Three-Dimensional Adaptive Grid Generation on a Composite-Block Grid

Hyun Jin Kim*
Korean Air Force, Korea
and
Joe F. Thompson†
Mississippi State University, Mississippi State, Mississippi

The EAGLE three-dimensional composite-block grid code has been extended to an adaptive grid to be coupled with a PDE solver. Both the adaptive control function formulation and the variational formulation were evaluated, and the former was found to be much faster and somewhat more effective. Results for the code coupled with an implicit Euler solver for a three-block configuration on a finite wing are compared with experimental data for transonic flow.

Introduction

NUMERICAL grid generation is a prelude to numerical solution of partial-differential equations (PDE). One problem in solving PDE with a fixed grid is that grid points are distributed in the physical domain before the solution is known. As a result, the grid may not be the best suited for the particular physical problem. The idea of an adaptive grid is to have the grid points move as the physical solution develops by concentrating points in the regions of large variation in the solution as they emerge. The grid points are redistributed by sensing the gradients in the evolving physical solution and evaluating the accuracy of the discrete representation of the solution.

This adaption can reduce the oscillations associated with inadequate resolution of large gradients, allowing for sharper shocks and better representation of boundary layers. Another advantageous feature is the fact that in the viscous regions where real diffusion effects must not be swamped, the numerical dissipation from upwind biasing is reduced by the adaption. Dynamic adaption is at the frontier of numerical grid generation and may well prove to be one of its most important aspects, along with the treatment of real three-dimensional configurations through the composite grid structure.

In the present effort, the EAGLE grid code[1,2] has been extended to be adaptive, incorporating both the control function adaptive approach and the variational adaptive approach, and has been coupled with an implicit Euler flow solver.[3,4]

EAGLE Grid Code

The construction of computational fluid dynamics (CFD) codes for complicated regions is greatly simplified by a composite-block grid structure since, with the use of a surrounding layer of points on each block, a flow code is only required basically to operate on rectangular computational regions. The necessary correspondence of points on the surrounding layers (image points) with interior points (object points) is set up by the grid code and made available to the CFD solution code.

The EAGLE grid code[1,2] is a general three-dimensional algebraic/elliptic grid generation code based on the block structure. This code allows any number of blocks to be used to fill an arbitrary three-dimensional region. Any block can be linked to any other block (or to itself) with complete (or lesser) continuity across the block interfaces as specified by input. This code uses an elliptic generation system with automatic evaluation of control functions, either directly from the initial algebraic grid (generated by transfinite interpolation) and then smoothed, or by interpolation from the boundary point distributions.[5] In the latter case, the arc length and curvature contributions to the control functions are evaluated and interpolated separately into the field from the appropriate boundaries, and the control function at each point in the field is then formed by combining the interpolated components. This procedure allows very general regions, with widely varying boundary curvature, to be treated.

The control functions can also be determined automatically to provide orthogonality at boundaries with specified normal spacing. Here, the iterative adjustments in the control functions are made by increments radiated from the boundary-points where orthogonality has not been attained. This allows the basic geometric control function structure evaluated from the algebraic grid, or from the boundary-point distributions, to be retained and thus relieves the iterative process of the need to establish this geometric form of the control functions.

Alternatively, boundary orthogonality can be achieved through Neumann boundary conditions, which allow the boundary points to move over a surface spline, the boundary-point locations being located by Newton iteration on the spline to be at the foot of normals to the adjacent field points. (This feature is used on the boundary in the present adaptive application.) Provision is also made for extrapolated zero-curvature boundary conditions on the surface spline and for mirror-image reflective boundary conditions on symmetry planes.

The grid is structured as follows. The entire three-dimensional physical region is filled with a set of interfacing hexahedrons with curved surfaces, each of which corresponds to a rectangular computational block. Each of these blocks has its own set of right-handed curvilinear coordinates (independent of those in the other blocks). The blocks do not have to be all the same size, and the size of each is specified by input. It is also not necessary for an entire side of one block to correspond to an entire side of an adjacent block. It is only necessary that all of the corresponding blocks fit together to fill the physical region.

Presented as Paper 88-0311 at the AIAA 26th Aerospace Sciences Meeting, Reno, NV, Jan. 11-14, 1988, received March 23, 1988; revision received Feb. 28, 1989.

*Lt. Colonel.
†Professor of Aerospace Engineering. Member AIAA.

Each computational block is surrounded by an extra layer of points in order to allow connections across the interfaces in the physical region to be formed. (Actually, provision is made for still another surrounding layer of points in order to provide connections for use in flow codes using three-point one-sided differences, but this extra layer is not used by the grid code.) The interfaces between blocks are branch cuts, and the code establishes a correspondence across the interface using the surrounding layer of points outside the blocks. This allows points on the interface to be treated just as all other points, so that there is no loss of continuity. The physical location of the interface is thus unspecified, being determined by the code.

The features of this code and its use are discussed in Ref. 1. Detailed discussion of both the use and the operation of the code is given in Ref. 2. The elliptic generation system is discussed in detail in Ref. 5, and some examples of applications have appeared in Refs. 3, 4, and 6–9.

Adaptive Grid Strategies

There are three basic strategies that may be employed in dynamically adaptive grids (cf. the survey given by Ref. 10) coupled with the PDE of the physical problem. (Combinations are also possible, of course.)

Redistribution of a Fixed Number of Points

In this approach points move from regions of relatively small error or solution gradient to regions of large error or gradient. While the global order of the approximation cannot be increased by such movement of points (cf. Ref. 11), it is possible to improve the approximation locally as significant gradients are better resolved. Thus, although there is no formal increase in global accuracy in the limit of infinitesimal spacing, there is definitely a practical increase at the finite spacings actually involved in CFD. As long as the redistribution of points does not seriously deplete the number of points in other regions of possible significant gradients, this is a viable approach. The increase in spacing that must occur somewhere is not of practical consequence if it occurs in regions of small error or gradient, even though in a formal mathematical sense the global approximation is not improved. The redistribution approach has the advantage of not increasing the computer time and storage during the solution and of being straightforward in coding and data structure. The disadvantages are the possible deleterious depletion of points in certain regions and the possibility of the grid becoming too skewed. Some examples of this adaptive approach in CFD are Ref. 12 for two dimensions and Ref. 13 for three dimensions.

Local Refinement of a Fixed Set of Points

In this approach, points are added (or removed) locally in a fixed point structure in regions of relatively large error or solution gradient. Here there is, of course, no depletion of points in other regions and therefore no formal increase of error occurs. Since the error is locally reduced in the area of refinement, the global error does formally decrease. The practical advantage of this approach is that the original point structure is preserved. The disadvantages are that the computer time and storage increase with the refinement and that the coding and data structure are difficult, especially for implicit flow solvers. Recent examples of this adaptive approach in CFD are Refs. 14 and 15, both in two dimensions.

Local Increase in Algorithm Order

In this approach the solution method is changed locally to a higher-order approximation in regions of relatively large error or solution gradient without changing the point distribution. This again increases the formal global accuracy since a local increase is achieved without an attendant decrease in formal accuracy elsewhere. The advantage is that the point distribution is not changed at all. The disadvantage is the great complexity of implementation in implicit flow solvers. This adaptive approach has not had any significant application in CFD in multiple dimensions.

Adaptive Formulation

With structured grids and implicit flow solvers, the adaptive strategy based on redistribution is by far the most simple to implement, requiring only the regeneration of the grid at each adaptive stage without modification of the flow solver unless time accuracy is desired. Time accuracy can be achieved, as far as the grid is concerned, by simply transforming the time derivatives, thus adding additional convective-like terms that do not alter the basic conservation form of the PDE.

Adaptive redistribution of points traces its roots to the principle of equidistribution of error (cf. Ref. 10) by which a point distribution is set so as to make the product of the spacing and a weight function constant over the points:

$$w\Delta x = \text{const} \tag{1}$$

With the point distribution defined by a function $x(\xi)$, where ξ varies by a unit increment between points, the equidistribution principle can be expressed as

$$wx_\xi = \text{const} \tag{2}$$

This one-dimensional equation can be applied in each direction in an alternating fashion, a good example of which is in Ref. 13 (cf. also Ref. 16). However, a direct extension to multiple dimensions can be made in either of two ways as follows.

Control Function Approach

This approach is developed by noting the correspondence between Eq. (2) and the one-dimensional form of the elliptic grid generation system (cf. Ref. 5, or Chapter 6 of Ref. 17),

$$\sum_i \sum_j g^{ij}\mathbf{r}_{\xi^i\xi^j} + \sum_k P_k g^{kk}\mathbf{r}_{\xi^k} = 0 \tag{3}$$

used in the EAGLE code and in many other grid generation codes. Here, the g^{ij} are the elements of the contravariant metric tensor:

$$g^{ij} = \nabla\xi^i \cdot \nabla\xi^j \tag{4}$$

These elements are more conveniently expressed computationally in terms of the elements of the covariant metric tensor, g_{ij}:

$$g_{ij} = \mathbf{r}_{\xi^i} \cdot \mathbf{r}_{\xi^j} \tag{5}$$

which can be calculated directly. Thus,

$$g^{ij} = \frac{1}{g}(g_{mk}g_{n\ell} - g_{m\ell}g_{nk}) \tag{6}$$

$$(i,m,n) \text{ cyclic}, \qquad (i,k,\ell) \text{ cyclic}$$

where g, the square of the Jacobian, is given by

$$g = \det|g_{ij}| = [\mathbf{r}_{\xi^1} \cdot (\mathbf{r}_{\xi^2} \times \mathbf{r}_{\xi^3})]^2 \tag{7}$$

In these relations, $\boldsymbol{r}$ is the Cartesian position vector of a grid point ($\boldsymbol{r} = \boldsymbol{i}x + \boldsymbol{j}y + \boldsymbol{k}z$) and the ξ^i ($i = 1, 2, 3$) are the curvilinear coordinates. The P_k are the "control functions," which serve to control the spacing and orientation of the grid lines in the field.

The one-dimensional form of this system is

$$x_{\xi\xi} + Px_\xi = 0 \tag{8}$$

Differentiation of Eq. (2) yields

$$wx_{\xi\xi} + w_\xi x_\xi = 0 \tag{9}$$

Then, from Eqs. (8) and (9),

$$-P = \frac{x_{\xi\xi}}{x_\xi} = -\frac{w_\xi}{w} \tag{10}$$

from which the control function can be taken as

$$P = \frac{w_\xi}{w} \tag{11}$$

It is natural then to represent the control functions in three dimensions as

$$P_k = \frac{w_{\xi^k}}{w} \qquad (k = 1, 2, 3) \tag{12}$$

This approach was proposed by Anderson[18,19] and has been applied with success to two-dimensional channel configurations,[20,21] to two-dimensional coastal configurations,[22] and now to a three-dimensional wing by Kim.[23] The first two applications involved static adaption to water depth, while the last two use an adaptive transonic Euler solver fully coupled dynamically with the EAGLE code. The weight function in the two depth-adaptive cases was

$$w = 1 + \text{depth} \tag{13}$$

and for the three-dimensional wing was based on the pressure gradient:

$$w = 1 + |\nabla p| \tag{14}$$

(These and more general forms of weight functions are discussed in Ref. 10). This control function adaptive approach has the significant advantage of being based on the same elliptic generation equations that are in common use in grid generation codes, and the adaptive control functions given by Eq. (12) can be added to those already evaluated from the configuration geometry.

The complete generalization of Eq. (11) is

$$P_i = \sum_j \frac{g^{ij}}{g^{ii}} \frac{(w_i)_{\xi^j}}{w_i} \tag{15}$$

involving three weight functions, w_i $(i = 1, 2, 3)$, as given by Eiseman.[24] This more general form was evaluated, but little advantage was found in the present cases. However, in some applications the availability of three separate control functions could be a definite advantage, with perhaps the velocity gradient used in the weight function in one direction and the pressure gradient in another. This general form is therefore included in the adaptive EAGLE code.

Variational Approach

From the calculus of variations, Eq. (2) can be shown (cf. Chapter 11 of Ref. 17) to be the Euler variational equation for the function $x(\xi)$, which minimizes the integral

$$I = \int w(\xi) x_\xi^2 \, d\xi \tag{16}$$

Generalizing this, a competitive enhancement of grid smoothness, orthogonality, and concentration can be accomplished by representing each of these features by integral measures over the grid and minimizing a weighted average of the three. This approach was put forward by Brackbill and Saltzman[12] and is discussed in detail in Chapter 11 of Ref. 17.

The smoothness integral is represented by

$$I_s = \iiint \sum_i (\nabla \xi^i \cdot \nabla \xi^i) \, dx \tag{17}$$

where the quantity $\nabla \xi^i \cdot \nabla \xi^i$ represents the variation of the curvilinear coordinates over the field and thus is a measure of the roughness of the grid. Therefore, to maximize the smoothness of the grid, it is appropriate to minimize the integral of this quantity over the field. Applying the variational equation to Eq. (17) yields

$$\nabla^2 \xi^i = 0 \qquad (i = 1, 2, 3)$$

showing that the Laplace equation gives the smoothest grid.

The orthogonality of the grid can be emphasized by minimizing the integral I_0, defined as

$$I_0 = \iiint \sum_i (\nabla \xi^j \cdot \nabla \xi^k)^2 g^{3/2} \, dx \tag{18}$$

where the quality $\nabla \xi^j \cdot \nabla \xi^k$ vanishes for an orthogonal grid. The inclusion of $g^{3/2}$, where g is the square of the Jacobian of the transformation, is somewhat arbitrary, causing orthogonality to be emphasized more strongly in the larger cells and also simplifying the Euler equations.

Finally, the concentration of the grid can be obtained by minimizing the integral I_w, defined as

$$I_w = \iiint w^2(x) \sqrt{g} \, dx \tag{19}$$

where $w(x)$ is a specified weight function and g is the square of the Jacobian, i.e., the cell volume. This causes the cells to be small where the weight function is large.

The grid generation system is then obtained from the Euler equations for minimization of the linear combination I of the smoothness, orthogonality, and concentration integrals given by Eqs. (17), (18), and (19):

$$I = c_s I_s + c_0 I_0 + c_w I_w \tag{20}$$

where c_s, c_0, and c_w are the specified coefficients that place more emphasis on smoothness, orthogonality, and concentration, respectively.

Since the three integrals I_s, I_0, and I_w have different units, they must be normalized, so that we have

$$I = c_s I_s + c_0 a_0 I_0 + c_w a_w \frac{1}{w_a} I_w \tag{21}$$

where a_0 and a_w are expressed in terms of N and L, where N is a characteristic number of points, L is a characteristic length, and w_a is the average weight function over the field, i.e.,

$$a_0 = \left(\frac{N}{L}\right)^7 \tag{22a}$$

$$a_w = \left(\frac{N}{L}\right)^5 \tag{22b}$$

$$w_a = \frac{1}{V} \iiint w^2 \, dx \tag{22c}$$

with V being the volume of the field, i.e., the sum of the Jacobian $\sqrt{g}$ over all cells. This scaling in the weighted sum is obtained as follows. From the given expressions for I_s, I_0, and I_w we have

$$I_s \sim \left(\frac{N}{L}\right)^2 L^3 \tag{23a}$$

$$I_0 \sim \left(\frac{L}{N}\right)^5 L^3 \tag{23b}$$

$$I_w \sim \left(\frac{L}{N}\right)^3 L^3 W^2 \tag{23c}$$

Therefore, the three terms in Eq. (21) should stand in the ratios given. In two dimensions both factors on I_0 and I_w become $(N/L)^4$, since the Jacobian is then proportional to $(L/N)^2$, the cell area, rather than to $(L/N)^3$, the cell volume.

The corresponding integrals in the transformed region can be obtained directly by transforming the integrals of Eqs. (17–19), with (i, j, k) cyclic:

$$I_s = \int\int\int \frac{1}{\sqrt{g}} \sum_i (g_{jj}g_{kk} - g_{jk}^2)\, \mathrm{d}\boldsymbol{\xi} \tag{24}$$

$$I_0 = \int\int\int \sum_i (g_{ij}g_{ik} - g_{ii}g_{jk})^2\, \mathrm{d}\boldsymbol{\xi} \tag{25}$$

$$I_w = \int\int\int w^2(\boldsymbol{x}) g\, \mathrm{d}\boldsymbol{\xi} \tag{26}$$

In general we have the integral

$$I = \int F[\boldsymbol{g}, w(\boldsymbol{x})]\, \mathrm{d}\boldsymbol{\xi} \tag{27}$$

The Euler equations are then given by

$$\sum_j \frac{\partial}{\partial \xi^j} \frac{\partial F}{\partial (x_i)_{\xi^j}} - \frac{\partial F}{\partial x_i} = 0 \qquad (i = 1, 2, 3) \tag{28}$$

For the three integrals given by Eqs. (24–26) we have, respectively, with (i, j, k) cyclic,

$$F_s = \frac{1}{\sqrt{g}} \sum_i (g_{jj}g_{kk} - g_{jk}^2) \tag{29}$$

$$F_0 = \sum_i (g_{ij}g_{ik} - g_{ii}g_{jk})^2 \tag{30}$$

$$F_w = gw^2(\boldsymbol{x}) \tag{31}$$

The resulting generation equation (cf. Ref. 23) is

$$\sum_j \sum_k \left[A_{jk} \boldsymbol{r}_{\xi^j \xi^k} + A'_{jk} (\nabla w \cdot \boldsymbol{r}_{\xi^j}) \boldsymbol{r}_{\xi^k} + \sum_m \sum_n \frac{\partial A_{jk}}{\partial g_{mn}} (\boldsymbol{r}_{\xi^m} \cdot \boldsymbol{r}_{\xi^j \xi^n} + \boldsymbol{r}_{\xi^n} \cdot \boldsymbol{r}_{\xi^j \xi^m}) \boldsymbol{r}_{\xi^k} \right] - F' \nabla w = 0 \tag{32}$$

where

$$A_{jk} = \frac{\partial F}{\partial g_{jk}} + \frac{\partial F}{\partial g_{kj}}$$

$$A'_{jk} = \frac{\partial A_{jk}}{\partial w}$$

$$F' = \frac{\partial F}{\partial w}$$

To solve Eq. (32), define

$$\boldsymbol{r}_{\xi^i \xi^j} = \bar{\boldsymbol{r}}_{\xi^i \xi^j} - \delta_{ij} 2\boldsymbol{r}$$

where δ_{ij} is the Kronecker delta and

$$\bar{\boldsymbol{r}}_{\xi^i \xi^j} = \boldsymbol{r}_{\xi^i \xi^j} + 2\boldsymbol{r} \qquad \text{for} \qquad i = j$$

$$= \boldsymbol{r}_{\xi^i \xi^j} \qquad \text{for} \qquad i \neq j$$

Then, by using the summation convention,

$$A_{ij}(\bar{\boldsymbol{r}}_{\xi^i \xi^j} - 2\delta_{ij}\boldsymbol{r})_p + \bar{A}_{ijmn}(\boldsymbol{r}_{\xi^m})_q (\bar{\boldsymbol{r}}_{\xi^i \xi^n} - 2\delta_{in}\boldsymbol{r})_q (\boldsymbol{r}_{\xi^j})_p + A'_{ij}(\nabla w)_p (\boldsymbol{r}_{\xi^i})_q (\boldsymbol{r}_{\xi^j})_p - F'(\nabla w)_p = 0 \tag{33}$$

where p and q are 3×3 matrix indices and

$$\bar{A}_{ijmn} = \frac{\partial A_{ij}}{\partial g_{mn}}$$

Then, Eq. (33) can be rewritten as

$$[2A_{ii}\delta_{pq} + 2\bar{A}_{ijmi}(\boldsymbol{r}_{\xi^m})_q (\boldsymbol{r}_{\xi^j})_p](\boldsymbol{r})_q = A_{ij}(\bar{\boldsymbol{r}}_{\xi^i \xi^j})_p + \bar{A}_{ijmn}(\boldsymbol{r}_{\xi^m})_q (\boldsymbol{r}_{\xi^i \xi^n})_q (\boldsymbol{r}_{\xi^j})_p + A'_{ij}(\nabla w)_q (\boldsymbol{r}_{\xi^i})_q (\boldsymbol{r}_{\xi^j})_p - F'(\nabla w)_p \tag{34}$$

or, in vector form,

$$\boldsymbol{B}\boldsymbol{r} = \boldsymbol{K} \tag{35}$$

where

$$\boldsymbol{r} = \begin{bmatrix} r_1 \\ r_2 \\ r_3 \end{bmatrix} = \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$

and

$$B_{pq} = 2[A_{ii}\delta_{pq} + \bar{A}_{ijmi}(\boldsymbol{r}_{\xi^m})_q (\boldsymbol{r}_{\xi^j})_p] \qquad (p = 1, 2, 3 \quad \text{and} \quad q = 1, 2, 3)$$

$$\boldsymbol{K} = A_{ij}\bar{\boldsymbol{r}}_{\xi^i \xi^j} + \bar{A}_{ijmn}(\boldsymbol{r}_{\xi^m} \cdot \boldsymbol{r}_{\xi^i \xi^n})\boldsymbol{r}_{\xi^j} + A'_{ij}(\nabla w \cdot \boldsymbol{r}_{\xi^i})\boldsymbol{r}_{\xi^j} - F'\nabla w$$

The solution, $\boldsymbol{r}$, is obtained by solving the 3×3 matrix equation, Eq. (35). Each element of the matrix has many function evaluations, thus causing a much longer computing time than in the control function adaptive grid generation system. The generality of the variational approach is appealing, however, and the formulation given here greatly reduces the programming complexity of this approach.

The adaptive three-dimensional Euler solver of Ref. 23 also includes the variational adaptive approach for comparison (as did the two-dimensional depth-adaptive code of Refs. 20 and 21). In both of these cases no advantage was found for the variational approach, and this approach was found to require up to an order of magnitude more computer time because of the complexity of its equations. Therefore, the control function adaptive approach has been adopted for the adaptive EAGLE code.

Results and Discussion

The algorithm developed in this study for generating an adaptive grid has been tested on several physical problems in two and three dimensions. Both the control function approach and the variational approach algorithms performed well. When the variational form was used, the specified concentration constant, c_w, was required to be a large number between 30 and 50, sometimes more than 100, or else the solutions were not improved and the grid movement could hardly be recognized. Moreover, it took more than three times the computer time to generate the variational adaptive grid as compared to the control function adaptive grid. On the other hand, when the control function algorithm was used, grids were moved rapidly and sensitively with the small value of c_w (the coefficient of the adaptive control functions when added to the geometric control functions) between 0.5 and 1.5.

Therefore, after preliminary tests, only the control function algorithm was used for the present study.

The simulations were done with the CRAY X-MP. Details of the use of the adaptive grid generation code are fully explained in Ref. 23.

Finite Wing

Results for the two-dimensional dynamic adaption for an NACA 0012 airfoil at Mach 0.8 and 1.25-deg angle of attack are given in Ref. 23. Only the three-dimensional results are included herein.

For the three-dimensional dynamic adaption, an ONERA M6 wing was considered. The initial grid was generated by the elliptic system with Neumann boundary conditions on the wing with the point movement restricted to the chordwise direction. The grid used in this study was a C-H mesh, dimensioned $97 \times 17 \times 17$ and divided into three blocks in the spanwise direction as shown in Fig. 1. Both nonadaptive and adaptive solutions were obtained with 210 cycles for the steady state under the conditions of 3.06-deg angle of attack at $M = 0.84$. Three adaptions, after 30, 60, and 120 cycles, were used for the adaptive solution. The pressure gradient was used for the weight variable, Eq. (14), and the concentration factor c_w was set to 1.2.

Figure 2 shows the final grid adapted to the pressure gradient at the 50% span location and over the upper wing surface. The corresponding pressure contours for the steady state are given in Fig. 3. A typical λ-type shock (two shocks merged into one toward the wing tip) on the upper wing surface can be easily recognized for the adaptive grid as shown in Figs. 2 and 3. The steady-state solutions for pressure on the adaptive and fixed grids can be compared in Figs. 3 and 4 at the 50% span location and on the upper wing surface. The static pressure coefficients on the entire wing are illustrated in Fig. 5, and the C_p curves at the 45% span locations are compared with experiment (Ref. 25) in Fig. 6. (Further comparisons at 65% span are given by Fig. 10.) It is clear that the adaptive solution is closer to the experiment than is the fixed solution, not only at the shock but also at the leading-edge suction peak. Finally, the convergence history of

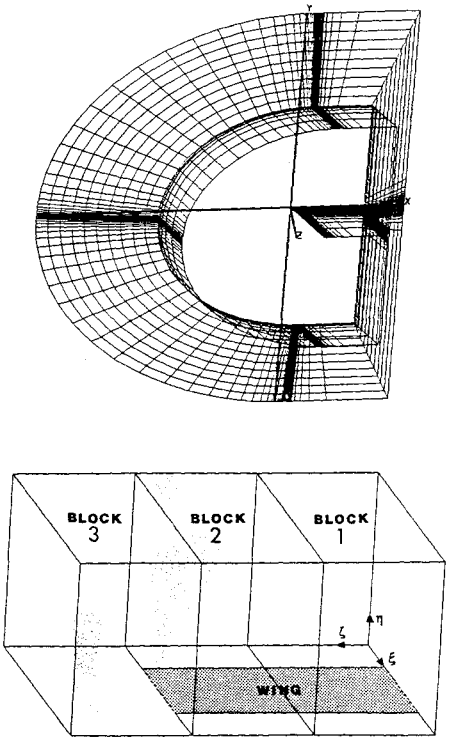
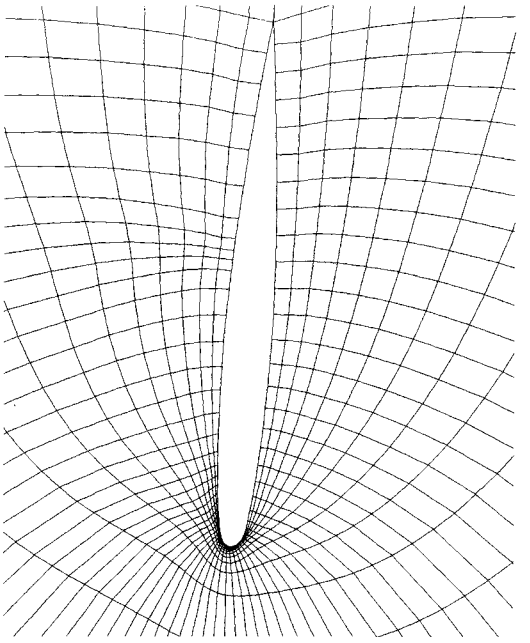
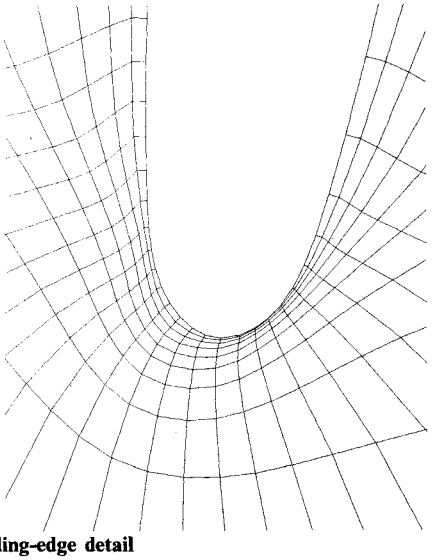


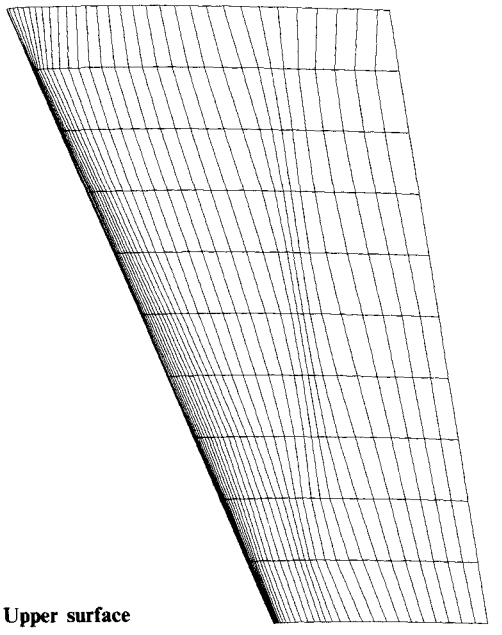
Fig. 1 ONERA M6 wing composite grid.



a) 50% span

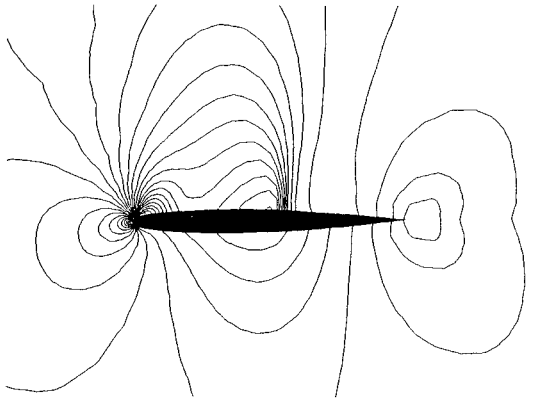


b) Leading-edge detail

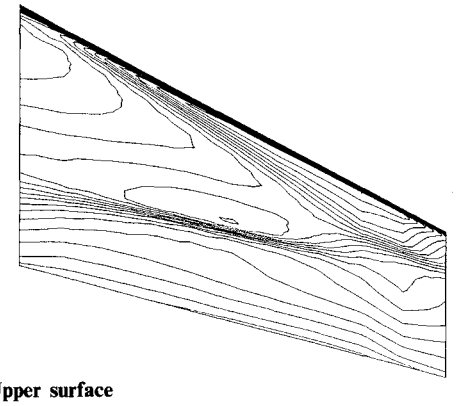


c) Upper surface

Fig. 2 Final adaptive grid.

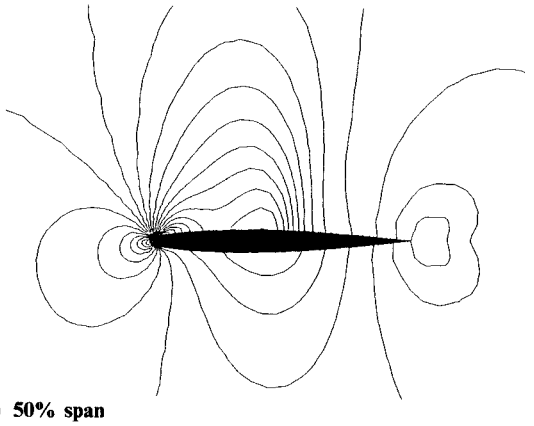


a) 50% span

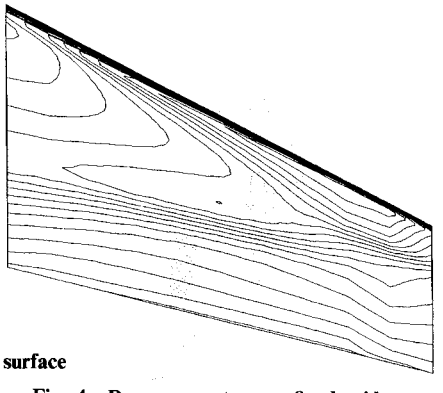


b) Upper surface

Fig. 3 Pressure contours—adaptive grid.

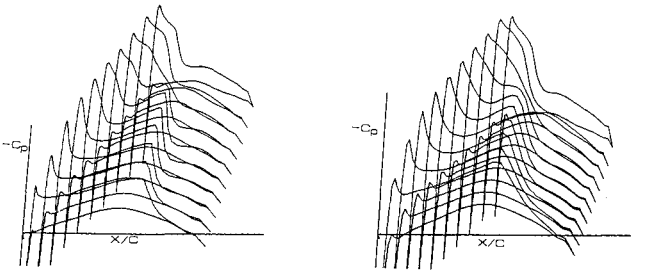


a) 50% span



b) Upper surface

Fig. 4 Pressure contours—fixed grid.



a) Fixed grid b) Adaptive grid

Fig. 5 Surface pressure coefficient.

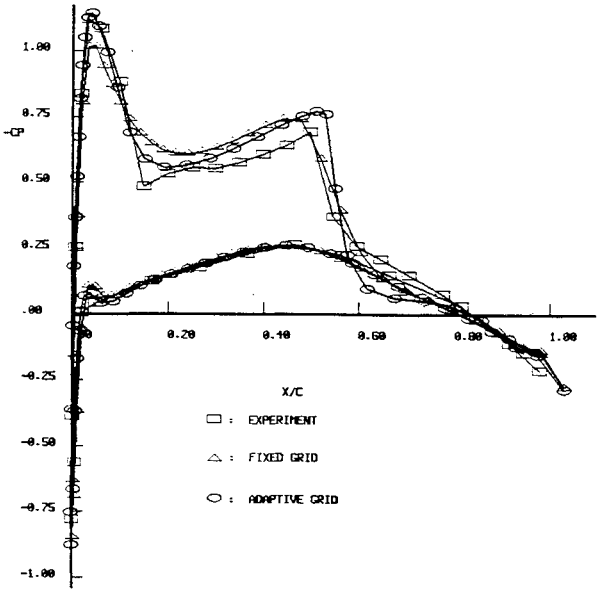
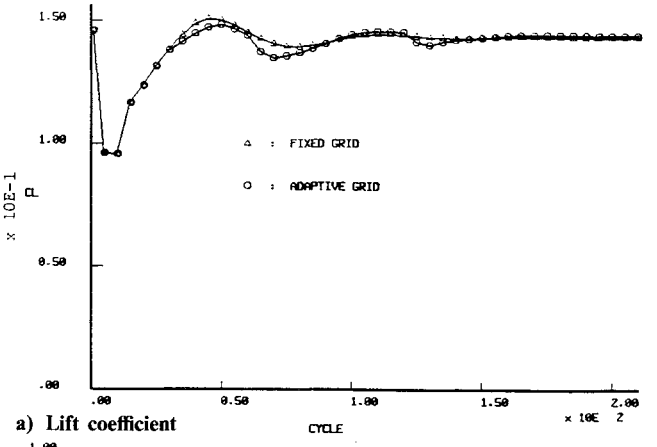
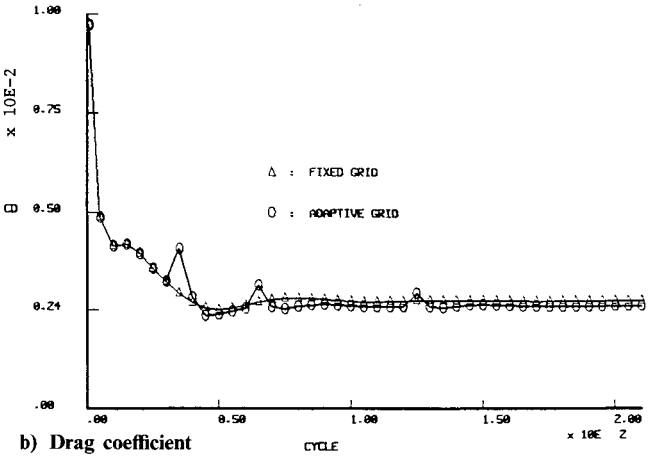


Fig. 6 Comparison of surface pressure coefficients.

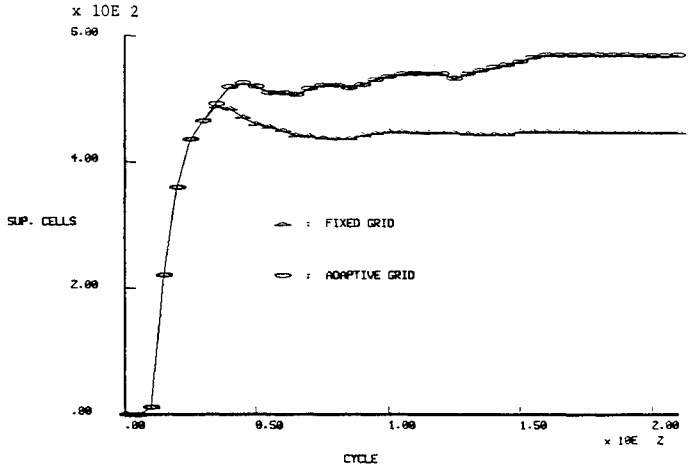


a) Lift coefficient

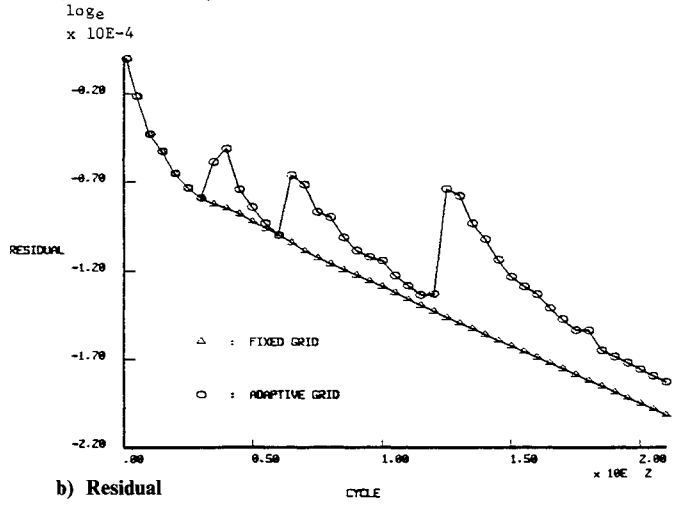


b) Drag coefficient

Fig. 7 Lift and drag convergence histories.



a) Number of supersonic cells



b) Residual

Fig. 8 Supersonic cells and residual convergence histories.

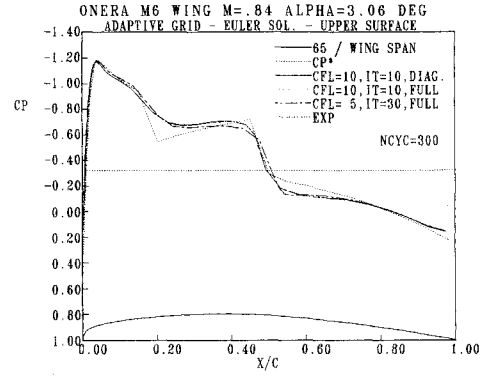
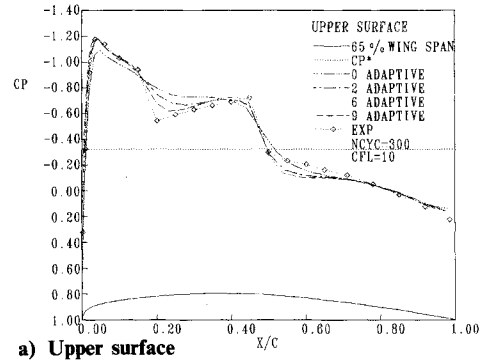
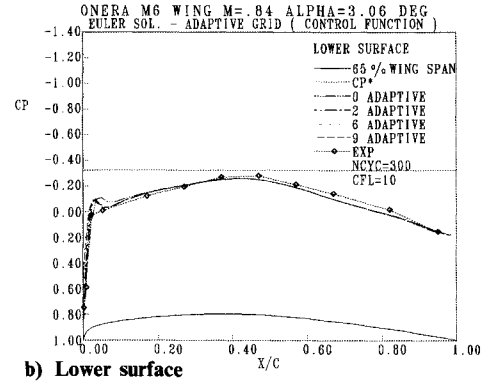


Fig. 9 Comparison of full [Eq. (15)] and diagonal [Eq. (12)] adaptive control functions.

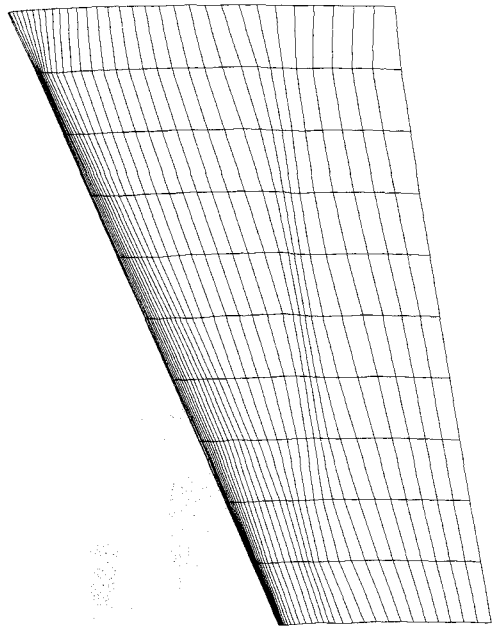


a) Upper surface

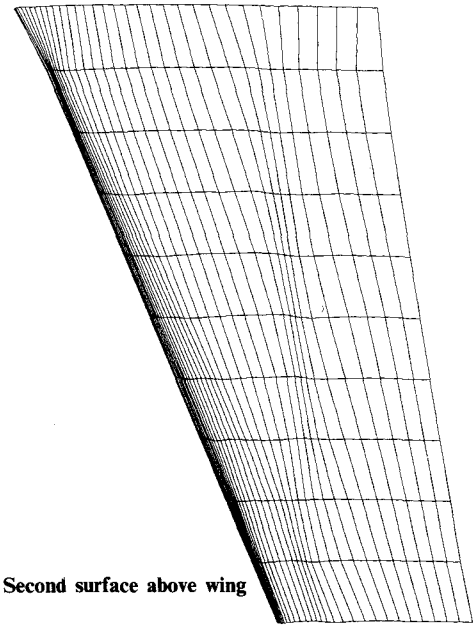


b) Lower surface

Fig. 10 Effect of number of grid adaptions.



a) First surface above wing



b) Second surface above wing

Fig. 11 Coordinate surfaces.

the lift and drag coefficients, the number of supersonic cells, and the residuals are given in Figs. 7 and 8.

Total computing time on the CRAY X-MP for the steady-state fixed-grid solution was 321 CPU-s and for the adaptive solution 350 CPU-s. Thus, the overall increase for the adaptive solution was 29 CPU-s (9%).

The completely generalized (full) form of the adaptive control functions, Eq. (15), is compared with the simpler (diagonal) form, Eq. (12), in Fig. 9. Since the difference is trivial, the diagonal form was used in all other results presented herein. This figure also shows some effect of the Courant number when a change is made from 5 to 10, with slightly better agreement with experimental results at the smaller Courant number.

Figure 10 shows the effect of the number of grid adaptions. Although six adaptions is noticeably better than two, the effect of a further increase to nine is small.

Finally, although the movement of the grid points on the wing surface via the Neumann boundary conditions was restricted to the chordwise direction, the grid adaption was fully three-dimensional in the field, as is evident in Fig. 11, which shows successive coordinate surfaces off the wing.

Conclusions

Two approaches for generating adaptive grids were investigated in the present study—the control function approach based on the elliptic generation system, and the variational approach based on the calculus of variations. Both methods have proven their capabilities for controlling the grids, but because of the longer computing time and less sensitivity in the variational method, the control function form may be the more promising tool for future applications. This adaptive control function formulation has been incorporated into the EAGLE grid code, providing a composite-block adaptive grid generation system for general three-dimensional regions to be coupled with PDE solvers.

Acknowledgments

The adaptive version of the EAGLE code was supported by Air Force Grant F0-8635-84-C-02281, Eglin AFB (Dr. Lawrence Lijewski, monitor). Abi A. Arabshahi and Yen Tu, graduate students at Mississippi State, assisted with the Euler solver.

References

[1]Thompson, J. F., "A Composite Grid Generation Code for General 3-D Regions—the EAGLE Code," *AIAA Journal*, Vol. 26, 1988, p. 271.

[2]Thompson, J. F. and Gatlin, B., *Program EAGLE User's Manual, Volume II—Surface Generation Code* and *Volume III—Grid Generation Code*, AFATL-TR-88-117, Eglin AFB, 1988.

[3]Belk, D. M. and Whitfield, D. L., "3-D Euler Solutions on Blocked Grid Using an Implicit Two-Pass Algorithm," AIAA Paper 87-0450, 1987.

[4]Belk, D. M. and Whitfield, D. L., "Time-Accurate Euler Equations Solutions on Dynamic Blocked Grids," AIAA Paper 87-1127-CP, 1987.

[5]Thompson, J. F., "A General Three-Dimensional Elliptic Grid Generation System on a Composite Block Structure," *Computer Methods in Applied Mechanics and Engineering*, Vol. 64, 1987, p. 377.

[6]Lijewski, L. E., "Transonic Flow Solutions on a Blunt, Body-Wing-Canard Configuration Using the Euler Equations," AIAA Paper 87–2273, 1987.

[7]Martinez, A., "Application of Numerical Grid Generation to Advanced Weapon Airframe," AIAA Paper 87-2294, 1987.

[8]Thompson, J. F., Lijewski, L. E., and Gatlin, B., "Efficient Application Techniques of the EAGLE Grid Code to Complex Missile Configurations," AIAA Paper 89-0361, 1989.

[9]Thompson, J. F. and Lijewski, L. E., "Composite Grid Generation for Aircraft Configurations with the EAGLE Code," *Three-Dimensional Grid Generation of Complex Configurations—Recent Progress*, edited by J. F. Thompson and J. L. Steger, AGARD-AG-309, 1988, p. 85.

[10]Thompson, J. F., "A Survey of Dynamically-Adaptive Grids in the Numerical Solution of Partial Differential Equations," *Applied Numerical Mathematics*, Vol. 1, 1985, p. 3.

[11]Thompson, J. F. and Mastin, C. W., "Order of Difference Expressions in Curvilinear Coordinate Systems," *Journal of Fluids Engineering*, Vol. 107, 1985, p. 241.

[12]Brackbill, J. U. and Saltzman, J. S., "Adaptive Zoning for Singular Problems in Two Dimensions," *Journal of Computational Physics*, Vol. 46, 1982, p. 342.

[13]Nakahashi, K. and Deiwert, G. S., "Three-Dimensional Adaptive Grid Method," *AIAA Journal*, Vol. 24, 1986, p. 948.

[14]Dannenhoffer, J. F. III and Baron, J. R., "Grid Adaption for the 2-D Euler Equations," AIAA Paper 85-0484, 1985.

[15]Oden, J. T., Strouboulis, T., and Devloo, P., "An Adaptive Finite Element Strategy for Complex Flow Problems," AIAA Paper 87-0557, 1987.

[16]Eiseman, P. R., "Alternating Direction Adaptive Grid Generation," AIAA Paper 83-1937, 1983.

[17]Thompson, J. F., Warsi, Z. U. A., and Mastin, C. W., *Numerical Grid Generation: Foundations and Applications*, North-Holland, Amsterdam, 1985.

[18]Anderson, D. A., "Generating Adaptive Grids with a Conventional Grid Scheme," AIAA Paper 86-0427, 1986.

[19]Anderson, D. A., "Equidistribution Schemes, Poisson Generators, and Adaptive Grids," *Applied Mathematics and Computation*, Vol. 24, 1987, p. 211.

[20]Thompson, J. F. and Johnson, B. H., "Development of an Adaptive Boundary-Fitted Coordinate Code for Use in Coastal and Estuarine Areas," Miscellaneous Paper HL-85-5, U.S. Army Engineer Waterways Experiment Station, Vicksburg, MS, 1986.

[21]Johnson, B. H. and Thompson, J. F., "Discussion of a Depth-Dependent Adaptive Grid Generator for Use in Computational Hydraulics," *Numerical Grid Generation in Computational Fluid Dynamics*, edited by J. Hauser and C. Taylor, Pineridge Press, Swansea, UK, 1986.

[22]Thompson, J. F., unpublished research, Coastal Engineering Research Center, U.S. Army Engineering Waterways Experiment Station, Vicksburg, MS, 1986.

[23]Kim, H. J., "Three-Dimensional Adaptive Grid Generation on a Composite Structure," Ph.D. Dissertation, Mississippi State University, Mississippi State, 1987.

[24]Eiseman, P. R., "Adaptive Grid Generation," *Computer Methods in Applied Mechanics and Engineering*, Vol. 64, 1987, p. 321.

[25]Schmitt, V. and Charpin, F., "Pressure Distributions on the ONERA M6-WING at Transonic Mach Numbers," *Experimental Data Base for Computer Program Assessment*, AGARD AR-138, 1979.