Copyright © 1997, American Institute of Aeronautics and Astronautics, Inc.

97-1771

13th AIAA Computational Fluid Dynam. Conf.
June 29 - July 2, 1997
Snowmass, Colorado

A97-32518

Computation of Flows with Moving Boundaries and Fluid-Structure Interactions

S. A. Bayyuk *

CFD Research Corporation
Huntsville, AL 35805

K. G. Powell †, and B. van Leer ‡

Department of Aerospace Engineering
The University of Michigan
Ann Arbor, MI 48109-2118

June 30, 1997

Abstract

An algorithm for the computation of steady and unsteady inviscid, compressible, two-dimensional flows is presented. Grid generation utilizes the quadtree spatial subdivision algorithm, allowing automatic meshing of boundaries of arbitrary geometry, and allowing h-adaptivity to geometric and solution features. Internal boundaries and interfaces may move, deform, coalesce, and disintegrate arbitrarily. The flow-solver is based on the MUSCL scheme. For moving boundaries, a novel feature of the algorithm is that the grid is stationary and boundaries are allowed to move across grid-lines. This is enabled by merging cells in the vicinity of boundaries to form composite cells that are topologically invariant during individual motion steps. This merging also eliminates the small-cell time-step constraint introduced by grid non-conformality. The motion of boundaries is tracked by a Lagrangian scheme that eliminates any diffusive distortion of the geometry. The motion of boundaries may be prescribed, or computed from aerodynamic, body, or contact forces in a coupled fashion. The capacity for exact preservation of the geometry makes the technique suitable for accurate solution of problems with aero-elasticity or large-scale rigid-body motion. Computations illustrating the capabilities and some potential applications are presented.

1 Introduction

Moving boundary problems arise in a large variety of physical situations. Examples include interaction of fluid interfaces, free-surface flows, solidification, and fluid-structure interactions. The characteristic feature is the presence of relative motion between different parts of a boundary or between different boundaries. This could additionally result in topologic changes such as coalescence, or formation of enclosures. Adequate simulation of such problems frequently demands

*Research Engineer
†Assistant Professor
‡Professor

Copyright ©1996 by AIAA. All rights reserved.

Copyright © 1997, American Institute of Aeronautics and Astronautics, Inc.

high accuracy in the computation of the location of boundaries and of the solution in their vicinity.

In regard to computing the location and motion of boundaries, the techniques developed so far have tended to either follow a predominantly Eulerian formulation, or a predominantly Lagrangian one. The advantage of the Eulerian formulation is that it allows interfaces to undergo arbitrary motion, deformation, and topologic transformation (such as merging, formation of voids, or disintegration). Its disadvantage is that interfaces are represented diffusely, and their shape cannot be well-preserved during advection. The Lagrangian representation (which includes fixed-topology grids [1] and chimera grids [2]) is computationally efficient and allows accurate preservation of the interface geometry, but cannot handle arbitrary motions, deformations, or topologic transformations, except if remeshing is used [5, 6, 7].

The work presented here attempts to combine the strengths of both approaches. A Lagrangian scheme that allows exact definition and preservation of interfaces is used for modeling the interface motion. However, instead of using a moving-grid that remains attached to the interface as with traditional Lagrangian approaches, a single stationary grid is used as with traditional Eulerian approaches. This requires interfaces to freely move across grid-lines, and this is enabled by use of a cell-merging procedure that combines cells in the vicinity of boundaries into composite cells that undergo a change in area but remain intersected along the same edges during a motion step. The use of a flexible data-structure for representing boundaries allows large-scale geometric and topologic transformations to be handled generally, and allows precise control over interface behavior during topologic transformations, offering a good opportunity to model the actual physical mechanisms involved.

2 Algorithm Description

2.1 Spatial Discretization

The spatial discretization algorithm is based on the Quadtree idea: an initial square cell containing all boundaries and the entire solution domain is subdivided into four equal cells. Each resulting cell is similarly subdivided until every boundary segment within each cell is sufficiently resolved. This is achieved when the segment within each cell is adequately approximated by the straight line joining the points of intersection of the segment with the cell. This algorithm can robustly generate grids for boundaries of arbitrary complexity. The discretization of the geometry is locally second-order accurate, and the grid quality is satisfactory for isotropic flows.

If boundaries move, their intersection pattern with the grid is re-evaluated after every discrete motion step and, depending on this, cells in the vicinity of boundaries may be coarsened or subdivided. In this manner, the geometry remains represented within the required order of truncation error at all times while the number of cells is kept to the minimum possible. The grid generation procedure imposes no restrictions on changes in the grid topology neither between nor along boundaries, in common with the traditional Eulerian approaches.

Use of a tree data-structure and use of cells that have simple shapes offers two main advantages over traditional unstructured-grid approaches; namely, reduced storage requirements, and more economical computation of geometric properties.

2.2 Representation of Boundaries

Each boundary in the grid is specified by a set of coordinate pairs and the continuities (either C^0, C^1, or C^2) thereat. These defining points are then interpolated by a Composite Parametric Cubic Spline. The representation of a boundary

Copyright © 1997, American Institute of Aeronautics and Astronautics, Inc.

on the Cartesian grid is then obtained by computing the intersections of all its spline segments with the Cartesian cells, and connecting the intersection points by straight line segments. For moving boundaries, a separate representation is needed for each of the positions required by the time-integration scheme; for example, for a two-step scheme, only two positions (the initial and final ones) are required.

Boundary motion is specified by specifying the motion of the defining points of the boundary, without any regard to the intersection points. The defining points may be moved by specifying individual velocity or displacement functions, or in the case of flow-coupled motion, by computing the force on each of the points and solving a kinetic equation. In the latter case, the force on each point is obtained by integrating the force along the two spline segments connected to it.

The points and spline segments are retained in a link-list data-structure. This data-structure readily allows the introduction or deletion of points, as may be necessary during large deformations. It also allows automatic disruption and reconnection of spline segments (either along the same boundary or between different boundaries) during a computation, as shown in figure 1. Since the spline interpolation satisfies coordinate invariance, rigid body motion preserves the original geometry to the truncation error of the representation of the geometry on the grid.

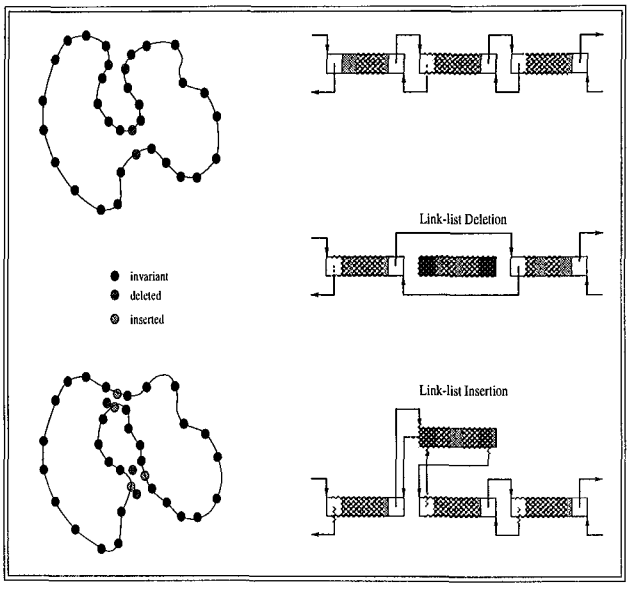


Figure 1: Link-list data-structure for splines, showing boundary disruption and reconnection.

2.3 Discretization and Solution of the Governing Equations

The governing equations for an arbitrary, moving, control-volume can be expressed as

$$\frac{d(\tilde{\mathbf{U}}\mathbf{V})}{dt} + \oint \tilde{\mathbf{F}}_\mathbf{n} \cdot \vec{\mathbf{ds}} = \vec{\mathbf{0}} \qquad (1)$$

where $\mathbf{V}$ is the volume enclosed by the control-surface, $\tilde{\mathbf{F}}_\mathbf{n}$ the flux tensor (formulated to account for control-surface velocity), $\tilde{\mathbf{U}}$ the state-vector, i.e. $(\rho, \rho u, \rho v, \rho E)^T$, and the other symbols have their usual denotations. The discrete analog employed has the form

$$\frac{\Delta(\tilde{\mathbf{U}}\mathbf{V})}{\Delta t} = -\sum_{faces} \bar{\tilde{\mathbf{F}}}_\mathbf{n} \cdot \vec{\mathbf{ds}} \qquad (2)$$

where $\bar{\tilde{\mathbf{F}}}_\mathbf{n}$ is an appropriate time-average of the flux tensor, and the other symbols are as above.

Interface fluxes are computed either by exact or approximate Riemann solvers, except on boundaries, where the following formula is used to evaluate the flux

$$\tilde{\mathbf{F}}_\mathbf{n} \cdot \vec{\mathbf{ds}} = \begin{pmatrix} 0 \\ p\Delta y \\ p\Delta x \\ pv_x\Delta y - pv_y\Delta x \end{pmatrix} \qquad (3)$$

where v_x and v_y are the local velocity components of the boundary, and Δx and Δy are the components of $\vec{\mathbf{ds}}$.

Copyright © 1997, American Institute of Aeronautics and Astronautics, Inc.

Second order temporal accuracy is accomplished by Predictor-Corrector time-stepping, a two-step scheme; viz.:

$$
\begin{aligned}
(\tilde{\mathbf{U}}\mathbf{V})^{(0)} &= (\tilde{\mathbf{U}}\mathbf{V})^{(n)}, \\
(\tilde{\mathbf{U}}\mathbf{V})^{(1)} &= (\tilde{\mathbf{U}}\mathbf{V})^{(0)} + \mathbf{\Delta t}[\mathbf{Res}((\tilde{\mathbf{U}}\mathbf{V})^{(0)})], \\
(\tilde{\mathbf{U}}\mathbf{V})^{(2)} &= (\tilde{\mathbf{U}}\mathbf{V})^{(0)} + \tfrac{\mathbf{\Delta t}}{2}[\mathbf{Res}((\tilde{\mathbf{U}}\mathbf{V})^{(0)}) \\
&\qquad + \mathbf{Res}((\tilde{\mathbf{U}}\mathbf{V})^{(1)})], \\
(\tilde{\mathbf{U}}\mathbf{V})^{(n+1)} &= (\tilde{\mathbf{U}}\mathbf{V})^{(2)}.
\end{aligned}
$$

Second-order spatial accuracy is accomplished by piece-wise linear reconstruction of the primitive variables, using either a Least-Squares or a Green-Gauss evaluation. At boundary segments, the value of the pressure, p, in equation (3) is extrapolated from the interior of the domain to the segment's mid-point, using the reconstructed gradients in the cell cut by the segment. Monotonicity is enforced by limiting the reconstruction gradients near discontinuities and extrema. The overall scheme satisfies the High-Resolution definition.

2.4 Solution-Adaptation and Cell-Merging

Spatial solution-adaptation is accomplished by subdivision or coarsening of cells. If the gradients (of the primitive variables) in a cell are too high, the cell is subdivided; if they are too low, the cell is coarsened. Refinement and coarsening is performed as needed after each time step to ensure that rapidly evolving geometric and solution features remain captured within the required resolution levels.

During a motion-step, the topologic status of cells in the vicinity of boundaries may switch from one of the three possible states (in, out, or intersected) to another. Satisfactory handling of the solution in such cells is the major obstacle to allowing motion of boundaries across grid-lines. The approach adopted in this work is to merge cells in the vicinity of boundaries to form polygonal composite cells that straddle the boundary and that remain topologically invariant during individual motion steps.

Figure 2 illustrates the merging process in the vicinity of a moving boundary. There, the shading indicates which cells are combined together into composite computational cells. The cells that remain un-intersected throughout the motion step are shown by dotted lines.

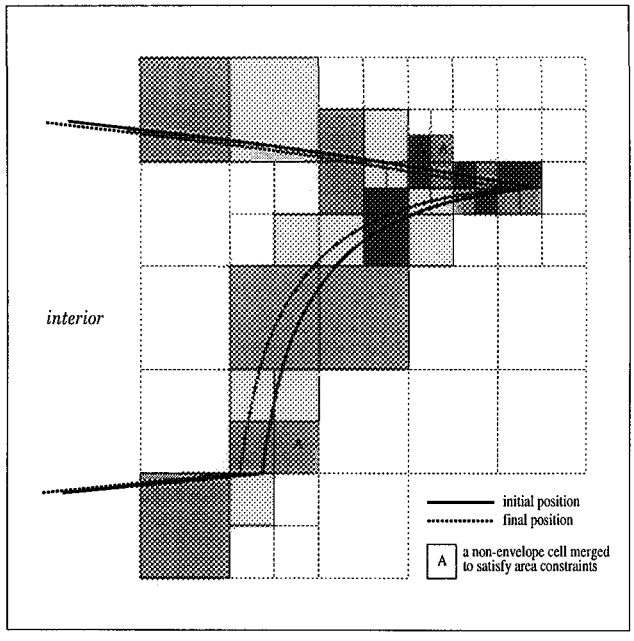


Figure 2: Cell merging in the motion envelope of a moving boundary.

The figure illustrates how although the individual components of composite cells may change their topologic status during a motion step, none of the composite cells do so. The figure also indicates how merging can be used to meet prespecified requirements on the minimum area of computational cells at the start and end of the motion step, eliminating any time-step restrictions imposed by small cells that arise from the non-boundary-conformality.

Within each composite cell, the fluid-dynamic problem is reduced to the motion of an impermeable boundary across a variable-area cell. The update procedure and boundary conditions for this problem are easily implemented within the Finite-Volume method, as outlined above.

Copyright © 1997, American Institute of Aeronautics and Astronautics, Inc.

After the discrete motion step is completed, the updated solution in each composite cell is projected back conservatively to its member cells according to their final areas and topologic status, and the solution gradient in the composite cell. This sequence of operations is repeated for every motion step.

By sufficiently refining the grid in the vicinity of any geometric feature, the merging procedure described above can always be carried out for any boundary geometry.

While merging must locally increase the truncation error, grid convergence studies of specific flow-fields, as in [3], provide strong evidence that the effect on the *order* of accuracy is small.

3 Demonstrative Results

3.1 Shock-Cylinder Interaction

An experimental investigation of the collision of planar shocks with a stationary cylinder was reported in [4]. Figure 3 shows the grid and density contour plots for the two times shown in [4] for the Mach 2.81-2.82 case. However, in the computation, an arbitrary Galilean transformation was imposed on the velocity of the initial flow-field and the cylinder. Throughout the flow-field, the computation agrees to within a few percent with the results of another computation with a stationary cylinder, and the computational results are in excellent agreement with the experimental ones.

3.2 Supersonic Inlet Flow

Figure 4 shows the Mach number contours and corresponding grids at three times during the prescribed deformation of a variable-geometry supersonic inlet in a free-stream with Mach number 2.54. The first figure in the sequence shows the initial inlet geometry while the last shows the terminal geometry; the geometries for intermediate times were obtained by linear interpolation between these two. The final figure shows the configuration with the fully-ingested shock pattern, with the flow successfully decelerated through a series of oblique shocks to the required Mach 1.45 at the exit.

3.3 Store Separation

Figure 5 shows a computation in which the motion of a store away from an airfoil is determined by the applied aerodynamic and (the much less significant) gravitational force. In order to increase the drag on the store, its trailing edge deforms gradually to a prescribed shape during the initial phase of the motion. Other than for this, the airfoil and the store are treated as rigid bodies, and motion starts from rest.

3.4 Projectile Impact

Figure 6 shows a projectile penetrating an elasto-plastic target. The projectile is accelerated from rest under the aerodynamic forces due to the pressurized gas in its rear cavity. Upon attaining a speed of Mach 3, the pressure and temperature of the gas in the projectile's front cavity instantaneously rise causing the projectile to fracture and its front and rear parts to separate and accelerate apart. The front part subsequently collides with, breaches, and enters a rectangular enclosure. The motion of the enclosure's walls is computed from the applied aerodynamic and impact forces using a simplified structural-dynamic model. The figure shows density contour and grid plots at three stages in the simulation. This problem demonstrates the automatic handling of boundaries that move, deform, and undergo topological changes. It also demonstrates the ability to simulate separation, impact, and fluid-structure interaction problems.

4 Computational Requirements

For stationary boundaries, 29 4-byte words of memory are required per leaf cell of the tree. The corresponding figure for moving boundaries is 39.

Copyright © 1997, American Institute of Aeronautics and Astronautics, Inc.

On a workstation rated at SPECfp92:150.6, current processing speeds are approximately 3,000 and 2,000 cells per second for stationary and moving boundaries respectively. For typical aerodynamic applications, roughly half the total computational effort is consumed in computation of fluxes and reconstruction of gradients.

5. Concluding Remarks

The main feature of the approach adopted in this work is that a Lagrangian technique is used for tracking interfaces, while an Eulerian technique is used for computing the flowfield. This is enabled by using a stationary grid, while allowing boundaries to move across grid-lines. The major benefits of this approach include elimination of the need to re-evaluate geometric properties of moving or deforming cells, and simplification of the enforcement of Geometric Conservation Laws [8]. The major drawback of this approach is the need for additional geometric computations to determine composite cells. In terms of capabilities, the algorithm presented here differs from those that have appeared in the literature in its truly automatic handling of arbitrary geometries, motions, and changes in boundary topology.

The technique and capabilities described above have been demonstrated for 2-D unsteady flows and for 3-D space marching applications and only for interfaces between ideal gases, or between ideal gases and solids (which may be rigid, elastic, plastic, or elasto-plastic). Although the emphasis in this paper was on moving-boundary applications, the algorithm is applicable to unsteady problems with stationary boundaries, as well as to steady-state problems.

Acknowledgments

This work was funded in part by a grant, monitored by Dr. August Verhoff, from McDonnell Aircraft Company. The first author gratefully acknowledges the support and encouragement received from Drs. Z.J. Wang, A. Krishnan, A. Przekwas, and A. Singhal of CFD Research Corporation.

References

[1] J. Batina, "Unsteady euler algorithm with unstructured dynamic mesh for complex aircraft aeroelastic analysis," AIAA Paper 89-1189., 1989.

[2] J. Benek, P. Buning, and J. Steger, "A 3-d chimera grid-embedding technique," AIAA Paper 85-1523-CP, 1985.

[3] W. Coirier, K. Powell, and M. Berger, "An accuracy assessment of cartesian mesh approaches for the euler equations," *Journal of Computational Physics*, vol. 117, pp. 121–131, 1995.

[4] R. Bryson and K. Gross, "Diffraction of strong shocks by cones, cylinders, and spheres," *Journal of Fluid Mechanics*, Vol. 10, Part 1, pp. 1–16, 1961.

[5] R. Lohner, "Adaptive remeshing for transient problems," *Computational Methods in Applied Mechanical Engineering*, vol. 75, pp. 195–214, 1989.

[6] E. Probert, O. Hassan, K. Morgan, and J. Peraire, "An adaptive finite element method for transient compressible flows with moving boundaries," *International Journal for Nummerical Methods in Engineering*, vol. 32, pp. 751–765, 1991.

[7] M. Reggio, J.Y. Trepanier, H. Zang, and R. Camarero, "Numerical simulation of the gas flow in a circuit breaker," *International Journal for Numerical Methods in Engineering*, vol. 34, pp. 607–618, 1992.

[8] C. Farhat, M. Lesoinne, and N. Maman, "Mixed explicit/implicit time-integration of coupled aeroelastic problems: three-field formulation, geometric conservation, and distributed solution," *International Journal of Numerical Methods in Fluids*, Vol. 21, pp. 807–835, 1995.

Copyright © 1997, American Institute of Aeronautics and Astronautics, Inc.

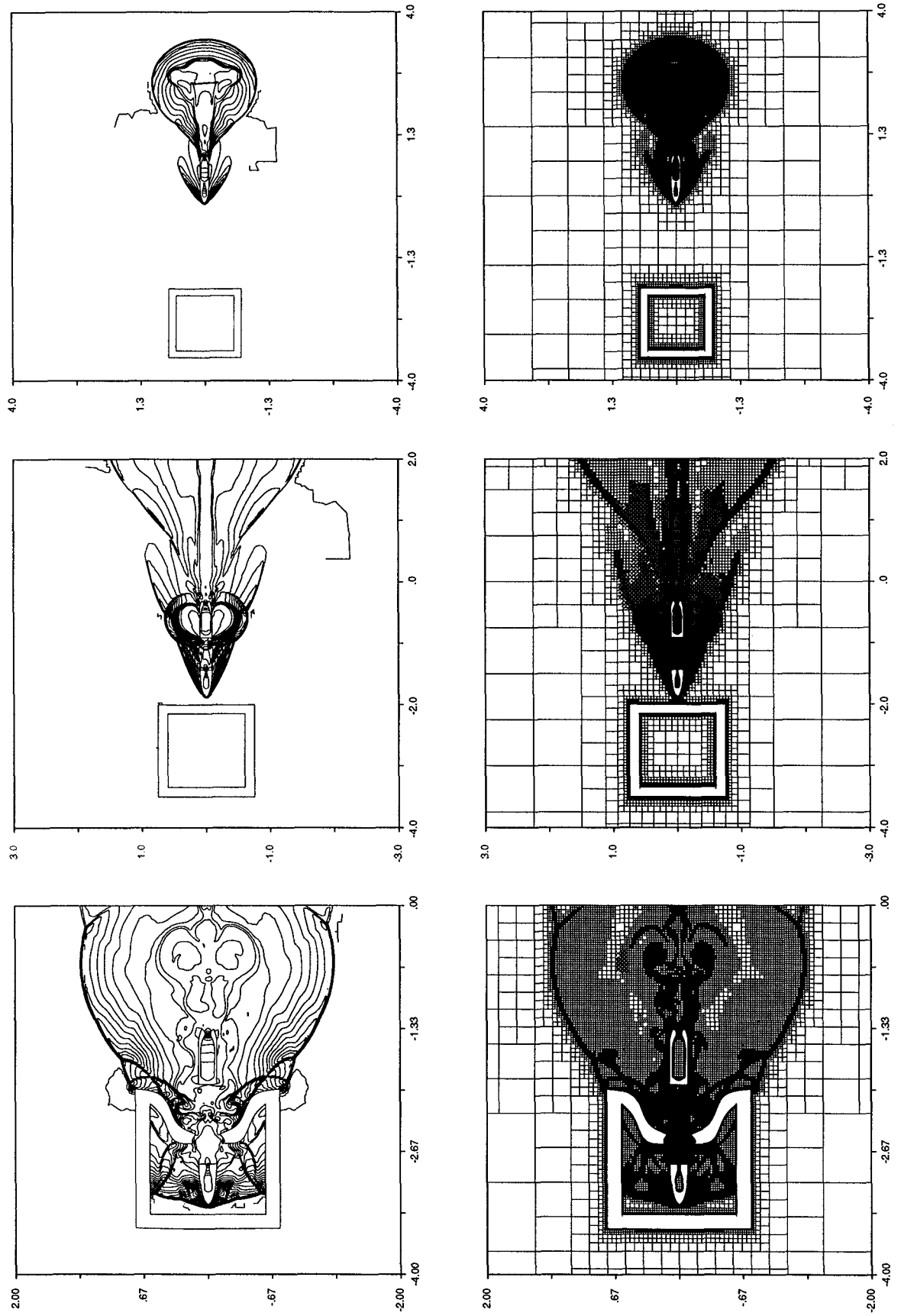
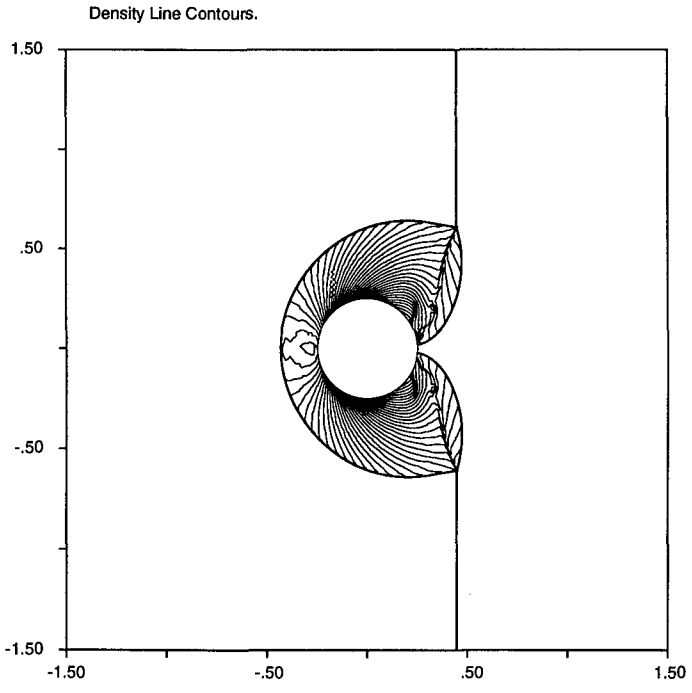
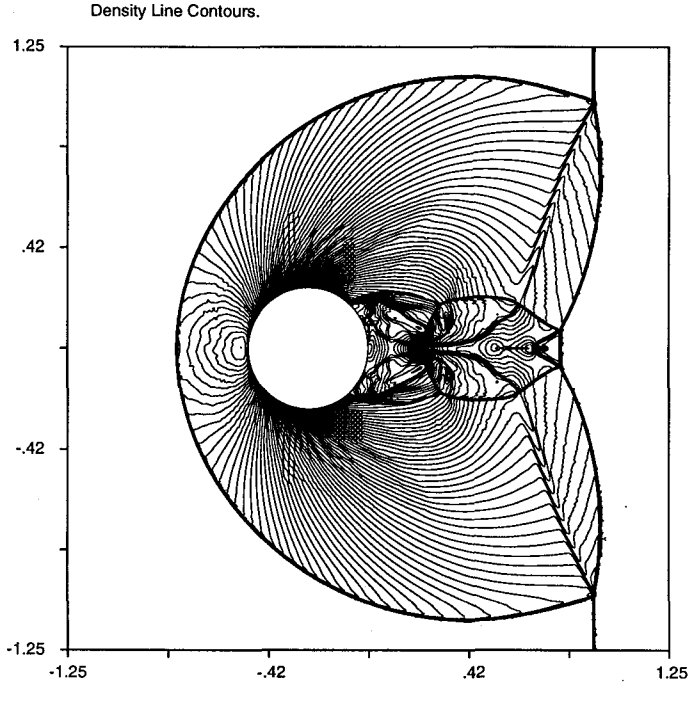


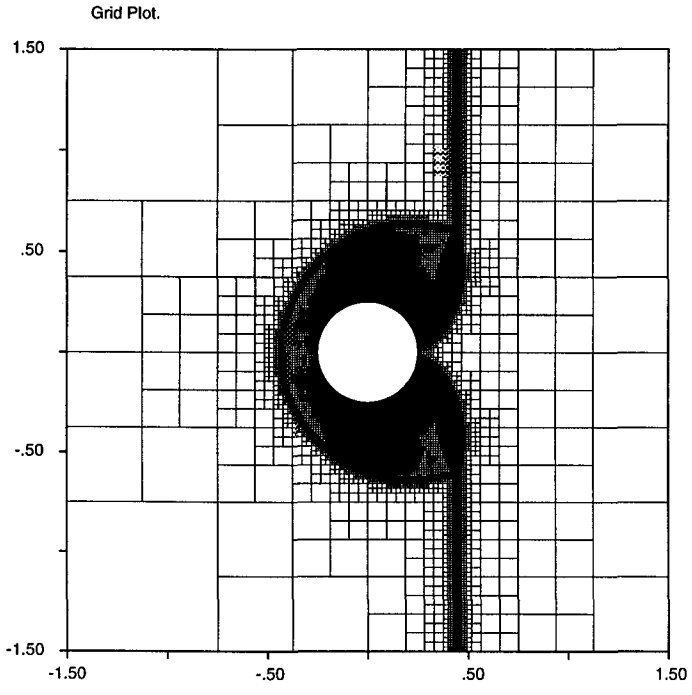
Figure 1: Density Contours and Grids for a Projectile Impact Problem.

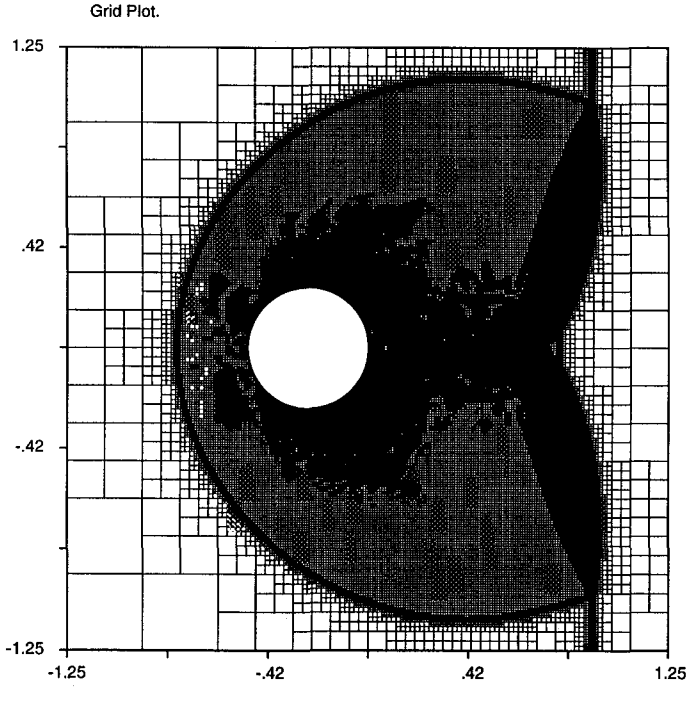
Copyright © 1997, American Institute of Aeronautics and Astronautics, Inc.

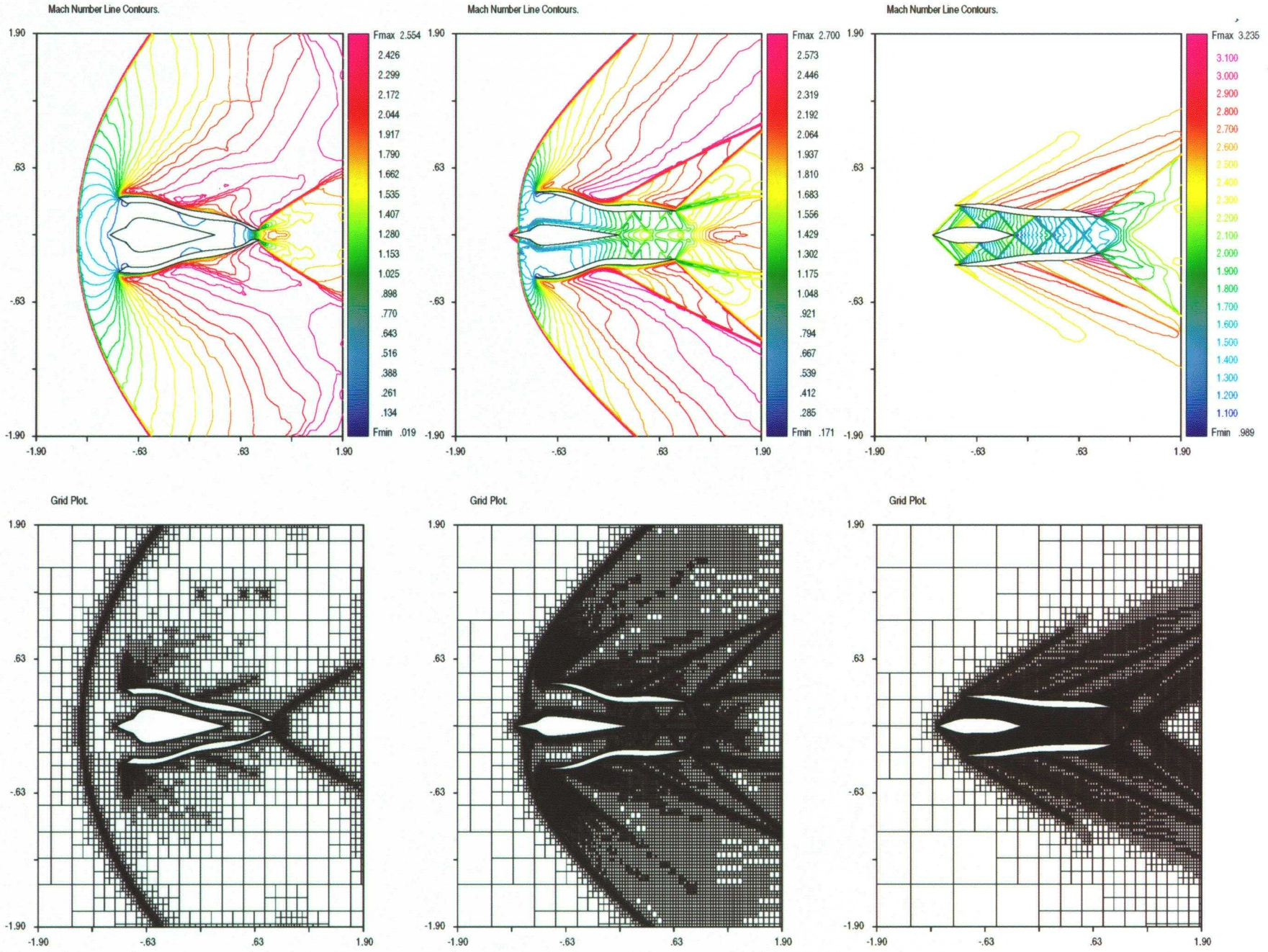




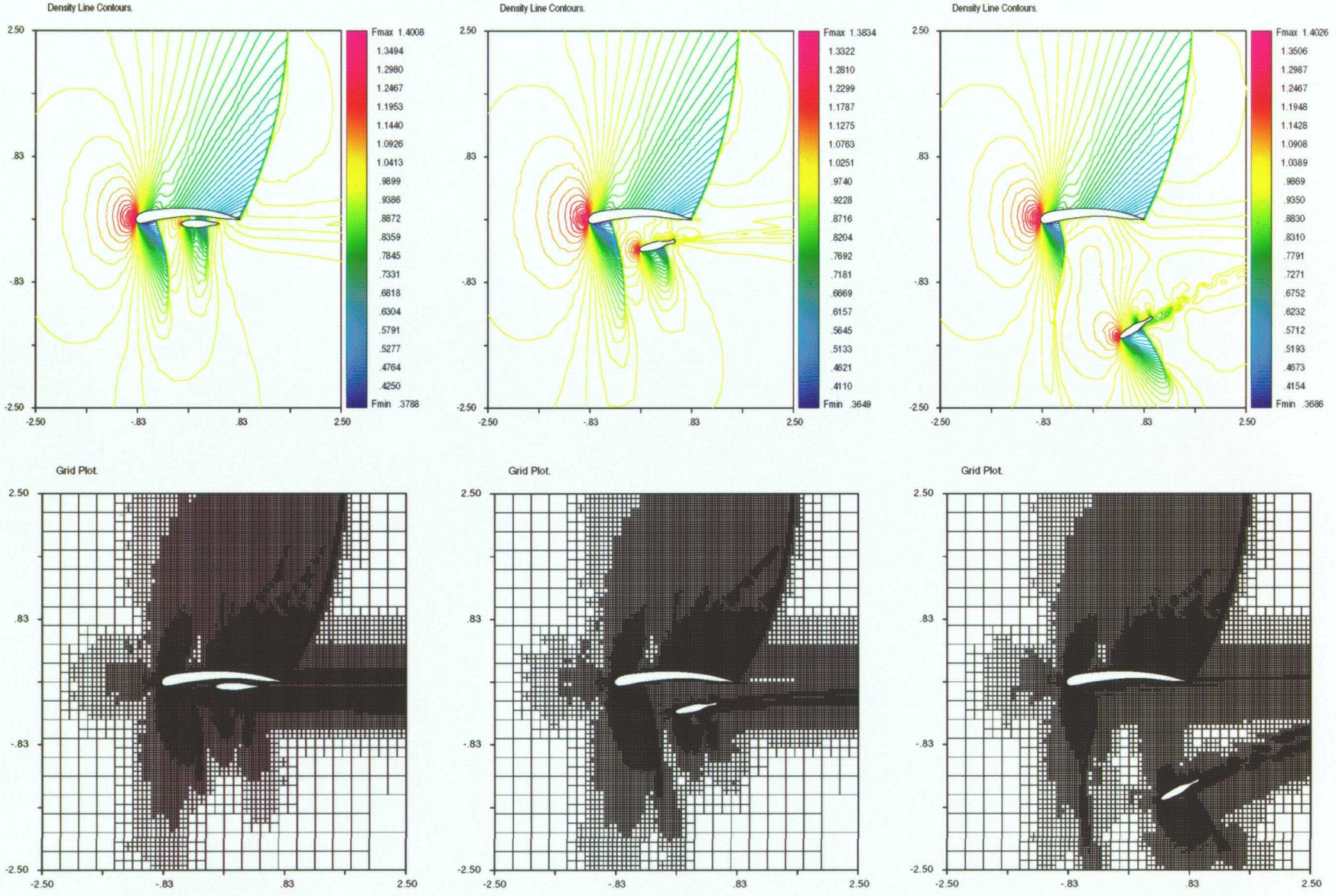
Copyright © 1997, American Institute of Aeronautics and Astronautics, Inc.







Copyright © 1997, American Institute of Aeronautics and Astronautics, Inc



Copyright © 1997, American Institute of Aeronautics and Astronautics, Inc