NASA Technical Memorandum 81198

A Computer Program to Generate Two-Dimensional Grids About Airfoils and Other Shapes by the Use of Poisson's Equation

Reese L. Sorenson

(NASA-TM-81198) A COMPUTER PROGRAM TO GENERATE TWO-DIMENSIONAL GRIDS ABOUT AIRFOILS AND OTHER SHAPES BY THE USE OF POISSON'S EQUATION (NASA) 62 p HC A04/MF A01

N80-26266

CSCL 01A G3/02

Unclas 23549

MAY 1980

REPRODUCED BY
U.S. DEPARTMENT OF COMMERCE
NATIONAL TECHNICAL
INFORMATION SERVICE
SPRINGFIELD, VA 22161

NASA
National Aeronautics and
Space Administration

NASA Technical Memorandum 81198

A Computer Program to Generate Two-Dimensional Grids About Airfoils and Other Shapes by the Use of Poisson's Equation

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

National Aeronautics and
Space Administration

Ames Research Center
Moffett Field, California 94035

A COMPUTER PROGRAM TO GENERATE TWO DIMENSIONAL GRIDS ABOUT AIRFOILS AND OTHER SHAPES BY THE USE OF POISSON'S EQUATION

AMES RESEARCH CENTER
MOFFETT FIELD, CA

MAY 80

TABLE OF CONTENTS

A COMPUTER PROGRAM TO GENERATE TWO-DIMENSIONAL GRIDS ABOUT AIRFOILS AND OTHER SHAPES BY THE USE OF POISSON'S EQUATION

Reese L. Sorenson

Ames Research Center

SUMMARY

A method for generating two-dimensional finite-difference grids about airfoils and other shapes by the use of the Poisson differential equation is developed. The inhomogeneous terms are automatically chosen such that two important effects are imposed on the grid at the inner (airfoil) boundary and at the outer boundary. The first effect is control of the spacing between mesh points, along mesh lines intersecting the boundaries. The second effect is control of the angles with which mesh lines intersect the boundaries. A FORTRAN computer program has been written to use this method. The program is available upon request from the Applied Computational Aerodynamics Branch, Mail Stop 202A-14, NASA Ames Research Center, Moffett Field, Calif. 94035. A description of the program, a discussion of the control parameters, and a set of sample cases are included.

INTRODUCTION

One of the most desirable characteristics of a method for generating grids, including those about airfoils, is that it be able to treat arbitrary boundary shapes. In a grid used for computing aerodynamic flow over an airfoil, or over any other body shape, the surface of the body is usually treated as boundary (hereinafter referred to as the "inner boundary") and it is desirable that the method offer complete freedom in choosing that body shape. Aerodynamic surfaces in the real world are often not represented as analytic functions and can include a number of "sharp corners," or points where the slope would be discontinuous. A method that requires that aerodynamic surfaces be several-times-differentiable analytic functions is one with a severe limitation. The same comments often apply to the outer boundary as well. It is also desirable that one be able to arbitrarily choose the distribution of points on boundaries, so that one could cluster points at regions where the greatest difficulty is encountered in solving the governing equations of flow.

Another critically important characteristic in a grid-generation technique is the ability to specify the spacing between mesh points at the boundary, in the direction normal to the boundary. The spacing required between the body and the adjacent mesh line of the same family can vary by orders of magnitude, depending on, for example, whether the flow model being used is viscous or inviscid.

Orthogonality is another desirable feature. If absolute analytic orthogonality at every mesh point can be guaranteed, certain terms can be deleted from the governing flow equations, thus facilitating their solution. Even if nonorthogonality is to be accepted, one still desires near-orthogonality. The opposite situation, extreme cell skewness, brings about slow numerical convergence or inaccuracies or both. This is especially important at the boundaries, since it is at the boundaries that most difficulties usually arise.

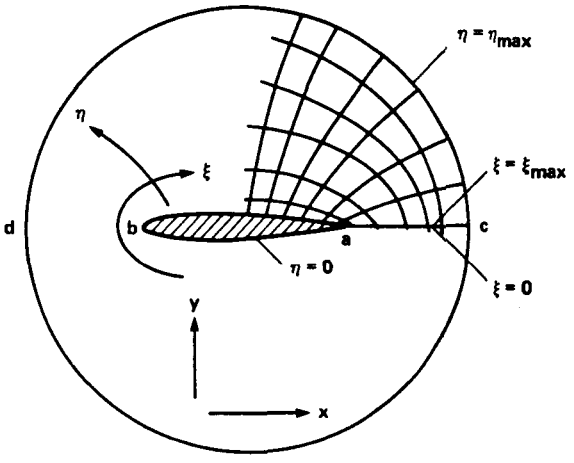
Of the many methods for generating grids, two classes of methods that have had widespread application are geometric construction and conformal mappings. By geometric construction it is meant that simple geometric shapes, such as lines, conic sections, quadratic curves, etc., are combined to form grids. Although these methods have good features, such as simplicity and computational ease in the case of geometric construction and orthogonality in the case of conformal mappings, they generally leave something to be desired in the area of applicability. The classes of problems they can treat are limited.

The method presented in this paper has no such limitation. It can utilize any boundary shape, even one specified by tabulated points and including a limited number of sharp corners. Any distribution of points on boundaries is acceptable. Spacing normal to the boundaries can be arbitrarily specified, and mesh spacing varies smoothly between boundaries. Control of angles at boundaries is imposed.

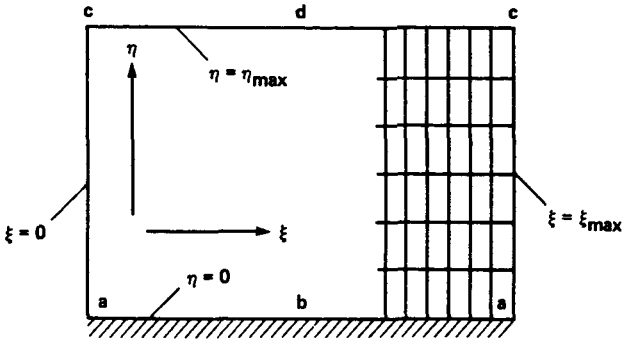
The computer program to implement this method is modular, and otherwise logically simple, contains many comments, and should be easily transportable. It is numerically stable and computationally fast. The following sections of this paper provide generous documentation. It is the intention of the Applied Computational Aerodynamics Branch at Ames Research Center to continue to support the code.

THEORETICAL DEVELOPMENT

Let the Cartesian coordinates x,y denote points in the real, or physical space, wherein the airfoil and airflow reside. A grid is essentially a mapping between real space and a computational space ξ,η for $0 \le \xi \le \xi_{max}$ and $0 \le \eta \le \eta_{max}$ (see fig. 1). The boundary $\eta = 0$ is mapped into the inner boundary (the airfoil) with $\xi = 0$ at the trailing edge and ξ increasing clockwise around the airfoil. The boundary $\eta = \eta_{max}$ is mapped into the outer boundary in a similar manner. The boundary $\xi = 0$ is mapped into the grid line proceeding rearward from the trailing edge to the outer boundary. The grid is said to be periodic in that if there was a grid line at $\xi = \xi_{max} + \Delta\xi$, it would be coincident with the line at $\xi = 0$. The grid is made of two families of lines: the ξ = constant family, which connect the inner boundary to the outer boundary, and the η = constant family, which form closed curves around the airfoil. Because one family of lines forms closed curves, the grid described above is referred to as an "O-type" grid.



(a) Physical space.



(b) Computational space.

Figure 1.- Topology for O-type grids.

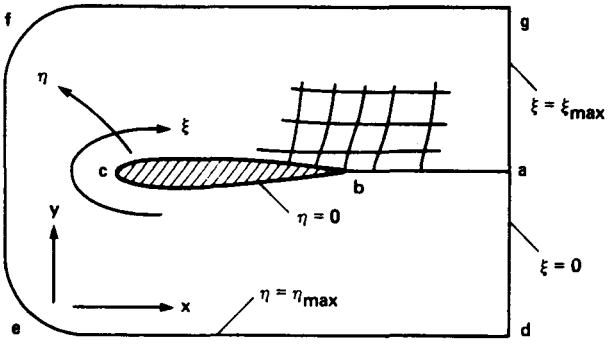
The program described in the following sections can generate either the above O-type grids or "C-type" grids, as illustrated in figure 2. In a C-type grid the $\eta = 0$ boundary goes forward from the rear boundary to the trailing edge, clockwise around the airfoil, and then rearward again. The η = constant family of lines form open curves resembling a letter C.

Let $\xi = \xi(x,y)$ and $\eta = \eta(x,y)$ specify the mapping from the physical space to the computational space. The basis of this method is that, following Thompson et al. (ref. 1), the mapping functions are required to satisfy the Poisson equations

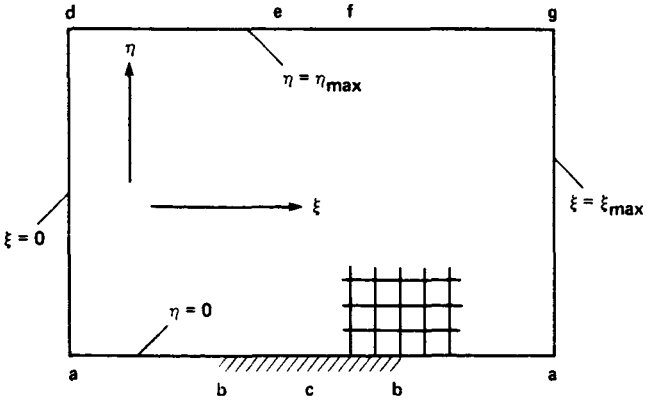
$$\xi_{xx} + \xi_{yy} = P \qquad (1a)$$

$$\eta_{xx} + \eta_{yy} = Q \qquad (1b)$$

The following relations are useful in transforming equations between computational space and physical space:



(a) Physical space.



(b) Computational space.

Figure 2.- Topology for C-type grids.

$$\xi_x = y_\eta / J \tag{2a}$$

$$\xi_y = -x_\eta / J \tag{2b}$$

$$\eta_x = -y_\xi / J \tag{2c}$$

$$\eta_y = x_\xi / J \tag{2d}$$

where

$$J = x_\xi y_\eta - x_\eta y_\xi \tag{2e}$$

Applying equations (2) to equations (1) yields the transformed Poisson equations

$$\alpha x_{\xi\xi} - 2\beta x_{\xi\eta} + \gamma x_{\eta\eta} = -J^2(Px_\xi + Qx_\eta) \tag{3a}$$

$$\alpha y_{\xi\xi} - 2\beta y_{\xi\eta} + \gamma y_{\eta\eta} = -J^2(Py_\xi + Qy_\eta) \tag{3b}$$

where

$$\alpha = x_\eta^2 + y_\eta^2 \tag{3c}$$

$$\beta = x_\xi x_\eta + y_\xi y_\eta \tag{3d}$$

$$\gamma = x_\xi^2 + y_\xi^2 \tag{3e}$$

Solving equations (3), for a particular choice of inhomogeneous terms P and Q (also known as right-hand-side terms or forcing functions), and for a particular set of boundary conditions, causes a grid to be generated.

However, a great latitude exists in grids so generated due to the ability to choose the P and Q terms. If $P = Q = 0$, the Poisson equations degenerate to Laplace equations, and a basic grid results. Different choices for P and Q produce different grids. The challenge is to choose P and Q so that a desirable grid results with a reasonable amount of effort, both computational and human. In the present work, which is an extension of an idea of Steger and Sorenson (ref. 2), P and Q are defined in terms of four new variables. Four geometrical constraints are set forth which translate into new equations in the new variables. Including the Poisson equations we have six equations in six unknowns, which can then be solved in a straightforward iterative manner.

For computational purposes, considerable simplification results if ξ and η take on integer values. Thus, indices j and k are defined as

$$j = \xi + 1 \quad \text{for} \quad 1 \le j \le j_{max} \tag{4a}$$

$$k = \eta + 1 \quad \text{for} \quad 1 \le k \le k_{max} \tag{4b}$$

As a result, ordered pairs j,k of integers correspond to grid nodes.

Let P and Q be defined as

$$P(\xi,\eta) = p(\xi)e^{-a\eta} + r(\xi)e^{-c(\eta_{max}-\eta)} \tag{5a}$$

$$Q(\xi,\eta) = q(\xi)e^{-b\eta} + s(\xi)e^{-d(\eta_{max}-\eta)} \tag{5b}$$

where a, b, c, and d are positive constants. The first of the geometric constraints which we wish to impose on the grid is that the spacing along ξ = constant lines between the body at $\eta = 0$ and the next grid node at $\eta = 1$ is specified by the user. Let this desired spacing, in real space, be denoted by $\Delta s\big|_{k=1}$. Let the differences in x and y over this interval be denoted by Δx and Δy, respectively. Thus, we have the requirement that x and y satisfy the equation

$$\Delta s\Big|_{k=1} = [(\Delta x)^2 + (\Delta y)^2]^{1/2}_{k=1} \tag{6}$$

In the limit as Δx and Δy approach zero, this approaches the differential relationship

$$ds\Big|_{k=1} = [(dx)^2 + (dy)^2]^{1/2}_{k=1} \tag{7}$$

Applying the chain rule for partial differentiation we obtain

$$ds\Big|_{k=1} = [(x_\xi d\xi + x_\eta d\eta)^2 + (y_\xi d\xi + y_\eta d\eta)^2]^{1/2}_{k=1} \tag{8}$$

Since ξ is constant along the interval under consideration, the above reduces to

$$ds\Big|_{k=1} = [(x_\eta{}^2 + y_\eta{}^2)^{1/2} d\eta]_{k=1} \tag{9}$$

or equivalently

$$s_\eta\Big|_{k=1} = [x_\eta{}^2 + y_\eta{}^2]^{1/2}_{k=1} \tag{10}$$

Note that while the user specifies Δs, the method uses s_η, which could be thought of as a function having the desired value only in the limit as $\Delta\eta$ approaches zero. Thus, some small amount of decay in the restriction can occur between the body ($\eta = 0$) and the next node ($\eta = 1$). The spacing will be correct in the limit as $\Delta\eta$ approaches zero.

The second geometric requirement we wish to impose is that the angle of the intersection between the body and the ξ = constant line is specified by the user. Let $\theta|_{k=1}$ denote this angle such that $\theta|_{k=1} = 90°$ means that the lines are perpendicular. Thus, from the definition of the dot product, we must have

$$[\nabla\xi \cdot \nabla\eta]_{k=1} = [|\nabla\xi|\ |\nabla\eta| \cos\theta]_{k=1} \tag{11}$$

Expanding, we obtain

$$[\xi_x\eta_x + \xi_y\eta_y]_{k=1} = [(\xi_x^2 + \xi_y^2)^{1/2}(\eta_x^2 + \eta_y^2)^{1/2} \cos\theta]_{k=1} \tag{12}$$

Applying the relations in equations (2) to equation (12) yields

$$[-y_\eta y_\xi - x_\eta x_\xi]_{k=1} = [(y_\eta^2 + x_\eta^2)^{1/2}(y_\xi^2 + x_\xi^2)^{1/2} \cos\theta]_{k=1} \tag{13}$$

Combining equations (10) and (13) to find $x_\eta|_{k=1}$ and $y_\eta|_{k=1}$ is a straightforward but lengthy algebraic exercise resulting in

$$x_\eta\Big|_{k=1} = \left[\frac{s_\eta(-x_\xi \cos\theta - y_\xi \sin\theta)}{(x_\xi^2 + y_\xi^2)^{1/2}}\right]_{k=1} \tag{14a}$$

$$y_\eta\Big|_{k=1} = \left[\frac{s_\eta(-y_\xi \cos\theta + x_\xi \sin\theta)}{(x_\xi^2 + y_\xi^2)^{1/2}}\right]_{k=1} \tag{14b}$$

The third and fourth geometric constraints are equivalent to the first and second, respectively, with the exception that they apply at the outer boundary. A similar development results in the relations

$$x_\eta\Big|_{k=k_{max}} = \left[\frac{s_\eta(-x_\xi \cos\theta - y_\xi \sin\theta)}{(x_\xi^2 + y_\xi^2)^{1/2}}\right]_{k=k_{max}} \tag{14c}$$

$$y_\eta\Big|_{k=k_{max}} = \left[\frac{s_\eta(-y_\xi \cos\theta + x_\xi \sin\theta)}{(x_\xi^2 + y_\xi^2)^{1/2}}\right]_{k=k_{max}} \tag{14d}$$

The desired end result of the preceding manipulations is that the four geometric constraints be embedded in the P and Q terms. In pursuit of this we use equations (14) to compute x_η and y_η. We then assume that the mapping functions do satisfy the transformed Poisson equations at the inner and outer boundaries, then "back-solve" for P and Q. In equations (5) the coefficients of r and s become vanishingly small at the body (η = 0), and thus at the body equations (5) reduce to

$$P(\xi,0) = p(\xi) \tag{15a}$$

$$Q(\xi,0) = q(\xi) \tag{15b}$$

Combining equations (3) and (15) and reducing yields

$$p(\xi) = \left[\frac{y_\eta R_1 - x_\eta R_2}{J}\right]_{k=1} \tag{16a}$$

$$q(\xi) = \left[\frac{-y_\xi R_1 + x_\xi R_2}{J}\right]_{k=1} \tag{16b}$$

where

$$R_1 = \left[\frac{-(\alpha x_{\xi\xi} - 2\beta x_{\xi\eta} + \gamma x_{\eta\eta})}{J^2}\right]_{k=1} \tag{16c}$$

$$R_2 = \left[\frac{-(\alpha y_{\xi\xi} - 2\beta y_{\xi\eta} + y_{\eta\eta})}{J^2}\right]_{k=1} \tag{16d}$$

The same procedure applied at the outer boundary ($\eta = \eta_{max}$) leads to

$$r(\xi) = \left[\frac{y_\eta R_3 - x_\eta R_4}{J}\right]_{k=k_{max}} \tag{16e}$$

$$s(\xi) = \left[\frac{-y_\xi R_3 + x_\xi R_4}{J}\right]_{k=k_{max}} \tag{16f}$$

where

$$R_3 = \left[\frac{-(\alpha x_{\xi\xi} - 2\beta x_{\xi\eta} + \gamma x_{\eta\eta})}{J^2}\right]_{k=k_{max}} \tag{16g}$$

$$R_4 = \left[\frac{-(\alpha y_{\xi\xi} - 2\beta y_{\xi\eta} + \gamma y_{\eta\eta})}{J^2}\right]_{k=k_{max}} \tag{16h}$$

Combining p, q, r, and s, as computed above, with equations (5) yields the desired $P(\xi,\eta)$ and $Q(\xi,\eta)$.

The preceding development would have been unchanged if the four input variables $\theta|_{k=1}$, $\theta|_{k=k_{max}}$, $S_\eta|_{k=1}$, and $S_\eta|_{k=k_{max}}$ representing the four

geometric constraints, and the positive constants a, b, c, and d in equations (5), had been functions of ξ. The computer program described in the following section does include this generality.

Although the four geometric effects are imbedded in p, q, r, and s, their use in equations (5) indicates that their effect decays exponentially as one moves away from the $\eta = 0$ and $\eta = \eta_{max}$ boundaries. Thus, some measurable decay in the geometric effects can occur between the boundary and the interior grid lines. The effects are ensured only in the limit as $\Delta\eta$ approaches zero. The four positive constants a, b, c, and d in equations (5) determine the rate of the exponential decay. Small values (e.g., 0.2) cause slow decay; that is, the geometric effects are propagated far out into the field, but small values also lead to more difficult numerical convergence. Large values (e.g., 0.7) have the opposite effects.

In an iterative procedure, instabilities can result if the p, q, r, and s terms are used exactly as shown in equations (16). Therefore, the changes in these variables are damped by a combination of under-relaxing and limiting the changes in these variables to a small coefficient times their present value. For $p^{(n)}$ being the value of p used on the last, or nth iteration, ω_p satisfying $0 < \omega_p < 1$, and p_{lim} being that "small coefficient" (e.g., 0.5), the value of p to be used in the new, or (n+1)-st iteration is computed from

$$p^{(n+1)} = p^{(n)} + \text{SIGN}\left\{\min\left[\omega_p\left|p - p^{(n)}\right|, p_{lim}\max\left(\left|p^{(n)}\right|, 1\right)\right], p - p^{(n)}\right\} \qquad (17)$$

where the SIGN function returns the magnitude of the first argument with the sign of the second argument. Similar procedures are used for $q^{(n+1)}$, $r^{(n+1)}$, and $s^{(n+1)}$.

It has also been found that instabilities can result from using the above procedure on points on inner or outer boundaries that are sharp corners, such as in an O-type grid at a sharp trailing edge. At such points the computed values for p, q, r, and s must be over-written by values computed as averages of the computed values on either side of the point. Thus, the control of angles and spacing at sharp corners is compromised.

To use the formulations presented above for p, q, r, and s, it is necessary to have values for all of the derivatives appearing in equations (16). Since at the inner ($k = 1$) and outer ($k = k_{max}$) boundaries η is fixed, ξ varies, and x and y are fixed, the derivatives x_ξ, y_ξ, $x_{\xi\xi}$, and $y_{\xi\xi}$ at $k = 1$ and $k = k_{max}$ can be computed by simply differencing known boundary points. These values are fixed for all iteration levels. Given these derivatives, along with input values for θ and S_η at $k = 1$ and $k = k_{max}$, the derivatives x_η and y_η at $k = 1$ and $k = k_{max}$ can be computed from equations (14). These also are fixed at all iteration levels. Derivatives $x_{\xi\eta}$ and $y_{\xi\eta}$ at $k = 1$ and $k = k_{max}$ can be computed by differencing x_η and y_η with respect to ξ. Of all derivatives at $k = 1$ and $k = k_{max}$ appearing in equations (16), only $x_{\eta\eta}$ and $y_{\eta\eta}$ change with iteration level. They are computed by differencing the existing x,y field using

$$x_{\eta\eta}\Big|_{k=1} = \frac{-7x\big|_{k=1} + 8x\big|_{k=2} - x\big|_{k=3}}{2(\Delta\eta)^2} - \frac{3x_\eta\big|_{k=1}}{\Delta\eta} \tag{18a}$$

$$y_{\eta\eta}\Big|_{k=1} = \frac{-7y\big|_{k=1} + 8y\big|_{k=2} - y\big|_{k=3}}{2(\Delta\eta)^2} - \frac{3y_\eta\big|_{k=1}}{\Delta\eta} \tag{18b}$$

$$x_{\eta\eta}\Big|_{k=k_{max}} = \frac{-7x\big|_{k=k_{max}} + 8x\big|_{k=k_{max}-1} - x\big|_{k=k_{max}-2}}{2(\Delta\eta)^2} + \frac{3x_\eta\big|_{k=k_{max}}}{\Delta\eta} \tag{18c}$$

$$y_{\eta\eta}\Big|_{k=k_{max}} = \frac{-7y\big|_{k=k_{max}} + 8y\big|_{k=k_{max}-1} - y\big|_{k=k_{max}-2}}{2(\Delta\eta)^2} + \frac{3y_\eta\big|_{k=k_{max}}}{\Delta\eta} \tag{18d}$$

All derivatives discussed in this paragraph are functions of ξ and thus must be computed for all values of j.

The iterative method for solving the Poisson equations to generate grids, as implemented in the program discussed in the following section, can be summarized as follows:

1. Values for x and y at inner and outer boundaries are computed. Initial conditions for the interior of the grid are computed by linearly interpolating between inner and outer boundary points having the same j values, using a predetermined exponential stretching. Zeros are used for initial conditions for p, q, r, and s. Input values for θ and S_η at $k = 1$ and $k = k_{max}$ are specified. All of the derivatives appearing in equations (16) which are fixed for all iteration levels are computed.

2. Given the initial conditions or the results of the previous iteration, $x_{\eta\eta}$ and $y_{\eta\eta}$ at $k = 1$ and $k = k_{max}$ are computed using equations (18).

3. Values for p, q, r, and s are computed using the procedure presented in the discussions of equations (16) and (17).

4. $P(\xi,\eta)$ and $Q(\xi,\eta)$ are computed at all grid points from equations (5).

5. One step of a successive line over-relaxation (SLOR) solution procedure is performed to find new values for x and y. The lines run in the ξ direction. Periodic or Dirichlet boundary conditions at $\xi = 0$ and $\xi = \xi_{max}$ are used depending on whether an O-type or C-type grid, respectively, is being made. A difference scheme for the transformed Poisson equations (3) is chosen which seeks to maximize diagonal dominance and thereby numerical stability.

Solution steps (2) through (5) are iterated to convergence.

The program discussed in the following section employs the above method in addition to a technique known as coarse-fine sequencing (CFS) which greatly accelerates numerical convergence. In the CFS technique the equations are iterated to convergence twice, with the first solution effected on a coarse grid consisting of every third or fourth point in the ξ direction and every third point in the η direction. Thus, the coarse solution uses roughly one-tenth of all grid points. The computer coding of this method is not difficult if approached from the standpoint of simply relaxing the restriction that $\Delta\xi = \Delta\eta = 1$. Once the coarse solution is finished, a cubic spline interpolation routine is used to fill in the rest of the points. The resulting x,y field is then used as initial conditions for the second, or fine, solution, which is a standard solution procedure using all of the points. A fairly tight convergence criterion is used on the coarse solution, for example, that the maximum correction should be reduced by four orders of magnitude. A much less restrictive condition is placed on the fine solution, for example, that the maximum correction should be reduced another one order of magnitude.

CFS imposes restrictions on the values of j_{max} and k_{max}, for example, that k_{max} must be of the form $3m + 1$ where m is an integer. But these restrictions are usually found to be acceptable in light of the fact that CFS offers a speedup by a factor of roughly 15 over a standard or "fine only" SLOR procedure. The FORTRAN program described in following sections can compute grids for simple cases in as little as 0.65 sec per thousand grid points (e.g., a 100 × 49 grid in 3.2 sec) on a CDC 7600 computer, including "set-up" overhead.

THE PROGRAM GRAPE

General Discussion

The computer program described below has been given the name GRAPE, an acronym derived from GRids about Airfoils using Poisson's Equation. It consists of a main program and 14 subroutines, and is written in FORTRAN IV. The program generates curvilinear finite-difference grids of the O-type or C-type about airfoils or about any other user-specified shape. The program has stored internally three airfoil shapes: (1) an NACA OOXX, with either the open trailing edge as defined, or modified to have a closed trailing edge (ref. 3); (2) an NACA 64A410 (ref. 3); and (3) a Garabedian-Korn 75-06-12 (ref. 4). Other airfoil shapes can be read in. The distribution of points on the airfoil can be specified in several ways, and the number of points on the airfoil can be changed by interpolation. The cell size and cell skewness at inner and outer boundaries is controlled and can be specified in several ways. Three outer boundary shapes are available for O-type grids: a circle, a rectangle with rounded corners, and a cascade shape. Two outer boundary shapes are available for C-type grids: a rectangle with rounded corners on the left side, and a cascade shape. In addition, for either O-type or C-type grids the outer boundary shape can be specified by the user. The distribution of points on the outer boundary can be specified in several ways. The program has some built-in capabilities for computer-graphic display of the resulting grid. An

exhaustive treatment of the program's capabilities is found in the detailed discussion of control parameters given in appendix A.

In the interest of code maintenance and adaptability the program is written in a modular form, with each module performing a particular task. The modules are as follows:

1. The Input Module. The input module contains the DATA statements which define the default case. The default case is the first entry in the set of sample cases following in appendix B. The input module also reads the input data cards. No calculations are done in this module. This module consists of subroutine INPUT.

2. The Input-Checking Module. It is recognized that it is not possible to screen out all invalid, meaningless, or contradictory combinations of input data, but a serious attempt to do so is made. A check is made to see if given data are within limits, for those parameters for which limits can be established. A check on consistency between data is attempted. A check for smoothness and monotonicity, as appropriate, is made in those input data that are arrays. This module also prints the input data. This module consists of subroutines INCHK and CKSMTH.

3. The Airfoil Boundary Module. This module fixes the x,y points on the inner (airfoil) boundary. It finds the inner boundary shape and distributes points thereon. This module consists of subroutines INNER, CSPLIN, and TRIB.

4. The Outer Boundary Module. This module fixes the x,y points on the outer boundary. It consists of subroutine OUTER.

5. The Solution Module. This module does the coarse-fine sequencing solution of the equations. It consists of subroutines SOLVE, IC, INTERP, RELAX, TRIP, and shares subroutines CSPLIN and TRIB with module No. 3.

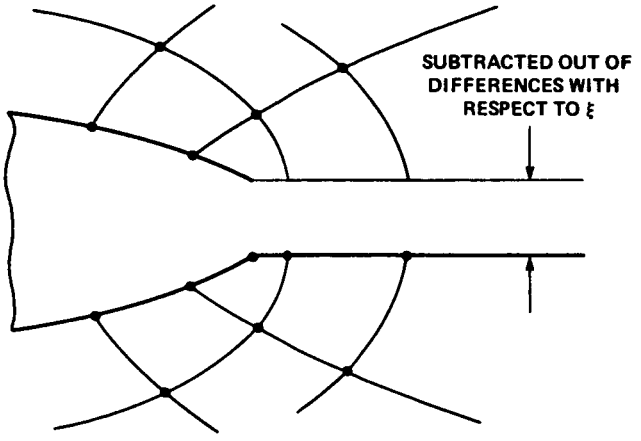
6. The Output Module. This module prints the final solution, writes the final solution for placing on a mass-storage device, and plots the grid. It consists of subroutines OUTPUT and PLAWT.

One benefit of this modular construction is adaptability. If, for example, one has a subroutine that computes part of what would be the input data (such as the shape of a boundary), it would be tedious to compute that data, print it out, punch it into data cards, then read it in and run this grid generation code. Alternatively, one could modify that subroutine to do its computations and store the results by selectively overwriting the appropriate arrays in GRAPE's common blocks. One could then modify GRAPE to call that subroutine between the input and input-checking modules. Thus, one would start with the default case, which could be modified by reading data cards. Those data would be further modified by the new subroutine, and all the data would be checked and printed.

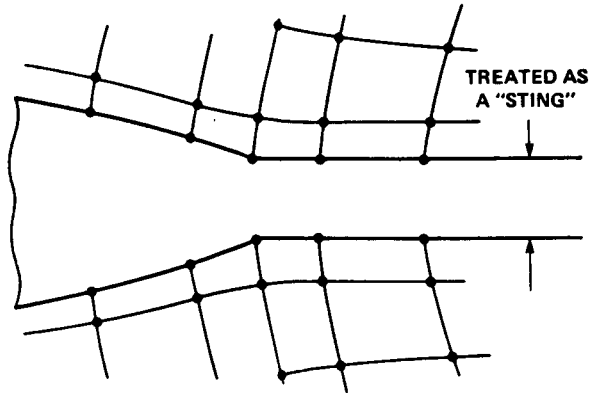
Computer-graphic display of results is always desirable, but especially when those results are a grid. However, it also seems to be true that

computer-graphic display code is among the least transportable of all code. This is due to different computers at different installations having different word-lengths, different graphical software, different graphical output devices, and different implementations. GRAPE includes code to plot the grid using ISSCO DISSPLA software (a proprietary and copyrighted product of Integrated Software Systems Corp., P.O. Box 9906, San Diego, Calif. 92109) and a C.O.M. device for making microfiche. If the user has exactly the same hardware and software, the display code may work as is. Otherwise the user should modify, ignore, remove, or replace that code as he sees fit. That code consists of subroutine PLAWT, and the calls to PLAWT and DONEPL in subroutine OUTPUT.

FORTRAN implicit type specification is used for all input variables and generally throughout the code. The only exceptions are a few LOGICAL variables which are used internally and are not part of the input. Thus, the names of all integer variables begin with the letters I, J, K, L, M, or N, and the names of all floating-point variables begin with the letters A through H and O through Z.



(a) O-type grids.



(b) C-type grids.

Figure 3.- Open trailing edges.

One difficult question in grid generation about airfoils is what to do about airfoils with open, or blunt trailing edges (e.g., the NACA OOXX); that is, airfoils for which the upper surface and lower surfaces do not quite meet at the trailing edge. This question also arises when the flow analysis code uses a boundary-layer displacement thickness, which produces an artificial openness at the trailing edge. GRAPE handles this question in the case of an O-type grid (see fig. 3(a)) by assuming that the trailing-edge openness is the beginning of a cut through real space proceeding rearward. The code differences across it as if it did not exist; that is, the thickness is subtracted out of the appropriate differences with respect to ξ. In the case of an open trailing-edge airfoil in a C-grid, that thickness is treated as a "sting" (see fig. 3(b)).

In many grid generation codes it is assumed that the body is located in the interval 0 to 1 on the x-axis, and that all distances are thus "normalized by chord." Although this is possible

with GRAPE, and in fact is the default, the program is not limited to this approach. The grid can be thought of as being imposed on a Cartesian x,y field, with the airfoil shifted (in the x-direction) and scaled any amount. The airfoil coordinates may be specified using arbitrary units provided only that the leading and trailing edges are on the x-axis; the leading edge need not be at $x = 0$ and the trailing edge need not be at $x = 1$. The outer boundary and all other input and output would then be in the same arbitrary units.

There is one artifice used in the input variables which is potentially confusing but, if mastered, is both handy and versatile. In the preceding section there are several variables introduced that are functions of ξ, for example, $\theta|_{k=1}$. This is specified by either the scalar input variable THETAI or the input array THETA(J). The default value is set into the array, such that THETA(J) = 90.0 for all J (thus indicating that we want 90° angles, or orthogonality, everywhere on the body). If the user wishes to override this default with a set of angles, varying with J, those angles should be input to the array THETA(J). The scalar variable THETAI is given the initial value zero, a physically unreasonable number, which indicates that values for $\theta|_{k=1}$ are to be taken from the array THETA(J). If, however, the user wishes to override the default (90° everywhere) with an angle that does not vary with j (e.g., 85° everywhere) then that angle should be input to the scalar THETAI. Thus, the scalar THETAI being equal to zero indicates that the values in the array THETA(J) (either the 90° default or as over-written by the user) are to be used for $\theta|_{k=1}$. The scalar THETAI being not equal to zero causes that value to be used for $\theta|_{k=1}$ for all j. This method of input is also used for $\theta|_{k=k_{max}}$ (given by THOBI and THETOB(J)); for $S_\eta|_{k=k_{max}}$ (given by DSOBI AND DSOB(J)); and for a, b, c, and d, as in equations (5) (given, respectively, by either AAAI, BBBI, CCCI, and DDDI or AAA(J), BBB(J), CCC(J), and DDD(J)). This method of input is also used for $S_\eta|_{k=1}$ if the input variable NDS equals 2 (see the discussion of NDS following). In this case, $S_\eta|_{k=1}$ is given by the scalar DSI for the array DS(J).

It should be pointed out that the ability to control the angles and spacing at the inner and outer boundaries need not be used in every case. It is possible to disengage the mesh control at the inner or outer boundaries. This produces a grid that locally resembles a Laplacian grid, and speeds numerical convergence. In fact, it is recommended that the effects not be used at the outer boundary in cases wherein the outer boundary conditions are those of free stream. The mesh control is disengaged at the outer boundary by setting input parameters OMEGR and OMEGS to zero, and at the inner boundary by setting OMEGP and OMEGQ to zero.

Most input arrays are checked for smoothness. This is done by successively examining each point by fitting a parabola through the three nearest surrounding points, evaluating the parabola to predict a value for the given point, and examining the difference between the given and predicted values for the point. If that difference is greater than the product of some tolerance times the predicted value, a warning message is printed. For different arrays the tolerances vary from 0.05 to 0.5. It is expected that this procedure will prove helpful, since a common error in punching data cards (e.g., those

describing airfoil ordinates) is to drop a leading zero for one element of an array. Thus, that element will be in error by an order of magnitude; this procedure will find such an error. A drawback to this method is that at the end points of an array the parabola is used for extrapolation rather than interpolation, and thus some elements that are correct are sometimes flagged as erroneous. Such warnings should be ignored.

The control parameters are specified using the FORTRAN feature NAMELIST. With NAMELIST, default values for all variables are initialized by DATA statements in the code — then only those input variables for which a value is required other than the default value need appear on input data cards. Thus, any data case is thought of and presented as an excursion from the default case. It is recognized that the use of NAMELIST will cause some degradation in the "transportability" of the code, since although NAMELIST is a fairly standard FORTRAN feature, it is not supported on some computers. However, this is believed to be justified because of the ease with which the user can "get the code running" when this approach is used. One need simply include three essentially blank data cards, and the default case will result. The alternative, with standard formatted READ statements, is to have to estimate reasonable values for each of the control parameters (74 in number, some of which are arrays), and correctly punch all of them on data cards before the program first runs.

Some computers, for example, those made by IBM, do not allow DATA statements for labeled COMMON variables to appear in subroutines. In such cases the DATA statements defining the default case, all of which are in subroutine INPUT, should be moved to a BLOCK DATA subroutine.

The 74 control parameters are divided into three NAMELIST groups: $GRID1, $GRID2, and $GRID3. $GRID1 includes the scalar variables (as opposed to arrays) that would be changed most often; $GRID2 includes the scalar variables that would be changed least often; and $GRID3 consists of input parameters that are arrays. Two exceptions to the above divisions are control parameters NORDA and MAXITA, arrays having two elements each, and found in $GRID1.

NAMELIST is used in the following way. Suppose one wants to run a case that is identical to the default case except that JMAX is to be set to 120 instead of the default value of 100. The following three data cards would be used (beginning in column 2):

```
$GRID1 JMAX=120$

$GRID2 $

$GRID3 $
```

For further details on the use of NAMELIST, the FORTRAN manual for the user's computer installation should be consulted.

Appendix A gives a complete list of input variables for all three NAMELIST groups and a complete description of their functions. Appendix B gives five

sample cases, and appendix C provides flow charts for the main program and the principal subroutines.

Reading the Output and Diagnosing Errors

Regardless of the value chosen for control parameter JPRT, error messages will appear for any errors found in the input. These messages should be self-explanatory.

For JPRT $\geq$ 0, all of the following printing will be done. The input variables will be printed, and it is suggested that any arrays read (using $GRID3) be checked, including a check to determine that the correct number of elements has been read. Also, if arrays AIRFX, AIRFY, XOB, and YOB, etc., are read in, they should be checked to determine that they are ordered correctly (clockwise from the rear).

For most cases, the inner boundary, that is, x and y for all j, and for k = 1, will be printed. In addition, if DS(J) is computed (NDS = 1), it is printed. TEOPEN is then printed.

For most cases, x and y on the outer boundary are printed along with the values of ARCLENGTH or PHI corresponding to each point. These indicate what distribution of points was used (by equal arc-length increments or by some angular distribution). In addition, for NOBSHP > 1 the variable KEY is printed, indicating to what region (straight line or circular arc) each point belongs (see fig. 5).

A series of numbers exponentially increasing from 0 to 1 is then printed indicating what distribution of points was used along ξ = constant lines for the initial conditions.

Then follow convergence histories for the coarse or fine solutions, along with brief examinations of the inner and outer boundaries after each solution. In the convergence histories there is one line of print for each iteration. Each line consists of several numbers arranged in columns. The numbers are:

1. ITER, the iteration count

2. CSUM, the sum of the absolute values of the corrections on x and y for all j and k

3. CMAX, the absolute value of the largest correction on x and y, followed by the values of j and k at its location

4. P1CM, the absolute value of the largest correction on p

5. P1M, the absolute value of the largest value of p, along with the value of j at its location

6. Q1CM, Q1M, and j, giving similar information for q

7. R1CM, R1M, and j, giving similar information for r

8. S1CM, S1M, and j, giving similar information for s

The solution is said to be converged when CMAX is reduced by the number of orders of magnitude indicated by control parameter NORDA. During the solution process P1M will reach a steady-state value. Thus, P1CM, in a sense the correction on P1M, will become several (e.g., four) orders of magnitude less than P1M. Similar behavior is seen for Q1M and Q1CM, R1M and R1CM, and S1M and S1CM.

GRAPE has been extensively tested, including a series of over 50 specific test cases, and in every case convergence was achieved. However, failure to converge will remain a possibility. Some suggestions can be offered in the event that a solution does fail to converge. First, the convergence history should be examined to determine where within the grid the problem lies. If P1CM fails to be several orders of magnitude less than P1M, or Q1CM fails to be several orders of magnitude less than Q1M, or both, then control of cell size and cell skewness at the inner boundary has not been achieved, and it is at the inner boundary that the problem most likely is to be found. If R1CM is not several orders of magnitude less than R1M or S1CM is not several orders of magnitude less than S1M, or both, then control of cell size and cell skewness at the outer boundary has not been achieved, and the problem is most likely to be found there. By noting the value of J corresponding to the suspect forcing function, one can determine at what part of the suspect boundary to look. All input data referring to the problem area should be examined, and the location of the probable problem area should be kept in mind as further suggestions are pursued.

The most likely cause of a failure to converge is that input variables defining k_{max}, $S_\eta\big|_{k=1}$, $S_\eta\big|_{k=k_{max}}$, a, b, c, d, and the size of the outer boundary have combined to form a physically impossible situation. In other words, (1) S_η at $k = 1$ and $k = k_{max}$ define the step size at the ends of ξ = constant lines, (2) that step size is to be increased exponentially toward the midfield at a rate determined by a, b, c, and d, but (3) k_{max} points are not sufficient to connect the given boundary points using those step sizes. In this case it is suggested that the user do one or more of the following:

1. Increase k_{max}

2. Increase S_η at $k = 1$ or at $k = k_{max}$ or both

3. Increase a, b, c, and d

4. Decrease the size of the outer boundary

For cases with NDS = 1, indicating that $S_\eta\big|_{k=1}$ is to be determined as a coefficient (DSI) times the body-surface arc-length, one is advised to check the values computed for DS and printed along with x and y for the inner boundary. It is especially advised that the value computed for DS at the leading edge be checked, since the body-surface arc-length is usually at a

minimum there, causing DS to be at a minimum there. One can thus be subtly drawn into the problem described in the preceding paragraph. For this case, suggestions are to

1. Increase DSI

2. Use NIBDST = 2 and a large value for BINN (e.g., 1.5)

3. Switch to NDS = 2

Difficulties can result from an outer boundary that is much taller than it is wide or wider than it is tall. For such cases, suggestions are:

1. Carefully adjust $S_\eta\big|_{k=k_{max}}$, c, and d such that they vary for varying j

2. Change the shape of the outer boundary

It is rare but possible for numerical instabilities to be introduced by excessive values for relaxation parameters ω, ω_p, ω_q, ω_r, ω_s, and limitation factors p_{lim}, q_{lim}, r_{lim}, and s_{lim}. When this is the case, the solution process "blows up," meaning that CMAX and several others of the numbers printed in convergence history become large without bound. In this case the user need do nothing because GRAPE will automatically reduce the parameters and factors, reset the x,y field to the initial conditions, and restart the solution process. If this does not work, then the problem lies elsewhere.

If the initial examination of the convergence history indicates that the problem lies on the inner boundary at a sharp leading edge, or at a sharp trailing edge in an O-type grid, then it is suggested that airfoil boundary points be reclustered to move more points toward that sharp edge.

Following the convergence history, a brief examination of the inner boundary is printed. For every j in the solution, there is first printed the angle, in degrees, between the boundary and the ξ = constant line. This is measured the same way as, and should be compared to, $\theta\big|_{k=1}$. The angle between the ξ = constant line and the x-axis is then printed. This is measured in classical polar coordinate fashion, counterclockwise from the positive x-axis. Next, the distance between the node at the boundary and the adjacent node in the field for each j, labeled "DISTANCE TO BOUNDARY," is printed. This should be compared to $S_\eta\big|_{k=1}$ as either read or computed. Following this, the x,y coordinates of the inner boundary points are printed. Next are the x,y coordinates of the adjacent node in the field, for each j. Following are the computed values for p and q. Similar data at the outer boundary are then printed. Printed last is a list of XMIN, XMAX, YMIN, and YMAX for each plot, if any.

APPENDIX A

INPUT VARIABLES

Alphabetical List of Input Variables

Following is an alphabetized list of input variables together with the number of the NAMELIST group in which each appears (e.g., 1 indicates $GRID1).

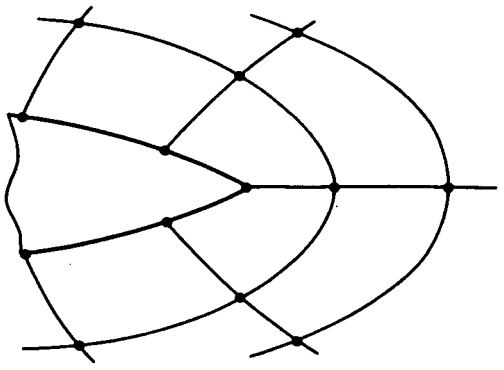
Variable name	Group number	Variable name	Group number	Variable name	Group number	Variable name	Group number
AAA	3	DSOBI	2	NTETYP	1	THETOB	3
AAAI	2	JAIRF	1	OBANGS	3	THOBI	2
AIRFX	3	JCAMBR	2	OMEGA	2	TR	1
AIRFY	3	JDIST	2	OMEGP	2	XLE	1
ALAMF	1	JMAX	1	OMEGQ	2	XLEFT	1
ALAMR	1	JPRT	1	OMEGR	2	XMAX	3
BBB	3	JTEBOT	1	OMEGS	2	XMIN	3
BBBI	2	JTETOP	1	PLIM	2	XOB	3
BINN	2	KMAX	1	QLIM	2	XOBCNT	2
CAMBRX	3	MAXITA	1	RADOB	1	XRIGHT	1
CAMBRY	3	NAIRF	1	RCORN	1	XTE	1
CCC	3	NDS	1	RLIM	2	XTFRAC	2
CCCI	2	NIBDST	1	ROTANG	2	YBOTOM	1
DDD	3	NLETYP	2	ROTCTR	2	YMAX	3
DDDI	2	NOBDST	1	SLIM	2	YMIN	3
DIST	3	NOBSHP	1	TEOPEN	2	YOB	3
DS	3	NORDA	1	THETA	3	YTOP	1
DSI	1	NOUT	1	THETAI	2	WAKEP	2
DSOB	3	NPLT	1				

Variables in NAMELIST $GRID1

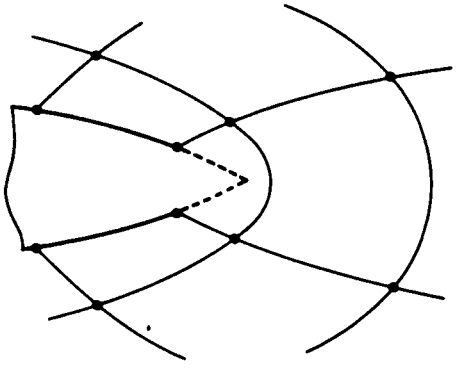
Following is a detailed description of each of the input variables in NAMELIST $GRID1. This list should serve as both a reference for those who are using the program, and as an introduction to all of the program's capabilities for the first-time reader.

Variable name	Description
JMAX	The maximum value of the subscript j, that is, the number of ξ values, that is, the number of points around the airfoil (in the case of an O-type grid) or the number of points along the "c" (in a C-type grid). If coarse-fine sequencing is to be used, there are restrictions on this value, and those restrictions are dependent on the value of NTETYP (see below). If NTETYP = 1, JMAX must be a multiple of 4. If NTETYP = 2, JMAX must be a multiple of 3. If

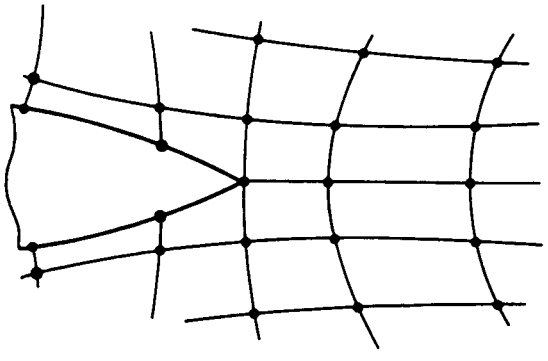
Variable name	Description
	NTETYP = 3, JMAX must be of the form $3n + 1$ where n is some integer. If these restrictions are not observed, the code will run, but coarse-fine sequencing will not be used. Range of acceptable values: 4 to 140. Default value: 100.
KMAX	The maximum value of the subscript k, that is, the number of η values, that is, the number of points from the body to outer boundary, inclusive. If coarse-fine sequencing is to be used, KMAX must be of the form $3m + 1$ where m is some integer. Range of acceptable values: 4 to 70. Default value: 49.
NTETYP	This variable determines the location of the point or points at the trailing edge, and it determines whether the grid is of the O-type or C-type. If NTETYP = 1 an O-type grid results with the point $j = 1$ at the trailing edge (see fig. 4(a)). If NTETYP = 2, an O-type grid is made with the trailing edge of the airfoil located midway between the two grid points at $j = 1$ and $j =$ JMAX. If NTETYP = 3, a C-type grid results with the lower surface trailing-edge point at $j =$ JTEBOT and the upper-surface trailing-edge point at $j =$ JTETOP (see below). Acceptable values for NTETYP: 1, 2, and 3. Default value: 1.
NAIRF	Determines which airfoil shape is to be used. If NAIRF = 1 an NACA OOXX airfoil shape will be calculated. The thickness ratio is given by TR (see below). Note that this airfoil shape has an open trailing edge. If NAIRF = 2, the airfoil shape used will be an NACA OOXX modified to have a closed trailing-edge by extrapolating the analytic defining function to find a zero and re-normalizing. The thickness ratio is given by TR (see below). If NAIRF = 3, a Garabedian-Korn airfoil will be used. If NAIRF = 4, an NACA 64A410 will be used. If NAIRF = 5 the airfoil or body shape will be supplied by the user. See JAIRF below and AIRFX and AIRFY in $GRID3. NAIRF must equal 5 if NIBDST (see below) equals 3 or 5. Acceptable values: 1, 2, 3, 4, and 5. Default value: 2.
JAIRF	The number of data points used to specify a user-supplied airfoil or body-shape. See AIRFX and AIRFY in $GRID3. JAIRF is ignored unless NAIRF equals 5, in which case JAIRF must be ≥ 4. Acceptable values: 0 or 4 to 241 inclusive. Default value: 0.
NIBDST	Specifies how points will be distributed on the airfoil. If NIBDST = 1, points will be distributed using a pre-stored circle-plane-mapping distribution taken from a conformal mapping grid about an NACA 0012 airfoil. This distribution is symmetric front to back about the midchord point, symmetric top to bottom, and produces a fairly high degree of clustering toward the leading and trailing edges. If NIBDST = 2, points will be clustered using an algorithm involving a parameter, BINN (in $GRID2), which can be adjusted to move points toward the leading and trailing edges or



(a) O-type grid with point at trailing edge (NTETYP = 1).



(b) O-type grid with trailing edge midway between two points (NTETYP = 2).

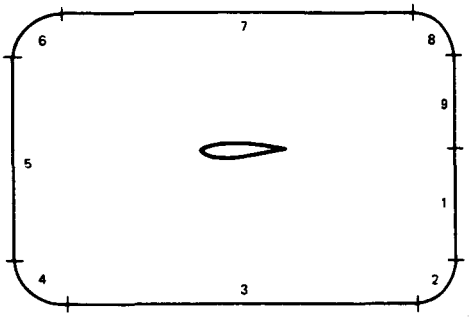


(c) C-type grid (NTETYP = 3).

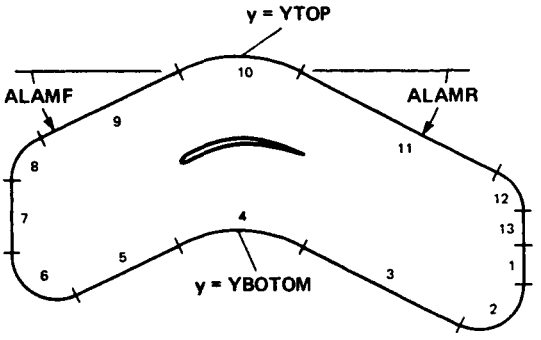
Figure 4.- Trailing-edge treatments.

Variable name	Description
	toward equal spacing with respect to arc-length. If NIBDST = 3, it is assumed that the user is supplying an airfoil (or other body) on data cards. In this case the given data points will not be used exactly as read; rather, they will be interpolated using a cubic spline method so that the number of points on the airfoil can be varied. However, in this case the distribution function will be taken from the given data points. For example, if the given airfoil data points have a great deal of clustering at the leading edge, the resulting airfoil will likewise have many points at the leading edge. In this case NAIRF must equal 5 and JAIRF must be ≥4 (see above). See also AIRFX and AIRFY in $GRID3. If NIBDST = 4, it is assumed that the user will supply a distribution function (a sequence of numbers normalized to go from 0 to 1) on data cards (see array DIST in $GRID3). If NIBDST = 5, it is assumed that the user is supplying an airfoil (or other body) on data cards and that that body is to be used exactly as read, with no interpolation. In this case NAIRF must equal 5 and JAIRF must be ≥4 (see above). See also AIRFX and AIRFY in $GRID3. Acceptable values: 1, 2, 3, 4, and 5. Default value: 1.
NDS	This parameter determines how the grid spacing normal to the body along ξ = constant lines, over the interval k = 1 to k = 2, denoted as $S_\eta\big\|_{k=1}$ in the Theoretical Development section, is to be determined. If NDS = 1, the spacing will be taken as a coefficient times the spacing along the body surface in the ξ direction. Thus, if this option is used and the coefficient is 1, grid cells that are roughly square or equilateral parallelograms will result at the body surface. If NDS = 2, the spacing will be either a scalar constant or any array of constants (see DSI and DS(J)). Acceptable values: 1 and 2. Default value: 2.
DSI	The meaning of this parameter is dependent on the value chosen for NDS. If NDS = 1, then DSI is the "coefficient" referred to in the discussion of NDS above. A value of 1.0 is suggested. If NDS = 2, and it is desired that a constant value be used for $S_\eta\big\|_{k=1}$ for every j in $1 \le j \le j_{max}$, then that value should be entered in DSI. Alternatively, as indicated by NDS = 2 and DSI = 0, any set of values may be entered into array DS(J) in $GRID3. Distances are measured in x,y "units." It is recommended that values for $S_\eta\big\|_{k=1}$ be less than one fourth of that which would result if the grid points were equally spaced between inner and outer boundaries. Acceptable values: all non-negative real numbers. Default value: 0. Note that in the default case the constant value 0.01 is stored in every element of the array DS(J).
JTEBOT	The value of the index j at the lower-surface trailing-edge point. In the case of an O-type grid the value given for JTEBOT will be

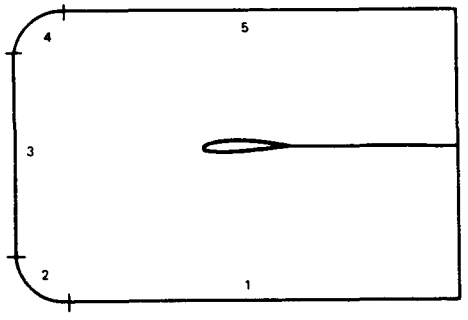
Variable name	Description
	ignored and overwritten with 1. For C-type grids this parameter, along with JTETOP, determines the number of points in the ξ direction in the wake region, the region behind the airfoil. Acceptable values: positive integers less than JTETOP. Default value: 15.
JTETOP	The value of the index j at the upper-surface trailing-edge point. In the case of an O-type grid, the value given for JTETOP will be ignored and overwritten with JMAX. For C-type grids this parameter is used, and JTETOP and JTEBOT must satisfy the relation JTEBOT - 1 = JMAX - JTETOP to ensure that there will be the same number of points in the ξ direction above and below the wake-line. Acceptable values: integers greater than JTEBOT and less than 140. Default value: 86.
TR	The thickness ratio of the NACA OOXX airfoil (e.g., TR = 0.12 yields an NACA 0012). Ignored if NAIRF > 2. Acceptable values: real numbers between 0.0 and 1.0. Default value: 0.12.
XLE	The value of x at the leading edge of the airfoil. If NIBDST < 5, the body will be shifted to place the leading edge at x = XLE. If NIBDST = 5, XLE will be overwritten with the value appropriate to the given body shape. Acceptable values: all real numbers. Default value: 0.0.
XTE	The value of x at the trailing edge of the airfoil. If NIBDST < 5, the body will be scaled to place the trailing edge at x = XTE. If NIBDST = 5, XLE will be overwritten with the value appropriate to the given body shape. Acceptable values: all real numbers greater than XLE. Default value: 1.0.
NOBSHP	This parameter determines what shape will be used for the outer boundary. For O-type grids (NTETYP equals 1 or 2) the following three shapes are available. If NOBSHP = 1, a circle will be used. See RADOB (below) and XOBCNT (in $GRID2). If NOBSHP = 2, a rectangle with corners rounded by circular arcs will be used (see fig. 5(a)). If NOBSHP = 3, a cascade shape consisting of straight lines connected by circular arcs will be used (see fig. 5(b)). For C-type grids (NTETYP = 3) two shapes are available. If NOBSHP = 4, a rectangle with corners on the left rounded by circular arcs will be used (see fig. 5(c)). If NOBSHP = 5, a cascade shape consisting of straight lines connected by circular arcs except on the right end where the straight lines intersect will be used (see fig. 5(d)). For NOBSHP equals 2 through 5, see XLEFT, XRIGHT, YBOTOM, YTOP, and RCORN below. For NOBSHP equals 3 or 5, see also ALAMF and ALAMR below. If NOBSHP = 6, the outer boundary will be supplied by the user on data cards. See XOB and YOB in $GRID3. These points will be used exactly as read, with no interpolation. Acceptable values: 1, 2, 3, 4, 5, and 6. Default value: 1.



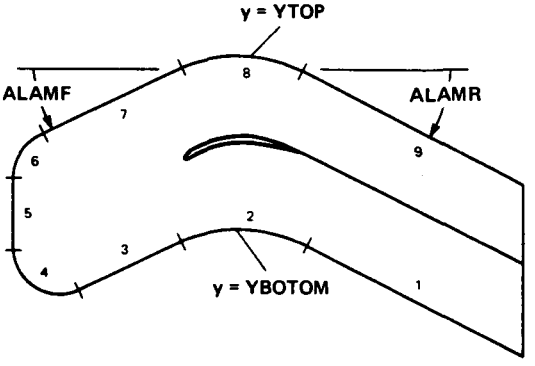
(a) Rectangular outer boundary for O-type grids (NOBSHP = 2).



(b) Cascade outer boundary for O-type grids (NOBSHP = 3).



(c) Rectangular outer boundary for C-type grids (NOBSHP = 4).



(d) Cascade outer boundary for C-type grids (NOBSHP = 5).

Figure 5.- Outer boundary shapes.

NOBDST — This parameter is ignored if NOBSHP = 6. If NOBSHP < 6, it determines how the points are to be distributed on the outer boundary. If NOBDST = 1, the points will be distributed by equal increments of the arc-length on the outer boundary. If NOBDST = 2, the points will be distributed by equal angular increments. For NOBDST = 2 and a circular outer boundary (NOBSHP = 1), the angles are measured about the point x = XOBCNT on the x-axis. For NOBDST = 2 and a rectangular outer boundary (NOBSHP equal 2 or 4), the angles are measured about the origin. If NOBDST = 3, outer boundary points will be distributed in an angular fashion, as in the case of NOBDST = 2, but using a set of angles that the user has read in on data cards (see array OBANGS in $GRID3). In the case of a cascade outer boundary shape (NOBSHP equals 3 or 5) NOBDST must equal 1. Acceptable values: 1, 2, and 3. Default value: 1.

RADOB — Radius of the circular outer boundary. It is ignored if NOBSHP > 1. Acceptable values: all positive real numbers. Default value: 6.0.

Variable name	Description
XLEFT XRIGHT	XLEFT and XRIGHT are the x-coordinates of the left and right ends, respectively, of the outer boundary. They are used for rectangular and cascade outer boundary shapes (NOBSHP equals 2, 3, 4, or 5) and are ignored if NOBSHP equals 1 or 6. Note that they are coordinates, not displacements. Therefore, in most cases XLEFT will have a negative value. XLEFT must be less than XRIGHT. Acceptable values: all real numbers. Default values: XLEFT = -6.0, XRIGHT = 6.0.
YBOTOM YTOP	For rectangular outer boundary shapes (NOBSHP equals 2 or 4), YBOTOM and YTOP are the y-coordinates of the bottom and top of the rectangle, respectively. For cascade outer boundaries (NOBSHP equals 3 or 5), YBOTOM is the y-coordinate of the uppermost point on the arc below the airfoil, and YTOP is the y-coordinate of the uppermost point on the arc above the airfoil. Since these are coordinates, not displacements, YBOTOM will in most cases be negative. YBOTOM and YTOP are ignored if NOBSHP equals 1 or 6. YBOTOM must be less than YTOP. Acceptable values: all real numbers. Default values: YBOTOM = -4.0, YTOP = 4.0.
RCORN	The radius of the circular arcs used to round the corners of the rectangular and cascade shapes. RCORN is ignored if NOBSHP equals 1 or 6. To prevent a physically impossible situation, the inequalities RCORN $\leq$ 1/2 (YTOP-YBOTOM) and RCORN $\leq$ 1/2 (XRIGHT-XLEFT) must be satisfied. Acceptable values: all positive real numbers. Default value: 1.
ALAMF ALAMR	ALAMF and ALAMR are the declination angles, in degrees, of the front and rear of the cascade outer boundaries, respectively (see fig. 5). Either but not both may be zero. Although values up to 90.0 are acceptable, values above approximately 45.0 are not recommended, since the grids can become meaningless and convergence difficulties can result. Acceptable values: real numbers in the range -90.0 to +90.0. Default values: 0.0, and 0.0.
NORDA	NORDA is an array having two elements, being the convergence criteria for the coarse-fine solutions. NORDA(1) and NORDA(2) are the numbers of orders of magnitude by which the maximum correction is to be reduced for the coarse and fine iterative procedures, respectively. Note that these criteria are subject to the limits imposed by MAXITA, below. Acceptable values: all non-negative integers. Default values: 4 and 1.
MAXITA	MAXITA is an array having two elements. MAXITA(1) and MAXITA(2) are the limits on the numbers of iterations allowed in the coarse and fine solutions, respectively. If it is desired that the coarse or fine solutions be entirely skipped, then zero may be entered for the appropriate element in MAXITA. Acceptable values: all non-negative integers. Default values: 200 and 100.

Variable name	Description
JPRT	A control parameter specifying how much printing is to be done. If JPRT < 0, no printing will be done, with the exception of error messages. If JPRT = 0 the input parameters, the inner and outer boundaries, a convergence history, a brief examination of the solution at the boundaries, and any error messages will be printed. For the default case this is 19 pages of print. If JPRT > 0, then all information listed for JPRT = 0 will be printed, in addition to the solution. The solution will be printed for all k in $1 \le k \le k_{max}$ and j as given by the FORTRAN DO loop "DO 26 J=1, JMAX,JPRT." For example, if JPRT = 10 and JMAX = 100 then the solution will be printed for J equals 1, 11, 21, 31, 41, 51, 61, 71, 81, and 91. Thus, JPRT = 1 yields the printing of the solution at all j and k. What is meant by "the solution," that is, which variables will be printed for each of the indicated j and k, is dependent on the value chosen for NOUT (see below). If NOUT < 2, the arrays X and Y will be printed. If NOUT = 2, X, Y and the Jacobians (see eq. (11e)) will be printed. If NOUT = 3, X, Y, the Jacobians and the metric quantities x_ξ, x_η, y_ξ, and y_η will be printed. The roles of JPRT and NOUT are summarized in the table following. Acceptable values: all integers. Default value: 0.
NPLT	The number of plots to be made of the finished grid, assuming that the user retains the ISSCO DISSPLA software, as called in subroutine PLAWT. See arrays XMIN, XMAX, YMIN, and YMAX in $GRID3. Acceptable values: integers in the range 0 to 100 inclusive. Default value: 0.
NOUT	A parameter controlling the output of the grid by unformatted writes on unit 7 for placing on a mass storage device. If NOUT = 0 there will be no such writing. If NOUT = 1, then X and Y arrays will be written. If NOUT = 2, the X and Y arrays and the Jacobians (see eq. (11e)) will be written. If NOUT = 3, the X and Y arrays, the Jacobians, and the metric quantities x_ξ, x_η, y_ξ, and y_η will be written. The unformatted writes on unit 7 are for all j in $1 \le j \le j_{max}$ and all k in $1 \le k \le k_{max}$. The record structure of these data, which one must know before coding the corresponding read statements, is easily ascertained by examining subroutine OUTPUT. Acceptable values: 0, 1, 2, and 3. Default value: 1.

Table for JPRT and NOUT

This table shows what is written on the printer (logical unit no. 6) and as an unformatted write (logical unit no. 7) for all acceptable combinations of input values for JPRT and NOUT. Note that arrays X, Y, the Jacobian, and the metric quantities x_ξ, x_η, y_ξ, and y_η are printed for only selected values of j (see JPRT in $GRID1), but unformatted writes are for all j. Not shown on this chart are error messages, which are printed unconditionally.

Written as unformatted (unit 7) \ Written on printer (unit 6)	JPRT < 0	JPRT = 0	JPRT > 0
NOUT = 0	Nothing / Nothing	Input, boundaries, conv. history / Nothing	Input, boundaries, conv. history X, Y / Nothing
NOUT = 1	Nothing / X, Y	Input, boundaries, conv. history (default) / X, Y	Input, boundaries, conv. history X, Y / X, Y
NOUT = 2	Nothing / X, Y Jacobians	Input, boundaries, conv. history / X, Y Jacobians	Input, boundaries, conv. history X, Y Jacobians / X, Y Jacobians
NOUT = 3	Nothing / X, Y Jacobians, Metric quantities	Input, boundaries, conv. history / X, Y Jacobians, Metric quantities	Input, boundaries, conv. history X, Y, Jacobians Metric quanti-ties / X, Y Jacobians, Metric quantities

Variables in NAMELIST $GRID2

Variable name	Description
NLETYP	Control variable specifying what type of leading edge is used. If NLETYP = 1, it is assumed that the leading edge is blunt (rounded). NLETYP should be set to 2 if the user is supplying his own airfoil (NIBDST = 5) and that airfoil has a sharp leading edge and there is a grid point at the leading edge. NLETYP should be set to 3 if the user is supplying his own airfoil and that airfoil has a sharp leading edge and the leading edge is midway between two grid points. Acceptable values: 1, 2, and 3. Default value: 1.
BINN	The parameter used in clustering points on the airfoil for NIBDST = 2. Reducing BINN causes points to be moved toward the leading and trailing edges. Increasing BINN causes the inner boundary point distribution to approach equal spacing as a function of surface arc-length. BINN is ignored if NIBDST is not equal to 2. Acceptable values: all real numbers greater than 1. Default value: 1.1.
JDIST	The number of points supplied by the user in defining his own distribution function for the points on the body (airfoil). See array DIST in $GRID3. JDIST need not equal JMAX. Ignored if NIBDST is not equal to 4. Acceptable values: 0 or integers in the range 4 to 241, inclusive. Default value: 0.
JCAMBR	If JCAMBR = 0 the airfoil will not be cambered. If JCAMBR > 0 then the airfoil, regardless of the shape or how it was specified, will be cambered by GRAPE, using a camber line defined by JCAMBR points stored in arrays CAMBRX and CAMBRY in $GRID3. Acceptable values: 0 or integers in the range 4 to 140, inclusive. Default value: 0.
XTFRAC	Ignored for O-type grids (NTETYP < 3). For C-type grids (NTETYP = 3) the x-coordinates of points on the inner boundary rearward of the trailing edge will be distributed by an exponential stretching giving increasing step-size with increasing x. The stretching is calculated so that the x-spacing of the first interval rearward of the trailing edge is equal to some constant multiplier times the x-spacing of the last interval on the body. XTFRAC is that constant multiplier. Acceptable values: all positive real numbers. Default value: 1.0.
ROTANG	If ROTANG = 0.0, the airfoil will not be rotated, that is, placed at an angle of attack, by the program. If ROTANG is not equal to zero, then it is taken as the angle, in degrees, that the airfoil is to be rotated. Acceptable values: real numbers between -90.0 and +90.0. Default value: 0.0.

Variable name	Description
ROTCTR	Ignored if ROTANG = 0.0. If ROTANG is not equal to zero, then the airfoil will be rotated about a point on the x-axis at x = ROTCTR. Acceptable values: all real numbers. Default value: 0.0.
XOBCNT	Used only in cases having a circular outer boundary (NOBSHP = 1). The circle will be centered on the x-axis at x = XOBCNT. Must be consistent with values given for XLE, XTE, and RADOB, that is, the outer boundary must not pass inside of the body. Acceptable values: all real numbers. Default value: 0.0.
OMEGA	Relaxation parameter used in SLOR solution processes for x and y. Increasing OMEGA may lead to more rapid convergence, but numerical instability may also result. Decreasing OMEGA has the opposite effects. Acceptable values: real numbers in the range 0.0 to 2.0. Default value: 1.3.
OMEGP OMEGQ	OMEGP and OMEGQ are the relaxation parameters used in solving for p and q as in equation (17). Changes in these parameters produce results similar to those in OMEGA. The effects of controlling angles and spacing at the inner boundary are disengaged by setting OMEGP and OMEGQ to zero. Acceptable values: real numbers in the range 0.0 to 2.0. Default values: 0.3.
OMEGR OMEGS	OMEGR and OMEGS are the relaxation parameters used in solving for r and s as in equation (17). Changes in these parameters produce results similar to those in OMEGA. The effects of controlling angles and spacing at the outer boundary are disengaged by setting OMEGR and OMEGS to zero. Acceptable values: real numbers in the range 0.0 to 2.0. Default values: 0.3.
PLIM QLIM RLIM SLIM	PLIM, QLIM, RLIM, and SLIM are limitation factors used in solving for p, q, r, and s as in equation (17). Changes in these parameters produce results similar to those in OMEGA. Acceptable values: real numbers between 0.0 and 100.0. Default values: 1.0.
DSOBI	If it is desired that a constant value be used for $S_\eta\big\|_{k=k_{max}}$, that is, the distance to be imposed along ξ = constant lines between the outer boundary (at $k = k_{max}$) and the adjacent grid node (at $k = k_{max} - 1$), for every j in $1 \le j \le j_{max}$, then that value should be entered in DSOBI. Alternatively, as indicated by DSOBI = 0.0, any set of values may be entered into array DSOB(J) in \$GRID3. Distances are measured in x,y "units." It is recommended that values for $S_\eta\big\|_{k=k_{max}}$ be less than 4 times that which would result if the grid points were equally spaced between inner and outer boundaries. Acceptable values: all nonnegative real numbers. Default value: 0.0 (in the default case the constant value 0.2 is stored in every element of the array DSOB(J)).

Variable name	Description
THETAI	If it is desired that a constant value be used for $\theta\|_{k=1}$, that is, the angles with which ξ = constant lines intersect the inner boundary, for every j in $1 \le j \le j_{max}$, then that value should be entered in THETAI. Alternatively, as indicated by THETAI = 0, any set of values may be entered into array THETA(J) in $GRID3. These angles are measured in degrees, and are measured about the inner boundary points from the inner boundary clockwise to the ξ = constant lines in the interior, with THETAI = 90.0 indicating orthogonality. Acceptable values: real numbers between 0.0 and 180.0. Default value: 0.0 (in the default case the constant value 90.0 is stored in every element of the array THETA(J)).
THOBI	For cascade cases (NOBSHP equals 3 or 5) the $\theta\|_{k=k_{max}}$, that is, the angles with which ξ = constant lines intersect the outer boundary, are determined internally by GRAPE such that periodicity between cascade elements is required. That is, in these cases, for those parts of the outer boundary which touch on adjacent cascade elements, the $\theta\|_{k=k_{max}}$ are chosen to require that the ξ = constant lines are vertical (parallel to the y-axis) at the outer boundary regardless of values chosen for ALAMF and ALAMR. Thus, for NOBSHP equal to 3 or 5, THOBI is ignored. For NOBSHP not equal to 3 or 5, if it is desired that a constant value be used for $\theta\|_{k=k_{max}}$ for every j in $1 \le j \le j_{max}$, then that value should be entered in THOBI. Alternatively, as indicated by THETAI = 0.0, any set of values may be entered into the array THETOB(J) in $GRID3. These angles are measured in degrees, and are measured about the outer boundary points from the outer boundary clockwise to the ξ = constant lines in the interior, with THOBI = 90.0 indicating orthogonality. Acceptable values: real numbers between 0.0 and 180.0. Default value: 0.0 (in the default case the constant value 90.0 is stored in every element of the array THETOB(J)).
AAAI BBBI	If it is desired that constant values be used for a and b, positive constants appearing in equations (5), for every j in $1 \le j \le j_{max}$, then those values should be entered in AAAI and BBBI. Alternatively, as indicated by AAAI and BBBI equal to zero, any set of values may be entered into the arrays AAA(J) and BBB(J) in $GRID3. Smaller values (e.g., 0.2) for a and b cause the effects of angle control and control of S_η at the inner boundary to be propagated far into the interior of the grid, but convergence difficulties can result. Larger values (e.g., 0.7) have the opposite effect. Acceptable values: all real numbers greater than 0.0. Default value: 0.0 (in the default case the constant value 0.45 is stored in every element of the arrays AAA(J) and BBB(J)).
CCCI DDDI	If it is desired that constant values be used for c and d, positive constants appearing in equation (3), for every j in

Variable name	Description
	$1 \le j \le j_{max}$, then those values should be entered in CCCI and DDDI. Alternatively, as indicated by CCCI and DDDI equal to zero, any set of values may be entered into the arrays CCC(J) and DDD(J) in $GRID3. Smaller values (e.g., 0.2) for c and d cause the effects of angle control and control of S_η at the outer boundary to be propagated far into the interior of the grid, but convergence difficulties can result. Larger values (e.g., 0.7) have the opposite effect. Acceptable values: all real numbers greater than 0.0. Default value: 0.0 (in the default case the constant value 0.45 is stored in every element of the arrays CCC(J) and DDD(J).
TEOPEN	The distance across the open trailing edge of the airfoil. For NIBDST < 5, input values for this parameter are ignored and TEOPEN is computed internally. For NIBDST = 5, the user supplies the body shape, not necessarily an airfoil, and that shape is to be used exactly as read, with no interpolation. In this case, for O-type grids, TEOPEN must be supplied by the user. TEOPEN = 0.0 indicates that the body shape has a closed trailing edge. Acceptable values: all nonnegative real numbers. Default value: 0.0.
WAKEP	Ignored for O-type grids (NTETYP < 3). For C-type grids (NTETYP = 3) this parameter effects the exponential stretching in the x-direction of inner boundary points in the wake region. Decreasing WAKEP causes wake points to be shifted closer to the airfoil, away from the rear boundary, that is, decreasing WAKEP causes the exponential stretching to increase more slowly with increasing x near the airfoil and more rapidly near the rear boundary. Increasing WAKEP has the opposite effect. Extreme values of WAKEP can cause a fatal error in subroutine INNER. Recommended values are in the range 0.5 to 1.5. Acceptable values: all positive real numbers. Default value: 1.0.

Variables in NAMELIST $GRID3

Variable name	Description
DS(J)	Ignored if NDS (in $GRID1) equals 1. For NDS = 2, DS is the array into which a set of values for $S_\eta\big\|_{k=1}$ which vary for different j may be entered. See DSI in $GRID1. Acceptable values: all positive real numbers. Default values: 0.01 for every j. Number of elements that must be specified (if any): JMAX.
DSOB(J)	The array into which a set of values for $S_\eta\big\|_{k=k_{max}}$ which vary for different j may be entered. See DSOBI in $GRID2. Acceptable values: all positive real numbers. Default values: 0.2 for every j. Number of elements that must be specified (if any): JMAX.

Variable name	Description
THETA(J)	The array into which a set of values for $\theta\|_{k=1}$ which vary for different j may be entered. See THETAI in $GRID2. Acceptable values: real numbers between 0.0 and 180.0. Default values: 90.0 for every j. Number of elements that must be specified (if any): JMAX.
THETOB(J)	The array into which a set of values for $\theta\|_{k=k_{max}}$ which vary for different j may be entered. See THOBI in $GRID2. THETOB is ignored for cascade outer boundaries (NOBSHP equals 3 or 5). Acceptable values: real numbers between 0.0 and 180.0. Default values: 90.0 for every j. Number of elements that must be specified (if any): JMAX.
DIST(J)	Ignored if NIBDST is not equal to 4. For NIBDST = 4, DIST is an array of values defining the distribution function for inner (airfoil) boundary points. The inner boundary point distribution could be thought of as a function having as a range the inner boundary arc-length normalized to go from 0 to 1, and as domain ξ for $0 \le \xi \le \xi_{max}$. Successive values of DIST changing by a small amount indicates dense airfoil points for the corresponding values of ξ. The converse is also true. Acceptable values: a monotonically increasing array of numbers with DIST(1) = 0.0 and DIST(JDIST) = 1.0. See JDIST in $GRID2. Default values: 0.0 for every j (in the default case DIST is ignored). Number of elements that must be specified (if any): JDIST.
AIRFX(J) AIRFY(J)	AIRFX and AIRFY are ignored if NAIRF is not equal to 5. For NAIRF = 5, the abscissas of points defining the user-supplied airfoil should be entered in AIRFX, and the ordinates should be entered in AIRFY. Points should be ordered clockwise from the trailing edge. If the given airfoil is to be interpolated (NIBDST = 3), the first and last values of AIRFX that are specified must be equal (thus the entire airfoil is described). Acceptable values: all real numbers. Default values: 0.0 for all j (in the default case AIRFX and AIRFY are ignored and an NACA 0012 airfoil is computed). Number of elements that must be specified (if any): JAIRF.
CAMBRX(J) CAMBRY(J)	Ignored if JCAMBER (in $GRID2) is equal to zero. If JCAMBR is not equal to zero, then coordinates describing a camber line should be stored in CAMBRX and CAMBRY. Abscissas are stored in CAMBRX and ordinates are stored in CAMBRY. Acceptable values for CAMBRX: a monotonically increasing array with CAMBRX(1) = 0.0 and CAMBRX(JCAMBR) = 1.0. Acceptable values for CAMBRY: all real numbers. Default values: 0.0 for every j (in the default case CAMBRX and CAMBRY are ignored). Number of elements that must be specified (if any): JCAMBR.

Variable name	Description
OBANGS(J)	Ignored if NOBDST < 3 (see $GRID1). For NOBDST = 3, OBANGS is an array of angles, measured in degrees, defining the distribution of points on the outer boundary. Caution is recommended in the use of this parameter, since the range of values required is never exactly 0.0 to 360.0, and varies depending on NTETYP. If NTETYP = 1, OBANGS must go from 0.0 to one angular increment less than 360.0. If NTETYP = 2, OBANGS must go from one half of an angular increment to one half of an angular increment less than 360.0. Further complicating the above is the trailing-edge openness, if any. If NTETYP = 3, OBANGS must go from arctangent (YBOTOM/XRIGHT) to 360.0-arctangent(YTOP/XRIGHT). Acceptable values: a monotonically increasing array of numbers, with range as indicated above. Default values: 0.0 for every j (in the default case OBANGS is ignored). Number of elements that must be specified (if any): JMAX.
XOB(J) YOB(J)	Ignored if NOBSHP < 6 (see $GRID1). For NOBSHP = 6, the user supplies an outer boundary by entering the x and y coordinates of points defining that shape in XOB and YOB, respectively. These points will be used exactly as read with no interpolation. The points should be ordered clockwise from the rear. Acceptable values: all real numbers. Default values: 0.0 for all j (in the default case XOB and YOB are ignored). Number of elements that must be specified (if any): JMAX.
XMIN(N) XMAX(N) YMIN(N) YMAX(N)	These four arrays are ignored if the user does not retain the code that calls the ISSCO DISSPLA plotting software, of if NPLT = 0 (see $GRID1). Otherwise plots of the finished grid will be made, and these arrays define the square region of the grid which will fill the picture. If a small square region of the grid is specified it will be enlarged to fill the picture. Thus, a series of pictures of the whole grid or "snapshot enlargements" of any part(s) of the grid or both can be made. For the Nth picture, XMIN(N) is the x-coordinate of the left side of the square, XMAX(N) is the x-coordinate of the right side of the square, YMIN(N) is the y-coordinate of the bottom of the square, and YMAX(N) is the y-coordinate of the top of the square. If a non-square rectangle is indicated, the shorter sides of the rectangle will be lengthened to make a square and avoid a "stretched" plot. Thus, there is a trick that can facilitate use of these four arrays. For example, if XMIN and XMAX are set, as specified above, their values can be used to not only define the size of the square (their difference is the length of a side), but to also indicate the x-coordinate of its center (their average). The y-coordinate of the center can then be entered in both YMIN and YMAX. For example, entering 0.9, 1.1, 0.0, and 0.0 in XMIN, XMAX, YMIN, and YMAX, respectively, produces a plot of a square region 0.2 units on a side, centered on the x-axis at x = 1.0. The roles of x and y can be interchanged

Variable name	Description
	in this trick. Acceptable values: all real numbers such that XMIN(N) $\leq$ XMAX(N) and YMIN(N) $\leq$ YMAX(N), and in at least one of those relationships the inequality obtains, for all N in $1 \leq N \leq$ NPLT. Default values: 0.0 for all N (in the default case no plots are made). Number of elements that must be specified (if any): NPLT.
AAA(J) BBB(J) CCC(J) DDD(J)	Arrays into which sets of values for the positive constants a, b, c, and d in equation (3) may be entered. See AAAI, BBBI, CCCI, and DDDI in $GRID2. Acceptable values: all real numbers greater than 0.0. Default values: 0.45 for every j. Number of elements that must be specified (if any): JMAX.

APPENDIX B

SAMPLE CASES

Sample Case No. 1

Required data cards

```
$GRID1 NPLT=4$
$GRID2  $
$GRID3 XMIN=-6.,-.5,-.1,.0,  XMAX=6.,1.5,.1,1.1$
```

Number of iterations required for convergence
89 coarse
6 fine

7600 C.P.U. time for run
3.206 sec without plots
7.805 sec with plots

Number of points in mesh: 4900

With the exception that plots are made, this is the default case. It is a 100 × 49 point, O-type grid about an NACA 0012 airfoil, modified to have a sharp trailing edge. There is a grid point at the trailing edge. Spacing normal to the airfoil, $S_\eta\big|_{k=1}$, is requested to be 0.01 units.

The airfoil spans the interval 0 to 1 on the x-axis, therefore the chord length equals 1 unit. Distribution of points on the airfoil is taken from a circle plane mapping. The outer boundary is a circle of radius 6 units, centered at the origin. Points are distributed on the outer boundary by equal angular increments. The convergence criteria are that CMAX be reduced by four orders of magnitude on the coarse solution and by one on the fine. The resulting grid is illustrated by the plots made by this run, which appear in figure 6.

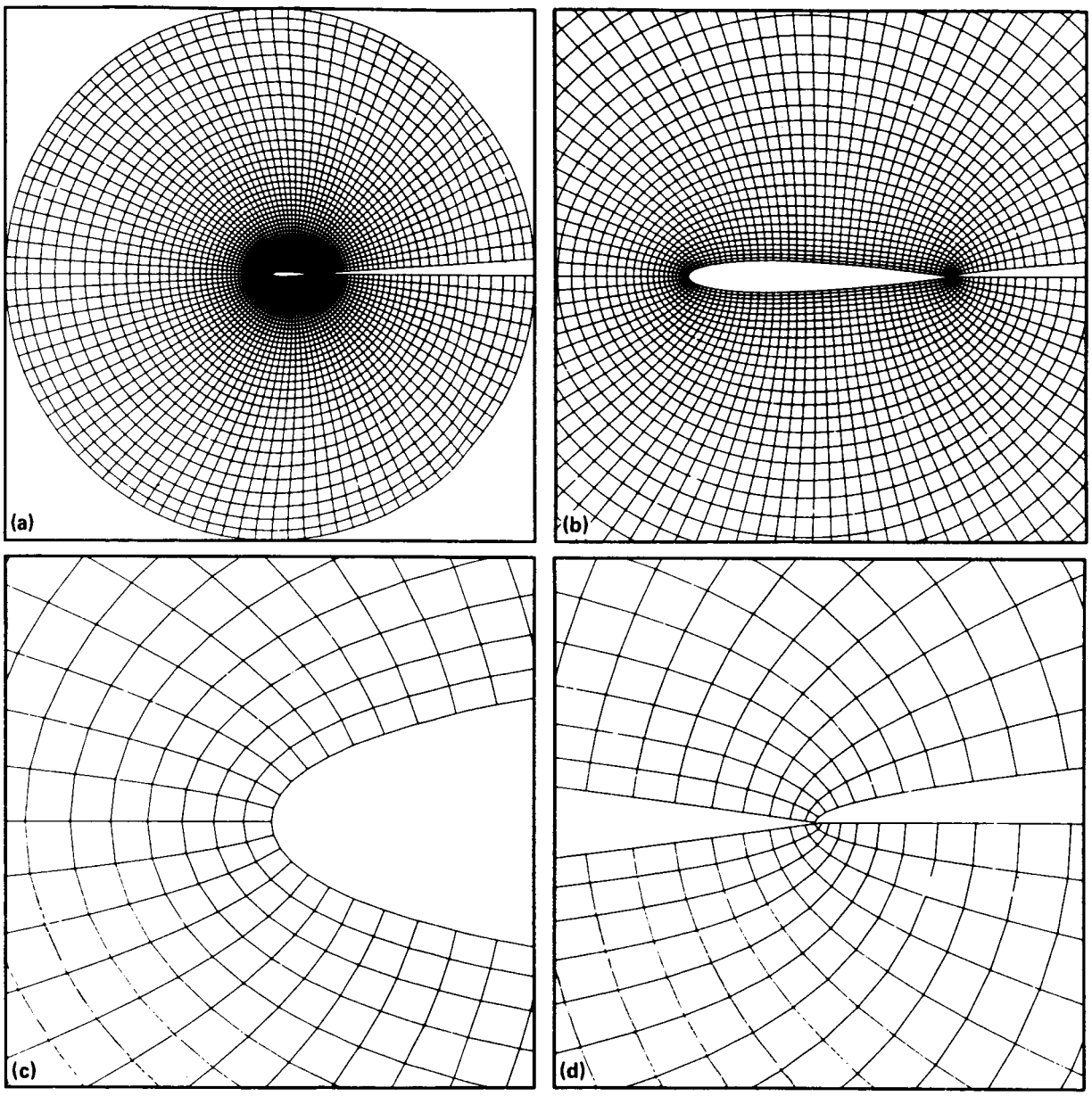


Figure 6.- Sample case no. 1 (default case). (a) Entire grid, showing outer boundary. (b) Region near airfoil. (c) Close-up of leading edge. (d) Close-up of trailing edge.

Sample Case No. 2

Required data cards

```
$GRID1 NDS=1,  DSI=1.,  NPLT=4$
$GRID2  $
$GRID3 XMIN=-6.,-.5,-.1,.9,  XMAX=6.,1.5,.1,1.1$
```

Number of iterations required for convergence
100 coarse
6 fine

7600 C.P.U. time for run
3.397 sec without plots
7.908 sec with plots

Number of points in mesh: 4900

This case is identical to case no. 1 except that it illustrates the NDS = 1 feature. Grid spacing normal to the airfoil is requested to be equal to the spacing along the airfoil surface. This grid is illustrated in figure 7.

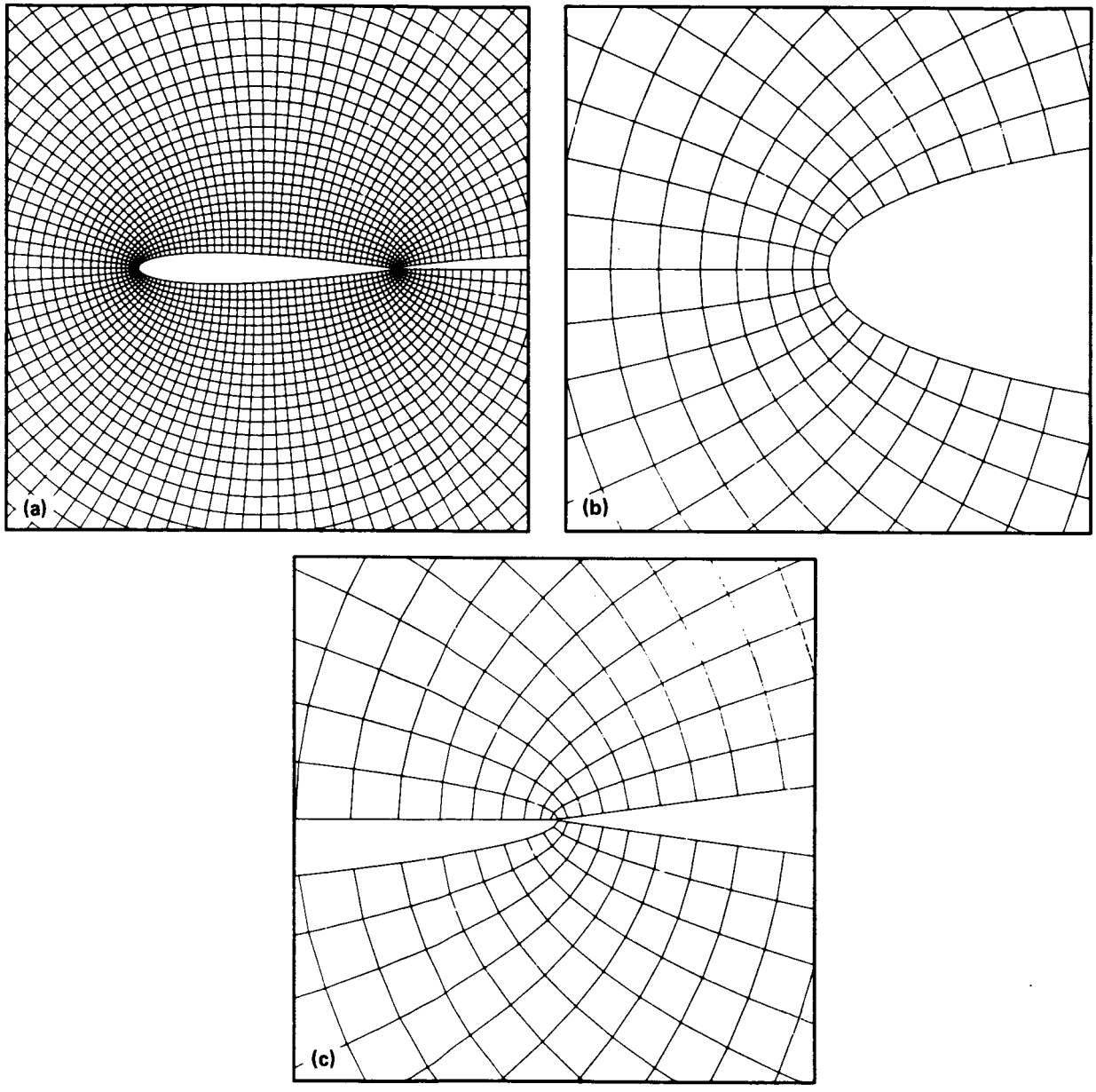


Figure 7.- Sample case no. 2 (identical to default case except NDS = 1). (a) Region near airfoil. (b) Close-up of leading edge. (c) Close-up of trailing edge.

Sample Case No. 3

Required data cards

```
$GRID1 KMAX=70, NTETYP=3, NAIRF=1, DSI=.0005, NOBSHP=4, NIBDST=2,
NPLT=5, NORDA=6,3$
$GRID2 BINN=1.2, OMEGR=0., OMEGS=0., AAAI=.3, BBBI=.3$
$GRID3 XMIN=-6.,-.5,-.05,.9,.99, XMAX=6.,1.5,.05,1.1,1.01$
```

Number of iterations required for convergence
102 coarse
42 fine

7600 C.P.U. time for run
14.507 sec without plots
21.896 sec with plots

Number of points in mesh: 7000

This case is a C-type grid, brought about by setting NTETYP = 3. For C-type grids it is necessary to choose a proper outer boundary shape, in this case a rectangle (NOBSHP = 4). This case illustrates the distribution of airfoil boundary points by formula (NIBDST = 2, BINN = 1.2). Also seen in this case is a Laplacian outer boundary treatment (OMEGR = 0., OMEGS = 0.).

This case also has a small spacing normal to the airfoil surface (DSI = 0.005). This is not exactly a viscous spacing, but it is the smallest that is legible in the accompanying figures. Some of the additional measures that are usually required to produce a viscous grid are seen. It was found (in a previous run) that the angles with which the ξ = constant lines intersected the x-axis, in the wake region, were scattered, that is, not smoothly varying. This is corrected by requiring more convergence (NORDA = 6,3). It was then found that those angles were not close enough to 90°. This was corrected by reducing a and b from their default values of 0.45 (AAAI = 0.3, BBBI = 0.3). The resulting grid is shown in figure 8.

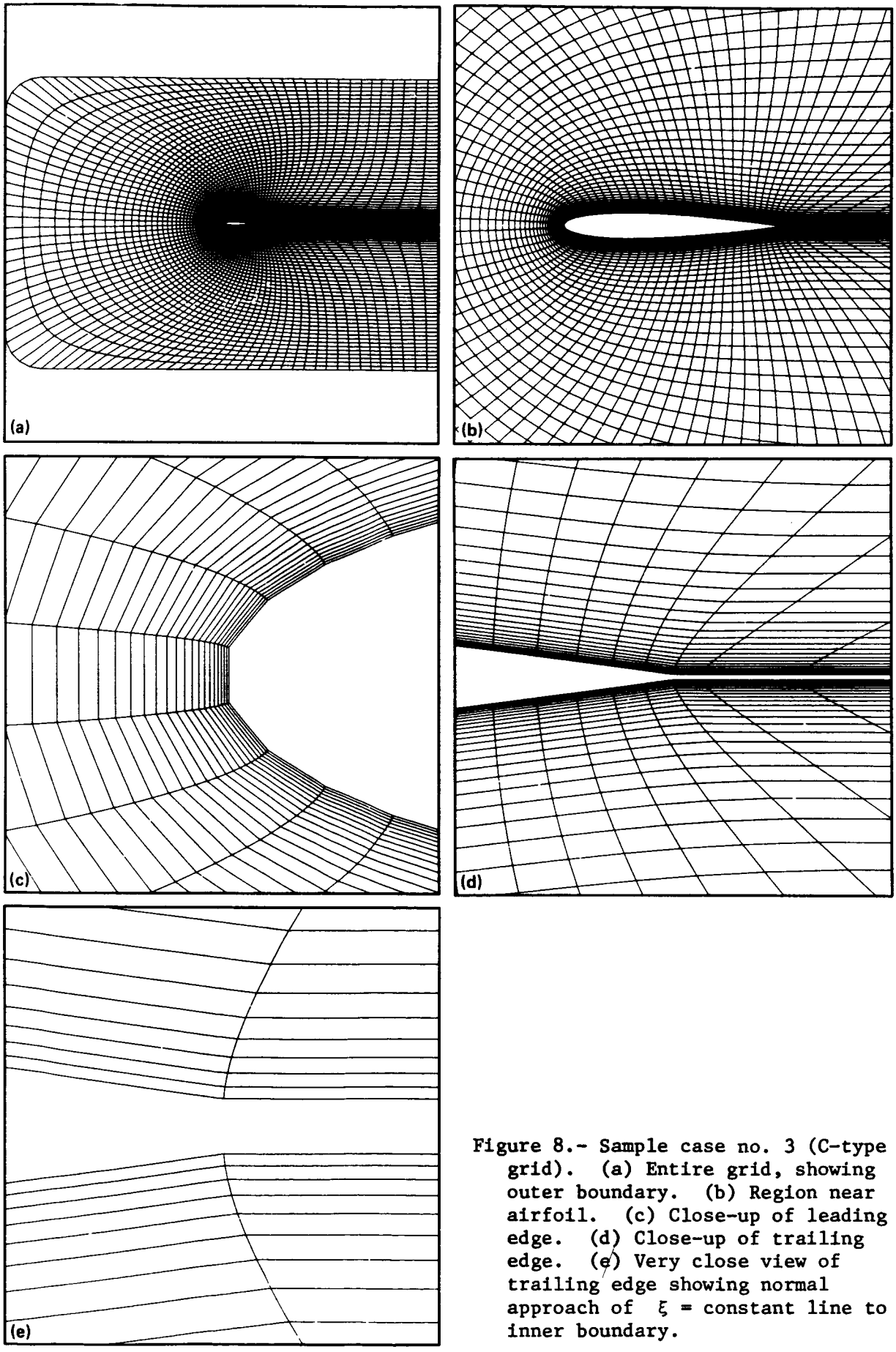


Figure 8.- Sample case no. 3 (C-type grid). (a) Entire grid, showing outer boundary. (b) Region near airfoil. (c) Close-up of leading edge. (d) Close-up of trailing edge. (e) Very close view of trailing edge showing normal approach of ξ = constant line to inner boundary.

Sample Case No. 4

Required data cards

```
$GRID1 JMAX=133,  KMAX=70,  NTETYP=2,   NOBSHP=3,  YBOTOM=-2.,  YTOP=2.,
ALAMF=30.,  ALAMR=20.,  NPLT=9$
$GRID2 JCAMBR=25,  DSORI=.05, CCCI=.3,  DDDI=.3$
$GRID3 CAMBRX=0.,
              .04166667,   .08333333,   .12500000,   .16666667,   .20833333,
   .25000000, .29166667,   .33333333,   .37500000,   .41666667,   .45833333,
   .50000000, .54166667,   .58333333,   .62500000,   .66666667,   .70833333,
   .75000000, .79166667,   .83333333,   .87500000,   .91666667,   .95833333,
  1.00000000,                CAMBRY=0.,
              .03194444,   .06111111,   .08750000,   .11111111,   .13194444,
   .15000000, .16527778,   .17777778,   .18750000,   .19444444,   .19861111,
   .20000000, .19861111,   .19444444,   .18750000,   .17777778,   .16527778,
   .15000000, .13194444,   .11111111,   .08750000,   .06111111,   .03194444,
   .00000000,
XMIN=-6.,-6.,-1.,1.,-.5,-.1,.9,.99,.999,  XMAX=6.,0.,2.,6.,1.5,.1,1.1,1.01,
1.001,  YMIN=0.,-5.,-2.75,-4.,  YMAX=0.,2.,2.,2.$
```

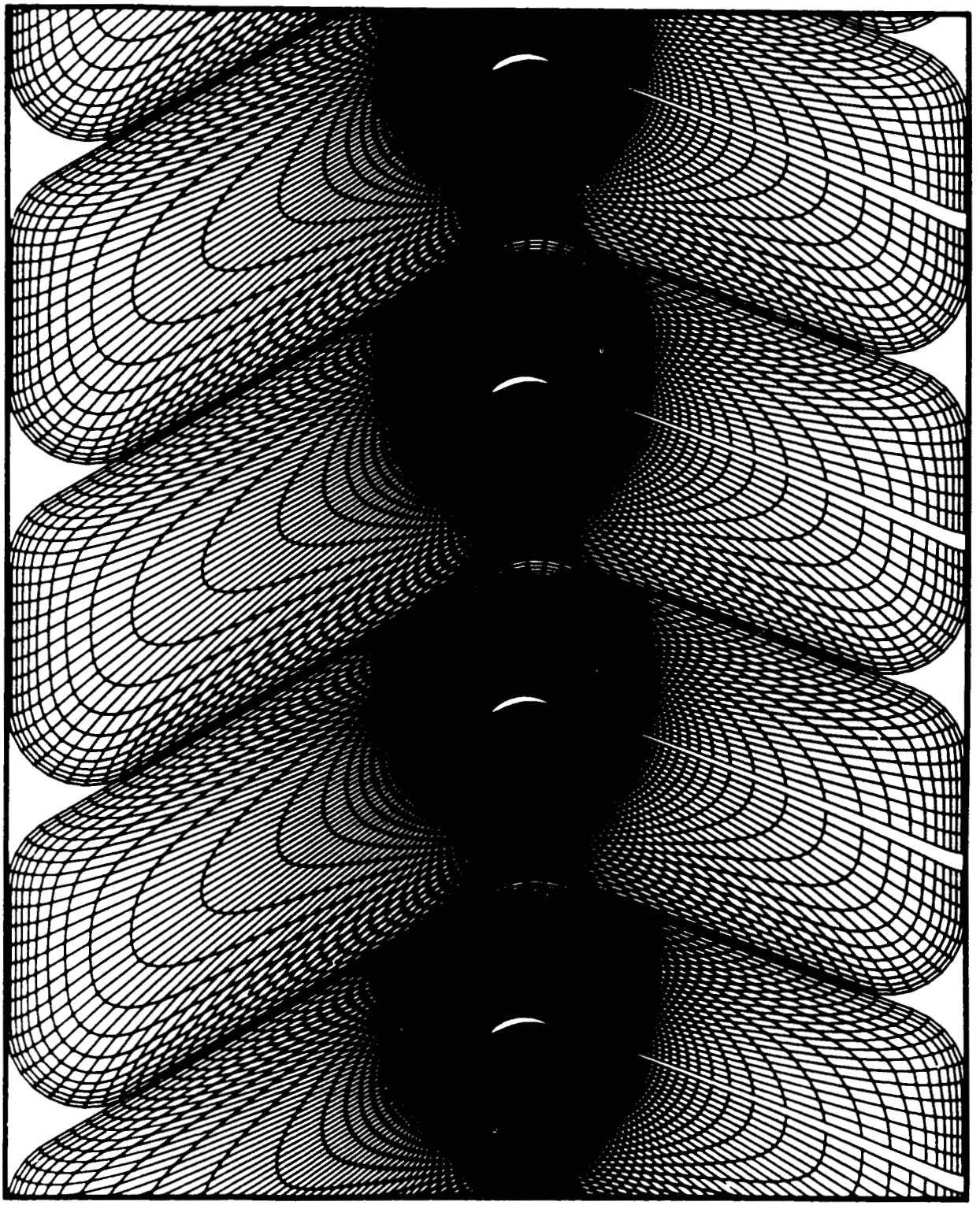
Number of iterations required for convergence
- 163 coarse
- 6 fine

7600 C.P.U. time for run
- 10.411 sec without plots
- 29.553 sec with plots

Number of points in mesh: 9660

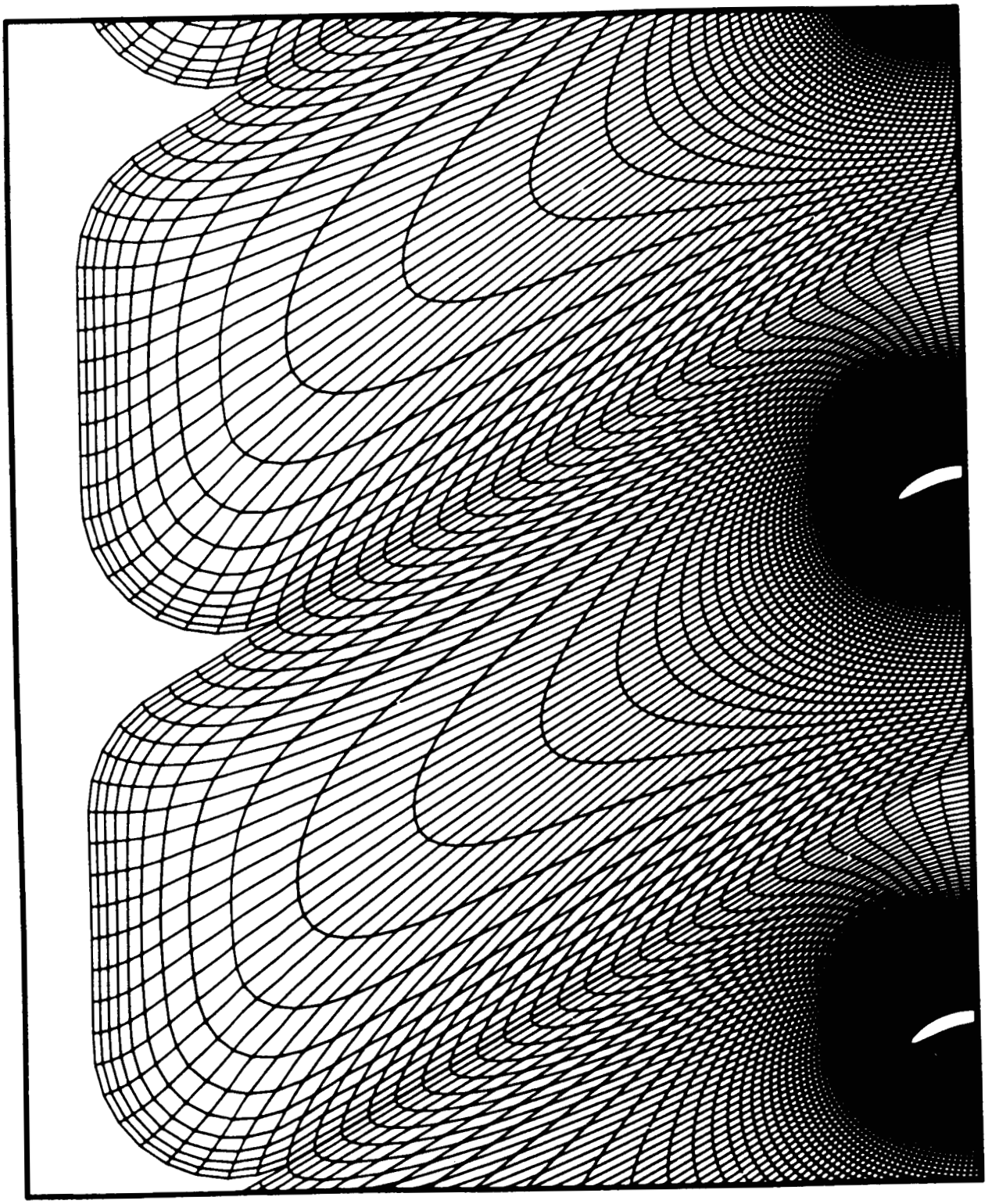
This case is an O-type grid for a cascade (NOBSHP = 3). The airfoil is an NACA 0012, cambered according to a circular arc (given by CAMBRX and CAMBRY). Although the airfoil is modified to have a closed trailing edge, the spacing on the airfoil is chosen such that the trailing edge is midway between two points (NTETYP = 2). The front (upstream) end of the grid is declined at 30° from the x-axis and the rear (downstream) at 20° (specified by ALAMF and ALAMR).

A previous run showed the ξ = constant lines intersecting the upper and lower boundaries in a manner significantly deviant from vertical. To improve this c and d were decreased (CCCI = 0.3, DDDI = 0.3). The results are shown in figure 9.



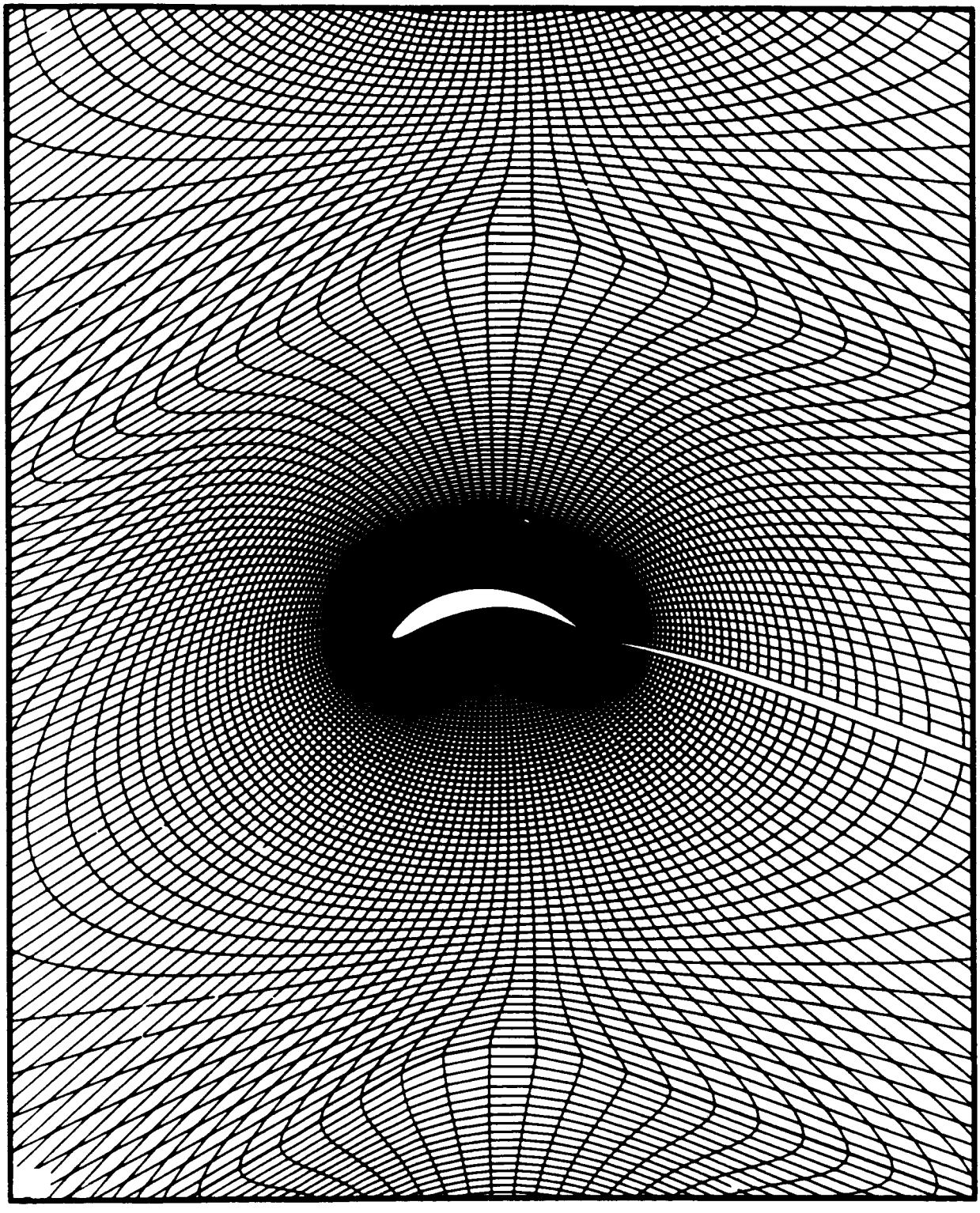
(a) Several cycles of entire grid, showing periodicity in vertical direction.

Figure 9.- Sample case no. 4 (cascade).



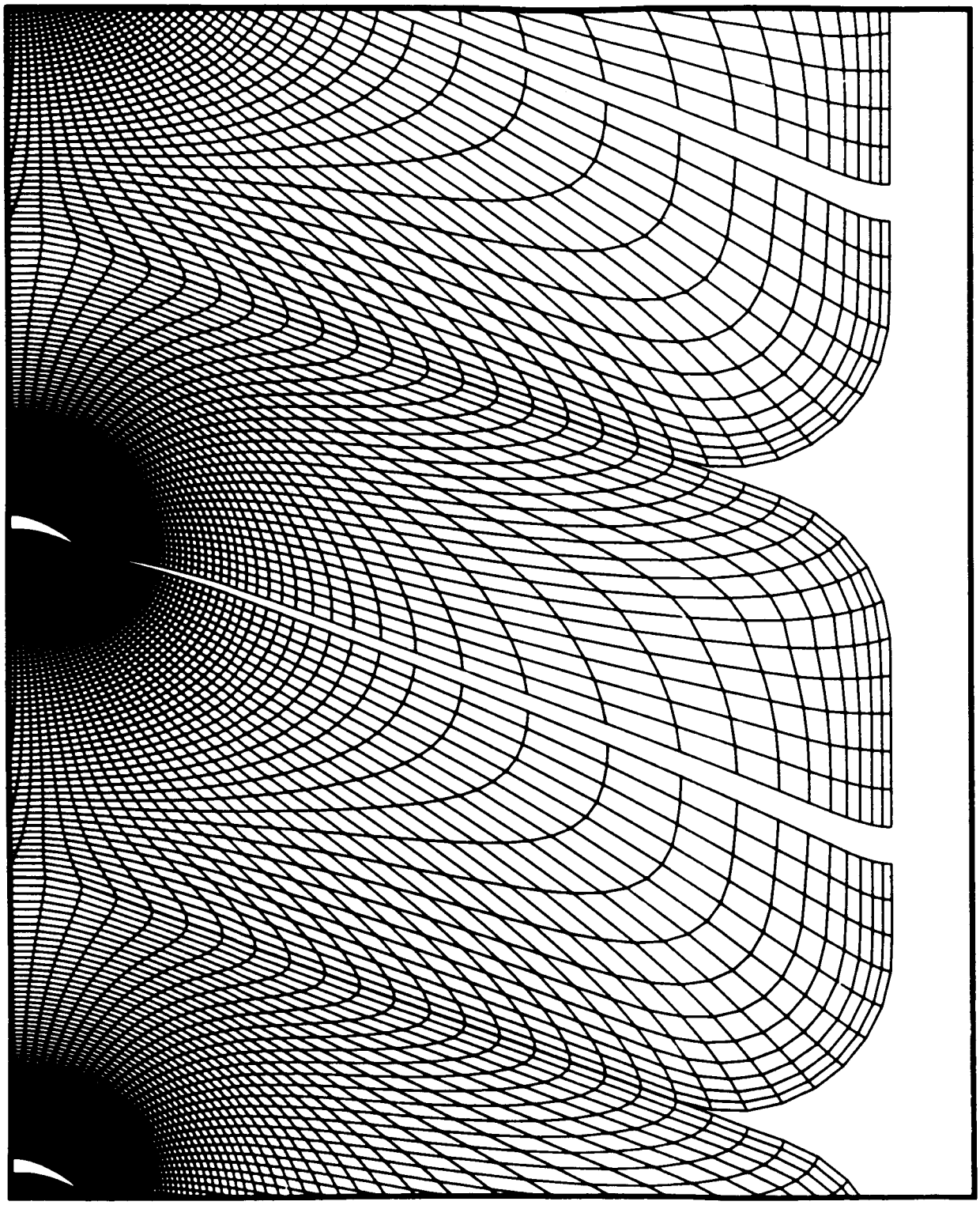
(b) Front (upstream) end.

Figure 9.- Continued.



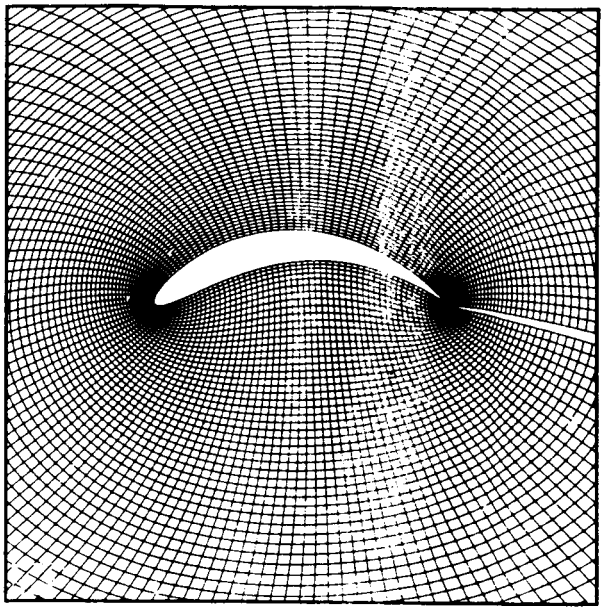
(c) Region near airfoil.

Figure 9.- Continued.

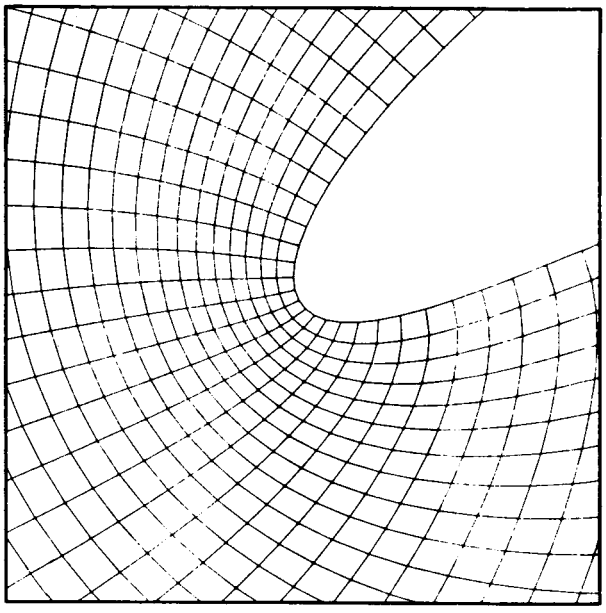


(d) Rear (downstream) end.

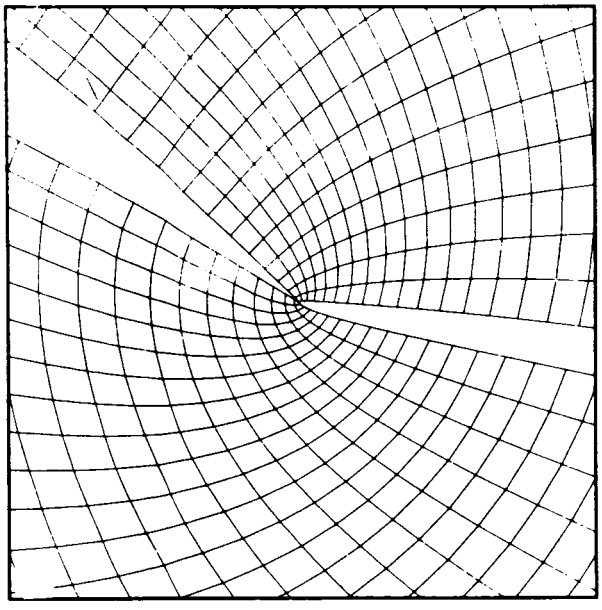
Figure 9.- Continued.



(e) Airfoil.



(f) Close-up of leading edge.



(g) Close-up of trailing edge.

Figure 9.- Concluded.

Sample Case No. 5

Required data cards

```
$GRID1  NTETYP=3,  NIBDST=5,  JTEBOT=1,  JTETOP=61,  JMAX=61,  JAIRF=61,
NOBSHP=6,  NAIRF=5,  NPLT= 6,  DS1=.5E-4,  NORDA=6,3$
$GRID2 DS0BI=.05,  AAAI=.6,  BBBI=.6,  CCCI=.6,  DDDI=.6,
OMEGA=1.1,  OMEGP=.1,  OMEGQ=.1,  OMEGR=.1,  OMEGS=.1,
PLIM=.5,  QLIM=.5,  RLIM=.5,  SLIM=.5$
$GRID3 AIRFX=2.,
                1.91504336,  1.83008672,  1.74513008,  1.66017344,  1.57521680,
  1.49026016,  1.40530352,  1.32034688,  1.23539024,  1.15043360,  1.06547696,
   .98052032,   .89556368,   .81062990,   .72673285,   .64486819,   .56564480,
   .48965190,   .41745466,   .34959007,   .28656284,   .22884175,   .17685609,
   .13099249,   .09159207,   .05894786,   .03330266,   .01484719,   .00371871,
  0.00000000,   .00371871,   .01484719,   .03330266,   .05894786,   .09159207,
   .13099249,   .17685609,   .22884175,   .28656284,   .34959007,   .41745466,
   .48965190,   .56564480,   .64486819,   .72673285,   .81062990,   .89556368,
   .98052032,  1.06547696,  1.15043360,  1.23539024,  1.32034688,  1.40530352,
  1.49026016,  1.57521680,  1.66017344,  1.74513008,  1.83008672,  1.91504336,
  2.00000000,
AIRFY=-1.19175359,
               -1.17677344, -1.16179330, -1.14681315, -1.13183300, -1.11685285,
 -1.10187271, -1.08689256, -1.07191241, -1.05693226, -1.04195211, -1.02697197,
 -1.01199182,  -.99701167,  -.98193578,  -.96193818,  -.93481624,  -.90074167,
  -.85996791,  -.81279821,  -.75958338,  -.70071922,  -.63664350,  -.56783281,
  -.49479890,  -.41808496,  -.33826155,  -.25592234,  -.17167973,  -.08616027,
  0.00000000,   .08616027,   .17167973,   .25592234,   .33826155,   .41808496,
   .49479890,   .56783281,   .63664350,   .70071922,   .75958338,   .81279821,
   .85996791,   .90074167,   .93481624,   .96193818,   .98190578,   .99701167,
  1.01199182,  1.02697197,  1.04195211,  1.05693226,  1.07191241,  1.08689256,
  1.10187271,  1.11685285,  1.13183300,  1.14681315,  1.16179330,  1.17677344,
  1.19175359,
XOB=2.,         1.90275660,  1.80551319,  1.70826979,  1.61102638,  1.51378298,
  1.41653957,  1.31929617,  1.22205276,  1.12480936,  1.02756595,   .93032255,
   .83307914,   .73583574,   .63859233,   .54134893,   .44410552,   .34686212,
   .24961871,   .15237531,   .05513190,  -.04211150,  -.13935491,  -.23659831,
  -.33384172,  -.43108512,  -.52508413,  -.60085368,  -.65569366,  -.68888709,
  -.70000000,  -.68888709,  -.65569366,  -.60085368,  -.52508413,  -.43108512,
  -.33384172,  -.23659831,  -.13935491,  -.04211150,   .05513190,   .15237531,
   .24961871,   .34686212,   .44410552,   .54134893,   .63859233,   .73583574,
   .83307914,   .93032255,  1.02756595,  1.12480936,  1.22205276,  1.31929617,
  1.41653957,  1.51378298,  1.61102638,  1.70826979,  1.80551319,  1.90275660,
  2.00000000,
YOB=-5.13205081,
               -4.96362029, -4.79518977, -4.62675926, -4.45832874, -4.28989822,
 -4.12146770, -3.95303718, -3.78460667, -3.61617615, -3.44774563, -3.27931511,
 -3.11088459, -2.94245408, -2.77402356, -2.60559304, -2.43716252, -2.26873200,
 -2.10030149, -1.93187097, -1.76344045, -1.59500993, -1.42657941, -1.25814890,
 -1.08971838,  -.92128786,  -.75107815,  -.57207297,  -.38558850,  -.19406284,
  0.00000000,   .19406284,   .38558850,   .57207297,   .75107815,   .92128786,
  1.08971838,  1.25814890,  1.42657941,  1.59500993,  1.76344045,  1.93187097,
  2.10030149,  2.26873200,  2.43716252,  2.60559304,  2.77402356,  2.94245408,
  3.11088459,  3.27931511,  3.44774563,  3.61617615,  3.78460667,  3.95303718,
  4.12146770,  4.28989822,  4.45832874,  4.62675926,  4.79518977,  4.96362029,
  5.13205081,
THETA=80.,82.,84.,86.,88.,51*90.,92.,94.,96.,98.,100.,
THETOB=30.,40.,50.,60.,70.,80.,49*90.,100.,110.,120.,130.,140.,150.,
XMIN=0.,0.,-.75,-.25,  XMAX=0.,0.,.25,2.,2.,2.,
YMIN=-5.25,0.,0.,-1.25,1.14,3.14,  YMAX=5.25,5.25,0.,1.25,3.14,5.14$
```

Number of iterations required for convergence
120 coarse
67 fine

7600 C.P.U. time for run
9.288 sec without plots
13.658 sec with plots

Number of points in mesh: 2999

This case illustrates the capability to generate a grid about a body other than an airfoil. A grid with viscous spacing (DSI = 0.00005) was desired about a sphere-cone reentry body. A shock-fitting technique was to be used, hence the outer boundary likewise was to be a sphere-cone, but with a larger cone-angle. The grid was, of course, a two-dimensional cut through the above described three-dimensional shape (see fig. 10(a)).

Since the η = constant lines are not closed curves with periodicity, but rather, resemble the letter "C," the grid was selected to be of the C-type (NTETYP = 3). The inner and outer boundaries, each having 61 points, were read from cards using AIRFX, AIRFY, XOB, and YOB. Likewise it was necessary to specify, on cards, THETA and THETOB.

To achieve good angle control at boundaries in a viscous grid, more convergence was required (NORDA = 6,3). To facilitate numerical stability during convergence, AAAI, BBBI, CCCI, and DDDI were each set to 0.6; OMEGA was set to 1.1; OMEGP, OMEGQ, OMEGR, and OMEGS were set to 0.1; and PLIM, QLIM, RLIM, and SLIM were set to 0.5.

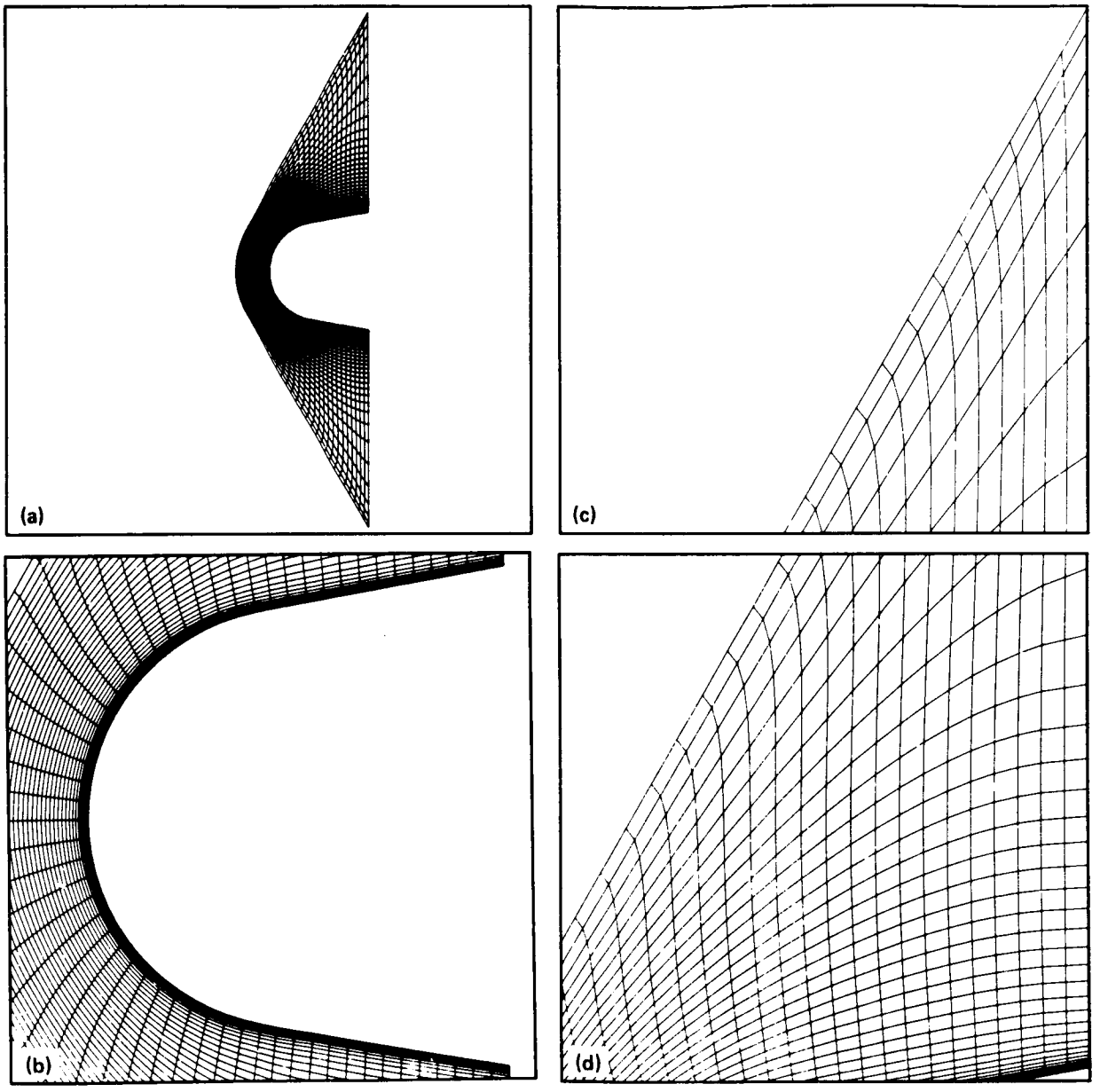


Figure 10.- Sample case no. 5 (sphere-cone reentry body). (a) Entire grid, showing outer boundary. (b) Entire body. (c) Upper half of rear boundary. (d) Lower half of rear boundary.

APPENDIX C

SUBROUTINES AND FLOWCHARTS

The following is an alphabetical list of the subroutines, a short description of the function of each, and a list of subroutines which call and are called by each subroutine.

Name	Function	Calls	Called by
CKSMTH	Checks arrays for smoothness		INCHK
CSPLIN	Cubic spline fit	TRIB	INNER, INTERP
IC	Initial conditions for X, Y, p, q, r, s		RELAX, SOLVE
INCHK	Checks input data	CKSMTH	MAIN
INNER	Locates points on inner (airfoil) boundary	CSPLIN	MAIN
INPUT	Reads initial data		MAIN
INTERP	Interpolates from coarse solution to obtain initial conditions for fine solution	CSPLIN	SOLVE
OUTER	Locates points on outer boundary		MAIN
OUTPUT	Writes finished grid	PLAWT plotting software	MAIN
PLAWT	Plots finished grid	Plotting software	OUTPUT
RELAX	Applies SLOR solution procedure to Poisson equation	TRIB, TRIP IC	SOLVE
SOLVE	Effects coarse and fine solutions of Poisson equation	RELAX, IC, INTERP	MAIN
TRIB	Solves tridiagonal system of linear equations		RELAX, CSPLIN
TRIP	Solves tridiagonal system of linear equations with periodic boundary conditions		RELAX

Subroutine Flowcharts

Figures 11 through 18 are flowcharts of the main program; subroutines INCHK, INNER, OUTER, SOLVE, and OUTPUT, which are called by the main program; and subroutines RELAX and INTERP, called by SOLVE. The starting points of the main program and of each of the subroutines, RETURN statements, and STOP statements are enclosed by circles. Blocks of code, each performing a particular function are indicated by rectangles composed of solid lines. Words inside the rectangles describe the functions of the blocks of code. Functions performed by subroutines, invoked by calling those subroutines, are indicated similarly by rectangles composed of dashed lines. The names of the called subroutines are indicated above the upper right-hand corners of the dashed line rectangles. Branch points having two or three exit paths are indicated by diamond shapes. Branches having more than three exit paths are indicated by circles. DO statements are indicated in a manner easily understood by users of FORTRAN. The numbers immediately above some of the symbols are statement label numbers.

These flowcharts are not exhaustive in detail; if they were their length would be unmanageable. Simplifications have been effected for brevity and clarity. Nevertheless, the flowcharts should provide a good orientation and introduction for someone determined to understand and modify the code.

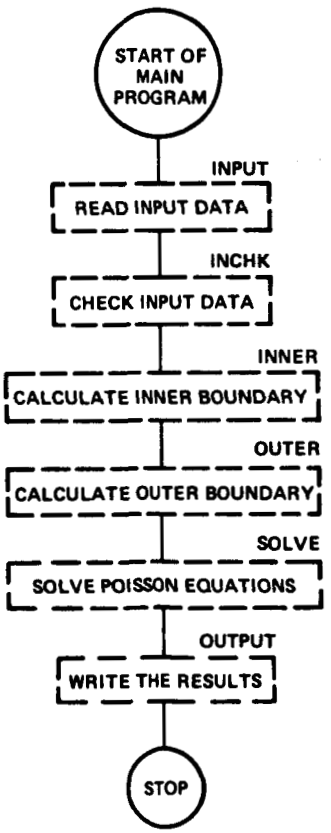


Figure 11.- Main program.

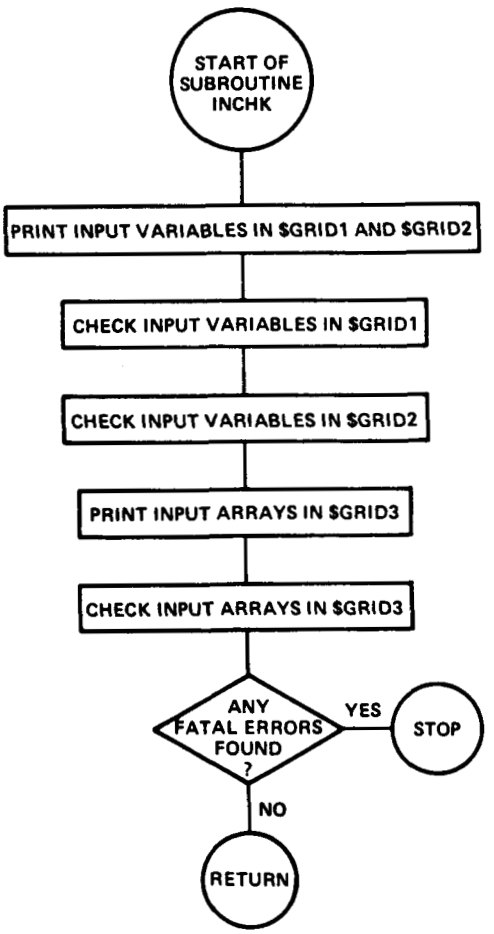


Figure 12.- Subroutine INCHK.

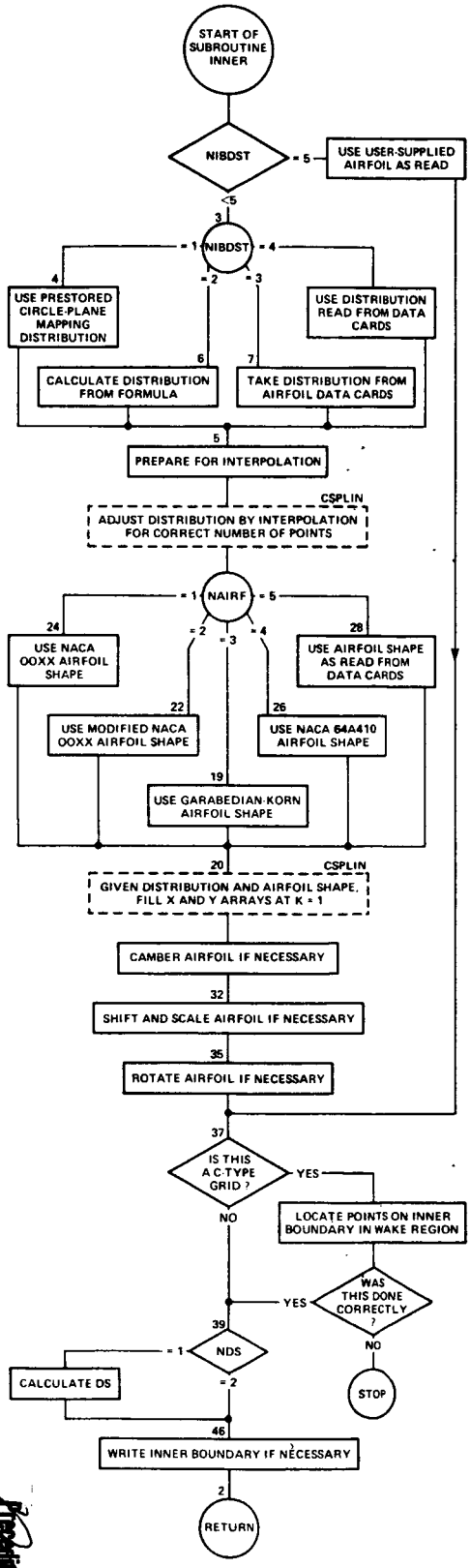


Figure 13.- Subroutine INNER.

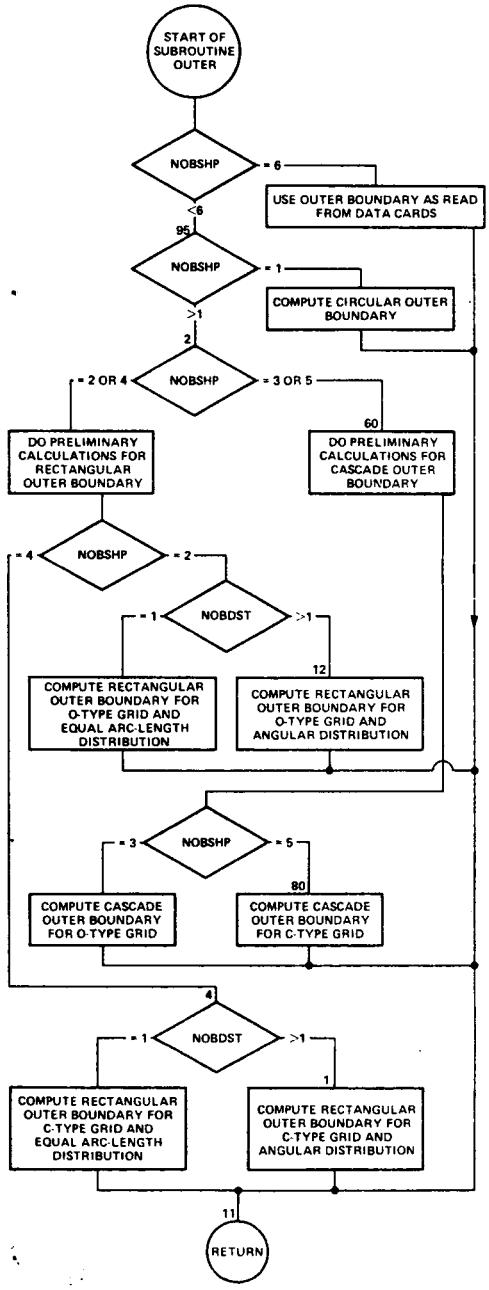


Figure 14.- Subroutine OUTER.

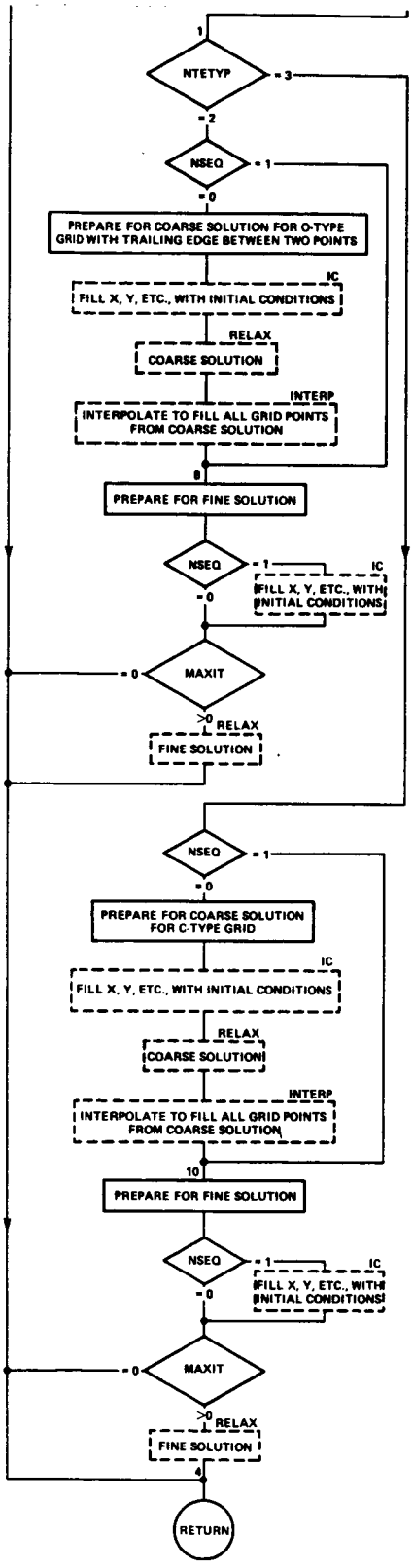
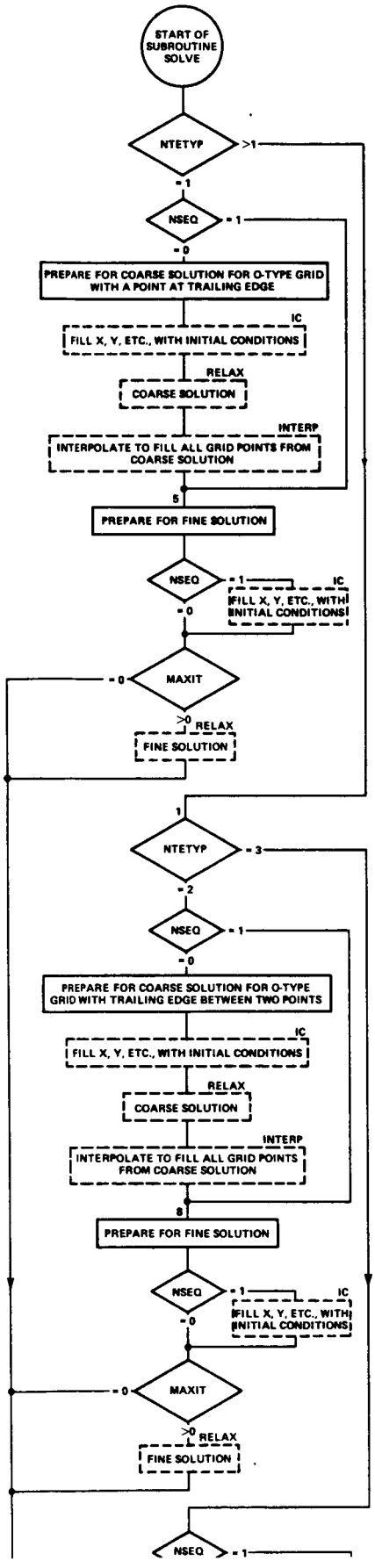


Figure 15.- Subroutine SOLVE.



54p.

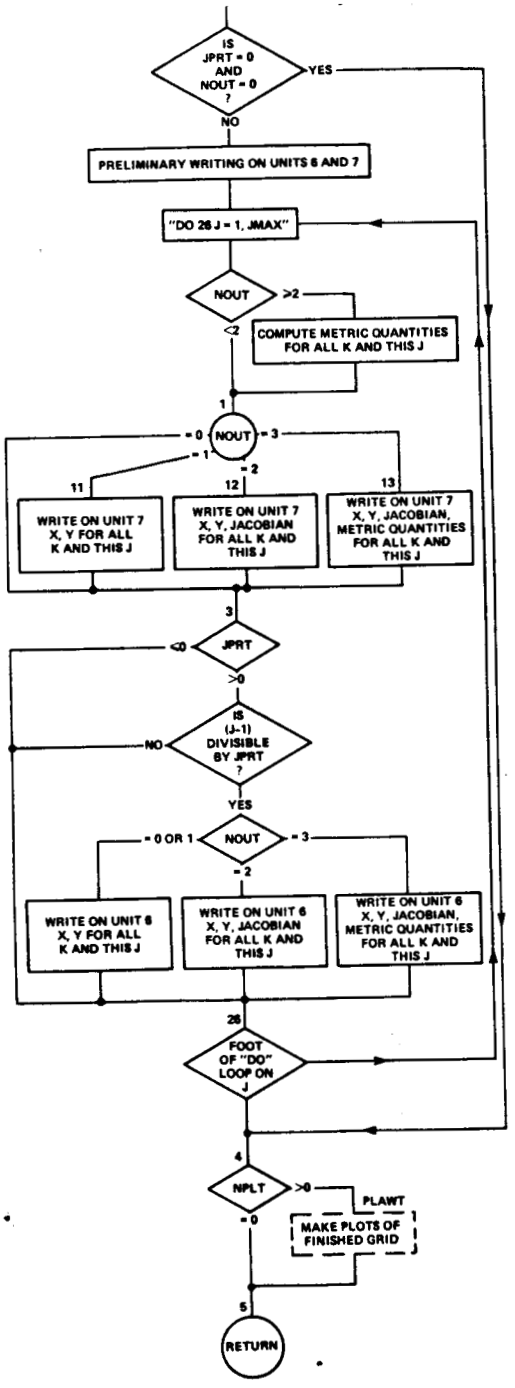


Figure 16.- Subroutine OUTPUT.

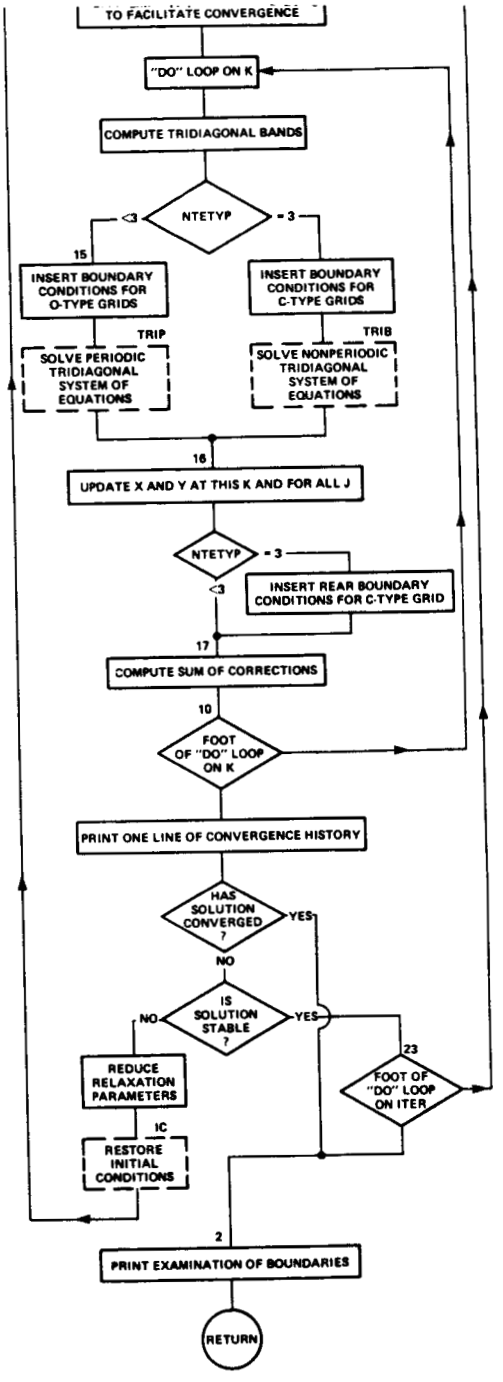
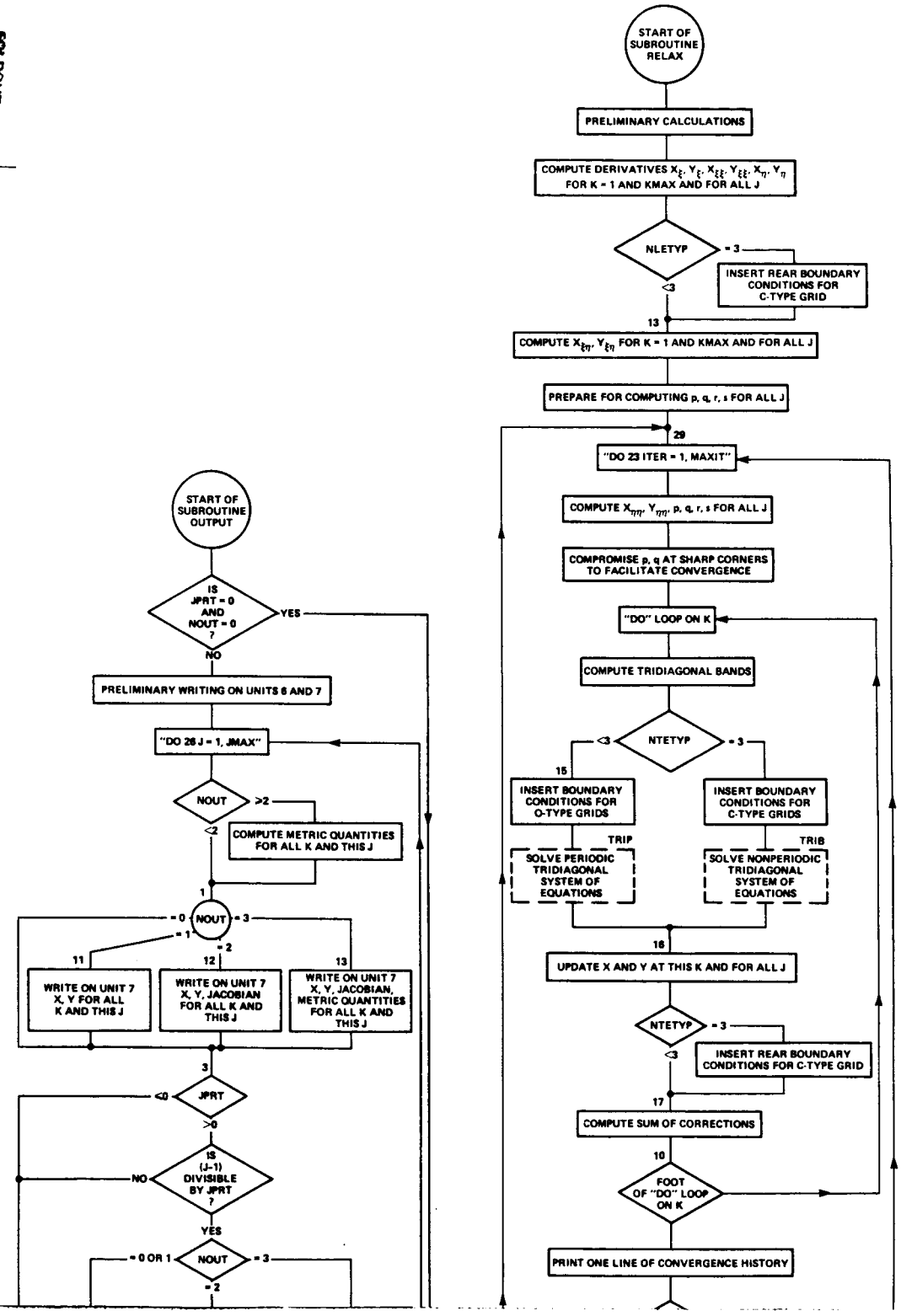


Figure 17.- Subroutine RELAX.

FOLDOUT FRAME 2

FOLDOUT FRAME 3


START OF SUBROUTINE RELAX
PRELIMINARY CALCULATIONS
COMPUTE DERIVATIVES X_ξ, Y_ξ, $X_{\xi\xi}$, $Y_{\xi\xi}$, X_η, Y_η FOR K = 1 AND KMAX AND FOR ALL J
NLETYP
= 3
INSERT REAR BOUNDARY CONDITIONS FOR C-TYPE GRID
<3
13
COMPUTE $X_{\xi\eta}$, $Y_{\xi\eta}$ FOR K = 1 AND KMAX AND FOR ALL J
PREPARE FOR COMPUTING p, q, r, s FOR ALL J
29
"DO 23 ITER = 1, MAXIT"
COMPUTE $X_{\eta\eta}$, $Y_{\eta\eta}$, p, q, r, s FOR ALL J
COMPROMISE p, q AT SHARP CORNERS TO FACILITATE CONVERGENCE
"DO" LOOP ON K
COMPUTE TRIDIAGONAL BANDS
NTETYP
<3
= 3
15
INSERT BOUNDARY CONDITIONS FOR O-TYPE GRIDS
INSERT BOUNDARY CONDITIONS FOR C-TYPE GRIDS
TRIP
SOLVE PERIODIC TRIDIAGONAL SYSTEM OF EQUATIONS
TRIB
SOLVE NONPERIODIC TRIDIAGONAL SYSTEM OF EQUATIONS
18
UPDATE X AND Y AT THIS K AND FOR ALL J
NTETYP
= 3
<3
INSERT REAR BOUNDARY CONDITIONS FOR C-TYPE GRID
17
COMPUTE SUM OF CORRECTIONS
10
FOOT OF "DO" LOOP ON K
PRINT ONE LINE OF CONVERGENCE HISTORY
START OF SUBROUTINE OUTPUT
IS JPRT = 0 AND NOUT = 0 ?
YES
NO
PRELIMINARY WRITING ON UNITS 6 AND 7
"DO 26 J = 1, JMAX"
NOUT
>2
<2
COMPUTE METRIC QUANTITIES FOR ALL K AND THIS J
1
NOUT
= 0
= 1
= 2
= 3
11
WRITE ON UNIT 7 X, Y FOR ALL K AND THIS J
12
WRITE ON UNIT 7 X, Y, JACOBIAN FOR ALL K AND THIS J
13
WRITE ON UNIT 7 X, Y, JACOBIAN, METRIC QUANTITIES FOR ALL K AND THIS J
3
JPRT
≤0
>0
IS (J-1) DIVISIBLE BY JPRT ?
NO
YES
NOUT
= 0 OR 1
= 2
= 3

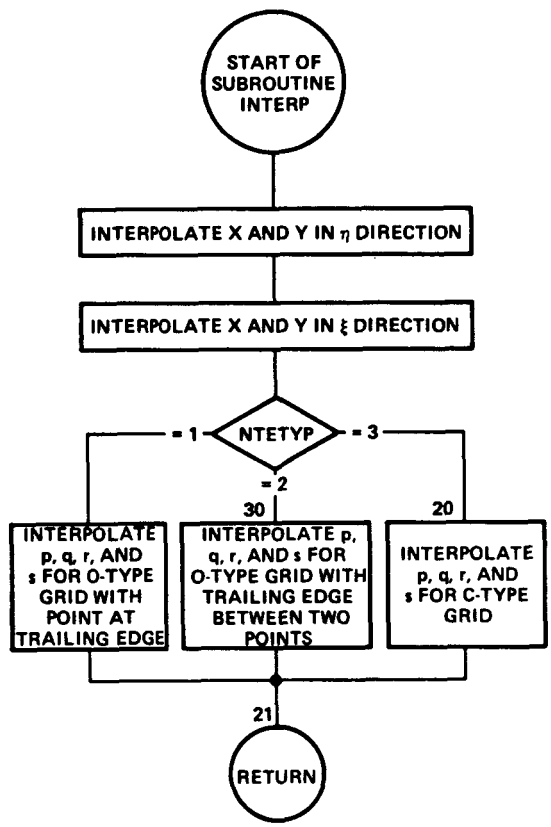


Figure 18.- Subroutine INTERP.

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-Dimensional Bodies. J. Comp. Phys., vol. 15, no. 3, July 1974, pp. 299-319.

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. Phys., vol. 33, no. 3, Dec. 1979, pp. 405-410.

3. Abbott, I. H.; and Von Doenhoff, A. E.: Theory of Wing Sections. Dover Publications, Inc., New York, 1959.

4. Bauer, F.; Garabedian, P.; Korn, D.; and Jameson, A.: Supercritical Wing Sections II. Lecture Notes in Economical and Mathematical Systems, Springer-Verlag, New York, 1975.

1. Report No.	2. Government Accession No.	3. Recipient's Catalog No.
NASA TM-81198		

4. Title and Subtitle	5. Report Date
A COMPUTER PROGRAM TO GENERATE TWO-DIMENSIONAL GRIDS ABOUT AIRFOILS AND OTHER SHAPES BY THE USE OF POISSON'S EQUATION	6. Performing Organization Code

7. Author(s)	8. Performing Organization Report No.
Reese L. Sorenson	A-8178
	10. Work Unit No. 505-31-11
9. Performing Organization Name and Address	11. Contract or Grant No.
Ames Research Center, NASA Moffett Field, Calif. 94035	
	13. Type of Report and Period Covered
12. Sponsoring Agency Name and Address	Technical Memorandum
National Aeronautics and Space Administration Washington, D.C. 20546	14. Sponsoring Agency Code

15. Supplementary Notes

16. Abstract

A method for generating two-dimensional finite-difference grids about airfoils and other shapes by the use of the Poisson differential equation is developed. The inhomogeneous terms are automatically chosen such that two important effects are imposed on the grid at the inner (airfoil) boundary and at the outer boundary. The first effect is control of the spacing between mesh points, along mesh lines intersecting the boundaries. The second effect is control of the angles with which mesh lines intersect the boundaries. A FORTRAN computer program has been written to use this method. The program is available upon request from the Applied Computational Aerodynamics Branch, Mail Stop 202A-14, NASA Ames Research Center, Moffett Field, Calif. 94035. A description of the program, a discussion of the control parameters, and a set of sample cases are included.

17. Key Words (Suggested by Author(s))	18. Distribution Statement
Numerical methods Grid generation	Unlimited STAR Category - 02

19. Security Classif. (of this report)	20. Security Classif. (of this page)
Unclassified	Unclassified

*For sale by the National Technical Information Service, Springfield, Virginia 22161