1995120329

# 3DGRAPE/AL: THE AMES / LANGLEY TECHNOLOGY UPGRADE

Reese L. Sorenson
National Aeronautics and Space Administration
Ames Research Center
Moffett Field, CA 94035-1000

Stephen J. Alter
Lockheed Engineering & Sciences Co.
Langley Research Center
Hampton, VA 23666

## SUMMARY

This paper describes a new three-dimensional structured multiple-block volume grid generator called 3DGRAPE/AL. It is a significantly improved version of the previously-released and widely-distributed program 3DGRAPE, with many of the improvements taken from the grid-generator program 3DMAGGS[1]. It generates volume grids by iteratively solving the Poisson Equations in three-dimensions. The right-hand-side terms are designed so that user-specified grid cell heights and user-specified grid cell skewness near boundary surfaces result automatically, with little user intervention. Versatility was a high priority in this code's development, and as a result it can generate grids in almost any three-dimensional physical domain. Improvements include added kinds of forcing functions, improved control of cell skewness, improved initial conditions, convergence acceleration, the ability to take as input the output from GRIDGEN, and a simple but powerful graphical user interface(GUI).

## INTRODUCTION

The original program, 3DGRAPE[2,3], of which 3DGRAPE/AL is an updated version, is a batch-type program. It reads in pre-defined input data, generates the grid, and writes it out. For those boundary surfaces which are of interest (i.e., the body) it expects to read X,Y,Z coordinates of surface grid points which the user has pre-defined using other software. Other boundary surfaces of less interest (i.e., the outer boundary) can be found by the program itself using simple analytic shapes. The grid can consist of multiple blocks, and the program is capable of finding its own internal block-to-block boundary surfaces. Volume grid points are found by numerically solving the Poisson equations. The Steger & Sorenson (S&S) Right-Hand-Side (RHS) terms (i.e., forcing functions) in those equations are of a type which allows the user to choose the desired cell height on a read-in boundary, after which the program automatically finds the actual numerical values for the RHS terms which yield the desired cell heights. In the process the RHS terms attempt to give local near-orthogonality in the region of those same read-in surfaces. The cell heights the user requires may be of any magnitude (limited only by the precision of the computer), appropriate for both viscous and inviscid aerodynamic flow modeling. The input data is ordinary text, with required formatting. The output grid may be any of three formats, including the commonly used PLOT3D[4] formats.

All the features described above for the original program are preserved in the new program, and a significant suite of new features is added. Those new features include:

- Grid quality is enhanced by re-formulated Steger & Sorenson (S&S) control terms in the Poisson Equations. The user may specify arbitrary angles with which lines are to intersect boundaries, rather than that specification being limited to 90° everywhere. The treatment of sharp corners which transverse boundary surfaces (e.g., a grid wrapping around an airplane fuselage which has a strake) is improved using this capability.
- Another improvement to grid quality is the addition of Thomas & Middlecoff (T&M) clustering terms for cases where all six faces of a block are read-in, as found in GRIDGEN and 3DMAGGS. The user can choose either the Steger & Sorenson terms (as in the original code and improved as described above), the

Thomas & Middlecoff type terms, or a blending between the two which gives good cell-size and skewness control at both the boundaries and the interior.

- Grid quality is evaluated by computing and printing maxima, minima, medians, and averages of cell heights and non-orthogonality, at boundaries and in the interiors of the blocks of the finished grid.
- Initialization is improved by Trans-Finite Interpolation[5] (for cases with six fixed boundary surfaces). In some cases grids initialized thusly can serve as the final grid, in others this improved initialization speeds convergence.
- Erlich's Ad Hoc Method for computing locally optimum relaxation parameters is available for the code's SOR solver. This also can accelerate convergence
- When installed on CRAY computers the code is vectorized in all three coordinate directions, allowing the longest possible vector length in each block. This, too, accelerates convergence.
- The grid generation iteration schedule can be divided into parts. Parameters which effect convergence (such as relaxation rates), as well as the type of clustering terms used and their associated decay rates, are adjustable with each part. Intermediate solutions and restart files can be written after each part. Thus, in practical operation, as much can sometimes be accomplished in one run with this program as in multiple runs with other grid generators.
- An input filter called PREGRAPE/AL, taken from 3DMAGGS, is supplied as a companion program. It inputs the output from the GRIDGEN[6] code, which contains blocking strategy and surface grids, and turns that into input for 3DGRAPE/AL.
- Required cell heights and skewness at read-in surfaces can be specified by the user at each point from a file. One possible application of the ability to specify the skewness at each point is in hypersonic flow where in the exit plane near the shock the flow is aligned with the shock and so should be the grid, while in that same face near the body the grid should be aligned with the body surface, and the angles are blended in-between.
- A complete grid generated elsewhere can be read-in, and the elliptic solver can be run a few steps to smooth the grid.
- A Graphical User Interface, coded in FORTRAN-77 and calling the IRIS Graphics Library, allows the user to watch selected grid surfaces while the grid solver is iterating. A full suite of transforms and other features is included.

The code is written in FORTRAN-77. It can be installed as an ordinary batch program, and in that form it should run on almost any computer. Alternately, on a Silicon Graphics Inc. (SGI) workstation it can be installed along with its graphical user interface. The GUI is also written in FORTRAN-77, and calls functions in the IRIS Graphics Library. For compiling on a CRAY supercomputer there is a vectorized batch version.

## THEORETICAL DEVELOPMENT OF THE POISSON EQUATIONS IN PHYSICAL SPACE

The original 3DGRAPE program, the 3DMAGGS program, and the new 3DGRAPE/AL program all generate grids by iteratively solving the Poisson Equations in three-dimensions. A mapping is thus found between the computational coordinates $\xi,\eta,\zeta$ and the physical coordinates X,Y,Z. The equations are typically given in the computational space as

$$\xi_{xx} + \xi_{yy} + \xi_{zz} = P(\xi,\eta,\zeta) \tag{1a}$$

$$\eta_{xx} + \eta_{yy} + \eta_{zz} = Q(\xi,\eta,\zeta) \tag{1b}$$

$$\zeta_{xx} + \zeta_{yy} + \zeta_{?zz} = R(\xi,\eta,\zeta) \tag{1c}$$

However, it is natural to apply them in the physical space. It is natural to specify the grid boundary conditions by giving X,Y,Z at fixed values of $\xi,\eta,\zeta$ rather than to give values of $\xi,\eta,\zeta$ at fixed values of X,Y,Z. The transformation of Eqs. 1 to physical space proceeds as follows. Clearly, we must have

$$\xi = \xi(x,y,z) \tag{2a}$$

$$\eta = \eta(x,y,z) \tag{2b}$$

$$\zeta = \zeta(x,y,z) \tag{2c}$$

To effect this transformation we must also have

$$x = x(\xi,\eta,\zeta) \tag{3a}$$

$$y = y(\xi,\eta,\zeta) \tag{3b}$$

$$z = z(\xi,\eta,\zeta) \tag{3c}$$

Differentiating Eqs. 2 and applying the chain rule gives

$$\begin{vmatrix} d\xi \\ d\eta \\ d\zeta \end{vmatrix} = \begin{vmatrix} \xi_x & \xi_y & \xi_z \\ \eta_x & \eta_y & \eta_z \\ \zeta_x & \zeta_y & \zeta_z \end{vmatrix} \begin{vmatrix} dx \\ dy \\ dz \end{vmatrix} \tag{4}$$

Likewise, differentiating Eqs. 3 and applying the chain rule gives

$$\begin{vmatrix} dx \\ dy \\ dz \end{vmatrix} = \begin{vmatrix} x_\xi & x_\eta & x_\zeta \\ y_\xi & y_\eta & y_\zeta \\ z_\xi & z_\eta & z_\zeta \end{vmatrix} \begin{vmatrix} d\xi \\ d\eta \\ d\zeta \end{vmatrix} \tag{5}$$

We designate the 3 x 3 matrix in Eq. 5 as M, assume that its inverse exists, and pre-multiply both sides of Eq. 5 by $M^{-1}$. This gives

$$M^{-1} \begin{vmatrix} dx \\ dy \\ dz \end{vmatrix} = \begin{vmatrix} d\xi \\ d\eta \\ d\zeta \end{vmatrix} \tag{6}$$

Substituting from Eq. 6 into Eq. 4 gives

$$M^{-1}\begin{vmatrix} dx \\ dy \\ dz \end{vmatrix} = \begin{vmatrix} \xi_x & \xi_y & \xi_z \\ \eta_x & \eta_y & \eta_z \\ \zeta_x & \zeta_y & \zeta_z \end{vmatrix} \begin{vmatrix} dx \\ dy \\ dz \end{vmatrix} \tag{7}$$

We know that, in general, if

$$A\vec{v} = B\vec{v} \tag{8a}$$

and if $B^{-1}$ exists then

$$B^{-1}\, A\, \vec{v} = \vec{v} \tag{8b}$$

Therefore it must be true that

$$B^{-1}\, A = I \tag{8c}$$

Pre-multiplying by B gives

$$A = B \tag{8d}$$

Applying Eqs. 8 to Eq. 7 gives

$$M^{-1} = \begin{vmatrix} \xi_x & \xi_y & \xi_z \\ \eta_x & \eta_y & \eta_z \\ \zeta_x & \zeta_y & \zeta_z \end{vmatrix} \tag{9}$$

For this to be useful, we must find $M^{-1}$. It is known, in general, that

$$A^{-1} = \frac{\mathrm{Adj}(A)}{\mathrm{Det}(A)} \tag{10}$$

Where Adj(A) is the adjoint of A and Det(A) is the determinant of A. The adjoint of A is a matrix having as each element the corresponding cofactor of A. Thus, from Eq. 9, we have

$$\begin{vmatrix} \xi_x & \xi_y & \xi_z \\ \eta_x & \eta_y & \eta_z \\ \zeta_x & \zeta_y & \zeta_z \end{vmatrix} = \begin{vmatrix} \gamma_{11} & \gamma_{12} & \gamma_{13} \\ \gamma_{21} & \gamma_{22} & \gamma_{23} \\ \gamma_{31} & \gamma_{32} & \gamma_{33} \end{vmatrix} / J \tag{11}$$

where $\gamma_{ij}$ is the ij-th cofactor of M and J is the determinant of M. By inspection of Eq. 11 we see that

$$\xi_x = \gamma_{11}/J \tag{12a}$$

$$\xi_y = \gamma_{12}/J \tag{12b}$$

$$\xi_z = \gamma_{13}/J \tag{12c}$$

$$\eta_x = \gamma_{21}/J \tag{12d}$$

$$\eta_y = \gamma_{22}/J \tag{12e}$$

$$\eta_z = \gamma_{23}/J \tag{12f}$$

$$\zeta_x = \gamma_{31}/J \tag{12g}$$

$$\zeta_y = \gamma_{32}/J \tag{12h}$$

$$\zeta_z = \gamma_{33}/J \tag{12i}$$

Completion of the derivation of the transformed Poisson equations requires further differentiating the metrics in Eqs. 12, substituting them into Eqs. 1, and collecting terms. This process is simple calculus, but very lengthy and beyond the scope of this paper. The result is

$$\begin{aligned} &\alpha_{11}\vec{r}_{\xi\xi} + \alpha_{22}\vec{r}_{\eta\eta} + \alpha_{33}\vec{r}_{\zeta\zeta} \\ &+ 2\,(\,\alpha_{12}\vec{r}_{\xi\eta} + \alpha_{13}\vec{r}_{\xi\zeta} + \alpha_{23}\vec{r}_{\eta\zeta}) \\ &= -J^2\,(P\vec{r}_\xi + Q\vec{r}_\eta + R\vec{r}_\zeta) \end{aligned} \tag{13a}$$

where:

$$\vec{r} = \left|\overrightarrow{\begin{matrix} x \\ y \\ z \end{matrix}}\right| \tag{13b}$$

and

$$\alpha_{ij} = \sum_{m=1}^{3} \gamma_{mi}\gamma_{mj} \tag{13c}$$

## THEORETICAL DEVELOPMENT OF IMPROVED S&S RHS TERMS

The distribution of points in the grid results primarily from the influence of the Right-Hand Side (RHS) terms, or forcing functions. We are free to choose them as we please. In both new and old programs they are:

$$\begin{aligned} P(\xi,\eta,\zeta) = {} & P_1(\eta,\zeta)e^{-a\xi} + P_2(\eta,\zeta)e^{-a(\xi_{max}-\xi)} \\ & + \; P_3(\xi,\zeta)e^{-a\eta} + P_4(\xi,\zeta)e^{-a(\eta_{max}-\eta)} \\ & + \; P_5(\xi,\eta)e^{-a\zeta} + P_6(\xi,\eta)e^{-a(\zeta_{max}-\zeta)} \end{aligned} \tag{14a}$$

$$Q(\xi,\eta,\zeta) = Q_1(\eta,\zeta)e^{-a\xi} + Q_2(\eta,\zeta)e^{-a(\xi_{max}-\xi)} + Q_3(\xi,\zeta)e^{-a\eta} + Q_4(\xi,\zeta)e^{-a(\eta_{max}-\eta)} + Q_5(\xi,\eta)e^{-a\zeta} + Q_6(\xi,\eta)e^{-a(\zeta_{max}-\zeta)} \tag{14b}$$

$$R(\xi,\eta,\zeta) = R_1(\eta,\zeta)e^{-a\xi} + R_2(\eta,\zeta)e^{-a(\xi_{max}-\xi)} + R_3(\xi,\zeta)e^{-a\eta} + R_4(\xi,\zeta)e^{-a(\eta_{max}-\eta)} + R_5(\xi,\eta)e^{-a\zeta} + R_6(\xi,\eta)e^{-a(\zeta_{max}-\zeta)} \tag{14c}$$

Clearly, these RHS terms P,Q,R are simply superpositions of other terms $P_n,Q_n,R_n$ for $1 \le n \le 6$, multiplied by exponentials which are at their maximum value, one, at the boundary surfaces and which decay with distance into the interior of the block. The positive constant "a" in Eqs. 14 is set by the user, and determines the rate of exponential decay in the size and influence of the RHS terms.

At this point we must introduce a nomenclature for the face numbers. It is seen in Table I. By examining that nomenclature we see that at each of the boundaries the terms in P,Q,R having their subscripts equal to the face number are non-zero, and the other terms in P,Q,R approach zero due to the behavior of their exponential factors. At face 3, for example, Eqs. 14 reduce to:

$$P(\xi,\eta,\zeta) = P_3(\xi,\zeta) \tag{15a}$$

$$Q(\xi,\eta,\zeta) = Q_3(\xi,\zeta) \tag{15b}$$

$$R(\xi,\eta,\zeta) = R_3(\xi,\zeta) \tag{15c}$$

So then we can find the terms $P_n,Q_n,R_n$ at face n by considering each face in turn. At each point on each face we:

- Assume that the Poisson Equations, Eqs. 13, are satisfied.
- Find values for all first and second partial derivatives required by Eqs. 13.
- Eqs. 13 reduce to a 3 x 3 set of linear equations in the three unknowns $P_n,Q_n,R_n$. Solve them.

Having found all the $P_n,Q_n,R_n$, for $1 \le n \le 6$, we can calculate P,Q,R at all points in the grid from Eqs. 14.

However, finding values for all first and second partial derivatives at each face is not trivial. To further illustrate this we must restrict our attention to a particular face. We choose face 3 to illustrate. On face 3 the derivatives $\vec{r}_\xi$, $\vec{r}_\zeta$, $\vec{r}_{\xi\zeta}$, $\vec{r}_{\xi\xi}$, and $\vec{r}_{\zeta\zeta}$ can be found by differencing known boundary face points. The derivatives $\vec{r}_{\eta\eta}$ are found by differencing the grid solution at the current time step, as described on page 78 of Ref. 2. If we could find derivatives $\vec{r}_\eta$ we could then difference them to find derivatives $\vec{r}_{\xi\eta}$ and $\vec{r}_{\eta\zeta}$.

We find derivatives $\vec{r}_\eta$ by adding additional equations which embody the user's requirements on cell height and skewness. In the old 3DGRAPE method we added the three equations

$$\vec{r}_\xi \cdot \vec{r}_\eta = 0 \tag{16a}$$

$$\vec{r}_\eta \cdot \vec{r}_\zeta = 0 \tag{16b}$$

$$\vec{r}_\eta \cdot \vec{r}_\eta = S^2 \tag{16c}$$

As seen in Table 1, $\xi$ and $\zeta$ vary over face 3, and $\eta$ varies along lines intersecting the face. Thus Eqs. 16a and 16b require orthogonality between the lines intersecting the face and the coordinate lines running over the face. Eq. 16c requires that the cell height on the surface be the positive constant S.

It is at this point that the old 3DGRAPE method and the new 3DGRAPE/AL method differ. In the new method we realize that when making grids about real-world configurations, with singularities and slope discontinuities, it is sometimes necessary to have grid cells which are skewed in a specified way. Lacking this ability, an inconsistency can develop which can either cause the elliptic solver to not converge, or result in an unsuitable grid. And so Eqs. 16 are replaced by

$$\vec{r}_\xi \cdot \vec{r}_\eta = |\vec{r}_\xi| \cdot |\vec{r}_\eta| \cos\theta_1 \tag{17a}$$

$$\vec{r}_\eta \cdot \vec{r}_\zeta = |\vec{r}_\eta| \cdot |\vec{r}_\zeta| \cos\theta_2 \tag{17b}$$

$$\vec{r}_\eta \cdot \vec{r}_\eta = S^2 \tag{17c}$$

where $\theta_1$ is the angle between the coordinate line intersecting face 3 and the line of varying $\xi$ on face 3, and $\theta_2$ is the angle between the coordinate line intersecting face 3 and the line of varying $\zeta$ on face 3. For $\theta_1$ and $\theta_2$ equal to 90°, Eqs. 17 reduce to Eqs. 16.

We now proceed to solve Eqs. 17 for $\vec{r}_\eta$. Expanding, we have

$$x_\xi x_\eta + y_\xi y_\eta + z_\xi z_\eta = c_1 \tag{18a}$$

$$x_\zeta x_\eta + y_\zeta y_\eta + z_\zeta z_\eta = c_2 \tag{18b}$$

$$x_\eta^2 + y_\eta^2 + z_\eta^2 = S^2 \tag{18c}$$

where

$$c_1 = |\vec{r}_\xi|\, S \cos\theta_1$$

$$c_2 = |\vec{r}_\zeta|\, S \cos\theta_2$$

$C_1$ and $c_2$ are constants because $\theta_1$, $\theta_2$, S, and the points on face 3 are user-defined inputs. Equations 18 are three equations in the three unknowns $x_\eta, y_\eta, z_\eta$ which are the elements of $\vec{r}_\eta$. But because Eq. 18c is quadratic, solving this set of equations is not straightforward. We will make an assumption about one of the unknowns and solve, make that assumption about another of the unknowns and solve, and then make that assumption about the last of the unknowns and solve. We will then select the answer which is "best."

The first assumption we make is that $x_\eta$ is a constant. Terms involving $x_\eta$ in Eqs. 18a and 18b are brought to the right side of the equations, and then the equations are solved, yielding

$$y_\eta = x_\eta \gamma_{22} / \gamma_{12} + k_1 \tag{19a}$$

$$z_\eta = x_\eta \gamma_{32} / \gamma_{12} + k_2 \tag{19b}$$

where

$$k_1 = \frac{c_1 z_\zeta - c_2 z_\xi}{-\gamma_{12}}$$

$$k_2 = \frac{c_2 y_\xi - c_1 y_\zeta}{-\gamma_{12}}$$

K1 and k2 are constants. Then from Eq. 18c

$$x_\eta = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a} \tag{20}$$

where

$$a = 1 + (\gamma_{22}/\gamma_{12})^2 + (\gamma_{32}/\gamma_{12})^2$$

$$b = \frac{2}{\gamma_{12}}(k_1\gamma_{12} + k_2\gamma_{32})$$

$$c = k_1^2 + k_2^2 - S^2$$

The second assumption is that $y_\eta$ is a constant. Terms involving $y_\eta$ in Eqs. 18a and 18b are brought to the right side of the equations, and then the equations are solved, yielding

$$x_\eta = y_\eta\gamma_{12} / \gamma_{22} + k_1 \tag{21a}$$

$$z_\eta = y_\eta\gamma_{32} / \gamma_{22} + k_2 \tag{21b}$$

where

$$k_1 = \frac{c_1 z_\zeta - c_2 z_\xi}{\gamma_{22}}$$

$$k_2 = \frac{c_2 x_\xi - c_1 x_\zeta}{\gamma_{22}}$$

Then from Eq. 18c

$$y_\eta = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a} \tag{22}$$

where

$$a = 1 + (\gamma_{12}/\gamma_{22})^2 + (\gamma_{32}/\gamma_{22})^2$$

$$b = \frac{2}{\gamma_{22}}(k_1\gamma_{12} + k_2\gamma_{32})$$

and c is the same as above, in Eq. 20.

The third assumption is that $z_\eta$ is a constant. Terms involving $z_\eta$ in Eqs. 18a and 18b are brought to the right side of the equations, and then the equations are solved, yielding

$$x_\eta = z_\eta \gamma_{12} / \gamma_{32} + k_1 \tag{23a}$$

$$y_\eta = z_\eta \gamma_{22} / \gamma_{32} + k_2 \tag{23b}$$

where

$$k_1 = \frac{c_1 y_\zeta - c_2 y_\xi}{-\gamma_{32}}$$

$$k_2 = \frac{c_2 x_\xi - c_1 x_\zeta}{-\gamma_{32}}$$

Then from Eq. 17c

$$z_\eta = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a} \tag{24}$$

where

$$a = 1 + (\gamma_{12}/\gamma_{32})^2 + (\gamma_{22}/\gamma_{32})^2$$

$$b = \frac{2}{\gamma_{32}}(k_1\gamma_{12} + k_2\gamma_{22})$$

and c is the same as above, in Eq. 20.

In general, none of these three assumptions is strictly correct. However, it usually turns out that at least one of them is close enough to correct for this method to generate suitable grids. It was said that we would choose whichever of these three solutions was "best." However, Eqs. 20, 22, and 24 each include an ambiguous sign from a square-root operation. Therefore, we actually have six solutions to choose from. Using each of the six solutions we compute the Jacobian. If the coordinates in the block are right-handed (with the "handedness" being a user-defined input) we choose the solution which yields the largest positive Jacobian. If the coordinates in the block are left-handed we choose the solution which yields the largest negative Jacobian. The logic behind choosing based upon the Jacobian is that Jacobians, as defined above, having large absolute values seem to be present in grids which are more orthogonal, and, conversely, Jacobians having small absolute values seem to be present in grids which are highly skewed. Thus the elements of $\vec{r}_\eta$ are found.

The foregoing is the analysis for face 3. The analysis for face 4 appears identical, differing only in some of the difference formulas. The analyses for faces 1, 2, 5, and 6 follow in a straightforward manner from the foregoing example.

This formulation for the S&S RHS terms requires a lot of computation but most of it is done only once, at the start of the iteration schedule. It was said, above, that having all values for the derivatives at the face, those derivatives are substituted into Eqs. 13, yielding a 3 x 3 set of linear equations in the three unknowns $P_n, Q_n, R_n$. Their solution shows $P_n, Q_n, R_n$ to be linear functions of the second derivatives $\vec{r}_{\eta\eta}$ which are found by differencing at each time step. The coefficients in those linear functions are fixed for all computational time. Therefore, the only computation necessary to find the RHS terms in each iteration is to re-evaluate $\vec{r}_{\eta\eta}$ , re-evaluate the linear functions to get $P_n, Q_n, R_n$ at each face, and then use Eqs. 14 to re-compute the P,Q,R at every point in the grid.

The effectiveness of this method is seen in Figure 1. When wrapping a grid around a sharp edge it is necessary to cause the lines intersecting the surface near the edge to bend toward the edge for best results. The ultimate example of wrapping a grid around a sharp edge is to wrap it around the edge of a flat plate. Figure 1 shows a wing with a zero-thickness extension in the spanwise direction, and a C-H type grid around it. Thus, it is necessary to wrap a C-type grid around the leading-edge of that wing and its flat-plate extension. This would not have been possible with the old type RHS terms.

## THOMAS AND MIDDLECOFF CLUSTERING TERMS

When making grids in regions where all six faces of the computational cube are fixed it is sometimes advantageous to use clustering functions where the spacing normal to a face is determined by the spacing on the side walls. The Thomas and Middlecoff[7] clustering terms are included here for that purpose. However, the Thomas and Middlecoff clustering terms

$$P = \Phi(\nabla\xi \,.\, \nabla\xi) \tag{25a}$$

$$Q = \Psi(\nabla\eta \,.\, \nabla\eta) \tag{25b}$$

$$R = \Omega(\nabla\zeta \,.\, \nabla\zeta) \tag{25c}$$

where

$$\Phi = \frac{\vec{r}_\xi \,.\, \vec{r}_{\xi\xi}}{\vec{r}_\xi \,.\, \vec{r}_\xi} \tag{25d}$$

$$\psi = \frac{\vec{r}_\eta \,.\, \vec{r}_{\eta\eta}}{\vec{r}_\eta \,.\, \vec{r}_\eta} \tag{25e}$$

$$\Omega = \frac{\vec{r}_\zeta \,.\, \vec{r}_{\zeta\zeta}}{\vec{r}_\zeta \,.\, \vec{r}_\zeta} \tag{25f}$$

are given in the computational space, and to be useful here they must be converted to physical space. Applying the definition of the $\nabla$ operator, illustrated by

$$\nabla\xi = \xi_x\hat{j} + \xi_y\hat{k} + \xi_z\hat{l} \tag{26}$$

where $\hat{j}$, $\hat{k}$, and $\hat{l}$ are the unit normal vectors, and reducing, gives

$$P = \Phi(\xi_x^2 + \xi_y^2 + \xi_z^2) \tag{27a}$$

$$Q = \psi(\eta_x^2 + \eta_y^2 + \eta_z^2) \tag{27b}$$

$$R = \Omega(\zeta_x^2 + \zeta_y^2 + \zeta_z^2) \tag{27c}$$

Substituting the metrics shown in Eqs. 12 into Eqs. 27, and expanding and re-grouping, gives

$$P = \Phi\left[\left(\vec{r}_\eta \cdot \vec{r}_\eta\right)\left(\vec{r}_\zeta \cdot \vec{r}_\zeta\right) - \left(\vec{r}_\eta \cdot \vec{r}_\zeta\right)\right]/J^2 \tag{28a}$$

$$Q = \psi\left[\left(\vec{r}_\xi \cdot \vec{r}_\xi\right)\left(\vec{r}_\zeta \cdot \vec{r}_\zeta\right) - \left(\vec{r}_\xi \cdot \vec{r}_\zeta\right)\right]/J^2 \tag{28b}$$

$$R = \Omega\left[\left(\vec{r}_\xi \cdot \vec{r}_\xi\right)\left(\vec{r}_\eta \cdot \vec{r}_\eta\right) - \left(\vec{r}_\xi \cdot \vec{r}_\eta\right)\right]/J^2 \tag{28c}$$

These RHS terms generate good grids in many applications. An exception is the situation where the opposing side boundaries, from which the T&M terms are calculated, have very different clustering characteristics. In these cases instabilities in the Poisson solver can result.

It was found in the development of 3DMAGGS that S&S clustering terms tend to give the most-nearly-orthogonal grids near boundaries, while T&M clustering terms give the best clustering in the interior of the blocks. And so a blending between the two kinds of RHS terms was developed, and is included in 3DGRAPE/AL.

## OPTIMUM RELAXATION PARAMETER

3DGRAPE/AL solves the 3-D Poisson equations using Point Successive Over Relaxation (PSOR). In PSOR there is a relaxation parameter, $\Omega$, which determines the rate of convergence and stability of the method. In the old program the $\Omega$ was fixed for all computational time. That option is still available in the new code as well. However, the new code also has an algorithm to compute an optimum relaxation parameter at every point in the grid using the method of Erlich.[8]

That method requires the equations being solved, here Eq. 13a, to be represented as a difference equation of the following form:

$$\begin{aligned} a_0\vec{r}_{j,k,l} &+ a_1\vec{r}_{j+1,k,l} + a_2\vec{r}_{j,k+1,l} + a_3\vec{r}_{j,k,l+1} \\ &+ a_4\vec{r}_{j-1,k,l} + a_5\vec{r}_{j,k-1,l} + a_6\vec{r}_{j,k,l-1} = \vec{b}_{j,k,l} \end{aligned} \tag{29}$$

Applying standard central differences to all first and second partial derivatives in Eq. 13a, and collecting terms, we arrive at the form of Eq. 29, where

$$a_0 = -2\left(\frac{\alpha_{11}}{(\Delta\xi)^2} + \frac{\alpha_{22}}{(\Delta\eta)^2} + \frac{\alpha_{33}}{(\Delta\zeta)^2}\right) \tag{30a}$$

$$a_1 = \frac{\alpha_{11}}{(\Delta\xi)^2} + \frac{J^2 P}{2\Delta\xi} \tag{30b}$$

$$a_2 = \frac{\alpha_{22}}{(\Delta\eta)^2} + \frac{J^2 Q}{2\Delta\eta} \tag{30c}$$

$$a_3 = \frac{\alpha_{33}}{(\Delta\zeta)^2} + \frac{J^2 R}{2\Delta\zeta} \tag{30d}$$

$$a_4 = \frac{\alpha_{11}}{(\Delta\xi)^2} - \frac{J^2 P}{2\Delta\xi} \tag{30e}$$

$$a_5 = \frac{\alpha_{22}}{(\Delta\eta)^2} - \frac{J^2 Q}{2\Delta\eta} \tag{30f}$$

$$a_6 = \frac{\alpha_{33}}{(\Delta\zeta)^2} - \frac{J^2 R}{2\Delta\zeta} \tag{30g}$$

The complex eigenvalues of Eq. 29 at each point, ignoring wave numbers above 1, are

$$\mu = \mu_r + \mu_i = \frac{2}{a_0}\left(\sqrt{a_1 a_4}\cos\frac{\pi}{j_{max}+1} + \sqrt{a_2 a_5}\cos\frac{\pi}{k_{max}+1} + \sqrt{a_3 a_6}\cos\frac{\pi}{l_{max}+1}\right) \tag{31}$$

where $\mu_r$ and $\mu_i$ are the real and imaginary parts of $\mu$, respectively. It is required that $|\mu_r| < 1$.

Continuing with Erlich's method, as formulated by Steinbrenner, Chawner, and Fouts on pages 6-6 and 6-7 of Ref. 6 (with typographical errors corrected), we let

$$A = \mu_r^2 + \mu_i^2 \tag{32a}$$

$$B = \mu_r^2 - \mu_i^2 \tag{32b}$$

$$C = A^2 - B^2 \tag{32c}$$

$$D = A^2 - B \tag{32d}$$

$$E = \sqrt{C + D^2} \tag{32e}$$

$$F = \sqrt[3]{C} \tag{32f}$$

Then

$$\overline{\omega} = \frac{(3D + E)\,F\,\sqrt[3]{E-D} - (3D - E)\,F\,\sqrt[3]{E+D} + A^2 + 3B^2 - 4A^2B}{A^2 D} \tag{33}$$

and the relaxation parameter $\omega$ is

$$\omega = \left|\begin{array}{ll} -\left(\overline{\omega} - \sqrt{\overline{\omega}^2 + 4\overline{\omega}}\right)/2 & \text{if } D > 0 \\ -\left(\overline{\omega} + \sqrt{\overline{\omega}^2 + 4\overline{\omega}}\right)/2 & \text{if } D < 0 \end{array}\right| \tag{34}$$

This method can reduce the number of iterations required to achieve convergence. The $\omega$ so computed can sometimes be a little too large, and so cause instabilities. Therefore, in the code, they are multiplied by a limiting factor. The default value of this factor is 0.7, but a value of 0.6 was found to be necessary in one of the sample cases. These $\omega$ are dependent on the grid at its current time step, and so they are re-calculated each time step. As can be inferred from the above, computing them requires a significant amount of computer time, so the code has an option wherein they are re-calculated every n time steps.

## CURRENT AND FUTURE WORK

Extensions and improvements to 3DGRAPE/AL planned for the near future, or currently in progress, include:

- Adding cylindrical topology -- solving in $\rho,\theta,x$ -- to improve solutions about cylindrical axes, just as use of spherical topology is offered to improve solutions about polar axes.
- Write a translator to translate input for the old program 3DGRAPE into input for the new program 3DGRAPE/AL. This will make datasets used in the old program workable in the new program. Also, the latest release of GRIDGEN, Version 9, can generate input to the earlier 3DGRAPE program, and so this translator will provide an alternate pathway between the two codes.
- We will try again to build multigrid into the Poisson solver. We tried once before, but did not succeed. We know that in grids produced by this code the cell size increases in an approximately exponential fashion with distance from a boundary, but we cannot predict exactly what that rate of increase in cell height will be. Therefore, although we know the desired distance to the 1st node away from the boundary, we cannot say, exactly, what should be the distances to the 2nd, 4th, 8th, 16th, etc., nodes from the boundary. But we need those distances to operate at the various multigrid levels. And so we estimate. But those estimates are not accurate. Thus, there are inconsistencies between the equations being solved at the different multigrid levels, and therefore it does not converge. But we are determined to build in a multigrid capability if at all possible.
- Optimize the code to run faster on SGI machines, such as ONYX. Portability to other platforms will be preserved.
- Write a GUI for PREGRAPE/AL.
- Put in a new boundary condition wherein the angles of lines intersecting a fixed internal boundary is mimicked across that boundary.
- Make a version of the code which uses PVM to run the code in parallel on multiple platforms.

## ACKNOWLEDGMENTS.

The authors are grateful to Professor Joe Thompson for his pioneering work in elliptic grid generation, and for his gracious and encouraging example. Dr. Jeffrey Hultquist of NASA Ames Research Center provided invaluable help with the graphics; were it not for his patience, generosity, and expertise the GUI would not have been possible. Mr. Scott Thomas of Sterling Software assisted in vectorizing the code on the CRAY, and Dr. Jin Chou of Computer Sciences Corporation helped assisted with vector analysis. Dr. Jamshid S. Abolhassani of Computer Sciences Corporation provided various valuable insights and explanations. Mr. William Kleb of NASA Langley Research Center assisted in the formulation of LARCS and other interpolation issues. The second author gratefully acknowledges support from contract NAS1-19000. This work is dedicated to the memory of Joseph L. Steger: scientist, mentor, teacher, and friend.

## REFERENCES

[1]Alter, S. J., Weilmuenster, K. J., "The Three-Dimensional Multi-Block Advanced Grid Generation System (3DMAGGS)," NASA TM 108985, May, 1993.

[2]Sorenson, R. L., "The 3DGRAPE Book: Theory, Users' Manual, Examples," NASA TM 102224, July, 1989.

[3]Sorenson, R. L., "Three-Dimensional Zonal Grids About Arbitrary Shapes by Poisson's Equation," appearing in Sengupta, S., Häuser, J., Eiseman, P.R., and Taylor, C., eds., Numerical Grid Generation in Computational Fluid Mechanics, Pineridge Press Ltd., 1988.

[4]Walatka, P. P., Buning, P. G.,and Elson, P. A., "PLOT3D User's Manual," NASA TM 101067, July, 1992.

[5]Soni, B. K., "Two- and Three-Dimensional Grid Generation for Internal Flow Applications of Computational Fluid Dynamics," AIAA 85-1526, 1985.

[6]Steinbrenner, J. P., Chawner, J. R., and Fouts, C.L., The GRIDGEN 3D Multiple Block Grid Generation System," Wright Research and Development Center Report WRDC TR90-33022, October, 1989.

[7]Thomas, P. D., Middlecoff, J, F., "Direct Control of the Grid Point Distribution in Meshes Generated by Elliptic Equations," AIAA Journal, vol 18, pp, 652-656, June, 1979.

[8]Erlich, L. W., "An Ad Hoc SOR Method, "Journal of Computationall Physics, vol. 44, pp. 31-45, March, 1981.

**Table I. Face numbers.**

| Face no. n : | j | k | l | $\xi$ | $\eta$ | $\zeta$ |
|---|---|---|---|---|---|---|
| 1 | 1 | varying | varying | 0. | varying | varying |
| 2 | jmax | varying | varying | $\xi_{max}$ | varying | varying |
| 3 | varying | 1 | varying | varying | 0. | varying |
| 4 | varying | $k_{max}$ | varying | varying | $\eta_{max}$ | varying |
| 5 | varying | varying | 1 | varying | varying | 0. |
| 6 | varying | varying | $l_{max}$ | varying | varying | $\zeta_{max}$ |

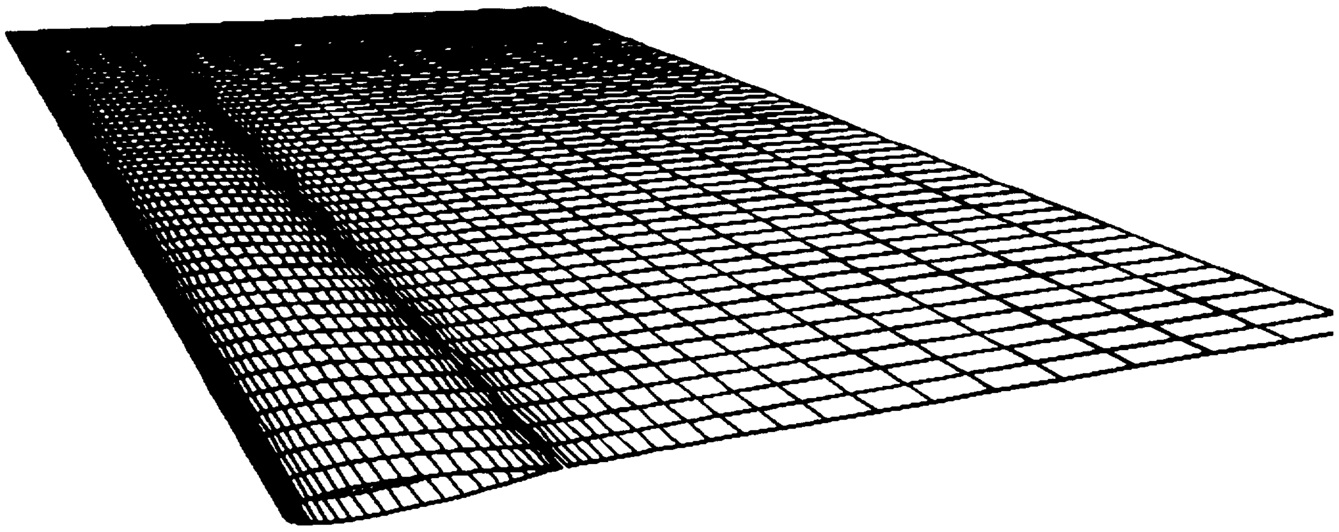

Figure 1a. Wing With Flat-Plate Extension in Both Spanwise and Downstream Directions.

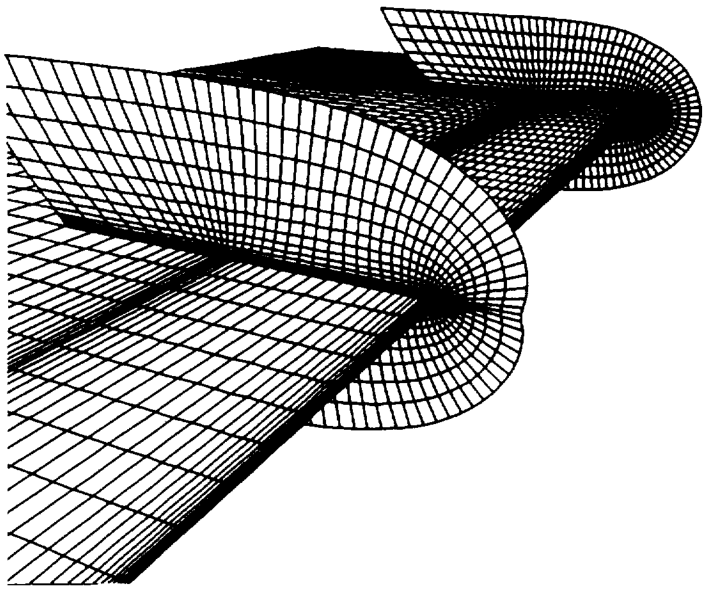

Figure 1b. View From Outboard End Showing Portions of Two Interior Grid Surfaces Wrapping Around Wing and Extension

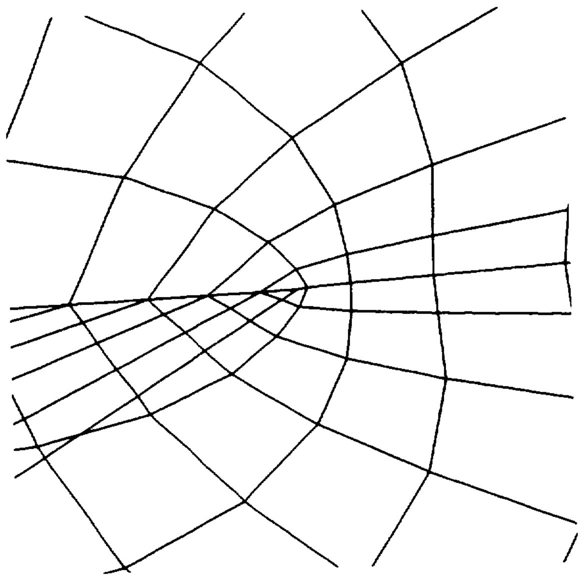

Figure 1c. Close-Up of Grid Surface Wrapping Around Leading Edge of Flat-Plate Extension