Application of Lagrangian Blending Functions for Grid Generation Around Airplane Geometries

Jamshid Samareh-Abolhassani* and Ideen Sadrehaghighi†
Old Dominion University, Norfolk, Virginia 23529
Robert E. Smith‡
NASA Langley Research Center, Hampton, Virginia 23665
and
Surendra N. Tiwari§
Old Dominion University, Norfolk, Virginia 23529

A simple procedure has been developed and applied for the grid generation around an airplane geometry. This approach is based on a transfinite interpolation with Lagrangian interpolation for the blending functions. By using a Lagrangian interpolation function, it is possible to enforce the grid continuity across the block interfaces without the derivative information. Monotonic rational quadratic spline interpolation has been employed for the grid distributions on the boundaries. This allows any arbitrary grid spacing without overlapping of the grid lines. An efficient computer program has been developed to generate a multiblock grid around a generic airplane geometry. This procedure has proven to be very simple and effective.

Nomenclature

a,b,c = constants for Eqs. (1) and (2)
A,B,C = surfaces defined in ξ, η, and ζ directions, respectively
F = position vector (x, y, and z)
F_1,F_2 = intermediate values for transfinite interpolations [Eqs. (8b–c)]
K = stretching coefficient [Eqs. (6)]
L,M,N = number of surfaces defined in ξ, η, and ζ directions, respectively
P,Q,R = number of derivatives defined in ξ, η, and ζ directions, respectively
x,y,z = physical coordinates
α, β, γ = blending function for constant ξ, η, and ζ, respectively
δ_{nm} = Kronecker delta
ξ, η, ζ = computational coordinates

Subscripts

i,j,k = grid number in ξ, η, and ζ directions, respectively

Superscripts

n = nth derivative
T = transpose

Introduction

To study the flowfield around any aerodynamic configuration, a system of nonlinear partial differential equations must be solved over a highly complex geometry. Regardless of the computational approach, the domain of interest should be discretized into a set of points (for the finite-difference methods) or a set of elements (for the finite element methods). This step is commonly referred to as grid generation.

Selection of the grid topology is the first step in the generation of a structured grid. In this step, orientation of the computational coordinates must be selected relative to the physical coordinates. For complex geometries, one may have to select different computational coordinate systems for different regions of the physical domain. In this case, one physical domain is mapped into several computational subdomains, and each subdomain is referred to as a block. Therefore, it is possible to have a boundary-fitted coordinate system for a highly complex configuration. The objective of this study is to present a simple procedure for generating a relatively orthogonal grid for a generic airplane geometry.

Topology of an Airplane

To establish a grid topology for any geometry, it is essential to examine each component separately.[1] A typical airplane geometry has two important components: fuselage and wing. A fuselage has a circular-like cross section, which suggests a natural O-type grid. This topology produces a nearly orthogonal grid with one line of polar singularity at the nose. In the streamwise direction, it is possible to have either a C- or an H-type grid. If the fuselage has a small slope near the nose, then it is better to use an H-type grid in the streamwise direction. The other choice is to have a C-type grid in the streamwise direction, which is good for the fuselage with a blunt nose. If the nose is sharp, C-type topology may generate a slightly skewed grid near the nose. In short, the use of an O-H or an O-C topology will result in a nearly orthogonal grid with one line of polar singularity at the nose.

In general, a wing possesses its own natural coordinate system, which may not be compatible with the fuselage's coordinate system. However, it is possible to generate an H-, O-, or C-type grid in the streamwise direction, and a C- or an H-type grid in the crosswise direction. It is also possible to generate a single-block grid about these two components, but this grid will be skewed for any practical applications. A dual-block grid possesses much less skewness than a single-block grid. However, the grid lines must be continuous at the interface between the two blocks. If both blocks share the same grid

Presented as Paper 90-0009 at the AIAA 28th Aerospace Sciences Meeting, Reno, NV, Jan. 8–11, 1990; received Feb. 15, 1990; revision received April 4, 1990.

*Visiting Assistant Professor, Department of Mechanical Engineering and Mechanics, Member AIAA.

†Graduate Research Assistant, Department of Mechanical Engineering and Mechanics. Student Member AIAA.

‡Senior Research Engineer, ACD-Computer Applications Branch. Member AIAA.

§Eminent Professor, Department of Mechanical Engineering and Mechanics. Member AIAA.

points at the interface, then the grid is referred to as being C^0 continuous across the interface. If both blocks share not only the same grid points at the interface but also nth derivatives, then the grid is referred to as being C^n continuous across the interface. To maintain a minimum of C^0 continuity at the interfaces, it is essential to select a compatible topology for the wing and the fuselage.

The dual-block grid consists of two large blocks, one covering the top part of the physical domain, and the other block covering the bottom part of the physical domain. The dual-block topology is a direct result of using an H-type grid for the wing. For higher continuity (C^1 and above), an oscillatory transfinite interpolation can be used to generate the interior grid. Then, it is possible to ensure the orthogonality at the interface as well. In general, the orthogonality of the interior grid cannot be guaranteed. This is the only limitation of the algebraic technique.

If the wingtip has a finite area, then the topology of the grid needs to be changed accordingly. This change may create additional blocks that are not desirable. Furthermore, addition of vertical and horizontal tail surfaces may also change the grid topology. These changes depend on the geometry of the tails and the required grid topology. If the leading and trailing edges of the tails are sharp, then it is a good idea to use an H-type grid around them. This will not change the dual-block topology. However, there would be one singular line emanating from the leading and trailing edges of each tail surface.

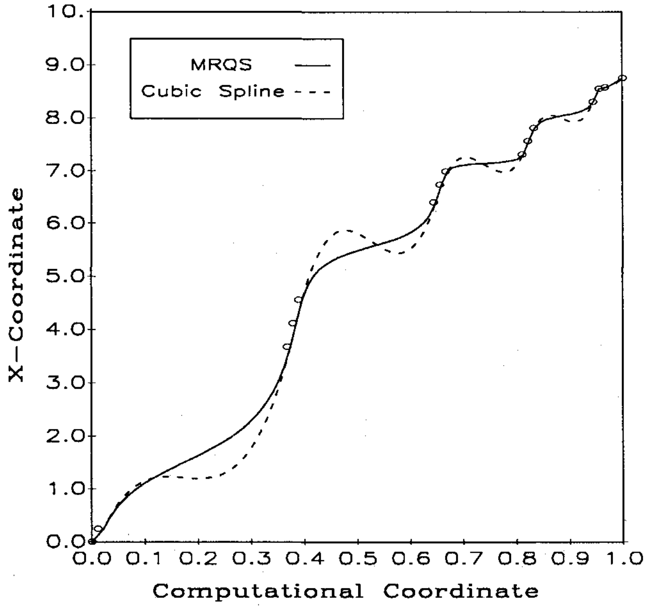

Fig. 1 Point distribution in streamwise direction.

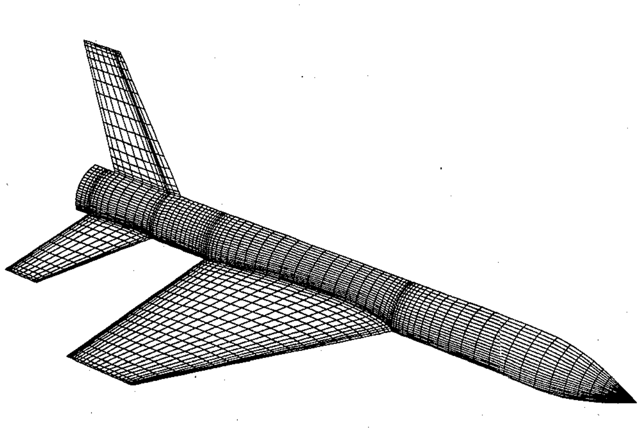

Fig. 2 Grid on solid surfaces.

These singular lines are located on the boundaries of the computational domain and will not have a dramatic effect on the flow code. It is also possible to have an O- or a C-type grid around the tails; this choice will result in creation of additional blocks.

Once the grid topology has been selected, the grid on the boundary surfaces can be generated. This step depends on how a surface is defined. A surface can be defined either by a set of analytical functions or by a set of cross sections. The former requires no interpolation, and the latter requires some sort of bidirectional interpolation. In this study, the fuselage surface is given by analytical functions and the wing and tails by their cross sections.

All solid surfaces are generated by analytical functions. In the present study, solid surfaces are composed of fuselage, wing, and horizontal and vertical tail surfaces. The fuselage consists of a cylindrical section and a nose. The nose is either sharp or blunt. A sharp nose is approximated by rotating a parabola about the centerline. This parabola can be expressed as

$$y = ax^2 + bx + c \tag{1}$$

Similarly, a quarter of an ellipse is used for the blunt nose:

$$1 = \left[(x-a)/a\right]^2 + \left[(y-b)/b\right]^2 \tag{2}$$

In Eqs. (1) and (2), constants a, b, and c are selected such that the surface is continuous (at least C^1) at the intersection of the nose and the cylindrical section. In the streamwise direction, the grid points are generated using a monotonic rational spline. In the crosswise direction, the grid points are generated using an exponential function.

The surfaces of the wing and tails are described by several cross sections using standard four-digit NACA airfoils. The wing has a NACA 0010 for a cross section with a closed wingtip. The grid points are concentrated more toward the leading and trailing edges. In the crosswise direction, grid points are concentrated near the fuselage. The intersection of the wing and fuselage is generated analytically. Grid points for the vertical and horizontal tails are generated in a similar fashion. The cross sections of the horizontal and vertical tails are NACA 0006 and 0005, respectively.

Discretization of a Curved Line

Before generating the surface grid, one needs to compute the grid-point distribution along all or part of the boundary edges. This distribution must be monotonic in the parameter space, and it can be computed by an analytical function or by a numerical approximation. In general, analytical functions are limited to simple curves. However, a complex curve can be decomposed into several sections, and analytical functions can be used for each section.[2] The advantages of analytical functions are their simplicity and the guaranteed monotonicity.

Similarly, a numerical approximation can be used to compute the grid-point distribution on a curve. This approach is widely used and care must be taken to ensure monotonicity and high accuracy. For example, the natural cubic spline is C^2 continuous, and it can generate smooth grid-point distribution. If the data have a high second derivative, the result may not be monotonic. The cubic spline can be modified to control monotonicity[3] and still be C^2 continuous. However, this scheme may not reproduce the initial data. This method has been used for two-boundary grid generation with much success.[2] The other option is to use a lower order polynomial (e.g., C^1) with guaranteed monotonicity. For instance, a monotonic rational quadratic spline (MRQS) interpolation is always monotonic[4] and smooth. Figure 1 shows results from a cubic spline and an MRQS. It can be seen clearly that the MRQS can generate a monotonic grid distribution unlike a cubic spline. The MRQS scheme is an explicit scheme and does

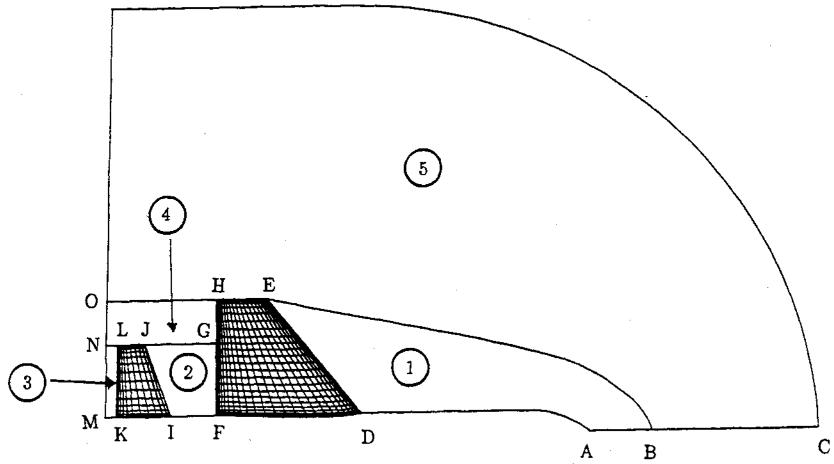

Fig. 3 Symmetry surface (x-y plane).

not require any matrix inversion. The disadvantage of the MRQS scheme is its accuracy (C^1). The algorithm for this method can be found in Ref. 4.

Surface Grid Generation

Once the grid is generated on the boundary edges, the surface grid can be generated by analytical functions or by an appropriate bidirectional interpolation. The surfaces are either physical (solid) or nonphysical (fictitious). Figure 2 shows the grid on the physical surfaces of a typical airplane geometry. The next step is to generate grid points on the remaining surface portions (nonphysical). For example, the symmetry surface in the x-y plane (Fig. 3) is surrounded by a number of lines. This region can be divided into a number of subregions, as shown in Fig. 3. In this case, it is divided into five subregions. This subdivision is arbitrary; however, it is a good idea to subdivide along a computational coordinate direction (e.g., constant ξ).

As mentioned earlier, each subregion may be defined by two or more grid lines. For each subregion, grid points can be generated by either an algebraic or a differential method. An extensive discussion of both methods can be found in Ref. 5. Most algebraic methods are based on either a transfinite[6,7] or multisurface[8] method. In this investigation, a transfinite interpolation has been used with the application of a Lagrangian interpolation for the blending functions. A significant extension of the original formulation by Gordon and Hall[6] has made it possible to generate grids for highly complex geometries with a high degree of local control.[7]

If the coordinates of some part of a surface $[F(\xi,\eta)]$ with their derivatives are known, then it is possible to generate the interior grid by a transfinite interpolation. In general, a two-step transfinite interpolation (or multivariant interpolation) for curved surfaces can be expressed as

$$F(\xi,\eta) = \{x(\xi,\eta),\ y(\xi,\eta),\ z(\xi,\eta)\}^T \tag{3a}$$

$$F_1(\xi,\eta) = \sum_{l=1}^{L} \sum_{n=0}^{P} \alpha_l^{(n)}(\xi) A_l^{(n)}(\eta) \tag{3b}$$

$$F(\xi,\eta) = F_1(\xi,\eta) + \sum_{l=1}^{M} \sum_{n=0}^{Q} \beta_l^{(n)}(\eta) \left[B_l^{(n)}(\xi) - \frac{\partial^n F_1}{\partial \eta^n}(\xi,\eta_l) \right] \tag{3c}$$

where the $A_l^{(n)}$ and $B_l^{(n)}$ are the known coordinate lines on the surface with their derivatives

$$\frac{\partial^n F}{\partial \xi^n}(\xi_l,\eta) = A_l^{(n)}(\eta), \quad l = 1,2,\ldots,L, \quad n = 0,1,\ldots,P \tag{3d}$$

$$\frac{\partial^n F}{\partial \eta^n}(\xi,\eta_l) = B_l^{(n)}(\xi), \quad l = 1,2,\ldots,M, \quad n = 0,1,\ldots,Q \tag{3e}$$

and $\alpha_l^{(n)}(\xi)$ and $\beta_l^{(n)}(\eta)$ are the univariant blending functions. These functions are subjected to the following cardinal conditions:

$$\frac{\partial^m \alpha_l^{(n)}(\xi_i)}{\partial \xi^m} = \delta_{l,i}\delta_{n,m} \tag{4a}$$

$$\frac{\partial^m \beta_l^{(n)}(\eta_i)}{\partial \eta^m} = \delta_{l,i}\delta_{n,m} \tag{4b}$$

These conditions allow the input boundaries to be reproduced. Equations (1) and (2) are based on an oscillatory interpolation that makes it possible to enforce orthogonality of the grid at the boundary surfaces. Eriksson[9] has used a similar formulation with orthogonality defined at one surface.

Selection of the blending function depends on the number of specified boundaries. If only two boundaries are defined in one computational direction, then the Lagrangian interpolation would convert to a simple linear interpolation

$$\alpha_1(\xi) = \frac{(\xi_2-\xi)}{(\xi_2-\xi_1)} \tag{5a}$$

$$\alpha_2(\xi) = \frac{(\xi-\xi_1)}{(\xi_2-\xi_1)} \tag{5b}$$

This works if the boundaries do not contain sharp discontinuities. Otherwise, these discontinuities will propagate into the interior regions. One way to alleviate this problem is to construct a blending function that has a very small value away from the boundaries. For example, the following blending functions demonstrate these criteria[7]:

$$\alpha_1(\xi) = \left\{ \exp\left[K \frac{(\xi_2-\xi)}{(\xi_2-\xi_1)} \right] - 1 \right\} \Big/ \left[\exp(K) - 1 \right] \tag{6a}$$

$$\alpha_2(\xi) = \left\{ \exp\left[K \frac{(\xi-\xi_1)}{(\xi_2-\xi_1)} \right] - 1 \right\} \Big/ \left[\exp(K) - 1 \right] \tag{6b}$$

where K is a negative number greater than 1. The larger the value of K, the less the discontinuity will propagate. A similar blending function can be constructed for the η direction.

The other choice for the blending function is the Lagrangian interpolation, which satisfies the cardinal conditions. For example, if the lines in the ξ direction are given at $\xi_1, \xi_2, \ldots, \xi_n$, then the blending function $\alpha_l(\xi)$ can be defined as

$$\alpha_l(\xi) = \prod_{\substack{j=1 \\ j\neq l}}^{n} \frac{(\xi-\xi_j)}{(\xi_l-\xi_j)} \tag{7}$$

For $n=2$, this equation will reduce to Eq. (5). For a surface that is defined by several lines, one can use the general definition in Eq. (7).

In Fig. 3, grids on the wing (DFHE) and the horizontal tail (IJLK) have been generated by using airfoil definitions. Similarly, the other boundary lines (AC, AM, CP, and MP) are defined by analytical functions [Eqs. (1) and (2)]. The remaining regions can be subdivided into smaller subregions (zones), and a grid in each zone can be generated based on the previous zones. For example, the interior grid of zone 1 can be generated by using lines DE, a line parallel to DE that is located on the wing, AB, AD, and the normal derivative at AB. Line BE is computed as part of the solution. Zones 2 and 3 can be generated in the same way. The grid in zone 4 can be generated by using lines GJLN, a line below GJLN, GH, a line on the wing parallel to GH, and line NO. Then, the interface (HO) is computed as part of the solution. By using the interface (GJLN) and the line below it, the grid lines are C^1 continuous across the interface. Lastly, the grid for zone 5 can be generated by using lines at the interface (BEHO), a line below BEHO, OP, BC, and the derivative at BC. Therefore, it is possible to generate a grid that is C^1 continuous at the interfaces without specifying the interfaces or their derivatives. This procedure can be extended for the volume grid generation.

A general-purpose subroutine is written for Eqs. (3) and (7), which can be found in Ref. 10. Some results of this procedure are shown in Figs. 3 and 4, and a nonphysical surface can be generated in a similar fashion. The outer boundary and the outflow boundary are shown in Figs. 5 and 6, respectively. The solid lines show the grid on the solid surfaces; and the dotted lines show the grid on the nonphysical boundaries.

Interior Generation Procedure

In general, decomposition of the physical domain produces several blocks. Each block is usually defined by six sides, and each side can be defined by either a surface, plane, line, or a point. If one side of a block collapses to a line or a point, then there would be a singularity in the block. In some instances, a block may have been defined by less than six surfaces. Once the surfaces are defined, the interior grid can be computed by any standard grid-generation technique. In this study, a transfinite interpolation has been used to generate the interior grid points.

Once the boundary surfaces $[F(\xi,\eta,\zeta)]$ are known, it is possible to generate the interior grid by a transfinite interpolation. The three-step transfinite interpolation can be expressed for the vector $F(\xi,\eta,\zeta)$ as

$$F(\xi,\eta,\zeta) = \{x(\xi,\eta,\zeta),\ y(\xi,\eta,\zeta),\ z(\xi,\eta,\zeta)\}^T \tag{8a}$$

$$F_1(\xi,\eta,\zeta) = \sum_{l=1}^{L}\sum_{n=0}^{P} \alpha_l^{(n)}(\xi) A_l^{(n)}(\eta,\zeta) \tag{8b}$$

$$F_2(\xi,\eta,\zeta) = F_1(\xi,\eta,\zeta) + \sum_{l=1}^{M}\sum_{n=0}^{Q} \beta_l^{(n)}(\eta) \times \left[B_l^{(n)}(\xi,\zeta) - \frac{\partial^n F_1}{\partial \eta^n}(\xi,\eta_l,\zeta) \right] \tag{8c}$$

$$F(\xi,\eta,\zeta) = F_2(\xi,\eta,\zeta) + \sum_{l=1}^{N}\sum_{n=0}^{R} \gamma_l^{(n)}(\zeta) \times \left[C_l^{(n)}(\xi,\eta) - \frac{\partial^n F_2}{\partial \zeta^n}(\xi,\eta,\zeta_l) \right] \tag{8d}$$

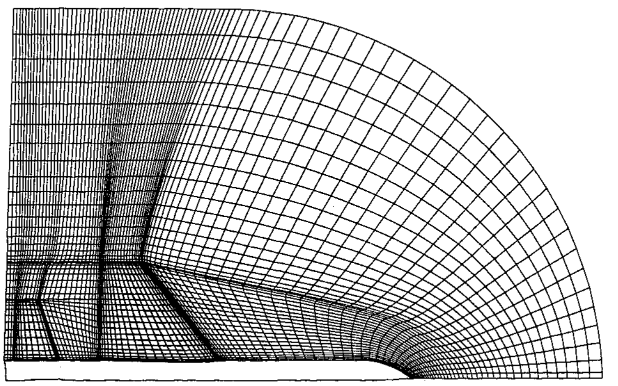

Fig. 4 Grid on symmetry surface (x-y plane).

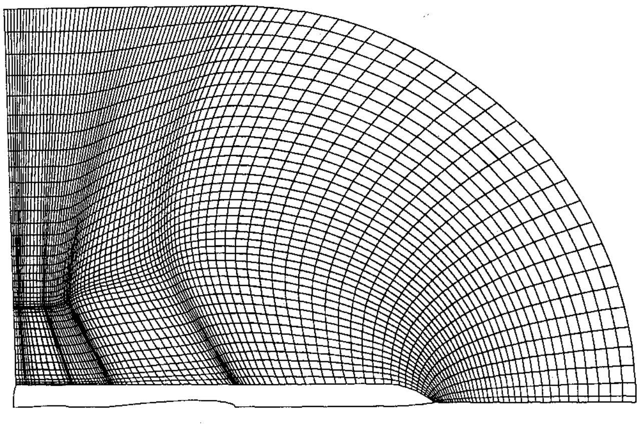

Fig. 5 Grid on symmetry surface (x-z plane).

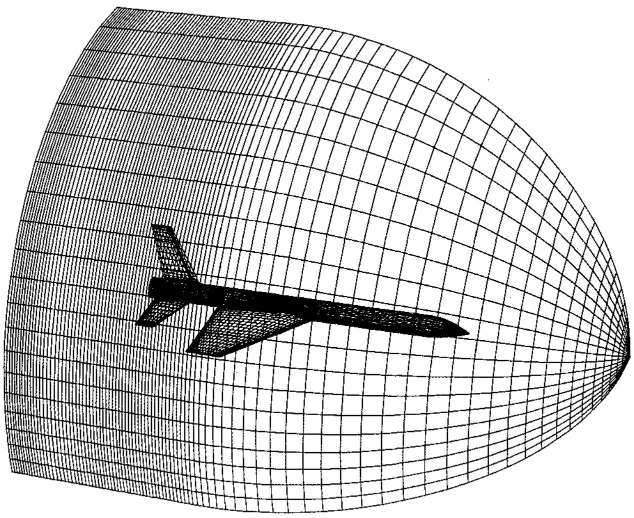

Fig. 6 Grid on outer boundary.

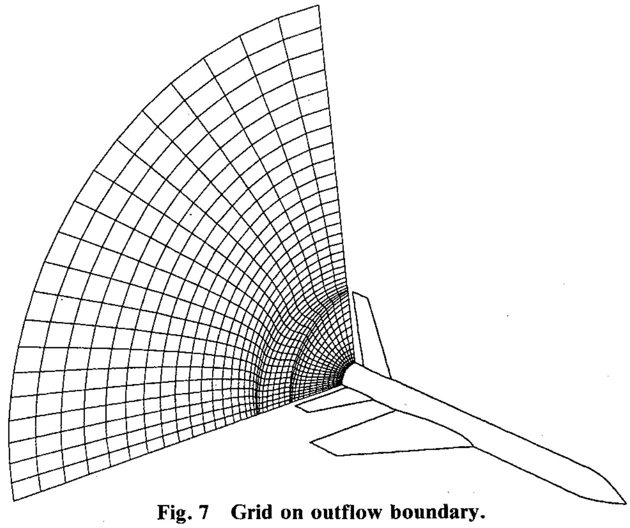

Fig. 7 Grid on outflow boundary.

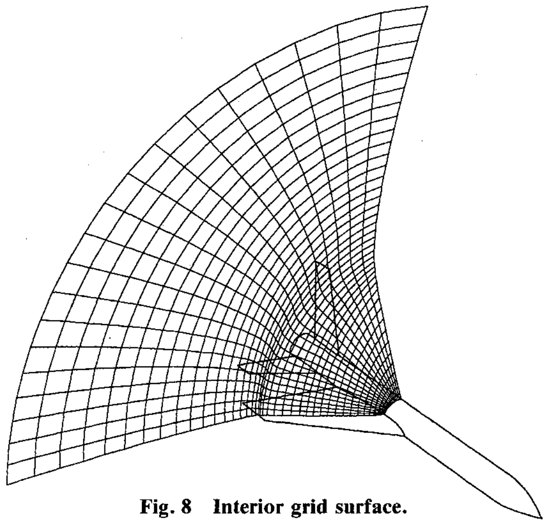

Fig. 8 Interior grid surface.

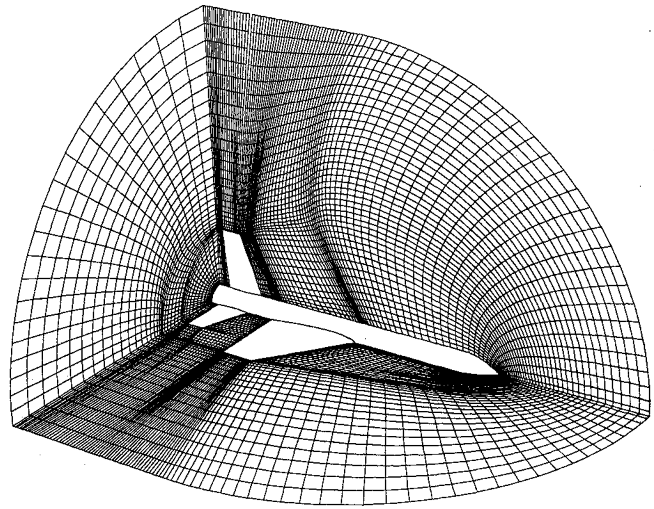

Fig. 9 Grid on physical and nonphysical surfaces.

where $A_l^{(n)}$, $B_l^{(n)}$, and $C_l^{(n)}$ are the known surface locations and their derivatives.

In this study, the interior grid points are generated based on the definition of six surfaces, and the derivatives at the boundary are not included. Equations are used for the blending functions in all directions. The method described here has been used to write a computer code capable of generating a grid over an airplane with fuselage, wing, and tails. Some of these results are shown in Figs. 7–9.

Results and Conclusions

A computer program has been developed to generate a multiblock grid around an airplane geometry. The technique is based on a three-dimensional transfinite interpolation with a Lagrangian interpolation function for the blending functions. By using a Lagrangian interpolation, it is possible to enforce continuity across the interfaces without the derivative information. This procedure is proven to be very simple and effective. It is also possible to control the grid spacing by using a monotonic rational quadratic spline interpolation.

Acknowledgments

The authors would like to thank Joan I. Pitts of NASA Langley (Computer Application Branch) for her technical assistance. The first, second, and fourth authors were partially supported by a grant from NASA Langley Research Center (NCCI-68). The authors would also like to thank Dr. Joseph Shang of Wright Research and Development Center (Flight Dynamics Laboratory) for initiating this investigation.

References

[1]Abolhassani, J. S., and Smith, R. E., "Three-Dimensional Grid Generation About a Submarine," *Numerical Grid Generation in Computational Fluid Mechanics,* edited by S. Sengupta et al., Pineridge, Swansea, England, 1988, pp. 505–515.

[2]Smith, R. E., and Wiese, M. R., "Interactive Algebraic Grid Generation," NASA TP-2533, March 1986.

[3]Reinsch, C. H., "Smoothing by Spline Functions," *Numerical Mathematics*, Vol. 10, No. 3, 1967, pp. 177–183.

[4]Gregory, J. A., and Delbourgo, R., "Piecewise Rational Quadratic Interpolation to Monotone Data," *IMA Journal of Numerical Analysis*, Vol. 2, April 1982, pp. 123–130.

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

[6]Gordon, W. J., and Hall, C. A., "Construction of Curvilinear Coordinate Systems and Applications to Mesh Generation," *International Journal of Numerical Methods in Engineering*, Vol. 7, No. 4, 1973, pp. 461–477.

[7]Smith, R. E., and Eriksson, L.-E., "Algebraic Grid Generation," *Computer Methods in Applied Mechanics and Engineering*, Vol. 64, Oct. 1987, pp. 285–300.

[8]Eiseman, P. R., "Grid Generation for Fluid Mechanics Computations," *Annual Review of Fluid Mechanics*, Vol. 17, 1985, pp. 487–522.

[9]Eriksson, L. E., "Practical Three-Dimensional Mesh Generation Using Transfinite Interpolation," *SIAM Journal of Scientific and Statistical Computations,* Vol. 6, No. 3, July 1985, pp. 712–741.

[10]Abolhassani, J. S., Sadrehaghighi, I., Smith, R. E., and Tiwari, S. N., "Applications of Lagrangian Blending Functions for Grid Generation Around Airplane Geometries," AIAA Paper 90-0009, Jan. 1990.