MULTIGRID SOLUTION OF THE EULER EQUATIONS ON UNSTRUCTURED AND ADAPTIVE MESHES

Dimitri Mavriplis

Institute for Computer Applications in Science and Engineering
NASA Langley Research Center
Hampton, VA

Antony Jameson

Princeton University
Princeton, NJ

ABSTRACT

A multigrid algorithm has been developed for solving the steady-state Euler equations in two dimensions on unstructured triangular meshes. The method assumes the various coarse and fine grids of the multigrid sequence to be independent of one another, thus decoupling the grid generation procedure from the multigrid algorithm. The transfer of variables between the various meshes employs a tree-search algorithm which rapidly identifies regions of overlap between coarse and fine grid cells. Finer meshes are obtained either by regenerating new globally refined meshes, or by adaptively refining the previous coarser mesh. For both cases, the observed convergence rates are comparable to those obtained with structured multigrid Euler solvers. The adaptively generated meshes are shown to produce solutions of higher accuracy with fewer mesh points.

1. INTRODUCTION

The ability to predict flow patterns and aerodynamic forces about complex configurations in the transonic regime is of primary importance to the aircraft designer. For slender bodies at small angles of attack, the flow remains attached, and the effect of viscosity is confined to relatively small boundary-layer and wake regions. Thus, an accurate description of the flow can be achieved using the inviscid Euler equations. These represent a system of non-linear partial differential equations in space and time.

Steady-state solutions of the Euler equations about simple geometries in two and three dimensions have become fairly widespread over the past few years. However, for more complex geometries, the generation of suitable meshes remains an obstacle. One approach which has recently received increased attention in the literature is the use of unstructured triangular or tetrahedral meshes in two or three

dimensions respectively [1,2]. The advantages of unstructured meshes are two-fold. Firstly, they provide a means for generating meshes about arbitrarily complex configurations. Secondly, they provide a natural setting for the use of adaptive meshing techniques, where local flow properties or error estimates are used to determine the distribution of mesh nodes. Because adaptive mesh refinement is a procedure which generally destroys the structure of an existing mesh, its implementation has often been constrained by the need to preserve the mesh structure. This has led to block structured composite meshes, where zonal regions are refined uniformly to preserve structure [3]. With unstructured meshes, these constraints are removed, and much more effective refinement strategies may be devised to develop "optimum" meshes.

On the other hand, unstructured mesh flow solvers are generally much less efficient than available structured mesh solvers. Unstructured mesh solvers suffer from inherent limitations, such as the need to store the mesh connectivity, and the use of gather-scatter operations on vector computers. However, it is also evident that the development of unstructured mesh flow solvers has not kept pace with advances in structured mesh solvers. Many of the ideas developed for structured mesh solvers, such as approximate factorization and nested multigrid methods, cannot be applied to unstructured meshes. They must either be modified or abandoned in favor of more general algorithms.

In this work, a multigrid algorithm for unstructured meshes is presented. The algorithm operates on a sequence of coarse and fine meshes and assumes no relation exists between the various meshes of the sequence. The meshes are generated by triangulating a given set of points in the flow-field using the Delaunay triangulation algorithm [4]. The distribution of mesh points is either determined by conformal mapping techniques, or by adaptive refinement of the previous coarser grid. The decision to adopt a multigrid strategy involving a sequence of unrelated meshes was motivated by the desire to optimize both the accuracy and the efficiency of the solver. This type of approach has previously been attempted by Lohner and Morgan [5] for elliptic problems. Other approaches [6] have suggested using a sequence of unstructured nested meshes, where finer meshes are constructed by successively subdividing the cells of a coarse unstructured grid in some manner. However, for a multigrid algorithm, the accuracy of the solution is determined uniquely by the finest grid, whereas the convergence rate is determined by the

coarsest grid of the sequence. The present approach provides the maximum flexibility for determining the configuration of the coarse and fine grids of the sequence, thus optimizing the efficiency and accuracy of the solver. Furthermore, when adaptive meshing techniques are employed, sequences of nested meshes can be obtained only by resorting to local mesh enrichment. However, much more sophisticated adaptive techniques are presently being advocated in the literature, mainly in the interest of obtaining directional refinements and smoothly varying meshes. These include a combination of mesh enrichment and moving meshes [7], and complete remeshing using coarse grid flow variables as weighting functions [8]. The present multigrid strategy can be used in conjunction with any of these techniques.

2. DISCRETIZATION OF THE GOVERNING EQUATIONS

The variables to be determined are the pressure, density, Cartesian velocity components, total energy and total enthalpy denoted by p, ρ, u, v, E, and H, respectively. Since for a perfect gas we have

$$E = \frac{P}{(\gamma-1)\rho} + \frac{u^2+v^2}{2}, \qquad H = E + \frac{P}{\rho}$$

where γ is the ratio of specific heats, we need only solve for the four variables ρ, ρu, ρv, and ρE.

These values are determined by solving the Euler equations, which in integral form read:

$$\frac{\partial}{\partial t}\iint_{\Omega} w\, dxdy + \int_{\partial\Omega} (fdy - gdx) = 0$$

where Ω is a fixed area with boundary $\partial\Omega$, x and y are Cartesian coordinates, and

$$w = \begin{bmatrix} \rho \\ \rho u \\ \rho v \\ \rho E \end{bmatrix} \quad f = \begin{bmatrix} \rho u \\ \rho u^2 + p \\ \rho uv \\ \rho uH \end{bmatrix} \quad g = \begin{bmatrix} \rho v \\ \rho vu \\ \rho v^2 + p \\ \rho vH \end{bmatrix}$$

The w variables are stored at the vertices of each triangle. The control volume for vertex i is defined as the union of all triangles having a vertex at i, as shown in Figure 1. The boundary flux integral in equation (1) is approximated by first calculating the values of the fluxes f and g at the nodes on the outer boundary of this control volume. These can then be integrated about the control volume boundary by assuming that on each edge, the value of the flux can be taken as the average of the two values on either end of the edge. This finite-volume formulation can be shown to be equivalent to a Galerkin

finite-element approximation, with a lumped mass matrix, and is second-order accurate in space [1].

Additional dissipative terms are needed to prevent odd-even point decoupling, and to prevent the formation of numerical oscillations near a shock. Artificial dissipation terms are constructed as a blend of second and fourth differences in the flow variables, where the differences are taken along each edge of the mesh. Thus, for example, the second differences of w at node i are calculated as

$$\nabla^2 w_i = \sum_{k=1}^{n} (w_i - w_k) \tag{2}$$

where n is the number of edges meeting at node i, and w_k represents the value of w at the other end of each edge (cf. Figure 1). Fourth differences are constructed by first computing and storing the second differences, as shown above, and then differencing these values again. This can be achieved by replacing the flow variables in equation (2) with the previously calculated second differences. The fourth difference terms form the background dissipation, which is applied throughout the flow-field. These terms can be shown to be third-order accurate, and thus the second order-accuracy of the scheme is preserved. The second differences represent stronger first order dissipation which is needed to prevent oscillations near shocks. Because this strong dissipation compromises the accuracy of the scheme, it is applied only in the vicinity of a shock, and is turned off elsewhere. This behavior is achieved by multiplying the second differences by an adaptive coefficient, constructed as a second difference in the pressure. This coefficient assumes a small value (order Δx^2) in regions of smooth flow, and becomes of order 1 near a shock. The present formulation of the dissipative terms is analogous to that used by Jameson for structured quadrilateral meshes [9], and provides a scheme which is second-order accurate everywhere, except in the vicinity of a shock where it becomes locally first-order accurate.

3. INTEGRATION TO A STEADY-STATE

Discretization of the Euler equations in space transforms the governing equations into a set of coupled ordinary-differential equations which must be integrated in time to obtain the steady-state solution. Thus, equation (1) becomes the set

$$S_i \frac{dw_i}{dt} + [Q(w_i) - D(w_i)] = 0, \qquad i=1,2,3,...$$

where S_i is the area of the control volume i, and is independent of time. The convective operator Q(w)

represents the discrete approximation to the flux integral in (1), and the dissipative operator D(w) represents the artificial dissipation terms. These equations are integrated in time using a fully explicit 5-stage hybrid time-stepping scheme, where the operator Q(w) is evaluated at each stage in the time step, and the operator D(w) is only evaluated in the first two stages, and then frozen at that value. Thus we advance in time as

$$
\begin{aligned}
w^{(0)} &= w^{n} \\
w^{(1)} &= w^{(0)} - \alpha_1 \frac{\Delta t}{S}\left[Q(w^{(0)}) - D(w^{(0)})\right] \\
w^{(2)} &= w^{(0)} - \alpha_2 \frac{\Delta t}{S}\left[Q(w^{(1)}) - D(w^{(1)})\right] \\
w^{(3)} &= w^{(0)} - \alpha_3 \frac{\Delta t}{S}\left[Q(w^{(2)}) - D(w^{(1)})\right] \\
w^{(4)} &= w^{(0)} - \alpha_4 \frac{\Delta t}{S}\left[Q(w^{(3)}) - D(w^{(1)})\right] \\
w^{(5)} &= w^{(0)} - \alpha_5 \frac{\Delta t}{S}\left[Q(w^{(4)}) - D(w^{(1)})\right] \\
w^{n+1} &= w^{(5)}
\end{aligned}
$$

where w^n and w^{n+1} are the values at the beginning and the end of the nth time step. The standard values of the coefficients are

$$\alpha_1 = 1/4 \qquad \alpha_2 = 1/6 \qquad \alpha_3 = 3/8 \qquad \alpha_4 = 1/2 \qquad \alpha_5 = 1$$

This scheme represents a particular case of a large class of hybrid time-stepping schemes, which has been specifically designed to produce strong damping characteristics of high frequency error modes. It is thus well suited to drive the multigrid algorithm.

Convergence to a steady-state is also accelerated by using the maximum permissible time step at each point in the flow-field, as determined by local stability analysis, by the use of enthalpy damping [9], and implicit residual averaging [10].

4. THE FULL MULTIGRID ALGORITHM

The basic idea of a multigrid strategy is to perform time steps on coarser meshes to calculate corrections to a solution on a finer mesh. The advantages of time-stepping on coarse meshes are two-fold: first, the permissible time step is much larger, since it is proportional to the mesh width, and secondly the work involved is much less because of the smaller number of grid points. In order to

combine, without compromise, the advantages of unstructured meshes with those of a multigrid strategy, it proves convenient to decouple the grid generation procedure from the multigrid algorithm. Thus, a multigrid method which operates on a sequence of unrelated meshes is needed. The key to such a strategy is the efficient transfer of flow variables back and forth between these meshes.

The full multigrid algorithm begins by computing the solution to the problem at hand on a coarse mesh. When convergence has been reached, a new finer mesh is generated. This can either be performed by globally regenerating a new mesh with a higher density of mesh points in all regions of the flow-field, or by adaptively refining the existing mesh. Next, the patterns for transferring the flow variables back and forth between these two meshes must be determined. Since the meshes are unnested, this is a non-trivial task. It is performed using a tree-search algorithm, which is described in detail in a following section. For any given flow calculation, this operation is only performed once, immediately after the generation of the new mesh. Transfer coefficients and transfer addresses are computed and stored, and used subsequently in the flow calculations. For each fine mesh point, three transfer addresses determine the three coarse grid nodes of the cell enclosing the fine grid node, to which the variables are to be transferred (see Figure 2), and the weighting is given by the corresponding transfer coefficients. The flow variables are then transferred to the new fine mesh, and these serve as the initial conditions for time stepping on this mesh. A multigrid saw-tooth cycle is then used to solve the equations on the new finer mesh, using the previous mesh as the background coarse grid. When convergence is obtained, a third finer mesh is generated, the transfer patterns are determined, and the flow variables are transferred to the new mesh. Time stepping resumes on this mesh using all three meshes as a sequence in the multigrid saw-tooth cycle. This procedure can be repeated as many times as necessary to obtain the desired accuracy, each time adding another mesh to the multigrid sequence. The full multigrid algorithm for a sequence of four meshes, beginning on the second mesh of the sequence, is depicted in Figure 3.

4.1. Multigrid Saw-Tooth Cycle

For a given sequence of meshes, the multigrid saw-tooth cycle is initiated by performing a single time step on the finest mesh of the sequence. The flow variables and residuals are then transferred to the next coarser grid. The equations on the coarse grids must be modified to ensure that they represent

the fine grid solution. If R' represents the transferred residuals and w' the transferred flow variables, a forcing function on the coarse grid may be defined as

$$P = R' - R(w').$$

Now, on the coarse grids, time stepping proceeds as

$$w^{(q)} = w^{(0)} - \alpha_q \frac{\Delta t}{S} (R(w^{(q-1)}) + P)$$

for the qth stage. In the first stage, $w^{(q-1)}$ reduces to the transferred flow variable w'. Thus, the calculated residuals on the coarse grid are canceled by the second term in the forcing function P, leaving only the R' term. This indicates that the coarse grid solution is driven by the fine grid residuals. This procedure is repeated on successively coarser grids, performing one time step on each grid level. When the coarsest grid is reached, the corrections are transferred back to the finer grids without any intermediate time stepping.

4.2. Grid Transfers

Flow variables, residuals, and corrections are transferred between coarse and fine grids in different manners. Flow variables at a coarse grid node P are taken as the linear interpolation of the corresponding values at nodes 1, 2, and 3, as shown in Figure 2, which are the vertices of the fine grid triangle enclosing P. These three nodes include the fine grid node which is closest to P, thus ensuring an accurate representation of the flow-field on the coarse grid. The fine grid residual R_a at "a" in Figure 2 is linearly distributed to the coarse grid nodes A, B, and C, which are the vertices of the coarse grid triangle enclosing "a". This linear distribution is accomplished by the use of shape functions which have the value 1 at one of the coarse grid triangle vertices, and vanish at the other two vertices. This implies that the sum of the residual contribution to A, B, and C equals the residual at "a", and the weighting is such that, if "a" and A coincide, then the contribution at A is equal to R_a, and the contributions at B and C vanish. This type of transfer is conservative. When transferring the corrections from the coarse grid back to the fine grid, a simple linear interpolation formula is used. Thus, the correction at the fine grid node "a" is taken as the linear interpolation of the three corrections at nodes A, B, and C which enclose "a" on the coarse grid.

4.3. Search Algorithm

The remaining difficulty lies in the determination of the nodes A, B, and C to be associated with each fine grid node "a". This is equivalent to the problem of locating the address of the coarse grid cell which encloses a particular fine grid node. A naive search over all the coarse grid cells would require $0(N^2)$ operations, where N is the number of grid points, and thus would be prohibitively expensive, requiring more time than the flow solution itself. Hence, an efficient search algorithm is needed. In this work, a tree-search algorithm has been adopted. It requires that information about the neighbors of each node or cell be stored for both the coarse and fine grids. It is initiated by providing an initial guess IC_1 for the coarse grid cell, and then testing IC_1 to see if it encloses the fine grid node NF. Since we are free to begin the search with any fine grid node and any coarse grid cell, we choose points whose locations are known (such as trailing edge values). If the test is negative, then the neighbors of IC_1 are tested. If these test also fail, then the neighbors of these neighbors are tested. This process is continued until, after n tries, the address IC_n of the cell enclosing NF is located. This entire procedure is repeated for every node of the fine grid. The next fine grid node NF_2 is thus chosen as a neighbor of NF, and the initial guess for the enclosing cell is taken as IC_n, the coarse grid cell which is now known to enclose the previous NF. In this manner, we are assured of a good initial guess, since IC_n and NF_2 must be located in the same region of the computational domain. This type of search can be achieved in 0(N logN) operations. In practice, of the order of 10 searches are required to locate an enclosing cell. Furthermore, this value is found to be insensitive to the size of the mesh. Because this operation is performed only once, just after the generation of the new mesh, the total amount of work involved is negligible when compared with the flow solution phase.

5. MESH GENERATION

Since the unstructured multigrid algorithm assumes the coarse and fine meshes of the multigrid sequence are independent of one another, any suitable mesh generation scheme may be employed. In this work, two approaches are illustrated, one where the global mesh point distribution is determined by conformal mapping techniques, and one where the mesh point distribution is determined by adaptive refinement techniques.

For both cases, the generation of unstructured triangular meshes is accomplished in three independent steps. First a distribution of mesh points in the flow-field is determined. These points are then joined together by line segments to form a set of triangular elements using the Delaunay triangulation algorithm. There exist many ways of triangulating a given set of points. The Delaunay algorithm represents a unique construction of this type. It also has the desirable property of minimizing the aspect ratios of the triangular cells. Further details on Delaunay triangulation can be found in [4,11]. The resulting mesh is then post-processed by a smoothing filter which slightly repositions the mesh points to ensure a distribution of smoothly varying elements. The new position of a mesh point is calculated as

$$x_i^{new} = x_i^{old} + \frac{\omega}{n}\sum_{k=1}^{n}(x_k - x_i)$$

with a similar expression for the y-coordinate. ω is a relaxation factor, and the sum is over all edges meeting at point i.

5.1. Mesh Point Distribution by Conformal Mapping

Conformal mapping techniques are used to generate global mesh point distributions about the multi-element airfoil configurations studied in this work. Each airfoil element of the configuration can be mapped to a circle by the application of a Karman-Trefftz transformation, followed by a shearing transformation. The resulting circle is fitted with a polar mesh. Upon mapping the circle back to the airfoil, a body-fitted regular quadrilateral O-mesh is obtained. When this procedure is repeated for each element of the configuration, a series of overlapping O-meshes is obtained. If the mesh cells are ignored, and the points which overlap with neighboring airfoil elements are omitted, a distribution of points in the flow-field is obtained. These points are then used as a basis for the triangulation procedure. Global refinement is achieved by prescribing twice as many points in the radial and circumferential directions of each mapped airfoil element, and remeshing the new point distribution.

5.2. Mesh Point Distribution by Adaptive Techniques

Adaptive mesh techniques offer the advantage of obtaining higher solution accuracy with fewer mesh points. This is achieved by concentrating the mesh points only in areas where large discretization

errors are observed. In principle, any type of adaptive meshing technique may be employed, since the present multigrid algorithm is decoupled from the mesh generation procedure. Presently, a simple refinement technique is employed, based on the extensive investigation of Danenhoffer [12], for structured quadrilateral grids. The undivided first difference of density is used as a refinement criterion, since the density varies with all important flow features. For each edge of the mesh, that is, any line segment of the mesh which joins two nodes, the difference of the density between the two end nodes is examined. If this difference is larger than some fraction (i.e. taken as 0.5 in this work) of the RMS average difference over all mesh edges, a new mesh point is created midway along that edge. For mesh edges approximating a curved boundary, such as the airfoil surfaces, the new mesh point will not coincide with the boundary, and must be projected back onto the airfoil surface. Once all new mesh points have been determined, they are combined with old mesh points and retriangulated. Splitting along edges in such a manner, rather than subdividing entire triangular cells, avoids the introduction of unnecessary mesh points, and offers the possibility of directional refinement.

In both of the above cases, the new refined mesh point distribution may contain points from the previous coarser mesh. However, the connectivity of these meshes is determined by the retriangulation procedure, and in general, the sequence of meshes will be unnested. Furthermore, the mesh points are displaced in the post-processing smoothing operation, and thus, none of the refined mesh points will coincide with the previous coarse mesh points. Thus, for both cases, the coarse and fine meshes of the sequence are independent from each other.

6. RESULTS

Results are presented for a two-element airfoil system in transonic flow. The basic configuration, which consists of a main airfoil fitted with a leading edge slat, has been the subject of an extensive study to determine the effectiveness of slats as a transonic maneuvering aid for fighter configurations [13]. The Mach number is 0.7, and the angle of attack is 2.8°. Figure 4 depicts the sequence of four globally refined meshes used in the full multigrid algorithm. The finest mesh contains a total of 5629 points. The computed pressure distribution on the finest mesh of this sequence is shown in Figure 5. Large suction peaks are evident near the leading edges of both airfoil elements. A very small supersonic

zone terminating with a shock is visible on the lower surface of the slat, near its leading edge. A strong shock at about mid-chord on the main airfoil is also observed. The convergence rate of the multigrid algorithm on this sequence of meshes is shown in Figure 6, as measured by the RMS average of the density residuals in the flow-field. On the finest grid, an average residual reduction of 0.897 per multigrid cycle is observed, reducing the residuals by 5 orders of magnitude in 100 cycles. The convergence rate is roughly the same on all meshes of the sequence, thus validating the effectiveness of the multigrid algorithm.

The same case has been computed with adaptively generated meshes. A sequence of six meshes is employed. The first two meshes are identical to the two coarsest meshes of the globally refined sequence in Figure 4. The next four meshes of the sequence, depicted in Figure 7, were obtained by successive adaptive refinements. The finest mesh of the sequence contains 4697 points, roughly 16% less than the finest mesh of Figure 4. Figure 8 shows the surface pressure distribution computed on this mesh. The accuracy of this solution is clearly superior to that of Figure 5. The definition of the shock on the main airfoil as well as the shock on the slat is much sharper than in the former case, due to the higher density of mesh points in these regions. The suction peaks on both airfoils, as well as the "hook" on the lower surface near the leading edge of the main airfoil are resolved in much better detail. The lower surface of the main airfoil contains fewer points than in the previous case. However, the accuracy of the solution is not affected, since no large flow gradients are present in this region. On the other hand, the resolution at the trailing edge of the main airfoil is somewhat lower than desired. Figure 9 shows the convergence rate for this sequence of meshes. An average residual reduction of 0.895 per multigrid cycle is achieved on the finest mesh, reducing the residuals by 5 orders of magnitude in 100 cycles. This rate is also seen to be roughly equivalent on all meshes of the sequence. The adaptive mesh technique is thus seen to produce more accurate solutions for less work. A solution with equivalent accuracy to the globally refined mesh solution depicted in Figure 4, can be obtained with roughly 1/3 the number of mesh points.

For both cases, the multigrid convergence rates are comparable to convergence rates obtained with a structured multigrid Euler solver [14]. A better assessment of the real efficiency of the present

multigrid algorithm is given in Figures 10 and 11, where the multigrid convergence rates for both of the above cases are plotted versus the number of work units. A work unit is defined as the amount of CPU time required to perform a single grid cycle on the finest mesh of the multigrid sequence. For comparison, the appropriate single grid convergence rates are also plotted on the same figures. The multigrid convergence histories include the time spent calculating transfer addresses and coefficients, performing inter-grid transfers of variables, and time stepping on coarse grids. They do not, however, include the mesh generation time. In all cases, the time spent calculating the transfer addresses and coefficients between any two meshes is of the order of 2 to 3 multigrid cycles on the newly generated mesh. In any particular multigrid saw-tooth cycle, the total fraction of time spent transferring variables back and forth between meshes is about 2%. The solution efficiency of the adaptive mesh sequence was found to be less than optimal, since time stepping occurs on all mesh levels, even in regions such as the far-field, where no mesh refinement takes place. In fact, little or no deterioration in the multigrid convergence rate was observed for this case, when time stepping on every second mesh of the sequence in the saw-tooth cycle was omitted.

7. CONCLUSION

The idea of uncoupling the multigrid algorithm from the grid generation procedure is an effective means for accelerating the convergence to a steady state of the Euler equations on arbitrary grids. The adaptive meshing technique produces significant increases in efficiency over global mesh refinement. Further work is required to determine more effective adaptive criteria, and to optimize the amount of work spent on each grid of adaptively generated multigrid sequences.

REFERENCES

1. Jameson, A., Baker, T. J., and Weatherill, N. P., "Calculation of Inviscid Transonic Flow over a Complete Aircraft", AIAA Paper 86-0103, January, 1986.
2. Angrand, F., Billey, V., Dervieux, A., Periaux, J., Pouletty, C., and Stoufflet, B., "2-D and 3-D Euler Flow Calculations with a Second Order Galerkin Finite Element Method", AIAA 18th Fluid Dynamics and Plasmodynamics and Lasers Conference, Cincinatti, July, 1985.
3. Benek, J. A., Buning, P. G., and Steger, J. L., " A 3-D Chimera Grid Embedding Technique", AIAA Paper 85-1523, 1985.
4. Weatherill, N. P., "The Generation of Unstructured Grids Using Dirichlet Tessalations", Princeton University Report MAE 1715, July, 1985.

5. Lohner, R., and Morgan, K., "An Unstructured Multigrid Method for Elliptic Problems", Int. J. Num. Meth. Eng. 24, pp 101-115, 1987.

6. Perez, E., "Finite Element and Multigrid Solution of the Two-Dimensional Euler Equations on a Non-Structured Mesh",INRIA Report No.442, September, 1985.

7. Palmerio, B., and Dervieux, A., "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.

8. Peraire, J., and Morgan, K., "A General Triangular Mesh Generator", to be published in Int. J. Num. Meth. Eng., 1987.

9. Jameson, A., Schmidt, W., and Turkel, E., "Numerical Solution of the Euler Equations by Finite Volume Methods Using Runge-Kutta Time Stepping Schemes", AIAA paper 81-1259, 1981.

10. Mavriplis, D. J., and Jameson, A., "Multigrid Solution of the Two-Dimensional Euler Equations on Unstructured Triangular Meshes", AIAA paper 87-0353, January, 1987.

11. Mavriplis, D. J., "Solution of the Two-Dimensional Euler Equations on Unstructured Triangular Meshes", Ph.D. Thesis, Princeton University, 1987.

12. Danenhoffer, J. F., and Baron, J. R., "Robust Grid Adaptation for Complex Transonic Flows", AIAA Paper 86-0495, 1986.

13. Volpe, G., "A Multigrid Method for Computing the Transonic Flow Over Two Closely-Coupled Airfoil Components", Paper presented at the 14th ICAS Congress, Toulouse, France, September, 1984.

14. Jameson, A., "Solution of the Euler Equations by a Multigrid Method", Applied Math. and Computation, Vol 13, 1983, pp. 327-356.

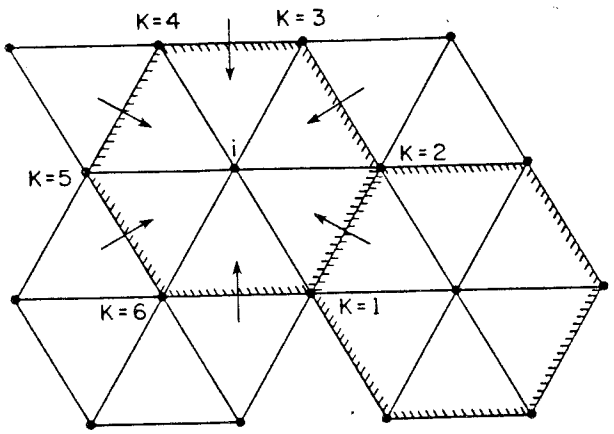

Figure 1
Control Volume for the Triangle Vertex Discretization Scheme

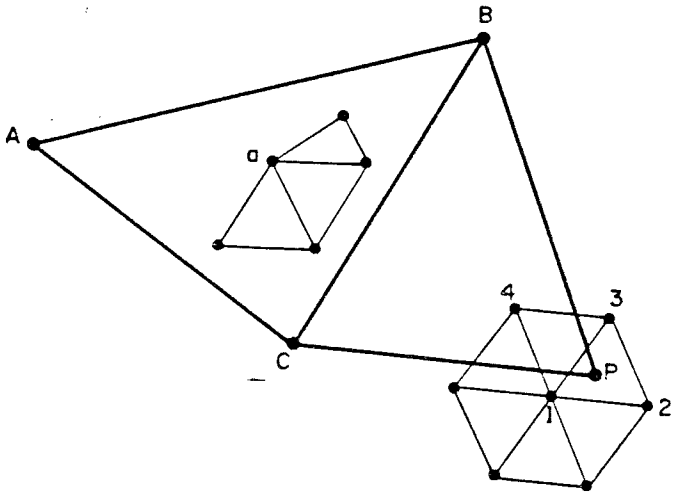

Figure 2
Grid Transfers for the Unstructured Multigrid Algorithm:
Residual at "a" is distributed to A, B, and C
Flow Variable at P is the Linear Interpolation
of Values at 1, 2, and 3.

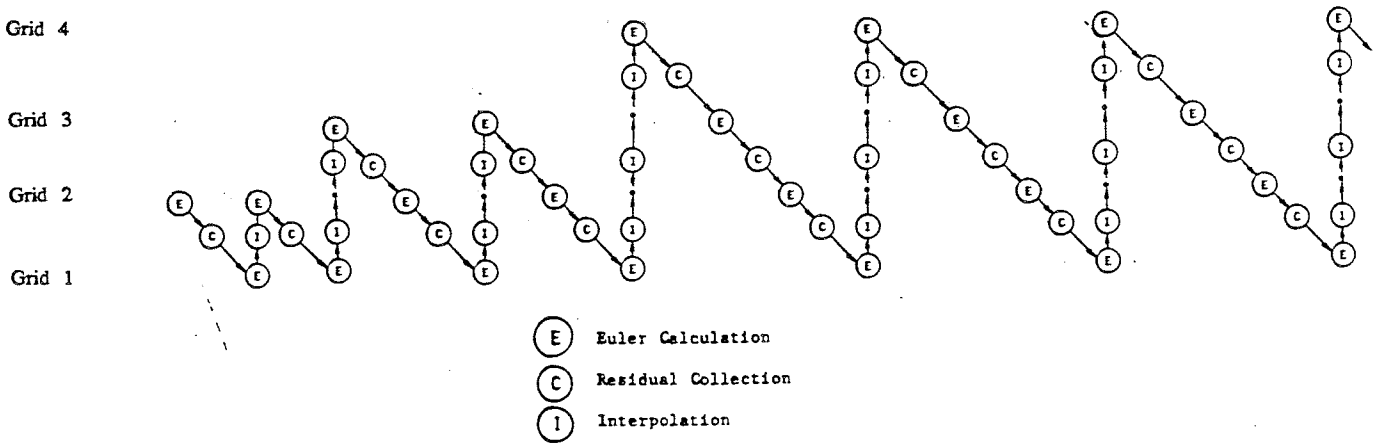

Figure 3
Full Multigrid Algorithm using the Saw-Tooth Cycle

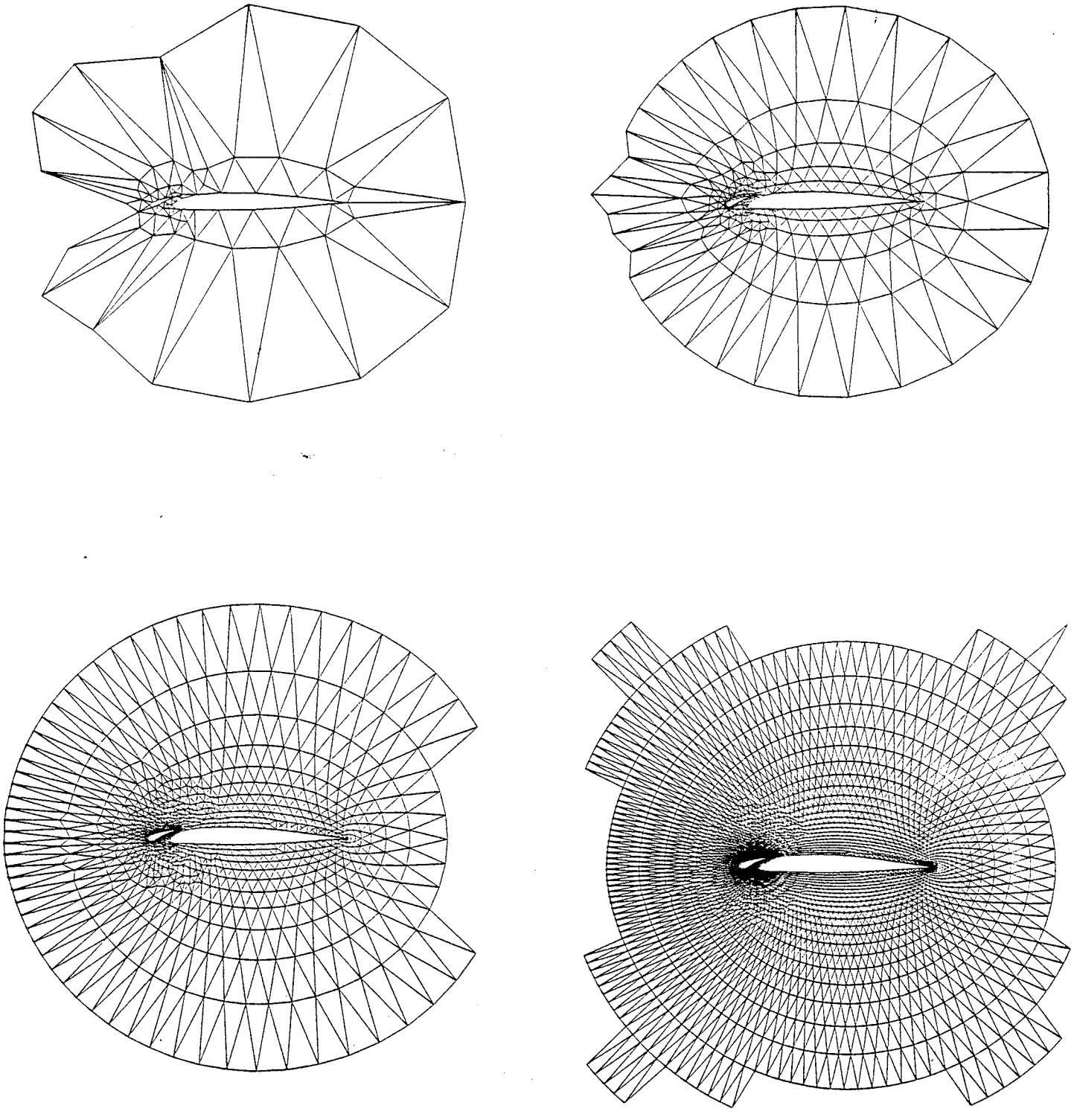

Figure 4
Sequence of Meshes Generated by Global Refinement for the
Unstructured Multigrid Algorithm
Mesh 1 : 114 Nodes
Mesh 2 : 382 Nodes
Mesh 3 : 1458 Nodes
Mesh 4 : 5629 Nodes

(Partial View of Meshes Only)

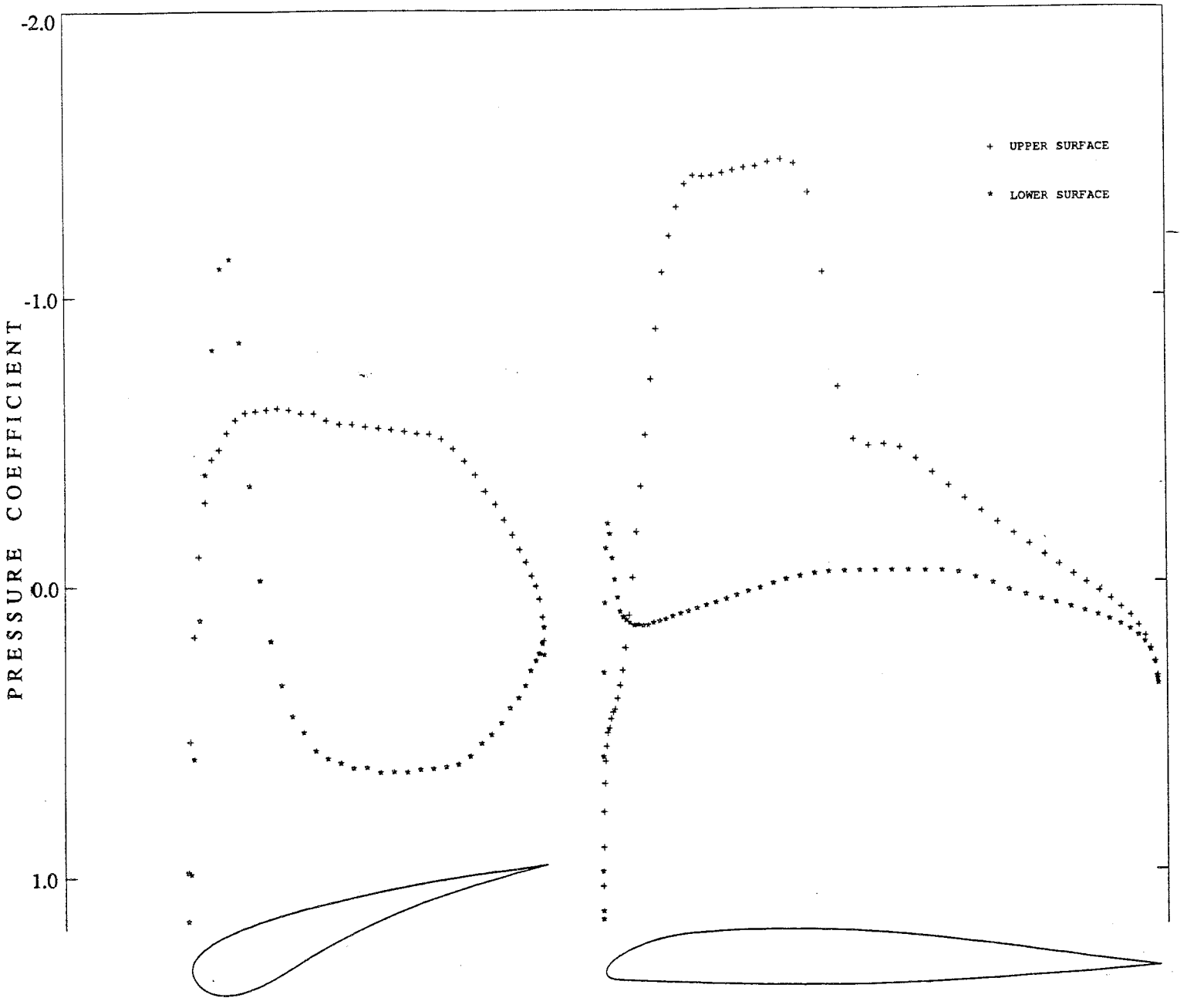

Figure 5
Surface Pressure Distribution on the Main Airfoil and the Leading Edge Slat
Calculated on the Finest Mesh of the Globally Refined Mesh Sequence.
Mach = 0.7 Incidence = 2.8^0

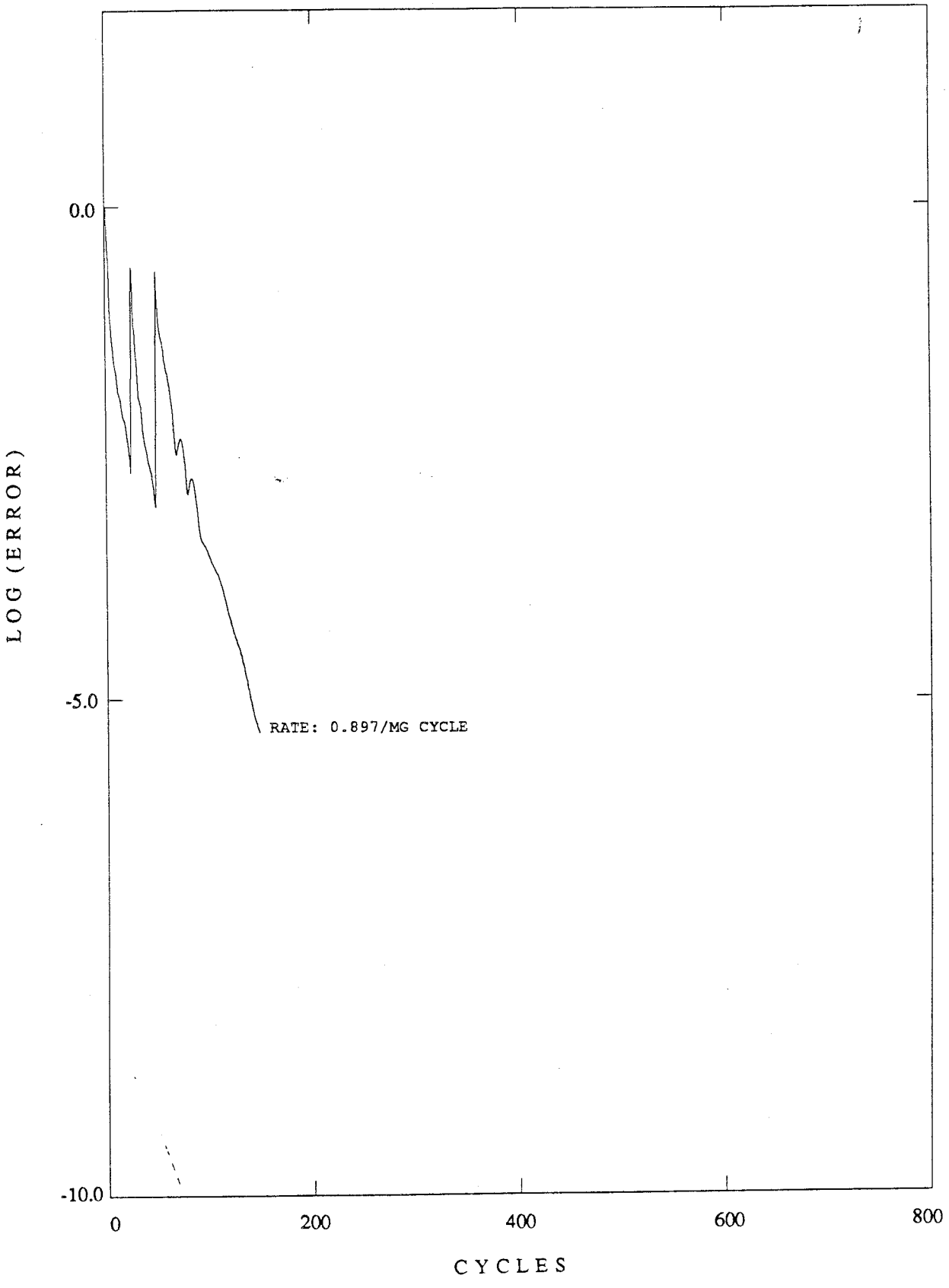

Figure 6
Convergence Rate as Measured by the RMS Average of the Density Residuals throughout the Flow-field versus the Number of Multigrid Cycles for the Globally Refined Mesh Sequence Beginning on the Second Mesh of the Sequence.

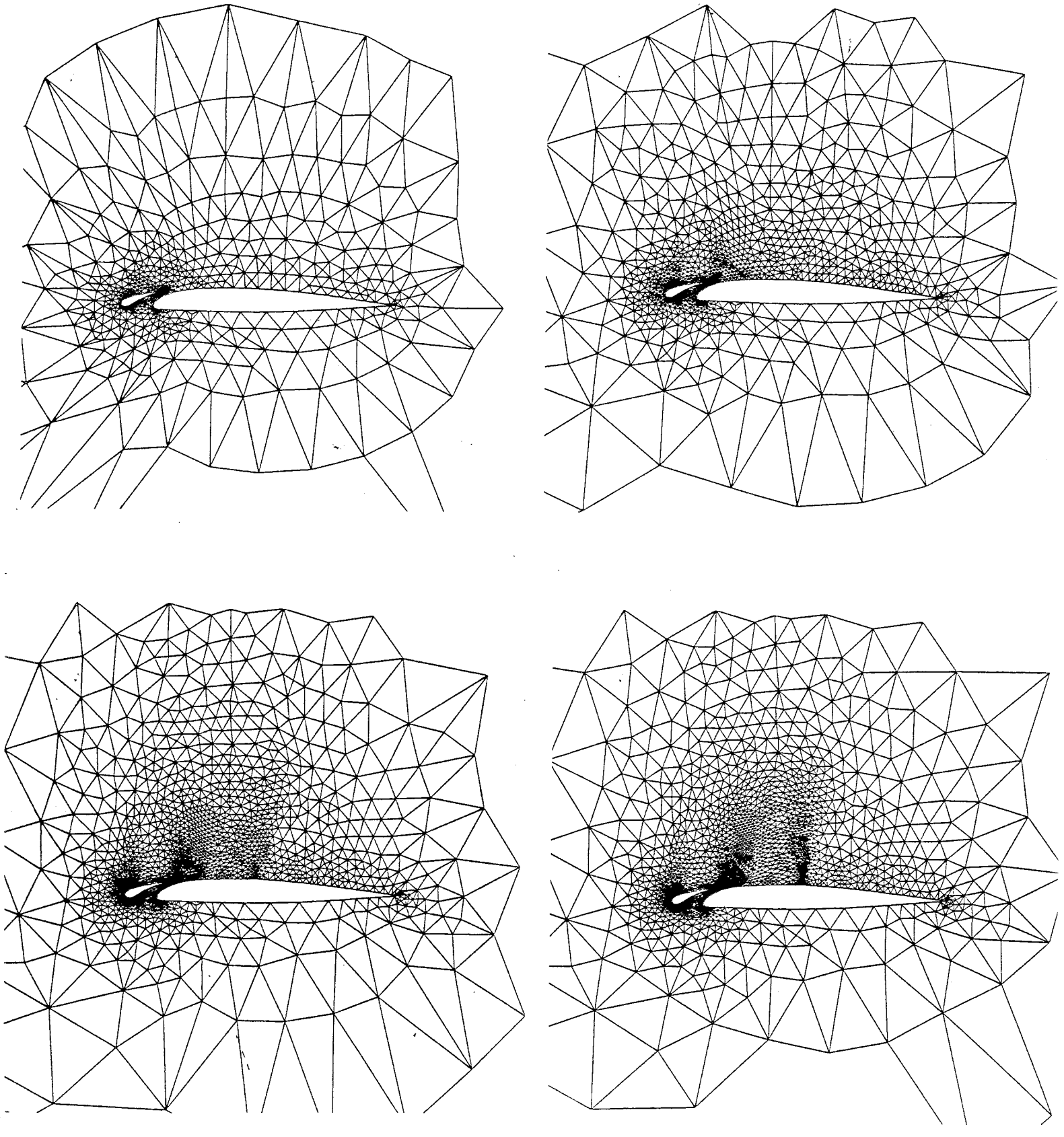

Figure 7
Sequence of 4 Adaptively Generated Meshes Used in the Multigrid Algorithm in Conjunction with the 2 First Meshes of Figure 4
Mesh 3 : 790 Nodes
Mesh 4 : 1631 Nodes
Mesh 5 : 3107 Nodes
Mesh 6 : 4697 Nodes
(Partial View of Meshes Only)

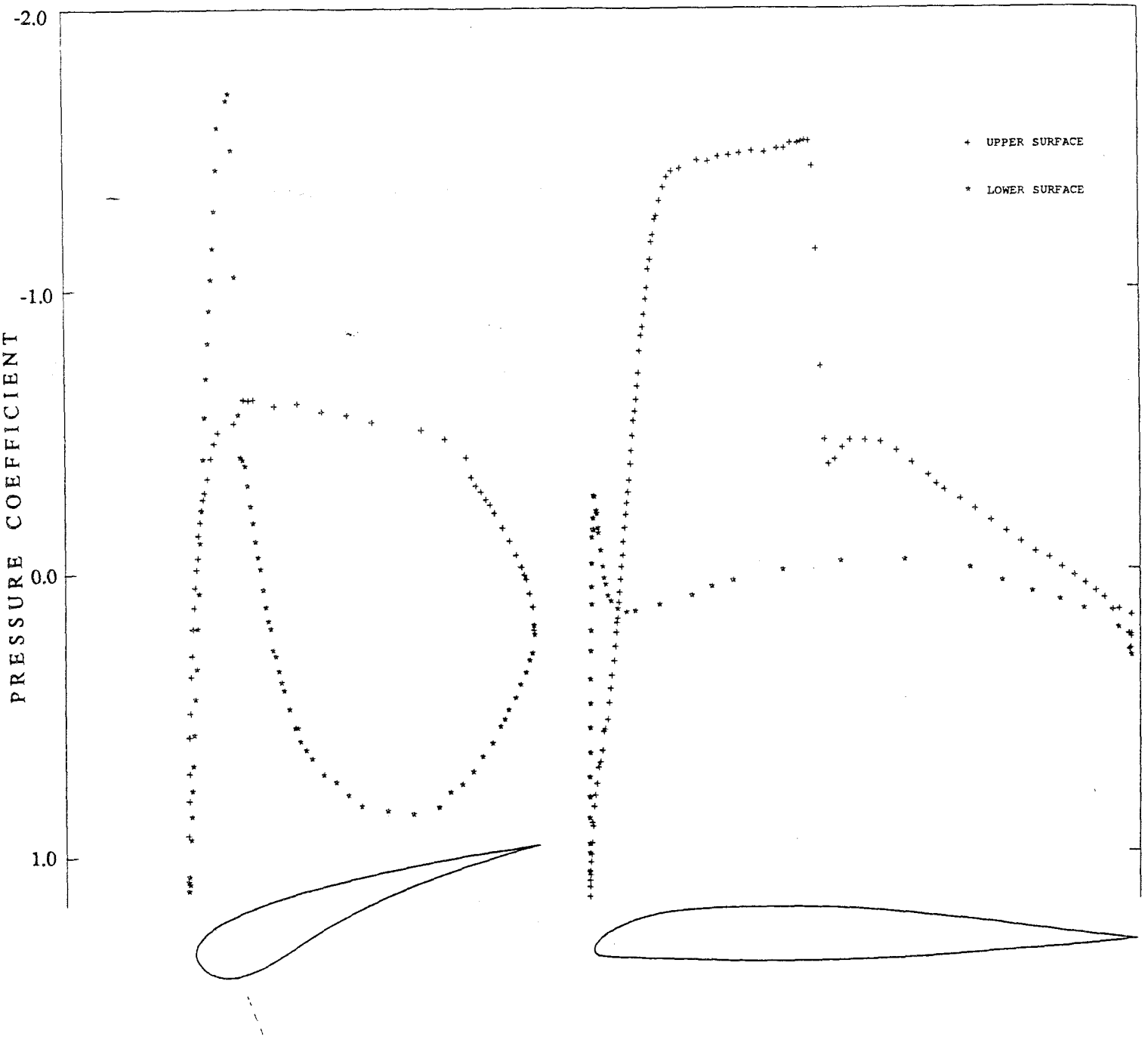

Figure 8
Surface Pressure Distribution on the Main Airfoil and the Leading Edge Slat
Calculated on the Finest Mesh of the Adaptively Refined Mesh Sequence.
Mach = 0.7 Incidence = 2.8^0

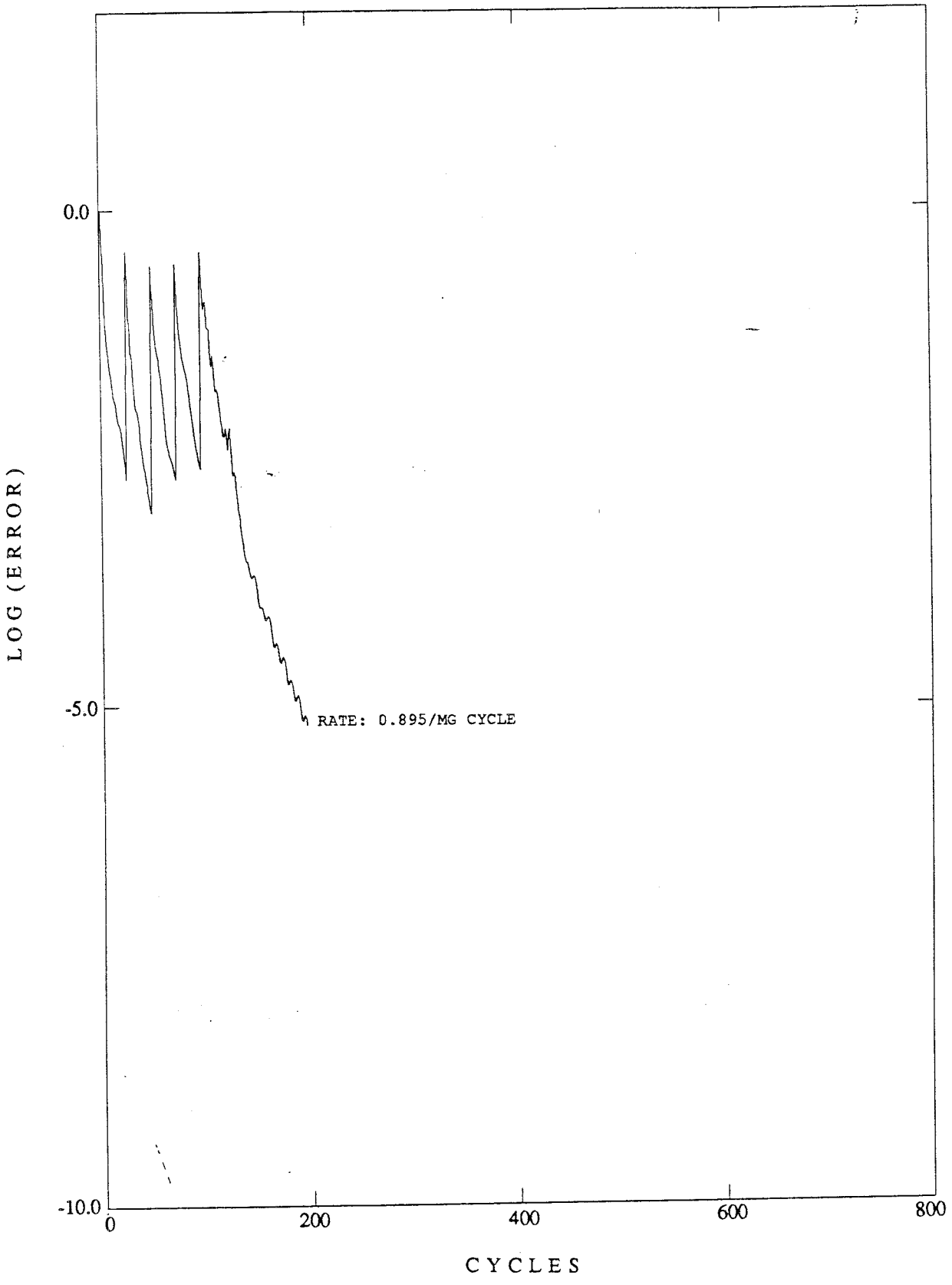

Figure 9
Convergence Rate as Measured by the RMS Average of the Density Residuals throughout the Flow-field versus the Number of Multigrid Cycles for the Adaptively Refined Mesh Sequence Beginning on the Second Mesh of the Sequence.

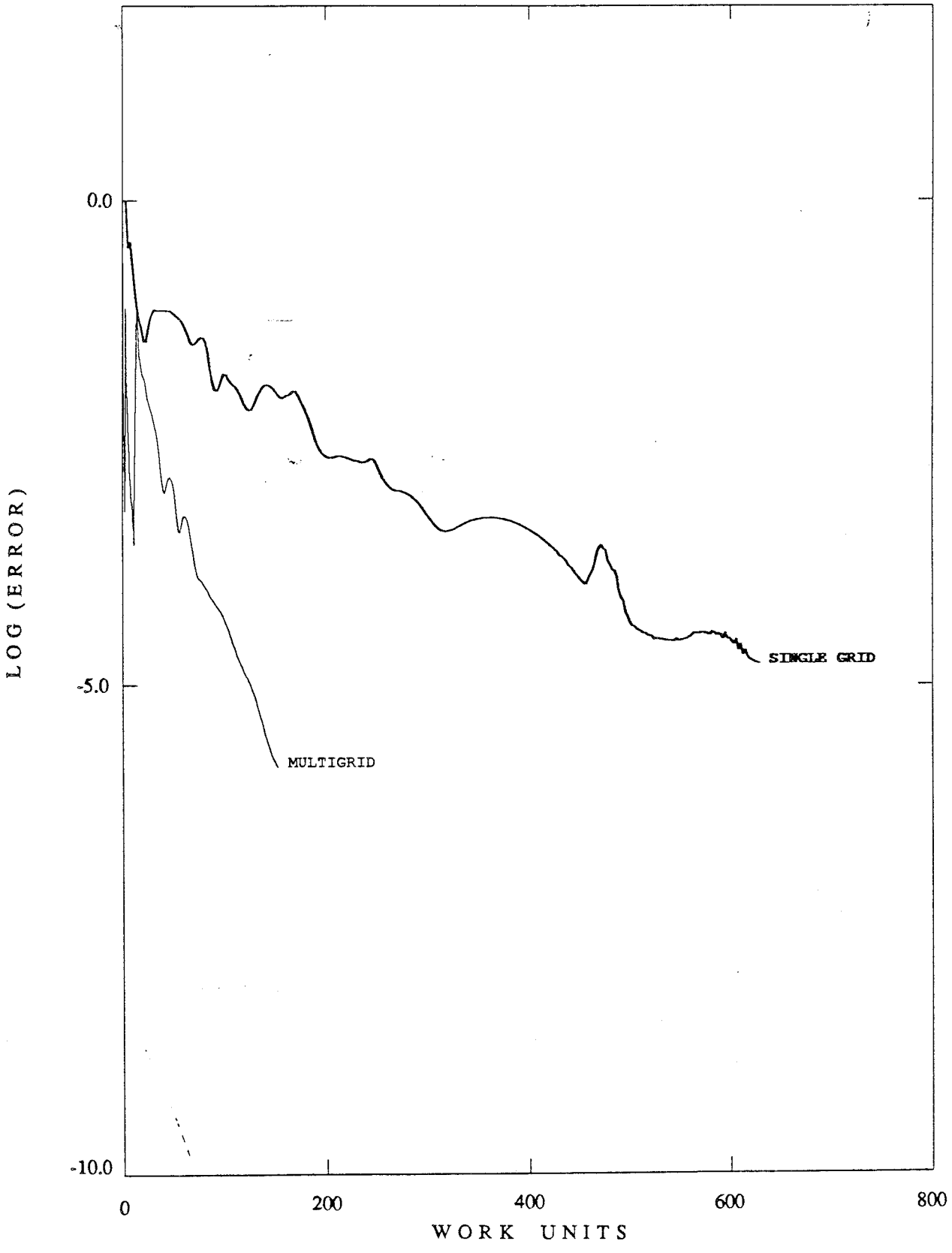

Figure 10
Convergence Rate as Measured by the Number of Work Units for the Full Multigrid Algorithm on the Globally Refined Mesh Sequence Beginning on the Second Mesh of the Sequence Compared with the Convergence Rate on the Single Finest Grid of the Sequence.

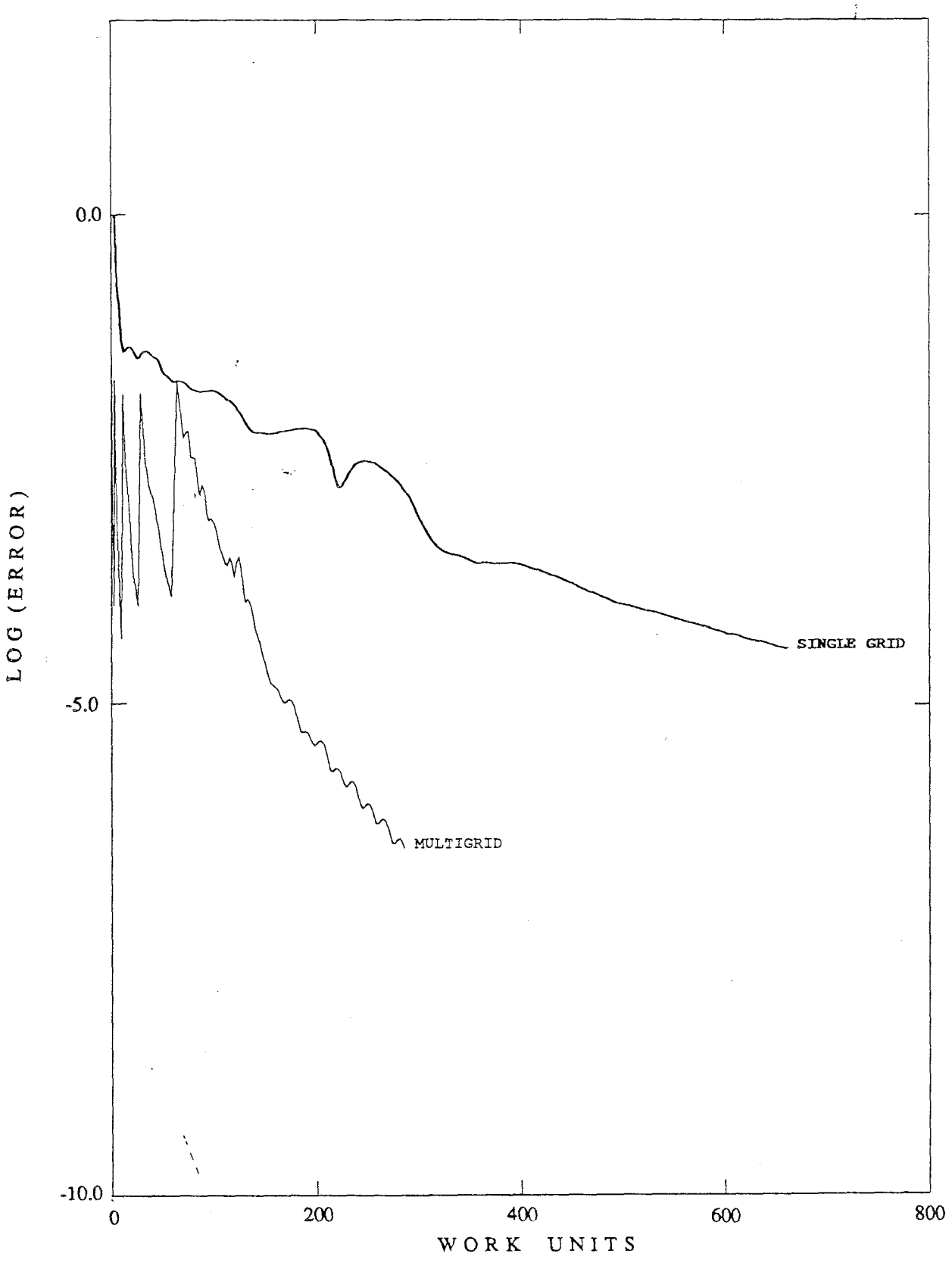

Figure 11
Convergence Rate as Measured by the Number of Work Units for the Full Multigrid Algorithm on the Adaptively Refined Mesh Sequence Beginning on the Second Mesh of the Sequence Compared with the Convergence Rate on the Single Finest Grid of the Sequence.