JOURNAL OF COMPUTATIONAL PHYSICS 47, 463–472 (1982)

A Segmentation Approach to Grid Generation Using Biharmonics

J. B. BELL, G. R. SHUBIN, AND A. B. STEPHENS

Applied Mathematics Branch (R44),
Naval Surface Weapons Center, Silver Spring, Maryland 20910

Received July 14, 1981; revised November 3, 1981

We present a method for the numerical generation of finite difference grids. In this method, a complicated shape is divided into simpler parts, then each part is transformed to a rectangle and is individually gridded. It is demonstrated that requiring the transformation function defining each subgrid to satisfy a linear fourth order PDE system provides the flexibility necessary to smoothly patch together the subgrids. The ability to prescribe mesh point locations on subgrid boundaries gives a simple method for controlling the locations of mesh points which are interior with respect to the composite grid.

1. INTRODUCTION

The numerical generation of curvilinear coordinate systems is a valuable tool for formulating finite difference models of physical problems. One often wants to specify the number and approximate distribution of mesh points in areas where the solution is of particular interest, such as in boundary layers or near singularities. Consequently, a good deal of work has been done on methods that generate grid systems which allow the user to specify mesh point locations on the boundary and, to some extent, control their distribution in the interior.

It is a common practice to transform a region Ω in physical space to a simpler region R (generally a rectangle) in a computational space. It is then easy to define difference equations for the transformed equations on a regular grid in R. The inverse of this transformation defines a curvilinear coordinate grid on Ω. The trick is to choose the transformation in such a way that the grid on Ω has certain desirable features while maintaining control over the placement of mesh points. For instance, it is obvious that grid lines from the same family should not cross. Moreover, since abrupt changes in finite difference grid spacing degrades the numerical results, the gradation of the spacing should be smooth and grids should not be excessively skewed. Additionally, the transformations defining the grids may be time or iteration dependent. This allows the coordinate system in the physical plane to deform with a surface while computations are done in a fixed computational region R.

In the last few years, a method has been developed which generates a coordinate system in physical space by considering the physical coordinates as solutions of a

0021-9991/82/090463-10$02.00/0

quasilinear elliptic system with Dirichlet boundary conditions in the transformed region R. This approach has been fully and systematically developed by Thompson *et al.* [1], who employ a second-order elliptic system. By and large, most such methods directly specify grid points on the boundary of Ω. Indirect control over interior points is obtained by adding various forcing terms to the elliptic system [2]. Recent contributions to mesh generation techniques using this and other methods may be found in [3].

In this paper, we present a variation of the above method which allows the user to specify the placement of grid points not only on the boundary of Ω but also at various interior points. Basically, our approach is to construct a grid system on Ω as a union of subgrids which are smoothly joined at boundaries. To insure that the subgrids fit together smoothly, additional boundary conditions are imposed to specify the slopes and spacings of mesh lines at the boundaries of subgrids. Requiring the transformation functions to satisfy a fourth-order elliptic system on each subgrid allows the specification of these conditions and insures the smoothness of the grid lines in the subregion interiors. The use of a higher order elliptic system together with finite elements has been examined by Dickson [4].

The system of equations which is used to determine the transformation functions on a subgrid can be discretized, yielding a linear set of difference equations. Since each system is relatively small, there are only modest demands on computer storage and the total computational time for the overall grid is quite small. We remark, however, that the use of subgrids may result in a more complex data structure than would be required when using a single grid in solving a partial differential equation on Ω.

Finally, we note that the method presented here extends naturally to three-dimensional grid generation [5].

2. Analytic Formulation

In order to construct a grid composed of smaller subgrids, it is necessary to control the location of grid points on subgrid boundaries and ensure that the small grids fit together smoothly. To ensure that the subgrids fit together, we must specify the slope of the grid lines as they intersect the subgrid boundaries and control the grid sizes near the boundaries. Failure to control these factors can lead to grids which are excessively skewed or which have a nonsmooth variation of mesh spacing (see Fig. 1).

Specification of the coordinate values, slopes, and local mesh spacing requires four boundary conditions on the boundary of each subgrid. Attempting to smoothly extend the grid lines to the interior of a subgrid using a second-order elliptic system would produce an overspecified problem for which there is, in general, no solution. To accomodate the extra boundary conditions we will use a fourth-order elliptic system to extend the grid lines to the interior.

Formulating the problem mathematically involves two steps. Following [1], we

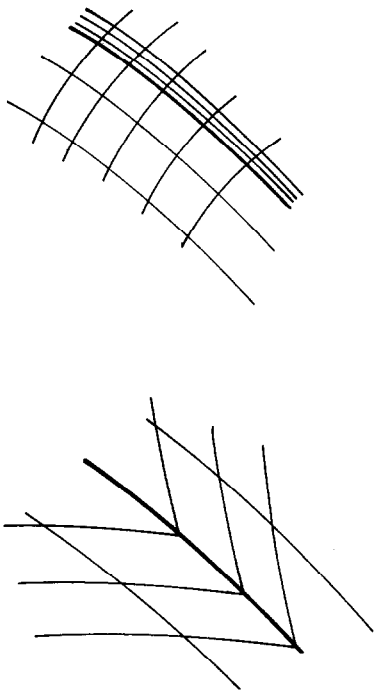

FIG. 1. Undesirable grids.

first formulate equations describing the transformation from physical space to the computational space; then we reverse the role of dependent and independent variables, resulting in equations for the transformation from computational space to physical space. We shall consider a $\xi - \eta$ computational space R corresponding to a subgrid region Ω_s in an $x - y$ physical space as in Fig. 2. Computational boundaries are specified as indicated.

We now formulate a boundary value problem to determine the transformation

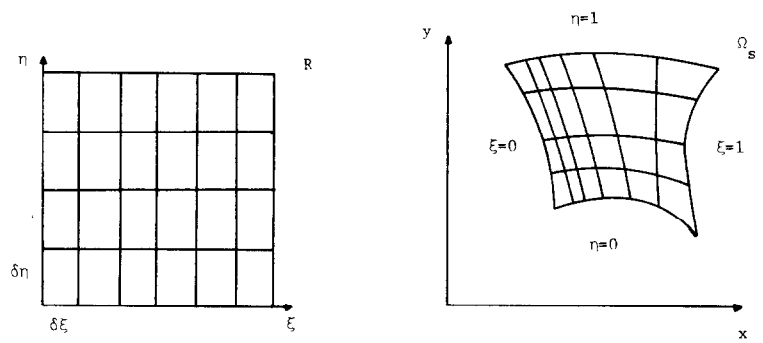

FIG. 2. The image R in computational space (ξ, η) of a subgrid Ω_s in physical space (x, y).

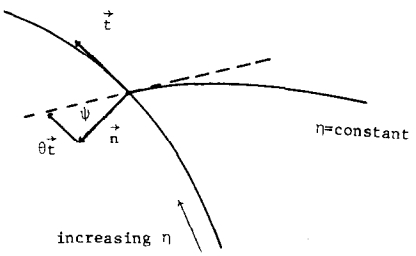

FIG. 3. Specification of the angle that a line where $\eta = \text{const}$ makes with the $\xi = 0$ boundary.

$\xi(x, y)$, $\eta(x, y)$ from Ω_s to R. The location of grid points on the boundary of Ω_s specifies boundary values for ξ and η. To specify the slopes of the grid lines at the boundaries, we prescribe the angle ψ the grid line makes with the normal to the boundary at each point, and take $\Theta = \tan\psi$. On a boundary where $\xi = 0, 1$, we specify the slope of the intersecting η grid line by requiring

$$\nabla\eta \cdot (\mathbf{n} + \Theta(\eta)_t) = 0,$$

where $\mathbf{n}$ is the unit outer normal at the boundary and $\mathbf{t}$ is the unit tangent in $x - y$ space (see Fig. 3). Analogously, on lines where $\eta = 0, 1$, we specify that $\nabla\xi$ be orthogonal to $\mathbf{n} + \Theta(\xi)\mathbf{t}$.

There are several possibilities involving first- and second-order derivative conditions which can be used to control the grid spacing near subgrid boundaries. Numerical experiments indicated that specification of $\partial\xi/\partial n$ on $\xi = 0, 1$ and $\partial\eta/\partial n$ on $\eta = 0, 1$ are good choices, and our discussion will be limited to this case. Special care should be used in specifying these values to ensure consistency of the boundary conditions near corners. This issue will be discussed in detail in Section 3.

We wish our transformation functions to satisfy a system of nonlinear elliptic differential equations in the interior of Ω_s. The differential operator will be denoted L and will be determined later. Thus, for a subgrid Ω_s, the transformation to the computational domain R satisfies

$$L\xi = 0, \quad \text{in } \Omega_s, \tag{1.1}$$

$$L\eta = 0 \quad \text{in } \Omega_s, \tag{1.2}$$

$$\xi, \eta \quad \text{given}, \quad \text{on } \partial\Omega_s, \tag{2.1}$$

$$\nabla\eta \cdot (\mathbf{n} + \Theta(\eta)\mathbf{t}) = 0, \quad \text{on } \xi = 0, 1, \tag{2.2}$$

$$\frac{\partial\xi}{\partial n} \quad \text{given}, \quad \text{on } \xi = 0, 1, \tag{2.3}$$

$$\nabla\xi \cdot (\mathbf{n} + \Theta(\xi)\mathbf{t}) = 0, \quad \text{on } \eta = 0, 1, \tag{2.4}$$

$$\frac{\partial\eta}{\partial n} \quad \text{given}, \quad \text{on } \eta = 0, 1. \tag{2.5}$$

We are interested in a transformation $x(\xi, \eta)$, $y(\xi, \eta)$ from computational space to physical space, but the formulated equations are for the inverse of the transformation. As usual [1], the assumed invertibility of the transformation can now be used to recast the equations by reversing the role of dependent and independent variables. This process is based on the fact that

$$\frac{\partial(\xi, \eta)}{\partial(x, y)} = \left[\frac{\partial(x, y)}{\partial(\xi, \eta)}\right]^{-1},$$

i.e.,

$$\begin{pmatrix} \xi_x & \xi_y \\ \eta_x & \eta_y \end{pmatrix} = \frac{1}{x_\xi y_\eta - x_\eta y_\xi} \begin{pmatrix} y_\eta & -x_\eta \\ -y_\xi & x_\xi \end{pmatrix}.$$

The boundary conditions corresponding to (2.1)–(2.5) become

$$x, y, \quad \text{given}, \quad \text{on } \partial R; \tag{3.1}$$

$$\beta \pm \Theta J = 0, \quad \text{on} \quad \xi = 0, \quad \xi = 1, \quad \text{respectively}; \tag{3.2}$$

$$-\alpha/J, \text{ given}, \quad \text{on} \quad \xi = 0, 1; \tag{3.3}$$

$$\beta \pm \Theta J = 0, \quad \text{on} \quad \eta = 0, \quad \eta = 1, \quad \text{respectively}; \tag{3.4}$$

$$-\gamma/J, \text{ given}, \quad \text{on} \quad \eta = 0, 1, \tag{3.5}$$

where $\alpha^2 = x_\eta^2 + y_\eta^2$, $\beta = x_\xi x_\eta + y_\xi y_\eta$, $\gamma^2 = x_\xi^2 + y_\xi^2$, and $J = x_\xi y_\eta - x_\eta y_\xi$. We note that by using the known data on the boundary of R, the above boundary conditions are linear.

Finally, we shall choose the operator L so that when the roles of the variables are switched, $L\xi = 0$ and $L\eta = 0$ become the linear equations

$$\Delta^2 x = 0, \tag{3.6}$$

$$\Delta^2 y = 0. \tag{3.7}$$

It should be noted that there is no maximum principle for this system of equations; consequently, it is possible to lose invertibility of the transformation for injudicious choices of subgrids.

3. Discretization

In this section, we shall discuss the numerical solution of the system of equations (3.1)–(3.7). There are two steps in the development of the numerical procedure. First, the differential equations must be discretized; then, the resulting linear finite difference equations must be solved. The x and y solution values of the system will provide the physical coordinates of the grid points.

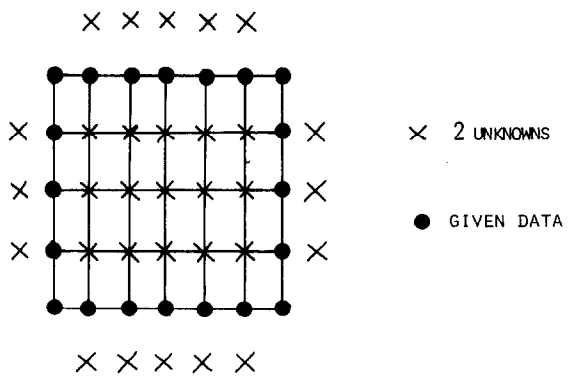

FIG. 4. Discretized computational space (ξ, η).

At each interior point, we apply the standard 13-point discretization of the biharmonic operator to $\Delta^2 x, \Delta^2 y$. Note (see Fig. 4) that the biharmonic approximation at points just inside the boundary introduces a point *outside* the domain. Since there are two unknons (x and y) at each point, the outside points have each introduced two new unknowns. Thus, if

$$N = 1/\delta\xi - 1, \qquad M = 1/\delta\eta - 1,$$

we have NM interior points with $2MN$ unknowns. The discretization of the biharmonic yields $2MN$ equations; however, it introduces $2(M+N)$ outside points and $4(M+N)$ new unknowns. The discretized boundary conditions will provide the extra $4(M+N)$ equations needed to close the system.

The first step in discretizing the boundary conditions is to specify $\partial\xi/\partial n$ on $\xi = 0, 1$ and $\partial\eta/\partial n$ on $\eta = 0, 1$ as required to evaluate (3.3) and (3.5). The case for $\xi = 0$ will be considered; other boundaries are treated analogously. We first note that the location of the grid point at $(\delta\xi, \delta\eta)$ has already essentially been determined by specifying the slopes of the grid lines $\eta = \delta\eta$ at $\xi = 0$ (Eq. 3.2) and $\xi = \delta\xi$ at $\eta = 0$ (Eq. (3.4)). The value of $\partial\xi/\partial n$ to be specified at $\xi = \delta\xi$, $\eta = \delta\eta$ is directly determined by using this information. This is done in order to avoid an inconsistency in the corner, and is related to requiring that the two second-order mixed partial derivatives agree at the corner point. Using a similar analysis, we then determine the value of $\partial\xi/\partial n$ to be specified at $\xi = 0$, $\eta = 1 - \delta\eta$. The specified values of $\partial\xi/\partial n$ at the remaining boundary points along $\xi = 0$ are determined by linear interpolation.

We are now ready to discretize the boundary conditions (3.2)–(3.5). Again we will consider only $\xi = 0$. To approximate (3.2) and (3.3), we need only approximate x_η, y_η, x_ξ, y_ξ for $\eta = \delta\eta, \ldots, 1 - \delta\eta$. A 2-point centered difference for the tangential derivatives x_η and y_η was found to be satisfactory.

Approximation of the normal derivatives x_ξ and y_ξ requires more care. A two-point one-sided outward difference decouples the boundary points from the biharmonic and

can lead to kinks in the grid lines just inside the boundary. Using only a two-point centered difference approximation produces the opposite effect; viz., it fails to insure sufficient adherence of the approximate solution to the boundary conditions. A three-point inward one-sided difference is also unsuitable; when used throughout, it yields a singular system.

When assembling subgrids, slope specification (3.2) is crucial, whereas some flexibility in spacing conditions (3.3) does not adversely affect the results. This observation suggests using a 3-point one-sided inward difference to evaluate x_ξ and y_ξ in (3.2) and a two-point centered difference for (3.3). This combination was found to yield good results, and was used for all examples in Section 4.

The discretized system now contains $2MN$ linear equations corresponding to discretization of (3.6) and (3.7) and $4(M+N)$ linear equations corresponding to discretized boundary conditions. Consequently, for the subgrid Ω_s in question, we have a banded linear system with $2MN + 4(M+N)$ unknowns and equations, which for our ordering of unknowns has a bandwidth of $8N + 17$. The solution of this linear system comprises the majority of the computational work necessary to generate a subgrid. Since the systems are small, Gaussian elimination is fast and consequently is used in all of the examples in Section 4.

For generating subgrids with a large number of points, Gaussian elimination can become costly. In that case, the system can be solved iteratively. Since little accuracy is needed, the iterative scheme converges adequately in very few steps. Numerical solution of the system using iterative methods will be discussed more fully in a forthcoming paper covering 3-D grid generation [5].

4. Results

In this section, we present some relatively simple grids generated by the above method. These examples show how subgrids can be joined smoothly across boundaries, and also how specification of grid point locations on subgrid boundaries which are interior to the composite grid can help control the placement of interior grid points. When subgrids are used, the interior subgrid boundaries are shown as darkened lines.

In the first example, we consider two approaches for gridding a triangle. In Fig. 5a, there is only one subgrid region. Point B is placed on edge AC and A, B, C, and D map into the corners of the computational space. The grid slope specifications are taken to be linear interpolations between the given corner slopes, except for the grid lines nearest point B where an angle of 60° with respect to the normal direction was arbitrarily chosen. This grid has the disadvantage that at point B the transformation is singular. This leads to difficulties both in forming the grid and using the computed grid. (For purposes of generating the grid, the location of line BC was slightly perturbed). By contrast, a very different gridding was obtained (Fig. 5b) by subdividing the triangle into three quadrilaterals and enforcing normality at the subgrid boundaries to join the subgrids smoothly.

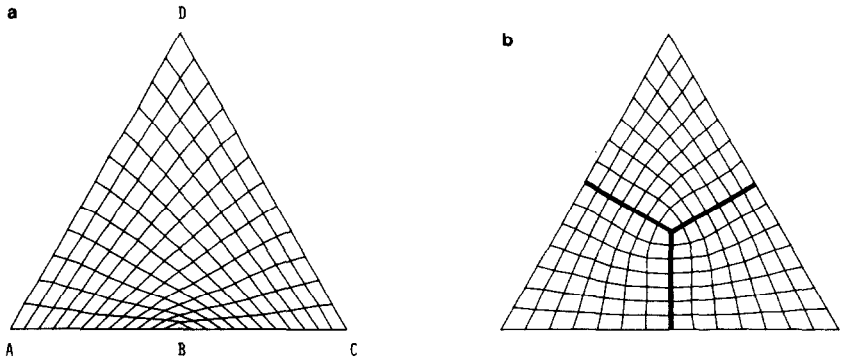

FIG. 5. Two griddings of a triangle (a) as one grid, (b) as the composite of three subgrids.

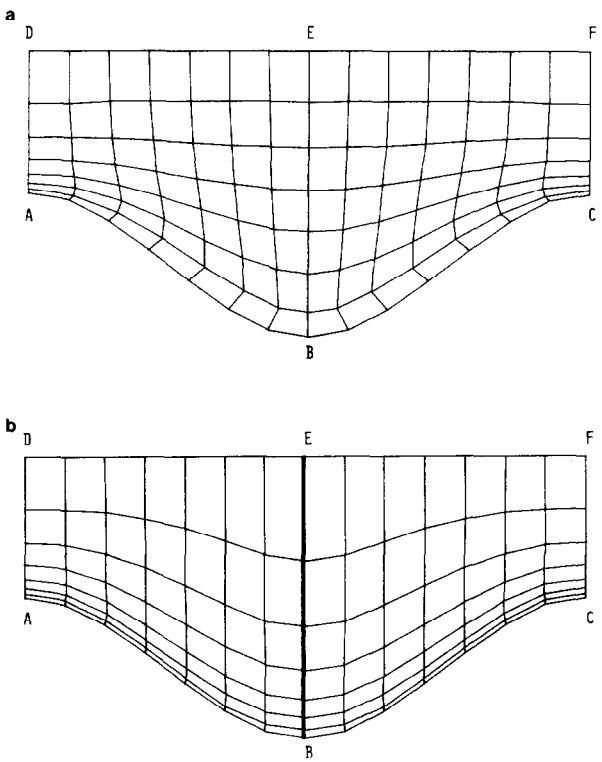

FIG. 6. Two griddings for flow over a dent: (a) as one grid, (b) with BE as a subgrid boundary, to control grid clustering near point B.

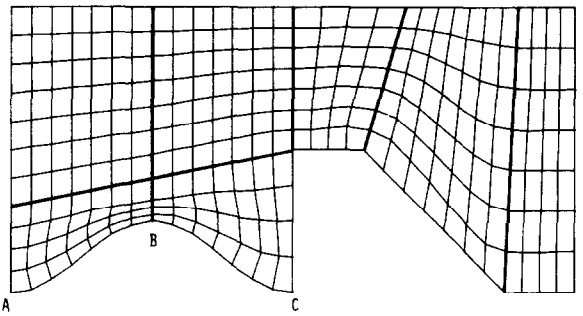

FIG. 7. One gridding of an oddly shaped region as the union of seven subgrids.

The next example (Fig. 6) shows a region in which we want to compute a flow from left to right. We wish to cluster grid points near the bottom boundary ABC and to have the grid lines nearly orthogonal to that boundary. In Fig. 6a, no subgridding was used and an exponential stretching of grid points along AD and CF was specified. We see that without some effort at control, the interior grid points fail to cluster closely to the boundary ABC near point B. While some manipulation of the spacing conditions (Eqs. (3.3), (3.5)) could improve the clustering, a more simple and direct approach is to use subgridding. In Figure 6b we choose BE to be a subgrid boundary, and by specifying the grid point locations on BE, we directly control the clustering near point B. Smooth joining of the subgrids is again controlled by specifying approximate normality.

Finally, in Fig. 7 we show one method of gridding a more complex shape. This example illustrates the flexibility that the user has in his choice of decomposing the composite grid into subgrids. Here, we have chosen seven subgrids, the largest of which is 8×8 $(M = N = 6)$. The computational work to construct such a subgrid essentially requires only the solution of a banded linear system with 120 unknowns and bandwidth 65. The slope conditions at the outer boundaries were chosen in various ways—for example, to enforce approximate normality along ABC. The smoothness of joining at subgrid boundaries is apparent. With only very little experience using the method, the user can wisely choose a subgrid decomposition and thereby effectively control the placement of grid points.

REFERENCES

1. J. F. THOMPSON, F. C. THAMES, AND C. W. MASTIN, *J. Comput. Phys.* **15**, (1974), 299.
2. J. F. THOMPSON, F. C. THAMES, C. W. MASTIN, AND S. P. SHANKS, *in* "Proceedings, AIAA Second Computational Fluid Dynamics Conference, Hartford, Connecticut, June 1975," p. 68.
3. "Numerical Grid Generation Techniques," NASA Conference Publication 2166, Workshop at NASA–Langley Research Center, Hampton, Virginia, October 1980.

4. L. J. DICKSON, *in* "Numerical Grid Generation Techniques," NASA Conference Publication 2166, Hampton, Virginia.
5. G. R. SHUBIN, A. B. STEPHENS, AND J. B. BELL, *in* "Numerical Grid Generation, Proceedings of the Symposium on Numerical Generation of Curvilinear Coordinate Systems Nashville, Tennessee, April 1982" (J. Thompson, ed.), Elsevier, New York, to appear.