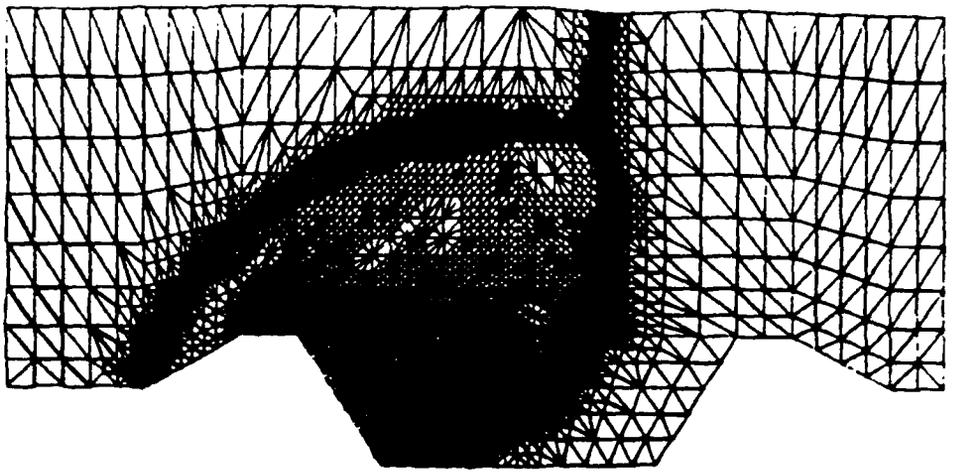
NELEM=2348, NPOIN=1236



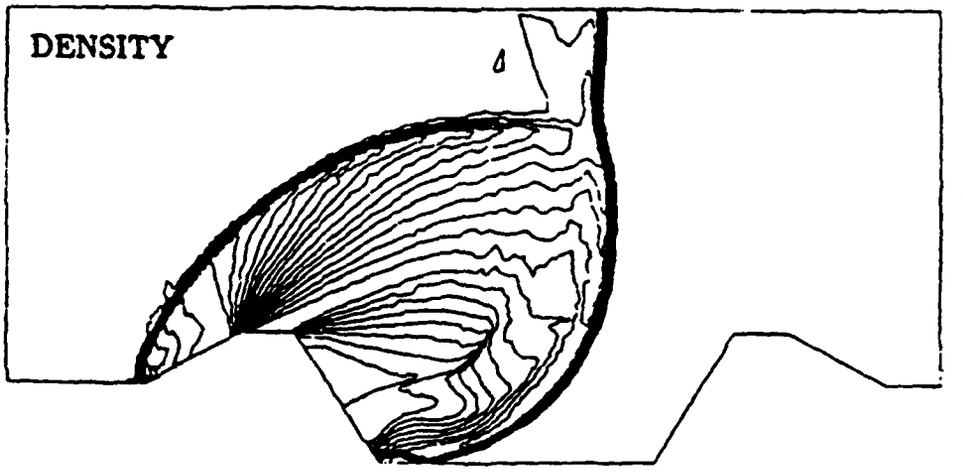


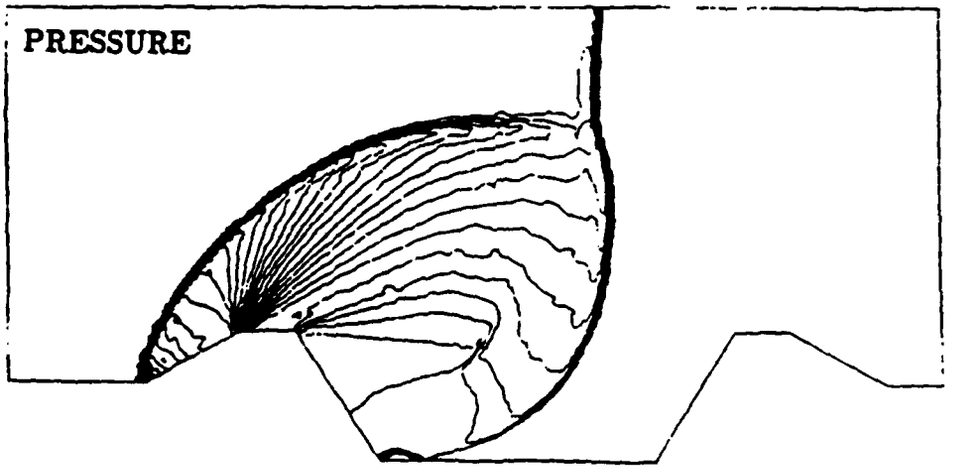
T=0.02

Figure 4a: Shock over indentation



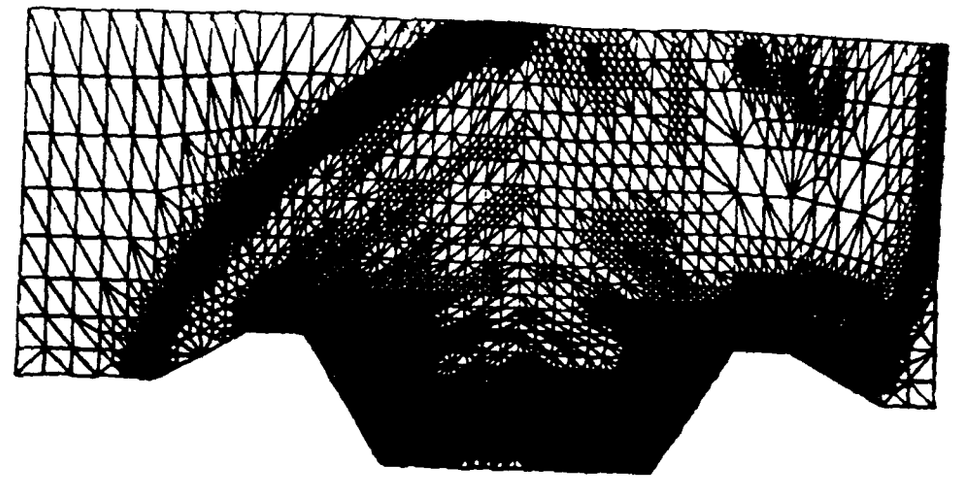
NELEM=9741, NPOIN=4956



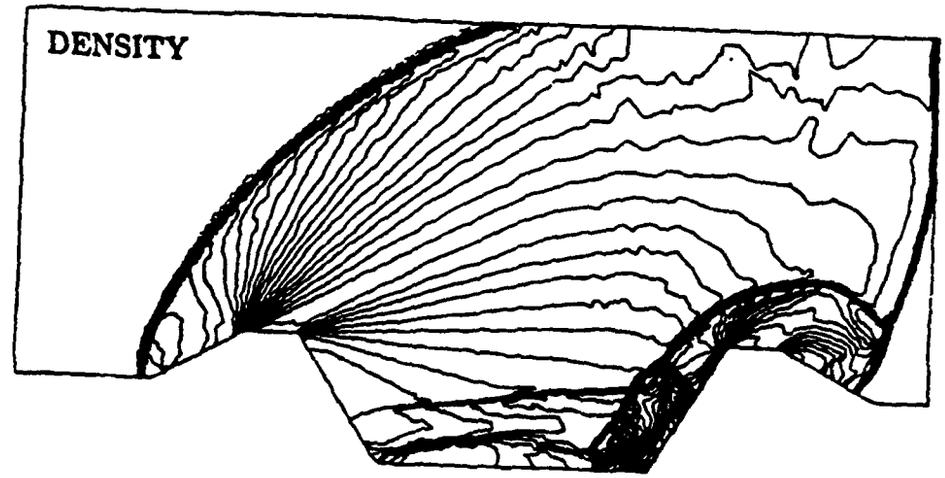


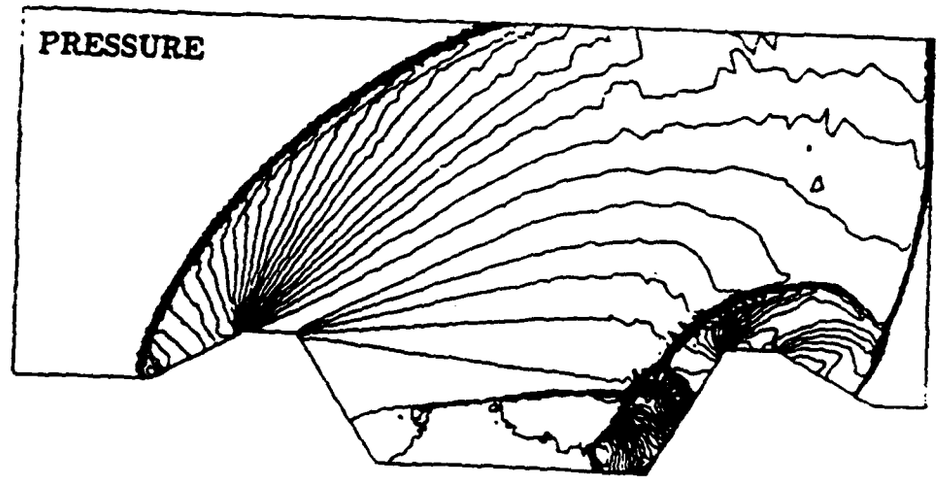
T=0.06

Figure 4b: Shock over indentation (cont.)



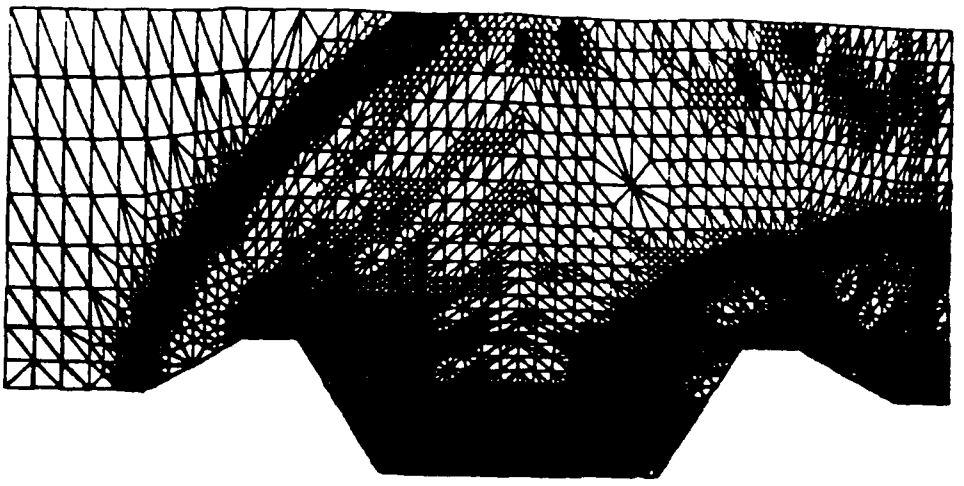
NELEM=13489, NPOIN=6891



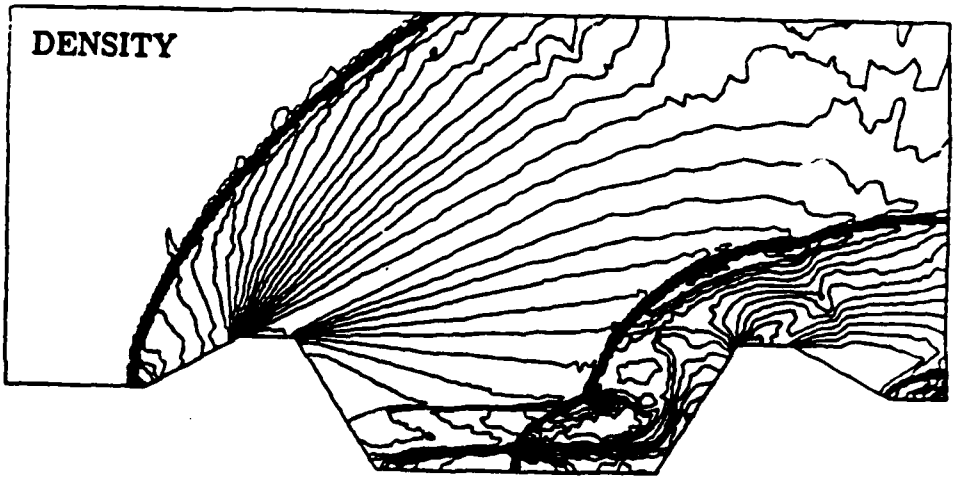


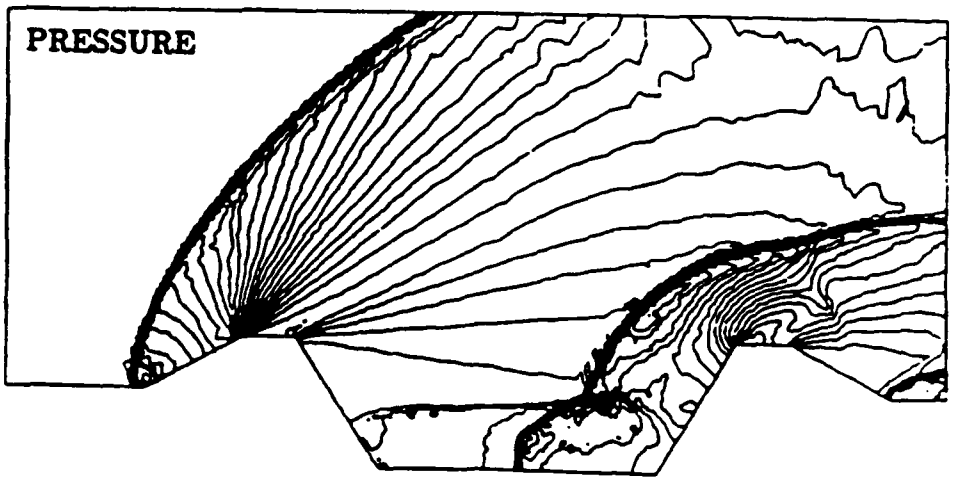
T=0.10

Figure 4c: Shock over indentation (cont.)



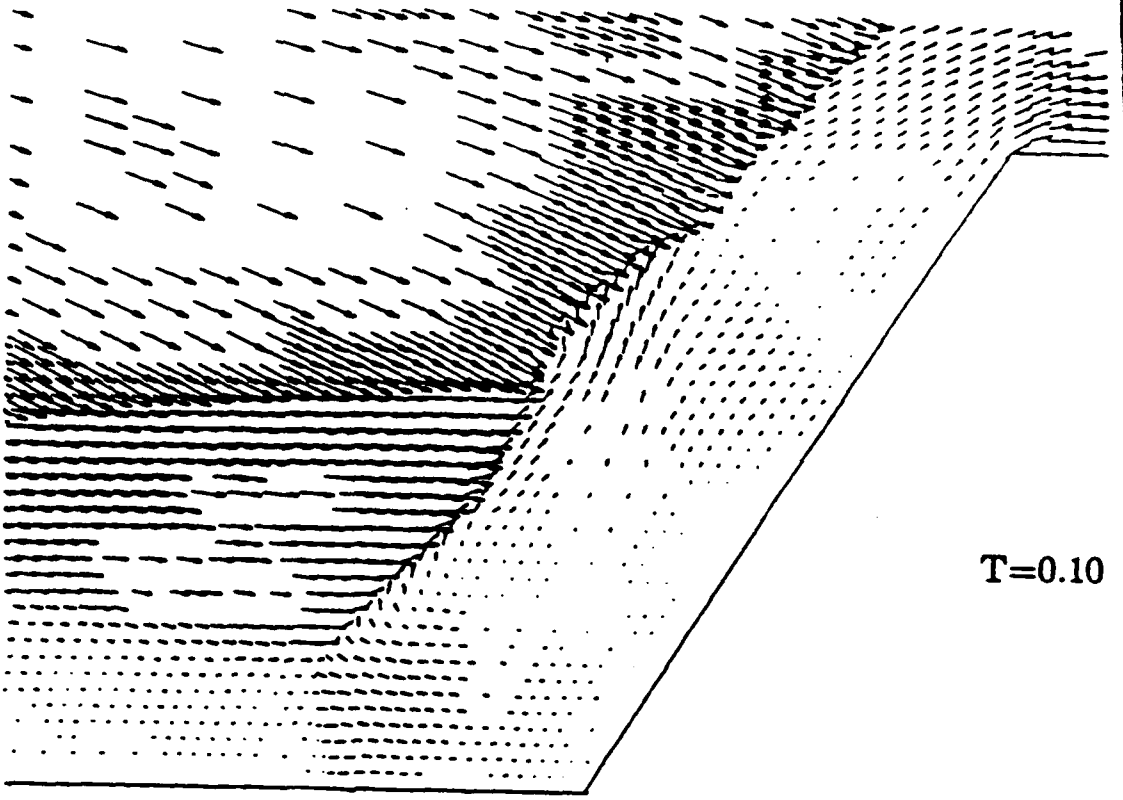
NELEM=13681, NPOIN=6984





T=0.12

Figure 4d: Shock over indentation (cont.)



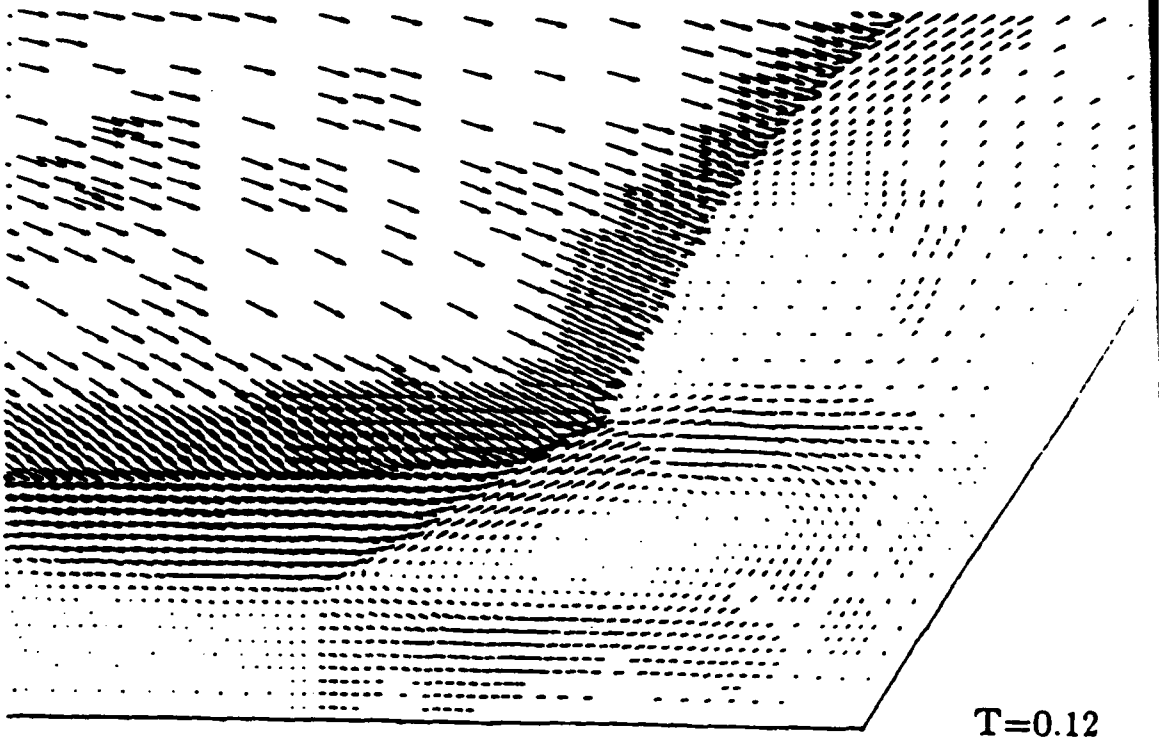


Figure 4e: Shock over indentation. Velocity Fields.

APPENDIX Y.

Adaptive Grid Refinement for the Compressible Euler Equations

Accuracy Estimates and Adaptive Refinements in Finite Element Computations
Edited by I. Babuška, O. C. Zienkiewicz, J. Gago, and E. R. de A. Oliveira

Chapter 15

Adaptive Grid Refinement for the Compressible Euler Equations

R. Löhner, K. Morgan, and O. C. Zienkiewicz

15.1 INTRODUCTION

The overall aim of our research work is to use the finite element method to solve large realistic three-dimensional high-speed compressible flow problems of practical interest. To date, we have concentrated on developing algorithms for inviscid fluids where the flow is governed by a system of hyperbolic conservation laws of the form

$$\frac{\partial \mathbf{U}}{\partial t} + \frac{\partial \mathbf{F}_j}{\partial x_j} = \mathbf{0} \tag{15.1}$$

where the summation convention has been employed and the range of j depends upon the number of space dimensions. With the overall aim of the work in mind, we have, from the outset, restricted consideration to solution algorithms which appear to be optimal in terms of CPU-time and storage requirements. It is clear that such an optimal algorithm will be characterized by the ability to produce an accurate solution with the use of the minimum amount of nodes/elements. This in turn implies that the basic solution algorithm should be capable of adaptively refining the mesh as the solution proceeds. If it is required to accurately follow the transient behaviour of the solution of Equation (15.1), then the adaptive procedure should also be capable of derefining certain portions of the mesh.

In this chapter, we describe our first attempts to develop an adaptive refinement strategy. The methods proposed will be designed to produce accurate steady-state solutions to Equation (15.1) and will not make any assumptions about the structure of the finite element grid.

15.2 EXPLICIT TIME-STEPPING SCHEME

The solution of Equation (15.1) is advanced towards the steady state by means of an explicit Euler–Taylor–Galerkin procedure. The Taylor–Galerkin family

of solution methods for advection-dominated problems was first introduced by Donea [1]. The authors [2,3] have described the use of an explicit Euler–Taylor–Galerkin procedure for obtaining both transient and steady-state solutions to Equation (15.1). The procedure uses Taylor expansions in time to produce the time-stepping scheme

$$\mathbf{U}^{n+1} = \mathbf{U}^n - \Delta t \frac{\partial \mathbf{F}_j^n}{\partial x_j} + \frac{\Delta t^2}{2} \frac{\partial}{\partial x_j}\left(\mathbf{A}_j^n \frac{\partial \mathbf{F}_k^n}{\partial x_k} \right) \tag{15.2}$$

where a superscript n denotes an evaluation at time $t = t_n$, $t_{n+1} = t_n + \Delta t$, and

$$\mathbf{A}_j = \frac{\mathrm{d}\mathbf{F}_j}{\mathrm{d}\mathbf{U}} \tag{15.3}$$

Piecewise linear interpolation is used for $\mathbf{U}$, $\mathbf{F}_j$, and piecewise constant interpolation is used for $\mathbf{A}_j$. The application of the Galerkin weighted residual method [4] then leads to an equation system of the form

$$\mathbf{M}(\hat{\mathbf{U}}^{n+1} - \hat{\mathbf{U}}^n) = \mathbf{f}^n \tag{15.4}$$

where $\mathbf{M}$ is the standard mass matrix and $\hat{\mathbf{U}}$ is the vector of nodal values of $\mathbf{U}$. If only steady-state solutions are of interest, the mass matrix may be lumped and the value of $\hat{\mathbf{U}}^{n+1}$ obtained immediately. The use of the $\mathbf{A}_j$ matrices is computationally expensive [5] and this method has been superceded by a two-step scheme [3] of improved efficiency. A detailed discussion of the application of the commonly encountered boundary indications within this formulation is discussed elsewhere [6]. For problems involving strong shocks the solution of Equation (15.4) is smoothed by the application of an artificial viscosity before proceeding to the next time step. This has been accomplished using the models of Lapidus [7] and also MacCormack and Baldwin [8], but we are currently favouring a method based upon a consistent extension of the Lapidus model to multidimensional configurations [9].

15.3 REQUIREMENT FOR ADAPTIVE REFINEMENT

Ideally, we would like to ensure that at any stage in the solution process

$$\max_{\Omega_\varepsilon} |U_l - \hat{U}_l| < \varepsilon \tag{15.5}$$

where U_l denotes the lth component of the solution vector $\mathbf{U}$, $\hat{U}_l$ denotes the corresponding component of the approximate solution, ε is a specified tolerance, $|\cdot|$ denotes an appropriate measure, and Ω_ε is that part of the solution domain Ω in which we wish to achieve this accuracy. In practice we choose $l = 1$ so that it is the density which is subjected to the requirement of Equation (15.5). For hyperbolic problems, errors introduced in certain parts of the domain

propagate throughout the entire domain and so we are, in general at least, forced to require that

$$\max_{\Omega} |U_1 - \hat{U}_1| < \varepsilon \tag{15.6}$$

For reasons of programming convenience and computer execution speed, we have attempted to satisfy this criterion by means of an *a posteriori* adaptation of the grid. As only a limited number of degrees of freedom can (and will, eventually) be employed, the optimal mesh will have the property that the error is evenly distributed over the elements [10]. Within the context of our explicit time-stepping procedure, the idea is then to advance the solution to the steady state on a fixed mesh, then adapt the grid, and produce a new solution. The adaption process is repeated until a solution of the desired accuracy results.

15.4 MEASUREMENT OF THE ERROR

Nothing has been said about the way in which the measurement of the error in Equation (15.6) is accomplished. Several possibilities exist and these have been discussed and compared elsewhere [11]. We have chosen to employ locally classic theoretical (*a priori*) error estimates [12, 13]. This method does not require the introduction of further degrees of freedom and only first or second derivatives need to be evaluated. For elliptic problems, the appropriate error measure appears naturally as the energy norm, whereas for hyperbolic problems the theory is far from complete. Nevertheless, one can assume, in a least squares sense, that

$$|U_1 - \hat{U}_1| < ch^l |U_1|_l \tag{15.7}$$

where c is an unknown constant, h is a representative element length, $|\cdot|$ denotes the L_2 norm, and $|\cdot|_l$ the appropriate seminorm [14]. As only a uniform distribution of errors for all elements is the aim of the refinement process, and since the exact solution U is unknown, the practical form of the requirement becomes

$$h^l |\hat{U}_1|_l = \text{constant} \tag{15.8}$$

In the commonly used finite element manner, this can be employed with $l = 1$ or $l = 2$.

15.5 REFINEMENT STRATEGY

Two methods of mesh refinement can be readily implemented in conjunction with the refinement indicator of Equation (15.8). In the first, the total number of nodes and elements remains fixed but their position is altered, whereas in the second, the nodes are fixed but new nodes and elements are added. We have pursued both approaches and will describe the implementations here.

15.5.1 Mesh movement

When a mesh refinement is required, the element sides are considered as springs of prescribed stiffness and the nodes are then moved until the spring system is in equilibrium. Consider two adjacent nodes i and j belonging to elements E_1 and E_2, as shown in Figure 15.1. The force $\mathbf{f}_{ij}$ exerted by the spring connecting these two nodes is taken to be

$$\mathbf{f}_{ij} = c_{ij}(\mathbf{X}_i - \mathbf{X}_j) \tag{15.9}$$

where c_{ij} is the stiffness of the spring and $\mathbf{X}_i$ and $\mathbf{X}_j$ are the position vectors of nodes i and j respectively. Assuming that

$$|\mathbf{X}_i - \mathbf{X}_j| \approx h \tag{15.10}$$

the refinement requirement of Equation (15.8) will be satisfied, when $l = 1$, if we define

$$c_{ij} = |\hat{U}_1|_1 = |\hat{U}_1|_1^{E_1} + |\hat{U}_1|_1^{E_2} \tag{15.11}$$

and allow the mesh to move to an equilibrium position. This means that the vector $\mathbf{f}(\mathbf{X})$ of nodal forces should equal zero; this is achieved by viscous relaxation [15], i.e. by solving the system of equations

$$\mathbf{c}\frac{d\mathbf{X}}{dt} = \mathbf{f}(\mathbf{X}) \tag{15.12}$$

where $\mathbf{c}$ is a lumped matrix with diagonal entries

$$c_{ii} = \sum_j c_{ij} \tag{15.13}$$

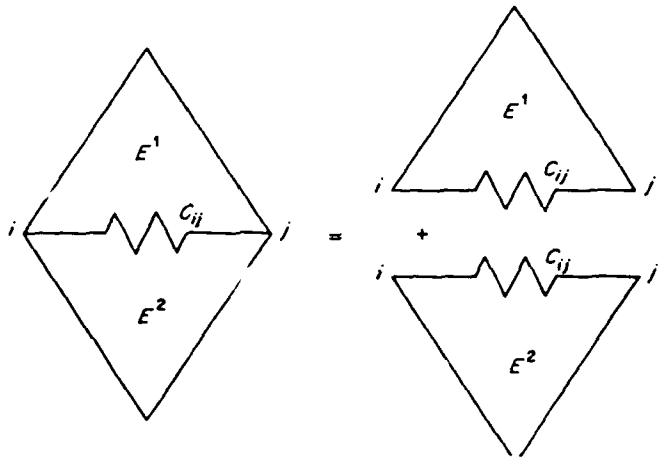


Figure 15.1 Idealization of the element sides as a system of springs

and **X** is the vector of nodal coordinates. The system is solved by using an explicit time integration and convergence is assumed when the decrease of the residual R becomes slow. With the residual after step $(m+1)$ defined by

$$R^m = \sqrt{[(\mathbf{X}^{m+1} - \mathbf{X}^m)^T(\mathbf{X}^{m+1} - \mathbf{X}^m)]} \tag{15.14}$$

the solution is taken to be converged when

$$|R^{m+1} - R^m| < \alpha R^m \tag{15.15}$$

with α usually of the order of 0.05.

This technique is not always guaranteed to produce meshes of a better quality, as badly formed elements can appear in regions (such as shocks) in which the spring coefficients c_{ij} vary rapidly over a small distance. To avoid this problem, the definition of c_{ij} given in Equation (15.11) is replaced by the expression

$$c_{ij}^* = 1 + \frac{Ac_{ij}}{B + c_{ij}} \tag{15.16}$$

This blending function definition for the spring stiffnesses (shown diagrammatically in Figure 15.2, ensures that excessively small or excessively large elements are avoided and that meshes of acceptable quality are produced.

After having moved the mesh, the values of the unknowns at the new grid points are found by interpolation. In order to avoid a costly, general interpolation procedure, the assumption is made that the nodes will not move beyond the patch of elements surrounding them. This is justified, as in zones of high gradients this will always be the case, since the elements are 'compressed', while for zones of low gradients, the error thus introduced is small. The elements

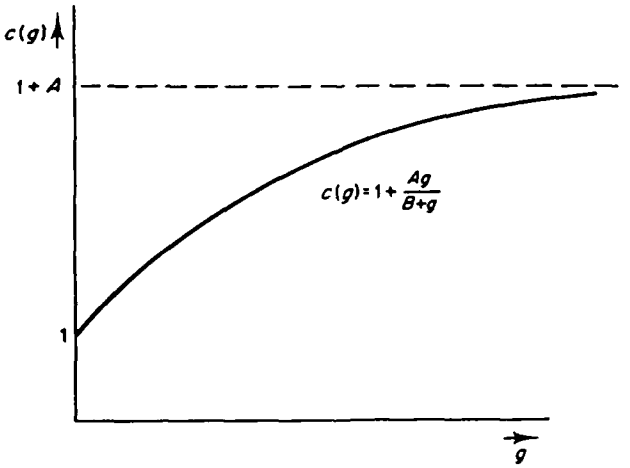


Figure 15.2 Relation between spring stiffness and g

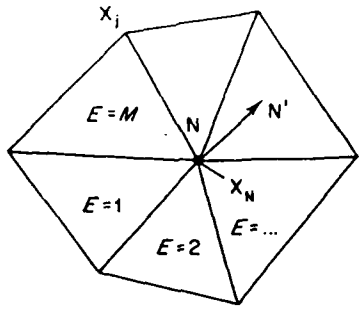


Figure 15.3 Patch of elements surrounding node N

surrounding each node are stored before the mesh is moved. When the new position of the notes is reached ($N \rightarrow N'$ in Figure 15.3), the element whose centre is closest to N' is used for the interpolation.

As the changes of element size which are produced can be considerable, badly deformed elements may appear during the course of the computation. It has been found that the best way of dealing with this problem is simply to remove such elements from the calculation. With reference to Figure 15.4, let us define the following geometrical quantities for a particular element: S_{min} is the shortest element side; i, j the nodes of the shortest element side; S_1, S_2 the remaining element sides, $D = \min(S_1/S_{min}, S_2/S_{min})$. An element is generally considered to be ill-deformed if $D > b$, where b is a chosen but arbitrary number. When such an ill-deformed element is encountered, three different possibilities exist:

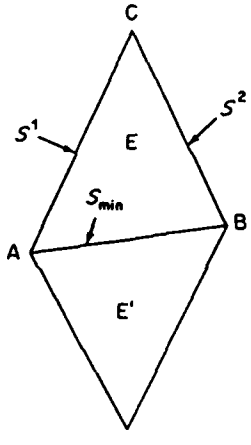


Figure 15.4 Treatment of badly deformed elements

(a) The nodes i and j lie on the boundary of the solution domain. If $D > 2$ and one of the nodes if fixed, then the free node is removed together with the element.
(b) Either node i or j lies on the boundary of the solution domain. If $D > 2$ then the other element that is contiguous with the side ij is examined. If it also satisfies $D > 2$ then the boundary node is kept while the two elements as well as the other node are removed.
(c) Node i and j lie within the solution domain. If $D > 2.3$ the contiguous element is examined and if it also satisfies $D > 2.3$ then both elements and one node are removed from the mesh.

15.5.2 Mesh enrichment

When the mesh is refined using mesh enrichment, we sweep over all elements of the mesh and determine β, which is the maximum value of the refinement indicator, i.e.

$$\beta = \max_{\text{elements}} h^l |\hat{U}_1|_l \tag{15.17}$$

The enrichment strategy is then to refine all these elements for which the indicator is larger than a certain proportion of this maximum value, i.e. all elements for which

$$h^l |\hat{U}_1|_l > a\beta \tag{15.18}$$

where a is normally chosen to be 0.6. The basic refinement process is to divide each element into four smaller elements as shown in Figure 15.5. To avoid the problem of 'hanging nodes' the refined region has to be surrounded by a transition region in which certain elements are subdivided into two smaller elements. A full discussion of this method has been given elsewhere [3, 11, 13, 16] and need not be repeated here. A recent paper by Palmerio [17] uses a similar refinement strategy but a different refinement indicator.

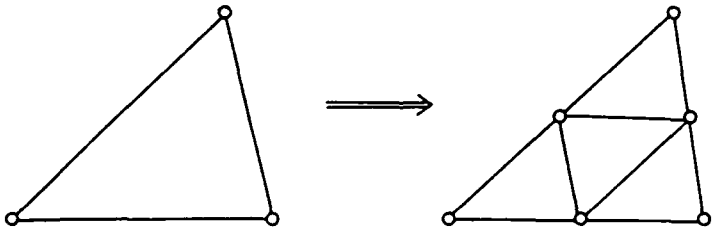


Figure 15.5

15.6 NUMERICAL EXAMPLES

Two problems have been chosen to illustrate the application of the adaptive refinement processes discussed above. It should be noted that all the results shown were obtained using a staggering procedure which enables the solution to be advanced with different time steps in different regions of the mesh [18]. This technique is essential for economic computing as the element sizes may vary by orders of magnitude.

15.6.1 Supersonic flow past a wedge

The problem specification and the solution domain are shown in Figure 15.6(a). Also shown in this figure is the initial mesh discretization for the case in which

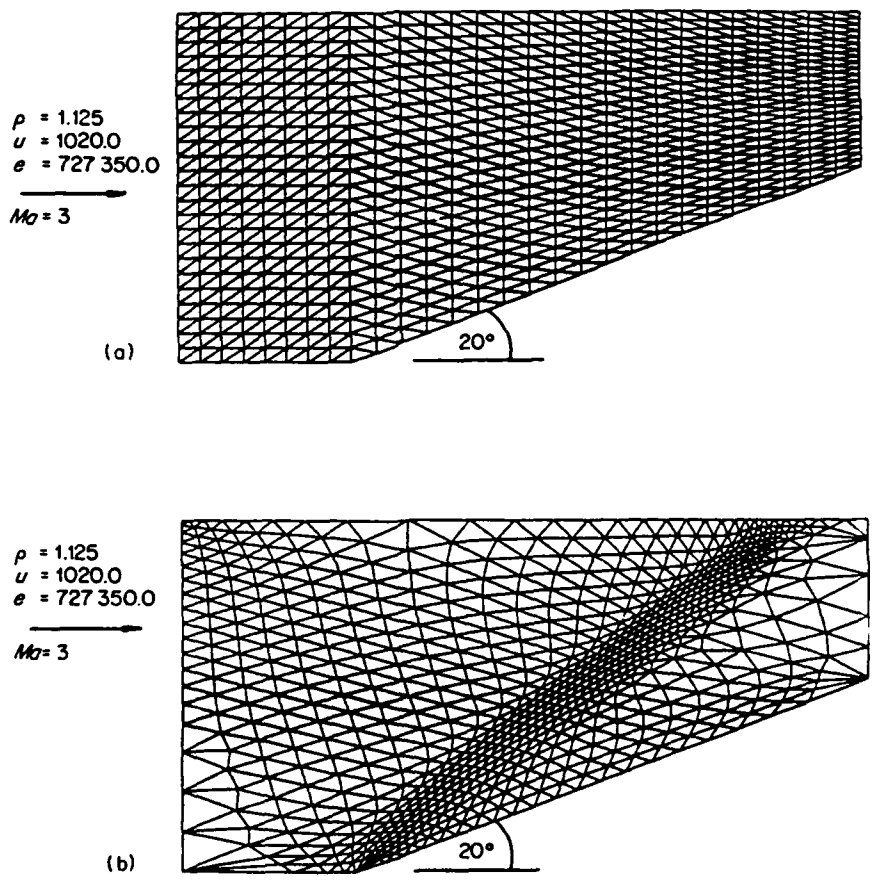


Figure 15.6 Supersonic flow past a wedge: (a) initial configuration and (b) after 300 steps

adaptive mesh refinement is to be achieved by mesh movement. This mesh was kept fixed for 100 time steps and then moved every 40 steps over the next 200 steps. The final mesh is illustrated in Figure 15.6(b). During the mesh movement procedure about 120 elements (or 10 per cent. of the total elements present) were removed from the calculation because they were judged to be badly deformed according to the criteria arrived in the previous section. To illustrate the gain in accuracy attainable by moving the mesh, the solutions obtained on the original mesh and on the moved mesh are shown in Figure 15.7. The superiority of the moved mesh solution is clearly apparent.

When this problem is solved using mesh enrichment, the chosen initial mesh is coarser, as shown in Figure 15.8. The solution was advanced on this mesh for 100 steps and then the mesh was refined. A further refinement was performed after another 100 steps of the calculation. The computation ceased after a further 50 time steps. The sequence of grids produced is shown in Figure 15.8 and the

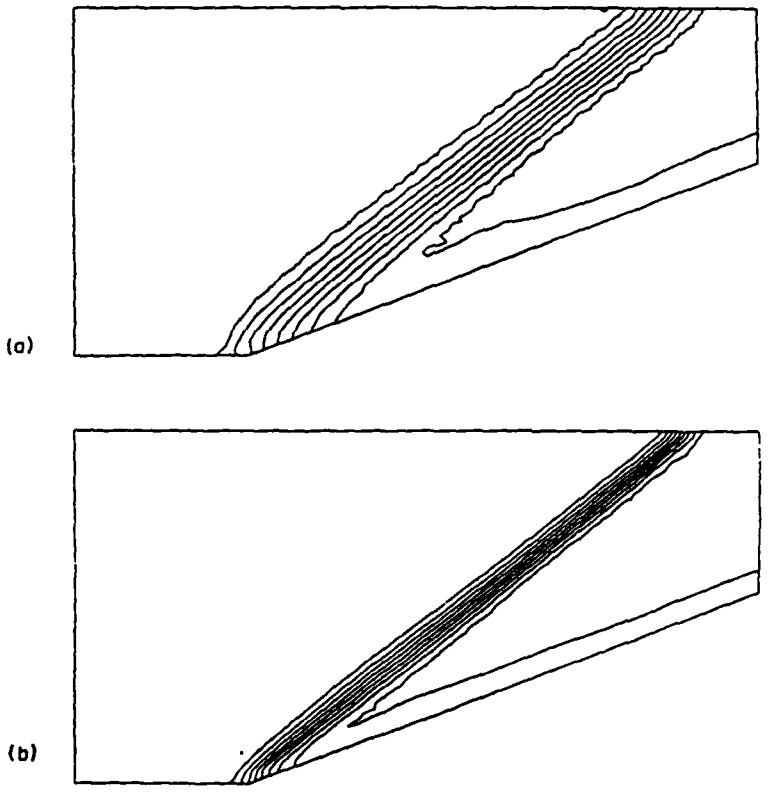


Figure 15.7 Density solution obtained (a) without mesh movement and (b) with mesh movement

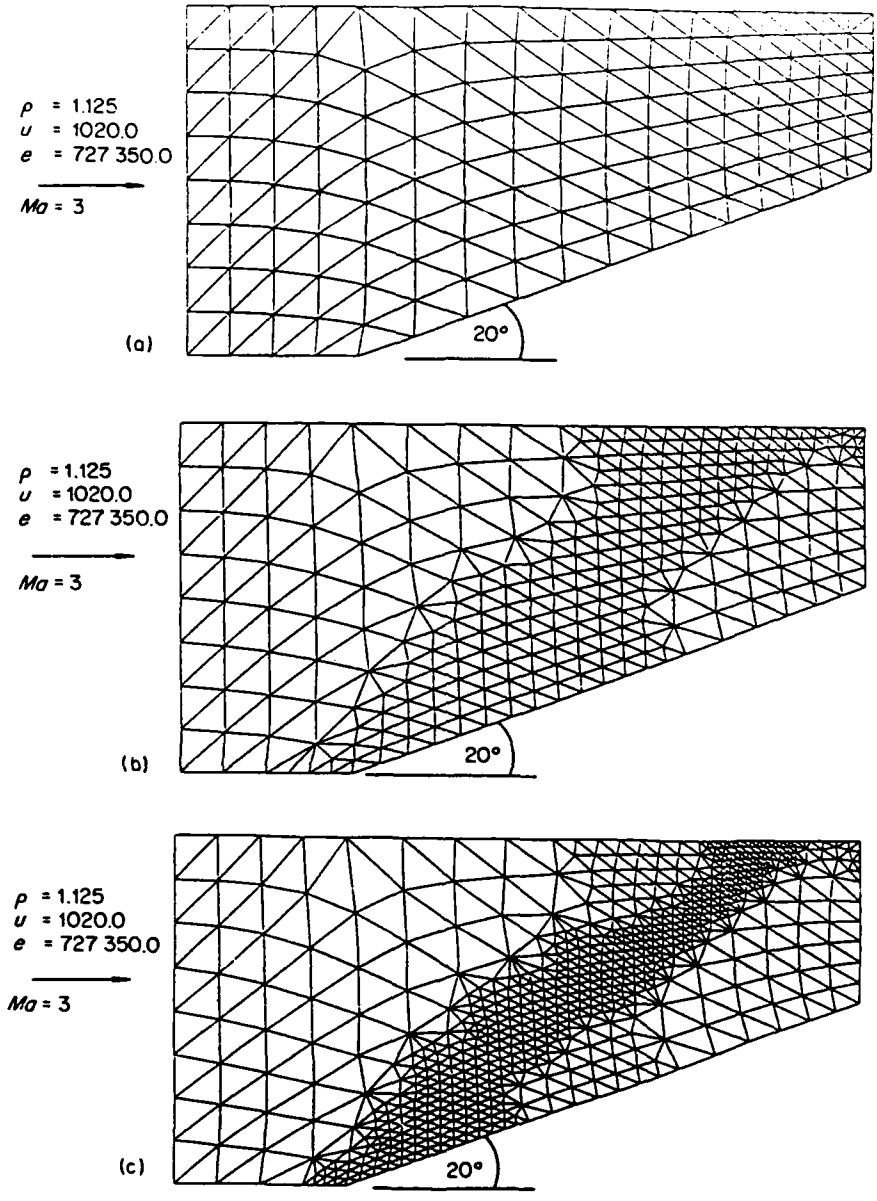


Figure 15.8 Supersonic flow past a wedge: (a) initial configuration, (b) after 101 steps, and (c) after 201 steps

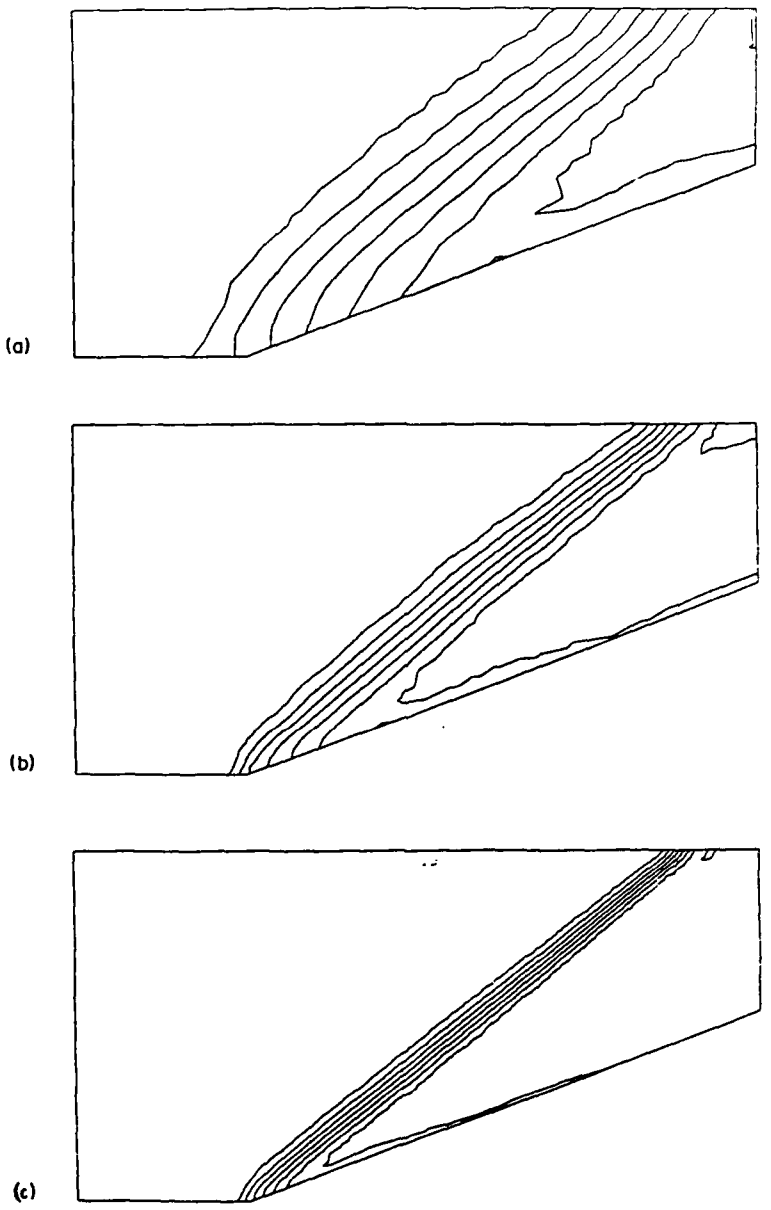


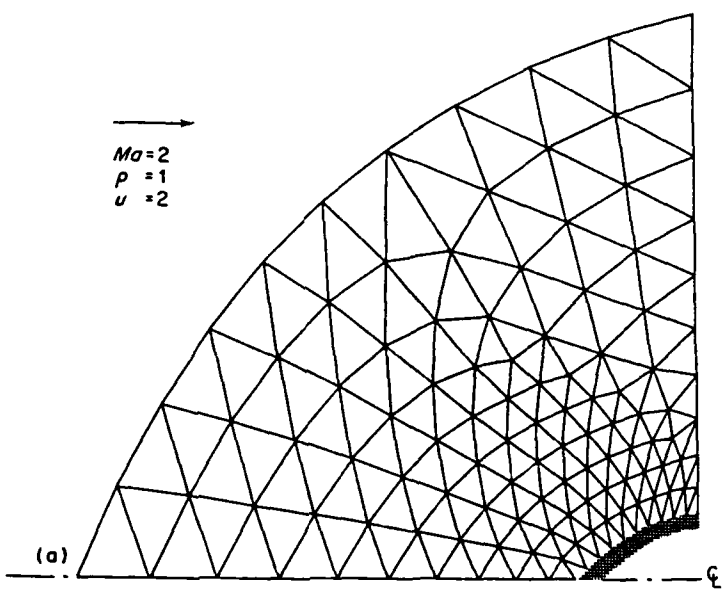
Figure 15.9 Density after (a) 100 steps, (b) 200 steps, and (c) 250 steps

solutions obtained are given in Figure 15.9. The improvement in solution quality is again evident.

15.6.2 Supersonic flow past a nose cone

The problem specification and the solution domain are shown in Figure 15.10(a). Also shown is the initial discretization for a run employing mesh movement. The mesh was moved every 25 steps for the first 200 steps and then moved again after a further 50 steps. The meshes and solutions produced are given in Figure 15.10(b) and (c). It should be noted that, although in the initial discretization consideration was given to the fact that more elements were needed near the cone, the changes in element size which have been produced are still considerable; near the inflow boundary the element size has nearly doubled, whereas near the cone the element size has almost halved.

For the solution of this problem using mesh enrichment, the initial discretization is illustrated in Figure 15.11. This mesh was enriched every 100 steps for 300 time steps and then a further 20 steps were performed. The grids produced and the corresponding solutions are shown in Figures 15.11 and 15.12, and it can be seen how the detached shock becomes the driving refinement phenomenon after the first two refinement steps.



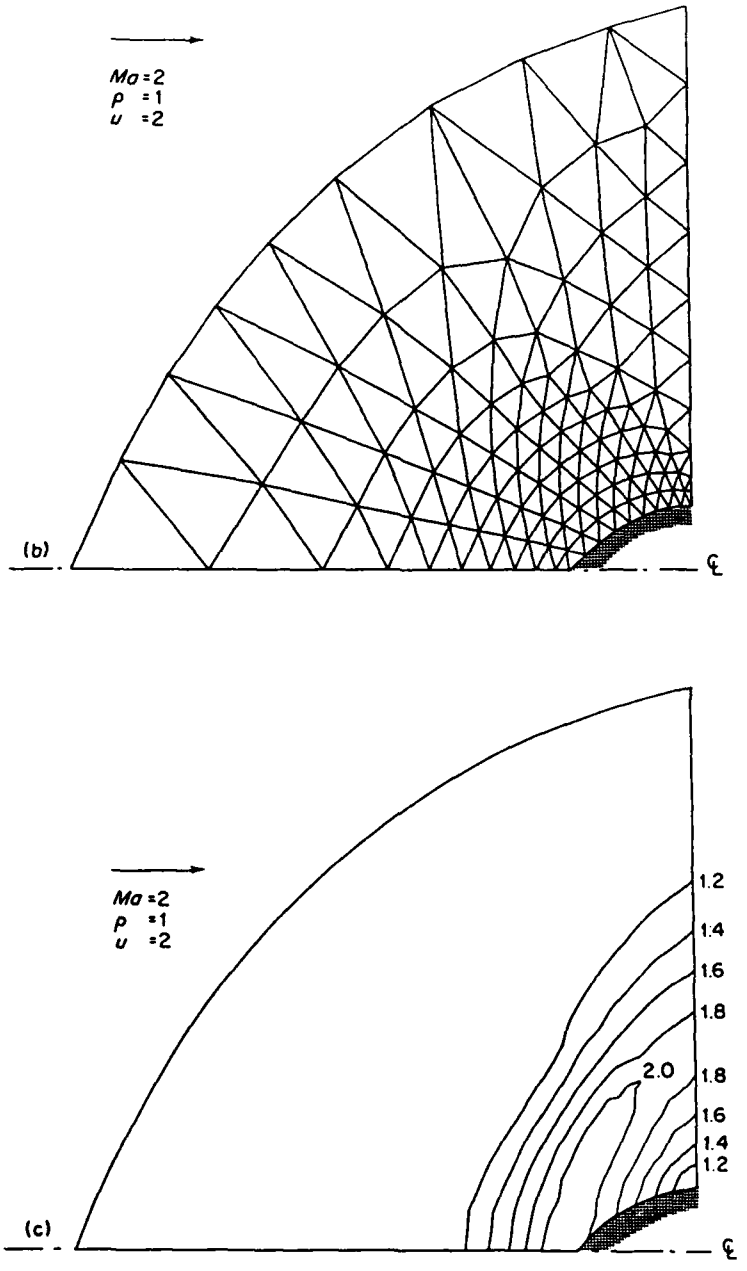


Figure 15.10 Supersonic flow past a nose cone: (a) original mesh, (b) mesh after 250 global steps (also mesh after 200 global steps), and (c) density after 250 global steps

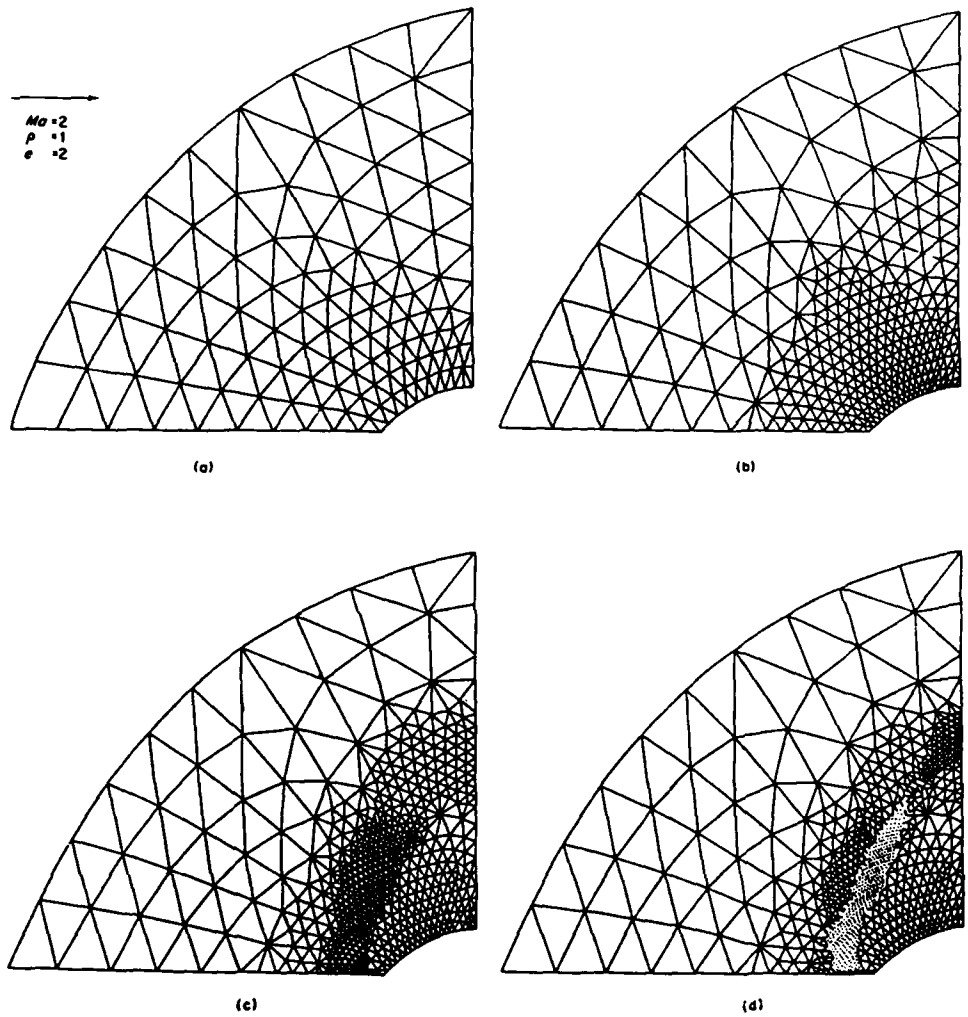


Figure 15.11 Grids produced after (a) 100 steps, (b) 200 steps, (c) 300 steps, and (d) 320 steps

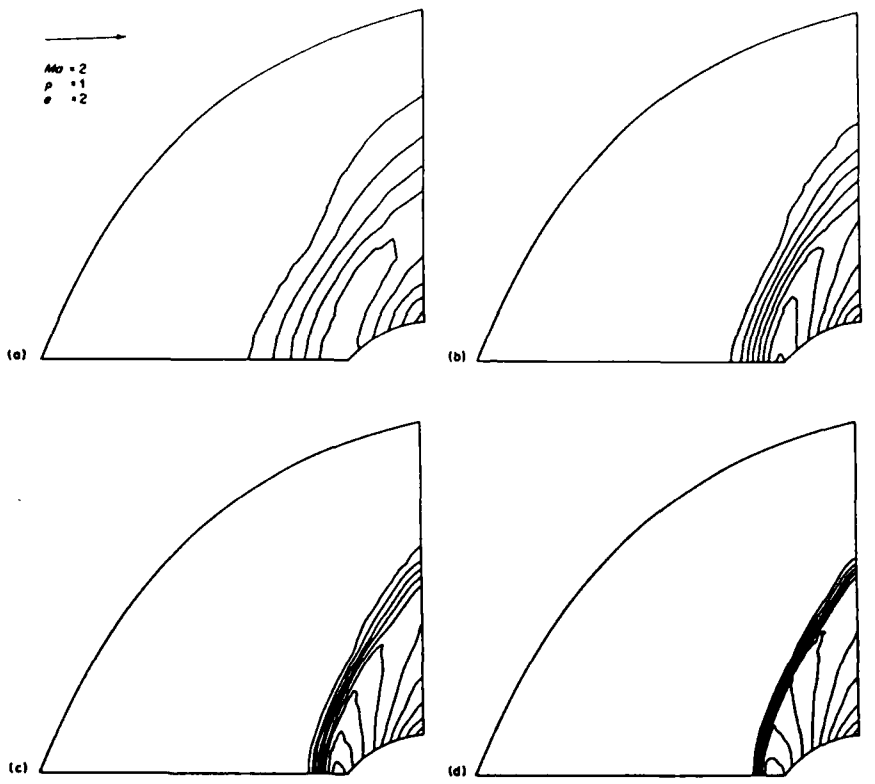


Figure 15.12 Density after (a) 100 steps, (b) 200 steps, (c) 300 steps, and (d) 320 steps

15.7 CONCLUSIONS

An adaptive grid refinement procedure for producing steady-state solutions of the compressible Euler equations has been described. It can be seen from the numerical examples presented above that adaptive mesh refinement can greatly improve the quality of the solution with minimal extra costs. At present it is not yet clear which of the two strategies discussed is to be preferred. Obviously mesh movement has the limitation that only a certain degree of change in element size can be achieved by it—usually not more than 1:4—whereas no such limitation exists for mesh enrichment strategies. Therefore, if no sophisticated mesh generator is available, or if the analyst has no knowledge of the final solution, mesh enrichment will be preferable. On the other hand, for many problems the analyst has some knowledge about the final solution, and accordingly can construct a mesh that is near optimal. In this case mesh movement, due to its simplicity, would be the obvious choice.

A final remark that has to be made is that the adaptive refinement techniques presented here were conceived for steady-state problems and for transient problems in which the areas that are to be refined do not vary greatly in time (boundary layer separation, buffeting, etc.). For transient problems where the areas that need refinement traverse large portions of the domain under consideration (e.g. shock propagation) a modified strategy will be required.

ACKNOWLEDGEMENT

The authors would like to thank the Aerothermal Loads Branch of the NASA Langley Research Center for supporting this research under Grant No. NAGW-478 and especially A. R. Wieting and K. S. Bey for their continued interest and encouragement.

REFERENCES

1. J. Donea, 'A Taylor–Galerkin method for convective transport problems', *Int. J. Numer. Meth. Engng*, **20**, 101–120 (1984).
2. R. Löhner, K. Morgan, and O. C. Zienkiewicz, 'The solution of non-linear hyperbolic equation systems by the finite element method', *Int. J. Numer. Meth. Fluids*, **4**, 1048–1063 (1984).
3. O. C. Zienkiewicz, R. Löhner, and K. Morgan, 'High speed inviscid compressible flow by the finite element method', in *The mathematics of finite elements are applications V*, MAFELAP 1984, (Ed. J. R. Whiteman), Academic Press, pp. 1–25 (1985).
4. O. C. Zienkiewicz and K. Morgan, *Finite Elements and Approximation*, John Wiley and Sons, New York (1983).
5. P. J. Roache, *Computational Fluid Dynamics*, Hermosa, Albuquerque (1976).
6. J. Peraire, R. Löhner, and K. Morgan, 'The implementation of boundary conditions in the finite element solution of supersonic flow', *Int. J. Numer. Meth. Engng*, (to appear).
7. A. Lapidus, 'A detached shock calculation by second-order finite differences', *J. Comp. Phys.*, **2**, 154–177 (1967).
8. R. W. MacCormack and B. S. Baldwin, 'A numerical method for solving the Navier–Stokes equations with application to shock-boundary interactions', AIAA Paper 75-1 (1975).
9. R. Löhner, K. Morgan, and J. Peraire, 'A simple extension to multidimensional problems of the artificial viscosity due to Lapidus', *Comm. Appl. Numer. Meth.* **1**, 41–47 (1985).
10. J. T. Oden, 'Notes on grid optimisation and adaptive methods for finite element methods', TICOM Report (1983).
11. O. C. Zienkiewicz, R. Löhner, K. Morgan, and J. Peraire, 'High speed compressible flow and other advection dominated problems of fluid dynamics', Ch. 2, pp. 41–88, *Finite Elements in Fluids*, Vol. 6, (Eds. R. H. Gallegher, G. F. Carey, J. T. Oden and O. C. Zienkiewicz), John Wiley and Sons, Chichester (1985).
12. A. R. Diaz, N. Kikuchi, and J. E. Taylor, 'A method of grid optimisation for the finite element method', *Comp. Meth. Appl. Mech. Engng*, **41**, 29–45 (1983).
13. R. Löhner, K. Morgan, and O. C. Zienkiewicz, 'An adaptive finite element method for high speed compressible flows', *Proc. Ninth Int. Conf. Numer. Meth. Fluids*, Paris, Lecture notes in physics, **218**, 388–392, Springer (1985).

14. G. Strang and G. J. Fix, *An Analysis of the Finite Element Method*, Prentice-Hall, Englewood Cliffs (1973).
15. O. C. Zienkiewicz and R. Löhner, 'Accelerated "relaxation" or direct solution? Future prospects for the finite element methods', *Int. J. Numer. Meth. Engng*, **21**, 1–11 (1985).
16. R. Löhner, K. Morgan, and O. C. Zienkiewicz, 'An adaptive finite element procedure for compressible high speed flows', *Comp. Meth. Appl. Mech. Engng*, **51**, 441–465 (1985).
17. B. Palmerio, 'Self adaptive finite element method algorithms for the Euler equations', INRIA Report No. 338 (1984).
18. R. Löhner, K. Morgan, and O. C. Zienkiewicz, 'The use of domain splitting with an explicit hyperbolic solver', *Comp. Meth. Appl. Mech. Engng*, **45**, 313–329 (1984).

APPENDIX Z.

An Adaptive Refinement Procedure for Transient Problems Arising in CFD (First World Congress on Comp. Mech.)

AN ADAPTIVE REFINEMENT PROCEDURE FOR TRANSIENT PROBLEMS ARISING IN CFD

R. Löhner
Berkeley Research Associates, Springfield, VA 22150

J.P. Boris
Laboratory for Computational Physics
Naval Research Laboratory, Washington, D.C., 20375

Our efforts are directed towards the efficient solution of transient problems which require a high degree of spatio-temporal resolution. Typical practical problems which fall under this category are impact of shock waves on geometrically complicated structures, detonations, shock-shock interactions and shock reflections. We first devised a high resolution scheme for unstructured grids [1] which handles the geometrical complexity typically encountered when solving practical problems, and then to employed adaptive refinement in 2-D [2] in order to enhance the resolution in regions where this is needed. The extension of the adaptive refinement scheme to 3-D is at present under intensive development.

Any adaptive refinement scheme for transient problems: 1) must be fast (and therefore must lend itself to vectorization/parallelization), as the grid adaptation has to be performed many times, 2) should not be storage intensive, as the grid adaptation process becomes an integral part of any code, 3) should recover the original grid after the feature has passed, as the feature that has been refined may pass again (e.g. shock reflection).

In order to meet these requirements we decided to use classic refinement, allowing only one level of refinement/coarsening per mesh-change. In this way, only 6 refinement and 3 coarsening cases are possible in 2-D. These are depicted in figure 1. This very low number of cases greatly simplifies the vectorization of the algorithm.

As error indicators we employ a modified interpolation error estimate, which in 1-D for a regular grid takes the form:

$$E_i = \frac{|U_{i+1} - 2 \cdot U_i + U_{i-1}|}{|U_{i+1} - U_i| + |U_i - U_{i-1}| + \epsilon \left[|U_{i+1}| + 2 \cdot |U_i| + |U_{i-1}| \right]} \qquad (1)$$

This modified error indicator has the following properties: 1) by dividing the second derivative by the 'jumps' (gradients) the 'eating-up' effect in the presence of a very strong shock is avoided (i.e. only the value of the normalized H2-seminorm is of importance, not the magnitude of the H2-seminorm as such), 2) normalizing in this way also has the advantage that the error indicator becomes dimensionless, so that more than one 'key-variable' can be used without encountering dimensioning problems, 3) moreover, the modified error indicator is now bounded ($0 \leq E_i < 1$), so that preset tolerances can be employed (this is of particular importance for transient problems), 4) the terms following ϵ are added as a 'noise' filter in order not to refine 'wiggles' or 'ripples' which may appear due to loss of monotonicity. The value for ϵ thus depends on the algorithm chosen to solve the PDEs describing the physical process at hand. This error indicator can be generalized for multidimensional problems (see [2]).

An example: Weak shock hitting a double wedge: as a typical example we consider the simulation of a weak shock ($M_s = 1.29$) that collides with a double wedge. The solution at T=6.5 is shown in figure 2. The savings in CPU-time and storage as compared

to a uniformly refined grid were more than a factor of 10, and the CPU time spent in non-vectorizable loops was less than 1 % on a CRAY-XMP-12. These numbers show clearly that also for transient problems, adaptive refinement techniques offer considerable advantages.

We plan to present more results and a better descripton of the schemes employed at the conference.

References:

[1] R. Löhner, K. Morgan, M. Vahdati, J.P. Boris and D.L. Book - FEM-FCT: Combining High Resolution with Unstructured Grids; Submitted to J.Comp.Phys. (1986).

[2] R. Löhner - An Adaptive Finite Element Scheme for Transient Problems in CFD; Submitted to Comp.Meth.Appl.Mech.Eng. (1986).

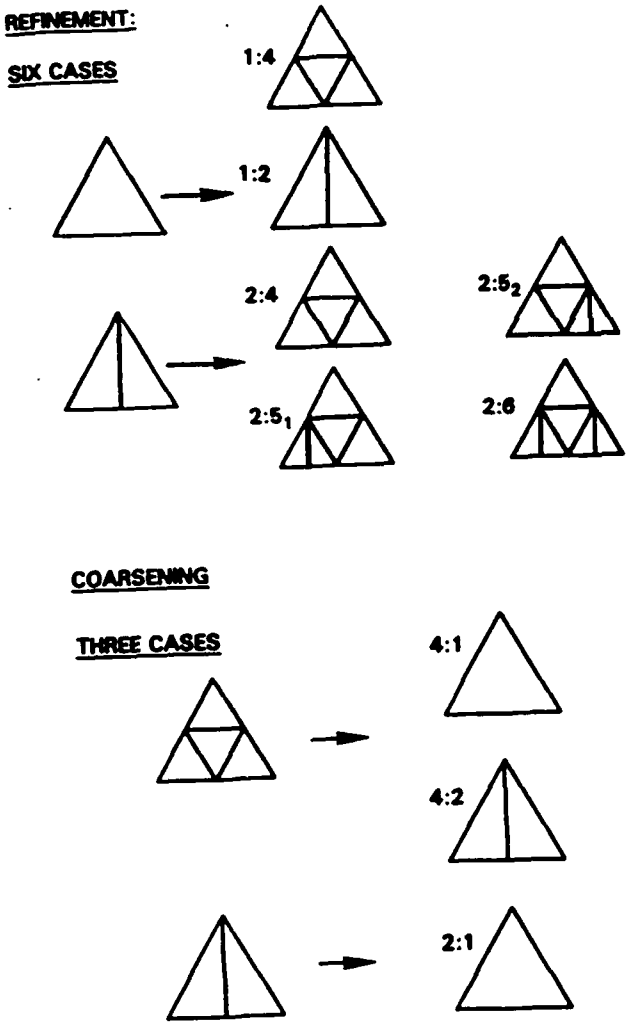


Figure 1

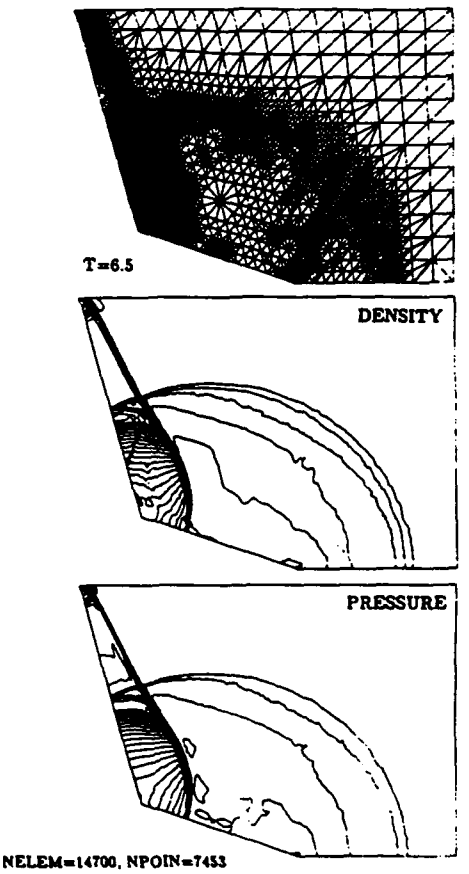


Figure 2