LA-11009-MS

2

NASA-VOF3D: A Three-Dimensional Computer Program for Incompressible Flows with Free Surfaces

Los Alamos National Laboratory
Los Alamos, New Mexico 87545

This work was supported by the US Department of Energy and the National Aeronautics and Space Administration, Lewis Research Center.

Prepared by Juanita Salazar, Group T-3

LA11009-MS

UC-32
Issued: July 1987

NASA-VOF3D: A Three-Dimensional Computer Program for Incompressible Flows with Free Surfaces

Martin D. Torrey
Raymond C. Mjolsness
Leland R. Stein

Los Alamos
Los Alamos National Laboratory
Los Alamos, New Mexico 87545

TABLE OF CONTENTS

NASA-VOF3D: A THREE-DIMENSIONAL COMPUTER PROGRAM FOR INCOMPRESSIBLE FLOWS WITH FREE SURFACES*

by

Martin D. Torrey, Raymond C. Mjolsness, and Leland R. Stein

ABSTRACT

We present the NASA-VOF3D three-dimensional, transient, free-surface hydrodynamics program. This three-dimensional extension of NASA-VOF2D will, in principle, permit treatment in full three-dimensional generality of the wide variety of applications that could be treated by NASA-VOF2D only within the two-dimensional idealization. In particular, it, like NASA-VOF2D, is specifically designed to calculate confined flows in a low g environment. The code is presently restricted to cylindrical geometry. The code is based on the fractional volume-of-fluid method and allows multiple free surfaces with surface tension and wall adhesion. It also has a partial cell treatment that allows curved boundaries and internal obstacles. This report provides a brief discussion of the numerical method, a code listing, and some sample problems.

I. INTRODUCTION

The NASA-VOF3D program simulates three-dimensional incompressible flows with free surfaces using the volume-of-fluid (VOF) algorithm.[1-3] It is specifically designed to calculate flows in a low g environment, in which surface physics must be accurately treated. This technique is based on the use of donor-acceptor differencing[4] to track the free surface across an Eulerian grid. Similar procedures were developed contemporarily by other workers[5-9] to track material interfaces as well as free surfaces. A similar, but 2D, program has been written specifically for simulating the draining of spacecraft propellant tanks.[10] Both the NASA-VOF2D and the NASA-VOF3D programs contain several improvements over these codes, and these improvements will be described along with the governing equations, the numerical algorithm, a program description and several numerical solutions. A program

*The code is available through the National Energy Software Center, 9700 South Cass Avenue, Argonne, IL 60439, telephone (312/972-7250).

listing will be presented. Because much of the numerical methodology is very similar in 2D and in 3D, we will frequently refer to the NASA-VOF2D report[11] for a fuller description of topics mentioned here.

In particular, Reference 11 discusses the developments leading to the use of SOLA-VOF methodology in low g flows. We have incorporated the NASA-VOF2D improvements in donor-acceptor differencing, and in the inclusion of a conjugate residual option as well as a successive-over-relaxation (SOR) option for advancing the pressures (and velocities) in time throughout the computing mesh. The conjugate residual method is vectorized for the Cray and employs a scaled coefficient matrix, as in NASA-VOF2D. We found it necessary for accuracy to radically alter the procedure for calculating 3D surface tension effects. The choice of surface orientation, greatly generalized from the 2D procedure, may require further attention. We have <u>not</u> included the 3D extension of our 2D wall-adhesion procedure for curved boundaries; this must be done to obtain a truly general-purpose code.

The basic algorithm is based on the Navier-Stokes equations for an incompressible fluid, which we record here in cylindrical coordinates only. We introduce:

x, θ, z for the radial, azimuthal, and axial coordinates

u, v, w for the corresponding velocity components

g_x, g_y, g_z, for the corresponding components of external acceleration

p, ρ, and Θ for the pressure, density, and partial cell parameter.[11]

Then the continuity equation is

$$\Delta \cdot (\Theta \mathbf{u}) = \frac{1}{x}\frac{\partial (x \Theta u)}{\partial x} + \frac{1}{x}\frac{\partial (\Theta v)}{\partial \theta} + \frac{\partial (\Theta w)}{\partial z} = 0 \ , \tag{1}$$

and the momentum equations are

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} + \frac{v}{x}\frac{\partial u}{\partial \theta} + w\frac{\partial u}{\partial z} - \frac{v^2}{x}$$

$$= g_x - \frac{1}{\rho}\frac{\partial p}{\partial x} + \nu \left| \frac{\partial^2 u}{\partial x^2} + \frac{1}{x}\frac{\partial u}{\partial x} - \frac{1}{x^2}u + \frac{1}{x^2}\frac{\partial^2 u}{\partial \theta^2} + \frac{\partial^2 u}{\partial z^2} - \frac{2}{x^2}\frac{\partial v}{\partial \theta} \right| \ , \tag{2}$$

$$
\frac{\partial v}{\partial t} + u \frac{\partial v}{\partial x} + \frac{v}{x} \frac{\partial v}{\partial \theta} + w \frac{\partial v}{\partial z} + \frac{1}{x} u v
$$

$$
= g_y - \frac{1}{\rho x} \frac{\partial p}{\partial \theta} + \nu \left[\frac{\partial^2 v}{\partial x^2} + \frac{1}{x} \frac{\partial v}{\partial x} - \frac{1}{x^2} v + \frac{1}{x^2} \frac{\partial^2 v}{\partial \theta^2} + \frac{\partial^2 v}{\partial z^2} + \frac{2}{x^2} \frac{\partial u}{\partial \theta} \right] , \tag{3}
$$

and

$$
\frac{\partial w}{\partial t} + u \frac{\partial}{\partial x} w + \frac{v}{x} \frac{\partial w}{\partial \theta} + w \frac{\partial}{\partial z} w
$$

$$
= g_z - \frac{1}{\rho} \frac{\partial p}{\partial z} + \nu \left[\frac{\partial^2 w}{\partial x^2} + \frac{1}{x} \frac{\partial w}{\partial x} + \frac{1}{x^2} \frac{\partial^2 w}{\partial \theta^2} + \frac{\partial^2 w}{\partial z^2} \right] , \tag{4}
$$

where ν is the kinematic viscosity. These equations apply at every point inside the fluid. When implemented numerically, all equations use the arc length azimuthal coordinate $y = x_{IM1}\,\theta$ rather than θ, where x_{IM1} is the radius of the computational mesh; this emphasizes the similarity of our equations to Cartesian equations.

The free surfaces are treated by introducing a function $F(x, \theta, z, t)$ that is defined to be unity at any point occupied by the fluid and zero elsewhere. When averaged over a cell of the computing mesh, $\langle F \rangle_{cell}$ is the fractional volume of the cell occupied by fluid. Thus, $\langle F \rangle_{cell} = 1$ implies a cell full of fluid, while $\langle F \rangle_{cell} = 0$ denotes an empty cell. Finite-difference solution of the governing equations for F, (Eq. 5), gives $\langle F \rangle_{cell}$ directly. Cells with F values between zero and one are partially filled with fluid; they are either intersected by a free surface or contain voids (bubbles) smaller than cell mesh dimensions. The F function is used to determine which cells contain a boundary and where the fluid is located in the cells. Gradients of F determine the mean local surface normal, and, together with F values, permit construction of an approximate interface. Surface-tension and wall-adhesion forces may then be evaluated. The special requirements for construction of gradients of the step function F are discussed in Reference 11. The governing equation for F is

$$\frac{\partial F}{\partial t} + u\frac{\partial F}{\partial x} + \frac{v}{x}\frac{\partial F}{\partial \theta} + w\frac{\partial F}{\partial z} = 0 \ . \qquad (5)$$

This approach allows the fluid interfaces to be represented simply via different equations. However, the flux of F through each face of an Eulerian grid will be required. Standard finite-difference techniques would lead to a smearing of F values, degrading the interface definition. Using the step function character of F, we employ a form of donor-acceptor differencing that preserves the discontinuous nature of F.

Section II describes the numerical method briefly, while Sec. III describes the construction of the program. In Sec. IV we present examples of the performance of the surface tension algorithm, while in Sec. V we present runs of an (initially) spherical bubble moving in an ellipsoidal tank. In Sec. VI we summarize what may be done to obtain from NASA-VOF3D a program of more general applicability for low g flows, but emphasize that the present form of the program has a very significant range of applicability. Finally, in Appendix A, we present a program listing.

II. THE NUMERICAL METHOD

This section provides brief descriptions of many features of the numerical method. Those features that are very similar to the 2D method are discussed at greater length in Reference 11. The program is Eulerian and allows for an arbitrary number of segments of free surface with any reasonable shape. Variable mesh spacing is permitted in all three coordinate directions, as is necessary for efficient numerical solution of many problems. The complete Navier-Stokes equations for an incompressible fluid are solved by finite differences, with surface-tension effects included. Wall adhesion may be included or neglected via a user option.

A. The Computational Mesh

The mesh is very similar to the 2D mesh of Reference 11. But note that the y coordinate of Reference 11 is our z coordinate. Thus $(\delta y_j)_{2D} \leftrightarrow (\delta z_k)_{3D}$, and δy_j for us denotes an increment in the azimuthal coordinate. There is a plane of fictitious cells on each boundary of the computational mesh and δx_i, δy_j, and δz_k denote the coordinate increments in the (i, j, k) cell. Thus

$i = 1, i = IMAX$

refers to non-physical cells and

$$i = 2, i = IM1 (\equiv IMAX - 1)$$

refers to the first and last physical cells in the x direction. Of course, because of the code's capability to treat curved boundaries, we mean only that i = IM1 is a physical cell for some value of j and k. Then

$$x_{i+1/2} = \sum_{i'=2}^{i} \Delta x_{i'} , \quad 2 \le i \le IM1$$

$$y_{j+1/2} = \sum_{j'=2}^{j} \Delta y_{j'} , \quad 2 \le j \le JM1$$

$$z_{k+1/2} = \sum_{k'=2}^{k} \Delta z_{k'} , \quad 2 \le k \le KM1$$

denote coordinate values on the right, back, and top faces (faces of maximal coordinate value) of the (i, j, k) cell and x_i, y_j, and z_k denote coordinates at cell center, e.g.,

$$x_i = \frac{1}{2} (x_{i+1/2} + x_{i-1/2}) .$$

We emphasize the program will readily accept variable mesh spacing, i.e., Δx_i may depend on i, Δy_j on j and Δz_k on k, subject only to reasonable requirements on avoiding neighboring cells of grossly different size and avoiding cells of large aspect ratio.[11]

The pressure $p_{i,j,k}$, fractional volume of fluid $F_{i,j,k}$, and fraction of the cell volume open to flow $AC_{i,j,k}$ are cell-centered quantities, while the velocities $u_{i+1/2,j,k}$, $v_{i,j+1/2,k}$, and $w_{i,j,k+1/2}$ and the fractional areas open to flow $AR_{i+1/2,j,k}$, $ABK_{i,j+1/2,k}$, and $AT_{i,j,k+1/2}$ are face-centered quantities, e.g., $v_{i,j+1/2,k}$, and $ABK_{i,j+1/2,k}$, are defined at $(x_i, y_{j+1/2}, z_k)$. The fractional areas and volumes open to flow are related to Θ of the continuity equation as described in Reference 11 (with the obvious extension to 3D geometry), e.g.,

$$ABK_{i,j+1/2,k} = \Theta_{i,j+1/2,k} \ .$$

Interior obstacles are defined as any nonflow region within the computational mesh; they are specified by a series of 2D conic sections (in x, z plane)

$$0(x,z) = a_2 x^2 + a_1 x + b_2 z^2 + b_1 z + c_2 x z + c_1 = 0 \ ,$$

such that some portion of each locus forms a portion of the obstacle, and by a parameter that flags a cell overlapping an obstacle as closed to flow, partially closed to flow, or open to flow. We emphasize that only cylindrical obstacles can be treated by the present version of the code. The operation of the obstacle setting procedure is described in Reference 11. With the obstacles specified, the fractional areas and volumes open to flow, e.g., ABK and AC, are specified as in 2D.[11]

B. Boundary Conditions

The boundary conditions are imposed in a standard way on the six boundaries (left, right, forward, back, bottom, and top) of the computational mesh by six flags WL, WR, WF, WBK, WB, and WT. Integer numerical values for the flags are as follows: 1 denotes rigid free-slip wall, 2 denotes rigid no-slip wall, 3 denotes continuative boundary, 4 denotes periodic boundary (this value must appear on both flags for two opposite boundaries), and 5 denotes specified pressure outflow boundary. Details on implementation are discussed in the SOLA-VOF report.[3] Other types of boundary conditions, if needed, must be supplied by the user; and they will typically be placed at the end of SUBROUTINE BC. An example of this (a tank-drain hole) will be displayed in Sec. III.

C. Initial Conditions

The program specifies the location of obstacles, and the initial locations of fluid, void, and fluid-void interfaces. When appropriate, a fluid-void interface may be calculated as an equilibrium surface; otherwise, its shape must be specified by the user in advance, and must conform to the geometric capabilities of the code, discussed for obstacle generation in Sec. II.A. The machinery for accomplishing all this is a direct 3D extension of our 2D capabilities.[11]

The initial fluid velocities are taken to be constant or zero. It would, in principle, be possible to take the fluid to be initially in nonuniform motion; the appropriate place for the user to insert such changes would be in SUBROUTINE SETUP. However, it is very rare for a dynamic velocity field to be known with sufficient precision to be suitable as an initial condition, except when it is obtained from some other numerical simulation. Thus, it is likely that the user would wish to start an initially dynamic configuration from a tape dump rather than to build the initial conditions into the program's subroutines.

While the pressure field is initially zeroed, this command is overriden to produce the hydrostatic head as the system's initial pressure field. It is important that this more accurate initial pressure field be employed; problems with zero initial pressure frequently have difficulty. As with the initial velocity fields, it will frequently be more useful to pick up the more intricate pressure field of an initially dynamic configuration from a tape dump, rather than building into the program's subroutines.

D. Surface Physics

All the surface physics algorithms must be improved and generalized considerably from their 2D analogs and from the initial algorithms used in our 3D program; it is more difficult to calculate surface physics in 3D than in 2D. In 2D, one surface curvature may be calculated algebraically, and only one curvature need be calculated by finite-difference approximations. Moreover, the algebraic curvature is typically the larger one in the crucial near-axis cells. In 3D, both surface curvatures must be calculated by less accurate finite-difference approximations. We describe the 3D surface physics algorithms. While the radial coordinate is denoted by x elsewhere in the text and throughout the computer program, it is denoted by r in the following Secs. 1 and 2.

1. Orientation of the Surface. The basic orientation of a surface cell is described by the parameter (array) NF through its numerical values 1, 2, 3, 4, 5, or 6. The definition of numerical values for the array is specified in Table I. The differential geometry describing the surface curvatures depends strongly on the numerical value selected for NF. Inspection of **Table I** indicates that NF = 1 or 2 corresponds to a surface defined by $r = R(\theta, z)$, and NF = 3 or 4 to one defined by $\theta = \Phi(r,z)$, and NF = 5 or 6 to one defined by $z = H(r,\theta)$; each surface description leads to a distinct surface differential geometry. Moreover, with only imprecise numerical data available to the program, the value obtained for the local surface pressure, p_s, depends strongly on the numerical value selected for NF; it is important to make an appropriate choice for NF.

Nevertheless, it has not been possible to devise a completely satisfactory algorithm of general applicability for the choice of NF values. We summarize here the algorithms that are currently implemented in the code; they suffice to provide reasonable simulations for the problems to which the code has been applied. The algorithms are activated by the choice of numerical value for the input parameter NFCAL.

TABLE I

DEFINITION OF VALUES FOR NF ARRAY

NF = 0	fluid cell, i.e., contains fluid and has no void cell adjacent to any of its faces
NF = 1	surface cell, fluid surface most nearly parallel to cylinder $r = \text{constant}$, defined by $r = R(\theta, z)$, interpolation neighbor fluid cell is on the left
NF = 2	surface cell, fluid surface most nearly parallel to cylinder $r = \text{constant}$, defined by $r = R(\theta, z)$, interpolation neighbor fluid cell is on the right
NF = 3	surface cell, fluid surface most nearly parallel to plane $\theta = \text{constant}$, defined by $\theta = \Phi(r,z)$, interpolation neighbor fluid cell is to the front
NF = 4	surface cell, fluid surface is most nearly parallel to plane $\theta = \text{constant}$, defined by $\theta = \Phi(r,z)$, interpolation neighbor fluid cell is to the back
NF = 5	surface cell, fluid surface most nearly parallel to plane $z = \text{constant}$, defined by $z = H(r,\theta)$, interpolation neighbor fluid cell is to the bottom
NF = 6	surface cell, fluid surface most nearly parallel to plane $z = \text{constant}$, defined by $z = H(r,\theta)$, interpolation neighbor fluid cell is to the top
NF = 7	isolated cell, contains fluid but all cells adjacent to one of its faces are void
NF = 8	void cell, contains no fluid.

We calculate a preliminary value of NF, which is needed for the earliest uses of NF in the computational cycle. This provisional NF value is determined from the "wettest surface" of the computational cycle by the same numerical ordering as described for the "interpolation neighbor fluid cell" in **Table I**. For example, if the "wettest surface" is to the left of the computational cell the provisional algorithm chooses NF = 1, if to the right it chooses NF = 2, etc. The "wettest surface" of the computational cell is obtained by adding the fractional fluid occupancies, i.e., by summing the F values, in the 9 adjacent cells (in a 3 x 3 array) centered on each one of the computational cells' surfaces. For example, if the computational cell had the cell index (i,j,k), the 9 adjacent cells "to the left" would be labeled by (i-1,j-1,k-1), (i-1,j,k-1), (i-1,j + 1,k-1), ..., (i-1,j + 1,k + 1), corresponding to the interface between (i,j,k) and (i-1,j,k). The "wettest surface" is chosen to be that surface with the largest total fluid occupancy. The provisional set of NF values is updated each cycle, after a new array of F values becomes available. By specifying NFCAL = 1, one may require that this provisional set of NF values also be adopted as the final set of NF values to be used throughout the calculational cycle.

A second set of "slope" NF values is also calculated. The "slope" NF values choose the basic surface mathematical description, and thus the basic surface differential geometry, by picking the surface description that yields a surface normal with minimum angle with respect to the normal of the corresponding coordinate surface, i.e., the surface is "most nearly parallel to" that coordinate surface. This limits the NF values to one of two possible numerical values. The choice of NF is made definite by choosing from the two adjacent cells sharing the preferred coordinate surface, that cell with the largest F value, i.e., containing the most fluid, to be the interpolation neighbor fluid cell. For example, if the "slope" value picks NF = 1 for the computational cell (i,j,k), then (a) defining the surface by $r = R(\theta,z)$ yields a surface normal $\hat{n}$ more clearly parallel to the normal $\hat{r}$ of the coordinate surface r = constant than would follow from one of the other modes of surface description, and (b) $F_{i-1jk} > F_{i+1jk}$, so it is reasonable to use cell (i-1,j,k) as the "base" of the surface, i.e., as the interpolation neighbor fluid cell. By specifying NFCAL = 2, one may require that the "slope" NF values are adopted as the final set of NF values.

Finally, by specifying NFCAL = 3, one may adopt a mixed strategy for choosing between the provisional NF values and the "slope" NF values that has proven to be adequate for the problems to which the program has been applied. With NFCAL = 3, one uses the provisional NF values, except a) if there is an obstacle cell adjacent to the computational cell (i,j,k) in the direction indicated by the "slope" value for NF, i.e., if when the "slope" NF = 1 or 2 either of the cells (i-1,j,k) or (i + 1,j,k) is an obstacle cell the "slope" value of NF is used, or (b) if all the previously computed part of the array of NF has a value of NF in those of the 16 (= 3 x 3 + 3 x 2 + 1) cells "adjoining" the computational cell that are surface cells that agrees with the "slope" NF value for the computational cell, then the "slope" NF value is used. To partially offset the occasional failure of even the NFCAL = 3 algorithm, and for another reason, we introduce a postprocessing algorithm that "clips" anomalous values of p_S, as explained in detail later.

*2. **Surface-Tension.*** Considerable refinement of our original 3D surface tension algorithms was necessary in order to produce algorithms that give accurate values for p_S. We first record here the differential geometric expressions for p_S arising for the three basic types of surface orientation mentioned earlier. We consider the (i,j,k) computational cell to be bounded by

$$r_I \le r \le r_O$$

$$\theta_W \le \theta \le \theta_E$$

$$z_S \le z \le z_N \ ,$$

with mean coordinate values

$$r_m = 1/2\,(r_I + r_O)$$

$$\theta_m = 1/2\,(\theta_W + \theta_E)$$

$$z_m = 1/2\,(z_S + z_N)\ ,$$

and coordinate differences

$$\Delta r = (r_O - r_I)$$

$$\Delta\theta = (\theta_E - \theta_W)$$

$$\Delta z = (z_N - z_S) \ .$$

The surface-tension of the fluid-void interface is denoted by σ. It should be pointed out that the present program actually works with the reduced pressure $\phi = p\rho^{-1}$, even though the reduced pressure is still denoted p. Thus, the σ, as implemented in the program, actually denotes $\sigma\rho^{-1}$. Similarly, densities do not appear explicitly in places, e.g., the calculation of provisional, or TILDE, velocities where one would expect to find the factor $p\rho^{-1}$.

When the fluid-void interface is described locally by a surface of the form $r = R(\theta, z)$ (so that NF = 1 or 2) we obtain a particular geometric form for p_s in a surface computational cell which we display for the case NF = 1, namely

$$p_s(i,j,k) = -\sigma \left\{ \frac{1}{\Delta z} \left[\left(\frac{R_z}{D} \right)_{r=R(\theta_m, Z_N)} - \left(\frac{R_z}{D} \right)_{r=R(\theta_m, z_S)} \right] \right.$$

$$- \frac{1}{2} \left[\left(\frac{1+R_z^2}{rD} \right)_{r=R(\theta_E, z_m)} + \left(\frac{1+R_z^2}{rD} \right)_{r=R(\theta_W, z_m)} \right]$$

$$\left. + \frac{1}{2R(\theta_m, z_m)} \, ctn\left(\frac{\Delta\theta}{2} \right) \left[\left(\frac{R_\theta}{rD} \right)_{r=R(\theta_E, z_m)} - \left(\frac{R_\theta}{rD} \right)_{r=R(\theta_W, z_m)} \right] \right\} , \qquad (6)$$

where

$$D = \left(1 + \frac{1}{r^2} R_\theta^2 + R_Z^2 \right)^{\frac{1}{2}} ,$$

and R_θ, R_z are the gradients of the radial coordinate describing the surface. Similarly, we display the expression for p_s in the NF=3 case of the surface definition $\theta=\Phi(r,z)$ as

$$p_s(i,j,k) = -\sigma \left\{ \frac{1}{\Delta z} \left| \left(\frac{r\Phi_z}{D} \right)_{\substack{r=r_m \\ z=z_N}} \cos\delta_N - \left(\frac{r^2\Phi_r\Phi_z}{D} \right)_{\substack{r=r_m \\ z=z_N}} \sin\delta_N \right. \right.$$

$$\left. - \left(\frac{r\Phi_z}{D} \right)_{\substack{r=r_m \\ z=z_S}} \cos\delta_S + \left(\frac{r^2\Phi_r\Phi_z}{D} \right)_{\substack{r=r_m \\ z=z_S}} \sin\delta_S \right|$$

$$+ \frac{1}{\Delta r} \left[\left(\frac{r\Phi r}{D} \right)_{\substack{r=r_O \\ z=z_m}} \cos\delta_E + \left(\frac{1+r^2\Phi_z^2}{D} \right)_{\substack{r=r_O \\ z=z_m}} \sin\delta_E \right.$$

$$\left. \left. - \left(\frac{r\Phi_r}{D} \right)_{\substack{r=r_I \\ z=z_m}} \cos\delta_W - \left(\frac{1+r^2\Phi_z^2}{D} \right)_{\substack{r=r_I \\ z=z_m}} \sin\delta_W \right| \right\} , \qquad (7)$$

where

$$D = \left(1 + r^2 \left[\Phi_r^2 + \Phi_z^2 \right] \right)^{\frac{1}{2}} ,$$

and, if $\hat{r}(\theta)$, $\hat{\theta}(\theta)$ denote the radial and azimuthal unit vectors evaluated at angle θ, then at the north face of the computational cell, i.e., at $z = z_N$,

$$\hat{\theta}(\theta_N) \cdot \hat{\theta}(\theta_m) = \cos\delta_N$$

$$r(\theta_N) \cdot \hat{\theta}(\theta_m) = \sin\delta_N ,$$

with

$$\theta_N = \Phi(r_m, z_N) \ ,$$

and similar expressions hold for the other angles δ evaluated at the S, E, and W faces of the computational cell. Finally, we display p_s in the NF = 5 case of the surface definition z = H(r,θ) as

$$p_s(i,j,k) = -\frac{\sigma}{r_m}\left\{\frac{1}{\Delta r}\left[\left(\frac{rH_r}{D}\right)_{\substack{r=r_O\\ \theta=\theta_m}} - \left(\frac{rH_r}{D}\right)_{\substack{r=r_I\\ \theta=\theta_m}}\right] + \frac{1}{\Delta\theta}\left[\left(\frac{H_\theta}{rD}\right)_{\substack{r=r_m\\ \theta=\theta_E}} - \left(\frac{H_\theta}{rD}\right)_{\substack{r=r_m\\ \theta=\theta_W}}\right]\right\} \ , \tag{8}$$

where

$$D = \left(1 + H_r^2 + \frac{1}{r^2} H_\theta^2\right)^{\frac{1}{2}} \ .$$

In Eqs. (7) and (8), Φ_r, Φ_z, H_r, and H_θ denote the gradients of the coordinate defining the surface.

In NASA-VOF2D and in earlier versions of our 3D program, the height functions defining the surface locally are computed from three cell summation formulas which convert the information available to the program, namely values of F in an array of cells near the computational cell, to a reconstruction of the local fluid-void interface. When this is done using reasonable zoning for the computational mesh and Eqs. (6)-(8) for p_s it is fairly common to obtain rather poor values for p_s. The problem is that regions in which the surface is nearly parallel to a particular height function are very poorly represented by the three-cell formula. Thus, in the interior of the computational mesh, we have defined the height functions by a five-cell summation formula which evolves smoothly (via a four-point summation formula) to the old three-cell formula still used at the edge of the mesh. For example, in the mesh interior we compute H for cell (i,j,k) from

$$H(i,j,k) = \sum_{k'=k-2}^{k+2} \left(1.0 + AC_{i,\,j,\,k'} * (F(i,j,k') - 1.0)\right) * \Delta Z_{k'} * FOF_{k'} \qquad (9)$$

where $FOF_{k'}$ is a cut-off factor which excludes a cell from the sum when appropriate. While Eq. (9) and its analogs substantially solve the problem of inaccurate height functions due to geometric effects, it is still necessary to be sure that the zoning is appropriate to the problem at hand; even Eq. (9) can yield inaccurate p_S values locally if the zoning is too coarse.

In 2D and in the earlier 3D algorithms, gradients of height functions were evaluated from interpolation formulae which do not maintain accuracy when variable zoning is used. It is important to overcome this defect in 3D because: a) the 3D formulas are more sensitive to errors in gradients, b) variable zoning is essential for many 3D problems to reduce problem running time, and c) we have seen cases where the older 3D formulation led to very large errors in p_S even for a uniform mesh. For our present algorithms we consider 9 height functions defined in a 3 × 3 array of cells centered on the computational cell, and define gradients of the height function evaluated at the center of a face of the computational cell by either 6 or 7 point formulas which use the information contained in the 9 height functions to:

(1) eliminate all constant, linear, and quadratic coefficients of the Taylor series expansions of the functions, except for the coefficient of the gradient being evaluated (which is set equal to unity)

(2) balance the two principal cubic errors of the Taylor series expansions.

The result of these operations is either a 6-point formula (if the gradient is parallel to the face where the gradient is being evaluated) or a 7-point formula (if the gradient is perpendicular to that face). The old 3D gradient formulas were all 6-point formulas, but they were different, and less satisfactory, formulas. The Taylor series being used in the preceding operations are all centered on the point where the gradient is being evaluated. We illustrate these processes by displaying two gradients of the height function $z = H(r,\theta)$, H_{rE} and H_{rN}, evaluated at the center of the east and north faces of the computational cell (i,j,k) respectively. The first gradient is evaluated by a 7-point sum

$$H_{rE} = \sum_{ii=1}^{3} \sum_{jj=1}^{3} c_{jj\,ii}\, H(i', j') \ , \qquad (10)$$

where

$$i' = i - 2 + ii$$

and

$$j' = j - 2 + jj \ .$$

As stated, the computational cell is in the center of the 3×3 array of cells used in the sum. Using the definitions

$$\delta r_O = 1/2\ \Delta r_{i+1}$$

$$\delta r = 1/2\ \Delta r_i$$

$$\delta r_I = 1/2\ \Delta r_{i-1}$$

$$\delta\theta_E = 1/2\ \Delta\theta_{j+1}$$

$$\delta\theta = 1/2\ \Delta\theta_j$$

$$\delta\theta_W = 1/2\ \Delta\theta_{j-1}$$

$$\Delta r_+ = \delta r_O + \delta r$$

$$\Delta r_- = \delta r_I + \delta r$$

$$\Delta\theta_+ = \delta\theta_E + \delta\theta$$

$$\Delta\theta_- = \delta\theta_W + \delta\theta \ , \tag{10a}$$

we express the 7 nonzero $c_{jj\,ii}$ coefficients as

$$c_{33} = \frac{1}{3}\ \frac{1}{\Delta r_+^2\,\Delta r_-(\Delta r_+ + \Delta r_-)}\ \frac{\Delta\theta_-}{(\Delta\theta_+ + \Delta\theta_-)}\left((\delta r_O - \delta r)(\Delta r_- + \delta r)^2 + (\Delta r_- + 2\delta r)\delta r_O^2\right)$$

$$c_{23} = -\left(1.0 + \frac{\Delta\theta_+}{\Delta\theta_-}\right)c_{33} + \frac{(\Delta r_- + 2\delta r)}{\Delta r_+(\Delta r_+ + \Delta r_-)}$$

$$c_{13} = \frac{\Delta\theta_+}{\Delta\theta_-}\,c_{33}$$

$$c_{32} = -c_{33}$$

$$c_{22} = \left(1.0 + \frac{\Delta\theta_+}{\Delta\theta_-}\right)c_{33} + \frac{(\delta r_O - \delta r - \Delta r_-)}{\Delta r_+\,\Delta r_-}$$

$$c_{12} = -\frac{\Delta\theta_+}{\Delta\theta_-}\,c_{33}$$

$$c_{21} = -\frac{(\delta r_O - \delta r)}{\Delta r_-(\Delta r_+ + \Delta r_-)} \ . \tag{10b}$$

The second gradient is evaluated by the 6-point sum

$$H_{rN} = \sum_{ii=1}^{3}\ \sum_{jj=1}^{3} a_{jj\,ii}\,H(i', j') \ . \tag{11}$$

In this case, the 6 nonzero $a_{jj\,ii}$ coefficients are given by

$$a_{33} = \frac{1}{3\Delta r_+}\left(1.0 + 3.0\,\frac{\delta\theta}{\Delta\theta_+}\,\frac{\Delta r_-}{(\Delta r_+ + \Delta r_-)}\right)$$

$$a_{32} = -\left(1.0 + \frac{\Delta r_+}{\Delta r_-}\right)a_{33} + \frac{1}{\Delta r_-}\,\frac{\delta\theta}{\Delta\theta_+}$$

$$a_{31} = \frac{\Delta r_+}{\Delta r_-}\,a_{33} - \frac{1}{\Delta r_-}\,\frac{\delta\theta}{\Delta\theta_+}$$

$$a_{23} = -\,a_{33} + \frac{\Delta r_-}{\Delta r_+(\Delta r_+ + \Delta r_-)}$$

$$a_{22} = \left(1.0 + \frac{\Delta r_+}{\Delta r_-}\right)a_{33} - \frac{1}{\Delta r_-}\,\frac{\delta\theta}{\Delta\theta_+} + \frac{(\Delta r_+ - \Delta r_-)}{\Delta r_+\Delta r_-}$$

$$a_{21} = -\,\frac{\Delta r_+}{\Delta r_-}\,a_{33} + \frac{1}{\Delta r_-}\,\frac{\delta\theta}{\Delta\theta_+} - \frac{\Delta r_+}{\Delta r_-(\Delta r_+ + \Delta r_-)}\,. \tag{11a}$$

Corresponding 6 and 7 point formulas hold for the other 22 gradients of the height functions. The various gradient formulas must be altered somewhat for the axial cylinder of cells ($i = 2$ computational cells). In this case, the singularity of the coordinate system at $r = 0$ must be respected and the 3×3 array of cells must be offset from the computational cell by one integer of the i index (to include $i = 2$, 3, and 4 cells) for the cases of NF = 3, 4, 5, and 6. The offset 3×3 array of cells still contains enough information to determine accurate values of the gradient, but the 6 and 7 point formulas must be altered somewhat to allow for the fact that the computational cell is now at the edge, rather than at the center, of the 3×3 array.

The expressions in which the radial coordinate of the surface must be evaluated with high precision to obtain accurate values for p_s are indicated in Eq. (6). When necessary, e.g., in the evaluation of $R(\theta_E, z_m)$, 3 point interpolation formulas are used to obtain sufficiently accurate r values from the 3×3 array of R values available locally at the computational cell.

The p_s algorithm concludes by a set of postprocessing procedures which 'clip' anomalous values of p_s. Specifically, the procedures compute an average value of the surface pressure, p_{sav}, in the surface cells adjacent to, i.e., within a 3×3 array of cells centered on, the computational cell and a) insist that the algebraic sign of p_s agree with the algebraic sign of p_{sav} (this eliminates any local "pimples" in the surface), and b) insist that $|p_s|$ differ by no more than 50% from $|p_{sav}|$. For simplicity, the calculation proceeds from $i = 3, 4, \ldots$ and then comes back with a slightly different algorithm for $i = 2$, reflecting the topological difference of the "inner edge" of the computing mesh. This procedure allows for: (1) the occasional malfunctioning of the decision procedure leading to the choice of NF values, and (2) the possibility that an inappropriate zoning of the problem produces locally some inaccurate values of p_s. These procedures limit the pressure gradient acceleration of fluid in the surface cells, and have proven to be an essential part of an algorithm which prevents an unphysical growth of such velocities, which, if unchecked, will eventually terminate the calculation in the F convection routine.

3. Wall-Adhesion. The present version of the program treats wall-adhesion accurately only for straight-walled containers. Since curved-container walls may give rise to substantial errors in wall-adhesion forces, we introduce an input parameter NOWALL, which can force the surface-tension option to be chosen. When NOWALL = 0 the program calculates surface-tension or wall-adhesion as directed by the general surface-physics algorithm; but when NOWALL = 1 the surface-physics algorithm will always select the surface-tension option.

Considerable effort was spent in obtaining a realistic wall-adhesion algorithm for curved boundaries in 2D problems. (It is a standard feature of NASA-VOF2D). This turned out to be of crucial importance to calculating thin liquid film features, such as occur on the tank wall during the initial phases of a fuel reorientation problem or the final phases of a tank-draining problem. In 2D the task was accomplished by a) defining a cell index array NW, redefined each computational cycle, which determines whether the wall-adhesion option or the surface-tension option is to be selected, and b) defining some angles, defined once and for all by the container geometry, which determine the direction in which any wall-adhesion force is to point. For 2D the wall-adhesion calculation simplifies because any wall-adhesion force points either up or down the container wall. (This neglects the, usually very small, contact angle of the fluid.) In contrast, a successful 3D wall-adhesion algorithm must introduce a further angle, redefined each computational cycle, which determines from the local surface geometry what direction along the container surface any wall-adhesion force is to point.

We emphasize that the present program, without a universally accurate calculation of wall-adhesion, still has a very significant range of applicability which permits a wide range of design questions to be meaningfully addressed. For example, questions concerning the bulk movement of fluid in a given geometry seldom involve thin liquid films and thus may be addressed without having the wall interaction of the fluid and its containers specified fully correctly.

E. Incrementing the Variables

Several sets of quantities must be incremented to advance the calculation through one time step. Some procedures are applied to all cells open to flow, others to fluid cells only, with alternative procedures for surface cells. We discuss the main stages of this process below. Velocities and pressures, having been determined at cycle n, i.e., at $t = n\delta t$, are to be determined at cycle n + 1.

1. Provisional (TILDE) Velocities. As an aid to incrementing the pressure and velocity variables, we calculate provisional, or TILDE, velocities, i.e., $\tilde{u}_{i+\frac{1}{2},j,k}$, etc., including the gravitational acceleration, and we use the cycle n variables to provide provisional values for the specific pressure acceleration SPGX, the advection ADVX, and the viscous acceleration VISX. Thus, the TILDE velocities are determined by equations of the form

$$u_{i+\frac{1}{2},j,,k} = u_{i+\frac{1}{2},j,k} + \delta t\left[g_x + SPGX_{i+\frac{1}{2},j,k} + ADVX_{i+\frac{1}{2},j,k} + VISX_{i+\frac{1}{2},j,k}\right] , \qquad (12)$$

with similar expressions for the y and z velocity components. The expressions used for the x, y, and z components of SPG, ADV, and VIS are the direct 3D analogs of the 2D algorithms adopted in the NASA-VOF2D program.[11] In particular, the components of the advection ADV are evaluated as a linear combination of centered differencing and donor-cell differencing, with the input parameter α (ALPHA) specifying the donor-cell fraction. The operation of the algorithms and useful rules of thumb for specifying α are discussed in Reference 11. The present program actually works with the reduced pressure $\phi = p\rho^{-1}$, although it is still denoted p. Thus, densities do not appear explicitly in places, i.e., the TILDE calculation, where $\rho\rho^{-1}$ occurs in the 2D program. Similarly, in this program σ denotes $\sigma\rho^{-1}$. TILDE velocities are obtained for all computational cells.

2. SOR Option for Pressures and Velocities. TILDE velocities must be further corrected in order to obtain cycle n + 1 velocities that satisfy the continuity equation. This is done by using one of two options to solve equations for the pressure increments δp^{n+1} in all fluid cells (excluding surface cells, isolated cells, and void cells) which relate the pressures at successive cycles, i.e.,

$$p^{n+1} = p^n + \delta p^{n+1} \; .$$

In the successive over-relaxation (SOR) option, a straightforward generalization of the 2D SOR option,[11] δp is calculated from

$$\delta p = -\beta S \; , \tag{13}$$

where

$$\beta = \frac{1}{\dfrac{\partial S}{\partial p}} \; ,$$

and S is the continuity equation written in finite-difference form. An algorithm is used to obtain updated velocities from the δp's, and then the most updated values for velocity and pressure are used in re-evaluating S. The process is continued iteratively until finally

$$|S| < EPSI \; , \tag{14}$$

where EPSI is an input parameter to the program. Usually convergence is accelerated by multiplying the δp of Eq. (13) by an over-relaxation parameter ω, i.e., by an input parameter OMG. The 2D version of the process, with guidance for the choice of input parameters, is discussed in some detail in Reference 11.

3. CR Option for Pressures and Velocities. In the conjugate residual (CR) option for solving for δp, which also applies to fluid cells only, the pressure increments are assembled into a vector **P**, the cycle n + 1 velocities are expressed in terms of the TILDE velocities and of **P**. The continuity equation at cycle n + 1 is then written as the vector equation

$$A\mathbf{P} = h \ , \tag{15}$$

where the matrix A is symmetric and the vector **h** contains the TILDE velocities and boundary conditions. The conjugate residual algorithm is then used to iteratively solve this equation for the pressure increments. The CR algorithm is terminated when the convergence criterion based on EPSI is satisfied. When **P** has been obtained, the cycle n + 1 pressure and velocity fields are constructed from their defining relations in terms of **P**. The 3D CR algorithm is the same as the 2D CR algorithm, which is described in more detail in Reference 11. At present we prefer the CR option to the SOR option; it saves computer time in running a problem and, in general, it is a more computationally robust algorithm.

4. Pressures and Velocities for Surface Cells. Surface cell pressures are included indirectly in both pressure iteration options. For the CR option, this must be done carefully to maintain the symmetry of the matrix A. The pressure of a surface cell is determined by an interpolation procedure that involves the surface pressure, p_s, and the pressure, p_n, of the interpolation neighbor fluid cell, which, in turn, is determined by the orientation index, NF, of the surface cell. For example, if the (i,j,k) surface cell has NF = 1, then the interpolation neighbor fluid cell is (i-1,j,k). The program uses surface cell geometry and F value to evaluate an interpolation parameter η (implemented in the program through the PETA array), and the pressure in the surface cell is then evaluated from

$$p = (1 - \eta)p_n + \eta p_s \ . \tag{16}$$

The evaluation of the PETA array is the direct extension of the 2D procedure described in Reference 11.

The velocities of surface cells are incremented beyond the TILDE values by essentially the 2D procedure;[11] namely, the gradients of the new pressure increments are used to increment the velocities, with due attention to limiting gradients not likely to be reliable (because they would, without modification, involve pressures not computed from the pressure iteration mesh), and the normal velocity on the cell face directly opposite to the interpolation neighbor fluid cell is reset afterward by imposing the continuity equation on surface cell velocities. However, this procedure still permits the slow growth of velocities by some nonphysical pressure gradients, typically resulting eventually in large nonphysical azimuthal velocities in cells near the coordinate axis. Thus the velocities in surface cells and void cells are further limited to be not greater than corresponding velocities in neighboring fluid cells.

5. Incrementing the F Values. The algorithms used to advance the F values throughout the mesh from cycle n to cycle n + 1 are the direct 3D extension of the 2D algorithms discussed in detail in Reference 11 and in the earlier literature. Here we merely reiterate the principal considerations involved in carrying out the calculation.

The convection algorithms use a form of donor-acceptor differencing which is designed to (1) preserve the sharp definition of free boundaries, which we denote here as fluid interfaces; (2) avoid negative diffusion truncation errors; and (3) not flux more fluid, or more void, across a computing cell interface than the cell losing the flux contains. The algorithms also contain features designed to suppress the appearance of spurious small wisps of fluid.

At each cell boundary the two adjacent cells are distinguished, one as the donor cell and the other as the acceptor cell, according to the algebraic sign of the normal velocity at the cell boundary. The continuity equation for F is written in finite difference form, for which an F value is specified at each boundary of the computational cell. The primary choice of F for each cell boundary is given by a fairly subtle algorithm, simple to state but based on a great deal of fluid dynamics experience, which picks either the donor or the acceptor value of F, F_D or F_A. This primary value of F is then limited to require that neither too much flux nor too much void is fluxed from the donor to the acceptor cell. This limiting feature also contains a provision which helps suppress the nonphysical convection of small wisps of fluid throughout the computing mesh.

When the "defoamer" control parameter IDIV is on, the value of F is further corrected by a term proportional to the divergence of the actual velocity field (the S value referred to previously) which is not rigorously zero. However, when the "defoamer" control parameter IDEFM is on (correcting the S value used in the pressure iteration subroutine), the IDIV option is skipped. Both "defoamer" options work much as they do in 2D.[11] However, the NPACK option is not included in NASA-VOF3D.

As in the 2D program,[11] large and small values of F^{n+1} are rounded so that F may be used as a boundary cell flag. F is reset to zero (an empty cell) when its computed value is less than ε_f (EMF in the program) and neighboring full cells become surface cells by having their F value reduced by 1.1 ε_f. F is reset to unity (a full cell) when its computed value is greater than 1 - ε_f. A typical value for ε_f is 10^{-6}. The rounding error in F values is included in the accumulated volume error. The accumulated volume error is typically very small during the course of a calculation.

As described in detail in Reference 11, the indexing of the F values is not strictly accurate, though it becomes accurate for a problem with constant time step, δt, which is started from rest. The program ignores the change in F occurring in the first half time step and the graphics display F values at the nth time step t_n instead of at the more accurate $t_n + \frac{1}{2}\delta t_n$.

6. Time Step Limitations. Limitations on the time step δt are in very close parallel to the limitations in 2D.[11] Specifically, there is a convective limit (the Courant condition), a diffusive limit, and a capillary limit.

The absolute convective limit may be stated

$$\delta t < min\left\{\frac{\delta x_i}{|u_{i,j,k}|}, \frac{\delta y_j}{|v_{i,j,k}|}, \frac{\delta z_k}{|w_{i,j,k}|}\right\}, \tag{17}$$

where the minimum is taken over all cells of the computing mesh, but it is usual to require that δt be no more than a small fraction, e.g., 0.25, of the minimum. The parameter α describing the proportion of donor cell differencing must be at least as large as, roughly, 1.2 to 1.5 times the value adopted for the fraction described above, to preserve the stability of the convective differencing.

The avoidance of diffusive instabilities imposes a further limit

$$\nu\delta t < 0.5\left(\delta x_i^{-2} + \delta y_j^{-2} + \delta z_k^{-2}\right)^{-1}. \tag{18}$$

Unlike the convective limit of Eq. (17), no harm results from imposing a δt limit very close to the critical value specified by Eq. (18), and this is usually done.

The limitation on the time step from capillary forces stems from the requirement that a capillary wave not travel more than one cell width in one time step. This is enforced in the program by the rough requirement

$$8\sigma\,\delta t^2 < p\,\delta x_m^3 \, , \tag{19}$$

where δx_m is the minimum average cell width in any direction anywhere in the mesh. In practice, Eq. (19) is usually dominated by the average azimuthal dimension of the cell nearest the axis (i = 2 cells).

All the above time step limitations are imposed automatically by setting the input parameter AUTOT = 1.0. The choice AUTOT = 0.0 permits a constant δt to be used. The input parameter ALPHA (needed, as described above, for numerical stability) must be set separately. The operation of the time step limitation procedure is described more fully in Reference 11.

III. PROGRAM DESCRIPTION

The NASA-VOF3D program, like its NASA-VOF2D predecessor, is highly structured so that individual components may be easily modified to fit specific problems or to accept subsequent code upgrades. This approach allows greater efficiency in operation, simplifies problem setup, and facilitates code modification. While NASA-VOF3D can run many problems without alteration, a specific application will in general require program alterations for special inflow or outflow ports, complicated geometries, or unusual initial conditions. Seldom does anything have to be deleted; rather the modifications are usually additions to the existing structure. An example would be the modification of the wall adhesion model, which would be most naturally included primarily in SUBROUTINE SURF10N, with some indices and geometric factors specified elsewhere, to treat cases in which the wall adhesion force significantly affects important features of a simulation.

A. Program Architecture

We present here a discussion of the execution of a calculation by tracing the operation of the program flow chart and listing the subroutines, with a short specification of their functions, for NASA-VOF3D. Note that while NASA-VOF2D is the direct 2D predecessor of the present 3D program NASA-VOF3D, the historical origins of their respective program architectures are somewhat different; NASA-VOF2D evolved most directly from SOLA-VOF, whereas the present NASA-VOF3D is most directly related to SOLA-3D, which, however, is undocumented.

*1. **Description of the Flow Chart***. Figure 1 displays the program's logical sequence. On the left is illustrated the flow through the main program, ROUTINE CONTROL; on the right is an expanded version of the flow through the main subprogram, where problem execution takes place, SUBROUTINE SOLA. In the SOLA flow chart it has been assumed that the CR option has been selected for the pressure iteration.

In SUBROUTINE RINPUT the problem input data are read, and default values are specified for data that might not be read in. In addition, the film package is prepared for execution by calling SUBROUTINE GRPHCFT and the associated ROUTINES GPLOT, GRPHLUN, and SETFLSH and the input data read in from NAMELIST XPUT are written on tapes 9 and 12. The preparation of the film package is specific to the Cray systems at Los Alamos National Laboratory; for other systems the user must substitute his own comparable routines and/or eliminate these calling statements. When TD>0 all subsequent set-up routines are skipped, the problem's initial conditions are read from the problem restart tape and control is transferred to SUBROUTINE SOLA, where problem execution begins. This problem restart capability is the natural way to continue a problem run previously; one needs only to have arranged a convenient tape dump from the previous run. It will also usually represent the simplest way to initiate a problem with "complex" initial conditions, i.e., with non-trivial initial velocity and/or pressure fields. (Only occasionally will such "complex" initial conditions be known with sufficient precision to be reasonably coded into the problem setup.)

When the problem is not to be restarted from a tape dump, CONTROL begins to read problem set-up data and then execute the problem set-up. First, SUBROUTINE MESHSET reads data from NAMELIST MESHGN, sets up the computational mesh, calculates areas of partial flow cells, prints obstacle data and calculates data for use in SUBROUTINE BC. The SUBROUTINE RGRAPHIC reads data from NAMELIST GRAFIC and prepares data, formats and captions for the graphs. Next, SUBROUTINE RCONTUR reads in data from NAMELIST CONTUR, although default values are given in RCONTUR, and sets up formats and titles for contour plots of cell-centered quantities. The graphic systems are again specific to the Cray systems and at Los Alamos National Laboratory, so the user must make appropriate adjustments for other systems.

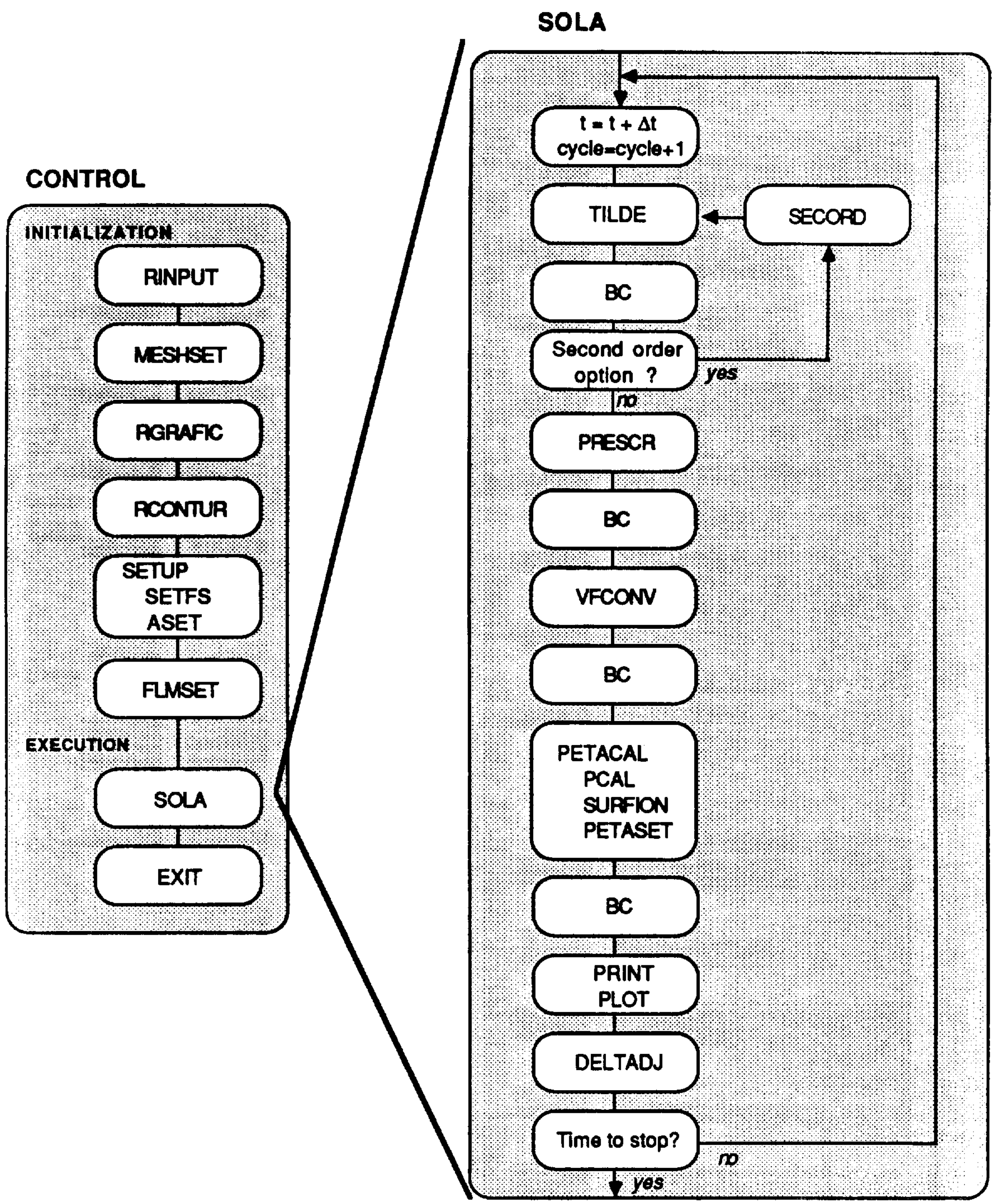

Fig. 1. Flow chart for NASA-VOF3D

In SUBROUTINE SETUP many of the problem initial values are specified; it is here that altered or more general initial conditions would be most appropriately inserted by the user. First, a number of constant parameters are specified and a number of arrays are initialized to zero. Then the initial fluid volume fraction F is calculated for each cell. When ICSURF = 0 (axisymmetric void volume) and IEQUIB >0 the fluid surface is calculated by SUBROUTINE EQUIB; otherwise it is specified by NAMELIST FLUIDGN and calculated in SUBROUTINE SETFS, which is called to complete the specification of initial values of F, unless the initial surface is horizontal or is an equilibrium surface. In SETFS the fluid surface is specified by means of general quadratic functions. The cases of axisymmetric and nonaxisymmetric volumes are treated separately. Cell F values are specified by calculating the fraction of the cell volume that is "inside" the surface.

SUBROUTINE SETUP continues by calculating the hydrostatic pressure as an initial value for the pressure field, calculating partial flow areas for partial flow cells (this is done by calling SUBROUTINE ASET, which makes use of the obstacle information read by NAMELIST MESHGN, which has already been called by MESHSET and thus is available information), calculating the BETA array for the mesh (done by calling SUBROUTINE BETACAL), calculating the viscous limit to the time step and specifying the initial values of the velocity fields to be zero. The obstacle information (used in ASET) and the fluid surface information (used in SETFS) are set by quadratic functions that are completely analogous to the quadratic functions used in the 2D program and discussed in detail in Reference 11.

Finally, the set-up is completed by calling SUBROUTINE FLMSET, which calculates the mesh boundaries for use in various graphics routines. Then control is passed to SUBROUTINE SOLA where the problem is executed and results are printed, plotted and dumped on tape.

SUBROUTINE SOLA begins by setting some timing information. It then calls SUBROUTINE BC to set the initial boundary conditions for the problem. Then SOLA begins its principal business of advancing the problem through one time step.

First the time variable t is incremented to $t + \delta t$ and the cycle indicator CYCLE is incremented to CYCLE + 1. Then the provisional (TILDE) velocities are obtained by calling SUBROUTINE TILDE. The continuing boundary conditions are obtained by calling SUBROUTINE BC, which, in turn, calls SUBROUTINE BCFS to set velocities on boundaries of surface cells and their adjacent void cells. If the second-order accurate option is specified, SUBROUTINE SECORD is called and the calculation recycles through TILDE and BC. The calculation then proceeds to advance the pressures, and further advance the velocities, in the fluid cells by calling SUBROUTINE PRESCR if the CR option is specified or calling SUBROUTINE PRESSIT if the SOR option is specified. Boundary conditions are reimposed, but principally the velocities on the boundaries of the surface cells and their adjacent void cells are readjusted by recalling SUBROUTINE BC.

SOLA is then ready to advance the F values by one time step, which it does by calling SUBROUTINE VFCONV, which also computes the total change in fluid volume (by calling SUBROUTINE VCHGCAL). Next, the relevant velocities are required to accommodate the new information on F values by recalling BC. SOLA then treats the surface physics by calling SUBROUTINE PETACAL, which initializes some arrays, calls SUBROUTINE PCAL to determine provisional values of the NF array, and determines final NF values and values of the surface pressure, p_s, in the surface cells by calling SUBROUTINE SURF10N. Continuing, it calculates values of the η (PETA) array for surface cells, sets η in the neighboring interpolation cells (for reasons explained in Reference 11) by calling SUBROUTINE PETASET, and resets pressures in obstacle cells, surface cells, void cells, and isolated fluid cells by calling SUBROUTINE PRESCK. The major surface physics, described in Sec. II.D., is carried out in SUBROUTINE SURF10N, which is the most appropriate place to add the bulk of a wall adhesion algorithm when the program is upgraded in that fashion.

The surface physics now having been carried forward for an additional time cycle, a new set of NF values, and hence a new geometry of surface cells, is now available. SOLA thus calls BC to update the velocities to this new geometric information. SOLA is then ready to print and/or plot the field variables (by calling various graphics routines), set the advanced time field arrays into the time n arrays, adjust the time step δt (by calling SUBROUTINE DELTADJ), and test whether the problem has been completed. If the problem has been completed, SOLA will print, plot, and dump information on tape. CONTROL will then call EXIT and END. If the problem has not been completed, control will remain in SOLA which then begins to advance the problem through another time step δt.

The preceding abbreviated description omits numerous calls to subroutines which accomplish very limited tasks in the calculation; their full inclusion would not only make the calculational path seem much more intricate, and thus harder to understand, but also raise numerous distracting questions (often not really of interest to the user) as to why the particular calculational path, rather than some other path, was chosen. For example, the calculation frequently needs the ordinal index IJK and a number of "neighboring" indexes formed from the cell indices (i,j,k); thus there are numerous calls to SUBROUTINE CALCIJK which provides this information. But sometimes only IJK is needed, which is provided by calls to IJKONLY, an entry point into CALCIJK. Moreover, while it is frequently advantageous to operate with problem vectors and matrices having a "loose" structure with numerous zeros, which is provided by IJK, for certain matrix operations one wants a "fight" structure in which no vectors contain elements which are identically zero. This is provided by calls to IJKAJCT, another entry point into CALCIJK, and use of the ordinal index LIJK.

We have attempted to provide a reasonable explanatory path for the program by enumerating the main features of the flow chart, identifying the main players (the subroutines, and the variables and arrays in common) of the program, specifying the problem input, and discussing the user options to influence the way in which the calculation is carried out. But there may well be <u>no</u> mode of explanation that successfully avoids the potential pitfalls of either explaining to the user many details which are of no interest to him or not providing a clear explanation of what the program is doing and, for the few that are interested, a rationale for why the program does what it does. We conclude this section by noting that the user would be well advised to be alert for systems calls that are specific to the Cray systems at Los Alamos National Laboratory; these he may need to do something about.

2. ***SUBROUTINES and Their Functions***. We specify here the SUBROUTINES of NASA-VOF3D, and define their functions briefly.

ASET (interior obstacle generator) Generates the problem variables, parameters, and flags to implement the partial flow volumes. Sets BETA to − 1.0 in fictitious cells and in obstacle cells.

BC (boundary conditions) Sets the appropriate fluid variables at the mesh boundaries and interior obstacles to produce specified boundary conditions; calls BCFS to set velocities on free surface and adjacent void cell boundaries in the absence of real pressure gradients. Proper place to insert special boundary conditions not built into code.

BCFS (boundary conditions) Sets velocities on free surfaces and adjacent void cell boundaries in the absence of real pressure gradients (called by BC).

BETACAL (β for mesh interior) Calculates BETA for nonobstacle cells. Includes effects of over-relaxation parameter ω.

CALCIJK	(cell indices) Evaluates IJK and other cell indices for cell i,j,k.
CNTR	(graphics) Plots contours of cell-centered quantities. Set-up in SUBROUTINE RCONTUR. User must input array to be plotted if other than pressure.
CONTRJB	(graphics) Sets scales of quantities to be displayed.
CONTROL	(controlling main program) Exercises logical control over the sequence of calculations. Calls SUBROUTINE SOLA for main calculation.
DELTADJ	(time step adjustment) Computes maximum allowable δt for stability, adjusts δt considering stability limit and number of iterations required for pressure convergence, recomputes relaxation factors (BETA) used with SOR method.
DGAP	(graphics) Defines graph area for perspective plots.
DRAW	(graphics) Plots velocity vectors, free surfaces, contours of F and P, and perspectives of surface on film.
DRAWQ	(graphics) Displays composite graphs of two different J-plane results in one graph, converting 3D results to a 2D display. This is the first of four supplementary subroutines needed for this purpose.
DRF	(graphics) Draws frame around computing mesh in plane view. Calls on SUBROUTINE DRFCYL for cylindrical coordinates.
DRFCYL	(graphics) Provides k = constant frame for cylindrical coordinates to SUBROUTINE DRF.
DRFP	(graphics) Draws frame around computing mesh in perspective.
DROBSP	(graphics) Draws interior obstacles in perspective.
DRVEC	(graphics) Provides a system-dependent call opportunity to draw a line segment in any coordinate plane.
DRWOBS	(graphics) Draws interior obstacles in plane view.
DRWOBSQ	(graphics) Draws the obstacles used in the computational mesh for the composite displays of DRAWQ.
DRWVEC	(graphics) Draws a vector for the composite graphics displays of DRAWQ.
EQUIB	(equilibrium surface) Solves two-point boundary value problem to obtain (cylindrically symmetric) equilibrium shape of initial free surface and gives it to SUBROUTINE SETUP.
FLMSET	(graphics) Calculates mesh boundaries for plotting.

FRAMEQ	(graphics) Draws a frame around the composite graphics displays of DRAWQ.
LPRT	(graphics) Prints data on film.
MESHSET	(mesh generator) Generates computing mesh, the fractional volumes and areas open to flow, and geometric information related to the mesh. Calls on SUBROUTINES ASET, MESHX, MESHY, and MESHZ for specific parts of the calculation.
MESHX	(x-coordinate generator) Computes x-coordinate values and their reciprocals. Called by SUBROUTINE MESHSET.
MESHY	(y-coordinate generator) Computes y-coordinate values and their reciprocals. Called by SUBROUTINE MESHSET.
MESHZ	(z-coordinate generator) Computes z-coordinate values and their reciprocals. Called by SUBROUTINE MESHSET.
PCAL	(surface physics) Calculates preliminary values of NF array and supplies them to SUBROUTINE PETACAL.
PCNV	(graphics) Produces coordinates suitable for perspective views.
PDFCALC	(partial derivatives) Calculates partial derivatives for the F function.
PETACAL	(surface physics) Determines preliminary values of surface orientation index NF and of interpolation parameter η (PETA). Calls on SUBROUTINE PCAL for preliminary NF and PETASET for PETA in neighboring interpolation cells.
PETASET	(surface physics) Calculates PETA in neighboring interpolation cells and supplies them to SUBROUTINE PETACAL.
PLTPT	(graphics) Plots a point. It is called by DRAWQ.
PRESCK	(surface physics) Resets pressure in surface cells, obstacle cells, void cells and isolated fluid cells. It is called by PETACAL.
PRESCR	(conjugate residual) Increments pressures and velocities by conjugate residual technique, as in 2D version of code.
PRESSIT	(successive-over-relaxation) Increments pressures and velocities by successive-over-relaxation technique.
RCONTUR	(graphics) Prepares formats for contour plots and supplies them to SUBROUTINE CNTR.
RDTAPE	(input) Reads input tape.
RGRAFIC	(graphics) Reads and writes data for formats of computer-drawn graphics.

RINPUT	(input) Reads problem input data.
SECORD	(velocity adjustment) Adjusts TILDE velocities to second-order accuracy for SUBROUTINE SOLA.
SETACC	(gravitational acceleration) Generates r and θ components (from x and y components) of a general gravitational field.
SETFS	(initial conditions) Generates initial configuration of fluid and gives it to SUBROUTINE SETUP.
SETUP	(initial conditions) Calculates problem parameters, problem geometry, initial arrays, fluid configuration and velocities. Calls SUBROUTINES ASET, BETACAL, EQUIB, and SETFS for specific phases of the calculation.
SOLA	(main calculational program) Increments pressures, velocities, and F values by one time step δt. Calls numerous SUBROUTINES for specific phases of calculation.
SURCNTR	(graphics) Draws the free surface on particular plane plots of velocity vectors.
SURFPLT	(graphics) Draws free surface in perspective.
SURF10N	(surface physics) Gives final NF values and calculates surface tension effects. Appropriate place to insert most of the needed wall adhesion modifications; present program uses very early version of wall adhesion algorithm which is known to be inadequate.
TILDE	(velocity increment) Calculates explicitly a set of approximate velocity increments.
VELV	(graphics) Draws velocity vectors in plane or in perspective plots.
VCHGCAL	(void volume) Computes volume of disjoint void (F = 0.0) regions.
VFCONV	(F increment) Computes the advective fluxes of F from the newly determined velocity field and updates the F array.
WRTAPE	(output) Writes tapes of output.
XINDF	(initialization) Sets arrays to indefinite values.

B. Communication Among Subroutines

In communication, in the sense of one subroutine calling another one, there is a sharing of information among subroutines by means of variables and arrays stored in COMMON. We enumerate these elements of COMMON here, defining them briefly.

1. COMMON Variables. We first supply definition of the variables listed in COMMON.

VARIABLES IN COMMON (EXCLUDING ARRAYS AND NAMELIST INPUT PARAMETERS)

AVE	Second-order accurate option parameter
CLK	System furnished time of day for run identification
CSANG	Cosine of contact angle
CYCLE	Calculational time cycle number
DAT	Date furnished by system for run identification
DELMN	Smallest cell dimension in problem
DELXRL	Ratio of cell spacing on left side of mesh (= DELX(1)/DELX(2))
DELXRR	Ratio of cell spacing on right side of mesh (= DELX(IMAX)/DELX(IMI))
DELYRBK	Ratio of cell spacing on back side of mesh (= DELY(JMAX)/DELY(JMI))
DELYRF	Ratio of cell spacing on front side of mesh (= DELY(1)/DELY(2))
DELZRB	Ratio of cell spacing at bottom of mesh (= DELZ(1)/DELZ(2))
DELZRT	Ratio of cell spacing at top of mesh (= DELZ(KMAX)/DELZ(KM1))
DLZ	Contour plot parameter (= 0.0 program determines interval, = value specifies interval)
DPMX	Contour plot parameter (= length of x-axis)
DPMY	Contour plot parameter (= length of y-axis)
DPMZ	Contour plot parameter (= length of z-axis)
DTVIS	Maximum DELT value allowed by the viscous forces stability criterion
EMF	Small value used to negate round-off error effects in F convection (= 10^{-6})
EMFI	1.0 – EMF
EM6	10^{-6}
EPSI	Input parameter
FIXL(m)	Plot parameter (= left plot boundary in plotting coordinates for each plot plane)
FIXR	Plot parameter (= right plot boundary)
FIYB	Plot parameter (= bottom plot boundary)

FIYT	Plot parameter (= top plot boundary)
FLG	Pressure iteration convergence indicator (= 0.0 converged)
FLGC	Volume of fluid convection limit indicator
FNOC	Pressure convergence failure indication
I	INDEX value (normally refers to R(x) coordinate cell index)
IBAR	Number of real cells in x direction (excludes fictitious cells)
IGRD	Plotting parameter (= 0. Plot routines determine number of grid lines, = value gives IGRD + 1 grid lines)
II0	Data storage parameter (= NQ*IBAR)
II1	Data storage parameter (= NQ*IMAX)
II2	Data storage parameter (= NQ*IMAX*JMAX)
II3	Data storage parameter (= NQ*IMAX*JBAR)
II4	Data storage parameter (= NQ*IMAX*JMAX*KBAR)
II5	Data storage parameter (= IMAX*JMAX)
II6	Data storage parameter (= IMAX*JBAR)
II7	Data storage parameter (= IMAX*JMI)
IJK	Single index value for cell i,j,k
IJKM	Single index value for cell i,j,k-1
IJKP	Single index value for cell i,j,k + 1
IJMK	Single index value for cell i,j – 1,k
IJMKP	Single index value for cell i,j – 1,k + 1
IJPK	Single index value for cell i,j + 1,k
IJPKM	Single index value for cell i,j + 1,k-1
IJPKP	Single index value for cell i,j + 1,k + 1
IM1	Value of the index I at the last real cell in the x-direction (= IMAX-1)
IM2	Value of the index I at the next to last real cell in the x-direction (= IMAX-2)
IMAX	Total number of cells in x-direction (= IBAR + 2)
IMJK	Single index value for cell i – 1,j,k
IMJKM	Single index value for cell i – 1,j,k-1
IMJKP	Single index value for cell i – 1,j,k + 1
IMJPK	Single index value for cell i – 1,j + 1,k
IPJK	Single index value for cell i + 1,j,k
IPJKM	Single index value for cell i + 1,j,k – 1
IPJKP	Single index value for cell i + 1,j,k + 1

IPJMK	Single index value for cell i + 1,j – 1,k
IPJPK	Single index value for cell i + 1,j + 1,k
IMJMK	Single index value for cell i – 1,j – 1,k
IJMKM	Single index value for cell i,j – 1,k-1
IMJPKP	Single index value for cell i – 1,j + 1,k + 1
IPJMKP	Single index value for cell i + 1,j – 1,k + 1
IPJPKM	Single index value for cell i + 1,j + 1,k – 1
IPJPKP	Single index value for cell i + 1,j + 1,k + 1
ITER	Pressure iteration counter
J	Azimuthal (y) cell index
JBAR	Number of real cells in y-direction (= JMAX – 2)
JC2PI	Indicator of 360 degree geometry (CYL = 1.0)
JM1	Value of the index J at the last real cell in the y-direction (= JMAX – 1)
JM2	Value of the index J at the next to last real cell in the y-direction (= JMAX – 2)
JMAX	Total number of mesh cells in the y-direction (= JBAR + 2)
K	Axial (z) cell index
KBAR	Number of real cells in z-direction (= KMAX – 2)
KM1	Value of the index K at the last real cell in the z-direction (= KMAX – 1)
KM2	Value of the index K at the next to last cell in the z-direction (= KMAX – 2)
KMAX	Total number of mesh cells in the z-direction (= KBAR + 2)
NCR1	Combined length of common blocks SSCM1 through SSCM6
NCR2	Individual length of common blocks SLCM1 through SLCM4
NFLGC	Accumulated F convection limit excesses
NOCON	Accumulated pressure convergence failures
NUMTD	Restart tape dump counter
PI	= 3.141592654
RIJK	Reciprocal of the number of real nonobstacle cells on computational mesh
SANG	Sine of the contact angle
STIM	System clock time when problem commences
TANCA	Tangent of contact angle

TLM	Problem control parameter (=system request time less orderly ending increment)
TWPLT	Problem time for next plot and/or data print to be sent to film
TWPRT	Problem time for next paper data print to be sent to output
TWTD	Problem time for next restart tape dump
UDUM	Temporary storage for u component of velocity
VCHGT	Accumulated fluid volume change in mesh
VDUM	Temporary storage velocity
VOFTOT	Total fluid volume on mesh at cycle n
WDUM	Temporary storage w velocity
XBL	Plot parameter--left boundary of plotting mesh in problem units (= x (1))
XBLC	Left boundary of plotting mesh in problem units for R, θ velocity vector plot (= x (1) normally)
XBR	Right boundary of plotting mesh in problem units (= x (IM1))
XLABLE	Contour plot x-axis label
YBBK	Back boundary of plotting mesh in problem units (= y (JM1))
YBF	Front boundary of plotting mesh in problem units (= y (1))
YBFC	Front boundary of plotting mesh in problem units for R,θ velocity vector plots (= y (1) normally)
YBABLE	Contour plot y-axis label
ZBB	Bottom boundary of plotting mesh in problem units (= z (1))
ZBT	Top boundary of plotting mesh in problem units (= z (KM1))
ZLABLE	Contour plot z-axis label
ZMN	Contour plot parameter
ZMX	Contour plot parameter

2. ***COMMON Arrays***. Finally, we enumerate and supply definitions for the arrays listed in COMMON.

ARRAYS IN COMMON (EXCLUDING NAMELIST ARRAYS)

ABK(i,j,k)	Fractional area open to flow on back wall of cell (i,j,k)
AC(i,j,k)	Fractional volume open to flow in cell (i,j,k)
AR(i,j,k)	Fractional area open to flow on right wall of cell (i,j,k)
AT(i,j,k)	Fractional area open to flow on top wall of cell (i,j,k)
BETA(i,j,k)	Pressure iteration relaxation factor in cell (i,j,k)

CTHJ(j)	Cosine of θ (at cell center) = cos (YJ(j)/X(IM1))
CTHJBK(j)	Cosine of θ (at cell back) = cos (Y(j)/X(IM1))
D(i,j,k)	The residual ($\nabla \cdot \mathbf{u}$) for cell (i,j,k) after convergence of the pressure iteration
DELX(i)	Mesh spacing of the i-th cell along the radial (x) coordinate
DELY(j)	Mesh spacing of the j-th cell along the azimuthal (y) coordinate at the maximum R(x) mesh value
DELZ(k)	Mesh spacing of the k-th cell along the axial (z) coordinate
FIXL(n)	Left boundary of mesh in plot coordinates for each plotting plane
F(i,j,k)	Volume of fluid per unit volume of cell (i,j,k) at time level n + 1
FN(i,j,k)	Volume of fluid per unit volume of cell (i,j,k) at time level n
GRDBN(n)	Array of values of plot boundaries in problem coordinates for use in DGAP plot routine
GXA(i,j)	Radial component of acceleration due to applied body force for cell (i,j,k)
GYA(i,j)	Azimuthal component of acceleration for cell (i,j,k) due to applied body force
JOP(j)	Index value of plane opposite jth J plane (appropriate for CYL = 1.0 only)
NF(i,j,k)	Flag indicating cell type at time level n + 1
NFO(i,j,k)	Flag indicating cell type at time level n
NFP(i,j,k)	Flag indicating provisional cell type at time level n + 1 (from fluid surface height function)
NFS(i,j,k)	Flag indicating provisional cell type at time level n + 1 (from fluid surface slope function)
P(i,j,k)	Pressure in cell (i,j,k) at time level N + 1
PETA(i,j,k)	Pressure interpolation factor for cell (i,j,k)
PN(i,j,k)	Pressure in cell (i,j,k) at time level N
PR(l)	Pressure in void region l (nominally = 0)
PS(i,j,k)	Surface pressure in cell (i,j,k) computed from surface tension forces
Q(m,n)	Temporary storage for plot routine
RDX(i)	Reciprocal of DELX(i)
RDXP(i)	2.0/(DELX(i) + DELX(i + 1))

RDY(j)	Reciprocal of DELY(j)
RDYP(j)	2.0/(DELY(j) + DELY(j + 1))
RDZ(k)	Reciprocal of DELZ(k)
RDZP(k)	2.0/(DELZ(k) + DELZ(k + 1))
RX(i)	Reciprocal of X(i)
RXI(i)	Reciprocal of XI(i)
RY(j)	Reciprocal of Y(j)
RYJ(j)	Reciprocal of YJ(j)
RZ(k)	Reciprocal of Z(k)
RZK(l)	Reciprocal of ZK(k)
STHJ(j)	Sine of θ (at cell center) = sin (YJ(j)/X(IM1))
STHJBK(j)	Sine of θ (at cell back) = sin (Y(j)/X(IM1))
U(i,j,k)	x-direction velocity component in cell (i,j,k) at time level n + 1
UN(i,j,k)	x-direction velocity component in cell (i,j,k) at time level n
UVV(m,n)	x-component of velocity at center of cell (i,j,k) (used in velocity vector plots)
V(i,j,k)	y-direction velocity component in cell (i,j,k) at time level n + 1
VN(i,j,k)	y-direction velocity component in cell (i,j,k) at time level n
VVV(m,n)	y-component of velocity at center of cell (i,j,k)(used in velocity vector plots)
W(i,j,k)	z-direction velocity component in cell (i,j,k) at time level n + 1
WN(i,j,k)	z-direction velocity component in cell (i,j,k) at time level n
WVV(m,n)	z-component of velocity at center of cell (i,j,k)(used in velocity vector plots)
X(i)	Location of the right-hand boundary of the i-th cell along the x-axis
XCONV(n)	Scale factors used to convert from problem coordinates to plot coordinates (horizontal)
XI(i)	Location of the center of the i-th cell along the x-axis
XPC(m,n)	x-coordinate of center of cell (i,j,k)(used in velocity vector plots)
Y(j)	Location of the back boundary of the j-th cell along the y-axis
YCONV(n)	Scaling factors used to convert from problem coordinates to plot coordinates (vertical)
YJ(j)	Location of the center of the j-th cell along the y-axis

YPC(m,n) y-coordinate of center of cell (i,j,k)(used in velocity vector plots)

Z(k) Location of the top boundary of the k-th cell along the z-axis

ZCQ(n) Contour plot parameters

ZK(k) Location of the center of the k-th cell along the z-axis

ZPC(m,n) z-coordinate of center of cell (i,j,k)(used in velocity vector plots)

C. Problem Input

Problem input data are furnished the code by the NAMELIST format-free option. Most have default values specified by DATA or DEFINING statements located in the subroutine containing the NAMELIST list. Data sets for problem input need only refer to variables or parameters whose values differ from these defaults.

1. NAMELIST XPUT. NAMELIST XPUT parameters specify the material properties of the fluid, the solution method options, problem-dependent code parameters, and output requirements. NAMELIST XPUT is read in subroutine RINPUT

INPUT PARAMETERS (NAMELIST /XPUT/)

ALPHA Controls amount of donor-cell fluxing (= 1.0 for full donor-cell differencing, = 0.0 for central differencing)

AUTOT Automatic time step flag (= 1.0 for automatic DELT adjustment, = 0.0 for constant DELT)

CANGLE Contact angle, in degrees, between fluid and wall

CYL Mesh geometry indicator (= 1.0 for cylindrical coordinates = 0.0 for Cartesian coordinates)

DELT Initial time step for calculation (subsequently becomes current time step)

EPSI Pressure iteration convergence criterion

FLHT Initial fluid height in computing mesh (if appropriate)

GX Body acceleration in positive x-direction

GY Body acceleration in positive y-direction

GZ Body acceleration in positive z-direction

ICLIP Indicator used with zoom option (= 0 normal, = 1 eliminate extraneous lines)

ICSURF Indicator for fluid generator

= 0 , horizontal or equilibrium surface

	= 1 ,	axisymmetric surface
	= 2 ,	axisymmetric surface

IDEFM Defoamer option flag

= 0 , defoamer off

= 1 , defoamer on

IEQUIB Indicator for fluid generator

= 0 Do not generate equilibrium free surface

= 1 Generate equilibrium free surface

IORDER Indicator for second-order accurate velocity calculation option

ISOR Pressure iteration solution method (conjugate residual = 0, SOR = 1)

ISURFT Indicator for surface tension

= 0 No surface tension

= 1 Calculate surface tension

IZOOM Indicator for plot routines (= 0 normal, = 1 magnifies a portion of the mesh)

JNM Graphics plot job identification

LPR Print output parameter

LPR = 0 , No paper or film data prints or film plots

= 1 , Film plots only

= 2 , Film plots plus film data prints

= 3 , Paper data prints only

NAME(N) Problem identification

NFCAL NF Determination algorithm flag

NFCAL = 1 , Use provisional value

NFCAL = 2 , Use slope value

NFCAL = 3 , Use decision-making algorithm to decide between provisional and slope values

NOWALL Indicator for wall adhesion (= 1 no wall adhesion, = 0 wall adhesion)

NU Coefficient of kinematic viscosity

OMG Over-relaxation factor used in SOR pressure iteration

PLTDT Time increment between plots and/or data prints to be output on film

PRTDT Time increment between data prints to be output on paper

RADPS Constant angular rotation velocity

RHOF Fluid density

SIGMA Surface tension coefficient

TD	Dump number on restart tape to continue problem
T	Problem time
TDDT	Time increment between restart tape dumps
TLIMD	Problem run parameter (= 1.0 terminate gracefully, = 0.0 run to system time limit)
TWFIN	Problem time to end calculation
UI	x-direction velocity used for initializing mesh
VELMX	Estimate of maximum velocity in problem, used to scale velocity vectors in plots
VI	y-direction velocity used for initializing mesh
WB	Indicator for boundary condition along bottom of mesh (K = 1)
WBK	Indicator for boundary condition along back of mesh (J = JMI)
WF	Indicator for boundary condition along front of mesh (J = 1)
WI	z-direction velocity used for initializing mesh
WL	Indicator for boundary condition along left side of mesh (I = 1)
WR	Indicator for boundary condition along right side of mesh (I = IM1)
WT	Indicator for boundary condition along top of mesh (K = KM1)

2. NAMELIST MESHGN. NAMELIST MESHGN is used to define the computational region dimensions and the zoning within that region. This function was performed by NAMELIST MSHSET in NASA-VOF2D. The computing mesh is constructed from a number of submeshes defined in each coordinate direction. The namelist input specifies the boundaries, the number of cells, the minimum cell dimension, and the convergence point of each submesh. Variable spacings are accommodated by linking a group of submeshes together to achieve any desired distribution of cell spacing. The number of cells is specified in each submesh on each side (i.e., to the left and to the right) of the convergence point. Both cells directly adjacent to the convergence point will have a cell spacing equal to the minimum value specified in the input as DXMN, DYMN, or DZMN. The cell spacing is then expanded quadratically from these cells to the left and right edges of the submesh in accordance with the desired number of cells (NXR,NXL,NYR,NYL, or NZR,NZL) in the input list. If the number of cells specified on the left (right) should produce a uniform cell size that is less than the minimum specified cell size DXMN (or DYMN), a uniform spacing is then used on the left (right). The number of cells to the left and to the right of the convergence point need not be equal, but there must be at least one on both sides.

When two or more submeshes are linked together, it is imperative that the location of the left edge of the right submesh be the same as the location of the right edge of the left submesh. In addition, large disparities in cell spacing should be avoided within a submesh and going from one submesh to another by adjusting the number of cells in the various submeshes. As a general rule the spacing of adjacent cells should not differ by more than 10-20%. The aspect ratio (DELX/DELY) for a given cell should not exceed 1.5 and should be greater than 0.67.

An example of the proper format to be used to specify a mesh spanning the x-dimension LW < x < RW with n submeshes is

$$NKX = n, \quad XL = LW,\ XL_2,\ XL_3, \ldots, XL_n,$$

$$XC = XC_1,\ XC_2, \ldots XC_n, \quad XR = XL_2,\ XL_3, \ldots, XL_n,\ RW,$$

$$NXL = NL_1,\ NL_2, \ldots, NL_n, \quad NXR = NR_1,\ NR_2, \ldots, NR_n,$$

$$DXMN = DXMN_1,\ DXMN_2, \ldots, DXMN_n$$

in which NL_i represents the number of cells to the left of XC_i and NR_i is the number of cells to the right of XC_i in each submesh i, i = 1, 2, ..., n.

References 2 and 10 contain several examples of mesh generation input and resultant computational meshes.

MESH SETUP INPUT (NAMELIST/MESHGN/)

DXMN(n)	Minimum space increment in x-direction in submesh n
DYMN(n)	Minimum space increment in y-direction in submesh n
DZMN(n)	Minimum space increment in z-direction in submesh n
NKX	Number of submeshes defining radial (x) coordinate spacing
NKY	Number of submeshes defining azimuthal (y) coordinate spacing
NKZ	Number of submeshes defining axial (z) coordinate spacing
NXL(n)	Number of cells between locations XL(n) and XC(n) in submesh n
NXR(n)	Number of cells between locations XC(n) and cells between locations XC(n) and XL(n + 1) in submesh n
NYL(n)	Number of cells between locations YL(n) and YC(n) in submesh n
NYR(n)	Number of cells between locations YL(n) and YL(n + 1) in submesh n
NZL(n)	Number of cells between locations ZL(n) and ZC(n) in submesh n
NZR(n)	Number of cells between locations ZC(n) and ZL(n + 1) in submesh n
XC(n)	x-coordinates of the convergence point in submesh n

XL(n) Location of the left edge of submesh n (NKX + 1 values required since XR(N) ≡ XL(n + 1))

YC(n) y-coordinate of the convergence point in submesh n

YL(n) Location of the front edge of submesh n (NKY + 1 values required since YR(n) ≡ YL(n + 1))

ZC(n) Location of the convergence point of submesh n

ZL(n) Location of the bottom edge of submesh n (NKZ + 1 values required since ZR(n) = ZL(n + 1))

NAMELIST MESHGN also provides information to the interior-obstacle-generating subroutine. The latter function was performed by NAMELIST ASETIN in NASA-VOF2D. Interior obstacles are defined as any nonflow regions within the computational mesh. As with the mesh generator, it is convenient to have the necessary flags and parameters produced automatically by the code for arbitrary-shaped axisymmetric obstacles. The generator uses a series of conic sections to define obstacle surfaces plus an additional parameter that directs the generator to designate computational cells as either flow or nonflow regions. The conic sections may overlap one another and this feature is utilized to form complex axisymmetric surfaces, representing baffles or container walls.

This procedure is as follows: coefficients of the general conic function

$$F(x,z) = a_2x^2 + a_1x + b_2z^2 + b_1z + c_2xz + c_1$$

are chosen such that some portion of the defining surface coincides with that of the desired internal obstacle. Computational cells partially or completely inside the conic surface [i.e., $F(x_c, z_c) < 0.0$, x_c and z_c being cell boundary coordinates] are flagged closed to flow, partially closed to flow, or fully open to flow depending upon the additional parameter IOH (= 1.0 closed, = 0.0 open) associated with each function. Generally, additional functions can be utilized to remove unwanted obstacle cells added by other functions. For example, given reasonable zoning, a 1.0 × 1.0 cm obstacle in the lower right corner of a 2.0 × 2.0 cm mesh would result from $f_1 = -x + 1.0$, $IOH_1 = 1.0$, which defines all cells, regardless of z value, to the right of x = 1.0 to be obstacles, and followed by $f_2 = -z + 1.0$, $IOH_2 = 0.0$, which removes all obstacle cells above z = 1.0. Nonaxisymmetric obstacles (or container boundaries) can be generated by inserting additional coding at the end of subroutine ASET to define AR(i,j,k), AT(i,j,k), ABK(i,j,k), AC(i,j,k), and BETA(i,j,k) for each nonflow or partial flow cell (i,j,k).

After all sets of obstacle data have been processed, the computing mesh is swept and individual cell area and volume open-to-flow variables are assigned. Aside from geometric factors for cylindrical coordinates, they range from 0.0 (not open to flow) to 1.0 (fully open to flow).

OBSTACLE SETUP (NAMELIST/MESHGN/CONTINUED)

IOH(n)	Indicator to add obstacles inside function (n = 1) or subtract obstacles (n = 0)
NOBS	Number of sets of obstacle data input
OA2(n)	Coefficient of x^2-term in function F
OA1(n)	Coefficient of x-term in function F
OB2(n)	Coefficient of z^2-term in function F
OB1(n)	Coefficient of z-term in function F
OC2(n)	Coefficient of xz-term in function F
OC1(n)	Coefficient of constant term in function F

3. ***NAMELIST GRAFIC.*** The input information needed to prepare the data, formats, and captions for the graphics is supplied by NAMELIST GRAFIC. We define the information here, presenting it in the order required by the calling sequence associated with SUBROUTINE RGRAFIC. The information is further broken down by specific functions which are separated by lines of asterisks (****...): first there is information connected with the eye location in perspective plots, next there is information connected with velocity vector plots, following that, contour plot data, and, finally, information for surface plots.

FORMAT PARAMETERS FOR GRAPHS (NAMELIST/GRAFIC/)

XEA(n)	The x-coordinate of the eye position in problem units.
YEA(n)	The y-coordinate of the eye position in problem units.
ZEA(n)	The z-coordinate of the eye position in problem units.
XCA(n)	The x-coordinate of the projection plane in problem units. This coordinate is usually taken near the origin.
YCA(n)	The y-coordinate of the projection plane in problem units. This coordinate is usually taken near the origin.
ZCA(n)	The z-coordinate of the projection plane in problem units. This coordinate is usually taken near the origin.

NVEWS	The number of eye views. That is the number of sets of eye point-plane points. That is XEA etc. above. Up to 10.
NVPLTS	The number of velocity vector plots. Up to 10. If NVPLTS = 0, no velocity plots will appear.
IVVEW(n)	The eye view to be associated with this velocity vector plot. IVVEW = 1 to 10.
IV1(n)	Defines the value of I for the left boundary of the velocity vectors to be plotted. If IV1 > imax, IV1 = IM1.
IV2(n)	Defines the value of I for the right boundary of the velocity vectors to be plotted. If IV2 > imax, IV2 = IM1.
JV1(n)	Defines the value of J for the front boundary of the velocity vectors to be plotted. If JV1 > jmax, JV1 = JM1.
JV2(n)	Defines the value for J for the back boundary of the velocity vectors to be plotted. If JV2 > jmax, JV2 = JM1.
KV1(n)	Defines the value of K for the bottom boundary of the velocity vectors to be plotted. If KV1 > kmax, KV1 = KM1.
KV2(n)	Defines the value of K for the top boundary of the velocity vectors to be plotted. If KV2 > kmax, KV2 = KM1.
NAV(n)	The number of frames on the film to be advanced before the velocity vector plot is made. Normally it is 1.
IPERV(n)	Toggle switch for perspective plot. If IPERV = 1, normal plot. If IPERV = 2, perspective plot.
NCPLTS	The number of contour plots. Up to 10. If NCPLTS = 0, no contour plots will appear.
ICVEW(n)	The eye view to be associated with this contour plot. ICVEW = 1 to 10.
IC1(n)	Defines the value of I for the left boundary of the contours to be plotted. If IC1 > imax, IC1 = IM1.
IC2(n)	Defines the value of I for the right boundary of the contours to be plotted. If IC2 > imax, IC2 = IM1.
JC1(n)	Defines the value of J for the front boundary of the contours to be plotted. If JC1 > jmax, JC1 = JM1.
JC2(n)	Defines the value of J for the back boundary of the contours to be plotted. If JC2 > jmax, JC2 = JM1.
KC1(n)	Defines the value of K for the bottom boundary of the contours to be plotted. If KC1 > kmax, KC1 = KM1.

KC2(n)	Defines the value of K for the top boundary of the contours to be plotted. If KC2>kmax, KC2 = KM1.
NCV(n)	The number of frames on the film to be advanced before the contour plot is made. Normally it is 1.
IPERC(n)	Toggle switch for perspective plot. Not operative. Use 1. If IPERC = 1, normal plot. If IPERC = 2, perspective plot.
NSPLTS	The number of surface plots. Up to 10. If NSPLTS = 0, no surface plots will appear.
ISVEW(n)	The eye view to be associated with this surface plot. ISVEW = 1 to 10.
IS1(n)	Defines the value of I for the left boundary of the surface to be plotted. If IS1>imax, IS1 = IM1.
IS2(n)	Defines the value of I for the right boundary of the surface to be plotted. If IS2>imax, IS2 = IM1.
JS1(n)	Defines the value of J for the front boundary of the surface to be plotted. If JS1>jmax, JS1 = JM1.
JS2(n)	Defines the value of J for the back boundary of the surface to be plotted. If JS2>jmax, JS2 = JM1.
KS1(n)	Defines the value of K for the bottom boundary of the surface to be plotted. If KS1>kmax, KS1 = JM1.
KS2(n)	Defines the value of K for the top boundary of the surface to be plotted. If KS2>kmax, KS2 = KM1.
NSV(n)	The number of frames on the film to be advanced before the surface plot is made. Normally it is 1.
IPERS(n)	Toggle switch for perspective plot. Not operative. Use 1. If IPERS = 1, normal plot. If IPERS = 2, perspective plot.

4. NAMELIST CONTUR. The input information needed to construct the contour plots is supplied by NAMELIST CONTUR. We enumerate and define the information here, presenting it in the order required by the calling sequence associated with SUBROUTINE CONTRJB.

INPUT PARAMETERS FOR CONTOUR PLOTS (NAMELIST/CONTUR/)

Q	Origin of matrix of values of the function to be contoured. Q should be stored so that Q(I,J) is the value of Q at [X(I), Y(J)].
NNX	Number of points in x table to be used. If NNX is negative, ZMN will be calculated by CONTRJB as the minimum value of Q.

NNY	Number of points in Y table to be used. If NNY is negative, ZMX will be calculated by CONTRJB as the maximum value of Q.
NNZ	Number of points in Z table to be used. If NNZ is negative, ZMX will be calculated by CONTRJB as the maximum value of Q.
NZX	Length of a column of Q matrix. Variable dimensioning is used. For example: If Q is dimensioned Q(10,20), then nzx = 10, and nzy = 20.
NZY	Number of columns in Q matrix. If NZY<0, and NC>0, ZCQ table is computed such that NC is the minimum of NC and (ZMX-ZMN)/DLZ. ZMX and ZMN are made multiples of DLZ.
NZZ	Similar to NZY. Used in the three dimensional extension. See the calling sequence in SUBROUTINE CNTR to CONTRJB.
NC	Number of contours to be plotted. If NC<0, ZCQ table is assumed to be user supplied and ZMN, ZMX, and DLZ are not used. The absolute value of NC must be less than or equal to 50. (See DLZ)
ZMN	Minimum value of contour to be used in contouring. If NNX<0, ZMN will be computed from minimum value of Q over contour grid.
ZMX	Maximum value of contour to be used in contouring. If NNY<0, ZMX will be computed from maximum value of Q over contour grid.
DLZ	Interval between contours. If DLZ is less than or equal to zero, DLZ will be set to (ZMX – ZMN)/(NC – 1).
ZCQ	Table of contour values. If NC>0, ZCQ table is filled by CONTRJB. If NC<0, ZCQ must be supplied by user. At least the absolute value of NC values must be supplied and values must be stored in increasing order in ZCQ, i.e., ZCQ(I + 1)>ZCQ(I).
DMPX	Length of X axes.
DMPY	Length of Y axes.
DMPZ	Length of Z axes.
IGRD	If IGRD ≤0, grid lines will appear only at intervals numbered by scaling subroutine. For a finer mesh, IGRD>0 will add IGRD-1 extra lines per interval.
ITITLE1	BCD array containing title to be written at top of graph 1.
NTITLE1	Number of characters in title 1.
ITITLE2	BCD array containing title to be written at top of graph 2.

NTITLE2	Number of characters in title 2.
XLABEL	BCD arrays containing label to be written on the X axes.
YLABEL	BDC arrays containing label to be written on the Y axes.
ZLABEL	BCD arrays containing label to be written on the Z axes.
NXLBL	Number of characters in X label.
NYLBL	Number of characters in Y label.
NZLBL	Number of characters in Z label.

5. NAMELIST FLUIDGN. Generating the initial fluid configuration is done in SUBROUTINE SETUP. If the surface is other than a horizontal or equilibrium surface, (ICSURF>0), then subroutine SETFS is called.

A complex initial free surface configuration is specified through NAMELIST FLUIDGN input data parameters. The generation of the free surfaces is accomplished in a manner analogous to the generation of interior obstacles through the use of conic functions. Computational cells completely or partially enclosed by the surface defined by a function are either filled, partially filled, or empty of fluid as specified by the input parameters of that function. General functions may be used in conjunction to define very general configurations. Subroutine SETFS has two logic paths, one for axisymmetric volumes and a second for nonaxisymmetric volumes. The distinction was made for computational efficiency since the algorithms for generating off axis (i.e., nonaxisymmetric volumes) lengthens the set-up time considerably.

FREE SURFACE INPUT DATA (NAMELIST/FLUIDGN/)

IQH(n)	Indicator for adding or subtracting fluid inside function n
NQBS	Number of functions to be used in generating free surfaces A
Axisymmetric (ICSURF = 1)	
QA1(n)	Coefficient of x term function n
QA2(n)	Coefficient of x^2 term function n
QB1(n)	Coefficient of z term function n
QB2(n)	Coefficient of z^2 term function n
QC1(n)	Constant term function n
OC2(n)	Coefficient of xz term function n
NonAxisymmetric (ICSURF> 1)	
QA1(n)	Value of x-coordinate at center of volume n
QA2(n)	Lower limit of I index
QA3(n)	Upper limit of I index

QB1(n) Value of y-coordinate at center of volume n

QB2(n) Lower limit of J-index

QB3(n) Upper limit of J-index

QC1(n) Value of z-coordinate at center of volume n

QC2(n) Lower limit of K index

QC3(n) Upper limit of K index

QD1(n) Constant term in function n

QD2(n) Indicator specifying if mesh is initially fluid or void (= 1.0 fluid, = 0.0 void)

The upper and lower limits on the three indices restrict the algorithm to that portion of the computing mesh known to contain volume n.

D. <u>User Options</u>

We summarize here the principal user options to influence the manner in which the calculation is to be carried out. We list principally option flags, i.e., integer valued variables which specify alternate logic paths for the calculations.

ALPHA	0.0	Problem runs with central differencing of convection.
	1.0	Problem runs with donor-cell differencing.
	1.0	and >0.0 (usual case) partial donor-cell differencing.
AUTOT	0	Problem is to be run with constant δt.
	1	Program will automatically impose time step controls; user must check for consistency of parameters.
CYL	0.0	Problem is to be run in Cartesian coordinates. (This option is not presently implemented.)
	1.0	Problem is to be run in cylindrical coordinates.
ICLIP	0	Normal zoom operation (if used).
	1	Eliminate extraneous lines with zoom.
ICSURF	0	Surface is horizontal or an equilibrium surface.
	1	Surface is axisymmetric. Its shape is specified in SETFS by quadratic functions.
	2	Surface is nonaxisymmetric. Its shape is again specified in SETFS, but by alternate quadratic function formalism.
IDEFM	0	Program will not use defoamer.
	1	Defoamer option on.
IEQUIB	0	Program will not generate equilibrium free surface.
	1	Program generates equilibrium free surface.
IORDER	1	Program does not call second-order accuracy option.

	2	Program calls second-order accuracy option.
ISOR	0	Program executes Conjugate Residual option.
	1	Program executes SOR option.
ISURFT	0	Program does not calculate surface tension.
	1	Program calculates a surface pressure, p_s, from surface tension algorithms.
IZOOM	0	Normal (no zoom) operation of plot routines.
	1	A portion of the mesh will be magnified.
LPR	0	No paper or film data prints or film plots.
	1	Film plots only.
	2	Film plots plus film data prints.
	3	Paper data prints only.
NFCAL	1	Use provisional values for NF.
	2	Use slope determination of NF value.
	3	Use decision-making algorithm to choose between provisional and slope values for NF.
NOWALL	0	Program imposes wall adhesion (this option is not presently implemented).
	1	Program does not impose wall adhesion at physical boundaries. Only active option in present version of code.
TD	≤ 0	Problem started from initial data.
	>0	Dump tape7 is read to start problem.
TLIMD	0.0	Problem runs to system time limit.
	1.0	Problem will terminate gracefully.

E. Example of Problem Setup

Table II is an example of a typical input data set for NASA-VOF3D. The code is to simulate a small nearly full glass vessel with cylindrical walls and hemispherical bottom draining under the influence of reduced gravity, surface-tension, and wall-adhesion. The discharge flow rate is specified and constant in time. Here cgs units are used for input and output; any consistent set of units may be employed. From the tabulated input parameters one may see that the conjugate residual method was chosen (ISOR = 0), the defoam option is on (IDEFM = 1), and the standard fluid generator is specified (ICSURF = 0). The initial fluid volume is specified by the fluid height above tank bottom (FLHT = 6.0). The surface-tension flag is on (ISURFT = 1), cylindrical coordinates are chosen (CYL = 1.0), and the equilibrium surface calculation is requested (IEQUIB = 1). The slope method for determining NF is specified

(NFCAL = 2). The initial DELT is computed from the outflow rate, discharge opening area, and mesh spacing in a fashion that satisfies stability constraints. The convergence criterion is set sufficiently small (EPSI = 0.001) that negligible pressure errors result from nonzero ($\nabla \cdot \mathbf{u}$)'s. Film prints and plots are requested (LPR = 2).

Figure 2 displays the initial problem configuration generated from XPUT and MESHGN data. The x coordinate has two submeshes with a total of 11 real cells, the spacing being finer near the tank wall. The y coordinate mesh consists of a single (wedge-shaped) physical cell of arbitrary thickness (NYL = 1, NYR = 0, DYNMN = 1.0). The z coordinate has a single submesh with 36 equally spaced real cells. The curved boundary at the bottom was drawn from cell variables and parameters produced by the obstacle generator subroutine from MESHGN data. Computational cells with areas intersected by the curve are defined to be partial flow cells whose boundaries are identified as partial flow areas. SUBROUTINE ASET computes the fraction of each face open to flow and the fractional cell volume open to flow (i.e., cell parameters AR, AT, ABK, AC). Computational mesh cells to the right and below the curve are designated obstacle cells, while those to the left and above are fluid cells. All interior obstacles and exterior, i.e., container, walls are simulated.

TABLE II

EXAMPLE OF INPUT DATA

```
$XPUT
JNM = 8H VOF3D7 .NAME = 45HVOF3D TANK DRAIN J = 2 PLANE ONLY    11/13/86
NFCAL = 2, IEQUIB = 1, ICSURF = 0, IDEFM = 1, RHOF = 1.58,
CYL = 1.0, DELT = 0.0002, GZ = - 14.72, VELMX = 1.0, NU = .00443, ISOR = 0,
EPSI = 0.001, VI = 0.0, SIGMA = 18.6, LPR = 2, CANGLE = 5.0, ISURFT = 1,
WF = 1, WBK = 1, FLHT = 6.0, TWFIN = 2.5, OMG = 1.8, PLTDT = .10, TDDT = 5.0
$END
$MESHGN
NKX = 2, XL = 0.0, 0.40, 2.0, XC = 0.20, 1.90, NXL = 1, 8, NXR = 1, 1,
DXMN = 0.20, 0.10,
NKY = 1, YL = 0., 1.0, YC = 1.0, NYL = 1, NYR = 0, DYMN = 1.0,
NKZ = 1, ZL = 0.0, 7.20, ZC = 3.60, NZL = 18, NZR = 18, DZMN = 0.20,
NOBS = 2, OA2 = - 1.0, 0.0, OA1 = 0.0, OB2 = - 1.0, 0.0, OB1 = 4.0, - 1.0,
OC2 = 0.0, 0.0, OC1 = 0.0, 2.0, IOH = 1,0
$END
$GRAFIC
NVEWS = 1, XEA(1) = 700., YEA(1) = 50., ZEA(1) = 1000., XCZ(1) = 150., YCA(1) = 15.,
ZCA(1) = 400., NVPLTS = 1, NCPLTS = 1, NSPLTS = 0,
IVVEW(1) = 1,  NAV(1) = 1,  IPERV(1) = 1,
IV1(1) = 2,  IV2(1) = 99,
JV1(1) = 2,  JV2(1) = 2
KV1(1) = 2, KV2(1) = 99,
ICVEW(1) = 1,    NAC(1) = 1,
IPERC = 1,
IC1(1) = 2,  IC2(1) = 99,
JC1(1) = 2,  JC2(1) = 2,
KC1(1) = 2, KC2(1) = 99,
$END
$CONTUR
$END
$FLUIDGN
$END
```

The initial free surface shape is furnished by SUBROUTINE EQUIB, which solves the two-point boundary value problem for the equilibrium position of the free surface. The parameters of the equation are the contact angle (cangle) and the Bond number ($\rho g R^2/\sigma$, where R is the tank radius). This free surface configuration is utilized to specify void cells, partially filled fluid cells, and completely full fluid cells in the computational mesh. The free surface is represented in the plot by the $F = 0.5$ contour, which generally does not correctly display the surface meniscus at the walls.

Table III contains code changes necessary to run the example. Programs for maintaining and updating source decks (UPDATE or HISTORIAN) afford a convenient way to incorporate problem-dependent modifications into the executable code. These particular modifications arise since we wish to simulate a small pipe located on the tank bottom at the cylindrical axis, whereas the code lacks provision for partially open mesh boundaries.

First, the parameter statement (No. 2 of SLCOM1) is replaced, specifying minimum storage requirements for the various arrays. Statements inserted at the end of SUBROUTINE BETACAL redefine the cell area-open-to-flow variables for cell $i = 2$, $j = 2$, $k = 1$, so that a discharge port is created. A second insertion (in SUBROUTINE BC) specifies the constant outflow velocity for this port. Also, changes are made in SUBROUTINE PRESCR to correctly apply outflow velocity boundary conditions for discharge port cells in the pressure iteration.

Efficiency is enhanced by a problem end test inserted into SUBROUTINE SOLA and by an insertion into SUBROUTINE VFCONV which excludes the fictitious port cell from contributing to the time step limitations arising from stability considerations. Finally, the insertion into SUBROUTINE SURF10N facilitates comparison with experimental data by estimating the heights of the free surface at the axis ($i = 2$) and at the container wall ($i = IM1$).

Table IV is an echo print of the example problem input data. It displays the complete set of problem run and output control parameters.

A partial cycle print is included in **Table V** as an example of code output and as an aid to checkouts. It is the solution after one time step; the flow field is established and the pressure and velocity fields are consistent. Surface pressures are defined only for surface cells ($0<NF(i,j,k)<7$) and are negative, indicating that unbalanced surfaces are directed upward (acting to straighten the surface). Cell $i = 2$, $j = 2$, $k = 30$ has $F = 0.111$ and $NF = 5$, indicating that the surface passes through the cell, is nearly horizontal and is above the fluid.

Figure 3 displays the time history of the free surface heights at the axis (below) and at the container wall (above). The experimental data reported by Symons[12] were obtained from analysis of high speed photography. Calculational results

TABLE III

EXAMPLE PROBLEM CODE MODIFICATIONS

```
*IDENT TANKD
*D,SLCOM1.2
C
C       CHANGE STORAGE DIMENSIONS
C
      PARAMETER (NQ=6, IBAR2=13, JBAR2=3, KBAR2=38, IBASC=NQ*IBAR2
*I,BC.233
C
C       IMPOSE CONSTANT VELOCITY AT OUTFLOW BOUNDARY
C
      DO 600 J=2,JM1
      IJK=NQ*(IMAX*(J-1)+1)+1
      WOUTB=249.7936
      W(IJK)=-WOUTB
  600 CONTINUE
*I,BETACAL.38
C
C       MODIFY AREA OPEN TO FLOW TO CREATE OUTFLOW OPENING
C
      DO 600 J=1,JMAX
      IJK=NQ*(IMAX*(J-1)+1)+1
      BETA(IJK)=+1.0
      AT(IJK)=1.0
      AC(IJK)=1.0
      ABK(IJK)=1.0
  600 CONTINUE
*I,PRESCR.117
C
C       MODIFY PRESSURE SOLUTION TO IMPOSE CONSTANT VELOCITY OUTFLOW
C       BOUNDARY CONDITION
C
      IF(I.EQ.2.AND.K.EQ.1) GO TO 340
*I,PRESCR.175
C
C       MODIFY PRESSURE SOLUTION TO IMPOSE CONSTANT VELOCITY OUTFLOW
C       BOUNDARY CONDITION
C
      IF(I.EQ.2.AND.K.EQ.2) GO TO 290
*I,VFCONV.21
C
C       SKIP STABILITY REQUIREMENTS FOR FICTITIOUS OUTFLOW CELL
C
      IF(K.EQ.1) GO TO 20
*I,SOLA.124
C
C     END PROB IF VAPOR INGESTED
C
      IJK=NQ*(IMAX*JMAX+IMAX+1)+1
      IF(F(IJK).LT.0.1) TWFIN=T
*I,SURF1ON.508
C
C       OUTPUT FREE SURFACE HEIGHTS AT AXIS AND WALL
C
      IF(MOD(CYCLE,20).NE.0) GO TO 9092
      IF(I.EQ.2.OR.I.EQ.IM1) HISURF=Z(K-1)-DELZ(K-1)+F(IJKM)*DELZ(K-1)+
     1 F(IJK)*DELZ(K)+F(IJKP)*DELZ(K+1)
      IF(I.EQ.2) WRITE(12,9090) I,J,K,T,HISURF
      IF(I.EQ.IM1) WRITE(12,9091) I,J,K,T,HISURF
 9090 FORMAT(5X,3I3,1PE12.5,*   AXIS HEIGHT=*E12.5)
 9091 FORMAT(5X,3I3,1PE12.5,*   WALL HEIGHT=*E12.5)
 9092 CONTINUE
```

TABLE IV

EXAMPLE INPUT ECHO PRINT

```
VOF3D TANK DRAIN J=2 PLANE ONLY        11/13/86      #'#'#'#'#'#'#'#'
              ALPHA=    1.00000E+00
              AUTOT=    1.00000E+00
             CANGLE=    8.72665E-02
                CYL=    1.00000E+00
               DELT=    2.00000E-04
               EPSI=    1.00000E-03
               FLHT=    6.00000E+00
                 GX=    0.00000E+00
                 GY=    0.00000E+00
                 GZ=   -1.47200E+01
              ICLIP=    0
             ICSURF=    0
              IDEFM=    1
             IEQUIB=    1
               ISOR=    0
             IORDER=    1
             ISURFT=    1
              IZOOM=    0
                JNM=    VOF3D7
                LPR=    2
              NFCAL=    2
             NOWALL=    0
                 NU=    4.43000E-03
                OMG=    1.80000E+00
              PLTDT=    1.00000E-01
              PRTDT=    1.00000E+00
              RADPS=    0.00000E+00
               RHOF=    1.58000E+00
              SIGMA=    1.86000E+01
                  T=    0.00000E+00
               TDDT=    5.00000E+00
                 TD=     -1
              TLIMD=    1.00000E+00
              TWFIN=    2.00000E-04
                 UI=    0.00000E+00
              VELMX=    1.00000E+00
                 VI=    0.00000E+00
                 WB=    1
                WBK=    1
                 WF=    1
                 WI=    0.00000E+00
                 WL=    1
                 WR=    1
                 WT=    1
NKX=     2
MESH-X=     1   XL=  0.00000E+00   XC=  2.00000E-01   XR=  4.00000E-01   NXL=   1   NXR=   1   DXMN=  2.00000E-01
MESH-X=     2   XL=  4.00000E-01   XC=  1.90000E+00   XR=  2.00000E+00   NXL=   8   NXR=   1   DXMN=  1.00000E-01
NKY=     1
MESH-Y=     1   YL=  0.00000E+00   YC=  1.00000E+00   YR=  1.00000E+00   NYL=   1   NYR=   0   DYMN=  1.00000E+00
NKZ=     1
MESH-Z=     1   ZL=  0.00000E+00   ZC=  3.60000E+00   ZR=  7.20000E+00   NZL=  18   NZR=  18   DZMN=  2.00000E-01

 IBAR= 11   JBAR=  1   KBAR= 36
```

TABLE V

EXAMPLE PROBLEM DATA PRINT

VOF3D TANK DRAIN J=2 PLANE ONLY 11/13/86 ?0?0?0?0?0?0?0?0 VOF3D7 03/12/87 11:06:53

ITER= 82 TIME= 2.04000E-04 DELT= 2.04000E-04 CYCLE= 1

I	J	K.	U	V	W	P	PS	F	NF	NFP	NFS	NFO	BETA	PETA	D
1	2	2	4.3354E+01	0.0000E+00	-8.5091E+01	-2.1991E+05	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
1	2	3	1.2575E+01	0.0000E+00	-3.4791E+01	-1.3650E+05	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
1	2	4	4.2926E+00	0.0000E+00	-1.7620E+01	-1.0239E+05	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
1	2	5	1.7430E+00	0.0000E+00	-1.0648E+01	-8.5121E+04	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
1	2	6	8.2757E-01	0.0000E+00	-7.3382E+00	-7.4685E+04	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
1	2	7	4.4477E-01	0.0000E+00	-5.5592E+00	-6.7493E+04	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
1	2	8	2.6175E-01	0.0000E+00	-4.5122E+00	-6.2046E+04	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
1	2	9	1.6411E-01	0.0000E+00	-3.8558E+00	-5.7625E+04	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
1	2	10	1.0736E-01	0.0000E+00	-3.4265E+00	-5.3848E+04	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
1	2	11	7.2129E-02	0.0000E+00	-3.1379E+00	-5.0492E+04	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
1	2	12	4.9215E-02	0.0000E+00	-2.9410E+00	-4.7418E+04	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
1	2	13	3.3814E-02	0.0000E+00	-2.8057E+00	-4.4538E+04	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
1	2	14	2.3235E-02	0.0000E+00	-2.7127E+00	-4.1790E+04	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
1	2	15	1.5859E-02	0.0000E+00	-2.6492E+00	-3.9133E+04	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
1	2	16	1.0633E-02	0.0000E+00	-2.6066E+00	-3.6539E+04	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
1	2	17	6.8453E-03	0.0000E+00	-2.5793E+00	-3.3987E+04	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
1	2	18	3.9878E-03	0.0000E+00	-2.5633E+00	-3.1461E+04	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
1	2	19	1.6650E-03	0.0000E+00	-2.5566E+00	-2.8951E+04	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
1	2	20	4.2962E-04	0.0000E+00	-2.5583E+00	-2.6447E+04	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
1	2	21	2.5705E-03	0.0000E+00	-2.5685E+00	-2.3942E+04	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
1	2	22	5.0764E-03	0.0000E+00	-2.5887E+00	-2.1427E+04	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
1	2	23	8.2503E-03	0.0000E+00	-2.6216E+00	-1.8892E+04	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
1	2	24	1.2521E-02	0.0000E+00	-2.6716E+00	-1.6325E+04	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
1	2	25	1.8425E-02	0.0000E+00	-2.7453E+00	-1.3708E+04	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
1	2	26	2.6667E-02	0.0000E+00	-2.8519E+00	-1.1020E+04	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
1	2	27	3.8179E-02	0.0000E+00	-3.0046E+00	-8.2266E+03	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
1	2	28	5.4102E-02	0.0000E+00	-3.2210E+00	-5.2839E+03	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
1	2	29	7.5544E-02	0.0000E+00	-3.5231E+00	-2.1290E+03	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
1	2	30	7.5544E-02	0.0000E+00	-3.8253E+00	1.3221E+03	0.0000E+00	1.1147E-01	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
1	2	31	1.2422E-02	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
1	2	32	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
1	2	33	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
1	2	34	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
1	2	35	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
1	2	36	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
1	2	37	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
1	2	38	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
2	2	1	-8.6708E+01	0.0000E+00	-2.4979E+02	-2.1991E+05	0.0000E+00	1.0000E+00	0	0	0	0	1.000E+00	1.000E+00	0.000E+00
2	2	2	-8.6708E+01	0.0000E+00	-8.5091E+01	-2.1991E+05	0.0000E+00	1.0000E+00	0	0	0	0	1.210E+02	1.000E+00	2.124E-07
2	2	3	-2.5150E+01	0.0000E+00	-3.4791E+01	-1.3650E+05	0.0000E+00	1.0000E+00	0	0	0	0	9.000E+01	1.000E+00	-1.777E-05
2	2	4	-8.5852E+00	0.0000E+00	-1.7620E+01	-1.0239E+05	0.0000E+00	1.0000E+00	0	0	0	0	9.000E+01	1.000E+00	-1.212E-04
2	2	5	-3.4861E+00	0.0000E+00	-1.0648E+01	-8.5121E+04	0.0000E+00	1.0000E+00	0	0	0	0	9.000E+01	1.000E+00	-2.212E-04
2	2	6	-1.6551E+00	0.0000E+00	-7.3382E+00	-7.4685E+04	0.0000E+00	1.0000E+00	0	0	0	0	9.000E+01	1.000E+00	-5.144E-04
2	2	7	-8.8954E-01	0.0000E+00	-5.5592E+00	-6.7493E+04	0.0000E+00	1.0000E+00	0	0	0	0	9.000E+01	1.000E+00	-3.240E-04
2	2	8	-5.2349E-01	0.0000E+00	-4.5122E+00	-6.2046E+04	0.0000E+00	1.0000E+00	0	0	0	0	9.000E+01	1.000E+00	-9.607E-05
2	2	9	-3.2821E-01	0.0000E+00	-3.8558E+00	-5.7625E+04	0.0000E+00	1.0000E+00	0	0	0	0	9.000E+01	1.000E+00	6.351E-06
2	2	10	-2.1471E-01	0.0000E+00	-3.4265E+00	-5.3848E+04	0.0000E+00	1.0000E+00	0	0	0	0	9.000E+01	1.000E+00	-3.225E-04
2	2	11	-1.4426E-01	0.0000E+00	-3.1379E+00	-5.0492E+04	0.0000E+00	1.0000E+00	0	0	0	0	9.000E+01	1.000E+00	1.003E-04
2	2	12	-9.8430E-02	0.0000E+00	-2.9410E+00	-4.7418E+04	0.0000E+00	1.0000E+00	0	0	0	0	9.000E+01	1.000E+00	2.017E-04
2	2	13	-6.7628E-02	0.0000E+00	-2.8057E+00	-4.4538E+04	0.0000E+00	1.0000E+00	0	0	0	0	9.000E+01	1.000E+00	2.240E-04
2	2	14	-4.6469E-02	0.0000E+00	-2.7127E+00	-4.1790E+04	0.0000E+00	1.0000E+00	0	0	0	0	9.000E+01	1.000E+00	2.521E-04
2	2	15	-3.1719E-02	0.0000E+00	-2.6492E+00	-3.9133E+04	0.0000E+00	1.0000E+00	0	0	0	0	9.000E+01	1.000E+00	3.545E-04
2	2	16	-2.1266E-02	0.0000E+00	-2.6066E+00	-3.6539E+04	0.0000E+00	1.0000E+00	0	0	0	0	9.000E+01	1.000E+00	2.104E-04
2	2	17	-1.3691E-02	0.0000E+00	-2.5793E+00	-3.3987E+04	0.0000E+00	1.0000E+00	0	0	0	0	9.000E+01	1.000E+00	5.721E-05
2	2	18	-7.9756E-03	0.0000E+00	-2.5633E+00	-3.1461E+04	0.0000E+00	1.0000E+00	0	0	0	0	9.000E+01	1.000E+00	-6.485E-05

TABLE V (Continued)

EXAMPLE PROBLEM DATA PRINT

/?

VOF3D TANK DRAIN J=2 PLANE ONLY 11/13/86 ?0?0?0?0?0?0?0?0 VOF3D7 03/12/87 11:06:53

ITER= 82 TIME= 2.04000E-04 DELT= 2.04000E-04 CYCLE= 1

I	J	K,	U	V	W	P	PS	F	NF	NFP	NFS	NFO	BETA	PETA	D
2	2	19	-3.3299E-03	0.0000E+00	-2.5566E+00	-2.8951E+04	0.0000E+00	1.0000E+00	0	0	0	0	9.000E+01	1.000E+00	1.819E-04
2	2	20	8.5924E-04	0.0000E+00	-2.5583E+00	-2.6447E+04	0.0000E+00	1.0000E+00	0	0	0	0	9.000E+01	1.000E+00	4.071E-04
2	2	21	5.1409E-03	0.0000E+00	-2.5685E+00	-2.3942E+04	0.0000E+00	1.0000E+00	0	0	0	0	9.000E+01	1.000E+00	1.853E-04
2	2	22	1.0153E-02	0.0000E+00	-2.5887E+00	-2.1427E+04	0.0000E+00	1.0000E+00	0	0	0	0	9.000E+01	1.000E+00	6.446E-04
2	2	23	1.6501E-02	0.0000E+00	-2.6216E+00	-1.8892E+04	0.0000E+00	1.0000E+00	0	0	0	0	9.000E+01	1.000E+00	3.576E-04
2	2	24	2.5042E-02	0.0000E+00	-2.6716E+00	-1.6325E+04	0.0000E+00	1.0000E+00	0	0	0	0	9.000E+01	1.000E+00	2.706E-04
2	2	25	3.6851E-02	0.0000E+00	-2.7453E+00	-1.3708E+04	0.0000E+00	1.0000E+00	0	0	0	0	9.000E+01	1.000E+00	2.259E-04
2	2	26	5.3334E-02	0.0000E+00	-2.8519E+00	-1.1020E+04	0.0000E+00	1.0000E+00	0	0	0	0	9.000E+01	1.000E+00	8.206E-05
2	2	27	7.6358E-02	0.0000E+00	-3.0046E+00	-8.2266E+03	0.0000E+00	1.0000E+00	0	0	0	0	9.000E+01	1.000E+00	2.430E-04
2	2	28	1.0820E-01	0.0000E+00	-3.2210E+00	-5.2839E+03	0.0000E+00	1.0000E+00	0	0	0	0	9.000E+01	1.000E+00	1.770E-04
2	2	29	1.5109E-01	0.0000E+00	-3.5231E+00	-2.1290E+03	0.0000E+00	1.0000E+00	0	0	0	0	9.000E+01	7.742E-01	8.343E-05
2	2	30	1.5109E-01	0.0000E+00	-3.8253E+00	1.3221E+03	-6.2259E+00	1.1147E-01	5	5	5	5	9.000E+01	1.635E+00	0.000E+00
2	2	31	2.4844E-02	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	8	8	0	8	9.000E+01	1.000E+00	0.000E+00
2	2	32	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	8	8	0	8	9.000E+01	1.000E+00	0.000E+00
2	2	33	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	8	8	0	8	9.000E+01	1.000E+00	0.000E+00
2	2	34	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	8	8	0	8	9.000E+01	1.000E+00	0.000E+00
2	2	35	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	8	8	0	8	9.000E+01	1.000E+00	0.000E+00
2	2	36	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	8	8	0	8	9.000E+01	1.000E+00	0.000E+00
2	2	37	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	8	8	0	8	1.200E+02	1.000E+00	0.000E+00
2	2	38	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
3	2	1	-2.9489E+01	0.0000E+00	0.0000E+00	-1.3491E+05	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
3	2	2	-2.9489E+01	0.0000E+00	-2.3527E+01	-1.3491E+05	0.0000E+00	1.0000E+00	0	0	0	0	1.244E+02	1.000E+00	6.924E-06
3	2	3	-1.6551E+01	0.0000E+00	-1.8226E+01	-1.1184E+05	0.0000E+00	1.0000E+00	0	0	0	0	9.500E+01	1.000E+00	-2.159E-05
3	2	4	-8.5713E+00	0.0000E+00	-1.2521E+01	-9.3977E+04	0.0000E+00	1.0000E+00	0	0	0	0	9.500E+01	1.000E+00	-5.199E-05
3	2	5	-4.5210E+00	0.0000E+00	-8.8175E+00	-8.1704E+04	0.0000E+00	1.0000E+00	0	0	0	0	9.500E+01	1.000E+00	-3.466E-04
3	2	6	-2.5113E+00	0.0000E+00	-6.5726E+00	-7.3062E+04	0.0000E+00	1.0000E+00	0	0	0	0	9.500E+01	1.000E+00	-2.822E-04
3	2	7	-1.4794E+00	0.0000E+00	-5.1932E+00	-6.6621E+04	0.0000E+00	1.0000E+00	0	0	0	0	9.500E+01	1.000E+00	-2.999E-04
3	2	8	-9.1890E-01	0.0000E+00	-4.3170E+00	-6.1533E+04	0.0000E+00	1.0000E+00	0	0	0	0	9.500E+01	1.000E+00	-9.296E-05
3	2	9	-5.9513E-01	0.0000E+00	-3.7423E+00	-5.7303E+04	0.0000E+00	1.0000E+00	0	0	0	0	9.500E+01	1.000E+00	-2.387E-04
3	2	10	-3.9709E-01	0.0000E+00	-3.3560E+00	-5.3637E+04	0.0000E+00	1.0000E+00	0	0	0	0	9.500E+01	1.000E+00	-4.265E-07
3	2	11	-2.7006E-01	0.0000E+00	-3.0921E+00	-5.0350E+04	0.0000E+00	1.0000E+00	0	0	0	0	9.500E+01	1.000E+00	2.643E-05
3	2	12	-1.8559E-01	0.0000E+00	-2.9102E+00	-4.7322E+04	0.0000E+00	1.0000E+00	0	0	0	0	9.500E+01	1.000E+00	1.789E-04
3	2	13	-1.2803E-01	0.0000E+00	-2.7846E+00	-4.4472E+04	0.0000E+00	1.0000E+00	0	0	0	0	9.500E+01	1.000E+00	2.146E-04
3	2	14	-8.8149E-02	0.0000E+00	-2.6980E+00	-4.1744E+04	0.0000E+00	1.0000E+00	0	0	0	0	9.500E+01	1.000E+00	1.339E-04
3	2	15	-6.0189E-02	0.0000E+00	-2.6388E+00	-3.9102E+04	0.0000E+00	1.0000E+00	0	0	0	0	9.500E+01	1.000E+00	5.320E-04
3	2	16	-4.0359E-02	0.0000E+00	-2.5991E+00	-3.6518E+04	0.0000E+00	1.0000E+00	0	0	0	0	9.500E+01	1.000E+00	3.018E-04
3	2	17	-2.5982E-02	0.0000E+00	-2.5735E+00	-3.3973E+04	0.0000E+00	1.0000E+00	0	0	0	0	9.500E+01	1.000E+00	8.599E-05
3	2	18	-1.5126E-02	0.0000E+00	-2.5587E+00	-3.1453E+04	0.0000E+00	1.0000E+00	0	0	0	0	9.500E+01	1.000E+00	9.122E-05
3	2	19	-6.3288E-03	0.0000E+00	-2.5524E+00	-2.8947E+04	0.0000E+00	1.0000E+00	0	0	0	0	9.500E+01	1.000E+00	1.057E-04
3	2	20	1.6053E-03	0.0000E+00	-2.5540E+00	-2.6448E+04	0.0000E+00	1.0000E+00	0	0	0	0	9.500E+01	1.000E+00	1.154E-04
3	2	21	9.7651E-03	0.0000E+00	-2.5635E+00	-2.3947E+04	0.0000E+00	1.0000E+00	0	0	0	0	9.500E+01	1.000E+00	3.905E-04
3	2	22	1.9240E-02	0.0000E+00	-2.5823E+00	-2.1437E+04	0.0000E+00	1.0000E+00	0	0	0	0	9.500E+01	1.000E+00	2.199E-04
3	2	23	3.1330E-02	0.0000E+00	-2.6131E+00	-1.8908E+04	0.0000E+00	1.0000E+00	0	0	0	0	9.500E+01	1.000E+00	1.888E-04
3	2	24	4.7619E-02	0.0000E+00	-2.6598E+00	-1.6349E+04	0.0000E+00	1.0000E+00	0	0	0	0	9.500E+01	1.000E+00	1.716E-04
3	2	25	7.0189E-02	0.0000E+00	-2.7288E+00	-1.3744E+04	0.0000E+00	1.0000E+00	0	0	0	0	9.500E+01	1.000E+00	1.736E-04
3	2	26	1.0179E-01	0.0000E+00	-2.8289E+00	-1.1072E+04	0.0000E+00	1.0000E+00	0	0	0	0	9.500E+01	1.000E+00	2.693E-04
3	2	27	1.4612E-01	0.0000E+00	-2.9728E+00	-8.3015E+03	0.0000E+00	1.0000E+00	0	0	0	0	9.500E+01	1.000E+00	3.558E-04
3	2	28	2.0814E-01	0.0000E+00	-3.1781E+00	-5.3900E+03	0.0000E+00	1.0000E+00	0	0	0	0	9.500E+01	1.000E+00	2.661E-04
3	2	29	2.9468E-01	0.0000E+00	-3.4702E+00	-2.2771E+03	0.0000E+00	1.0000E+00	0	0	0	0	9.500E+01	8.034E-01	1.949E-04
3	2	30	2.9468E-01	0.0000E+00	-3.7624E+00	1.1221E+03	-6.3405E+00	1.6443E-01	5	5	5	5	9.500E+01	1.505E+00	0.000E+00
3	2	31	8.7998E-02	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	8	8	0	8	9.500E+01	1.000E+00	0.000E+00
3	2	32	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	8	8	0	8	9.500E+01	1.000E+00	0.000E+00
3	2	33	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	8	8	0	8	9.500E+01	1.000E+00	0.000E+00
3	2	34	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	8	8	0	8	9.500E+01	1.000E+00	0.000E+00
3	2	35	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	8	8	0	8	9.500E+01	1.000E+00	0.000E+00

TABLE V (Continued)

EXAMPLE PROBLEM DATA PRINT

/?

VOF3D TANK DRAIN J=2 PLANE ONLY 11/13/86 ?0?0?0?0?0?0?0?0 VOF3D7 03/12/87 11:06:53

ITER= 82 TIME= 2.04000E-04 DELT= 2.04000E-04 CYCLE= 1

I	J	K,	U	V	W	P	PS	F	NF	NFP	NFS	NFO	BETA	PETA	D
3	2	36	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	8	8	0	8	9.500E+01	1.000E+00	0.000E+00
3	2	37	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	8	8	0	8	1.291E+02	1.000E+00	0.000E+00
3	2	38	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
4	2	1	-1.2113E+01	0.0000E+00	0.0000E+00	-1.0057E+05	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
4	2	2	-1.2113E+01	0.0000E+00	-8.1633E+00	-1.0057E+05	0.0000E+00	1.0000E+00	0	0	0	0	1.320E+02	1.000E+00	5.984E-05
4	2	3	-9.1646E+00	0.0000E+00	-8.7508E+00	-9.2574E+04	0.0000E+00	1.0000E+00	0	0	0	0	1.142E+02	1.000E+00	1.938E-05
4	2	4	-6.2172E+00	0.0000E+00	-7.7116E+00	-8.3998E+04	0.0000E+00	1.0000E+00	0	0	0	0	1.142E+02	1.000E+00	-7.044E-06
4	2	5	-4.0813E+00	0.0000E+00	-6.4309E+00	-7.6440E+04	0.0000E+00	1.0000E+00	0	0	0	0	1.142E+02	1.000E+00	-8.791E-05
4	2	6	-2.6747E+00	0.0000E+00	-5.3473E+00	-7.0138E+04	0.0000E+00	1.0000E+00	0	0	0	0	1.142E+02	1.000E+00	-1.858E-04
4	2	7	-1.7742E+00	0.0000E+00	-4.5276E+00	-6.4899E+04	0.0000E+00	1.0000E+00	0	0	0	0	1.142E+02	1.000E+00	-1.747E-04
4	2	8	-1.1961E+00	0.0000E+00	-3.9325E+00	-6.0463E+04	0.0000E+00	1.0000E+00	0	0	0	0	1.142E+02	1.000E+00	-1.106E-04
4	2	9	-8.1841E-01	0.0000E+00	-3.5071E+00	-5.6611E+04	0.0000E+00	1.0000E+00	0	0	0	0	1.142E+02	1.000E+00	-3.049E-05
4	2	10	-5.6598E-01	0.0000E+00	-3.2052E+00	-5.3175E+04	0.0000E+00	1.0000E+00	0	0	0	0	1.142E+02	1.000E+00	-1.074E-04
4	2	11	-3.9364E-01	0.0000E+00	-2.9918E+00	-5.0036E+04	0.0000E+00	1.0000E+00	0	0	0	0	1.142E+02	1.000E+00	5.295E-05
4	2	12	-2.7411E-01	0.0000E+00	-2.8419E+00	-4.7106E+04	0.0000E+00	1.0000E+00	0	0	0	0	1.142E+02	1.000E+00	1.242E-04
4	2	13	-1.9042E-01	0.0000E+00	-2.7372E+00	-4.4322E+04	0.0000E+00	1.0000E+00	0	0	0	0	1.142E+02	1.000E+00	1.460E-04
4	2	14	-1.3153E-01	0.0000E+00	-2.6648E+00	-4.1642E+04	0.0000E+00	1.0000E+00	0	0	0	0	1.142E+02	1.000E+00	4.052E-05
4	2	15	-8.9900E-02	0.0000E+00	-2.6152E+00	-3.9032E+04	0.0000E+00	1.0000E+00	0	0	0	0	1.142E+02	1.000E+00	1.253E-04
4	2	16	-6.0241E-02	0.0000E+00	-2.5820E+00	-3.6471E+04	0.0000E+00	1.0000E+00	0	0	0	0	1.142E+02	1.000E+00	2.232E-04
4	2	17	-3.8731E-02	0.0000E+00	-2.5606E+00	-3.3943E+04	0.0000E+00	1.0000E+00	0	0	0	0	1.142E+02	1.000E+00	1.939E-04
4	2	18	-2.2519E-02	0.0000E+00	-2.5482E+00	-3.1435E+04	0.0000E+00	1.0000E+00	0	0	0	0	1.142E+02	1.000E+00	2.169E-04
4	2	19	-9.4191E-03	0.0000E+00	-2.5430E+00	-2.8940E+04	0.0000E+00	1.0000E+00	0	0	0	0	1.142E+02	1.000E+00	1.890E-04
4	2	20	2.3792E-03	0.0000E+00	-2.5443E+00	-2.6450E+04	0.0000E+00	1.0000E+00	0	0	0	0	1.142E+02	1.000E+00	1.378E-04
4	2	21	1.4514E-02	0.0000E+00	-2.5522E+00	-2.3958E+04	0.0000E+00	1.0000E+00	0	0	0	0	1.142E+02	1.000E+00	8.696E-05
4	2	22	2.8670E-02	0.0000E+00	-2.5680E+00	-2.1459E+04	0.0000E+00	1.0000E+00	0	0	0	0	1.142E+02	1.000E+00	1.873E-04
4	2	23	4.6784E-02	0.0000E+00	-2.5937E+00	-1.8944E+04	0.0000E+00	1.0000E+00	0	0	0	0	1.142E+02	1.000E+00	1.102E-04
4	2	24	7.1297E-02	0.0000E+00	-2.6330E+00	-1.6405E+04	0.0000E+00	1.0000E+00	0	0	0	0	1.142E+02	1.000E+00	1.997E-04
4	2	25	1.0541E-01	0.0000E+00	-2.6913E+00	-1.3826E+04	0.0000E+00	1.0000E+00	0	0	0	0	1.142E+02	1.000E+00	1.218E-04
4	2	26	1.5341E-01	0.0000E+00	-2.7763E+00	-1.1191E+04	0.0000E+00	1.0000E+00	0	0	0	0	1.142E+02	1.000E+00	1.179E-04
4	2	27	2.2111E-01	0.0000E+00	-2.8991E+00	-8.4716E+03	0.0000E+00	1.0000E+00	0	0	0	0	1.142E+02	1.000E+00	1.695E-04
4	2	28	3.1631E-01	0.0000E+00	-3.0753E+00	-5.6323E+03	0.0000E+00	1.0000E+00	0	0	0	0	1.142E+02	1.000E+00	1.220E-04
4	2	29	4.4931E-01	0.0000E+00	-3.3262E+00	-2.6202E+03	0.0000E+00	1.0000E+00	0	0	0	0	1.142E+02	8.719E-01	8.472E-05
4	2	30	4.4931E-01	0.0000E+00	-3.5771E+00	6.3783E+02	-6.9015E+00	2.9862E-01	5	5	5	5	1.142E+02	1.252E+00	0.000E+00
4	2	31	2.4801E-01	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	8	8	0	8	1.142E+02	1.000E+00	0.000E+00
4	2	32	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	8	8	0	8	1.142E+02	1.000E+00	0.000E+00
4	2	33	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	8	8	0	8	1.142E+02	1.000E+00	0.000E+00
4	2	34	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	8	8	0	8	1.142E+02	1.000E+00	0.000E+00
4	2	35	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	8	8	0	8	1.142E+02	1.000E+00	0.000E+00
4	2	36	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	8	8	0	8	1.142E+02	1.000E+00	0.000E+00
4	2	37	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	8	8	0	8	1.673E+02	1.000E+00	0.000E+00
4	2	38	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
5	2	1	0.0000E+00	0.0000E+00	0.0000E+00	-8.4987E+04	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
5	2	2	0.0000E+00	0.0000E+00	-4.2933E+00	-8.4987E+04	0.0000E+00	1.0000E+00	0	0	0	0	5.857E+01	1.000E+00	2.663E-04
5	2	3	-5.8446E+00	0.0000E+00	-4.8824E+00	-8.0781E+04	0.0000E+00	1.0000E+00	0	0	0	0	1.192E+02	1.000E+00	1.750E-04
5	2	4	-4.5089E+00	0.0000E+00	-4.9082E+00	-7.5998E+04	0.0000E+00	1.0000E+00	0	0	0	0	1.093E+02	1.000E+00	9.686E-05
5	2	5	-3.3280E+00	0.0000E+00	-4.5847E+00	-7.1189E+04	0.0000E+00	1.0000E+00	0	0	0	0	1.093E+02	1.000E+00	2.279E-05
5	2	6	-2.4051E+00	0.0000E+00	-4.1654E+00	-6.6697E+04	0.0000E+00	1.0000E+00	0	0	0	0	1.093E+02	1.000E+00	-4.481E-05
5	2	7	-1.7235E+00	0.0000E+00	-3.7688E+00	-6.2616E+04	0.0000E+00	1.0000E+00	0	0	0	0	1.093E+02	1.000E+00	-1.048E-04
5	2	8	-1.2318E+00	0.0000E+00	-3.4368E+00	-5.8924E+04	0.0000E+00	1.0000E+00	0	0	0	0	1.093E+02	1.000E+00	-6.743E-05
5	2	9	-8.7932E-01	0.0000E+00	-3.1758E+00	-5.5558E+04	0.0000E+00	1.0000E+00	0	0	0	0	1.093E+02	1.000E+00	-3.397E-05
5	2	10	-6.2583E-01	0.0000E+00	-2.9790E+00	-5.2447E+04	0.0000E+00	1.0000E+00	0	0	0	0	1.093E+02	1.000E+00	3.005E-05
5	2	11	-4.4297E-01	0.0000E+00	-2.8349E+00	-4.9529E+04	0.0000E+00	1.0000E+00	0	0	0	0	1.093E+02	1.000E+00	6.068E-05
5	2	12	-3.1124E-01	0.0000E+00	-2.7320E+00	-4.6753E+04	0.0000E+00	1.0000E+00	0	0	0	0	1.093E+02	1.000E+00	1.119E-04
5	2	13	-2.1690E-01	0.0000E+00	-2.6599E+00	-4.4077E+04	0.0000E+00	1.0000E+00	0	0	0	0	1.093E+02	1.000E+00	1.004E-04
5	2	14	-1.4976E-01	0.0000E+00	-2.6101E+00	-4.1473E+04	0.0000E+00	1.0000E+00	0	0	0	0	1.093E+02	1.000E+00	7.243E-05

TABLE V (Continued)

EXAMPLE PROBLEM DATA PRINT

/?

VOF3D TANK DRAIN J=2 PLANE ONLY 11/13/86 ?0?0?0?0?0?0?0?0 VOF3D7 03/12/87 11:06:53

ITER= 82 TIME= 2.04000E-04 DELT= 2.04000E-04 CYCLE= 1

I	J	K,	U	V	W	P	PS	F	NF	NFP	NFS	NFO	BETA	PETA	D
5	2	15	-1.0215E-01	0.0000E+00	-2.5763E+00	-3.8917E+04	0.0000E+00	1.0000E+00	0	0	0	0	1.093E+02	1.000E+00	1.959E-04
5	2	16	-6.8282E-02	0.0000E+00	-2.5538E+00	-3.6394E+04	0.0000E+00	1.0000E+00	0	0	0	0	1.093E+02	1.000E+00	1.280E-04
5	2	17	-4.3793E-02	0.0000E+00	-2.5394E+00	-3.3893E+04	0.0000E+00	1.0000E+00	0	0	0	0	1.093E+02	1.000E+00	1.651E-04
5	2	18	-2.5418E-02	0.0000E+00	-2.5310E+00	-3.1406E+04	0.0000E+00	1.0000E+00	0	0	0	0	1.093E+02	1.000E+00	1.366E-04
5	2	19	-1.0620E-02	0.0000E+00	-2.5275E+00	-2.8928E+04	0.0000E+00	1.0000E+00	0	0	0	0	1.093E+02	1.000E+00	2.052E-04
5	2	20	2.6647E-03	0.0000E+00	-2.5284E+00	-2.6453E+04	0.0000E+00	1.0000E+00	0	0	0	0	1.093E+02	1.000E+00	1.187E-04
5	2	21	1.6332E-02	0.0000E+00	-2.5337E+00	-2.3977E+04	0.0000E+00	1.0000E+00	0	0	0	0	1.093E+02	1.000E+00	5.803E-05
5	2	22	3.2330E-02	0.0000E+00	-2.5442E+00	-2.1496E+04	0.0000E+00	1.0000E+00	0	0	0	0	1.093E+02	1.000E+00	5.823E-05
5	2	23	5.2904E-02	0.0000E+00	-2.5616E+00	-1.9005E+04	0.0000E+00	1.0000E+00	0	0	0	0	1.093E+02	1.000E+00	2.744E-05
5	2	24	8.0918E-02	0.0000E+00	-2.5883E+00	-1.6496E+04	0.0000E+00	1.0000E+00	0	0	0	0	1.093E+02	1.000E+00	6.624E-05
5	2	25	1.2019E-01	0.0000E+00	-2.6283E+00	-1.3962E+04	0.0000E+00	1.0000E+00	0	0	0	0	1.093E+02	1.000E+00	6.110E-05
5	2	26	1.7592E-01	0.0000E+00	-2.6874E+00	-1.1388E+04	0.0000E+00	1.0000E+00	0	0	0	0	1.093E+02	1.000E+00	1.233E-04
5	2	27	2.5515E-01	0.0000E+00	-2.7742E+00	-8.7561E+03	0.0000E+00	1.0000E+00	0	0	0	0	1.093E+02	1.000E+00	1.084E-04
5	2	28	3.6771E-01	0.0000E+00	-2.9008E+00	-6.0393E+03	0.0000E+00	1.0000E+00	0	0	0	0	1.093E+02	1.000E+00	6.526E-05
5	2	29	5.2944E-01	0.0000E+00	-3.0872E+00	-3.1984E+03	0.0000E+00	1.0000E+00	0	0	0	0	1.093E+02	1.028E+00	5.238E-05
5	2	30	5.2944E-01	0.0000E+00	-3.2737E+00	-1.7463E+02	-7.3962E+00	5.5198E-01	5	5	5	5	1.093E+02	9.506E-01	0.000E+00
5	2	31	4.7114E-01	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	8	8	0	8	1.093E+02	1.000E+00	0.000E+00
5	2	32	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	8	8	0	8	1.093E+02	1.000E+00	0.000E+00
5	2	33	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	8	8	0	8	1.093E+02	1.000E+00	0.000E+00
5	2	34	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	8	8	0	8	1.093E+02	1.000E+00	0.000E+00
5	2	35	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	8	8	0	8	1.093E+02	1.000E+00	0.000E+00
5	2	36	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	8	8	0	8	1.093E+02	1.000E+00	0.000E+00
5	2	37	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	8	8	0	8	1.570E+02	1.000E+00	0.000E+00
5	2	38	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
6	2	1	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
6	2	2	0.0000E+00	0.0000E+00	0.0000E+00	-7.9482E+04	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
6	2	3	-4.0005E+00	0.0000E+00	-3.2962E+00	-7.3977E+04	0.0000E+00	1.0000E+00	0	0	0	0	1.080E+02	1.000E+00	1.056E-04
6	2	4	-3.4138E+00	0.0000E+00	-3.5059E+00	-7.0748E+04	0.0000E+00	1.0000E+00	0	0	0	0	1.001E+02	1.000E+00	1.493E-04
6	2	5	-2.6943E+00	0.0000E+00	-3.4887E+00	-6.7314E+04	0.0000E+00	1.0000E+00	0	0	0	0	1.001E+02	1.000E+00	7.449E-05
6	2	6	-2.0691E+00	0.0000E+00	-3.3561E+00	-6.3897E+04	0.0000E+00	1.0000E+00	0	0	0	0	1.001E+02	1.000E+00	-4.565E-05
6	2	7	-1.5599E+00	0.0000E+00	-3.1850E+00	-6.0609E+04	0.0000E+00	1.0000E+00	0	0	0	0	1.001E+02	1.000E+00	-8.350E-05
6	2	8	-1.1602E+00	0.0000E+00	-3.0181E+00	-5.7490E+04	0.0000E+00	1.0000E+00	0	0	0	0	1.001E+02	1.000E+00	-2.134E-05
6	2	9	-8.5277E-01	0.0000E+00	-2.8748E+00	-5.4534E+04	0.0000E+00	1.0000E+00	0	0	0	0	1.001E+02	1.000E+00	-1.793E-05
6	2	10	-6.1806E-01	0.0000E+00	-2.7618E+00	-5.1718E+04	0.0000E+00	1.0000E+00	0	0	0	0	1.001E+02	1.000E+00	9.625E-06
6	2	11	-4.4087E-01	0.0000E+00	-2.6785E+00	-4.9014E+04	0.0000E+00	1.0000E+00	0	0	0	0	1.001E+02	1.000E+00	2.692E-05
6	2	12	-3.0971E-01	0.0000E+00	-2.6200E+00	-4.6391E+04	0.0000E+00	1.0000E+00	0	0	0	0	1.001E+02	1.000E+00	3.709E-05
6	2	13	-2.1483E-01	0.0000E+00	-2.5802E+00	-4.3825E+04	0.0000E+00	1.0000E+00	0	0	0	0	1.001E+02	1.000E+00	1.046E-04
6	2	14	-1.4742E-01	0.0000E+00	-2.5536E+00	-4.1298E+04	0.0000E+00	1.0000E+00	0	0	0	0	1.001E+02	1.000E+00	1.265E-04
6	2	15	-9.9950E-02	0.0000E+00	-2.5361E+00	-3.8798E+04	0.0000E+00	1.0000E+00	0	0	0	0	1.001E+02	1.000E+00	6.272E-05
6	2	16	-6.6458E-02	0.0000E+00	-2.5247E+00	-3.6314E+04	0.0000E+00	1.0000E+00	0	0	0	0	1.001E+02	1.000E+00	7.757E-05
6	2	17	-4.2454E-02	0.0000E+00	-2.5175E+00	-3.3842E+04	0.0000E+00	1.0000E+00	0	0	0	0	1.001E+02	1.000E+00	7.920E-05
6	2	18	-2.4573E-02	0.0000E+00	-2.5135E+00	-3.1377E+04	0.0000E+00	1.0000E+00	0	0	0	0	1.001E+02	1.000E+00	1.053E-04
6	2	19	-1.0262E-02	0.0000E+00	-2.5117E+00	-2.8916E+04	0.0000E+00	1.0000E+00	0	0	0	0	1.001E+02	1.000E+00	8.470E-05
6	2	20	2.5539E-03	0.0000E+00	-2.5121E+00	-2.6456E+04	0.0000E+00	1.0000E+00	0	0	0	0	1.001E+02	1.000E+00	1.195E-04
6	2	21	1.5730E-02	0.0000E+00	-2.5147E+00	-2.3996E+04	0.0000E+00	1.0000E+00	0	0	0	0	1.001E+02	1.000E+00	1.118E-04
6	2	22	3.1190E-02	0.0000E+00	-2.5198E+00	-2.1534E+04	0.0000E+00	1.0000E+00	0	0	0	0	1.001E+02	1.000E+00	1.394E-05
6	2	23	5.1194E-02	0.0000E+00	-2.5283E+00	-1.9066E+04	0.0000E+00	1.0000E+00	0	0	0	0	1.001E+02	1.000E+00	-6.668E-06
6	2	24	7.8638E-02	0.0000E+00	-2.5416E+00	-1.6590E+04	0.0000E+00	1.0000E+00	0	0	0	0	1.001E+02	1.000E+00	7.563E-05
6	2	25	1.1748E-01	0.0000E+00	-2.5621E+00	-1.4102E+04	0.0000E+00	1.0000E+00	0	0	0	0	1.001E+02	1.000E+00	6.052E-05
6	2	26	1.7321E-01	0.0000E+00	-2.5933E+00	-1.1593E+04	0.0000E+00	1.0000E+00	0	0	0	0	1.001E+02	1.000E+00	2.976E-05
6	2	27	2.5309E-01	0.0000E+00	-2.6405E+00	-9.0532E+03	0.0000E+00	1.0000E+00	0	0	0	0	1.001E+02	1.000E+00	5.518E-05
6	2	28	3.6505E-01	0.0000E+00	-2.7087E+00	-6.4674E+03	0.0000E+00	1.0000E+00	0	0	0	0	1.001E+02	1.000E+00	6.996E-05
6	2	29	5.0832E-01	0.0000E+00	-2.7900E+00	-3.8147E+03	0.0000E+00	1.0000E+00	0	0	0	0	1.001E+02	1.100E+00	6.540E-05
6	2	30	5.0832E-01	0.0000E+00	-2.8712E+00	-1.0824E+03	-8.6578E+00	8.8990E-01	5	5	5	5	1.001E+02	7.195E-01	0.000E+00
6	2	31	5.0832E-01	0.0000E+00	-8.4349E-01	0.0000E+00	0.0000E+00	0.0000E+00	8	8	0	8	1.001E+02	1.000E+00	0.000E+00

TABLE V (Continued)

EXAMPLE PROBLEM DATA PRINT

/?

VOF3D TANK DRAIN J=2 PLANE ONLY 11/13/86 ?0?0?0?0?0?0?0?0 VOF3D7 03/12/87 11:06:53

ITER= 82 TIME= 2.04000E-04 DELT= 2.04000E-04 CYCLE= 1

I	J	K,	U	V	W	P	PS	F	NF	NFP	NFS	NFO	BETA	PETA	D
6	2	32	1.5914E-01	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	8	8	0	8	1.001E+02	1.000E+00	0.000E+00
6	2	33	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	8	8	0	8	1.001E+02	1.000E+00	0.000E+00
6	2	34	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	8	8	0	8	1.001E+02	1.000E+00	0.000E+00
6	2	35	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	8	8	0	8	1.001E+02	1.000E+00	0.000E+00
6	2	36	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	8	8	0	8	1.001E+02	1.000E+00	0.000E+00
6	2	37	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	8	8	0	8	1.387E+02	1.000E+00	0.000E+00
6	2	38	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
7	2	1	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
7	2	2	0.0000E+00	0.0000E+00	0.0000E+00	-6.9810E+04	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
7	2	3	0.0000E+00	0.0000E+00	-2.6729E+00	-6.9810E+04	0.0000E+00	1.0000E+00	0	0	0	0	2.006E+01	1.000E+00	6.619E-04
7	2	4	-2.6137E+00	0.0000E+00	-2.7414E+00	-6.7192E+04	0.0000E+00	1.0000E+00	0	0	0	0	1.082E+02	1.000E+00	1.224E-04
7	2	5	-2.2183E+00	0.0000E+00	-2.8244E+00	-6.4507E+04	0.0000E+00	1.0000E+00	0	0	0	0	8.960E+01	1.000E+00	1.200E-04
7	2	6	-1.7713E+00	0.0000E+00	-2.8150E+00	-6.1741E+04	0.0000E+00	1.0000E+00	0	0	0	0	8.960E+01	1.000E+00	1.143E-05
7	2	7	-1.3796E+00	0.0000E+00	-2.7602E+00	-5.8984E+04	0.0000E+00	1.0000E+00	0	0	0	0	8.960E+01	1.000E+00	-4.309E-05
7	2	8	-1.0519E+00	0.0000E+00	-2.6915E+00	-5.6281E+04	0.0000E+00	1.0000E+00	0	0	0	0	8.960E+01	1.000E+00	-6.416E-05
7	2	9	-7.8764E-01	0.0000E+00	-2.6254E+00	-5.3646E+04	0.0000E+00	1.0000E+00	0	0	0	0	8.960E+01	1.000E+00	2.528E-05
7	2	10	-5.7452E-01	0.0000E+00	-2.5735E+00	-5.1075E+04	0.0000E+00	1.0000E+00	0	0	0	0	8.960E+01	1.000E+00	-9.440E-06
7	2	11	-4.0744E-01	0.0000E+00	-2.5391E+00	-4.8554E+04	0.0000E+00	1.0000E+00	0	0	0	0	8.960E+01	1.000E+00	-9.398E-05
7	2	12	-2.8230E-01	0.0000E+00	-2.5192E+00	-4.6068E+04	0.0000E+00	1.0000E+00	0	0	0	0	8.960E+01	1.000E+00	-2.613E-05
7	2	13	-1.9283E-01	0.0000E+00	-2.5086E+00	-4.3601E+04	0.0000E+00	1.0000E+00	0	0	0	0	8.960E+01	1.000E+00	6.013E-05
7	2	14	-1.3056E-01	0.0000E+00	-2.5032E+00	-4.1145E+04	0.0000E+00	1.0000E+00	0	0	0	0	8.960E+01	1.000E+00	7.239E-05
7	2	15	-8.7597E-02	0.0000E+00	-2.5005E+00	-3.8694E+04	0.0000E+00	1.0000E+00	0	0	0	0	8.960E+01	1.000E+00	1.626E-04
7	2	16	-5.7809E-02	0.0000E+00	-2.4992E+00	-3.6245E+04	0.0000E+00	1.0000E+00	0	0	0	0	8.960E+01	1.000E+00	4.172E-05
7	2	17	-3.6736E-02	0.0000E+00	-2.4986E+00	-3.3798E+04	0.0000E+00	1.0000E+00	0	0	0	0	8.960E+01	1.000E+00	8.492E-05
7	2	18	-2.1196E-02	0.0000E+00	-2.4983E+00	-3.1351E+04	0.0000E+00	1.0000E+00	0	0	0	0	8.960E+01	1.000E+00	6.355E-05
7	2	19	-8.8368E-03	0.0000E+00	-2.4981E+00	-2.8905E+04	0.0000E+00	1.0000E+00	0	0	0	0	8.960E+01	1.000E+00	9.602E-05
7	2	20	2.1825E-03	0.0000E+00	-2.4981E+00	-2.6459E+04	0.0000E+00	1.0000E+00	0	0	0	0	8.960E+01	1.000E+00	5.124E-05
7	2	21	1.3508E-02	0.0000E+00	-2.4982E+00	-2.4012E+04	0.0000E+00	1.0000E+00	0	0	0	0	8.960E+01	1.000E+00	4.564E-05
7	2	22	2.6828E-02	0.0000E+00	-2.4985E+00	-2.1566E+04	0.0000E+00	1.0000E+00	0	0	0	0	8.960E+01	1.000E+00	1.227E-05
7	2	23	4.4151E-02	0.0000E+00	-2.4991E+00	-1.9120E+04	0.0000E+00	1.0000E+00	0	0	0	0	8.960E+01	1.000E+00	-1.240E-04
7	2	24	6.8114E-02	0.0000E+00	-2.5003E+00	-1.6672E+04	0.0000E+00	1.0000E+00	0	0	0	0	8.960E+01	1.000E+00	-2.799E-05
7	2	25	1.0244E-01	0.0000E+00	-2.5029E+00	-1.4224E+04	0.0000E+00	1.0000E+00	0	0	0	0	8.960E+01	1.000E+00	2.845E-06
7	2	26	1.5271E-01	0.0000E+00	-2.5085E+00	-1.1773E+04	0.0000E+00	1.0000E+00	0	0	0	0	8.960E+01	1.000E+00	8.980E-06
7	2	27	2.2772E-01	0.0000E+00	-2.5215E+00	-9.3168E+03	0.0000E+00	1.0000E+00	0	0	0	0	8.960E+01	1.000E+00	1.618E-04
7	2	28	3.4336E-01	0.0000E+00	-2.5565E+00	-6.8477E+03	0.0000E+00	1.0000E+00	0	0	0	0	8.960E+01	1.000E+00	1.088E-04
7	2	29	5.3674E-01	0.0000E+00	-2.6685E+00	-4.3442E+03	0.0000E+00	1.0000E+00	0	0	0	0	8.960E+01	1.000E+00	-6.301E-05
7	2	30	9.3262E-01	0.0000E+00	-2.1547E+00	-1.7310E+03	0.0000E+00	9.9887E-01	0	0	0	0	8.960E+01	9.046E-01	-7.118E-05
7	2	31	9.3262E-01	0.0000E+00	-2.6943E+00	3.7853E+02	-9.5724E+00	3.1307E-01	5	5	5	5	8.960E+01	1.231E+00	0.000E+00
7	2	32	7.5017E-01	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	8	8	0	8	8.960E+01	1.000E+00	0.000E+00
7	2	33	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	8	8	0	8	8.960E+01	1.000E+00	0.000E+00
7	2	34	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	8	8	0	8	8.960E+01	1.000E+00	0.000E+00
7	2	35	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	8	8	0	8	8.960E+01	1.000E+00	0.000E+00
7	2	36	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	8	8	0	8	8.960E+01	1.000E+00	0.000E+00
7	2	37	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	8	8	0	8	1.193E+02	1.000E+00	0.000E+00
7	2	38	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
8	2	1	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
8	2	2	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
8	2	3	0.0000E+00	0.0000E+00	0.0000E+00	-6.7300E+04	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
8	2	4	0.0000E+00	0.0000E+00	-2.3708E+00	-6.4790E+04	0.0000E+00	1.0000E+00	0	0	0	0	3.460E+01	1.000E+00	5.498E-04
8	2	5	-1.8192E+00	0.0000E+00	-2.4053E+00	-6.2469E+04	0.0000E+00	1.0000E+00	0	0	0	0	9.330E+01	1.000E+00	7.376E-05
8	2	6	-1.5329E+00	0.0000E+00	-2.4479E+00	-6.0113E+04	0.0000E+00	1.0000E+00	0	0	0	0	7.764E+01	1.000E+00	5.300E-05
8	2	7	-1.2171E+00	0.0000E+00	-2.4530E+00	-5.7716E+04	0.0000E+00	1.0000E+00	0	0	0	0	7.764E+01	1.000E+00	-4.347E-05
8	2	8	-9.3879E-01	0.0000E+00	-2.4438E+00	-5.5314E+04	0.0000E+00	1.0000E+00	0	0	0	0	7.764E+01	1.000E+00	6.197E-06
8	2	9	-7.1225E-01	0.0000E+00	-2.4256E+00	-5.2922E+04	0.0000E+00	1.0000E+00	0	0	0	0	7.764E+01	1.000E+00	-1.226E-05
8	2	10	-5.1579E-01	0.0000E+00	-2.4169E+00	-5.0546E+04	0.0000E+00	1.0000E+00	0	0	0	0	7.764E+01	1.000E+00	-2.506E-05

TABLE V (Continued)

EXAMPLE PROBLEM DATA PRINT

/?

VOF3D TANK DRAIN J=2 PLANE ONLY 11/13/86 ?0?0?0?0?0?0?0?0 VOF3D7 03/12/87 11:06:53

ITER= 82 TIME= 2.04000E-04 DELT= 2.04000E-04 CYCLE= 1

I	J	K,	U	V	W	P	PS	F	NF	NFP	NFS	NFO	BETA	PETA	D
8	2	11	-3.5664E-01	0.0000E+00	-2.4218E+00	-4.8180E+04	0.0000E+00	1.0000E+00	0	0	0	0	7.764E+01	1.000E+00	-6.956E-06
8	2	12	-2.3878E-01	0.0000E+00	-2.4353E+00	-4.5809E+04	0.0000E+00	1.0000E+00	0	0	0	0	7.764E+01	1.000E+00	-1.047E-04
8	2	13	-1.5843E-01	0.0000E+00	-2.4502E+00	-4.3424E+04	0.0000E+00	1.0000E+00	0	0	0	0	7.764E+01	1.000E+00	-1.671E-05
8	2	14	-1.0507E-01	0.0000E+00	-2.4629E+00	-4.1025E+04	0.0000E+00	1.0000E+00	0	0	0	0	7.764E+01	1.000E+00	1.702E-04
8	2	15	-6.9558E-02	0.0000E+00	-2.4726E+00	-3.8613E+04	0.0000E+00	1.0000E+00	0	0	0	0	7.764E+01	1.000E+00	6.558E-05
8	2	16	-4.5511E-02	0.0000E+00	-2.4794E+00	-3.6192E+04	0.0000E+00	1.0000E+00	0	0	0	0	7.764E+01	1.000E+00	-1.327E-05
8	2	17	-2.8763E-02	0.0000E+00	-2.4840E+00	-3.3764E+04	0.0000E+00	1.0000E+00	0	0	0	0	7.764E+01	1.000E+00	3.917E-06
8	2	18	-1.6537E-02	0.0000E+00	-2.4867E+00	-3.1332E+04	0.0000E+00	1.0000E+00	0	0	0	0	7.764E+01	1.000E+00	6.898E-05
8	2	19	-6.8828E-03	0.0000E+00	-2.4878E+00	-2.8897E+04	0.0000E+00	1.0000E+00	0	0	0	0	7.764E+01	1.000E+00	2.908E-05
8	2	20	1.6946E-03	0.0000E+00	-2.4875E+00	-2.6461E+04	0.0000E+00	1.0000E+00	0	0	0	0	7.764E+01	1.000E+00	9.402E-06
8	2	21	1.0508E-02	0.0000E+00	-2.4857E+00	-2.4025E+04	0.0000E+00	1.0000E+00	0	0	0	0	7.764E+01	1.000E+00	2.351E-05
8	2	22	2.0891E-02	0.0000E+00	-2.4823E+00	-2.1591E+04	0.0000E+00	1.0000E+00	0	0	0	0	7.764E+01	1.000E+00	4.358E-05
8	2	23	3.4433E-02	0.0000E+00	-2.4767E+00	-1.9160E+04	0.0000E+00	1.0000E+00	0	0	0	0	7.764E+01	1.000E+00	-1.253E-04
8	2	24	5.3279E-02	0.0000E+00	-2.4682E+00	-1.6735E+04	0.0000E+00	1.0000E+00	0	0	0	0	7.764E+01	1.000E+00	-5.860E-05
8	2	25	8.0473E-02	0.0000E+00	-2.4558E+00	-1.4318E+04	0.0000E+00	1.0000E+00	0	0	0	0	7.764E+01	1.000E+00	-6.669E-06
8	2	26	1.2063E-01	0.0000E+00	-2.4382E+00	-1.1914E+04	0.0000E+00	1.0000E+00	0	0	0	0	7.764E+01	1.000E+00	-1.313E-04
8	2	27	1.8094E-01	0.0000E+00	-2.4131E+00	-9.5261E+03	0.0000E+00	1.0000E+00	0	0	0	0	7.764E+01	1.000E+00	8.194E-06
8	2	28	2.7267E-01	0.0000E+00	-2.3752E+00	-7.1633E+03	0.0000E+00	1.0000E+00	0	0	0	0	7.764E+01	1.000E+00	-5.804E-05
8	2	29	4.1098E-01	0.0000E+00	-2.2974E+00	-4.8376E+03	0.0000E+00	1.0000E+00	0	0	0	0	7.764E+01	1.000E+00	-4.253E-05
8	2	30	5.9266E-01	0.0000E+00	-2.0150E+00	-2.5882E+03	0.0000E+00	1.0000E+00	0	0	0	0	7.764E+01	1.100E+00	-9.530E-05
8	2	31	5.9266E-01	0.0000E+00	-1.7326E+00	-6.1559E+02	-1.1403E+01	8.0438E-01	5	5	5	5	7.764E+01	7.666E-01	0.000E+00
8	2	32	5.9266E-01	0.0000E+00	-5.4748E-01	0.0000E+00	0.0000E+00	0.0000E+00	8	8	0	8	7.764E+01	1.000E+00	0.000E+00
8	2	33	2.2037E-01	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	8	8	0	8	7.764E+01	1.000E+00	0.000E+00
8	2	34	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	8	8	0	8	7.764E+01	1.000E+00	0.000E+00
8	2	35	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	8	8	0	8	7.764E+01	1.000E+00	0.000E+00
8	2	36	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	8	8	0	8	7.764E+01	1.000E+00	0.000E+00
8	2	37	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	8	8	0	8	9.898E+01	1.000E+00	0.000E+00
8	2	38	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
9	2	1	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
9	2	2	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
9	2	3	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
9	2	4	0.0000E+00	0.0000E+00	0.0000E+00	-6.2905E+04	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
9	2	5	0.0000E+00	0.0000E+00	-2.1727E+00	-6.1019E+04	0.0000E+00	1.0000E+00	0	0	0	0	3.424E+01	1.000E+00	4.801E-04
9	2	6	-1.3007E+00	0.0000E+00	-2.1912E+00	-5.8892E+04	0.0000E+00	1.0000E+00	0	0	0	0	7.781E+01	1.000E+00	6.789E-05
9	2	7	-1.0825E+00	0.0000E+00	-2.2269E+00	-5.6747E+04	0.0000E+00	1.0000E+00	0	0	0	0	6.435E+01	1.000E+00	-3.262E-05
9	2	8	-8.3122E-01	0.0000E+00	-2.2597E+00	-5.4567E+04	0.0000E+00	1.0000E+00	0	0	0	0	6.435E+01	1.000E+00	-1.839E-05
9	2	9	-6.4398E-01	0.0000E+00	-2.2660E+00	-5.2354E+04	0.0000E+00	1.0000E+00	0	0	0	0	6.435E+01	1.000E+00	-6.963E-05
9	2	10	-4.5413E-01	0.0000E+00	-2.2876E+00	-5.0136E+04	0.0000E+00	1.0000E+00	0	0	0	0	6.435E+01	1.000E+00	-2.913E-05
9	2	11	-2.9714E-01	0.0000E+00	-2.3260E+00	-4.7896E+04	0.0000E+00	1.0000E+00	0	0	0	0	6.435E+01	1.000E+00	3.361E-05
9	2	12	-1.8590E-01	0.0000E+00	-2.3700E+00	-4.5618E+04	0.0000E+00	1.0000E+00	0	0	0	0	6.435E+01	1.000E+00	-4.080E-05
9	2	13	-1.1787E-01	0.0000E+00	-2.4068E+00	-4.3298E+04	0.0000E+00	1.0000E+00	0	0	0	0	6.435E+01	1.000E+00	-1.518E-05
9	2	14	-7.6178E-02	0.0000E+00	-2.4340E+00	-4.0941E+04	0.0000E+00	1.0000E+00	0	0	0	0	6.435E+01	1.000E+00	2.452E-05
9	2	15	-4.9713E-02	0.0000E+00	-2.4530E+00	-3.8558E+04	0.0000E+00	1.0000E+00	0	0	0	0	6.435E+01	1.000E+00	8.757E-05
9	2	16	-3.2268E-02	0.0000E+00	-2.4658E+00	-3.6156E+04	0.0000E+00	1.0000E+00	0	0	0	0	6.435E+01	1.000E+00	-2.538E-05
9	2	17	-2.0299E-02	0.0000E+00	-2.4740E+00	-3.3741E+04	0.0000E+00	1.0000E+00	0	0	0	0	6.435E+01	1.000E+00	-4.559E-05
9	2	18	-1.1642E-02	0.0000E+00	-2.4788E+00	-3.1319E+04	0.0000E+00	1.0000E+00	0	0	0	0	6.435E+01	1.000E+00	-5.236E-05
9	2	19	-4.8375E-03	0.0000E+00	-2.4808E+00	-2.8891E+04	0.0000E+00	1.0000E+00	0	0	0	0	6.435E+01	1.000E+00	-7.092E-05
9	2	20	1.1892E-03	0.0000E+00	-2.4804E+00	-2.6462E+04	0.0000E+00	1.0000E+00	0	0	0	0	6.435E+01	1.000E+00	-6.150E-05
9	2	21	7.3832E-03	0.0000E+00	-2.4773E+00	-2.4033E+04	0.0000E+00	1.0000E+00	0	0	0	0	6.435E+01	1.000E+00	-4.558E-05
9	2	22	1.4683E-02	0.0000E+00	-2.4713E+00	-2.1607E+04	0.0000E+00	1.0000E+00	0	0	0	0	6.435E+01	1.000E+00	-6.298E-05
9	2	23	2.4220E-02	0.0000E+00	-2.4613E+00	-1.9188E+04	0.0000E+00	1.0000E+00	0	0	0	0	6.435E+01	1.000E+00	-1.283E-04
9	2	24	3.7544E-02	0.0000E+00	-2.4461E+00	-1.6777E+04	0.0000E+00	1.0000E+00	0	0	0	0	6.435E+01	1.000E+00	-1.270E-04
9	2	25	5.6832E-02	0.0000E+00	-2.4232E+00	-1.4382E+04	0.0000E+00	1.0000E+00	0	0	0	0	6.435E+01	1.000E+00	-1.183E-04
9	2	26	8.5442E-02	0.0000E+00	-2.3891E+00	-1.2010E+04	0.0000E+00	1.0000E+00	0	0	0	0	6.435E+01	1.000E+00	-1.152E-04
9	2	27	1.2854E-01	0.0000E+00	-2.3386E+00	-9.6702E+03	0.0000E+00	1.0000E+00	0	0	0	0	6.435E+01	1.000E+00	1.100E-04

TABLE V (Continued)

EXAMPLE PROBLEM DATA PRINT

/?

UOF3D TANK DRAIN J=2 PLANE ONLY 11/13/86 ?0?0?0?0?0?0?0?0 UOF3D7 03/12/87 11:06:53

ITER= 82 TIME= 2.04000E-04 DELT= 2.04000E-04 CYCLE= 1

I	J	K,	U	V	U	P	PS	F	NF	NFP	NFS	NFO	BETA	PETA	D
9	2	28	1.9399E-01	0.0000E+00	-2.2628E+00	-7.3805E+03	0.0000E+00	1.0000E+00	0	0	0	0	6.435E+01	1.000E+00	1.336E-04
9	2	29	2.9319E-01	0.0000E+00	-2.1498E+00	-5.1649E+03	0.0000E+00	1.0000E+00	0	0	0	0	6.435E+01	1.000E+00	-4.619E-05
9	2	30	4.4187E-01	0.0000E+00	-2.0134E+00	-3.0602E+03	0.0000E+00	1.0000E+00	0	0	0	0	6.435E+01	1.000E+00	-1.492E-05
9	2	31	6.9187E-01	0.0000E+00	-1.2598E+00	-1.0893E+03	0.0000E+00	9.9887E-01	0	0	0	0	6.435E+01	9.538E-01	-3.909E-05
9	2	32	7.3382E-01	0.0000E+00	-1.4724E+00	1.4290E+02	-1.2333E+01	3.7183E-01	5	5	5	5	6.435E+01	1.148E+00	0.000E+00
9	2	33	7.2950E-01	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	8	8	0	8	6.435E+01	1.000E+00	0.000E+00
9	2	34	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	8	8	0	8	6.435E+01	1.000E+00	0.000E+00
9	2	35	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	8	8	0	8	6.435E+01	1.000E+00	0.000E+00
9	2	36	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	8	8	0	8	6.435E+01	1.000E+00	0.000E+00
9	2	37	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	8	8	0	8	7.836E+01	1.000E+00	0.000E+00
9	2	38	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
10	2	1	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
10	2	2	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
10	2	3	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
10	2	4	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
10	2	5	0.0000E+00	0.0000E+00	0.0000E+00	-5.9518E+04	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
10	2	6	0.0000E+00	0.0000E+00	-2.0412E+00	-5.8016E+04	0.0000E+00	1.0000E+00	0	0	0	0	2.547E+01	1.000E+00	-1.021E-04
10	2	7	-9.2018E-01	0.0000E+00	-2.0541E+00	-5.6017E+04	0.0000E+00	1.0000E+00	0	0	0	0	6.248E+01	1.000E+00	-7.690E-05
10	2	8	-7.2712E-01	0.0000E+00	-2.1310E+00	-5.4006E+04	0.0000E+00	1.0000E+00	0	0	0	0	5.007E+01	1.000E+00	4.161E-05
10	2	9	-5.9655E-01	0.0000E+00	-2.1355E+00	-5.1920E+04	0.0000E+00	1.0000E+00	0	0	0	0	5.007E+01	1.000E+00	-2.578E-05
10	2	10	-3.9593E-01	0.0000E+00	-2.1797E+00	-4.9830E+04	0.0000E+00	1.0000E+00	0	0	0	0	5.007E+01	1.000E+00	-5.885E-05
10	2	11	-2.3434E-01	0.0000E+00	-2.2496E+00	-4.7696E+04	0.0000E+00	1.0000E+00	0	0	0	0	5.007E+01	1.000E+00	-8.816E-07
10	2	12	-1.2855E-01	0.0000E+00	-2.3232E+00	-4.5493E+04	0.0000E+00	1.0000E+00	0	0	0	0	5.007E+01	1.000E+00	7.616E-07
10	2	13	-7.6544E-02	0.0000E+00	-2.3782E+00	-4.3218E+04	0.0000E+00	1.0000E+00	0	0	0	0	5.007E+01	1.000E+00	-2.761E-05
10	2	14	-4.8165E-02	0.0000E+00	-2.4158E+00	-4.0890E+04	0.0000E+00	1.0000E+00	0	0	0	0	5.007E+01	1.000E+00	1.324E-05
10	2	15	-3.1063E-02	0.0000E+00	-2.4410E+00	-3.8524E+04	0.0000E+00	1.0000E+00	0	0	0	0	5.007E+01	1.000E+00	1.810E-05
10	2	16	-2.0045E-02	0.0000E+00	-2.4576E+00	-3.6134E+04	0.0000E+00	1.0000E+00	0	0	0	0	5.007E+01	1.000E+00	-1.042E-04
10	2	17	-1.2568E-02	0.0000E+00	-2.4681E+00	-3.3727E+04	0.0000E+00	1.0000E+00	0	0	0	0	5.007E+01	1.000E+00	-1.308E-04
10	2	18	-7.1914E-03	0.0000E+00	-2.4741E+00	-3.1311E+04	0.0000E+00	1.0000E+00	0	0	0	0	5.007E+01	1.000E+00	-5.750E-05
10	2	19	-2.9877E-03	0.0000E+00	-2.4767E+00	-2.8888E+04	0.0000E+00	1.0000E+00	0	0	0	0	5.007E+01	1.000E+00	-1.723E-04
10	2	20	7.2644E-04	0.0000E+00	-2.4761E+00	-2.6463E+04	0.0000E+00	1.0000E+00	0	0	0	0	5.007E+01	1.000E+00	-1.694E-04
10	2	21	4.5598E-03	0.0000E+00	-2.4723E+00	-2.4038E+04	0.0000E+00	1.0000E+00	0	0	0	0	5.007E+01	1.000E+00	-1.563E-04
10	2	22	9.0627E-03	0.0000E+00	-2.4647E+00	-2.1617E+04	0.0000E+00	1.0000E+00	0	0	0	0	5.007E+01	1.000E+00	-2.344E-04
10	2	23	1.4961E-02	0.0000E+00	-2.4522E+00	-1.9204E+04	0.0000E+00	1.0000E+00	0	0	0	0	5.007E+01	1.000E+00	-1.631E-04
10	2	24	2.3214E-02	0.0000E+00	-2.4328E+00	-1.6803E+04	0.0000E+00	1.0000E+00	0	0	0	0	5.007E+01	1.000E+00	-2.540E-04
10	2	25	3.5192E-02	0.0000E+00	-2.4035E+00	-1.4421E+04	0.0000E+00	1.0000E+00	0	0	0	0	5.007E+01	1.000E+00	-2.240E-05
10	2	26	5.2996E-02	0.0000E+00	-2.3595E+00	-1.2067E+04	0.0000E+00	1.0000E+00	0	0	0	0	5.007E+01	1.000E+00	1.382E-04
10	2	27	7.9860E-02	0.0000E+00	-2.2936E+00	-9.7569E+03	0.0000E+00	1.0000E+00	0	0	0	0	5.007E+01	1.000E+00	8.391E-05
10	2	28	1.2081E-01	0.0000E+00	-2.1946E+00	-7.5112E+03	0.0000E+00	1.0000E+00	0	0	0	0	5.007E+01	1.000E+00	6.428E-05
10	2	29	1.8412E-01	0.0000E+00	-2.0476E+00	-5.3626E+03	0.0000E+00	1.0000E+00	0	0	0	0	5.007E+01	1.000E+00	-1.073E-05
10	2	30	2.8685E-01	0.0000E+00	-1.8415E+00	-3.3581E+03	0.0000E+00	1.0000E+00	0	0	0	0	5.007E+01	1.000E+00	-9.113E-05
10	2	31	4.8835E-01	0.0000E+00	-1.2629E+00	-1.5556E+03	0.0000E+00	9.9967E-01	0	0	0	0	5.007E+01	1.000E+00	3.541E-05
10	2	32	6.8286E-01	0.0000E+00	-1.2629E+00	-2.1310E+02	-5.8452E+00	9.9412E-01	2	5	2	2	5.007E+01	6.456E-01	0.000E+00
10	2	33	4.9345E-01	0.0000E+00	-9.4443E-02	0.0000E+00	0.0000E+00	0.0000E+00	8	8	0	8	5.007E+01	1.000E+00	0.000E+00
10	2	34	3.4592E-01	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	8	8	0	8	5.007E+01	1.000E+00	0.000E+00
10	2	35	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	8	8	0	8	5.007E+01	1.000E+00	0.000E+00
10	2	36	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	8	8	0	8	5.007E+01	1.000E+00	0.000E+00
10	2	37	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	8	8	0	8	5.816E+01	1.000E+00	0.000E+00
10	2	38	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
11	2	1	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
11	2	2	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
11	2	3	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
11	2	4	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
11	2	5	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
11	2	6	0.0000E+00	0.0000E+00	0.0000E+00	-5.6763E+04	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00

TABLE V (Continued)

EXAMPLE PROBLEM DATA PRINT

/?

VOF3D TANK DRAIN J=2 PLANE ONLY 11/13/86 ?0?0?0?0?0?0?0?0 VOF3D7 03/12/87 11:06:53

ITER= 82 TIME= 2.04000E-04 DELT= 2.04000E-04 CYCLE= 1

I	J	K,	U	V	W	P	PS	F	NF	NFP	NFS	NFO	BETA	PETA	D
11	2	7	0.0000E+00	0.0000E+00	-1.9455E+00	-5.5510E+04	0.0000E+00	1.0000E+00	0	0	0	0	1.357E+01	1.000E+00	-2.823E-04
11	2	8	0.0000E+00	0.0000E+00	-2.0575E+00	-5.3605E+04	0.0000E+00	1.0000E+00	0	0	0	0	5.308E+01	1.000E+00	-4.170E-05
11	2	9	-5.8218E-01	0.0000E+00	-2.0226E+00	-5.1591E+04	0.0000E+00	1.0000E+00	0	0	0	0	3.763E+01	1.000E+00	-1.993E-04
11	2	10	-3.4291E-01	0.0000E+00	-2.0888E+00	-4.9611E+04	0.0000E+00	1.0000E+00	0	0	0	0	3.763E+01	1.000E+00	-1.509E-04
11	2	11	-1.7270E-01	0.0000E+00	-2.1901E+00	-4.7566E+04	0.0000E+00	1.0000E+00	0	0	0	0	3.763E+01	1.000E+00	-7.484E-05
11	2	12	-7.1192E-02	0.0000E+00	-2.2940E+00	-4.5422E+04	0.0000E+00	1.0000E+00	0	0	0	0	3.763E+01	1.000E+00	-6.617E-05
11	2	13	-3.9312E-02	0.0000E+00	-2.3622E+00	-4.3176E+04	0.0000E+00	1.0000E+00	0	0	0	0	3.763E+01	1.000E+00	-9.988E-05
11	2	14	-2.4210E-02	0.0000E+00	-2.4062E+00	-4.0863E+04	0.0000E+00	1.0000E+00	0	0	0	0	3.763E+01	1.000E+00	-3.034E-05
11	2	15	-1.5501E-02	0.0000E+00	-2.4348E+00	-3.8507E+04	0.0000E+00	1.0000E+00	0	0	0	0	3.763E+01	1.000E+00	-2.778E-05
11	2	16	-9.9707E-03	0.0000E+00	-2.4534E+00	-3.6123E+04	0.0000E+00	1.0000E+00	0	0	0	0	3.763E+01	1.000E+00	-1.559E-04
11	2	17	-6.2496E-03	0.0000E+00	-2.4651E+00	-3.3721E+04	0.0000E+00	1.0000E+00	0	0	0	0	3.763E+01	1.000E+00	-3.239E-04
11	2	18	-3.5742E-03	0.0000E+00	-2.4718E+00	-3.1307E+04	0.0000E+00	1.0000E+00	0	0	0	0	3.763E+01	1.000E+00	-2.836E-04
11	2	19	-1.4910E-03	0.0000E+00	-2.4746E+00	-2.8886E+04	0.0000E+00	1.0000E+00	0	0	0	0	3.763E+01	1.000E+00	-3.382E-04
11	2	20	3.5947E-04	0.0000E+00	-2.4739E+00	-2.6463E+04	0.0000E+00	1.0000E+00	0	0	0	0	3.763E+01	1.000E+00	-5.947E-05
11	2	21	2.2474E-03	0.0000E+00	-2.4698E+00	-2.4041E+04	0.0000E+00	1.0000E+00	0	0	0	0	3.763E+01	1.000E+00	-4.061E-04
11	2	22	4.4924E-03	0.0000E+00	-2.4614E+00	-2.1622E+04	0.0000E+00	1.0000E+00	0	0	0	0	3.763E+01	1.000E+00	-2.205E-04
11	2	23	7.4184E-03	0.0000E+00	-2.4475E+00	-1.9212E+04	0.0000E+00	1.0000E+00	0	0	0	0	3.763E+01	1.000E+00	-1.082E-04
11	2	24	1.1512E-02	0.0000E+00	-2.4261E+00	-1.6816E+04	0.0000E+00	1.0000E+00	0	0	0	0	3.763E+01	1.000E+00	-2.522E-04
11	2	25	1.7475E-02	0.0000E+00	-2.3935E+00	-1.4440E+04	0.0000E+00	1.0000E+00	0	0	0	0	3.763E+01	1.000E+00	6.241E-05
11	2	26	2.6325E-02	0.0000E+00	-2.3444E+00	-1.2096E+04	0.0000E+00	1.0000E+00	0	0	0	0	3.763E+01	1.000E+00	7.014E-05
11	2	27	3.9707E-02	0.0000E+00	-2.2706E+00	-9.8009E+03	0.0000E+00	1.0000E+00	0	0	0	0	3.763E+01	1.000E+00	-2.327E-05
11	2	28	6.0165E-02	0.0000E+00	-2.1590E+00	-7.5778E+03	0.0000E+00	1.0000E+00	0	0	0	0	3.763E+01	1.000E+00	1.808E-04
11	2	29	9.2066E-02	0.0000E+00	-1.9898E+00	-5.4641E+03	0.0000E+00	1.0000E+00	0	0	0	0	3.763E+01	1.000E+00	1.110E-04
11	2	30	1.4438E-01	0.0000E+00	-1.7282E+00	-3.5163E+03	0.0000E+00	1.0000E+00	0	0	0	0	3.763E+01	1.000E+00	-8.944E-05
11	2	31	2.3617E-01	0.0000E+00	-1.2629E+00	-1.8249E+03	0.0000E+00	1.0000E+00	0	0	0	0	3.763E+01	1.000E+00	2.026E-04
11	2	32	3.1804E-01	0.0000E+00	-6.0220E-01	-5.8968E+02	0.0000E+00	1.0000E+00	0	0	0	0	3.763E+01	1.100E+00	2.085E-04
11	2	33	2.3989E-01	0.0000E+00	-1.3472E-01	-4.0535E+01	-1.6753E+01	7.0102E-01	2	5	2	2	3.763E+01	8.326E-01	0.000E+00
11	2	34	5.2418E-02	0.0000E+00	-2.1400E-02	0.0000E+00	0.0000E+00	0.0000E+00	8	8	0	8	3.763E+01	1.000E+00	0.000E+00
11	2	35	3.4436E-02	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	8	8	0	8	3.763E+01	1.000E+00	0.000E+00
11	2	36	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	8	8	0	8	3.763E+01	1.000E+00	0.000E+00
11	2	37	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	8	8	0	8	4.202E+01	1.000E+00	0.000E+00
11	2	38	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
12	2	1	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
12	2	2	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
12	2	3	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
12	2	4	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
12	2	5	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
12	2	6	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
12	2	7	0.0000E+00	0.0000E+00	0.0000E+00	-5.5510E+04	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
12	2	8	0.0000E+00	0.0000E+00	0.0000E+00	-5.2456E+04	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
12	2	9	0.0000E+00	0.0000E+00	-1.9030E+00	-5.1306E+04	0.0000E+00	1.0000E+00	0	0	0	0	2.704E+01	1.000E+00	-9.059E-05
12	2	10	0.0000E+00	0.0000E+00	-2.0037E+00	-4.9443E+04	0.0000E+00	1.0000E+00	0	0	0	0	4.993E+01	1.000E+00	-1.551E-04
12	2	11	0.0000E+00	0.0000E+00	-2.1393E+00	-4.7482E+04	0.0000E+00	1.0000E+00	0	0	0	0	5.918E+01	1.000E+00	-7.228E-05
12	2	12	0.0000E+00	0.0000E+00	-2.2781E+00	-4.5387E+04	0.0000E+00	1.0000E+00	0	0	0	0	6.104E+01	1.000E+00	-3.254E-05
12	2	13	0.0000E+00	0.0000E+00	-2.3547E+00	-4.3157E+04	0.0000E+00	1.0000E+00	0	0	0	0	6.104E+01	1.000E+00	-1.017E-05
12	2	14	0.0000E+00	0.0000E+00	-2.4019E+00	-4.0851E+04	0.0000E+00	1.0000E+00	0	0	0	0	6.104E+01	1.000E+00	-1.126E-04
12	2	15	0.0000E+00	0.0000E+00	-2.4321E+00	-3.8499E+04	0.0000E+00	1.0000E+00	0	0	0	0	6.104E+01	1.000E+00	2.900E-05
12	2	16	0.0000E+00	0.0000E+00	-2.4515E+00	-3.6118E+04	0.0000E+00	1.0000E+00	0	0	0	0	6.104E+01	1.000E+00	-1.618E-04
12	2	17	0.0000E+00	0.0000E+00	-2.4637E+00	-3.3717E+04	0.0000E+00	1.0000E+00	0	0	0	0	6.104E+01	1.000E+00	-1.478E-04
12	2	18	0.0000E+00	0.0000E+00	-2.4707E+00	-3.1305E+04	0.0000E+00	1.0000E+00	0	0	0	0	6.104E+01	1.000E+00	-1.999E-04
12	2	19	0.0000E+00	0.0000E+00	-2.4737E+00	-2.8886E+04	0.0000E+00	1.0000E+00	0	0	0	0	6.104E+01	1.000E+00	-1.499E-04
12	2	20	0.0000E+00	0.0000E+00	-2.4730E+00	-2.6463E+04	0.0000E+00	1.0000E+00	0	0	0	0	6.104E+01	1.000E+00	-9.214E-05
12	2	21	0.0000E+00	0.0000E+00	-2.4686E+00	-2.4042E+04	0.0000E+00	1.0000E+00	0	0	0	0	6.104E+01	1.000E+00	-1.262E-04
12	2	22	0.0000E+00	0.0000E+00	-2.4599E+00	-2.1625E+04	0.0000E+00	1.0000E+00	0	0	0	0	6.104E+01	1.000E+00	-2.502E-04
12	2	23	0.0000E+00	0.0000E+00	-2.4455E+00	-1.9216E+04	0.0000E+00	1.0000E+00	0	0	0	0	6.104E+01	1.000E+00	-9.694E-05

TABLE V (Continued)

EXAMPLE PROBLEM DATA PRINT

?

VOF3D TANK DRAIN J=2 PLANE ONLY   11/13/86   ?0?0?0?0?0?0?0?0   VOF3D7   03/12/87 11:06:53

ITER= 82   TIME= 2.04000E-04   DELT= 2.04000E-04   CYCLE= 1

I	J	K.	U	V	W	P	PS	F	NF	NFP	NFS	NFO	BETA	PETA	D
12	2	24	0.0000E+00	0.0000E+00	-2.4231E+00	-1.6821E+04	0.0000E+00	1.0000E+00	0	0	0	0	6.104E+01	1.000E+00	-1.238E-04
12	2	25	0.0000E+00	0.0000E+00	-2.3890E+00	-1.4449E+04	0.0000E+00	1.0000E+00	0	0	0	0	6.104E+01	1.000E+00	-5.823E-05
12	2	26	0.0000E+00	0.0000E+00	-2.3377E+00	-1.2109E+04	0.0000E+00	1.0000E+00	0	0	0	0	6.104E+01	1.000E+00	1.753E-04
12	2	27	0.0000E+00	0.0000E+00	-2.2603E+00	-9.8204E+03	0.0000E+00	1.0000E+00	0	0	0	0	6.104E+01	1.000E+00	-1.444E-06
12	2	28	0.0000E+00	0.0000E+00	-2.1431E+00	-7.6073E+03	0.0000E+00	1.0000E+00	0	0	0	0	6.104E+01	1.000E+00	1.388E-04
12	2	29	0.0000E+00	0.0000E+00	-1.9636E+00	-5.5092E+03	0.0000E+00	1.0000E+00	0	0	0	0	6.104E+01	1.000E+00	3.909E-05
12	2	30	0.0000E+00	0.0000E+00	-1.6823E+00	-3.5870E+03	0.0000E+00	1.0000E+00	0	0	0	0	6.104E+01	1.000E+00	1.943E-05
12	2	31	0.0000E+00	0.0000E+00	-1.2220E+00	-1.9407E+03	0.0000E+00	1.0000E+00	0	0	0	0	6.104E+01	1.000E+00	1.831E-04
12	2	32	0.0000E+00	0.0000E+00	-6.0220E-01	-7.4558E+02	0.0000E+00	1.0000E+00	0	0	0	0	6.104E+01	1.000E+00	2.514E-04
12	2	33	0.0000E+00	0.0000E+00	-1.3472E-01	-1.5813E+02	0.0000E+00	1.0000E+00	0	0	0	0	6.104E+01	1.100E+00	1.985E-05
12	2	34	0.0000E+00	0.0000E+00	-3.2575E-02	-2.8992E+01	-2.8973E+01	6.5696E-01	2	5	2	2	6.104E+01	1.000E+00	0.000E+00
12	2	35	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	8	8	0	8	6.104E+01	1.000E+00	0.000E+00
12	2	36	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	8	8	0	8	6.104E+01	1.000E+00	0.000E+00
12	2	37	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	8	8	0	8	7.351E+01	1.000E+00	0.000E+00
12	2	38	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
13	2	1	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
13	2	2	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
13	2	3	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
13	2	4	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
13	2	5	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
13	2	6	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
13	2	7	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
13	2	8	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
13	2	9	0.0000E+00	0.0000E+00	-1.9030E+00	-5.1306E+04	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
13	2	10	0.0000E+00	0.0000E+00	-2.0037E+00	-4.9443E+04	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
13	2	11	0.0000E+00	0.0000E+00	-2.1393E+00	-4.7482E+04	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
13	2	12	0.0000E+00	0.0000E+00	-2.2781E+00	-4.5387E+04	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
13	2	13	0.0000E+00	0.0000E+00	-2.3547E+00	-4.3157E+04	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
13	2	14	0.0000E+00	0.0000E+00	-2.4019E+00	-4.0851E+04	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
13	2	15	0.0000E+00	0.0000E+00	-2.4321E+00	-3.8499E+04	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
13	2	16	0.0000E+00	0.0000E+00	-2.4515E+00	-3.6118E+04	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
13	2	17	0.0000E+00	0.0000E+00	-2.4637E+00	-3.3717E+04	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
13	2	18	0.0000E+00	0.0000E+00	-2.4707E+00	-3.1305E+04	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
13	2	19	0.0000E+00	0.0000E+00	-2.4737E+00	-2.8886E+04	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
13	2	20	0.0000E+00	0.0000E+00	-2.4730E+00	-2.6463E+04	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
13	2	21	0.0000E+00	0.0000E+00	-2.4686E+00	-2.4042E+04	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
13	2	22	0.0000E+00	0.0000E+00	-2.4599E+00	-2.1625E+04	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
13	2	23	0.0000E+00	0.0000E+00	-2.4455E+00	-1.9216E+04	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
13	2	24	0.0000E+00	0.0000E+00	-2.4231E+00	-1.6821E+04	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
13	2	25	0.0000E+00	0.0000E+00	-2.3890E+00	-1.4449E+04	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
13	2	26	0.0000E+00	0.0000E+00	-2.3377E+00	-1.2109E+04	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
13	2	27	0.0000E+00	0.0000E+00	-2.2603E+00	-9.8204E+03	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
13	2	28	0.0000E+00	0.0000E+00	-2.1431E+00	-7.6073E+03	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
13	2	29	0.0000E+00	0.0000E+00	-1.9636E+00	-5.5092E+03	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
13	2	30	0.0000E+00	0.0000E+00	-1.6823E+00	-3.5870E+03	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
13	2	31	0.0000E+00	0.0000E+00	-1.2220E+00	-1.9407E+03	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
13	2	32	0.0000E+00	0.0000E+00	-6.0220E-01	-7.4558E+02	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
13	2	33	0.0000E+00	0.0000E+00	-1.3472E-01	-1.5813E+02	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
13	2	34	0.0000E+00	0.0000E+00	-3.2575E-02	-2.8992E+01	0.0000E+00	6.5696E-01	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
13	2	35	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
13	2	36	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
13	2	37	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
13	2	38	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
1	3	1	4.3354E+01	-2.9552E+01	-2.4979E+02	-2.1991E+05	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00
1	3	2	4.3354E+01	-2.9552E+01	-8.5091E+01	-2.1991E+05	0.0000E+00	1.0000E+00	0	0	0	0	-1.000E+00	1.000E+00	0.000E+00

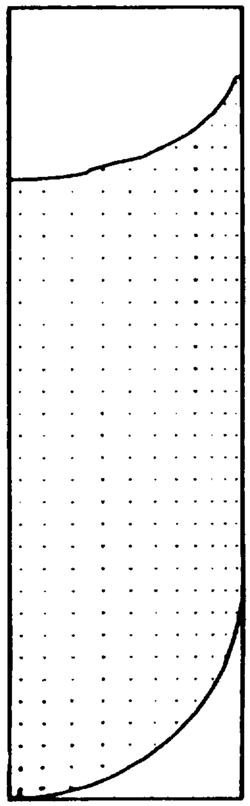

Fig. 2. Example problem: velocity vectors and free-surface contour at t=0.0002 s.

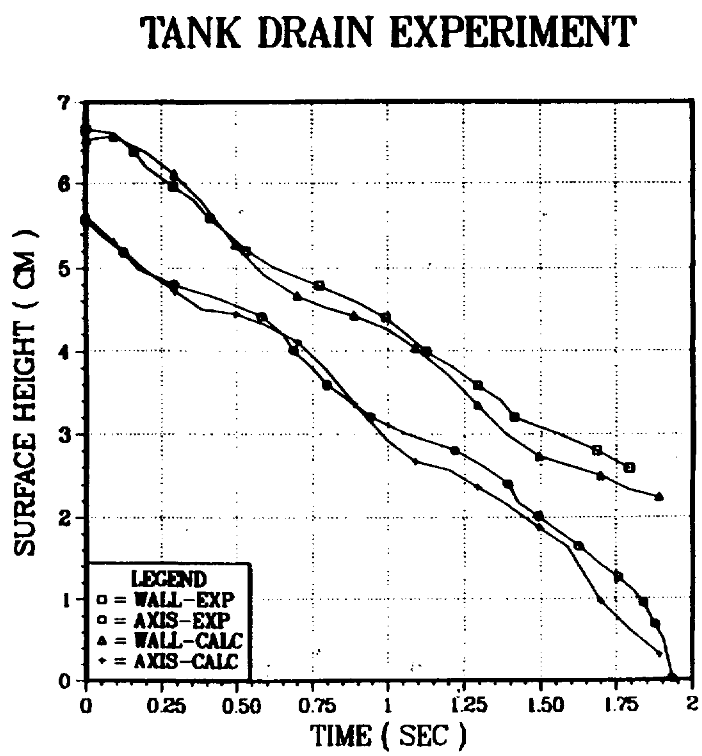

Fig. 3. Free surface height history.

reflect somewhat arbitrary definition of free surface location at the container wall. The maximal error resulting from this should be less than one cell height (0.2 cm).

IV. EXAMPLES OF SURFACE TENSION CALCULATIONS

We display the comparatively high accuracy of the present surface tension algorithms by presenting selected results for three geometrically simple examples. We also discuss the errors that can arise from the present algorithms and the (larger) errors that will arise from the more standard algorithms.

The class of example problem most difficult for our algorithms is that of the plane interface. Here the surface tension forces on opposite or adjacent faces of a computational cell should cancel exactly, and this is most difficult to arrange numerically using necessarily inexact data. The example we discuss is that of a vertical plane surface which is 0.55 cm from the axis of the ellipsoidal tank used for the experiments and simulations discussed in Sec. V. Fluid lies to the left of the interface, which, in turn, lies to the right of the axis. The fluid-vapor interface has surface-tension $\sigma = 28.2$ dyne cm^{-1}, thus a surface pressure $|p_S| \sim 40$ dyne cm^{-2} corresponds to a strongly curved interface. In **Table VI** we display the surface pressures resulting from our present surface-tension algorithms as a function of the azimuthal zone number J. The first physical cell is $j = 2$, while $j = 15$ constitutes the last azimuthal zone inter-sected by the interface. It may be seen that the absolute surface pressures are mostly two to three orders of magnitude smaller than a strongly curved surface would possess, with the smallest value being $p_S = 0.015$. The multiple entries for $j = 14$ and $j = 15$ denote adjacent radial zones (values of i) that contain part of the surface.

The new surface tension algorithms generally work quite well. We have tested several hundred values of p_S, obtained from two off-axis spherical bubbles of varying radius and surface tension in different geometrical locations in the same ellipsoidal tank referred to above and in Sec. V. The tank is otherwise filled with a liquid. We obtain many values of the surface pressure, p_S, that are within 1% of the correct (analytic) values and almost all are within several percent of correct values. However, all 3D surface-tension calculations, including ours, are highly amplifying of certain types of numerical errors. As an example of the sorts of difficulty that must be handled, we note that we have seen an example in a spherical bubble where 1-2% errors on the height functions defining the surface result in 20-

TABLE VI

SURFACE PRESSURES FOR PLANE INTERFACE IN ELLIPSOIDAL TANK

Azimuthal Zone Number J	Surface Pressure P_S (dynes · cm^{-2})
2	−0.808
3	−0.228
4	−0.905
5	−0.406
6	−0.649
7	0.055
8	−0.205
9	0.248
10	−0.120
11	−0.339
12	−0.266
13	0.055
14	0.399
	−0.161
15	0.015
	−0.327
	0.068
	0.219

30% errors in some values for components of surface gradient vectors and as much as 50-60% error in a local value for p_S. By contrast, in the same example using standard older formulations for evaluating p_S (e.g., those used in the early phases of our work) it would be common to have 100-200% error in components of surface gradient vectors. So our algorithms represent an improvement in the state of the art, but they can still encounter problems. In general, it is more difficult to calculate p_S in 3D than in 2D. In 3D both surface curvatures must be evaluated by finite-difference approximations to second derivatives, whereas in 2D one of the curvatures may be obtained from a more accurate algebraic formula. Moreover, in cylindrical geometry, the algebraic curvature is typically the largest one in the crucial region near the coordinate axis ($r = 0.0$). But in 3D the full numerical complexities of evaluating p_s must be faced routinely.

V. SAMPLE APPLICATIONS

We present here numerical simulations of three experiments[13] done at the NASA LeRC Zero Gravity Facility to model the liquid oxygen dynamics in a Centaur G-Prime Tank during separation from the shuttle. Graphs of the data have been presented, along with numerical simulations using HYDR3-D,[14] as Figs. 1, 2, and 3 of Reference 15. We will be referring to these figures as we discuss our simulations.

The experiments are done in an ellipsoidal tank which is partially filled with either Ethanol or FC-43 and then dropped from a drop tower. After two seconds of free fall the tank is accelerated along a principal axis for roughly 0.1 sec and the position of the vapor bubble is recorded on film for roughly 0.6 to 0.8 sec. The two-dimensional bubble profiles recorded on the film and displayed in Figs. 1, 2, and 3 of Reference 15 can give only a very approximate impression of the actual three-dimensional bubble contours.

We present data for the numerical simulations in several computational planes. Our basic plot is a composite plot of the first ($j = 2$) and the last ($j = JM1$) azimuthal planes. It is very nearly the result that would be seen at the midplane of the ellipsoid. Other plots are given in azimuthal planes (j = constant) and horizontal planes (k = constant). The calculational mesh of physical cells contains 13 radial cells, 18 azimuthal cells, and 20 vertical cells. To simplify the running of the problems, the applied acceleration is ramped linearly from zero to its experimental value in a time interval Δt_r which is typically 1% of the time interval Δt_a over which the acceleration is applied.

Experiment R-12-7 was done with Ethanol, and had an applied acceleration of 0.22 g lasting $\Delta t_a = 0.109$ sec. Numerical results of our simulation are displayed in Figs. 4-17, which give velocity vector and surface contour plots in our basic composite plot at various times during the run. A fairly rich display of computational planes (j = constant, and k = constant) is also given at $t = 0.10$ sec. It is seen from Figs. 6-12 and from Fig. 3 of Reference 15 that our simulation correctly reproduces the location of the bubble at $t = 0.10$ sec, including the jetting of the fluid into the central bubble region. The fluid jet is somewhat wider than indicated by experiment.

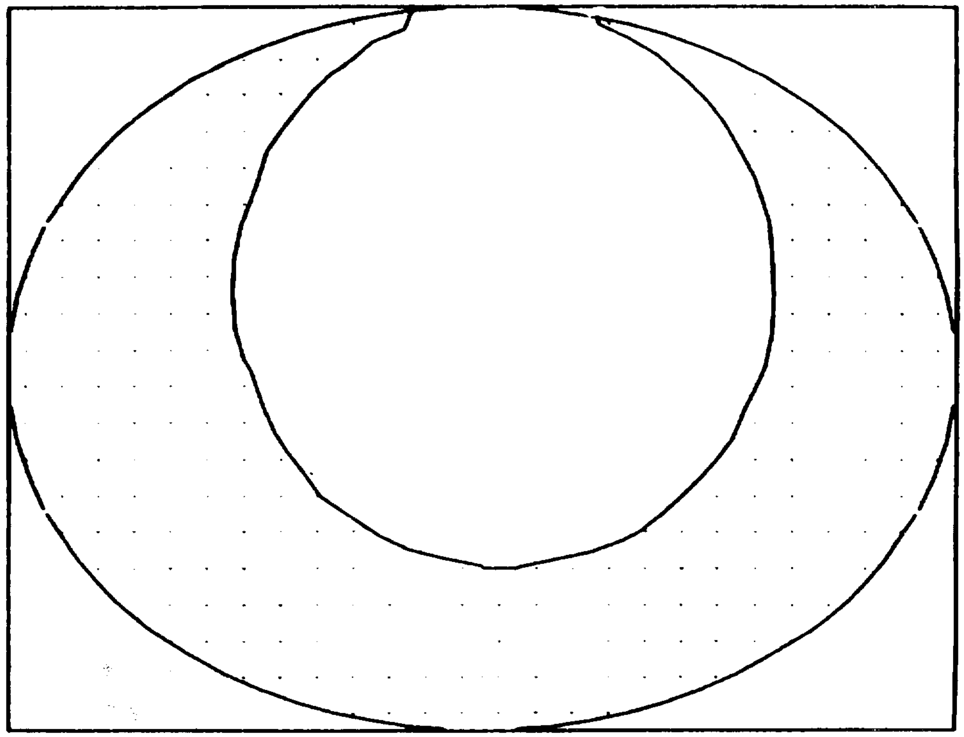

Fig. 4. Velocity vector plot for problem R-12-7: t=0.0 s.

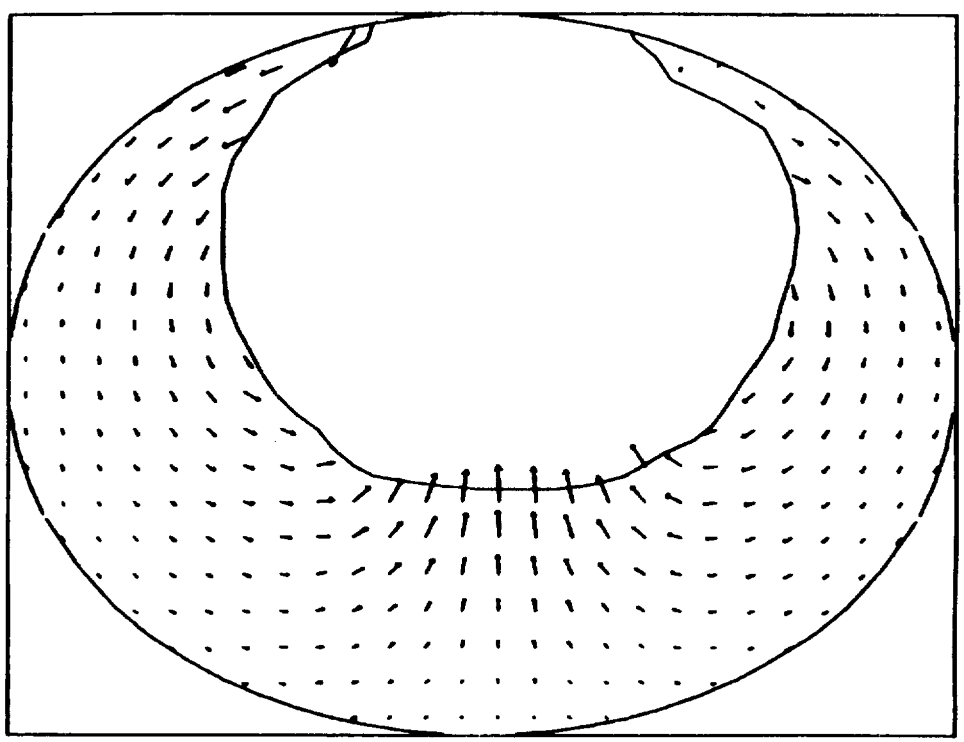

Fig. 5. Velocity vector plot for problem R-12-7: t=0.05 s.

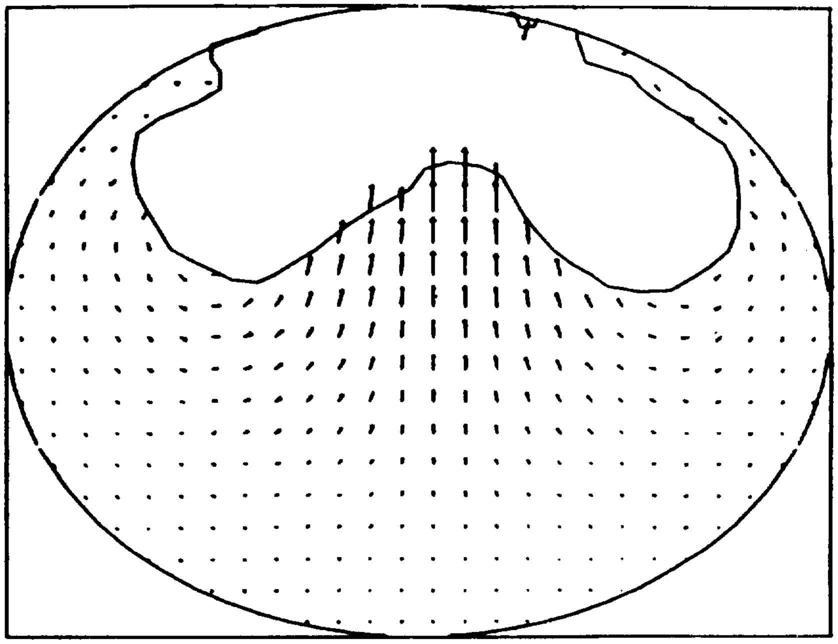

Fig. 6. Velocity vector plot for problem R-12-7: t=0.10 s.

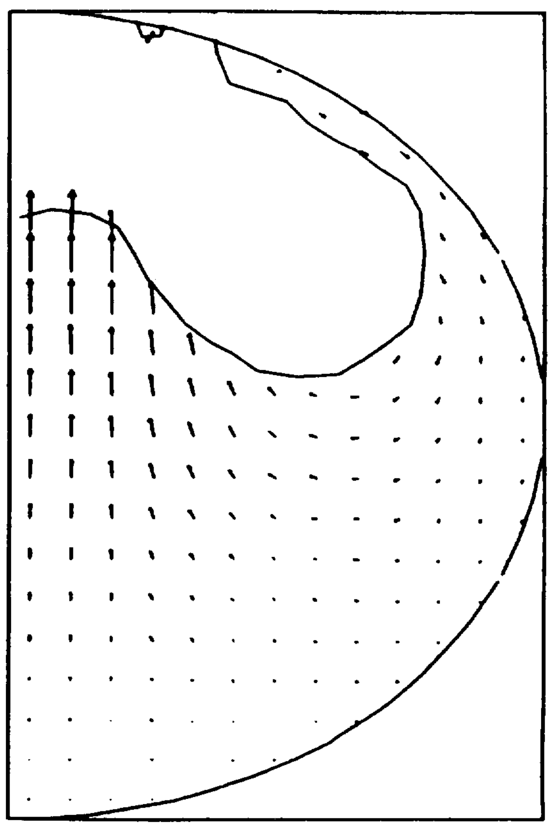

Fig. 7. Velocity vector plot for problem R-12-7: t= 0.10 s. J=2 surface.

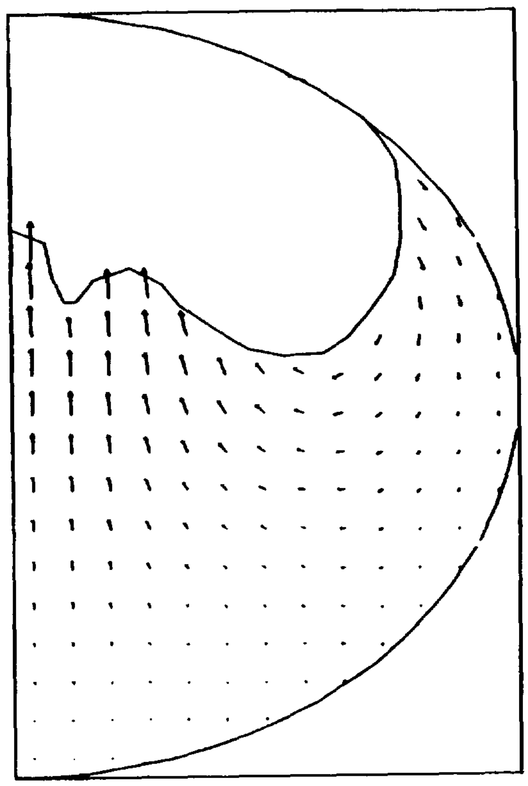

Fig. 8. Velocity vector plot for problem R-12-7: t=0.10 s. J=5 surface.

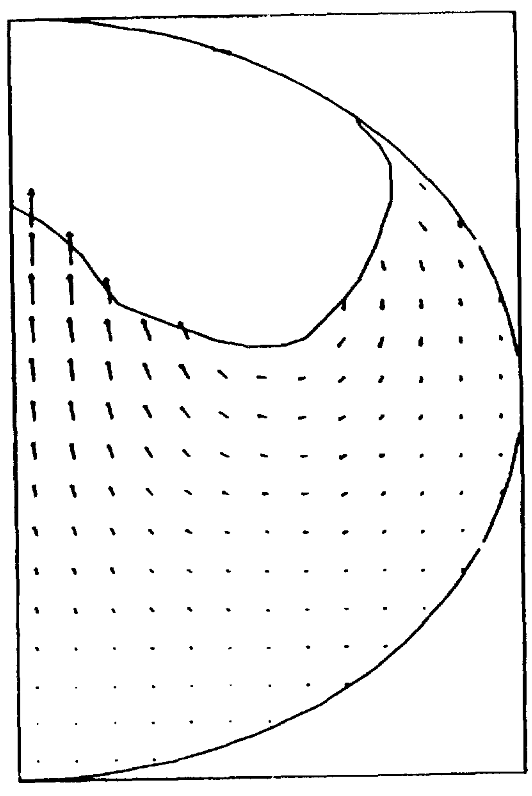

Fig. 9. Velocity vector plot for problem R-12-7: t=0.10 s. J=12 surface.

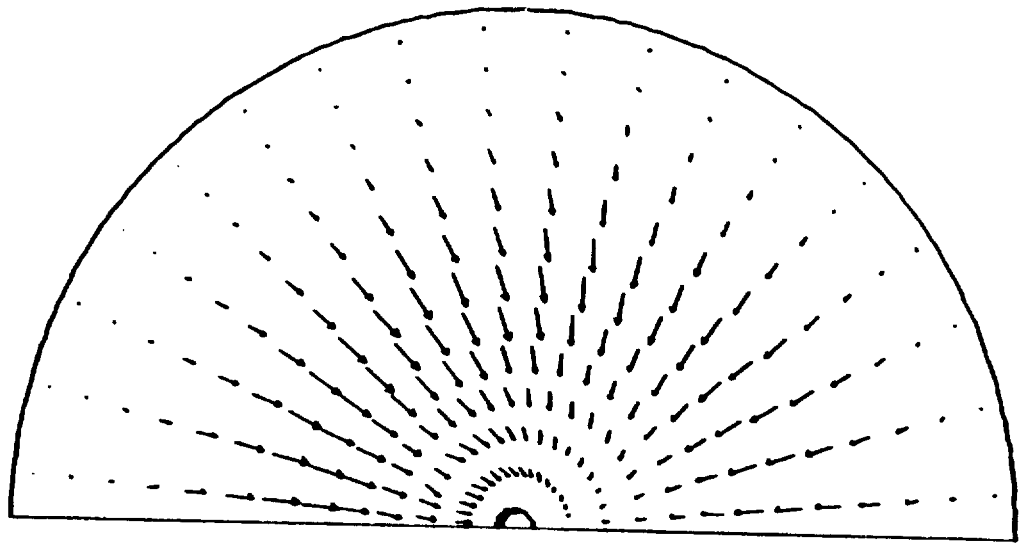

Fig. 10. Velocity vector plot for problem R-12-7: t=0.10 s. K=12 surface.

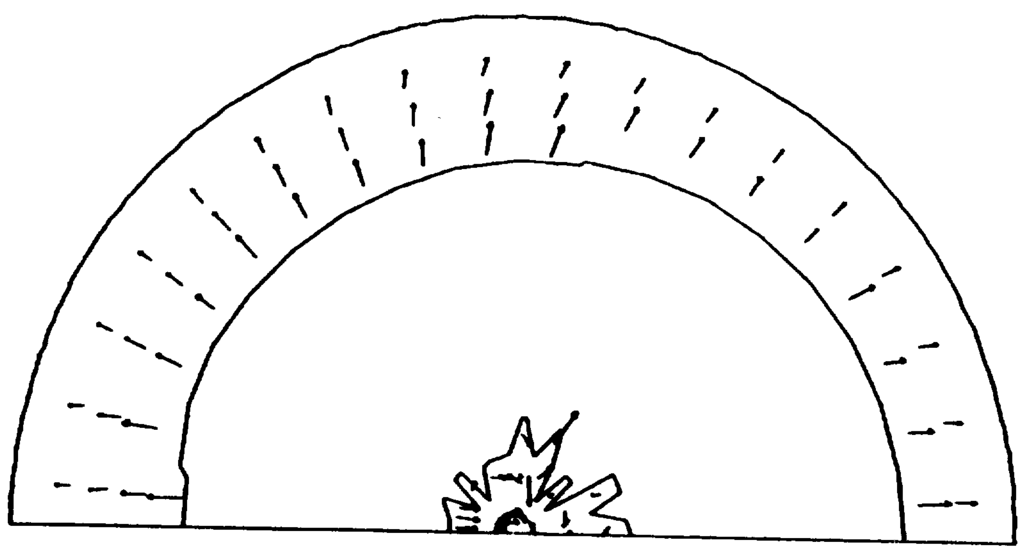

Fig. 11. Velocity vector plot for problem R-12-7: t=0.10 s. K=15 surface.

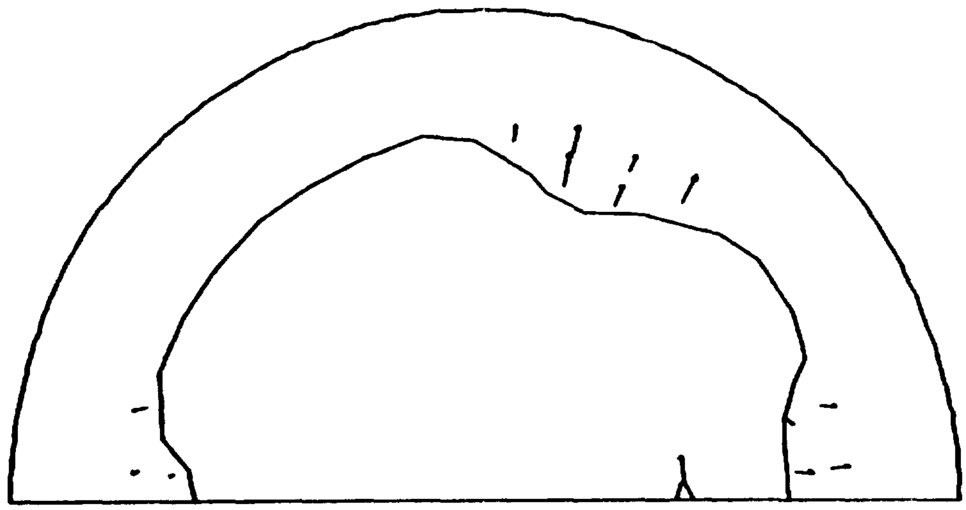

Fig. 12. Velocity vector plot for problem R-12-7: t=0.10 s. K=18 surface.

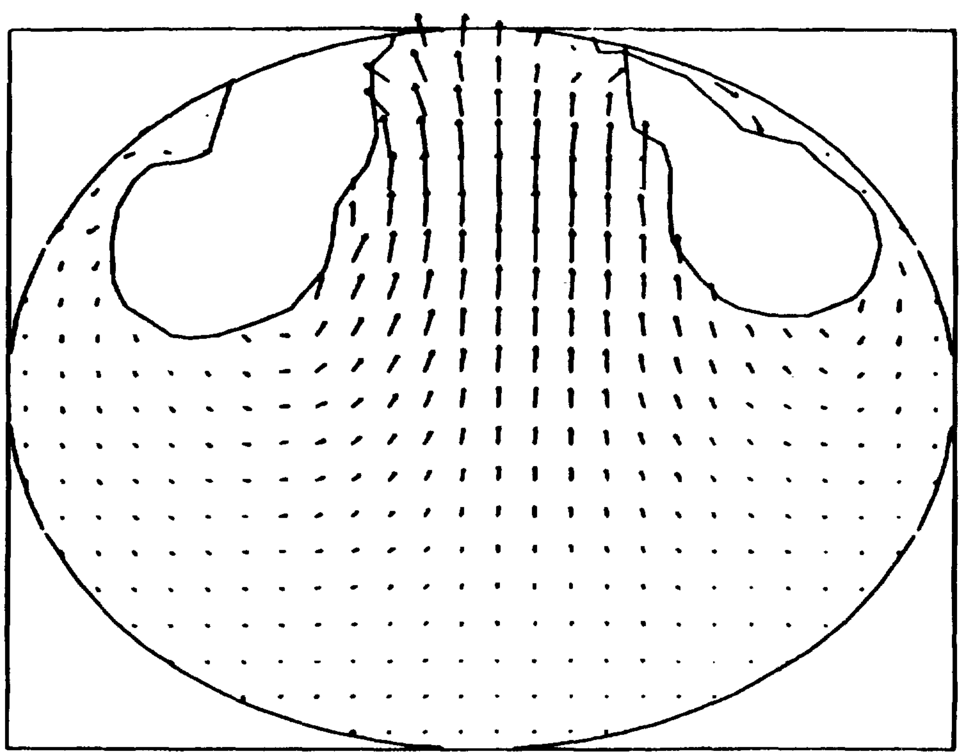

Fig. 13. Velocity vector plot for problem R-12-7: t=0.15 s.

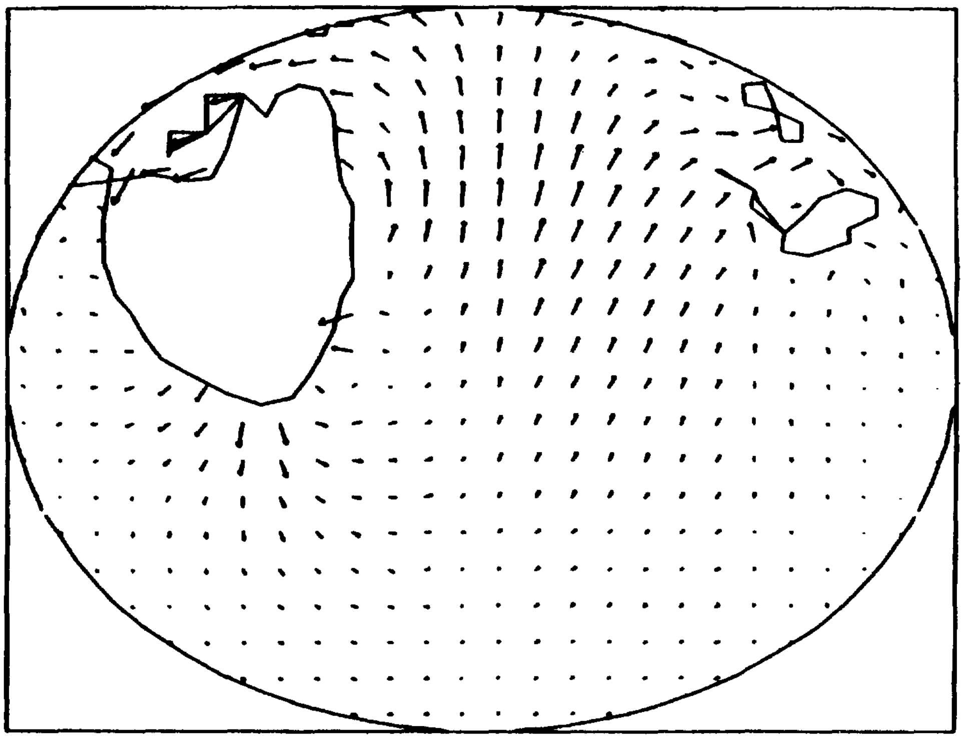

Fig. 14. Velocity vector plot for problem R-12-7: t=0.25 s.

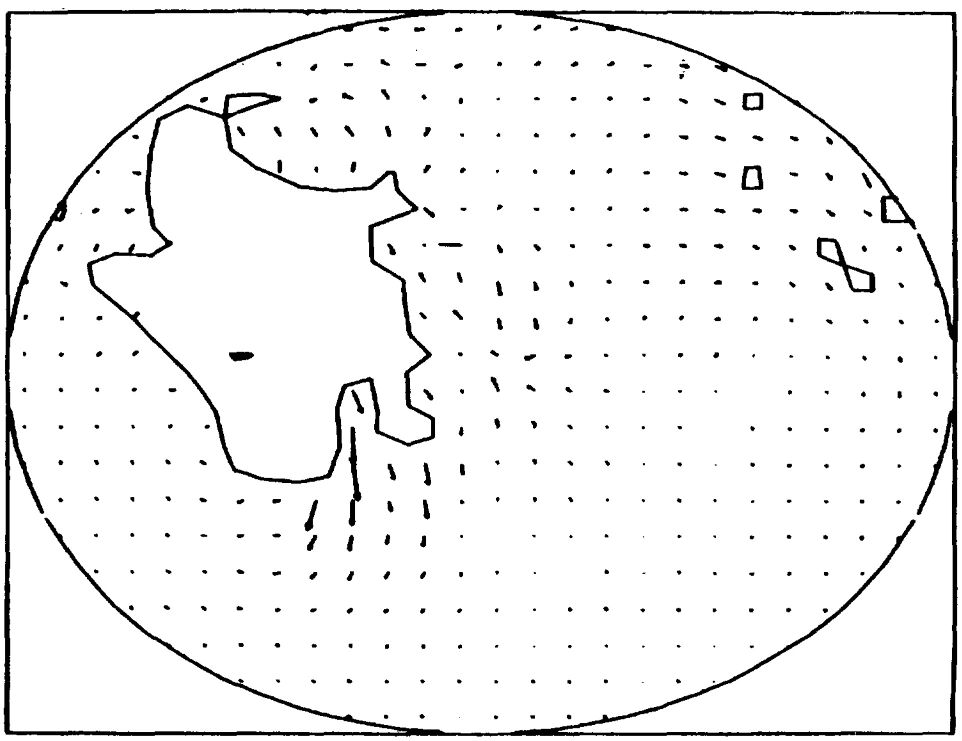

Fig. 15. Velocity vector plot for problem R-12-7: t=0.45 s.

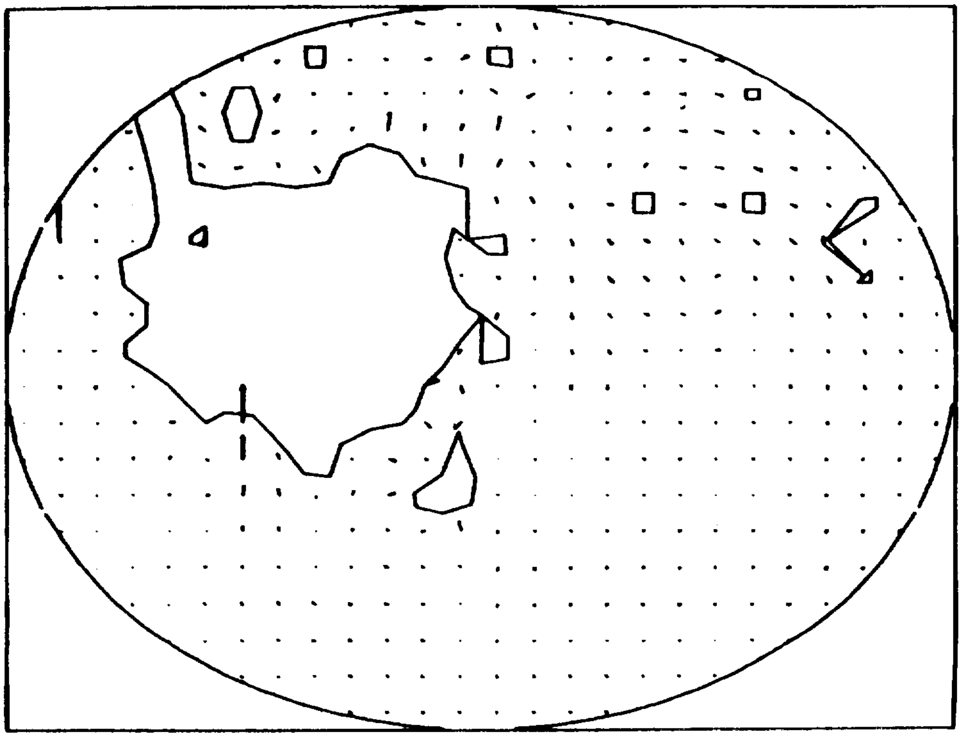

Fig. 16. Velocity vector plot for problem R-12-7: t=0.65 s.

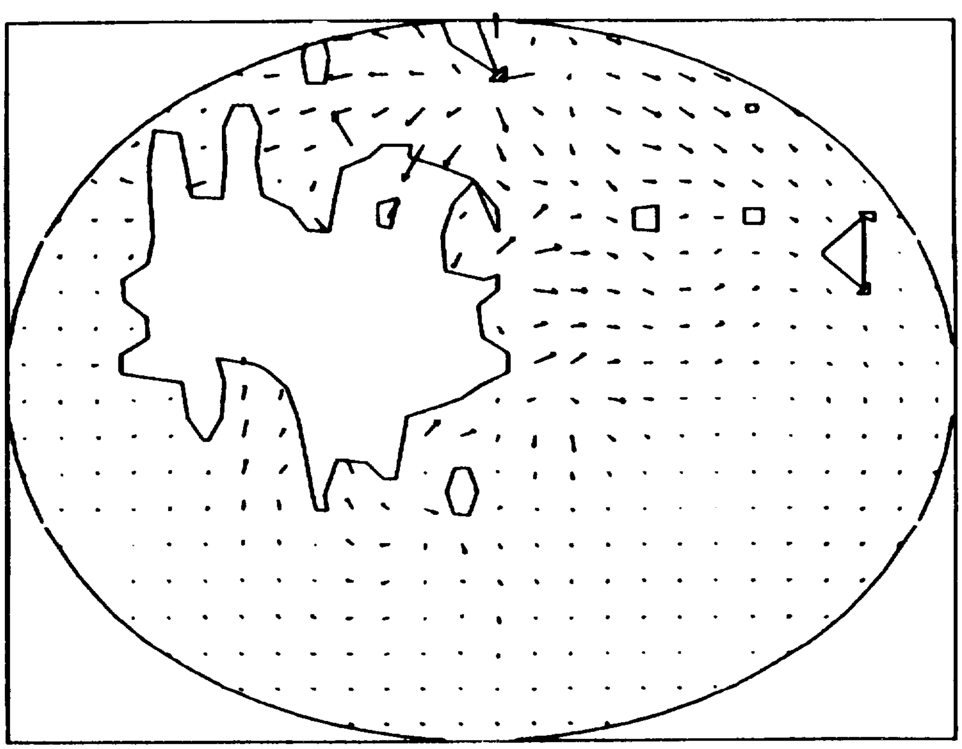

Fig. 17. Velocity vector plot for problem R-12-7: t=0.70 s.

Comparing our composite plot at $t = 0.65$ sec, Fig. 16, with the experimental plot at time $t = 0.67$ sec, Fig. 3 of Reference 15, we see a general breaking away of the bubble from the negative z tank wall and a general shift toward negative x values for the bubble location. There is thus a general agreement between our simulation and experiment; however, the numerical results indicate a less complete separation from the negative z tank wall and a more fragmented bubble structure than does the experiment. There appears to be a reliable long term predictive capability for this type of problem from numerical simulations based on NASA-VOF3D.

Experiment R-12-8 was done with FC-43, and had an applied acceleration of 0.14 g lasting $\Delta t_a = 0.132$ sec. Numerical results of our simulation are displayed in Figs. 18-31 and may be compared at $t = 0.13$ sec with experiment. Our simulation indicates very general qualitative agreement with experiment as to the location of the bubble. The bubble is to be found at negative z values and positive x values and has fluid jetting into its central region. However, the simulation does not agree in detail with experiment; the bubble displays too extended a structure, the jets are too wide and the bubble intersection with the tank wall is not perfectly represented. The NASA-VOF3D simulation has a bubble intersection with the tank wall more nearly in agreement with experiment, but its fluid jet has a double hump structure that is not apparent in the data. Of course, one must not place too much emphasis on detailed agreement with experiment in view of the difficulty of interpreting the two-dimensional trace of three-dimensional data (contained in the experimental plots) in terms of a truly two-dimensional structure displayed in the simulation plots. By comparing our Fig. 31 at $t = 0.85$ sec with the data at $t = 0.84$ sec one again sees general agreement; the bubble has moved toward the middle of the tank in the z coordinate and has shifted to slightly negative x values. However, the numerical results indicate some bubble contact with the negative z tank wall and a fairly complex bubble interface, in contrast to the data. There again appears to be an improvement in the long term predictive ability for general bubble location in NASA-VOF3D.

Experiment R-12-9 was done with Ethanol, and had an applied acceleration of 0.25 g lasting $\Delta t_a = 0.101$ sec. Our simulation results are displayed in Figs. 32-43 and may be compared with experiment in Fig. 1 of Reference 15. There is again qualitative agreement between simulation and data at $t = 0.10$ sec, but the left lobe of our bubble is not firmly in contact with the negative z tank wall. In contrast to the data, the bubble has not moved quite far enough to the positive x direction, and

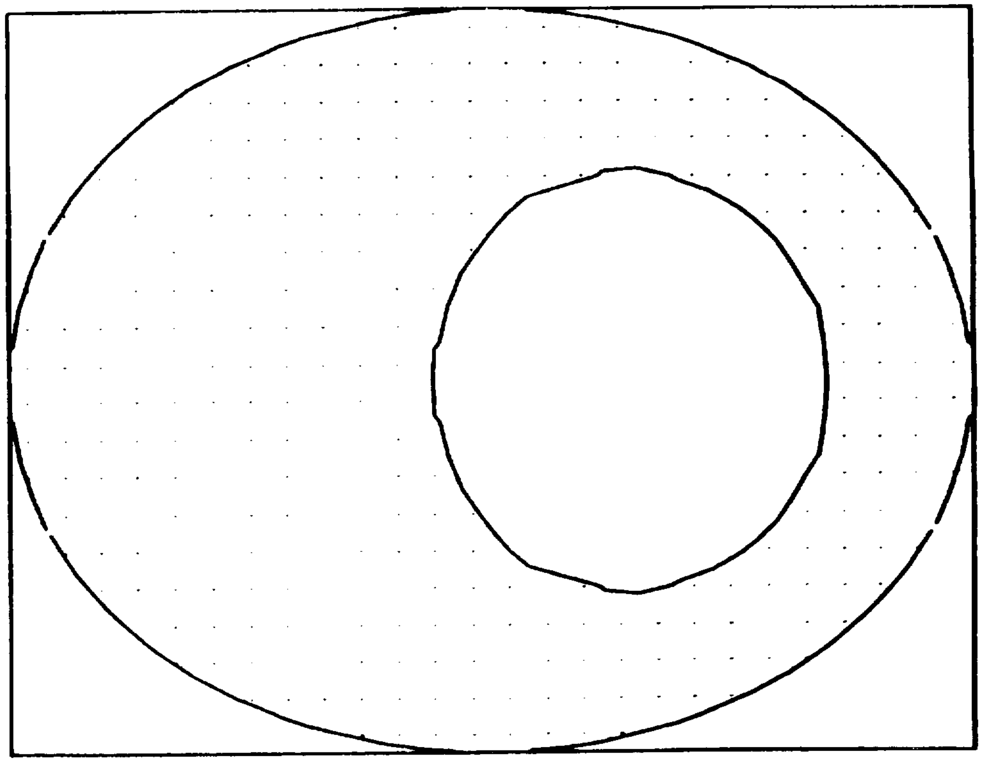

Fig. 18. Velocity vector plot for problem R-12-8: t=0.00 s.

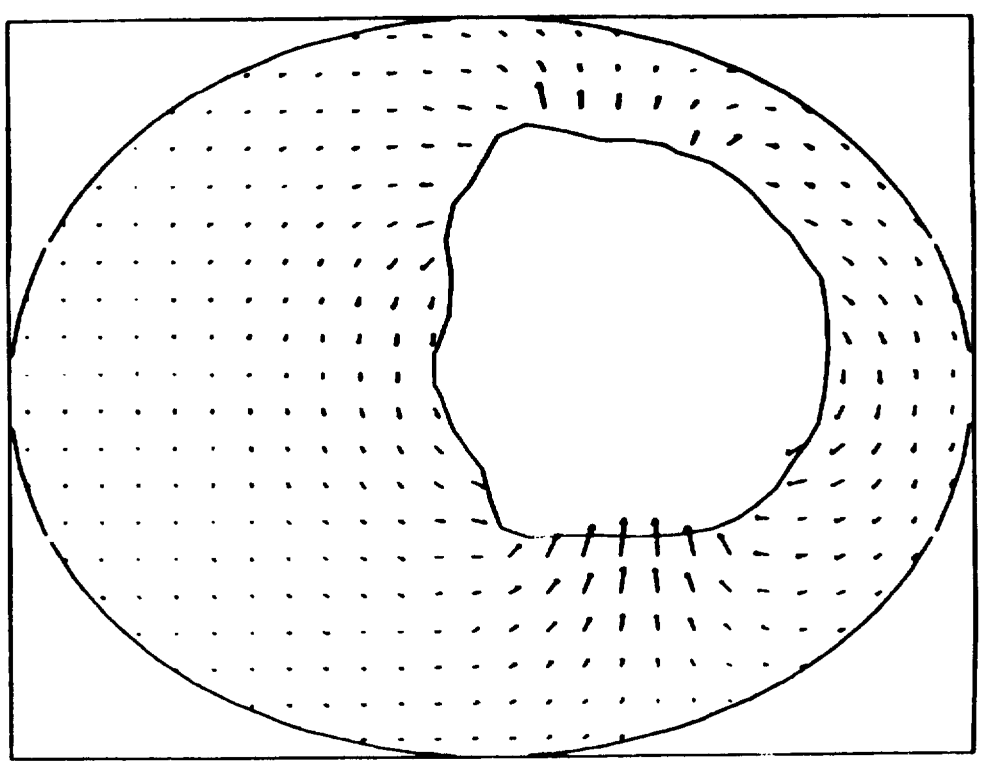

Fig. 19. Velocity vector plot for problem R-12-8: t=0.05 s.

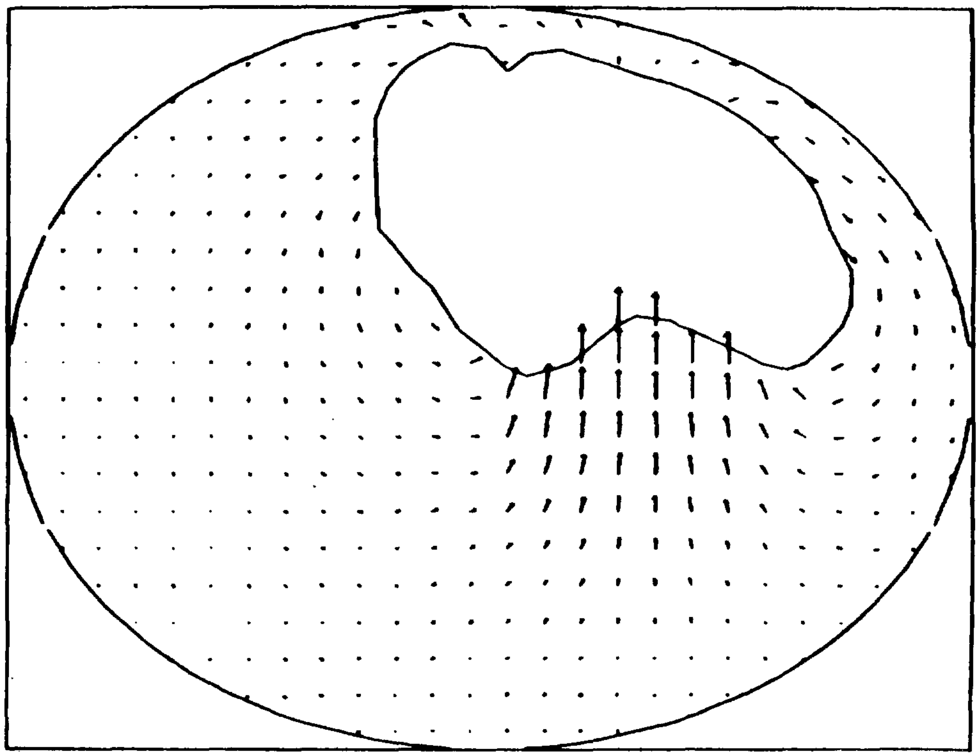

Fig. 20. Velocity vector plot for problem R-12-8: t=0.10 s.

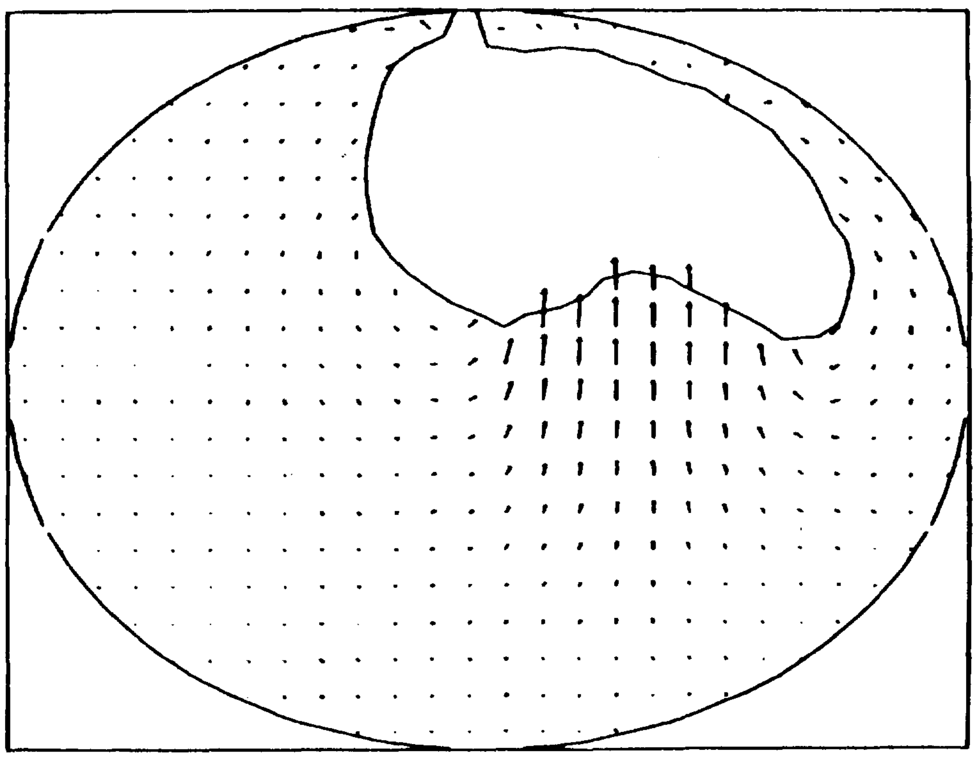

Fig. 21. Velocity vector plot for problem R-12-8: t=0.11 s.

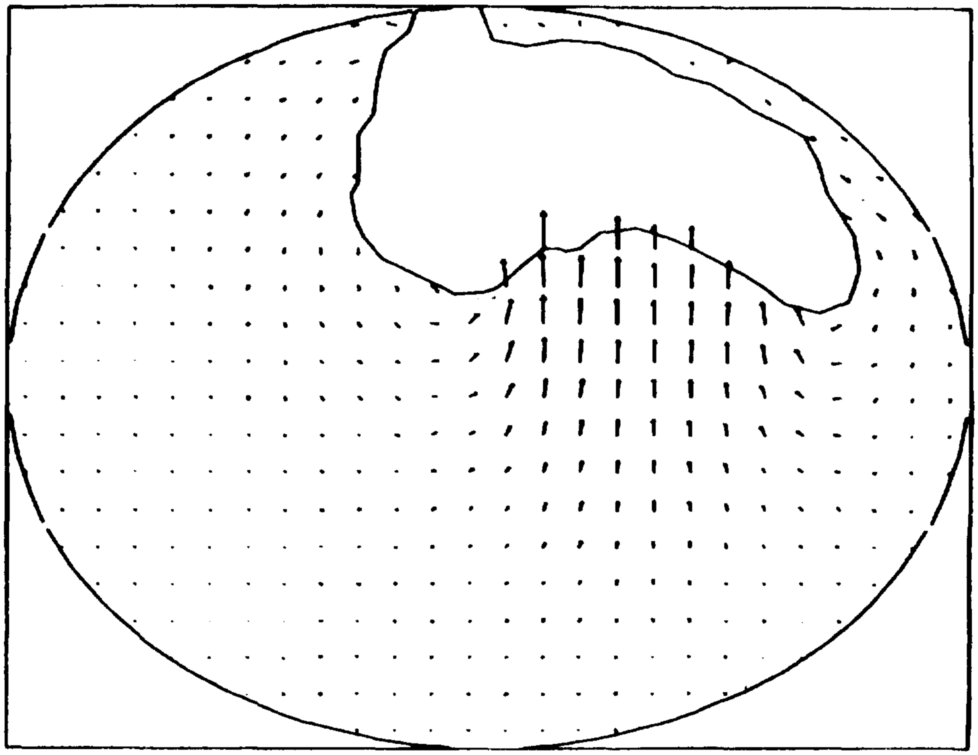

Fig. 22. Velocity vector plot for problem R-12-8: t=0.12 s.

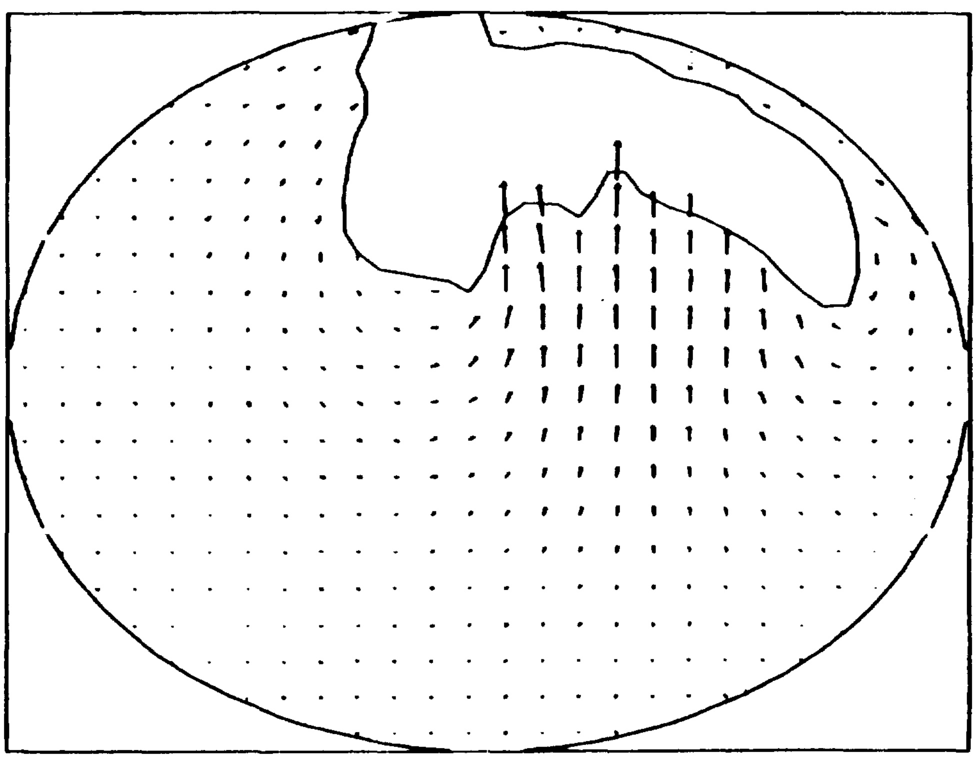

Fig. 23. Velocity vector plot for problem R-12-8: t=0.13 s.

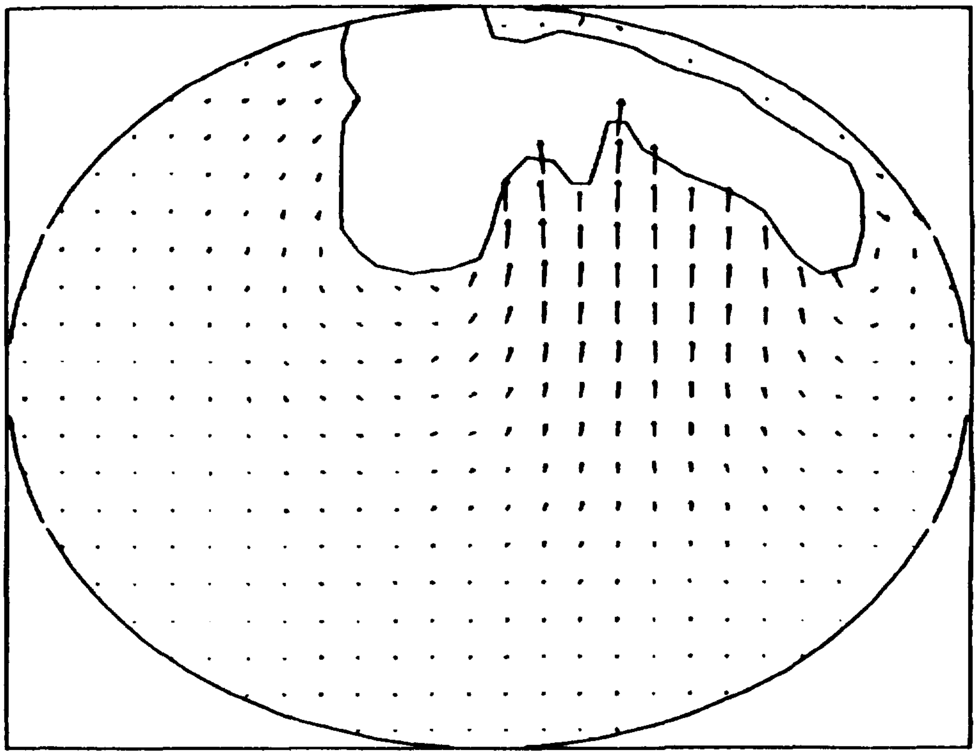

Fig. 24. Velocity vector plot for problem R-12-8: t=0.14 s.

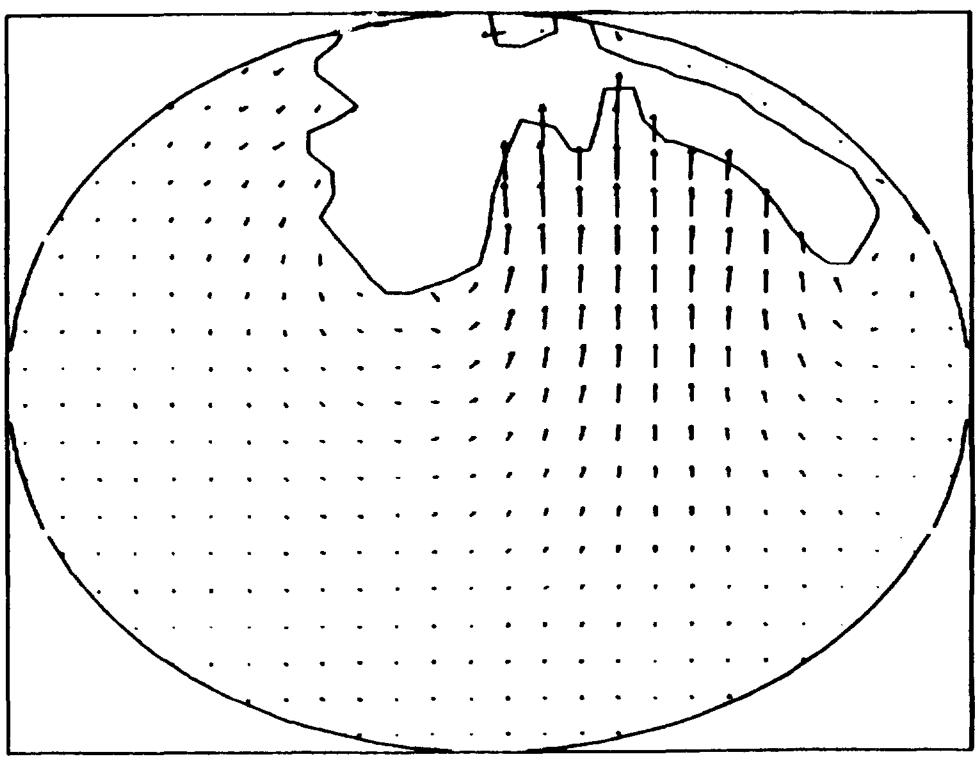

Fig. 25. Velocity vector plot for problem R-12-8: t=0.15 s.

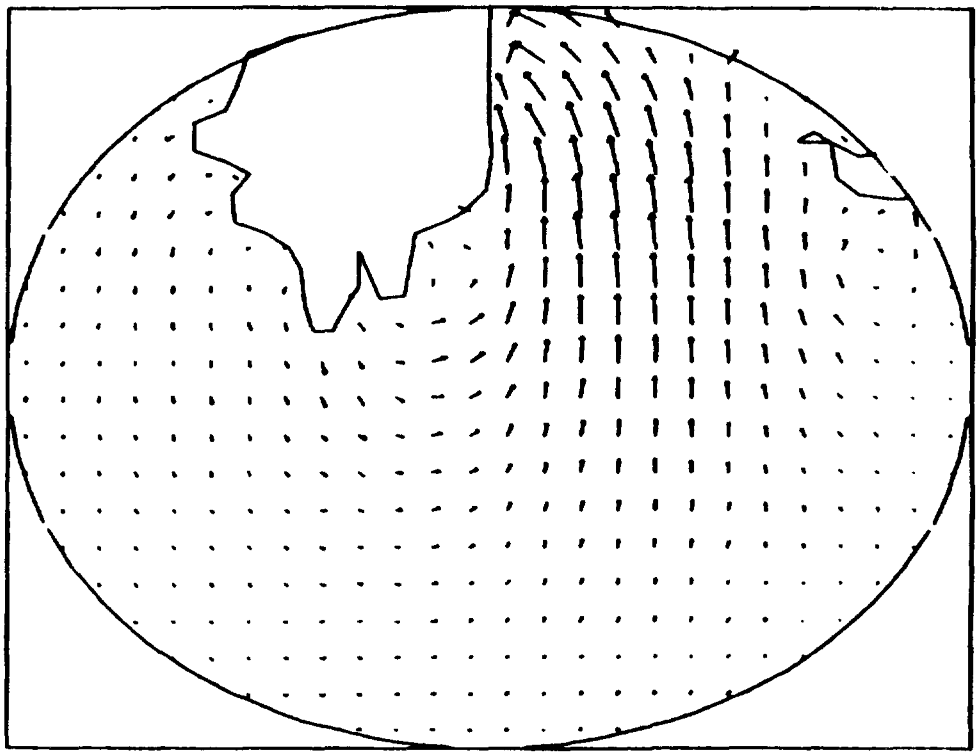

Fig. 26. Velocity vector plot for problem R-12-8: t=0.25 s.

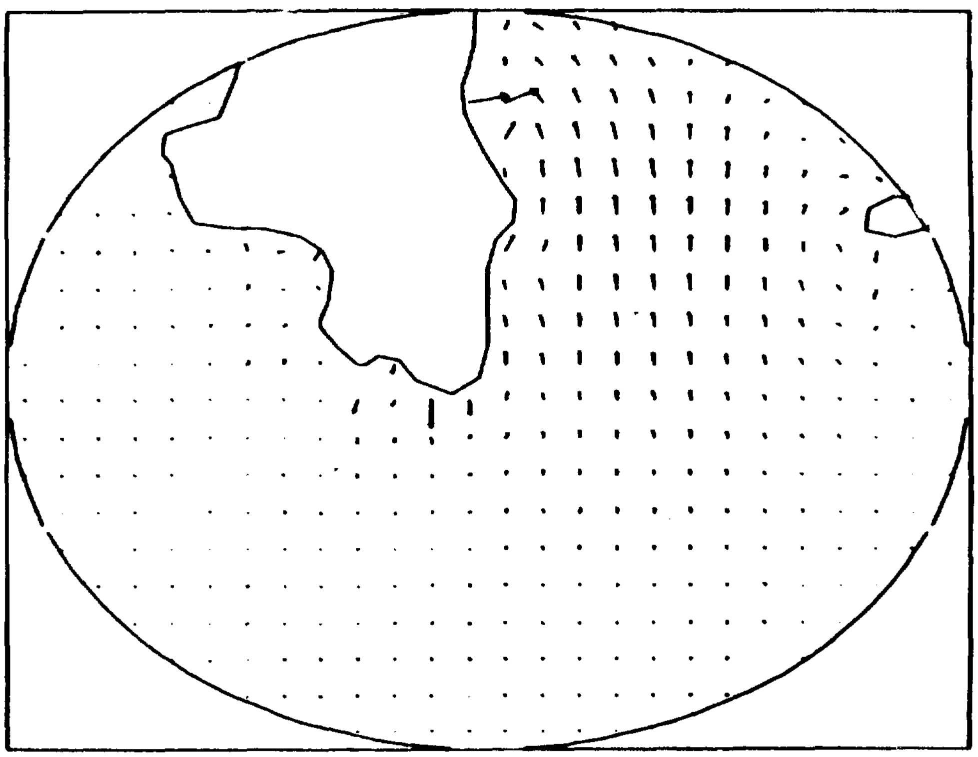

Fig. 27. Velocity vector plot for problem R-12-8: t=0.35 s.

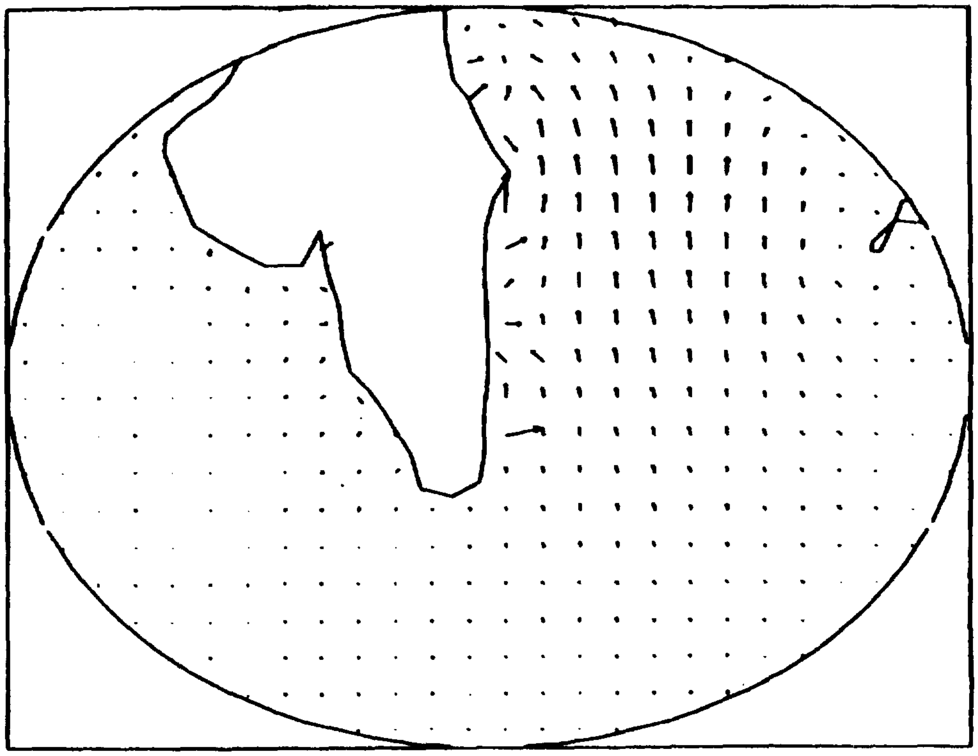

Fig. 28. Velocity vector plot for problem R-12-8: t=0.45 s.

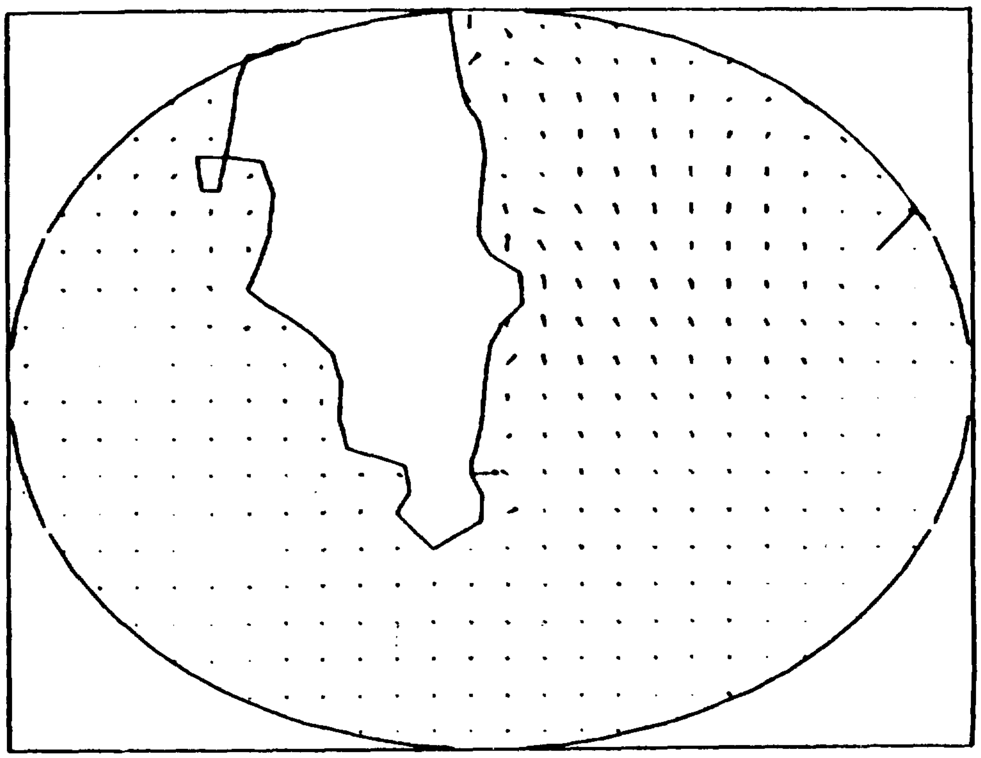

Fig. 29. Velocity vector plot for problem R-12-8: t=0.65 s.

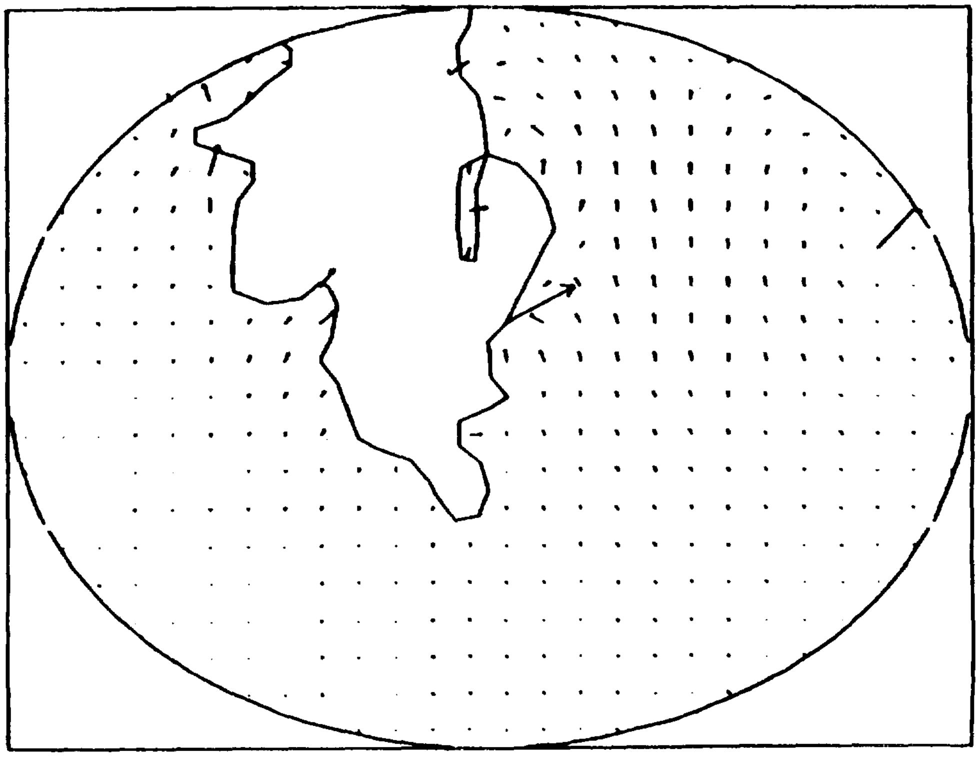

Fig. 30. Velocity vector plot for problem R-12-8: t=0.80 s.

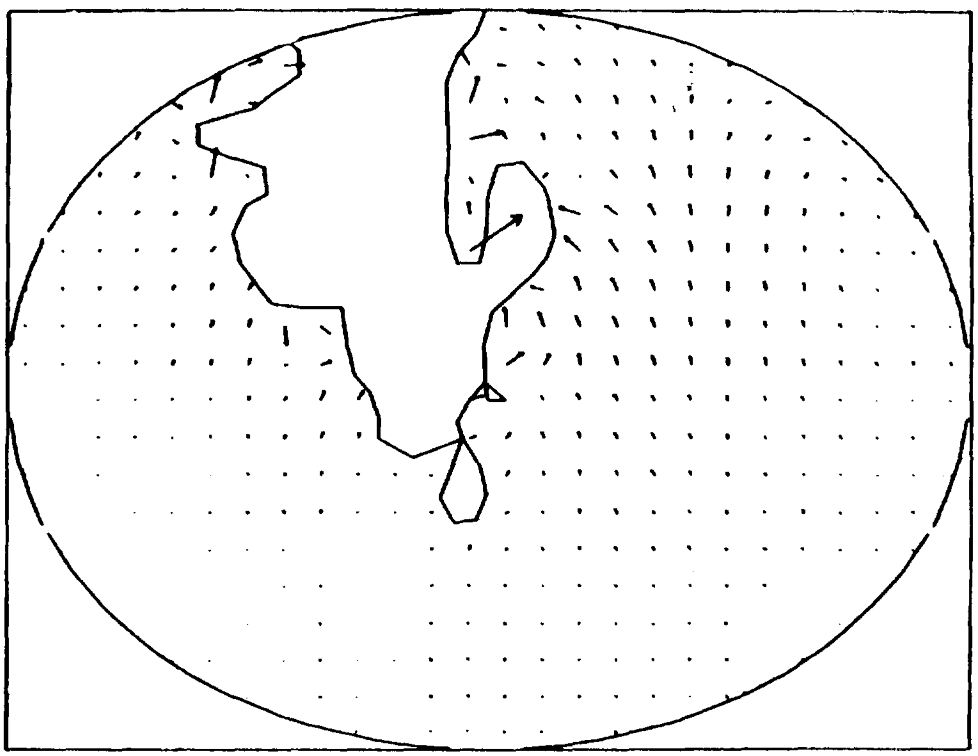

Fig. 31. Velocity vector plot for problem R-12-8: t=0.85 s.

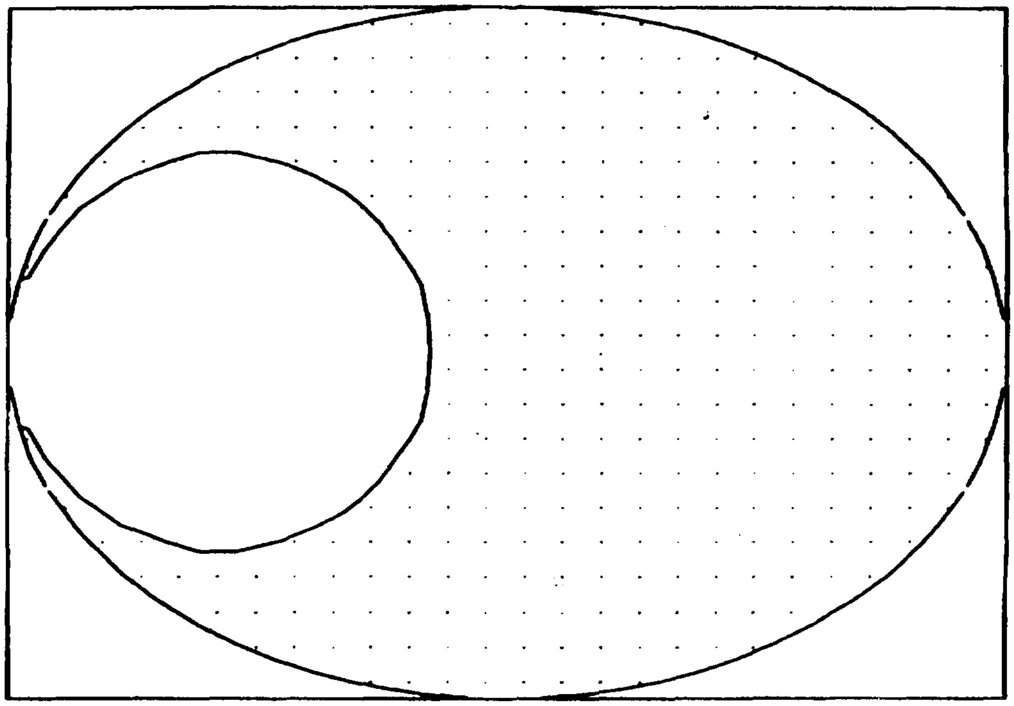

Fig. 32. Velocity vector plot for problem R-12-9: t=0.0 s.

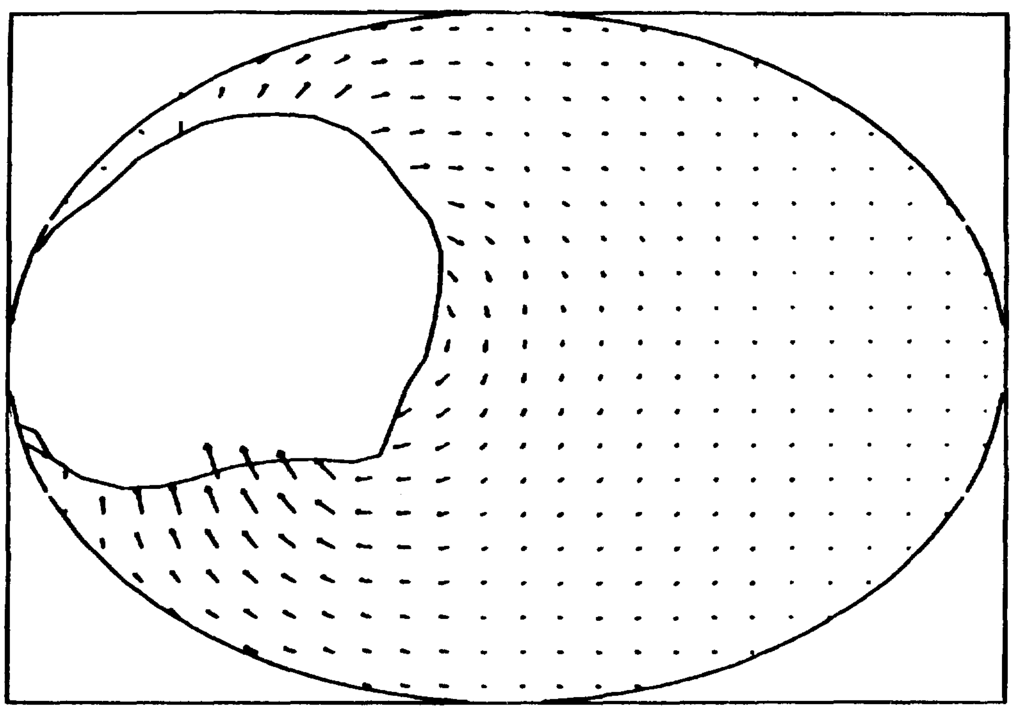

Fig. 33. Velocity vector plot for problem R-12-9: t=0.05 s.

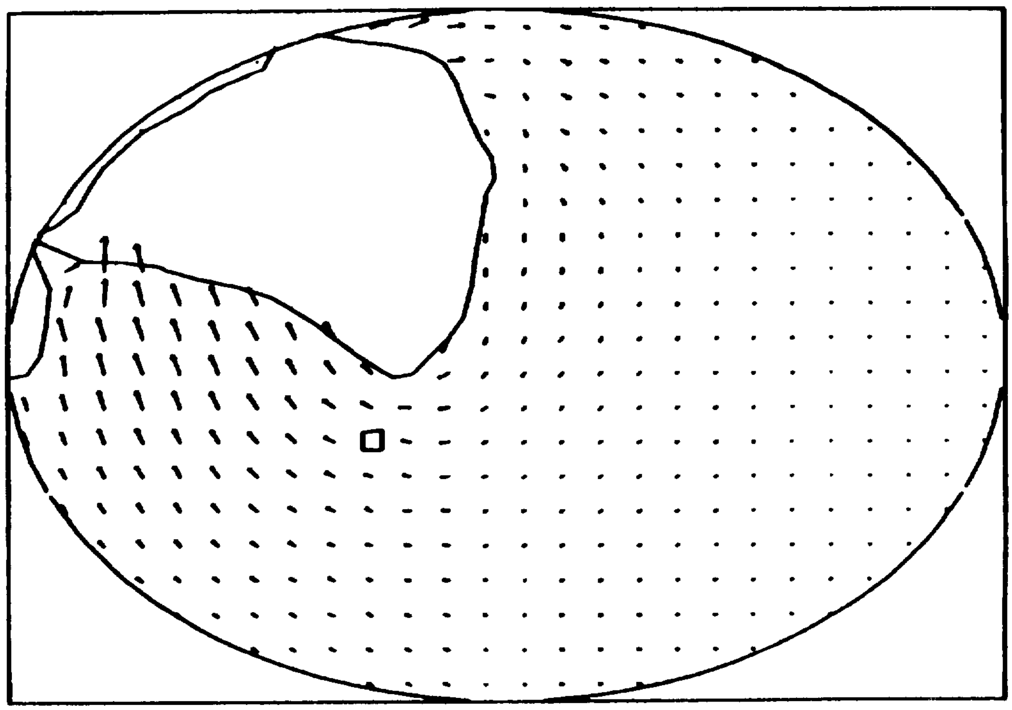

Fig. 34. Velocity vector plot for problem R-12-9: t=0.10 s.

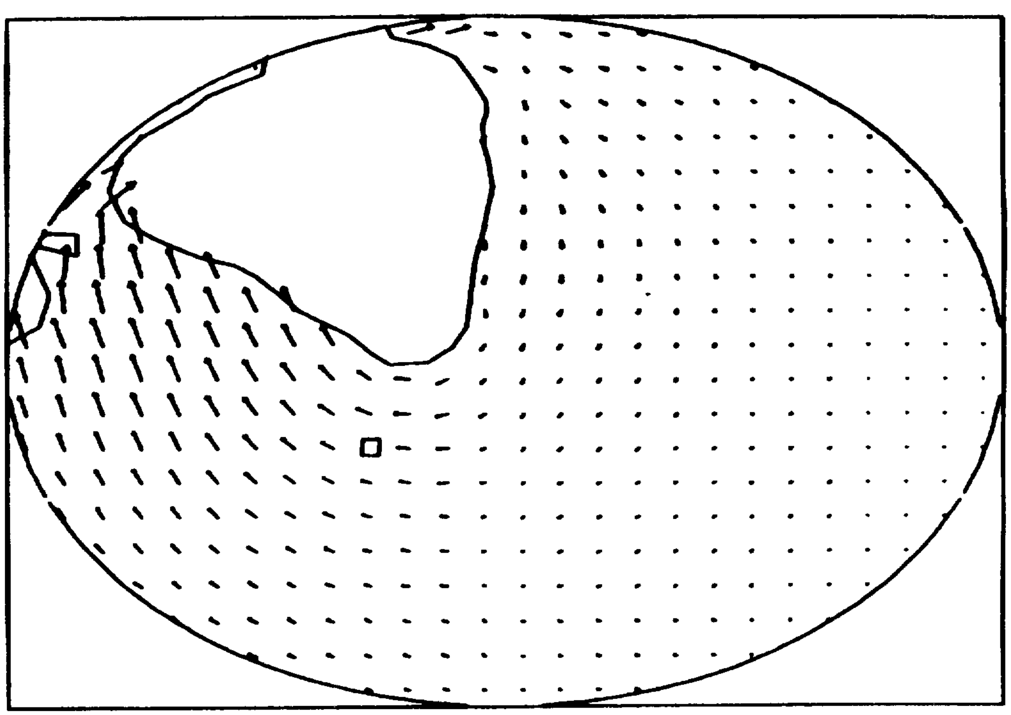

Fig. 35. Velocity vector plot for problem R-12-9: t=0.11 s.

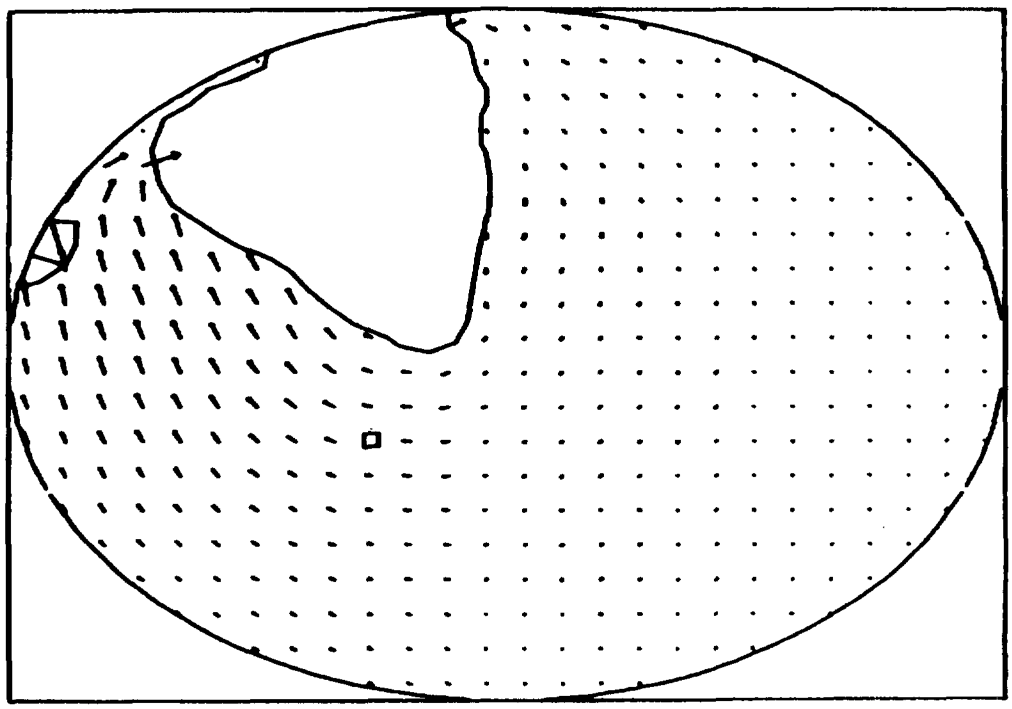

Fig. 36. Velocity vector plot for problem R-12-9: t=0.12 s.

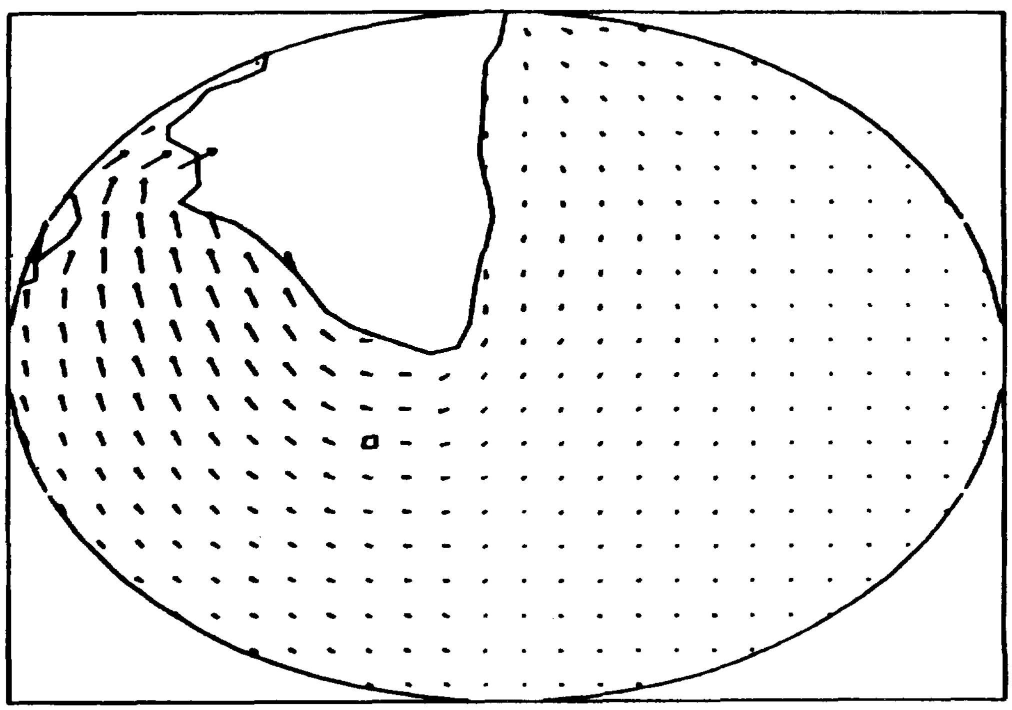

Fig. 37. Velocity vector plot for problem R-12-9: t=0.13 s.

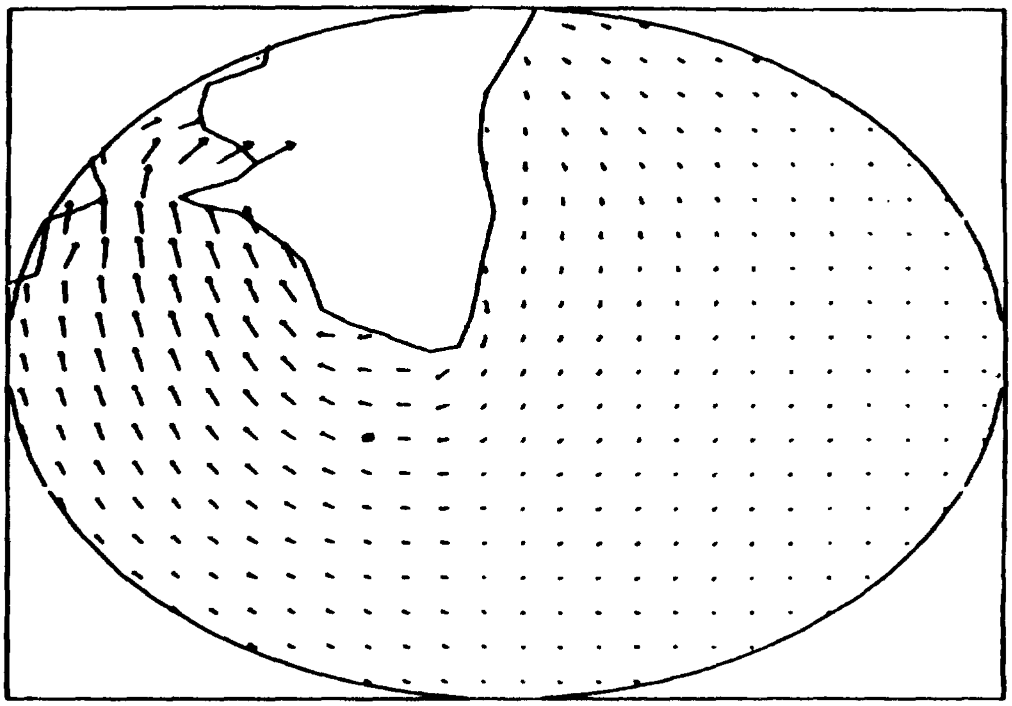

Fig. 38. Velocity vector plot for problem R-12-9: t=0.14 s.

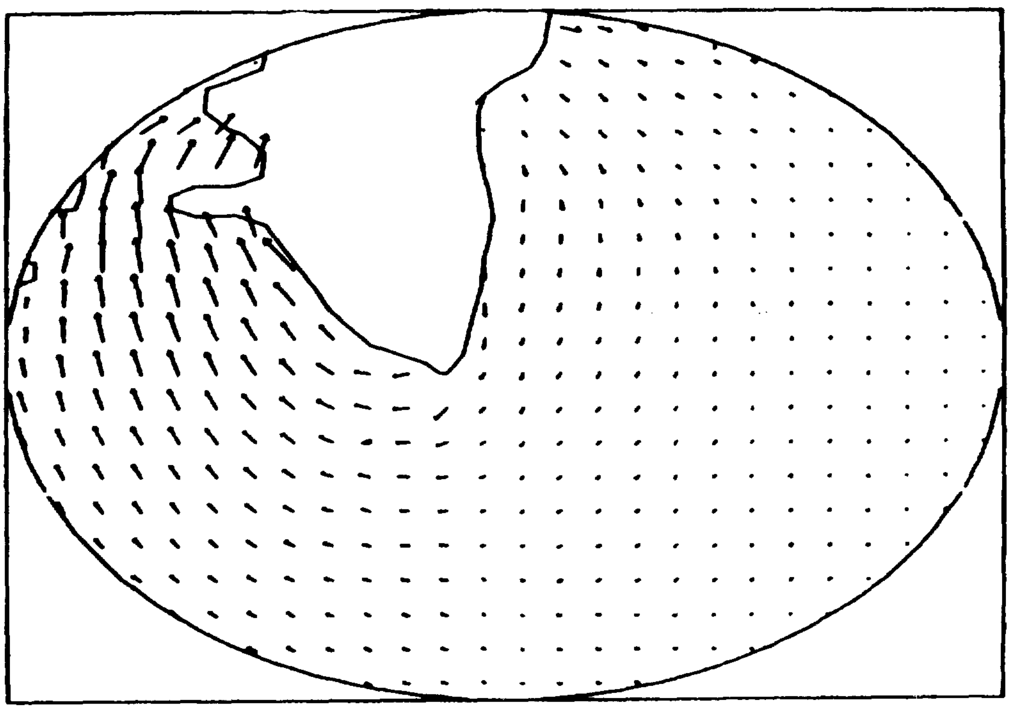

Fig. 39. Velocity vector plot for problem R-12-9: t=0.15 s.

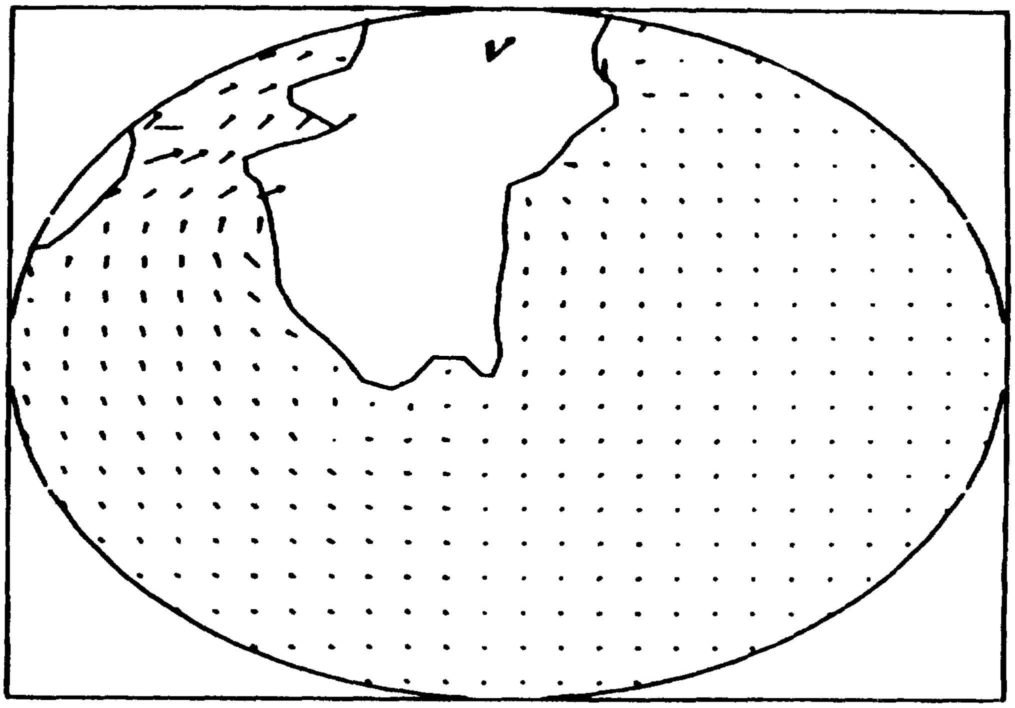

Fig. 40. Velocity vector plot for problem R-12-9: t=0.25 s.

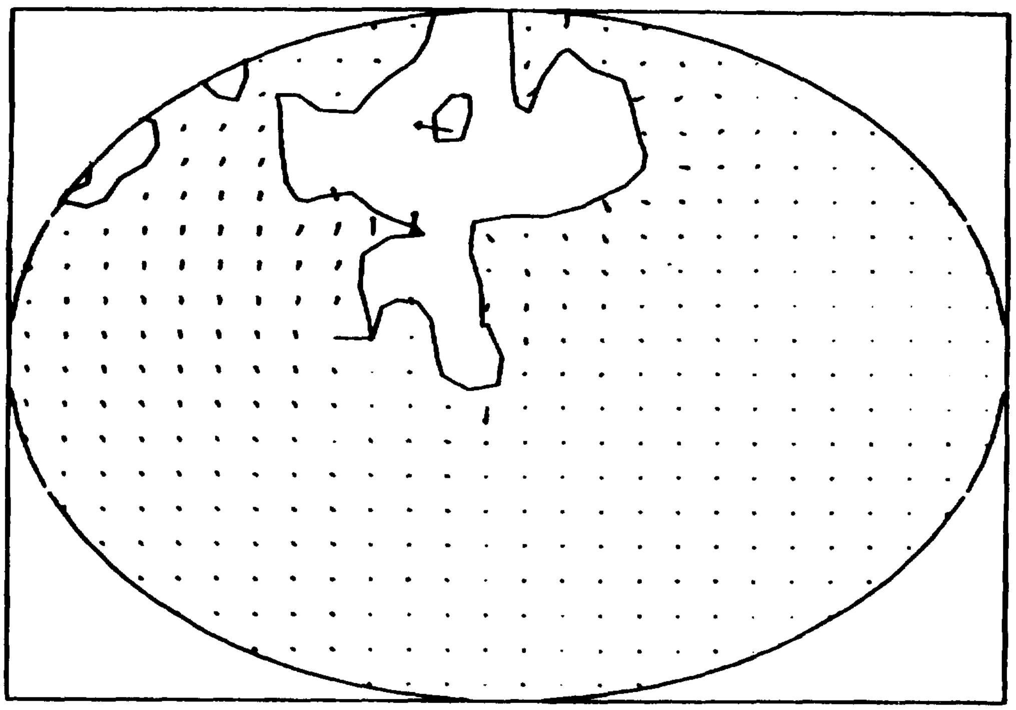

Fig. 41. Velocity vector plot for problem R-12-9: t=0.35 s.

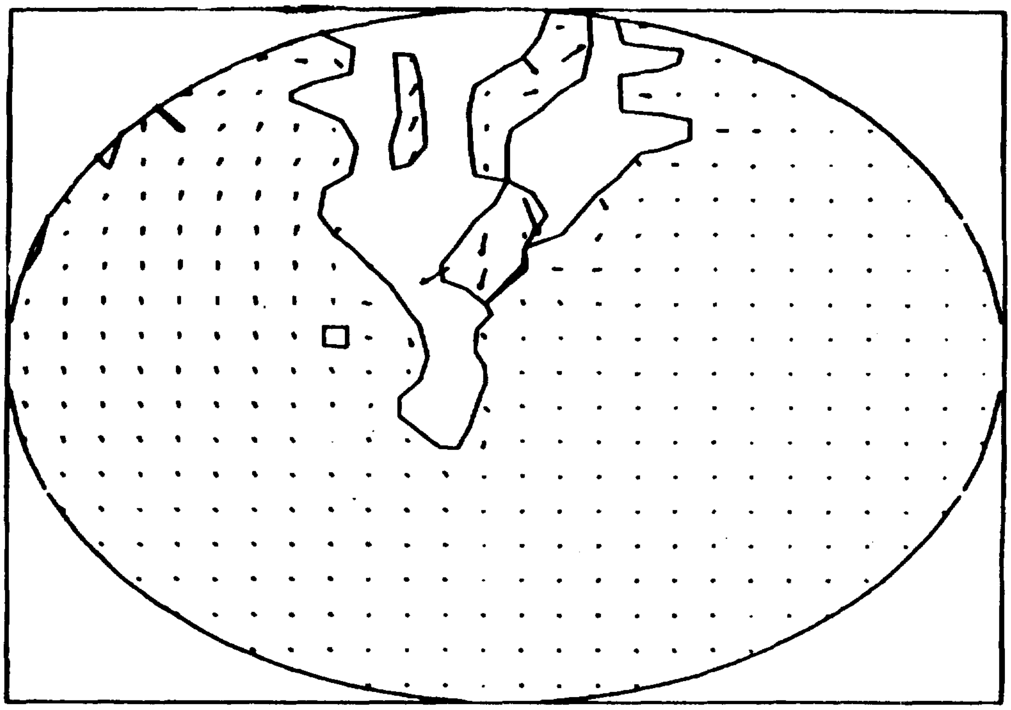

Fig. 42. Velocity vector plot for problem R-12-9: t=0.55 s.

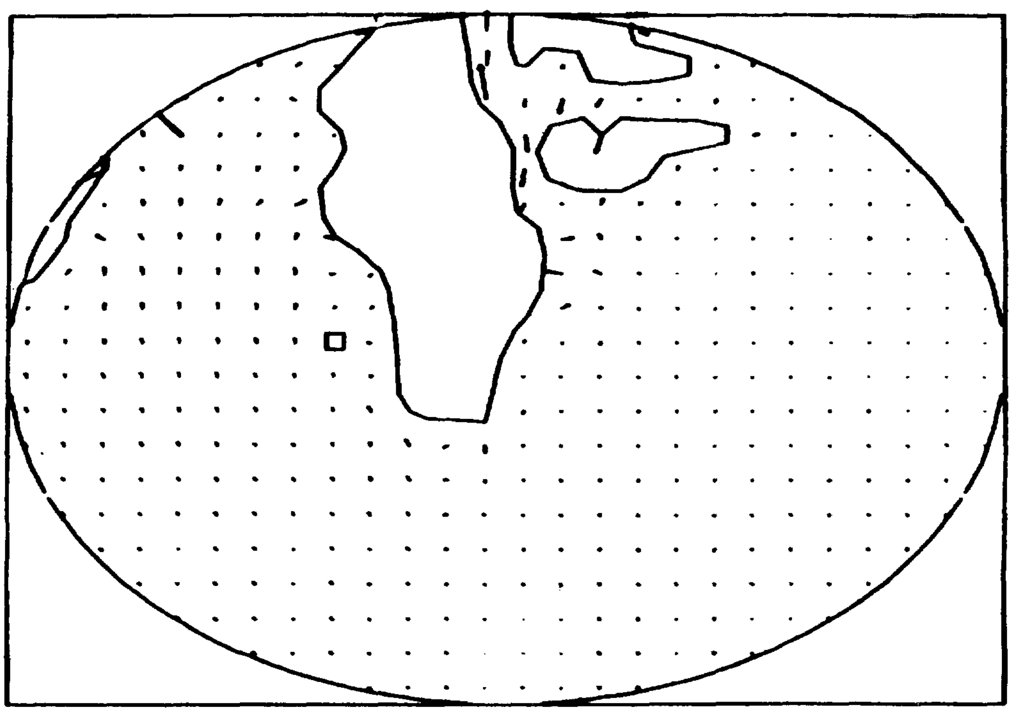

Fig. 43. Velocity vector plot for problem R-12-9: t=0.62 s.

the fluid jetting is too extended a structure. In comparison of our Figs. 42 and 43, at $t = 0.60$ and 0.65 sec respectively, with the data at $t = 0.62$ sec, there is a general qualitative agreement that the bubble has broken away from the negative z tank wall and is roughly in the center of the system, but the numerical bubble structure is much more fragmented than indicated by the data, the numerical bubble does not extend as far to positive x values as does the data, and the numerical bubble is concentrated at negative z values while the data are at positive z values. There seems also in this problem to be an improvement in long-term predictive capability for bubble location in NASA-VOF3D.

VI. FURTHER DEVELOPMENTS

The most urgent needs for improvements in the present version of NASA-VOF3D are: a) inclusion of a realistic, yet tractable, wall-adhesion algorithm, b) improvements in the decision-making algorithms which choose cell-by-cell between several possible evaluations of the NF array, c) a general "shakedown" period of improving the running of the code, and d) a physical model for wall-friction. We discuss these items in order.

The prototype of an acceptable and accurate wall-adhesion algorithm is given for 2D in Reference 11. The 2D version involves 1) initial calculation of trigonometric factors defining the orientation of obstacles with respect to the coordinate surfaces, 2) initial calculation of an array specifying the cell geometry of obstacles, 3) a time-dependent decision making procedure that determines whether surface-tension or wall-adhesion is to be applied at a cell interface and, if wall-adhesion is selected, whether the adhesion force points "up" or "down" the obstacle wall. In 3D there must be, in addition, a time-dependent calculation of trigonometric factors related to the angle which the adhesion force makes with the "up" direction at the obstacle wall. There will also be a somewhat more intricate procedure for selecting the wall-adhesion option and specifying the adhesion force. It has been shown in 2D that accurate specification of the wall-adhesion force is necessary for accurate prediction of some flow features.

The present decision-making algorithms for selection of surface-cell orientation can scarcely be regarded as a final solution for this important topic; they suffice for the present problems, and, presumably, for many others. We have abundant numerical evidence that there will always be <u>some</u> NF value that yields a good surface-tension calculation; the problem is to find the appropriate NF value in almost all cases. The present algorithms give appropriate NF values a large percentage of the time for a number of examples. They also limit fairly effectively the damage that an inappropriate choice of NF will do to the simulation. The problem of devising algorithms that optimize the percentage of appropriate NF values in the widest variety of cases must be left for future research.

The code requires a period of time devoted to running many problems and correcting little awkwardnesses that will inevitably show up during this process. Possibly, some improvements in the conjugate residual technique will further speed up the code. Certainly the number of cases to which the code can be routinely applied would increase. The anticipated result of such a "shakedown" period would be a substantial increase in the amount of "user-friendliness" displayed by the code.

NASA-VOF3D was developed as a design tool to study bulk fluid and vapor motion in a low-g environment. It meets this goal. Even in its present form it probably suffices for many applications. But if it were to be used for precision comparison with experiment, then the aesthetically annoying feature that a physical mechanism--wall friction--is here modeled by a variety of sometimes competing numerical errors will begin to degrade the code's overall performance. In such a context it would be useful to model the wall-friction explicitly.

We close by noting that there are contexts in which one is interested in a highly viscous fluid in a low-g environment. To treat such a case by the present code would require a substantial revision of the solution algorithm. Viscous forces would have to be included more directly in the pressure iterations.

ACKNOWLEDGMENT

We are indebted to Drs. John K. Dukowicz and John R. Travis for assistance in implementing the conjugate algorithms.

REFERENCES

1. Nichols, B. D. and Hirt, C. W., "Methods for Calculating Multi-Dimensional, Transient Free Surface Flows Past Bodies," Proc. of the First Intern. Conf. Num. Ship Hydrodynamics, Gaithersburg, MD, October 1975.

2. Nichols, B. D., Hirt, C. W., and Hotchkiss, R. S., "SOLA-VOF: A Solution Algorithm for Transient Fluid Flow with Multiple Free Boundaries," Los Alamos Scientific Laboratory report LA-8355, 1980.

3. Hirt, C. W. and Nichols, B. D., 1981, J. Comput. Phys. 39, 201.

4. Johnson, W. E., "Development and Application of Computer Programs Related to Hypervelocity Impact," Systems, Science and Software report 3SR-353, 1970.

5. Ramshaw, J. D. and Trapp, J. A, 1976, J. Comput. Phys. 21, 438.

6. McMaster, W. H. and Gong, E. Y., "PELE-IC User's Manual," Lawrence Livermore Laboratory report UCRL-52609, 1979.

7. McMaster, W. H., Quiñones, D. F., Landram, C. S., Norris, D. M., Gong, E. Y., Machen, N. A., and Nickell, R. E., "Applications of the Coupled Fluid-Structure Code PELE-IC to Pressure Suppression Analysis, Annual Report to NRC for 1979," NUREG/CR-1179, UCRL-52733, 1980.

8. Kershner, J. D. and Mader, C. L., "2DE: A Two-Dimensional Continuous Eulerian Hydrodynamic Code for Computing Multicomponent Reactive Hydrodynamic Problems," Los Alamos Scientific Laboratory report LA-4846, 1972.

9. Noh, W. H. and Woodward, P., "The SLIC (Simple Line Interface Calculation)," in Lecture Notes in Physics, Vol. 59; proceedings of the Fifth Intern. Conf. on Numerical Methods in Fluid Dynamics, eds. A. I. van de Vooren and P. J. Zandbergen, Springer-Verlag, 1976.

10. Hotchkiss, R. S., "Simulation of Tank Draining Phenomena with the NASA SOLA-VOF Code," Los Alamos Scientific Laboratory report LA-8163-MS, 1979.

11. Torrey, M. D., Cloutman, L. D., Mjolsness, R. C., and Hirt, C. W., "NASA-VOF2D: A Computer Program for Incompressible Flows with Free Surfaces," Los Alamos National Laboratory report LA-10612-MS, 1985.

12. Symons, E. P., "Draining Characteristics of Hemispherically Bottomed Cylinders in a Low-Gravity Environment," NASA Technical paper 1297, 1978.

13. Carney, M., "Results of Shuttle/Centaur Zero-G Tests," NASA LeRC Internal report, March 1985.

14. Sicilian, J. M. and Hirt, C. W., "HYDR3-D, A Solution Algorithm for Transient 3D Flows," Flow Science, Inc. report FSI-84-00-3, 1985.

15. Hochstein, J. I., "Computational Prediction of Propellant Motion During Separation of a Centaur G-Prime Vehicle from the Shuttle," Washington University report WU/CFDL-85/1, December 1985.

APPENDIX

FORTRAN LISTING OF THE NASA-VOF3D SOURCE CODE

```
*CD SLCOM1                                                              SLCOM1
      PARAMETER (NQ=6, IBAR2=20, JBAR2=20, KBAR2=20, IBASC=NQ*IBAR2     SLCOM1
     1 *JBAR2*KBAR2)                                                    SLCOM1
C                                                                       SLCOM1
C     ****  NOTE: IBAR2,JBAR2,KBAR2 MUST EQUAL IMAX,JMAX,KMAX OF MESH   SLCOM1
C                 GENERATION WHEN CONJUGATE RESIDUAL SOLUTION METHOD    SLCOM1
C                 IS USED IN THE PRESSURE ITERATION                     SLCOM1
C                                                                       SLCOM1
      PARAMETER (NQ2=6, NVEC=NQ2*IBAR2*KBAR2)                           SLCOM1
C                                                                       SLCOM1
C     **** NOTE: TWX(NVEC) IS WORKING ARRAY FOR PLOT ROUTINES           SLCOM1
C                   NVEC MUST EQUAL NQ2 TIMES THE LARGER OF             SLCOM1
C                   IBAR2*JBAR2, IBAR2*KBAR2, JBAR2*KBAR2               SLCOM1
C                                                                       SLCOM1
      COMMON /SLCM1/ BASC(IBASC) /SLCM2/ BASC1(IBASC) /SLCM3/ BASC2     SLCOM1
     1 (IBASC) /SLCM4/ BASC3(IBASC) /SLCM5/ TWX(NVEC)                   SLCOM1
C                                                                       SLCOM1
      COMMON /SSCM1/ AA(1), ALPHA, AUTOT, CLK, CYL, CYCLE, DAT, DELMN,  SLCOM1
     1 DELT, DELX(100), DELXRL, DELXRR, DELY(100), DELYRBK, DELYRF, DELZ SLCOM1
     2 (100), DELZRB, DELZRT, DLZ, DMPX, DMPY, DMPZ, DTVIS, DXMN(10),   SLCOM1
     3 DYMN(10), DZMN(10), EPSI, FIXL(3), FIXR, FIYB, FIYT, GRDBN(24),  SLCOM1
     4 GX, GY, GZ, I, IBAR, ICLIP, IGRD, II0, II1, II2, II3, II4, IJK,  SLCOM1
     5 IJKM, IJKP, IJMK, IJMKP, IJPK, IJPKM, IJPKP, IM1, IM2, II5, II6, SLCOM1
     6 II7, IMAX, IMJK, IMJKM, IMJKP, IMJPK, IPJK, IPJKM, IPJKP, IPJMK, SLCOM1
     7 IPJPK, IMJMK, IJMKM, IMJPKP, IPJMKP, IPJPKM, IPJPKP, IDEFM, ISOR, SLCOM1
     8 ITER, ITITLE1(3), ITITLE2(3), IZOOM, J, JBAR, JM1, JM2, JMAX, JNM SLCOM1
     9 , K, KBAR, KM1, KM2, KMAX, LPR, NAME(8), NC, NCR1, NCR2, NNX, NNY SLCOM1
C                                                                       SLCOM1
      COMMON /SSMC2/ NNZ, NFCAL, NZX, NZY, NZZ, NTITLE1, NTITLE2, NU,   SLCOM1
     1 NUMTD, NXL(11), NXLBL, NXR(10), NYL(11), NYLBL, NYR(10), NZL(11), SLCOM1
     2 NZLBL, NZR(10), OMG, PLTDT, PRTDT, Q(50,50), RDX(100), RDY(100), SLCOM1
     3 RDZ(100), RIJK, RX(100), RXI(100), RDXP(100), RDYP(100), RDZP(100 SLCOM1
     4 ), RY(100), RYJ(100), RZ(100), RZK(100), STIM, T, TWTD, TDDT, TD, SLCOM1
     5 TLIMD, TLM, TWFIN, TWPLT, TWPRT, UI, VELMX, VI, WB, WBK, WF, WI,  SLCOM1
     6 WL, WR, WT, X(100), XBL, XBLC, XBR, XC(10), XCONV(3), XI(100), XL SLCOM1
     7 (11), XLABLE, Y(100), YBBK, YBF, YBFC, YC(10), YCONV(3), YJ(100), SLCOM1
     8 YL(11), YLABLE, Z(100), ZBB, ZBT, ZC(10), ZCQ(50), ZK(100), ZL(11 SLCOM1
     9 ), ZLABLE, ZMN, ZMX                                              SLCOM1
C                                                                       SLCOM1
      COMMON /SSCM4/ JOP(100), JC2PI, SANG, CSANG, ICSURF, IEQUIB, SIGMA SLCOM1
     1 , CANGLE, ISURFT, TANCA, STHJ(99), IORDER, CTHJ(99), CTHJBK(99), SLCOM1
     2 STHJBK(99), FLHT, AVE, FLG, UDUM, VDUM, WDUM, EMF, EMF1, NOWALL, SLCOM1
     3 RHOF, VCHGT, FLGC, FNOC, PR(100), NFLGC, NOCON, VOFTOT, PI, EM6  SLCOM1
C                                                                       SLCOM1
      COMMON /SSCM5/ GXA(NVEC), GYA(NVEC), RADPS                        SLCOM1
C                                                                       SLCOM1
      COMMON /SSCM5A/ NOBS, OA2(10), OA1(10), OB2(10), OB1(10), OC2(10), SLCOM1
     1 OC1(10), IOH(10), NQBS, QA2(10), QA1(10), QB2(10), QB1(10), QC2(1 SLCOM1
     2 0), QC1(10), IQH(10), QA3(10), QB3(10), QC3(10), QD3(10), QD2(10) SLCOM1
     3 , QD1(10)                                                        SLCOM1
C                                                                       SLCOM1
      COMMON /SSCM6/ IC1(10), IC2(10), ICVEW(10), IPERC(10), IPERS(10), SLCOM1
     1 IPERV(10), IS1(10), IS2(10), ISVEW(10), IV1(10), IV2(10), IVVEW(1 SLCOM1
     2 0), JC1(10), JC2(10), JS1(10), JS2(10), JV1(10), JV2(10), KC1(10) SLCOM1
     3 , KC2(10), KS1(10), KS2(10), KV1(10), KV2(10), LPMAX, NAC(10),   SLCOM1
     4 NAS(10), NAV(10), NCPLTS, NSPLTS, NVEWS, NVPLTS, RR(100), RRI(100 SLCOM1
     5 ), XCA(10), XEA(10), YCA(10), YEA(10), ZCA(10), ZEA(10), ZLAST   SLCOM1
C                                                                       SLCOM1
C * * NOTE ZLAST IS END OF COMMON BLOCK DUMPPED TO TAPE                 SLCOM1
C                                                                       SLCOM1
      DIMENSION U(1), V(1), W(1), UN(1), VN(1), WN(1), BETA(1), D(1), P( SLCOM1
     1 1)                                                               SLCOM1
C                                                                       SLCOM1
      EQUIVALENCE (BASC(1),U), (BASC(2),V), (BASC(3),W), (BASC(4),UN),  SLCOM1
     1 (BASC(5),VN), (BASC(6),WN), (BASC1(1),BETA), (BASC1(2),D), (BASC1 SLCOM1
     2 (3),P)                                                           SLCOM1
C                                                                       SLCOM1
      DIMENSION UVV(1), VVV(1), WVV(1), XPC(1), YPC(1), ZPC(1)          SLCOM1
C                                                                       SLCOM1
      EQUIVALENCE (TWX(1),UVV), (TWX(2),VVV), (TWX(3),WVV), (TWX(4),XPC) SLCOM1
     1 , (TWX(5),YPC), (TWX(6),ZPC)                                     SLCOM1
C                                                                       SLCOM1
      DIMENSION F(1), NF(1), PS(1), PETA(1), FN(1), AR(1), AT(1), ABK(1) SLCOM1
     1 , AC(1), PN(1)                                                   SLCOM1
C                                                                       SLCOM1
      EQUIVALENCE (BASC2(1),F), (BASC2(2),FN), (BASC2(3),PETA), (BASC2(4 SLCOM1
     1 ),NF), (BASC2(5),PS), (BASC2(6),PN)                              SLCOM1
C                                                                       SLCOM1
      EQUIVALENCE (BASC3(1),AR), (BASC3(2),AT), (BASC3(3),ABK), (BASC3(4 SLCOM1
     1 ),AC)                                                            SLCOM1
```

```
      C
            DIMENSION NFP(1), NFS(1), NFO(1)
      C
            EQUIVALENCE (BASC1(4),NFP), (BASC1(5),NFS), (BASC1(6),NFO)
      C
            INTEGER CYCLE, WL, WR, WT, WB, WF, WBK, TD
            REAL LONG, NU
      C
      *CD SLCOM2
      C * * SSCM8 IS FOR PERSPECTIVE GRAPHICS
      C
            COMMON /SSCM8/ CSPHI, CSTH, ETAMN, ETAMX, GDRAT, SNPHI, SNTH, XCC,
           1 XEB, XIMN, XIMX, YCC, YEB, ZCC, ZEB
      C
      *DK CONTROL
            PROGRAM CONTROL (INPUT,XOUTPUT,TTY,TAPE59=TTY,TAPE10=INPUT,TAPE9
           1 =XOUTPUT,TAPE7,TAPE8,TAPE2)
      C
      *CA SLCOM1
      C * * READ PROBLEM INPUT DATA
      C
            CALL RINPUT
      C
      C * * SKIP ALL SET-UP ROUTINES FOR TAPE RESTART RUN
      C
            IF (TD) 10,10,20
         10 CONTINUE
      C
      C * * READ PROBLEM SET-UP DATA FOR THE MESH, GRAPHICS, ETC.
      C
            CALL MESHSET
            CALL RGRAFIC
            CALL RCONTUR
            CALL SETUP
            CALL FLMSET
      C
      C * * THE NUMERICAL SOLUTION ALORITHM AND CYCLIC CONTROL
      C     OF THE CALCULATION IS PROVIDED BY SUBROUTINE SOLA
      C
            CALL SOLA
            CALL EXITA (405)
      C
      C * * TAPE DUMP RESTART
      C
         20 CALL SOLA
            CALL EXITA (100)
      C
            END
      *DK ASET
            SUBROUTINE ASET
      *CA SLCOM1
      C ***
      C *** CONIC FCN=OA2*X*X+OA1*X+OB2*Y*Y+OB1*Y+OC2*X*Y+OC1
      C *** INSIDE FCN=NEGATIVE VALUE
      C *** IOH=1 ADD OBS INSIDE FCN, IOH=0 SUBTRACT OBS INSIDE FCN
      C ***
            DIMENSION IFLG(5), DIS(4), XM(5), YM(5)
      C
            EM6=1.0E-6
            DO 20 K=1,KMAX
            DO 20 J=1,JMAX
            DO 20 I=1,IMAX
            IJK=NQ*(II5*(K-1)+IMAX*(J-1)+(I-1))+1
            AR(IJK)=1.0
            AT(IJK)=1.0
            ABK(IJK)=1.0
            AC(IJK)=1.0
            BETA(IJK)=0.0
            IF (I.EQ.IM1.AND.WR.LE.2) AR(IJK)=0.0
            IF (K.EQ.KM1.AND.WT.LE.2) AT(IJK)=0.0
            IF (J.EQ.JM1.AND.WBK.LE.2) ABK(IJK)=0.0
            IF (I.EQ.1.AND.WL.LE.2) GO TO 10
            IF (I.EQ.IMAX.AND.WR.LE.2) GO TO 10
            IF (K.EQ.1.AND.WB.LE.2) GO TO 10
            IF (K.EQ.KMAX.AND.WT.LE.2) GO TO 10
            IF (J.EQ.1.AND.WF.LE.2) GO TO 10
            IF (J.EQ.JMAX.AND.WBK.LE.2) GO TO 10
            GO TO 20
         10 AR(IJK)=0.0
            AT(IJK)=0.0
```

```
      ABK(IJK)=0.0
      AC(IJK)=0.0
      BETA(IJK)=-1.0
   20 CONTINUE
C
      IF (NOBS.LE.0) GO TO 300
C
      J=2
      DO 250 N=1,NOBS
      DO 240 K=2,KM1
      DO 240 I=2,IM1
      CALL IJKONLY
      RDXDY=1.0/(DELX(I)*DELZ(K))
      DO 80 M=1,4
      GO TO (30,40,50,60), M
   30 X1=X(I)
      Y1=Z(K-1)
      DIS(1)=DELZ(K)
      GO TO 70
   40 Y1=Z(K)
      X1=X(I)
      DIS(2)=DELX(I)
      GO TO 70
   50 X1=X(I-1)
      Y1=Z(K)
      DIS(3)=DELZ(K)
      GO TO 70
   60 Y1=Z(K-1)
      X1=X(I-1)
      DIS(4)=DELX(I)
   70 IFLG(M)=0
      FCONIC=OA2(N)*X1*X1+OA1(N)*X1+OB2(N)*Y1*Y1+OB1(N)*Y1+OC2(N)*X1*Y1
     1 +OC1(N)
      IF (FCONIC.LE.0.0) IFLG(M)=1
      XM(M)=X1
      YM(M)=Y1
   80 CONTINUE
      IFLG(5)=IFLG(1)
      XM(5)=XM(1)
      YM(5)=YM(1)
      IFLGS=0
      DO 90 M=1,4
   90 IFLGS=IFLGS+IFLG(M)
      BRIJ=0.0
      BTIJ=0.0
      IF (IFLGS.EQ.0) GO TO 240
      IF (IFLGS.LT.4) GO TO 100
      BIJ=1.0
      BRIJ=1.0
      BTIJ=1.0
      GO TO 220
  100 IF (IFLG(1).EQ.1.AND.IFLG(2).EQ.1) BRIJ=1.0
      IF (IFLG(2).EQ.1.AND.IFLG(3).EQ.1) BTIJ=1.0
      DO 180 M=1,4
      IF (IFLG(M).EQ.IFLG(M+1)) GO TO 180
      X1=XM(M)
      Y1=YM(M)
      X2=XM(M+1)
      Y2=YM(M+1)
      IF (IFLG(M).EQ.0) GO TO 110
      X2=XM(M)
      Y2=YM(M)
      X1=XM(M+1)
      Y1=YM(M+1)
  110 EPSIF=0.001*(ABS(X2-X1)+ABS(Y2-Y1))
      SMN=0.0
      FMN=OA2(N)*X2*X2+OA1(N)*X2+OB2(N)*Y2*Y2+OB1(N)*Y2+OC2(N)*X2*Y2+OC1
     1 (N)
      SMX=1.0
      FMX=OA2(N)*X1*X1+OA1(N)*X1+OB2(N)*Y1*Y1+OB1(N)*Y1+OC2(N)*X1*Y1+OC1
     1 (N)
      S=0.5
  120 XT=S*X1+(1.0-S)*X2
      YT=S*Y1+(1.0-S)*Y2
      FS=OA2(N)*XT*XT+OA1(N)*XT+OB2(N)*YT*YT+OB1(N)*YT+OC2(N)*XT*YT+OC1
     1 (N)
      IF (ABS(FS).LT.EPSIF) GO TO 150
      IF (FS.GE.0.0) GO TO 130
      FDEN=ABS(FS-FMN)+1.0E-10
      SE=S-FS*(S-SMN)/FDEN
```

```
      IF (SE.GT.SMX) SE=SMX
      FMN=FS
      SMN=S
      GO TO 140
  130 FDEN=ABS(FMX-FS)+1.0E-10
      SE=S-FS*(SMX-S)/FDEN
      IF (SE.LT.SMN) SE=SMN
      FMX=FS
      SMX=S
  140 SI=S-FS*(SMX-SMN)/(FMX-FMN)
      S=0.5*(SE+SI)
      GO TO 120
  150 DIS(M)=SQRT((XT-X2)**2+(YT-Y2)**2)
      GO TO (160,170,180,180), M
  160 BRIJ=DIS(1)/DELZ(K)
      GO TO 180
  170 BTIJ=DIS(2)/DELX(I)
  180 CONTINUE
      M=0
      BIJ=0.0
  190 CONTINUE
      M=M+1
      IF (M.EQ.5) GO TO 210
      IF (IFLG(M).EQ.0) GO TO 190
      MP1=M+1
      IF (MP1.EQ.5) MP1=1
      MM1=M-1
      IF (MM1.EQ.0) MM1=4
      BIJ=BIJ+DIS(M)*DIS(MM1)
      IF (IFLG(MP1).EQ.1) GO TO 200
      DIS2=DIS(M)
  200 CONTINUE
      IF (IFLG(MM1).EQ.1) GO TO 190
      DIS1=DIS(MM1)
      GO TO 190
  210 CONTINUE
      IF (IFLGS.EQ.3) BIJ=BIJ-DIS1*DIS2
      BIJ=0.5*BIJ*RDXDY
      IF (BIJ.GT.1.0) BIJ=1.0
  220 CONTINUE
      IF (IOH(N).EQ.0) GO TO 230
      BIJ=-BIJ
      BRIJ=-BRIJ
      BTIJ=-BTIJ
  230 AC(IJK)=AC(IJK)+BIJ
      IF (AC(IJK).GT.0.99) AC(IJK)=1.0
      IF (AC(IJK).LT.0.01) AC(IJK)=0.0
      ABK(IJK)=AC(IJK)
      AR(IJK)=AR(IJK)+BRIJ
      IF (AR(IJK).GT.0.99) AR(IJK)=1.0
      IF (AR(IJK).LT.0.01) AR(IJK)=0.0
      IF (I.EQ.IM1.AND.WR.LE.2) AR(IJK)=0.0
      AT(IJK)=AT(IJK)+BTIJ
      IF (AT(IJK).GT.0.99) AT(IJK)=1.0
      IF (AT(IJK).LT.0.01) AT(IJK)=0.0
      IF (K.EQ.KM1.AND.WT.LE.2) AT(IJK)=0.0
  240 CONTINUE
  250 CONTINUE
C
      DO 270 K=1,KMAX
      IF (WL.LE.2) GO TO 260
      IJKL=NQ*(II5*(K-1)+IMAX+1)+1
      IJKLM=IJKL-NQ
      AR(IJKLM)=AR(IJKL)
      AT(IJKLM)=AT(IJKL)
      ABK(IJKLM)=ABK(IJKL)
      AC(IJKLM)=AC(IJKL)
  260 IF (WR.LE.2) GO TO 270
      IJKR=NQ*(II5*(K-1)+IMAX+(IM1-1))+1
      IJKRP=IJKR+NQ
      AR(IJKRP)=AR(IJKR)
      AT(IJKRP)=AT(IJKR)
      ABK(IJKRP)=ABK(IJKR)
      AC(IJKRP)=AC(IJKR)
  270 CONTINUE
C
      DO 290 I=1,IMAX
      IF (WB.LE.2) GO TO 280
      IJKB=NQ*(II5+IMAX+(I-1))+1
      IJKBM=IJKB-II2
```

```
      AT(IJKBM)=AT(IJKB)
      AR(IJKBM)=AR(IJKB)
      ABK(IJKBM)=ABK(IJKB)
      AC(IJKBM)=AC(IJKB)
  280 IF (WT.LE.2) GO TO 290
      IJKT=NQ*(II5*(KM1-1)+IMAX+(I-1))+1
      IJKTP=IJKT+II2
      AT(IJKTP)=AT(IJKT)
      AR(IJKTP)=AR(IJKT)
      ABK(IJKTP)=ABK(IJKT)
      AC(IJKTP)=AC(IJKT)
  290 CONTINUE
C
  300 CONTINUE
C
      J=2
      DO 310 K=2,KM1
      DO 310 I=2,IM1
      CALL IJKAJCT
      IF (AC(IJK).GT.EM6) GO TO 310
      AR(IJK)=0.0
      AR(IMJK)=0.0
      AT(IJK)=0.0
      AT(IJKM)=0.0
      ABK(IJK)=0.0
      ABK(IJMK)=0.0
      BETA(IJK)=-1.0
  310 CONTINUE
C
C     NOTE: ALL JPLANES HAVE SAME VALUES
C
      DO 360 J=1,JMAX
      IF (J.EQ.2) GO TO 360
      DO 350 K=1,KMAX
      DO 340 I=1,IMAX
      I2K=NQ*(II5*(K-1)+IMAX+(I-1))+1
      IJK=NQ*(II5*(K-1)+IMAX*(J-1)+(I-1))+1
      IF (J.EQ.1.AND.WF.LE.2) GO TO 330
      IF (J.EQ.JMAX.AND.WBK.LE.2) GO TO 320
      AR(IJK)=AR(I2K)
      AT(IJK)=AT(I2K)
      ABK(IJK)=ABK(I2K)
      AC(IJK)=AC(I2K)
      BETA(IJK)=BETA(I2K)
      GO TO 340
  320 IJMK=IJK-II1
      ABK(IJMK)=0.0
  330 AR(IJK)=0.0
      AT(IJK)=0.0
      ABK(IJK)=0.0
      AC(IJK)=0.0
      BETA(IJK)=-1.0
  340 CONTINUE
  350 CONTINUE
  360 CONTINUE
C ***
C *** SET SPECIAL VALUES OF AR,AT,ABK HERE
C ***
      RETURN
      END
*DK BC
      SUBROUTINE BC
*CA SLCOM1
C
C * * GENERAL-SURFACE BOUNDARY CONDITIONS
C
C * * SET LEFT AND RIGHT BDY CONDITIONS
C
      DO 150 K=1,KMAX
      IF (CYL.EQ.0.0) GO TO 20
      SUM1=0.0
      SUM2=0.0
      SUM3=0.0
      DO 10 J=2,JM1
      IJK=NQ*(II5*(K-1)+IMAX*(J-1)+1)+1
      SUM1=SUM1+U(IJK)*CTHJ(J)-V(IJK)*STHJBK(J)
      SUM2=SUM2+U(IJK)*STHJ(J)+V(IJK)*CTHJBK(J)
      SUM3=SUM3+W(IJK)
   10 CONTINUE
      IF (ABS(SUM1).LT.EM6) SUM1=0.0
```

```
      IF (ABS(SUM2).LT.EM6) SUM2=0.0
      IF (ABS(SUM3).LT.EM6) SUM3=0.0
      SUM1=0.5*SUM1/JBAR
      SUM2=0.5*SUM2/JBAR
      SUM3=SUM3/JBAR
      PHIO=1.570796327
      IF (SUM1.NE.0.0) PHIO=ATAN(SUM2/SUM1)
      CPHIO=COS(PHIO)
      SPHIO=SIN(PHIO)
      VELXY=SQRT(SUM1**2+SUM2**2)
   20 CONTINUE
      DO 140 J=1,JMAX
      ILJK=NQ*(II5*(K-1)+IMAX*(J-1)+1)+1
      ILMJK=ILJK-NQ
      ILPJK=ILJK+NQ
      IRJK=ILJK+IIO
      IRMJK=IRJK-NQ
      F(ILMJK)=F(ILJK)
      P(ILMJK)=P(ILJK)
      F(IRJK)=F(IRMJK)
      P(IRJK)=P(IRMJK)
      IF (CYL.EQ.1.0) GO TO 70
      GO TO (30,40,50,60,50), WL
   30 U(ILMJK)=0.0
      V(ILMJK)=V(ILJK)*(1.0-CYL+CYL*XI(2)*RXI(1))
      W(ILMJK)=W(ILJK)
      GO TO 80
   40 U(ILMJK)=0.0
      V(ILMJK)=-V(ILJK)*DELXRL
      W(ILMJK)=-W(ILJK)*DELXRL
      GO TO 80
   50 IF (ITER.GT.0) GO TO 80
      U(ILMJK)=U(ILJK)*RR(1)/RR(2)
      V(ILMJK)=V(ILJK)
      W(ILMJK)=W(ILJK)
      GO TO 80
   60 U(ILMJK)=U(IRMJK)
      V(ILMJK)=V(IRMJK)
      W(ILMJK)=W(IRMJK)
      F(ILMJK)=F(IRMJK)
      NF(ILMJK)=NF(IRMJK)
      P(ILMJK)=P(IRMJK)
      PN(ILMJK)=PN(IRMJK)
      PS(ILMJK)=PS(IRMJK)
      PETA(ILMJK)=PETA(IRMJK)
      GO TO 80
   70 IOPJK=NQ*(II5*(K-1)+IMAX*(JOP(J)-1)+1)+1
      F(ILMJK)=F(IOPJK)
      P(ILMJK)=P(IOPJK)
      U(ILMJK)=VELXY*(CTHJ(J)*CPHIO+STHJ(J)*SPHIO)
      V(ILMJK)=-VELXY*(STHJBK(J)*CPHIO-CTHJBK(J)*SPHIO)
      W(ILMJK)=SUM3
   80 GO TO (90,100,110,120,110), WR
   90 U(IRMJK)=0.0
      V(IRJK)=V(IRMJK)*(1.0-CYL+CYL*XI(IM1)*RXI(IMAX))
      W(IRJK)=W(IRMJK)
      GO TO 130
  100 U(IRMJK)=0.0
      V(IRJK)=-V(IRMJK)*DELXRR
      W(IRJK)=-W(IRMJK)*DELXRR
      GO TO 130
  110 IF (ITER.GT.0) GO TO 130
      U(IRJK)=U(IRMJK)*RR(IMAX)/RR(IM1)
      V(IRJK)=V(IRMJK)
      W(IRJK)=W(IRMJK)
      GO TO 130
  120 U(IRJK)=U(ILMJK)
      V(IRJK)=V(ILMJK)
      W(IRJK)=W(ILMJK)
      F(IRJK)=F(ILJK)
      NF(IRJK)=NF(ILJK)
      P(IRJK)=P(ILJK)
      PN(IRJK)=PN(ILJK)
      PS(IRJK)=PS(ILJK)
      PETA(IRJK)=PETA(ILJK)
  130 CONTINUE
  140 CONTINUE
  150 CONTINUE
C
C * * SET FRONT AND BACK BDY. CONDITIONS
```

```
C
      DO 270 K=1,KMAX
      DO 260 I=1,IMAX
      IFJK=NQ*(II5*(K-1)+IMAX+I-1)+1
      IFJMK=IFJK-II1
      IFJPK=IFJK+II1
      IBKJK=IFJK+II3
      IBKJMK=IBKJK-II1
      F(IFJMK)=F(IFJK)
      P(IFJMK)=P(IFJK)
      F(IBKJK)=F(IBKJMK)
      P(IBKJK)=P(IBKJMK)
      GO TO (160,170,180,190,180), WF
  160 U(IFJMK)=U(IFJK)
      V(IFJMK)=0.0
      W(IFJMK)=W(IFJK)
      GO TO 200
  170 U(IFJMK)=-U(IFJK)*DELYRF
      V(IFJMK)=0.0
      W(IFJMK)=-W(IFJK)*DELYRF
      GO TO 200
  180 IF (ITER.GT.0) GO TO 200
      U(IFJMK)=U(IFJK)
      V(IFJMK)=V(IFJK)
      W(IFJMK)=W(IFJK)
      GO TO 200
  190 U(IFJMK)=U(IBKJMK)
      V(IFJMK)=0.5*(V(IFJMK)+V(IBKJMK))
      IF (ISOR.EQ.0) V(IFJMK)=V(IBKJMK)
      W(IFJMK)=W(IBKJMK)
      F(IFJMK)=F(IBKJMK)
      NF(IFJMK)=NF(IBKJMK)
      P(IFJMK)=P(IBKJMK)
      PN(IFJMK)=PN(IBKJMK)
      PS(IFJMK)=PS(IBKJMK)
      PETA(IFJMK)=PETA(IBKJMK)
  200 GO TO (210,220,230,240,230), WBK
  210 U(IBKJK)=U(IBKJMK)
      V(IBKJMK)=0.0
      W(IBKJK)=W(IBKJMK)
      GO TO 250
  220 U(IBKJK)=-U(IBKJMK)*DELYRBK
      V(IBKJMK)=0.0
      W(IBKJK)=-W(IBKJMK)*DELYRBK
      GO TO 250
  230 IF (ITER.GT.0) GO TO 250
      U(IBKJK)=U(IBKJMK)
      V(IBKJK)=V(IBKJMK)
      W(IBKJK)=W(IBKJMK)
      GO TO 250
  240 U(IBKJK)=U(IFJK)
      W(IBKJK)=W(IFJK)
      V(IBKJK)=V(IFJK)
      V(IBKJMK)=V(IFJMK)
      F(IBKJK)=F(IFJK)
      NF(IBKJK)=NF(IFJK)
      P(IBKJK)=P(IFJK)
      PN(IBKJK)=PN(IFJK)
      PS(IBKJK)=PS(IFJK)
      PETA(IBKJK)=PETA(IFJK)
  250 CONTINUE
  260 CONTINUE
  270 CONTINUE
C
C * * SET BOTTOM AND TOP BDY. CONDITIONS
C
      DO 390 J=2,JM1
      DO 380 I=2,IM1
      IBJK=NQ*(II5+IMAX*(J-1)+I-1)+1
      IBJKM=IBJK-II2
      ITJK=IBJK+II4
      ITJKM=ITJK-II2
      F(IBJKM)=F(IBJK)
      P(IBJKM)=P(IBJK)
      F(ITJK)=F(ITJKM)
      P(ITJK)=P(ITJKM)
      GO TO (280,290,300,310,300), WB
  280 U(IBJKM)=U(IBJK)
      V(IBJKM)=V(IBJK)
      W(IBJKM)=0.0
```

```
      GO TO 320
  290 U(IBJKM)=-U(IBJK)*DELZRB
      V(IBJKM)=-V(IBJK)*DELZRB
      W(IBJKM)=0.0
      GO TO 320
  300 IF (ITER.GT.0) GO TO 320
      U(IBJKM)=U(IBJK)
      V(IBJKM)=V(IBJK)
      W(IBJKM)=W(IBJK)
      GO TO 320
  310 U(IBJKM)=U(ITJKM)
      V(IBJKM)=V(ITJKM)
      W(IBJKM)=0.5*(W(IBJKM)+W(ITJKM))
      F(IBJKM)=F(ITJKM)
      NF(IBJKM)=NF(ITJKM)
      P(IBJKM)=P(ITJKM)
      PN(IBJKM)=PN(ITJKM)
      PS(IBJKM)=PS(ITJKM)
      PETA(IBJKM)=PETA(ITJKM)
  320 GO TO (330,340,350,360,350), WT
  330 U(ITJK)=U(ITJKM)
      V(ITJK)=V(ITJKM)
      W(ITJKM)=0.0
      GO TO 370
  340 U(ITJK)=-U(ITJKM)*DELZRT
      V(ITJK)=-V(ITJKM)*DELZRT
      W(ITJKM)=0.0
      GO TO 370
  350 IF (ITER.GT.0) GO TO 370
      U(ITJK)=U(ITJKM)
      V(ITJK)=V(ITJKM)
      W(ITJK)=W(ITJKM)
      GO TO 370
  360 U(ITJK)=U(IBJK)
      V(ITJK)=V(IBJK)
      W(ITJK)=W(IBJKM)
      F(ITJK)=F(IBJK)
      NF(ITJK)=NF(IBJK)
      P(ITJK)=P(IBJK)
      PN(ITJK)=PN(IBJK)
      PS(ITJK)=PS(IBJK)
      PETA(ITJK)=PS(IBJK)
  370 CONTINUE
  380 CONTINUE
  390 CONTINUE
C
C * * SET FREE SURFACE BOUNDARY CONDITIONS
C
      CALL BCFS
C
C * * SPECIAL BOUNDARY CONDITIONS
C * * (FOR SPECIFIED IN OR OUT FLOW SET BETA=+1.0 IN FICTICIOUS CELLS)
C
      RETURN
      END
*DK BCFS
      SUBROUTINE BCFS
*CA SLCOM1
C
C * * SET FREE SURFACE BOUNDARY CONDITIONS
C
      DO 220 K=2,KM1
      DO 220 J=2,JM1
      DO 220 I=2,IM1
      KNFF=0
      CALL CALCIJK
C
C * * SET F AND P VALUES IN OBSTACLE EDGE CELLS
C
      IF (BETA(IJK).GE.0.0) GO TO 10
      ABR=1.0
      IF (BETA(IPJK).LT.0.0) ABR=0.0
      ABL=1.0
      IF (BETA(IMJK).LT.0.0) ABL=0.0
      ABBK=1.0
      IF (BETA(IJPK).LT.0.0) ABBK=0.0
      ABF=1.0
      IF (BETA(IJMK).LT.0.0) ABF=0.0
      ABT=1.0
      IF (BETA(IJKP).LT.0.0) ABT=0.0
```

```
      ABB=1.0
      IF (BETA(IJKM).LT.0.0) ABB=0.0
      ABSUM=ABR+ABL+ABBK+ABF+ABT+ABB
      IF (ABSUM.LT.0.5) GO TO 10
      F(IJK)=(ABR*F(IPJK)+ABL*F(IMJK)+ABBK*F(IJPK)+ABF*F(IJMK)+ABT*F
     1 (IJKP)+ABB*F(IJKM))/ABSUM
      P(IJK)=(ABR*P(IPJK)+ABL*P(IMJK)+ABBK*P(IJPK)+ABF*P(IJMK)+ABT*P
     1 (IJKP)+ABB*P(IJKM))/ABSUM
      GO TO 220
   10 CONTINUE
      IF (NMAT.EQ.2) GO TO 230
      NFF=NF(IJK)
      IF (NFF.EQ.0.OR.NFF.GT.7) GO TO 220
C
C * * SET VELOCITY ON CELL BOUNDARY BETWEEN PARTIALLY
      GO TO (20,30,40,50,60,70,80), NFF
   20 IF (AR(IJK).GT.EM6) U(IJK)=U(IMJK)*AR(IMJK)*X(I-1)/(AR(IJK)*X(I))
      IF (ABK(IJK).GT.EM6) V(IJK)=V(IMJK)
      IF (ABK(IJMK).GT.EM6) V(IJMK)=V(IMJMK)
      IF (AT(IJK).GT.EM6) W(IJK)=W(IMJK)
      IF (AT(IJKM).GT.EM6) W(IJKM)=W(IMJKM)
      GO TO 80
   30 IF (AR(IMJK).GT.EM6) U(IMJK)=U(IJK)*AR(IJK)*X(I)/(AR(IMJK)*X(I-1))
      IF (ABK(IJK).GT.EM6) V(IJK)=V(IPJK)
      IF (ABK(IJMK).GT.EM6) V(IJMK)=V(IPJMK)
      IF (AT(IJK).GT.EM6) W(IJK)=W(IPJK)
      IF (AT(IJKM).GT.EM6) W(IJKM)=W(IPJKM)
      GO TO 80
   40 IF (ABK(IJK).GT.EM6) V(IJK)=V(IJMK)
      IF (AR(IJK).GT.EM6) U(IJK)=U(IJMK)
      IF (AR(IMJK).GT.EM6) U(IMJK)=U(IMJMK)
      IF (AT(IJK).GT.EM6) W(IJK)=W(IJMK)
      IF (AT(IJKM).GT.EM6) W(IJKM)=W(IJMKM)
      GO TO 80
   50 IF (ABK(IJK).GT.EM6) V(IJMK)=V(IJK)
      IF (AR(IJK).GT.EM6) U(IJK)=U(IJPK)
      IF (AR(IMJK).GT.EM6) U(IMJK)=U(IMJPK)
      IF (AT(IJK).GT.EM6) W(IJK)=W(IJPK)
      IF (AT(IJKM).GT.EM6) W(IJKM)=W(IJPKM)
      GO TO 80
   60 IF (AT(IJK).GT.EM6) W(IJK)=W(IJKM)
      IF (AR(IJK).GT.EM6) U(IJK)=U(IJKM)
      IF (AR(IMJK).GT.EM6) U(IMJK)=U(IMJKM)
      IF (ABK(IJK).GT.EM6) V(IJK)=V(IJKM)
      IF (ABK(IJKM).GT.EM6) V(IJMK)=V(IJMKM)
      GO TO 80
   70 IF (AT(IJKM).GT.EM6) W(IJKM)=W(IJK)
      IF (AR(IJK).GT.EM6) U(IJK)=U(IJKP)
      IF (AR(IMJK).GT.EM6) U(IMJK)=U(IMJKP)
      IF (ABK(IJK).GT.EM6) V(IJK)=V(IJKP)
      IF (ABK(IJMK).GT.EM6) V(IJMK)=V(IJMKP)
      GO TO 80
C
C * * CALCULATE VELOCITY ON NF SELECTED CELL BOUNDARY
C
   80 GO TO (90,100,110,120,130,140,140), NFF
   90 IF (NF(IPJK).LE.7.OR.AR(IJK).LT.EM6) GO TO 150
      DENOM=-RRI(I)*RDX(I)*AR(IJK)/RR(I)
      U(IJK)=(RRI(I)*(RDX(I)*(-U(IMJK)*AR(IMJK)/RR(I-1))+RDY(J)*(V(IJK)
     1 *ABK(IJK)-V(IJMK)*ABK(IJMK)))+RDZ(K)*(W(IJK)*AT(IJK)-W(IJKM)*AT
     2 (IJKM)))/DENOM
      GO TO 160
  100 IF (NF(IMJK).LE.7.OR.AR(IMJK).LT.EM6) GO TO 150
      DENOM=RRI(I)*RDX(I)*AR(IMJK)/RR(I-1)
      U(IMJK)=(RRI(I)*(RDX(I)*(U(IJK)*AR(IJK)/RR(I))+RDY(J)*(V(IJK)*ABK
     1 (IJK)-V(IJMK)*ABK(IJMK)))+RDZ(K)*(W(IJK)*AT(IJK)-W(IJKM)*AT(IJKM)
     2 ))/DENOM
      GO TO 160
  110 IF (NF(IJPK).LE.7.OR.ABK(IJK).LT.EM6) GO TO 150
      DENOM=-RRI(I)*RDY(J)*ABK(IJK)
      V(IJK)=(RRI(I)*(RDX(I)*(U(IJK)*AR(IJK)/RR(I)-U(IMJK)*AR(IMJK)/RR(I
     1 -1))+RDY(J)*(-V(IJMK)*ABK(IJMK)))+RDZ(K)*(W(IJK)*AT(IJK)-W(IJKM)
     2 *AT(IJKM)))/DENOM
      GO TO 160
  120 IF (NF(IJMK).LE.7.OR.ABK(IJMK).LT.EM6) GO TO 150
      DENOM=RRI(I)*RDY(J)*ABK(IJMK)
      V(IJMK)=(RRI(I)*(RDX(I)*(U(IJK)*AR(IJK)/RR(I)-U(IMJK)*AR(IMJK)/RR
     1 (I-1))+RDY(J)*(V(IJK)*ABK(IJK)))+RDZ(K)*(W(IJK)*AT(IJK)-W(IJKM)
     2 *AT(IJKM)))/DENOM
      GO TO 160
```

```
  130 IF (NF(IJKP).LE.7.OR.AT(IJK).LT.EM6) GO TO 150
      DENOM=-RDZ(K)*AT(IJK)
      W(IJK)=(RRI(I)*(RDX(I)*(U(IJK)*AR(IJK)/RR(I)-U(IMJK)*AR(IMJK)/RR(I
     1 -1))+RDY(J)*(V(IJK)*ABK(IJK)-V(IJMK)*ABK(IJMK)))+RDZ(K)*(-W(IJKM)
     2 *AT(IJKM)))/DENOM
      GO TO 160
  140 IF (NF(IJKM).LE.7.OR.AT(IJKM).LT.EM6) GO TO 150
      DENOM=RDZ(K)*AT(IJKM)
      W(IJKM)=(RRI(I)*(RDX(I)*(U(IJK)*AR(IJK)/RR(I)-U(IMJK)*AR(IMJK)/RR
     1 (I-1))+RDY(J)*(V(IJK)*ABK(IJK)-V(IJMK)*ABK(IJMK)))+RDZ(K)*(W(IJK)
     2 *AT(IJK)))/DENOM
      GO TO 160
  150 CONTINUE
C
C * * IF NF DESIGNATED CELL IS NOT EMPTY TRY ALL NEIGHBORING
C     CELLS TO FIND EMPTY NEIGHBOR
C
      KNFF=KNFF+1
      NFF=NFF+1
      IF (NFF.GT.7) NFF=1
      IF (KNFF.GT.7) GO TO 160
      GO TO 80
  160 CONTINUE
C
C * * SET VELOCITIES IN EMPTY CELLS ADJACENT TO PARTIAL FLUID CELLS
C
      IF (FLG.GT.0.5.AND.ITER.GT.0.AND.ISOR.EQ.1) GO TO 220
      IF (NF(IPJK).LE.7.OR.BETA(IPJK).LT.0.0) GO TO 170
      IF (NF(IPJPK).GT.7.AND.ABK(IPJK).GT.EM6) V(IPJK)=F(IJK)*V(IJK)*RRI
     1 (I)/RRI(I+1)
      IF (NF(IPJMK).GT.7.AND.ABK(IPJMK).GT.EM6) V(IPJMK)=F(IJK)*V(IJMK)
     1 *RRI(I)/RRI(I+1)
      IF (NF(IPJKP).GT.7.AND.AT(IPJK).GT.EM6) W(IPJK)=F(IJK)*W(IJK)
      IF (NF(IPJKM).GT.7.AND.AT(IPJKM).GT.EM6) W(IPJKM)=F(IJK)*W(IJKM)
  170 IF (NF(IMJK).LE.7.OR.BETA(IMJK).LT.0.0) GO TO 180
      IF (NF(IMJPK).GT.7.AND.ABK(IMJK).GT.EM6) V(IMJK)=F(IJK)*V(IJK)*RRI
     1 (I)/RRI(I-1)
      IF (NF(IMJMK).GT.7.AND.ABK(IMJMK).GT.EM6) V(IMJMK)=F(IJK)*V(IJMK)
     1 *RRI(I)/RRI(I-1)
      IF (NF(IMJKP).GT.7.AND.AT(IMJK).GT.EM6) W(IMJK)=F(IJK)*W(IJK)
      IF (NF(IMJKM).GT.7.AND.AT(IMJKM).GT.EM6) W(IMJKM)=F(IJK)*W(IJKM)
  180 IF (NF(IJPK).LE.7.OR.BETA(IJPK).LT.0.0) GO TO 190
      IF (NF(IMJPK).GT.7.AND.AR(IMJPK).GT.EM6) U(IMJPK)=F(IJK)*U(IMJK)
      IF (NF(IPJPK).GT.7.AND.AR(IJPK).GT.EM6) U(IJPK)=F(IJK)*U(IJK)
      IF (NF(IJPKP).GT.7.AND.AT(IJPK).GT.EM6) W(IJPK)=F(IJK)*W(IJK)
      IF (NF(IJPKM).GT.7.AND.AT(IJPKM).GT.EM6) W(IJPKM)=F(IJK)*W(IJKM)
  190 IF (NF(IJMK).LE.7.OR.BETA(IJMK).LT.0.0) GO TO 200
      IF (NF(IMJMK).GT.7.AND.AR(IMJMK).GT.EM6) U(IMJMK)=F(IJK)*U(IMJK)
      IF (NF(IPJMK).GT.7.AND.AR(IJMK).GT.EM6) U(IJMK)=F(IJK)*U(IJK)
      IF (NF(IJMKP).GT.7.AND.AT(IJMK).GT.EM6) W(IJMK)=F(IJK)*W(IJK)
      IF (NF(IJMKM).GT.7.AND.AT(IJMKM).GT.EM6) W(IJMKM)=F(IJK)*W(IJKM)
  200 IF (NF(IJKP).LE.7.OR.BETA(IJKP).LT.0.0) GO TO 210
      IF (NF(IPJKP).GT.7.AND.AR(IJKP).GT.EM6) U(IJKP)=F(IJK)*U(IJK)
      IF (NF(IMJKP).GT.7.AND.AR(IMJKP).GT.EM6) U(IMJKP)=F(IJK)*U(IMJK)
      IF (NF(IJPKP).GT.7.AND.ABK(IJKP).GT.EM6) V(IJKP)=F(IJK)*V(IJK)
      IF (NF(IJMKP).GT.7.AND.ABK(IJMKP).GT.EM6) V(IJMKP)=F(IJK)*V(IJMK)
  210 IF (NF(IJKM).LE.7.OR.BETA(IJKM).LT.0.0) GO TO 220
      IF (NF(IPJKM).GT.7.AND.AR(IJKM).GT.EM6) U(IJKM)=F(IJK)*U(IJK)
      IF (NF(IMJKM).GT.7.AND.AR(IMJKM).GT.EM6) U(IMJKM)=F(IJK)*U(IMJK)
      IF (NF(IJPKM).GT.7.AND.ABK(IJKM).GT.EM6) V(IJKM)=F(IJK)*V(IJK)
      IF (NF(IJMKM).GT.7.AND.ABK(IJMKM).GT.EM6) V(IJMKM)=F(IJK)*V(IJMK)
  220 CONTINUE
  230 CONTINUE
      RETURN
      END
*DK BETACAL
 .    SUBROUTINE BETACAL
*CA SLCOM1
C
C * * CALCULATE BETA(IJK) FOR MESH INTERIOR
C * * NOTE THAT ALL FICTICIOUS CELLS HAVE BETA SET TO ZERO IN ASET
C * * NOTE THAT ALL OBSTACLE CELLS HAVE BETA SET TO -1.0 IN ASET
C
C * * CALCULATE BETA(IJK) FOR NON-OBSTACLE CELLS
C
      RIJK=0.0
      DO 10 K=2,KM1
      DO 10 J=2,JM1
      DO 10 I=2,IM1
      CALL IJKAJCT
```

```
      IF (BETA(IJK).LT.0.0) GO TO 10
      RIJK=RIJK+1.
      ABR=AR(IJK)
      IF (BETA(IPJK).LT.0.0.OR.BETA(IPJK).EQ.1.0) ABR=0.0
      ABL=AR(IMJK)
      IF (BETA(IMJK).LT.0.0.OR.BETA(IMJK).EQ.1.0) ABL=0.0
      ABBK=ABK(IJK)
      IF (BETA(IJPK).LT.0.0.OR.BETA(IJPK).EQ.1.0) ABBK=0.0
      ABF=ABK(IJMK)
      IF (BETA(IJMK).LT.0.0.OR.BETA(IJMK).EQ.1.0) ABF=0.0
      ABT=AT(IJK)
      IF (BETA(IJKP).LT.0.0.OR.BETA(IJKP).EQ.1.0) ABT=0.0
      ABB=AT(IJKM)
      IF (BETA(IJKM).LT.0.0.OR.BETA(IJKM).EQ.1.0) ABB=0.0
      XX=2.0*DELT*(RDX(I)*(ABR/(DELX(I)+DELX(I+1))+ABL/(DELX(I)+DELX(I-1
     1 )))+RDY(J)*RRI(I)*(ABBK*RRI(I)/(DELY(J)+DELY(J+1))+ABF*RRI(I)/
     2 (DELY(J)+DELY(J-1)))+RDZ(K)*(ABT/(DELZ(K)+DELZ(K+1))+ABB/(DELZ(K)
     3 +DELZ(K-1)))+CYL*0.5*(ABR/(DELX(I)+DELX(I+1))-ABL/(DELX(I-1)+DELX
     4 (I)))*RRI(I)/X(IM1))
      BETA(IJK)=OMG/XX*AC(IJK)
   10 CONTINUE
C
      RIJK=1.0/RIJK
      RETURN
      END
*DK CALCIJK
      SUBROUTINE CALCIJK
*CA SLCOM1
C
C * * CALCULATE "IJK" AND OTHER INDICES FOR CELL (I,J,K)
C
      IJK=NQ*(II5*(K-1)+IMAX*(J-1)+(I-1))+1
      IMJK=IJK-NQ
      IPJK=IJK+NQ
      IJMK=IJK-II1
      IJPK=IJK+II1
      IJKM=IJK-II2
      IJKP=IJK+II2
      IPJMK=IJK+NQ-II1
      IPJKM=IJK+NQ-II2
      IMJPK=IJK-NQ+II1
      IMJMK=IJK-NQ-II1
      IJPKM=IJK+II1-II2
      IMJKP=IJK-NQ+II2
      IJMKP=IJK-II1+II2
      IJMKM=IJK-II1-II2
      IPJKP=IJK+NQ+II2
      IMJKM=IJK-NQ-II2
      IPJPK=IJK+NQ+II1
      IJPKP=IJK+II1+II2
      IMJPKP=IJK-NQ+II1+II2
      IPJMKP=IJK+NQ-II1+II2
      IPJPKM=IJK+NQ+II1-II2
      IPJPKP=IJK+NQ+II1+II2
C
      RETURN
C
      ENTRY IJKONLY
C
      IJK=NQ*(II5*(K-1)+IMAX*(J-1)+(I-1))+1
      RETURN
C
      ENTRY IJKAJCT
C
      IJK=NQ*(II5*(K-1)+IMAX*(J-1)+(I-1))+1
      IMJK=IJK-NQ
      IPJK=IJK+NQ
      IJMK=IJK-II1
      IJPK=IJK+II1
      IJKM=IJK-II2
      IJKP=IJK+II2
      RETURN
C
      END
*DK CNTR
      SUBROUTINE CNTR (I1,I2,J1,J2,K1,K2,NA,IPER)
*CA SLCOM1
C
C * * CONTOUR PLOTTING ROUTINE (SET-UP IN SUBROUTINE RCONTUR)
C
```

```
      COMMON /SSCM10/ MPLNC, IFMC, JFMC, KFMC, JNMC, DATC, CLKC, TC,
     1 ITERC, CYCLEC, NAMEC(10), I1C, I2C, J1C, J2C, K1C, K2C, IPERK
C
      INTEGER CYCLEC
      DO 10 I=1,50
      DO 10 J=1,50
   10 Q(I,J)=0.0
C
C * * DETERMINE CONSTANT PLANE, MPLN
C
      IFM=0
      JFM=0
      KFM=0
      IF (I1.NE.I2) GO TO 20
      IFM=I1
      MPLN=1
   20 IF (J1.NE.J2) GO TO 30
      JFM=J1
      MPLN=2
   30 IF (K1.NE.K2) GO TO 40
      KFM=K1
      MPLN=3
   40 CONTINUE
C
C * * SET CELL(2,2,2) REFERENCE INDEX
C
      I=2
      J=2
      K=2
      CALL CALCIJK
      IJKC=IJK
C
C * * CALCULATE PRESSURE VALUES, Q-ARRAY, TO BE PLOTTED
C
      DO 80 K=2,KM1
      DO 80 J=2,JM1
      DO 80 I=2,IM1
      CALL CALCIJK
      IQ=I-1
      JQ=J-1
      KQ=K-1
      GO TO (50,60,70), MPLN
   50 IF (I.NE.I1) GO TO 80
      IF (J.LT.J1) GO TO 80
      IF (J.GT.J2) GO TO 80
      IF (K.LT.K1) GO TO 80
      IF (K.GT.K2) GO TO 80
      Q(JQ,KQ)=P(IJK)-P(IJKC)
      GO TO 80
   60 IF (J.NE.J1) GO TO 80
      IF (K.LT.K1) GO TO 80
      IF (K.GT.K2) GO TO 80
      IF (I.LT.I1) GO TO 80
      IF (I.GT.I2) GO TO 80
      Q(IQ,KQ)=P(IJK)-P(IJKC)
      GO TO 80
   70 IF (K.NE.K1) GO TO 80
      IF (J.LT.J1) GO TO 80
      IF (J.GT.J2) GO TO 80
      IF (I.LT.I1) GO TO 80
      IF (I.GT.I2) GO TO 80
      Q(IQ,JQ)=P(IJK)-P(IJKC)
   80 CONTINUE
      MPLNC=MPLN
      IFMC=IFM
      JFMC=JFM
      KFMC=KFM
      JNMC=JNM
      DATC=DAT
      CLKC=CLK
      TC=T
      ITERC=ITER
      CYCLEC=CYCLE
      DO 90 N=1,10
   90 NAMEC(N)=NAME(N)
      I1C=I1
      I2C=I2
      J1C=J1
      J2C=J2
      K1C=K1
```

```
      K2C=K2                                                            CNTR      87
C                                                                       CNTR      88
C * * CALL CONTOUR PLOTTING ROUTINES                                    CNTR      89
C                                                                       CNTR      90
      NYM=NNY+1                                                         CNTR      91
      GO TO (100,110,120), MPLN                                         CNTR      92
  100 CALL CONTRJB (YJ(2),NYM,ZK(2),NNZ,Q,NZX,NZY,NC,ZMN,ZMX,DLZ,ZCQ    CNTR      93
     1 ,DMPY,DMPZ,IGRD,ITITLE1,NTITLE1,YLABLE,NYLBL,ZLABLE,NZLBL)       CNTR      94
      GO TO 130                                                         CNTR      95
  110 CALL CONTRJB (XI(2),NNX,ZK(2),NNZ,Q,NZX,NZY,NC,ZMN,ZMX,DLZ,ZCQ    CNTR      96
     1 ,DMPX,DMPZ,IGRD,ITITLE1,NTITLE1,XLABLE,NXLBL,ZLABLE,NZLBL)       CNTR      97
      GO TO 130                                                         CNTR      98
  120 CALL CONTRJB (XI(2),NNX,YJ(2),NYM,Q,NZX,NZY,NC,ZMN,ZMX,DLZ,ZCQ    CNTR      99
     1 ,DMPX,DMPY,IGRD,ITITLE1,NTITLE1,XLABLE,NXLBL,YLABLE,NYLBL)       CNTR     100
  130 CONTINUE                                                          CNTR     101
C                                                                       CNTR     102
      RETURN                                                            CNTR     103
      END                                                               CNTR     104
*DK CONTRJB                                                             CONTRJB    1
      SUBROUTINE CONTRJB (X,NNX,Y,NNY,Z,NZX,NZY,NC,ZMN,ZMX,DLZ,ZPLAN    CONTRJB    2
     1 ,DMPX,DMPY,IGRD,ITITLE,NTITLE,XLABEL,NXLBL,YLABEL,NYLBL)         CONTRJB    3
      COMMON /CNTRCOM/ ISYM(50)                                         CONTRJB    4
      COMMON /SSCM10/ MPLNC, IFMC, JFMC, KFMC, JNMC, DATC, CLKC, TC,    CONTRJB    5
     1 ITERC, CYCLEC, NAMEC(10), I1C, I2C, J1C, J2C, K1C, K2C, IPERK    CONTRJB    6
      DIMENSION XSCALE(2), YSCALE(2)                                    CONTRJB    7
      EQUIVALENCE (XMIN,XSCALE(1)), (XMAX,XSCALE(2))                    CONTRJB    8
      EQUIVALENCE (YMIN,YSCALE(1)), (YMAX,YSCALE(2))                    CONTRJB    9
      DIMENSION X(1), Y(1), Z(NZX,1), ZPLAN(1)                          CONTRJB   10
      DIMENSION ZT(4)                                                   CONTRJB   11
      INTEGER CYCLE, CYCLEC                                             CONTRJB   12
      NOC=MIN0(IABS(NC),50)                                             CONTRJB   13
      ZMIN=ZMN                                                          CONTRJB   14
      ZMAX=ZMX                                                          CONTRJB   15
      DLZZ=DLZ                                                          CONTRJB   16
      DMAPX=DMPX                                                        CONTRJB   17
      DMAPY=DMPY                                                        CONTRJB   18
      NOX=IABS(NNX)                                                     CONTRJB   19
      NOY=IABS(NNY)                                                     CONTRJB   20
      DO 10 I=1,50                                                      CONTRJB   21
   10 ISYM(I)=0                                                         CONTRJB   22
C                                                                       CONTRJB   23
C * * ESTABLISH SCALES                                                  CONTRJB   24
C                                                                       CONTRJB   25
      CALL MINV (X,1,NOX,I,XMIN)                                        CONTRJB   26
      CALL MAXV (Y,1,NOY,I,YMAX)                                        CONTRJB   27
      CALL MAXV (X,1,NOX,I,XMAX)                                        CONTRJB   28
      CALL MINV (Y,1,NOY,I,YMIN)                                        CONTRJB   29
      FGRD=0.                                                           CONTRJB   30
      IF (IGRD.GT.0) FGRD=-IGRD                                         CONTRJB   31
      CALL PLOJB (XSCALE,YSCALE,2,1,1,1,FGRD,DMAPX,DMAPY,ITITLE,-NTITLE CONTRJB   32
     1 ,XLABEL,NXLBL,YLABEL,NYLBL)                                      CONTRJB   33
      IF (NC.LT.0) GO TO 50                                             CONTRJB   34
      IF (NNX.LE.0) CALL MINM (Z,NZX,NOX,NOY,I,J,ZMIN)                  CONTRJB   35
      IF (NNY.LE.0) CALL MAXM (Z,NZX,NOX,NOY,I,J,ZMAX)                  CONTRJB   36
      IF (DLZZ.GT.0) GO TO 20                                           CONTRJB   37
      DLZZ=(ZMAX-ZMIN)/(NOC-1.)                                         CONTRJB   38
   20 IF (NZY.GT.0) GO TO 30                                            CONTRJB   39
      ZMAX=ZMAX-AMOD(ZMAX,DLZZ)                                         CONTRJB   40
      ZMIN=ZMIN-AMOD(ZMIN,DLZZ)                                         CONTRJB   41
      NOC=MIN0(NOC,IFIX((ZMAX-ZMIN)/DLZZ+1.01))                         CONTRJB   42
   30 ZPLAN(1)=ZMIN                                                     CONTRJB   43
      DO 40 I=2,NOC                                                     CONTRJB   44
   40 ZPLAN(I)=ZPLAN(I-1)+DLZZ                                          CONTRJB   45
   50 DO 90 NY=2,NOY                                                    CONTRJB   46
      IX=MOD(NY,2)                                                      CONTRJB   47
      DY=Y(NY)-Y(NY-1)                                                  CONTRJB   48
      DO 80 INX=2,NOX                                                   CONTRJB   49
      NX=INX                                                            CONTRJB   50
      IF (IX.NE.0) NX=NOX-INX+2                                         CONTRJB   51
      ZT(1)=Z(NX-1,NY-1)                                                CONTRJB   52
      ZT(2)=Z(NX,NY-1)                                                  CONTRJB   53
      ZT(3)=Z(NX,NY)                                                    CONTRJB   54
      ZT(4)=Z(NX-1,NY)                                                  CONTRJB   55
      DX=X(NX)-X(NX-1)                                                  CONTRJB   56
      IF (ABS(ZT(3)-ZT(1))-ABS(ZT(4)-ZT(2))) 70,60,60                   CONTRJB   57
   60 CALL TRICJB (X(NX),Y(NY),-DX,-DY,NOC,ZPLAN,ZT(4),ZT(3),ZT(2))     CONTRJB   58
      CALL TRICJB (X(NX-1),Y(NY-1),DX,DY,NOC,ZPLAN,ZT(2),ZT(1),ZT(4))   CONTRJB   59
      GO TO 80                                                          CONTRJB   60
   70 CALL TRICJB (X(NX-1),Y(NY),DX,-DY,NOC,ZPLAN,ZT(3),ZT(4),ZT(1))    CONTRJB   61
      CALL TRICJB (X(NX),Y(NY-1),-DX,DY,NOC,ZPLAN,ZT(1),ZT(2),ZT(3))    CONTRJB   62
```

```
   80 CONTINUE
   90 CONTINUE
      CALL LINCNT (1)
      WRITE (12,230) (NAMEC(I),I=1,5),JNMC,DATC,CLKC
      CALL LINCNT (59)
      WRITE (12,240) ITERC,TC,CYCLEC
      GO TO (100,110,120), MPLNC
  100 WRITE (12,250) IFMC,J1C,J2C,K1C,K2C
      GO TO 130
  110 WRITE (12,260) JFMC,I1C,I2C,K1C,K2C
      GO TO 130
  120 WRITE (12,270) KFMC,I1C,I2C,J1C,J2C
  130 CONTINUE
      N=1
      DO 140 I=1,NOC
      ZTT=ABS(ZPLAN(I))
      IF (ZTT.EQ.0.OR.(ZTT.GT..01.AND.(ZTT.LT.10000))) GO TO 140
      N=2
      GO TO 150
  140 CONTINUE
  150 I1=1
      I2=MINO(NOC,30)
  160 CALL FADV (1)
      CALL LINCNT (56)
      WRITE (12,230) (NAMEC(I),I=1,5),JNMC,DATC,CLKC
      WRITE (12,240) ITERC,TC,CYCLEC
      GO TO (170,180,190), MPLNC
  170 WRITE (12,250) IFMC,J1C,J2C,K1C,K2C
      GO TO 200
  180 WRITE (12,260) JFMC,I1C,I2C,K1C,K2C
      GO TO 200
  190 WRITE (12,270) KFMC,I1C,I2C,J1C,J2C
  200 CONTINUE
      KYS=20
      CALL EXL
      CALL DLCH (MAXO(0,460-NTITLE*6),KYS,NTITLE,ITITLE,1)
      KYS=KYS+60
      CALL DLCH (250,KYS,35,35HIDENTIFICATION         CONTOUR VALUE,1)
      DO 220 I=I1,I2
      KYS=KYS+30
      IF (N.EQ.1) ENCODE (8,280,BCDA) ZPLAN(I)
      IF (N.EQ.2) ENCODE (8,290,BCDA) ZPLAN(I)
      CALL EXH
      CALL DRVEC (250,KYS,400,KYS)
      CALL EXL
      DO 210 K=1,4
  210 CALL DLCH (K*50+200,KYS,0,I,1)
  220 CALL DLCH (550,KYS-6,10,BCDA,1)
      CALL EXH
      IF (I2.EQ.NOC) RETURN
      I1=I2+1
      I2=NOC
      GO TO 160
C
  230 FORMAT (1H ,2X,5A8,1X,A8,2(1X,A8))
  240 FORMAT (1X,5HITER=,I4,18X,6HTIME= ,1PE12.5,6X,7HCYCLE= ,I4)
  250 FORMAT (13H  CONSTANT I=,I4,17H   SURFACE     J=,I3,3H TO,I3,7H
     1  K=,I3,4H  TO,I3)
  260 FORMAT (13H  CONSTANT J=,I4,17H   SURFACE     I=,I3,3H TO,I3,7H
     1  K=,I3,4H  TO,I3)
  270 FORMAT (13H  CONSTANT K=,I4,17H   SURFACE     I=,I3,3H TO,I3,7H
     1  J=,I3,4H  TO,I3)
  280 FORMAT (F8.3)
  290 FORMAT (1PE8.1)
      END
*DK DELTADJ
      SUBROUTINE DELTADJ
*CA SLCOM1
C
      DATA CON, ITMIN, ITMOST, LITER, ITCRMX /0.40,15,30,50,200/
C * * ADJUST TIME STEP (DELT)
C
      DELTN=DELT
      IF (FLGC.LT.0.5) GO TO 20
C
C * * F CONVECTION LIMIT EXCEEDED
C
      T=T-DELT
      CYCLE=CYCLE-1
      DELT=0.5*DELT
```

```
      DO 10 K=1,KMAX
      DO 10 J=1,JMAX
      DO 10 I=1,IMAX
      CALL CALCIJK
      F(IJK)=FN(IJK)
      U(IJK)=UN(IJK)
      V(IJK)=VN(IJK)
      W(IJK)=WN(IJK)
   10 CONTINUE
      NFLGC=NFLGC+1
   20 CONTINUE
      IF (AUTOT.LT.0.5.AND.FNOC.LT.0.5) GO TO 90
C
C * * DETERMINE MAXIMUM VELOCITY TERM
C
      IDUMX=1
      JDUMX=1
      KDUMX=1
      IDVMX=1
      JDVMX=1
      KDVMX=1
      IDWMX=1
      JDWMX=1
      KDWMX=1
      IMX=1
      JMX=1
      KMX=1
      IJKMX=1
      IJKUMX=1
      IJKVMX=1
      IJKWMX=1
      DUMX=1.0E-10*DELX(2)/DELT
      DVMX=1.0E-10*DELY(2)/RRI(2)/DELT
      DWMX=1.0E-10*DELZ(2)/DELT
      DUVW=AMAX1(DUMX,DVMX,DWMX)
      IF (FNOC.GT.0.5) DELT=0.5*DELT
      DO 60 K=2,KM1
      DO 60 J=2,JM1
      DO 60 I=2,IM1
      CALL IJKONLY
      IF (NF(IJK).NE.0) GO TO 60
      IF (BETA(IJK).LT.0.0) GO TO 60
      UDM=ABS(UN(IJK))/(XI(I+1)-XI(I))
      VDM=ABS(VN(IJK))/(YJ(J+1)-YJ(J))*RRI(I)
      WDM=ABS(WN(IJK))/(ZK(K+1)-ZK(K))
      IF (UDM.LT.DUMX) GO TO 30
      DUMX=UDM
      IDUMX=I
      JDUMX=J
      KDUMX=K
      IJKUMX=IJK
   30 IF (VDM.LT.DVMX) GO TO 40
      DVMX=VDM
      JDVMX=J
      IDVMX=I
      KDVMX=K
      IJKVMX=IJK
   40 IF (WDM.LT.DWMX) GO TO 50
      DWMX=WDM
      KDWMX=K
      IDWMX=I
      JDWMX=J
      IJKWMX=IJK
   50 UVWS=AMAX1(UDM,VDM,WDM)
      IF (DUVW.GT.UVWS) GO TO 60
      DUVW=UVWS
      IMX=I
      JMX=J
      KMX=K
      IJKMX=IJK
   60 CONTINUE
C
C * * ADJUST DELT TO REFLECT NUMBER OF ITERATIONS
C
      DTMP=1.02
      IF (ITER.LT.ITMIN.AND.LITER.LT.ITMIN) DTMP=1.04
      IF (ISOR.NE.1) GO TO 70
      IF (ITER.GT.ITMOST.AND.LITER.GT.ITMOST) DTMP=0.99
      GO TO 80
   70 IF (ITER.GT.ITCRMX.AND.LITER.GT.ITCRMX) DTMP=0.98
```

```
   80 DELTO=DELT*DTMP
C
C * * SELECT SMALLEST DELT
C
      DELT=AMIN1(DELTO,CON/DUMX,CON/DVMX,CON/DWMX,DTVIS)
   90 IF (DELT.EQ.DELTN) GO TO 110
C
C * * ADJUST BETA ARRAY TO REFLECT DELT CHANGE
C
      IF (ISOR.NE.1) GO TO 110
      DO 100 K=1,KMAX
      DO 100 J=1,JMAX
      DO 100 I=1,IMAX
      CALL IJKONLY
      IF (BETA(IJK).EQ.1.0) GO TO 100
      IF (BETA(IJK).LT.0.0) GO TO 100
      BETA(IJK)=BETA(IJK)*DELTN/DELT
      IF (BETA(IJK).EQ.1.0) BETA(IJK)=1.10
  100 CONTINUE
  110 CONTINUE
C
      LITER=ITER
      IF (MOD(CYCLE,20).NE.0) RETURN
      WRITE (9,120) CYCLE,IMX,JMX,KMX,IJKMX,U(IJKMX),V(IJKMX),W(IJKMX)
      WRITE (12,120) CYCLE,IMX,JMX,KMX,IJKMX,U(IJKMX),V(IJKMX),W(IJKMX)
      RETURN
C
  120 FORMAT (2X,*DELTADJ  *,I5,3I4,I6,3(3X,1PE12.5))
      END
*DK DGAP
      SUBROUTINE DGAP (XE,YE,ZE,XC1,YC1,ZC1)
*CA SLCOM2
*CA SLCOM1
C
C * * DEFINE GRAPH AREA FOR PERSPECTIVE VIEW PLOTS (NO FILM PRODUCED)
C
      XCC=XC1
      YCC=YC1
      ZCC=ZC1
      YEMYC=YE-YC1+1.0E-20
      XXSQ=(XE-XC1)**2
      YYSQ=YEMYC**2
      ZZSQ=(ZE-ZC1)**2
      RTXY=SQRT(XXSQ+YYSQ)
      RHP=1.0/RTXY
      RHP2=1.0/SQRT(XXSQ+YYSQ+ZZSQ)
      CSTH=-YEMYC*RHP
      SNTH=(XE-XC1)*RHP
      TRCR=YEMYC*CSTH-(XE-XC1)*SNTH
      CSPHI=RTXY*RHP2
      SNPHI=(ZE-ZC1)*RHP2
      IF (ABS(SNPHI).GT.0.866) GO TO 10
      CSPHI=1.0
      SNPHI=0.0
   10 CONTINUE
      XEB=(XE-XC1)*CSTH+(YE-YC1)*SNTH
      YEB=(ZC1-ZE)*SNPHI+TRCR*CSPHI
      ZEB=(ZE-ZC1)*CSPHI+TRCR*SNPHI
      IF (IZOOM.EQ.1) GO TO 30
      XIMX=-1.0E+15
      ETAMX=-1.0E+15
      XIMN=+1.0E+15
      ETAMN=+1.0E+15
      DO 20 IXR=1,23,3
      XBND=GRDBN(IXR)
      YBND=GRDBN(IXR+1)
      ZBND=GRDBN(IXR+2)
      TRCR=(YBND-YC1)*CSTH-(XBND-XC1)*SNTH
      XB=(XBND-XC1)*CSTH+(YBND-YC1)*SNTH
      YB=(ZC1-ZBND)*SNPHI+TRCR*CSPHI
      ZZB=(ZBND-ZC1)*CSPHI+TRCR*SNPHI
      YDEN=AMAX1((YB-YEB),1.0E-6)
      YRAT=YEB/YDEN
      XIBND=XEB-(XB-XEB)*YRAT
      ETABND=ZEB-(ZZB-ZEB)*YRAT
      XIMX=AMAX1(XIMX,XIBND)
      XIMN=AMIN1(XIMN,XIBND)
      ETAMX=AMAX1(ETAMX,ETABND)
      ETAMN=AMIN1(ETAMN,ETABND)
   20 CONTINUE
```

```
      DXI=XIMX-XIMN                                                     DGAP  52
      DETA=ETAMX-ETAMN                                                  DGAP  53
      XIETAMX=AMAX1(DXI,DETA)                                           DGAP  54
      GSZ=900.0                                                         DGAP  55
      IF (LPR.EQ.0) GSZ=1020.0                                          DGAP  56
      GDRAT=GSZ/XIETAMX                                                 DGAP  57
   30 CONTINUE                                                          DGAP  58
C                                                                       DGAP  59
      RETURN                                                            DGAP  60
      END                                                               DGAP  61
*DK DRAW                                                                DRAW   1
      SUBROUTINE DRAW                                                   DRAW   2
*CA SLCOM1                                                              DRAW   3
C                                                                       DRAW   4
C * * PLOT VELOCITY VECTORS, PARTICLES, ETC. ON FILM                    DRAW   5
C                                                                       DRAW   6
C * * GRID PLOT                                                         DRAW   7
C                                                                       DRAW   8
C * * VELOCITY VECTOR AND CONSTANT PLANE FREE SURFACE PLOT              DRAW   9
C                                                                       DRAW  10
C******* PLOT A TWO DIMENSIONAL VECTOR AND SURFACE J=2 AND J=JM1        DRAW  11
C                                                                       DRAW  12
      IF (CYL.GT.0.5.AND.(ABS(Y(JM1)/X(IM1)-6.2831853).LT.0.01.OR.ABS(Y DRAW  13
     1 (JM1)/X(IM1)-3.14159265).LT.0.01)) CALL DRAWQ                    DRAW  14
C                                                                       DRAW  15
      IVEWO=1                                                           DRAW  16
      CALL DGAP (XEA(IVEWO),YEA(IVEWO),ZEA(IVEWO),XCA(IVEWO),YCA(IVEWO) DRAW  17
     1 ,ZCA(IVEWO))                                                     DRAW  18
      IF (NVPLTS.EQ.0) GO TO 30                                         DRAW  19
      DO 20 NI=1,NVPLTS                                                 DRAW  20
      IF (IVVEW(NI).EQ.IVEWO) GO TO 10                                  DRAW  21
      IVEWO=IVVEW(NI)                                                   DRAW  22
      CALL DGAP (XEA(IVEWO),YEA(IVEWO),ZEA(IVEWO),XCA(IVEWO),YCA(IVEWO) DRAW  23
     1 ,ZCA(IVEWO))                                                     DRAW  24
   10 CALL VELV (IV1(NI),IV2(NI),JV1(NI),JV2(NI),KV1(NI),KV2(NI),NAV(NI) DRAW  25
     1 ,IPERV(NI))                                                      DRAW  26
   20 CONTINUE                                                          DRAW  27
   30 CONTINUE                                                          DRAW  28
C                                                                       DRAW  29
C * * CONTOUR PLOTS                                                     DRAW  30
C                                                                       DRAW  31
      IVEWO=1                                                           DRAW  32
      IF (NCPLTS.EQ.0) GO TO 60                                         DRAW  33
      DO 50 NI=1,NCPLTS                                                 DRAW  34
      IF (ICVEW(NI).EQ.IVEWO) GO TO 40                                  DRAW  35
      IVEWO=ICVEW(NI)                                                   DRAW  36
      CALL DGAP (XEA(IVEWO),YEA(IVEWO),ZEA(IVEWO),XCA(IVEWO),YCA(IVEWO) DRAW  37
     1 ,ZCA(IVEWO))                                                     DRAW  38
   40 CALL CNTR (IC1(NI),IC2(NI),JC1(NI),JC2(NI),KC1(NI),KC2(NI),NAC(NI) DRAW  39
     1 ,IPERC(NI))                                                      DRAW  40
   50 CONTINUE                                                          DRAW  41
   60 CONTINUE                                                          DRAW  42
C                                                                       DRAW  43
C * * PERSPECTIVE SURFACE PLOTS (NO HIDDEN LINES IN THIS VERSION)       DRAW  44
C                                                                       DRAW  45
      IVEWO=1                                                           DRAW  46
      IF (NSPLTS.EQ.0) GO TO 90                                         DRAW  47
      DO 80 NI=1,NSPLTS                                                 DRAW  48
      IF (ISVEW(NI).EQ.IVEWO) GO TO 70                                  DRAW  49
      IVEWO=ISVEW(NI)                                                   DRAW  50
      CALL DGAP (XEA(IVEWO),YEA(IVEWO),ZEA(IVEWO),XCA(IVEWO),YCA(IVEWO) DRAW  51
     1 ,ZCA(IVEWO))                                                     DRAW  52
   70 CALL SURFPLT (IS1(NI),IS2(NI),JS1(NI),JS2(NI),KS1(NI),KS2(NI),NAS DRAW  53
     1 (NI),IPERS(NI))                                                  DRAW  54
   80 CONTINUE                                                          DRAW  55
   90 CONTINUE                                                          DRAW  56
      RETURN                                                            DRAW  57
      END                                                               DRAW  58
*DK DRF                                                                 DRF    1
      SUBROUTINE DRF (IFM,JFM,KFM)                                      DRF    2
*CA SLCOM1                                                              DRF    3
C                                                                       DRF    4
C * * DRF - DRAW FRAME (PLANE VIEW)                                     DRF    5
C       DRFP -DRAW FRAME IN PERSPECTIVE (ENTRY POINT)                   DRF    6
C                                                                       DRF    7
C * * SELECT CONSTANT PLANE, MPLN                                       DRF    8
C                                                                       DRF    9
      IF (IFM.NE.0) MPLN=1                                              DRF   10
      IF (JFM.NE.0) MPLN=2                                              DRF   11
      IF (KFM.NE.0) MPLN=3                                              DRF   12
```

```
      GO TO (10,20,30), MPLN
C
C * * CONSTANT I PLANE
C
   10 YLEFT=0.0
      YRIGHT=YBBK-YBF
      ZBOT=0.0
      ZTOP=ZBT-ZBB
      IXL=FIXL(MPLN)+YLEFT*XCONV(MPLN)
      IXR=FIXL(MPLN)+YRIGHT*XCONV(MPLN)
      IXB=FIYB+ZBOT*YCONV(MPLN)
      IXT=FIYB+ZTOP*YCONV(MPLN)
      CALL FRAME (IXL,IXR,IXT,IXB)
      CALL FRAME (IXL,IXR,IXT,IXB)
      GO TO 50
C
C * * CONSTANT J PLANE
C
   20 XLEFT=0.0
      XRIGHT=XBR-XBL
      ZBOT=0.0
      ZTOP=ZBT-ZBB
      IXL=FIXL(MPLN)+XLEFT*XCONV(MPLN)
      IXR=FIXL(MPLN)+XRIGHT*XCONV(MPLN)
      IXB=FIYB+ZBOT*YCONV(MPLN)
      IXT=FIYB+ZTOP*YCONV(MPLN)
      CALL FRAME (IXL,IXR,IXT,IXB)
      CALL FRAME (IXL,IXR,IXT,IXB)
      GO TO 50
C
C * * CONSTANT K PLANE
C
   30 IF (CYL.EQ.1.0) GO TO 40
      XLEFT=0.0
      XRIGHT=XBR-XBL
      YBOT=0.0
      YTOP=YBBK-YBF
      IXL=FIXL(MPLN)+XLEFT*XCONV(MPLN)
      IXR=FIXL(MPLN)+XRIGHT*XCONV(MPLN)
      IXB=FIYB+YBOT*YCONV(MPLN)
      IXT=FIYB+YTOP*YCONV(MPLN)
      CALL FRAME (IXL,IXR,IXT,IXB)
      CALL FRAME (IXL,IXR,IXT,IXB)
      GO TO 50
C
C * * DRAW FRAME FOR CONSTANT K-PLANE IN CYLINDRICAL COORDINATES
C
   40 CONTINUE
      CALL DRFCYL (MPLN)
C
C * * LABEL COORDINATE AXES
C
   50 CALL LINCNT (59)
      IX1=AMAX1(0.0,(FIXL(MPLN)-8))
      IY1=FIYB-32
      IX2=FIXL(MPLN)+16
C
C       DRAW OBSTACLE BOUNDARIES FOR JPLANE PLOT
C
      CALL DRWOBS (MPLN)
C
      GO TO (60,70,80), MPLN
C
C * * CONSTANT I PLANE
C
   60 CALL PLT (IX1,IY1,57)
      CALL PLT (IX2,IY2,56)
      RETURN
C
C * * CONSTANT J PLANE
C
   70 CALL PLT (IX1,IY1,57)
      CALL PLT (IX2,IY2,55)
      RETURN
C
C * * CONSTANT K PLANE
C
   80 CONTINUE
      RETURN
C
```

```
      END
*DK DRFCYL
      SUBROUTINE DRFCYL (MPLN)
*CA SLCOM1
C
C * * DRAW FRAME FOR CONSTANT K-PLANE IN CYLINDRICAL COORDINATES
C
      X1=X(1)
      X2=X(IM1)
      CRX=CYL/X2
      XLEFT=XBLC
      YLEFT=YBFC
      JPL=JM1
      DO 20 J=2,JPL
      TH1=CRX*Y(J-1)
      TH2=CRX*Y(J)
      STH1=SIN(TH1)
      CTH1=COS(TH1)
      STH2=SIN(TH2)
      CTH2=COS(TH2)
      XX1=X1*CTH1
      YY1=X1*STH1+Y(J-1)*(1.0-CYL)
      XX2=X1*CTH2
      YY2=X1*STH2+Y(J)*(1.0-CYL)
      XRIGHT1=XX1-XLEFT
      YRIGHT1=YY1-YLEFT
      IX1=FIXL(MPLN)+XRIGHT1*XCONV(MPLN)
      IY1=FIYB+YRIGHT1*YCONV(MPLN)
      XRIGHT2=XX2-XLEFT
      YRIGHT2=YY2-YLEFT
      IX2=FIXL(MPLN)+XRIGHT2*XCONV(MPLN)
      IY2=FIYB+YRIGHT2*YCONV(MPLN)
      CALL DRV (IX1,IY1,IX2,IY2)
      DTH21=(TH2-TH1)/10.0
      THO=TH1
      DO 10 L=1,10
      THO1=THO+(L-1)*DTH21
      THO2=THO+L*DTH21
      CTH1=COS(THO1)
      CTH2=COS(THO2)
      STH1=SIN(THO1)
      STH2=SIN(THO2)
      XX1=X2*CTH1
      XX2=X2*CTH2
      YY1=X2*STH1+Y(J-1)*(1.0-CYL)
      YY2=X2*STH2+Y(J)*(1.0-CYL)
      XRIGHT1=XX1-XLEFT
      YRIGHT1=YY1-YLEFT
      IX1=FIXL(MPLN)+XRIGHT1*XCONV(MPLN)
      IY1=FIYB+YRIGHT1*YCONV(MPLN)
      XRIGHT2=XX2-XLEFT
      YRIGHT2=YY2-YLEFT
      IX2=FIXL(MPLN)+XRIGHT2*XCONV(MPLN)
      IY2=FIYB+YRIGHT2*YCONV(MPLN)
   10 CALL DRV (IX1,IY1,IX2,IY2)
   20 CONTINUE
      IF (ABS(Y(JPL)*CRX-6.28327853).LT.0.0001) RETURN
      Y1=Y(1)
      Y2=Y(JM1)
      TH1=CRX*Y1
      TH2=CRX*Y2
      STH1=SIN(TH1)
      CTH1=COS(TH1)
      STH2=SIN(TH2)
      CTH2=COS(TH2)
      XX1=X1*CTH1
      YY1=X1*STH1+Y1*(1.0-CYL)
      XX2=X2*CTH1
      YY2=X2*STH1+Y1*(1.0-CYL)
      XRIGHT1=XX1-XLEFT
      YRIGHT1=YY1-YLEFT
      IX1=FIXL(MPLN)+XRIGHT1*XCONV(MPLN)
      IY1=FIYB+YRIGHT1*YCONV(MPLN)
      XRIGHT2=XX2-XLEFT
      YRIGHT2=YY2-YLEFT
      IX2=FIXL(MPLN)+XRIGHT2*XCONV(MPLN)
      IY2=FIYB+YRIGHT2*YCONV(MPLN)
      CALL DRV (IX1,IY1,IX2,IY2)
      XX1=X1*CTH2
      YY1=X1*STH2+Y2*(1.0-CYL)
```

```
      XX2=X2*CTH2
      YY2=X2*STH2+Y2*(1.0-CYL)
      XRIGHT1=XX1-XLEFT
      YRIGHT1=YY1-YLEFT
      IX1=FIXL(MPLN)+XRIGHT1*XCONV(MPLN)
      IY1=FIYB+YRIGHT1*YCONV(MPLN)
      XRIGHT2=XX2-XLEFT
      YRIGHT2=YY2-YLEFT
      IX2=FIXL(MPLN)+XRIGHT2*XCONV(MPLN)
      IY2=FIYB+YRIGHT2*YCONV(MPLN)
      CALL DRV (IX1,IY1,IX2,IY2)
C
      RETURN
      END
*DK DRFP
      SUBROUTINE DRFP
*CA SLCOM1
C
C * * DRAW FRAME IN PERSPECTIVE
C
      X1=X(1)
      X2=X(IM1)
      Y1=Y(1)
      Y2=Y(JM1)
      Z1=Z(1)
      Z2=Z(KM1)
      CRX=CYL/X2
      TH1=CRX*Y1
      TH2=CRX*Y2
      CTH1=COS(TH1)
      CTH2=COS(TH2)
      STH1=SIN(TH1)
      STH2=SIN(TH2)
      XX1=X1*CTH1
      YY1=X1*STH1+Y1*(1.0-CYL)
      XX2=X2*CTH1
      YY2=X2*STH1+Y1*(1.0-CYL)
      CALL PCNV (IXI1,IETA1,XX1,YY1,Z1)
      CALL PCNV (IXI2,IETA2,XX2,YY2,Z1)
      CALL DRVEC (IXI1,IETA1,IXI2,IETA2)
      CALL PCNV (IXI3,IETA3,XX1,YY1,Z2)
      CALL PCNV (IXI4,IETA4,XX2,YY2,Z2)
      CALL DRVEC (IXI3,IETA3,IXI4,IETA4)
      CALL PCNV (IXI5,IETA5,XX1,YY1,Z1)
      CALL PCNV (IXI6,IETA6,XX1,YY1,Z2)
      CALL DRVEC (IXI5,IETA5,IXI6,IETA6)
      CALL PCNV (IXI7,IETA7,XX2,YY2,Z1)
      CALL PCNV (IXI8,IETA8,XX2,YY2,Z2)
      CALL DRVEC (IXI7,IETA7,IXI8,IETA8)
      XX1=X1*CTH2
      YY1=X1*STH2+Y2*(1.0-CYL)
      XX2=X2*CTH2
      YY2=X2*STH2+Y2*(1.0-CYL)
      CALL PCNV (IXI1,IETA1,XX1,YY1,Z1)
      CALL PCNV (IXI2,IETA2,XX2,YY2,Z1)
      CALL DRVEC (IXI1,IETA1,IXI2,IETA2)
      CALL PCNV (IXI3,IETA3,XX1,YY1,Z2)
      CALL PCNV (IXI4,IETA4,XX2,YY2,Z2)
      CALL DRVEC (IXI3,IETA3,IXI4,IETA4)
      CALL PCNV (IXI5,IETA5,XX1,YY1,Z1)
      CALL PCNV (IXI6,IETA6,XX1,YY1,Z2)
      CALL DRVEC (IXI5,IETA5,IXI6,IETA6)
      CALL PCNV (IXI7,IETA7,XX2,YY2,Z1)
      CALL PCNV (IXI8,IETA8,XX2,YY2,Z2)
      CALL DRVEC (IXI7,IETA7,IXI8,IETA8)
      JPL=JM1
      DO 10 J=2,JPL
      TH1=CRX*Y(J-1)
      TH2=CRX*Y(J)
      STH1=SIN(TH1)
      CTH1=COS(TH1)
      STH2=SIN(TH2)
      CTH2=COS(TH2)
      XX1=X1*CTH1
      YY1=X1*STH1+Y(J-1)*(1.0-CYL)
      XX2=X1*CTH2
      YY2=X1*STH2+Y(J)*(1.0-CYL)
      CALL PCNV (IXI1,IETA1,XX1,YY1,Z1)
      CALL PCNV (IXI2,IETA2,XX2,YY2,Z1)
      CALL DRVEC (IXI1,IETA1,IXI2,IETA2)
```

```
      CALL PCNV (IXI3,IETA3,XX1,YY1,Z2)
      CALL PCNV (IXI4,IETA4,XX2,YY2,Z2)
      CALL DRVEC (IXI3,IETA3,IXI4,IETA4)
      XX1=X2*CTH1
      XX2=X2*CTH2
      YY1=X2*STH1+Y(J-1)*(1.0-CYL)
      YY2=X2*STH2+Y(J)*(1.0-CYL)
      CALL PCNV (IXI1,IETA1,XX1,YY1,Z1)
      CALL PCNV (IXI2,IETA2,XX2,YY2,Z1)
      CALL DRVEC (IXI1,IETA1,IXI2,IETA2)
      CALL PCNV (IXI3,IETA3,XX1,YY1,Z2)
      CALL PCNV (IXI4,IETA4,XX2,YY2,Z2)
      CALL DRVEC (IXI3,IETA3,IXI4,IETA4)
   10 CONTINUE
C
C * * DRAW OBSTACLE IN PERSPECTIVE
C
C     CALL DROBSP
C
      RETURN
      END
*DK DROBSP
      SUBROUTINE DROBSP
*CA SLCOM1
C
C * * DRAW OBSTACLES IN PERSPECTIVE
C
      IPL=IM1
      JPL=JM1
      KPL=KM1
C
      DO 40 I=1,IPL
      DO 40 J=1,JPL
      DO 40 K=1,KPL
      ACS=1.0
      ARS=1.0
      ABKS=1.0
      ATS=1.0
      IF (BETA(IJK).LT.0.0) ACS=0.0
      IF (BETA(IPJK).LT.0.0) ARS=0.0
      IF (BETA(IJPK).LT.0.0) ABKS=0.0
      IF (BETA(IJKP).LT.0.0) ATS=0.0
C
      IPLX=0
      IF (ACS.EQ.0.0) IPLX=1
      IF (BETA(IJPKP).LT.0.0) IPLX=IPLX+1
      IPLD=0
      IF (ATS.EQ.0.0) IPLD=1
      IF (ABKS.EQ.0.0) IPLD=IPLD+1
      IPLX=IPLX-IPLD
C
      IPLY=0
      IF (ACS.EQ.0.0) IPLY=1
      IF (BETA(IPJKP).LT.0.0) IPLY=IPLY+1
      IPLD=0
      IF (ARS.EQ.0.0) IPLD=1
      IF (ATS.EQ.0.0) IPLD=IPLD+1
      IPLY=IPLY-IPLD
C
      IPLZ=0
      IF (ACS.EQ.0.0) IPLZ=1
      IF (BETA(IPJPK).LT.0.0) IPLZ=IPLZ+1
      IPLD=0
      IF (ARS.EQ.0.0) IPLD=1
      IF (ABKS.EQ.0.0) IPLD=IPLD+1
      IPLZ=IPLZ-IPLD
C
      IF (IPLX.EQ.0.AND.IPLY.EQ.0.AND.IPLZ.EQ.0) GO TO 40
      XB1=X(I)
      YB1=Y(J)
      ZZ1=Z(K)
      CRX=CYL/X(IM1)
      TH1=CRX*YB1
      CTH1=COS(TH1)
      STH1=SIN(TH1)
      XX1=XB1*CTH1
      YY1=XB1*STH1+YB1*(1.0-CYL)
      CALL PCNV (IXI1,IETA1,XX1,YY1,ZZ1)
      IRET=1
      IF (IPLX.EQ.0.OR.I.EQ.1) GO TO 10
```

```
      XX2=X(I-1)*CTH1
      YY2=X(I-1)*STH1+YB1*(1.0-CYL)
      ZZ2=ZZ1
      GO TO 30
   10 IRET=2
      IF (IPLZ.EQ.O.OR.K.EQ.1.) GO TO 20
      XX2=XX1
      YY2=YY1
      ZZ2=Z(K-1)
      GO TO 30
   20 IRET=3
      IF (IPLY.EQ.O.OR.J.EQ.1) GO TO 40
      YB2=Y(J-1)
      TH2=CRX*YB2
      CTH2=COS(TH2)
      STH2=SIN(TH2)
      XX2=XB1*CTH2
      YY2=XB1*STH2+YB2*(1.0-CYL)
      ZZ2=ZZ1
   30 CALL PCNV (IXI2,IETA2,XX2,YY2,ZZ2)
      CALL DRVEC (IXI1,IETA1,IXI2,IETA2)
      GO TO (10,20,40), IRET
   40 CONTINUE
C
      RETURN
      END
*DK DRWOBS
      SUBROUTINE DRWOBS (MPLN)
*CA SLCOM1
C
C     *** DRAW AROUND ALL OBSTACLES
C
      TPI=2.0*PI
C
      GO TO (200,10,200), MPLN
C
   10 CONTINUE
      J=2
      DO 190 I=2,IM1
      ATR=1.0-CYL+CYL*(1.0-EM6)
      ATL=1.0-CYL+CYL*(1.0-EM6)
      ATC=1.0-CYL+CYL*(1.0-EM6)
      DO 190 K=2,KM1
      CALL IJKAJCT
      IF (AC(IJK).LT.EM6) GO TO 190
      AFR=1.0
      AFT=1.0
      AFL=1.0
      AFB=1.0
      IF (AR(IJK).LT.ATR) AFR=AR(IJK)/ATR
      IF (AT(IJK).LT.ATC) AFT=AT(IJK)/ATC
      IF (AR(IMJK).LT.ATL) AFL=AR(IMJK)/ATL
      IF (AT(IJKM).LT.ATC) AFB=AT(IJKM)/ATC
      IF (AC(IJK).GE.ATC) GO TO 130
      IF (I.EQ.2) AFL=AFR-EM6
      IF (I.EQ.IM1) AFR=AFL-EM6
      IF (K.EQ.2) AFB=AFT-EM6
      IF (K.EQ.KM1) AFT=AFB-EM6
      IF ((AFT+AFB).LT.EM6.OR.(AFL+AFR).LT.EM6) GO TO 190
      M=1
      AMN=AFB+AFR
      IF ((AFR+AFT).GT.AMN) GO TO 20
      M=2
      AMN=AFR+AFT
   20 IF ((AFT+AFL).GT.AMN) GO TO 30
      M=3
      AMN=AFT+AFL
   30 IF ((AFL+AFB).GT.AMN) GO TO 40
      M=4
   40 GO TO (50,70,90,110), M
   50 X1=X(I-1)+AFT*DELX(I)
      Y1=Z(K)
      IF (AFT.LT.1.0) GO TO 60
      Y1=Y1-AFR*DELZ(K)
   60 X2=X(I-1)
      Y2=Z(K)-AFL*DELZ(K)
      IF (AFL.LT.1.0) GO TO 170
      X2=X2+AFB*DELX(I)
      GO TO 170
   70 X1=X(I-1)
```

```
      Y1=Z(K-1)+AFL*DELZ(K)
      IF (AFL.LT.1.0) GO TO 80
      X1=X1+AFT*DELX(I)
   80 X2=X(I-1)+AFB*DELX(I)
      Y2=Z(K-1)
      IF (AFB.LT.1.0) GO TO 170
      Y2=Y2+AFR*DELZ(K)
      GO TO 170
   90 X1=X(I)-AFB*DELX(I)
      Y1=Z(K-1)
      IF (AFB.LT.1.0) GO TO 100
      Y1=Y1+AFL*DELZ(K)
  100 X2=X(I)
      Y2=Z(K-1)+AFR*DELZ(K)
      IF (AFR.LT.1.0) GO TO 170
      X2=X2-AFT*DELX(I)
      GO TO 170
  110 X1=X(I)
      Y1=Z(K)-AFR*DELZ(K)
      IF (AFR.LT.1.0) GO TO 120
      X1=X1-AFB*DELX(I)
  120 X2=X(I)-AFT*DELX(I) .
      Y2=Z(K)
      IF (AFT.LT.1.0) GO TO 170
      Y2=Y2-AFL*DELZ(K)
      GO TO 170
  130 IF (AFR.GT.EM6) GO TO 140
      X1=X(I)
      Y1=Z(K-1)
      X2=X1
      Y2=Z(K)
      ASSIGN 140 TO KR1
      GO TO 180
  140 IF (AFT.GT.EM6) GO TO 150
      X1=X(I-1)
      Y1=Z(K)
      X2=X(I)
      Y2=Y1
      ASSIGN 150 TO KR1
      GO TO 180
  150 IF (AFL.GT.EM6) GO TO 160
      X1=X(I-1)
      Y1=Z(K)
      X2=X1
      Y2=Z(K-1)
      ASSIGN 160 TO KR1
      GO TO 180
  160 IF (AFB.GT.EM6) GO TO 190
      X1=X(I-1)
      Y1=Z(K-1)
      X2=X(I)
      Y2=Y1
      ASSIGN 170 TO KR1
      GO TO 180
  170 ASSIGN 190 TO KR1
  180 IX1=FIXL(MPLN)+X1*XCONV(MPLN)
      IX2=FIXL(MPLN)+X2*XCONV(MPLN)
      IY1=FIYB+Y1*YCONV(MPLN)
      IY2=FIYB+Y2*YCONV(MPLN)
      CALL DRV (IX1,IY1,IX2,IY2)
      GO TO KR1, (140,150,160,170,190)
  190 CONTINUE
  200 CONTINUE
      RETURN
      END
*DK DRVEC
      SUBROUTINE DRVEC (IXI1,IETA1,IXI2,IETA2)
*CA SLCOM1
C
C * * AFTER CLIPPING,DRAW VECTOR (A PART OF ZOOM FEATURE)
C
C * * WORKING IN FLOATING POINT NUMBERS
C
      IF (ICLIP.EQ.0) GO TO 90
      IF ((IXI1+IETA1).EQ.0.OR.(IXI2+IETA2).EQ.0) GO TO 100
      XIL=123.0
      XIR=1023.0
      IF (LPR.EQ.0) XIL=2.0
      IF (LPR.EQ.0) XIR=1022.0
      ETAT=1.0
```

```
      ETAB=901.0
      IF (LPR.EQ.0) ETAT=2.0
      IF (LPR.EQ.0) ETAB=1022.0
      CX1=IXI1
      X11=CX1
      CX2=IXI2
      X22=CX2
      CY1=IETA1
      Y11=CY1
      CY2=IETA2
      Y22=CY2
      DX=X22-X11
      DY=Y22-Y11
      IF (ABS(DX).LT.1.0E-3) GO TO 40
      SM=DY/DX
      IF (CX1.GE.XIL) GO TO 10
      CY1=Y11+SM*(XIL-X11)
      CX1=XIL
   10 IF (CX1.LE.XIR) GO TO 20
      CY1=Y11+SM*(XIR-X11)
      CX1=XIR
   20 IF (CX2.GE.XIL) GO TO 30
      CY2=Y11+SM*(XIL-X11)
      CX2=XIL
   30 IF (CX2.LE.XIR) GO TO 40
      CY2=Y11+SM*(XIR-X11)
      CX2=XIR
   40 IF (CX1.LT.XIL.OR.CX1.GT.XIR) GO TO 100
      IF (ABS(DY).LT.1.0E-3) GO TO 80
      SM=DX/DY
      IF (CY1.LE.ETAB) GO TO 50
      CX1=X11+SM*(ETAB-Y11)
      CY1=ETAB
   50 IF (CY1.GE.ETAT) GO TO 60
      CX1=X11+SM*(ETAT-Y11)
      CY1=ETAT
   60 IF (CY2.LE.ETAB) GO TO 70
      CX2=X11+SM*(ETAB-Y11)
      CY2=ETAB
   70 IF (CY2.GE.ETAT) GO TO 80
      CX2=X11+SM*(ETAT-Y11)
      CY2=ETAT
   80 CONTINUE
      IF (CY1.LT.ETAT.OR.CY1.GT.ETAB) GO TO 100
      DS=ABS(CX2-CX1)+ABS(CY2-CY1)
      IF (DS.LT.1.0E-3) GO TO 100
      IXI1=CX1
      IXI2=CX2
      IETA1=CY1
      IETA2=CY2
   90 CALL DRV (IXI1,IETA1,IXI2,IETA2)
  100 RETURN
      END
*DK EQUIB
      SUBROUTINE EQUIB (Y,Z,NX1,BOND,CANGLE,CYL)
      DIMENSION Y(1), Z(1)
      DATA PI /3.141592654/, EPS /1.E-04/
C +++
C +++ USEFUL INTERMEDIATE QUANTITIES
C +++
      RANGLE=CANGLE
      COSTST=COS(RANGLE)
      COSCA=(1.+CYL)*COS(RANGLE)
      OMCYL=1.-CYL
      NX=NX1-1
      DR=1./FLOAT(NX)
      ITER=0
C +++
C +++ SET INITIAL GUESS FOR Y(1)
C +++
      Y(1)=0.
      IF (ABS(BOND).GT.0.) Y(1)=-COSCA/(2.*ABS(BOND))
C +++
C +++ NUMERICAL INTEGRATION
C +++
   10 Z(1)=0.
      ITER=ITER+1
      Z(NX1+1)=0.
      WRITE (2,100) Y(1)
      DO 20 J=2,NX1
```

```
      RJM=DR*FLOAT(J-2)*CYL+OMCYL
      RJ=DR*FLOAT(J-1)*CYL+OMCYL
      RJH=0.5*(RJM+RJ)
      Z(J)=Z(J-1)*RJM/RJ+DR*(RJH/RJ)*(COSCA-BOND*(Y(J-1)+0.5*DR*Z(J-1)
     1 /SQRT(1.-Z(J-1)**2)))
      IF (Z(J).GE.1.) GO TO 30
      SLOPE=Z(J)/SQRT(1.-Z(J)**2)
      SLOPF=Z(J-1)/SQRT(1.-Z(J-1)**2)
      Z(NX1+J)=(SLOPE-SLOPF)/(DR*(1.+SLOPE**2)**1.5)
      Y(J)=Y(J-1)+0.5*DR*(Z(J)/SQRT(1.-Z(J)**2)+Z(J-1)/SQRT(1.-Z(J-1)**2
     1 ))
   20 CONTINUE
      GO TO 40
   30 CONTINUE
      WRITE (2,110) J,Z(J)
      Y(1)=Y(1)*1.05
      WRITE (2,120) Z(NX1)
      IF (ITER.GT.400) GO TO 60
      GO TO 10
C +++
C +++ CHECK CONSTRAINT AND CONVERGENCE
C +++
   40 CONTINUE
      WRITE (2,130)
      YSUM=0.5*DR*Y(NX1)
      DO 50 J=2,NX
      YSUM=YSUM+(FLOAT(J-1)*DR*CYL+OMCYL)*DR*Y(J)
   50 CONTINUE
      WRITE (2,140) YSUM,Y(1),Z(NX1),COSTST
      IF (ABS(Z(NX1)-COS(RANGLE)).LE.EPS.AND.BOND.NE.0.) GO TO 70
      IF (ABS(YSUM).LT.EPS.AND.BOND.EQ.0.) GO TO 70
      Y(1)=Y(1)-YSUM/(OMCYL+CYL*2.)
      IF (ITER.GT.400) GO TO 60
      GO TO 10
C +++
C +++ EXIT IF CONVERGENCE FAILS
C +++
   60 CONTINUE
      WRITE (59,150)
      WRITE (12,150)
      WRITE (9,150)
      CALL EXIT
   70 CONTINUE
      WRITE (2,80) (I,Y(I),Z(I),Z(NX1+I),I=1,NX1)
      WRITE (59,90)
      WRITE (2,90)
      RETURN
C
   80 FORMAT (///3X,1HI,8X,1HY,11X,1HZ,9X,3HKXY/(1X,I3,1P3E12.4))
   90 FORMAT (15H FINISHED EQUIB)
  100 FORMAT (1X,17HSTEP B WITH Y(1)=,1PE14.6)
  110 FORMAT (1X,9HZ TOO BIG,I5,1PE12.4)
  120 FORMAT (1X,17HGO TO STEP B   Z=,1PE12.4)
  130 FORMAT (1X,23HPARTS B AND C CONVERGED)
  140 FORMAT (1X,5HYSUM=,1PE13.5,4H Y1=,E13.5,4H ZN=E13.5,5H COS=,E13.5)
  150 FORMAT (1X,28H*** EQUIB FAILED TO CONVERGE)
      END
*DK FLMSET
      SUBROUTINE FLMSET
*CA SLCOM1
C
C * * CALCULATE MESH BOUNDARIES FOR PLOTTING
C
C * * SET BOUNDARIES FOR RECTANGULAR MESH
C
      XBR=X(IM1)
      Y2=Y(JM1)
      YBBK=Y2
      YBF=Y(1)
      XBL=X(1)
      ZBT=Z(KM1)
      ZBB=Z(1)
      IF (CYL.LT.0.5) GO TO 10
C
C * * RESET BOUNDARIES FOR CYCLINDRICAL COORDINATES
C
      XBL=XBR*COS(Y2/XBR)
      IF (Y2/XBR.GT.0.5*PI) XBL=XBR*COS(Y2/XBR)
      IF ((Y2/XBR).GT.PI) XBL=-XBR
      YBBK=XBR*SIN(Y2/XBR)
```

```
      IF ((Y2/XBR).GT.(0.5*PI)) YBBK=XBR*SIN(Y(1)/XBR)
      IF (Y2/XBR.GE.PI) YBBK=XBR
      YBF=X(1)*SIN(Y(1)/XBR)
      IF ((Y2/XBR).LT.0.5*PI) YBF=X(1)*SIN(Y2/XBR)
      IF (Y2/XBR.GT.PI) YBF=-YBBK
   10 CONTINUE
C
C * * SET BOUNDARY VALUES INTO GRDBN-ARRAY FOR USE IN DGAP SUBROUTINE
C
      GRDBN(1)=XBL
      GRDBN(2)=YBF
      GRDBN(3)=ZBB
      GRDBN(4)=XBR
      GRDBN(5)=YBF
      GRDBN(6)=ZBB
      GRDBN(7)=XBL
      GRDBN(8)=YBBK
      GRDBN(9)=ZBB
      GRDBN(10)=XBR
      GRDBN(11)=YBBK
      GRDBN(12)=ZBB
      GRDBN(13)=XBL
      GRDBN(14)=YBF
      GRDBN(15)=ZBT
      GRDBN(16)=XBR
      GRDBN(17)=YBF
      GRDBN(18)=ZBT
      GRDBN(19)=XBL
      GRDBN(20)=YBBK
      GRDBN(21)=ZBT
      GRDBN(22)=XBR
      GRDBN(23)=YBBK
      GRDBN(24)=ZBT
C
C * * SCALE MESH BOUNDARY VALUES FOR CRT PLOTTING DEVICE
C
C * * FOR K PLANE
C
      FIYB=916.0
      XD=(XBR-XBL)/(YBBK-YBF)
      YY=0.0
      IF (XD.LT.1.13556) YY=1.0
      FIXL(3)=AMAX1(0.0,(511.0-450.0*XD)*YY)
      FIXR=(511.0+450.0*XD)*YY+1022.0*(1.0-YY)
      FIYT=16.0*YY+(916.0-1022.0/XD)*(1.0-YY)
      XCONV(3)=(FIXR-FIXL(3))/(XBR-XBL)
      YCONV(3)=(FIYT-FIYB)/(YBBK-YBF)
C
C * * RESET LIMITS FOR RECTANGULAR MESH FOR I AND J PLANE
C
      XBLC=XBL
      YBFC=YBF
      XBR=X(IM1)
      YBBK=Y(JM1)
      XBL=X(1)
      YBF=Y(1)
C
C * * FOR J PLANE
C
      XD=(XBR-XBL)/(ZBT-ZBB)
      YY=0.0
      IF (XD.LT.1.13556) YY=1.0
      FIXL(2)=AMAX1(0.0,(511.0-450.0*XD)*YY)
      FIXR=(511.0+450.0*XD)*YY+1022.0*(1.0-YY)
      FIYT=16.0*YY+(916.0-1022.0/XD)*(1.0-YY)
      XCONV(2)=(FIXR-FIXL(2))/(XBR-XBL)
      YCONV(2)=(FIYT-FIYB)/(ZBT-ZBB)
C
C * * FOR I PLANE
C
      XD=(YBBK-YBF)/(ZBT-ZBB)
      YY=0.0
      IF (XD.LT.1.13556) YY=1.0
      FIXL(1)=AMAX1(0.0,(511.0-450.0*XD)*YY)
      FIXR=(511.0+450.0*XD)*YY+1022.0*(1.0-YY)
      FIYT=16.0*YY+(916.0-1022.0/XD)*(1.0-YY)
      XCONV(1)=(FIXR-FIXL(1))/(YBBK-YBF)
      YCONV(1)=(FIYT-FIYB)/(ZBT-ZBB)
C
      RETURN
```

```
      END
*DK LPRT
      SUBROUTINE LPRT
*CA SLCOM1
      IF (LPR.EQ.0) RETURN
C
C * * PRINT DATA ON FILM
C
      CALL LINCNT (64)
      CALL FADV (1)
      ASSIGN 30 TO KR1
      GO TO 20
   10 LINES=LINES+1
      IF (LINES.LT.55) GO TO 40
   20 LINES=0
      WRITE (12,70)
      WRITE (12,100) NAME,JNM,DAT,CLK
      WRITE (12,110) ITER,T,DELT,CYCLE
      WRITE (12,90)
      GO TO KR1, (30,40)
   30 ASSIGN 40 TO KR1
      DO 50 J=1,JMAX
      DO 50 I=1,IMAX
      DO 50 K=1,KMAX
      CALL IJKONLY
      WRITE (12,80) I,J,K,U(IJK),V(IJK),W(IJK),P(IJK),PS(IJK),F(IJK),NF
     1 (IJK),NFP(IJK),NFS(IJK),NFO(IJK),BETA(IJK),PETA(IJK),D(IJK)
      GO TO 10
   40 CONTINUE
   50 CONTINUE
      IF (LPR.LE.2) RETURN
C
C * * PRINT DATA ON PAPER
C
      ENTRY LPRT2
      WRITE (9,70)
      WRITE (9,100) NAME,JNM,DAT,CLK
      WRITE (9,110) ITER,T,DELT,CYCLE
      WRITE (9,90)
      DO 60 J=1,JMAX
      DO 60 I=1,IMAX
      DO 60 K=1,KMAX
      CALL IJKONLY
      WRITE (9,80) I,J,K,U(IJK),V(IJK),W(IJK),P(IJK),PS(IJK),F(IJK),NF
     1 (IJK),NFP(IJK),NFS(IJK),NFO(IJK),BETA(IJK),PETA(IJK),D(IJK)
   60 CONTINUE
      RETURN
C
   70 FORMAT (1H1)
   80 FORMAT (3I3,6(1X,1PE11.4),2I3,2I4,3(1X,1PE10.3))
   90 FORMAT (10H   I  J  K,6X,1HU,11X,1HV,11X,1HW,11X,1HP,10X,2HPS,10X,1
     1 HF,6X,2HNF,1X,3HNFP,1X,3HNFS,1X,3HNFO,4X,4HBETA,7X,4HPETA,8X,1HD)
  100 FORMAT (1H ,18X,8A8,1X,A8,2(1X,A8))
  110 FORMAT (5X,6HITER= ,I5,5X,6HTIME= ,1PE12.5,5X,6HDELT= ,1PE12.5,5X,
     1 7HCYCLE= ,I4)
      END
*DK MESHSET
      SUBROUTINE MESHSET
*CA SLCOM1
C
C * * MESH GENERATOR FOR X,Y,Z-COORDINATES
C
      NAMELIST /MESHGN/ NKX,XL,XC,NXL,NXR,DXMN,NKY,YL,YC,NYL,NYR,DYMN
     1 ,NKZ,ZL,ZC,NZL,NZR,DZMN,NOBS,OA2,OA1,OB2,OB1,OC2,OC1,IOH
      DATA NOBS /0/
C
C * * READ IN DATA FROM NAMELIST / MESHGN /
C
      READ (10,MESHGN)
C
C * * WRITE INPUT DATA FOR MESH GENERATOR ONTO TAPES 9 AND 12.
C
      WRITE (9,220) NKX
      WRITE (12,220) NKX
      DO 10 L=1,NKX
      WRITE (9,230) L,XL(L),XC(L),XL(L+1),NXL(L),NXR(L),DXMN(L)
      WRITE (12,230) L,XL(L),XC(L),XL(L+1),NXL(L),NXR(L),DXMN(L)
   10 CONTINUE
C
      WRITE (9,240) NKY
```

```
      WRITE (12,240) NKY
      DO 20 L=1,NKY
      WRITE (9,250) L,YL(L),YC(L),YL(L+1),NYL(L),NYR(L),DYMN(L)
      WRITE (12,250) L,YL(L),YC(L),YL(L+1),NYL(L),NYR(L),DYMN(L)
   20 CONTINUE
C
      WRITE (9,260) NKZ
      WRITE (12,260) NKZ
      DO 30 L=1,NKZ
      WRITE (9,270) L,ZL(L),ZC(L),ZL(L+1),NZL(L),NZR(L),DZMN(L)
      WRITE (12,270) L,ZL(L),ZC(L),ZL(L+1),NZL(L),NZR(L),DZMN(L)
   30 CONTINUE
C
C * * COMPUTE X-COORDINATE VALUES
C
      CALL MESHX (NUMXP1,NKX)
C
C * * COMPUTE Y-COORDINATE VALUES
C
      CALL MESHY (NUMYP1,NKY)
C
C * * COMPUTE Z-COORDINATE VALUES
C
      CALL MESHZ (NUMZP1,NKZ)
C
      WRITE (9,280) IBAR,JBAR,KBAR
      WRITE (12,280) IBAR,JBAR,KBAR
      WRITE (59,280) IBAR,JBAR,KBAR
C
C * * COMPUTE CONSTANT TERMS
C
      IMAX=IBAR+2
      JMAX=JBAR+2
      KMAX=KBAR+2
      IM1=IMAX-1
      JM1=JMAX-1
      KM1=KMAX-1
      IM2=IMAX-2
      JM2=JMAX-2
      KM2=KMAX-2
      II5=IMAX*JMAX
      II0=NQ*IBAR
      II1=NQ*IMAX
      II2=NQ*II5
      II3=NQ*IMAX*JBAR
      II4=NQ*II5*KBAR
      II6=IMAX*JBAR
      II7=IMAX*JM1
C
C * * SET ARRAYS TO INDEFINITE
C
      CALL XINDF
C
      CALL DATEH (DAT)
      CALL CLOCK1 (CLK)
C
C * * INITIALIZE RADII FOR CYL. COORDINATES
C     NOTE, Y=X(IM1)*THETA
C
      DO 40 L=2,IMAX
      RR(L)=1.0-CYL+CYL*X(IM1)/X(L)
   40 RRI(L)=1.0-CYL+CYL*X(IM1)/XI(L)
      RR(1)=1.0
      RRI(1)=1.0
      IF (X(1).EQ.0.0) GO TO 50
      RR(1)=X(IM1)/X(1)
      RRI(1)=X(IM1)/XI(1)
   50 CONTINUE
C
      DO 60 J=1,JMAX
      STHJ(J)=SIN(YJ(J)/X(IM1))
      STHJBK(J)=SIN(Y(J)/X(IM1))
      CTHJ(J)=COS(YJ(J)/X(IM1))
      CTHJBK(J)=COS(Y(J)/X(IM1))
   60 CONTINUE
      JC2PI=0
      IF (CYL.LT.0.5) GO TO 110
      IF (ABS(Y(JM1)/X(IM1)-6.2831853).GT.001) GO TO 90
      JC2PI=1
      INC=JBAR/2
```

```
      JLAST=INC+1
      JSTART=JLAST+1
      DO 70 J=1,JLAST
      JOP(J)=J+INC
   70 CONTINUE
      DO 80 J=JSTART,JMAX
      JOP(J)=J-INC
   80 CONTINUE
      GO TO 110
   90 DO 100 J=1,JMAX
      JOP(J)=JMAX-J+1
  100 CONTINUE
  110 CONTINUE
C * * WRITE MESH GENERATOR OUTPUT
C
      WRITE (9,290)
      WRITE (12,290)
      DO 120 N=1,NUMXP1
      WRITE (9,300) N,X(N),N,RX(N),N,DELX(N),N,RDX(N),N,XI(N),N,RXI(N)
      WRITE (12,300) N,X(N),N,RX(N),N,DELX(N),N,RDX(N),N,XI(N),N,RXI(N)
  120 CONTINUE
      WRITE (9,290)
      WRITE (12,290)
      DO 130 N=1,NUMYP1
      WRITE (9,310) N,Y(N),N,DELY(N),N,RDY(N),N,YJ(N),N,RYJ(N)
      WRITE (12,310) N,Y(N),N,DELY(N),N,RDY(N),N,YJ(N),N,RYJ(N)
  130 CONTINUE
      WRITE (9,290)
      WRITE (12,290)
      DO 140 N=1,NUMZP1
      WRITE (9,320) N,Z(N),N,DELZ(N),N,RDZ(N),N,ZK(N),N,RZK(N)
      WRITE (12,320) N,Z(N),N,DELZ(N),N,RDZ(N),N,ZK(N),N,RZK(N)
  140 CONTINUE
C
       CALCULATE PARTIAL AREAS
C
      CALL ASET
C
      PRINT OBSTACLE DATA
C
      IF (LPR.EQ.0) GO TO 160
      IF (NOBS.LE.0) GO TO 160
      WRITE (9,210)
      WRITE (12,210)
      DO 150 I=1,NOBS
      WRITE (9,200) I,OA2(I),OA1(I),OB2(I),OB1(I),OC2(I),OC1(I),IOH(I)
      WRITE (12,200) I,OA2(I),OA1(I),OB2(I),OB1(I),OC2(I),OC1(I),IOH(I)
  150 CONTINUE
  160 CONTINUE
C * * CALC RATIOS FOR USE IN NO SLIP BDY. CONDITIONS IN SUBROUTINE BC
C
      DELXRL=DELX(1)/DELX(2)
      DELXRR=DELX(IMAX)/DELX(IM1)
      DELYRF=DELY(1)/DELY(2)
      DELYRBK=DELY(JMAX)/DELY(JM1)
      DELZRB=DELZ(1)/DELZ(2)
      DELZRT=DELZ(KMAX)/DELZ(KM1)
C
C * * DETERMINE MINIMUM CELL SIZE (LENGTH)
C
      DELXMN=1.0E+20
      DELYMN=1.0E+20
      DELZMN=1.0E+20
      DO 170 N=1,IMAX
  170 DELXMN=AMIN1(DELX(N),DELXMN)
      DO 180 N=1,JMAX
  180 DELYMN=AMIN1(DELY(N),DELYMN)
      DO 190 N=1,KMAX
  190 DELZMN=AMIN1(DELZ(N),DELZMN)
      DELMN=AMIN1(DELXMN,DELYMN,DELZMN)
      RETURN
C
  200 FORMAT (2X,2HI=,I1,2X,5HOA2= ,1PE12.5,2X,5HOA1= ,E12.5,2X,5HOB2=
     1 ,E12.5,2X,5HOB1= ,E12.5,2X,5HOC2= ,E12.5,2X,5HOC1= ,E12.5,2X,5HIO
     2H= ,I2)
  210 FORMAT (///20X,25HCONIC OBSTACLE PARAMETERS)
  220 FORMAT (2X,5HNKX= ,I4)
  230 FORMAT (2X,8HMESH-X= ,I4,3X,4HXL= ,1PE12.5,3X,4HXC= ,E12.5,3X,4HXR
     1= ,E12.5,3X,5HNXL= ,I4,3X,5HNXR= ,I4,3X,6HDXMN= ,E12.5)
  240 FORMAT (2X,5HNKY= ,I4)
```

```
  250 FORMAT (2X,8HMESH-Y= ,I4,3X,4HYL= ,1PE12.5,3X,4HYC= ,E12.5,3X,4HYR MESHSET     185
     1= ,E12.5,3X,5HNYL= ,I4,3X,5HNYR= ,I4,3X,6HDYMN= ,E12.5)          MESHSET     186
  260 FORMAT (2X,5HNKZ= ,I4)                                            MESHSET     187
  270 FORMAT (2X,8HMESH-Z= ,I4,3X,4HZL= ,1PE12.5,3X,4HZC= ,E12.5,3X,4HZR MESHSET     188
     1= ,E12.5,3X,5HNZL= ,I4,3X,5HNZR= ,I4,3X,6HDZMN= ,E12.5)          MESHSET     189
  280 FORMAT (/,7H  IBAR=,I3,8H   JBAR=,I3,8H   KBAR=,I3)               MESHSET     190
  290 FORMAT (1H1)                                                      MESHSET     191
  300 FORMAT (1X,2HX(,I2,2H)=,1PE12.5,2X,3HRX(,I2,2H)=,1PE12.5,2X,5HDELX MESHSET     192
     1(,I2,2H)=,1PE12.5,1X,4HRDX(,I2,2H)=,1PE12.5,2X,3HXI(,I2,2H)=,1PE12 MESHSET     193
     2 .5,2X,4HRXI(,I2,2H)=,1PE12.5)                                    MESHSET     194
  310 FORMAT (1X,2HY(,I2,2H)=,1PE12.5,3X,5HDELY(,I2,2H)=,1PE12.5,3X,4HRD MESHSET     195
     1Y(,I2,2H)=,1PE12.5,3X,3HYJ(,I2,2H)=,1PE12.5,3X,4HRYJ(,I2,2H)=,1PE1 MESHSET     196
     2 2.5)                                                             MESHSET     197
  320 FORMAT (1X,2HZ(,I2,2H)=,1PE12.5,3X,5HDELZ(,I2,2H)=,1PE12.5,3X,4HRD MESHSET     198
     1Z(,I2,2H)=,1PE12.5,3X,3HZK(,I2,2H)=,1PE12.5,3X,4HRZK(,I2,2H)=,1PE1 MESHSET     199
     2 2.5)                                                             MESHSET     200
      END                                                               MESHSET     201
*DK MESHX                                                               MESHX         1
      SUBROUTINE MESHX (NUMXP1,NKX)                                     MESHX         2
*CA SLCOM1                                                              MESHX         3
C                                                                       MESHX         4
C * * COMPUTE X-COORDINATE VALUES AND RECIPROCALS                       MESHX         5
C                                                                       MESHX         6
      X(1)=XL(1)                                                        MESHX         7
      I11=1                                                             MESHX         8
      DO 30 L=1,NKX                                                     MESHX         9
      DXML=(XC(L)-XL(L))/NXL(L)                                         MESHX        10
      DXMN1=DXMN(L)                                                     MESHX        11
      NT=NXL(L)                                                         MESHX        12
      TN=NT                                                             MESHX        13
      TN=AMAX1(TN,1.0+1.0E-14)                                          MESHX        14
      DXMN(L)=AMIN1(DXMN1,DXML)                                         MESHX        15
      CMC=(XC(L)-XL(L)-TN*DXMN(L))*TN/(TN-1.0)                          MESHX        16
      IF (NT.EQ.1) CMC=0.0                                              MESHX        17
      BMC=XC(L)-XL(L)-CMC                                               MESHX        18
      DO 10 L1=1,NT                                                     MESHX        19
      I11=I11+1                                                         MESHX        20
      RLN=(FLOAT(L1)-TN)/TN                                             MESHX        21
   10 X(I11)=XC(L)+BMC*RLN-CMC*RLN*RLN                                  MESHX        22
      DXMR=(XL(L+1)-XC(L))/NXR(L)                                       MESHX        23
      DXMN(L)=AMIN1(DXMN1,DXMR)                                         MESHX        24
      NT=NXR(L)                                                         MESHX        25
      TN=NT                                                             MESHX        26
      TN=AMAX1(TN,1.0+1.0E-14)                                          MESHX        27
      CMC=(XL(L+1)-XC(L)-TN*DXMN(L))*TN/(TN-1.0)                        MESHX        28
      IF (NT.EQ.1) CMC=0.0                                              MESHX        29
      BMC=XL(L+1)-XC(L)-CMC                                             MESHX        30
      DO 20 L1=1,NT                                                     MESHX        31
      I11=I11+1                                                         MESHX        32
      RLN=FLOAT(L1)/TN                                                  MESHX        33
   20 X(I11)=XC(L)+BMC*RLN+CMC*RLN*RLN                                  MESHX        34
   30 CONTINUE                                                          MESHX        35
      NUMX=I11                                                          MESHX        36
      NUMXM1=NUMX-1                                                     MESHX        37
      NUMXP1=NUMX+1                                                     MESHX        38
      IBAR=NUMX-1                                                       MESHX        39
      DO 40 I11=2,NUMX                                                  MESHX        40
   40 DELX(I11)=X(I11)-X(I11-1)                                         MESHX        41
      DELX(1)=DELX(2)                                                   MESHX        42
      DELX(NUMXP1)=DELX(NUMX)                                           MESHX        43
      X(NUMXP1)=X(NUMX)+DELX(NUMXP1)                                    MESHX        44
C                                                                       MESHX        45
      DO 60 I11=1,NUMXP1                                                MESHX        46
      IF (X(I11).EQ.0.0) GO TO 50                                       MESHX        47
      RX(I11)=1.0/X(I11)                                                MESHX        48
      GO TO 60                                                          MESHX        49
   50 RX(I11)=0.0                                                       MESHX        50
   60 CONTINUE                                                          MESHX        51
      DO 70 I11=2,NUMXP1                                                MESHX        52
      XI(I11)=0.5*(X(I11-1)+X(I11))                                     MESHX        53
      RXI(I11)=1.0/XI(I11)                                              MESHX        54
      IF (I11.LT.NUMXP1) RDXP(I11)=2.0/(DELX(I11)+DELX(I11+1))          MESHX        55
   70 RDX(I11)=1.0/DELX(I11)                                            MESHX        56
      XI(1)=XI(2)-DELX(2)                                               MESHX        57
      RXI(1)=1.0/XI(1)                                                  MESHX        58
      RDX(1)=1.0/DELX(1)                                                MESHX        59
      RDXP(1)=2.0/(DELX(1)+DELX(2))                                     MESHX        60
C                                                                       MESHX        61
      RETURN                                                            MESHX        62
      END                                                               MESHX        63
```

```
*DK MESHY
      SUBROUTINE MESHY (NUMYP1,NKY)
*CA SLCOM1
C
C * * COMPUTE Y-COORDINATE VALUES AND RECIPROCALS
C
      IF (NKY.EQ.1.AND.(NYL(1).EQ.0.OR.NYR(1).EQ.0)) GO TO 40
      Y(1)=YL(1)
      J11=1
      DO 30 L=1,NKY
      DYML=(YC(L)-YL(L))/NYL(L)
      DYMN1=DYMN(L)
      NT=NYL(L)
      TN=NT
      TN=AMAX1(TN,1.0+1.0E-14)
      DYMN(L)=AMIN1(DYMN1,DYML)
      CMC=(YC(L)-YL(L)-TN*DYMN(L))*TN/(TN-1.0)
      IF (NT.EQ.1) CMC=0.0
      BMC=YC(L)-YL(L)-CMC
      DO 10 L1=1,NT
      J11=J11+1
      RLN=(FLOAT(L1)-TN)/TN
   10 Y(J11)=YC(L)+BMC*RLN-CMC*RLN*RLN
      DYMR=(YL(L+1)-YC(L))/NYR(L)
      DYMN(L)=AMIN1(DYMN1,DYMR)
      NT=NYR(L)
      TN=NT
      TN=AMAX1(TN,1.0+1.0E-14)
      CMC=(YL(L+1)-YC(L)-TN*DYMN(L))*TN/(TN-1.0)
      IF (NT.EQ.1) CMC=0.0
      BMC=YL(L+1)-YC(L)-CMC
      DO 20 L1=1,NT
      J11=J11+1
      RLN=FLOAT(L1)/TN
   20 Y(J11)=YC(L)+BMC*RLN+CMC*RLN*RLN
   30 CONTINUE
      GO TO 50
   40 J11=2
      Y(J11)=1.0
      Y(J11-1)=0.0
   50 CONTINUE
      NUMY=J11
      NUMYM1=NUMY-1
      NUMYP1=NUMY+1
      JBAR=NUMY-1
      DO 60 J11=2,NUMY
   60 DELY(J11)=Y(J11)-Y(J11-1)
      DELY(1)=DELY(2)
      DELY(NUMYP1)=DELY(NUMY)
      Y(NUMYP1)=Y(NUMY)+DELY(NUMYP1)
C
      DO 80 J11=1,NUMYP1
      IF (Y(J11).EQ.0.0) GO TO 70
      RY(J11)=1.0/Y(J11)
      GO TO 80
   70 RY(J11)=0.0
   80 CONTINUE
      DO 90 J11=2,NUMYP1
      YJ(J11)=0.5*(Y(J11-1)+Y(J11))
      RYJ(J11)=1.0/YJ(J11)
      IF (J11.LT.NUMYP1) RDYP(J11)=2.0/(DELY(J11)+DELY(J11+1))
   90 RDY(J11)=1.0/DELY(J11)
      YJ(1)=YJ(2)-DELY(2)
      RYJ(1)=1.0/YJ(1)
      RDY(1)=1.0/DELY(1)
      RDYP(1)=2.0/(DELY(1)+DELY(2))
C
      RETURN
      END
*DK MESHZ
      SUBROUTINE MESHZ (NUMZP1,NKZ)
*CA SLCOM1
C
C * * COMPUTE Z-COORDINATE VALUES AND RECIPROCALS
C
      Z(1)=ZL(1)
      K11=1
      DO 30 L=1,NKZ
      DZML=(ZC(L)-ZL(L))/NZL(L)
      DZMN1=DZMN(L)
```

```
      NT=NZL(L)
      TN=NT
      TN=AMAX1(TN,1.0+1.0E-14)
      DZMN(L)=AMIN1(DZMN1,DZML)
      CMC=(ZC(L)-ZL(L)-TN*DZMN(L))*TN/(TN-1.0)
      IF (NT.EQ.1) CMC=0.0
      BMC=ZC(L)-ZL(L)-CMC
      DO 10 L1=1,NT
      K11=K11+1
      RLN=(FLOAT(L1)-TN)/TN
   10 Z(K11)=ZC(L)+BMC*RLN-CMC*RLN*RLN
      DZMR=(ZL(L+1)-ZC(L))/NZR(L)
      DZMN(L)=AMIN1(DZMN1,DZMR)
      NT=NZR(L)
      TN=NT
      TN=AMAX1(TN,1.0+1.0E-14)
      CMC=(ZL(L+1)-ZC(L)-TN*DZMN(L))*TN/(TN-1.0)
      IF (NT.EQ.1) CMC=0.0
      BMC=ZL(L+1)-ZC(L)-CMC
      DO 20 L1=1,NT
      K11=K11+1
      RLN=FLOAT(L1)/TN
   20 Z(K11)=ZC(L)+BMC*RLN+CMC*RLN*RLN
   30 CONTINUE
      NUMZ=K11
      NUMZM1=NUMZ-1
      NUMZP1=NUMZ+1
      KBAR=NUMZ-1
      DO 40 K11=2,NUMZ
   40 DELZ(K11)=Z(K11)-Z(K11-1)
      DELZ(1)=DELZ(2)
      DELZ(NUMZP1)=DELZ(NUMZ)
      Z(NUMZP1)=Z(NUMZ)+DELZ(NUMZP1)
C
      DO 60 K11=1,NUMZP1
      IF (Z(K11).EQ.0.0) GO TO 50
      RZ(K11)=1.0/Z(K11)
      GO TO 60
   50 RZ(K11)=0.0
   60 CONTINUE
      DO 70 K11=2,NUMZP1
      ZK(K11)=0.5*(Z(K11-1)+Z(K11))
      RZK(K11)=1.0/ZK(K11)
      IF (K11.LT.NUMZP1) RDZP(K11)=2.0/(DELZ(K11)+DELZ(K11+1))
   70 RDZ(K11)=1.0/DELZ(K11)
      ZK(1)=ZK(2)-DELZ(2)
      RZK(1)=1.0/ZK(1)
      RDZ(1)=1.0/DELZ(1)
      RDZP(1)=2.0/(DELZ(1)+DELZ(2))
C
      RETURN
      END
*DK PCNV
      SUBROUTINE PCNV (IXI,IETA,XX1,YY1,ZZ1)
*CA SLCOM1
*CA SLCOM2
C
C * * PERSPECTIVE CONVERT SUBR - PRODUCES 4020 COORDINATES FOR
C     PERSPECTIVE VIEWS
C
      IXI=0
      IETA=0
      TRCR=(YY1-YCC)*CSTH-(XX1-XCC)*SNTH
      XB=(XX1-XCC)*CSTH+(YY1-YCC)*SNTH
      YB=(ZCC-ZZ1)*SNPHI+TRCR*CSPHI
      ZZB=(ZZ1-ZCC)*CSPHI+TRCR*SNPHI
      YDEN=YB-YEB
      IF (YDEN.LT.1.0E-6) GO TO 10
      YRAT=YEB/YDEN
      XID=XEB-(XB-XEB)*YRAT
      ETA=ZEB-(ZZB-ZEB)*YRAT
      ISH=61
      IF (LPR.EQ.0) ISH=0
      XIA=0.5*(XIMX+XIMN)
      ETAA=0.5*(ETAMX+ETAMN)
      IETA=512-(ETA-ETAA)*GDRAT-ISH
      IXI=512+(XID-XIA)*GDRAT+ISH
   10 CONTINUE
C
      RETURN
```

```
      END
*DK PDFCALC
*DK PETACAL
      SUBROUTINE PETACAL
*CA SLCOM1
C
C * * DETERMINE PROVISIONAL NF VALUES, SURFACE TENSION PRESSURES, AND
C * * PRESSURE INTERPOLATION FACTOR PETA FOR SURFACE AND NEIGHBOR CELLS
C
      DO 10 K=1,KMAX
      DO 10 J=1,JMAX
      DO 10 I=1,IMAX
      CALL CALCIJK
      NFO(IJK)=NF(IJK)
      NF(IJK)=0
      PETA(IJK)=1.0
      PS(IJK)=0.0
      NFP(IJK)=0
      NFS(IJK)=0
   10 CONTINUE
C
      CALL PCAL
C     CALC SURFACE TENSION
      IF (ISURFT.EQ.1) CALL SURF1ON
C
      DO 100 K=2,KM1
      DO 100 J=2,JM1
      DO 100 I=2,IM1
      CALL IJKAJCT
      IF (BETA(IJK).LT.0.0) GO TO 100
      L=I
      M=J
      N=K
      NFC=NF(IJK)+1
      GO TO (100,20,30,40,50,60,70,80,100), NFC
   20 DSUR=DELX(I)
      DNBR=DELX(I-1)
      L=I-1
      GO TO 90
   30 DSUR=DELX(I)
      DNBR=DELX(I+1)
      L=I+1
      GO TO 90
   40 DSUR=DELY(J)
      DNBR=DELY(J-1)
      M=J-1
      GO TO 90
C       RADIAL DISTANCE FACTOR CANCELS
   50 DSUR=DELY(J)
      DNBR=DELY(J+1)
      M=J+1
      GO TO 90
   60 DSUR=DELZ(K)
      DNBR=DELZ(K-1)
      N=K-1
      GO TO 90
   70 DSUR=DELZ(K)
      DNBR=DELZ(K+1)
      N=K+1
      GO TO 90
   80 P(IJK)=0.1666667*(P(IPJK)+P(IJPK)+P(IJKP)+P(IMJK)+P(IJMK)+P(IJKM))
      GO TO 100
   90 CONTINUE
C
C * * CALCULATE PETA
C
      LMN=NQ*(II5*(N-1)+IMAX*(M-1)+(L-1))+1
      SDIS=F(IJK)*DSUR+F(LMN)*DNBR*0.5
      DCNT=0.5*(DSUR+DNBR)
      SDIS=AMAX1(SDIS,0.5*DCNT)
      PETA(IJK)=DCNT/SDIS
      IF (BETA(LMN).LT.0.0.OR.NF(LMN).NE.0) PETA(IJK)=1.0
      IF (PETA(IJK).GT.2.0) PETA(IJK)=2.0
      IF (PETA(IJK).LT.0.0) PETA(IJK)=0.0
  100 CONTINUE
C
C * * CALCULATE PETA IN NEIGHBORING INTERPOLATION CELLS
C
      CALL PETASET
C
```

```
      IF (CYCLE.GT.O) CALL PRESCK
C
      RETURN
      END
*DK PCAL
      SUBROUTINE PCAL
*CA SLCOM1
C
C * * DETERMINE PROVISIONAL NF VALUES
C
      DO 10 K=1,KMAX
      DO 10 J=1,JMAX
      DO 10 I=1,IMAX
      CALL IJKONLY
      NF(IJK)=7
      IF (I.EQ.1.OR.I.EQ.IMAX.OR.J.EQ.1.OR.J.EQ.JMAX.OR.K.EQ.1.OR.K.EQ
     1 .KMAX.OR.BETA(IJK).LT.O.O) NF(IJK)=O
   10 CONTINUE
      DO 160 K=2,KM1
      DO 160 J=2,JM1
      DO 160 I=2,IM1
      CALL CALCIJK
      IF (BETA(IJK).LT.O.O) GO TO 20
      IF (F(IJK).GT.EMF1.OR.AC(IJK).LT.EMF) GO TO 20
      IF (F(IJK).GT.EMF) GO TO 30
      NF(IJK)=8
      NFP(IJK)=NF(IJK)
      GO TO 160
   20 NF(IJK)=O
      NFP(IJK)=NF(IJK)
      GO TO 160
C
C      LOOKING ONLY AT PARTIALLY FILLED CELLS TO FIND SURFACE CELLS
C        DETERMINE NF VALUE FOR PROVISIONAL SURFACE ORIENTATION
C
   30 FXM=F(IJK)
      FXP=F(IJK)
      FZM=F(IJK)
      FZP=F(IJK)
      FYP=F(IJK)
      FYM=F(IJK)
      IF (CYL.EQ.1.O.AND.I.EQ.2) IMJK=NQ*(II5*(K-1)+IMAX*(JOP(J)-1)+(I-1
     1 ))+1
      IF (AR(IMJK).GT.EMF) FXM=F(IMJK)
      IF (AR(IJK).GT.EMF) FXP=F(IPJK)
      IF (AT(IJKM).GT.EMF) FZM=F(IJKM)
      IF (AT(IJK).GT.EMF) FZP=F(IJKP)
      IF (ABK(IJK).GT.EMF) FYP=F(IJPK)
      IF (ABK(IJMK).GT.EMF) FYM=F(IJMK)
C
C       INITIALIZE 3 INDICES FOR NF DECISIONS AND 3 VARIABLES FOR
C       VOLUME OF FLUID CALCULATIONS
C
      MOBS=1
      INF=1
      IOBS=1
      VF=O.O
      VFXM=O.O
      VFXP=O.O
C
C       SET UP DO LOOPS FOR VOLUME OF FLUID CALCULATIONS AT XM AND XP
C
      DO 50 KK=1,3
      N=K-2+KK
      DO 40 JJ=1,3
      M=J-2+JJ
      LMN=NQ*(IMAX*JMAX*(N-1)+IMAX*(M-1)+(I-1))+1
      LMNM=LMN-NQ
      LMNP=LMN+NQ
      VFXM=VFXM+F(LMNM)
      VFXP=VFXP+F(LMNP)
   40 CONTINUE
   50 CONTINUE
C
C      CALCULATE NF=1 AND RESET INDICES IF APPROPIATE
C
      IF (BETA(IMJK).LT.O.O) IOBS=2
      MOBS=MOBS+(IOBS-1)
      IF (IOBS.EQ.2) GO TO 60
      IF (FXM.LT.EMF) GO TO 60
```

```
      INF=INF+1
      IF (VFXM.GT.VF) NF(IJK)=1
      IF (NF(IJK).EQ.1) VF=VFXM
   60 CONTINUE
      IOBS=1
C
C      CALCULATE NF=2 AND RESET INDICES IF APPROPIATE
C
      IF (BETA(IPJK).LT.0.0) IOBS=2
      MOBS=MOBS+(IOBS-1)
      IF (IOBS.EQ.2) GO TO 70
      IF (FXP.LT.EMF) GO TO 70
      INF=INF+1
      IF (VFXP.GT.VF) NF(IJK)=2
      IF (NF(IJK).EQ.2) VF=VFXP
   70 CONTINUE
      IOBS=1
C
C      INITIALIZE 2 VARIABLES FOR VOLUME OF FLUID CALCULATIONS
C
      VFZM=0.0
      VFZP=0.0
C
C      SET UP DO LOOPS FOR VOLUME OF FLUID CALCULATIONS AT ZM AND ZP
C
      DO 90 II=1,3
      L=I-2+II
      DO 80 JJ=1,3
      M=J-2+JJ
      LMN=NQ*(IMAX*JMAX*(K-1)+IMAX*(M-1)+(L-1))+1
      LMNM=LMN-II2
      LMNP=LMN+II2
      VFZM=VFZM+F(LMNM)
      VFZP=VFZP+F(LMNP)
   80 CONTINUE
   90 CONTINUE
C
C     CALCULATE NF=5 AND RESET INDICES IF APPROPIATE
C
      IF (BETA(IJKM).LT.0.0) IOBS=2
      MOBS=MOBS+(IOBS-1)
      IF (IOBS.EQ.2) GO TO 100
      IF (FZM.LT.EMF) GO TO 100
      INF=INF+1
      IF (VFZM.GT.VF) NF(IJK)=5
      IF (NF(IJK).EQ.5) VF=VFZM
  100 CONTINUE
      IOBS=1
C
C      CALCULATE NF=6 AND RESET INDICES IF APPROPIATE
C
      IF (BETA(IJKP).LT.0.0) IOBS=2
      MOBS=MOBS+(IOBS-1)
      IF (IOBS.EQ.2) GO TO 110
      IF (FZP.LT.EMF) GO TO 110
      INF=INF+1
      IF (VFZP.GT.VF) NF(IJK)=6
      IF (NF(IJK).EQ.6) VF=VFZP
  110 CONTINUE
      IOBS=1
C
C      INITIALIZE 2 VARIABLES FOR VOLUME OF FLUID CALCULATIONS
C
      VFYM=0.0
      VFYP=0.0
C
C      SET UP DO LOOPS FOR VOLUME OF FLUID CALCULATIONS AT YM AND YP
C
      DO 130 KK=1,3
      N=K-2+KK
      DO 120 II=1,3
      L=I-2+II
      LMN=NQ*(IMAX*JMAX*(N-1)+IMAX*(J-1)+(L-1))+1
      LMNM=LMN-II1
      LMNP=LMN+II1
      VFYM=VFYM+F(LMNM)
      VFYP=VFYP+F(LMNP)
  120 CONTINUE
  130 CONTINUE
C
```

```
C      CALCULATE NF=3 AND RESET INDICES IF APPROPIATE
C
       IF (BETA(IJMK).LT.0.0) IOBS=2
       MOBS=MOBS+(IOBS-1)
       IF (IOBS.EQ.2) GO TO 140
       IF (FYM.LT.EMF) GO TO 140
       INF=INF+1
       IF (VFYM.GT.VF) NF(IJK)=3
       IF (NF(IJK).EQ.3) VF=VFYM
   140 CONTINUE
       IOBS=1
C
C       CALCULATE NF=4 AND RESET INDICES IF APPROPIATE
C
       IF (BETA(IJPK).LT.0.0) IOBS=2
       MOBS=MOBS+(IOBS-1)
       IF (IOBS.EQ.2) GO TO 150
       IF (FYP.LT.EMF) GO TO 150
       INF=INF+1
       IF (VFYP.GT.VF) NF(IJK)=4
       IF (NF(IJK).EQ.4) VF=VFYP
   150 CONTINUE
C
C       IF WE DO NOT RESET NF VALUE DEFAULT CORRECTLY GIVES ISOLATED
C       CELL : NF=7
C
C       IF WE DO RESET NF VALUE WE WILL HAVE CORRECT NF VALUE - EXCEPT
C       WHEN THERE IS FLUID IN EACH NHBR CELL THAT IS NOT AN OBSTACLE
C       CELL
C
C       IN THIS EXCEPTIONAL CASE WE SHOULD HAVE NF=0
C
C       THIS IS DONE BY DEFINING A CRITICAL VALUE FOR INF AND
C       TESTING ON THAT VALUE
C
       INFCR=8-MOBS
       IF (INF.EQ.INFCR.AND.INFCR.GT.1) NF(IJK)=0
C
       NFP(IJK)=NF(IJK)
C
   160 CONTINUE
       RETURN
       END
*DK PETASET
       SUBROUTINE PETASET
*CA SLCOM1
C
C * * CALCULATE PETA IN NEIGHBORING INTERPOLATION CELLS
C
       DO 90 K=1,KMAX
       DO 90 J=1,JMAX
       DO 90 I=1,IMAX
       CALL CALCIJK
       NFF=NF(IJK)
       IF (NFF.EQ.0.OR.BETA(IJK).LT.0.0) GO TO 90
       IF (NFF.GT.7) GO TO 80
       L=I
       M=J
       N=K
       GO TO (10,20,30,40,50,60,90), NFF
    10 L=I-1
       AMN=AR(IMJK)
       DNBR=DELX(L)
       DCNT=0.5*(DNBR+DELX(I))
       GO TO 70
    20 L=I+1
       AMN=AR(IJK)
       DNBR=DELX(L)
       DCNT=0.5*(DNBR+DELX(I))
       GO TO 70
    30 M=J-1
       AMN=ABK(IJMK)
       DNBR=DELY(M)/RRI(I)
       DCNT=0.5*(DNBR+DELY(J)/RRI(I))
       GO TO 70
    40 M=J+1
       AMN=ABK(IJK)
       DNBR=DELY(M)/RRI(I)
       DCNT=0.5*(DNBR+DELY(J)/RRI(I))
       GO TO 70
```

```
   50 N=K-1
      AMN=AT(IJKM)
      DNBR=DELZ(N)
      DCNT=0.5*(DNBR+DELZ(K))
      GO TO 70
   60 N=K+1
      AMN=AT(IJK)
      DNBR=DELZ(N)
      DCNT=0.5*(DNBR+DELZ(K))
   70 CONTINUE
      LMN=NQ*(II5*(N-1)+IMAX*(M-1)+(L-1))+1
      IF (NF(LMN).GT.0.OR.AMN.LT.EM6) GO TO 90
      BPD=1.0/PETA(LMN)-BETA(LMN)*(1.0-PETA(IJK))*AMN/AC(LMN)*DELT/(DNBR
     1 *DCNT)
      PETA(LMN)=AMIN1(1.0/BPD,1.98/OMG)
      GO TO 90
   80 CONTINUE
C
C * * SET VOID REGION NFF PRESSURE INTO CELL IJK
C
      P(IJK)=PR(NFF)
   90 CONTINUE
C
      RETURN
      END
*DK PRESCK
      SUBROUTINE PRESCK
C
*CA SLCOM1
      DATA PSAT /0.0/
C
      DO 150 K=2,KM1
      DO 150 J=2,JM1
      DO 150 I=2,IM1
      CALL IJKAJCT
      IF (AC(IJK).LT.EM6.OR.BETA(IJK).LE.0.0) GO TO 130
      NFF=NF(IJK)
      NFFO=NFO(IJK)
      IF (NFF.EQ.NFFO) GO TO 150
      IF (NFF.EQ.0) GO TO 140
      IF (NFF.GT.7) GO TO 100
      L=I
      M=J
      N=K
      GO TO (10,20,30,40,50,60,70), NFF
   10 L=I-1
      GO TO 80
   20 L=I+1
      GO TO 80
   30 M=J-1
      GO TO 80
   40 M=J+1
      GO TO 80
   50 N=K-1
      GO TO 80
   60 N=K+1
      GO TO 80
   70 GO TO 90
   80 LMN=NQ*(II5*(N-1)+IMAX*(M-1)+(L-1))+1
      NFEL=NF(IMJK)
      NFER=NF(IPJK)
      NFEF=NF(IJMK)
      NFEBK=NF(IJPK)
      NFEB=NF(IJKM)
      NFET=NF(IJKP)
      NFE=MAX0(NFEL,NFER,NFEBK,NFEF,NFEB,NFET,1)
      PSURF=PS(IJK)+PR(NFE)
      PLMN=P(LMN)
      IF (NF(LMN).NE.0.AND.BETA(IJK).GT.0.0) PLMN=PSURF
      P(IJK)=(1.0-PETA(IJK))*PLMN+PETA(IJK)*PSURF
      GO TO 150
   90 IF (PSAT.LE.0.0) GO TO 120
      IF (F(IJK).LT.EMF1) GO TO 110
      PMPS=0.0
      IF (PN(IJK).LT.PSAT) PMPS=P(IJK)-PSAT
      DIJK=(RRI(I)*(RDX(I)*(U(IJK)*AR(IJK)/RR(I)-U(IMJK)*AR(IMJK)/RR(I-1
     1 ))+RDY(J)*(V(IJK)*ABK(IJK)-V(IJMK)*ABK(IJMK)))+RDZ(K)*(W(IJK)*AT
     2 (IJK)-W(IJKM)*AT(IJKM)))/AC(IJK)
      DIJK=DIJK-PMPS**2/DELT
      P(IJK)=PN(IJK)-DIJK*BETA(IJK)/(1.0-2.0*PMPS*BETA(IJK)/DELT)
```

```
      GO TO 150                                                         PRESCK  56
  100 P(IJK)=PR(NFF)                                                    PRESCK  57
      GO TO 150                                                         PRESCK  58
  110 P(IJK)=PSAT                                                       PRESCK  59
      GO TO 150                                                         PRESCK  60
  120 P(IJK)=PN(IJK)                                                    PRESCK  61
      GO TO 150                                                         PRESCK  62
  130 P(IJK)=0.0                                                        PRESCK  63
  140 CONTINUE                                                          PRESCK  64
  150 CONTINUE                                                          PRESCK  65
C                                                                       PRESCK  66
      IF (CYCLE.LT.9999) RETURN                                         PRESCK  67
C                                                                       PRESCK  68
      DO 160 K=2,KM1                                                    PRESCK  69
      DO 160 J=2,JM1                                                    PRESCK  70
      DO 160 I=2,IM1                                                    PRESCK  71
      CALL IJKAJCT                                                      PRESCK  72
      IF (AC(IJK).LT.EM6.OR.BETA(IJK).LE.0.0) GO TO 160                 PRESCK  73
      NFF=NF(IJK)                                                       PRESCK  74
      NFFO=NFO(IJK)                                                     PRESCK  75
      IF (NFF.EQ.NFFO) GO TO 160                                        PRESCK  76
      WRITE (9,170) CYCLE,I,J,K,NF(IJK),NFO(IJK),D(IJK),P(IJK),PN(IJK),U PRESCK  77
     1 (IJK),U(IMJK),V(IJK),V(IJMK),W(IJK),W(IJKM)                      PRESCK  78
  160 CONTINUE                                                          PRESCK  79
      RETURN                                                            PRESCK  80
C                                                                       PRESCK  81
  170 FORMAT (* NF SWITCH  AFTER PETACAL  *,6I4/9(2X,1PE12.5))          PRESCK  82
      END                                                               PRESCK  83
*DK PLTPT                                                               PLTPT    1
      SUBROUTINE PLTPT (XONE,YONE,ICHAR,ISYM)                           PLTPT    2
*CA SLCOM1                                                              PLTPT    3
      PARAMETER (IBAR2Q=2*IBAR2-2)                                      PLTPT    4
      COMMON BETAQ(IBAR2Q,KBAR2), FQ(IBAR2Q,KBAR2), ACQ(IBAR2Q,KBAR2),  PLTPT    5
     1 ARQ(IBAR2Q,KBAR2), ATQ(IBAR2Q,KBAR2), UQ(IBAR2Q,KBAR2), VQ(IBAR2Q PLTPT    6
     2 ,KBAR2), XQ(IBAR2Q), XIQ(IBAR2Q), YQ(KBAR2), YJQ(KBAR2), IM1Q,   PLTPT    7
     3 JM1Q, JBAR2Q, JMAXQ, IMAXQ, SF, XSHFT, YSHFT, DELXQ(IBAR2Q),     PLTPT    8
     4 DELYQ(KBAR2), XMINQ, XMAXQ, YMINQ, YMAXQ                         PLTPT    9
C                                                                       PLTPT   10
C +++ PLOT (DRAW) A POINT                                               PLTPT   11
C +++ PROVIDES A SYSTEM DEPENDANT CALL                                  PLTPT   12
C                                                                       PLTPT   13
      IC=0                                                              PLTPT   14
      X1=XONE                                                           PLTPT   15
      Y1=YONE                                                           PLTPT   16
      XO1=(X1-XMINQ)*SF+XSHFT                                           PLTPT   17
      YO1=(Y1-YMINQ)*SF+YSHFT                                           PLTPT   18
      IX1=16.+900.0*XO1                                                 PLTPT   19
      IY1=16.+900.0*(1.0-YO1)                                           PLTPT   20
      CALL PLT (IX1,IY1,42)                                             PLTPT   21
      IF (ABS(X1).LE.EM6) GO TO 10                                      PLTPT   22
   10 RETURN                                                            PLTPT   23
      END                                                               PLTPT   24
*DK PRESCR                                                              PRESCR   1
      SUBROUTINE PRESCR                                                 PRESCR   2
*CA SLCOM1                                                              PRESCR   3
C                                                                       PRESCR   4
      DIMENSION FF(IBAR2*JBAR2*KBAR2), AF(IBAR2*JBAR2*KBAR2), AD(IBAR2  PRESCR   5
     1 *JBAR2*KBAR2), QQ(IBAR2*JBAR2*KBAR2), VLM(IBAR2*JBAR2*KBAR2), SSM PRESCR   6
     2 (IBAR2*JBAR2*KBAR2), RXR(IBAR2*JBAR2*KBAR2), RXL(IBAR2*JBAR2     PRESCR   7
     3 *KBAR2), RZT(IBAR2*JBAR2*KBAR2), RZB(IBAR2*JBAR2*KBAR2), RYBK    PRESCR   8
     4 (IBAR2*JBAR2*KBAR2), RYF(IBAR2*JBAR2*KBAR2), DIV(IBAR2*JBAR2     PRESCR   9
     5 *KBAR2), DP(IBAR2*JBAR2*KBAR2), CQ(IBAR2*JBAR2*KBAR2)            PRESCR  10
C                                                                       PRESCR  11
      DATA ADEFM, BDEFM /100.0,0.10/                                    PRESCR  12
      DATA ITMAX, ITMIN, ITMOST /500,5,90/                              PRESCR  13
      DATA KTRAN /1/                                                    PRESCR  14
      DATA PSAT /0.0/                                                   PRESCR  15
C                                                                       PRESCR  16
C     THIS SUBROUTINE MUST BE MODIFIED FOR BOUNDARY CONDITIONS          PRESCR  17
C     OTHER THAN NORMAL VELOCITY EQUAL TO ZERO                          PRESCR  18
C                                                                       PRESCR  19
      GO TO (10,200), KTRAN                                             PRESCR  20
   10 KTRAN=2                                                           PRESCR  21
C                                                                       PRESCR  22
      LVEC=II5*KMAX                                                     PRESCR  23
      IF (LVEC.NE.IBAR2*JBAR2*KBAR2) GO TO 760                          PRESCR  24
      EPSIP=2.0*EPSI                                                    PRESCR  25
      EPSIM=0.5*EPSI                                                    PRESCR  26
      DO 20 K=1,KMAX                                                    PRESCR  27
      DO 20 J=1,JMAX                                                    PRESCR  28
```

```
      DO 20 I=1,IMAX
      CALL IJKONLY
      LIJK=II5*(K-1)+IMAX*(J-1)+I
      VLM(LIJK)=AC(IJK)*DELX(I)*DELZ(K)*DELY(J)/RRI(I)
   20 CONTINUE
C
      IF (CYCLE.GT.1) GO TO 200
      IF (GZ.EQ.0.0) GO TO 40
      DO 30 J=2,JM1
      DO 30 I=2,IM1
      PSADD=0.0
      DO 30 KK=2,KM1
      K=KM1-KK+2
      CALL IJKONLY
      IF (PS(IJK).NE.0.0) PSADD=PS(IJK)-P(IJK)
      P(IJK)=P(IJK)+PSADD
   30 CONTINUE
C
   40 DO 190 K=1,KMAX
      DO 190 J=1,JMAX
      DO 190 I=1,IMAX
      CALL IJKAJCT
      LIJK=II5*(K-1)+IMAX*(J-1)+I
      VLM(LIJK)=AC(IJK)*DELX(I)*DELZ(K)*DELY(J)/RRI(I)
      IF (AC(IJK).LT.EM6.OR.BETA(IJK).LE.0.0) GO TO 170
      NFF=NF(IJK)
      IF (NFF.EQ.0) GO TO 180
      IF (NFF.GT.7) GO TO 140
      L=I
      M=J
      N=K
      GO TO (50,60,70,80,90,100,110), NFF
   50 L=I-1
      GO TO 120
   60 L=I+1
      GO TO 120
   70 M=J-1
      GO TO 120
   80 M=J+1
      GO TO 120
   90 N=K-1
      GO TO 120
  100 N=K+1
      GO TO 120
  110 GO TO 130
  120 LMN=NQ*(II5*(N-1)+IMAX*(M-1)+(L-1))+1
      NFEL=NF(IMJK)
      NFER=NF(IPJK)
      NFEF=NF(IJMK)
      NFEBK=NF(IJPK)
      NFEB=NF(IJKM)
      NFET=NF(IJKP)
      NFE=MAX0(NFEL,NFER,NFEBK,NFEF,NFEB,NFET,1)
      PSURF=PS(IJK)+PR(NFE)
      PLMN=P(LMN)
      IF (NF(LMN).NE.0.AND.BETA(IJK).GT.0.0) PLMN=PSURF
      P(IJK)=(1.0-PETA(IJK))*PLMN+PETA(IJK)*PSURF
      GO TO 180
  130 IF (PSAT.LE.0.0) GO TO 160
      IF (F(IJK).LT.EMF1) GO TO 150
      PMPS=0.0
      IF (PN(IJK).LT.PSAT) PMPS=P(IJK)-PSAT
      DIJK=(RRI(I)*(RDX(I)*(U(IJK)*AR(IJK)/RR(I)-U(IMJK)*AR(IMJK)/RR(I-1
     1 ))+RDY(J)*(V(IJK)*ABK(IJK)-V(IJMK)*ABK(IJMK)))+RDZ(K)*(W(IJK)*AT
     2 (IJK)-W(IJKM)*AT(IJKM)))/AC(IJK)
      DIJK=DIJK-PMPS**2/DELT
      P(IJK)=PN(IJK)-DIJK*BETA(IJK)/(1.0-2.0*PMPS*BETA(IJK)/DELT)
      GO TO 180
  140 P(IJK)=PR(NFF)
      GO TO 180
  150 P(IJK)=PSAT
      GO TO 180
  160 P(IJK)=PN(IJK)
      GO TO 180
  170 P(IJK)=0.0
  180 PN(IJK)=P(IJK)
  190 CONTINUE
C
      CALL BC
C
```

```
  200 IF (CYCLE.LT.1) RETURN
C
      C1=1.0
      DD=1.0
      DO 360 K=1,KMAX
      DO 360 J=1,JMAX
      DO 360 I=1,IMAX
      CALL IJKAJCT
      LIJK=II5*(K-1)+IMAX*(J-1)+I
      IF (AC(IJK).LT.EM6.OR.BETA(IJK).LE.0.0) GO TO 340
      IF (NF(IJK).GT.0) GO TO 350
      DIJK=(RRI(I)*(RDX(I)*(U(IJK)*AR(IJK)/RR(I)-U(IMJK)*AR(IMJK)/RR(I-1
     1 ))+RDY(J)*(V(IJK)*ABK(IJK)-V(IJMK)*ABK(IJMK)))+RDZ(K)*(W(IJK)*AT
     2 (IJK)-W(IJKM)*AT(IJKM)))/AC(IJK)
C
      IF (IDEFM.EQ.0.OR.F(IJK).GE.EMF1) GO TO 210
C
      DIJK=DIJK+AMIN1(ADEFM*EPSI,BDEFM*(1.0-F(IJK))/DELT)
C
  210 DIV(LIJK)=DIJK*VLM(LIJK)
C
      SSM(LIJK)=0.0
      RXR(LIJK)=0.0
      RXRA=0.0
      RXRB=0.0
      IF (AR(IJK).LT.EM6) GO TO 230
      RXR(LIJK)=DELZ(K)*DELY(J)*RDXP(I)*AR(IJK)/RR(I)
      SSM(LIJK)=SSM(LIJK)+DELT*RXR(LIJK)
      IF (NF(IPJK).EQ.0) GO TO 230
      IF (NF(IPJK).GT.7) GO TO 220
      IF (NF(IPJK).NE.1) GO TO 220
      RXRA=(1.0-PETA(IPJK))*RXR(LIJK)
      RXRB=((1.0-PETA(IPJK))*PN(IJK)+PETA(IPJK)*(PS(IPJK)+PSAT)-PN(IPJK)
     1 )*RXR(LIJK)
      GO TO 220
  220 RXR(LIJK)=0.0
  230 RXL(LIJK)=0.0
      RXLA=0.0
      RXLB=0.0
      IF (AR(IMJK).LT.EM6) GO TO 250
      RXL(LIJK)=DELZ(K)*DELY(J)*RDXP(I-1)*AR(IMJK)/RR(I-1)
      SSM(LIJK)=SSM(LIJK)+DELT*RXL(LIJK)
      IF (NF(IMJK).EQ.0) GO TO 250
      IF (NF(IMJK).GT.7) GO TO 240
      IF (NF(IMJK).NE.2) GO TO 240
      RXLA=(1.0-PETA(IMJK))*RXL(LIJK)
      RXLB=((1.0-PETA(IMJK))*PN(IJK)+PETA(IMJK)*(PS(IMJK)+PSAT)-PN(IMJK)
     1 )*RXL(LIJK)
      GO TO 240
  240 RXL(LIJK)=0.0
  250 RZT(LIJK)=0.0
      RZTA=0.0
      RZTB=0.0
      IF (AT(IJK).LT.EM6) GO TO 270
      RZT(LIJK)=DELX(I)*DELY(J)*RRI(I)*RDZP(K)*AT(IJK)
      SSM(LIJK)=SSM(LIJK)+DELT*RZT(LIJK)
      IF (NF(IJKP).EQ.0) GO TO 270
      IF (NF(IJKP).GT.7) GO TO 260
      IF (NF(IJKP).NE.5) GO TO 260
      RZTA=(1.0-PETA(IJKP))*RZT(LIJK)
      RZTB=((1.0-PETA(IJKP))*PN(IJK)+PETA(IJKP)*(PS(IJKP)+PSAT)-PN(IJKP)
     1 )*RZT(LIJK)
      GO TO 260
  260 RZT(LIJK)=0.0
  270 RZB(LIJK)=0.0
      RZBA=0.0
      RZBB=0.0
      IF (AT(IJKM).LT.EM6) GO TO 290
      RZB(LIJK)=DELX(I)*DELY(J)/RRI(I)*RDZP(K-1)*AT(IJKM)
      SSM(LIJK)=SSM(LIJK)+DELT*RZB(LIJK)
      IF (NF(IJKM).EQ.0) GO TO 290
      IF (NF(IJKM).GT.7) GO TO 280
      IF (NF(IJKM).NE.6) GO TO 280
      RZBA=(1.0-PETA(IJKM))*RZB(LIJK)
      RZBB=((1.0-PETA(IJKM))*PN(IJK)+PETA(IJKM)*(PS(IJKM)+PSAT)-PN(IJKM)
     1 )*RZB(LIJK)
      GO TO 280
  280 RZB(LIJK)=0.0
  290 RYBK(LIJK)=0.0
      RYBKA=0.0
```

```
      RYBKB=0.0
      IF (ABK(IJK).LT.EM6) GO TO 310
      RYBK(LIJK)=DELX(I)*DELZ(K)*RDYP(J)*RRI(I)*ABK(IJK)
      SSM(LIJK)=SSM(LIJK)+DELT*RYBK(LIJK)
      IF (NF(IJPK).EQ.0) GO TO 310
      IF (NF(IJPK).GT.7) GO TO 300
      IF (NF(IJPK).NE.3) GO TO 300
      RYBKA=(1.0-PETA(IJPK))*RYBK(LIJK)
      RYBKB=((1.0-PETA(IJPK))*PN(IJK)+PETA(IJPK)*(PS(IJPK)+PSAT)-PN(IJPK
     1 ))*RYBK(LIJK)
      GO TO 300
  300 RYBK(LIJK)=0.0
  310 RYF(LIJK)=0.0
      RYFA=0.0
      RYFB=0.0
      IF (ABK(IJMK).LT.EM6) GO TO 330
      RYF(LIJK)=DELX(I)*DELZ(K)*RDYP(J-1)*RRI(I)*ABK(IJMK)
      SSM(LIJK)=SSM(LIJK)+DELT*RYF(LIJK)
      IF (NF(IJMK).EQ.0) GO TO 330
      IF (NF(IJMK).GT.7) GO TO 320
      IF (NF(IJMK).NE.4) GO TO 320
      RYFA=(1.0-PETA(IJMK))*RYF(LIJK)
      RYFB=((1.0-PETA(IJMK))*PN(IJK)+PETA(IJMK)*(PS(IJMK)+PSAT)-PN(IJMK)
     1 )*RYF(LIJK)
      GO TO 320
  320 RYF(LIJK)=0.0
  330 SSM(LIJK)=SSM(LIJK)-DELT*(RXRA+RXLA+RZTA+RZBA+RYBKA+RYFA)
      DIV(LIJK)=(DIV(LIJK)-DELT*(RXRB+RXLB+RZTB+RZBB+RYBKB+RYFB))/SSM
     1 (LIJK)
      FF(LIJK)=0.0
      AF(LIJK)=0.0
      QQ(LIJK)=0.0
      AD(LIJK)=0.0
      DP(LIJK)=0.0
      GO TO 360
  340 VLM(LIJK)=1.0
  350 DIV(LIJK)=0.0
      FF(LIJK)=0.0
      AF(LIJK)=0.0
      AD(LIJK)=0.0
      QQ(LIJK)=0.0
      SSM(LIJK)=1.0
      RXR(LIJK)=0.0
      RXL(LIJK)=0.0
      RZT(LIJK)=0.0
      RZB(LIJK)=0.0
      RYBK(LIJK)=0.0
      RYF(LIJK)=0.0
      DP(LIJK)=0.0
  360 CONTINUE
C
      IF (WBK.NE.4) GO TO 380
      DO 370 K=1,KMAX
      DO 370 I=1,IMAX
      LIJKF=II5*(K-1)+I
      LIJKFP=LIJKF+IMAX
      LIJKBK=LIJKFP+IMAX*JBAR
      LIJKBKM=LIJKBK-IMAX
      SSM(LIJKF)=SSM(LIJKBKM)
      VLM(LIJKF)=VLM(LIJKBKM)
      DIV(LIJKF)=DIV(LIJKBKM)
      RXR(LIJKF)=RXR(LIJKBKM)
      RXL(LIJKF)=RXL(LIJKBKM)
      RZT(LIJKF)=RZT(LIJKBKM)
      RZB(LIJKF)=RZB(LIJKBKM)
      RYBK(LIJKF)=RYBK(LIJKBKM)
      RYF(LIJKF)=RYF(LIJKBKM)
      SSM(LIJKBK)=SSM(LIJKFP)
      VLM(LIJKBK)=VLM(LIJKFP)
      DIV(LIJKBK)=DIV(LIJKFP)
      RXR(LIJKBK)=RXR(LIJKFP)
      RXL(LIJKBK)=RXL(LIJKFP)
      RZT(LIJKBK)=RZT(LIJKFP)
      RZB(LIJKBK)=RZB(LIJKFP)
      RYBK(LIJKBK)=RYBK(LIJKFP)
      RYF(LIJKBK)=RYF(LIJKFP)
  370 CONTINUE
C
C  **********
  380 CONTINUE
```

```
C   **********
  390 A=C1/DD
      CALL SAXPY (LVEC,A,FF,1,DP,1)
      CALL SAXPY (LVEC,-A,QQ,1,DIV,1)
C
      DO 400 K=1,KMAX
      KKK=II5*(K-1)
      DO 400 J=1,JMAX
      JJJ=KKK+IMAX*(J-1)
      DO 400 I=1,IMAX
C
C     THE FOLLOWING COMMENTED STATEMENTS REPLACED IN ORDER TO VECTORIZE
C     IJK=NQ*(II5*(K-1)+IMAX*(J-1)+(I-1))+1
C     LIJK=II5*(K-1)+IMAX*(J-1)+I
C     CQ(LIJK)=DIV(LIJK)*SSM(LIJK)/VLM(LIJK)
C     IF((NF(IJK).NE.O).OR.(BETA(IJK).LT.O.O))CQ(LIJK)=0.0
C
      CQ(JJJ+I)=DIV(JJJ+I)*SSM(JJJ+I)/VLM(JJJ+I)
      IF ((NF(NQ*(JJJ+I-1)+1).NE.O).OR.(BETA(NQ*(JJJ+I-1)+1).LT.O.O)) CQ
     1 (JJJ+I)=0.0
  400 CONTINUE
C
      IJMAX=ISAMAX(LVEC,CQ,1)
      DMAX=ABS(CQ(IJMAX))
      IF (DMAX.LE.EPSI) GO TO 450
C
      DO 410 K=1,KMAX
      KKK=II5*(K-1)
      DO 410 J=1,JMAX
      JJJ=KKK+IMAX*(J-1)
      DO 410 I=1,IMAX
C
C     THE FOLLOWING COMMENTED STATEMENTS REPLACED IN ORDER TO VECTORIZE
C     IJK=NQ*(II5*(K-1)+IMAX*(J-1)+(I-1))+1
C     LIJK=II5*(K-1)+IMAX*(J-1)+I
C     LIPJK=LIJK+1
C     LIMJK=LIJK-1
C     LIJPK=LIJK+IMAX
C     LIJMK=LIJK-IMAX
C     LIJKP=LIJK+II5
C     LIJKM=LIJK-II5
C     AD(LIJK)=DELT*(RXR(LIJK)*DIV(LIPJK)+RXL(LIJK)*DIV(LIMJK)+RZT(LIJK)
C    1 *DIV(LIJKP)+RZB(LIJK)*DIV(LIJKM)+RYBK(LIJK)*DIV(LIJPK)+RYF(LIJK)*
C    2 DIV(LIJMK))-SSM(LIJK)*DIV(LIJK)
C     IF((NF(IJK).NE.O).OR.(BETA(IJK).LT.O.O))AD(LIJK)=0.0
C
      AD(JJJ+I)=DELT*(RXR(JJJ+I)*DIV(JJJ+I+1)+RXL(JJJ+I)*DIV(JJJ+I-1)
     1 +RZT(JJJ+I)*DIV(JJJ+I+II5)+RZB(JJJ+I)*DIV(JJJ+I-II5)+RYBK(JJJ+I)
     2 *DIV(JJJ+I+IMAX)+RYF(JJJ+I)*DIV(JJJ+I-IMAX))-SSM(JJJ+I)*DIV(JJJ+I
     3 )
      IF ((NF(NQ*(JJJ+I-1)+1).NE.O).OR.(BETA(NQ*(JJJ+I-1)+1).LT.O.O)) AD
     1 (JJJ+I)=0.0
  410 CONTINUE
C
      IF (WBK.NE.4) GO TO 430
      DO 420 K=1,KMAX
      KKK=II5*(K-1)
CDIR$ IVDEP
      DO 420 I=1,IMAX
C
C     THE FOLLOWING COMMENTED STATEMENTS REPLACED IN ORDER TO VECTORIZE
C     LIJKF=II5*(K-1)+I
C     LIJKFP=LIJKF+IMAX
C     LIJKBK=LIJKFP+IMAX*JBAR
C     LIJKBKM=LIJKBK-IMAX
C     AD(LIJKF)=AD(LIJKBKM)
C     AD(LIJKBK)=AD(LIJKFP)
C
      AD(KKK+I)=AD(KKK+II6+I)
      AD(KKK+II7+I)=AD(KKK+IMAX+I)
  420 CONTINUE
C
  430 C2=SDOT(LVEC,DIV,1,AD,1)
      B=C2/C1
      CALL SSCAL (LVEC,B,FF,1)
      CALL SAXPY (LVEC,1.0,DIV,1,FF,1)
C
      CALL SSCAL (LVEC,B,AF,1)
      CALL SAXPY (LVEC,1.0,AD,1,AF,1)
C
```

```
      DO 440 K=1,KMAX
      DO 440 J=1,JMAX
      DO 440 I=1,IMAX
      LIJK=II5*(K-1)+IMAX*(J-1)+I
      QQ(LIJK)=AF(LIJK)/SSM(LIJK)
  440 CONTINUE
C
      DD=SDOT(LVEC,QQ,1,AF,1)
      C1=C2
      ITER=ITER+1
      IF (ITER.GT.ITMAX) GO TO 750
      GO TO 390
C
  450 IF (ITER.GT.ITMOST) EPSI=1.05*EPSI
      IF (ITER.LT.ITMIN) EPSI=0.95*EPSI
      EPSI=AMIN1(EPSI,EPSIP)
      EPSI=AMAX1(EPSI,EPSIM)
C
C  **********
C
C +++ UNFOLD PRESSURE FOR INTERIOR FLUID CELLS
C
      DO 460 K=1,KMAX
      KKK=II5*(K-1)
      DO 460 J=1,JMAX
      JJJ=KKK+IMAX*(J-1)
      DO 460 I=1,IMAX
      D(NQ*(JJJ+I-1)+1)=CQ(JJJ+I)
      IF ((NF(NQ*(JJJ+I-1)+1).LE.0).AND.(AC(NQ*(JJJ+I-1)+1).GE.EM6).AND.
     1 (BETA(NQ*(JJJ+I-1)+1).GE.0.0)) P(NQ*(JJJ+I-1)+1)=PN(NQ*(JJJ+I-1)+
     2 1)+DP(JJJ+I)
  460 CONTINUE
C
C +++ CALCULATE PRESSURE CHANGES FOR SURFACE CELLS
C
      DO 610 K=1,KMAX
      DO 610 J=1,JMAX
      DO 610 I=1,IMAX
      CALL IJKAJCT
      LIJK=II5*(K-1)+IMAX*(J-1)+I
      IF (AC(IJK).LT.EM6.OR.BETA(IJK).LT.0.0) GO TO 590
      NFF=NF(IJK)
      IF (NFF.EQ.0) GO TO 600
      IF (NFF.GT.7) GO TO 560
      L=I
      M=J
      N=K
      GO TO (470,480,490,500,510,520,530), NFF
  470 L=I-1
      GO TO 540
  480 L=I+1
      GO TO 540
  490 M=J-1
      GO TO 540
  500 M=J+1
      GO TO 540
  510 N=K-1
      GO TO 540
  520 N=K+1
      GO TO 540
  530 GO TO 550
  540 LMN=NQ*(II5*(N-1)+IMAX*(M-1)+(L-1))+1
      NFEL=NF(IMJK)
      NFER=NF(IPJK)
      NFEF=NF(IJMK)
      NFEBK=NF(IJPK)
      NFEB=NF(IJKM)
      NFET=NF(IJKP)
      NFE=MAX0(NFEL,NFER,NFEBK,NFEF,NFEB,NFET,1)
      PSURF=PS(IJK)+PR(NFE)
      PLMN=P(LMN)
      IF (NF(LMN).NE.0.AND.BETA(IJK).GT.0.0) PLMN=PSURF
      DP(LIJK)=(1.0-PETA(IJK))*PLMN+PETA(IJK)*PSURF-PN(IJK)
      GO TO 600
  550 IF (PSAT.LE.0.0) GO TO 580
      IF (F(IJK).LT.EMF1) GO TO 570
      PMPS=0.0
      IF (PN(IJK).LT.PSAT) PMPS=P(IJK)-PSAT
      DIJK=(RRI(I)*(RDX(I)*(U(IJK)*AR(IJK)/RR(I)-U(IMJK)*AR(IMJK)/RR(I-1
     1 ))+RDY(J)*(V(IJK)*ABK(IJK)-V(IJMK)*ABK(IJMK)))+RDZ(K)*(W(IJK)*AT
```

```
     2 (IJK)-W(IJKM)*AT(IJKM)))/AC(IJK)
      DIJK=DIJK-PMPS**2/DELT
      DP(LIJK)=-DIJK*BETA(IJK)/(1.0-2.0*PMPS*BETA(IJK)/DELT)
      GO TO 600
  560 DP(LIJK)=PR(NFF)-PN(IJK)
      GO TO 600
  570 DP(LIJK)=PSAT-PN(IJK)
      GO TO 600
  580 DP(LIJK)=0.0
      GO TO 600
  590 DP(LIJK)=0.0
  600 CONTINUE
  610 CONTINUE
C
C      CALC NEW VELOCITIES CONSISTENT WITH SURFACE PRESSURE ASSUMPTIONS
C     NOTE: YIELDS DIV=0.0 FOR INTERIOR CELLS ADJACENT TO SURFACE CELLS
C
      DO 730 K=1,KM1
      DO 730 J=1,JM1
      DO 730 I=1,IM1
      CALL IJKAJCT
      LIJK=II5*(K-1)+IMAX*(J-1)+I
      LIPJK=LIJK+1
      LIJPK=LIJK+IMAX
      LIJKP=LIJK+II5
      IF (AC(IJK).LT.EM6.OR.BETA(IJK).LT.0.0) GO TO 730
      IF (AR(IJK).LT.EM6) GO TO 650
      IF (NF(IJK).EQ.0.AND.NF(IPJK).EQ.0) GO TO 620
      IF (NF(IJK).EQ.0.AND.NF(IPJK).NE.1) GO TO 630
      IF (NF(IJK).NE.2.AND.NF(IPJK).EQ.0) GO TO 640
  620 U(IJK)=U(IJK)+DELT*(1.0*(DP(LIJK)-DP(LIPJK)))*RDXP(I)
      GO TO 650
  630 U(IJK)=U(IJK)+DELT*DP(LIJK)*RDXP(I)
      GO TO 650
  640 U(IJK)=U(IJK)+DELT*(-DP(LIPJK))*RDXP(I)
      GO TO 650
  650 IF (ABK(IJK).LT.EM6) GO TO 690
      IF (NF(IJK).EQ.0.AND.NF(IJPK).EQ.0) GO TO 660
      IF (NF(IJK).EQ.0.AND.NF(IJPK).NE.3) GO TO 670
      IF (NF(IJK).NE.4.AND.NF(IJPK).EQ.0) GO TO 680
  660 V(IJK)=V(IJK)+DELT*(1.0*(DP(LIJK)-DP(LIJPK)))*RDYP(J)*RRI(I)
      GO TO 690
  670 V(IJK)=V(IJK)+DELT*DP(LIJK)*RDYP(J)*RRI(I)
      GO TO 690
  680 V(IJK)=V(IJK)+DELT*(-DP(LIJPK))*RDYP(J)*RRI(I)
  690 IF (AT(IJK).LT.EM6) GO TO 730
      IF (NF(IJK).EQ.0.AND.NF(IJKP).EQ.0) GO TO 700
      IF (NF(IJK).EQ.0.AND.NF(IJKP).NE.5) GO TO 710
      IF (NF(IJK).NE.6.AND.NF(IJKP).EQ.0) GO TO 720
  700 W(IJK)=W(IJK)+DELT*(1.0*(DP(LIJK)-DP(LIJKP)))*RDZP(K)
      GO TO 730
  710 W(IJK)=W(IJK)+DELT*DP(LIJK)*RDZP(K)
      GO TO 730
  720 W(IJK)=W(IJK)+DELT*(-DP(LIJKP))*RDZP(K)
  730 CONTINUE
C
C +++ PUT IN PRESSURES FOR SURFACE CELLS, ISOLATED CELLS, VOIDS
C
      DO 740 K=1,KMAX
      KKK=II5*(K-1)
      DO 740 J=1,JMAX
      JJJ=KKK+IMAX*(J-1)
      DO 740 I=1,IMAX
      IF ((NF(NQ*(JJJ+I-1)+1).EQ.0).OR.(AC(NQ*(JJJ+I-1)+1).LT.EM6).OR.
     1 (BETA(NQ*(JJJ+I-1)+1).LT.0.0)) GO TO 740
      P(NQ*(JJJ+I-1)+1)=PN(NQ*(JJJ+I-1)+1)+DP(JJJ+I)
  740 CONTINUE
C
      RETURN
C
  750 WRITE (59,770) CYCLE
      WRITE (12,770) CYCLE
      WRITE (9,770) CYCLE
      CALL EXIT
C
  760 WRITE (59,780) CYCLE
      WRITE (12,780) CYCLE
      WRITE (9,780) CYCLE
      CALL EXIT
C
```

```
  770 FORMAT (1X,29HTOO MANY CR ITERATIONS, CYCLE,I7)
  780 FORMAT (1X,44HMESH DIMENSIONS AND STORAGE NOT EQUAL, CYCLE,I7)
      END
*DK PRESCRC
*DK PRESSIT
      SUBROUTINE PRESSIT
*CA SLCOM1
      DATA ITMAX /1000/
C
C * * PRESSURE ITERATION
C
   10 IF (IORDER.EQ.2) ALPHA=1.0
      IF (FLG.EQ.0.0) GO TO 180
      ITER=ITER+1
      IF (ITER.LT.ITMAX) GO TO 20
      FNOC=1.0
      NOCON=NOCON+1
      CALL LPRT2
      GO TO 180
   20 FLG=0.0
C
C * * COMPUTE UPDATED CELL PRESSURE AND VELOCITIES
C
      DO 170 K=2,KM1
      DO 170 J=2,JM1
      DO 170 I=2,IM1
      CALL CALCIJK
      IF (BETA(IJK).LT.0.0) GO TO 170
      IF (F(IJK).LT.EMF) GO TO 170
      IF (NF(IJK).EQ.0) GO TO 100
C
C * * CALCULATE PRESSURE CHANGE FOR SURFACE CELLS
C
      NFF=NF(IJK)
      L=I
      M=J
      N=K
      GO TO (30,40,50,60,70,80,170), NFF
   30 L=I-1
      GO TO 90
   40 L=I+1
      GO TO 90
   50 M=J-1
      GO TO 90
   60 M=J+1
      GO TO 90
   70 N=K-1
      GO TO 90
   80 N=K+1
   90 CONTINUE
      LMN=NQ*(II5*(N-1)+IMAX*(M-1)+(L-1))+1
      NFER=NF(IPJK)
      NFEL=NF(IMJK)
      NFEBK=NF(IJPK)
      NFEF=NF(IJMK)
      NFEB=NF(IJKM)
      NFET=NF(IJKP)
      NFE=MAX0(NFEL,NFER,NFEBK,NFEF,NFEB,NFET,1)
      PSURF=PR(NFE)+PS(IJK)
      PLMN=P(LMN)
      IF (NF(LMN).NE.0.AND.BETA(IJK).GT.0.0) PLMN=PSURF
      DELP=(1.0-PETA(IJK))*PLMN+PETA(IJK)*PSURF-P(IJK)
      GO TO 110
  100 CONTINUE
C
C * * CALCULATE PRESSURE CHANGE FOR FULL CELLS
C     AND TEST FOR CONVERGENCE
C
      D(IJK)=(RRI(I)*(RDX(I)*(U(IJK)*AR(IJK)/RR(I)-U(IMJK)*AR(IMJK)/RR(I
     1 -1))+RDY(J)*(V(IJK)*ABK(IJK)-V(IJMK)*ABK(IJMK)))+RDZ(K)*(W(IJK)
     2 *AT(IJK)-W(IJKM)*AT(IJKM)))/AC(IJK)
      DIJK=D(IJK)
      IF (ABS(DIJK).GE.EPSI) FLG=1.0
      DELP=-BETA(IJK)*DIJK*PETA(IJK)
C
C * * UPDATE CELL PRESSURE AND VELOCITY
C
  110 DTDELP=DELT*DELP
      P(IJK)=P(IJK)+DELP
      IF (BETA(IPJK).LT.0.0.OR.BETA(IPJK).EQ.1.0) GO TO 120
```

```
      U(IJK)=U(IJK)+DTDELP*RDXP(I)
  120 IF (BETA(IMJK).LT.0.0.OR.BETA(IMJK).EQ.1.0) GO TO 130
      U(IMJK)=U(IMJK)-DTDELP*RDXP(I-1)
  130 IF (BETA(IJPK).LT.0.0.OR.BETA(IJPK).EQ.1.0) GO TO 140
      V(IJK)=V(IJK)+DTDELP*RDYP(J)*RRI(I)
  140 IF (BETA(IJMK).LT.0.0.OR.BETA(IJMK).EQ.1.0) GO TO 150
      V(IJMK)=V(IJMK)-DTDELP*RDYP(J-1)*RRI(I)
  150 IF (BETA(IJKP).LT.0.0.OR.BETA(IJKP).EQ.1.0) GO TO 160
      W(IJK)=W(IJK)+DTDELP*RDZP(K)
  160 IF (BETA(IJKM).LT.0.0.OR.BETA(IJKM).EQ.1.0) GO TO 170
      W(IJKM)=W(IJKM)-DTDELP*RDZP(K-1)
  170 CONTINUE
C
C * * IMPOSE BOUNDARY CONDITIONS
C
      CALL BC
C
      GO TO 10
  180 CONTINUE
C
      RETURN
      END
*DK RCONTUR
      SUBROUTINE RCONTUR
*CA SLCOM1
C
C * * CONTOUR SET-UP
C
C * * THIS SET UP IS FOR CELL CENTERED QUANTITIES
C
      NAMELIST /CONTUR/ NNX,NNY,NNZ,NZX,NZY,NZZ,NC,ZMN,ZMX,DLZ,ZCQ,DMPX
     1 ,DMPY,DMPZ,IGRD,ITITLE1,NTITLE1,ITITLE2,NTITLE2,XLABLE,YLABLE
     2 ,ZLABLE,NXLBL,NYLBL,NZLBL
C
C * * DEFAULT VALUES FOR CONTUR
C
      NNX=-IBAR
      NNY=-JBAR
      NNZ=-KBAR
      NZX=50
      NZY=50
      NZZ=50
      NC=10
      ZMN=-1.0
      ZMX=-1.0
      DLZ=0.0
      ZCQ(1)=1.0
      DMPX=X(IM1)
      DMPY=Y(JM1)
      DMPZ=Z(KM1)
      IGRD=0
      ITITLE1(1)=8HPRESSURE
      ITITLE1(2)=8HCONTOURS
      NTITLE1=16
      ITITLE2(1)=8HVORTICIT
      ITITLE2(2)=8HY CONTOU
      ITITLE2(3)=8HRS
      NTITLE2=18
      XLABLE=6HX-AXIS
      YLABLE=6HY-AXIS
      ZLABLE=6HZ-AXIS
      NXLBL=6
      NYLBL=6
      NZLBL=6
C
C * * READ VALUES FOR CONTOUR FROM NAMELIST INPUT / CONTUR /
C
      READ (10,CONTUR)
C
C * * WRITE CONTOUR INPUT VALUES ONTO TAPE 9 AND TAPE 12.
C
C * * WRITE EFFECTIVE DATA FROM "CONTUR" TO TAPES 9 AND 12.
C
      IF (LPR.EQ.0) GO TO 30
C
      WRITE (9,40)
      WRITE (12,40)
      WRITE (9,300)
      WRITE (12,300)
C
```

```
      KPR=9
      ASSIGN 20 TO KRET
   10 WRITE (KPR,50) NNX
      WRITE (KPR,60) NNY
      WRITE (KPR,70) NNZ
      WRITE (KPR,80) NZX
      WRITE (KPR,90) NZY
      WRITE (KPR,100) NZZ
      WRITE (KPR,110) NC
      WRITE (KPR,120) ZMN
      WRITE (KPR,130) ZMX
      WRITE (KPR,140) DLZ
      WRITE (KPR,150) ZCQ
      WRITE (KPR,160) DMPX
      WRITE (KPR,170) DMPY
      WRITE (KPR,180) DMPZ
      WRITE (KPR,190) IGRD
      WRITE (KPR,200) (ITITLE1(L),L=1,3)
      WRITE (KPR,210) NTITLE1
      WRITE (KPR,220) (ITITLE2(L),L=1,3)
      WRITE (KPR,230) NTITLE2
      WRITE (KPR,240) XLABLE
      WRITE (KPR,250) YLABLE
      WRITE (KPR,260) ZLABLE
      WRITE (KPR,270) NXLBL
      WRITE (KPR,280) NYLBL
      WRITE (KPR,290) NZLBL
      GO TO KRET, (20,30)
   20 ASSIGN 30 TO KRET
      KPR=12
      GO TO 10
   30 CONTINUE
      RETURN
C
   40 FORMAT (1H1)
   50 FORMAT (10X,10H      NNX=   ,I3)
   60 FORMAT (10X,10H      NNY=   ,I3)
   70 FORMAT (10X,10H      NNZ=   ,I3)
   80 FORMAT (10X,10H      NZX=   ,I3)
   90 FORMAT (10X,10H      NZY=   ,I3)
  100 FORMAT (10X,10H      NZZ=   ,I3)
  110 FORMAT (10X,10H       NC=   ,I3)
  120 FORMAT (10X,10H      ZMN=   ,1PE12.5)
  130 FORMAT (10X,10H      ZMX=   ,1PE12.5)
  140 FORMAT (10X,10H      DLZ=   ,1PE12.5)
  150 FORMAT (10X,10H      ZCQ=   ,1PE12.5)
  160 FORMAT (10X,10H     DMPX=   ,1PE12.5)
  170 FORMAT (10X,10H     DMPY=   ,1PE12.5)
  180 FORMAT (10X,10H     DMPZ=   ,1PE12.5)
  190 FORMAT (10X,10H     IGRO=   ,I3)
  200 FORMAT (10X,10HITITLE1=   ,3A8)
  210 FORMAT (10X,10HNTITLE1=   ,I3)
  220 FORMAT (10X,10HITITLE2=   ,3A8)
  230 FORMAT (10X,10HNTITLE2=   ,I3)
  240 FORMAT (10X,10H XLABLE=   ,A8)
  250 FORMAT (10X,10H YLABLE=   ,A8)
  260 FORMAT (10X,10H ZLABLE=   ,A8)
  270 FORMAT (10X,10H  NXLBL=   ,I3)
  280 FORMAT (10X,10H  NYLBL=   ,I3)
  290 FORMAT (10X,10H  NXLBL=   ,I3)
  300 FORMAT (1HO,/,25X,26HCONTOUR PLOTTING SPECIFICS,/)
      END
*DK RDTAPE
      SUBROUTINE RDTAPE
*CA SLCOM1
C
C * * READ INPUT TAPE
C
      REWIND 7
      JTD=TD
      JNSC=LOCF(ZLAST)-LOCF(AA)+1
   10 READ (7) (AA(N),N=1,JNSC)
      READ (7) (BASC(N),N=1,NCR2)
      READ (7) (BASC1(N),N=1,NCR2)
      READ (7) (BASC2(N),N=1,NCR2)
      READ (7) (BASC3(N),N=1,NCR2)
      IF (JTD.NE.NUMTD) GO TO 10
      CALL CLOSE (7)
      CALL SECOND (STIM)
      TRL=STIM
```

```
      T1=TRL
      T2=T1
      CALL DATEH (DAT)
      CALL CLOCK1 (CLK)
      CALL GETJTL (TL)
      TLM=TL-150.0+(1.0-TLIMD)*1.0E+10
      WRITE (9,20) JTD
      WRITE (9,30) NAME,JNM,DAT,CLK
      WRITE (9,40) ITER,T,CYCLE
      WRITE (12,20) JTD
      WRITE (12,30) NAME,JNM,DAT,CLK
      WRITE (12,40) ITER,T,CYCLE
      WRITE (59,20) JTD
      WRITE (59,30) NAME,JNM,DAT,CLK
      WRITE (59,40) ITER,T,CYCLE
C
C * * CHANGES TO TAPE  DUMP INSERTED HERE
C
C      CHANGES TO ALLOW INPUT PARAMETER MODIFICATION ON RESTART
C
      CALL RINPUT
      CALL RGRAFIC
      RETURN
C
   20 FORMAT (22H0 RESTARTING FROM TD= ,I3)
   30 FORMAT (1H ,18X,10A8,1X,A8,2(1X,A8))
   40 FORMAT (6X,6HITER= ,I5,18X,6HTIME= ,1PE12.5,12X,7HCYCLE= ,I4)
      END
*DK RGRAFIC
      SUBROUTINE RGRAFIC
*CA SLCOM1
C
      NAMELIST /GRAFIC/ XEA,YEA,ZEA,XCA,YCA,ZCA,NVEWS,NVPLTS,IVVEW,IV1
     1 ,IV2,JV1,JV2,KV1,KV2,NAV,IPERV,NCPLTS,ICVEW,IC1,IC2,JC1,JC2,KC1
     2 ,KC2,NAC,IPERC,NSPLTS,ISVEW,IS1,IS2,JS1,JS2,KS1,KS2,NAS,IPERS
C*******
C******* DEFAULT VALUES FOR NAMELIST /GRAFIC/
C*******
C******* DEFAULT VALUES ARE FOR ONE VELOCITY  VECTOR VIEW.  WITH  FULL
C******* RANGE ON  I AND FULL RANGE ON J WITH K=10 CONSTANT.
C*******
      IF (TD.GE.1) GO TO 10
      XEA(1)=50.0
      YEA(1)=50.0
      ZEA(1)=50.0
      XCA(1)=0.05
      YCA(1)=0.05
      ZCA(1)=0.05
      NVEWS=1
      NVPLTS=1
      IV1(1)=2
      IV2(1)=99
      JV1(1)=2
      JV2(1)=99
      KV1(1)=10
      KV2(1)=10
      NCPLTS=0
      NSPLTS=0
C
C * * READ IN PLOT-OUTPUT SPECIFICS
C
   10 READ (10,GRAFIC)
C
      IPMX=IM1
      JPMX=JM1
      KPMX=KM1
      IF (NVPLTS.EQ.0) GO TO 30
      DO 20 N=1,NVPLTS
      IF (IV1(N).GT.IMAX) IV1(N)=IPMX
      IF (IV2(N).GT.IMAX) IV2(N)=IPMX
      IF (JV1(N).GT.JMAX) JV1(N)=JPMX
      IF (JV2(N).GT.JMAX) JV2(N)=JPMX
      IF (KV1(N).GT.KMAX) KV1(N)=KPMX
   20 IF (KV2(N).GT.KMAX) KV2(N)=KPMX
   30 IF (NCPLTS.EQ.0) GO TO 50
      DO 40 N=1,NCPLTS
      IF (IC1(N).GT.IMAX) IC1(N)=IPMX
      IF (IC2(N).GT.IMAX) IC2(N)=IPMX
      IF (JC1(N).GT.JMAX) JC1(N)=JPMX
      IF (JC2(N).GT.JMAX) JC2(N)=JPMX
```

```
      IF (KC1(N).GT.KMAX) KC1(N)=KPMX                                   RGRAFIC  53
   40 IF (KC2(N).GT.KMAX) KC2(N)=KPMX                                   RGRAFIC  54
   50 IF (NSPLTS.EQ.0) GO TO 70                                         RGRAFIC  55
      DO 60 N=1,NSPLTS                                                  RGRAFIC  56
      IF (IS1(N).GT.IMAX) IS1(N)=IPMX                                   RGRAFIC  57
      IF (IS2(N).GT.IMAX) IS2(N)=IPMX                                   RGRAFIC  58
      IF (JS1(N).GT.JMAX) JS1(N)=JPMX                                   RGRAFIC  59
      IF (JS2(N).GT.JMAX) JS2(N)=JPMX                                   RGRAFIC  60
      IF (KS1(N).GT.KMAX) KS1(N)=KPMX                                   RGRAFIC  61
   60 IF (KS2(N).GT.KMAX) KS2(N)=KPMX                                   RGRAFIC  62
   70 CONTINUE                                                          RGRAFIC  63
C                                                                       RGRAFIC  64
C * * PRINT/WRITE(4020) PLOTTING INPUT DATA                             RGRAFIC  65
C                                                                       RGRAFIC  66
      KPR=9                                                             RGRAFIC  67
      ASSIGN 90 TO KRET                                                 RGRAFIC  68
   80 WRITE (KPR,110)                                                   RGRAFIC  69
      WRITE (KPR,120) (XEA(N),YEA(N),ZEA(N),XCA(N),YCA(N),ZCA(N),N=1    RGRAFIC  70
     1 ,NVEWS)                                                          RGRAFIC  71
      WRITE (KPR,130) (IVVEW(N),IV1(N),IV2(N),JV1(N),JV2(N),KV1(N),KV2(N RGRAFIC  72
     1 ),NAV(N),IPERV(N),N=1,NVPLTS)                                    RGRAFIC  73
      WRITE (KPR,140) (ICVEW(N),IC1(N),IC2(N),JC1(N),JC2(N),KC1(N),KC2(N RGRAFIC  74
     1 ),NAC(N),IPERC(N),N=1,NCPLTS)                                    RGRAFIC  75
      WRITE (KPR,150) (ISVEW(N),IS1(N),IS2(N),JS1(N),JS2(N),KS1(N),KS2(N RGRAFIC  76
     1 ),NAS(N),IPERS(N),N=1,NSPLTS)                                    RGRAFIC  77
      GO TO KRET, (90,100)                                              RGRAFIC  78
   90 ASSIGN 100 TO KRET                                                RGRAFIC  79
      KPR=12                                                            RGRAFIC  80
      GO TO 80                                                          RGRAFIC  81
  100 CONTINUE                                                          RGRAFIC  82
      RETURN                                                            RGRAFIC  83
C                                                                       RGRAFIC  84
  110 FORMAT (1H0,30X,32H  PLOTTING OUTPUT SPECIFICATIONS//)            RGRAFIC  85
  120 FORMAT (38H0           EYE POINT-VIEW PLANE TABLE/(5H  XE=,1PE12.5 RGRAFIC  86
     1 ,5H  YE=,E12.5,5H  ZE=,E12.5,5H  XC=,E12.5,5H  YC=,E12.5,5H  ZC= RGRAFIC  87
     2 ,E12.5))                                                         RGRAFIC  88
  130 FORMAT (32H0           VELOCITY VECTOR PLOTS/(8H  IVVEW=,I3,6H  IV1 RGRAFIC  89
     1=,I3,6H  IV2=,I3,6H  JV1=,I3,6H  JV2=,I3,6H  KV1=,I3,6H  KV2=,I3,6 RGRAFIC  90
     2 H  NAV=,I3,8H  IPERV=,I3))                                       RGRAFIC  91
  140 FORMAT (24H0           CONTOUR PLOTS /(8H  ICVEW=,I3,6H  IC1=,I3,6H RGRAFIC  92
     1 IC2=,I3,6H  JC1=,I3,6H  JC2=,I3,6H  KC1=,I3,6H  KC2=,I3,6H  NAC= RGRAFIC  93
     2 ,I3,8H  IPERC=,I3))                                              RGRAFIC  94
  150 FORMAT (35H0           PERSPECTIVE SURFACE PLOTS/(8H  ISVEW=,I3,6H RGRAFIC  95
     1IS1=,I3,6H  IS2=,I3,6H  JS1=,I3,6H  JS2=,I3,6H  KS1=,I3,6H  KS2= RGRAFIC  96
     2 ,I3,6H  NAS=,I3,8H  IPERS=,I3))                                  RGRAFIC  97
      END                                                               RGRAFIC  98
*DK RINPUT                                                              RINPUT    1
      SUBROUTINE RINPUT                                                 RINPUT    2
*CA SLCOM1                                                              RINPUT    3
C                                                                       RINPUT    4
C * * READ PROBLEM INPUT DATA                                           RINPUT    5
C                                                                       RINPUT    6
      NAMELIST /XPUT/ ALPHA,PLTDT,PRTDT,CYL,DELT,EPSI,GX,GY,GZ,ICSURF   RINPUT    7
     1 ,IEQUIB,JNM,LPR,NAME,NFCAL,NU,OMG,TDDT,TD,TLIMD,AUTOT,SIGMA      RINPUT    8
     2 ,CANGLE,IDEFM,ISOR,ISURFT,T,RADPS,NOWALL,TWFIN,UI,VELMX,VI,WB,WBK RINPUT    9
     3 ,WF,WI,WL,WR,WT,FLHT,IORDER,ICLIP,IZOOM,RHOF                     RINPUT   10
C                                                                       RINPUT   11
      DATA KTRAN /1/                                                    RINPUT   12
C * * DEFAULT VALUES FOR NAMELIST / XPUT / .                            RINPUT   13
C                                                                       RINPUT   14
      GO TO (10,20), KTRAN                                              RINPUT   15
   10 CONTINUE                                                          RINPUT   16
      ALPHA=1.0                                                         RINPUT   17
      AUTOT=1.0                                                         RINPUT   18
      PLTDT=1.0                                                         RINPUT   19
      PRTDT=1.0                                                         RINPUT   20
      CYL=0.0                                                           RINPUT   21
      DELT=0.02                                                         RINPUT   22
      EPSI=0.001                                                        RINPUT   23
      GX=0.0                                                            RINPUT   24
      GY=0.0                                                            RINPUT   25
      GZ=0.0                                                            RINPUT   26
      ICLIP=0                                                           RINPUT   27
      ICSURF=0                                                          RINPUT   28
      IEQUIB=0                                                          RINPUT   29
      IORDER=1                                                          RINPUT   30
      IDEFM=0                                                           RINPUT   31
      ISOR=0                                                            RINPUT   32
      IZOOM=0                                                           RINPUT   33
      JNM=8H RUN 1                                                      RINPUT   34
```

```
      LPR=2
      NAME(1)=8HPROB. NO
      NAME(2)=8HNAME
      NFCAL=3
      NOWALL=1
      NU=0.0
      OMG=1.0
      FLHT=1.0
      RADPS=0.0
      RHOF=1.0
      SIGMA=0.0
      ISURFT=0
      CANGLE=0.0
      T=0.0
      TDDT=+10
      TD=-1.0
      TLIMD=1.0
      TWFIN=10.0
      UI=0.0
      VELMX=2.0
      VI=0.0
      WB=1
      WBK=1
      WF=1
      WI=0.0
      WL=1
      WR=1
      WT=1
   20 CONTINUE
C
C * * READ INPUT DATA FROM NAMELIST /XPUT /.
C
      READ (10,XPUT)
C
      IF (CANGLE.EQ.90.0) CANGLE=CANGLE-EM6
      CANGLE=CANGLE*0.0174532925
      TANCA=TAN(CANGLE)
      SANG=SIN(CANGLE)
      CSANG=COS(CANGLE)
C * * SET UP FILM (CGSCFT) PACKAGE
C
      GO TO (30,40), KTRAN
   30 KTRAN=2
      CALL GRPHCFT
      CALL GPLOT (1HU,14HT3MDT SOLA-3DV,14)
      CALL LIB4020
      CALL GRPHLUN (12,4HPLOT)
      CALL SETFLSH
   40 CONTINUE
C
C * * WRITE EFFECTIVE DATA FROM XPUT TO TAPES 9 AND 12.
C
      IF (LPR.EQ.0) GO TO 70
C
      KPR=9
      ASSIGN 60 TO KRET
   50 WRITE (KPR,80) (NAME(I),I=1,8)
      WRITE (KPR,90) ALPHA
      WRITE (KPR,430) AUTOT
      WRITE (KPR,480) CANGLE
      WRITE (KPR,120) CYL
      WRITE (KPR,130) DELT
      WRITE (KPR,140) EPSI
      WRITE (KPR,360) FLHT
      WRITE (KPR,150) GX
      WRITE (KPR,160) GY
      WRITE (KPR,170) GZ
      WRITE (KPR,380) ICLIP
      WRITE (KPR,390) ICSURF
      WRITE (KPR,490) IDEFM
      WRITE (KPR,400) IEQUIB
      WRITE (KPR,500) ISOR
      WRITE (KPR,370) IORDER
      WRITE (KPR,510) ISURFT
      WRITE (KPR,420) IZOOM
      WRITE (KPR,180) JNM
      WRITE (KPR,190) LPR
      WRITE (KPR,410) NFCAL
      WRITE (KPR,450) NOWALL
      WRITE (KPR,200) NU
```

```
      WRITE (KPR,210) OMG
      WRITE (KPR,100) PLTDT
      WRITE (KPR,110) PRTDT
      WRITE (KPR,440) RADPS
      WRITE (KPR,460) RHOF
      WRITE (KPR,470) SIGMA
      WRITE (KPR,520) T
      WRITE (KPR,220) TDDT
      WRITE (KPR,230) TD
      WRITE (KPR,240) TLIMD
      WRITE (KPR,250) TWFIN
      WRITE (KPR,260) UI
      WRITE (KPR,270) VELMX
      WRITE (KPR,280) VI
      WRITE (KPR,290) WB
      WRITE (KPR,300) WBK
      WRITE (KPR,310) WF
      WRITE (KPR,320) WI
      WRITE (KPR,330) WL
      WRITE (KPR,340) WR
      WRITE (KPR,350) WT
      GO TO KRET, (60,70)
   60 ASSIGN 70 TO KRET
      KPR=12
      GO TO 50
   70 CONTINUE
      RETURN
C
   80 FORMAT (1H1,10A8,//)
   90 FORMAT (10X,10H  ALPHA=  ,1PE12.5)
  100 FORMAT (10X,10H  PLTDT=  ,1PE12.5)
  110 FORMAT (10X,10H  PRTDT=  ,1PE12.5)
  120 FORMAT (10X,10H    CYL=  ,1PE12.5)
  130 FORMAT (10X,10H   DELT=  ,1PE12.5)
  140 FORMAT (10X,10H   EPSI=  ,1PE12.5)
  150 FORMAT (10X,10H     GX=  ,1PE12.5)
  160 FORMAT (10X,10H     GY=  ,1PE12.5)
  170 FORMAT (10X,10H     GZ=  ,1PE12.5)
  180 FORMAT (10X,10H    JNM=  ,A8)
  190 FORMAT (10X,10H    LPR=  ,I2)
  200 FORMAT (10X,10H     NU=  ,1PE12.5)
  210 FORMAT (10X,10H    OMG=  ,1PE12.5)
  220 FORMAT (10X,10H   TDDT=  ,1PE12.5)
  230 FORMAT (10X,10H     TD=  ,I4)
  240 FORMAT (10X,10H  TLIMD=  ,1PE12.5)
  250 FORMAT (10X,10H  TWFIN=  ,1PE12.5)
  260 FORMAT (10X,10H     UI=  ,1PE12.5)
  270 FORMAT (10X,10H  VELMX=  ,1PE12.5)
  280 FORMAT (10X,10H     VI=  ,1PE12.5)
  290 FORMAT (10X,10H     WB=  ,I2)
  300 FORMAT (10X,10H    WBK=  ,I2)
  310 FORMAT (10X,10H     WF=  ,I2)
  320 FORMAT (10X,10H     WI=  ,1PE12.5)
  330 FORMAT (10X,10H     WL=  ,I2)
  340 FORMAT (10X,10H     WR=  ,I2)
  350 FORMAT (10X,10H     WT=  ,I2)
  360 FORMAT (10X,10H   FLHT=  ,1PE12.5)
  370 FORMAT (10X,10H IORDER=  ,I2)
  380 FORMAT (10X,10H  ICLIP=  ,I2)
  390 FORMAT (10X,10H  ICSURF= ,I2)
  400 FORMAT (10X,10H  IEQUIB= ,I2)
  410 FORMAT (10X,10H  NFCAL=  ,I2)
  420 FORMAT (10X,10H  IZOOM=  ,I2)
  430 FORMAT (10X,10H  AUTOT=  ,1PE12.5)
  440 FORMAT (10X,10H  RADPS=  ,1PE12.5)
  450 FORMAT (10X,10H NOWALL=  ,I2)
  460 FORMAT (10X,10H   RHOF=  ,1PE12.5)
  470 FORMAT (10X,10H  SIGMA=  ,1PE12.5)
  480 FORMAT (10X,10H CANGLE=  ,1PE12.5)
  490 FORMAT (10X,10H  IDEFM=  ,I2)
  500 FORMAT (10X,10H   ISOR=  ,I2)
  510 FORMAT (10X,10H ISURFT=  ,I2)
  520 FORMAT (10X,10H      T=  ,1PE12.5)
      END
*DK SECORD
      SUBROUTINE SECORD
*CA SLCOM1
C
C * * SECOND ORDER OPTION
C
```

```
      DO 20 J=1,JMAX
      DO 20 I=1,IMAX
      KT1=KT1+1
      DO 10 K=1,KMAX
      CALL CALCIJK
      UDUM=UN(IJK)
      UN(IJK)=U(IJK)
      U(IJK)=UDUM
      VDUM=VN(IJK)
      VN(IJK)=V(IJK)
      V(IJK)=VDUM
      WDUM=WN(IJK)
      WN(IJK)=W(IJK)
      W(IJK)=WDUM
   10 CONTINUE
   20 CONTINUE
      ALPHA=-1.0
      AVE=0.5
C
      RETURN
      END
*DK SETACC
      SUBROUTINE SETACC
C
*CA SLCOM1
      IJ=I+(J-1)*IMAX
      GXA(IJ)=GX*CTHJ(J)+GY*STHJ(J)
      GYA(IJ)=GX*STHJBK(J)+GY*CTHJBK(J)
      RETURN
      END
*DK SETFS
      SUBROUTINE SETFS
C
*CA SLCOM1
      DIMENSION IFLG(5), DIS(4), XM(5), ZM(5)
C
      NAMELIST /FLUIDGN/ NQBS,QA3,QA2,QA1,QB3,QB2,QB1,QC3,QC2,QC1,QD3
     1 ,QD2,QD1,IQH
      DATA NQBS /0/
      DATA QA2, QA1, QB2, QB1, QC2, QC1, IQH /10*0.0,10*0.0,10*0.0,10*0.
     1 0,10*0.0,10*0.0,10*0/
      DATA QA3, QB3, QC3, QD3, QD2, QD1 /10*0.0,10*0.0,10*0.0,10*0.0,10*
     1 0.0,10*0.0/
C
C * * READ IN DATA FROM NAMELIST / FLUIDGN /
C
      READ (10,FLUIDGN)
C
C * * WRITE INPUT DATA FOR FLUID GENERATOR ONTO TAPES 9 AND 12.
C
      IF (NQBS.LE.0) GO TO 410
      IF (LPR.EQ.0) GO TO 20
      WRITE (9,450)
      WRITE (12,450)
      WRITE (9,420) NQBS
      WRITE (12,420) NQBS
      DO 10 I=1,NQBS
      WRITE (9,430) I,QA2(I),QA1(I),QB2(I),QB1(I),QC2(I),QC1(I),IQH(I)
      WRITE (9,440) I,QA3(I),QB3(I),QC3(I),QD3(I),QD2(I),QD1(I)
      WRITE (12,430) I,QA2(I),QA1(I),QB2(I),QB1(I),QC2(I),QC1(I),IQH(I)
      WRITE (12,440) I,QA3(I),QB3(I),QC3(I),QD3(I),QD2(I),QD1(I)
   10 CONTINUE
C
   20 IF (ICSURF.GT.1) GO TO 300
C
C +++  THIS PORTION FOR AXISYMMETRIC SURFACES
C +++ CONIC FCN=QA2*X*X+QA1*X+QB2*Z*Z+QB1*Z+QC2*X*Z+QC1
C +++ INSIDE FCN=NEGATIVE VALUE
C +++ IQH=1 ADD FLUID INSIDE FCN, IQH=0 SUBTRACT FLUID INSIDE FCN
C +++
      J=2
      DO 250 L=1,NQBS
      DO 240 K=2,KM1
      DO 240 I=2,IM1
      CALL IJKONLY
      RDXDZ=1.0/(DELX(I)*DELZ(K))
      DO 80 M=1,4
      GO TO (30,40,50,60), M
   30 X1=X(I)
      Z1=Z(K-1)
```

```
      DIS(1)=DELZ(K)
      GO TO 70
   40 Z1=Z(K)
      X1=X(I)
      DIS(2)=DELX(I)
      GO TO 70
   50 X1=X(I-1)
      Z1=Z(K)
      DIS(3)=DELZ(K)
      GO TO 70
   60 Z1=Z(K-1)
      X1=X(I-1)
      DIS(4)=DELX(I)
   70 IFLG(M)=0
      FCONIC=QA2(L)*X1*X1+QA1(L)*X1+QB2(L)*Z1*Z1+QB1(L)*Z1+QC2(L)*X1*Z1
     1 +QC1(L)
      IF (FCONIC.LE.0.0) IFLG(M)=1
      XM(M)=X1
      ZM(M)=Z1
   80 CONTINUE
      IFLG(5)=IFLG(1)
      XM(5)=XM(1)
      ZM(5)=ZM(1)
      IFLGS=0
      DO 90 M=1,4
   90 IFLGS=IFLGS+IFLG(M)
      BRIK=0.0
      BTIK=0.0
      IF (IFLGS.EQ.0) GO TO 240
      IF (IFLGS.LT.4) GO TO 100
      BIK=1.0
      BRIK=1.0
      BTIK=1.0
      GO TO 220
  100 IF (IFLG(1).EQ.1.AND.IFLG(2).EQ.1) BRIK=1.0
      IF (IFLG(2).EQ.1.AND.IFLG(3).EQ.1) BTIK=1.0
      DO 180 M=1,4
      IF (IFLG(M).EQ.IFLG(M+1)) GO TO 180
      X1=XM(M)
      Z1=ZM(M)
      X2=XM(M+1)
      Z2=ZM(M+1)
      IF (IFLG(M).EQ.0) GO TO 110
      X2=XM(M)
      Z2=ZM(M)
      X1=XM(M+1)
      Z1=ZM(M+1)
  110 EPSIF=0.001*(ABS(X2-X1)+ABS(Z2-Z1))
      SMN=0.0
      FMN=QA2(L)*X2*X2+QA1(L)*X2+QB2(L)*Z2*Z2+QB1(L)*Z2+QC2(L)*X2*Z2+QC1
     1 (L)
      SMX=1.0
      FMX=QA2(L)*X1*X1+QA1(L)*X1+QB2(L)*Z1*Z1+QB1(L)*Z1+QC2(L)*X1*Z1+QC1
     1 (L)
      S=0.5
  120 XT=S*X1+(1.0-S)*X2
      ZT=S*Z1+(1.0-S)*Z2
      FS=QA2(L)*XT*XT+QA1(L)*XT+QB2(L)*ZT*ZT+QB1(L)*ZT+QC2(L)*XT*ZT+QC1
     1 (L)
      IF (ABS(FS).LT.EPSIF) GO TO 150
      IF (FS.GE.0.0) GO TO 130
      FDEN=ABS(FS-FMN)+1.0E-10
      SE=S-FS*(S-SMN)/FDEN
      IF (SE.GT.SMX) SE=SMX
      FMN=FS
      SMN=S
      GO TO 140
  130 FDEN=ABS(FMX-FS)+1.0E-10
      SE=S-FS*(SMX-S)/FDEN
      IF (SE.LT.SMN) SE=SMN
      FMX=FS
      SMX=S
  140 SI=S-FS*(SMX-SMN)/(FMX-FMN)
      S=0.5*(SE+SI)
      GO TO 120
  150 DIS(M)=SQRT((XT-X2)**2+(ZT-Z2)**2)
      GO TO (160,170,180,180), M
  160 BRIK=DIS(1)/DELZ(K)
      GO TO 180
  170 BTIK=DIS(2)/DELX(I)
```

```
  180 CONTINUE
      M=0
      BIK=0.0
  190 CONTINUE
      M=M+1
      IF (M.EQ.5) GO TO 210
      IF (IFLG(M).EQ.0) GO TO 190
      MP1=M+1
      IF (MP1.EQ.5) MP1=1
      MM1=M-1
      IF (MM1.EQ.0) MM1=4
      BIK=BIK+DIS(M)*DIS(MM1)
      IF (IFLG(MP1).EQ.1) GO TO 200
      DIS2=DIS(M)
  200 CONTINUE
      IF (IFLG(MM1).EQ.1) GO TO 190
      DIS1=DIS(MM1)
      GO TO 190
  210 CONTINUE
      IF (IFLGS.EQ.3) BIK=BIK-DIS1*DIS2
      BIK=0.5*BIK*RDXDZ
      IF (BIK.GT.1.0) BIK=1.0
  220 CONTINUE
      IF (IQH(L).EQ.1) GO TO 230
      BIK=-BIK
  230 F(IJK)=F(IJK)+BIK
      IF (F(IJK).GT.0.99) F(IJK)=1.0
      IF (F(IJK).LT.0.01) F(IJK)=0.0
  240 CONTINUE
  250 CONTINUE
C
      NOTE: ALL JPLANES HAVE SAME VALUES
C
      DO 290 J=1,JMAX
      IF (J.EQ.2) GO TO 290
      DO 280 K=1,KMAX
      DO 270 I=1,IMAX
      I2K=NQ*(II5*(K-1)+IMAX+(I-1))+1
      IJK=NQ*(II5*(K-1)+IMAX*(J-1)+(I-1))+1
      IF (J.EQ.1.AND.WF.LE.2) GO TO 260
      IF (J.EQ.JMAX.AND.WBK.LE.2) GO TO 260
      F(IJK)=F(I2K)
      GO TO 270
  260 F(IJK)=0.0
  270 CONTINUE
  280 CONTINUE
  290 CONTINUE
C
      GO TO 410
C
C +++  THIS PORTION FOR NON-AXISYMMETRIC SURFACES
C
C +++  FCN=(X-QA1)**2+(Y-QB1)**2+(Z-QC1)**2+QD1
C
C      FOR GENERAL SURFACES CHANGE DEFINITION OF FCN
C
C       QA2,QA3 ARE USED IN THIS FORMULATION TO DENOTE APPROXIMATE
C       UPPER AND LOWER LIMITS FOR I INDEX OF MESH CONTAING SURFACE FOR
C       EFFICIENCY PURPOSES.  QB3,QB2 FOR J INDEX, QC3,QC2 FOR K INDEX
C
C       QD2 IS USED TO INDICATE IF MESH IS INITIALLY FLUID OR VOID
C       =1.0 FLUID, =0.0 VOID
C
C +++ INSIDE FCN=NEGATIVE VALUE
C +++ IQH=1 ADD FLUID INSIDE FCN, IQH=0 SUBTRACT FLUID INSIDE FCN
C +++
C
      DATA LMAX, MMAX, NMAX /20,40,20/
      DATA SUMVOID, SUMLIQD, SUMVOL /0.0,0.0,0.0/
C
  300 RXIM1=1.0/X(IM1)
      FLMAX=1.0/FLOAT(LMAX)
      FMMAX=1.0/FLOAT(MMAX)
      FNMAX=1.0/FLOAT(NMAX)
      DO 400 NS=1,NQBS
      IBOT=QA2(NS)
      ITOP=QA3(NS)
      JBOT=QB2(NS)
      JTOP=QB3(NS)
      KBOT=QC2(NS)
```

```
      KTOP=QC3(NS)
      FIQH=IQH(NS)
C
      DO 390 K=1,KMAX
      DO 390 J=1,JMAX
      DO 390 I=1,IMAX
      CALL IJKONLY
      IF (NS.GT.1) GO TO 310
      F(IJK)=1.0
      IF (QD2(NS).EQ.0.0) F(IJK)=0.0
  310 IF (I.EQ.1.OR.I.EQ.IMAX) GO TO 390
      IF (J.EQ.1.OR.J.EQ.JMAX) GO TO 390
      IF (K.EQ.1.OR.K.EQ.KMAX) GO TO 390
      VOLLC=DELX(I)*DELZ(K)*DELY(J)/RRI(I)
      IF (I.LT.IBOT.OR.I.GT.ITOP) GO TO 380
      IF (J.LT.JBOT.OR.J.GT.JTOP) GO TO 380
      IF (K.LT.KBOT.OR.K.GT.KTOP) GO TO 380
C
      CTEST=0.0
      DO 320 N=K-1,K
      DO 320 M=J-1,J
      DO 320 L=I-1,I
      X1=X(L)*CTHJBK(M)
      Y1=X(L)*STHJBK(M)
      Z1=Z(N)
      FNC=(X1-QA1(NS))**2+(Y1-QB1(NS))**2+(Z1-QC1(NS))**2+QD1(NS)
      CTEST=CTEST+CVMGP(1.0,0.0,FNC)
  320 CONTINUE
      IF (CTEST.GT.7.5) GO TO 380
      IF (CTEST.GT.0.5) GO TO 330
      F(IJK)=FIQH
      GO TO 380
C
  330 SVOLSC=0.0
      DELTAY=DELY(J)*FMMAX*RXIM1
      DELTAX=DELX(I)*FLMAX
      DELTAZ=DELZ(K)*FNMAX
      RSC=X(I-1)-0.5*DELTAX
      RSD2=X(I-1)
      DO 360 L=1,LMAX
      RSC=RSC+DELTAX
      RSD1=RSD2
      RSD2=RSD1+DELTAX
      RA=SQRT(0.5*(RSD2**2+RSD1**2))
      DELTAS=DELTAY*RSC
      VOLSC=DELTAX*DELTAZ*DELTAS
      YSC=Y(J-1)*RSC*RXIM1-0.5*DELTAS
      DO 350 M=1,MMAX
      YSC=YSC+DELTAS
      THSC=YSC/RSC
      X1=RA*COS(THSC)
      Y1=RA*SIN(THSC)
      ZSC=Z(K-1)-0.5*DELTAZ
      DO 340 N=1,NMAX
      ZSC=ZSC+DELTAZ
      Z1=ZSC
C
      FNC=(X1-QA1(NS))**2+(Y1-QB1(NS))**2+(Z1-QC1(NS))**2+QD1(NS)
C
      SVOLSC=SVOLSC+CVMGP(0.0,VOLSC,FNC)
C
  340 CONTINUE
  350 CONTINUE
  360 CONTINUE
C
      FVOL=SVOLSC/VOLLC
      IF (IQH(NS).EQ.1) F(IJK)=F(IJK)+FVOL
      IF (IQH(NS).EQ.0) F(IJK)=F(IJK)-FVOL
      IF (F(IJK).LE.1.0) GO TO 370
      F(IJK)=1.0
  370 IF (F(IJK).GE.0.0) GO TO 380
      F(IJK)=0.0
  380 CONTINUE
      SUMVOID=SUMVOID+(1.0-F(IJK))*VOLLC
      SUMLIQD=SUMLIQD+F(IJK)*VOLLC
      SUMVOL=SUMVOL+VOLLC
C
  390 CONTINUE
C
  400 CONTINUE
```

```
C
  410 RETURN
C
  420 FORMAT (2X,6HNQBS= ,I2)
  430 FORMAT (2X,2HI=,I1,2X,5HQA2= ,1PE12.5,2X,5HQA1= ,E12.5,2X,5HQB2=
     1 ,E12.5,2X,5HQB1= ,E12.5,2X,5HQC2= ,E12.5,2X,5HQC1= ,E12.5,2X,5HIQ
     2H= ,I2)
  440 FORMAT (2X,2HI=,I1,2X,5HQA3= ,1PE12.5,2X,5HQB3= ,E12.5,2X,5HQC3=
     1 ,E12.5,2X,5HQD3= ,E12.5,2X,5HQD2= ,E12.5,2X,5HQD1= ,E12.5)
  450 FORMAT (///20X,25HCONIC FLUID PARAMETERS    )
      END
*DK SETUP
      SUBROUTINE SETUP
*CA SLCOM1
C
C * * SET CONSTANT PARAMETERS
C
      PI=3.141592654
      EMF=1.0E-6
      EMF1=1.0-EMF
      FNOC=0.0
      FLGC=0.0
      NFLGC=0.0
      NOCON=0
      FLG=0.0
      VCHGT=0.0
      NUMTD=1
      T=0.0
      ITER=0
      CYCLE=0
      IF (IORDER.EQ.2) ALPHA=1.0
      TWTD=TDDT
      TWPLT=PLTDT
      TWPRT=PRTDT
      SIGMA=SIGMA/RHOF
C
C * * SPECIAL INPUT DATA
C
C * * SET PRESSURE FIELD AND OTHER ARRAYS TO ZERO
C
      DO 10 K=1,KMAX
      DO 10 J=1,JMAX
      DO 10 I=1,IMAX
      CALL IJKONLY
      P(IJK)=0.0
      PN(IJK)=0.0
      F(IJK)=0.0
      NF(IJK)=0
      PETA(IJK)=0.0
      FN(IJK)=0.0
      D(IJK)=0.0
   10 CONTINUE
      DO 20 L=1,100
   20 PR(L)=0.0
C
C * * COMPUTE INITIAL VOLUME FRACTION FUNCTION F IN CELLS
C
      IF (ICSURF.GT.0) GO TO 70
      IF (IEQUIB.EQ.0) GO TO 30
      BOND=GZ*X(IM1)**2/SIGMA
      CALL EQUIB (BASC(1),BASC(1001),1000,BOND,CANGLE,CYL)
      SFLHT=FLHT
   30 DO 60 I=1,IMAX
      DO 60 J=1,JMAX
      IBJK=NQ*(II5+IMAX*(J-1)+I-1)+1
      IBJKM=IBJK-II2
      DO 50 K=2,KMAX
      CALL IJKONLY
      F(IJK)=1.0
      IF (IEQUIB.EQ.0) GO TO 40
      LDCK=1000.*XI(I)/X(IM1)+1.000001
      LDCK=MIN0(1000,LDCK)
      LDCK=MAX0(1,LDCK)
      FLHT=SFLHT+BASC(LDCK)*X(IM1)
   40 IF (FLHT.GT.Z(K-1).AND.FLHT.LT.Z(K)) F(IJK)=RDZ(K)*(FLHT-Z(K-1))
      IF (Z(K-1).GE.FLHT) F(IJK)=0.0
   50 CONTINUE
      F(IBJKM)=F(IBJK)
   60 CONTINUE
      GO TO 80
```

```
C * * GENERATE SPECIAL F-FUNCTION (FLUID) CONFIGURATION
   70 CALL SETFS
C
C * * CALCULATE HYDROSTATIC PRESSURE
C
   80 DO 90 I=2,IM1
      DO 90 J=2,JM1
      DO 90 KK=2,KM1
      K=KM1-KK+2
      CALL IJKAJCT
      P(IJK)=P(IJKP)-GZ*(AMIN1(F(IJKP),0.5)*DELZ(K+1)+AMAX1(F(IJK)-0.5,0
     1 .0)*DELZ(K))
   90 CONTINUE
C
C * * CALCULATE DELT TO ENSURE VISCOUS DIFFUSION STABILITY
C
      DTVIS=DELT*1.0E+10
      DO 100 K=2,KM1
      DO 100 J=2,JM1
      DO 100 I=2,IM1
      DXSQ=DELX(I)**2
      DYSQ=DELY(J)**2
      DZSQ=DELZ(K)**2
      RDSQ1=DXSQ*DYSQ*DZSQ/(DXSQ+DYSQ+DZSQ)
      RDSQ=RDSQ1/(3.0*NU+DELT*1.0E-10)
      DTVIS=AMIN1(DTVIS,RDSQ)
  100 CONTINUE
C
C       CALCULATE PARTIAL AREAS
C
      CALL ASET
C
C * * CALCULATE BETA(IJK) FOR MESH
C
      CALL BETACAL
C
C    ****** SKIP PRINT **** REMOVE FOR DEBUG *********************
C
      GO TO 140
C
C     PRINT BETAS AC AR AT ABK FOR ALL PLANES
      KPR=9
      ASSIGN 130 TO KRET
      IF (LPR.NE.3) KPR=12
      IF (LPR.NE.3) ASSIGN 140 TO KRET
  110 WRITE (KPR,160)
      WRITE (KPR,170)
      DO 120 K=1,KMAX
      DO 120 J=1,JMAX
      DO 120 I=1,IMAX
      IJK=NQ*(II5*(K-1)+IMAX*(J-1)+(I-1))+1
      WRITE (KPR,180) I,J,K,BETA(IJK),AC(IJK),AR(IJK),AT(IJK),ABK(IJK)
  120 CONTINUE
      GO TO KRET, (130,140)
  130 ASSIGN 140 TO KRET
      KPR=12
      GO TO 110
  140 CONTINUE
C * * SET INITIAL VELOCITY FIELDS INTO U,V AND W ARRAYS
C
      DO 150 J=1,JMAX
      DO 150 I=1,IMAX
      DO 150 K=1,KMAX
      CALL IJKONLY
      CALL SETACC
      U(IJK)=0.0
      V(IJK)=0.0
      W(IJK)=0.0
      IF (F(IJK).LE.EMF) GO TO 150
      IF (BETA(IPJK).GT.0.0.AND.BETA(IPJK).NE.1.0) U(IJK)=UI
      IF (BETA(IJPK).GT.0.0.AND.BETA(IJPK).NE.1.0) V(IJK)=VI
      IF (BETA(IJKP).GT.0.0.AND.BETA(IJKP).NE.1.0) W(IJK)=WI
  150 CONTINUE
C
      RETURN
C
  160 FORMAT (1H1)
  170 FORMAT (4X,1HI,4X,1HJ,4X,1HK,7X,4HBETA,12X,2HAC,13X,2HAR,13X,2HAT,
     1 13X,3HABK)
  180 FORMAT (3I5,1P5E15.5)
```

```
      END
*DK SOLA
      SUBROUTINE SOLA
*CA SLCOM1
C
C * * NUMERICAL SOLUTION ALGORITHM
C
      MSGCODE=O
C * * REQUEST THAT MSGCODE BE SET TO 1
C * * WHEN A MESSAGE FROM A CONTROLLER ARRIVES.
      CALL MSGFLAG (MSGCODE,2)
      CALL SECOND (STIM)
      CALL FADV (1)
      WRITE (59,190)
      TRL=STIM
      T1=TRL
      T2=T1
      CALL GETJTL (TL)
      TLM=TL-150.0+(1.0-TLIMD)*1.0E+10
      IF (TD.GT.O) GO TO 160
C
C * * SET INITIAL BOUNDARY CONDITIONS
C
      CALL BC
C
      GO TO 40
C
C * * START TIME CYCLE
C
   10 CONTINUE
      T=T+DELT
      AVE=1.0
      CYCLE=CYCLE+1
      IF (NFLGC.GE.200.OR.NOCON.GE.25) T=1.0E+10
      IF (MOD(CYCLE,20).EQ.0) WRITE (59,180) T,CYCLE,ITER,DELT,EPSI
      ITER=0
      FLG=1.0
      FNOC=0
   20 CONTINUE
C
C * * EXPLICITLY APPROXIMATE NEW TIME LEVEL VELOCITIES
C
      CALL TILDE
C
C * * SET BOUNDARY CONDITIONS
C
      CALL BC
C
C * * SECOND ORDER OPTION
C
      IF (IORDER.EQ.1.OR.AVE.LT.0.75) GO TO 30
      CALL SECORD
      GO TO 20
C
C * * ITERATIVELY ADJUST CELL PRESSURE AND VELOCITY
C
   30 IF (ISOR.EQ.1) CALL PRESSIT
      IF (ISOR.EQ.0) CALL PRESCR
      CALL BC
C * * UPDATE FLUID CONFIGURATION
C
   40 CALL VFCONV
      IF (FLGC.GT.0.5) GO TO 110
C
C * * SET BOUNDARY CONDITIONS
C
      CALL BC
C
C * * DETERMINE PRESSURE INPERPOLATION FACTOR AND NEIGHBOR
C
      CALL PETACAL
      CALL BC
      IF (CYCLE.LT.1.AND.ISOR.NE.2) CALL PRESCR
C
C * * PLOT VELOCITY VECTORS AND FREE SURFACE ON FILM
C * * LPR=1,  DRAW ONLY AT TWPLT
C * * LPR=2,  DRAW AND PRINT DATA ON FILM AT TWPLT
C * * PRINT DATA ON LISTING ONLY AT TWPRT
C
      IF (CYCLE.EQ.O.OR.CYCLE.EQ.1) GO TO 60
```

```
      IF (T+1.0E-10.GE.TWFIN) GO TO 50                                  SOLA
      IF (T+1.0E-6.LT.TWPLT) GO TO 70                                   SOLA
   50 TWPLT=TWPLT+PLTDT                                                 SOLA
   60 IF (LPR.EQ.0) GO TO 70                                            SOLA
      IF (LPR.EQ.1.OR.LPR.EQ.2) CALL DRAW                               SOLA
      IF (LPR.EQ.2) CALL LPRT                                           SOLA
C                                                                       SOLA
C * * PRINT FIELD VARIABLE DATA ON PAPER                                SOLA
C                                                                       SOLA
   70 CONTINUE                                                          SOLA
      IF (CYCLE.EQ.0.OR.CYCLE.EQ.1) GO TO 80                            SOLA
      IF (T+1.0E-6.LT.TWPRT) GO TO 90                                   SOLA
      IF (TWPRT.EQ.TWPLT-PLTDT.AND.LPR.EQ.2) GO TO 90                   SOLA
      TWPRT=TWPRT+PRTDT                                                 SOLA
   80 IF (LPR.NE.3) GO TO 90                                            SOLA
      CALL LPRT2                                                        SOLA
C                                                                       SOLA
C * * SET THE ADVANCE TIME ARRAYS INTO THE TIME-N ARRAYS                SOLA
C                                                                       SOLA
   90 CONTINUE                                                          SOLA
      DO 100 K=1,KMAX                                                   SOLA
      DO 100 J=1,JMAX                                                   SOLA
      DO 100 I=1,IMAX                                                   SOLA
      CALL IJKONLY                                                      SOLA
      UN(IJK)=U(IJK)                                                    SOLA
      VN(IJK)=V(IJK)                                                    SOLA
      WN(IJK)=W(IJK)                                                    SOLA
      FN(IJK)=F(IJK)                                                    SOLA
      PN(IJK)=P(IJK)                                                    SOLA
  100 CONTINUE                                                          SOLA
C                                                                       SOLA
C * * ADJUST DELT                                                       SOLA
C                                                                       SOLA
  110 CALL DELTADJ                                                      SOLA
C                                                                       SOLA
      IF (MSGCODE.NE.1) GO TO 120                                       SOLA
C * * GET THE MESSAGE FROM THE CONTROLLER.                              SOLA
C * * THIS ALSO RESETS MSGCODE TO ZERO.                                 SOLA
      CALL GETMSGR (IBUFF,NW,1)                                         SOLA
C * * CHECK FOR THE MESSAGE "HALT"                                      SOLA
      IF (IBUFF.EQ.4HHALT) GO TO 140                                    SOLA
  120 CONTINUE                                                          SOLA
C                                                                       SOLA
C * * TEST IF PROBLEM IS FINISHED                                       SOLA
C                                                                       SOLA
      IF (T+1.0E-10.GE.TWFIN) GO TO 140                                 SOLA
      IF (CYCLE.LE.0) GO TO 10                                          SOLA
C                                                                       SOLA
C * * CALCULATE GRIND TIME AND PRINT OUT PROBLEM DATA                   SOLA
C                                                                       SOLA
      TOLD=T2                                                           SOLA
      IF (MOD(CYCLE,20).NE.0) GO TO 130                                 SOLA
      CALL SECOND (T2)                                                  SOLA
      XXX=(T2-TOLD)                                                     SOLA
      XX=XXX*RIJK                                                       SOLA
      IF (LPR.NE.0) WRITE (12,170) T,CYCLE,ITER,XX,DELT,EPSI,VCHGT      SOLA
     1 ,VOFTOT,XXX                                                      SOLA
      IF (LPR.NE.0) WRITE (9,170) T,CYCLE,ITER,XX,DELT,EPSI,VCHGT,VOFTOT SOLA
     1 ,XXX                                                             SOLA
  130 CONTINUE                                                          SOLA
      IF (T2-STIM.GE.TLM) GO TO 140                                     SOLA
C                                                                       SOLA
C * * WRITE PROBLEM DUMP TAPE                                           SOLA
C                                                                       SOLA
      IF (T.GE.TWTD) CALL WRTAPE                                        SOLA
      GO TO 10                                                          SOLA
  140 CALL WRTAPE                                                       SOLA
      IF (T+1.0E-10.GE.TWFIN) GO TO 150                                 SOLA
      IF (LPR.EQ.1.OR.LPR.EQ.2) CALL DRAW                               SOLA
      IF (LPR.EQ.2) CALL LPRT                                           SOLA
  150 CONTINUE                                                          SOLA
      CALL GDONE                                                        SOLA
      RETURN                                                            SOLA
C                                                                       SOLA
C * * READ TAPE DUMP                                                    SOLA
C                                                                       SOLA
  160 CALL RDTAPE                                                       SOLA
      GO TO 10                                                          SOLA
C                                                                       SOLA
  170 FORMAT (3H T=,1PE12.5,7H CYCLE=,I4,6H ITER=,I4,8H GRINDS=,E10.3,6H SOLA
```

```
     1 DELT=,E11.4,6H EPSI=,E10.3,7H VCHGT=,E10.3,8H VOFTOT=,E12.5,/5X,*
    2TOTAL TIME FOR CYCLE = *,E11.4)
  180 FORMAT (/,3H T=,1PE10.3,7H CYCLE=,I5,6H ITER=,I5,6H DELT=,E10.3,6H
     1 EPSI=,E10.3)
  190 FORMAT (/,25HENTERING SUBROUTINE SOLA:,/)
      END
*DK SURCNTR
      SUBROUTINE SURCNTR (I1,I2,J1,J2,K1,K2,MPLN,AXP,AYP)
*CA SLCOM1
C
C * * DRAW FREE SURFACE ON SELECTED PLANE VELOCITY VECTOR PLOT
C
      FPL=0.5
C
C * * SELECT INDICES AND COORDINATES FOR SURFACE PLANE TO BE PLOTTED
C
      GO TO (10,20,30), MPLN
   10 L1=J1
      L2=J2
      M1=I1
      N1=K1
      N2=K2
      GO TO 40
   20 L1=I1
      L2=I2
      M1=J1
      N1=K1
      N2=K2
      GO TO 40
   30 L1=I1
      L2=I2
      M1=K1
      N1=J1
      N2=J2
   40 CONTINUE
C
C * * MULTIPLIERS OF 1 OR 0 TO DETERMINE PLANE PLOTTED
C
      AXYP=AXP*AYP
      AXP1=1.0-AXP
      AYP1=1.0-AYP
      M=M1
C
C * * DRAW FREE SURFACE
C
      DO 140 L=L1,L2
      DO 140 N=N1,N2
      XLR=1.0
      XLL=1.0
      GO TO (50,60,70), MPLN
   50 I=M
      J=L
      K=N
      XLCP=YJ(J+1)
      XLC=YJ(J)
      XLCM=YJ(J-1)
      YNCP=ZK(K+1)
      YNC=ZK(K)
      YNCM=ZK(K-1)
      GO TO 80
   60 J=M
      I=L
      K=N
      XLCP=XI(I+1)
      XLC=XI(I)
      XLCM=XI(I-1)
      YNCP=ZK(K+1)
      YNC=ZK(K)
      YNCM=ZK(K-1)
      GO TO 80
   70 K=M
      I=L
      J=N
      XLCP=XI(I+1)
      XLC=XI(I)
      XLCM=XI(I-1)
      YNCP=YJ(J+1)/RR(I)
      YNC=YJ(J)/RR(I)
      YNCM=YJ(J-1)/RR(I)
      IF (CYL.LT.0.5.OR.MPLN.NE.3) GO TO 80
```

```
      YNCP=YJ(J+1)/X(IM1)
      YNC=YJ(J)/X(IM1)
      YNCM=YJ(J-1)/X(IM1)
      XLR=X(I)
      XLL=X(I-1)
      IF (XLL.EQ.0.0) XLL=EM6
   80 CONTINUE
      CALL CALCIJK
      LMN=IJK
      LPMN=IPJK*AXP+IJPK*AXP1
      LMNP=IJPK*AYP1+IJKP*AYP
      LPMNP=IJPKP*AXP1+IPJKP*AXYP+IPJPK*AYP1
      LMNM=IJMK*AYP1+IJKM*AYP
      LMMN=IMJK*AXP+IJMK*AXP1
      LPMNM=IJPKM*AXP1+IPJKM*AXYP+IPJMK*AYP1
      LMMNP=IJMKP*AXP1+IMJKP*AXYP+IMJPK*AYP1
      IF (BETA(LMN).LT.0.0) GO TO 140
      FATR=0.25*(F(LMN)+F(LPMN)+F(LMNP)+F(LPMNP))
      FXTR=0.5*(F(LPMNP)+F(LPMN)-F(LMNP)-F(LMN))/(XLCP-XLC)
      FYTR=0.5*(F(LMNP)+F(LPMNP)-F(LMN)-F(LPMN))/(YNCP-YNC)
      FYTR=FYTR/XLR
      FTRS=FXTR**2+FYTR**2
      IF (FTRS.EQ.0.0) FTRS=1.0E+10
      XTR=0.5*(XLCP+XLC)+(FPL-FATR)*FXTR/FTRS
      XTR=AMAX1(XTR,XLC)
      XTR=AMIN1(XTR,XLCP)
      YTR=0.5*(YNC+YNCP)+(FPL-FATR)*FYTR/(FTRS*XLR)
      YTR=AMAX1(YTR,YNC)
      YTR=AMIN1(YTR,YNCP)
      X2EF=XTR-(AXP*XBL+AXP1*YBF)
      Y2EF=YTR-(AYP*ZBB+AYP1*YBF)
      IF (F(LMN).GT.0.5.AND.F(LPMN).GT.0.5) GO TO 110
      IF (F(LMN).LT.0.5.AND.F(LPMN).LT.0.5) GO TO 110
      FABR=0.25*(F(LMN)+F(LPMN)+F(LMNM)+F(LPMNM))
      FXBR=0.5*(F(LPMN)+F(LPMNM)-F(LMN)-F(LMNM))/(XLCP-XLC)
      FYBR=0.5*(F(LMN)+F(LPMN)-F(LMNM)-F(LPMNM))/(YNC-YNCM)
      FYBR=FYBR/XLR
      FBRS=FXBR**2+FYBR**2
      IF (FBRS.EQ.0.0) FBRS=1.0E+10
      XBMR=0.5*(XLCP+XLC)+(FPL-FABR)*FXBR/FBRS
      XBMR=AMAX1(XBMR,XLC)
      XBMR=AMIN1(XBMR,XLCP)
      YBMR=0.5*(YNC+YNCM)+(FPL-FABR)*FYBR/(FBRS*XLR)
      YBMR=AMAX1(YBMR,YNCM)
      YBMR=AMIN1(YBMR,YNC)
      X1EF=XBMR-(AXP*XBL+AXP1*YBF)
      Y1EF=YBMR-(AYP*ZBB+AYP1*YBF)
      GO TO (100,100,90), MPLN
   90 IF (CYL.LT.0.5) GO TO 100
      X2EF=XTR*COS(YTR)-XBLC
      Y2EF=XTR*SIN(YTR)-YBFC
      X1EF=XBMR*COS(YBMR)-XBLC
      Y1EF=XBMR*SIN(YBMR)-YBFC
  100 CONTINUE
      IX1=FIXL(MPLN)+X1EF*XCONV(MPLN)
      IY1=FIYB+Y1EF*YCONV(MPLN)
      IX2=FIXL(MPLN)+X2EF*XCONV(MPLN)
      IY2=FIYB+Y2EF*YCONV(MPLN)
      CALL DRV (IX1,IY1,IX2,IY2)
  110 CONTINUE
      IF (F(LMN).GT.0.5.AND.F(LMNP).GT.0.5) GO TO 140
      IF (F(LMN).LT.0.5.AND.F(LMNP).LT.0.5) GO TO 140
      FATL=0.25*(F(LMN)+F(LMNP)+F(LMMN)+F(LMMNP))
      FXTL=0.5*(F(LMNP)+F(LMN)-F(LMMNP)-F(LMMN))/(XLC-XLCM)
      FYTL=0.5*(F(LMMNP)+F(LMNP)-F(LMMN)-F(LMN))/(YNCP-YNC)
      FYTL=FYTL/XLL
      FTLS=FXTL**2+FYTL**2
      IF (FTLS.EQ.0.0) FTLS=1.0E+10
      XTL=0.5*(XLCM+XLC)+(FPL-FATL)*FXTL/FTLS
      XTL=AMAX1(XTL,XLCM)
      XTL=AMIN1(XTL,XLC)
      YTL=0.5*(YNC+YNCP)+(FPL-FATL)*FYTL/(FTLS*XLL)
      YTL=AMAX1(YTL,YNC)
      YTL=AMIN1(YTL,YNCP)
      X1EF=XTL-(AXP*XBL+AXP1*YBF)
      Y1EF=YTL-(AYP*ZBB+AYP1*YBF)
      GO TO (130,130,120), MPLN
  120 IF (CYL.LT.0.5) GO TO 130
      X2EF=XTR*COS(YTR)-XBLC
      Y2EF=XTR*SIN(YTR)-YBFC
```

```
      X1EF=XTL*COS(YTL)-XBLC
      Y1EF=XTL*SIN(YTL)-YBFC
  130 CONTINUE
      IX1=FIXL(MPLN)+X1EF*XCONV(MPLN)
      IY1=FIYB+Y1EF*YCONV(MPLN)
      IX2=FIXL(MPLN)+X2EF*XCONV(MPLN)
      IY2=FIYB+Y2EF*YCONV(MPLN)
      CALL DRV (IX1,IY1,IX2,IY2)
  140 CONTINUE
C
      RETURN
      END
*DK SURFPLT
      SUBROUTINE SURFPLT (I1,I2,J1,J2,K1,K2,NA,IPER)
*CA SLCOM1
C
C * * DRAW FREE SURFACE IN PERSPECTIVE
C
      CALL FADV (NA)
      CALL DRFP
      CALL LINCNT (59)
      WRITE (12,70)
      WRITE (12,80) NAME,JNM,DAT,CLK
      WRITE (12,90) ITER,T,CYCLE
      FPL=0.5
      I1M=I1-1
      J1M=J1-1
      K1M=K1-1
      DO 60 I=I1M,I2
      DO 60 J=J1M,J2
      DO 60 K=K1M,K2
      CALL CALCIJK
      FAIM=0.0
      FAJM=0.0
      FAKM=0.0
      CRX=CYL/X(IM1)
      IF (BETA(IJK).LT.0.0.AND.BETA(IJPK).LT.0.0.AND.BETA(IJPKP).LT.0.0
     1 .AND.BETA(IJKP).LT.0.0) GO TO 10
      IF (F(IJK).GT.FPL.AND.F(IJPK).GT.FPL.AND.F(IJPKP).GT.FPL.AND.F
     1 (IJKP).GT.FPL) GO TO 10
      IF (F(IJK).LT.FPL.AND.F(IJPK).LT.FPL.AND.F(IJPKP).LT.FPL.AND.F
     1 (IJKP).LT.FPL) GO TO 10
      FAIM=0.125*(F(IJK)+F(IJPK)+F(IJPKP)+F(IJKP)+F(IMJK)+F(IMJPK)+F
     1 (IMJPKP)+F(IMJKP))
      FAIX=0.5*(F(IJK)+F(IJPK)+F(IJPKP)+F(IJKP)-4.0*FAIM)/(XI(I)-XI(I-1)
     1 )
      FAIY=0.5*(F(IMJK)+F(IJK)+F(IJKP)+F(IMJKP)-4.0*FAIM)*RR(I-1)/(YJ(J)
     1 -YJ(J+1))
      FAIZ=0.5*(F(IJK)+F(IJPK)+F(IMJPK)+F(IMJK)-4.0*FAIM)/(ZK(K)-ZK(K+1)
     1 )
      FAIS=FAIX**2+FAIY**2+FAIZ**2
      IF (FAIS.EQ.0.0) FAIS=1.0E+10
      XAIM=0.5*(XI(I-1)+XI(I))+(FPL-FAIM)*FAIX/FAIS
      XAIM=AMAX1(XAIM,XI(I-1))
      XAIM=AMIN1(XAIM,XI(I))
      YAIM=0.5*(YJ(J)+YJ(J+1))+(FPL-FAIM)*FAIY*RR(I-1)/FAIS
      YAIM=AMAX1(YAIM,YJ(J))
      YAIM=AMIN1(YAIM,YJ(J+1))
      TH1=CYL*RX(IM1)*YAIM
      YAIM=YAIM*(1.0-CYL)+XAIM*SIN(TH1)
      XAIM=XAIM*COS(TH1)
      ZAIM=0.5*(ZK(K)+ZK(K+1))+(FPL-FAIM)*FAIZ/FAIS
      ZAIM=AMAX1(ZAIM,ZK(K))
      ZAIM=AMIN1(ZAIM,ZK(K+1))
      TH1=CRX*YAIM
      YAIM=XAIM*SIN(TH1)+YAIM*(1.0-CYL)
      XAIM=XAIM*COS(TH1)
   10 CONTINUE
      IF (BETA(IJK).LT.0.0.AND.BETA(IPJK).LT.0.0.AND.BETA(IPJKP).LT.0.0
     1 .AND.BETA(IJKP).LT.0.0) GO TO 20
      IF (F(IJK).GT.FPL.AND.F(IPJK).GT.FPL.AND.F(IPJKP).GT.FPL.AND.F
     1 (IJKP).GT.FPL) GO TO 20
      IF (F(IJK).LT.FPL.AND.F(IPJK).LT.FPL.AND.F(IPJKP).LT.FPL.AND.F
     1 (IJKP).LT.FPL) GO TO 20
      FAJM=0.125*(F(IJK)+F(IPJK)+F(IPJKP)+F(IJKP)+F(IJMK)+F(IPJMK)+F
     1 (IPJMKP)+F(IJMKP))
      FAJX=0.5*(F(IJK)+F(IJKP)+F(IJMKP)+F(IJMK)-4.0*FAJM)/(XI(I)-XI(I+1)
     1 )
      FAJY=0.5*(F(IJK)+F(IPJK)+F(IPJKP)+F(IJKP)-4.0*FAJM)*RR(I)/(YJ(J)
     1 -YJ(J-1))
```

```
      FAJZ=0.5*(F(IJK)+F(IJMK)+F(IPJMK)+F(IPJK)-4.0*FAJM)/(ZK(K)-ZK(K+1)
     1 )
      FAJS=FAJX**2+FAJY**2+FAJZ**2
      IF (FAJS.EQ.0.0) FAJS=1.0E+10
      XAJM=0.5*(XI(I)+XI(I+1))+(FPL-FAJM)*FAJX/FAJS
      XAJM=AMAX1(XAJM,XI(I))
      XAJM=AMIN1(XAJM,XI(I+1))
      YAJM=0.5*(YJ(J-1)+YJ(J))+(FPL-FAJM)*FAJY*RR(I)/FAJS
      YAJM=AMAX1(YAJM,YJ(J-1))
      YAJM=AMIN1(YAJM,YJ(J))
      TH1=CYL*RX(IM1)*YAJM
      YAJM=YAJM*(1.0-CYL)+XAJM*SIN(TH1)
      XAJM=XAJM*COS(TH1)
      ZAJM=0.5*(ZK(K)+ZK(K+1))+(FPL-FAJM)*FAJZ/FAJS
      ZAJM=AMAX1(ZAJM,ZK(K))
      ZAJM=AMIN1(ZAJM,ZK(K+1))
      TH1=CRX*YAJM
      YAJM=XAJM*SIN(TH1)+YAJM*(1.0-CYL)
      XAJM=XAJM*COS(TH1)
   20 CONTINUE
      IF (BETA(IJK).LT.0.0.AND.BETA(IPJK).LT.0.0.AND.BETA(IPJPK).LT.0.0
     1 .AND.BETA(IJPK).LT.0.0) GO TO 30
      IF (F(IJK).GT.FPL.AND.F(IPJK).GT.FPL.AND.F(IPJPK).GT.FPL.AND.F
     1 (IJPK).GT.FPL) GO TO 30
      IF (F(IJK).LT.FPL.AND.F(IPJK).LT.FPL.AND.F(IPJPK).LT.FPL.AND.F
     1 (IJPK).LT.FPL) GO TO 30
      FAKM=0.125*(F(IJK)+F(IPJK)+F(IPJPK)+F(IJPK)+F(IJKM)+F(IPJKM)+F
     1 (IPJPKM)+F(IJPKM))
      FAKX=0.5*(F(IJK)+F(IJPK)+F(IJPKM)+F(IJKM)-4.0*FAKM)/(XI(I)-XI(I+1)
     1 )
      FAKY=0.5*(F(IJK)+F(IPJK)+F(IPJKM)+F(IJKM)-4.0*FAKM)*RR(I)/(YJ(J)
     1 -YJ(J+1))
      FAKZ=0.5*(F(IJK)+F(IPJK)+F(IPJPK)+F(IJPK)-4.0*FAKM)/(ZK(K)-ZK(K-1)
     1 )
      FAKS=FAKX**2+FAKY**2+FAKZ**2
      IF (FAKS.EQ.0.0) FAKS=1.0E+10
      XAKM=0.5*(XI(I)+XI(I+1))+(FPL-FAKM)*FAKX/FAKS
      XAKM=AMAX1(XAKM,XI(I))
      XAKM=AMIN1(XAKM,XI(I+1))
      YAKM=0.5*(YJ(J)+YJ(J+1))+(FPL-FAKM)*FAKY*RR(I)/FAKS
      YAKM=AMAX1(YAKM,YJ(J))
      YAKM=AMIN1(YAKM,YJ(J+1))
      TH1=CYL*RX(IM1)*YAKM
      YAKM=YAKM*(1.0-CYL)+XAKM*SIN(TH1)
      XAKM=XAKM*COS(TH1)
      ZAKM=0.5*(ZK(K-1)+ZK(K))+(FPL-FAKM)*FAKZ/FAKS
      ZAKM=AMAX1(ZAKM,ZK(K-1))
      ZAKM=AMIN1(ZAKM,ZK(K))
      TH1=CRX*YAKM
      YAKM=XAKM*SIN(TH1)+YAKM*(1.0-CYL)
      XAKM=XAKM*COS(TH1)
   30 CONTINUE
      IF (FAIM.EQ.0.0.AND.FAJM.EQ.0.0.AND.FAKM.EQ.0.0) GO TO 60
      FAC=0.125*(F(IJK)+F(IPJK)+F(IPJPK)+F(IJPK)+F(IJKP)+F(IPJKP)+F
     1 (IPJPKP)+F(IJPKP))
      FACX=0.5*(F(IJK)+F(IJPK)+F(IJPKP)+F(IJKP)-4.0*FAC)/(XI(I)-XI(I+1))
      FACY=0.5*(F(IJK)+F(IPJK)+F(IPJKP)+F(IJKP)-4.0*FAC)*RR(I)/(YJ(J)-YJ
     1 (J+1))
      FACZ=0.5*(F(IJK)+F(IPJK)+F(IPJPK)+F(IJPK)-4.0*FAC)/(ZK(K)-ZK(K+1))
      FACS=FACX**2+FACY**2+FACZ**2
      IF (FACS.EQ.0.0) FACS=1.0E+10
      XAC=0.5*(XI(I)+XI(I+1))+(FPL-FAC)*FACX/FACS
      XAC=AMAX1(XAC,XI(I))
      XAC=AMIN1(XAC,XI(I+1))
      YAC=0.5*(YJ(J)+YJ(J+1))+(FPL-FAC)*FACY*RR(I)/FACS
      YAC=AMAX1(YAC,YJ(J))
      YAC=AMIN1(YAC,YJ(J+1))
      TH1=CYL*RX(IM1)*YAC
      YAC=YAC*(1.0-CYL)+XAC*SIN(TH1)
      XAC=XAC*COS(TH1)
      ZAC=0.5*(ZK(K)+ZK(K+1))+(FPL-FAC)*FACZ/FACS
      ZAC=AMAX1(ZAC,ZK(K))
      ZAC=AMIN1(ZAC,ZK(K+1))
      TH1=CRX*YAC
      YAC=XAC*SIN(TH1)+YAC*(1.0-CYL)
      XAC=XAC*COS(TH1)
      CALL PCNV (IXIC,IETAC,XAC,YAC,ZAC)
      IF (FAIM.EQ.0.0.OR.I.EQ.I1M) GO TO 40
      CALL PCNV (IXIIM,IETAIM,XAIM,YAIM,ZAIM)
      CALL DRVEC (IXIC,IETAC,IXIIM,IETAIM)
```

```
   40 IF (FAJM.EQ.O.O.OR.J.EQ.J1M) GO TO 50
      CALL PCNV (IXIJM,IETAJM,XAJM,YAJM,ZAJM)
      CALL DRVEC (IXIC,IETAC,IXIJM,IETAJM)
   50 IF (FAKM.EQ.O.O.OR.K.EQ.K1M) GO TO 60
      CALL PCNV (IXIKM,IETAKM,XAKM,YAKM,ZAKM)
      CALL DRVEC (IXIC,IETAC,IXIKM,IETAKM)
   60 CONTINUE
      RETURN
C
   70 FORMAT (21H PERSPECTIVE SURFACE )
   80 FORMAT (1H ,2X,10A8,1X,A8,2(1X,A8))
   90 FORMAT (6X,5HITER=,I5,18X,6HTIME= ,1PE12.5,12X,7HCYCLE= ,I4)
      END
*DK SURF1ON
      SUBROUTINE SURF1ON
C
*CA SLCOM1
      DIMENSION HX(3,3), HY(3,3), HZ(3,3)
      DIMENSION ANE(3,3), ANW(3,3), AEN(3,3), AES(3,3), CEE(3,3), CEW(3,
     1 3), CNN(3,3), CNS(3,3)
      DATA NOWALL /1/
      DATA SANG, CSANG /0.087155742,0.996194698/
      PSLIM=25.0/AMIN1(X(IM1),Y(JM1),Z(KM1))
      DO 1070 K=2,KM1
      DO 1070 J=2,JM1
      DO 1070 I=2,IM1
      CALL CALCIJK
      IF (BETA(IJK).LT.0.0) GO TO 1070
      NFC=NF(IJK)
      IF (NFC.EQ.0.OR.NFC.GT.6) GO TO 1070
C
C     CHECK IF SUFFICIENT FLUID PRESENT
C
      FSUM=0.0
      DO 10 N=K-1,K+1
      DO 10 M=J-1,J+1
      DO 10 L=I-1,I+1
      LMN=NQ*(IMAX*JMAX*(N-1)+IMAX*(M-1)+(L-1))+1
      FSUM=FSUM+F(LMN)
   10 CONTINUE
      IF (FSUM.LT.1.0) GO TO 1070
C
C     VARIABLES FOR USE IN SIX AND SEVEN POINT DERIVATIVE FORMULAS
C
C     Z COORDINATE PATTERN
      DZN=0.5*DELZ(K+1)
      DZ=0.5*DELZ(K)
      DZS=0.5*DELZ(K-1)
      DZP=DZN+DZ
      DZM=DZS+DZ
      ZP=DZP/DZM
      ZM=1.0/ZP
      RDZS=1.0/(DZP+DZM)
      DZD=DZP-DZM
      RPDZ=1.0/(DZP*DZM)
      FRZP=DZ/DZP
      FRZM=DZ/DZM
      RATZP=DZP*RDZS
      RATZM=DZM*RDZS
      DZPA=DZP+DZ
      DZMA=DZM+DZ
      DZDN=DZN-DZ
      DZDS=DZS-DZ
C
C     MIRROR PATTERN FOR THETA COORDINATE (Y)
C
      DTHE=0.5*DELY(J+1)*RX(IM1)
      DTH=0.5*DELY(J)*RX(IM1)
      DTHW=0.5*DELY(J-1)*RX(IM1)
      DTHP=DTHE+DTH
      DTHM=DTH+DTHW
      THP=DTHP/DTHM
      THM=1.0/THP
      RDTHS=1.0/(DTHP+DTHM)
      DTHD=DTHP-DTHM
      RPDTH=1.0/(DTHP*DTHM)
      FRTHP=DTH/DTHP
      FRTHM=DTH/DTHM
      RATTHP=DTHP*RDTHS
      RATTHM=DTHM*RDTHS
```

```
      DTHPA=DTHP+DTH
      DTHMA=DTHM+DTH
      DTHDE=DTHE-DTH
      DTHDW=DTHW-DTH
C
C     MIRROR PATTERN FOR RADIAL COORDINATE
C
      DRO=0.5*DELX(I+1)
      DR=0.5*DELX(I)
      DRI=0.5*DELX(I-1)
      IF (I.NE.2) GO TO 20
      DRO=0.5*DELX(4)
      DR=0.5*DELX(3)
      DRI=0.5*DELX(2)
   20 CONTINUE
      DRP=DRO+DR
      DRM=DR+DRI
      RP=DRP/DRM
      RM=1.0/RP
      RDRS=1.0/(DRP+DRM)
      DRD=DRP-DRM
      RPDR=1.0/(DRP*DRM)
      FRRP=DR/DRP
      FRRM=DR/DRM
      RATRP=DRP*RDRS
      RATRM=DRM*RDRS
      DRPA=DRP+DR
      DRMA=DRM+DR
      DRDO=DRO-DR
      DRDI=DRI-DR
C
      DXR=0.5*(DELX(I)+DELX(I+1))
      DXL=0.5*(DELX(I)+DELX(I-1))
      IF (I.EQ.2) DXL=0.5*(DELX(2)+DELX(3))
      IF (I.EQ.2) DXR=0.5*(DELX(3)+DELX(4))
      DZT=0.5*(DELZ(K)+DELZ(K+1))
      DZB=0.5*(DELZ(K)+DELZ(K-1))
      DYBK=0.5*(DELY(J)+DELY(J+1))/RRI(I)
      DYBKP=0.5*(DELY(J)+DELY(J+1))/RRI(I+1)
      DYBKM=0.5*(DELY(J)+DELY(J+1))/RRI(I-1)
      IF (CYL.EQ.1.0.AND.I.EQ.2) DYBKM=DYBK
      DYF=0.5*(DELY(J)+DELY(J-1))/RRI(I)
      DYFP=0.5*(DELY(J)+DELY(J-1))/RRI(I+1)
      DYFM=0.5*(DELY(J)+DELY(J-1))/RRI(I-1)
      IF (CYL.EQ.1.0.AND.I.EQ.2) DYFM=DYF
      RXDEN=1.0/(DXL*DXR*(DXL+DXR))
      RYDEN=1.0/(DYF*DYBK*(DYF+DYBK))
      RYDENP=1.0/(DYFP*DYBKP*(DYFP+DYBKP))
      RYDENM=1.0/(DYFM*DYBKM*(DYFM+DYBKM))
      RZDEN=1.0/(DZT*DZB*(DZT+DZB))
C
C     ASSUME NEAR CYLINDRICAL SURFACE - COLLAPSE IN RADIAL DIRECTION
C
C      ISCR, ISCV, ISCH ARE FLAGS SET DURING COLLAPSING DOWN LOOPS
C      IF EQUAL TO 0 THEN SUBSEQUENT SLOPES MAY BE ACCURATE
C      IF EQUAL TO 1 THEN SLOPES CANNOT BE CORRECT : AT LEAST ONE ROW
C      OR COLUMN OF THE COLLAPSING ARRAYS DOES NOT CONTAIN A
C      SURFACE CELL
C
      ISCR=0
      ISCV=0
      ISCH=0
C
      ISCRE=0
      ISCRF=0
      DO 150 KK=1,3
      N=K-2+KK
      DO 140 JJ=1,3
      M=J-2+JJ
      LMN=NQ*(IMAX*JMAX*(N-1)+IMAX*(M-1)+(I-1))+1
      LMNM=LMN-NQ
      LMNP=LMN+NQ
      IF (CYL.LT.0.5) GO TO 70
      IF (M.EQ.1) GO TO 30
      IF (M.EQ.JMAX) GO TO 50
      GO TO 70
C
C     M=1
C
   30 MM=2
```

```
      IF (WF.EQ.4) MM=JM1                                               SURF1ON
   40 LMN=NQ*(IMAX*JMAX*(N-1)+IMAX*(MM-1)+(I-1))+1                      SURF1ON
      GO TO 60                                                          SURF1ON
C                                                                       SURF1ON
C       M=JMAX                                                          SURF1ON
C                                                                       SURF1ON
   50 MM=JM1                                                            SURF1ON
      IF (WBK.EQ.4) MM=2                                                SURF1ON
      GO TO 40                                                          SURF1ON
   60 LMNM=LMN-NQ                                                       SURF1ON
      LMNP=LMN+NQ                                                       SURF1ON
   70 CONTINUE                                                          SURF1ON
      FOFM=1.0                                                          SURF1ON
      IF (AC(LMNM).LE.EM6.AND.F(IMJK).LT.EMF) FOFM=0.0                  SURF1ON
      FOFP=1.0                                                          SURF1ON
      IF (AC(LMNP).LE.EM6.AND.F(IPJK).LT.EMF) FOFP=0.0                  SURF1ON
      IF (I.GE.3.AND.I.LE.IM2) GO TO 100                                SURF1ON
      IF (I.EQ.2) GO TO 80                                              SURF1ON
      FOFMM=1.0                                                         SURF1ON
      LMNMM=LMNM-NQ                                                     SURF1ON
      IF (AC(LMNMM).LT.EM6.AND.F(IMJK).LT.EMF) FOFMM=0.0                SURF1ON
      DELTA=2.0*((1.0+AC(LMNM)*(F(LMNM)-1.0))*XI(I-1)*DELX(I-1)*FOFM+(1.SURF1ON
     1 0+AC(LMN)*(F(LMN)-1.0))*XI(I)*DELX(I)+(1.0+AC(LMNMM)*(F(LMNMM)-1.SURF1ON
     2 0))*XI(I-2)*DELX(I-2)*FOFMM)                                     SURF1ON
      DELTB=2.0*(XI(I-2)*DELX(I-2)*FOFMM+XI(I-1)*DELX(I-1)*FOFM+XI(I)   SURF1ON
     1 *DELX(I))                                                        SURF1ON
      IF (NFC.NE.2) HX(KK,JJ)=SQRT(X(I-3)**2+DELTA)-X(I-3)              SURF1ON
      IF (NFC.EQ.2) HX(KK,JJ)=X(I)-SQRT(X(I)**2-DELTA)                  SURF1ON
      IF (NFC.NE.2) HXCOL=SQRT(X(I-3)**2+DELTB)-X(I-3)-EM6              SURF1ON
      IF (NFC.EQ.2) HXCOL=X(I)-SQRT(X(I)**2-DELTB)-EM6                  SURF1ON
      IF (HX(KK,JJ).LE.0.0) ISCRE=ISCRE+1                               SURF1ON
      IF (HX(KK,JJ).GE.HXCOL) ISCRF=ISCRF+1                             SURF1ON
      GO TO 90                                                          SURF1ON
   80 FOFPP=1.0                                                         SURF1ON
      LMNPP=LMNP+NQ                                                     SURF1ON
      IF (AC(LMNPP).LT.EM6.AND.F(IPJK).LT.EMF) FOFPP=0.0                SURF1ON
      DELTA=2.0*((1.0+AC(LMN)*(F(LMN)-1.0))*XI(I)*DELX(I)+(1.0+AC(LMNP)*SURF1ON
     1 (F(LMNP)-1.0))*XI(I+1)*DELX(I+1)*FOFP+(1.0+AC(LMNPP)*(F(LMNPP)-1.SURF1ON
     2 0))*XI(I+2)*DELX(I+2)*FOFPP)                                     SURF1ON
      DELTB=2.0*(XI(I+2)*DELX(I+2)*FOFPP+XI(I+1)*DELX(I+1)*FOFP+XI(I)   SURF1ON
     1 *DELX(I))                                                        SURF1ON
      IF (NFC.NE.2) HX(KK,JJ)=SQRT(DELTA)                               SURF1ON
      IF (NFC.EQ.2) HX(KK,JJ)=X(I+2)-SQRT(X(I+2)**2-DELTA)              SURF1ON
      IF (NFC.NE.2) HXCOL=SQRT(DELTB)-EM6                               SURF1ON
      IF (NFC.EQ.2) HXCOL=X(I+2)-SQRT(X(I+2)**2-DELTB)-EM6              SURF1ON
      IF (HX(KK,JJ).LE.0.0) ISCRE=ISCRE+1                               SURF1ON
      IF (HX(KK,JJ).GE.HXCOL) ISCRF=ISCRF+1                             SURF1ON
   90 CONTINUE                                                          SURF1ON
      GO TO 140                                                         SURF1ON
  100 LMNMM=LMNM-NQ                                                     SURF1ON
      LMNPP=LMNP+NQ                                                     SURF1ON
      FOFMM=1.0                                                         SURF1ON
      FOFPP=1.0                                                         SURF1ON
      IMM=I-3                                                           SURF1ON
      IPP=I+2                                                           SURF1ON
      IF (I.NE.3) GO TO 110                                             SURF1ON
      LMNMM=LMNM                                                        SURF1ON
      FOFMM=0.0                                                         SURF1ON
      IMM=I-2                                                           SURF1ON
      IF (AC(LMNPP).LT.EM6.AND.F(IPJK).LT.EMF) FOFPP=0.0                SURF1ON
      GO TO 130                                                         SURF1ON
  110 IF (I.NE.IM2) GO TO 120                                           SURF1ON
      LMNPP=LMNP                                                        SURF1ON
      FOFPP=0.0                                                         SURF1ON
      IPP=I+1                                                           SURF1ON
      IF (AC(LMNMM).LT.EM6.AND.F(IMJK).LT.EMF) FOFMM=0.0                SURF1ON
      GO TO 130                                                         SURF1ON
  120 IF (AC(LMNMM).LT.EM6.AND.F(IMJK).LT.EMF) FOFMM=0.0                SURF1ON
      IF (AC(LMNPP).LT.EM6.AND.F(IPJK).LT.EMF) FOFPP=0.0                SURF1ON
  130 DELTA=2.0*((1.0+AC(LMNM)*(F(LMNM)-1.0))*XI(I-1)*DELX(I-1)*FOFM+(1.SURF1ON
     1 0+AC(LMN)*(F(LMN)-1.0))*XI(I)*DELX(I)+(1.0+AC(LMNP)*(F(LMNP)-1.0)SURF1ON
     2 )*XI(I+1)*DELX(I+1)*FOFP+(1.0+AC(LMNMM)*(F(LMNMM)-1.0))*XI(I-2)  SURF1ON
     3 *DELX(I-2)*FOFMM+(1.0+AC(LMNPP)*(F(LMNPP)-1.0))*XI(I+2)*DELX(I+2)SURF1ON
     4 *FOFPP)                                                          SURF1ON
      DELTB=2.0*(XI(I-2)*DELX(I-2)*FOFMM+XI(I-1)*DELX(I-1)*FOFM+XI(I)   SURF1ON
     1 *DELX(I)+XI(I+1)*DELX(I+1)*FOFP+XI(I+2)*DELX(I+2)*FOFPP)         SURF1ON
      IF (NFC.NE.2) HX(KK,JJ)=SQRT(X(IMM)**2+DELTA)-X(IMM)              SURF1ON
      IF (NFC.EQ.2) HX(KK,JJ)=X(IPP)-SQRT(X(IPP)**2-DELTA)              SURF1ON
      IF (NFC.NE.2) HXCOL=SQRT(X(IMM)**2+DELTB)-X(IMM)-EM6              SURF1ON
      IF (NFC.EQ.2) HXCOL=X(IPP)-SQRT(X(IPP)**2-DELTB)-EM6              SURF1ON
```

```
      IF (HX(KK,JJ).LE.O.O) ISCRE=ISCRE+1
      IF (HX(KK,JJ).GE.HXCOL) ISCRF=ISCRF+1
  140 CONTINUE
  150 CONTINUE
C
      IF (K.NE.KM1) GO TO 160
      HX(3,1)=HX(2,1)
      HX(3,2)=HX(2,2)
      HX(3,3)=HX(2,3)
      GO TO 170
  160 IF (K.NE.2) GO TO 170
      HX(1,1)=HX(2,1)
      HX(1,2)=HX(2,2)
      HX(1,3)=HX(2,3)
  170 CONTINUE
C
      PHXZ=RZDEN*((HX(3,2)-HX(2,2))*DZB**2+(HX(2,2)-HX(1,2))*DZT**2)
      PHXY=RYDEN*((HX(2,3)-HX(2,2))*DYF**2+(HX(2,2)-HX(2,1))*DYBK**2)
C
C      ASSUME NEAR VERTICAL SURFACE - COLLAPSE IN AZMUTHAL DIRECTION
C
      IF (JBAR.EQ.1) GO TO 350
      ISCVE=O
      ISCVF=O
      DO 320 KK=1,3
      N=K-2+KK
      DO 310 II=1,3
      L=I-2+II
      IF (I.EQ.2) L=1+II
      LMN=NQ*(IMAX*JMAX*(N-1)+IMAX*(J-1)+(L-1))+1
      LMNM=LMN-II1
      LMNP=LMN+II1
      IF (CYL.LT.O.5) GO TO 250
      IF (J.EQ.2) GO TO 180
      IF (J.EQ.JM1) GO TO 210
      IF (I.EQ.2) GO TO 240
      GO TO 250
C
C     J=2
C
  180 IF (I.EQ.2) GO TO 200
  190 LL=L
      M=2
      IF (WF.EQ.4) M=JM1
      LMNM=NQ*(IMAX*JMAX*(N-1)+IMAX*(M-1)+(LL-1))+1
      GO TO 250
  200 IF (L.GT.1) GO TO 190
      M=JOP(J)
      LL=2
      LMN=NQ*(IMAX*JMAX*(N-1)+IMAX*(M-1)+(LL-1))+1
      LMNP=LMN-II1
      LMNM=LMN+II1
      IF (WF.EQ.4) GO TO 250
      LMNP=LMN
      LMNM=LMN-II1
      GO TO 250
C
C     J=JM1
C
  210 IF (I.EQ.2) GO TO 230
  220 LL=L
      M=JM1
      IF (WBK.EQ.4) M=2
      LMNP=NQ*(IMAX*JMAX*(N-1)+IMAX*(M-1)+(LL-1))+1
      GO TO 250
  230 IF (L.GT.1) GO TO 220
      M=JOP(J)
      LL=2
      LMN=NQ*(IMAX*JMAX*(N-1)+IMAX*(M-1)+(LL-1))+1
      LMNM=LMN+II1
      LMNP=LMN-II1
      IF (WBK.EQ.4) GO TO 250
      LMNM=LMN
      LMNP=LMN+II1
      GO TO 250
C
C      I=2
C
  240 IF (L.GT.1) GO TO 250
      LL=2
```

```
      M=JOP(J)
      LMN=NQ*(IMAX*JMAX*(N-1)+IMAX*(M-1)+(LL-1))+1
      LMNM=LMN+II1
      LMNP=LMN-II1
      IF (WF.EQ.4) GO TO 250
      LMNM=LMN-II1
      LMNP=LMN+II1
  250 CONTINUE
      FOFM=1.0
      IF (AC(LMNM).LE.EM6.AND.F(IJMK).LT.EMF) FOFM=0.0
      IF (J.EQ.2.AND.WF.NE.4.AND.F(LMN).LT.EMF) FOFM=0.0
      FOFP=1.0
      IF (AC(LMNP).LE.EM6.AND.F(IJPK).LT.EMF) FOFP=0.0
      IF (J.EQ.JM1.AND.WBK.NE.4.AND.F(LMN).LT.EMF) FOFP=0.0
      IF (J.GE.3.AND.J.LE.JM2) GO TO 270
      IF (J.EQ.2) GO TO 260
      FOFMM=1.0
      LMNMM=LMNM-II1
      IF (AC(LMNMM).LT.EM6.AND.F(IJMK).LT.EMF) FOFMM=0.0
      HY(II,KK)=(1.0+AC(LMNM)*(F(LMNM)-1.0))*DELY(J-1)*FOFM+(1.0+AC(LMN)
     1 *(F(LMN)-1.0))*DELY(J)+(1.0+AC(LMNMM)*(F(LMNMM)-1.0))*DELY(J-2)
     2 *FOFMM
      HY(II,KK)=HY(II,KK)/X(IM1)
      HYCOL=(DELY(J-2)*FOFMM+DELY(J-1)*FOFM+DELY(J))/X(IM1)-EM6
      IF (HY(II,KK).LE.0.0) ISCVE=ISCVE+1
      IF (HY(II,KK).GE.HYCOL) ISCVF=ISCVF+1
      GO TO 310
  260 FOFPP=1.0
      LMNPP=LMNP+II1
      IF (AC(LMNPP).LT.EM6.AND.F(IJPK).LT.EMF) FOFPP=0.0
      HY(II,KK)=(1.0+AC(LMN)*(F(LMN)-1.0))*DELY(J)+(1.0+AC(LMNP)*(F(LMNP
     1 )-1.0))*DELY(J+1)+(1.0+AC(LMNPP)*(F(LMNPP)-1.0))*DELY(J+2)
      HY(II,KK)=HY(II,KK)/X(IM1)
      HYCOL=(DELY(J+2)*FOFPP+DELY(J+1)*FOFP+DELY(J))/X(IM1)-EM6
      IF (HY(II,KK).LE.0.0) ISCVE=ISCVE+1
      IF (HY(II,KK).GE.HYCOL) ISCVF=ISCVF+1
      GO TO 310
  270 LMNMM=LMNM-II1
      LMNPP=LMNP+II1
      FOFMM=1.0
      FOFPP=1.0
      IF (J.NE.3) GO TO 280
      LMNMM=LMNM
      FOFMM=0.0
      IF (AC(LMNPP).LT.EM6.AND.F(IJPK).LT.EMF) FOFPP=0.0
      GO TO 300
  280 IF (J.NE.JM2) GO TO 290
      LMNPP=LMNP
      FOFPP=0.0
      IF (AC(LMNMM).LT.EM6.AND.F(IJMK).LT.EMF) FOFMM=0.0
      GO TO 300
  290 IF (AC(LMNMM).LT.EM6.AND.F(IJMK).LT.EMF) FOFMM=0.0
      IF (AC(LMNPP).LT.EM6.AND.F(IJPK).LT.EMF) FOFPP=0.0
  300 HY(II,KK)=(1.0+AC(LMNM)*(F(LMNM)-1.0))*DELY(J-1)*FOFM+(1.0+AC(LMN)
     1 *(F(LMN)-1.0))*DELY(J)+(1.0+AC(LMNP)*(F(LMNP)-1.0))*DELY(J+1)
     2 *FOFP+(1.0+AC(LMNMM)*(F(LMNMM)-1.0))*DELY(J-2)*FOFMM+(1.0+AC
     3 (LMNPP)*(F(LMNPP)-1.0))*DELY(J+2)*FOFPP
      HY(II,KK)=HY(II,KK)/X(IM1)
      HYCOL=(DELY(J-2)*FOFMM+DELY(J-1)*FOFM+DELY(J)+DELY(J+1)*FOFP+DELY
     1 (J+2)*FOFPP)/X(IM1)-EM6
      IF (HY(II,KK).LE.0.0) ISCVE=ISCVE+1
      IF (HY(II,KK).GE.HYCOL) ISCVF=ISCVF+1
  310 CONTINUE
  320 CONTINUE
C
      IF (K.NE.KM1) GO TO 330
      HY(1,3)=HY(1,2)
      HY(2,3)=HY(2,2)
      HY(3,3)=HY(3,2)
      GO TO 340
  330 IF (K.NE.2) GO TO 340
      HY(1,1)=HY(1,2)
      HY(2,1)=HY(2,2)
      HY(3,1)=HY(3,2)
  340 CONTINUE
C
      PHYX=RXDEN*((HY(3,2)-HY(2,2))*DXL**2+(HY(2,2)-HY(1,2))*DXR**2)
      PHYZ=RZDEN*((HY(2,3)-HY(2,2))*DZB**2+(HY(2,2)-HY(2,1))*DZT**2)
      IF (I.EQ.2) PHYX=RXDEN*((HY(2,2)-HY(3,2))*(DXL**2)+(HY(2.2)-HY(1,2
     1 ))*DXR*(DXR+2.0*DXL))
```

```
      IF (I.EQ.2) PHYZ=RZDEN*((HY(1,3)-HY(1,2))*(DZB**2)+(HY(1,2)-HY(1,1
     1 ))*(DZT**2))
      PHYX=XI(I)*PHYX
      PHYZ=XI(I)*PHYZ
      GO TO 360
  350 PHYX=1.0E10
      PHYZ=1.0E10
  360 CONTINUE
C
C      ASSUME NEAR HORIZONTAL SURFACE - COLLAPSE IN AXIAL DIRECTION
C
      ISCHE=0
      DO 520 II=1,3
      L=I-2+II
      IF (I.EQ.2) L=1+II
      DO 510 JJ=1,3
      M=J-2+JJ
      LMN=NQ*(IMAX*JMAX*(K-1)+IMAX*(M-1)+(L-1))+1
      LMNM=LMN-II2
      LMNP=LMN+II2
      IF (CYL.LT.0.5) GO TO 450
      IF (M.EQ.1) GO TO 370
      IF (M.EQ.JMAX) GO TO 400
      IF (L.EQ.1) GO TO 430
      GO TO 450
C
C     M=1
C
  370 IF (L.EQ.1) GO TO 390
      LL=L
      MM=2
      IF (WF.EQ.4) MM=JM1
  380 LMN=NQ*(IMAX*JMAX*(K-1)+IMAX*(MM-1)+(LL-1))+1
      GO TO 440
  390 LL=2
      MM=JM2
      IF (WF.EQ.4) MM=JOP(J)+1
      GO TO 380
C
C     M=JMAX
C
  400 IF (L.EQ.1) GO TO 420
      LL=L
      MM=JM1
      IF (WBK.EQ.4) M=2
  410 LMN=NQ*(IMAX*JMAX*(K-1)+IMAX*(MM-1)+(LL-1))+1
      GO TO 440
  420 LL=2
      MM=3
      IF (WBK.EQ.4) MM=JOP(JM1)-1
      GO TO 410
C
C      I=2
C
  430 LL=2
      IF (JJ.EQ.1) MM=JOP(J+1)
      IF (JJ.EQ.2) MM=JOP(J)
      IF (JJ.EQ.3) MM=JOP(J-1)
      GO TO 380
  440 LMNM=LMN-II2
      LMNP=LMN+II2
  450 CONTINUE
      FOFM=1.0
      IF (AC(LMNM).LE.EM6.AND.F(IJKM).LT.EMF) FOFM=0.0
      FOFP=1.0
      IF (AC(LMNP).LE.EM6.AND.F(IJKP).LT.EMF) FOFP=0.0
      IF (K.GE.3.AND.K.LE.KM2) GO TO 470
      IF (K.EQ.2) GO TO 460
      FOFMM=1.0
      LMNMM=LMNM-II2
      IF (AC(LMNMM).LT.EM6.AND.F(IJKM).LT.EMF) FOFMM=0.0
      HZ(II,JJ)=(1.0+AC(LMNM)*(F(LMNM)-1.0))*DELZ(K-1)*FOFM+(1.0+AC(LMN)
     1 *(F(LMN)-1.0))*DELZ(K)+(1.0+AC(LMNMM)*(F(LMNMM)-1.0))*DELZ(K-2)
     2 *FOFMM
      HZCOL=DELZ(K-2)*FOFMM+DELZ(K-1)*FOFM+DELZ(K)-EM6
      IF (HZ(II,JJ).LE.0.0) ISCHE=ISCHE+1
      IF (HZ(II,JJ).GE.HZCOL) ISCHF=ISCHF+1
      GO TO 510
  460 FOFPP=1.0
      LMNPP=LMNP+II2
```

```
      IF (AC(LMNPP).LT.EM6.AND.F(IJKP).LT.EMF) FOFPP=0.0
      HZ(II,JJ)=(1.0+AC(LMN)*(F(LMN)-1.0))*DELZ(K)+(1.0+AC(LMNP)*(F(LMNP
     1 )-1.0))*DELZ(K+1)+(1.0+AC(LMNPP)*(F(LMNPP)-1.0))*DELZ(K+2)
      HZCOL=DELZ(K+2)*FOFPP+DELZ(K+1)*FOFP+DELZ(K)-EM6
      IF (HZ(II,JJ).LE.0.0) ISCHE=ISCHE+1
      IF (HZ(II,JJ).GE.HZCOL) ISCHF=ISCHF+1
      GO TO 510
  470 LMNMM=LMNM-II2
      LMNPP=LMNP+II2
      FOFMM=1.0
      FOFPP=1.0
      IF (K.NE.3) GO TO 480
      LMNMM=LMNM
      FOFMM=0.0
      IF (AC(LMNPP).LT.EM6.AND.F(IJKP).LT.EMF) FOFPP=0.0
      GO TO 500
  480 IF (K.NE.KM2) GO TO 490
      LMNPP=LMNP
      FOFPP=0.0
      IF (AC(LMNMM).LT.EM6.AND.F(IJKM).LT.EMF) FOFMM=0.0
      GO TO 500
  490 IF (AC(LMNMM).LT.EM6.AND.F(IJKM).LT.EMF) FOFMM=0.0
      IF (AC(LMNPP).LT.EM6.AND.F(IJKP).LT.EMF) FOFPP=0.0
  500 HZ(II,JJ)=(1.0+AC(LMNM)*(F(LMNM)-1.0))*DELZ(K-1)*FOFM+(1.0+AC(LMN)
     1 *(F(LMN)-1.0))*DELZ(K)+(1.0+AC(LMNP)*(F(LMNP)-1.0))*DELZ(K+1)
     2 *FOFP+(1.0+AC(LMNMM)*(F(LMNMM)-1.0))*DELZ(K-2)*FOFMM+(1.0+AC
     3 (LMNPP)*(F(LMNPP)-1.0))*DELZ(K+2)*FOFPP
      HZCOL=DELZ(K-2)*FOFMM+DELZ(K-1)*FOFM+DELZ(K)+DELZ(K+1)*FOFP+DELZ(K
     1 +2)*FOFPP-EM6
      IF (HZ(II,JJ).LE.0.0) ISCHE=ISCHE+1
      IF (HZ(II,JJ).GE.HZCOL) ISCHF=ISCHF+1
  510 CONTINUE
  520 CONTINUE
C
      PHZX=RXDEN*((HZ(3,2)-HZ(2,2))*DXL**2+(HZ(2,2)-HZ(1,2))*DXR**2)
      PHZY=RYDEN*((HZ(2,3)-HZ(2,2))*DYF**2+(HZ(2,2)-HZ(2,1))*DYBK**2)
      IF (I.EQ.2) PHZX=RXDEN*((HZ(2,2)-HZ(3,2))*(DXL**2)+(HZ(2,2)-HZ(1,2
     1 ))*(DXR+2.0*DXL)*DXR)
      IF (I.EQ.2) PHZY=RYDEN*((HZ(1,3)-HZ(1,2))*(DYF**2)+(HZ(1,2)-HZ(1,1
     1 ))*(DYBK**2))
C
      IF (ISCR+ISCV+ISCH.EQ.3) GO TO 720
      QX=PHXY**2+PHXZ**2
      QY=PHYX**2+PHYZ**2
      QZ=PHZX**2+PHZY**2
      IF (QX+QY+QZ.GT.1.0E-10) GO TO 530
      PS(IJK)=0.0
      GO TO 1070
C
  530 IF (QX.LT.QY.AND.QX.LT.QZ) GO TO 540
      IF (QY.LT.QX.AND.QY.LT.QZ) GO TO 600
      IF (QZ.LT.QX.AND.QZ.LT.QY) GO TO 660
C
  540 NFSLP=1
      IF (F(IPJK).GT.F(IMJK)) NFSLP=2
      IF (NFSLP.EQ.NF(IJK)) GO TO 720
      IF (NFCAL.EQ.2) GO TO 590
      IF (NFCAL.EQ.3) GO TO 550
      GO TO 720
C      CONFLICT
  550 IF (BETA(IMJK).LT.0.0.OR.BETA(IPJK).LT.0.0) GO TO 590
C
      INB=0
      NFREF=0
      DO 560 JJ=1,3
      DO 560 II=1,3
      N=K-1
      M=J-2+JJ
      L=I-2+II
      LMN=NQ*(II5*(N-1)+IMAX*(M-1)+(L-1))+1
      IF (NF(LMN).LT.1.OR.NF(LMN).GT.6) GO TO 560
      NFNB=NF(LMN)
      IF (NFREF.EQ.0) NFREF=NFNB
      IF (NFNB.NE.NFREF) INB=1
  560 CONTINUE
      DO 570 KK=2,3
      DO 570 II=1,3
      N=K-2+KK
      M=J-1
      L=I-2+II
```

```
      LMN=NQ*(II5*(N-1)+IMAX*(M-1)+(L-1))+1
      IF (NF(LMN).LT.1.OR.NF(LMN).GT.6) GO TO 570
      NFNB=NF(LMN)
      IF (NFREF.EQ.0) NFREF=NFNB
      IF (NFNB.NE.NFREF) INB=1
  570 CONTINUE
C
      IF (NF(IMJK).LT.1.OR.NF(IMJK).GT.6) GO TO 580
      NFNB=NF(IMJK)
      IF (NFREF.EQ.0) NFREF=NFNB
      IF (NFNB.NE.NFREF) INB=1
  580 CONTINUE
C
      IF (NFREF.EQ.0) GO TO 720
C
      IF (INB.EQ.1) GO TO 720
C
      IF (NFREF.NE.NFSLP) GO TO 720
C
  590 NF(IJK)=NFSLP
      GO TO 720
C
  600 NFSLP=3
      IF (F(IJPK).GT.F(IJMK)) NFSLP=4
      IF (NFSLP.EQ.NF(IJK)) GO TO 720
      IF (NFCAL.EQ.2) GO TO 650
      IF (NFCAL.EQ.3) GO TO 610
      GO TO 720
C      CONFLICT
  610 IF (BETA(IJMK).LT.0.0.OR.BETA(IJPK).LT.0.0) GO TO 650
C
      INB=0
      NFREF=0
      DO 620 JJ=1,3
      DO 620 II=1,3
      N=K-1
      M=J-2+JJ
      L=I-2+II
      LMN=NQ*(II5*(N-1)+IMAX*(M-1)+(L-1))+1
      IF (NF(LMN).LT.1.OR.NF(LMN).GT.6) GO TO 620
      NFNB=NF(LMN)
      IF (NFREF.EQ.0) NFREF=NFNB
      IF (NFNB.NE.NFREF) INB=1
  620 CONTINUE
      DO 630 KK=2,3
      DO 630 II=1,3
      N=K-2+KK
      M=J-1
      L=I-2+II
      LMN=NQ*(II5*(N-1)+IMAX*(M-1)+(L-1))+1
      IF (NF(LMN).LT.1.OR.NF(LMN).GT.6) GO TO 630
      NFNB=NF(LMN)
      IF (NFREF.EQ.0) NFREF=NFNB
      IF (NFNB.NE.NFREF) INB=1
  630 CONTINUE
C
      IF (NF(IMJK).LT.1.OR.NF(IMJK).GT.6) GO TO 640
      NFNB=NF(IMJK)
      IF (NFREF.EQ.0) NFREF=NFNB
      IF (NFNB.NE.NFREF) INB=1
  640 CONTINUE
C
      IF (NFREF.EQ.0) GO TO 720
C
      IF (INB.EQ.1) GO TO 720
C
      IF (NFREF.NE.NFSLP) GO TO 720
C
  650 NF(IJK)=NFSLP
      GO TO 720
C
  660 NFSLP=5
      IF (F(IJKP).GT.F(IJKM)) NFSLP=6
      IF (NFSLP.EQ.NF(IJK)) GO TO 720
      IF (NFCAL.EQ.2) GO TO 710
      IF (NFCAL.EQ.3) GO TO 670
      GO TO 720
C     CONFLICT
  670 IF (BETA(IJKM).LT.0.0.OR.BETA(IJKP).LT.0.0) GO TO 710
C
```

```
      INB=0
      NFREF=0
      DO 680 JJ=1,3
      DO 680 II=1,3
      N=K-1
      M=J-2+JJ
      L=I-2+II
      LMN=NQ*(II5*(N-1)+IMAX*(M-1)+(L-1))+1
      IF (NF(LMN).LT.1.OR.NF(LMN).GT.6) GO TO 680
      NFNB=NF(LMN)
      IF (NFREF.EQ.0) NFREF=NFNB
      IF (NFNB.NE.NFREF) INB=1
  680 CONTINUE
      DO 690 KK=2,3
      DO 690 II=1,3
      N=K-2+KK
      M=J-1
      L=I-2+II
      LMN=NQ*(II5*(N-1)+IMAX*(M-1)+(L-1))+1
      IF (NF(LMN).LT.1.OR.NF(LMN).GT.6) GO TO 690
      NFNB=NF(LMN)
      IF (NFREF.EQ.0) NFREF=NFNB
      IF (NFNB.NE.NFREF) INB=1
  690 CONTINUE
C
      IF (NF(IMJK).LT.1.OR.NF(IMJK).GT.6) GO TO 700
      NFNB=NF(IMJK)
      IF (NFREF.EQ.0) NFREF=NFNB
      IF (NFNB.NE.NFREF) INB=1
  700 CONTINUE
C
      IF (NFREF.EQ.0) GO TO 720
C
      IF (INB.EQ.1) GO TO 720
C
      IF (NFREF.NE.NFSLP) GO TO 720
C
  710 NF(IJK)=NFSLP
      GO TO 720
C
  720 CONTINUE
C
      NFS(IJK)=NFSLP
      NFF=NF(IJK)
      CTH=1.0
      STH=0.0
      FLGE=1.0
      FLGW=1.0
      FLGN=1.0
      FLGS=1.0
      GO TO (730,730,790,790,880,880), NFF
  730 CONTINUE
      RDEW=RRI(I)*RDY(J)
      RDNS=RDZ(K)
      IF (CYL.LT.0.5) GO TO 760
      HXE=(DTH*DTHMA*RDTHS/DTHP)*HX(2,3)+(DTHE*DTHMA*RPDTH)*HX(2,2)-
     1 (DTHE*DTH*RDTHS/DTHM)*HX(2,1)
      HXW=(DTH*DTHPA*RDTHS/DTHM)*HX(2,1)+(DTHW*DTHPA*RPDTH)*HX(2,2)-
     1 (DTHW*DTH*RDTHS/DTHP)*HX(2,3)
      TH=0.5*DELY(J)*RX(IM1)
      CTH=COS(TH)
      STH=SIN(TH)
      RDEW=1.0/(XI(I)*2.0*STH)
      IF (NFF.EQ.2) GO TO 740
      IF (I.GT.3) RSURF=X(I-3)+HX(2,2)
      IF (I.GT.3) RSURFE=X(I-3)+HXE
      IF (I.GT.3) RSURFW=X(I-3)+HXW
      IF (I.EQ.3) RSURF=X(I-2)+HX(2,2)
      IF (I.EQ.3) RSURFE=X(I-2)+HXE
      IF (I.EQ.3) RSURFW=X(I-2)+HXW
      IF (I.EQ.2) RSURF=X(I-1)+HX(2,2)
      IF (I.EQ.2) RSURF=AMAX1(RSURF,0.1*DELX(2))
      IF (I.EQ.2) RSURFE=X(I-1)+HXE
      IF (I.EQ.2) RSURFE=AMAX1(RSURFE,0.1*DELX(2))
      IF (I.EQ.2) RSURFW=X(I-1)+HXW
      IF (I.EQ.2) RSURFW=AMAX1(RSURFW,0.1*DELX(2))
      GO TO 750
  740 IF (I.LT.IM2) RSURF=X(I+2)-HX(2,2)
      IF (I.LT.IM2) RSURFE=X(I+2)-HXE
      IF (I.LT.IM2) RSURFW=X(I+2)-HXW
```

```
      IF (I.EQ.IM2) RSURF=X(I+1)-HX(2,2)
      IF (I.EQ.IM2) RSURFE=X(I+1)-HXE
      IF (I.EQ.IM2) RSURFW=X(I+1)-HXW
      IF (I.EQ.IM1) RSURF=X(I)-HX(2,2)
      IF (I.EQ.IM1) RSURFE=X(I)-HXE
      IF (I.EQ.IM1) RSURFW=X(I)-HXW
  750 RDEW=1.0/(RSURF*2.0*STH)
      IF (NFF.EQ.2) STH=-STH
  760 CONTINUE
      AFE=ABK(IJK)
      IF (CYL.EQ.1.0.AND.AFE.LT.EM6) AFE=1.0
      AFW=ABK(IJMK)
      IF (CYL.EQ.1.0.AND.AFW.LT.EM6) AFW=1.0
      AFN=AT(IJK)
      AFS=AT(IJKM)
      AFLOOR=AR(IMJK)
      IF (NFF.EQ.2) AFLOOR=AR(IJK)
      IF (NOWALL.EQ.1) AFLOOR=1.0
C  **   USE FLG- TO INDICATE WALL ADHESION AT CELL BOTTOM
      IF (AFE.GT.EM6.AND.F(IJPK).LT.EMF) FLGE=AFLOOR
      IF (AFW.GT.EM6.AND.F(IJMK).LT.EMF) FLGW=AFLOOR
      IF (AFN.GT.EM6.AND.F(IJKP).LT.EMF) FLGN=AFLOOR
      IF (AFS.GT.EM6.AND.F(IJKM).LT.EMF) FLGS=AFLOOR
      PHXZB=(HX(2,2)-HX(1,2))/DZB
      PHXZT=(HX(3,2)-HX(2,2))/DZT
      IF (CYL.LT.0.5) GO TO 990
      DHNA=0.5*(PHXZT+PHXZB)
      FCYL=0.5*DELZ(K)*DHNA*RXI(I)
      IF (NFF.EQ.2) FCYL=-FCYL
      FLGN=FLGN*(1.0+FCYL)
      FLGS=FLGS*(1.0-FCYL)
C
C       BUILDING FOUR SETS OF SIX POINT COEFFICIENTS
C
C       COORDINATES ARE SUCH THAT (E,W) GOES WITH (J+1,J-1)
C       COORDINATES ARE SUCH THAT (N,S) GOES WITH (K+1,K-1)
C
C       Z DERIVATIVE ON E FACE (FOR DHNE)
C
      ANE(3,3)=(1.0+3.0*FRTHP*RATZM)/(3.0*DZP)
      ANE(2,3)=-(1.0+ZP)*ANE(3,3)+(FRTHP/DZM)
      ANE(1,3)=ZP*ANE(3,3)-(FRTHP/DZM)
      ANE(3,2)=-ANE(3,3)+ZM*RDZS
      ANE(2,2)=(1.0+ZP)*ANE(3,3)-(FRTHP/DZM)+DZD*RPDZ
      ANE(1,2)=-ZP*ANE(3,3)+(FRTHP/DZM)-ZP*RDZS
      ANE(3,1)=0.0
      ANE(2,1)=0.0
      ANE(1,1)=0.0
C
C      Z DERIVATIVE ON W FACE (FOR DHNW)
C
      ANW(1,1)=-(1.0+3.0*FRTHM*RATZP)/(3.0*DZM)
      ANW(2,1)=-(1.0+ZM)*ANW(1,1)-(FRTHM/DZP)
      ANW(3,1)=ZM*ANW(1,1)+(FRTHM/DZP)
      ANW(1,2)=-ANW(1,1)-ZP*RDZS
      ANW(2,2)=(1.0+ZM)*ANW(1,1)+(FRTHM/DZP)+DZD*RPDZ
      ANW(3,2)=-ZM*ANW(1,1)-(FRTHM/DZP)+ZM*RDZS
      ANW(1,3)=0.0
      ANW(2,3)=0.0
      ANW(3,3)=0.0
C
C       THETA DERIVATIVE ON N FACE  (FOR DHEN)
C
      AEN(3,3)=(1.0+3.0*FRZP*RATTHM)/(3.0*DTHP)
      AEN(3,2)=-(1.0+THP)*AEN(3,3)+(FRZP/DTHM)
      AEN(3,1)=THP*AEN(3,3)-(FRZP/DTHM)
      AEN(2,3)=-AEN(3,3)+THM*RDTHS
      AEN(2,2)=(1.0+THP)*AEN(3,3)-(FRZP/DTHM)+DTHD*RPDTH
      AEN(2,1)=-THP*AEN(3,3)+(FRZP/DTHM)-THP*RDTHS
      AEN(1,3)=0.0
      AEN(1,2)=0.0
      AEN(1,1)=0.0
C
C       THETA DERIVATIVE ON S FACE (FOR DHES)
C
      AES(1,1)=-(1.0+3.0*FRZM*RATTHP)/(3.0*DTHM)
      AES(1,2)=-(1.0+THM)*AES(1,1)-(FRZM/DTHP)
      AES(1,3)=THM*AES(1,1)+(FRZM/DTHP)
      AES(2,1)=-AES(1,1)-THP*RDTHS
      AES(2,2)=(1.0+THM)*AES(1,1)+(FRZM/DTHP)+DTHD*RPDTH
```

```
      AES(2,3)=-THM*AES(1,1)-(FRZM/DTHP)+THM*RDTHS
      AES(3,1)=0.0
      AES(3,2)=0.0
      AES(3,3)=0.0
C
C       BUILDING FOUR SETS OF SEVEN POINT COEFFICIENTS
C
C       THETA DERIVATIVE ON E FACE (FOR DHEE)
C
      CEE(3,3)=RPDTH*RDTHS*RATZM*(DTHDE*(DTHMA**2)+(DTHMA+DTH)*(DTHE**2)
     1 )/(3.0*DTHP)
      CEE(2,3)=-(1.0+ZP)*CEE(3,3)+(DTHMA+DTH)*RDTHS/DTHP
      CEE(1,3)=ZP*CEE(3,3)
      CEE(3,2)=-CEE(3,3)
      CEE(2,2)=(1.0+ZP)*CEE(3,3)+(DTHDE-DTHM)*RPDTH
      CEE(1,2)=-ZP*CEE(3,3)
      CEE(3,1)=0.0
      CEE(2,1)=-DTHDE*RDTHS/DTHM
      CEE(1,1)=0.0
C
C      THETA DERIVATIVE ON W FACE  (FOR DHEW)
C
      CEW(1,1)=-RPDTH*RDTHS*RATZP*(DTHDW*(DTHPA**2)+(DTHPA+DTH)*(DTHW**2
     1 ))/(3.0*DTHM)
      CEW(2,1)=-(1.0+ZM)*CEW(1,1)-(DTHPA+DTH)*RDTHS/DTHM
      CEW(3,1)=ZM*CEW(1,1)
      CEW(1,2)=-CEW(1,1)
      CEW(2,2)=(1.0+ZM)*CEW(1,1)-(DTHDW-DTHP)*RPDTH
      CEW(3,2)=-ZM*CEW(1,1)
      CEW(1,3)=0.0
      CEW(2,3)=DTHDW*RDTHS/DTHP
      CEW(3,3)=0.0
C
C      Z DERIVATIVE ON N FACE  (FOR DHNN)
C
      CNN(3,3)=RPDZ*RDZS*RATTHM*(DZDN*(DZMA**2)+(DZMA+DZ)*(DZN**2))/(3.0
     1 *DZP)
      CNN(3,2)=-(1.0+THP)*CNN(3,3)+(DZMA+DZ)*RDZS/DZP
      CNN(3,1)=THP*CNN(3,3)
      CNN(2,3)=-CNN(3,3)
      CNN(2,2)=(1.0+THP)*CNN(3,3)+(DZDN-DZM)*RPDZ
      CNN(2,1)=-THP*CNN(3,3)
      CNN(1,3)=0.0
      CNN(1,2)=-DZDN*RDZS/DZM
      CNN(1,1)=0.0
C
C       Z DERIVATIVE ON S FACE  (FOR DHNS)
C
      CNS(1,1)=-RPDZ*RDZS*RATTHP*(DZDS*(DZPA**2)+(DZPA+DZ)*(DZS**2))/(3.
     1 0*DZM)
      CNS(1,2)=-(1.0+THM)*CNS(1,1)-(DZPA+DZ)*RDZS/DZM
      CNS(1,3)=THM*CNS(1,1)
      CNS(2,1)=-CNS(1,1)
      CNS(2,2)=(1.0+THM)*CNS(1,1)-(DZDS-DZP)*RPDZ
      CNS(2,3)=-THM*CNS(1,1)
      CNS(3,1)=0.0
      CNS(3,2)=DZDS*RDZS/DZP
      CNS(3,3)=0.0
C
C      OBTAIN RADIAL COORDINATES FOR CONVERTING THETA
C      DERIVATIVES TO GRADIENTS
C
      RRSURF=1.0/RSURF
      RRSURFE=1.0/RSURFE
      RRSURFW=1.0/RSURFW
C
C       INITIALIZE GRADIENTS FOR USE IN DO LOOPS
C
      DHEE=0.0
      DHEW=0.0
      DHEN=0.0
      DHES=0.0
      DHNE=0.0
      DHNW=0.0
      DHNN=0.0
      DHNS=0.0
C
C       NESTED DO LOOPS FOR CONSTRUCTION OF GRADIENTS
C
C       SIX OR SEVEN POINT FORMULAS FOR GRADIENTS IMPOSED BY
```

```
C      NINE POINT SUMMATIONS (3 OR 2 TERMS HAVE BEEN SET TO ZERO)
C
      DO 780 KK=1,3
      DO 770 JJ=1,3
      DHNE=DHNE+ANE(KK,JJ)*HX(KK,JJ)
      DHNW=DHNW+ANW(KK,JJ)*HX(KK,JJ)
      DHEN=DHEN+AEN(KK,JJ)*HX(KK,JJ)*RRSURF
      DHES=DHES+AES(KK,JJ)*HX(KK,JJ)*RRSURF
      DHEE=DHEE+CEE(KK,JJ)*HX(KK,JJ)*RRSURFE
      DHEW=DHEW+CEW(KK,JJ)*HX(KK,JJ)*RRSURFW
      DHNN=DHNN+CNN(KK,JJ)*HX(KK,JJ)
      DHNS=DHNS+CNS(KK,JJ)*HX(KK,JJ)
  770 CONTINUE
  780 CONTINUE
      GO TO 990
  790 CONTINUE
      RDEW=RDX(I)
      RDNS=RDZ(K)
      PHINW=0.25*(HY(1,3)+HY(2,3)+HY(1,2)+HY(2,2))
      PHINE=0.25*(HY(2,3)+HY(3,3)+HY(2,2)+HY(3,2))
      PHISW=0.25*(HY(1,2)+HY(2,2)+HY(1,1)+HY(2,1))
      PHISE=0.25*(HY(2,2)+HY(3,2)+HY(2,1)+HY(3,1))
      PHIAVE=0.5*(PHINE+PHISE)
      PHIAVW=0.5*(PHINW+PHISW)
      PHIAVN=0.5*(PHINW+PHINE)
      PHIAVS=0.5*(PHISW+PHISE)
      IF (J.EQ.2) PHIM=(YJ(2)+DELY(1))/X(IM1)
      IF (J.NE.2) PHIM=(YJ(J)-Y(J-2))/X(IM1)
C ******  Y=2 SPECIAL CASE ********
      DELE=PHIAVE-PHIM
      DELW=PHIAVW-PHIM
      IF (I.EQ.2) DELW=0.0
      DELN=PHIAVN-PHIM
      DELS=PHIAVS-PHIM
      ACOSE=COS(DELE)
      ASINE=SIN(DELE)
      ACOSW=COS(DELW)
      ASINW=SIN(DELW)
      ACOSN=COS(DELN)
      ASINN=SIN(DELN)
      ACOSS=COS(DELS)
      ASINS=SIN(DELS)
      AFE=AR(IJK)
      AFW=AR(IMJK)
      IF (CYL.GT.0.5.AND.I.EQ.2) AFW=1.0
      AFN=AT(IJK)
      AFS=AT(IJKM)
      AFLOOR=ABK(IJMK)
      IF (NFF.EQ.4) AFLOOR=ABK(IJK)
      IF (NOWALL.EQ.1) AFLOOR=1.0
      IF (CYL.EQ.1.0.AND.AFLOOR.LT.EM6) AFLOOR=1.0
      IF (AFE.GT.EM6.AND.F(IPJK).LT.EMF) FLGE=AFLOOR
      IF (AFW.GT.EM6.AND.F(IMJK).LT.EMF) FLGW=AFLOOR
      IF (AFN.GT.EM6.AND.F(IJKP).LT.EMF) FLGN=AFLOOR
      IF (AFS.GT.EM6.AND.F(IJKM).LT.EMF) FLGS=AFLOOR
C
C      BUILDING FOUR SETS OF SIX POINT COEFFICIENTS
C
C      COORDINATES ARE SUCH THAT (E,W) GOES WITH (I+1,I-1)
C      COORDINATES ARE SUCH THAT (N,S) GOES WITH (K+1,K-1)
C
C      Z DERIVATIVE ON E FACE (FOR DHNE)
C
      ANE(3,3)=(1.0+3.0*FRRP*RATZM)/(3.0*DZP)
      ANE(2,3)=-(1.0+ZP)*ANE(3,3)+(FRRP/DZM)
      ANE(1,3)=ZP*ANE(3,3)-(FRRP/DZM)
      ANE(3,2)=-ANE(3,3)+ZM*RDZS
      ANE(2,2)=(1.0+ZP)*ANE(3,3)-(FRRP/DZM)+DZD*RPDZ
      ANE(1,2)=-ZP*ANE(3,3)+(FRRP/DZM)-ZP*RDZS
      ANE(3,1)=0.0
      ANE(2,1)=0.0
      ANE(1,1)=0.0
      IF (I.NE.2) GO TO 800
      ANE(3,1)=ANE(3,2)
      ANE(2,1)=ANE(2,2)
      ANE(1,1)=ANE(1,2)
      ANE(3,2)=ANE(3,3)
      ANE(2,2)=ANE(2,3)
      ANE(1,2)=ANE(1,3)
      ANE(3,3)=0.0
```

```
      ANE(2,3)=0.0
      ANE(1,3)=0.0
  800 CONTINUE
C
C     Z DERIVATIVE ON W FACE (FOR DHNW)
C
      ANW(1,1)=-(1.0+3.0*FRRM*RATZP)/(3.0*DZM)
      ANW(2,1)=-(1.0+ZM)*ANW(1,1)-(FRRM/DZP)
      ANW(3,1)=ZM*ANW(1,1)+(FRRM/DZP)
      ANW(1,2)=-ANW(1,1)-ZP*RDZS
      ANW(2,2)=(1.0+ZM)*ANW(1,1)+(FRRM/DZP)+DZD*RPDZ
      ANW(3,2)=-ZM*ANW(1,1)-(FRRM/DZP)+ZM*RDZS
      ANW(1,3)=0.0
      ANW(2,3)=0.0
      ANW(3,3)=0.0
C
C      R DERIVATIVE ON N FACE  (FOR DHEN)
C
      AEN(3,3)=(1.0+3.0*FRZP*RATRM)/(3.0*DRP)
      IF (I.EQ.2) AEN(3,3)=-(1.0+RM+3.0*FRZP*RATRM)/(3.0*DRP)
      AEN(3,2)=-(1.0+RP)*AEN(3,3)+(FRZP/DRM)
      AEN(3,1)=RP*AEN(3,3)-(FRZP/DRM)
      AEN(2,3)=-AEN(3,3)+RM*RDRS
      AEN(2,2)=(1.0+RP)*AEN(3,3)-(FRZP/DRM)+DRD*RPDR
      AEN(2,1)=-RP*AEN(3,3)+(FRZP/DRM)-RP*RDRS
      AEN(1,3)=0.0
      AEN(1,2)=0.0
      AEN(1,1)=0.0
      IF (I.NE.2) GO TO 810
      AEN(2,3)=AEN(2,3)-2.0*RM*RDRS
      AEN(2,2)=AEN(2,2)+(2.0/DRP)
      AEN(2,1)=AEN(2,1)-(2.0*RDRS)
  810 CONTINUE
C
C     R DERIVATIVE ON S FACE (FOR DHES)
C
      AES(1,1)=-(1.0+3.0*FRZM*RATRP)/(3.0*DRM)
      AES(1,2)=-(1.0+RM)*AES(1,1)-(FRZM/DRP)
      AES(1,3)=RM*AES(1,1)+(FRZM/DRP)
      AES(2,1)=-AES(1,1)-RP*RDRS
      AES(2,2)=(1.0+RM)*AES(1,1)+(FRZM/DRP)+DRD*RPDR
      AES(2,3)=-RM*AES(1,1)-(FRZM/DRP)+RM*RDRS
      AES(3,1)=0.0
      AES(3,2)=0.0
      AES(3,3)=0.0
      IF (I.NE.2) GO TO 820
      AES(1,3)=(1.0+RM-3.0*FRZM*RATRM)/(3.0*DRP)
      AES(1,2)=-(1.0+RP)*AES(1,3)+(FRZM/DRM)
      AES(1,1)=RP*AES(1,3)-(FRZM/DRM)
      AES(2,3)=-AES(1,3)-(RM*RDRS)
      AES(2,2)=(1.0+RP)*AES(1,3)-(FRZM/DRM)+(1.0/DRP)+(1.0/DRM)
      AES(2,1)=-RP*AES(1,3)+(FRZM/DRM)-(1.0/DRM)-(RDRS)
  820 CONTINUE
C
C      BUILDING FOUR SETS OF SEVEN POINT COEFFICIENTS
C
C      R DERIVATIVE ON E FACE (FOR DHEE)
C
       CEE(3,3)=RPDR*RDRS*RATZM*(DRDO*(DRMA**2)+(DRMA+DR)*(DRO**2))/(3.0
     1 *DRP)
       CEE(2,3)=-(1.0+ZP)*CEE(3,3)+(DRMA+DR)*RDRS/DRP
       CEE(1,3)=ZP*CEE(3,3)
       CEE(3,2)=-CEE(3,3)
       CEE(2,2)=(1.0+ZP)*CEE(3,3)+(DRDO-DRM)*RPDR
       CEE(1,2)=-ZP*CEE(3,3)
       CEE(3,1)=0.0
       CEE(2,1)=-DRDO*RDRS/DRM
       CEE(1,1)=0.0
       IF (I.NE.2) GO TO 830
       CEE(1,1)=-ZP*CEE(3,3)
       CEE(2,1)=-(1.0+ZM)*CEE(1,1)-(DRPA+DR)*RDRS/DRM
       CEE(3,1)=ZM*CEE(1,1)
       CEE(1,2)=-CEE(1,1)
       CEE(2,2)=(1.0+ZM)*CEE(1,1)-(DRDI-DRP)*RPDR
       CEE(3,2)=-ZM*CEE(1,1)
       CEE(1,3)=0.0
       CEE(2,3)=DRDI*RDRS/DRP
       CEE(3,3)=0.0
  830 CONTINUE
C
```

```
C      R DERIVATIVE ON W FACE  (FOR DHEW)
C
       CEW(1,1)=-RPDR*RDRS*RATZP*(DRDI*(DRPA**2)+(DRPA+DR)*(DRI**2))/(3.0
      1 *DRM)
       CEW(2,1)=-(1.0+ZM)*CEW(1,1)-(DRPA+DR)*RDRS/DRM
       CEW(3,1)=ZM*CEW(1,1)
       CEW(1,2)=-CEW(1,1)
       CEW(2,2)=(1.0+ZM)*CEW(1,1)-(DRDI-DRP)*RPDR
       CEW(3,2)=-ZM*CEW(1,1)
       CEW(1,3)=0.0
       CEW(2,3)=DRDI*RDRS/DRP
       CEW(3,3)=0.0
C
C      Z DERIVATIVE ON N FACE  (FOR DHNN)
C
       CNN(3,3)=RPDZ*RDZS*RATRM*(DZDN*(DZMA**2)+(DZMA+DZ)*(DZN**2))/(3.0
      1 *DZP)
       CNN(3,2)=-(1.0+RP)*CNN(3,3)+(DZMA+DZ)*RDZS/DZP
       CNN(3,1)=RP*CNN(3,3)
       CNN(2,3)=-CNN(3,3)
       CNN(2,2)=(1.0+RP)*CNN(3,3)+(DZDN-DZM)*RPDZ
       CNN(2,1)=-RP*CNN(3,3)
       CNN(1,3)=0.0
       CNN(1,2)=-DZDN*RDZS/DZM
       CNN(1,1)=0.0
       IF (I.NE.2) GO TO 840
       CNN(3,2)=-(1.0+RP)*CNN(3,3)
       CNN(3,1)=RP*CNN(3,3)+(1.0/DZP)-DZDN*RDZS/DZP
       CNN(2,3)=-CNN(3,3)
       CNN(2,2)=(1.0+RP)*CNN(3,3)
       CNN(2,1)=-RP*CNN(3,3)+(DZDN-DZM)*RPDZ
       CNN(1,2)=0.0
       CNN(1,1)=-DZDN*RDZS/DZM
   840 CONTINUE
C
C       Z DERIVATIVE ON S FACE  (FOR DHNS)
C
       CNS(1,1)=-RPDZ*RDZS*RATRP*(DZDS*(DZPA**2)+(DZPA+DZ)*(DZS**2))/(3.0
      1 *DZM)
       CNS(1,2)=-(1.0+RM)*CNS(1,1)-(DZPA+DZ)*RDZS/DZM
       CNS(1,3)=RM*CNS(1,1)
       CNS(2,1)=-CNS(1,1)
       CNS(2,2)=(1.0+RM)*CNS(1,1)-(DZDS-DZP)*RPDZ
       CNS(2,3)=-RM*CNS(1,1)
       CNS(3,1)=0.0
       CNS(3,2)=DZDS*RDZS/DZP
       CNS(3,3)=0.0
       IF (I.NE.2) GO TO 850
       CNS(1,3)=RM*CNS(1,1)
       CNS(1,2)=-(1.0+RP)*CNS(1,3)
       CNS(1,1)=RP*CNS(1,3)-(1.0/DZM)+DZDS*RDZS/DZM
       CNS(2,3)=-CNS(1,3)
       CNS(2,2)=(1.0+RP)*CNS(1,3)
       CNS(2,1)=-RP*CNS(1,3)-(DZDS-DZP)*RPDZ
       CNS(3,2)=0.0
       CNS(3,1)=DZDS*RDZS/DZP
   850 CONTINUE
C
C       INITIALIZE GRADIENTS FOR USE IN DO LOOPS
C
       DHEE=0.0
       DHEW=0.0
       DHEN=0.0
       DHES=0.0
       DHNE=0.0
       DHNW=0.0
       DHNN=0.0
       DHNS=0.0
C
C       NESTED DO LOOPS FOR CONSTRUCTION OF GRADIENTS
C
C       MULTIPLY GRADIENTS BY APPROPIATE RADIUS ON EACH FACE
C       TO OBTAIN VARIABLES WITH DIMENSIONS OF ANGLES
C
C       SIX OR SEVEN POINT FORMULAS FOR GRADIENTS IMPOSED BY
C       NINE POINT SUMMATIONS (3 OR 2 TERMS HAVE BEEN SET TO ZERO)
C
       DO 870 KK=1,3
       DO 860 II=1,3
       DHNE=DHNE+ANE(KK,II)*HY(II,KK)*X(I)
```

```
      DHNW=DHNW+ANW(KK,II)*HY(II,KK)*X(I-1)
      DHEN=DHEN+AEN(KK,II)*HY(II,KK)*XI(I)
      DHES=DHES+AES(KK,II)*HY(II,KK)*XI(I)
      DHEE=DHEE+CEE(KK,II)*HY(II,KK)*X(I)
      DHEW=DHEW+CEW(KK,II)*HY(II,KK)*X(I-1)
      DHNN=DHNN+CNN(KK,II)*HY(II,KK)*XI(I)
      DHNS=DHNS+CNS(KK,II)*HY(II,KK)*XI(I)
  860 CONTINUE
  870 CONTINUE
      GO TO 990
  880 CONTINUE
      RDEW=RDX(I)
      RDNS=RDY(J)*RRI(I)
      AFE=AR(IJK)
      AFW=AR(IMJK)
      IF (I.EQ.2.AND.CYL.EQ.1.0.AND.X(1).EQ.0.0) AFW=1.0
      AFN=ABK(IJK)
      IF (J.EQ.JM1.AND.CYL.EQ.1.0.AND.JC2PI.EQ.0) AFN=1.0
      AFS=ABK(IJMK)
      IF (J.EQ.2.AND.CYL.EQ.1.0.AND.JC2PI.EQ.0) AFS=1.0
      AFLOOR=AT(IJKM)
      IF (NFF.EQ.6) AFLOOR=AT(IJK)
      IF (NOWALL.EQ.1) AFLOOR=1.0
      IF (AFE.GT.EM6.AND.F(IPJK).LT.EMF) FLGE=AFLOOR
      IF (AFW.GT.EM6.AND.F(IMJK).LT.EMF) FLGW=AFLOOR
      IF (AFN.GT.EM6.AND.F(IJPK).LT.EMF) FLGN=AFLOOR
      IF (AFS.GT.EM6.AND.F(IJMK).LT.EMF) FLGS=AFLOOR
      IF (CYL.GT.0.5) FLGE=FLGE*X(I)*RXI(I)
      IF (CYL.GT.0.5) FLGW=FLGW*X(I-1)*RXI(I)
C
C      BUILDING FOUR SETS OF SIX POINT COEFFICIENTS
C
C      COORDINATES ARE SUCH THAT (E,W) GOES WITH (I+1,I-1)
C      COORDINATES ARE SUCH THAT (N,S) GOES WITH (J+1,J-1)
C
C      THETA DERIVATIVE ON E FACE (FOR DHNE)
C
      ANE(3,3)=(1.0+3.0*FRRP*RATTHM)/(3.0*DTHP)
      ANE(2,3)=-(1.0+THP)*ANE(3,3)+(FRRP/DTHM)
      ANE(1,3)=THP*ANE(3,3)-(FRRP/DTHM)
      ANE(3,2)=-ANE(3,3)+THM*RDTHS
      ANE(2,2)=(1.0+THP)*ANE(3,3)-(FRRP/DTHM)+DTHD*RPDTH
      ANE(1,2)=-THP*ANE(3,3)+(FRRP/DTHM)-THP*RDTHS
      ANE(3,1)=0.0
      ANE(2,1)=0.0
      ANE(1,1)=0.0
      IF (I.NE.2) GO TO 890
      ANE(3,1)=ANE(3,2)
      ANE(2,1)=ANE(2,2)
      ANE(1,1)=ANE(1,2)
      ANE(3,2)=ANE(3,3)
      ANE(2,2)=ANE(2,3)
      ANE(1,2)=ANE(1,3)
      ANE(3,3)=0.0
      ANE(2,3)=0.0
      ANE(1,3)=0.0
  890 CONTINUE
C
C     THETA DERIVATIVE ON W FACE (FOR DHNW)
C
      ANW(1,1)=-(1.0+3.0*FRRM*RATTHP)/(3.0*DTHM)
      ANW(2,1)=-(1.0+THM)*ANW(1,1)-(FRRM/DTHP)
      ANW(3,1)=THM*ANW(1,1)+(FRRM/DTHP)
      ANW(1,2)=-ANW(1,1)-THP*RDTHS
      ANW(2,2)=(1.0+THM)*ANW(1,1)+(FRRM/DTHP)+DTHD*RPDTH
      ANW(3,2)=-THM*ANW(1,1)-(FRRM/DTHP)+THM*RDTHS
      ANW(1,3)=0.0
      ANW(2,3)=0.0
      ANW(3,3)=0.0
C
C      R DERIVATIVE ON N FACE  (FOR DHEN)
C
      AEN(3,3)=(1.0+3.0*FRTHP*RATRM)/(3.0*DRP)
      IF (I.EQ.2) AEN(3,3)=-(1.0+RM+3.0*FRTHP*RATRM)/(3.0*DRP)
      AEN(3,2)=-(1.0+RP)*AEN(3,3)+(FRTHP/DRM)
      AEN(3,1)=RP*AEN(3,3)-(FRTHP/DRM)
      AEN(2,3)=-AEN(3,3)+RM*RDRS
      AEN(2,2)=(1.0+RP)*AEN(3,3)-(FRTHP/DRM)+DRD*RPDR
      AEN(2,1)=-RP*AEN(3,3)+(FRTHP/DRM)-RP*RDRS
      AEN(1,3)=0.0
```

```
      AEN(1,2)=0.0
      AEN(1,1)=0.0
      IF (I.NE.2) GO TO 900
      AEN(2,3)=AEN(2,3)-2.0*RM*RDRS
      AEN(2,2)=AEN(2,2)+(2.0/DRP)
      AEN(2,1)=AEN(2,1)-(2.0*RDRS)
  900 CONTINUE
C
C     R DERIVATIVE ON S FACE (FOR DHES)
C
      AES(1,1)=-(1.0+3.0*FRTHM*RATRP)/(3.0*DRM)
      AES(1,2)=-(1.0+RM)*AES(1,1)-(FRTHM/DRP)
      AES(1,3)=RM*AES(1,1)+(FRTHM/DRP)
      AES(2,1)=-AES(1,1)-RP*RDRS
      AES(2,2)=(1.0+RM)*AES(1,1)+(FRTHM/DRP)+DRD*RPDR
      AES(2,3)=-RM*AES(1,1)-(FRTHM/DRP)+RM*RDRS
      AES(3,1)=0.0
      AES(3,2)=0.0
      AES(3,3)=0.0
      IF (I.NE.2) GO TO 910
      AES(1,3)=(1.0+RM-3.0*FRTHM*RATRM)/(3.0*DRP)
      AES(1,2)=-(1.0+RP)*AES(1,3)+(FRTHM/DRM)
      AES(1,1)=RP*AES(1,3)-(FRTHM/DRM)
      AES(2,3)=-AES(1,3)-(RM*RDRS)
      AES(2,2)=(1.0+RP)*AES(1,3)-(FRTHM/DRM)+(1.0/DRP)+(1.0/DRM)
      AES(2,1)=-RP*AES(1,3)+(FRTHM/DRM)-(1.0/DRM)-(RDRS)
  910 CONTINUE
C
C      BUILDING FOUR SETS OF SEVEN POINT COEFFICIENTS
C
C      R DERIVATIVE ON E FACE (FOR DHEE)
C
      CEE(3,3)=RPDR*RDRS*RATTHM*(DRDO*(DRMA**2)+(DRMA+DR)*(DRO**2))/(3.O
     1 *DRP)
      CEE(2,3)=-(1.0+THP)*CEE(3,3)+(DRMA+DR)*RDRS/DRP
      CEE(1,3)=THP*CEE(3,3)
      CEE(3,2)=-CEE(3,3)
      CEE(2,2)=(1.0+THP)*CEE(3,3)+(DRDO-DRM)*RPDR
      CEE(1,2)=-THP*CEE(3,3)
      CEE(3,1)=0.0
      CEE(2,1)=-DRDO*RDRS/DRM
      CEE(1,1)=0.0
      IF (I.NE.2) GO TO 920
      CEE(1,1)=-THP*CEE(3,3)
      CEE(2,1)=-(1.0+THM)*CEE(1,1)-(DRPA+DR)*RDRS/DRM
      CEE(3,1)=THM*CEE(1,1)
      CEE(1,2)=-CEE(1,1)
      CEE(2,2)=(1.0+THM)*CEE(1,1)-(DRDI-DRP)*RPDR
      CEE(3,2)=-THM*CEE(1,1)
      CEE(1,3)=0.0
      CEE(2,3)=DRDI*RDRS/DRP
      CEE(3,3)=0.0
  920 CONTINUE
C
C     R DERIVATIVE ON W FACE  (FOR DHEW)
C
      CEW(1,1)=-RPDR*RDRS*RATTHP*(DRDI*(DRPA**2)+(DRPA+DR)*(DRI**2))/(3.
     1 0*DRM)
      CEW(2,1)=-(1.0+THM)*CEW(1,1)-(DRPA+DR)*RDRS/DRM
      CEW(3,1)=THM*CEW(1,1)
      CEW(1,2)=-CEW(1,1)
      CEW(2,2)=(1.0+THM)*CEW(1,1)-(DRDI-DRP)*RPDR
      CEW(3,2)=-THM*CEW(1,1)
      CEW(1,3)=0.0
      CEW(2,3)=DRDI*RDRS/DRP
      CEW(3,3)=0.0
C
C     THETA DERIVATIVE ON N FACE  (FOR DHNN)
C
      CNN(3,3)=RPDTH*RDTHS*RATRM*(DTHDE*(DTHMA**2)+(DTHMA+DTH)*(DTHE**2)
     1 )/(3.0*DTHP)
      CNN(3,2)=-(1.0+RP)*CNN(3,3)+(DTHMA+DTH)*RDTHS/DTHP
      CNN(3,1)=RP*CNN(3,3)
      CNN(2,3)=-CNN(3,3)
      CNN(2,2)=(1.0+RP)*CNN(3,3)+(DTHDE-DTHM)*RPDTH
      CNN(2,1)=-RP*CNN(3,3)
      CNN(1,3)=0.0
      CNN(1,2)=-DTHDE*RDTHS/DTHM
      CNN(1,1)=0.0
      IF (I.NE.2) GO TO 930
```

```
      CNN(3,2)=-(1.0+RP)*CNN(3,3)
      CNN(3,1)=RP*CNN(3,3)+(1.0/DTHP)-DTHDE*RDTHS/DTHP
      CNN(2,3)=-CNN(3,3)
      CNN(2,2)=(1.0+RP)*CNN(3,3)
      CNN(2,1)=-RP*CNN(3,3)+(DTHDE-DTHM)*RPDTH
      CNN(1,2)=0.0
      CNN(1,1)=-DTHDE*RDTHS/DTHM
  930 CONTINUE
C
C       THETA DERIVATIVE ON S FACE  (FOR DHNS)
C
      CNS(1,1)=-RPDTH*RDTHS*RATRP*(DTHDW*(DTHPA**2)+(DTHPA+DTH)*(DTHW**2
     1 ))/(3.0*DTHM)
      CNS(1,2)=-(1.0+RM)*CNS(1,1)-(DTHPA+DTH)*RDTHS/DTHM
      CNS(1,3)=RM*CNS(1,1)
      CNS(2,1)=-CNS(1,1)
      CNS(2,2)=(1.0+RM)*CNS(1,1)-(DTHDW-DTHP)*RPDTH
      CNS(2,3)=-RM*CNS(1,1)
      CNS(3,1)=0.0
      CNS(3,2)=DTHDW*RDTHS/DTHP
      CNS(3,3)=0.0
      IF (I.NE.2) GO TO 940
      CNS(1,3)=RM*CNS(1,1)
      CNS(1,2)=-(1.0+RP)*CNS(1,3)
      CNS(1,1)=RP*CNS(1,3)-(1.0/DTHM)+DTHDW*RDTHS/DTHM
      CNS(2,3)=-CNS(1,3)
      CNS(2,2)=(1.0+RP)*CNS(1,3)
      CNS(2,1)=-RP*CNS(1,3)-(DTHDW-DTHP)*RPDTH
      CNS(3,2)=0.0
      CNS(3,1)=DTHDW*RDTHS/DTHP
  940 CONTINUE
C
C       INITIALIZE GRADIENTS FOR USE IN DO LOOPS
C
      DHEE=0.0
      DHEW=0.0
      DHEN=0.0
      DHES=0.0
      DHNE=0.0
      DHNW=0.0
      DHNN=0.0
      DHNS=0.0
C
C       NESTED DO LOOPS FOR CONSTRUCTION OF GRADIENTS
C
C       SIX OR SEVEN POINT FORMULAS FOR GRADIENTS IMPOSED BY
C       NINE POINT SUMMATIONS (3 OR 2 TERMS HAVE BEEN SET TO ZERO)
C
      DO 980 JJ=1,3
      DO 970 II=1,3
      DHNE=DHNE+ANE(JJ,II)*HZ(II,JJ)*RX(I)
      IF (I.EQ.2) GO TO 950
      DHNW=DHNW+ANW(JJ,II)*HZ(II,JJ)*RX(I-1)
      GO TO 960
  950 CONTINUE
      DHNW=1.0
  960 CONTINUE
      DHEN=DHEN+AEN(JJ,II)*HZ(II,JJ)
      DHES=DHES+AES(JJ,II)*HZ(II,JJ)
      DHEE=DHEE+CEE(JJ,II)*HZ(II,JJ)
      DHEW=DHEW+CEW(JJ,II)*HZ(II,JJ)
      DHNN=DHNN+CNN(JJ,II)*HZ(II,JJ)*RXI(I)
      DHNS=DHNS+CNS(JJ,II)*HZ(II,JJ)*RXI(I)
  970 CONTINUE
  980 CONTINUE
  990 CONTINUE
      IF (NOWALL.EQ.0) GO TO 1000
      AFE=1.0
      AFW=1.0
      AFN=1.0
      AFS=1.0
 1000 CONTINUE
      TERM1=1.0+DHNE*DHNE
      TERM2=1.0+DHNW*DHNW
      RHDE=SQRT(TERM1+DHEE*DHEE)
      RHDW=SQRT(TERM2+DHEW*DHEW)
      IF (NFF.EQ.3.OR.NFF.EQ.4) GO TO 1020
      FEW=FLGE*(AFE*DHEE/RHDE+(1.0-AFE)*CSANG)-FLGW*(AFW*DHEW/RHDW-(1.0
     1 -AFW)*CSANG)
C       ADD CYL TERMS FOR NF=1 OR 2
```

```
      IF (STH.NE.0.0) GO TO 1010
      GO TO 1030
 1010 CONTINUE
      FEW=FEW*CTH-STH*(FLGE*(AFE*TERM1/RHDE+(1.0-AFE)*SANG)+FLGW*(AFW
     1 *TERM2/RHDW+(1.0-AFW)*SANG))
      GO TO 1030
 1020 CONTINUE
      T2RE=(1.0+DHNE**2)/RHDE
      T2RW=(1.0+DHNW**2)/RHDW
      FEW=FLGE*(AFE*(DHEE/RHDE*ACOSE+T2RE*ASINE)+(1.0-AFE)*CSANG)-FLGW*
     1 (AFW*(DHEW/RHDW*ACOSW+T2RW*ASINW)-(1.0-AFW)*CSANG)
C
 1030 CONTINUE
      TERM1=1.0+DHEN*DHEN
      TERM2=1.0+DHES*DHES
      RHDN=SQRT(TERM1+DHNN*DHNN)
      RHDS=SQRT(TERM2+DHNS*DHNS)
      IF (NFF.EQ.3.OR.NFF.EQ.4) GO TO 1040
      FNS=FLGN*(AFN*DHNN/RHDN+(1.0-AFN)*CSANG)-FLGS*(AFS*DHNS/RHDS-(1.0
     1 -AFS)*CSANG)
      GO TO 1050
 1040 CONTINUE
      T2RN=-DHEN*DHNN/RHDN
      T2RS=-DHES*DHNS/RHDS
      FNS=FLGN*(AFN*(DHNN/RHDN*ACOSN+T2RN*ASINN)+(1.0-AFN)*CSANG)-FLGS*
     1 (AFS*(DHNS/RHDS*ACOSS+T2RS*ASINS)-(1.0-AFS)*CSANG)
 1050 CONTINUE
C     ADD WALL ADHESION AT BOTTOM OF CELL
      IF (NOWALL.EQ.1) GO TO 1060
      FEW=FEW-(2.0-FLGE-FLGW)*SANG
      FNS=FNS-(2.0-FLGN-FLGS)*SANG
C     LIMIT CURVATURE BY CELL SIZE
 1060 AFEW=ABS(FEW)
      FEW=RDEW*AMIN1(AFEW,2.0)*SIGN(1.0,FEW)
      AFNS=ABS(FNS)
      FNS=RDNS*AMIN1(AFNS,2.0)*SIGN(1.0,FNS)
C
      FEWFNS=FEW+FNS
      PS(IJK)=-SIGMA*AMIN1(ABS(FEWFNS),PSLIM)*SIGN(1.0 FEWFNS)
C
 1070 CONTINUE
C
C       FIX FLOOR AND CEILING FOR ALL J AND K I=3,4......IM1
C
      PSLIMI=PSLIM*SIGMA
      DO 1150 K=2,KM1
      DO 1150 J=2,JM1
      DO 1150 I=2,IM1
      CALL IJKAJCT
      PN(IJK)=0.0
      IF (BETA(IJK).LE.0.0) GO TO 1150
      IF (NF(IJK).LT.1.OR.NF(IJK).GT.6) GO TO 1150
C
C     TREAT INTERIOR CELLS OF I,J,K MESH BY GENERAL RULE,
C     EXCEPTION FOR EDGE CELLS
C
      PSAV=0.0
      RIAV=0.0
      DO 1140 MM=1,3
      DO 1140 NN=1,3
      DO 1140 LL=1,3
      L=I-2+LL
      M=J-2+MM
      N=K-2+NN
      LMN=NQ*(II5*(N-1)+IMAX*(M-1)+(L-1))+1
      IF (LL.EQ.2.AND.MM.EQ.2.AND.NN.EQ.2) GO TO 1140
      IF (L.NE.1) GO TO 1080
      LPMN=LMN+NQ
      PS(LMN)=PS(LPMN)
 1080 IF (L.NE.IMAX) GO TO 1090
      LMMN=LMN-NQ
      PS(LMN)=PS(LMMN)
 1090 IF (M.NE.1) GO TO 1100
      LMPN=LMN+NQ*IMAX
      PS(LMN)=PS(LMPN)
 1100 IF (M.NE.JMAX) GO TO 1110
      LMMN=LMN-NQ*IMAX
      PS(LMN)=PS(LMMN)
 1110 IF (N.NE.1) GO TO 1120
      LMNP=LMN+NQ*II5
```

```
      PS(LMN)=PS(LMNP)
 1120 IF (N.NE.KMAX) GO TO 1130
      LMNM=LMN-NQ*II5
      PS(LMN)=PS(LMNM)
 1130 CONTINUE
      IF (ABS(PS(LMN)).LT.1.0) GO TO 1140
      RIAV=RIAV+1.0
      PSAV=PSAV+PS(LMN)
 1140 CONTINUE
      IF (RIAV.EQ.0.0) GO TO 1150
      PSAVI=PSAV/RIAV
      PN(IJK)=PS(IJK)
      IF (ABS(PS(IJK)).GT.1.5*ABS(PSAVI).OR.ABS(PS(IJK)).GT.PSLIMI) PN
     1 (IJK)=AMIN1(PSLIMI,1.5*ABS(PSAVI))*SIGN(1.0,PSAVI)
      IF (ABS(PS(IJK)).LT.0.5*ABS(PSAVI)) PN(IJK)=0.5*ABS(PSAVI)*SIGN(1.
     1 0,PSAVI)
 1150 CONTINUE
C
      DO 1160 K=1,KMAX
      DO 1160 J=1,JMAX
      DO 1160 I=1,IMAX
      CALL IJKONLY
      PS(IJK)=PN(IJK)
      IF (I.EQ.1.OR.I.EQ.IMAX.OR.J.EQ.1.OR.J.EQ.JMAX.OR.K.EQ.1.OR.K.EQ
     1 .KMAX) PS(IJK)=0.0
 1160 CONTINUE
C
C       FIX FOR I=2 CASE
C
      I=2
      DO 1180 K=2,KM1
      DO 1180 J=2,JM1
      CALL IJKAJCT
      IF (BETA(IJK).LE.0.0) GO TO 1180
      IF (NF(IJK).LT.1.OR.NF(IJK).GT.6) GO TO 1180
C
C     TREAT INTERIOR CELLS OF J,K MESH BY GENERAL RULE,
C     EXCEPTION FOR EDGE CELLS
C
      PSAV=0.0
      RIAV=0.0
      DO 1170 MM=1,3
      DO 1170 NN=1,3
      M=J-2+MM
      N=K-2+NN
      LMN=NQ*(II5*(N-1)+IMAX*(M-1)+2)+1
      LMPN=LMN+NQ*IMAX
      LMMN=LMN-NQ*IMAX
      LMNP=LMN+NQ*II5
      LMNM=LMN-NQ*II5
      IF (M.EQ.1) PS(LMN)=PS(LMPN)
      IF (N.EQ.1) PS(LMN)=PS(LMNP)
      IF (M.EQ.JMAX) PS(LMN)=PS(LMMN)
      IF (N.EQ.KMAX) PS(LMN)=PS(LMNM)
      IF (ABS(PS(LMN)).LT.1.0) GO TO 1170
      RIAV=RIAV+1.0
      PSAV=PSAV+PS(LMN)
 1170 CONTINUE
      IF (RIAV.EQ.0.0) GO TO 1180
      PSAVI=PSAV/RIAV
      IF (ABS(PS(IJK)).GT.1.5*ABS(PSAVI).OR.ABS(PS(IJK)).GT.PSLIMI) PS
     1 (IJK)=AMIN1(PSLIMI,1.5*ABS(PSAVI))*SIGN(1.0,PSAVI)
      IF (ABS(PS(IJK)).LT.0.5*ABS(PSAVI)) PS(IJK)=0.5*ABS(PSAVI)*SIGN(1.
     1 0,PSAVI)
 1180 CONTINUE
C
      I=2
      DO 1190 K=1,KMAX,KM1
      DO 1190 J=1,JMAX,JM1
      CALL IJKONLY
      PS(IJK)=0.0
 1190 CONTINUE
C
C  **********
      RETURN
      END
*DK SURF100
*DK TILDE
      SUBROUTINE TILDE
*CA SLCOM1
```

```
C
C * * EXPLICITLY APPROXIMATE NEW TIME LEVEL VELOCITIES
C
      DO 50 J=2,JM1
      DO 50 I=2,IM1
      DO 40 K=2,KM1
      CALL CALCIJK
      UDUM=0.0
      VDUM=0.0
      WDUM=0.0
      IF (BETA(IJK).LT.0.0) GO TO 30
C
      IJ=I+(J-1)*IMAX
      GX=GXA(IJ)
      GY=GYA(IJ)
C * * U-VELOCITY
C
      IF (BETA(IPJK).LT.0.0) GO TO 10
      IF (F(IJK)+F(IPJK).LT.EMF) GO TO 10
      SGU=SIGN(1.0,UN(IJK))
      DUDR=(UN(IPJK)-UN(IJK))*RDX(I+1)
      DUDL=(UN(IJK)-UN(IMJK))*RDX(I)
      IF (AR(IPJK).LT.EM6) DUDR=0.0
      IF (AR(IMJK).LT.EM6) DUDL=0.0
      DXAV=DELX(I)+DELX(I+1)+ALPHA*SGU*(DELX(I+1)-DELX(I))
      FUX=UN(IJK)*(DELX(I)*DUDR+DELX(I+1)*DUDL+ALPHA*SGU*(DELX(I+1)*DUDL
     1 -DELX(I)*DUDR))/DXAV
      VBDYT=(DELX(I)*VN(IPJK)+DELX(I+1)*VN(IJK))/(DELX(I)+DELX(I+1))
      VBDYB=(DELX(I)*VN(IPJMK)+DELX(I+1)*VN(IJMK))/(DELX(I)+DELX(I+1))
      DVDY=VBDYT-VBDYB
      IF (ABK(IJK).LT.EM6.OR.ABK(IPJK).LT.EM6.OR.ABK(IJMK).LT.EM6.OR.ABK
     1 (IPJMK).LT.EM6) DVDY=0.0
      VAV=0.5*(VBDYT+VBDYB)
      DYF=0.5*(DELY(J)+DELY(J+1))
      DYA=0.5*(DELY(J-1)+DELY(J))
      DUDF=RR(I)*(UN(IJPK)-UN(IJK))/DYF
      DUDA=RR(I)*(UN(IJK)-UN(IJMK))/DYA
      IF (AR(IJPK).LT.EM6) DUDF=0.0
      IF (AR(IJMK).LT.EM6) DUDA=0.0
      SGV=SIGN(1.0,VAV)
      DYAV=DYF+DYA+ALPHA*SGV*(DYF-DYA)
      FUY=(DYA*DUDF+DYF*DUDA+ALPHA*SGV*(DYF*DUDA-DYA*DUDF))*(VAV/DYAV)
      WBDYT=(DELX(I+1)*WN(IJK)+DELX(I)*WN(IPJK))/(DELX(I)+DELX(I+1))
      WBDYB=(DELX(I+1)*WN(IJKM)+DELX(I)*WN(IPJKM))/(DELX(I)+DELX(I+1))
      WAV=0.5*(WBDYT+WBDYB)
      DZT=0.5*(DELZ(K)+DELZ(K+1))
      DZB=0.5*(DELZ(K-1)+DELZ(K))
      DUDT=(UN(IJKP)-UN(IJK))/DZT
      DUDB=(UN(IJK)-UN(IJKM))/DZB
      IF (AR(IJKP).LT.EM6) DUDT=0.0
      IF (AR(IJKM).LT.EM6) DUDB=0.0
      SGW=SIGN(1.0,WAV)
      DZAV=DZT+DZB+ALPHA*SGW*(DZT-DZB)
      FUZ=(DZB*DUDT+DZT*DUDB+ALPHA*SGW*(DZT*DUDB-DZB*DUDT))*(WAV/DZAV)
      DUDXSQ=2.0*(DUDR-DUDL)/(DELX(I)+DELX(I+1))
      DUDYSQ=RDY(J)*(DUDF-DUDA)*RR(I)
      DUDZSQ=RDZ(K)*(DUDT-DUDB)
      DUDCYL=1.0*RX(I)*(DELX(I)*DUDR+DELX(I+1)*DUDL)/(DELX(I)+DELX(I+1))
     1 -2.0*RR(I)*RX(I)*RDY(J)*DVDY-UN(IJK)*RX(I)**2
      VISX=NU*(DUDXSQ+DUDYSQ+DUDZSQ+CYL*DUDCYL)
      FUC=CYL*(-0.25*(VBDYT+VBDYB)**2)*RX(I)
      FRU=-RADPS*(VBDYT+VBDYB)-X(I)*RADPS**2
      UDUM=UN(IJK)+DELT*((P(IJK)-P(IPJK))*2.0/(DELX(I)+DELX(I+1))+GX-FUX
     1 -FUY-FUZ+VISX-FUC-FRU)
   10 CONTINUE
C
C * * V-VELOCITY
C
      IF (BETA(IJPK).LT.0.0) GO TO 20
      IF (F(IJK)+F(IJPK).LT.EMF) GO TO 20
      VBDYR=(DELX(I+1)*VN(IJK)+DELX(I)*VN(IPJK))/(DELX(I)+DELX(I+1))
      VBDYL=(DELX(I)*VN(IMJK)+DELX(I-1)*VN(IJK))/(DELX(I-1)+DELX(I))
      UBDYR=(DELY(J)*UN(IJPK)+DELY(J+1)*UN(IJK))/(DELY(J)+DELY(J+1))
      UBDYL=(DELY(J)*UN(IMJPK)+DELY(J+1)*UN(IMJK))/(DELY(J)+DELY(J+1))
      UAV=0.5*(UBDYR+UBDYL)
      DXR=0.5*(DELX(I)+DELX(I+1))
      DXL=0.5*(DELX(I-1)+DELX(I))
      DVDR=(VN(IPJK)-VN(IJK))/DXR
      DVDL=(VN(IJK)-VN(IMJK))/DXL
      IF (ABK(IPJK).LT.EM6) DVDR=0.0
```

```
      IF (ABK(IMJK).LT.EM6) DVDL=0.0
      SGU=SIGN(1.0,UAV)
      DXAV=DXR+DXL+ALPHA*SGU*(DXR-DXL)
      FVX=(DXR*DVDL+DXL*DVDR+ALPHA*SGU*(DXR*DVDL-DXL*DVDR))*(UAV/DXAV)
      FVC=CYL*(VBDYR*UBDYR+VBDYL*UBDYL)*0.5*RXI(I)
      SGV=SIGN(1.0,VN(IJK))
      DVDF=(VN(IJPK)-VN(IJK))*RDY(J+1)*RRI(I)
      DVDA=(VN(IJK)-VN(IJMK))*RDY(J)*RRI(I)
      IF (ABK(IJPK).LT.EM6) DVDF=0.0
      IF (ABK(IJMK).LT.EM6) DVDA=0.0
      DYAV=DELY(J)+DELY(J+1)+ALPHA*SGV*(DELY(J+1)-DELY(J))
      FVY=VN(IJK)*(DELY(J)*DVDF+DELY(J+1)*DVDA+ALPHA*SGV*(DELY(J+1)*DVDA
     1 -DELY(J)*DVDF))/DYAV
      WBDYT=(DELY(J+1)*WN(IJK)+DELY(J)*WN(IJPK))/(DELY(J)+DELY(J+1))
      WBDYB=(DELY(J+1)*WN(IJKM)+DELY(J)*WN(IJPKM))/(DELY(J)+DELY(J+1))
      WAV=0.5*(WBDYT+WBDYB)
      DZT=0.5*(DELZ(K)+DELZ(K+1))
      DZB=0.5*(DELZ(K-1)+DELZ(K))
      DVDT=(VN(IJKP)-VN(IJK))/DZT
      DVDB=(VN(IJK)-VN(IJKM))/DZB
      IF (ABK(IJKP).LT.EM6) DVDT=0.0
      IF (ABK(IJKM).LT.EM6) DVDB=0.0
      SGW=SIGN(1.0,WAV)
      DZAV=DZT+DZB+ALPHA*SGW*(DZT-DZB)
      FVZ=(DZT*DVDB+DZB*DVDT+ALPHA*SGW*(DZT*DVDB-DZB*DVDT))*(WAV/DZAV)
      DVDXSQ=RDX(I)*(DVDR-DVDL)
      DVDYSQ=2.0*RRI(I)*(DVDF-DVDA)/(DELY(J)+DELY(J+1))
      DVDZSQ=RDZ(K)*(DVDT-DVDB)
      DUDY=UN(IJPK)+UN(IMJPK)-UN(IJK)-UN(IMJK)
      IF (AR(IJPK).LT.EM6.OR.AR(IMJPK).LT.EM6.OR.AR(IJK).LT.EM6.OR.AR
     1 (IMJK).LT.EM6) DUDY=0.0
      DVDCYL=RXI(I)*0.5*(DVDR+DVDL)+2.0*RXI(I)*RRI(I)*DUDY/(DELY(J)+DELY
     1 (J+1))-VN(IJK)*RXI(I)**2
      VISY=NU*(DVDXSQ+DVDYSQ+DVDZSQ+CYL*DVDCYL)
      FRV=RADPS*(UBDYR+UBDYL)
      VDUM=VN(IJK)+DELT*((P(IJK)-P(IJPK))*2.0/(DELY(J)+DELY(J+1))*RRI(I)
     1 +GY-FVX-FVY-FVZ+VISY-FVC-FRV)
   20 CONTINUE
C
C * * W-VELOCITY
C
      IF (BETA(IJKP).LT.0.0) GO TO 30
      IF (F(IJK)+F(IJKP).LT.EMF) GO TO 30
      UBDYR=(DELZ(K+1)*UN(IJK)+DELZ(K)*UN(IJKP))/(DELZ(K)+DELZ(K+1))
      UBDYL=(DELZ(K+1)*UN(IMJK)+DELZ(K)*UN(IMJKP))/(DELZ(K)+DELZ(K+1))
      UAV=0.5*(UBDYR+UBDYL)
      DXR=0.5*(DELX(I)+DELX(I+1))
      DXL=0.5*(DELX(I-1)+DELX(I))
      DWDR=(WN(IPJK)-WN(IJK))/DXR
      DWDL=(WN(IJK)-WN(IMJK))/DXL
      IF (AT(IPJK).LT.EM6) DWDR=0.0
      IF (AT(IMJK).LT.EM6) DWDL=0.0
      SGU=SIGN(1.0,UAV)
      DXAV=DXR+DXL+ALPHA*SGU*(DXR-DXL)
      FWX=(DXR*DWDL+DXL*DWDR+ALPHA*SGU*(DXR*DWDL-DXL*DWDR))*(UAV/DXAV)
      VBDYF=(DELZ(K+1)*VN(IJK)+DELZ(K)*VN(IJKP))/(DELZ(K)+DELZ(K+1))
      VBDYA=(DELZ(K+1)*VN(IJMK)+DELZ(K)*VN(IJMKP))/(DELZ(K)+DELZ(K+1))
      VAV=0.5*(VBDYF+VBDYA)
      DYF=0.5*(DELY(J)+DELY(J+1))
      DYA=0.5*(DELY(J-1)+DELY(J))
      DWDF=RRI(I)*(WN(IJPK)-WN(IJK))/DYF
      DWDA=RRI(I)*(WN(IJK)-WN(IJMK))/DYA
      IF (AT(IJPK).LT.EM6) DWDF=0.0
      IF (AT(IJMK).LT.EM6) DWDA=0.0
      SGV=SIGN(1.0,VAV)
      DYAV=DYF+DYA+ALPHA*SGV*(DYF-DYA)
      FWY=(DYF*DWDA+DYA*DWDF+ALPHA*SGV*(DYF*DWDA-DYA*DWDF))*(VAV/DYAV)
      SGW=SIGN(1.0,WN(IJK))
      DWDT=(WN(IJKP)-WN(IJK))*RDZ(K+1)
      DWDB=(WN(IJK)-WN(IJKM))*RDZ(K)
      IF (AT(IJKP).LT.0.0) DWDT=0.0
      IF (AT(IJKM).LT.EM6) DWDB=0.0
      DZAV=DELZ(K)+DELZ(K+1)+ALPHA*SGW*(DELZ(K+1)-DELZ(K))
      FWZ=(DELZ(K)*DWDT+DELZ(K+1)*DWDB+ALPHA*SGW*(DELZ(K+1)*DWDB-DELZ(K)
     1 *DWDT))*(WN(IJK)/DZAV)
      DWDXSQ=RDX(I)*(DWDR-DWDL)
      DWDYSQ=RRI(I)*RDY(J)*(DWDF-DWDA)
      DWDZSQ=2.0*(DWDT-DWDB)/(DELZ(K+1)+DELZ(K))
      DWDCYL=RXI(I)*0.5*(DWDR+DWDL)
      VISZ=NU*(DWDXSQ+DWDYSQ+DWDZSQ+CYL*DWDCYL)
```

```
      WDUM=WN(IJK)+DELT*((P(IJK)-P(IJKP))*2.0/(DELZ(K)+DELZ(K+1))+GZ-FWX
     1 -FWY-FWZ+VISZ)
   30 CONTINUE
C
C * * COMPUTE VELOCITIES CONSIDERING SECOND ORDER
C     ACCURATE OPTION PARAMETER AVE
C
      U(IJK)=AVE*UDUM+(1.0-AVE)*U(IJK)
      V(IJK)=AVE*VDUM+(1.0-AVE)*V(IJK)
      W(IJK)=AVE*WDUM+(1.0-AVE)*W(IJK)
   40 CONTINUE
   50 CONTINUE
C
C  **********
      RETURN
      END
*DK VELV
      SUBROUTINE VELV (I1,I2,J1,J2,KK1,KK2,NA,IPER)
*CA SLCOM1
C
C * * DRAW VELOCITY VECTORS IN PLANE OR PERSPECTIVE PLOTS
C
C * * DETERMINE PLANE TO BE PLOTTED
C
      IFM=0
      JFM=0
      KFM=0
      IF (I1.NE.I2) GO TO 10
      IFM=I1
      MPLN=1
      ICON=J2-J1+1
      JCON=KK2-KK1+1
   10 IF (J1.NE.J2) GO TO 20
      JFM=J1
      MPLN=2
      ICON=I2-I1+1
      JCON=KK2-KK1+1
   20 IF (KK1.NE.KK2) GO TO 30
      KFM=KK1
      MPLN=3
      ICON=I2-I1+1
      JCON=J2-J1+1
   30 CONTINUE
C
C * * DRAW PLANE OR PERSPECTIVE PLOT FRAME
C
      CALL FADV (NA)
      IF (IPER.EQ.2) GO TO 40
      AXP=1.0
      IF (MPLN.EQ.1) AXP=0.0
      AYP=1.0
      IF (MPLN.EQ.3) AYP=0.0
      IF (NA.GT.0) CALL DRF (IFM,JFM,KFM)
      GO TO 50
   40 IF (NA.GT.0) CALL DRFP
C
C * * SET UP TO DRAW VELOCITY VECTORS
C
   50 IJ=1
      KNTIJ=1
      VMAX=0.0
      VMAXI=0.0
      VMAXJ=0.0
      VMAXK=0.0
      DO 110 K=2,KMAX
      DO 110 J=2,JMAX
      DO 110 I=2,IMAX
      CALL CALCIJK
      IF (F(IJK).LT.0.5) GO TO 110
      IF (AC(IJK).LT.0.5) GO TO 110
      GO TO (60,70,80), MPLN
   60 IF (I.NE.I1) GO TO 110
      IF (J.LT.J1) GO TO 110
      IF (J.GT.J2) GO TO 110
      IF (K.LT.KK1) GO TO 110
      IF (K.GT.KK2) GO TO 110
      GO TO 90
   70 IF (J.NE.J1) GO TO 110
      IF (K.LT.KK1) GO TO 110
      IF (K.GT.KK2) GO TO 110
```

```
      IF (I.LT.I1) GO TO 110
      IF (I.GT.I2) GO TO 110
      GO TO 90
   80 IF (K.NE.KK1) GO TO 110
      IF (J.LT.J1) GO TO 110
      IF (J.GT.J2) GO TO 110
      IF (I.LT.I1) GO TO 110
      IF (I.GT.I2) GO TO 110
   90 CONTINUE
      STH1=0.0
      CTH1=1.0
      CYLM=1.0
      IF (MPLN.NE.3.AND.IPER.EQ.1) GO TO 100
      TH1=CYL*RX(IM1)*YJ(J)
      STH1=SIN(TH1)
      CTH1=COS(TH1)
      CYLM=1.0-CYL
  100 CONTINUE
C
C * * DETERMINE MAXIMUM VECTOR LENGTH
C
      VINER=XI(I)*RADPS
      VINER=0.0
      UAVE=0.5*(U(IJK)+U(IMJK))
      VAVE=0.5*(V(IJK)+V(IJMK))+VINER
      UVV(IJ)=UAVE*CTH1-VAVE*STH1
      VVV(IJ)=UAVE*STH1+VAVE*CTH1
      WVV(IJ)=0.5*(W(IJK)+W(IJKM))
      XPC(IJ)=XI(I)*CTH1
      YPC(IJ)=YJ(J)*CYLM+XI(I)*STH1
      ZPC(IJ)=ZK(K)
      VMAX=AMAX1(VMAX,ABS(UVV(IJ)),ABS(VVV(IJ)),ABS(WVV(IJ)))
      VMAXI=AMAX1(VMAXI,ABS(VVV(IJ)),ABS(WVV(IJ)))
      VMAXJ=AMAX1(VMAXJ,ABS(UVV(IJ)),ABS(WVV(IJ)))
      VMAXK=AMAX1(VMAXK,ABS(UVV(IJ)),ABS(VVV(IJ)))
      IF (BETA(IJK).LT.0.0.OR.F(IJK).LT.0.5) UVV(IJ)=1.0E+10
      KT1=KM1
      IF (K.GT.KT1) UVV(IJ)=+1.0E+10
      KNTIJ=KNTIJ+1
      IJ=IJ+NQ2
      IF (IJ.GT.NVEC) GO TO 310
  110 CONTINUE
      KNTIJ=KNTIJ-1
C
C * * SCALE VECTOR LENGTH
C
      GO TO (120,130,140), MPLN
  120 DROU=VELMX*DELMN/(VMAXI+1.0E-10)
      GO TO 150
  130 DROU=VELMX*DELMN/(VMAXJ+1.0E-10)
      GO TO 150
  140 DROU=VELMX*DELMN/(VMAXK+1.0E-10)
  150 CONTINUE
      IJ=1
      IF (MPLN.NE.3) GO TO 160
      XBL=XBLC
      YBF=YBFC
  160 CONTINUE
      DO 210 L=1,KNTIJ
      X1P=XPC(IJ)
      Y1P=YPC(IJ)
      Z1P=ZPC(IJ)
      U1P=UVV(IJ)
      V1P=VVV(IJ)
      W1P=WVV(IJ)
      IF (U1P.GT.1.0E+06) GO TO 210
      GO TO (170,190), IPER
C
C * * PLANE VECTOR PLOT
C
  170 CONTINUE
      X1EF=AXP*(X1P-XBL)+(1.0-AXP)*(Y1P-YBF)
      Y1EF=AYP*(Z1P-ZBB)+(1.0-AYP)*(Y1P-YBF)
      X2EF=AXP*(X1P+U1P*DROU-XBL)+(1.0-AXP)*(Y1P+V1P*DROU-YBF)
      Y2EF=AYP*(Z1P+W1P*DROU-ZBB)+(1.0-AYP)*(Y1P+V1P*DROU-YBF)
      IX1=FIXL(MPLN)+X1EF*XCONV(MPLN)
      IY1=FIYB+Y1EF*YCONV(MPLN)
      IX2=FIXL(MPLN)+X2EF*XCONV(MPLN)
      IY2=FIYB+Y2EF*YCONV(MPLN)
      CALL DRV (IX1,IY1,IX2,IY2)
```

```
C     DRAW ARROWHEADS
      FIX1=IX1
      FIX2=IX2
      FIY1=IY1
      FIY2=IY2
      X34=FIX1+0.85*(FIX2-FIX1)
      Y34=FIY1+0.85*(FIY2-FIY1)
      DX234=FIX2-X34
      DY234=FIY2-Y34
      D234=SQRT(DX234**2+DY234**2)
      DA234=D234*0.57735027
C       TAN(15 DEG)=0.57735027
      COSB=DA234
      SINB=0.0
      IF (DX234.EQ.0.0) GO TO 180
      BETA234=1.57079633-ATAN(DY234/DX234)
      COSB=COS(BETA234)*DA234
      SINB=SIN(BETA234)*DA234
  180 IX3=X34-COSB
      IX4=X34+COSB
      IY3=Y34+SINB
      IY4=Y34-SINB
      CALL DRV (IX2,IY2,IX3,IY3)
      CALL DRV (IX2,IY2,IX4,IY4)
      GO TO 210
C
C * * PERSPECTIVE VECTOR PLOT
C
  190 CALL PCNV (IXI1,IETA1,X1P,Y1P,Z1P)
      X2P=X1P+U1P*DROU
      Y2P=Y1P+V1P*DROU
      Z2P=Z1P+W1P*DROU
      CALL PCNV (IXI2,IETA2,X2P,Y2P,Z2P)
      CALL DRVEC (IXI1,IETA1,IXI2,IETA2)
C     DRAW ARROWHEADS
      FIX1=IXI1
      FIX2=IXI2
      FIY1=IETA1
      FIY2=IETA2
      X34=FIX1+0.85*(FIX2-FIX1)
      Y34=FIY1+0.85*(FIY2-FIY1)
      DX234=FIX2-X34
      DY234=FIY2-Y34
      D234=SQRT(DX234**2+DY234**2)
      DA234=D234*0.57735027
C       TAN(15 DEG)=0.57735027
      COSB=DA234
      SINB=0.0
      IF (DX234.EQ.0.0) GO TO 200
      BETA234=1.57079633-ATAN(DY234/DX234)
      COSB=COS(BETA234)*DA234
      SINB=SIN(BETA234)*DA234
  200 IX3=X34-COSB
      IX4=X34+COSB
      IY3=Y34+SINB
      IY4=Y34-SINB
      CALL DRV (IXI2,IETA2,IX3,IY3)
      CALL DRV (IXI2,IETA2,IX4,IY4)
  210 IJ=IJ+NQ2
C
C * * DRAW CONSTANT PLANE FREE SURFACE
C
      IF (IPER.NE.1) GO TO 220
      CALL SURCNTR (I1,I2,J1,J2,KK1,KK2,MPLN,AXP,AYP)
C
C * * WRITE TITLES AND INFO ON PLOT
C
  220 CONTINUE
      XBL=X(1)
      YBF=Y(1)
      CALL LINCNT (59)
      GO TO (230,240,250), MPLN
  230 WRITE (12,350) VMAX,VMAXI
      GO TO 260
  240 WRITE (12,360) VMAX,VMAXJ
      GO TO 260
  250 WRITE (12,370) VMAX,VMAXK
  260 CONTINUE
      WRITE (12,380) CYCLE,T,(NAME(I),I=1,5),JNM
      IF (IPER.EQ.2) GO TO 300
```

```
      GO TO (270,280,290), MPLN
  270 WRITE (12,320) IFM,J1,J2,KK1,KK2
      GO TO 300
  280 WRITE (12,330) JFM,I1,I2,KK1,KK2
      GO TO 300
  290 WRITE (12,340) KFM,I1,I2,J1,J2
  300 RETURN
C
C * * ERROR - STOP AND PRINT
C
  310 CONTINUE
      WRITE (9,390)
      WRITE (59,390)
      WRITE (12,390)
      CALL EXITA (3020)
C
  320 FORMAT (13H  CONSTANT I=,I4,17H   SURFACE     J=,I3,3H TO,I3,7H
     1  K=,I3,3H TO,I3)
  330 FORMAT (13H  CONSTANT J=,I4,17H   SURFACE     I=,I3,3H TO,I3,7H
     1  K=,I3,3H TO,I3)
  340 FORMAT (13H  CONSTANT K=,I4,17H   SURFACE     I=,I3,3H TO,I3,7H
     1  J=,I3,3H TO,I3)
  350 FORMAT (18H  VELOCITY VECTORS,4X,7H  VMAX=,1PE12.5,4X,6HVMAXI=,E12
     1 .5)
  360 FORMAT (18H  VELOCITY VECTORS,4X,7H  VMAX=,1PE12.5,4X,6HVMAXJ=,E12
     1 .5)
  370 FORMAT (18H  VELOCITY VECTORS,4X,7H  VMAX=,1PE12.5,4X,6HVMAXK=,E12
     1 .5)
  380 FORMAT (8H  CYCLE=,I5,4H  T=,1PE12.5,5X,5A8,1X,A8)
  390 FORMAT (51HO - - - - ERROR - - TWX ARRAY FOR PLOTTING EXCEEDED)
      END
*DK VCHGCAL
      SUBROUTINE VCHGCAL
*CA SLCOM1
C
C * * CALCULATE TOTAL VOLUME CHANGE
      VOFTOT=0.0
C
      DO 40 K=2,KM1
      DO 40 J=2,JM1
      DO 40 I=2,IM1
      CALL CALCIJK
      IF (BETA(IJK).LT.0.0) GO TO 40
      VOFTOT=VOFTOT+F(IJK)*DELX(I)*DELY(J)/RRI(I)*DELZ(K)*AC(IJK)
      VCHG=0.0
      IF (F(IJK).GT.EMF.AND.F(IJK).LT.EMF1) GO TO 20
      IF (F(IJK).GE.EMF1) GO TO 10
      VCHG=F(IJK)
      F(IJK)=0.0
      GO TO 20
   10 CONTINUE
      VCHG=-(1.0-F(IJK))
      F(IJK)=1.0
   20 CONTINUE
      VCHGT=VCHGT+VCHG*DELX(I)*DELY(J)/RRI(I)*DELZ(K)*AC(IJK)
      IF (F(IJK).LT.EMF1) GO TO 40
      IF (F(IPJK).LT.EMF) GO TO 30
      IF (F(IMJK).LT.EMF) GO TO 30
      IF (F(IJPK).LT.EMF) GO TO 30
      IF (F(IJMK).LT.EMF) GO TO 30
      IF (F(IJKP).LT.EMF) GO TO 30
      IF (F(IJKM).LT.EMF) GO TO 30
      GO TO 40
   30 F(IJK)=F(IJK)-1.1*EMF
      VCHG=1.1*EMF
      VCHGT=VCHGT+VCHG*DELX(I)*DELY(J)/RRI(I)*DELZ(K)*AC(IJK)
   40 CONTINUE
C
      RETURN
      END
*DK VFCONV
      SUBROUTINE VFCONV
*CA SLCOM1
C
C * * CONVECT VOLUME OF FLUID FUNCTION F
C
      DATA FLGCS, KOUNT /0.0,0/
      IF (CYCLE.LT.1) GO TO 100
      KOUNT=0
      FLGCS=FLGCS+FLGC
```

```
      FLGC=0.0
      DO 90 K=1,KM1
      DO 90 J=1,JM1
      DO 90 I=1,IM1
      CALL CALCIJK
      VX=U(IJK)*DELT
      VY=V(IJK)*DELT
      VZ=W(IJK)*DELT
      ABVX=ABS(VX)
      ABVY=ABS(VY)
      ABVZ=ABS(VZ)
      IF (NF(IJK).GT.7) GO TO 20
      IF (BETA(IJK).LT.0.0) GO TO 20
      IF (ABVX.LT.0.9*DELX(I).AND.ABVY.LT.0.9*DELY(J)/RRI(I).AND.ABVZ.LT
     1 .0.9*DELZ(K)) GO TO 20
      FLGC=1.0
      WRITE (9,130) I,J,K,U(IJK),V(IJK),W(IJK),T,DELT,CYCLE,ITER
      WRITE (12,130) I,J,K,U(IJK),V(IJK),W(IJK),T,DELT,CYCLE,ITER
      IF (FLGCS.LT.100.0) GO TO 10
      WRITE (12,110) FLGCS
      WRITE (9,110) FLGCS
      CALL EXIT
   10 CONTINUE
      KOUNT=KOUNT+1
      IF (KOUNT.LT.20) GO TO 20
      WRITE (12,120) KOUNT
      WRITE (9,120) KOUNT
      CALL EXIT
   20 IF (BETA(IJK).LE.0.0) GO TO 90
      IF (BETA(IPJK).LE.0.0) GO TO 40
C
C * * CONVECT F IN X-DIRECTION
C
      IA=I+1
      ID=I
      IAJK=IPJK
      IDJK=IJK
      IDM=MAX0(I-1,1)
      RB=AR(IJK)*(1.0-CYL+CYL*X(I))
      RA=AC(IPJK)*(1.0-CYL+CYL*XI(I+1))
      RD=AC(IJK)*(1.0-CYL+CYL*XI(I))
      IF (VX.GE.0.0) GO TO 30
      IA=I
      ID=I+1
      IAJK=IJK
      IDJK=IPJK
      IDM=MIN0(I+2,IMAX)
      RA=AC(IJK)*(1.0-CYL+CYL*XI(I))
      RD=AC(IPJK)*(1.0-CYL+CYL*XI(I+1))
   30 CONTINUE
      IAD=IA
      IDMJK=NQ*(II5*(K-1)+IMAX*(J-1)+(IDM-1))+1
      IF (NF(IDJK).EQ.3.OR.NF(IDJK).EQ.4) IAD=ID
      IF (NF(IDJK).EQ.5.OR.NF(IDJK).EQ.6) IAD=ID
      IF (FN(IAJK).LT.EMF.OR.FN(IDMJK).LT.EMF) IAD=IA
      IADJK=NQ*(II5*(K-1)+IMAX*(J-1)+(IAD-1))+1
      FDM=AMAX1(FN(IDMJK),FN(IDJK))
      IF (F(IDMJK).LT.EMF.AND.F(IAJK).LT.EMF) FDM=AMAX1(FDM,0.10)
      IF (BETA(IDMJK).LT.0.0) FDM=1.0
      FX1=FN(IADJK)*ABVX+AMAX1((FDM-FN(IADJK))*ABVX-(FDM-FN(IDJK))*DELX
     1 (ID),0.0)
      FX=AMIN1(FX1,FN(IDJK)*DELX(ID))
      F(IDJK)=F(IDJK)-FX*RDX(ID)*(RB/RD)
      F(IAJK)=F(IAJK)+FX*RDX(IA)*(RB/RA)
   40 IF (BETA(IJPK).LE.0.0) GO TO 60
C
C * * CONVECT F IN Y-DIRECTION
C
      JA=J+1
      JD=J
      IJAK=IJPK
      IJDK=IJK
      JDM=MAX0(J-1,1)
      RB=ABK(IJK)
      RA=AC(IJPK)
      RD=AC(IJK)
      IF (VY.GE.0.0) GO TO 50
      JA=J
      JD=J+1
      IJAK=IJK
```

```
      IJDK=IJPK
      JDM=MINO(J+2,JMAX)
      RA=AC(IJK)
      RD=AC(IJPK)
   50 CONTINUE
      JAD=JA
      IJDMK=NQ*(II5*(K-1)+IMAX*(JDM-1)+(I-1))+1
      IF (NF(IJDK).EQ.5.OR.NF(IJDK).EQ.6) JAD=JD
      IF (NF(IJDK).EQ.1.OR.NF(IJDK).EQ.2) JAD=JD
      IF (FN(IJAK).LT.EMF.OR.FN(IJDMK).LT.EMF) JAD=JA
      IJADK=NQ*(II5*(K-1)+IMAX*(JAD-1)+(I-1))+1
      FDM=AMAX1(FN(IJDMK),FN(IJDK))
      IF (F(IJDMK).LT.EMF.AND.F(IJAK).LT.EMF) FDM=AMAX1(FDM,0.10)
      IF (BETA(IJDMK).LT.0.0) FDM=1.0
      FY1=FN(IJADK)*ABVY+AMAX1((FDM-FN(IJADK))*ABVY-(FDM-FN(IJDK))*DELY
     1 (JD)/RRI(I),0.0)
      FY=AMIN1(FY1,FN(IJDK)*DELY(JD)/RRI(I))
      F(IJDK)=F(IJDK)-FY*RDY(JD)*RRI(I)*(RB/RD)
      F(IJAK)=F(IJAK)+FY*RDY(JA)*RRI(I)*(RB/RA)
C
C * * CONVECT F IN Z-DIRECTION
C
   60 IF (BETA(IJKP).LE.0.0) GO TO 80
      KA=K+1
      KD=K
      IJKA=IJKP
      IJKD=IJK
      KDM=MAXO(K-1,1)
      RB=AT(IJK)
      RA=AC(IJKP)
      RD=AC(IJK)
      IF (VZ.GE.0.0) GO TO 70
      KA=K
      KD=K+1
      IJKA=IJK
      IJKD=IJKP
      KDM=MINO(K+2,KMAX)
      RA=AC(IJK)
      RD=AC(IJKP)
   70 CONTINUE
      KAD=KA
      IJKDM=NQ*(II5*(KDM-1)+IMAX*(J-1)+(I-1))+1
      IF (NF(IJKD).EQ.1.OR.NF(IJKD).EQ.2) KAD=KD
      IF (NF(IJKD).EQ.3.OR.NF(IJKD).EQ.4) KAD=KD
      IF (FN(IJKA).LT.EMF.OR.FN(IJKDM).LT.EMF) KAD=KA
      IJKAD=NQ*(II5*(KAD-1)+IMAX*(J-1)+(I-1))+1
      FDM=AMAX1(FN(IJKDM),FN(IJKD))
      IF (F(IJKDM).LT.EMF.AND.F(IJKA).LT.EMF) FDM=AMAX1(FDM,0.10)
      IF (BETA(IJKDM).LT.0.0) FDM=1.0
      FZ1=FN(IJKAD)*ABVZ+AMAX1((FDM-FN(IJKAD))*ABVZ-(FDM-FN(IJKD))*DELZ
     1 (KD),0.0)
      FZ=AMIN1(FZ1,FN(IJKD)*DELZ(KD))
      F(IJKD)=F(IJKD)-FZ*RDZ(KD)*(RB/RD)
      F(IJKA)=F(IJKA)+FZ*RDZ(KA)*(RB/RA)
   80 IF (NF(IJK).NE.0) GO TO 90
      IF (IDEFM.GT.0) GO TO 90
      IF (BETA(IJK).EQ.1.0) GO TO 90
      F(IJK)=F(IJK)+DELT*FN(IJK)*D(IJK)
   90 CONTINUE
C
C * * CALCULATE TOTAL VOLUME CHANGE
C
  100 CALL VCHGCAL
C
      RETURN
C
  110 FORMAT (*     VFCONV - TOO MANY LARGE VELOCITIES  FLGCS=*,F10.5)
  120 FORMAT (*     VFCONV - TOO MANY LARGE VELOCITIES  KOUNT=*,F10.5)
  130 FORMAT (*   VFCONV - VELOCITIES TOO LARGE    */3X,3I3,5F12.5,2I5)
      END
*DK WRTAPE
      SUBROUTINE WRTAPE
*CA SLCOM1
C
C * * WRITE MAGMETIC TAPE
C
      WRITE (9,10) NUMTD,T,CYCLE
      IF (LPR.GT.0) WRITE (12,10) NUMTD,T,CYCLE
      WRITE (59,10) NUMTD,T,CYCLE
      TWTD=T+TDDT
```

```
      WRITE (8) (AA(N),N=1,NCR1)
      WRITE (8) (BASC(N),N=1,NCR2)
      WRITE (8) (BASC1(N),N=1,NCR2)
      WRITE (8) (BASC2(N),N=1,NCR2)
      WRITE (8) (BASC3(N),N=1,NCR2)
      NUMTD=NUMTD+1
C
      RETURN
C
   10 FORMAT (12HO TAPE DUMP=,I3,7H  AT T=,1PE12.5,7H CYCLE=,I5)
      END
*DK XINDF
      SUBROUTINE XINDF
*CA SLCOM1
C
C * * SET ARRAYS TO INDEFINITE
C
      ILIM=II5*KMAX*NQ
      DATA XINDEF /17770000000000777777B/
      IUP=IBASC
      DO 10 I=1,IUP
      BASC(I)=XINDEF
      BASC2(I)=XINDEF
   10 BASC3(I)=XINDEF
      NCR1=LOCF(ZLAST)-LOCF(AA)+1
      NCR2=LOCF(BASC(ILIM))-LOCF(BASC)+1
C
      RETURN
      END
*DK DRAWQ
      SUBROUTINE DRAWQ
*CA SLCOM1
C
      PARAMETER (IBAR2Q=2*IBAR2-2)
      COMMON BETAQ(IBAR2Q,KBAR2), FQ(IBAR2Q,KBAR2), ACQ(IBAR2Q,KBAR2),
     1 ARQ(IBAR2Q,KBAR2), ATQ(IBAR2Q,KBAR2), UQ(IBAR2Q,KBAR2), VQ(IBAR2Q
     2 ,KBAR2), XQ(IBAR2Q), XIQ(IBAR2Q), YQ(KBAR2), YJQ(KBAR2), IM1Q,
     3 JM1Q, JBAR2Q, JMAXQ, IMAXQ, SF, XSHFT, YSHFT, DELXQ(IBAR2Q),
     4 DELYQ(KBAR2), XMINQ, XMAXQ, YMINQ, YMAXQ
C
C       MOVE QUANITIES FROM 3D MESH TO 2D PLOT MESH
C
C       PLOTTING J=2 AND J=JM1 PLANES FOR 180 DEG PROBLEM
C       PLOTTING J=2 AND J=JDP(2) PLANES FOR 360 DEG PROBLEM
C
      IM1Q=IBAR2Q-1
      JM1Q=KM1
      JBAR2Q=KBAR2
      JMAXQ=KMAX
      IMAXQ=IBAR2Q
C
      J=2
      DO 10 L=IMAX,IBAR2Q
      DO 10 K=1,KMAX
      I=L-IMAX+2
      CALL IJKONLY
      BETAQ(L,K)=BETA(IJK)
      FQ(L,K)=F(IJK)
      ARQ(L,K)=AR(IJK)
      ATQ(L,K)=AT(IJK)
      ACQ(L,K)=AC(IJK)
      UQ(L,K)=U(IJK)
      VQ(L,K)=W(IJK)
   10 CONTINUE
C
      J=JDP(2)
C
      DO 20 L=1,IM1
      DO 20 K=1,KMAX
      I=IMAX-L+1
      CALL IJKAJCT
      BETAQ(L,K)=BETA(IJK)
      FQ(L,K)=F(IJK)
      ARQ(L,K)=AR(IMJK)
      IF (L.EQ.IM1) ARQ(L,K)=1.0
      ATQ(L,K)=AT(IJK)
      ACQ(L,K)=AC(IJK)
      UQ(L,K)=-U(IMJK)
      IF (L.EQ.IM1) UQ(L,K)=0.0
      VQ(L,K)=W(IJK)
```

```
   20 CONTINUE
C
      DO 30 L=IMAX,IBAR2Q
      I=L-IM1+1
      XQ(L)=X(I)+X(IM1)
      DELXQ(L)=DELX(I)
   30 CONTINUE
C
      DO 40 L=1,IM1
      I=IMAX-L
      XQ(L)=X(IM1)-X(I)
      DELXQ(L)=DELX(I)
   40 CONTINUE
C
      DO 50 L=2,IBAR2Q
      XIQ(L)=(XQ(L)+XQ(L-1))*0.5
   50 CONTINUE
      XIQ(1)=-XIQ(2)
C
      DO 60 K=1,KMAX
      YQ(K)=Z(K)
      YJQ(K)=ZK(K)
      DELYQ(K)=DELZ(K)
   60 CONTINUE
C
C +++ SET CONSTANT TERMS FOR PLOTTING
C
      XMINQ=XQ(1)
      XMAXQ=XQ(IM1Q)
      YMINQ=YQ(1)
      YMAXQ=YQ(JM1Q)
      D1=XMAXQ-XMINQ
      D2=YMAXQ-YMINQ
      D3=AMAX1(D1,D2)
      SF=1.0/D3
      XSHFT=0.5*(1.0-D1*SF)
      YSHFT=0.5*(1.0-D2*SF)
C
C * * DETERMINE MAXIMUM VECTOR LENGTH
C
      VMAX=0.0
      VMAXJ=0.0
      DO 70 K=2,KM1
      DO 70 J=2,JM1
      DO 70 I=2,IM1
      CALL IJKAJCT
      IF (F(IJK).LT.0.5) GO TO 70
      IF (AC(IJK).LT.0.5) GO TO 70
      IF (J.NE.2.AND.J.NE.JOP(2)) GO TO 70
      UAVE=0.5*(U(IJK)+U(IMJK))
      VAVE=0.5*(V(IJK)+V(IJMK))
      WAVE=0.5*(W(IJK)+W(IJKM))
      VMAX=AMAX1(VMAX,ABS(UAVE),ABS(VAVE),ABS(WAVE))
      VMAXJ=AMAX1(VMAXJ,ABS(UAVE),ABS(WAVE))
   70 CONTINUE
C
C * * SCALE VECTOR LENGTH
C
      VELMX1=VELMX*DELMN/(VMAXJ+1.0E-10)
C
C +++
C +++ DRAW VELOCITY VECTOR PLOT
C +++
      CALL ADV (1)
      CALL LINCNT (1)
C     WRITE(12,200) T,CYCLE,NAME
      CALL FRAMEQ (XMINQ,XMAXQ,YMAXQ,YMINQ)
      CALL DRWOBSQ
      DO 80 I=2,IM1Q
      ACELH=0.5*CYL+1.0-CYL
      DO 80 J=2,JM1Q
      IF (FQ(I,J).LT.0.5) GO TO 80
      IF (ACQ(I,J).LT.ACELH) GO TO 80
      XXCC=XIQ(I)
      YYCC=0.5*(YQ(J)+YQ(J-1))
      UMPL=0.0
      VMPL=0.0
      IF (ARQ(I,J).GT.EM6) UMPL=1.0
      IF (ARQ(I-1,J).GT.EM6) UMPL=1.0/(1.0+UMPL)
      IF (ATQ(I,J).GT.EM6) VMPL=1.0
```

```
      IF (ATQ(I,J-1).GT.EM6) VMPL=1.0/(1.0+VMPL)
      UPLT=UMPL*(UQ(I-1,J)+UQ(I,J))
      VPLT=VMPL*(VQ(I,J-1)+VQ(I,J))
      UVEC=UPLT*VELMX1+XXCC
      VVEC=VPLT*VELMX1+YYCC
      CALL DRWVECA (XXCC,YYCC,UVEC,VVEC,1)
      CALL PLTPT (XXCC,YYCC,53B,1)
   80 CONTINUE
C
C +++ DRAW FREE SURFACE
C
      FPL=0.5
      DO 100 I=2,IM1Q
      DO 100 J=2,JM1Q
      IF (BETAQ(I,J).LT.0.0) GO TO 100
      FATR=0.25*(FQ(I,J)+FQ(I+1,J)+FQ(I,J+1)+FQ(I+1,J+1))
      FXTR=0.5*(FQ(I+1,J+1)+FQ(I+1,J)-FQ(I,J+1)-FQ(I,J))/(XIQ(I+1)-XIQ(I
     1 ))
      FYTR=0.5*(FQ(I,J+1)+FQ(I+1,J+1)-FQ(I,J)-FQ(I+1,J))/(YJQ(J+1)-YJQ(J
     1 ))
      FTRS=FXTR**2+FYTR**2
      IF (FTRS.EQ.0.0) FTRS=1.0E+10
      XTR=0.5*(XIQ(I+1)+XIQ(I))+(FPL-FATR)*FXTR/FTRS
      XTR=AMAX1(XTR,XIQ(I))
      XTR=AMIN1(XTR,XIQ(I+1))
      YTR=0.5*(YJQ(J)+YJQ(J+1))+(FPL-FATR)*FYTR/FTRS
      YTR=AMAX1(YTR,YJQ(J))
      YTR=AMIN1(YTR,YJQ(J+1))
      IF (FQ(I,J).GT.0.5.AND.FQ(I+1,J).GT.0.5) GO TO 90
      IF (FQ(I,J).LT.0.5.AND.FQ(I+1,J).LT.0.5) GO TO 90
      FABR=0.25*(FQ(I,J)+FQ(I+1,J)+FQ(I,J-1)+FQ(I+1,J-1))
      FXBR=0.5*(FQ(I+1,J)+FQ(I+1,J-1)-FQ(I,J)-FQ(I,J-1))/(XIQ(I+1)-XIQ(I
     1 ))
      FYBR=0.5*(FQ(I,J)+FQ(I+1,J)-FQ(I,J-1)-FQ(I+1,J-1))/(YJQ(J)-YJQ(J-1
     1 ))
      FBRS=FXBR**2+FYBR**2
      IF (FBRS.EQ.0.0) FBRS=1.0E+10
      XXBR=0.5*(XIQ(I+1)+XIQ(I))+(FPL-FABR)*FXBR/FBRS
      XXBR=AMAX1(XXBR,XIQ(I))
      XXBR=AMIN1(XXBR,XIQ(I+1))
      YBR=0.5*(YJQ(J)+YJQ(J-1))+(FPL-FABR)*FYBR/FBRS
      YBR=AMAX1(YBR,YJQ(J-1))
      YBR=AMIN1(YBR,YJQ(J))
      CALL DRWVEC (XXBR,YBR,XTR,YTR,1)
   90 CONTINUE
      IF (FQ(I,J).GT.0.5.AND.FQ(I,J+1).GT.0.5) GO TO 100
      IF (FQ(I,J).LT.0.5.AND.FQ(I,J+1).LT.0.5) GO TO 100
      FATL=0.25*(FQ(I,J)+FQ(I,J+1)+FQ(I-1,J)+FQ(I-1,J+1))
      FXTL=0.5*(FQ(I,J+1)+FQ(I,J)-FQ(I-1,J+1)-FQ(I-1,J))/(XIQ(I)-XIQ(I-1
     1 ))
      FYTL=0.5*(FQ(I-1,J+1)+FQ(I,J+1)-FQ(I-1,J)-FQ(I,J))/(YJQ(J+1)-YJQ(J
     1 ))
      FTLS=FXTL**2+FYTL**2
      IF (FTLS.EQ.0.0) FTLS=1.0E+10
      XTL=0.5*(XIQ(I-1)+XIQ(I))+(FPL-FATL)*FXTL/FTLS
      XTL=AMAX1(XTL,XIQ(I-1))
      XTL=AMIN1(XTL,XIQ(I))
      YTL=0.5*(YJQ(J)+YJQ(J+1))+(FPL-FATL)*FYTL/FTLS
      YTL=AMAX1(YTL,YJQ(J))
      YTL=AMIN1(YTL,YJQ(J+1))
      CALL DRWVEC (XTL,YTL,XTR,YTR,1)
  100 CONTINUE
C
      CALL LINCNT (59)
      WRITE (12,120) VMAX,VMAXJ
      WRITE (12,130) CYCLE,T,(NAME(I),I=1,5),JNM
      WRITE (12,110) 2,IM1,2,KM1
      RETURN
C
  110 FORMAT (21H CONSTANT J=2 AND JM1,17H   SURFACE     I=,I3,3H TO,I3,
     1 7H     K=,I3,3H TO,I3)
  120 FORMAT (18H  VELOCITY VECTORS,4X,7H  VMAX=,1PE12.5,4X,6HVMAXJ=,E12
     1 .5)
  130 FORMAT (8H  CYCLE=,I5,4H  T=,1PE12.5,5X,5A8,1X,A8)
      END
*DK DRWOBSQ
      SUBROUTINE DRWOBSQ
*CA SLCOM1
C
      PARAMETER (IBAR2Q=2*IBAR2-2)
```

```
      COMMON BETAQ(IBAR2Q,KBAR2), FQ(IBAR2Q,KBAR2), ACQ(IBAR2Q,KBAR2),
     1 ARQ(IBAR2Q,KBAR2), ATQ(IBAR2Q,KBAR2), UQ(IBAR2Q,KBAR2), VQ(IBAR2Q
     2 ,KBAR2), XQ(IBAR2Q), XIQ(IBAR2Q), YQ(KBAR2), YJQ(KBAR2), IM1Q,
     3 JM1Q, JBAR2Q, JMAXQ, IMAXQ, SF, XSHFT, YSHFT, DELXQ(IBAR2Q),
     4 DELYQ(KBAR2), XMINQ, XMAXQ, YMINQ, YMAXQ
C
C +++ DRAW AROUND ALL OBSTACLES
C
      DO 170 I=2,IM1Q
      ATR=1.0-EM6
      ATL=1.0-EM6
      IF (I.EQ.2.) ATL=-EM6
      ATC=1.0-EM6
      DO 170 J=2,JM1Q
      IF (ACQ(I,J).LT.EM6) GO TO 170
      AFR=1.0
      AFT=1.0
      AFL=1.0
      AFB=1.0
      IF (ARQ(I,J).LT.ATR) AFR=ARQ(I,J)/ATR
      IF (ATQ(I,J).LT.ATC) AFT=ATQ(I,J)/ATC
      IF (ARQ(I-1,J).LT.ATL) AFL=ARQ(I-1,J)/ATL
      IF (ATQ(I,J-1).LT.ATC) AFB=ATQ(I,J-1)/ATC
      IF (ACQ(I,J).GE.ATC) GO TO 120
      IF (I.EQ.2) AFL=AFR-EM6
      IF (I.EQ.IM1Q) AFR=AFL-EM6
      IF (J.EQ.2) AFB=AFT-EM6
      IF (J.EQ.JM1Q) AFT=AFB-EM6
      IF ((AFT+AFB).LT.EM6.OR.(AFL+AFR).LT.EM6) GO TO 170
      M=1
      AMN=AFB+AFR
      IF ((AFR+AFT).GT.AMN) GO TO 10
      M=2
      AMN=AFR+AFT
   10 IF ((AFT+AFL).GT.AMN) GO TO 20
      M=3
      AMN=AFT+AFL
   20 IF ((AFL+AFB).GT.AMN) GO TO 30
      M=4
   30 GO TO (40,60,80,100), M
   40 X1=XQ(I-1)+AFT*DELXQ(I)
      Y1=YQ(J)
      IF (AFT.LT.1.0) GO TO 50
      Y1=Y1-AFR*DELYQ(J)
   50 X2=XQ(I-1)
      ZY2=YQ(J)-AFL*DELYQ(J)
      IF (AFL.LT.1.0) GO TO 160
      X2=X2+AFB*DELXQ(I)
      GO TO 160
   60 X1=XQ(I-1)
      Y1=YQ(J-1)+AFL*DELYQ(J)
      IF (AFL.LT.1.0) GO TO 70
      X1=X1+AFT*DELXQ(I)
   70 X2=XQ(I-1)+AFB*DELXQ(I)
      ZY2=YQ(J-1)
      IF (AFB.LT.1.0) GO TO 160
      ZY2=ZY2+AFR*DELYQ(J)
      GO TO 160
   80 X1=XQ(I)-AFB*DELXQ(I)
      Y1=YQ(J-1)
      IF (AFB.LT.1.0) GO TO 90
      Y1=Y1+AFL*DELYQ(J)
   90 X2=XQ(I)
      ZY2=YQ(J-1)+AFR*DELYQ(J)
      IF (AFR.LT.1.0) GO TO 160
      X2=X2-AFT*DELXQ(I)
      GO TO 160
  100 X1=XQ(I)
      Y1=YQ(J)-AFR*DELYQ(J)
      IF (AFR.LT.1.0) GO TO 110
      X1=X1-AFB*DELXQ(I)
  110 X2=XQ(I)-AFT*DELXQ(I)
      ZY2=YQ(J)
      IF (AFT.LT.1.0) GO TO 160
      ZY2=ZY2-AFL*DELYQ(J)
      GO TO 160
  120 IF (AFR.GT.EM6) GO TO 130
      X1=XQ(I)
      Y1=YQ(J-1)
      X2=X1
```

```
      ZY2=YQ(J)
      CALL DRWVEC (X1,Y1,X2,ZY2,1)
  130 IF (AFT.GT.EM6) GO TO 140
      X1=XQ(I-1)
      Y1=YQ(J)
      X2=XQ(I)
      ZY2=Y1
      CALL DRWVEC (X1,Y1,X2,ZY2,1)
  140 IF (AFL.GT.EM6) GO TO 150
      X1=XQ(I-1)
      Y1=YQ(J)
      X2=X1
      ZY2=YQ(J-1)
      CALL DRWVEC (X1,Y1,X2,ZY2,1)
  150 IF (AFB.GT.EM6) GO TO 170
      X1=XQ(I-1)
      Y1=YQ(J-1)
      X2=XQ(I)
      ZY2=Y1
      CALL DRWVEC (X1,Y1,X2,ZY2,1)
  160 CALL DRWVEC (X1,Y1,X2,ZY2,1)
  170 CONTINUE
      RETURN
      END
*DK DRWVEC
      SUBROUTINE DRWVEC (XONE,YONE,XTWO,YTWO,ISYM)
*CA SLCOM1
      PARAMETER (IBAR2Q=2*IBAR2-2)
      COMMON BETAQ(IBAR2Q,KBAR2), FQ(IBAR2Q,KBAR2), ACQ(IBAR2Q,KBAR2),
     1 ARQ(IBAR2Q,KBAR2), ATQ(IBAR2Q,KBAR2), UQ(IBAR2Q,KBAR2), VQ(IBAR2Q
     2 ,KBAR2), XQ(IBAR2Q), XIQ(IBAR2Q), YQ(KBAR2), YJQ(KBAR2), IM1Q,
     3 JM1Q, JBAR2Q, JMAXQ, IMAXQ, SF, XSHFT, YSHFT, DELXQ(IBAR2Q),
     4 DELYQ(KBAR2), XMINQ, XMAXQ, YMINQ, YMAXQ
C
C +++ DRAW A VECTOR
C +++ PROVIDES A SYSTEM DEPENDANT CALL
C
      IC=0
      X1=XONE
      Y1=YONE
      X2=XTWO
      ZY2=YTWO
      XO1=(X1-XMINQ)*SF+XSHFT
      YO1=(Y1-YMINQ)*SF+YSHFT
      XO2=(X2-XMINQ)*SF+XSHFT
      YO2=(ZY2-YMINQ)*SF+YSHFT
      IX1=16.+900.0*XO1
      IX2=16.+900.0*XO2
      IY1=16.+900.0*(1.0-YO1)
      IY2=16.+900.0*(1.0-YO2)
      CALL DRV (IX1,IY1,IX2,IY2)
      RETURN
C
      ENTRY DRWVECA(XONE,YONE,XTWO,YTWO,ISYM)
C
C +++ DRAW A VECTOR WITH ARROWHEADS
C
      IC=0
      X1=XONE
      Y1=YONE
      X2=XTWO
      ZY2=YTWO
      XO1=(X1-XMINQ)*SF+XSHFT
      YO1=(Y1-YMINQ)*SF+YSHFT
      XO2=(X2-XMINQ)*SF+XSHFT
      YO2=(ZY2-YMINQ)*SF+YSHFT
      IX1=16.+900.0*XO1
      IX2=16.+900.0*XO2
      IY1=16.+900.0*(1.0-YO1)
      IY2=16.+900.0*(1.0-YO2)
      CALL DRV (IX1,IY1,IX2,IY2)
C    DRAW ARROWHEADS
      FIX1=IX1
      FIX2=IX2
      FIY1=IY1
      FIY2=IY2
      X34=FIX1+0.85*(FIX2-FIX1)
      Y34=FIY1+0.85*(FIY2-FIY1)
      DX234=FIX2-X34
      DY234=FIY2-Y34
```

```
      D234=SQRT(DX234**2+DY234**2)
      DA234=D234*0.57735027
C       TAN(15 DEG)=0.57735027
      COSB=DA234
      SINB=0.0
      IF (DX234.EQ.0.0) GO TO 10
      BETA234=1.57079633-ATAN(DY234/DX234)
      COSB=COS(BETA234)*DA234
      SINB=SIN(BETA234)*DA234
   10 IX3=X34-COSB
      IX4=X34+COSB
      IY3=Y34+SINB
      IY4=Y34-SINB
      CALL DRV (IX2,IY2,IX3,IY3)
      CALL DRV (IX2,IY2,IX4,IY4)
      RETURN
      END
*DK FRAME
      SUBROUTINE FRAMEQ (XXL,XXR,YYT,YYB)
C
C +++ DRAW A FRAME AROUND THE PLOT
C
      CALL DRWVEC (XXL,YYT,XXR,YYT,0)
      CALL DRWVEC (XXL,YYT,XXL,YYB,0)
      CALL DRWVEC (XXL,YYB,XXR,YYB,0)
      CALL DRWVEC (XXR,YYB,XXR,YYT,0)
      RETURN
      END
```

Printed in the United States of America
Available from
National Technical Information Service
US Department of Commerce
5285 Port Royal Road
Springfield, VA 22161

Microfiche (A01)

Page Range	NTIS Price Code	Page Range	NTIS Price Code	Page Range	NTIS Price Code	Page Range	NTIS Price Code
001-025	A02	151-175	A08	301-325	A14	451-475	A20
026-050	A03	176-200	A09	326-350	A15	476-500	A21
051-075	A04	201-225	A10	351-375	A16	501-525	A22
076-100	A05	226-250	A11	376-400	A17	526-550	A23
101-125	A06	251-275	A12	401-425	A18	551-575	A24
126-150	A07	276-300	A13	426-450	A19	576-600	A25
						601-up*	A99

*Contact NTIS for a price quote.

Los Alamos