NASA Technical Memorandum 101018

NASA-TM-101018 19880018658

Three-Dimensional Zonal Grids About Arbitrary Shapes by Poisson's Equation

Reese L. Sorenson

August 1988

National Aeronautics and
Space Administration

NASA Technical Memorandum 101018

Three-Dimensional Zonal Grids About Arbitrary Shapes by Poisson's Equation

Reese L. Sorenson, Ames Research Center, Moffett Field, California

August 1988

National Aeronautics and
Space Administration

Ames Research Center
Moffett Field, California 94035

THREE-DIMENSIONAL ZONAL GRIDS ABOUT ARBITRARY SHAPES BY POISSON'S EQUATION

Reese L. Sorenson*
NASA Ames Research Center, Moffett Field, CA

1. SUMMARY

A method for generating three-dimensional finite difference grids about or within arbitrary shapes is presented. The 3-D Poisson equations are solved numerically, with values for the inhomogeneous terms found automatically by the algorithm. Those inhomogeneous terms have the effect near boundaries of reducing cell skewness and imposing arbitrary cell height. The method allows the region of interest to be divided into zones (blocks), allowing the method to be applicable to almost any physical domain. A FORTRAN program called 3DGRAPE has been written to implement the algorithm. Lastly, a method for redistributing grid points along lines normal to boundaries will be described.

2. INTRODUCTION

The complex "real world" geometries to which computational physics is currently being applied demand a new level of flexibility in grid-generation. An airplane with strakes, canards, inlets, external stores, a plume, and a canopy is an example of the cases which cannot practically be gridded by a mapping into a single rectangular-solid computational domain. It becomes necessary to partition the physical domain into an arbitrary number of zones and then have a grid-generator which can treat such a zoned domain. Further complication derives from the needs imposed by the solvers of the equations of mathematical physics for which the grid is being generated. These include the need for orthogonality or near-orthogonality, especially near boundaries where high gradients typically occur; the ability to control grid cell height at boundaries for the same reason; smoothness in the interior of the domain; and smoothness as the grid makes the transition across zone-to-zone boundaries. Lastly, grid-generation is an applied science, and a grid-generation methodology has value only when it has been "packaged" in a well-documented, readily-available, well-written, robust, fast, user-friendly program.

---

*Research Scientist.

A grid-generation methodology is described which meets the requirements listed above. Grids are generated by the solution of Poisson's elliptic partial differential equation, which can be shown [1] to give the smoothest possible grid in the interior of a domain. The inhomogeneous terms, or right-hand-side (RHS) terms, in this equation are constructed so as to minimize grid cell skewness and control grid cell height near any or all of the six boundary faces of the zones (computational cubes). This is accomplished in an automated manner, requiring a minimum of user expertise. The values of the RHS terms which give the desired properties in the resultant grid are found automatically from side-condition equations, concurrently with the solution for the grid. Grid cell height can be specified independently for each face of each zone; alternatively the feature of controlling cell height and skewness can be disabled at the various faces at the user's option.

The method is zonal, meaning that the physical domain can be divided into an arbitrary number of contiguous zones. The user has complete freedom in topologically connecting the zones. The method solves for the grid in all of the zones simultaneously, assuring smoothness across zonal boundaries within the grid. The shapes of the zonal-interface surfaces, and the distribution of points thereon, need not be defined by the user a-priori; they can be the result of the Poisson equation solution process. This feature significantly reduces the amount of surface-fitting preparation required before the grid-generator can be used.

The technology described above is packaged in a grid-generator program called 3DGRAPE, an acronym for "Three-Dimensional Grids About Anything by Poisson's Equation." The problem of specifying how a complex, three-dimensional, multi-zone grid is to be constructed is not trivial. The variety of ways in which a user might want to twist and deform computational cubes in a realistic zonal application is difficult to predict. A way of specifying that topology (including the block-to-block interconnections), in a manner which is precise yet not pedantic, has been developed.

3. DEVELOPMENT OF GRID-GENERATON ALGORITHM

The Poisson equations in three dimensions are well-known, and can be found in Ref. [2]. But the equations in this form, yielding ξ,η,ζ for given x,y,z, would be awkward to use, especially when specifying the boundaries. Instead, the

transformed Poisson equations are used, wherein x,y,z are found from given ξ,η,ζ:

$$\alpha_{11}\vec{r}_{\xi\xi} + \alpha_{22}\vec{r}_{\eta\eta} + \alpha_{33}\vec{r}_{\zeta\zeta} + 2(\alpha_{12}\vec{r}_{\xi\eta} + a_{13}\vec{r}_{\xi\zeta} + \alpha_{23}\vec{r}_{\eta\zeta}) = -J^2(P\vec{r}_{\xi} + Q\vec{r}_{\eta} + R\vec{r}_{\zeta}) \qquad (1a)$$

where:

$$\vec{r} = \begin{bmatrix} x \\ y \\ z \end{bmatrix} \qquad \alpha_{ij} = \sum_{m=1}^{3} \gamma_{mi}\gamma_{mj} \qquad (1b,c)$$

γ_{ij} is the ijth cofactor of the matrix M:

$$M = \begin{bmatrix} x_{\xi} & x_{\eta} & x_{\zeta} \\ y_{\xi} & y_{\eta} & y_{\zeta} \\ z_{\xi} & z_{\eta} & z_{\zeta} \end{bmatrix} \qquad (1d)$$

and the Jacobian J is the determinant of M.

In such a grid-generation approach the inhomogeneous terms may be chosen as one wishes. Here, as in the two-dimensional grid generation method discussed in Ref. [3], they are chosen in a way which is complicated to describe, but simple to use. The actual term P, in Eq. (1a), is:

$$P(\xi,\eta,\zeta) = \sum_{n=1}^{6} P_n(\xi,\eta,\zeta) \qquad (2)$$

Terms Q and R are identical in form.

Space limitations prevent a detailed treatment of the derivation of each of the terms in the right side of Eq. (2). But those derivations are completely analogous, so a treatment of P_1 will have to suffice. In the remainder of this paper, as well as in the 3DGRAPE program, the faces of the computational cube are numbered. Face 1, for example, is the face wherein ξ is fixed at zero. The index n in Eq. (2) is the face number for $1 \le n \le 6$. So in deriving P_1, we will be considering how to control the grid cell height and skewness on face 1. The term P_1 is defined as:

$$P_1(\xi,\eta,\zeta) = p_1(\eta,\zeta)e^{-a\xi} \qquad (3)$$

where a is a positive constant. At face 1 the exponential factor becomes 1.0, so the problem for face 1 reduces to finding p_1. It can be shown that on face 1, the other terms P_2 through P_6 are of no effect. So on face 1, inhomogeneous term P reduces to just p_1. This is true similarly for Q reducing to q_1 and R reducing to r_1.

Thus on face 1, the transformed Poisson equations (1a) could be thought of as a coupled set of three linear equations in the three unknowns p_1, q_1, and r_1, with the left sides being constant on the boundary and within each computational time step. Those equations are solved for p_1, q_1, and r_1. That solution, however, requires that all of the derivatives present on the left sides be known at face 1. This means that all possible first and second partial derivatives of x,y,z with respect to ξ,η,ζ must be found. The derivatives involving only η, or ζ, or both η and ζ, are found by simply differencing the given fixed boundary points.

Derivatives involving only ξ are found in the following manner. Three geometric constraints are imposed upon a line of varying ξ which intersects face 1: (1) it must be normal to the intersecting line of varying η on the surface, (2) it must be normal to the intersecting line of varying ζ on the surface, (3) the length along that line from the surface to the next node must be controlled. These geometric constraints are equivalent to the three algebraic constraints:

$$\vec{r}_\xi \cdot \vec{r}_\eta = 0 \tag{4a}$$

$$\vec{r}_\xi \cdot \vec{r}_\zeta = 0 \tag{4b}$$

$$\vec{r}_\xi \cdot \vec{r}_\xi = S^2 \tag{4c}$$

where S is that desired distance to the first point off of the surface. From these equations, expressions can be obtained for the derivatives with respect to ξ at the surface in terms of S and the γs of Eq. (1). Again, space limitations prohibit a complete derivation, but an example is:

$$X_\xi = \frac{\gamma_{11} S}{\pm\sqrt{\gamma_{11}^2 + \gamma_{21}^2 + \gamma_{31}^2}} \tag{5}$$

The positive sign for the radical is chosen for a right-handed coordinate system. Note that the "handedness" can vary from

block to block in the same grid. Derivatives so obtained, then, can be differenced with respect to η and ζ to obtain the $\xi\eta$ and $\xi\zeta$ mixed second-partial derivatives. All of the derivatives discussed so far are fixed with respect to computational time, and need be computed only once.

The only derivatives remaining to be found are the second partial derivatives with respect to ξ. Those can be found by differencing the existing solution at the current computational time step. Equation (1a) then is treated as a 3-by-3 linear system with the left side being constant, and is solved for p_1, q_1, and r_1. Given these, the P_1, Q_1, and R_1 are known. Terms P_n, Q_n, and R_n, for $2 \le n \le 6$ are found similarly, enabling calculation of values for P,Q,R at the current time step.

The positive constant a that appears in Eq. (3), along with similar constants used in the definitions of Q and R, determine the rate at which the control of cell height and skewness dissipates with distance from that boundary. Reducing these constants causes the control to propagate far out into the field, with the possible consequence of instability in the grid-generation process.

4. BLOCK STRUCTURE

Complicated problems from the "real world" often require that the region of interest be divided into zones. Consider, for example, an effort to create a grid with cylindrical topology about an F-16 aircraft, as described in Ref. [4]. The surface grid, with zone numbers and zonal boundaries, is shown in Fig. 1. It is desired that control of cell height and skewness be applied on the fuselage and the surface of the vertical and horizontal tails, but not on the symmetry plane or the planform surface. These specifications argue for a multi-block approach, with block boundaries coinciding with the leading edges and trailing edges of those control surfaces. But the leading and trailing edges of the vertical and horizontal tails do not appear at the same axial locations, so the block structure indicated in Fig. 1 results. Note that the upper face of block 6 is divided into two sections: one mating with the lower face of block 10, and the other mating with part of the lower face of block 11. The remainder of the lower face of block 11 mates with part of the upper face of block 7, etc.

To compensate for this situation and a plethora of other possibilities, the faces of the blocks can be divided into

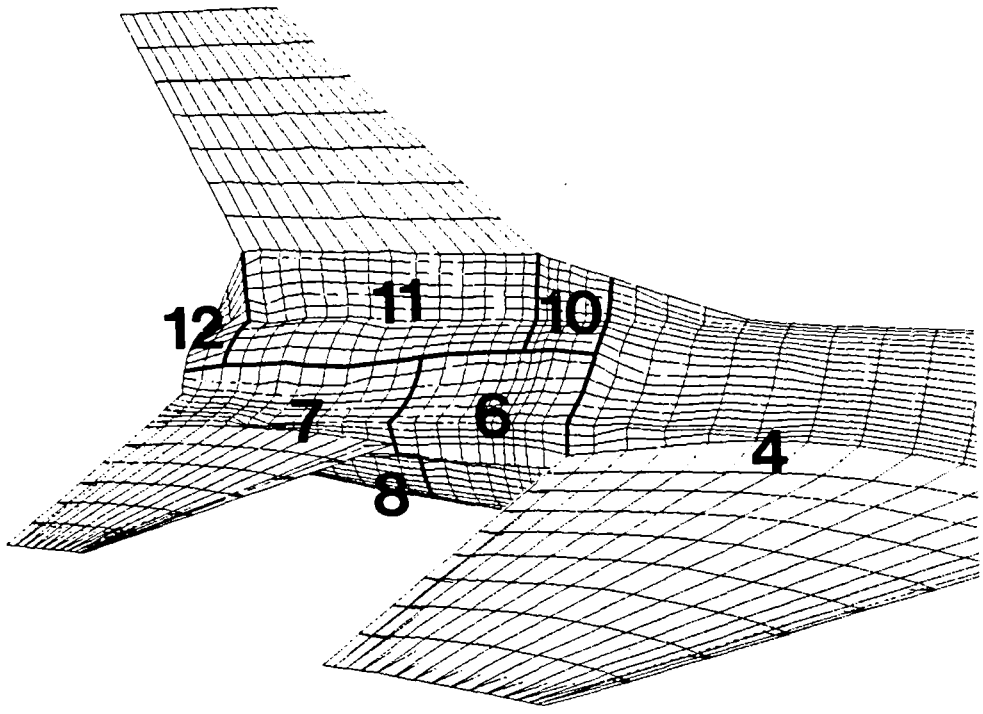


Fig. 1. Surface Grid, on Aft End of F-16 Aircraft Showing Block Structure

sections (subfaces), with each section having its own unique boundary treatment. Sections of boundary faces may be assigned the following boundary conditions:

1. They can abut other sections. The shape of the surface and the distribution of points on it are determined by the elliptic solution. Lines passing through the surface will be continuous, with optional spacing continuity along those lines.

2. They can consist of x,y,z locations specified by the user and made unchanging with computational time.

3. They can have points which float, as dictated by the elliptic solution, on a plane normal to one of the x,y,z axes.

4. They can have points which float, as dictated by the elliptic solution, on a cylinder having its axis coincident with one of the x,y,z axes.

5. They can have points which float, as dictated by the elliptic solution, on an ellipsoid having its semi-axes parallel to the x,y,z axes.

6. They can have points which float, as dictated by the elliptic solution, along a surface which is collapsed to a line coincident with one of the x,y,z axes.

7. They may be collapsed to a point.

The boundary treatments listed above for the various sections of the faces of the blocks are all effected in an explicit manner. After finding new values for the x,y,z in the interior of each block in each iteration step, new values for x,y,z on the boundary faces are found, as appropriate for each face. This explicit treatment makes the grid-generator run slower, but it greatly enhances the modularity of the code. New boundary treatments can be added in a fairly straightforward fashion. Control of grid cell height and skewness may be exercised or not, and the height S in Eqs. (4c) and (5) is specified individually for each face. Note that when two sections of faces abut, the two indices on each face may be different (e.g., if j, k,and l are the indices running in the three computational directions, we might have j and k running on one section, with k and l running on the other section to which it abuts). The indices on abutting sections can even run in opposite directions. These possibilities are allowed in the 3DGRAPE code. Note that the cell height, S, is user specified, and can be used to cluster points very close to the boundary, making viscous calculations possible.

The 3DGRAPE code has also been used to generate an inviscidly spaced grid about an F-16 aircraft, shown in Fig. 2.

5. A RECLUSTERING TECHNIQUE

A problem was encountered while testing 3DGRAPE. When a grid of spherical topology (as on a blunt-nosed aircraft fuselage) was being generated with grid cell height and skewness controlled on the surface, the RHS terms became ineffective near the axis. The problem appeared to be associated with the fact that the Jacobian [see Eq. (1)] became zero at the axis.

While addressing this problem a reclustering technique was discovered, and has proved to be so effective that it will probably be included in the 3DGRAPE code. A typical approach is to generate a Laplacian grid, retain the radial lines (those

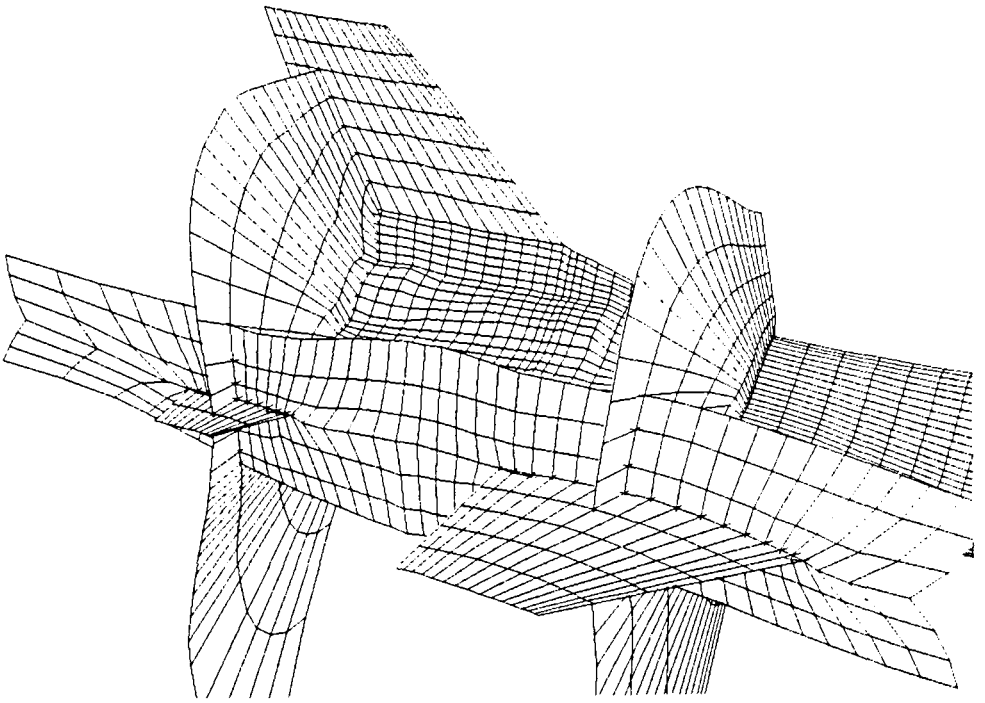
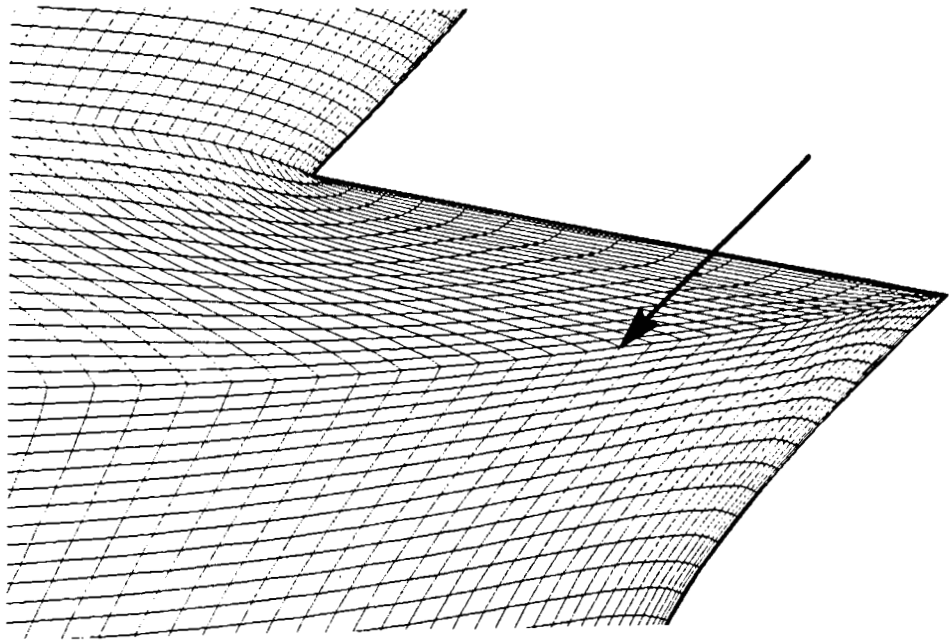


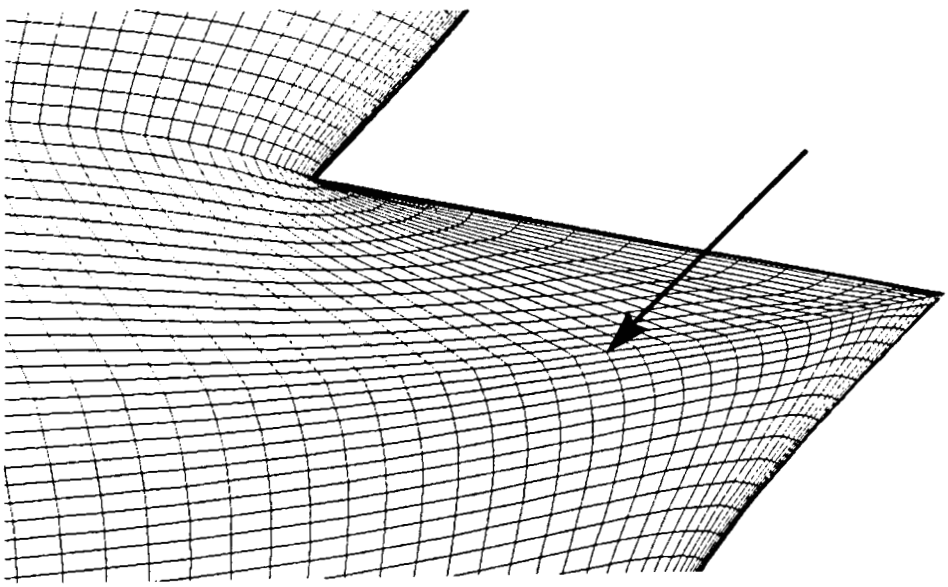
Fig. 2. Selected Surfaces in F-16 Field Grid

running from the body to the outer boundary), and discard the locations of the points on those lines, and recluster somehow along those lines to obtain a desired distance between the body and the first node in the field. Such techniques usually work for smooth bodies, but a problem arises for bodies having slope discontinuities, especially convex corners (see the example in Figure 3a). Such slope discontinuities tend to be propagated far out into the field, in some cases all the way to the outer boundary. It was thought that if the reclustering could start with a Laplacian grid or even a mildly clustered Poisson grid (control terms turned on), and if a reclustering method could be developed which somehow used the existing spacings along radial lines in that unclustered or mildly clustered grid, then the natural tendency of that grid to round off the sharp corners could influence the reclustered grid.

The present approach employs the principle that if a monotonically increasing function that passes through the origin and the point (1,1) is taken to a positive power, it will still increase monotonically and will still pass through the origin and the point (1,1). The procedure is as follows. Along each radial line, calculate the cumulative distances to each point. Normalize that sequence of numbers to range from 0



a) Typical Method, With Discontinuities Propagating Far into the Interior



b) Present Method, Smoother in the Interior

Fig. 3. Closeup of Two Grids with Slope-Discontinuities in Boundary

to 1; thereafter 0 represents the body and 1 the outer boundary. Then take that set of normalized spacings to some power (the same power on all lines), such that the first interval will become the desired first spacing. Then

unnormalize those new spacings, and interpolate the x,y,z to be functions of the new spacings.

Figure 3b shows the same reclustering problem done by the present method. The propagation of boundary-slope discontinuities into the field is significantly reduced. With a small increase in complexity, the present method can be used to modify the number of points along the radial line. Thus, for example, an inviscid spacing using 20 radial points can be generated, and that can be re-interpolated to be a viscous spacing using 50 radial points.

6. CONCLUSIONS

A grid-generation algorithm has been developed which can generate three-dimensional grids for any block-structured topology. On any or all of the six faces of the blocks, grid cell skewness is controlled and any arbitrary cell height (viscous or inviscid) may be imposed. The resulting grid is smooth in the interior of the blocks and across block-to-block boundaries. A FORTRAN program called 3DGRAPE has been written to implement the grid-generation algorithm. The program includes a scheme for redistributing points in existing grids is described which reduces the propagation of boundary slope discontinuities into the field.

7. REFERENCES

1. THOMPSON, J.F., WARSI, Z.U.A., and MASTIN, C.W. - Numerical Grid Generation, North-Holland, p. 192, 1985.

2. SORENSON, R. L. and STEGER, J. L. - Grid Generation in Three Dimensions by Poisson Equations with Control of Cell Size and Skewness at Boundary Surfaces, ASME FED, vol. 5, June 1983, pp. 181-188.

3. STEGER, J.L. and SORENSON, R.L. - Automatic Mesh-Point Clustering Near a Boundary in Grid Generation with Elliptic Partial Differential Equations. J. Comp. Phys., vol. 33, no. 3, Dec. 1979, pp. 405-410.

4. SORENSON, R. L. - Three-Dimensional Elliptic Grid Generation for an F-16, Three Dimensional Grid Generation for Complex Configurations -- Recent Progress, AGARDograph no. 309, March 1988, pp. 23-28.

NASA
National Aeronautics and Space Administration

Report Documentation Page

1. Report No.	2. Government Accession No.	3. Recipient's Catalog No.
NASA TM-101018		

4. Title and Subtitle	5. Report Date
Three-Dimensional Zonal Grids About Arbitrary Shapes by Poisson's Equation	August 1988
	6. Performing Organization Code

7. Author(s)	8. Performing Organization Report No.
Reese L. Sorenson	A-88258
	10. Work Unit No. 505-60

9. Performing Organization Name and Address	11. Contract or Grant No.
Ames Research Center Moffett Field, CA 94035	
	13. Type of Report and Period Covered
12. Sponsoring Agency Name and Address	Technical Memorandum
National Aeronautics and Space Administration Washington, DC 20546-0001	14. Sponsoring Agency Code

15. Supplementary Notes

Point of Contact: Reese L. Sorenson, Ames Research Center, MS 258-1
Moffett Field, CA 94035 (415) 694-2271 or FTS 464-2271

16. Abstract

A method for generating three-dimensional finite difference grids about or within arbitrary shapes is presented. The 3-D Poisson equations are solved numerically, with values for the inhomogeneous terms found automatically by the algorithm. Those inhomogeneous terms have the effect near boundaries of reducing cell skewness and imposing arbitrary cell height. The method allows the region of interest to be divided into zones (blocks), allowing the method to be applicable to almost any physical domain. A FORTRAN program called 3DGRAPE has been written to implement the algorithm. Lastly, a method for redistributing grid points along lines normal to boundaries will be described.

17. Key Words (Suggested by Author(s))	18. Distribution Statement
Grids Aerodynamics Computational fluid dynamics	Unclassified-Unlimited Subject Category - 02

19. Security Classif. (of this report)	20. Security Classif. (of this page)	21. No. of pages	22. Price
Unclassified	Unclassified	12	A02