NASA CONTRACTOR REPORT 166523

Generation of Orthogonal Grids By Boundary Grid Relaxation

S. Nakamura

CONTRACT NAS2– 0R565-101
September 1983

NASA

NASA CONTRACTOR REPORT 166523

Generation of Orthogonal Grids By Boundary Grid Relaxation

S. Nakamura
Department of Mechanical Engineering
The Ohio State University
Columbus, Ohio

Prepared for
Ames Research Center
under Contract NAS2-OR565-101

National Aeronautics and
Space Administration

Ames Research Center
Moffett Field, California 94035

GENERATION OF ORTHOGONAL GRIDS BY BOUNDARY GRID RELAXATION

S. NAKAMURA

The Ohio State University
Mechanical Engineering Department
206 W. 18th Avenue
Columbus, Ohio 43210

Abstract: Two elliptic grid generation schemes that yield orthogonal grids are presented (FGBR and UBGR). Each scheme is different from the original elliptic grid generation scheme developed by Thompson, et al., in the treatment of boundary points. With the UBGR scheme, the grid points on the flow boundaries are automatically determined by the algorithm, while with the FBGR scheme at most one half of the boundary grid points may be prespecified and the remainder of boundary grid points are determined automatically. Numerical examples show their capability of easy stretching, clustering and shock fitting while maintaining orthogonality of grid. The present method can be implemented into existing elliptic grid generation programs with relatively minor modifications.

§1. Introduction

The elliptic grid generation scheme developed by Thompson, et al.[1], has been regarded as a method to generate non-orthogonal grids. However, approximate orthogonality of grids in the vicinity of flow boundary may be achieved by the algorithm devised by Steger and Sorenson [2].

Analytical conformal mapping [3], or its numerical version, generates orthogonal grids. The advantages of completely orthogonal grids are (a) the difference equations on the transformed coordinates are simpler than on non-orthogonal grids, and thus both computing time and memory space may be saved, (b) accuracy of the difference equations on orthogonal grids is better than on non-orthogonal girds, (c) if the computational domain is rectangular, a fast direct solver may be applicable in place of iterative schemes.

The most fundamental difference between the conformal mapping and the elliptic grid generation method is that the grids on the flow boundary are prespecified in the elliptic grid generation, while in the former the grids on the boundary are determined by the confomal mapping.

The objective of the present paper is to show that orthogonal grids may be generated by using the elliptic grid generation equations by automatically determining at least one half of the boundary grid points as part of the grid generation. The present method is more versatile than conformal mapping because (a) clustering and stretching of grids are easy, (b) adapting to the flow solution is possible, (c) orthogonal grids may be generated even when one half of the boundary grid points are fixed, and (d) the algorithm may be extended to control orthogonality of grids in three-dimensional elliptic grid generation.

The coefficient of the cross derivative term in the elliptic partial differential equations on the computational domain, which will be called the B-coefficient, becomes zero if the transformation is orthogonal. The simplification of the flow equations on orthogonal grids is due to this fact. However, an orthogonal coordinate transformation does not necessarily mean that the B-coefficient in numerical computations is exactly zero [4]. This is because even if the coordinate transformation is analytically orthogonal, the B-coefficient in numerical computations is calculated by finite difference approximation. Therefore, from the view point of numerical computations, it is desirable to have the numerically calculated B-coefficient exactly zero, than to have analytically orthogonal grids as the case of conformal mapping. In this paper we use the term "numerical orthogonality" meaning that the B-coefficient calculated numerically is zero. Although the grids generated by the proposed shemes are not numerically orthogonal in this sense, but are rather approximation to analytically orthogonal grids. However, feasibility of generating numerically orthogonal grids by using elliptic systems is discussed later.

In the remainder of this paper, (1) the elliptic PDEs and the boundary conditions to generate analytically orthogonal grids are derived from orthogonality conditions, (2) the elliptic equations and boundary conditions are transformed onto the computational domain, (3) difference equations for the elliptic grid generation equations and boundary conditions are derived, (4) grids generated by the present method for a number of geometries are illustrated, and finally (5) feasibility of generating numerically orthogonal grid is discussed.

§2. Elliptic Partial Differential Equations and Boundary Conditions for Orthogonal Coordinate Transformations

Denoting x-y and ξ-η as physical and computational coordinates, respec-

tively, let us consider two families of curves, $\xi(x,y) = c$ and $\eta(x,y) = d$, on the physical domain as shown in Fig. 1, where c and d are parameters. We now determine the conditions for the two families of curves to become orthogonal. Along a curve, $\xi = c$, we have

$$d\xi = \xi_x dx + \xi_y dy = 0 \tag{1}$$

or equivalently,

$$(dy/dx)_1 = -\xi_x/\xi_y \tag{2}$$

where subscript 1 indicates that the derivative is the slope of $\xi = c$ on the x-y plane. Similarly, the slope of the curve $\eta = d$ is written as

$$(dy/dx)_2 = -\eta_x/\eta_y \tag{3}$$

The orthogonality between $\xi = c$ and $\eta = d$ requires that Eq.(2) be the negative reciprocal of Eq.(3), that is,

$$\frac{\eta_x}{\xi_y} = -\frac{\eta_y}{\xi_x} = ak(x,y) \tag{4}$$

where a is a constant, and $k(x,y)$ is a continuous function which may be arbitrarily specified. Equation (4) may be equivalently written as

$$\begin{aligned} \eta_x &= ak\xi_y \\ \eta_y &= -ak\xi_x \end{aligned} \tag{5}$$

The transformation between x-y and ξ-η is conformal if $ak = 1$. Otherwise the transformation is orthogonal but not conformal.

Equation (5) is a sufficient condition for orthogonality, but it is not suitable in determining $\xi(x,y)$ and $\eta(x,y)$. To find more appopriate equations, we eliminate ξ or η from Eq. (5) (one at a time) and obtain elliptic PDEs as follows

$$\begin{aligned} (k\xi_x)_x + (k\xi_y)_y &= 0 \\ (\frac{1}{k}\eta_x)_x + (\frac{1}{k}\eta_y)_y &= 0 \end{aligned} \tag{6}$$

Notice that the above equations do not involve the constant a. Equation (6) is a convenient form because the equations are elliptic PDEs, for which there are many solution methods available. Unfortunately, Eq. (6) is not a sufficient condition for orthogonality any more, but is only a necessary condition. This is because, while Eq. (6) is derived from Eq. (5), the reverse is not possible without additional conditions. However, the solution of Eq. (6) are uniquely determined when boundary conditions are specified. In fact, the orthogonal set of ξ and η are determined with appropriate boundary conditions.

In order to derive boundary conditions rigorously, we write the following theorem:

[Theorem] Suppose $\xi(x, y)$ and $\eta(x, y)$ are smooth functions on the x-y plane. In order that the family of curves $\eta(x, y) = d$ are orthogonal to a curve $\xi(x, y) = c$, the partial derivative of η normal to the curve $\xi = c$ must be zero.

[Proof] The gradient of the curve $\xi = c$ is given by Eq.(1). The derivative of $\eta(x, y)$ normal to $\xi = c$ may be written as

$$\partial\eta(x, y)/\partial n = \eta_x l_x + \eta_y l_y \tag{7}$$

where l_x and l_y are directional cosines of the outward normal to $\xi = c$. Referring to Fig. 2, the gradient of $\xi = c$ is related to the directional cosines as

$$(dy/dx)_1 = -l_x/l_y \tag{8}$$

Eliminating $(dy/dx)_1$ with Eq.(2) and a slight rewriting yields

$$l_y/\xi_y = l_x/\xi_x \tag{9}$$

Notice that, if l_y approaches zero, ξ_y also approaches zero ($\xi = c$ becomes vertical) while l_x and ξ_x are both nonzero. Therefore, even when l_y approaches zero, the left side of Eq.(9) remains nonzero and nonsingular.

Introducing Eq.(9) into Eq.(7) yields

$$\partial\eta(x, y)/\partial n = (l_y/\xi_y)(\eta_x\xi_x + \eta_y\xi_y) \tag{10}$$

If the left side of Eq.(10) is set to zero, then,

$$\eta_x \xi_x + \eta_y \xi_y = 0 \tag{11}$$

which is equivalent to Eq.(4). Thus $\xi = c$ and $\eta = d$ are orthogonal.

[Corollary] If a boundary of the x-y domain is represented by a curve $\xi = c$, the derivative of $\eta(x, y)$ normal to $\xi = 0$ must be zero along that part of the boundary. Similarly, if a boundary of the x-y domain is represented by a curve $\eta = d$, the derivative of $\xi(x, y)$ normal to $\eta = d$ must be zero.

The above corollary may be intepreted as that, when the value of ξ (or η) is set to a constant along a portion of the flow boundary, the normal derivative of η (or ξ respectively) must be zero along that portion.

In applying the above rule to grid generation, suppose the whole boundary of the x-y domain is divided into four parts, G_g, $g = 1$ to 4, each of which is transformed respectively to the left, top, right and bottom sides of the rectangular computational domain, $0 < \xi < \xi_{max}$ and $0 < \eta < \eta_{max}$ (see Fig.3a). Thus, the boundary values of ξ and η are specified as

$$\begin{aligned} &\xi = 0 \quad \text{along} \quad G_1 \\ &\eta = \eta_{max} \quad \text{along} \quad G_2 \\ &\xi = \xi_{max} \quad \text{along} \quad G_3 \\ &\eta = 0 \quad \text{along} \quad G_4 \end{aligned} \tag{12}$$

The additional boundary conditions are set in accordance with the corollary as follows:

$$\begin{aligned} &\partial\eta(x, y)/\partial n = 0 \quad \text{along} \quad G_1 \quad \text{and} \quad G_3 \\ &\partial\xi(x, y)/\partial n = 0 \quad \text{along} \quad G_2 \quad \text{and} \quad G_4 \end{aligned} \tag{13}$$

In conclusion, an orthogonal tranformation of the x-y domain to the ξ-η domain is completed if Eq. (6) with bounday conditions Eqs.(12) and (13) are solved.

§3. Derivation of Grid Generation Equations

The grid generation equations that map the uniformaly spaced Cartesian grids on the computational domain onto the physical domain are obtained

by inverting Eq.(6) and the boundary conditions, Eqs.(12)-(13), in accordance with

$$\begin{aligned} \xi_x &= \quad y_\eta / J \\ \xi_y &= -\xi_\eta / J \\ \eta_x &= -y_\xi / J \\ \eta_y &= \quad x_\xi / J \end{aligned} \tag{14}$$

where J is the Jacobian given by

$$J = x_\xi y_\eta - x_\eta y_\xi$$

Equation (6) is transformed to

$$\begin{aligned} x_\eta Ly - y_\eta Lx &= -(J/k)[y_\eta^2 k_\xi - y_\xi y_\eta k_\eta + x_\eta^2 k_\xi - x_\xi x_\eta k_\eta] \\ x_\xi Ly - y_\xi Lx &= -(J/k)[y_\xi^2 k_\eta - y_\eta y_\xi k_\xi + x_\xi^2 k_\eta - x_\eta x_\xi k_\xi] \end{aligned} \tag{15}$$

In the above equations, L is a quasi-linear elliptic partial differential operator given by

$$L = A\frac{\partial^2}{\partial \xi^2} - 2B\frac{\partial^2}{\partial \xi \partial \eta} + C\frac{\partial^2}{\partial \eta^2} \tag{16}$$

where

$$\begin{aligned} A &= \xi_\eta^2 + y_\eta^2 \\ B &= x_\xi x_\eta + y_\xi y_\eta \\ C &= x_\xi^2 + y_\xi^2 \end{aligned}$$

Separating Ly and Lx in Eq.(15) yields

$$\begin{aligned} Lx &= [\alpha\gamma + 2x_\xi x_\eta \beta] \\ Ly &= [\beta\gamma + 2y_\xi y_\eta \alpha] \end{aligned} \tag{17}$$

where

$$\begin{aligned} \alpha &= (y_\eta k_\xi - y_\xi k_\eta) \\ \beta &= (x_\eta k_\xi - x_\xi k_\eta) \\ \gamma &= (x_\xi y_\eta + x_\eta y_\xi) \end{aligned}$$

The boundary conditions, Eq.(12), are implemented on the computational domain as follows: First, define

$$f_g(x,y)=0 \quad \text{on} \quad F_g, \quad g=1,2,3 \quad \text{and} \quad 4 \tag{18}$$

where F_g , g = 1, 2, 3 and 4, are the left, top, right and bottom boundaries of the computational domain respectively, and $f_g = 0$ is the shape of the corresponding boundary on the x-y domain. By using Eqs.(7) and and (14), Eq.(13)becomes

$$\begin{aligned} -y_\xi l_x + x_\xi l_y &= 0 \quad \text{along} \quad G_1(\xi=0) \quad \text{and} \quad G_3(\xi=\xi_{max}) \\ y_\eta l_x - x_\eta l_y &= 0 \quad \text{along} \quad G_2(\eta=\eta_{max}) \quad \text{and} \quad G_4(\eta=0) \end{aligned} \tag{19}$$

where l_x and l_y are directional cosines of the outward normal on the boundary.

A central difference scheme is used to discretize Eq.(17) to yield

$$\begin{aligned} &A(x_{i-1,j}-2x_{i,j}+x_{i+1,j})+C(x_{i,j-1}-2x_{i,j}+x_{i,j+1}) \\ &\quad = B(x_{i+1,j+1}-x_{i-1,j+1}-x_{i+1,j-1}+x_{i-1,j-1})/2+R^{(x)}_{i,j} \\ &\quad \equiv F^{(x)}_{i,j} \\ &A(y_{i-1,j}-2y_{i,j}+y_{i+1,j})+C(y_{i,j-1}-2y_{i,j}+y_{i,j+1}) \\ &\quad = B(y_{i+1,j+1}-y_{i-1,j+1}-y_{i+1,j-1}+y_{i-1,j-1})/2+R^{(y)}_{i,j} \\ &\quad \equiv F^{(y)}_{i,j} \end{aligned} \tag{20}$$

where $R^{(x)}_{i,j}$ and $R^{(y)}_{i,j}$ represent the difference approximations for the right sides of Eq.(17).

Equation (20) is solved iteratively. Any of the iterative methods used for existing elliptic grid generation scheme should work for the present method.

§4. Undetermined Boundary Grid Relaxation (UBGR) scheme

The numerical treatment of boundary conditions is different in the two schemes proposed: UBGR and FBGR. In the UBGR scheme, all the grid

points on the boundary are undetermined, while up to one half of the boundary grid points may be prespecified in the FBGR scheme and the remaining boundary grid points are undetermined.

Detail of the treatment of the boundary conditions for the UBGR scheme is described in the remainder of this section. The boundary conditions for the FBGR scheme are explained in a later section since it is an extension of the UBGR scheme.

For internal grid points, the two equations in Eq.(20) detemine $x_{i,j}$ and $y_{i,j}$, provided that the coordinates of the surrounding eight grid points are given. Of course the coordinates of the surrounding grid points are iteratively revised in every iteration step. However, assuming temporarily that the coordinates of the surrounding grid points are known simplifies explanation of the derivation of the difference equations for the boundary grid points.

For a grid point on the boundary, the two equations in Eq.(20) can not be satisfied simultanously because of the following additional constraints: (1) boundary grid must be on the specified boundary line, and (2) orthogonality of grid lines at the bounday is required. Althogh the othogonality requirement can be incorporated into the elliptic equation as a bounday condition, the first constraint does not allow the two elliptic equations to be independent. This means that only one of the two equations can be used unless the two equations are linearly dependent.

In order to find an appropriate equation, consider a boundary grid point (i, j) to be determined on a boundary as depicted in Fig. 4a. We assume that the three surrounding grid points denoted by L, R, and B are temporarily fixed. The variables s and t are local Cartesian coordinates as shown in Fig. 4b. The elliptic equations for x and y may be written in terms of s and t as

$$As_{\xi\xi} - 2Bs_{\xi\eta} + Cs_{\eta\eta} = R^{(s)} \tag{21a}$$

$$At_{\xi\xi} - 2Bt_{\xi\eta} + Ct_{\eta\eta} = R^{(t)} \tag{21b}$$

Since Eqs.(21a) and (21b) are independent, one has to choose only one of the two equations. We choose Eq.(21a) and abandon (21b) for the following reasons: The coordinate t is in the normal direction from the boundary so $t_{i,j}$ on the boundary is not allowed to change. On the other hand, s is tangent to the boundary so $s_{i,j}$ has a freedom to change.

The difference form of the first and third terms of Eq.(21a) may be written as

$$s_{\xi\xi} = s_{i-1,j} - 2s_{i,j} + s_{i+1,j} \tag{22a}$$

$$s_{\eta\eta} = \frac{[s_\eta]_{i,j} - (s_{i,j} - s_{i,j-1})}{1/2} \tag{22b}$$

The orthogonality boundary condition requires that the first term on the right side of Eq.(22b) be zero,

$$[s_\eta]_{i,j} = 0 \tag{23}$$

By introducing Eq.(22) into Eq.(21a) and using Eq.(23), we obtain

$$A(s_{i-1,j} - 2s_{i,j} + s_{i+1,j}) + 2C(-s_{i,j} + s_{i,j+1}) = R^{(s)}_{i,j} \tag{24}$$

where B in Eq.(21a) is set to zero because of orthogonality at the boundary. If the coordinates of the surrounding grid points are known, $s_{i,j}$ is found by solving the above equation. Then $t_{i,j}$ is calculated by introducing the value of $s_{i,j}$ into the equation that defines the boundary,

$$f(s,t) = 0 \tag{25}$$

§5. Control of Grid Spacing

Control of grid spacing for clustering and stretching is achieved by the following two algorithms: (1) variable grid spacing on the computational grids, and (2) space dependent coefficient, $k(x,y)$.

In the first approach, the Cartesian grid points with variable spacing on the computational domain are mapped onto the physical domain as illustrated in Fig. 5. Small grid spacing on a part of the physical domain is obtained by imposing a small grid spacing on the computational grid in the corresponding area. This technique may be used on the ξ and η directions independently. Once the coordinates of the grid points on the physical domain are determined, the grid may used as if it were generated for an uniformely spaced computational grid.

The second approach is to change k in space. Since k in Eq.(4) is propotional to the aspect ratio of a grid cell, the clustering effect in the ξ-direction, for example, is obtained by reducing the value of k in the interested area on the physical domain. While the first approach can only change the grid spacing uniformly in the ξ or η direction, the second approach can cluster or stretch grid more flexibly on the physical domain. Uses of $k(x, y)$ are illustrated later with numerical examples.

In addition, these two techniques may be applied simultaneously in one grid generation as needed.

§6. Fixed Boundary Grid Relaxation (FBGR) Scheme

In the UBGR scheme two end points of a grid line are undetermined so both move along the physical boundary during iterations, while the grid line on the computational domain is fixed. Another way of satisfying the orthogonal grid generation equations is that: (a) one end point of a grid line is fixed, (b) the other end point of the grid line is left to be variable, and (c) grid spacings adjacent to the corresponding grid line on the computational domain are variable. This latter approach is denoted as the FBGR scheme.

This scheme can be implemented by modifying the UBGR scheme only in the treatment of the fixed boundary grid points as explained next.

Suppose that the grid points with black circles in Fig. 6 are obtained after an iteration step with a given set of grid spacings $\delta\xi_i \equiv \xi_i - \xi_{i-1}$ and $\delta\xi_{i+1} \equiv \xi_{i+1} - \xi_i$. First, the hypothetical coodinates to satisfy Eq. (24) are calculated (open circle marked as B') as if the grid point (i, j) on the boundary is an undetermined point. Second, the following two chord length ratios are calculated:

$$\begin{aligned} r' &= AB'/(AB' + B'C) \\ r &= AB/(AB + BC) \end{aligned} \tag{26}$$

where AB, BC, AB' and $B'C$ are chord lengths. Third, the computational grid spacings $\delta\xi_i$ and $\delta\xi_{i+1}$ are replaced by

$$\delta\xi'_i = ar^2 + br \tag{27}$$

$$\delta\xi'_{i+1} = (\delta\xi_i + \delta\xi_{i+1}) - \delta\xi'_i \tag{28}$$

where

$$a = \delta\xi_i/r' - \delta\xi_{i+1}/(r' - 1)$$

and

$$b = (\delta\xi_i + \delta\xi_{i+1}) - a$$

The geometrical interpretations of the above equations are :

(i) If $\delta\xi_i$ and $\delta\xi_{i+1}$ are the computational grid spacings, the orthogonal boundary condition requires that the boundary grid must be at B'.

(ii) If the boundary grid is to be located at B while satisfying the orthogonality boundary conditions, the computational grid line i must move toward line $i - 1$.

(iii) The functional relation among $\delta\xi_i$, $\delta\xi_{i+1}$ and r' are fitted by a quadratic polynomial, and then the revised value of the grid spacing $\delta\xi_i$ is calculated by the quadratic function thus obtained.

(iv) The total computational grid spacing between $(i - 1)$ and $(i + 1)$ is unchanged through this modification, namely

$$\delta\xi_i + \delta\xi_{i+1} = \delta\xi_i' + \delta\xi_{i+1}' \tag{29}$$

The grid spacings next to all the fixed boundary grid points are modified after each iteration step.

§7. Numerical Illustrations

Example 1:

Figure 7 shows the grid generated by the present method for a quadrilateral domain. An uniformly spaced Cartesian grid on the computational domain is mapped to the the physical domain by the UBGR method with $k = 1$.

Example 2:

Figure 8 shows the grid generated for a NACA-0012 airfoil by the UBGR scheme with $k = 1$ and no clustering. The grid is not, however, orthogonal in the vicinity of the trailing edge since the two grid points at the trailing edge are fixed without any FBGR treatment.

Example 3:

Figure 9 shows the grid generated for a NACA-0012 airfoil by the UBGR schem with a clustering effect created by a variable grid spacing on the intermediate computational domain.

Example 4:

Figure 10 shows the grid points clustered around a prescribed curve to illustrate the capability of solution adapting. This was done by setting $k(x, y)$ in Eq.(4), or equivalently Eq.(18), as

$$k(x, y) = c_1 + \frac{c_2}{1 + [h(x, y)]^2}$$

where c_1 and c_2 are positive constants and $h(x, y) = 0$ is the curve (for example, a shock profile) to which grid points are to be clustered. The function $h(x, y)$ used in the present example is

$$h(x, y) = sin(x) - y$$

Example 5:

Figure 11 shows the C-type grid generated by the UBGR scheme for a cambered airfoil. The grid points on the Kutta cut do not match because the points above and below were determined independently.

Example 6:

Figure 12 shows the C-type grid generated by the FBGR for NACA-0012 airfoil. The grid points on the airfoil surface and the Kutta cut are fixed, while the grid points on the outer boundaries are all determined by the UBGR scheme. The grid points near the trailing edge are clustered for this particular grid but this clustering is not essential to the present grid generation scheme.

Example 7:

Figure 13 is for the same configuration as Example 6 except the airfoil is cambered.

All the grids shown above were calculated by the point successive over relaxation iterative scheme, which was used for simplicity of programming. Although no effort has been paid in the present study to accelerating the convergence rate, the convergence of the iterative scheme is felt to be slow particularly in comparison with the conventional elliptic grid generation scheme with fixed boundary grid points. It seems possible to increase the computa-

tional efficiency for the proposed scheme by the ADI or multigrid method or both.

§8. Numerically Orthogonal Grids

The orthogonal grids generated by the UBGR and VBGR schemes are approximation to analytically orthogonal grids. This is because the basic equations are based on analytically orthogonal transfomations but the finite difference approximation was used to generate grids. In this section the possibility of generating numerically orthogonal grids is analyzed.

The condition of numerical orthogonality may be defined as

$$(\delta_i x)(\delta_j x) + (\delta_i y)(\delta_j y) = 0 \tag{30}$$

where δ_i and δ_j are central difference operators in the i and j directions respectively. Rewriting Eq.(30) yields

$$\delta_i y/\delta_j x = -\delta_i x/\delta_j y = ak(x, y) \tag{31}$$

which may be further rewritten as

$$\begin{aligned} \delta_i y &= \quad ak\delta_j x \\ \delta_i x &= -ak\delta_j y \end{aligned} \tag{32}$$

By rewriting, we obtain

$$\begin{aligned} \delta_i(\frac{1}{k}\delta_i x) + a^2\delta_j(k\delta_j x) &= 0 \\ \delta_i(\frac{1}{k}\delta_i y) + a^2\delta_j(k\delta_j y) &= 0 \end{aligned} \tag{33}$$

The coefficient a in Eq.(33) is eliminated as follows. The two equations in Eq.(32) are added after taking the square of each term, and the resulting equation is solved for a^2 as

$$a^2 = C/(k^2 A) \tag{34}$$

where

$$A = (\delta_j y)^2 + (\delta_j x)^2$$
$$C = (\delta_i x)^2 + (\delta_i y)^2 \tag{35}$$

By introducing Eq.(34), Eq.(33) becomes

$$Ak\delta_i(\frac{1}{k}\delta_i x) + C\frac{1}{k}\delta_j(k\delta_j x) = 0$$
$$Ak\delta_i(\frac{1}{k}\delta_i y) + C\frac{1}{k}\delta_j(k\delta_j y) = 0 \tag{36}$$

If the difference operator is replaced by the differential operator, Eq.(36) becomes equivalent to Eq.(17) except Eq.(36) is missing the B-coefficient. This last point is understandable because Eq.(36) has been derived from the orthogonality condition.

Difference equations are meaningless unless the meaning of the difference operator δ is clearly defined. So we define δ here as the central difference operator associated with one grid interval. Then Eq.(36) is in the well known five point difference form of elliptic partial differential equations. The five points involved are shown with black dots in Fig. 14, which are called here the primary grid points.

With the present definition of δ, however, Eq.(30) may be written more explicitly as

$$(x_{i+1/2,j+1/2} - x_{i-1/2,j+1/2})(x_{i,j+1} - x_{i,j})$$
$$-(y_{i+1/2,j+1/2} - y_{i-1/2,j+1/2})(y_{i,j+1} - y_{i,j}) = 0 \tag{37a}$$

and

$$(x_{i+1,j} - x_{i,j})(x_{i+1/2,j+1/2} - x_{i+1/2,j-1/2})$$
$$-(y_{i+1,j} - y_{i,j})(y_{i+1/2,j+1/2} - y_{i+1/2,j-1/2}) = 0 \tag{37b}$$

where $(i + 1/2, j + 1/2)$ is a grid point in the staggered position as shown with an open circle in Fig. 15. The grid points at the staggered positions are called the auxiliary grid. A system of elliptic difference equations in the same form as Eq.(36) can be derived for the auxiliary grid from Eq.(37). The important thing to be observed here is that Eq.(37) is interpreted as the condition for numerical orthogonality between the primary and auxiliary grids, but not for the numeical orthogonality among the primary grid points or

among the auxiliary gid points.. 5pt Another way of defining the meaning of δ is possible. For example, suppose δ is the central difference operator associated with two grid intervals. Then, the left side of Eq.(30) becomes exactly the central difference approximation of the B-coefficint explained earlier. Therefore, based on this definition the grid generated whould be numerically orthogonal. The problem, however, is that the same five difference equations that generates the primary grid can not be derived any more.

The present observation leads us to the conclusion that, if the grid is generated by the conventional five point difference equation, the B-coefficient calculated numerically by the central difference approximation with two grid intervals can not be made exactly zero.

On the orther hand, there is a possibility that numerically orthogonal staggered grid may be obtained. This will require generating both the primary and auxiliary grids by two sets of elliptic grid generation equations coupled with appropriate boundary conditions. The detail of such a scheme needs further research, however.

§9. Conclusions

The present paper showed that orthogonal grids may be generated by solving a finite difference approximation for the elliptic partial differential equations with either of the UBGR or FBGR schemes. These schemes are more versatile than conformal mapping because (1) the grid cell aspect ratio may be controlled through the function $k(x, y)$, and (2) they can be used for a local control of orthogonality of grid, and (3) the schemes can be extended to three-dimensional grid generation to control orthogonality of grids in one or two directions.

The proposed methods can be implemented in existing grid generation programs which are based on the elliptic grid generation equations with relatively minor effort.

One drawback of the proposed shemes is that the iterative convergence rate is slower than the conventional elliptic grid generation scheme with fixed boundary grid points. The first reason for this is that the boundary conditions are Neuman type in the proposed schemes while the boundary conditions of the conventional elliptic grid generation schemes are all Dirichlet type. The second reason is that the boundary relaxation schemes are nolinear. Since the convergence rate of the proposed schemes is strongly dependent of the

number of grid points, the multigrid method would be useful to increase the computing efficiency.

As an additional product of the present work, it was shown that the grids generated by the five-point central difference approximation to the elliptic PDEs can not be made numerically orthogonal if the B-cefficient is calculated by the central difference approximation associated with two grid intervals, but grids may be made numerically orthogonal between two grid systems that are in the staggered position to each other.

Acknowledgement

The author is indebted to Dr. Terry L. Holst of NASA Ames Research Center for his support and encouragement for this work.

REFERENCES

[1] Thompson, J. F., Thames, F. C. and Mastin, C. W., "Automatic Numerical Generation of Body-Fitted Curvilinear Coordinate System for Field Containing Any Number of Arbitrary Two-Dinmensional Bodies," *J. Comp. Physc.*, Vol. 15 (1974)

[2] Steger, J. L. and Sorenson, R. L., "Automatic Mesh-Point Clustering near a Boundary in Grid Generation with Elliptic Partial Differential Equations," *J. Comp. Physc.*, Vol. 33 (1979)

[3] Ives, D. C., "Conformal Grid Generation," *Numerical Grid Generation*, North-Holland (1982)

[4] Eiseman, P. R., "Orthogonal Grid Generation," *Numerical Grid Generation*, North-Holland (1982)

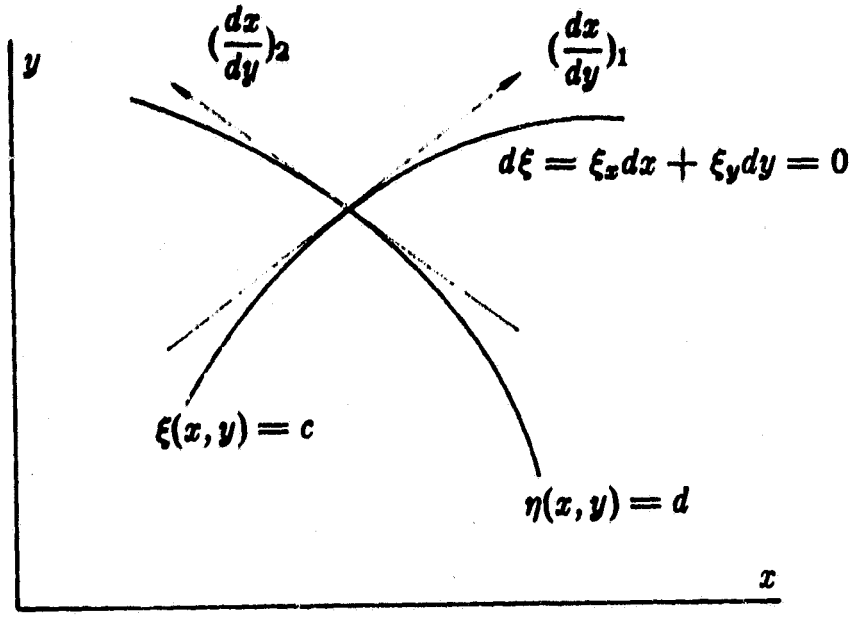


Figure 1 Two families of curves (c and d are parameters)

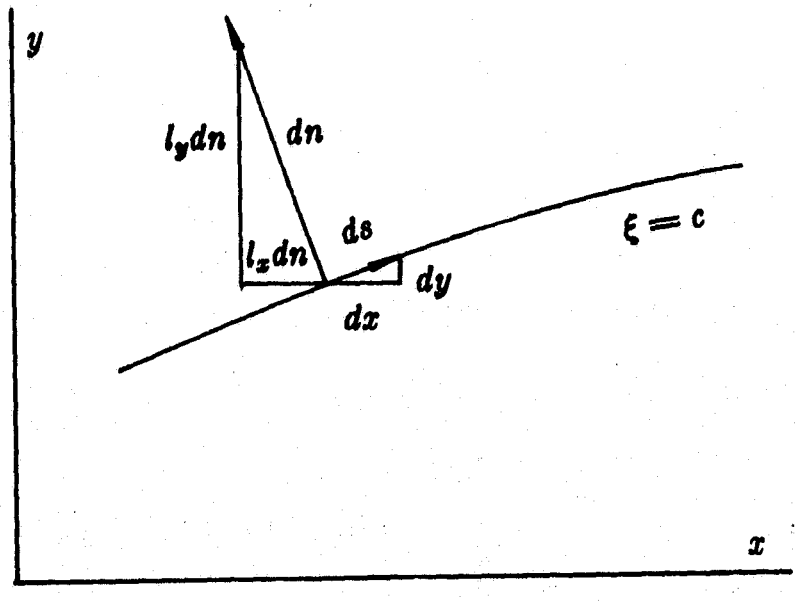


Figure 2 Directional cosines of the outward normal to $\xi = c$

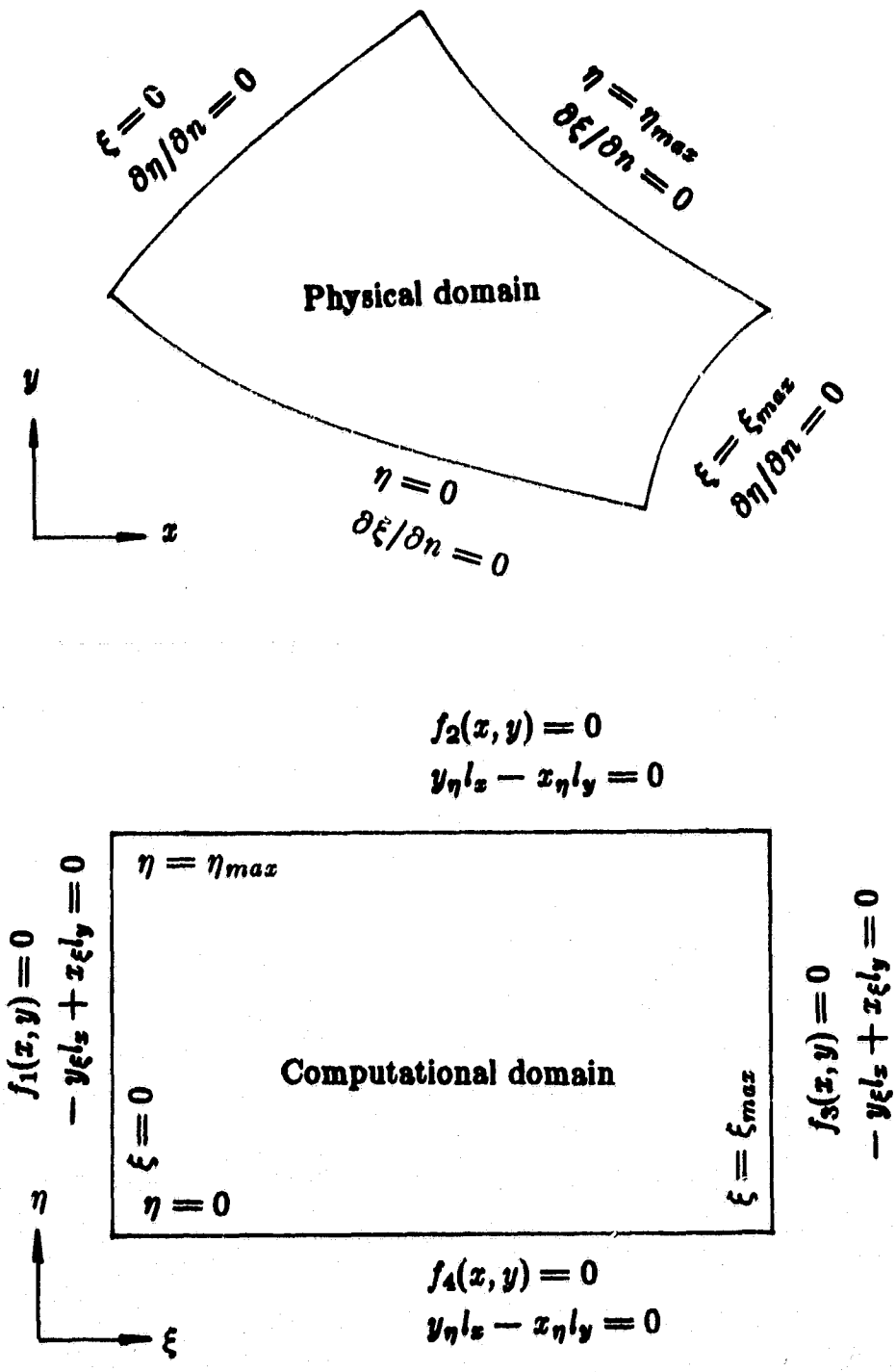


Figure 3 **Mapping of the computational boundaries onto the physical domain and the boundary conditions**

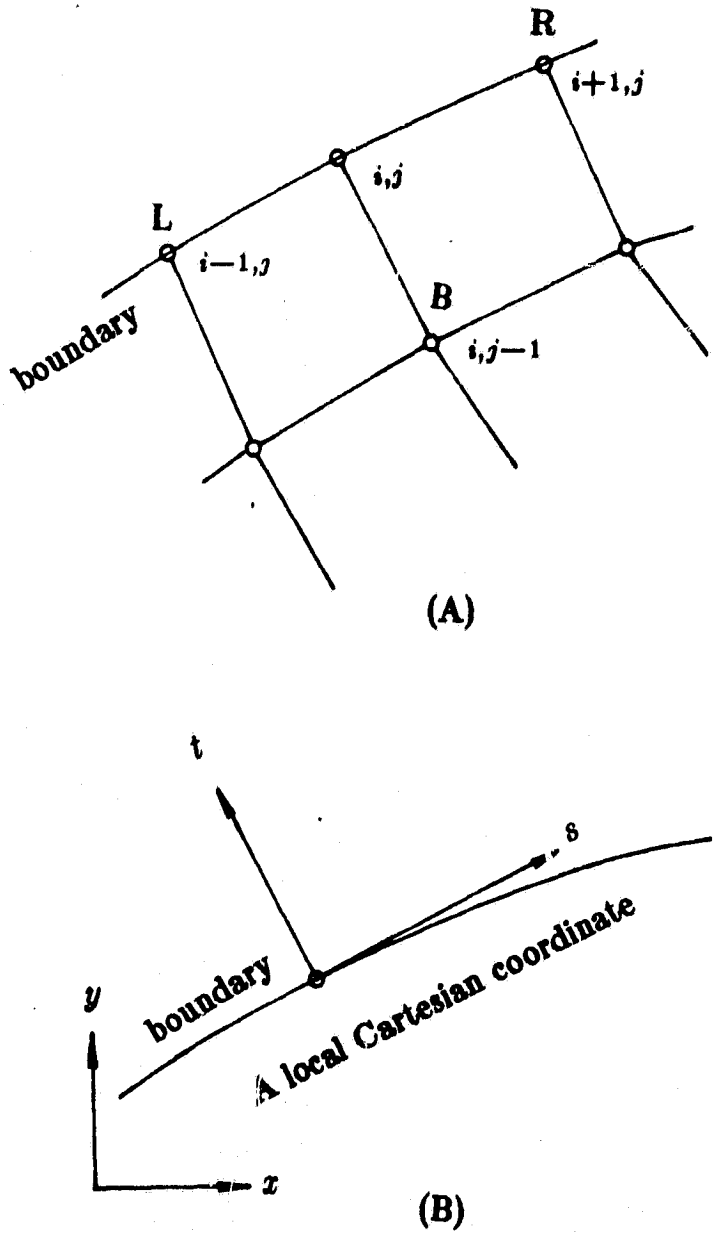


Figure 4 Boundary grid and local Cartesian coordinate

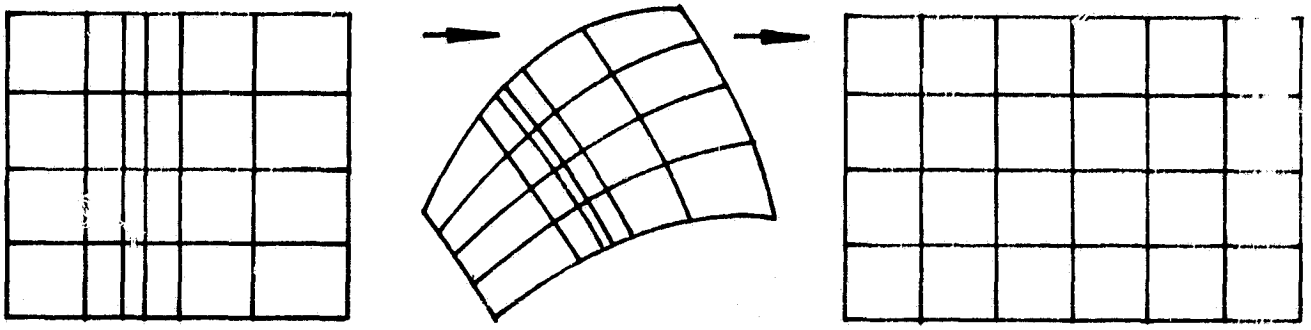


Figure 5 **The effect of variable computational grid spacing on the grids generated**

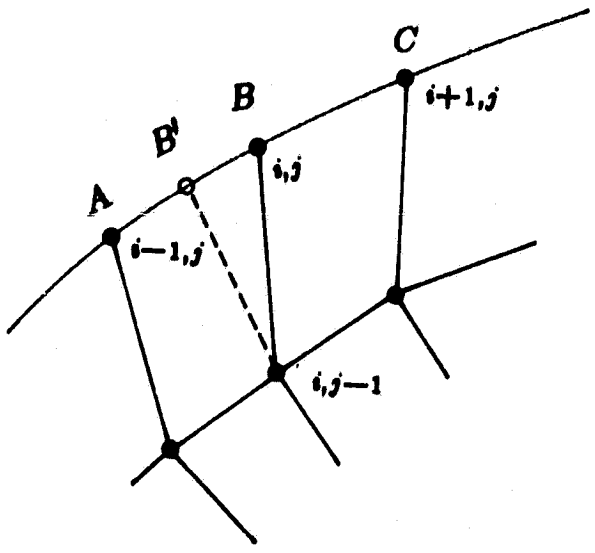


Figure 6 Notations for the VBGR scheme

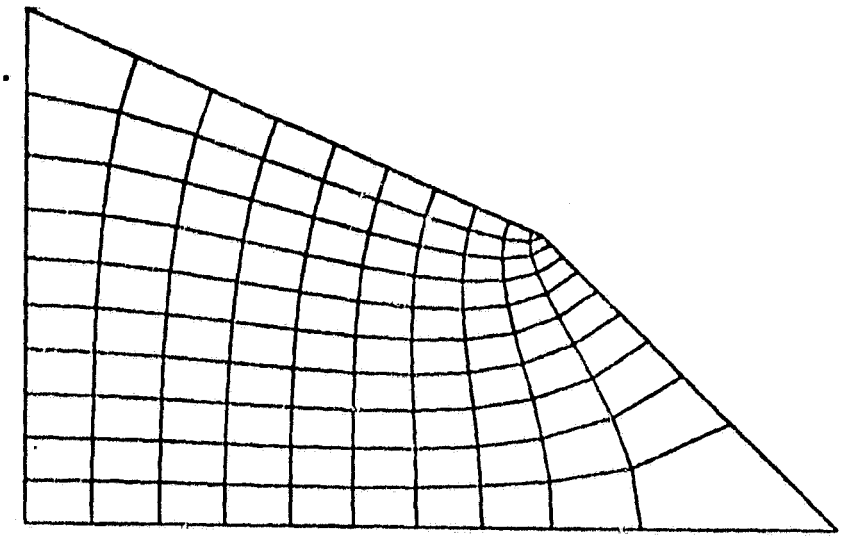


Figure 7 **Orthogonal grid generated for a quadrilateral domain with no grid spacing control**

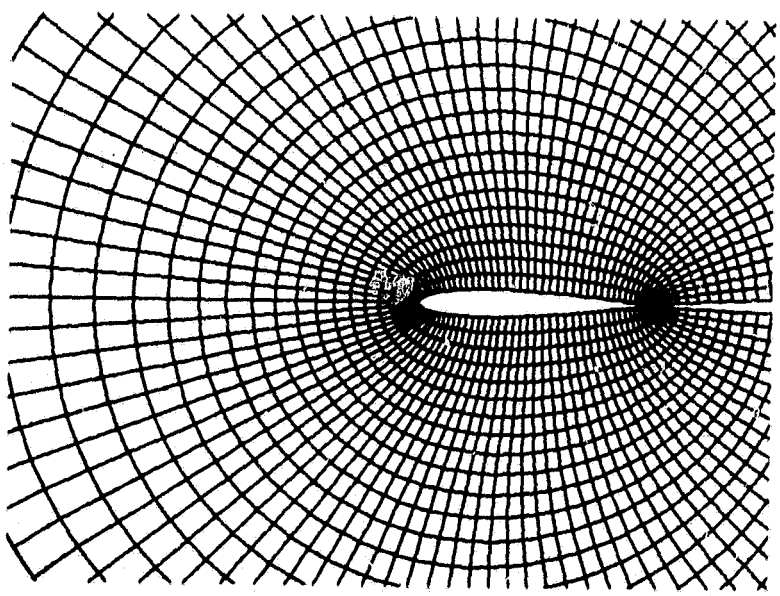


Figure 8 **O-grid for NACA-0012 airfoil generated by the UBGR scheme**

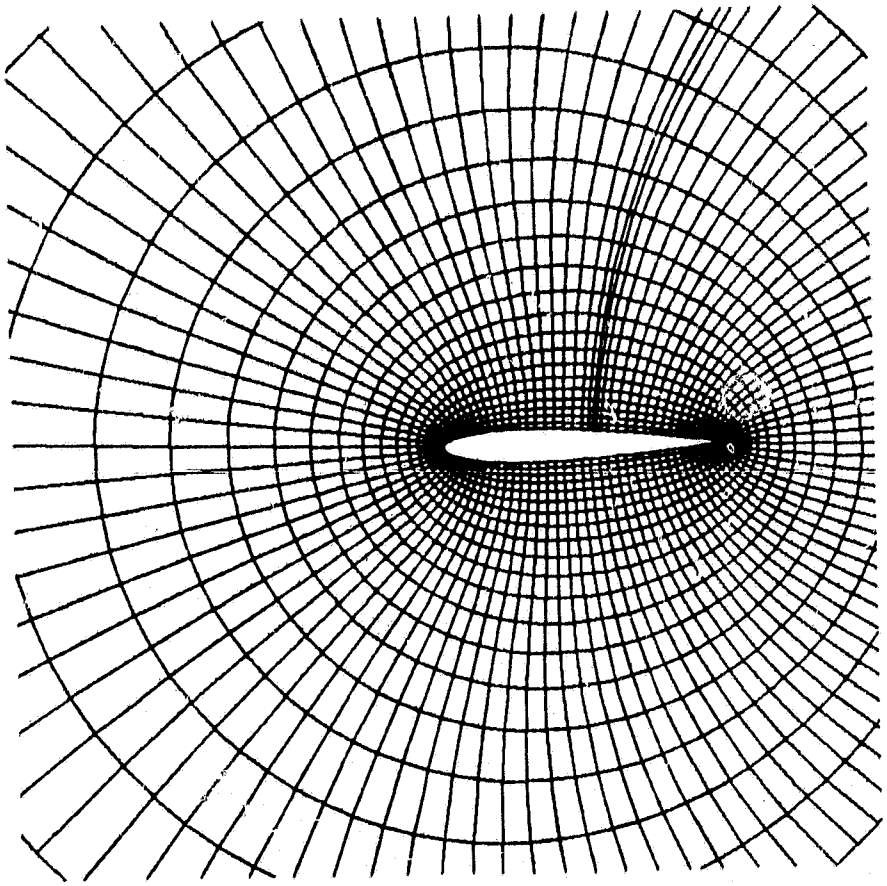


Figure 9 **Illustration of clustering by variable grid spacing on the intermediate compuational domain**

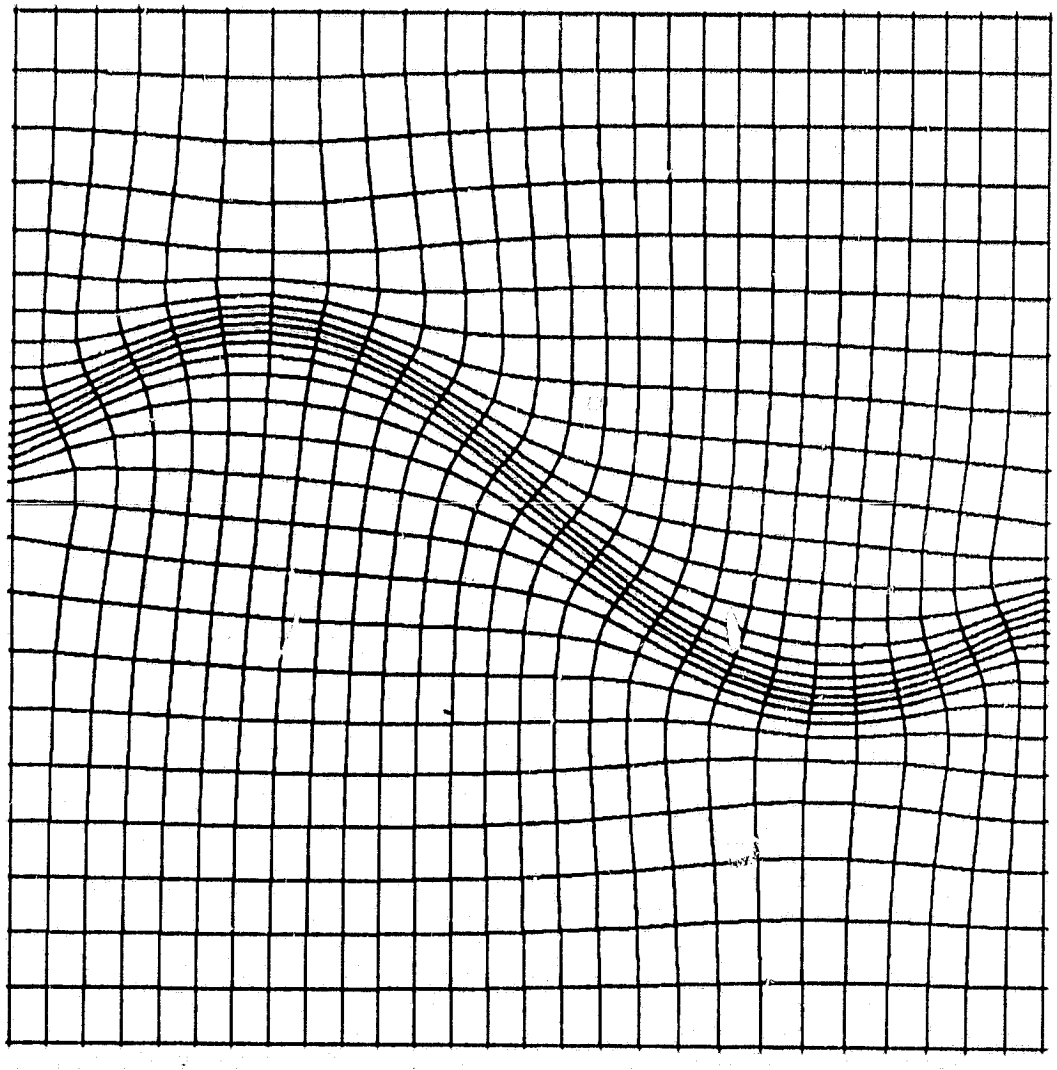
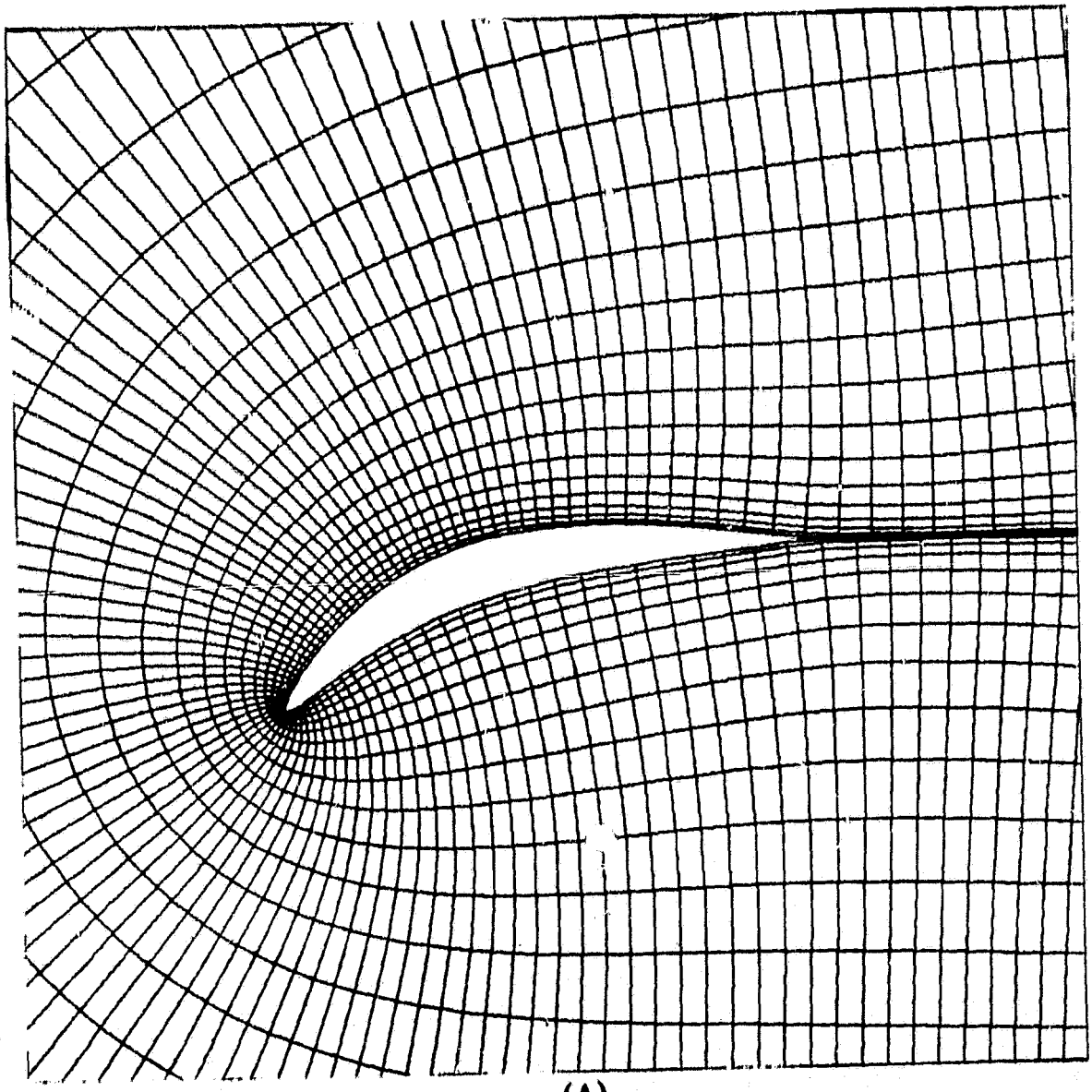
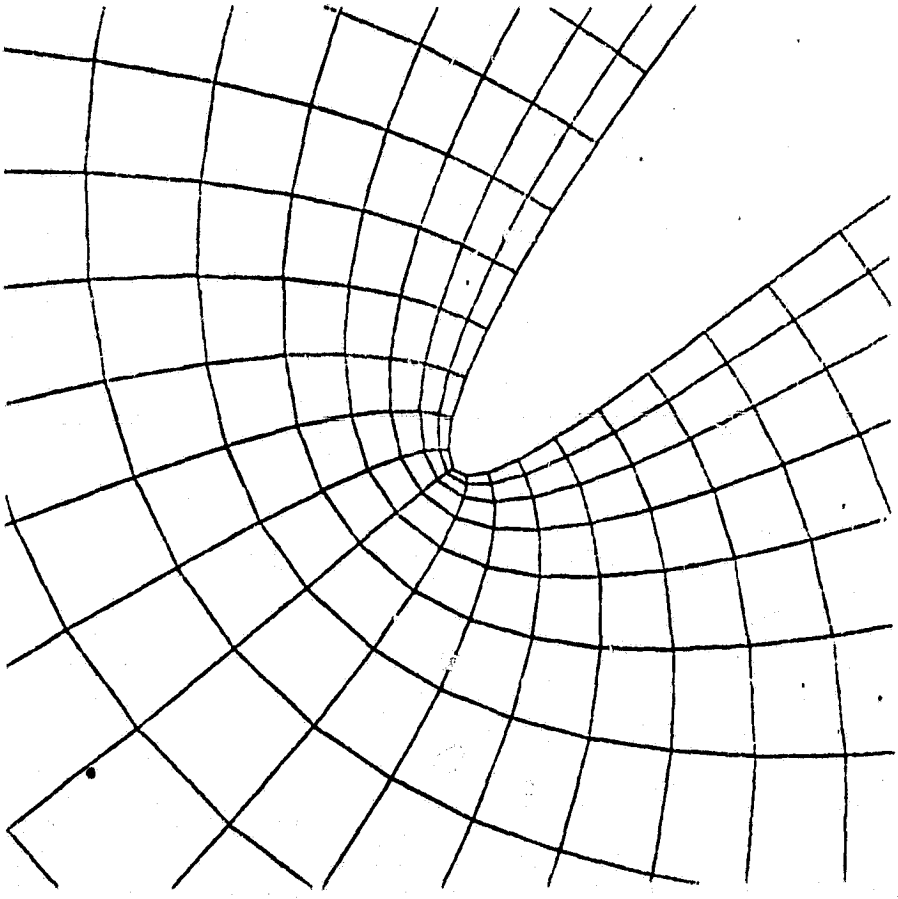


Figure 10 **Grid on a rectangular domain with adapting to a given curve**



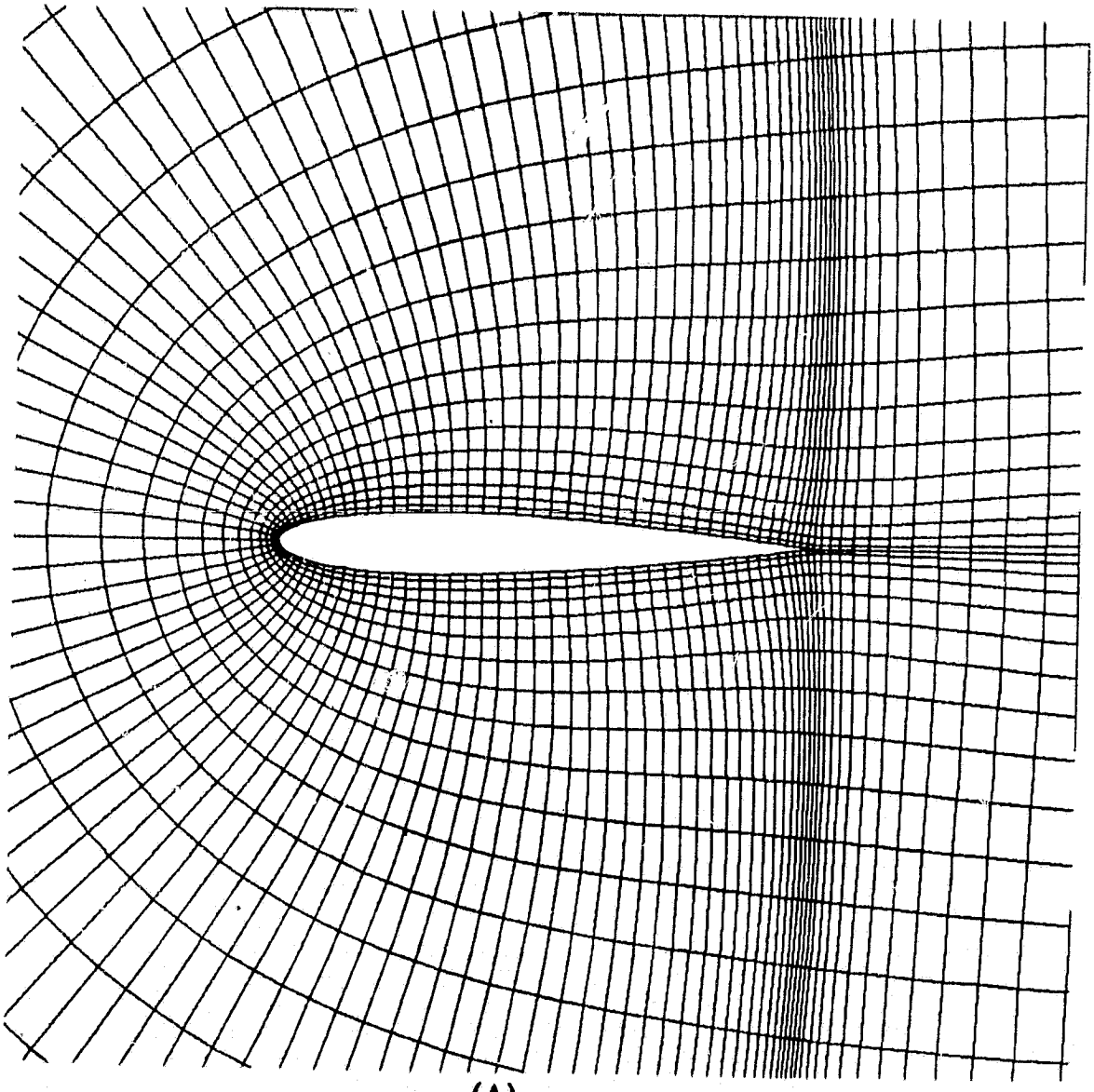
(A)

Figure 11 **C-grid generated by the UBGR scheme**



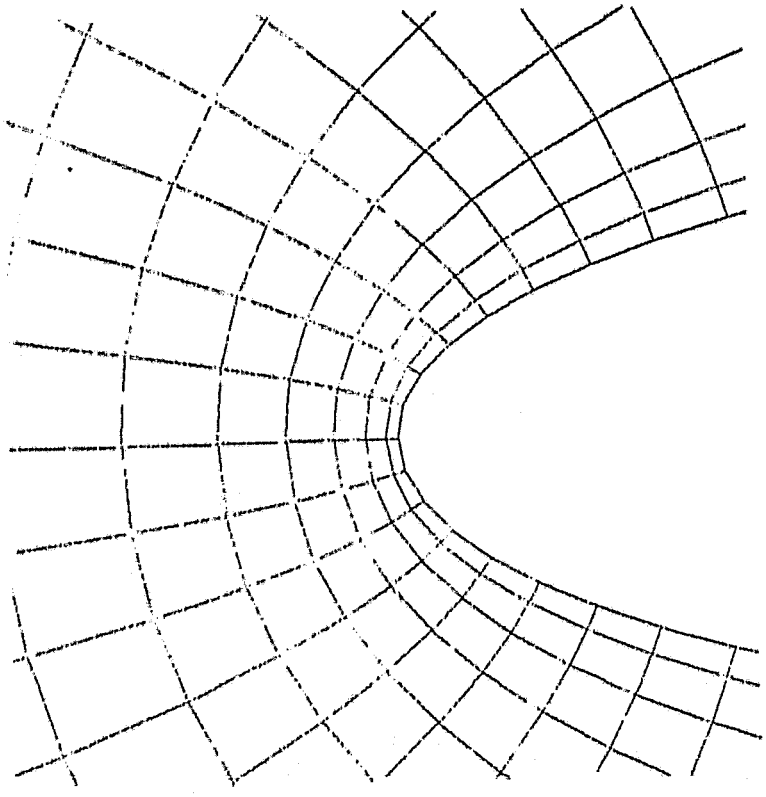
(B)

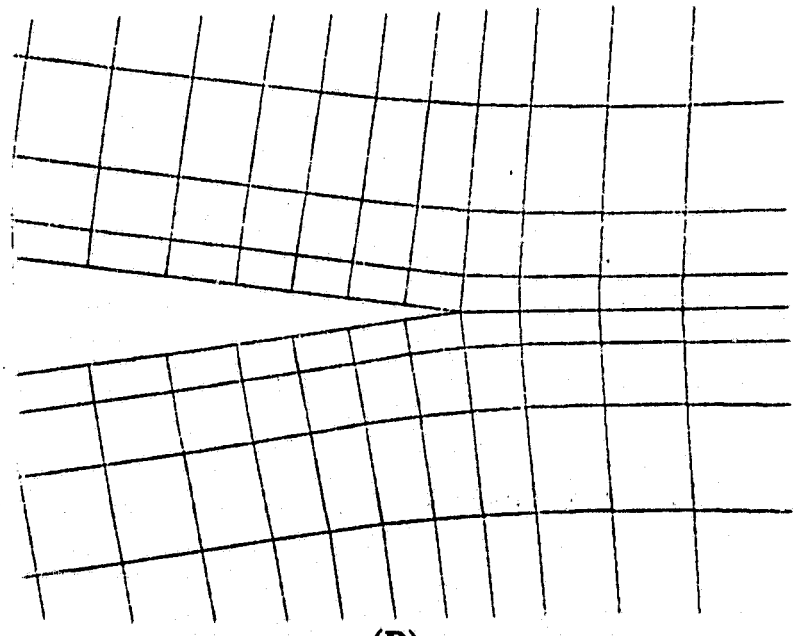
Figure 11 (continued) Close view of the leading edge



(A)

Figure 12 **C-grid generated for NACA-0012 airfoil by the FBGR scheme**





(B)

Figure 12 (continued) Close view of leading and trailing edges

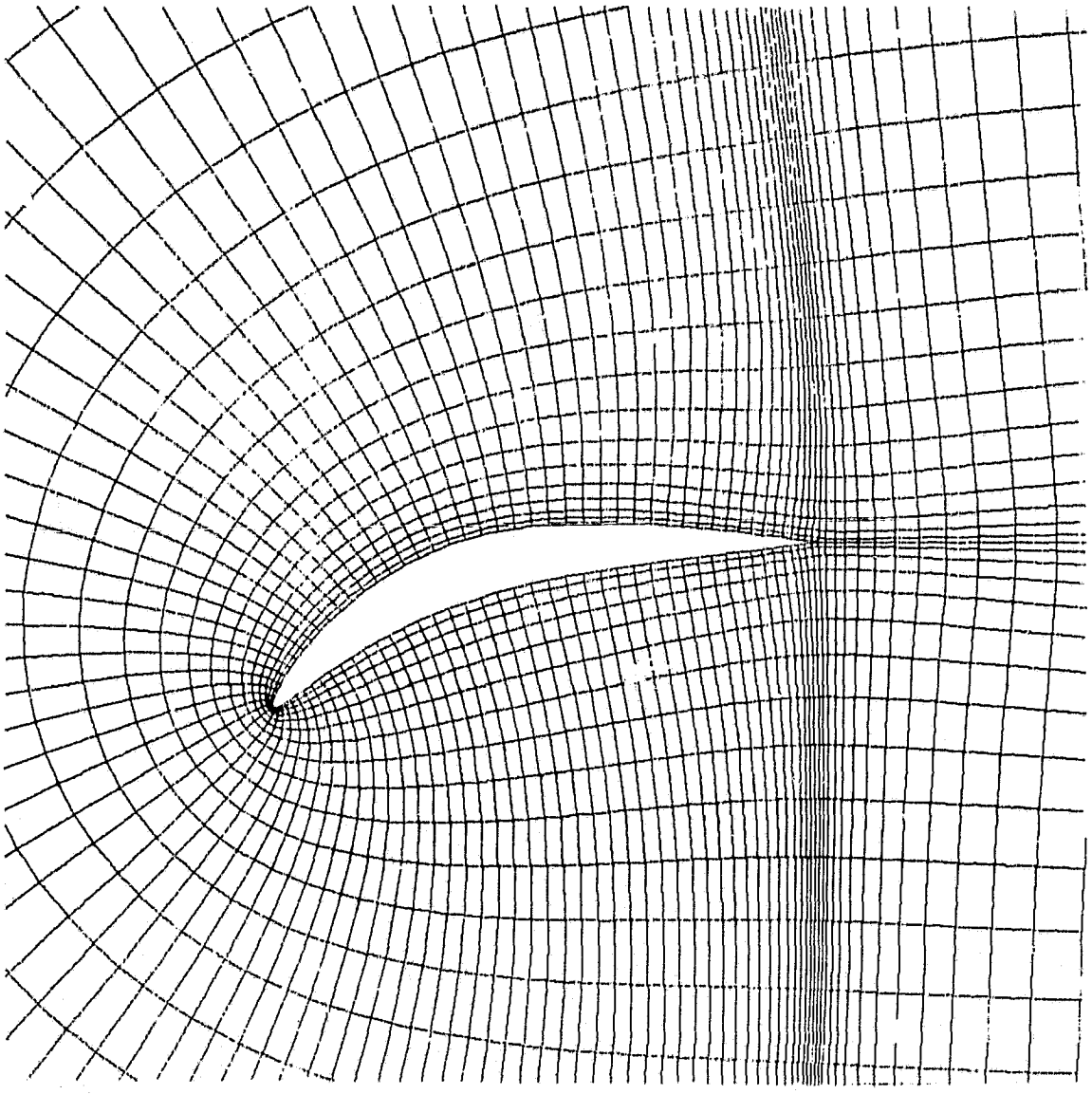


Figure 13 C-grid generated for an arbitrarily cambered airfoil by the FBGR scheme

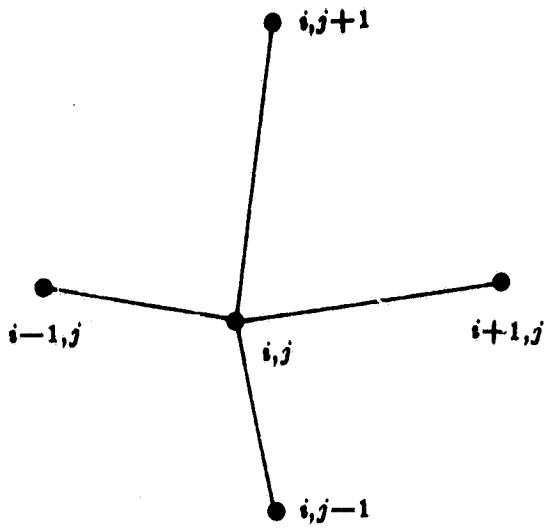


Figure 14 Primary grid

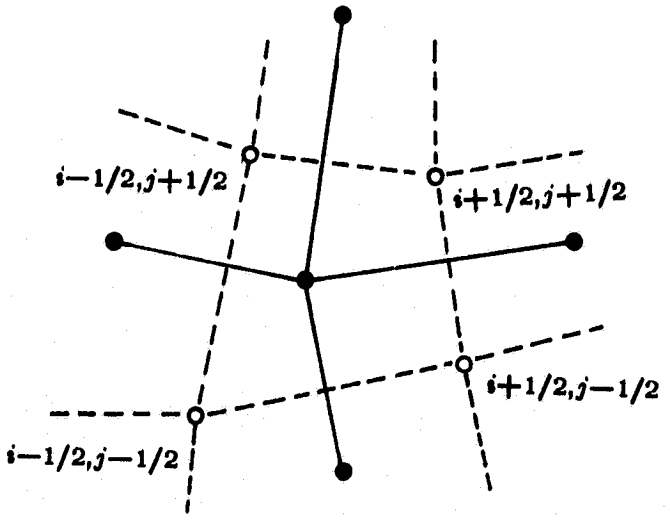


Figure 15 Auxiliary grid