LA-8355

2

See attached Errata Sheet

SOLA-VOF: A Solution Algorithm for Transient Fluid Flow with Multiple Free Boundaries

DO NOT CIRCULATE

PERMANENT RETENTION

REQUIRED BY CONTRACT

LOS ALAMOS NATIONAL LABORATORY
3 9338 00313 5075

University of California

LOS ALAMOS SCIENTIFIC LABORATORY
Post Office Box 1663 Los Alamos, New Mexico 87545

An Affirmative Action/Equal Opportunity Employer

This work was supported by the Electric Power Research Institute, the Office of Naval Research, and the National Aeronautics and Space Administration, Lewis Research Center.

Edited by Karyn Ames

This report was prepared as an account of work sponsored by the United States Government. Neither the United States nor the United States Department of Energy, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, mark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

UNITED STATES
DEPARTMENT OF ENERGY
CONTRACT W-7405-ENG. 36

ERRATTA FOR SOLA-VOF NUMBER 1

8/11/80

To insure convergence in the pressure iteration scheme, an under-relaxation is necessary for those cells used as interpolation neighbors for free-surface cells. When a cell is an interpolation neighbor for only one surface cell, the proper relaxation factor is that given by Eq. (15) in the SOLA-VOF report, LA-8355.

When more than one surface cell uses the same full cell as a neighbor, then a different factor must be used. This calculation is performed at the end of subroutine PETACAL following the comment "Set peta in adjacent full cell." The proper coding, which has been included in the publicly available program is

```
1527 C      *** SET PETA IN ADJACENT FULL CELL
1528 C
1529        DO 290 J=1,JMAX
1530        DO 290 I=1,IMAX
1531        NFF=NF(I,J)
1532        IF (NFF.EQ.0.OR.BETA(I,J).LT.0.0) GO TO 290
1533        IF (NFF.GT.5) GO TO 280
1534        L=I
1535        M=J
1536        GO TO (230,240,250,260,290), NFF
1537   230  L=I-1
1538        DMX=DELX(L)
1539        DMIN=0.5*(DMX+DELX(I))
1540        GO TO 270
1541   240  L=I+1
1542        DMX=DELX(L)
1543        DMIN=0.5*(DMX+DELX(I))
1544        GO TO 270
1545   250  M=J-1
1546        DMX=DELY(M)
1547        DMIN=0.5*(DMX+DELY(J))
1548        GO TO 270
1549   260  M=J+1
1550        DMX=DELY(M)
1551        DMIN=0.5*(DMX+DELY(J))
1552   270  CONTINUE
1553        IF (NF(L,M).GT.0) GO TO 290
1554        CTOS=DELT*RDTEXP
1555        COMG=AMIN1(CTOS**2,1.0)
1556        BPD=1.0/PETA(L,M)-BETA(L,M)*(1.0-PETA(I,J))
1557       1*DELT/(DMIN*DMX)*(COMG/RHOF)
1558        PETA(L,M)= 1.0/BPD
1559        GO TO 290
1560   280  CONTINUE
1561        P(I,J)=PR(NFF)
1562   290  CONTINUE
1563   300  CONTINUE
1564        RETURN
1565        END
```

LA-8355
UC-32 and UC-34
Issued: August 1980

SOLA-VOF: A Solution Algorithm for Transient Fluid Flow with Multiple Free Boundaries

B. D. Nichols
C. W. Hirt
R. S. Hotchkiss

LOS ALAMOS NATL. LAB. LIBS.
3 9338 00313 5075

CONTENTS

Abstract

I. INTRODUCTION 1

II. SOLA-VOF 4

A. Outline 5

B. Momentum Equation Approximations 8

C. Continuity Equation Approximation 11

D. Implicit vs Explicit Calculations 14

E. Approximations for Volume of Fluid Function 16

F. Surface Tension and Wall Adhesion 21

G. Marker Particles 25

H. Boundary Conditions 26

I. Numerical Stability Considerations 29

III. PROGRAM DETAILS 31

A. Subroutines 31

B. Basic Input Variables 35

C. Mesh Generation Input 37

D. Variables and Arrays Listed in COMMON 39

IV. PROBLEM RUNNING 42

A. General Procedure 42

B. Sample Test Problem 44

ACKNOWLEDGMENT 58

APPENDIX A. CODE LISTING 59

APPENDIX B. SAMPLE CALCULATIONS 108

REFERENCES 118

SOLA-VOF: A SOLUTION ALGORITHM FOR TRANSIENT FLUID FLOW WITH MULTIPLE FREE BOUNDARIES

by

B. D. Nichols, C. W. Hirt, and R. S. Hotchkiss

ABSTRACT

In this report a simple, but powerful, computer program is presented for the solution of two-dimensional transient fluid flow with free boundaries. The SOLA-VOF program, which is based on the concept of a fractional volume of fluid (VOF), is more flexible and efficient than other methods for treating arbitrary free boundaries.

SOLA-VOF has a variety of user options that provide capabilities for a wide range of applications. Its basic mode of operation is for single fluid calculations having multiple free surfaces. However, SOLA-VOF can also be used for calculations involving two fluids separated by a sharp interface. In either case, the fluids may be treated as incompressible or as having limited compressibility. Surface tension forces with wall adhesion are permitted in both cases. Internal obstacles may be defined by blocking out any desired combination of cells in the mesh, which is composed of rectangular cells of variable size.

SOLA-VOF is an easy-to-use program. Its logical parts are isolated in separate subroutines, and numerous special features have been included to simplify its operation, such as an automatic time-step control, a flexible mesh generator, extensive output capabilities, a variety of optional boundary conditions, and instructive internal documentation.

I. INTRODUCTION

In structural dynamics, it is customary to employ Lagrangian coordinates as the basis for numerical solution algorithms. In fluid dynamics, however, both Lagrangian and Eulerian coordinates have been used with considerable success. Because each coordinate representation has unique advantages and disadvantages, the choice of which representation to use depends on the characteristics of the problem to be solved. In this report, the emphasis is on Eulerian formulations

for fluid dynamics problems involving free boundaries. In particular, for problems where free boundaries undergo such large deformations that Lagrangian methods cannot be used.

Free boundaries are considered here to be free surfaces or material interfaces. Three types of problems arise in the numerical treatment of free boundaries: (1) their discrete representation, (2) their evolution in time, and (3) the manner in which boundary conditions are imposed on them.

Discrete Lagrangian representations for a fluid are conceptually simple because each zone of a grid that subdivides the fluid into elements remains identified with the same fluid element for all time. Body and surface forces on these elements are easy to define, so it is relatively straightforward to compute the dynamic response of the elements. In an Eulerian representation the grid remains fixed and the identity of individual fluid elements is not maintained. Nevertheless, it is customary to view the fluid in an Eulerian mesh cell as a fluid element on which body and surface forces may be computed, in a manner completely analogous to a Lagrangian calculation. The two methods differ, however, in the manner in which the fluid elements are moved to new positions after their new velocities have been computed. In the Lagrangian case the grid simply moves with the computed element velocities, whereas in an Eulerian or Arbitrary-Lagrangian-Eulerian calculation it is necessary to compute the flow of fluid through the mesh. This flow, or convective flux calculation, requires an averaging of the flow properties of all fluid elements that find themselves in a given mesh cell after some period of time.

It is this averaging process, inherent in convective flux approximations, that is the biggest drawback of Eulerian methods. Convective averaging results in a smoothing of all variations in flow quantities, and in particular, a smearing of surfaces of discontinuity such as free surfaces. The only way to overcome this loss in boundary resolution is to introduce some special treatment that recognizes a discontinuity and avoids averaging across it. Although there have been a variety of techniques developed to do this,[1-3] they all have limitations. A comparison[4] of the relative advantages and disadvantages of these methods leads to a new technique that is simple yet effective. This method, the fractional volume of fluid (VOF) method, forms the basis of the SOLA-VOF program described in this report.

The SOLA-VOF solution algorithm has been designed for a wide range of applications. It may be applied to problems involving a single fluid having any

number of free surfaces, or to two immiscible fluids separated by any number of free interfaces. Limited fluid compressibility[5] is allowed in either case, as are suface tension and wall adhesion forces. In addition, SOLA-VOF has provisions for constructing internal obstacles by blocking out mesh cells. Several automatic features are included in the program to aid the user, such as a generator for variable meshes, an automatic time-step control, and routines to identify and label disjoint void regions.

The basis of the SOLA-VOF method is the fractional volume of fluid scheme for tracking free boundaries. In this technique, a function F(x,y,t) is defined whose value is unity at any point occupied by fluid and zero elsewhere. When averaged over the cells of a computational mesh, the average value of F in a cell is equal to the fractional volume of the cell occupied by fluid. In particular, a unit value of F corresponds to a cell full of fluid, whereas a zero value indicates that the cell contains no fluid. Cells with F values between zero and one contain a free surface. The VOF method requires only one storage word for each mesh cell, which is consistent with the storage requirements for all other dependent variables.

In addition to defining which cells contain a boundary, the F function can be used to define where fluid is located in a boundary cell. The normal direction to the boundary lies in the direction in which the value of F changes most rapidly. Because F is a step function, however, its derivatives must be computed in a special way, as described below. When properly computed, the derivatives can then be used to determine the boundary normal. Finally, when the normal direction and the value of F in a boundary cell are known, a line cutting the cell can be constructed that approximates the interface there. Additionally, surface curvatures can be computed for the definition of surface tension forces.

The time dependence of F is governed by the equation,

$$\frac{\partial F}{\partial t} + u \frac{\partial F}{\partial x} + v \frac{\partial F}{\partial y} = 0 \ , \tag{1}$$

where (u,v) are fluid velocities in the coordinate directions (x,y). This equation states that F moves with the fluid. In an Eulerian mesh, the flux of F moving with the fluid through a cell must be computed, but as noted earlier,

standard finite-difference approximations would lead to a smearing of the F function and interfaces would lose their definition. Fortunately, the fact that F is a step function with values of zero or one, permits the use of a flux approximation that preserves its discontinuous nature. This approximation, referred to as a Donor-Acceptor method,[6] is described in more detail in the next section.

Thus, the VOF techniques provides a means of following fluid regions through an Eulerian mesh of stationary cells. The method uses a minimum of stored information, and because it follows regions rather than boundaries, it avoids all logic problems associated with intersecting surfaces. The VOF method is also readily extendible to three-dimensional computations, where its conservative use of stored information is particularly advantageous.

In principle, the VOF method could be used to track any surface of discontinuity in material properties, in tangential velocity, or any other property. The particular case being represented determines the specific boundary conditions that must be applied at the location of the boundary. For situations where the surface does not remain fixed in the fluid, but has some additional relative motion, the equation of motion, Eq. (1), must be modified. Examples of such applications are shock waves, chemical reaction fronts, and boundaries between single- and two-phase fluid regions.

A description of the SOLA-VOF algorithm is contained in Sec. II. Program details, including a list of input variables and subroutines, are contained in Sec. III. In Sec. IV a test problem is provided to aid in the checkout of the program when used at other installations. The program is listed in App. A, and finally, in App. B, sample calculations are presented that illustrate some of the many capabilities of the SOLA-VOF program.

II. SOLA-VOF

Eulerian finite-difference methods for computing the dynamics of incompressible fluids are well established. The first method to successfully treat problems involving complicated free surface motions was the Marker-and-Cell (MAC) method.[1] This method was also the first technique to use pressure and velocity as the primary dependent variables. MAC employed a distribution of marker particles to define fluid regions, and simply set free surface pressures at the centers of cells defined to contain the surface. No attempt was made to apply the pressure boundary condition at the actual location of the boundary with-

in the surface containing cell. This crude approximation was later improved,[7] and marker particles were eliminated in favor of particle chains on the free surfaces.[3]

A simplified version of the basic solution algorithm (SOLA) used in the MAC method is available[8] in a user-oriented code called SOLA. Although SOLA does not treat free surfaces, an extended version, SOLA-SURF, is also available[8] that uses a surface height function method. The basic simplicity and flexibility of the SOLA codes make them excellent foundations for the development of more sophisticated codes. For this reason, a variable mesh version of the SOLA code, SOLA-VM, was chosen as a basis for the VOF technique. An experimental version of this new code, SOLA-VOF, was first reported in Ref. 9. Since that time, many improvements have been made and the basic technique has matured through applications to a wide class of problems. In a related development, McMaster, et al. have recently combined the SOLA-SURF code with a different interface tracking technique based on a VOF-like concept.[10,11]

The following subsections provide details of the SOLA-VOF solution algorithm with particular attention given to the special considerations needed in making finite-difference approximations in nonuniform meshes and to the VOF algorithms for advection and locating interfaces.

A. Outline

SOLA-VOF uses an Eulerian mesh of rectangular cells having variable sizes, δx_i for the i^{th} column and δy_j for the j^{th} row (Fig. 1). Although it is not as flexible as a mesh composed of arbitrary quadrilaterals, the variable mesh capability of SOLA-VOF gives it a considerable advantage for localized resolution over methods using equal-sized rectangles.

The fluid equations to be solved are the Navier-Stokes equations

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} + v\frac{\partial u}{\partial y} = -\frac{1}{\rho}\frac{\partial p}{\partial x} + g_x + \nu\left[\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} + \xi\left(\frac{1}{x}\frac{\partial u}{\partial x} - \frac{u}{x^2}\right)\right] \tag{2}$$

and

$$\frac{\partial v}{\partial t} + u\frac{\partial v}{\partial x} + v\frac{\partial v}{\partial y} = -\frac{1}{\rho}\frac{\partial p}{\partial y} + g_y + \nu\left[\frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2} + \frac{\xi}{x}\frac{\partial v}{\partial x}\right] . \tag{3}$$

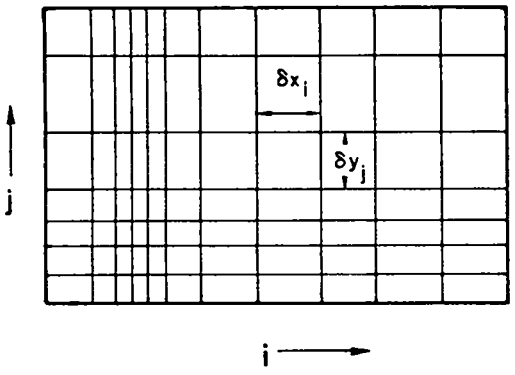

Fig. 1.
A finite-difference mesh with variable rectangular cells.

Velocity components (u,v) are in the Cartesian coordinate directions (x,y) or cylindrical coordinate directions (r,z), respectively. The choice of coordinate system is governed by the value of ξ, where $\xi = 0$ corresponds to Cartesian and $\xi = 1$ to cylindrical geometry. Body accelerations are denoted by (g_x, g_y) and ν is the coefficient of kinematic viscosity. Fluid density is denoted by ρ. In a one-fluid calculation with free surfaces, the ρ in Eq. (2) and Eq. (3) is assumed to be a constant. In two-fluid calculations, ρ may be equal to a different constant in each fluid, but assumes intermediate values in mesh cells containing an interface. For an incompressible fluid, the momentum equations, Eq. (2) and Eq. (3), must be supplemented with the incompressibility condition,

$$\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} + \frac{\xi u}{x} = 0 \quad . \tag{4}$$

Sometimes, it is desirable to allow limited compressibility effects[5] (e.g., acoustic waves), in which case Eq. (4) must be replaced with

$$\frac{1}{\rho c^2}\frac{\partial p}{\partial t} + \frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} + \frac{\xi u}{x} = 0 \quad , \tag{5}$$

where c is the adiabatic speed of sound in the fluid. Because Eq. (5) adds more flexibility with little additional complexity, it is incorporated as a standard feature in SOLA-VOF.

Discrete values of the dependent variables, including the fractional volume of fluid (F) variable used in the VOF technique, are located at cell positions shown in Fig. 2.

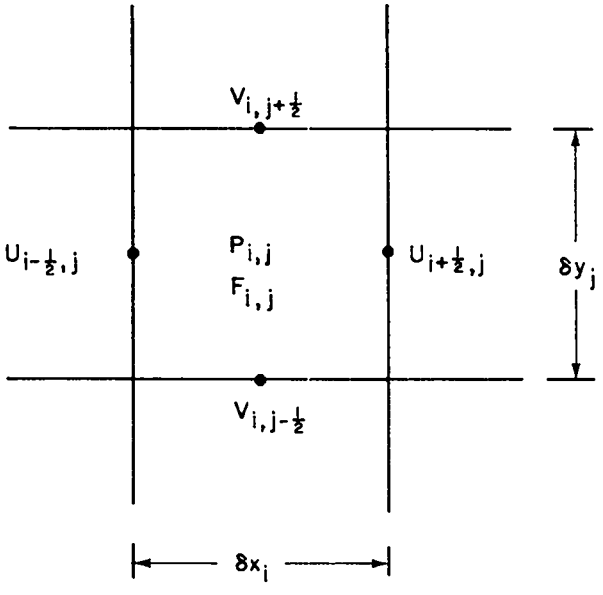

Fig. 2.
Location of variables in a typical mesh cell.

The volume of fluid function F is used to identify mesh cell that contain fluid of density ρ_F. A free surface or interface cell (i,j) is defined as a cell containing a nonzero value of F and having at least one neighboring cell (i ± 1,j) or (i,j ± 1) that contains a zero value of F. Cells with zero F values are empty or contain material of density ρ_c. Cells with nonzero F values and no empty neighbors are treated as cells full of ρ_F fluid. The SOLA-VOF code also has provisions for defining any cell or combination of cells in the mesh to be obstacle cells into which fluid cannot flow.

The choice of whether a one- or two-fluid calculation is performed is controlled by setting the integer input parameter NMAT to the number of fluids desired. When a one-fluid calculation is performed (NMAT=1), the fluid density is ρ_F in regions defined by nonzero values of F and the value of ρ_c is immaterial.

Briefly, the basic procedure for advancing a solution through one increment in time, δt, consists of three steps:

(1) Explicit approximations of Eq. (2) and Eq. (3) are used to compute the first guess for new time-level velocities using the initial conditions or previous time-level values for all advective, pressure, and viscous accelerations.

(2) To satisfy the continuity equation, Eq. (5), pressures are iteratively adjusted in each cell and the velocity changes induced by each pressure change are added to the velocities computed in step (1). An iteration is needed because the change in pressure needed in one cell to satisfy Eq. (5) will upset the balance in the four adjacent cells.

(3) Finally, the F function defining fluid regions must be updated to give the new fluid configuration.

Repetition of these steps will advance a solution through any desired time interval. At each step, of course, suitable boundary conditions must be imposed

at all mesh and free-surface boundaries. Details of these steps and boundary conditions are given in the following subsections.

B. Momentum Equation Approximations

In the following, the notation $Q^n_{i,j}$ stands for the value of Q(x,y,t) at time $n\delta t$ and at a location centered in the i^{th} cell in the x-direction and j^{th} cell in the y-direction. Half integer subscripts refer to cell boundary locations. For example, $Q^n_{i,j+\frac{1}{2}}$ refers to the value of Q on the boundary between the j and j + 1 cells in the y-direction.

A generic form for the finite-difference approximation of Eq. (2) and Eq. (3) is

$$u^{n+1}_{i+\frac{1}{2},j} = u^n_{i+\frac{1}{2},j} + \delta t \left[- \left(p^{n+1}_{i+1,j} - p^{n+1}_{i,j} \right) / \delta\rho x_{i+\frac{1}{2}} + g_x - FUX - FUY + VISX \right]$$

and (6)

$$v^{n+1}_{i,j+\frac{1}{2}} = v^n_{i,j+\frac{1}{2}} + \delta t \left[- \left(p^{n+1}_{i,j+1} - p^{n+1}_{i,j} \right) / \delta\rho y_{j+\frac{1}{2}} + g_y - FVX - FVY + VISY \right],$$

where

$$\delta\rho X_{i+\frac{1}{2}} = \frac{1}{2} \left\{ \left[\rho_c + (\rho_F - \rho_c) F_{i,j} \right] \delta x_{i+1} + \left[\rho_c + (\rho_F - \rho_c) F_{i+1,j} \right] \delta x_i \right\}$$

and

$$\delta\rho y_{j+\frac{1}{2}} = \frac{1}{2} \left\{ \left[\rho_c + (\rho_F - \rho_c) F_{i,j} \right] \delta y_{j+1} + \left[\rho_c + (\rho_F - \rho_c) F_{i,j+1} \right] \delta y_j \right\} .$$

The advective and viscous acceleration terms have an obvious meaning, e.g., FUX means the advective flux of u in the x-direction, etc. These terms are all evaluated using the old time level (n) values for velocities. Because the pressures at time level n + 1 are not known at the beginning of the cycle, Eq. (6) cannot be used directly to evaluate (u^{n+1}, v^{n+1}), but must be combined with the continuity equation as described below. In the first step of a solution, therefore, the p^{n+1} values in these equations are replaced by p^n to get a first guess for the new velocities.

In the basic solution procedure, the specific approximations chosen for the advective and viscous terms in Eq. (6) are relatively unimportant, provided they lead to a numerically stable algorithm. Special care must be exercised, however, when making approximations in a variable mesh like that of Fig. 1. The problem is best illustrated by considering the procedure used in the original MAC method[1] for Cartesian coordinates. In the MAC method, Eq. (2) through Eq. (4) were first combined so that the convective flux terms could be written in a divergence form (i.e., $\nabla\cdot\underline{uu}$ instead of $\underline{u}\cdot\nabla\underline{u}$). Thus, FUX would be, for example, $\frac{\partial u^2}{\partial x}$ rather than $u\frac{\partial u}{\partial x}$. The divergence form was preferred in MAC because it provided a simple way to insure conservation of momentum in the difference approximations. This may be seen by considering the control volume used for $u_{i+\frac{1}{2},j}$ that is indicated by dashed lines in Fig. 3. With the divergence form, Gauss' theorem may be used to convert the integrated value of FUX over the control volume to boundary fluxes at its sides. Then, the flux leaving one control volume will automatically be gained by the adjacent one and conservation during advection is guaranteed.

Unfortunately, conservation in a variable mesh does not automatically imply accuracy. To see this, suppose an upstream or donor-cell difference approximation is used for FUX = $\partial u^2/\partial x$, which is known to provide a conditionally stable algorithm. Assuming the u velocity is positive, the donor-cell approximation is

$$\text{FUX} = \left[u_{i+1,j} < u_{i+1,j} > - u_{i,j} < u_{i,j} > \right] / \delta x_{i+\frac{1}{2}} \quad , \tag{7}$$

where, e.g.,

$$u_{i,j} = (u_{i-\frac{1}{2},j} + u_{i+\frac{1}{2},j})/2$$

and

$$< u_{i,j} > = \begin{cases} u_{i-\frac{1}{2},j} \ , & \text{if } u_{i,j} \geq 0 \\ u_{i+\frac{1}{2},j} \ , & \text{if } u_{i,j} < 0 \ . \end{cases}$$

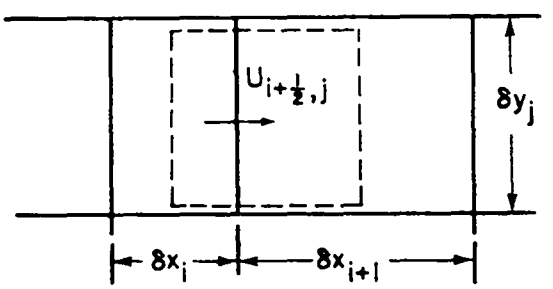

Fig. 3.

Control volume (dashed rectangle) used for constructing a finite-difference approximation for the u momentum equation at location $(i + \frac{1}{2}, j)$.

Expanding Eq. (7) in a Taylor series about the location $x_{i+\frac{1}{2}}$, where the u-equation is evaluated, yields

$$FUX = \frac{1}{2}\left(\frac{3\delta x_i + \delta x_{i+1}}{\delta x_i + \delta x_{i+1}}\right)\frac{\partial u^2}{\partial x} + O(\delta x) \ . \tag{8}$$

Thus, the zeroth order term is incorrect unless the cell widths are equal, $\delta x_i = \delta x_{i+1}$. In other words, the variable mesh reduces the order of approximation by one, and in this case leads to an incorrect zeroth order result. If a centered rather than a donor-cell approximation had been used, the result would have been first order accurate and not second order, as it is in a uniform mesh.

It does not follow, however, that variable meshes are necessarily less accurate because they do allow finer zoning in localized regions where flow variables are expected to vary most rapidly. Nevertheless, variable meshes must be used with care. It is best, for example, to allow for gradual variations in cell sizes to minimize the reduction in approximation order. It is also worthwhile to look for other approximations that do not lose their accuracy in a variable mesh. In this regard, it should be noted that the reason the conservation form of the advective terms lose accuracy is because the control volumes are not centered about the positions where variables are located. Because of this, the advective terms should be corrected to account for the difference in locations of the variables being updated and the centroids of their control volumes. When this is not done, a lower order error is introduced.

The stability advantages of the donor cell method can be retained in a variable mesh with no reduction in formal accuracy, if the $\underline{u}\cdot\nabla\underline{u}$ form is used for the advection flux. At the same time, it is also possible to combine the donor-cell and centered-difference approximations into a single expression with a parameter α that controls the relative amount of each one. The general form at $(i+\frac{1}{2},j)$ is

$$FUX = \left(u_{i+\frac{1}{2},j}/\delta x_\alpha\right)\left[\delta x_{i+1}\ DUL + \delta x_i\ DUR + \alpha \mathrm{sgn}(u)\ \left(\delta x_{i+1}\ DUL - \delta x_i\ DUR\right)\right], \tag{9}$$

where

$$DUL = (u_{i+\frac{1}{2},j} - u_{i-\frac{1}{2},j})/\delta x_i \quad ,$$

$$DUR = (u_{i+3/2,j} - u_{i+\frac{1}{2},j})/\delta x_{i+1} \quad ,$$

$$\delta x_\alpha = \delta x_{i+1} + \delta x_i + \alpha \mathrm{sgn}(u)\ (\delta x_{i+1} - \delta x_i) \quad ,$$

and sgn(u) means the sign of $u_{i+\frac{1}{2},j}$. When $\alpha = 0$, this approximation reduces to a second order accurate, centered-difference approximation. When $\alpha = 1$, the first order donor-cell form is recovered. Thus, using the approximation defined in Eq. (9), there is no loss of formal accuracy when a variable mesh is used.

The basic idea used in Eq. (9) is to weight the upstream derivative of the quantity being fluxed more than the downstream value. The weighting factors are $1 + \alpha$ and $1 - \alpha$, for the upstream and downstream derivatives, respectively. The derivatives are also weighted by cell sizes in such a way that the correct order of approximation is maintained in a variable mesh. This type of approximation is used in SOLA-VOF for all advective flux terms appearing in Eq. (6). Viscous accelerations are approximated with standard centered approximations.

C. Continuity Equation Approximation

Velocities computed from Eq. (6) must satisfy the continuity equation, Eq. (5). In order to satisfy this equation, the pressures (and velocities) must be adjusted in each computational cell occupied by fluid. The finite-difference form used for Eq. (5) is

$$\left(p_{i,j}^{n+1} - p_{i,j}^{n}\right)/(\rho\, c^2 \delta t) + D_{i,j}^{n+1} = 0 \quad , \tag{10}$$

where

$$D^{n+1}_{i,j} = \left(u^{n+1}_{i+\frac{1}{2},j} - u^{n+1}_{i-\frac{1}{2},j}\right) / \delta x_i + \left(v^{n+1}_{i,j+\frac{1}{2}} - v^{n+1}_{i,j-\frac{1}{2}}\right) / \delta y_j$$

$$+ \xi\left(u^{n+1}_{i+\frac{1}{2},j} + u^{n+1}_{i-\frac{1}{2},j}\right) / (2x_i) \quad ,$$

and

$$\rho = \rho_F F_{i,j} + \rho_c(1 - F_{i,j}) \quad .$$

Note that only one c^2 value is used in the program. In other words, it is assumed that the sound speeds of both fluids in a two-fluid calculation are equal. Because the velocities appearing in D are evaluated at the new time level, which depend on the n + 1 level pressures according to Eq. (6), this equation is an implicit relation for the new pressures. A solution may be obtained by the following iterative process. The computational mesh is swept row by row starting with the bottom and working upward. In each cell containing fluid, but not a free surface, the pressure change needed to satisfy Eq. (10) is

$$\delta p = - S/(\partial S/\partial p) \quad , \tag{11}$$

where S, denoting the left side of Eq. (10), is evaluated with the most updated values of p that are available, and the derivative is with respect to $p_{i,j}$. The new estimate for the cell pressure is then

$$P_{i,j} + \delta p \quad , \tag{12}$$

and new estimates for the velocities located on the sides of the cell are

$$u_{i+\frac{1}{2},j} + \delta t\Omega \; \delta p/\delta\rho x_{i+\frac{1}{2}}$$

$$u_{i-\frac{1}{2},j} - \delta t\Omega \; \delta p/\delta\rho x_{i-\frac{1}{2}}$$

$$v_{i,j+\frac{1}{2}} + \delta t\Omega \; \delta p/\delta\rho y_{j+\frac{1}{2}}$$

$$v_{i,j-\frac{1}{2}} - \delta t \Omega \, \delta p / \delta y_{j-\frac{1}{2}} \quad , \tag{13}$$

where the velocities are the most updated values available. The Ω factor controls the amount of implicitness used in solving Eq. (10) through Eq. (13) as described in Sec. II.D.

In one-fluid calculations a similar procedure is used in cells containing a free surface, except that the S used in Eq. (11) is not the left side of Eq. (10), but a relation that leads to the proper free-surface boundary condition when driven to zero by the iteration.[3] The boundary condition is satisfied by setting the surface cell pressure ($p_{i,j}$) equal to the value obtained by a linear interpolation between the pressure wanted at the surface (p_s) and a pressure inside the fluid (p_N). For this scheme to work, the adjacent cell chosen for the interpolation should be such that the line connecting its center to the center of the surface cell is closest to the normal to the free surface. Then the S function giving this result is

$$S = (1 - \eta) \, p_N + \eta \, p_s - p_{i,j} \quad , \tag{14}$$

where $\eta = d_c/d$ is the ratio of the distance between the cell centers and the distance between the free surface and the center of the interpolation cell (Fig. 4).

A complete iteration, therefore, consists of adjusting pressures and velocities in all cells occupied by fluid according to Eq. (11) through Eq. (13), where S is given by Eq. (10) for an interior cell and by Eq. (14) for a surface cell. Convergence of the iteration is achieved when all cells have S values whose magnitudes are below some small number, ε. Typically, ε is of order 10^{-3}, although it can vary with the problem being solved and the units chosen for the problem.

In some cases, convergence of the iteration can be accelerated by multiplying δp from Eq. (11) by an over-relaxation factor ω. A value of ω that is often optimum is 1.8, but in no case should it exceed 2.0; otherwise, an unstable iteration results.

In practice, the free-surface condition, Eq. (14), leads to an over-relaxation type of instability when the interpolation factor ω is greater than one. Stability can be insured by under-relaxing the pressure variations in cells used

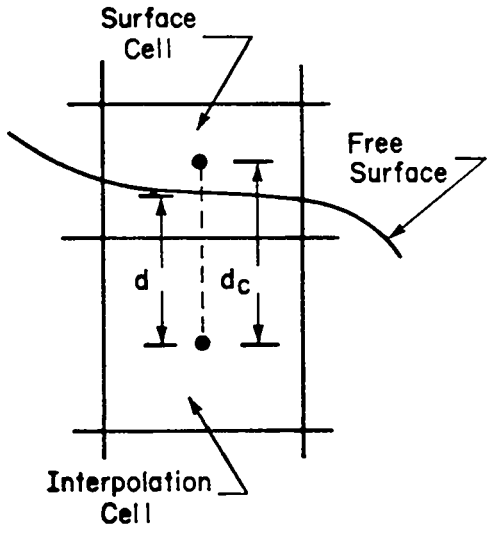

Fig. 4.

Definition of quantities used in defining free-surface pressure boundary condition.

as interpolation neighbors for surface cells. In particular, the relaxation factor ω used in a cell acting as an interpolation neighbor for a surface cell must be replaced with

$$\frac{\omega}{1 - \omega(1 - \eta)\delta t\ R'} \tag{15}$$

where

$$R = \left(\frac{\delta S}{\delta P}\ \Delta x\ d_c\right)^{-1} .$$

Here η and d_c refer to the surface cell and the S derivative is the value for the neighbor cell. Also Δx is δx of the surface cell if the neighbor lies in the x-direction; otherwise, Δx is equal to δy of the surface cell. The idea behind (15) is that the pressure change in the neighbor cell is coupled to the pressure in the surface cell, which in turn is dependent on the neighbor cell pressure through the linear interpolation, Eq. (14). To insure stability, this feedback type of coupling of the surface cell on its neighbor cell can be algebraically computed and used to define the stable relaxation limit, Eq. (15).

D. Implicit vs Explicit Calculations

The continuity equation has been numerically formulated as an implicit relation involving pressures from all fluid-occupied cells. This is necessary when the fluid is incompressible, because then a change in any one-cell pressure can influence the pressure in all other cells. However, when the limited compressibility option is used, compression waves will only travel a distance $C\delta t$ in one time step, where C is the sound speed. For sufficiently small δt, this distance may be less than the width of a computational mesh cell. In this limit, the implicit solution method for the continuity equation is no longer required and, in fact, is less accurate and more time-consuming than some simple explicit schemes. To see why this is so it is sufficient to investigate a linearized form of the equations, on which a linear stability analysis can be per-

formed. Consider a one-dimensional case, e.g., $P^n \sim A^n \exp\{ikx\}$. In the fully implicit case ($\Omega = 1$), as formulated in the previous section and with $\delta x = \delta y =$ constant, the amplification factor A is found to have the modulus

$$|A|^2 = \frac{1}{1 + \mu^2} ,$$

where $\mu = 2(C\delta t/\delta x) \sin(k\delta x/2)$. Thus, the implicit method is always stable because $|A|$ is always less than unity. Unfortunately, this also means that all Fourier components are damped in time unless μ is vanishingly small.

In contrast, if we had chosen $\Omega = 0$ so that velocities in Eq. (13) are not altered by the pressure iteration, then Eq. (10) becomes an explicit expression for each $P^{n+1}_{i,j}$ and the amplification factor determined from the linear stability analysis has a modulus identically equal to unity if $C\delta t \leq \delta x$, and greater than unity when $C\delta t > \delta x$. Therefore, the explicit method is conditionally stable ($C\delta t \leq \delta x$), and when operated in the stable range, introduces no damping of any Fourier components. The absence of all damping is not completely desirable, however, because there is still dispersion that can result in some high frequency numerical oscillations.

Ideally, a numerical method is wanted that only damps the highest frequencies (the shortest wavelengths of order $2\delta x$) to suppress dispersion-induced oscillations, but does not damp the longer and better resolved wave components. In SOLA-VOF, this goal can be approximated by properly choosing Ω as a function of $C\delta t/\delta x$. In particular, we have written the program so that $\Omega = \min(\mu_o^2, 1.0)$, where μ_o is the maximum value of $2C\delta t/\delta x_i$ and $2C\delta t/\delta y_j$ in the mesh. For purely incompressible calculations, Ω is always set to unity. With this prescription the linear stability analysis indicates that the amplification factor modulus will be

$$|A|^2 = \frac{1}{1 + \Omega\mu^2} .$$

Thus, when δt is reduced below the value needed for an explicit calculation to be stable, Ω rapidly decreases and $|A|$ approaches unity even more rapidly. Also, the calculations become more explicit as Ω is reduced and so less work is required to obtain a solution of the continuity equations.

Strictly speaking, the last remark is not correct unless the overrelaxation factor ω is also reduced to unity. This is because the explicit calculation ($\Omega = 0$) in Eq. (10) can be solved exactly with one iteration because it is linear in the unknown pressure. However, when ω is not unity, the correct pressure is altered by ω and several iterations are then required to relax it back to its correct value. The purpose of ω is to increase the convergence rate of the implicit form of Eq. (10), where changes in a cell pressure can influence neighboring cell pressures. Consequently, if we are to gain a computational advantage by allowing Ω to decrease for more explicit calculations when δt is small, it is also necessary to simultaneously push ω toward unity. This is done in the present version of SOLA-VOF by internally setting

$$\omega = (\omega_o - 1.0)\ \Omega + 1.0 \quad ,$$

where ω_o is the value of ω set in the input data.

In summary, the program internally determines when explicit calculations can be performed and automatically reduces the implicitness in such a way that calculations become more efficient and more accurate.

E. Approximations for Volume of Fluid Function

1. Advancing F in Time. The VOF function F is governed by Eq (1). For an incompressible fluid, Eq. (4) may be combined with Eq. (2) to yield

$$\frac{\partial F}{\partial t} + \frac{1}{r}\frac{\partial rFu}{\partial x} + \frac{\partial Fv}{\partial y} = 0 \quad , \tag{16}$$

where $r = x$ when $\xi = 1$ and $r = 1$ when $\xi = 0$. Even when the fluid is slightly compressible and Eq. (5) replaces Eq. (4), this equation for F is still an acceptable approximation. Equation (16), which is in divergence form, is here more convenient for numerical approximation and is the form used in the following discussion. When Eq. (16) is integrated over a computational cell, the changes in F in a cell reduce to fluxes of F across the cell faces. As previously noted, special care must be taken in computing these fluxes to preserve the sharp definition of free boundaries. The method employed in SOLA-VOF uses a type of Donor-Acceptor flux approximation.[6] The essential idea is to use information about F downstream as well as upstream of a flux boundary to establish a crude interface shape, and then to use this shape in computing the flux. Sever-

al researchers have previously used variations of this approach for tracking material interfaces.[6,12,13]

The basic method as developed for use in the VOF technique may be understood by considering the amount of F to be fluxed through the right-hand face of a cell during a time step of duration δt. The flux of volume crossing this cell face per unit cross sectional area is $V_x = u\delta t$, where u is the normal velocity at the face. The sign of u determines the donor and acceptor cells, i.e., the cells losing and gaining volume, respectively. For example, if u is positive the upstream or left cell is the donor and the downstream or right cell the acceptor. The amount of F fluxed across the cell face in one time step is δF times the face cross sectional area, where

$$\delta F = \text{MIN} \left\{ F_{AD} \, |V_x| + CF \, , \, F_D \, \delta x_D \right\} \tag{17}$$

and

$$CF = \text{MAX} \left\{ \left(1.0 - F_{AD}\right) |V_x| - \left(1.0 - F_D\right) \delta x_D \, , \, 0.0 \right\} \, .$$

Single subscripts denote the acceptor (A) and donor (D) cells. The double subscript, AD, refers to either A or D, depending on the orientation of the interface relative to the direction of flow as explained later.

Briefly, the MIN feature in Eq. (17) prevents the fluxing of more F from the donor cell than it has to give, while the MAX feature accounts for an additional F flux, CF, if the amount of void (1 - F) to be fluxed exceeds the amount available. Figure 5 provides a pictorial explanation of Eq. (17). The donor and acceptor cells are defined in Fig. 5a for fluxing across a vertical cell face. When AD = D, the flux is an ordinary donor cell value,

$$F = F_D \, |V_x| \, ,$$

in which the F value in the donor cell is used to define the fractional area of the cell face fluxing F (Fig. 5b). As discussed in Sec. II.I, numerical stability requires that $|V_x|$ be less than δx, so that it is not possible to empty the donor cell in this case.

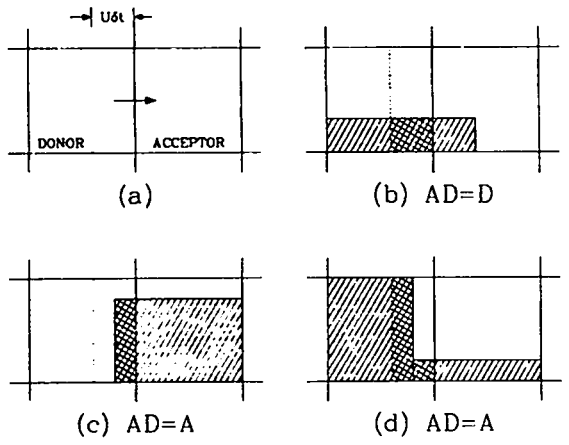

Fig. 5.

Examples of free-surface shapes used in the advection of F. The donor-acceptor arrangement is shown in (a) where the dashed line indicates the left boundary of the total volume being advected. The cross-hatched regions shown in (b-d) are the actual amounts of F fluxed.

When AD = A, the value of F in the acceptor cell is used to define the fractional area of the cell face across which F is flowing. In case (c) of Fig. 5, all the F fluid in the donor cell is fluxed because everything lying between the dashed line and the flux boundary moves into the acceptor cell. This is an example of the MIN test in Eq. (17). In case (d) of Fig. 5, more F fluid than the amount $F_A|V_x|$, must be fluxed, so this is an example of the MAX test. In particular, the extra fluid between the dashed line and the flux boundary is equal to the CF value in Eq. (17).

Whether the acceptor or donor cell is used to determine the fractional area for F flow depends on the mean surface orientation. The acceptor cell is used when the surface is advected mostly normal to itself; otherwise, the donor cell value is used. However, if the acceptor cell is empty of F or if the cell upstream of the donor cell is empty, then the acceptor cell F value is used to determine the flux regardless of the orientation of the surface. This means that a donor cell must fill before any F fluid can enter a downstream empty cell.

The reason for testing on surface orientation is that an incorrect steepening of surface waves will occur if the acceptor cell is always used to compute fluxes. Consider, for example, a horizontal surface with a small wave moving in the positive x-direction. A flux based on the downstream (acceptor) value of F will eventually steepen the wave into a step discontinuity. In effect, the acceptor method is numerically unstable because it introduces a negative diffusion of F (i.e., a diffusion-like transport with a negative coefficient). Instabilities do not grow unbounded; however, because of the MIN and MAX tests used in the flux definition. In contrast, when the surface is advecting normal to itself, a steepening that maintains the step-functiion character of F is exactly what is wanted.

Once the flux has been computed by the above method, it is multiplied by the flux boundary area to get the amount of F fluid to be subtracted from the donor cell and added to the acceptor cell. When the process is repeated for all cell boundaries in the mesh, the resulting F values correspond to the time-advanced values satisfying Eq. (16) and still sharply define all interfaces.

2. Bookkeeping Adjustments. The new F values determined by the above method ocassionally may have values slightly less than zero or slightly greater than unity. Therefore, after the advection calculation has been completed, a pass is made through the mesh to reset values of F less than zero back to zero and values of F greater than one back to one. Accumulated changes in fluid volume introduced by these adjustments during a calculation are recorded and may be printed out at any time.

There is one other adjustment needed in F so that it may be used as a boundary cell flag. Boundary cells have values of F lying between zero or one. However, in a numerical solution, F values cannot be tested against exact numbers like zero and one because roundoff errors would cause spurious results. Instead, a cell is defined to be empty of F when F is less than ε_F and full when F is greater than $1 - \varepsilon_F$, where ε_F is typically 10^{-6}. If, after advection, a cell has an F value less than ε_F, this F is set to zero and all neighboring full cells become surface cells by having their F values reduced from unity by an amount $1.1\varepsilon_F$. These changes in F are also included in the accumulated volume change. Volume errors after hundreds of cycles are typically observed to be a fraction of a per cent of the total F volume.

3. Determining Interfaces Within a Cell. For the accurate application of boundary conditions, knowledge of the boundary location within a surface cell is required. In the VOF technique, it is assumed that the boundary can be approximated by a straight line cutting through the cell. By first determining the slope of this line, it can then be moved across the cell to a position that intersects the known amount of F volume in the cell.

To determine the boundary slope, it must be recognized that the boundary can be represented either as a single-valued function Y(x) or as X(y), depending on its orientation. If the boundary is representable as Y(x), we must compute dY/dx. A good approximation to Y(x) is

$$Y_i = Y(x_i) = F(i,j-1)\delta y_{j-1} + F(i,j)\delta y_j + F(i,j+1)\delta y_{j+1} \quad ,$$

where Y = 0 has been taken as the bottom edge of the j - 1 row of cells. Then,

$$\left(\frac{dY}{dx}\right)_i = \left[(Y_{i+1} - Y_i)\delta x_{i-\frac{1}{2}}/\delta x_{i+\frac{1}{2}} + (Y_i - Y_{i-1})\delta x_{i+\frac{1}{2}}/\delta x_{i-\frac{1}{2}}\right]/(\delta x_{i-\frac{1}{2}} + \delta x_{i+\frac{1}{2}}) \ , \tag{18}$$

where, e.g., $\delta x_{i+\frac{1}{2}} = (\delta x_{i+1} + \delta x_i)/2$. A similar calculation can be made for dX/dy, where

$$X_j = X(y_j) = F(i-1,j)\delta x_{i-1} + F(i,j)\delta x_i + F(i+1,j)\delta x_{i+1} \ ,$$

and

$$\left(\frac{dX}{dy}\right)_j = \left[(X_{j+1} - X_j)\delta y_{j-\frac{1}{2}}/\delta y_{j+\frac{1}{2}} + (X_j - X_{j-1})\delta y_{j+\frac{1}{2}}/\delta y_{j-\frac{1}{2}}\right]/(\delta y_{j+\frac{1}{2}} + \delta y_{j-\frac{1}{2}}) \ . \tag{19}$$

If |dY/dx| is smaller than |dX/dy|, the boundary is more nearly horizontal than vertical, otherwise it is more nearly vertical. The derivative with the smallest magnitude gives the best approximation to the slope because the corresponding Y or X approximation is more accurate in that case.

Suppose |dY/dx| is smallest so the interface is more horizontal than vertical. If dX/dy is negative, F fluid lies below the boundary, and cell (i,j - 1) is used as the interpolation neighbor for the surface cell (i,j). Had dX/dy been positive, cell (i,j + 1) would be chosen for the neighboring interpolation cell because fluid would then be above the boundary.

Once the boundary slope and the side occupied by fluid have been determined, a line can be constructed in the cell with the correct amount of F volume lying on the F fluid side. This line is used as an approximation to the actual boundary and provides the information necessary to calculate η for the application of free-surface pressure boundary conditions, as described in Sec. II.C.

For cylindrical coordinates, the above computations are more complex because of the volume dependence on radius. Except for cells on the axis, however, the exact results differ little from the simpler Cartesian coordinate results; consequently, the latter are used in both cases.

F. Surface Tension and Wall Adhesion

Surface tension effects require little additional effort and are included in SOLA-VOF as an option.[14] The input parameter ISURF10 is set to unity when surface tension is wanted; otherwise, it is set to zero. The surface tension coefficient (a force per unit length) is denoted by σ.

Two essential steps are needed to include surface tension in a calculation. First, it is necessary to compute a local curvature in each boundary cell using the Y(x) or X(y) definitions from Eq. (18) and Eq. (19), so that a surface tension pressure can be evaluated. Once the curvature K is found, this surface tension pressure is given by

$$p_s = - \sigma K \quad .$$

Second, it is necessary to impose this pressure force on all interfaces.

In the first step, K is given by

$$K = K_{xy} + K_{cy\ell} = \frac{1}{R_{xy}} + \frac{1}{R_{cy\ell}} \quad ,$$

where R_{xy} is the principal radius of curvature in the x - y plane and $R_{cy\ell}$ is the principal radius of curvature associated with the azimuthal direction in the case of cylindrical coordinates. For plane coordinates, $K_{cy\ell} = 0$. When the interface is mostly horizontal, the curvature in the x - y plane is given by

$$K_{xy} = \frac{d}{dx} \left[\frac{\frac{dY}{dx}}{\sqrt{1 + \left(\frac{dY}{dx}\right)^2}} \right] \quad .$$

If the interface is mostly vertical, the roles of x and y must be reversed with X(y) replacing Y(x).

Numerical evaluation of K_{xy} must be done carefully because it contains a second derivative, which is sensitive to small errors in the first derivatives. We have found that the best technique is to evaluate surface slopes around the cell corners lying nearest to the surface. For example, to evaluate dY/dx at the right side of cell (i,j), we first determine whether the surface intersects

this side, $x = x_{i+\frac{1}{2}}$, nearer $y_{j+\frac{1}{2}}$ or $y_{j-\frac{1}{2}}$. If it is nearer the top corner at $y_{j+\frac{1}{2}}$, then dY/dx and dX/dy are evaluated about this corner. The derivative with the smallest magnitude is assumed to give the best estimate for the surface slope near that corner of the cell. In this way, we obtain a relatively accurate evaluation of curvature that is not sensitive to changes in interpolation cell neighbors or to unusual distributions of F.

To evaluate $K_{cy\ell}$, let θ be the angle between the interface tangent and the positive x axis when the interface is horizontal. If the interface is vertical, θ is the angle it makes with the positive y axis. In the near horizontal case,

$$K_{cy\ell} = \sin \theta / XI_i \quad ,$$

where XI_i is the distance in the x-direction from the axis of symmetry to the center of cell (i,j). For the near vertical case,

$$K_{cy\ell} = \cos \theta / r_{cy\ell} \quad ,$$

where

$$r_{cy\ell} = \begin{cases} -(X_{i-\frac{1}{2}} + F_{i,j}\,\delta x_i) & \text{, if fluid is left of interface} \\ X_{i+\frac{1}{2}} - F_{i,j}\,\delta X_i & \text{, if fluid is right of interface.} \end{cases}$$

Here $X_{i+\frac{1}{2}}$ is, e.g., the radius of the boundary between the i and i + 1 column of cells, and the minus sign used in the fluid left case insures the proper sign for the surface tension pressure.

This completes the necessary calculations for p_s except near walls where wall adhesion effects must be accounted for. A contact angle CANGLE must be specified in the input data list. If no wall adhesion is desired this angle is set to zero. In all other cases, CANGLE is the angle between the wall and the fluid interface that includes the F fluid. To impose this angle condition we adjust the Y(x) and X(y) values at the wall. For example, suppose the boundary cell (i,j) is as shown in Figs. 6a and 6b with F fluid assumed to be below the boundary. A surface tension pressure will be calculated in the same manner as

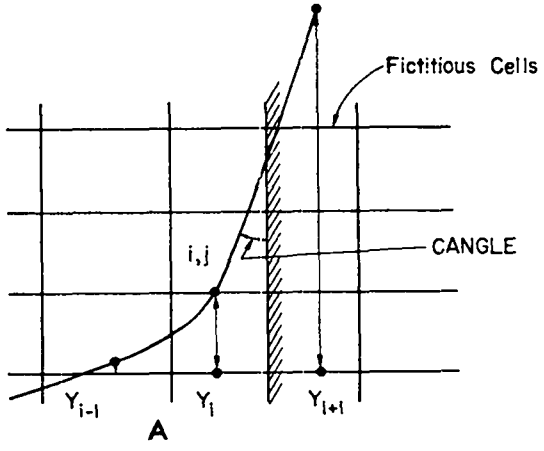

Fig. 6a.

Wall adhesion for the locally near-horizontal interface is modeled by setting the value of Y_{i+1} such that CANGLE is the angle between the wall and the interface.

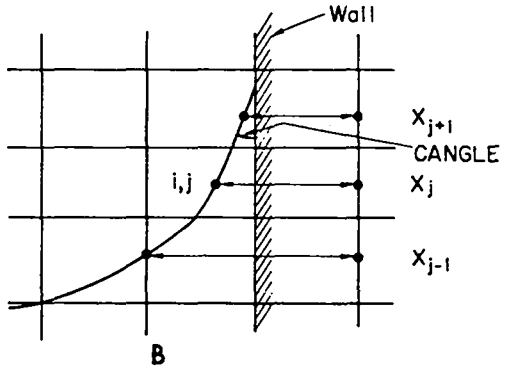

Fig. 6b.

Wall adhesion for the locally near-vertical interface is modeled by setting X_{j+1} such that CANGLE is the angle between the wall and the interface.

outlined above, except that the surface is assumed to make an angle CANGLE with the wall by adjusting the i + 1 value of Y(x) and the j + 1 value of X(y) as

$$Y_{i+1} = Y_i + 1/2(\delta x_i + \delta x_{i+1})/TANCA$$

and

$$X_{j+1} = X_j - 1/2(\delta y_j + \delta y_{j+1})TANCA \quad ,$$

where TANCA = tan(CANGLE). Similar adjustments are made to the appropriate variables for all other orientations of walls and interfaces. It should be noted that this procedure is implemented in SOLA-VOF not only for mesh boundary walls, but also at walls of internal obstacles constructed by blocking out combinations of mesh cells.

The second step required to add surface tension to a calculation is the manner in which p_s forces are added to the dynamics. In the case of a one-fluid free-surface calculation, p_s is used simply as an applied surface pressure according to the surface pressure equation, Eq. (14). In a two-fluid calculation,

no separate interface boundary conditions are used and so p_s forces must be added in some other way. In SOLA-VOF, these are included as additional accelerations affecting velocities on the sides of cells containing an interface, which is done as follows. First, assume the interface is near the horizontal. Then the y-direction force on the interface is

$$STFY = p_s \, \delta x_i \quad ,$$

and the force in the x-direction is

$$STFX = p_s \, \delta x_i \tan \theta \quad .$$

Accelerations produced by these forces are applied to the four velocity components at the sides of cell (i,j). For the near-horizontal case it is assumed that the forces act equally at the left and right cell sides, but an interpolation is used for the top and bottom forces. These forces, which are added to the right sides of Eq. (6) in a special subroutine (TMS10), are

$$u^{n+1}_{i+\frac{1}{2},j} = \ldots + \delta t \, \omega_R STFX/(\delta y_j \, \delta\rho x_{i+\frac{1}{2}}) \quad ,$$

$$u^{n+1}_{i-\frac{1}{2},j} = \ldots + \delta t \, \omega_L STFX/(\delta y_j \, \delta\rho x_{i-\frac{1}{2}}) \quad ,$$

$$v^{n+1}_{i,j+\frac{1}{2}} = \ldots + \delta t \, \omega_T STFY/(\delta x_i \, \delta\rho y_{j+\frac{1}{2}}) \quad ,$$

and

$$v^{n+1}_{i,j-\frac{1}{2}} = \ldots + \delta t \, \omega_B STFY/(\delta x_i \, \delta\rho y_{j-\frac{1}{2}}) \quad ,$$

where ω_R, etc. are the interpolation weighting factors.

A similar procedure is followed when the interface is near the vertical, except that δy replaces δx in the STFX and STFY forces and interpolation is used in the x-direction.

This procedure does allow some noise to be generated in the form of small local velocity perturbations along an interface. Generally, a small amount of viscosity (or smoothing automatically introduced by advection approximations) will keep this noise from causing any problems. The origin of the noise is associated with the fact that an interface can be a discontinuous vortex sheet, but the calculation cannot accurately resolve this.

G. Marker Particles

Sometimes it is desirable to mark specific regions of fluid so that the evolution (location and shape) of these regions can be followed. Marker particles are also useful for computer graphic displays (App. B). For this purpose, SOLA-VOF has an optional marker particle capability. Particles are points embedded in the fluid that move about with it, but do not affect the fluid dynamics.

To use this option, the user must program in his particular initial particle setup and number of particles in the SETUP subroutine at the locations marked for this purpose. If only one rectangular region of particles is desired, this can be initialized by input data (input data list Sec. III.B). When marker particles are not wanted, the parameters (NPX,NPY) should be set to zero (their default values).

Local velocities (u_p, v_p) used to move particles are computed from the nearest four mesh velocities in each coordinate direction using a standard bilinear interpolation. New particle coordinates after a time interval δt are obtained from the first-order difference expressions,

$$x_p^{n+1} = x_p^n + u_p \, \delta t$$

and

$$y_p^{n+1} = y_p^n + v_p \, \delta t.$$

Particles that cross a continuative or constant pressure boundary are deleted, and particles crossing a periodic boundary are reinserted at the opposite side of the computational mesh.

H. Boundary Conditions

1. Mesh Boundaries. In addition to the free-surface boundary conditions, it is necessary to set conditions at all mesh boundaries and at surfaces of all internal obstacles. At the mesh boundaries, a variety of conditions may be set using the layer of fictitious cells surrounding the mesh. Consider, for example, the left boundary. If this is a rigid free-slip wall, the normal velocity there must be zero and the tangential velocity should have no normal gradient, i.e.,

$$\left.\begin{aligned} u_{1,j} &= 0 \\ v_{1,j} &= v_{2,j} \\ p_{1,j} &= p_{2,j} \\ F_{1,j} &= F_{2,j} \end{aligned}\right\} \text{ for all } j.$$

If the left boundary is a no-slip rigid wall, then the tangential velocity component at the wall should also be zero, i.e.,

$$\left.\begin{aligned} u_{1,j} &= 0 \\ v_{1,j} &= -v_{2,j} \\ p_{1,j} &= p_{2,j} \\ F_{1,j} &= F_{2,j} \end{aligned}\right\} \text{ for all } j.$$

These conditions are imposed on the velocities computed from the momentum equations and after each pass through the mesh during the pressure iteration.

Continuative or outflow boundaries always pose a problem for low-speed calculations because whatever prescription is chosen can potentially affect the entire flow field. What is needed is a prescription that permits fluid to flow out of the mesh with a minimum of upstream influence. In SOLA-VOF, the continuative boundary conditions used at the left wall are

$$\left.\begin{aligned} u_{1,j} &= u_{2,j} \\ v_{1,j} &= v_{2,j} \\ p_{1,j} &= p_{2,j} \\ F_{1,j} &= F_{2,j} \end{aligned}\right\} \text{ for all } j.$$

These conditions, however, are only imposed after applying the momentum equations and not after each pass through the pressure iteration.

For periodic boundary conditions in the x-direction, the left and right boundaries must be set to reflect the periodicity. This is easiest when the period length is chosen equal to the distance from the left wall to the left boundary of the last interior cell in the mesh at the right side. That is, two columns of cells, i = IMAX and i = IMAX - 1, are reserved on the right side of the mesh for the setting of periodic boundary conditions. The conditions are then, on the left,

$$\left.\begin{aligned} u_{1,j} &= u_{IM2,j} \\ v_{1,j} &= v_{IM2,j} \\ F_{1,j} &= F_{IM2,j} \end{aligned}\right\} \text{ for all } j,$$

and on the right,

$$\left.\begin{aligned} u_{IM1,j} &= u_{2,j} \\ v_{IM1,j} &= v_{2,j} \\ p_{IM1,j} &= p_{2,j} \\ (p_s)_{IM1,j} &= (p_s)_{2,j} \\ v_{IMAX,j} &= v_{3,j} \\ F_{IMAX,j} &= F_{3,j} \\ F_{IM1,j} &= F_{2,j} \end{aligned}\right\} \text{ for all } j,$$

where IM1 = IMAX - 1 and IM2 = IMAX - 2. In this case, these conditions are imposed on velocities computed from the explicit momentum equations and after each pressure iteration.

A constant pressure boundary condition at the left wall is set by keeping the pressure in column i = 2 constant and otherwise treating the boundary as continuative.

Boundary conditions similar to those for the left wall are used at the right, top, and bottom boundaries of the mesh. Of course, the normal and tangential velocities at the top and bottom boundaries are v and u, respectively.

For convenience, the SOLA-VOF code has been written so that any of the above boundary conditions can be automatically imposed by setting input numbers. For example, for the left boundary

$$WL = \begin{cases} 1, \text{ rigid free-slip wall} \\ 2, \text{ rigid no-slip wall} \\ 3, \text{ continuative} \\ 4, \text{ periodic in x} \\ 5, \text{ specified pressure .} \end{cases}$$

In a similar fashion, WR controls the right boundary, WT the top boundary, and WB the bottom boundary.

Specified inflow and outflow conditions at mesh boundaries or at locations within the mesh must be programmed into the boundary condition subroutine of the code. A special location is reserved in the SOLA-VOF code for this purpose at the end of the regular boundary conditions.

2. Free-Surface Boundaries. In a one-fluid calculation, the inviscid free-surface boundary condition for normal stress is automatically satisfied by the implicit pressure calculation using Eq. (14). This condition must be supplemented with the specification of velocities immediately outside the surface, where these values are needed in the finite-difference approximations for points outside the surface. The specifications used in SOLA-VOF are identical to those used in many earlier MAC codes. Velocities must be set on every cell boundary between a surface cell and an empty cell. If the surface cell has only one neighboring empty cell, the boundary velocity is set to insure the vanishing of $D_{i,j}$, the velocity divergence defined in Eq. (10). When there are two or more empty cell neighbors, the individual contributions to the divergence, $\frac{1}{r}\frac{\partial ru}{\partial r}$ and $\frac{\partial v}{\partial y}$, are separately set to zero. Zero values for $\frac{\partial u}{\partial y}$ or $\frac{\partial v}{\partial x}$ are additionally used to set exterior tangent velocities to a free surface on boundaries between empty cells adjacent to a surface cell.

3. Internal-Obstacle Boundaries. The definition of internal obstacles is accomplished by flagging those cells of a mesh that are to be blocked out. Because the relaxation factor $BETA_{i,j}$ used in the pressure iteration must be positive, we can use negative values of this variable as a suitable flag for obstacle cells. A convenient, but arbitrary choice, is to assign a value of BETA = -1.0 to all obstacle cells. These flag values for BETA must be programmed into

the code for each application. A comment is included in the subroutine SETUP where this is to be done.

In the program no velocities or pressures are calculated in obstacle cells, and all velocity components on faces of obstacle cells are automatically set to zero. In the boundary condition subroutine values for volume fractions and pressures are set in all obstacle cells bordering fluid cells. These values are computed to be equal to the averages of these quantities in the adjacent fluid cells. All other obstacle cells have zero values for F and p. With this prescription, contour plots in fluid-occupied regions are smoothly drawn at obstacle boundaries. Even more important, this prescription prevents fluid-obstacle boundaries from being interpreted as free surfaces, as they would be without some sort of additional testing.

In addition, it should also be noted that because all velocity components within obstacle cells are set to zero, no-slip tangential velocity conditions at obstacle boundaries are only first-order accurate. That is, tangential velocities are zero at locations shifted into the obstacles one-half of a cell width from the actual boundary location.

I. Numerical Stability Considerations

Numerical calculations often have computed quantities that develop large, high-frequency oscillations in space, time, or both. This behavior is usually referred to as a numerical instability, especially if the physical problem being studied is known not to have unstable solutions. When the physical problem does have unstable solutions and if the calculated results exhibit significant variations over distances comparable to a cell width or over times comparable to the time increment, the accuracy of the results cannot be relied on. To prevent this type of numerical instability or inaccuracy certain restrictions must be observed in defining the mesh increments δx_i and δy_j, the time increment δt, and the upstream differencing parameter α.

For accuracy, the mesh increments must be chosen small enough to resolve the expected spatial variations in all dependent variables. When this is impossible because of limitations imposed by computing time or memory requirements, special care must be exercised in interpreting calculational results. For example, when computing the flow in a large chamber, it is usually impossible to resolve thin boundary layers along the confining walls. In many applications, however, the presence of thin boundary layers is unimportant and free-slip boundary conditions can be justified as a good approximation.

Once a mesh has been chosen, the choice of the time increment necessary for stability is governed by several restrictions. First, material cannot move through more than one cell in one time step because the difference equations assume fluxes only between adjacent cells. Therefore, the time increment must satisfy the inequality,

$$\delta t < \min \left\{ \frac{\delta x_i}{|u_{i,j}|} , \frac{\delta y_j}{|v_{i,j}|} \right\},$$

where the minimum is with respect to every cell in the mesh. Typically, δt is chosen equal to one-fourth to one-third of the minimum cell transit time. Second, when a nonzero value of kinematic viscosity is used, momentum must not diffuse more than approximately one cell in one time step. A linear stability analysis shows that this limitation implies

$$\nu \delta t < \frac{1}{2} \frac{\delta x_i^2 \, \delta y_j^2}{\delta x_i^2 + \delta y_j^2} .$$

Third, when surface tension is included there must also be a limit on δt to prevent capillary waves from traveling more than one cell width in one time step. A rough estimate for this stability condition is

$$\sigma \delta t^2 < \frac{\rho_m \, \delta x_m^3}{4(1 + CYL)} ,$$

where ρ_m is the minimum of ρ_F and ρ_c, δx_m is the minimum cell size in the mesh (either δx or δy), and CYL is 1.0 for cylindrical geometry and 0.0 for the Cartesian case. Finally, with δt chosen to satisfy the above inequalities, the last parameter needed to insure numerical stability is α. The proper choice for α is

$$1 \geqslant \alpha > \max \left\{ \left| \frac{u_{i,j} \delta t}{\delta x_i} \right| , \left| \frac{v_{i,j} \delta t}{\delta y_j} \right| \right\} .$$

As a rule of thumb, an α approximately 1.2 to 1.5 times larger than the right-hand member of the last inequality is a good choice. If α is too large, an unnecessary amount of numerical smoothing (diffusion-like truncation errors) may be introduced.[15]

All the above time-step controls are automatically satisfied by the program if the user selects this option (AUTOT=1.0) through the input data. Setting AUTOT=0.0 will permit a constant δt to be used. The AUTOT=1.0 option also adjusts the time step to keep pressure iterations at approximately 25 per cycle or less. In situations where the pressure iteration has not converged after 1000 iterations, or when an advective flux exceeds more than half the volume of a cell, the time step is automatically cut in half independent of the AUTOT selection. Further, in the case of an excessive advective flux, the cycle is restarted with the reduced δt.

III. PROGRAM DETAILS

A. Subroutines

The SOLA-VOF program is written in subroutine form such that each subroutine performs an individual task. A flow chart is given in Fig. 7 showing the calling sequence of the various subroutines. Subroutine names have been selected to indicate the function that each performs, and these are arranged alphabetically following the main program routine, SOLA-VOF. Each subroutine is listed here with a brief description of its major function.

SOLA-VOF (main program)

(1) Reads and prints the input and output data.

(2) Contains the calling sequences to the other subroutines and thus provides cyclic control over the calculation.

(3) t → t + DELT.

(4) Increments the cycle number by one each cycle.

(5) Provides a shutdown procedure in the event that a solution cannot be obtained that satisfies mass conservation.

BC (Boundary Conditions)

(1) Sets the values of appropriate variables at rigid free-slip, no-slip, continuative outflow, periodic, and specified pressure boundaries.

(2) Sets the values of appropriate variables around the boundaries established by free surfaces.

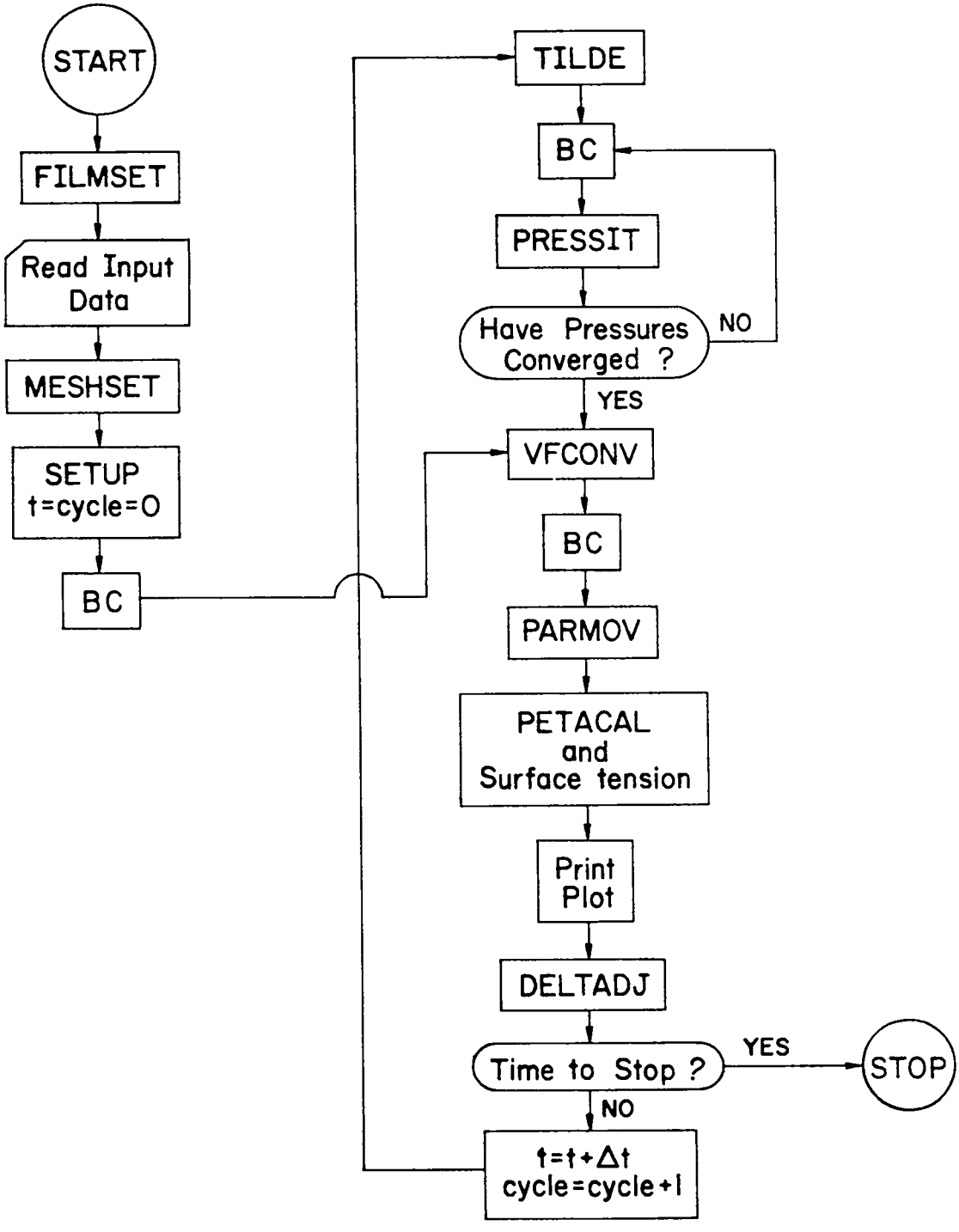

Fig. 7.

Flow chart for SOLA-VOF

(3) Allows for special boundary condition inclusions, such as inflow boundaries; these must be included by hand as needed for each problem.

(4) Sets average F and p values in obstacle cells adjacent to fluid cells.

CAVOVO (CAlculates VOid VOlumes)

(1) Computes the volume of disjoint void (F = 0) regions.

DELTADJ (Time step ADJustment)

(1) Computes maximum allowable δt for stability.

(2) Adjusts δt according to number of iterations and maximum allowed for stability.

(3) Recomputes relaxation factors BETA.

DRAW (Generates graphics output of problem data)

(1) Draws velocity vector and free-surface distributions.

(2) Draws the mesh during CYCLE = 0.

(3) Provides particle plots as requested.

DRWOBS (DRaW OBStables)

(1) Draws lines around all obstacles.

DRWVEC (DRaW a VECtor)

(1) Provides the graphics package system call to draw a line between points (x1,y1) and (x2,y2).

(2) Computes and plots the symmetric form of a given line if ISYMPLT is turned on (i.e., = 1).

FILMSET (FILM SET-up)

(1) Provides the necessary buffers, links, and logical units for the use of local graphics software. This routine must be written specifically for each graphics system.

FRAME

(1) Draws a frame around graphics output (the frame size is scaled to the mesh size).

LAVORE (LAbel VOid REgions)

(1) Detects and labels all disjoint void (F = 0) regions. Labels are NF values 6,7,8, etc.

MESHSET (MESH SET-up)

(1) Generates the computing mesh from the input data established in NAMELIST/MSHSET/.

(2) Evaluates all of the necessary geometric variables that are used throughout the code.

PARMOV (PARticle MOVement)

(1) Computes the movement of marker particles in the fluid velocity field.

(2) Provides the necessary bookkeeping to allow marker particles to exit the mesh or to be replaced by newly input particles. (NOTE: Particle initializations must be done by hand in the SETUP subroutine, except for the case of a single rectangular region, which can be defined by input data.)

PETACAL (PETA interpolation factor CALculation)

(1) Determines the slope of the surface in the surface cells.

(2) Determines the cell flag NF(i,j) to indicate the interpolation neighbor of the surface cell. (The interpolation neighbor is the cell adjacent to the surface cell containing fluid and with which surface cell pressures are interpolated to provide the proper value of P at the surface.)

NF = 1, implies the neighbor to the left.

NF = 2, neighbor to the right.

NF = 3, neighbor on the bottom.

NF = 4, neighbor on the top.

Thus, the above information describes the orientation of the surface (whether vertical or horizontal) and on which side of this surface fluid exists.

(3) Computes the surface pressure PS(i,j) caused by surface tension in surface cells, if the surface tension flag is set.

PLTPT (PLoT a PoinT)

(1) Provides the graphics system call to plot a single point (x1,y1).

(2) Computes and plots the symmetric point to be plotted if the flag ISYMPLT is on. (Symmetry is always assumed to exist only about the y axis.)

PRESSIT (PRESSure ITeration)

(1) Iterates the velocity and pressure field such that mass is conserved in each cell of the mesh.

(2) Computes a free-surface cell pressure adjustment based on the applied surface pressure; mass conservation in the surface is not iterated, it is set by application of the free-surface boundary conditions.

PRTPLT (PRinTs and PLoTs)

(1) Outputs data for paper prints and film plots.

SETUP (general SETUP)

(1) Initializes constants necessary to the calculation.

(2) Computes the scaling factors and centering shifts required for graphics output.

(3) Computes the initial hydrostatic pressure distribution to initialize the P(i,j) pressure array.

(4) Initializes marker particle number.

(5) Sets up the initial velocity with U(i,j) = UI and V(i,j) = VI everywhere in the mesh.

(6) Computes the relaxation factors (BETA(i,j)) that are used in the pressure iteration.

(7) Sets up obstacles by defining obstacle cells as having BETA(i,j) = -1.0. Obstacle definition, in general, must be coded by hand for each problem.

TILDE (Temporary Velocity Calculation)

(1) Computes an explicit solution for each of the momentum equations, (i.e., new values of velocities are obtained from the time n values of pressure, advective, and diffusive accelerations). These tilde values are advanced to time n + 1 in the pressure iteration.

TMS10 (Two Material Surface Tension)

(1) Adds interfacial surface tension forces to velocities near interface.

VFCONV (Volume Fraction CONVection)

(1) Computes the solution to the equation

$$\frac{\partial F}{\partial t} + \nabla \cdot \underset{\sim}{u}F = 0 \ .$$

(2) Computes and stores for printout any errors in volume (i.e., loss or gain) during the calculation of step (1).

B. Basic Input Variables

All of the input data to the SOLA-VOF code are in NAMELIST blocks with the exception of the problem title NAME, which is a 10A8 format. The NAMELIST variables occur in two blocks. The first block is /XPUT/ and contains all of the physical parameters necessary to specify and run a problem. The second block /MSHSET/ contains the geometrical information necessary for the creation of the variable mesh and the number of cells used in it.

The following is a list of all the input variables in the first block and a description of each. This same list is also reproduced at the beginning of the program listing.

1. 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 adjustments, =0.0 for constant DELT)
CANGLE	Contact angle, in degrees, between fluid and wall
CSQ	Material sound speed squared (=-1.0 for incompressible material)
DELT	Time step
EPSI	Pressure iteration convergence criterion
FLHT	Fluid height, in y-direction
GX	Body acceleration in positive x-direction
GY	Body acceleration in positive y-direction
ICYL	Mesh geometry indicator (=1 for cylindrical coordinates, =0 for plane coordinates)
IMOVY	Movie indicator (=1 for movie film output, =0 for other film output)
ISURF10	Surface tension indicator (=1 for surface tension, =0 for no surface tension)
ISYMPLT	Symmetry plot indicator (=1 for symmetry plot, =0 for no symmetry plot)
NMAT	Number of materials (1 or 2)
NPX	Number of particles in x-direction in rectangular particle setup
NPY	Number of particles in y-direction in rectangular particle setup
NU	Coefficient of kinematic viscosity
OMG	Over-relaxation factor used in pressure iteration
PLTDT	Time increment between plots and/or prints to be output on film
PRTDT	Time increment between prints on paper
RHOF	Fluid density (for F = 1.0 region)
RHOFC	Fluid density in complement of F region
SIGMA	Surface tension coefficient
TWFIN	Problem time to end calculation
UI	X-direction velocity used for initializing mesh
VI	Y-direction velocity used for initializing mesh
VELMX	Maximum velocity in problem used to scale velocity vector plots

WB Indicator for boundary condition to be used along the bottom of the mesh (=1 for rigid free-slip wall, =2 for rigid no-slip wall, =3 for continuative boundary, =4 for periodic boundary, =5 for constant pressure boundary)

WL Indicator for boundary condition along left side of mesh (see WB)

WR Indicator for boundary condition along right side of mesh (see WB)

WT Indicator for boundary condition along top of mesh (see WB)

XPL Location of left side of rectangular particle region

XPR Location of right side of rectangular particle region

YPB Location of bottom of rectangular particle region

YPT Location of top of rectangular particle region

C. Mesh Generation Input

The SOLA-VOF code utilizes a mesh composed of rectangular cells with variable sizes. Although the coordinates of each mesh line could be read in for each new problem, this is inconvenient in most cases. To circumvent this, SOLA-VOF has an automatic mesh generator that is general enough to satisfy almost all situations likely to be encountered. The basic idea is to build up a mesh by stacking together a series of submeshes in each coordinate direction. For example, to set up mesh intervals in the x-direction, the x dimension of the problem is subdivided into a set of NKX intervals. The k^{th} interval extends from its left end, XL(k), to the left end of the next interval, XL(k + 1). Within each interval there is a location XC(k) where the mesh cells will be smallest. The number of mesh cells between XL(k) and XC(k) is specified as NXL(k), and the number from XC(k) to XL(k + 1) is specified as NXR(k). The minimum cell size in the interval, which is located at XC(k), is specified as DXMN(k). Using this information, the mesh generator expands the cells starting with DXMN(k) at XC(k) and increases their size quadratically toward both ends of the interval. If DXMN(k) is larger than the cell size corresponding to uniform zoning then uniform zoning is produced by the generator. Thus, whenever uniform zoning is wanted in an interval, it is only necessary to input a large value for DXMN(k), say the width of the interval. The number of cells to the left and right of the convergence point need not be equal, but there must be at least one cell on both sides.

Any number of submeshes may be stacked together, provided sufficient storage has been allowed for the necessary input data defining these intervals. In this way it is possible to generate complicated meshes with locally fine resolu-

tion around any number of points. Furthermore, there should be no unexpected numerical stability problems because minimum cell sizes are specified as part of the input data.

A similar treatment is used in the y-direction in which YL(k) denotes the low end of the k^{th} interval and YC(k) the small cell convergence point in the interval. Cell numbers below and above this point for the k^{th} interval are NYL(k) and NYR(k), while DYMN(k) denotes the minimum cell size in the interval.

When periodic conditions are specified in the x-direction, there must be two cells reserved at the right end of the mesh for the setting of boundary conditions. It is also necessary that these cells have the same sizes as the first two interior cells on the left side of the mesh in order to insure periodicity. The mesh generator does this automatically, so the user only needs to input the dimensions of the periodic interval he wishes to use. However, in setting storage requirements, it must be remembered that periodic meshes need three rather than two additional cells for boundary conditions. Periodic conditions in the y-direction are treated in the same way with two cells needed at the top of the mesh to set boundary conditions, and this fact must be remembered in setting the PARAMETER statement at the beginning of the code.

An example of the proper format to be used to specify a mesh spanning the x region $LW \leqslant X \leqslant RW$ with n submeshes is

NKX=n, XL=LW, XL_2, ..., XL_n, RW,

XC=XC_1, XC_2, ..., XC_n,

NXL=NL_1, NL_2, ..., NL_n

NXR=NR_1, NR_2, ..., NR_n,

DXMN=$DXMN_1$, $DXMN_2$ ..., $DXMN_n$,

where NL_i represents the number of cells to the left of XC_i and NR_i the number to the right of XC_i in the i^{th} submesh.

The following is a list of all input variables in the second NAMELIST block used for mesh generation. This list is reproduced at the beginning of the program listing.

1. NAMELIST/MSHSET/.

DXMN(N)	Minimum space increment in x-direction in submesh N
DYMN(N)	Minimum space increment in y-direction in submesh N
NKX	Number of submesh regions in x-direction

NKY Number of submesh regions in y-direction

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 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 YC(N) and YL(N + 1) in submesh N

XC(N) X-coordinate of the convergence point in submesh N

XL(N) Location of the left edge of submesh N (NKX+1 values of XL(N) are necessary because the right edge of submesh N is determined by the left edge of submesh N + 1)

YC(N) Y-coordinate of the convergence point in submesh N

YL(N) Location of the bottom of submesh N (NKY + 1 values of YL(N) are necessary because the top edge of submesh N is determined by the bottom edge of submesh N + 1)

D. Variables and Arrays Listed in COMMON

The variables and arrays contained in COMMON are given alphabetically in the following list with a brief description of each one. Variables already defined in the NAMELIST inputs are omitted from this list.

1. COMMON VARIABLES.

CYCLE Calculational time cycle

CYL Mesh geometry indicator (=ICYL)

DTSFT Maximum DELT value allowed by the surface tension force stability criterion (DELT is automatically adjusted)

DTVIS Maximum DELT value allowed by the viscous force stability criterion (DELT is automatically adjusted)

EMF Small value, typically 1.0E - 06, used to negate round-off error effects when testing F = 1.0 or F = 0.0

EMF1 =1.0-EMF

EM6 =1.0E-06

EM10 =1.0E-10

EP10 =1.0E+10

FLG Pressure iteration convergence test indicator (=0.0 when the convergence test is satisfied, =1.0 when the convergence test is not satisfied)

FLGC Volume of fluid function advection limit indicator (DELT reduced and cycle started over if limit is exceeded)

FNOC	Pressure convergence failure indicator (=1.0, convergence failed and DELT is reduced, =0.0 otherwise)
IBAR	Number of real cells in x-direction (excludes fictitious cells)
IBAR2	=IBAR + 2 or IBAR + 3, specified in parameter statement
IMAX	Total number of mesh cells in x-direction (=IBAR + 2 or IBAR + 3)
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 the last real cell in the x-direction (=IMAX-2)
IPL	Leftmost pressure iteration index in x-direction (=3 for constant pressure boundary condition, =2 for all other cases)
IPR	Rightmost pressure iteration index in x-direction (=IM2 for constant pressure boundary condition, =IM1 for all other cases)
ITER	Pressure iteration counter
JBAR	Number of real cells in y-direction (excludes fictitious cells)
JBAR2	=JBAR + 2 or JBAR + 3, specified in parameter statement
JMAX	Total number at mesh cells in y-direction (=JBAR + 2 or JBAR + 3)
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 the last real cell in the y-direction (=JMAX - 2)
JPB	Bottom pressure iteration index in y-direction (=3 for constant pressure boundary condition, =2 for all other cases)
JPT	Top pressure iteration index in y-direction (=JM2 for constant pressure boundary condition, =JM1 for all other cases)
NFLGC	Number of cycles the volume of fluid function advection limit (FLGC) is exceeded
NOCON	Number of cycles pressure convergence has failed
NP	Total number of particles computed to be in mesh
NPRTS	Number of particles in mesh, specified in parameter statement (used to set array size, must be >0)
NREG	Number of void regions generated in calculation
NVOR	Maximum number of void regions allowed, specified in parameter statement
NVRM	Maximum number of void regions ($\leqslant$ NVOR)

MESHX	Number of submesh regions in x-direction, specified in parameter statement
MESHY	Number of submesh regions in y-direction, specified in parameter statement
PI	=3.141592654
RCSQ	Reciprocal of CSQ
RHOD	Difference in fluid densities (=RHOF - RHOFC)
RPD	Degrees to radians conversion factor
SF	Plot scaling factor
T	Problem time
TANGLE	Tangent of contact angle, CANGLE
VCHGT	Accumulated fluid volume change
VELMX1	VELMX normalized to minimum mesh cell dimension
XMAX	Location of right-hand side of mesh
XMIN	Location of left-hand side of mesh
XSHFT	Computed shift along the plotting abscissa to center the plot frame on film
YMAX	Location of the top of the mesh
YMIN	Location of the bottom of the mesh
YSHFT	Computed shift along the plotting ordinate to center the flot frame on film

2. COMMON ARRAYS.

ACOM(1)	First word in common
BETA(I,J)	Pressure iteration relaxation factor in cell (I,J)
DELX(I)	Mesh spacing of the Ith cell along the x-axis
DELY(J)	Mesh spacing of the Jth cell along the y-axis
F(I,J)	Volume of fluid per unit volume of cell (I,J) at time level N + 1
FN(I,J)	Volume of fluid per unit volume of cell (I,J) at time level N
IP(K)	Cell index for particle K along x-axis
JP(K)	Cell index for particle K along y-axis
NAME(10)	Problem identification line
NF(I,J)	Flag of surface cell (I,J) indicating the location of its neighboring pressure interpolation cell
NR(K)	Label of void region, K > 5
P(I,J)	Pressure in cell (I,J) at time level N + 1
PETA(I,J)	Pressure interpolation factor for cell (I,J)

PN(I,J)	Pressure in cell (I,J) at time level N
PR(K)	Pressure in void region NR(K)
PS(I,J)	Surface pressure in cell (I,J) computed from surface tension force
RDX(I)	Reciprocal of DELX(I)
RDY(J)	Reciprocal of DELY(J)
RX(I)	Reciprocal of X(I)
RXI(I)	Reciprocal of XI(I)
RYJ(J)	Reciprocal of YJ(J)
TANTH(I,J)	Slope of fluid surface in cell (I,J)
U(I,J)	X-direction velocity component in cell (I,J) at time level N + 1
UN(I,J)	X-direction velocity component in cell (I,J) at time level N
V(I,J)	Y-direction velocity component in cell (I,J) at time level N + 1
VN(I,J)	Y-direction velocity component in cell (I,J) at time level N
VOL(K)	Volume of void region NR(K)
X(I)	Location of the right-hand boundary of the Ith cell along the x-axis
XI(I)	Location of the center of the Ith cell along the x-axis
XP(K)	X-coordinate of particle K
Y(J)	Location of the top boundary of the Jth cell along the y-axis
YJ(J)	Location of the center of the Jth cell along the y-axis
YP(K)	Y-coordinate of particle K
ZCOM(1)	Last word in COMMON

IV. PROBLEM RUNNING

A. General Procedure

To run a problem with SOLA-VOF several things must be done. First, the user must specify the input data defining the mesh to be used. This data is entered in NAMELIST/MSHSET/. All other input data must be specified in NAMELIST/XPUT/. Except for the time step (DELT), time when to finish (TWFIN), and print and plot time intervals (PRTDT, PLTDT), all input variables in NAMELIST/XPUT/ have default values, which are listed at the beginning of the program. If the default values are acceptable they need not be redefined in the input.

Once the input variables have been selected, it is essential to check the PARAMETER statement at the beginning of the code to see if proper storage requirements have been requested for the problem. The PARAMETER statement has the form

PARAMETER(IBAR2= , JBAR2= , NPRTS= , MESHX= , MESHY= , NVOR=) .

The variables named in this statement are used to set the dimensions of the COMMON blocks at compilation time. These variables cannot be used elsewhere in the code. The proper values to be specified for these variables are easily determined from the input data,

$$\text{IBAR2} = 2 + \sum_{k}^{NKX} (\text{NXL}(k)+\text{NXR}(k)) + (1, \text{ if periodic in x-direction}) \quad ,$$

$$\text{JBAR2} = 2 + \sum_{k}^{NKY} (\text{NYL}(k)+\text{NYR}(k)) + (1, \text{ if periodic in y-direction}) \quad ,$$

$$\text{NPRTS} = (\text{NPX}+1)(\text{NPY}) + 1 \quad ,$$

$$\text{MESHX} = \text{NKX} \quad ,$$

$$\text{MESHY} = \text{NKY} \quad ,$$

and

NVOR = Maximum number of void regions expected.

Actual problem dimensions can be smaller than the values used in the PARAMETER statement, but never larger.

When special boundary conditions or internal obstacles are wanted that cannot be set by the proper selection of input data, the user must add these to the code in the locations reserved for this purpose. For special boundary conditions, the proper location is marked by a comment at the end of the BC subroutine. To set internal obstacle flags (BETA(i,j) = -1.0), a location in the routine SETUP is marked with a comment.

The last step in setting up a problem is to insure the proper initialization of all dependent variables. Input data is available for defining an initial uniform velocity (UI,VI) and a horizontal interface of height FLHT. For other fluid configurations it is best to input FLHT = 0.0 and then program in the desired distribution in the location indicated in the SETUP routine.

Thus, the general setup procedure for SOLA-VOF is relatively easy and straightforward. To summarize, the necessary steps are:

(1) Set mesh generator input data.

(2) Set remaining input data if not correctly defaulted.

(3) Check PARAMETER statement for memory size.

(4) Set special boundary conditions and obstacles, if needed.

(5) Set special initial values for dependent variables if not covered by input data.

B. Sample Test Problem

To help the new user get started, the SOLA-VOF program has been set up to solve a simple problem. Results from our calculations for this problem are provided in this section.

When a horizontal layer of water is pushed into a rigid, vertical wall there will be a step wave, or bore, produced that runs away from the wall. If the incident velocity is not too great, the bore front will have an undular shape, but at sufficiently high velocities the bore front will break and be highly irregular. In either case, conservation of mass and momentum principles may be used to derive "jump" conditions that should exist across the bore transition.[16]

SOLA-VOF was used to compute the undular bore evolution shown in Fig. 8. The initial configuration in Fig. 8a consists of a uniform mesh of 20 cells in the horizontal direction ($\delta x = 0.6$) and 8 cells in the vertical direction ($\delta y = 0.2$). Fluid initially fills the lowest five rows (depth 1.0) and is uniformly moving to the right with unit velocity. The right, bottom, and top walls are rigid free-slip boundaries. At the left boundary, a continuative boundary condition is used, which allows a continuous input of fluid that prevents any waves from being generated there. Gravity acts downward with unit magnitude.

Although this problem is very coarsely resolved, the results are remarkably good and provide a nice check on mass and momentum conservation. The computed jump height at the right wall is 1.210, whereas theory predicts 1.209.

The coarse mesh calculation took 14 s of CDC-7600 computer time to complete 48 cycles of calculation. No special boundary conditions or other modifications are needed for this problem. Thus, it can be set up completely with the following input data lists. Input data used to set up the mesh are:

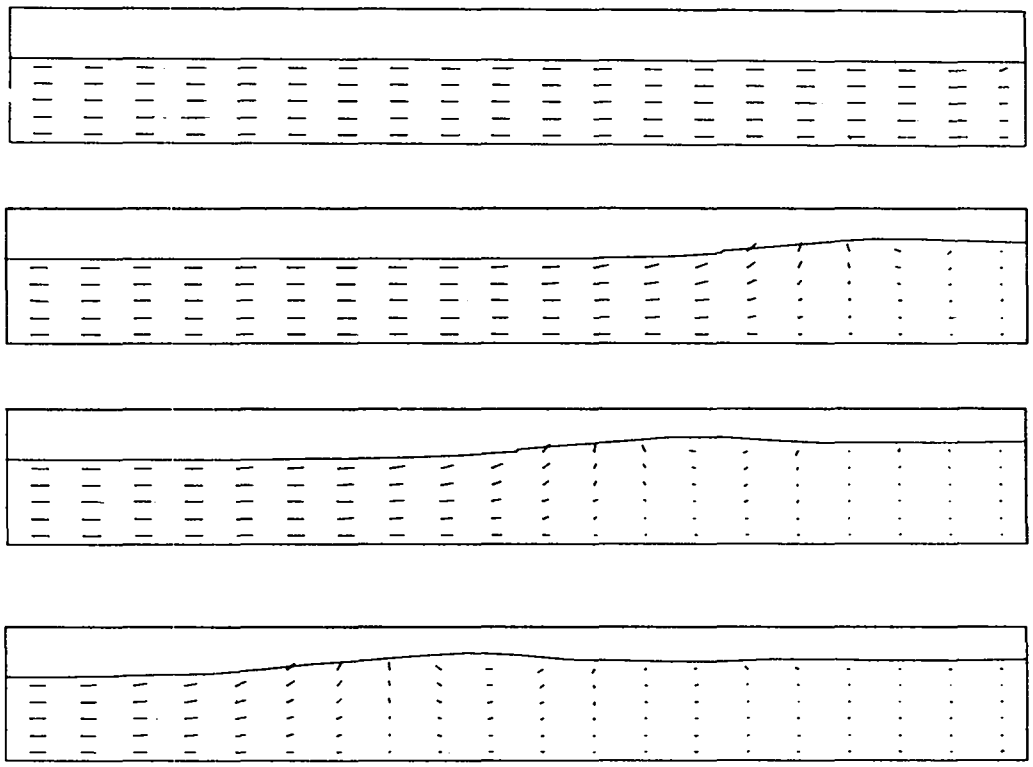

Fig. 8.

Velocities and free-surface configuration for undular bore calculation. Times are 0.0, 4.0, 7.0, and 10.0.

NKX=1, XL=0.0, 12.0, XC=6.0, NXL=10, NXR=10, DXMN=0.6

NKY=1, YL=0.0, 1.6, YC=0.8, NYL=4, NYR=4, DYMN=0.2.

The remaining input data are:

DELT=0.2, FLHT=1.0, GY=-1.0, PLTDT=1.0, PRTDT=5.0,

TWFIN=12.0, UI=0.2, VELMX=0.2, WL=3, AUTOT=0.0.

Results from cycle zero (the setup), from cycle 1, and from t = 10.0 (cycle 50) are contained in the following long print listings that are output on paper.

SOLA - VOF UNDULAR BORE COMPARISON

ITER= 0 TIME= 0. DELT= 2.00000E-01 CYCLE= 0 VCHGT= 2.64000E-06

NREG= 1

K	VOL(K)	PR(K)
6	7.20000E+00	0.

I	J	U	V	P	D	PS	F	NF	PETA
1	1	2.00000E-01	0.	9.00000E-01	3.33333E-01	0.	1.00000E+00	0	1.00000E+00
1	2	2.00000E-01	0.	9.00000E-01	3.33333E-01	0.	1.00000E+00	0	1.00000E+00
1	3	2.00000E-01	0.	7.00000E-01	3.33333E-01	0.	1.00000E+00	0	1.00000E+00
1	4	2.00000E-01	0.	5.00000E-01	3.33333E-01	0.	1.00000E+00	0	1.00000E+00
1	5	2.00000E-01	0.	3.00000E-01	3.33333E-01	0.	1.00000E+00	0	1.00000E+00
1	6	2.00000E-01	0.	1.00000E-01	3.33333E-01	0.	9.99999E-01	0	1.00000E+00
1	7	2.00000E-01	0.	0.	3.33333E-01	0.	0.	0	1.00000E+00
1	8	0.	0.	0.	0.	0.	0.	0	1.00000E+00
1	9	0.	0.	0.	0.	0.	0.	0	1.00000E+00
1	10	0.	0.	0.	0.	0.	0.	0	1.00000E+00
2	1	2.00000E-01	0.	9.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
2	2	2.00000E-01	0.	9.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
2	3	2.00000E-01	0.	7.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
2	4	2.00000E-01	0.	5.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
2	5	2.00000E-01	0.	3.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
2	6	2.00000E-01	0.	1.00000E-01	0.	0.	9.99999E-01	3	6.66667E-01
2	7	2.00000E-01	0.	0.	0.	0.	0.	6	1.00000E+00
2	8	0.	0.	0.	0.	0.	0.	6	1.00000E+00
2	9	0.	0.	0.	0.	0.	0.	6	1.00000E+00
2	10	0.	0.	0.	0.	0.	0.	0	1.00000E+00
3	1	2.00000E-01	0.	9.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
3	2	2.00000E-01	0.	9.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
3	3	2.00000E-01	0.	7.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
3	4	2.00000E-01	0.	5.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
3	5	2.00000E-01	0.	3.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
3	6	2.00000E-01	0.	1.00000E-01	0.	0.	9.99999E-01	3	6.66667E-01
3	7	2.00000E-01	0.	0.	0.	0.	0.	6	1.00000E+00
3	8	0.	0.	0.	0.	0.	0.	6	1.00000E+00
3	9	0.	0.	0.	0.	0.	0.	6	1.00000E+00
3	10	0.	0.	0.	0.	0.	0.	0	1.00000E+00
4	1	2.00000E-01	0.	9.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
4	2	2.00000E-01	0.	9.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
4	3	2.00000E-01	0.	7.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
4	4	2.00000E-01	0.	5.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
4	5	2.00000E-01	0.	3.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
4	6	2.00000E-01	0.	1.00000E-01	0.	0.	9.99999E-01	3	6.66667E-01
4	7	2.00000E-01	0.	0.	0.	0.	0.	6	1.00000E+00
4	8	0.	0.	0.	0.	0.	0.	6	1.00000E+00
4	9	0.	0.	0.	0.	0.	0.	6	1.00000E+00
4	10	0.	0.	0.	0.	0.	0.	0	1.00000E+00
5	1	2.00000E-01	0.	9.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00

5	2	2.00000E-01	0.	9.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
5	3	2.00000E-01	0.	7.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
5	4	2.00000E-01	0.	5.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
5	5	2.00000E-01	0.	3.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
5	6	2.00000E-01	0.	1.00000E-01	0.	0.	9.99999E-01	3	6.66667E-01
5	7	2.00000E-01	0.	0.	0.	0.	0.	6	1.00000E+00
5	8	0.	0.	0.	0.	0.	0.	6	1.00000E+00
5	9	0.	0.	0.	0.	0.	0.	6	1.00000E+00
5	10	0.	0.	0.	0.	0.	0.	0	1.00000E+00
6	1	2.00000E-01	0.	9.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
6	2	2.00000E-01	0.	9.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
6	3	2.00000E-01	0.	7.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
6	4	2.00000E-01	0.	5.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
6	5	2.00000E-01	0.	3.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
6	6	2.00000E-01	0.	1.00000E-01	0.	0.	9.99999E-01	3	6.66667E-01
6	7	2.00000E-01	0.	0.	0.	0.	0.	6	1.00000E+00
6	8	0.	0.	0.	0.	0.	0.	6	1.00000E+00
6	9	0.	0.	0.	0.	0.	0.	6	1.00000E+00
6	10	0.	0.	0.	0.	0.	0.	0	1.00000E+00
7	1	2.00000E-01	0.	9.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
7	2	2.00000E-01	0.	9.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
7	3	2.00000E-01	0.	7.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
7	4	2.00000E-01	0.	5.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
7	5	2.00000E-01	0.	3.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
7	6	2.00000E-01	0.	1.00000E-01	0.	0.	9.99999E-01	3	6.66667E-01
7	7	2.00000E-01	0.	0.	0.	0.	0.	6	1.00000E+00
7	8	0.	0.	0.	0.	0.	0.	6	1.00000E+00
7	9	0.	0.	0.	0.	0.	0.	6	1.00000E+00
7	10	0.	0.	0.	0.	0.	0.	0	1.00000E+00
8	1	2.00000E-01	0.	9.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
8	2	2.00000E-01	0.	9.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
8	3	2.00000E-01	0.	7.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
8	4	2.00000E-01	0.	5.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
8	5	2.00000E-01	0.	3.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
8	6	2.00000E-01	0.	1.00000E-01	0.	0.	9.99999E-01	3	6.66667E-01
8	7	2.00000E-01	0.	0.	0.	0.	0.	6	1.00000E+00
8	8	0.	0.	0.	0.	0.	0.	6	1.00000E+00
8	9	0.	0.	0.	0.	0.	0.	6	1.00000E+00
8	10	0.	0.	0.	0.	0.	0.	0	1.00000E+00
9	1	2.00000E-01	0.	9.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
9	2	2.00000E-01	0.	9.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
9	3	2.00000E-01	0.	7.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
9	4	2.00000E-01	0.	5.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
9	5	2.00000E-01	0.	3.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
9	6	2.00000E-01	0.	1.00000E-01	0.	0.	9.99999E-01	3	6.66667E-01
9	7	2.00000E-01	0.	0.	0.	0.	0.	6	1.00000E+00
9	8	0.	0.	0.	0.	0.	0.	6	1.00000E+00
9	9	0.	0.	0.	0.	0.	0.	6	1.00000E+00
9	10	0.	0.	0.	0.	0.	0.	0	1.00000E+00
10	1	2.00000E-01	0.	9.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
10	2	2.00000E-01	0.	9.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
10	3	2.00000E-01	0.	7.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
10	4	2.00000E-01	0.	5.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
10	5	2.00000E-01	0.	3.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
10	6	2.00000E-01	0.	1.00000E-01	0.	0.	9.99999E-01	3	6.66667E-01
10	7	2.00000E-01	0.	0.	0.	0.	0.	6	1.00000E+00
10	8	0.	0.	0.	0.	0.	0.	6	1.00000E+00
10	9	0.	0.	0.	0.	0.	0.	6	1.00000E+00
10	10	0.	0.	0.	0.	0.	0.	0	1.00000E+00
11	1	2.00000E-01	0.	9.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00

11	2	2.00000E-01	0.	9.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
11	3	2.00000E-01	0.	7.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
11	4	2.00000E-01	0.	5.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
11	5	2.00000E-01	0.	3.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
11	6	2.00000E-01	0.	1.00000E-01	0.	0.	9.99999E-01	3	6.66667E-01
11	7	2.00000E-01	0.	0.	0.	0.	0.	6	1.00000E+00
11	8	0.	0.	0.	0.	0.	0.	6	1.00000E+00
11	9	0.	0.	0.	0.	0.	0.	6	1.00000E+00
11	10	0.	0.	0.	0.	0.	0.	0	1.00000E+00
12	1	2.00000E-01	0.	9.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
12	2	2.00000E-01	0.	9.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
12	3	2.00000E-01	0.	7.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
12	4	2.00000E-01	0.	5.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
12	5	2.00000E-01	0.	3.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
12	6	2.00000E-01	0.	1.00000E-01	0.	0.	9.99999E-01	3	6.66667E-01
12	7	2.00000E-01	0.	0.	0.	0.	0.	6	1.00000E+00
12	8	0.	0.	0.	0.	0.	0.	6	1.00000E+00
12	9	0.	0.	0.	0.	0.	0.	6	1.00000E+00
12	10	0.	0.	0.	0.	0.	0.	0	1.00000E+00
13	1	2.00000E-01	0.	9.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
13	2	2.00000E-01	0.	9.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
13	3	2.00000E-01	0.	7.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
13	4	2.00000E-01	0.	5.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
13	5	2.00000E-01	0.	3.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
13	6	2.00000E-01	0.	1.00000E-01	0.	0.	9.99999E-01	3	6.66667E-01
13	7	2.00000E-01	0.	0.	0.	0.	0.	6	1.00000E+00
13	8	0.	0.	0.	0.	0.	0.	6	1.00000E+00
13	9	0.	0.	0.	0.	0.	0.	6	1.00000E+00
13	10	0.	0.	0.	0.	0.	0.	0	1.00000E+00
14	1	2.00000E-01	0.	9.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
14	2	2.00000E-01	0.	9.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
14	3	2.00000E-01	0.	7.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
14	4	2.00000E-01	0.	5.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
14	5	2.00000E-01	0.	3.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
14	6	2.00000E-01	0.	1.00000E-01	0.	0.	9.99999E-01	3	6.66667E-01
14	7	2.00000E-01	0.	0.	0.	0.	0.	6	1.00000E+00
14	8	0.	0.	0.	0.	0.	0.	6	1.00000E+00
14	9	0.	0.	0.	0.	0.	0.	6	1.00000E+00
14	10	0.	0.	0.	0.	0.	0.	0	1.00000E+00
15	1	2.00000E-01	0.	9.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
15	2	2.00000E-01	0.	9.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
15	3	2.00000E-01	0.	7.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
15	4	2.00000E-01	0.	5.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
15	5	2.00000E-01	0.	3.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
15	6	2.00000E-01	0.	1.00000E-01	0.	0.	9.99999E-01	3	6.66667E-01
15	7	2.00000E-01	0.	0.	0.	0.	0.	6	1.00000E+00
15	8	0.	0.	0.	0.	0.	0.	6	1.00000E+00
15	9	0.	0.	0.	0.	0.	0.	6	1.00000E+00
15	10	0.	0.	0.	0.	0.	0.	0	1.00000E+00
16	1	2.00000E-01	0.	9.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
16	2	2.00000E-01	0.	9.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
16	3	2.00000E-01	0.	7.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
16	4	2.00000E-01	0.	5.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
16	5	2.00000E-01	0.	3.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
16	6	2.00000E-01	0.	1.00000E-01	0.	0.	9.99999E-01	3	6.66667E-01
16	7	2.00000E-01	0.	0.	0.	0.	0.	6	1.00000E+00
16	8	0.	0.	0.	0.	0.	0.	6	1.00000E+00
16	9	0.	0.	0.	0.	0.	0.	6	1.00000E+00
16	10	0.	0.	0.	0.	0.	0.	0	1.00000E+00
17	1	2.00000E-01	0.	9.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00

17	2	2.00000E-01	0.	9.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
17	3	2.00000E-01	0.	7.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
17	4	2.00000E-01	0.	5.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
17	5	2.00000E-01	0.	3.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
17	6	2.00000E-01	0.	1.00000E-01	0.	0.	9.99999E-01	3	6.66667E-01
17	7	2.00000E-01	0.	0.	0.	0.	0.	6	1.00000E+00
17	8	0.	0.	0.	0.	0.	0.	6	1.00000E+00
17	9	0.	0.	0.	0.	0.	0.	6	1.00000E+00
17	10	0.	0.	0.	0.	0.	0.	0	1.00000E+00
18	1	2.00000E-01	0.	9.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
18	2	2.00000E-01	0.	9.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
18	3	2.00000E-01	0.	7.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
18	4	2.00000E-01	0.	5.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
18	5	2.00000E-01	0.	3.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
18	6	2.00000E-01	0.	1.00000E-01	0.	0.	9.99999E-01	3	6.66667E-01
18	7	2.00000E-01	0.	0.	0.	0.	0.	6	1.00000E+00
18	8	0.	0.	0.	0.	0.	0.	6	1.00000E+00
18	9	0.	0.	0.	0.	0.	0.	6	1.00000E+00
18	10	0.	0.	0.	0.	0.	0.	0	1.00000E+00
19	1	2.00000E-01	0.	9.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
19	2	2.00000E-01	0.	9.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
19	3	2.00000E-01	0.	7.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
19	4	2.00000E-01	0.	5.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
19	5	2.00000E-01	0.	3.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
19	6	2.00000E-01	0.	1.00000E-01	0.	0.	9.99999E-01	3	6.66667E-01
19	7	2.00000E-01	0.	0.	0.	0.	0.	6	1.00000E+00
19	8	0.	0.	0.	0.	0.	0.	6	1.00000E+00
19	9	0.	0.	0.	0.	0.	0.	6	1.00000E+00
19	10	0.	0.	0.	0.	0.	0.	0	1.00000E+00
20	1	2.00000E-01	0.	9.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
20	2	2.00000E-01	0.	9.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
20	3	2.00000E-01	0.	7.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
20	4	2.00000E-01	0.	5.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
20	5	2.00000E-01	0.	3.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
20	6	2.00000E-01	0.	1.00000E-01	0.	0.	9.99999E-01	3	6.66667E-01
20	7	2.00000E-01	0.	0.	0.	0.	0.	6	1.00000E+00
20	8	0.	0.	0.	0.	0.	0.	6	1.00000E+00
20	9	0.	0.	0.	0.	0.	0.	6	1.00000E+00
20	10	0.	0.	0.	0.	0.	0.	0	1.00000E+00
21	1	0.	0.	9.00000E-01	-3.33333E-01	0.	1.00000E+00	0	1.00000E+00
21	2	0.	0.	9.00000E-01	-3.33333E-01	0.	1.00000E+00	0	1.00000E+00
21	3	0.	0.	7.00000E-01	-3.33333E-01	0.	1.00000E+00	0	1.00000E+00
21	4	0.	0.	5.00000E-01	-3.33333E-01	0.	1.00000E+00	0	1.00000E+00
21	5	0.	0.	3.00000E-01	-3.33333E-01	0.	1.00000E+00	0	1.00000E+00
21	6	0.	6.66667E-02	1.00000E-01	0.	0.	9.99999E-01	3	6.66667E-01
21	7	0.	0.	0.	-6.66667E-01	0.	0.	6	1.00000E+00
21	8	0.	0.	0.	0.	0.	0.	6	1.00000E+00
21	9	0.	0.	0.	0.	0.	0.	6	1.00000E+00
21	10	0.	0.	0.	0.	0.	0.	0	1.00000E+00
22	1	0.	0.	9.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
22	2	0.	0.	9.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
22	3	0.	0.	7.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
22	4	0.	0.	5.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
22	5	0.	0.	3.00000E-01	0.	0.	1.00000E+00	0	1.00000E+00
22	6	0.	0.	1.00000E-01	0.	0.	9.99999E-01	0	1.00000E+00
22	7	0.	0.	0.	0.	0.	0.	0	1.00000E+00
22	8	0.	0.	0.	0.	0.	0.	0	1.00000E+00
22	9	0.	0.	0.	0.	0.	0.	0	1.00000E+00
22	10	0.	0.	0.	0.	0.	0.	0	1.00000E+00

SOLA - VOF UNDULAR BORE COMPARISON

ITER= 25 TIME= 2.00000E-01 DELT= 2.00000E-01 CYCLE= 1 VCHGT= 8.82104E-05

NREG= 1

K	VOL(K)	PR(K)
6	7.16006E+00	0.

I	J	U	V	P	D	PS	F	NF	PETA
1	1	2.00000E-01	0.	9.00000E-01	3.33333E-01	0.	1.00000E+00	0	1.00000E+00
1	2	2.00000E-01	0.	9.00000E-01	3.33333E-01	0.	1.00000E+00	0	1.00000E+00
1	3	2.00000E-01	-2.84217E-15	7.00000E-01	3.33333E-01	0.	1.00000E+00	0	1.00000E+00
1	4	2.00000E-01	0.	5.00000E-01	3.33333E-01	0.	1.00000E+00	0	1.00000E+00
1	5	2.00000E-01	0.	3.00000E-01	3.33333E-01	0.	1.00000E+00	0	1.00000E+00
1	6	2.00000E-01	-1.00000E-01	9.99998E-02	-1.66667E-01	0.	9.99999E-01	0	1.00000E+00
1	7	2.00000E-01	0.	0.	8.33333E-01	0.	0.	0	1.00000E+00
1	8	0.	0.	0.	0.	0.	0.	0	1.00000E+00
1	9	0.	0.	0.	0.	0.	0.	0	1.00000E+00
1	10	0.	0.	0.	0.	0.	0.	0	1.00000E+00
2	1	2.00000E-01	0.	9.00000E-01	-3.27191E-08	0.	1.00000E+00	0	1.00000E+00
2	2	2.00000E-01	6.61168E-09	9.00000E-01	3.39294E-10	0.	1.00000E+00	0	1.00000E+00
2	3	2.00000E-01	1.41900E-08	7.00000E-01	-1.80911E-11	0.	1.00000E+00	0	1.00000E+00
2	4	2.00000E-01	2.40787E-08	5.00000E-01	-2.86342E-11	0.	1.00000E+00	0	1.00000E+00
2	5	2.00000E-01	3.80960E-08	3.00000E-01	-3.49718E-11	0.	1.00000E+00	0	1.00000E+00
2	6	2.00000E-01	5.90671E-08	9.99998E-02	0.	0.	9.99999E-01	3	6.66667E-01
2	7	2.00000E-01	0.	0.	-4.00191E-07	0.	0.	6	1.00000E+00
2	8	0.	0.	0.	0.	0.	0.	6	1.00000E+00
2	9	0.	0.	0.	0.	0.	0.	6	1.00000E+00
2	10	0.	0.	0.	0.	0.	0.	0	1.00000E+00
3	1	2.00000E-01	0.	9.00000E-01	-1.88524E-08	0.	1.00000E+00	0	1.00000E+00
3	2	2.00000E-01	3.88049E-09	9.00000E-01	5.50111E-10	0.	1.00000E+00	0	1.00000E+00
3	3	2.00000E-01	7.56714E-09	7.00000E-01	-6.68083E-11	0.	1.00000E+00	0	1.00000E+00
3	4	2.00000E-01	1.09890E-08	5.00000E-01	-8.29254E-11	0.	1.00000E+00	0	1.00000E+00
3	5	2.00000E-01	1.36710E-08	3.00000E-01	-8.06486E-11	0.	1.00000E+00	0	1.00000E+00
3	6	2.00000E-01	1.45704E-08	9.99998E-02	-2.64698E-23	0.	9.99999E-01	3	6.66667E-01
3	7	2.00000E-01	0.	0.	-7.73487E-08	0.	0.	6	1.00000E+00
3	8	0.	0.	0.	0.	0.	0.	6	1.00000E+00
3	9	0.	0.	0.	0.	0.	0.	6	1.00000E+00
3	10	0.	0.	0.	0.	0.	0.	0	1.00000E+00
4	1	2.00000E-01	0.	9.00000E-01	-8.40995E-09	0.	1.00000E+00	0	1.00000E+00
4	2	2.00000E-01	1.78349E-09	9.00000E-01	5.07508E-10	0.	1.00000E+00	0	1.00000E+00
4	3	2.00000E-01	3.29818E-09	7.00000E-01	-3.30926E-10	0.	1.00000E+00	0	1.00000E+00
4	4	2.00000E-01	4.56248E-09	5.00000E-01	-3.62697E-10	0.	1.00000E+00	0	1.00000E+00
4	5	2.00000E-01	5.43469E-09	3.00000E-01	-2.17964E-10	0.	1.00000E+00	0	1.00000E+00
4	6	2.00000E-01	5.73995E-09	9.99998E-02	1.98523E-23	0.	9.99999E-01	3	6.66667E-01
4	7	2.00000E-01	0.	0.	-3.02261E-08	0.	0.	6	1.00000E+00
4	8	0.	0.	0.	0.	0.	0.	6	1.00000E+00
4	9	0.	0.	0.	0.	0.	0.	6	1.00000E+00
4	10	0.	0.	0.	0.	0.	0.	0	1.00000E+00
5	1	2.00000E-01	0.	9.00000E-01	-5.08350E-09	0.	1.00000E+00	0	1.00000E+00
5	2	2.00000E-01	5.96567E-10	9.00000E-01	-2.10066E-09	0.	1.00000E+00	0	1.00000E+00
5	3	2.00000E-01	1.22545E-09	7.00000E-01	-2.71290E-09	0.	1.00000E+00	0	1.00000E+00
5	4	2.00000E-01	1.92538E-09	5.00000E-01	-2.37965E-09	0.	1.00000E+00	0	1.00000E+00
5	5	2.00000E-01	2.69316E-09	3.00000E-01	-7.29696E-10	0.	1.00000E+00	0	1.00000E+00
5	6	2.00000E-01	2.99773E-09	9.99998E-02	1.32349E-23	0.	9.99999E-01	3	6.66667E-01
5	7	2.00000E-01	0.	0.	-1.65115E-08	0.	0.	6	1.00000E+00
5	8	0.	0.	0.	0.	0.	0.	6	1.00000E+00
5	9	0.	0.	0.	0.	0.	0.	6	1.00000E+00
5	10	0.	0.	0.	0.	0.	0.	0	1.00000E+00
6	1	2.00000E-01	0.	9.00000E-01	-1.70403E-08	0.	1.00000E+00	0	1.00000E+00

6	2	2.00000E-01	-1.98327E-09	9.00000E-01	-2.69567E-08	0.	1.00000E+00	0	1.00000E+00
6	3	2.00000E-01	-8.86791E-10	7.00000E-01	-1.77799E-08	0.	1.00000E+00	0	1.00000E+00
6	4	2.00000E-01	1.64757E-09	5.00000E-01	-1.22950E-08	0.	1.00000E+00	0	1.00000E+00
6	5	2.00000E-01	5.43392E-09	3.00000E-01	-7.93188E-10	0.	1.00000E+00	0	1.00000E+00
6	6	2.00000E-01	6.74891E-09	9.99998E-02	5.29396E-23	0.	9.99999E-01	3	6.66667E-01
6	7	2.00000E-01	0.	0.	-4.03195E-08	0.	0.	6	1.00000E+00
6	8	0.	0.	0.	0.	0.	0.	6	1.00000E+00
6	9	0.	0.	0.	0.	0.	0.	6	1.00000E+00
6	10	0.	0.	0.	0.	0.	0.	0	1.00000E+00
7	1	2.00000E-01	0.	9.00000E-01	-1.05297E-07	0.	1.00000E+00	0	1.00000E+00
7	2	2.00000E-01	-1.57628E-08	9.00000E-01	-1.84111E-07	0.	1.00000E+00	0	1.00000E+00
7	3	2.00000E-01	-6.06702E-09	7.00000E-01	-8.07642E-08	0.	1.00000E+00	0	1.00000E+00
7	4	2.00000E-01	1.08051E-08	5.00000E-01	-4.29506E-08	0.	1.00000E+00	0	1.00000E+00
7	5	2.00000E-01	3.18446E-08	3.00000E-01	1.08999E-08	0.	1.00000E+00	0	1.00000E+00
7	6	2.00000E-01	3.81311E-08	9.99998E-02	0.	0.	9.99999E-01	3	6.66667E-01
7	7	2.00000E-01	0.	0.	-2.22088E-07	0.	0.	6	1.00000E+00
7	8	0.	0.	0.	0.	0.	0.	6	1.00000E+00
7	9	0.	0.	0.	0.	0.	0.	6	1.00000E+00
7	10	0.	0.	0.	0.	0.	0.	0	1.00000E+00
8	1	2.00000E-01	0.	9.00000E-01	-5.27736E-07	0.	1.00000E+00	0	1.00000E+00
8	2	2.00000E-01	-7.26445E-08	9.00000E-01	-8.90959E-07	0.	1.00000E+00	0	1.00000E+00
8	3	2.00000E-01	-7.76979E-09	7.00000E-01	-2.59891E-07	0.	1.00000E+00	0	1.00000E+00
8	4	2.00000E-01	7.93862E-08	5.00000E-01	-9.84978E-08	0.	1.00000E+00	0	1.00000E+00
8	5	2.00000E-01	1.71413E-07	3.00000E-01	8.44728E-08	0.	1.00000E+00	0	1.00000E+00
8	6	2.00000E-01	1.96457E-07	9.99999E-02	2.54110E-21	0.	9.99999E-01	3	6.66667E-01
8	7	2.00000E-01	0.	0.	-1.10750E-06	0.	0.	6	1.00000E+00
8	8	0.	0.	0.	0.	0.	0.	6	1.00000E+00
8	9	0.	0.	0.	0.	0.	0.	6	1.00000E+00
8	10	0.	0.	0.	0.	0.	0.	0	1.00000E+00
9	1	1.99998E-01	0.	9.00001E-01	-2.12365E-06	0.	1.00000E+00	0	1.00000E+00
9	2	1.99998E-01	-2.31277E-07	9.00001E-01	-3.28004E-06	0.	1.00000E+00	0	1.00000E+00
9	3	1.99998E-01	8.05031E-08	7.00001E-01	-6.21412E-07	0.	1.00000E+00	0	1.00000E+00
9	4	1.99998E-01	4.33478E-07	5.00001E-01	-1.24803E-07	0.	1.00000E+00	0	1.00000E+00
9	5	1.99999E-01	7.61774E-07	3.00001E-01	3.61067E-07	0.	1.00000E+00	0	1.00000E+00
9	6	2.00000E-01	8.47135E-07	1.00000E-01	-5.08220E-21	0.	9.99999E-01	3	6.66667E-01
9	7	1.99999E-01	3.14768E-06	0.	9.78069E-06	0.	0.	6	1.00000E+00
9	8	1.99999E-01	0.	0.	3.33316E-01	0.	0.	6	1.00000E+00
9	9	0.	0.	0.	0.	0.	0.	6	1.00000E+00
9	10	0.	0.	0.	0.	0.	0.	0	1.00000E+00
10	1	1.99994E-01	0.	9.00006E-01	-7.23081E-06	0.	1.00000E+00	0	1.00000E+00
10	2	1.99994E-01	-4.91897E-07	9.00006E-01	-9.69030E-06	0.	1.00000E+00	0	1.00000E+00
10	3	1.99994E-01	6.91921E-07	7.00007E-01	-1.12499E-06	0.	1.00000E+00	0	1.00000E+00
10	4	1.99995E-01	1.88424E-06	5.00006E-01	7.96471E-08	0.	1.00000E+00	0	1.00000E+00
10	5	1.99997E-01	2.88863E-06	3.00004E-01	1.13622E-06	0.	1.00000E+00	0	1.00000E+00
10	6	1.99999E-01	3.14768E-06	1.00001E-01	1.35525E-20	0.	1.00000E+00	0	5.66574E-01
10	7	1.99999E-01	3.14768E-06	0.	0.	0.	2.04768E-06	3	1.99999E+00
10	8	1.99999E-01	0.	0.	-1.57384E-05	0.	0.	6	1.00000E+00
10	9	0.	0.	0.	0.	0.	0.	6	1.00000E+00
10	10	0.	0.	0.	0.	0.	0.	0	1.00000E+00
11	1	1.99981E-01	0.	9.00024E-01	-2.18103E-05	0.	1.00000E+00	0	1.00000E+00
11	2	1.99981E-01	-4.15397E-07	9.00024E-01	-2.38923E-05	0.	1.00000E+00	0	1.00000E+00
11	3	1.99982E-01	3.33514E-06	7.00025E-01	-1.45997E-06	0.	1.00000E+00	0	1.00000E+00
11	4	1.99985E-01	6.92824E-06	5.00021E-01	9.46525E-07	0.	1.00000E+00	0	1.00000E+00
11	5	1.99990E-01	9.67837E-06	3.00014E-01	2.88501E-06	0.	1.00000E+00	0	1.00000E+00
11	6	1.99997E-01	1.04027E-05	1.00005E-01	-1.35525E-20	0.	1.00000E+00	0	5.66581E-01
11	7	1.99997E-01	1.11271E-05	0.	4.06576E-20	0.	9.30274E-06	3	1.99996E+00
11	8	1.99997E-01	0.	0.	-5.92575E-05	0.	0.	6	1.00000E+00
11	9	0.	0.	0.	0.	0.	0.	6	1.00000E+00
11	10	0.	0.	0.	0.	0.	0.	0	1.00000E+00
12	1	1.99945E-01	0.	9.00081E-01	-6.05153E-05	0.	1.00000E+00	0	1.00000E+00

12	2	1.99945E-01	1.98591E-06	9.00081E-01	-5.05858E-05	0.	1.00000E+00	0	1.00000E+00
12	3	1.99948E-01	1.29359E-05	7.00079E-01	-9.64943E-07	0.	1.00000E+00	0	1.00000E+00
12	4	1.99958E-01	2.25083E-05	5.00067E-01	3.13237E-06	0.	1.00000E+00	0	1.00000E+00
12	5	1.99973E-01	2.95069E-05	3.00044E-01	6.20593E-06	0.	9.99999E-01	0	1.00000E+00
12	6	1.99991E-01	3.14260E-05	1.00015E-01	-5.42101E-20	0.	1.00000E+00	0	5.66602E-01
12	7	1.99991E-01	3.33451E-05	0.	1.08420E-19	0.	3.03260E-05	3	1.99988E+00
12	8	1.99991E-01	0.	0.	-1.76321E-04	0.	0.	6	1.00000E+00
12	9	0.	0.	0.	0.	0.	0.	6	1.00000E+00
12	10	0.	0.	0.	0.	0.	0.	0	1.00000E+00
13	1	1.99849E-01	0.	9.00248E-01	-1.59185E-04	0.	1.00000E+00	0	1.00000E+00
13	2	1.99849E-01	1.30290E-05	8.00248E-01	-9.40405E-05	0.	1.00000E+00	0	1.00000E+00
13	3	1.99861E-01	4.22497E-05	7.00235E-01	1.30892E-06	0.	1.00000E+00	0	1.00000E+00
13	4	1.99889E-01	6.67862E-05	5.00192E-01	7.38093E-06	0.	9.99999E-01	0	1.00000E+00
13	5	1.99929E-01	8.38793E-05	3.00126E-01	1.16494E-05	0.	9.99998E-01	0	1.00000E+00
13	6	1.99976E-01	8.88004E-05	1.00042E-01	-1.08420E-19	0.	1.00000E+00	0	5.66658E-01
13	7	1.99976E-01	9.37215E-05	0.	3.25261E-19	0.	8.77004E-05	3	1.99965E+00
13	8	1.99976E-01	0.	0.	-4.93213E-04	0.	0.	6	1.00000E+00
13	9	0.	0.	0.	0.	0.	0.	6	1.00000E+00
13	10	0.	0.	0.	0.	0.	0.	0	1.00000E+00
14	1	1.99605E-01	0.	9.00700E-01	-4.06012E-04	0.	1.00000E+00	0	1.00000E+00
14	2	1.99605E-01	4.99751E-05	9.00700E-01	-1.56137E-04	0.	1.00000E+00	0	1.00000E+00
14	3	1.99641E-01	1.24651E-04	7.00650E-01	6.25525E-06	0.	9.99999E-01	0	1.00000E+00
14	4	1.99714E-01	1.85738E-04	5.00526E-01	1.42256E-05	0.	9.99997E-01	0	1.00000E+00
14	5	1.99817E-01	2.26831E-04	3.00340E-01	1.94883E-05	0.	9.99996E-01	0	1.00000E+00
14	6	1.99939E-01	2.39230E-04	1.00113E-01	1.30104E-18	0.	1.00000E+00	0	5.66806E-01
14	7	1.99939E-01	2.51628E-04	0.	-2.16840E-18	0.	2.38130E-04	3	1.99905E+00
14	8	1.99939E-01	0.	0.	-1.32013E-03	0.	0.	6	1.00000E+00
14	9	0.	0.	0.	0.	0.	0.	6	1.00000E+00
14	10	0.	0.	0.	0.	0.	0.	0	1.00000E+00
15	1	1.98994E-01	0.	9.01884E-01	-1.01943E-03	0.	1.00000E+00	0	1.00000E+00
15	2	1.98994E-01	1.56920E-04	9.01884E-01	-2.34833E-04	0.	1.00000E+00	0	1.00000E+00
15	3	1.99089E-01	3.43698E-04	7.01727E-01	1.43269E-05	0.	9.99997E-01	0	1.00000E+00
15	4	1.99277E-01	4.94108E-04	5.01383E-01	2.37228E-05	0.	9.99995E-01	0	1.00000E+00
15	5	1.99538E-01	5.92957E-04	3.00889E-01	2.95150E-05	0.	9.99994E-01	0	1.00000E+00
15	6	1.99846E-01	6.23938E-04	1.00296E-01	-1.73472E-18	0.	1.00000E+00	0	5.67184E-01
15	7	1.99846E-01	6.54920E-04	0.	8.67362E-19	0.	6.22838E-04	3	1.99751E+00
15	8	1.99846E-01	0.	0.	-3.42951E-03	0.	0.	6	1.00000E+00
15	9	0.	0.	0.	0.	0.	0.	6	1.00000E+00
15	10	0.	0.	0.	0.	0.	0.	0	1.00000E+00
16	1	1.97469E-01	0.	9.04902E-01	-2.54184E-03	0.	1.00000E+00	0	1.00000E+00
16	2	1.97469E-01	4.43628E-04	9.04902E-01	-3.23702E-04	0.	1.00000E+00	0	1.00000E+00
16	3	1.97714E-01	9.06966E-04	7.04459E-01	2.51748E-05	0.	9.99995E-01	0	1.00000E+00
16	4	1.98188E-01	1.27696E-03	5.03552E-01	3.52481E-05	0.	9.99993E-01	0	1.00000E+00
16	5	1.98843E-01	1.51675E-03	3.02275E-01	4.09831E-05	0.	9.99992E-01	0	1.00000E+00
16	6	1.99614E-01	1.59395E-03	1.00758E-01	1.56125E-17	0.	1.00000E+00	0	5.68136E-01
16	7	1.99614E-01	1.67115E-03	0.	-1.21431E-17	0.	1.59285E-03	3	1.99365E+00
16	8	1.99614E-01	0.	0.	-8.74175E-03	0.	0.	6	1.00000E+00
16	9	0.	0.	0.	0.	0.	0.	6	1.00000E+00
16	10	0.	0.	0.	0.	0.	0.	0	1.00000E+00
17	1	1.93678E-01	0.	9.12496E-01	-6.31789E-03	0.	1.00000E+00	0	1.00000E+00
17	2	1.93678E-01	1.18093E-03	9.12496E-01	-4.13238E-04	0.	1.00000E+00	0	1.00000E+00
17	3	1.94295E-01	2.32844E-03	7.11315E-01	3.76637E-05	0.	9.99992E-01	0	1.00000E+00
17	4	1.95476E-01	3.24198E-03	5.08987E-01	4.76208E-05	0.	9.99990E-01	0	1.00000E+00
17	5	1.97110E-01	3.83013E-03	3.05745E-01	5.27331E-05	0.	9.99989E-01	0	1.00000E+00
17	6	1.99037E-01	4.02266E-03	1.01915E-01	-1.42247E-16	0.	1.00000E+00	0	5.70518E-01
17	7	1.99037E-01	4.21520E-03	0.	1.04083E-17	0.	4.02156E-03	3	1.98404E+00
17	8	1.99037E-01	0.	0.	-2.20387E-02	0.	0.	6	1.00000E+00
17	9	0.	0.	0.	0.	0.	0.	6	1.00000E+00
17	10	0.	0.	0.	0.	0.	0.	0	1.00000E+00
18	1	1.84290E-01	0.	9.31462E-01	-1.56465E-02	0.	1.00000E+00	0	1.00000E+00

18	2	1.84290E-01	3.03067E-03	9.31462E-01	-4.93168E-04	0.	1.00000E+00	0	1.00000E+00
18	3	1.85796E-01	5.87350E-03	7.28432E-01	5.01226E-05	0.	9.99990E-01	0	1.00000E+00
18	4	1.88698E-01	8.14470E-03	5.22558E-01	5.93493E-05	0.	9.99988E-01	0	1.00000E+00
18	5	1.92755E-01	9.60914E-03	3.14413E-01	6.34742E-05	0.	9.99987E-01	0	1.00000E+00
18	6	1.97585E-01	1.00931E-02	1.04804E-01	1.38778E-17	0.	1.00000E+00	0	5.76459E-01
18	7	1.97585E-01	1.05770E-02	0.	5.55112E-17	0.	1.00920E-02	3	1.96043E+00
18	8	1.97585E-01	0.	0.	-5.53044E-02	0.	0.	6	1.00000E+00
18	9	0.	0.	0.	0.	0.	0.	6	1.00000E+00
18	10	0.	0.	0.	0.	0.	0.	0	1.00000E+00
19	1	1.61311E-01	0.	9.78592E-01	-3.82991E-02	0.	1.00000E+00	0	1.00000E+00
19	2	1.61311E-01	7.54877E-03	9.78592E-01	-5.55200E-04	0.	1.00000E+00	0	1.00000E+00
19	3	1.64739E-01	1.45799E-02	7.71043E-01	6.08890E-05	0.	9.99988E-01	0	1.00000E+00
19	4	1.71532E-01	2.03157E-02	5.56463E-01	6.90807E-05	0.	9.99986E-01	0	1.00000E+00
19	5	1.81450E-01	2.40986E-02	3.36148E-01	7.21031E-05	0.	9.99986E-01	0	1.00000E+00
19	6	1.93817E-01	2.53548E-02	1.12049E-01	-1.11022E-16	0.	1.00000E+00	0	5.91309E-01
19	7	1.93817E-01	2.66109E-02	0.	0.	0.	2.53537E-02	3	1.90348E+00
19	8	1.93817E-01	0.	0.	-1.39336E-01	0.	0.	6	1.00000E+00
19	9	0.	0.	0.	0.	0.	0.	6	1.00000E+00
19	10	0.	0.	0.	0.	0.	0.	0	1.00000E+00
20	1	1.07451E-01	0.	1.09466E+00	-8.97665E-02	0.	1.00000E+00	0	1.00000E+00
20	2	1.07451E-01	1.78343E-02	1.09466E+00	-5.94841E-04	0.	1.00000E+00	0	1.00000E+00
20	3	1.13404E-01	3.49598E-02	8.76826E-01	6.87302E-05	0.	9.99986E-01	0	1.00000E+00
20	4	1.26254E-01	5.00677E-02	6.41866E-01	7.59059E-05	0.	9.99985E-01	0	1.00000E+00
20	5	1.48102E-01	6.11991E-02	3.91798E-01	7.79592E-05	0.	9.99984E-01	0	1.00000E+00
20	6	1.82701E-01	6.49044E-02	1.30599E-01	-1.44329E-15	0.	1.00000E+00	0	6.29242E-01
20	7	1.82701E-01	6.86097E-02	0.	1.11022E-15	0.	6.49033E-02	3	1.77021E+00
20	8	1.82701E-01	0.	0.	-3.61575E-01	0.	0.	6	1.00000E+00
20	9	0.	0.	0.	0.	0.	0.	6	1.00000E+00
20	10	0.	0.	0.	0.	0.	0.	0	1.00000E+00
21	1	0.	0.	1.37231E+00	-1.79085E-01	0.	1.00000E+00	0	1.00000E+00
21	2	0.	3.56945E-02	1.37231E+00	-6.12000E-04	0.	1.00000E+00	0	1.00000E+00
21	3	0.	7.35105E-02	1.13661E+00	7.31513E-05	0.	9.99985E-01	0	1.00000E+00
21	4	0.	1.15611E-01	8.63103E-01	7.95470E-05	0.	9.99984E-01	0	1.00000E+00
21	5	0.	1.64995E-01	5.47492E-01	8.09154E-05	0.	9.99984E-01	0	1.00000E+00
21	6	0.	2.25895E-01	1.82497E-01	1.77636E-15	0.	1.00000E+00	0	7.75872E-01
21	7	0.	2.86795E-01	0.	-5.32907E-15	0.	2.25894E-01	3	1.37761E+00
21	8	0.	0.	0.	-1.73848E+00	0.	0.	6	1.00000E+00
21	9	0.	0.	0.	0.	0.	0.	6	1.00000E+00
21	10	0.	0.	0.	0.	0.	0.	0	1.00000E+00
22	1	0.	0.	1.37231E+00	0.	0.	1.00000E+00	0	1.00000E+00
22	2	0.	3.56945E-02	1.37231E+00	1.78473E-01	0.	1.00000E+00	0	1.00000E+00
22	3	0.	7.35105E-02	1.13661E+00	1.89080E-01	0.	9.99985E-01	0	1.00000E+00
22	4	0.	1.15611E-01	8.63103E-01	2.10503E-01	0.	9.99984E-01	0	1.00000E+00
22	5	0.	1.64995E-01	5.47492E-01	2.46918E-01	0.	9.99984E-01	0	1.00000E+00
22	6	0.	2.25895E-01	1.82497E-01	3.04501E-01	0.	1.00000E+00	0	1.00000E+00
22	7	0.	0.	0.	-1.12948E+00	0.	2.25894E-01	0	1.00000E+00
22	8	0.	0.	0.	0.	0.	0.	0	1.00000E+00
22	9	0.	0.	0.	0.	0.	0.	0	1.00000E+00
22	10	0.	0.	0.	0.	0.	0.	0	1.00000E+00

SOLA - VOF UNDULAR BORE COMPARISON

ITER= 15 TIME= 1.00000E+01 DELT= 2.00000E-01 CYCLE= 50 VCHGT= -6.51366E-05

NREG= 1

K VOL(K) PR(K)

6 5.21223E+00 0.

I	J	U	V	P	D	PS	F	NF	PETA
1	1	1.71202E-01	0.	9.01134E-01	2.85336E-01	0.	1.00000E+00	0	1.00000E+00
1	2	1.71202E-01	-4.69811E-05	9.01134E-01	2.85101E-01	0.	1.00000E+00	0	1.00000E+00
1	3	1.72089E-01	-4.05679E-05	7.01120E-01	2.86847E-01	0.	1.00000E+00	0	1.00000E+00
1	4	1.73874E-01	1.70440E-05	5.01076E-01	2.90078E-01	0.	9.99993E-01	0	1.00000E+00
1	5	1.76561E-01	1.27146E-04	3.01000E-01	2.94819E-01	0.	9.99981E-01	0	1.00000E+00
1	6	1.80171E-01	2.60132E-04	1.00895E-01	3.00950E-01	0.	9.99988E-01	0	1.00000E+00
1	7	1.78231E-01	-3.17950E-01	-9.92293E-02	-1.29400E+00	0.	5.78688E-03	0	1.00000E+00
1	8	1.78231E-01	0.	0.	1.88680E+00	0.	0.	0	1.00000E+00
1	9	0.	0.	0.	0.	0.	0.	0	1.00000E+00
1	10	0.	0.	0.	0.	0.	0.	0	1.00000E+00
2	1	1.71016E-01	0.	9.01134E-01	-3.08860E-04	0.	1.00000E+00	0	1.00000E+00
2	2	1.71016E-01	2.42050E-05	9.01134E-01	-1.87834E-04	0.	1.00000E+00	0	1.00000E+00
2	3	1.71870E-01	9.74291E-05	7.01120E-01	1.90782E-06	0.	1.00000E+00	0	1.00000E+00
2	4	1.73600E-01	1.91640E-04	5.01076E-01	1.45020E-05	0.	9.99993E-01	0	1.00000E+00
2	5	1.76229E-01	3.06372E-04	3.01000E-01	2.07585E-05	0.	9.99981E-01	0	1.00000E+00
2	6	1.79802E-01	4.31738E-04	1.00895E-01	1.12637E-05	0.	9.99988E-01	0	5.70794E-01
2	7	1.77856E-01	5.56760E-04	-9.92293E-02	-3.46945E-18	0.	5.78688E-03	3	1.98294E+00
2	8	1.77856E-01	0.	0.	-3.40891E-03	0.	0.	6	1.00000E+00
2	9	0.	0.	0.	0.	0.	0.	6	1.00000E+00
2	10	0.	0.	0.	0.	0.	0.	0	1.00000E+00
3	1	1.65497E-01	0.	9.14507E-01	-9.19847E-03	0.	1.00000E+00	0	1.00000E+00
3	2	1.65497E-01	1.77150E-03	9.14507E-01	-3.40949E-04	0.	1.00000E+00	0	1.00000E+00
3	3	1.66354E-01	3.61620E-03	7.14211E-01	2.94632E-05	0.	9.99991E-01	0	1.00000E+00
3	4	1.68073E-01	5.46534E-03	5.13585E-01	3.29782E-05	0.	9.99978E-01	0	1.00000E+00
3	5	1.70657E-01	7.32991E-03	3.12618E-01	3.68552E-05	0.	9.99969E-01	0	1.00000E+00
3	6	1.74110E-01	9.23140E-03	1.11293E-01	2.21970E-05	0.	9.99979E-01	0	6.22985E-01
3	7	1.73641E-01	1.06364E-02	-9.04088E-02	0.	0.	5.83224E-02	3	1.79108E+00
3	8	1.73641E-01	0.	0.	-6.02072E-02	0.	0.	6	1.00000E+00
3	9	0.	0.	0.	0.	0.	0.	6	1.00000E+00
3	10	0.	0.	0.	0.	0.	0.	0	1.00000E+00
4	1	1.53082E-01	0.	9.30746E-01	-2.06916E-02	0.	1.00000E+00	0	1.00000E+00
4	2	1.53082E-01	4.04501E-03	9.30746E-01	-4.66508E-04	0.	1.00000E+00	0	1.00000E+00
4	3	1.54004E-01	8.16464E-03	7.30192E-01	1.52280E-05	0.	9.99980E-01	0	1.00000E+00
4	4	1.55861E-01	1.22439E-02	5.29049E-01	4.37229E-05	0.	9.99951E-01	0	1.00000E+00
4	5	1.58662E-01	1.62552E-02	3.27307E-01	6.48254E-05	0.	9.99936E-01	0	1.00000E+00
4	6	1.62420E-01	2.01605E-02	1.24953E-01	4.22933E-05	0.	9.99957E-01	0	6.92241E-01
4	7	1.64447E-01	2.32252E-02	-7.80141E-02	1.11022E-16	0.	1.32449E-01	3	1.58115E+00
4	8	1.64447E-01	0.	0.	-1.31449E-01	0.	0.	6	1.00000E+00
4	9	0.	0.	0.	0.	0.	0.	6	1.00000E+00
4	10	0.	0.	0.	0.	0.	0.	0	1.00000E+00
5	1	1.32652E-01	0.	9.52022E-01	-3.40515E-02	0.	1.00000E+00	0	1.00000E+00
5	2	1.32652E-01	6.67899E-03	9.52022E-01	-6.56544E-04	0.	1.00000E+00	0	1.00000E+00
5	3	1.33651E-01	1.34840E-02	7.51247E-01	1.03949E-04	0.	9.99948E-01	0	1.00000E+00
5	4	1.35644E-01	2.02429E-02	5.49628E-01	9.88511E-05	0.	9.99921E-01	0	1.00000E+00
5	5	1.38664E-01	2.69228E-02	3.47145E-01	6.84258E-05	0.	9.99891E-01	0	1.00000E+00
5	6	1.42767E-01	3.34820E-02	1.43769E-01	4.10681E-05	0.	9.99944E-01	0	7.82554E-01
5	7	1.46716E-01	3.93924E-02	-6.05361E-02	-4.44089E-16	0.	2.33555E-01	3	1.36322E+00
5	8	1.46716E-01	0.	0.	-2.26514E-01	0.	0.	6	1.00000E+00
5	9	0.	0.	0.	0.	0.	0.	6	1.00000E+00
5	10	0.	0.	0.	0.	0.	0.	0	1.00000E+00
6	1	1.03536E-01	0.	9.80072E-01	-4.85256E-02	0.	1.00000E+00	0	1.00000E+00

6	2	1.03536E-01	9.59386E-03	9.80072E-01	-5.56238E-04	0.	1.00000E+00	0	1.00000E+00
6	3	1.04450E-01	1.93309E-02	7.79128E-01	1.59568E-05	0.	9.99924E-01	0	1.00000E+00
6	4	1.06302E-01	2.91145E-02	5.77205E-01	1.49955E-05	0.	9.99889E-01	0	1.00000E+00
6	5	1.09160E-01	3.89642E-02	3.74250E-01	7.52387E-05	0.	9.99862E-01	0	1.00000E+00
6	6	1.13117E-01	4.88644E-02	1.70167E-01	8.49667E-05	0.	9.99901E-01	0	9.01226E-01
6	7	1.17764E-01	5.85149E-02	-3.51579E-02	-2.22045E-16	0.	3.74686E-01	3	1.14327E+00
6	8	1.17764E-01	0.	0.	-3.40827E-01	0.	0.	6	1.00000E+00
6	9	0.	0.	0.	0.	0.	0.	6	1.00000E+00
6	10	0.	0.	0.	0.	0.	0.	0	1.00000E+00
7	1	6.63208E-02	0.	1.01534E+00	-6.20257E-02	0.	1.00000E+00	0	1.00000E+00
7	2	6.63208E-02	1.22500E-02	1.01534E+00	-7.75862E-04	0.	1.00000E+00	0	1.00000E+00
7	3	6.68957E-02	2.47945E-02	8.14505E-01	1.31949E-04	0.	9.99891E-01	0	1.00000E+00
7	4	6.80334E-02	3.75861E-02	6.12746E-01	1.77198E-04	0.	9.99845E-01	0	1.00000E+00
7	5	6.98267E-02	5.07183E-02	4.09968E-01	1.06578E-04	0.	9.99782E-01	0	1.00000E+00
7	6	7.24176E-02	6.42919E-02	2.06047E-01	3.52614E-05	0.	9.99874E-01	0	1.00000E+00
7	7	7.57704E-02	7.82899E-02	7.94701E-04	-8.88178E-16	0.	5.68336E-01	3	9.36035E-01
7	8	7.57704E-02	0.	0.	-4.61439E-01	0.	0.	6	1.00000E+00
7	9	0.	0.	0.	0.	0.	0.	6	1.00000E+00
7	10	0.	0.	0.	0.	0.	0.	0	1.00000E+00
8	1	2.44519E-02	0.	1.05614E+00	1.36025E+00	0.	1.00000E+00	0	1.00000E+00
8	2	2.44519E-02	1.38794E-02	1.05614E+00	-3.84366E-04	0.	1.00000E+00	0	1.00000E+00
8	3	2.41762E-02	2.81307E-02	8.55678E-01	5.76300E-05	0.	9.99852E-01	0	1.00000E+00
8	4	2.36832E-02	4.29116E-02	6.54734E-01	-1.27246E-05	0.	9.99805E-01	0	1.00000E+00
8	5	2.32003E-02	5.84587E-02	4.53225E-01	2.49680E-05	0.	9.99761E-01	0	1.00000E+00
8	6	2.31376E-02	7.48993E-02	2.51035E-01	6.98324E-05	0.	9.99831E-01	0	1.00000E+00
8	7	2.41724E-02	9.20987E-02	4.80841E-02	8.88178E-16	0.	8.18601E-01	3	7.58380E-01
8	8	-2.86006E-01	1.02842E-01	0.	-5.49242E-01	0.	0.	6	1.00000E+00
8	9	-2.86006E-01	0.	0.	-9.90888E-01	0.	0.	6	1.00000E+00
8	10	-2.86006E-01	0.	0.	-4.76676E-01	0.	0.	0	1.00000E+00
9	1	-1.47377E-02	0.	1.09750E+00	1.36471E+00	0.	1.00000E+00	0	1.00000E+00
9	2	-1.47377E-02	1.29865E-02	1.09750E+00	-3.83721E-04	0.	1.00000E+00	0	1.00000E+00
9	3	-1.67178E-02	2.66393E-02	8.97824E-01	1.07595E-04	0.	9.99818E-01	0	1.00000E+00
9	4	-2.08479E-02	4.15094E-02	6.98337E-01	1.31983E-04	0.	9.99798E-01	0	1.00000E+00
9	5	-2.73672E-02	5.83754E-02	4.98720E-01	5.08354E-05	0.	9.99699E-01	0	1.00000E+00
9	6	-3.66710E-02	7.83136E-02	2.98382E-01	9.67626E-06	0.	9.99810E-01	0	1.00000E+00
9	7	-4.94141E-02	1.02842E-01	9.62297E-02	-1.62204E-07	0.	9.97697E-01	0	7.24570E-01
9	8	-2.86006E-01	1.02842E-01	-1.10829E-01	0.	0.	1.68047E-01	3	1.49690E+00
9	9	-2.86006E-01	0.	0.	-5.14212E-01	0.	0.	6	1.00000E+00
9	10	-2.86006E-01	0.	0.	0.	0.	0.	0	1.00000E+00
10	1	-3.58835E-02	0.	1.13259E+00	1.30120E+00	0.	9.99980E-01	0	1.00000E+00
10	2	-3.58835E-02	7.06871E-03	1.13259E+00	1.00496E-04	0.	9.99980E-01	0	1.00000E+00
10	3	-3.85574E-02	1.43372E-02	9.34658E-01	-5.69075E-05	0.	9.99834E-01	0	1.00000E+00
10	4	-4.40317E-02	2.20554E-02	7.39047E-01	-4.83489E-05	0.	9.99780E-01	0	1.00000E+00
10	5	-5.25837E-02	3.04628E-02	5.46090E-01	9.37173E-06	0.	9.99639E-01	0	1.00000E+00
10	6	-6.46277E-02	3.97873E-02	3.56369E-01	2.76825E-05	0.	9.99777E-01	0	1.00000E+00
10	7	-8.07383E-02	5.02330E-02	1.70864E-01	2.14387E-05	0.	9.99984E-01	0	9.93286E-01
10	8	-2.67288E-01	4.39938E-02	-1.12072E-02	3.33067E-16	0.	4.91242E-01	3	1.00884E+00
10	9	-2.67288E-01	0.	0.	-1.88773E-01	0.	0.	6	1.00000E+00
10	10	-2.67288E-01	0.	0.	3.11961E-02	0.	0.	0	1.00000E+00
11	1	-3.30322E-02	0.	1.14703E+00	1.09918E+00	0.	9.99975E-01	0	1.00000E+00
11	2	-3.30322E-02	-9.25193E-04	1.14703E+00	1.26104E-04	0.	9.99975E-01	0	1.00000E+00
11	3	-3.49919E-02	-2.10340E-03	9.49279E-01	5.14243E-05	0.	9.99781E-01	0	1.00000E+00
11	4	-3.89819E-02	-3.78291E-03	7.53943E-01	1.87385E-05	0.	9.99753E-01	0	1.00000E+00
11	5	-4.51480E-02	-6.26428E-03	5.61227E-01	-1.39956E-05	0.	9.99665E-01	0	1.00000E+00
11	6	-5.37220E-02	-9.90557E-03	3.71424E-01	-3.02055E-05	0.	9.99764E-01	0	1.00000E+00
11	7	-6.91378E-02	-1.37759E-02	1.84874E-01	-1.74859E-05	0.	9.99991E-01	0	9.72483E-01
11	8	-2.18886E-01	-2.99098E-02	-3.85225E-04	0.	0.	4.64332E-01	3	1.03699E+00
11	9	-2.18886E-01	0.	0.	2.30219E-01	0.	0.	6	1.00000E+00
11	10	-2.18886E-01	0.	0.	8.06696E-02	0.	0.	0	1.00000E+00
12	1	-1.22797E-02	0.	1.14065E+00	8.02197E-01	0.	9.99952E-01	0	1.00000E+00

12	2	-1.22797E-02	-6.86989E-03	1.14065E+00	2.38148E-04	0.	9.99952E-01	0	1.00000E+00
12	3	-1.25660E-02	-1.43431E-02	9.41924E-01	1.04256E-05	0.	9.99784E-01	0	1.00000E+00
12	4	-1.30737E-02	-2.29797E-02	7.44557E-01	-2.14848E-06	0.	9.99779E-01	0	1.00000E+00
12	5	-1.37293E-02	-3.34566E-02	5.48607E-01	-2.03261E-05	0.	9.99642E-01	0	1.00000E+00
12	6	-1.54426E-02	-4.62235E-02	3.54153E-01	-3.53204E-05	0.	9.99810E-01	0	1.00000E+00
12	7	-3.65205E-02	-5.71014E-02	1.61263E-01	-2.73783E-05	0.	9.99987E-01	0	8.15738E-01
12	8	-1.53522E-01	-7.88895E-02	-3.15271E-02	4.44089E-16	0.	2.72038E-01	3	1.29527E+00
12	9	-1.53522E-01	0.	0.	5.03388E-01	0.	0.	6	1.00000E+00
12	10	-1.53522E-01	0.	0.	1.08941E-01	0.	0.	0	1.00000E+00
13	1	1.13187E-02	0.	1.12166E+00	8.06940E-01	0.	9.99899E-01	0	1.00000E+00
13	2	1.13187E-02	-7.81019E-03	1.12166E+00	2.79720E-04	0.	9.99899E-01	0	1.00000E+00
13	3	1.26688E-02	-1.62330E-02	9.21468E-01	-5.62347E-05	0.	9.99814E-01	0	1.00000E+00
13	4	1.55713E-02	-2.57862E-02	7.21081E-01	-2.43617E-05	0.	9.99753E-01	0	1.00000E+00
13	5	2.02432E-02	-3.71137E-02	5.20348E-01	-1.65736E-05	0.	9.99647E-01	0	1.00000E+00
13	6	2.37593E-02	-5.01851E-02	3.19025E-01	-2.05511E-05	0.	9.99832E-01	0	1.00000E+00
13	7	-6.12947E-03	-6.03183E-02	1.16941E-01	-1.43199E-05	0.	9.99236E-01	0	5.02943E-01
13	8	-1.53522E-01	-6.03183E-02	-8.51124E-02	0.	0.	3.77410E-03	3	2.29189E+00
13	9	-1.53522E-01	0.	0.	3.01591E-01	0.	0.	6	1.00000E+00
13	10	-1.53522E-01	0.	0.	0.	0.	0.	0	1.00000E+00
14	1	2.31499E-02	0.	1.10281E+00	1.97186E-02	0.	9.99868E-01	0	1.00000E+00
14	2	2.31499E-02	-3.89795E-03	1.10281E+00	2.28810E-04	0.	9.99868E-01	0	1.00000E+00
14	3	2.49349E-02	-7.99463E-03	9.01556E-01	-3.97632E-05	0.	9.99802E-01	0	1.00000E+00
14	4	2.87280E-02	-1.23866E-02	6.98977E-01	-3.17492E-05	0.	9.99790E-01	0	1.00000E+00
14	5	3.46468E-02	-1.71920E-02	4.94819E-01	-2.11974E-05	0.	9.99677E-01	0	1.00000E+00
14	6	3.81003E-02	-2.19761E-02	2.88717E-01	-1.88802E-05	0.	9.99814E-01	0	1.00000E+00
14	7	1.21378E-02	-2.80652E-02	8.05345E-02	-1.11022E-16	0.	8.59473E-01	3	7.35579E-01
14	8	1.21378E-02	-6.03183E-02	0.	1.14834E-01	0.	0.	6	1.00000E+00
14	9	0.	0.	0.	5.57461E-01	0.	0.	6	1.00000E+00
14	10	0.	0.	0.	2.55870E-01	0.	0.	0	1.00000E+00
15	1	1.92194E-02	0.	1.09433E+00	-6.55072E-03	0.	9.99808E-01	0	1.00000E+00
15	2	1.92194E-02	1.34187E-03	1.09433E+00	1.58634E-04	0.	9.99808E-01	0	1.00000E+00
15	3	1.99846E-02	2.98878E-03	8.92963E-01	-1.59971E-05	0.	9.99780E-01	0	1.00000E+00
15	4	2.15725E-02	5.36780E-03	6.90119E-01	-3.08079E-05	0.	9.99741E-01	0	1.00000E+00
15	5	2.38088E-02	8.97567E-03	4.85544E-01	-2.39699E-05	0.	9.99649E-01	0	1.00000E+00
15	6	2.25000E-02	1.41729E-02	2.78841E-01	-1.42311E-05	0.	9.99877E-01	0	1.00000E+00
15	7	-4.57681E-03	1.97445E-02	6.97088E-02	-4.44089E-16	0.	8.52524E-01	3	7.39358E-01
15	8	-4.39488E-02	3.40938E-02	0.	-2.17310E-02	0.	0.	6	1.00000E+00
15	9	-4.39488E-02	0.	0.	-2.43717E-01	0.	0.	6	1.00000E+00
15	10	0.	0.	0.	0.	0.	0.	0	1.00000E+00
16	1	8.35293E-03	0.	1.09799E+00	2.01633E-01	0.	9.99779E-01	0	1.00000E+00
16	2	8.35293E-03	3.63554E-03	1.09799E+00	6.68409E-05	0.	9.99779E-01	0	1.00000E+00
16	3	7.96579E-03	7.64147E-03	8.97527E-01	-1.76568E-06	0.	9.99825E-01	0	1.00000E+00
16	4	7.16005E-03	1.24459E-02	6.96584E-01	1.41066E-06	0.	9.99802E-01	0	1.00000E+00
16	5	5.71694E-03	1.84751E-02	4.95108E-01	-7.26061E-06	0.	9.99745E-01	0	1.00000E+00
16	6	5.08704E-04	2.58029E-02	2.93005E-01	-1.27656E-05	0.	9.99815E-01	0	1.00000E+00
16	7	-2.94495E-02	3.40938E-02	9.01957E-02	-2.22045E-16	0.	9.78905E-01	0	5.17361E-01
16	8	-4.39488E-02	3.40938E-02	0.	0.	0.	1.84457E-03	3	2.21946E+00
16	9	-4.39488E-02	0.	0.	-1.70469E-01	0.	0.	6	1.00000E+00
16	10	-4.39488E-02	0.	0.	-7.32480E-02	0.	0.	0	1.00000E+00
17	1	9.71038E-04	0.	1.10562E+00	2.07441E-01	0.	9.99787E-01	0	1.00000E+00
17	2	9.71038E-04	2.47186E-03	1.10562E+00	5.61234E-05	0.	9.99787E-01	0	1.00000E+00
17	3	1.43715E-04	5.07864E-03	9.06213E-01	-2.86241E-06	0.	9.99840E-01	0	1.00000E+00
17	4	-1.59124E-03	7.99305E-03	7.07532E-01	-1.34257E-05	0.	9.99820E-01	0	1.00000E+00
17	5	-4.49682E-03	1.13959E-02	5.09848E-01	-8.81626E-06	0.	9.99673E-01	0	1.00000E+00
17	6	-1.08870E-02	1.51934E-02	3.13630E-01	-5.01013E-06	0.	9.99880E-01	0	1.00000E+00
17	7	-3.48666E-02	1.69985E-02	1.19529E-01	-2.91091E-06	0.	9.99941E-01	0	7.01803E-01
17	8	-4.39488E-02	1.69985E-02	-7.14386E-02	0.	0.	1.42910E-01	3	1.55543E+00
17	9	-4.39488E-02	0.	0.	-8.49927E-02	0.	0.	6	1.00000E+00
17	10	-4.39488E-02	0.	0.	0.	0.	0.	0	1.00000E+00
18	1	5.47753E-04	0.	1.10767E+00	2.19039E-01	0.	9.99773E-01	0	1.00000E+00

18	2	5.47753E-04	1.43718E-04	1.10767E+00	1.31149E-05	0.	9.99773E-01	0	1.00000E+00
18	3	3.32838E-04	8.18008E-05	9.08175E-01	5.61878E-06	0.	9.99827E-01	0	1.00000E+00
18	4	2.03769E-05	-4.54034E-04	7.09235E-01	6.85064E-06	0.	9.99826E-01	0	1.00000E+00
18	5	-1.79640E-04	-1.89326E-03	5.10980E-01	-8.24164E-07	0.	9.99774E-01	0	1.00000E+00
18	6	-7.39297E-04	-5.27707E-03	3.13614E-01	-6.20541E-06	0.	9.99851E-01	0	1.00000E+00
18	7	-1.15533E-02	-1.30490E-02	1.17402E-01	-4.46959E-06	0.	9.99864E-01	0	6.50946E-01
18	8	-4.39488E-02	-1.30490E-02	-7.73590E-02	0.	0.	8.79075E-02	3	1.70095E+00
18	9	-4.39488E-02	0.	0.	6.52452E-02	0.	0.	6	1.00000E+00
18	10	-4.39488E-02	0.	0.	0.	0.	0.	0	1.00000E+00
19	1	1.56423E-03	0.	1.10521E+00	-1.46860E-01	0.	9.99724E-01	0	1.00000E+00
19	2	1.56423E-03	-3.34333E-04	1.10521E+00	2.24567E-05	0.	9.99724E-01	0	1.00000E+00
19	3	1.59407E-03	-7.55088E-04	9.05067E-01	-1.72012E-06	0.	9.99889E-01	0	1.00000E+00
19	4	1.66926E-03	-1.30532E-03	7.04714E-01	-3.03296E-06	0.	9.99854E-01	0	1.00000E+00
19	5	1.97210E-03	-2.02311E-03	5.04006E-01	-2.71962E-06	0.	9.99770E-01	0	1.00000E+00
19	6	2.96487E-03	-3.25834E-03	3.02675E-01	-2.54447E-06	0.	9.99855E-01	0	1.00000E+00
19	7	7.41727E-03	-9.58187E-03	1.00250E-01	0.	0.	9.85692E-01	3	6.73087E-01
19	8	7.41727E-03	-1.30490E-02	0.	6.82743E-02	0.	0.	6	1.00000E+00
19	9	2.97108E-02	0.	0.	1.88011E-01	0.	0.	6	1.00000E+00
19	10	2.97108E-02	0.	0.	1.22766E-01	0.	0.	0	1.00000E+00
20	1	1.69267E-03	0.	1.10403E+00	-1.48340E-01	0.	9.99711E-01	0	1.00000E+00
20	2	1.69267E-03	-3.94098E-05	1.10403E+00	1.70143E-05	0.	9.99711E-01	0	1.00000E+00
20	3	1.88614E-03	-1.37777E-04	9.03821E-01	-5.05515E-06	0.	9.99879E-01	0	1.00000E+00
20	4	2.17165E-03	-3.05351E-04	7.03371E-01	-5.52644E-07	0.	9.99862E-01	0	1.00000E+00
20	5	2.63270E-03	-5.25487E-04	5.02596E-01	3.22633E-07	0.	9.99827E-01	0	1.00000E+00
20	6	3.56154E-03	-7.24570E-04	3.01362E-01	-9.75636E-07	0.	9.99849E-01	0	1.00000E+00
20	7	9.68248E-03	-1.47964E-03	9.94611E-02	0.	0.	9.95408E-01	3	6.68714E-01
20	8	0.	7.21095E-03	0.	3.10908E-02	0.	0.	6	1.00000E+00
20	9	0.	0.	0.	-8.55727E-02	0.	0.	6	1.00000E+00
20	10	2.97108E-02	0.	0.	0.	0.	0.	0	1.00000E+00
21	1	0.	0.	1.10469E+00	-2.82111E-03	0.	9.99676E-01	0	1.00000E+00
21	2	0.	5.67556E-04	1.10469E+00	1.66703E-05	0.	9.99676E-01	0	1.00000E+00
21	3	0.	1.19613E-03	9.04766E-01	-6.73955E-07	0.	9.99903E-01	0	1.00000E+00
21	4	0.	1.91956E-03	7.04941E-01	-2.26797E-06	0.	9.99862E-01	0	1.00000E+00
21	5	0.	2.79671E-03	5.05289E-01	-2.07819E-06	0.	9.99831E-01	0	1.00000E+00
21	6	0.	3.98360E-03	3.05934E-01	-1.48126E-06	0.	9.99834E-01	0	1.00000E+00
21	7	0.	7.21095E-03	1.07077E-01	-7.03552E-07	0.	9.99834E-01	0	6.12988E-01
21	8	0.	7.21095E-03	-9.09926E-02	0.	0.	4.78552E-02	3	1.82530E+00
21	9	0.	0.	0.	-3.60548E-02	0.	0.	6	1.00000E+00
21	10	0.	0.	0.	-4.95180E-02	0.	0.	0	1.00000E+00
22	1	0.	0.	1.10469E+00	0.	0.	9.99676E-01	0	1.00000E+00
22	2	0.	5.67556E-04	1.10469E+00	2.83778E-03	0.	9.99676E-01	0	1.00000E+00
22	3	0.	1.19613E-03	9.04766E-01	3.14289E-03	0.	9.99903E-01	0	1.00000E+00
22	4	0.	1.91956E-03	7.04941E-01	3.61715E-03	0.	9.99862E-01	0	1.00000E+00
22	5	0.	2.79671E-03	5.05289E-01	4.38575E-03	0.	9.99831E-01	0	1.00000E+00
22	6	0.	3.98360E-03	3.05934E-01	5.93441E-03	0.	9.99834E-01	0	1.00000E+00
22	7	0.	7.21095E-03	1.07077E-01	1.61368E-02	0.	9.99834E-01	0	1.00000E+00
22	8	0.	1.71145E-02	-9.09926E-02	4.95180E-02	0.	4.78552E-02	0	1.00000E+00
22	9	0.	0.	0.	-8.55727E-02	0.	0.	0	1.00000E+00
22	10	0.	0.	0.	0.	0.	0.	0	1.00000E+00

Results from additional sample problems are contained in App. B. These problems will give the prospective user some idea of the broad range of problems that can be investigated with the SOLA-VOF program.

ACKNOWLEDGMENT

Development of the SOLA-VOF program was made possible through the support of the Electric Power Research Institute, contract RP-965-3, and in part by the Office of Naval Research, contract NA-ONR-6-75, NR062-455 and the NASA Lewis Research Center, contract C-14898-D. This work was performed with the cooperation of the US Department of Energy. The authors especially wish to thank Mrs. Juanita Salazar for her superb typing and assembly of this report.

APPENDIX A

CODE LISTING

The SOLA-VOF code listing is a source file containing a single COMMON deck named COMMON1 at its beginning. This COMDECK is to be included in the main DECK (SOLA-VOF) at every occurrence of *CALL COMMON1 or *CALL,COMMON1. The inclusion of this COMDECK in the code produces a COMPILE file that can then be compiled and executed. Thus, the user can make a single change in the COMDECK and have it automatically inserted in each subroutine by his own UPDATE system or automated text editor.

The SOLA-VOF code is an ANSI standard code with the exception of the graphics routines. These routines are system dependent and must be replaced with their equivalents at each computing facility or bypassed in the event that a facility has no graphics capability. SOLA-VOF has been written to minimize the system graphics calls. It utilizes the SC-4020 to display graphic and alphanumeric information. The SC-4020 is a CRT device spanned by a 1024 x 1024 raster grid with an origin at the upper left corner. To maximize utility, SOLA-VOF first scales and centers the region to be displayed on a unit square with the origin at the lower left. SOLA-VOF then maps the unit image onto the SC-4020 1024 x 1024 screen. Thus, omission of the last procedure enables the use of a variety of devices that require a unit (or scaled unit) image.

The system-dependent graphics calls that appear in SOLA-VOF are defined below.

CALL ADV(n) - ADVances the film by n frames. This call occurs in SUBROUTINE DRAW.

CALL DRV(IX1,IY1,IX2,IY2) - DRaws a Vector between the SC-4020 raster points (IX1,IY1) and (IX2,IY2). This call occurs only once in SUBROUTINE DRWVEC.

CALL LINCNT(n) - Provides a LINe CouNT for writing formatted alphanumeric character strings on a display at line n. This call occurs in SUBROUTINE DRAW.

CALL PLT(IX1,IY1,ICHAR) - PLoTs the plotting character specified by ICHAR (usually a dot) at the SC-4020 raster point (IX1,IY1). This call occurs only once in SUBROUTINE PLTPT.

In addition to these calls, SUBROUTINE FILMSET makes calls to GFR80, GRPHLUN(12), LIB4020, GRPHFTN, and SETFLSH. These routines are specific to the

graphics system at LASL and are of no concern to the outside user. They do nothing more than initialize the film file, establish logical unit 12 as the graphics output unit for formatted writes, provide libraries for the graphics calls, and provide a mechanism to flush the buffers at the end of a calculation. Each user must recode SUBROUTINE FILMSET to provide similar information in the manner required by each facility.

```
 1  *COMDECK,COMMON1
 2        PARAMETER (IBAR2=22, JBAR2=10, NPRTS=1, MESHX=1, MESHY=1, NVOR=10)
 3        PARAMETER (MSHX=MESHX+1, MSHY=MESHY+1)
 4  C
 5        REAL NU, NORMX, NORMY
 6        INTEGER CYCLE, WL, WR, WT, WB
 7  C
 8        COMMON /FV/ ACOM(1), UN(IBAR2,JBAR2), VN(IBAR2,JBAR2), PN(IBAR2
 9       1 ,JBAR2), FN(IBAR2,JBAR2), U(IBAR2,JBAR2), V(IBAR2,JBAR2), P(IBAR2
10       2 ,JBAR2), F(IBAR2,JBAR2), PETA(IBAR2,JBAR2), BETA(IBAR2,JBAR2), NF
11       3 (IBAR2,JBAR2), TANTH(IBAR2,JBAR2), PS(IBAR2,JBAR2)
12  C
13        COMMON /ME/ X(IBAR2), XI(IBAR2), RXI(IBAR2), DELX(IBAR2), RDX
14       1 (IBAR2), RX(IBAR2), Y(JBAR2), YJ(JBAR2), RYJ(JBAR2), DELY(JBAR2),
15       2 RDY(JBAR2), XL(MSHX), XC(MESHX), DXMN(MESHX), NXL(MESHX), NXR
16       3 (MESHX), YL(MSHY), YC(MESHY), DYMN(MESHY), NYL(MESHY), NYR(MESHY)
17        COMMON /PV/ XP(NPRTS), YP(NPRTS), IP(NPRTS), JP(NPRTS), NR(NVOR),
18       1 PR(NVOR), VOL(NVOR), NAME(10)
19  C
20        COMMON /IV/ IBAR, JBAR, IMAX, JMAX, IM1, JM1, IM2, JM2, NKX, NKY,
21       1 CYCLE, DELT, T, AUTOT, PRTDT, TWPRT, PLTDT, TWPLT, TWFIN, FLHT,
22       2 NU, CSQ, RCSQ, NMAT, RHOF, RHOFC, RHOD, NVRM, NREG, VCHGT, RDTEXP
23       3 , ISURF10, SIGMA, CANGLE, TANCA, ICYL, CYL, GX, GY, UI, VI, OMG,
24       4 ALPHA, WL, WR, WB, WT, NP, ITER, EPSI, FLG, FLGC, FNOC, NOCON,
25       5 NFLGC, ISYMPLT, IMOVY, VELMX, VELMX1, XSHFT, YSHFT, XMIN, XMAX,
26       6 YMIN, YMAX, SF, XPL, XPR, YPB, YPT, NPX, NPY, IPL, IPR, JPB, JPT,
27       7 DTVIS, DTSFT
28  C
29        COMMON /CONST/ EMF, EMF1, EM6, EM10, EP10, PI, RPD
30  C
31        COMMON /LAST/ ZCOM
32  *DECK,SOLAVOF
33        PROGRAM SOLAVOF (INPUT,TAPE5=INPUT,OUTPUT,TAPE6=OUTPUT,TAPE7,TAPE8
34       1 )
35  C
36  C     *** SOLA-VOF  VOLUME OF FLUID METHOD ***
37  C
38  C     *** LIST OF PRIMARY VARIABLES ***
39  C
40  C     *** INPUT PARAMETERS (NAMELIST /XPUT/)
41  C
42  C     ALPHA      CONTROLS AMOUNT OF DONOR CELL FLUXING (=1.0 FOR FULL
43  C                DONOR CELL DIFFERENCING, =0.0 FOR CENTRAL DIFFERENCING)
44  C     AUTOT      AUTOMATIC TIME STEP FLAG (=1.0 FOR AUTOMATIC DELT
45  C                ADJUSTMENT, =0.0 FOR CONSTANT DELT)
46  C     CANGLE     CONTACT ANGLE, IN DEGRES, BETWEEN FLUID AND WALL
47  C     CSQ        MATERIAL SOUND SPEED SQUARED (=-1.0 FOR
48  C                INCOMPRESSIBLE MATERIAL)
49  C     DELT       TIME STEP
50  C     EPSI       PRESSURE ITERATION CONVERGENCE CRITERION
51  C     FLHT       FLUID HEIGHT, IN Y-DIRECTION
52  C     GX         BODY ACCELERATION IN POSITIVE X-DIRECTION
53  C     GY         BODY ACCELERATION IN POSITIVE Y-DIRECTION
54  C     ICYL       MESH GEOMETRY INDICATOR (=1 FOR CYLINDRICAL COORDINATES,
55  C                =0 FOR PLANE COORDINATES)
56  C     IMOVY      MOVIE INDICATOR (=1 FOR MOVIE FILM OUTPUT, =0 FOR
57  C                OTHER FILM OUTPUT)
58  C     ISURF10    SURFACE TENSION INDICATOR (=1 FOR SURFACE TENSION,
```

```
C                =0 FOR NO SURFACE TENSION)
C      ISYMPLT   SYMMETRY PLOT INDICATOR (=1 FOR SYMMETRY PLOT,
C                =0 FOR NO SYMMETRY PLOT)
C      NMAT      NUMBER OF MATERIALS
C      NPX       NUMBER OF PARTICLES IN X-DIRECTION IN RECTANGULAR SETUP
C      NPY       NUMBER OF PARTICLES IN Y-DIRECTION IN RECTANGULAR SETUP
C      NU        COEFFICIENT OF KINEMATIC VISCOSITY
C      OMG       OVER-RELAXATION FACTOR USED IN PRESSURE ITERATION
C      PLTDT     TIME INCREMENT BETWEEN PLOTS AND/OR PRINTS TO BE OUTPUT
C                ON FILM
C      PRTDT     TIME INCREMENT BETWEEN PRINTS ON PAPER
C      RHOF      FLUID DENSITY (FOR F=1.0 REGION)
C      RHOFC     FLUID DENSITY IN COMPLEMENT OF F REGION
C      SIGMA     SURFACE TENSION COEFFICIENT
C      TWFIN     PROBLEM TIME TO END CALCULATION
C      UI        X-DIRECTION VELOCITY USED FOR INITIALIZING MESH
C      VI        Y-DIRECTION VELOCITY USED FOR INITIALIZING MESH
C      VELMX     MAXIMUM VELOCITY EXPECTED IN PROBLEM, USED TO SCALE
C                VELOCITY VECTORS
C      WB        INDICATOR FOR BOUNDARY CONDITION TO BE USED ALONG THE
C                BOTTOM OF THE MESH (=1 FOR RIGID FREE-SLIP WALL,
C                =2 FOR RIGID NO-SLIP WALL, =3 FOR CONTINUATIVE
C                BOUNDARY, =4 FOR PERIODIC BOUNDARY, =5 FOR CONSTANT
C                PRESSURE BOUNDARY)
C      WL        INDICATOR FOR BOUNDARY CONDITION ALONG LEFT SIDE OF
C                MESH (SEE WB)
C      WR        INDICATOR FOR BOUNDARY CONDITION ALONG RIGHT SIDE OF
C                MESH (SEE WB)
C      WT        INDICATOR FOR BOUNDARY CONDITION ALONG TOP OF MESH
C                (SEE WB)
C      XPL       LOCATION OF LEFT SIDE OF RECTANGULAR PARTICLE REGION
C      XPR       LOCATION OF RIGHT SIDE OF RECTANGULAR PARTICLE REGION
C      YPB       LOCATION OF BOTTOM OF RECTANGULAR PARTICLE REGION
C      YPT       LOCATION OF TOP OF RECTANGULAR PARTICLE REGION
C
C      *** MESH SETUP INPUT (NAMELIST /MSHSET/)
C
C      DXMIN(N)     MINIMUM SPACE INCREMENT IN X-DIRECTION IN SUBMESH N
C      DYMIN(N)     MINIMUM SPACE INCREMENT IN Y-DIRECTION IN SUBMESH N
C      NKX          NUMBER OF SUBMESH REGIONS IN X-DIRECTION
C      NXL(N)       NUMBER OF CELLS BETWEEN LOCATIONS XL(N) AND XC(N) IN
C                   SUBMESH N
C      NXR(N)       NUMBER OF CELLS BETWEEN LOCATIONS XC(N) AND XL(N+1) IN
C                   SUBMESH N
C      NYL(N)       NUMBER OF CELLS BETWEEN LOCATIONS YL(N) AND YC(N) IN
C                   SUBMESH N
C      NYR(N)       NUMBER OF CELLS BETWEEN LOCATIONS YC(N) AND YL(N+1) IN
C                   SUBMESH N
C      XC(N)        X-COORDINATE OF THE CONVERGENCE POINT IN SUBMESH N
C      XL(N)        LOCATION OF THE LEFT EDGE OF SUBMESH N (NKX+1 VALUES
C                   OF XL(N) ARE NECESSARY BECAUSE THE RIGHT EDGE (XR) OF
C                   SUBMESH N IS DETERMINED BY THE LEFT EDGE OF
C                   SUBMESH N+1)
C      YC(N)        Y-COORDINATE OF THE CONVERGENCE POINT IN SUBMESH N
C      YL(N)        LOCATION OF THE BOTTOM OF SUBMESH N (NKY+1 VALUES OF
C                   YL(N) ARE NECESSARY BECAUSE THE TOP EDGE (YR) OF
C                   SUBMESH N IS DETERMINED BY THE BOTTOM EDGE OF
C                   SUBMESH N+1)
```

```
C
C     *** VARIABLES LISTED IN COMMON (EXCLUDING INPUT PARAMETERS)
C
C     CYCLE     CALCULATIONAL TIME CYCLE
C     CYL       MESH GEOMETRY INDICATOR (= ICYL)
C     DTSFT     MAXIMUM DELT VALUE ALLOWED BY THE SURFACE TENSION FORCES
C               STABILITY CRITERION (DELT IS AUTOMATICALLY ADJUSTED)
C     DTVIS     MAXIMUM DELT VALUE ALLOWED BY THE VISCOUS FORCES
C               STABILITY CRITERION (DELT IS AUTOMATICALLY ADJUSTED)
C     EMF       SMALL VALUE, TYPICALLY 1.0E-06, USED TO NEGATE ROUND-OFF
C               ERROR EFFECTS WHEN TESTING F=1.0 OR F=0.0
C     EMF1      =1.0-EMF
C     EM6       =1.0E-06
C     EM10      =1.0E-10
C     EP10      =1.0E+10
C     FLG       PRESSURE ITERATION CONVERGENCE TEST INDICATOR (=0.0 WHEN
C               THE CONVERGENCE TEST IS SATISFIED, =1.0 WHEN THE
C               CONVERGENCE TEST IS NOT SATISFIED)
C     FLGC      VOLUME OF FLUID FUNCTION CONVECTION LIMIT INDICATOR
C               (DELT REDUCED AND CYCLE STARTED OVER IF LIMIT
C               IS EXCEEDED)
C     FNOC      PRESSURE CONVERGENCE FAILURE INDICATOR (=1.0,
C               CONVERGENCE FAILED AND DELT IS REDUCED, =0.0 OTHERWISE)
C     IBAR      NUMBER OF REAL CELLS IN X-DIRECTION (EXCLUDES FICTICIOUS
C               CELLS)
C     IBAR2     =IBAR+2, SPECIFIED IN PARAMETER STATEMENT
C               (=IBAR+3, IF PERIODIC IN X-DIRECTION)
C     IMAX      TOTAL NUMBER OF MESH CELLS IN X-DIRECTION (=IBAR+2)
C               (=IBAR+3, IF PERIODIC IN X-DIRECTION)
C     IM1       VALUE OF THE INDEX I AT THE LAST REAL CELL IN THE
C               X-DIRECTION (=IMAX-1)
C     IM2       VALUE OF THE INDEX I AT THE NEXT TO THE LAST REAL CELL
C               IN THE X-DIRECTION (=IMAIN THE X-2)
C     IPL       LEFTMOST PRESSURE ITERATION INDEX IN X-DIRECTION
C               (=3 FOR CONSTANT PRESSURE BOUNDARY CONDITION, =2 FOR
C               ALL OTHER CASES)
C     IPR       RIGHTMOST PRESSURE ITERATION INDEX IN X-DIRECTION
C               (=IM2 FOR CONSTANT PRESSURE BOUNDARY CONDITION, =IM1 FOR
C               ALL OTHER CASES)
C     ITER      PRESSURE ITERATION COUNTER
C     JBAR      NUMBER OF REAL CELLS IN Y-DIRECTION (EXCLUDES FICTICIOUS
C               CELLS)
C     JBAR2     =JBAR+2, SPECIFIED IN PARAMETER STATEMENT
C               (=JBAR+3, IF PERIODIC IN Y-DIRECTION)
C     JMAX      TOTAL NUMBER OF MESH CELLS IN Y-DIRECTION (=JBAR+2)
C               (=JBAR+3, IF PERIODIC IN Y-DIRECTION)
C     JM1       VALUE OF THE INDEX J AT THE LAST REAL CELL IN THE
C               Y-DIRECTION (=JMAX-1)
C     JM2       VALUE OF THE INDEX J AT THE NEXT TO THE LAST REAL CELL
C               IN THE Y-DIRECTION (=JMAX-2)
C     JPB       BOTTOM PRESSURE ITERATION INDEX IN Y-DIRECTION
C               (=3 FOR CONSTANT PRESSURE BOUNDARY CONDITION, =2 FOR
C               ALL OTHER CASES)
C     JPT       TOP PRESSURE ITERATION INDEX IN Y-DIRECTION
C               (=JM2 FOR CONSTANT PRESSURE BOUNDARY CONDITION, =JM1 FOR
C               ALL OTHER CASES)
C     NFLGC     NUMBER OF CYCLES THE VOLUME OF FLUID FUNCTION CONVECTION
C               LIMIT (FLGC) IS EXCEEDED
```

```
175  C     NOCON     NUMBER OF CYCLES PRESSURE CONVEGENCE HAS FAILED
176  C     NP        TOTAL NUMBER OF PARTICLES COMPUTED TO BE IN MESH
177  C     NPRTS     NUMBER OF PARTICLES IN MESH, SPECFIED IN PARAMETER STATE
178  C               (USED TO SET ARRAY SIZE - MUST BE > 0)
179  C     NREG      NUMBER OF VOID REGIONS GENERATED IN CALCULATION
180  C     NVOR      MAXIMUM NUMBER OF VOID REGIONS ALLOWED, SPECIFIED IN
181  C               PARAMETER STATEMENT
182  C     NVRM      NUMBER OF VOID REGIONS
183  C     MESHX     NUMBER OF SUBMESH REGIONS IN X-DIRECTION, SPECIFIED
184  C.              IN PARAMETER STATEMENT
185  C     MESHY     NUMBER OF SUBMESH REGIONS IN Y-DIRECTION, SPECIFIED
186  C               IN PARAMETER STATEMENT
187  C     PI        =3.141592654
188  C     RCSQ      RECIPROCAL OF CSQ
189  C     RHOD      DIFFERENCE IN FLUID DENSITIES (=RHOF-RHOFC)
190  C     RPD       DEGREES TO RADIANS CONVERSION FACTOR
191  C     SF        PLOT SCALING FACTOR
192  C     T         PROBLEM TIME
193  C     TANGLE    TANGENT OF CONTACT ANGLE, CANGLE
194  C     VCHGT     ACCUMULATED FLUID VOLUME CHANGE
195  C     VELMXI    VELMX NORMALIZED TO MINIMUM MESH CELL DIMENSION
196  C     XMAX      LOCATION OF RIGHT-HAND SIDE OF MESH
197  C     XMIN      LOCATION OF LEFT-HAND SIDE OF MESH
198  C     XSHFT     COMPUTED SHIFT ALONG THE PLOTTING ABSCISSA TO CENTER
199  C               THE PLOT FRAME ON FILM
200  C     YMAX      LOCATION OF THE TOP OF THE MESH
201  C     YMIN      LOCATION OF THE BOTTOM OF THE MESH
202  C     YSHFT     COMPUTED SHIFT ALONG THE PLOTTING ORDINATE TO CENTER THE
203  C               PLOT FRAME ON FILM
204  C
205  C     *** ARRAYS IN COMMON (EXCLUDING MESH SETUP PARAMETERS)
206  C
207  C     ACOM(1)       FIRST WORD IN COMMON
208  C     BETA(I,J)     PRESSURE ITERATION RELAXATION FACTOR IN CELL (I,J)
209  C     DELX(I)       MESH SPACING OF THE I-TH CELL ALONG THE X-AXIS
210  C     DELY(J)       MESH SPACING OF THE J-TH CELL ALONG THE Y-AXIS
211  C     F(I,J)        VOLUME OF FLUID PER UNIT VOLUME OF CELL (I,J) AT TIME
212  C                   LEVEL N+1
213  C     FN(I,J)       VOLUME OF FLUID PER UNIT VOLUME OF CELL (I,J) AT TIME
214  C                   LEVEL N
215  C     IP(K)         CELL INDEX FOR PARTICLE K ALONG X-AXIS
216  C     JP(K)         CELL INDEX FOR PARTICLE K ALONG Y-AXIS
217  C     NAME(10)      PROBLEM IDENTIFICATION LINE
218  C     NF(I,J)       FLAG OF SURFACE CELL (I,J) INDICATING THE LOCATION
219  C                   OF ITS NEIGHBORING PRESSURE INTERPOLATION CELL
220  C     NR(K)         LABEL OF VOID REGION, K > 5
221  C     P(I,J)        PRESSURE IN CELL (I,J) AT TIME LEVEL N+1
222  C     PETA(I,J)     PRESSURE INTERPOLATION FACTOR FOR CELL (I,J)
223  C     PN(I,J)       PRESSURE IN CELL (I,J) AT TIME LEVEL N
224  C     PR(K)         PRESSURE IN VOID REGION NR(K)
225  C     PS(I,J)       SURFACE PRESSURE IN CELL (I,J) COMPUTED FROM SURFACE
226  C                   TENSION FORCES
227  C     RDX(I)        RECIPROCAL OF DELX(I)
228  C     RDY(J)        RECIPROCAL OF DELY(J)
229  C     RX(I)         RECIPROCAL OF X(I)
230  C     RXI(I)        RECIPROCAL OF XI(I)
231  C     RYJ(J)        RECIPROCAL OF YJ(J)
232  C     TANTH(I,J)    SLOPE OF FLUID SURFACE IN CELL (I,J)
```

```
C     U(I,J)      X-DIRECTION VELOCITY COMPONENT IN CELL (I,J) AT TIME
C                 LEVEL N+1
C     UN(I,J)     X-DIRECTION VELOCITY COMPONENT IN CELL (I,J) AT TIME
C                 LEVEL N
C     V(I,J)      Y-DIRECTION VELOCITY COMPONENT IN CELL (I,J) AT TIME
C                 LEVEL N+1
C     VN(I,J)     Y-DIRECTION VELOCITY COMPONENT IN CELL (I,J) AT TIME
C                 LEVEL N
C     VOL(K)      VOLUME OF VOID REGION NR(K)
C     X(I)        LOCATION OF THE RIGHT-HAND BOUNDARY OF THE I-TH CELL
C                 ALONG THE X-AXIS
C     XI(I)       LOCATION OF THE CENTER OF THE I-TH CELL ALONG THE
C                 X-AXIS
C     XP(K)       X-COORDINATE OF PARTICLE K
C     Y(J)        LOCATION OF THE TOP BOUNDARY OF THE J-TH CELL ALONG TH
C                 Y-AXIS
C     YJ(J)       LOCATION OF THE CENTER OF THE J-TH CELL ALONG THE
C                 Y-AXIS
C     YP(K)       Y-COORDINATE OF PARTICLE K
C     ZCOM(1)     LAST WORD IN COMMON
*CALL,COMMON1
      NAMELIST /XPUT/ DELT,NU,ICYL,EPSI,GX,GY,UI,VI,VELMX,TWFIN,PRTDT
     1 ,PLTDT,OMG,ALPHA,WL,WR,WT,WB,IMOVY,AUTOT,FLHT,ISYMPLT,SIGMA
     2 ,ISURF10,CANGLE,CSQ,NMAT,RHOF,RHOFC,XPL,XPR,YPB,YPT,NPX,NPY
      NAMELIST /MSHSET/ NKX,XL,XC,NXL,NXR,DXMN,NKY,YL,YC,NYL,NYR,DYMN
C
      DATA EMF /1.0E-06/, EM6 /1.0E-06/, EM10 /1.0E-10/
      DATA EP10 /1.0E+10/
      DATA PI /3.141592654/, RPD /0.0174532925/
C
C     *** DEFAULT INPUT DATA
C     *** NOTE    USER MUST SUPPLY THE FOLLOWING REGARDLESS
C                      OF DEFAULTS: DELT,TWFIN,PRTDT,PLTDT
C
      DATA NU /0.0/, ICYL /0/, EPSI /1.0E-03/, GX /0.0/, GY /0.0/, UI /0
     1 .0/, VI /0.0/, VELMX /1.0/, IMOVY /0/, OMG /1.7/, ALPHA /1.0/, WL
     2 /1/, WR /1/, WT /1/, WB /1/, CSQ /-1.0/, AUTOT /1.0/, ISYMPLT /0/
     3 , ISURF10 /0/, SIGMA /0.0/, CANGLE /90.0/, NMAT /1/, RHOF /1.0/,
     4 RHOFC /1.0/, FLHT /0.0/, XPL /0.0/, YPB /0.0/, XPR /0.0/, YPT /0.
     5 0/, NPX /0/, NPY /0/
C
C     *** SETUP FILM INITIALIZATION TO SYSTEM
C     *** NOTE    FILMSET IS SYSTEM DEPENDANT
C
      CALL FILMSET
C
C     *** READ PROBLEM TITLE (NAME)
C
      READ (5,110) NAME
C
C     *** READ INITIAL INPUT DATA
C
      READ (5,XPUT)
C
C     *** READ INPUT PARAMETERS FOR VARIABLE MESH
C
      READ (5,MSHSET)
C
```

```
C     *** CALCULATE VARIABLE MESH DATA
C
      CALL MESHSET
C
C     *** PRINT INITIAL INPUT DATA
C
      CALL PRTPLT (1)
C
C     *** SET INITIAL CONDITIONS
C
      CALL SETUP
C
C     *** SET INITIAL BOUNDARY CONDITIONS
C
      CALL BC
C
      GO TO 20
C
C     *** START TIME CYCLE
C
   10 CONTINUE
      ITER=0
      FLG=1.0
      FNOC=0.0
C
C     *** EXPLICITLY APPROXIMATE NEW TIME-LEVEL VELOCITIES
C
      CALL TILDE
C
      IF (NMAT.EQ.2.AND.ISURF10.EQ.1) CALL TMS10
C
C     *** SET BOUNDARY CONDITIONS
C
      CALL BC
C
C     *** ITERATIVELY ADJUST CELL PRESSURE AND VELOCITY
C
      CALL PRESSIT
C
      IF (T.GT.EPI0) GO TO 30
C
C     *** UPDATE FLUID CONFIGURATION
C
   20 CALL VFCONV
C
      IF (FLGC.GT.0.5) GO TO 90
C
C     *** SET BOUNDARY CONDITIONS
C
      CALL BC
C
C     *** MOVE MARKER PARTICLES
C
      CALL PARMOV
C
C     *** DETERMINE PRESSURE INTERPOLATION FACTOR AND NEIGHBOR
C     *** ALSO DETERMINE SURFACE TENSION PRESSURES AND
C     *** WALL ADHESION EFFECTS IN SURFACE CELLS
```

```
C
      CALL PETACAL
C
C     *** PRINT TIME AND CYCLE DATA ON PAPER AND/OR FILM
C
   30 CALL PRTPLT (2)
C
      IF (CYCLE.LE.0) GO TO 40
      IF (T+EM6.LT.TWPLT) GO TO 50
      TWPLT=TWPLT+PLTDT
   40 CONTINUE
C
C     *** PRINT FIELD VARIABLE DATA ON FILM
C
      CALL PRTPLT (3)
C
C     *** PLOT VELOCITY VECTOR, FREE SURFACE, MESH,
C     *** AND MARKER PARTICLE ON FILM
C
      CALL DRAW
C
   50 CONTINUE
      IF (CYCLE.LE.0) GO TO 60
      IF (T+EM6.LT.TWPRT) GO TO 70
      TWPRT=TWPRT+PRTDT
   60 CONTINUE
C
C     *** PRINT FIELD VARIABLE DATA ON PAPER
C
      CALL PRTPLT (4)
C
   70 CONTINUE
C
C     *** SET THE ADVANCE TIME ARRAYS INTO THE TIME-N ARRAYS
C
      DO 80 I=1,IMAX
      DO 80 J=1,JMAX
      UN(I,J)=U(I,J)
      VN(I,J)=V(I,J)
      U(I,J)=0.0
      V(I,J)=0.0
      PN(I,J)=P(I,J)
      FN(I,J)=F(I,J)
   80 CONTINUE
      NREGN=NREG
C
C     *** ADJUST DELT
C
   90 CALL DELTADJ
C
C     *** ADVANCE TIME
C
      T=T+DELT
      IF (T.GT.TWFIN) GO TO 100
C
C     *** ADVANCE CYCLE
C
      CYCLE=CYCLE+1
```

```
      IF (NFLGC.GE.25.OR.NOCON.GE.25) T=EP10
      GO TO 10
C
  100 CALL EXIT
C
  110 FORMAT (10A8)
      END
```

```
      SUBROUTINE BC
*CALL COMMON1
C
C     *** SET BOUNDARY CONDITIONS
C
      DO 100 J=1,JMAX
      F(1,J)=F(2,J)
      F(IMAX,J)=F(IM1,J)
      P(1,J)=P(2,J)
      P(IMAX,J)=P(IM1,J)
      GO TO (10,20,30,40,30), WL
   10 U(1,J)=0.0
      V(1,J)=V(2,J)
      GO TO 50
   20 U(1,J)=0.0
      V(1,J)=-V(2,J)*DELX(1)/DELX(2)
      GO TO 50
   30 IF (ITER.GT.0) GO TO 50
      U(1,J)=U(2,J)*(X(2)*RX(1)*CYL+1.0-CYL)
      V(1,J)=V(2,J)
      GO TO 50
   40 U(1,J)=U(IM2,J)
      V(1,J)=V(IM2,J)
      F(1,J)=F(IM2,J)
   50 GO TO (60,70,80,90,80), WR
   60 U(IM1,J)=0.0
      V(IMAX,J)=V(IM1,J)
      GO TO 100
   70 U(IM1,J)=0.0
      V(IMAX,J)=-V(IM1,J)*DELX(IMAX)/DELX(IM1)
      GO TO 100
   80 IF (ITER.GT.0) GO TO 100
      U(IM1,J)=U(IM2,J)*(X(IM2)*RX(IM1)*CYL+1.0-CYL)
      V(IMAX,J)=V(IM1,J)
      GO TO 100
   90 U(IM1,J)=U(2,J)
      V(IM1,J)=V(2,J)
      P(IM1,J)=P(2,J)
      PS(IM1,J)=PS(2,J)
      F(IM1,J)=F(2,J)
      V(IMAX,J)=V(3,J)
      F(IMAX,J)=F(3,J)
  100 CONTINUE
      DO 200 I=1,IMAX
      F(I,1)=F(I,2)
      F(I,JMAX)=F(I,JM1)
      P(I,1)=P(I,2)
      P(I,JMAX)=P(I,JM1)
      GO TO (110,120,130,140,130), WT
  110 V(I,JM1)=0.0
      U(I,JMAX)=U(I,JM1)
      GO TO 150
  120 V(I,JM1)=0.0
      U(I,JMAX)=-U(I,JM1)*DELY(JMAX)/DELY(JM1)
      GO TO 150
  130 IF (ITER.GT.0) GO TO 150
      V(I,JM1)=V(I,JM2)
      U(I,JMAX)=U(I,JM1)
```

```
      GO TO 150
  140 V(I,JM1)=V(I,2)
      U(I,JM1)=U(I,2)
      P(I,JM1)=P(I,2)
      PS(I,JM1)=PS(I,2)
      F(I,JM1)=F(I,2)
      U(I,JMAX)=U(I,3)
      F(I,JMAX)=F(I,3)
  150 GO TO (160,170,180,190,180), MB
  160 V(I,1)=0.0
      U(I,1)=U(I,2)
      GO TO 200
  170 V(I,1)=0.0
      U(I,1)=-U(I,2)*DELY(1)/DELY(2)
      GO TO 200
  180 IF (ITER.GT.0) GO TO 200
      V(I,1)=V(I,2)
      U(I,1)=U(I,2)
      GO TO 200
  190 V(I,1)=V(I,JM2)
      U(I,1)=U(I,JM2)
      F(I,1)=F(I,JM2)
  200 CONTINUE
C
C     *** FREE SURFACE AND SLOPED BOUNDARY CONDITIONS
C
      DO 440 I=2,IM1
      XRP=RDX(I)+0.5*RXI(I)
      RXRP=1./XRP
      XRM=RDX(I)-0.5*RXI(I)
      IF (XRM.GT.0.0) GO TO 210
      RXRM=0.0
      GO TO 220
  210 CONTINUE
      RXRM=1./XRM
  220 CONTINUE
      DO 440 J=2,JM1
      IF (BETA(I,J).GT.0.0) GO TO 230
      BMR=0.0
      BMT=0.0
      BML=0.0
      BMB=0.0
      F(I,J)=0.0
      P(I,J)=0.0
      IF (BETA(I+1,J).GT.0.0) BMR=1.0
      IF (BETA(I,J+1).GT.0.0) BMT=1.0
      IF (BETA(I-1,J).GT.0.0) BML=1.0
      IF (BETA(I,J-1).GT.0.0) BMB=1.0
      BMTOT=BMR+BMT+BML+BMB
      IF (BMTOT.LE.0.0) GO TO 440
      F(I,J)=(BMR*F(I+1,J)+BMT*F(I,J+1)+BML*F(I-1,J)+BMB*F(I,J-1))/BMTOT
      P(I,J)=(BMR*P(I+1,J)+BMT*P(I,J+1)+BML*P(I-1,J)+BMB*P(I,J-1))/BMTOT
      GO TO 440
  230 CONTINUE
      IF (NMAT.EQ.2) GO TO 440
      IF (F(I,J).LT.EMF.OR.F(I,J).GT.EMF1) GO TO 440
      NFSB=0
      IF (F(I+1,J).LT.EMF) NFSB=NFSB+1
```

```
      IF (F(I,J+1).LT.EMF) NFSB=NFSB+2
      IF (F(I-1,J).LT.EMF) NFSB=NFSB+4
      IF (F(I,J-1).LT.EMF) NFSB=NFSB+8
      IF (NFSB.EQ.0) GO TO 440
      IF (NFSB.GT.8) GO TO 240
      GO TO (250,260,270,280,290,300,310,320), NFSB
  240 NFSB1=NFSB-8
      GO TO (330,340,350,360,370,380,390), NFSB1
  250 U(I,J)=(U(I-1,J)-DELX(I)*RDY(J)*(V(I,J)-V(I,J-1)))*(1.0-CYL)+CYL*
     1 (U(I-1,J)*XRM*RXRP-RDY(J)*RXRP*(V(I,J)-V(I,J-1)))
      GO TO 400
  260 V(I,J)=(V(I,J-1)-DELY(J)*RDX(I)*(U(I,J)-U(I-1,J)))*(1.0-CYL)+CYL*
     1 (V(I,J-1)-DELY(J)*(XRP*U(I,J)-XRM*U(I-1,J)))
      GO TO 400
  270 U(I,J)=U(I-1,J)*(1.0-CYL)+CYL*U(I-1,J)
      GO TO 260
  280 U(I-1,J)=(U(I,J)+DELX(I)*RDY(J)*(V(I,J)-V(I,J-1)))*(1.0-CYL)+CYL*
     1 (U(I,J)*XRP*RXRM+RDY(J)*RXRM*(V(I,J)-V(I,J-1)))
      GO TO 400
  290 U(I-1,J)=U(I-1,J-1)
      GO TO 250
  300 U(I-1,J)=U(I,J)*(1.0-CYL)+CYL*U(I,J)
      GO TO 260
  310 U(I-1,J)=U(I-1,J-1)
      U(I,J)=U(I,J-1)
      GO TO 260
  320 V(I,J-1)=(V(I,J)+DELY(J)*RDX(I)*(U(I,J)-U(I-1,J)))*(1.0-CYL)+CYL*
     1 (V(I,J)+DELY(J)*(XRP*U(I,J)-XRM*U(I-1,J)))
      GO TO 400
  330 U(I,J)=U(I-1,J)*(1.0-CYL)+CYL*U(I-1,J)
      GO TO 320
  340 V(I,J)=V(I-1,J)
      GO TO 320
  350 V(I,J)=V(I-1,J)
      V(I,J-1)=V(I-1,J-1)
      GO TO 250
  360 U(I-1,J)=U(I,J)*(1.0-CYL)+CYL*U(I,J)
      GO TO 320
  370 U(I,J)=U(I,J+1)
      U(I-1,J)=U(I-1,J+1)
      GO TO 320
  380 V(I,J)=V(I+1,J)
      V(I,J-1)=V(I+1,J-1)
      GO TO 280
  390 U(I,J)=U(I-1,J)*(1.0-CYL)+CYL*U(I-1,J)*XRM*RXRP
      V(I,J-1)=V(I,J)
      V(I,J+1)=V(I,J)
      GO TO 400
C
C     *** SET VELOCITIES IN EMPTY CELLS ADJACENT TO PARTIAL FLUID CELLS
C
  400 CONTINUE
      IF (FLG.GT.0.5.AND.ITER.GT.0) GO TO 440
      IF (F(I+1,J).GT.EMF) GO TO 410
      IF (F(I+1,J+1).LT.EMF) V(I+1,J)=V(I,J)
      IF (F(I+1,J-1).LT.EMF) V(I+1,J-1)=V(I,J-1)
  410 IF (F(I,J+1).GT.EMF) GO TO 420
      IF (F(I+1,J+1).LT.EMF) U(I,J+1)=U(I,J)
```

```
588         IF (F(I-1,J+1).LT.EMF) U(I-1,J+1)=U(I-1,J)
589     420 IF (F(I-1,J).GT.EMF) GO TO 430
590         IF (F(I-1,J+1).LT.EMF) V(I-1,J)=V(I,J)
591         IF (F(I-1,J-1).LT.EMF) V(I-1,J-1)=V(I,J-1)
592     430 IF (F(I,J-1).GT.EMF) GO TO 440
593         IF (F(I+1,J-1).LT.EMF) U(I,J-1)=U(I,J)
594         IF (F(I-1,J-1).LT.EMF) U(I-1,J-1)=U(I-1,J)
595     440 CONTINUE
596   C
597   C     *** SPECIAL VELOCITY BOUNDARY CONDITIONS
598   C
599         RETURN
600         END
```

```
601         SUBROUTINE CAVOVO
602   *CALL,COMMON1
603   C
604   C       *** CALCULATE VOID VOLUMES
605   C
606   C       *** INITIALIZE VOID VOLUMES
607   C
608         DO 10 K=1,NVRM
609      10 VOL(K)=0.0
610   C
611   C       *** COMPUTE VOID REGION VOLUMES
612   C
613         DO 30 J=2,JM1
614         DO 30 I=2,IM1
615         INF=NF(I,J)
616         IF (INF.EQ.0.OR.BETA(I,J).LT.0.0) GO TO 30
617         VOLA=(1.0-F(I,J))*DELX(I)*DELY(J)*(1.0-CYL+CYL*2.0*3.14159*XI(I))
618         IF (INF.GT.5) GO TO 20
619         INFR=NF(I+1,J)
620         INFT=NF(I,J+1)
621         INFL=NF(I-1,J)
622         INFB=NF(I,J-1)
623         INF=MAX0(INFR,INFT,INFL,INFB)
624      20 VOL(INF)=VOL(INF)+VOLA
625      30 CONTINUE
626         RETURN
627         END
```

```
      SUBROUTINE DELTADJ
*CALL,COMMON1
C
C     *** DELT (TIME STEP) ADJUSTMENT
C
      DELTN=DELT
      IF (FLGC.LT.0.5) GO TO 20
      T=T-DELT
      CYCLE=CYCLE-1
      DELT=0.5*DELT
      DO 10 I=1,IMAX
      DO 10 J=1,JMAX
      P(I,J)=PN(I,J)
      F(I,J)=FN(I,J)
      U(I,J)=0.0
      V(I,J)=0.0
   10 CONTINUE
      NFLGC=NFLGC+1
   20 CONTINUE
      IF (AUTOT.LT.0.5.AND.FNOC.LT.0.5) GO TO 40
      DUMX=EM10
      DVMX=EM10
      IF (FNOC.GT.0.5) DELT=0.5*DELT
      DO 30 I=2,IM1
      DO 30 J=2,JM1
      UDM=ABS(UN(I,J))/(XI(I+1)-XI(I))
      VDM=ABS(VN(I,J))/(YJ(J+1)-YJ(J))
      DUMX=AMAX1(DUMX,UDM)
      DVMX=AMAX1(DVMX,VDM)
   30 CONTINUE
      DTMP=1.01
      IF (ITER.GT.25) DTMP=0.99
      DELTO=DELT*DTMP
      CON=0.25
      DELT=AMIN1(DELTO,CON/DUMX,CON/DVMX,DTVIS,DTSFT)
      IF (IMOVY.GT.0) DELT=AMIN1(DELT,PLTDT)
   40 IF (DELT.EQ.DELTN.AND.NMAT.EQ.1) GO TO 60
      CTOS=DELT*RDTEXP
      COMG=AMIN1(CTOS**2,1.0)
      OMG1=(OMG-1.0)*COMG+1.0
      DO 50 I=1,IMAX
      DO 50 J=1,JMAX
      IF (BETA(I,J).LT.0.0) GO TO 50
      RHXR=(RHOFC+RHOD*F(I,J))*DELX(I+1)+(RHOFC+RHOD*F(I+1,J))*DELX(I)
      RHXL=(RHOFC+RHOD*F(I-1,J))*DELX(I)+(RHOFC+RHOD*F(I,J))*DELX(I-1)
      RHYT=(RHOFC+RHOD*F(I,J))*DELY(J+1)+(RHOFC+RHOD*F(I,J+1))*DELY(J)
      RHYB=(RHOFC+RHOD*F(I,J-1))*DELY(J)+(RHOFC+RHOD*F(I,J))*DELY(J-1)
      XX=DELT*RDX(I)*(2.0/RHXL+2.0/RHXR)+DELT*RDY(J)*(2.0/RHYT+2.0/RHYB)
      RHOR=RHOF/(RHOFC+RHOD*F(I,J))
      BETA(I,J)=OMG1/(XX*COMG+RCSQ*RHOR/DELT)
   50 CONTINUE
   60 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE DRAW
*CALL,COMMON1
C
C     *** PLOT VELOCITY VECTOR, FREE SURFACE, MESH,
C     *** AND MARKER PARTICLE ON FILM
C
      IF (IMOVY.EQ.1.AND.NP.GT.0) GO TO 80
C
C     *** VELOCITY VECTOR PLOT
C
      CALL ADV (1)
      CALL FRAME (XMIN,XMAX,YMAX,YMIN)
      IF (IMOVY.EQ.1) GO TO 10
      CALL LINCNT (62)
      WRITE (12,140) NAME
      CALL LINCNT (64)
      WRITE (12,130) T,CYCLE
   10 CONTINUE
      CALL DRWOBS
      DO 20 J=2,JM1
      DO 20 I=2,IM1
      IF (F(I,J).LT.0.5.AND.NMAT.EQ.1) GO TO 20
      IF (BETA(I,J).LT.0.0) GO TO 20
      XCC=XI(I)
      YCC=0.5*(Y(J)+Y(J-1))
      UVEC=(U(I-1,J)+U(I,J))*0.5*VELMX1+XCC
      VVEC=(V(I,J-1)+V(I,J))*0.5*VELMX1+YCC
      CALL DRWVEC (XCC,YCC,UVEC,VVEC,1)
      CALL PLTPT (XCC,YCC,538,1)
   20 CONTINUE
C
C     *** DRAW FREE SURFACE
C
      FPL=0.5
      DO 40 I=2,IM1
      DO 40 J=2,JM1
      IF (BETA(I,J).LT.0.0) GO TO 40
      FATR=0.25*(F(I,J)+F(I+1,J)+F(I,J+1)+F(I+1,J+1))
      FXTR=0.5*(F(I+1,J+1)+F(I+1,J)-F(I,J+1)-F(I,J))/(XI(I+1)-XI(I))
      FYTR=0.5*(F(I,J+1)+F(I+1,J+1)-F(I,J)-F(I+1,J))/(YJ(J+1)-YJ(J))
      FTRS=FXTR**2+FYTR**2
      IF (FTRS.EQ.0.0) FTRS=EP10
      XTR=0.5*(XI(I+1)+XI(I))+(FPL-FATR)*FXTR/FTRS
      XTR=AMAX1(XTR,XI(I))
      XTR=AMIN1(XTR,XI(I+1))
      XTRM=-XTR
      YTR=0.5*(YJ(J)+YJ(J+1))+(FPL-FATR)*FYTR/FTRS
      YTR=AMAX1(YTR,YJ(J))
      YTR=AMIN1(YTR,YJ(J+1))
      IF (F(I,J).GT.0.5.AND.F(I+1,J).GT.0.5) GO TO 30
      IF (F(I,J).LT.0.5.AND.F(I+1,J).LT.0.5) GO TO 30
      FABR=0.25*(F(I,J)+F(I+1,J)+F(I,J-1)+F(I+1,J-1))
      FXBR=0.5*(F(I+1,J)+F(I+1,J-1)-F(I,J)-F(I,J-1))/(XI(I+1)-XI(I))
      FYBR=0.5*(F(I,J)+F(I+1,J)-F(I,J-1)-F(I+1,J-1))/(YJ(J)-YJ(J-1))
      FBRS=FXBR**2+FYBR**2
      IF (FBRS.EQ.0.0) FBRS=EP10
      XBR=0.5*(XI(I+1)+XI(I))+(FPL-FABR)*FXBR/FBRS
      XBR=AMAX1(XBR,XI(I))
```

```
      XBR=AMIN1(XBR,XI(I+1))
      YBR=0.5*(YJ(J)+YJ(J-1))+(FPL-FABR)*FYBR/FBRS
      YBR=AMAX1(YBR,YJ(J-1))
      YBR=AMIN1(YBR,YJ(J))
      CALL DRWVEC (XBR,YBR,XTR,YTR,1)
   30 CONTINUE
      IF (F(I,J).GT.0.5.AND.F(I,J+1).GT.0.5) GO TO 40
      IF (F(I,J).LT.0.5.AND.F(I,J+1).LT.0.5) GO TO 40
      FATL=0.25*(F(I,J)+F(I,J+1)+F(I-1,J)+F(I-1,J+1))
      FXTL=0.5*(F(I,J+1)+F(I,J)-F(I-1,J+1)-F(I-1,J))/(XI(I)-XI(I-1))
      FYTL=0.5*(F(I-1,J+1)+F(I,J+1)-F(I-1,J)-F(I,J))/(YJ(J+1)-YJ(J))
      FTLS=FXTL**2+FYTL**2
      IF (FTLS.EQ.0.0) FTLS=EP10
      XTL=0.5*(XI(I-1)+XI(I))+(FPL-FATL)*FXTL/FTLS
      XTL=AMAX1(XTL,XI(I-1))
      XTL=AMIN1(XTL,XI(I))
      YTL=0.5*(YJ(J)+YJ(J+1))+(FPL-FATL)*FYTL/FTLS
      YTL=AMAX1(YTL,YJ(J))
      YTL=AMIN1(YTL,YJ(J+1))
      CALL DRWVEC (XTL,YTL,XTR,YTR,1)
   40 CONTINUE
C
C     *** MESH PLOT
C
      IF (IMOVY.EQ.1) GO TO 120
      IF (T.GT.0.0) GO TO 70
      CALL ADV (1)
      CALL DRWOBS
      DO 50 J=1,JM1
      YCC=Y(J)
      CALL DRWVEC (XMIN,YCC,XMAX,YCC,0)
   50 CONTINUE
      DO 60 I=1,IM1
      XCC=X(I)
      CALL DRWVEC (XCC,YMIN,XCC,YMAX,1)
   60 CONTINUE
   70 CONTINUE
C
C     *** PLOT PARTICLES
C
      IF (NP.EQ.0) GO TO 110
   80 CONTINUE
      CALL ADV (1)
      CALL DRWOBS
      CALL FRAME (XMIN,XMAX,YMAX,YMIN)
      IF (IMOVY.EQ.1) GO TO 90
      CALL LINCNT (62)
      WRITE (12,140) NAME
      CALL LINCNT (64)
      WRITE (12,130) T,CYCLE
   90 CONTINUE
      DO 100 N=1,NP
      CALL PLTPT (XP(N),YP(N),3008,1)
  100 CONTINUE
  110 CONTINUE
  120 CONTINUE
      RETURN
C
```

```
798     130 FORMAT (1H+,80X,2HT=,1PE10.3,4X,6HCYCLE=,I4)
799     140 FORMAT (1H ,18X,10A8,1X,A10,2(1X,A8))
800         END
```

```
801         SUBROUTINE DRWOBS
802   *CALL,COMMON1
803   C
804   C       *** DRAW AROUND ALL OBSTACLES
805   C
806         DO 20 I=2,IM1
807         DO 20 J=2,JM1
808         IF ((BETA(I,J).LT.0.0.AND.BETA(I+1,J).LT.0.0).OR.(BETA(I,J).GT.0.0
809        1 .AND.BETA(I+1,J).GT.0.0)) GO TO 10
810         XONE=X(I)
811         XTWO=XONE
812         YONE=Y(J-1)
813         YTWO=Y(J)
814         CALL DRWVEC (XONE,YONE,XTWO,YTWO,1)
815       10 IF ((BETA(I,J).LT.0.0.AND.BETA(I,J+1).LT.0.0).OR.(BETA(I,J).GT.0.0
816        1 .AND.BETA(I,J+1).GT.0.0)) GO TO 20
817         XONE=X(I-1)
818         XTWO=X(I)
819         YONE=Y(J)
820         YTWO=YONE
821         CALL DRWVEC (XONE,YONE,XTWO,YTWO,1)
822       20 CONTINUE
823         RETURN
824         END
```

```
825         SUBROUTINE DRWVEC (XONE,YONE,XTWO,YTWO,ISYM)
826   *CALL,COMMON1
827   C
828   C       *** DRAW A VECTOR
829   C       *** PROVIDES A SYSTEM DEPENDANT CALL
830   C
831         IC=0
832         X1=XONE
833         Y1=YONE
834         X2=XTWO
835         Y2=YTWO
836      10 X01=(X1-XMIN)*SF+XSHFT
837         Y01=(Y1-YMIN)*SF+YSHFT
838         X02=(X2-XMIN)*SF+XSHFT
839         Y02=(Y2-YMIN)*SF+YSHFT
840         IX1=50.+920.0*X01
841         IX2=50.+920.0*X02
842         IY1=50.+920.0*(1.0-Y01)
843         IY2=50.+920.0*(1.0-Y02)
844         CALL DRV (IX1,IY1,IX2,IY2)
845         IF (ISYMPLT.EQ.0.OR.ISYM.EQ.0) GO TO 20
846         IC=IC+1
847         IF (IC.GT.1) GO TO 20
848         X1=-X1
849         X2=-X2
850         GO TO 10
851      20 RETURN
852         END
```

```
      SUBROUTINE FILMSET
C
C     *** SETUP OF OUTPUT FILM TYPE
C     *** THIS ROUTINE IS SYSTEM DEPENDANT
C
      CALL GFR80 (1HU,7HSOLAVOF,60,2H16,5HT3RSH,1HN)
C
      CALL GRPHLUN (12)
      CALL LIB4020
      CALL GRPHFTN
      CALL SETFLSH
      RETURN
      END
```

```
      SUBROUTINE FRAME (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
```

```
      SUBROUTINE LAVORE
*CALL,COMMON1
C
C     *** LABEL VOID REGIONS - - VOID REGIONS ARE NF.EQ.6 AND ABOVE
C
      NNR=6
      NVR=6
      DO 30 J=2,JM1
      DO 30 I=2,IM1
      IF (NF(I,J).LT.6) GO TO 30
      INFB=NF(I,J-1)
      INFL=NF(I-1,J)
      IF (INFB.LT.6.AND.INFL.LT.6) GO TO 20
      IF (INFB.LT.6.OR.INFL.LT.6) GO TO 10
      NF(I,J)=MIN0(INFB,INFL)
      INRB=NR(INFB)
      INRL=NR(INFL)
      INRMN=MIN0(INRB,INRL)
      NR(INFB)=INRMN
      NR(INFL)=INRMN
      GO TO 30
   10 NF(I,J)=INFB
      IF (INFB.LT.6) NF(I,J)=INFL
      GO TO 30
   20 NF(I,J)=NVR
      NR(NVR)=NNR
      NVR=NVR+1
      NNR=NNR+1
   30 CONTINUE
C
C     *** REDEFINE REGION NUMBERS TO BE CONSECUTIVE
C
      NVR1=NVR-1
      NNR1=NNR-1
      KKN=7
      DO 50 KK=7,NNR1
      KFLG=0
      DO 40 K=7,NVR1
      IF (NR(K).NE.KK) GO TO 40
      NR(K)=KKN
      KFLG=1
   40 CONTINUE
      IF (KFLG.EQ.1) KKN=KKN+1
   50 CONTINUE
      NREG=KKN-6
C
C     *** REDEFINE VOID NUMBERS TO BE CONSECUTIVE IF NREG.GT.1
C
      DO 60 J=2,JM1
      DO 60 I=2,IM1
      INF=NF(I,J)
      IF (INF.LT.6) GO TO 60
      NF(I,J)=NR(INF)
   60 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE MESHSET
*CALL,COMMON1
C
C     *** MESH SETUP  (GENERATION)
C
      I=1
      J=1
      X(1)=XL(1)
      Y(1)=YL(1)
      DO 30 K=1,NKX
      DXML=(XC(K)-XL(K))/NXL(K)
      DXMR=(XL(K+1)-XC(K))/NXR(K)
      DXMN1=DXMN(K)
      NT=NXL(K)
      TN=NT
      TN=AMAX1(TN,1.0+EM6)
      DXMN(K)=AMIN1(DXMN1,DXML)
      CMC=(XC(K)-XL(K)-TN*DXMN(K))*TN/(TN-1.0)
      IF (NT.EQ.1) CMC=0.0
      BMC=XC(K)-XL(K)-CMC
      DO 10 L=1,NT
      I=I+1
      RLN=(FLOAT(L)-TN)/TN
   10 X(I)=XC(K)+BMC*RLN-CMC*RLN*RLN
      NT=NXR(K)
      TN=NT
      TN=AMAX1(TN,1.0+EM6)
      DXMN(K)=AMIN1(DXMN1,DXMR)
      CMC=(XL(K+1)-XC(K)-TN*DXMN(K))*TN/(TN-1.0)
      IF (NT.EQ.1) CMC=0.0
      BMC=XL(K+1)-XC(K)-CMC
      DO 20 L=1,NT
      I=I+1
      RLN=FLOAT(L)/TN
   20 X(I)=XC(K)+BMC*RLN+CMC*RLN*RLN
   30 CONTINUE
      IF (WR.NE.4) GO TO 40
      I=I+1
      X(I)=X(I-1)+X(2)-X(1)
   40 CONTINUE
      DO 70 K=1,NKY
      DYML=(YC(K)-YL(K))/NYL(K)
      DYMR=(YL(K+1)-YC(K))/NYR(K)
      DYMN1=DYMN(K)
      NT=NYL(K)
      TN=NT
      TN=AMAX1(TN,1.0+EM6)
      DYMN(K)=AMIN1(DYMN1,DYML)
      CMC=(YC(K)-YL(K)-TN*DYMN(K))*TN/(TN-1.0)
      IF (NT.EQ.1) CMC=0.0
      BMC=YC(K)-YL(K)-CMC
      DO 50 L=1,NT
      J=J+1
      RLN=(FLOAT(L)-TN)/TN
   50 Y(J)=YC(K)+BMC*RLN-CMC*RLN*RLN
      NT=NYR(K)
      TN=NT
      TN=AMAX1(TN,1.0+EM6)
```

```
      DYMN(K)=AMIN1(DYMN1,DYMR)
      CMC=(YL(K+1)-YC(K)-TN*DYMN(K))*TN/(TN-1.0)
      IF (NT.EQ.1) CMC=0.0
      BMC=YL(K+1)-YC(K)-CMC
      DO 60 L=1,NT
      J=J+1
      RLN=FLOAT(L)/TN
   60 Y(J)=YC(K)+BMC*RLN+CMC*RLN*RLN
   70 CONTINUE
      IF (WT.NE.4) GO TO 80
      J=J+1
      Y(J)=Y(J-1)+Y(2)-Y(1)
   80 CONTINUE
      NUMX=I
      NUMY=J
      NUMXM1=NUMX-1
      NUMYM1=NUMY-1
      NUMXP1=NUMX+1
      NUMYP1=NUMY+1
      IBAR=NUMX-1
      JBAR=NUMY-1
      IMAX=IBAR+2
      JMAX=JBAR+2
      IM1=IMAX-1
      JM1=JMAX-1
      IM2=IMAX-2
      JM2=JMAX-2
C
C     *** CALCULATE VALUES NEEDED FOR VARIABLE MESH
C
      DO 100 I=1,NUMX
      IF (X(I).EQ.0.0) GO TO 90
      RX(I)=1.0/X(I)
      GO TO 100
   90 RX(I)=0.0
  100 CONTINUE
      DO 110 I=2,NUMX
      XI(I)=0.5*(X(I-1)+X(I))
      DELX(I)=X(I)-X(I-1)
      RXI(I)=1.0/XI(I)
  110 RDX(I)=1.0/DELX(I)
      DELX(1)=DELX(2)
      XI(1)=XI(2)-DELX(2)
      RXI(1)=1.0/XI(1)
      RDX(1)=1.0/DELX(1)
      DELXA=DELX(NUMX)
      IF (WR.EQ.4) DELXA=DELX(3)
      DELX(NUMXP1)=DELXA
      XI(NUMXP1)=XI(NUMX)+DELXA
      X(NUMXP1)=XI(NUMXP1)+0.5*DELX(NUMXP1)
      RXI(NUMXP1)=1.0/XI(NUMXP1)
      RDX(NUMXP1)=1.0/DELX(NUMXP1)
      DO 120 I=2,NUMY
      YJ(I)=0.5*(Y(I-1)+Y(I))
      RYJ(I)=1.0/YJ(I)
      DELY(I)=Y(I)-Y(I-1)
      RDY(I)=1.0/DELY(I)
  120 CONTINUE
```

```
      DELY(1)=DELY(2)
      RDY(1)=1.0/DELY(1)
      YJ(1)=YJ(2)-DELY(2)
      RYJ(1)=1.0/YJ(1)
      DELYA=DELY(NUMY)
      IF (WT.EQ.4) DELYA=DELY(3)
      DELY(NUMYP1)=DELYA
      YJ(NUMYP1)=YJ(NUMY)+DELYA
      Y(NUMYP1)=YJ(NUMYP1)+0.5*DELY(NUMYP1)
      RYJ(NUMYP1)=1.0/YJ(NUMYP1)
      RDY(NUMYP1)=1.0/DELY(NUMYP1)
      WRITE (6,190)
      DO 130 I=1,NUMXP1
      WRITE (6,200) I,X(I),I,RX(I),I,DELX(I),I,RDX(I),I,XI(I),I,RXI(I)
  130 CONTINUE
      WRITE (6,190)
      DO 140 I=1,NUMYP1
      WRITE (6,210) I,Y(I),I,DELY(I),I,RDY(I),I,YJ(I),I,RYJ(I)
  140 CONTINUE
      IF (IMOVY.EQ.1) GO TO 170
      WRITE (12,190)
      DO 150 I=1,NUMXP1
      WRITE (12,200) I,X(I),I,RX(I),I,DELX(I),I,RDX(I),I,XI(I),I,RXI(I)
  150 CONTINUE
      WRITE (12,190)
      DO 160 I=1,NUMYP1
      WRITE (12,210) I,Y(I),I,DELY(I),I,RDY(I),I,YJ(I),I,RYJ(I)
  160 CONTINUE
  170 CONTINUE
C
C     *** TEST ARRAY SIZE
C
      IF (IMAX.LE.IBAR2.AND.JMAX.LE.JBAR2) GO TO 180
      WRITE (6,220)
C
      CALL EXIT
C
  180 CONTINUE
      RETURN
C
  190 FORMAT (1H1)
  200 FORMAT (1X,2HX(,I2,2H)=,1PE12.5,2X,3HRX(,I2,2H)=,1PE12.5,2X,5HDELX
     1(,I2,2H)=,1PE12.5,1X,4HRDX(,I2,2H)=,1PE12.5,2X,3HXI(,I2,2H)=,1PE12
     2 .5,2X,4HRXI(,I2,2H)=,1PE12.5)
  210 FORMAT (1X,2HY(,I2,2H)=,1PE12.5,3X,5HDELY(,I2,2H)=,1PE12.5,3X,4HRD
     1Y(,I2,2H)=,1PE12.5,3X,3HYJ(,I2,2H)=,1PE12.5,3X,4HRYJ(,I2,2H)=,1PE1
     2 2.5)
  220 FORMAT (41H  MESH SIZE GREATER THAN ARRAY DIMENSIONS)
      END
```

```
      SUBROUTINE PARMOV
*CALL,COMMON1
C
C     *** MARKER PARTICLE MOVEMENT SECTION
C
      NPT=0
      NPN=0
      K=1
      KN=1
      PFLG=1.0
      IPER=IM1
      IF (WR.EQ.4) IPER=IM2
      JPER=JM1
      IF (WT.EQ.4) JPER=JM2
   10 IF (NP.EQ.NPT) GO TO 120
C
C     *** CLACULATE U WEIGHTED VELOCITY OF PARTICLE
C
      I=IP(K)
      J=JP(K)
      IF (YP(K).GT.YJ(J)) GO TO 20
      HPX=X(I)-XP(K)
      HMX=DELX(I)-HPX
      HPY=YJ(J)-YP(K)
      NORMY=(DELY(J)+DELY(J-1))*0.5
      HMY=NORMY-HPY
      UTOP=(U(I-1,J)*HPX+U(I,J)*HMX)*RDX(I)
      UBOT=(U(I-1,J-1)*HPX+U(I,J-1)*HMX)*RDX(I)
      UPART=(UTOP*HMY+UBOT*HPY)/NORMY
      GO TO 30
   20 HPX=X(I)-XP(K)
      HMX=DELX(I)-HPX
      HPY=YJ(J+1)-YP(K)
      NORMY=(DELY(J+1)+DELY(J))*0.5
      HMY=NORMY-HPY
      UTOP=(U(I-1,J+1)*HPX+U(I,J+1)*HMX)*RDX(I)
      UBOT=(U(I-1,J)*HPX+U(I,J)*HMX)*RDX(I)
      UPART=(UTOP*HMY+UBOT*HPY)/NORMY
C
C     *** CALCULATE V WEIGHTED VELOCITY OF PARTICLE
C
   30 IF (XP(K).GT.XI(I)) GO TO 40
      NORMX=(DELX(I)+DELX(I-1))*0.5
      RNORMX=1.0/NORMX
      HPX=XI(I)-XP(K)
      HMX=NORMX-HPX
      HPY=Y(J)-YP(K)
      HMY=DELY(J)-HPY
      VTOP=(V(I-1,J)*HPX+V(I,J)*HMX)*RNORMX
      VBOT=(V(I-1,J-1)*HPX+V(I,J-1)*HMX)*RNORMX
      VPART=(VTOP*HMY+VBOT*HPY)*RDY(J)
      GO TO 50
   40 NORMX=(DELX(I)+DELX(I+1))*0.5
      RNORMX=1.0/NORMX
      HPX=XI(I+1)-XP(K)
      HMX=NORMX-HPX
      HPY=Y(J)-YP(K)
      HMY=DELY(J)-HPY
```

```
1155         VTOP=(V(I,J)*HPX+V(I+1,J)*HMX)*RNORMX
1156         VBOT=(V(I,J-1)*HPX+V(I+1,J-1)*HMX)*RNORMX
1157         VPART=(VTOP*HMY+VBOT*HPY)*RDY(J)
1158      50 XPART=XP(K)+UPART*DELT
1159         YPART=YP(K)+VPART*DELT
1160         IF (XPART.GT.X(I)) IP(KN)=IP(K)+1
1161         IF (XPART.LT.X(I-1)) IP(KN)=IP(K)-1
1162         IF (YPART.GT.Y(J)) JP(KN)=JP(K)+1
1163         IF (YPART.LT.Y(J-1)) JP(KN)=JP(K)-1
1164         XP(KN)=XPART
1165         YP(KN)=YPART
1166         IF (XP(KN).LT.X(1)) GO TO 60
1167         IF (YP(KN).LT.Y(1)) GO TO 70
1168         IF (XP(KN).GT.X(IPER)) GO TO 80
1169         IF (YP(KN).GT.Y(JPER)) GO TO 90
1170         GO TO 100
1171      60 IF (NL.LE.2) GO TO 100
1172         IF (NL.NE.4) GO TO 110
1173         XP(KN)=XP(KN)+X(IM2)-X(1)
1174         IP(KN)=IP(KN)+IM2-1
1175         GO TO 100
1176      70 IF (NB.LE.2) GO TO 100
1177         IF (NB.NE.4) GO TO 110
1178         YP(KN)=YP(KN)+Y(JM2)-Y(1)
1179         JP(KN)=JP(KN)+JM2-1
1180         GO TO 100
1181      80 IF (NR.LE.2) GO TO 100
1182         IF (NR.NE.4) GO TO 110
1183         XP(KN)=XP(KN)-X(IM2)+X(1)
1184         IP(KN)=IP(KN)-IM2+1
1185         GO TO 100
1186      90 IF (NT.LE.2) GO TO 100
1187         IF (NT.NE.4) GO TO 110
1188         YP(KN)=YP(KN)-Y(JM2)+Y(1)
1189         JP(KN)=JP(KN)-JM2+1
1190     100 KN=KN+1
1191         NPN=NPN+1
1192     110 K=K+1
1193         NPT=NPT+1
1194         PFLG=1.0
1195         GO TO 10
1196     120 NP=NPN
1197         RETURN
1198         END
```

```
      SUBROUTINE PETACAL
*CALL,COMMON1
C
C     *** DETERMINE THE PRESSURE INTERPOLATION FACTOR PETA
C     *** DETERMINE THE SURFACE TENSION PRESSURE AND
C     *** WALL ADHESION EFFECTS IN SURFACE CELLS
C
      DO 10 I=1,IMAX
      DO 10 J=1,JMAX
      NF(I,J)=0
      PS(I,J)=0.0
   10 PETA(I,J)=1.0
      IPASS=0
      DO 140 I=2,IM1
      DO 140 J=2,JM1
      TANTH(I,J)=EP10
      IF (BETA(I,J).LT.0.0) GO TO 140
      IF (F(I,J).LT.EMF) NF(I,J)=6
      IF (F(I,J).LT.EMF.OR.F(I,J).GT.EMF1) GO TO 140
      IF (F(I+1,J).LT.EMF) GO TO 20
      IF (F(I,J+1).LT.EMF) GO TO 20
      IF (F(I-1,J).LT.EMF) GO TO 20
      IF (F(I,J-1).LT.EMF) GO TO 20
      GO TO 140
   20 CONTINUE
C
C     *** CALCULATE THE PARTIAL DERIVATIVES OF F
C
      DXR=0.5*(DELX(I)+DELX(I+1))
      DXL=0.5*(DELX(I)+DELX(I-1))
      DYT=0.5*(DELY(J)+DELY(J+1))
      DYB=0.5*(DELY(J)+DELY(J-1))
      RXDEN=1.0/(DXR*DXL*(DXR+DXL))
      RYDEN=1.0/(DYT*DYB*(DYT+DYB))
      FL=F(I-1,J+1)
      IF (BETA(I-1,J+1).LT.0.0.OR.(I.EQ.2.AND.WL.LT.3)) FL=1.0
      FC=F(I,J+1)
      IF (BETA(I,J+1).LT.0.0) FC=1.0
      FR=F(I+1,J+1)
      IF (BETA(I+1,J+1).LT.0.0.OR.(I.EQ.IM1.AND.WR.LT.3)) FR=1.0
      AVFT=FL*DELX(I-1)+FC*DELX(I)+FR*DELX(I+1)
      FL=F(I-1,J-1)
      IF (BETA(I-1,J-1).LT.0.0.OR.(I.EQ.2.AND.WL.LT.3)) FL=1.0
      FC=F(I,J-1)
      IF (BETA(I,J-1).LT.0.0) FC=1.0
      FR=F(I+1,J-1)
      IF (BETA(I+1,J-1).LT.0.0.OR.(I.EQ.IM1.AND.WR.LT.3)) FR=1.0
      AVFB=FL*DELX(I-1)+FC*DELX(I)+FR*DELX(I+1)
      FL=F(I-1,J)
      IF (BETA(I-1,J).LT.0.0.OR.(I.EQ.2.AND.WL.LT.3)) FL=1.0
      FR=F(I+1,J)
      IF (BETA(I+1,J).LT.0.0.OR.(I.EQ.IM1.AND.WR.LT.3)) FR=1.0
      AVFCY=FL*DELX(I-1)+F(I,J)*DELX(I)+FR*DELX(I+1)
      FB=F(I,J-1)
      IF (BETA(I,J-1).LT.0.0.OR.(J.EQ.2.AND.WB.LT.3)) FB=1.0
      FT=F(I,J+1)
      IF (BETA(I,J+1).LT.0.0.OR.(J.EQ.JM1.AND.WT.LT.3)) FT=1.0
      AVFCX=FB*DELY(J-1)+F(I,J)*DELY(J)+FT*DELY(J+1)
```

```
      FB=F(I-1,J-1)
      IF (BETA(I-1,J-1).LT.0.0.OR.(J.EQ.2.AND.WB.LT.3)) FB=1.0
      FC=F(I-1,J)
      IF (BETA(I-1,J).LT.0.0) FC=1.0
      FT=F(I-1,J+1)
      IF (BETA(I-1,J+1).LT.0.0.OR.(J.EQ.JM1.AND.WT.LT.3)) FT=1.0
      AVFL=FB*DELY(J-1)+FC*DELY(J)+FT*DELY(J+1)
      FB=F(I+1,J-1)
      IF (BETA(I+1,J-1).LT.0.0.OR.(J.EQ.2.AND.WB.LT.3)) FB=1.0
      FC=F(I+1,J)
      IF (BETA(I+1,J).LT.0.0) FC=1.0
      FT=F(I+1,J+1)
      IF (BETA(I+1,J+1).LT.0.0.OR.(J.EQ.JM1.AND.WT.LT.3)) FT=1.0
      AVFR=FB*DELY(J-1)+FC*DELY(J)+FT*DELY(J+1)
C
C     *** BOUNDARY CONDITIONS FOR WALL ADHESION
C
      IF (ISURF10.EQ.0.OR.CANGLE.EQ.0.0) GO TO 60
      IF (BETA(I+1,J).GE.0.0.AND.I.NE.IM1) GO TO 30
      AVFR=AVFCX+0.5*(DELX(I)+DELX(I+1))/TANCA
      IF (F(I,J+1).LT.EMF.AND.F(I,J-1).GE.EMF) AVFT=AVFCY-0.5*(DELY(J)
     1 +DELY(J+1))*TANCA
      IF (F(I,J-1).LT.EMF.AND.F(I,J+1).GE.EMF) AVFB=AVFCY-0.5*(DELY(J)
     1 +DELY(J-1))*TANCA
   30 IF (BETA(I,J+1).GE.0.0.AND.J.NE.JM1) GO TO 40
      AVFT=AVFCY+0.5*(DELY(J)+DELY(J+1))/TANCA
      IF (F(I+1,J).LT.EMF.AND.F(I-1,J).GE.EMF) AVFR=AVFCX-0.5*(DELX(I)
     1 +DELX(I+1))*TANCA
      IF (F(I-1,J).LT.EMF.AND.F(I+1,J).GE.EMF) AVFL=AVFCX-0.5*(DELX(I)
     1 +DELX(I-1))*TANCA
   40 IF (BETA(I,J-1).GE.0.0.AND.J.NE.2) GO TO 50
      AVFB=AVFCY+0.5*(DELY(J)+DELY(J-1))/TANCA
      IF (F(I+1,J).LT.EMF.AND.F(I-1,J).GE.EMF) AVFR=AVFCX-0.5*(DELX(I)
     1 +DELX(I+1))*TANCA
      IF (F(I-1,J).LT.EMF.AND.F(I+1,J).GE.EMF) AVFL=AVFCX-0.5*(DELX(I)
     1 +DELX(I-1))*TANCA
   50 IF (BETA(I-1,J).GE.0.0.AND.I.NE.2) GO TO 60
      IF (CYL.GT.0.5.AND.X(1).EQ.0.0) GO TO 60
      AVFL=AVFCX+0.5*(DELX(I)+DELX(I-1))/TANCA
      IF (F(I,J+1).LT.EMF.AND.F(I,J-1).GE.EMF) AVFT=AVFCY-0.5*(DELY(J)
     1 +DELY(J+1))*TANCA
      IF (F(I,J-1).LT.EMF.AND.F(I,J+1).GE.EMF) AVFB=AVFCY-0.5*(DELY(J)
     1 +DELY(J-1))*TANCA
   60 CONTINUE
      XTHM=3.0*AMAX1(AVFT,AVFCY,AVFB)/(DELX(I-1)+DELX(I)+DELX(I+1))
      YTHM=3.0*AMAX1(AVFL,AVFCX,AVFR)/(DELY(J-1)+DELY(J)+DELY(J+1))
      PFX=RXDEN*((AVFR-AVFCX)*DXL**2+(AVFCX-AVFL)*DXR**2)
      PFY=RYDEN*((AVFT-AVFCY)*DYB**2+(AVFCY-AVFB)*DYT**2)
      PF=PFX**2+PFY**2
      IF (PF.GT.EM10) GO TO 70
      NF(I,J)=5
      P(I,J)=0.25*(P(I+1,J)+P(I,J+1)+P(I-1,J)+P(I,J-1))
      GO TO 140
   70 CONTINUE
C
C     *** DETERMINE THE PRESSURE INTERPOLATION CELL NF
C
      ABPFX=ABS(PFX)
```

```
      ABPFY=ABS(PFY)
      L=I
      M=J
      IF (ABPFY.GE.ABPFX) GO TO 80
      DXDYR=DELY(J)*RDX(I)
      PFMN=PFY
      NF(I,J)=2
      L=I+1
      DMX=DELX(I)
      DMIN=0.5*(DMX+DELX(I+1))
      IF (PFX.GT.0.0) GO TO 90
      NF(I,J)=1
      PFMN=-PFY
      L=I-1
      DMX=DELX(I)
      DMIN=0.5*(DMX+DELX(I-1))
      GO TO 90
   80 CONTINUE
      DXDYR=DELX(I)*RDY(J)
      PFMN=-PFX
      NF(I,J)=4
      M=J+1
      DMX=DELY(J)
      DMIN=0.5*(DMX+DELY(J+1))
      IF (PFY.GT.0.0) GO TO 90
      NF(I,J)=3
      PFMN=PFX
      M=J-1
      DMX=DELY(J)
      DMIN=0.5*(DMX+DELY(J-1))
   90 CONTINUE
      TANTH(I,J)=PFMN
      ABTAN=ABS(TANTH(I,J))
C
C     *** DETERMINE THE CURVATURE AND SURFACE PRESSURE
C
      DFS=(0.5-F(I,J))*DMX
      IF (F(I,J).LT.0.5*ABTAN*DXDYR) DFS=0.5*DMX*(1.0+DXDYR*ABTAN-SQRT(8
     1 .0*F(I,J)*DXDYR*ABTAN))
      IF (ISURF10.LT.1) GO TO 130
      NFC=NF(I,J)
      PXR=(AVFR-AVFCX)/DXR
      PXL=(AVFCX-AVFL)/DXL
      PYT=(AVFT-AVFCY)/DYT
      PYB=(AVFCY-AVFB)/DYB
      YDFS=-DFS
      IF (NFC.EQ.2.OR.NFC.EQ.4) YDFS=DFS
      IF (NFC.GT.2) GO TO 100
      DXDN=DELY(J)
      XINB=YDFS+0.5*TANTH(I,J)*DXDN
      XINT=2.0*YDFS-XINB
      GP1=PYT
      PX1=PXL
      IF (XINT.GT.0.0) PX1=PXR
      IF (ABS(PX1).LT.ABS(GP1)) GP1=SIGN(1.0,GP1)/(ABS(PX1)+EM10)
      GP2=PYB
      PX2=PXR
      IF (XINB.LT.0.0) PX2=PXL
```

```
      IF (ABS(PX2).LT.ABS(GP2)) GP2=SIGN(1.0,GP2)/(ABS(PX2)+EM10)
      GO TO 110
  100 DXDN=DELX(I)
      YINR=YDFS+0.5*TANTH(I,J)*DXDN
      YINL=2.0*YDFS-YINR
      GP1=PXR
      PY1=PYT
      IF (YINR.LT.0.0) PY1=PYB
      IF (ABS(PY1).LT.ABS(GP1)) GP1=SIGN(1.0,GP1)/(ABS(PY1)+EM10)
      GP2=PXL
      PY2=PYB
      IF (YINL.GT.0.0) PY2=PYT
      IF (ABS(PY2).LT.ABS(GP2)) GP2=SIGN(1.0,GP2)/(ABS(PY2)+EM10)
  110 GP1D=1.0+GP1*GP1
      GP2D=1.0+GP2*GP2
      CURVXY=(GP2/SQRT(GP2D)-GP1/SQRT(GP1D))/DXDN
      CURVCYL=0.0
      IF (CYL.LT.1.0) GO TO 120
      XLITLR=XI(I)
      IF (NFC.EQ.1) XLITLR=X(I-1)+F(I,J)*DELX(I)
      IF (NFC.EQ.2) XLITLR=X(I)-F(I,J)*DELX(I)
      RLITLR=AMIN1(1.0/XLITLR,RX1(2))
      TRIG=ABS(SIN(ATAN(ABTAN)))
      IF (NFC.LE.2) TRIG=ABS(COS(ATAN(ABTAN)))
      CURVCYL=-CYL*TRIG*SIGN(1.0,PFX)*RLITLR
  120 CURV=CURVXY+CURVCYL
      PS(I,J)=SIGMA*CURV
      IF (XTHM.LT.1.0.OR.YTHM.LT.1.0) PS(I,J)=0.0
  130 CONTINUE
C
C     *** CALCULATE PETA
C
      NFSB=0
      IF (F(I+1,J).LT.EMF.OR.I.EQ.IM1.OR.BETA(I+1,J).LT.0.0) NFSB=NFSB+1
      IF (F(I,J+1).LT.EMF.OR.BETA(I,J+1).LT.0.0) NFSB=NFSB+2
      IF (F(I-1,J).LT.EMF.OR.BETA(I-1,J).LT.0.0) NFSB=NFSB+4
      IF (F(I,J-1).LT.EMF.OR.BETA(I,J-1).LT.0.0) NFSB=NFSB+8
      IF (NFSB.EQ.15) PS(I,J)=0.0
      IF (NMAT.EQ.2) GO TO 140
      PETA(I,J)=1.0/(1.0-DFS/DMIN)
      IF (L.EQ.1.OR.L.EQ.IMAX) PETA(I,J)=1.0
      IF (M.EQ.1.OR.M.EQ.JMAX) PETA(I,J)=1.0
      IF (BETA(L,M).LT.0.0) PETA(I,J)=1.0
  140 CONTINUE
C
      CALL LAVORE
C
      CALL CAVOVO
C
C     IF NECESSARY, DETERMINE PRESSURES PR FOR VOID REGIONS NF
C
      IF (NMAT.EQ.2) GO TO 220
C
C     *** SET PETA IN ADJACENT FULL CELL
C
      DO 210 J=1,JMAX
      DO 210 I=1,IMAX
      NFF=NF(I,J)
```

```
1431          IF (NFF.EQ.0.OR.BETA(I,J).LT.0.0) GO TO 210
1432          IF (NFF.GT.5) GO TO 200
1433          L=I
1434          M=J
1435          GO TO (150,160,170,180,210), NFF
1436      150 L=I-1
1437          DMX=DELX(I)
1438          DMIN=0.5*(DMX+DELX(I-1))
1439          GO TO 190
1440      160 L=I+1
1441          DMX=DELX(I)
1442          DMIN=0.5*(DMX+DELX(I+1))
1443          GO TO 190
1444      170 M=J-1
1445          DMX=DELY(J)
1446          DMIN=0.5*(DMX+DELY(J-1))
1447          GO TO 190
1448      180 M=J+1
1449          DMX=DELY(J)
1450          DMIN=0.5*(DMX+DELY(J+1))
1451      190 CONTINUE
1452          IF (NF(L,M).GT.0) GO TO 210
1453          BPD=1.0-BETA(L,M)*(1.0-PETA(I,J))*DELT/(DMIN*DMX)
1454          PETA(L,M)=AMIN1(PETA(L,M),1.0/BPD)
1455          GO TO 210
1456      200 CONTINUE
1457          P(I,J)=PR(NFF)
1458      210 CONTINUE
1459      220 CONTINUE
1460          RETURN
1461          END
```

```
1462        SUBROUTINE PLTPT (XONE,YONE,ICHAR,ISYM)
1463  *CALL,COMMON1
1464  C
1465  C     *** PLOT (DRAW) A POINT
1466  C     *** PROVIDES A SYSTEM DEPENDANT CALL
1467  C
1468        IC=0
1469        X1=XONE
1470        Y1=YONE
1471     10 XO1=(X1-XMIN)*SF+XSHFT
1472        YO1=(Y1-YMIN)*SF+YSHFT
1473        IX1=50.+920.0*XO1
1474        IY1=50.+920.0*(1.0-YO1)
1475        CALL PLT (IX1,IY1,42)
1476        IF (ABS(X1).LE.EMS) GO TO 20
1477        IF (ISYM*PLT.EQ.0.OR.ISYM.EQ.0) GO TO 20
1478        IC=IC+1
1479        IF (IC.GT.1) GO TO 20
1480        X1=-X1
1481        GO TO 10
1482     20 RETURN
1483        END
```

```
      SUBROUTINE PRESSIT
*CALL,COMMON1
C
C     *** PRESSURE ITERATION
C
C     *** TEST FOR CONVERGENCE
C
   10 IF (FLG.EQ.0.) GO TO 140
      ITER=ITER+1
      ITMAX=1000
      IF (ITER.LT.ITMAX) GO TO 20
      FNOC=1.0
      NOCON=NOCON+1
      GO TO 140
   20 FLG=0.0
C
C     *** COMPUTE/UPDATED CELL PRESSURE AND VELOCITIES
C
      DO 130 J=JP8,JPT
      DO 130 I=IPL,IPR
      IF (BETA(I,J).LT.0.0) GO TO 130
      IF (NMAT.EQ.2) GO TO 80
      IF (F(I,J).LT.EMF) GO TO 130
      IF (NF(I,J).EQ.0) GO TO 80
C
C     *** CALCULATE PRESSURE FOR SURFACE CELLS
C
      NFF=NF(I,J)
      L=I
      M=J
      GO TO (30,40,50,60,130), NFF
   30 L=I-1
      GO TO 70
   40 L=I+1
      GO TO 70
   50 M=J-1
      GO TO 70
   60 M=J+1
   70 CONTINUE
      NFEL=NF(I-1,J)
      NFER=NF(I+1,J)
      NFEB=NF(I,J-1)
      NFET=NF(I,J+1)
      NFE=MAX0(NFEL,NFER,NFEB,NFET)
      PSURF=PS(I,J)+PR(NFE)
      PLM=P(L,M)
      IF (NF(L,M).NE.0.AND.BETA(I,J).GT.0.0) PLM=PSURF
      DELP=(1.0-PETA(I,J))*PLM+PETA(I,J)*PSURF-P(I,J)
      GO TO 90
   80 CONTINUE
      DIJ=RDX(I)*(U(I,J)-U(I-1,J))+RDY(J)*(V(I,J)-V(I,J-1))+CYL*0.5*RXI
     1 (I)*(U(I,J)+U(I-1,J))
      RHOR=RHOF/(RHOFC+RHOD*F(I,J))
      DFUN=DIJ+RHOR*RCSQ*(P(I,J)-PN(I,J))/DELT
C
C     *** SET FLAG INDICATING CONVERGENCE
C
      IF (ABS(DFUN).GE.EPSI) FLG=1.0
```

```
1542        DELP=-BETA(I,J)*DFUN*PETA(I,J)
1543     90 CONTINUE
1544        P(I,J)=P(I,J)+DELP
1545        CTOS=DELT*RDTEXP
1546        COMG=AMIN1(CTOS**2,1.0)
1547        DPTC=2.0*DELT*DELP*COMG
1548        IF (BETA(I+1,J).LT.0.0) GO TO 100
1549        RHOXR=(RHOFC+RHOD*F(I,J))*DELX(I+1)+(RHOFC+RHOD*F(I+1,J))*DELX(I)
1550        U(I,J)=U(I,J)+DPTC/RHOXR
1551    100 IF (BETA(I-1,J).LT.0.0) GO TO 110
1552        RHOXL=(RHOFC+RHOD*F(I-1,J))*DELX(I)+(RHOFC+RHOD*F(I,J))*DELX(I-1)
1553        U(I-1,J)=U(I-1,J)-DPTC/RHOXL
1554    110 IF (BETA(I,J+1).LT.0.0) GO TO 120
1555        RHOYT=(RHOFC+RHOD*F(I,J))*DELY(J+1)+(RHOFC+RHOD*F(I,J+1))*DELY(J)
1556        V(I,J)=V(I,J)+DPTC/RHOYT
1557    120 IF (BETA(I,J-1).LT.0.0) GO TO 130
1558        RHOYB=(RHOFC+RHOD*F(I,J-1))*DELY(J)+(RHOFC+RHOD*F(I,J))*DELY(J-1)
1559        V(I,J-1)=V(I,J-1)-DPTC/RHOYB
1560    130 CONTINUE
1561        CALL BC
1562        GO TO 10
1563    140 CONTINUE
1564        RETURN
1565        END
```

```
      SUBROUTINE PRTPLT (N)
*CALL,COMMON1
C
C     *** PRINT AND PLOT
C     *** PROVIDES FORMATTED WRITES TO PAPER AND FILM
C
      GO TO (10,70,90,130), N
C
C     *** PRTPLT (1) WRITE OUT INITIAL DATA AND MESH DATA
C
   10 WRITE (6,170)
      WRITE (6,180) NAME
      WRITE (6,220) IBAR,JBAR,DELT,NU,ICYL,EPSI,GX,GY,UI,VI,VELMX,TWFIN
     1 ,PRTDT,PLTDT,OMG,ALPHA,WL,WR,WT,WB,IMOVY,AUTOT,FLHT,ISYMPLT,SIGMA
     2 ,ISURF10,CANGLE,CSQ,NMAT,RHOF,RHOFC
      IF (IMOVY.GT.0) GO TO 40
      WRITE (12,170)
      WRITE (12,180) NAME
      WRITE (12,220) IBAR,JBAR,DELT,NU,ICYL,EPSI,GX,GY,UI,VI,VELMX,TWFIN
     1 ,PRTDT,PLTDT,OMG,ALPHA,WL,WR,WT,WB,IMOVY,AUTOT,FLHT,ISYMPLT,SIGMA
     2 ,ISURF10,CANGLE,CSQ,NMAT,RHOF,RHOFC
C
C     *** WRITE ON FILM VARIABLE MESH INPUT DATA
C
      WRITE (12,250) NKX
      DO 20 I=1,NKX
      WRITE (12,260) I,XL(I),XC(I),XL(I+1),NXL(I),NXR(I),DXMN(I)
   20 CONTINUE
      WRITE (12,280) NKY
      DO 30 I=1,NKY
      WRITE (12,270) I,YL(I),YC(I),YL(I+1),NYL(I),NYR(I),DYMN(I)
   30 CONTINUE
   40 CONTINUE
C
C     *** PRINT VARIABLE MESH INPUT DATA
C
      WRITE (6,250) NKX
      DO 50 I=1,NKX
      WRITE (6,260) I,XL(I),XC(I),XL(I+1),NXL(I),NXR(I),DXMN(I)
   50 CONTINUE
      WRITE (6,280) NKY
      DO 60 I=1,NKY
      WRITE (6,270) I,YL(I),YC(I),YL(I+1),NYL(I),NYR(I),DYMN(I)
   60 CONTINUE
      GO TO 160
C
C     *** PRTPLT (2)  WRITE TIME STEP, CYCLE INFORMATION
C
   70 CONTINUE
      WRITE (6,210) ITER,T,DELT,CYCLE,VCHGT
      IF (IMOVY.EQ.1) GO TO 80
      IF (T.GT.0.) GO TO 80
      WRITE (12,210) ITER,T,DELT,CYCLE,VCHGT
   80 CONTINUE
      GO TO 160
C
C     *** PRTPLT (3)  WRITE FIELD VARIABLES TO FILM
C
```

```
1624      90 IF (IMOVY.EQ.1) GO TO 120
1625         CALL ADV (1)
1626         WRITE (12,240) NAME
1627         WRITE (12,210) ITER,T,DELT,CYCLE,VCHGT
1628         WRITE (12,230)
1629         WRITE (12,290) NREG
1630         WRITE (12,300)
1631         KNR=NREG+5
1632         DO 100 K=6,KNR
1633         WRITE (12,310) K,VOL(K),PR(K)
1634     100 CONTINUE
1635         WRITE (12,190)
1636         DO 110 I=1,IMAX
1637         DO 110 J=1,JMAX
1638         DIJ=RDX(I)*(U(I,J)-U(I-1,J))+RDY(J)*(V(I,J)-V(I,J-1))+CYL*0.5*RXI
1639        1 (I)*(U(I,J)+U(I-1,J))
1640         WRITE (12,200) I,J,U(I,J),V(I,J),P(I,J),DIJ,PS(I,J),F(I,J),NF(I,J)
1641        1 ,PETA(I,J)
1642     110 CONTINUE
1643     120 CONTINUE
1644         GO TO 160
1645   C
1646   C     *** PRTPLT (4)  WRITE FIELD VARIABLES TO PAPER
1647   C
1648     130 WRITE (6,170)
1649         WRITE (6,240) NAME
1650         WRITE (6,210) ITER,T,DELT,CYCLE,VCHGT
1651         WRITE (6,230)
1652         WRITE (6,290) NREG
1653         WRITE (6,300)
1654         KNR=NREG+5
1655         DO 140 K=6,KNR
1656         WRITE (6,310) K,VOL(K),PR(K)
1657     140 CONTINUE
1658         WRITE (6,230)
1659         WRITE (6,190)
1660         DO 150 I=1,IMAX
1661         DO 150 J=1,JMAX
1662         DIJ=RDX(I)*(U(I,J)-U(I-1,J))+RDY(J)*(V(I,J)-V(I,J-1))+CYL*0.5*RXI
1663        1 (I)*(U(I,J)+U(I-1,J))
1664         WRITE (6,200) I,J,U(I,J),V(I,J),P(I,J),DIJ,PS(I,J),F(I,J),NF(I,J)
1665        1 ,PETA(I,J)
1666     150 CONTINUE
1667     160 RETURN
1668   C
1669     170 FORMAT (1H1)
1670     180 FORMAT (10A8)
1671     190 FORMAT (4X,1HI,5X,1HJ,9X,1HU,14X,1HV,15X,1HP,15X,1HD,12X,2HPS,13X,
1672        1 1HF,11X,2HNF,9X,4HPETA)
1673     200 FORMAT (2X,I3,3X,I3,6(3X,1PE12.5),3X,I3,3X,E12.5)
1674     210 FORMAT (6X,6HITER= ,I5,5X,6HTIME= ,1PE12.5,5X,6HDELT= ,1PE12.5,5X,
1675        1 7HCYCLE= ,I4,5X,7HVCHGT= ,1PE12.5)
1676     220 FORMAT (1H ,5X,6HIBAR= ,I3/6X,6HJBAR= ,I3/6X,6HDELT= ,1PE12.5/8X,4
1677        1 HNU= ,E12.5/6X,6HICYL= ,I2/6X,6HEPSI= ,E12.5/8X,4HGX= ,E12.5/8X,4
1678        2 HGY= ,E12.5/8X,4HUI= ,E12.5/8X,4HVI= ,E12.5/5X,7HVELMX= ,E12.5/5X
1679        3 ,7HTWFIN= ,E12.5/5X,7HPRTDT= ,E12.5/5X,7HPLTDT= ,E12.5/7X,5HOMG=
1680        4 ,E12.5/5X,7HALPHA= ,E12.5/8X,4HWL= ,I2/8X,4HWR= ,I2/8X,4HWT= ,I2/
1681        5 8X,4HWB= ,I2/5X,7HIMOVY= ,E12.5/5X,7HAUTOT= ,E12.5/6X,6HFLHT=
```

```
1682        6 ,E12.5/3X,9HISYMPLT= ,I2/5X,7HSIGMA= ,E12.5/3X,9HISURF10= ,I2/4X,
1683        7 8HCANGLE= ,E12.5/7X,5HCSQ= ,E12.5/6X,6HNMAT= ,I2/6X,6HRHOF = ,E12.
1684        8 5/,5X,7HRHOFC= ,E12.5/)
1685      230 FORMAT (1H0)
1686      240 FORMAT (1H ,18X,10A8,1X,A10,2(1X,A8))
1687      250 FORMAT (2X,5HNKX= ,I4)
1688      260 FORMAT (2X,8HMESH-X= ,I4,3X,4HXL= ,1PE12.5,3X,4HXC= ,E12.5,3X,4HXR
1689         1= ,E12.5,3X,5HNXL= ,I4,3X,5HNXR= ,I4,3X,6HDXMN= ,E12.5)
1690      270 FORMAT (2X,8HMESH-Y= ,I4,3X,4HYL= ,1PE12.5,3X,4HYC= ,E12.5,3X,4HYR
1691         1= ,E12.5,3X,5HNYL= ,I4,3X,5HNYR= ,I4,3X,6HDYMN= ,E12.5)
1692      280 FORMAT (2X,5HNKY= ,I4)
1693      290 FORMAT (2X,6HNREG= ,I4)
1694      300 FORMAT (15X,1HK,6X,6HVOL(K),9X,5HPR(K))
1695      310 FORMAT (13X,I3,2X,1PE12.5,3X,E12.5)
1696          END
```

```
1697        SUBROUTINE SETUP
1698  *CALL,COMMON1
1699  C
1700  C     *** COMPUTE CONSTANT TERMS AND INITIALIZE NECESSARY VARIABLES
1701  C
1702  C     *** SET PARAMETER STATEMENT VALUE INTO CONSTANT
1703  C
1704        NVRM=NVOR
1705  C
1706        CYL=FLOAT(ICYL)
1707        EMF1=1.0-EMF
1708        T=0.0
1709        ITER=0
1710        CYCLE=0
1711        TWPRT=0.0
1712        TWPLT=0.0
1713        VCHGT=0.0
1714        NOCON=0
1715        NFLGC=0
1716        FNOC=0.0
1717        RCSQ=1.0/(RHOF*CSQ)
1718        IF (CSQ.LT.0.0) RCSQ=0.0
1719        IF (NMAT.EQ.1) RHOFC=RHOF
1720        RHOD=RHOF-RHOFC
1721        IF (CANGLE.EQ.90.0) CANGLE=CANGLE-EM6
1722        CANGLE=CANGLE*RPD
1723        TANCA=TAN(CANGLE)
1724        IPL=2
1725        IF (WL.EQ.5) IPL=3
1726        IPR=IM1
1727        IF (WR.EQ.5) IPR=IM2
1728        JPB=2
1729        IF (WB.EQ.5) JPB=3
1730        JPT=JM1
1731        IF (WT.EQ.5) JPT=JM2
1732  C
1733  C     *** SET CONSTANT TERMS FOR PLOTTING
1734  C
1735        XMIN=X(1)
1736        XMAX=X(IM1)
1737        IF (ISYMPLT.GT.0) XMIN=-XMAX
1738        YMIN=Y(1)
1739        YMAX=Y(JM1)
1740        D1=XMAX-XMIN
1741        D2=YMAX-YMIN
1742        D3=AMAX1(D1,D2)
1743        SF=1.0/D3
1744        XSHFT=0.5*(1.0-D1*SF)
1745        YSHFT=0.5*(1.0-D2*SF)
1746        DXMIN=EP10
1747        DO 10 I=2,IM1
1748     10 DXMIN=AMIN1(DELX(I),DXMIN)
1749        DYMIN=EP10
1750        DO 20 I=2,JM1
1751     20 DYMIN=AMIN1(DELY(I),DYMIN)
1752        VELMX1=AMIN1(DXMIN,DYMIN)/VELMX
1753  C
1754  C     *** DETERMINE SLOPED BOUNDARY LOCATION
```

```
1755  C
1756  C     *** COMPUTE INITIAL VOLUME FRACTION FUNCTION F  IN CELLS
1757  C
1758        DO 40 I=1,IMAX
1759        DO 30 J=2,JMAX
1760        F(I,J)=1.0
1761        IF (FLHT.GT.Y(J-1).AND.FLHT.LT.Y(J)) F(I,J)=RDY(J)*(FLHT-Y(J-1))
1762        IF (Y(J-1).GE.FLHT) F(I,J)=0.0
1763     30 CONTINUE
1764        F(I,1)=F(I,2)
1765     40 CONTINUE
1766  C
1767  C     *** GENERATE SPECIAL F-FUNCTION (FLUID) CONFIGURATION
1768  C
1769  C     *** CALCULATE DTVIS AND DTSFT
1770  C
1771        DS=1.0E+10
1772        DTVIS=1.0E+10
1773        DTSFT=1.0E+10
1774        DO 50 I=2,IM1
1775        DO 50 J=2,JM1
1776        DXSQ=DELX(I)**2
1777        DYSQ=DELY(J)**2
1778        RDSQ=DXSQ*DYSQ/(DXSQ+DYSQ)
1779        RDSQ=RDSQ/(3.0*NU+1.0E-10)
1780        DTVIS=AMIN1(DTVIS,RDSQ)
1781        DS=AMIN1(DELX(I),DELY(J),DS)
1782     50 CONTINUE
1783        SIGX=SIGMA
1784        RHOMN=AMIN1(RHOF,RHOFC)
1785        IF (SIGX.EQ.0.0) SIGX=EM10
1786        DTM=SQRT(RHOMN*DS**3/(SIGX*4.0*(1.0+CYL)))
1787        DTSFT=AMIN1(DTSFT,DTM)
1788  C
1789  C     *** CALCULATE BETA(I,J) FOR MESH
1790  C
1791        RDTEXP=2.0*SQRT(ABS(CSQ))/DS
1792        IF (CSQ.LT.0.0) RDTEXP=1.0E+10
1793        CTOS=DELT*RDTEXP
1794        COMG=AMIN1(CTOS**2,1.0)
1795        OMG1=(OMG-1.0)*COMG+1.0
1796        DO 60 I=2,IM1
1797        DO 60 J=2,JM1
1798        RHXR=(RHOFC+RHOD*F(I,J))*DELX(I+1)+(RHOFC+RHOD*F(I+1,J))*DELX(I)
1799        RHXL=(RHOFC+RHOD*F(I-1,J))*DELX(I)+(RHOFC+RHOD*F(I,J))*DELX(I-1)
1800        RHYT=(RHOFC+RHOD*F(I,J))*DELY(J+1)+(RHOFC+RHOD*F(I,J+1))*DELY(J)
1801        RHYB=(RHOFC+RHOD*F(I,J-1))*DELY(J)+(RHOFC+RHOD*F(I,J))*DELY(J-1)
1802        XX=DELT*RDX(I)*(2.0/RHXL+2.0/RHXR)+DELT*RDY(J)*(2.0/RHYT+2.0/RHYB)
1803        RHOR=RHOF/(RHOFC+RHOD*F(I,J))
1804        BETA(I,J)=OMG1/(XX*COMG+RCSQ*RHOR/DELT)
1805     60 CONTINUE
1806  C
1807  C     *** SET BETA(I,J)= -1.0 IN OBSTACLE CELLS
1808  C         MUST BE DONE BY HAND IN GENERAL
1809  C
1810  C     *** PRINT BETA(I,J) ON FILM AND PAPER
1811  C
1812        IF (IMOVY.EQ.1) GO TO 80
```

```
1813        WRITE (12,210)
1814        DO 70 J=1,JM1
1815        DO 70 I=1,IM1
1816        WRITE (12,220) I,J,BETA(I,J)
1817     70 CONTINUE
1818     80 CONTINUE
1819        WRITE (6,210)
1820        DO 90 J=1,JM1
1821        DO 90 I=1,IM1
1822        WRITE (6,220) I,J,BETA(I,J)
1823     90 CONTINUE
1824  C
1825  C     *** CALCULATE HYDROSTATIC PRESSURE
1826  C
1827        DO 100 I=2,IM1
1828        P(I,JMAX)=0.0
1829        DO 100 J=2,JM1
1830        JJ=JM1-J+2
1831        RHOYA=(RHOFC+RHOD*F(I,JJ))*DELY(JJ)*0.5+(RHOFC+RHOD*F(I,JJ+1))
1832       1 *DELY(JJ+1)*0.5
1833        IF (NMAT.EQ.1) RHOYA=(AMIN1(F(I,JJ+1),0.5)*DELY(JJ+1)+AMAX1(0.0,F
1834       1 (I,JJ)-0.5)*DELY(JJ))*RHOF
1835        P(I,JJ)=P(I,JJ+1)-GY*RHOYA
1836    100 CONTINUE
1837  C
1838  C     *** PARTICLE SET UP
1839  C
1840        NP=NPY*(1+NPX)
1841        IF (NP.EQ.0) GO TO 170
1842        DXP=(XPR-XPL)/FLOAT(NPX)
1843        DYP=(YPT-YPB)/FLOAT(NPY)
1844        K=0
1845        DO 110 JN=1,NPY,2
1846        DO 110 IN=1,NPX
1847        K=K+1
1848        XP(K)=XPL+(FLOAT(IN)-0.5)*DXP
1849        YP(K)=YPB+(FLOAT(JN)-1.0)*DYP
1850        IF (YP(K).GT.YPT) YP(K)=YPT
1851    110 CONTINUE
1852        DO 120 JN=2,NPY,2
1853        K=K+1
1854        XP(K)=XPL
1855        YP(K)=YPB+(FLOAT(JN)-1.0)*DYP
1856        IF (YP(K).GT.YPT) YP(K)=YPT
1857        DO 120 IN=1,NPX
1858        K=K+1
1859        XP(K)=XPL+FLOAT(IN)*DXP
1860        YP(K)=YPB+(FLOAT(JN)-1.0)*DYP
1861        IF (YP(K).GT.YPT) YP(K)=YPT
1862    120 CONTINUE
1863        NP=K
1864        DO 160 K=1,NP
1865        DO 130 I=2,IM1
1866        IF (XP(K).GE.X(I-1).AND.XP(K).LE.X(I)) IP(K)=I
1867        IF (X(I-1).GT.XPR) GO TO 140
1868    130 CONTINUE
1869    140 DO 150 J=2,JM1
1870        IF (YP(K).GE.Y(J-1).AND.YP(K).LE.Y(J)) JP(K)=J
```

```
1871          IF (Y(J-1).GT.YPT) GO TO 160
1872      150 CONTINUE
1873      160 CONTINUE
1874      170 CONTINUE
1875  C
1876  C       *** SET INITIAL SURFACE PRESSURE
1877  C
1878          DO 180 J=2,JM1
1879          DO 180 I=2,IM1
1880          PS(I,J)=0.0
1881      180 CONTINUE
1882  C
1883  C       *** SET INITIAL VELOCITY FIELD INTO U AND V ARRAYS
1884  C
1885          DO 190 I=2,IM1
1886          DO 190 J=2,JM1
1887          V(I,J)=VI
1888          U(I,J)=UI
1889          IF (F(I,J).GT.EMF.OR.NMAT.EQ.2) GO TO 190
1890          U(I,J)=0.0
1891          V(I,J)=0.0
1892      190 CONTINUE
1893  C
1894  C       *** SET INITIAL VOID REGION QUANTITIES
1895  C
1896          DO 200 K=1,NVRM
1897          NR(K)=0
1898          PR(K)=0.0
1899      200 VOL(K)=0.0
1900          RETURN
1901  C
1902      210 FORMAT (1H1)
1903      220 FORMAT (2X,5HBETA(,I2,1H,,I2,2H)=,1PE14.7)
1904          END
```

```
      SUBROUTINE TILDE
*CALL COMMON1
C
C     *** COMPUTE TEMPORARY U AND V EXPLICITLY
C
      DO 20 J=2,JM1
      DO 20 I=2,IM1
      U(I,J)=0.0
      RDELX=1.0/(DELX(I)+DELX(I+1))
      RDELY=1.0/(DELY(J)+DELY(J+1))
      IF (F(I,J)+F(I+1,J).LT.EMF.AND.NMAT.EQ.1) GO TO 10
      IF (BETA(I,J).LT.0.0.OR.BETA(I+1,J).LT.0.0) GO TO 10
      SGU=SIGN(1.0,UN(I,J))
      DUDR=(UN(I+1,J)-UN(I,J))*RDX(I+1)
      DUDL=(UN(I,J)-UN(I-1,J))*RDX(I)
      RDXA=DELX(I)+DELX(I+1)+ALPHA*SGU*(DELX(I+1)-DELX(I))
      RDXA=1.0/RDXA
      FUX=RDXA*UN(I,J)*(DELX(I)*DUDR+DELX(I+1)*DUDL+ALPHA*SGU*(DELX(I+1)
     1 *DUDL-DELX(I)*DUDR))
      VBT=(DELX(I)*VN(I+1,J)+DELX(I+1)*VN(I,J))*RDELX
      VBB=(DELX(I)*VN(I+1,J-1)+DELX(I+1)*VN(I,J-1))*RDELX
      VAV=0.5*(VBT+VBB)
      DYT=0.5*(DELY(J)+DELY(J+1))
      DYB=0.5*(DELY(J-1)+DELY(J))
      DUDT=(UN(I,J+1)-UN(I,J))/DYT
      DUDB=(UN(I,J)-UN(I,J-1))/DYB
      SGV=SIGN(1.0,VAV)
      DYA=DYT+DYB+ALPHA*SGV*(DYT-DYB)
      FUY=(VAV/DYA)*(DYB*DUDT+DYT*DUDB+ALPHA*SGV*(DYT*DUDB-DYB*DUDT))
      UBDYT=(DELY(J)*UN(I,J+1)+DELY(J+1)*UN(I,J))/(DELY(J)+DELY(J+1))
      UBDYB=(DELY(J-1)*UN(I,J)+DELY(J)*UN(I,J-1))/(DELY(J)+DELY(J-1))
      DUDXSQ=2.0*(UN(I-1,J)*RDX(I)/(DELX(I)+DELX(I+1))+UN(I+1,J)*RDX(I+1
     1 )/(DELX(I)+DELX(I+1))-UN(I,J)*RDX(I)*RDX(I+1))
      DUDYT=(UN(I,J+1)*DELY(J)*RDY(J+1)-UN(I,J)*DELY(J+1)*RDY(J)-UBDYT*
     1 (DELY(J)*RDY(J+1)-DELY(J+1)*RDY(J)))/(0.5*(DELY(J)+DELY(J+1)))
      DUDYB=(UN(I,J)*DELY(J-1)*RDY(J)-UN(I,J-1)*DELY(J)*RDY(J-1)-UBDYB*
     1 (DELY(J-1)*RDY(J)-DELY(J)*RDY(J-1)))/(0.5*(DELY(J-1)+DELY(J)))
      DUDYSQ=(DUDYT-DUDYB)*RDY(J)
      DUDXL=(UN(I,J)-UN(I-1,J))*RDX(I)
      DUDXR=(UN(I+1,J)-UN(I,J))*RDX(I+1)
      RXDUDX=RX(I)*(DELX(I+1)*DUDXL+DELX(I)*DUDXR)/(DELX(I)+DELX(I+1))
      RXSQU=UN(I,J)*RX(I)**2
      VISX=NU*(DUDXSQ+DUDYSQ+CYL*RXDUDX-CYL*RXSQU)
      RHOX=(RHOFC+RHOD*F(I,J))*DELX(I+1)+(RHOFC+RHOD*F(I+1,J))*DELX(I)
      U(I,J)=UN(I,J)+DELT*((P(I,J)-P(I+1,J))*2.0/RHOX+GX-FUX-FUY+VISX)
   10 CONTINUE
      V(I,J)=0.0
      IF (F(I,J)+F(I,J+1).LT.EMF.AND.NMAT.EQ.1) GO TO 20
      IF (BETA(I,J).LT.0.0.OR.BETA(I,J+1).LT.0.0) GO TO 20
      UBR=(DELY(J+1)*UN(I,J)+DELY(J)*UN(I,J+1))*RDELY
      UBL=(DELY(J+1)*UN(I-1,J)+DELY(J)*UN(I-1,J+1))*RDELY
      UAV=0.5*(UBR+UBL)
      DXR=0.5*(DELX(I)+DELX(I+1))
      DXL=0.5*(DELX(I)+DELX(I-1))
      SGU=SIGN(1.0,UAV)
      DXA=DXR+DXL+ALPHA*SGU*(DXR-DXL)
      DVDR=(VN(I+1,J)-VN(I,J))/DXR
      DVDL=(VN(I,J)-VN(I-1,J))/DXL
```

```
1963        FVX=(UAV/DXA)*(DXL*DVDR+DXR*DVDL+ALPHA*SGU*(DXR*DVDL-DXL*DVDR))
1964        SGV=SIGN(1.0,VN(I,J))
1965        DYA=DELY(J+1)+DELY(J)+ALPHA*SGV*(DELY(J+1)-DELY(J))
1966        DVDT=(VN(I,J+1)-VN(I,J))*RDY(J+1)
1967        DVDB=(VN(I,J)-VN(I,J-1))*RDY(J)
1968        FVY=(VN(I,J)/DYA)*(DELY(J)*DVDT+DELY(J+1)*DVDB+ALPHA*SGV*(DELY(J+1
1969       1 )*DVDB-DELY(J)*DVDT))
1970        VBDYR=(DELX(I+1)*VN(I,J)+DELX(I)*VN(I+1,J))/(DELX(I)+DELX(I+1))
1971        VBDYL=(DELX(I)*VN(I-1,J)+DELX(I-1)*VN(I,J))/(DELX(I)+DELX(I-1))
1972        DVDXR=(VN(I+1,J)*DELX(I)*RDX(I+1)-VN(I,J)*DELX(I+1)*RDX(I)-VBDYR*
1973       1 (DELX(I)*RDX(I+1)-DELX(I+1)*RDX(I)))/(0.5*(DELX(I+1)+DELX(I)))
1974        DVDXL=(VN(I,J)*DELX(I-1)*RDX(I)-VN(I-1,J)*DELX(I)*RDX(I-1)-VBDYL*
1975       1 (DELX(I-1)*RDX(I)-DELX(I)*RDX(I-1)))/(0.5*(DELX(I)+DELX(I-1)))
1976        DVDXSQ=(DVDXR-DVDXL)*RDX(I)
1977        DVDYSQ=2.0*(VN(I,J-1)*RDY(J)/(DELY(J+1)+DELY(J))-VN(I,J)*RDY(J+1)
1978       1 *RDY(J)+VN(I,J+1)*RDY(J+1)/(DELY(J+1)+DELY(J)))
1979        DVDXRX=(VBDYR-VBDYL)*RDX(I)*RXI(I)
1980        VISY=NU*(DVDXSQ+DVDYSQ+CYL*DVDXRX)
1981        RHOY=(RHOFC+RHOD*F(I,J))*DELY(J+1)+(RHOFC+RHOD*F(I,J+1))*DELY(J)
1982        V(I,J)=VN(I,J)+DELT*((P(I,J)-P(I,J+1))*2.0/RHOY+GY-FVX-FVY+VISY)
1983     20 CONTINUE
1984        RETURN
1985        END
```

```
      SUBROUTINE TMS10
*CALL,COMMON1
C
C     *** TWO MATERIAL SURFACE TENSION
C
C     *** NOTE: THIS ROUTINE INTRODUCES SOME NUMERICAL NOISE
C     ***       AND MAY BE REPLACED IN THE FUTURE
C
      DO 30 I=2,IM1
      DO 30 J=2,JM1
      IF (NF(I,J).EQ.0.OR.NF(I,J).GE.5.OR.BETA(I,J).LT.0.0) GO TO 30
      WHTL=0.5
      WHTR=0.5
      WHTT=0.5
      WHTB=0.5
      IF (NF(I,J).GT.2) GO TO 10
      WHTL=1.0-F(I,J)
      IF (NF(I,J).EQ.2) WHTL=1.0-WHTL
      WHTR=1.0-WHTL
      STFX=PS(I,J)*DELY(J)
      IF (NF(I,J).EQ.1) STFX=-STFX
      STFY=STFX*TANTH(I,J)
      GO TO 20
   10 WHTB=1.0-F(I,J)
      IF (NF(I,J).EQ.4) WHTB=1.0-WHTB
      WHTT=1.0-WHTB
      STFY=PS(I,J)*DELX(I)
      IF (NF(I,J).EQ.3) STFY=-STFY
      STFX=-STFY*TANTH(I,J)
   20 CONTINUE
      RHOXR=(RHOFC+RHOD*F(I,J))*DELX(I+1)+(RHOFC+RHOD*F(I+1,J))*DELX(I)
      U(I,J)=U(I,J)+2.0*DELT*WHTR*STFX/(RHOXR*DELY(J))
      RHOXL=(RHOFC+RHOD*F(I-1,J))*DELX(I)+(RHOFC+RHOD*F(I,J))*DELX(I-1)
      U(I-1,J)=U(I-1,J)+2.0*DELT*WHTL*STFX/(RHOXL*DELY(J))
      RHOYT=(RHOFC+RHOD*F(I,J))*DELY(J+1)+(RHOFC+RHOD*F(I,J+1))*DELY(J)
      V(I,J)=V(I,J)+2.0*DELT*WHTT*STFY/(RHOYT*DELX(I))
      RHOYB=(RHOFC+RHOD*F(I,J-1))*DELY(J)+(RHOFC+RHOD*F(I,J))*DELY(J-1)
      V(I,J-1)=V(I,J-1)+2.0*DELT*WHTB*STFY/(RHOYB*DELX(I))
   30 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE VFCONV
*CALL,COMMON1
C
C     *** CONVECT THE VOLUME OF FLUID FUNCTION F
C
      IF (CYCLE.LT.1) GO TO 40
      FLGC=0.0
      DO 30 J=1,JM1
      DO 30 I=1,IM1
      VX=U(I,J)*DELT
      VY=V(I,J)*DELT
      ABVX=ABS(VX)
      ABVY=ABS(VY)
      IF (ABVX.GT.0.5*DELX(I).OR.ABVY.GT.0.5*DELY(J)) FLGC=1.0
      IA=I+1
      ID=I
      IDM=MAX0(I-1,1)
      RB=X(I)
      RA=XI(I+1)
      RD=XI(I)
      IF (VX.GE.0.0) GO TO 10
      IA=I
      ID=I+1
      IDM=MIN0(I+2,IMAX)
      RA=XI(I)
      RD=XI(I+1)
   10 CONTINUE
      IAD=IA
      IF (NF(ID,J).EQ.3.OR.NF(ID,J).EQ.4) IAD=ID
      IF (FN(IA,J).LT.EMF.OR.FN(IDM,J).LT.EMF) IAD=IA
      FDM=AMAX1(FN(IDM,J),FN(ID,J))
      FX1=FN(IAD,J)*ABS(VX)+AMAX1((FDM-FN(IAD,J))*ABS(VX)-(FDM-FN(ID,J))
     1 *DELX(ID),0.0)
      FX=AMIN1(FX1,FN(ID,J)*DELX(ID))
      F(ID,J)=F(ID,J)-FX*RDX(ID)*((ABS(RB/RD))*CYL+(1.0-CYL))
      F(IA,J)=F(IA,J)+FX*RDX(IA)*((ABS(RB/RA))*CYL+(1.0-CYL))
      JA=J+1
      JD=J
      JDM=MAX0(J-1,1)
      IF (VY.GE.0.0) GO TO 20
      JA=J
      JD=J+1
      JDM=MIN0(J+2,JMAX)
   20 CONTINUE
      JAD=JA
      IF (NF(I,JD).EQ.1.OR.NF(I,JD).EQ.2) JAD=JD
      IF (FN(I,JA).LT.EMF.OR.FN(I,JDM).LT.EMF) JAD=JA
      FDM=AMAX1(FN(I,JDM),FN(I,JD))
      FY1=FN(I,JAD)*ABS(VY)+AMAX1((FDM-FN(I,JAD))*ABS(VY)-(FDM-FN(I,JD))
     1 *DELY(JD),0.0)
      FY=AMIN1(FY1,FN(I,JD)*DELY(JD))
      F(I,JD)=F(I,JD)-FY*RDY(JD)
      F(I,JA)=F(I,JA)+FY*RDY(JA)
   30 CONTINUE
   40 CONTINUE
      DO 80 J=2,JM1
      DO 80 I=2,IM1
      IF (BETA(I,J).LT.0.0) GO TO 80
```

```
2085          VCHG=0.0
2086          IF (F(I,J).GT.EMF.AND.F(I,J).LT.EMF1) GO TO 60
2087          IF (F(I,J).GE.EMF1) GO TO 50
2088          VCHG=F(I,J)
2089          F(I,J)=0.0
2090          GO TO 60
2091       50 CONTINUE
2092          VCHG=-(1.0-F(I,J))
2093          F(I,J)=1.0
2094       60 CONTINUE
2095          VCHGT=VCHGT+VCHG*DELX(I)*DELY(J)*(XI(I)*2.0*PI*CYL+(1.0-CYL))
2096          IF (F(I,J).LT.EMF1) GO TO 80
2097          IF (F(I+1,J).LT.EMF) GO TO 70
2098          IF (F(I-1,J).LT.EMF) GO TO 70
2099          IF (F(I,J+1).LT.EMF) GO TO 70
2100          IF (F(I,J-1).LT.EMF) GO TO 70
2101          GO TO 80
2102       70 F(I,J)=F(I,J)-1.1*EMF
2103          VCHG=1.1*EMF
2104          VCHGT=VCHGT+VCHG*DELX(I)*DELY(J)*(XI(I)*2.0*PI*CYL+(1.0-CYL))
2105       80 CONTINUE
2106     C
2107     C       *** SPECIAL BOUNDARY CONDITIONS FOR F
2108     C
2109          RETURN
2110          END
```

APPENDIX B

SAMPLE CALCULATIONS

Some idea of the power and flexibility of the SOLA-VOF program can be gained by reviewing the sample calculations briefly described in this Appendix. These examples have been selected to illustrate the wide range of problems that SOLA-VOF can address.

1. A BREAKING BORE.

The test problem provided for checking out new SOLA-VOF codes was an undular bore generated when a stream of fluid encountered a rigid wall. One advantage of the SOLA-VOF program is that it is not limited to simple free-surface configurations. This can be illustrated by increasing the height of the bore transition to a level where experimental data indicate a turbulent transition occurs. An easy way to increase the transition height is to reduce the gravitational acceleration used in the test problem. For example, reducing it from 1.0 to 0.4548 increases the bore transition height from 1.0 to 2.8.

A calculation for this case is shown in Fig. B-1. The computational mesh consisted of 50 equally spaced cells in the x-direction ($\delta x = 0.25$) and 20 cells with variable spacing in the y-direction. The variable spacing was selected to give the greatest resolution around $y = 1.0$ where a shear layer is formed as fluid flows into the bore.

Experimental evidence indicates that turbulent bore transitions have widths that are typically five times their elevation change. This is consistent with the calculational results, even though the calculation is not computing true three-dimensional turbulence. A better measure of the accuracy of the calculation is the final fluid height at the wall, which is 2.91 compared to the theoretical value of 2.8. A brief study of the entire calculation indicates that this 3.9% error would probably be smaller had the mesh length been increased to allow the bore to move further away from the wall.

2. BROKEN DAM PROBLEM.

In this example, a rectangular column of water, in hydrostatic equilibrium, is confined between two vertical walls (Fig. B-2). The water column is 1.0 unit wide and 2.0 units high. Gravity is acting downward with unit magnitude. At the beginning of the calculation, the right wall (dam) is removed and water is allowed to flow out along a dry horizontal floor. Experimental results for this problem have been reported[17] for the position vs time of the leading edge of the water as it flows to the right (Fig. B-3).

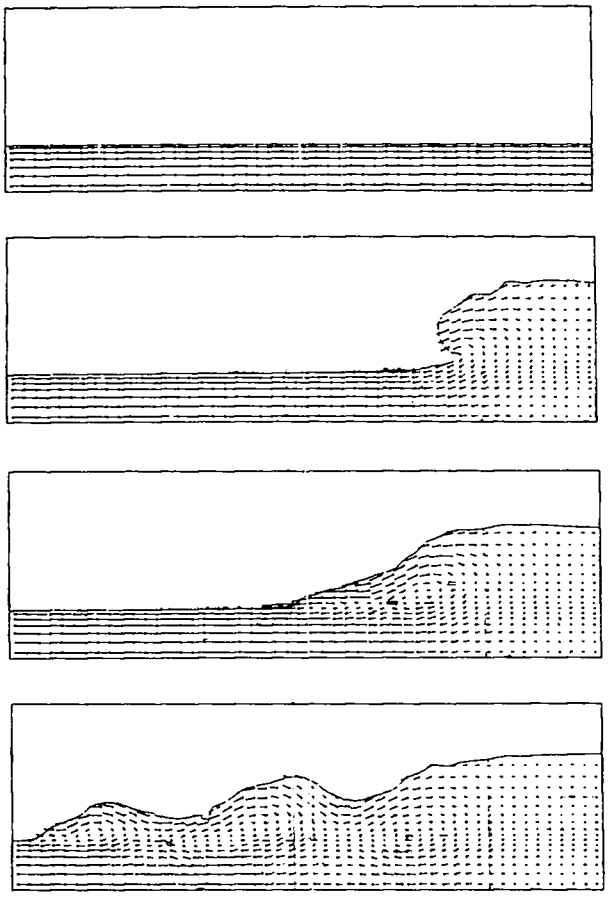

Fig. B-1.

Velocities and free-surface configurations for breaking bore calculation. Times are 0.0, 6.50, 8.51, and 14.01.

This is a good test problem because it has simple boundary conditions and a simple initial configuration. The appearance of both a vertical and horizontal free surface, however, provides a check on the capability of SOLA-VOF to treat free surfaces that are not single valued with respect to x or y. Results from two calculations are presented in Fig. B-3 in comparison with the experimental data. In both cases, the mesh consisted of 40 uniformly spaced columns ($\delta x = 0.1$) and 22 uniformly spaced rows. The smallest δy values are located at the bottom of the mesh where resolution is needed to define the thin leading edge of the advancing water. In the first calculation, the smallest δy was 0.05, while in the second it was 0.025. Figure B-3 shows that the best results are obtained with the smallest δy case, but both results are still quite good. The greatest deviation from experimental results is everywhere less than one cell width.

3. COLLAPSE OF A CYLINDRICAL FLUID COLUMN.

Although much like the Broken Dam problem, this calculation (Fig. B-4), illustrates several additional features of the SOLA-VOF program. The cylindrcal coordinate option was employed so that the resulting flow would approximate what might occur following the failure of a cylindrical storage tank for liquid fuel. A retaining collar surrounds the original tank, but as seen in Fig. B-4, it is too low and a large portion of the fluid splashes outside. The collar is defined by cells that are flagged as obstacles into which fluid cannot flow. It is interesting to note that almost no fluid splashes outside the collar when its height is increased to twice that shown in Fig. B-4.

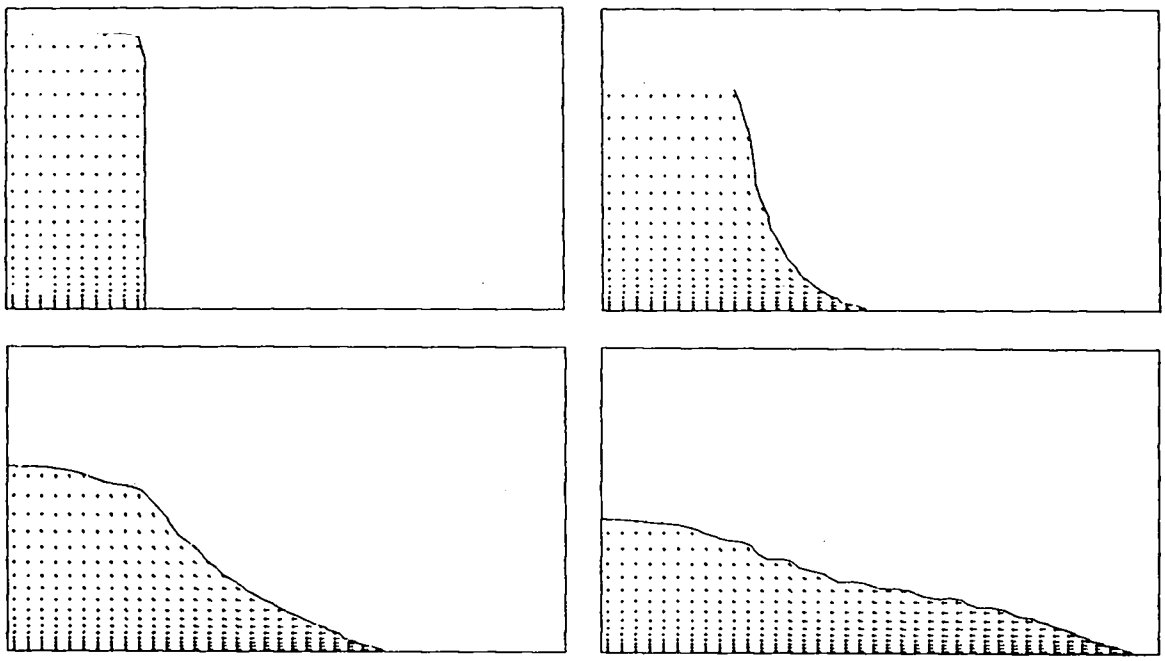

Fig. B-2.

Velocity vectors and fluid configurations computed for broken dam problem at times 0.0, 0.9, 1.4, and 2.0.

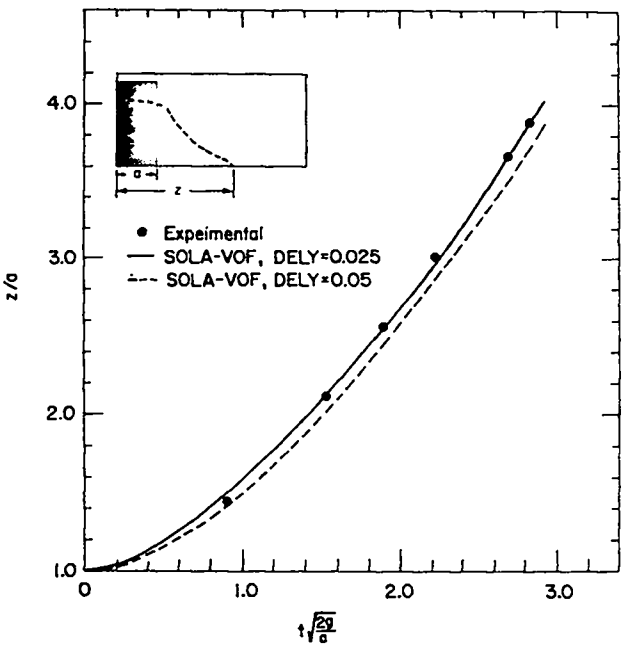

Fig. B-3.

Comparison of calculated results with experimental data for the broken dam problem.

This calculation also illustrates how the marker particle capability of the program can be used to provide a different type of flow visualization. The initial rectangular marker particle distribution was established by defining in the input data the region's boundaries and the number of particles wanted in each direction. The small scale structures evident in the later particle configurations must be interpreted with care because they are not adequately resolved by the finite-difference mesh.

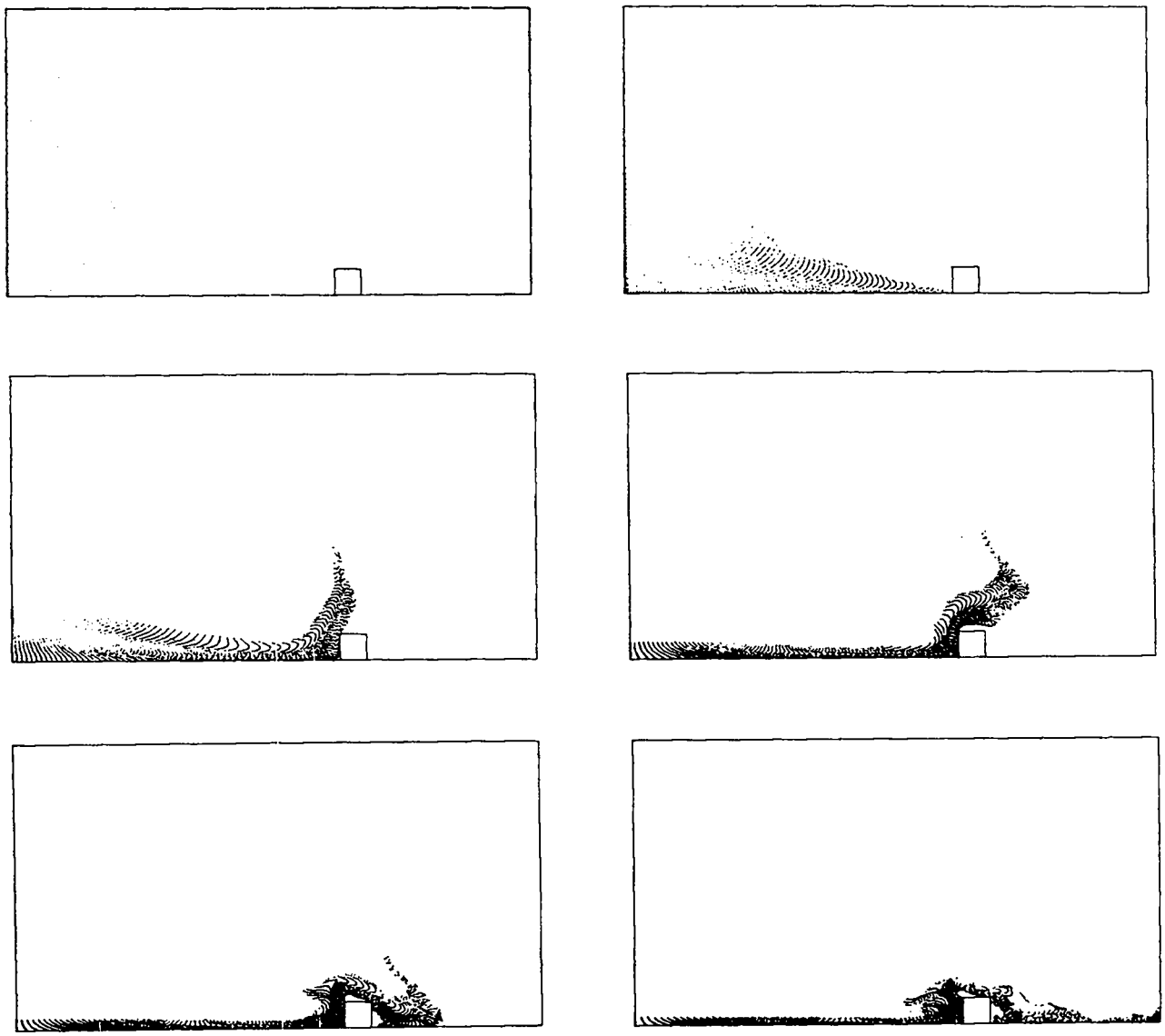

Fig. B-4.

Calculational results showing the collapse of a cylindrical column of fluid surrounded by a low retaining wall. Times shown are 0.0, 1.6, 2.5, 3.6, 4.2, and 4.6.

4. INSTABILITY OF A LIQUID COLUMN.

For some applications, such as the breakup of a thin liquid jet, surface tension forces must be considered. A classic problem of this type concerns the instability of a cylindrical column of fluid. When its free surface is perturbed by radial displacements, an exponential growth in perturbation amplitude may result that eventually causes the cylinder to break up into a series of discrete drops. Figure B-5 illustrates a SOLA-VOF calculation of this type of surface tension driven instability. The cylinder is initially perturbed with an axisymmetric displacement of its free surface that is sinusoidal in the axial direction. Two axial wave lengths λ are followed, with $\lambda = 4.598$ D, where D is the diameter of the undisturbed cylinder. The initial perturbation amplitude is 0.001 D. A comparison of the computed amplitude growth with linear theory[18] is shown in Fig. B-6. It is somewhat remarkable that the linear theory works so

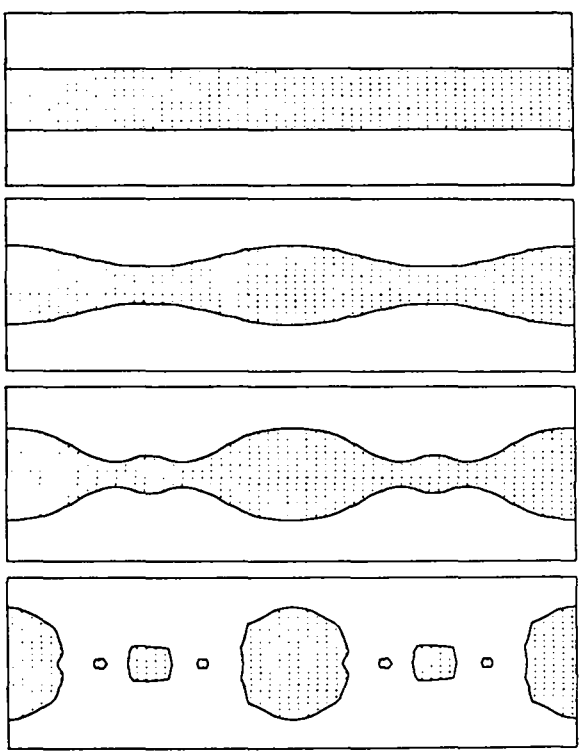

Fig. B-5.

Fluid configurations calculated for a surface tension driven instability of a cylindrical column of fluid. Times are approximately 0.0, 6.03, 6.49, and 7.0 in units of $\sqrt{D^3/\sigma}$.

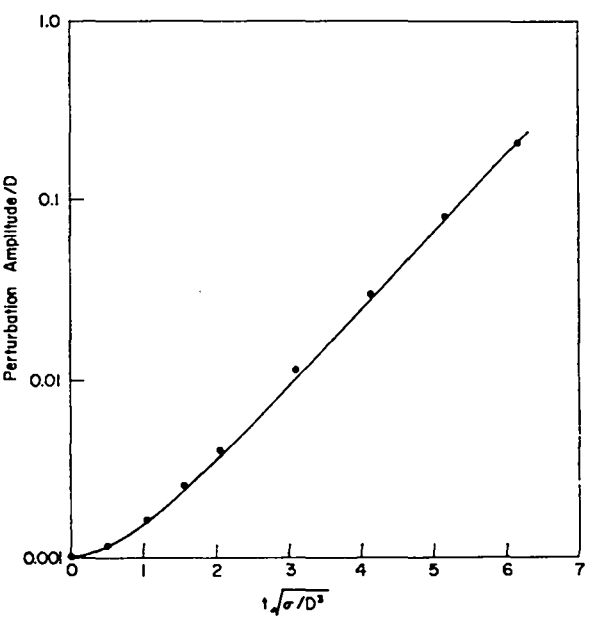

Fig. B-6.

Comparison of calculated perturbation amplitude vs time (dots) with linear theory (solid line).

far into the range of large amplitude displacements. Nevertheless, nonlinear effects are important because they are the cause of the secondary bulges that develop between the crests of the original harmonic disturbance. Following pinch-off, these secondary disturbances have resulted in small satellite drops between each large drop.

It is a strong testimony for the SOLA-VOF program to be able to carry this calculation completly through the pinch-off and droplet formation stage. For this to happen it was necessary to use the automatic time-step control provided in the program. The exponential growth of the perturbation requires a corresponding exponential decrease in time-step size to remain stable and to follow the evolution of the instability.

5. EXTRUSION OF IMMISCIBLE FLUIDS.

A mixture of oil and water provides an excellent example of a two-fluid system often encountered in practical situations. Because of their slightly differing densities and the action of interfacial surface tension forces, the mixture behaves dynamically quite different than either fluid separately.

Using the two-fluid and surface tension options, a variety of mixture problems can be investigated with the SOLA-VOF program. Figure B-7 contains results from a calculation of the passage of a liquid drop through a hole in a plate. The drop has a density equal to 9/10 of the density of the surrounding fluid. Surface tension σ at the interface between the two fluids is such that the Weber number ($\rho V^2 r/\sigma$) is equal to 0.2 based on the drop radius r and average flow rate V. This means that surface tension forces are more significant than those of inertia. In this example viscous forces are also relatively strong for the Reynolds number (Vr/ν) was chosen to be 1.00, and we assumed that both fluids have the same kinematic viscosity ν. If this were an oil-water mixture, it would correspond to a small oil drop ($r = 3 \times 10^{-4}$ cm) forced rapidly ($V = 46.1$ cm/s) through a hole in a thin plate. To force the drop through the constriction requires extra work to deform the interface against its surface tension forces. Much of this work, however, is recovered as the drop emerges on the downstream side of the constriction.

6. A REACTOR SAFETY APPLICATION.

Many boiling water reactors use a large pool of water to condense steam should a major steam leak occur. In some designs, steam is forced into the pool through long vertical pipes extending several pipe diameters below the surface of the pool. Before steam enters the pool, however, the air already in the

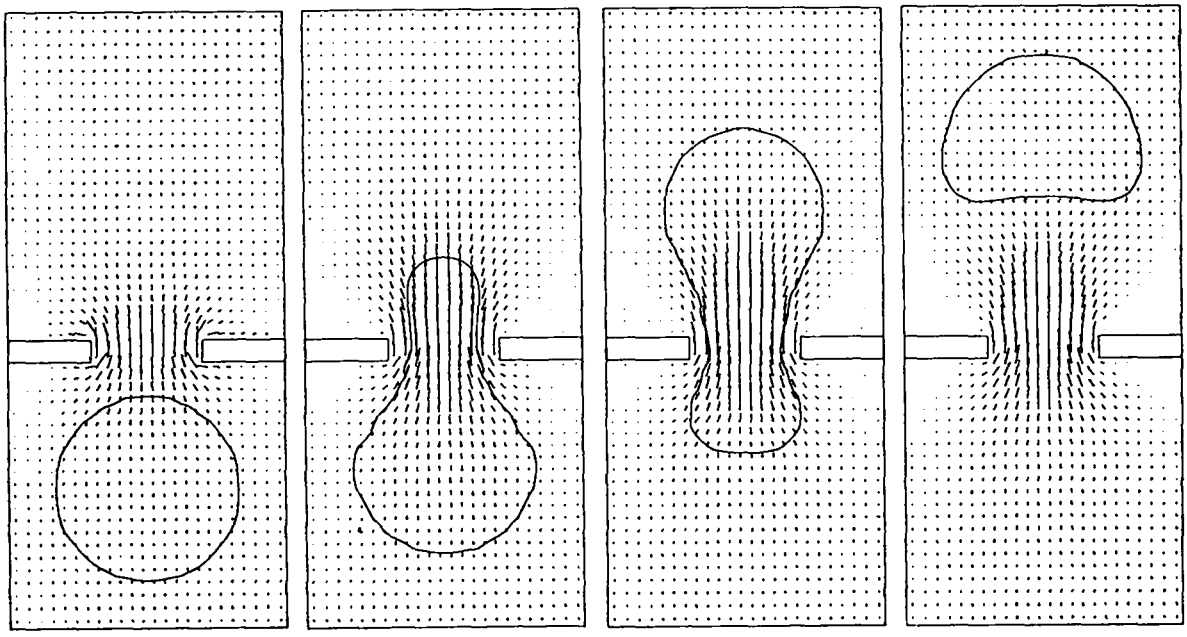

Fig. B-7.

Calculational results showing deformation of an oil drop in water passing through a hole in a rigid plate. Times shown in units of $2R/V_o$ are 0.0, 0.125, 0.5, and 0.875.

pipes must be pushed out. The ejection of this noncondensable air forms large bubbles in the pool and displaces the pool surface upward. Safety considerations require an understanding of the hydrodynamic forces generated during this process. For this purpose, several small scale experimental programs have been conducted and several groups have attempted supporting theoretical analysis.

A cross section of a single pipe apparatus used at the Massachusetts Institute of Technology (MIT)[19] is shown in Fig. B-8. It consists of a cylindrical vessel filled approximately half-way with water and an axisymmetric pipe extending down into the pool from above. At the beginning of a test, a valve is opened at the top end of the central pipe exposing it to a constant pressure plenum. Gas in the plenum flows through an orifice in the pipe and then into the lower pressure cylindrical tank by displacing water initially in the pipe.

To model this test apparatus with the SOLA-VOF code, it is necessary to supplement the code with calculations for the gas pressure in the pipe and for the pressure in the closed space above the pool surface. These pressures are then used as free-surface boundary pressures. A sequence of calculated results illustrating the fluid dynamics associated with the air clearing process are

contained in Fig. B-9. The free boundaries obviously undergo severe distortion, but the SOLA-VOF algorithm has no difficulty in following the fluid motion. Pressures measured at the center of the floor are compared with the corresponding calculated pressures in Fig. B-10. The agreement is reasonably good, except for some of the details associated with the initial pressure spike. There is some experimental evidence that the higher first spike and subsequent small second spike is a result of elastic flexibility in the apparatus, which was not included in the calculation. Similar results have also been obtained for many other test conditions and for other measured quantities.[20]

In some of the experimental tests,[19] it was observed that small air bubbles remaining in the fluid from earlier tests provided a springiness to the fluid and qualitatively changed the pressure history on the floor of the container. SOLA-VOF can be used to study this effect by treating the air bubbles as a mechanism that reduces the bulk modulus of the water. This treatment uses the limited compressibility option in the program with a fluid velocity of sound less than that of pure water. For example, Fig. B-11 shows the computed floor pressure histories from three calculations in which the only difference is the fluid sound speed. The three cases correspond to an incompressible calculation, a case in which the sound speed was one-fifth the normal speed of sound in pure water, and a case in which the speed was one-tenth the normal speed. A comparison of these results shows that as the sound speed is lowered large pressure oscillations develop on the floor and these increase in both magnitude and period as the fluid becomes more compressible.

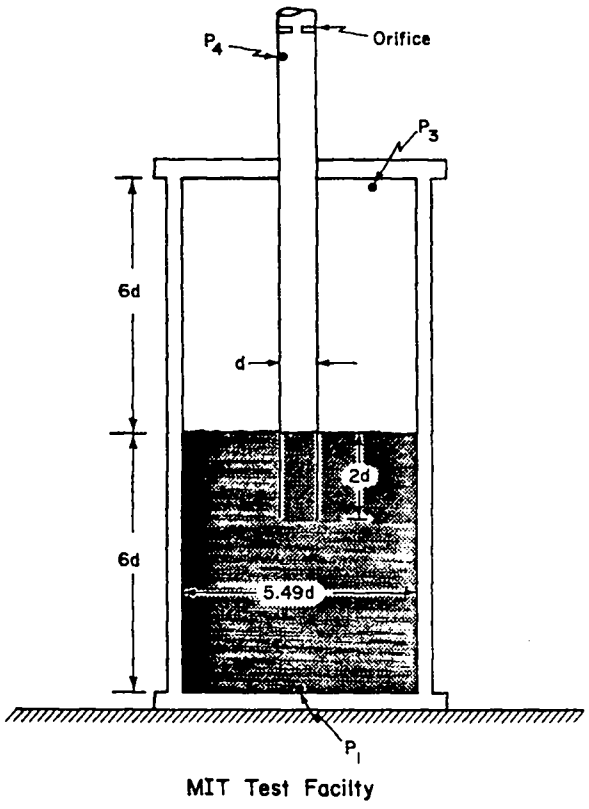

Fig. B-8.

The MIT single vent test apparatus.

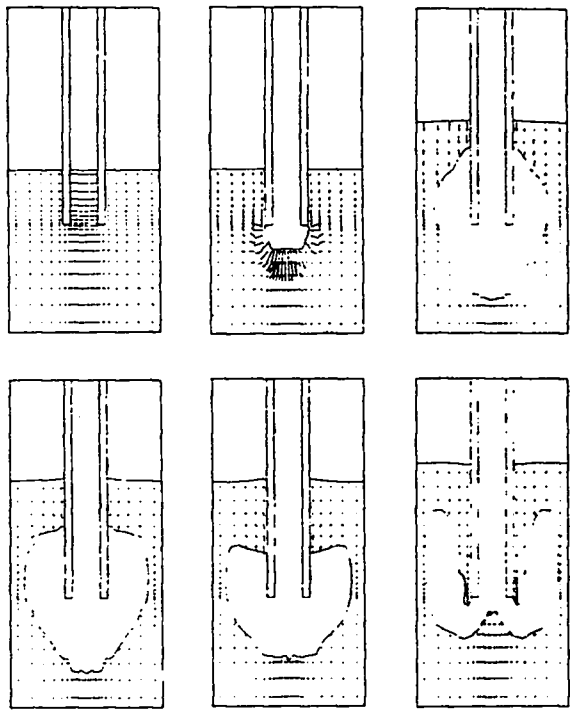

Fig. B-9.

Velocity vectors and free-surface configurations computed when air is forced through a submerged vent pipe.

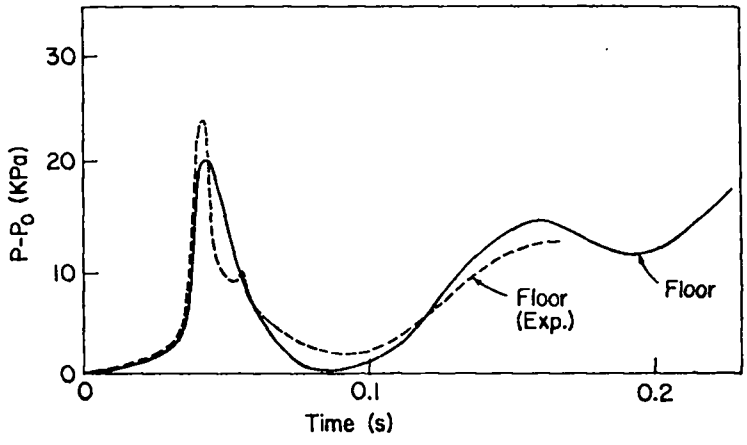

Fig. B-10.

Comparison of calculated and measured pressures on floor of pool chamber.

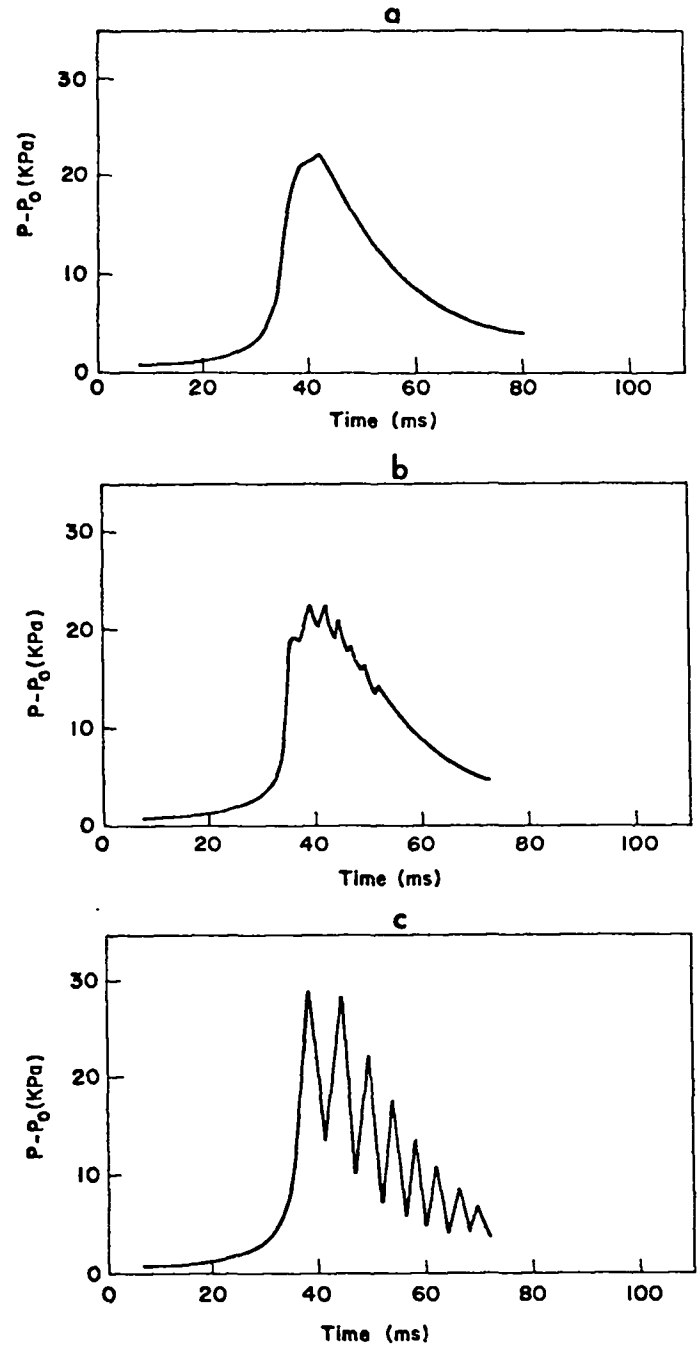

Fig. B-11.

Pressures computed on pool chamber floor during vent clearing (a) with an incompressible fluid, (b) with a fluid whose sound speed is one-fifth that of water, and (c) with a fluid whose sound speed is one-tenth that of water.

REFERENCES

1. F. H. Harlow and J. E. Welch, "Numerical Calculation of Time-Dependent Viscous Incompressible Flow," Phys. Fluids 8, 2182 (1965); J. E. Welch, F. H. Harlow, J. P. Shannon, and B. J. Daly, "THE MAC METHOD: A Computing Technique for Solving Viscous, Incompressible, Transient Fluid-Flow Problems Involving Free Surfaces," Los Alamos Scientific Laboratory report LA-3425 (March 1966).

2. R. K.-C. Chan and R. L. Street, "SUMMAC - A Numerical Model for Water Waves," Stanford University, Department of Civil Engineering, report 135 (August 1970).

3. B. D. Nichols and C. W. Hirt, "Improved Free-Surface Boundary Conditions for Numerical Incompressible Flow Calculations," J. Comput. Phys. 8, 434 (1971).

4. C. W. Hirt and B. D. Nichols, "Volume of Fluid (VOF) Method for the Dynamics of Free Boundaries," J. Comput. Phys., accepted for publication (1980).

5. C. W. Hirt and B. D. Nichols, "Adding Limited Compressibility to Incompressible Hydrocodes," J. Comput. Phys. 34, 390 (1980).

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

7. R. K.-C. Chan and R. L. Street, "A Computer Study of Finite Amplitude Water Waves," J. Comput. Phys. 6, 68 (1970).

8. C. W. Hirt, B. D. Nichols, and N. C. Romero, "SOLA - A Numerical Solution Algorithm for Transient Fluid Flows," Los Alamos Scientific Laboratory report LA-5852 (April 1975).

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

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

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

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

13. W. F. Noh and P. Woodward," The SLIC (Simple Line Interface Calculation) Method," Lawrence Livermore Laboratory report UCRL-52111 (August 1976).

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

15. C. W. Hirt, "Heuristic Stability Theory for Finite-Difference Equations," J. Comput. Phys. 2, 339 (1968).

16. J. J. Stoker, Water Waves (Interscience, New York, 1957).

17. J. C. Martin and W. J. Moyce, Phil. Trans. Roy. Soc. London A244, 312 (1952).

18. Sir Horace Lamb, Hydrodynamics (Dover Publications, New York, 1945).

19. W. G. Anderson, P. W. Huber, and A. A. Sonin, "Small Scale Modeling of Hydrodynamic Forces in Pressure Suppression Systems, Final Report," Department of Mechanical Engineering, Massachusetts Institute of Technology, Nuclear Regulatory Commission report NUREG/CR-0003 (March 1978).

20. B. D. Nichols and C. W. Hirt, "Numerical Simulation of BWR Vent-Clearing Hydrodynamics," Nucl. Sci. Eng. 73, 196 (1980).

* U.S. GOVERNMENT PRINTING OFFICE: 1980—677-115/152

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 $3.00

001-025	4.00	126-150	7.25	251-275	10.75	376-400	13.00	501-525	15.25
026-050	4.50	151-175	8.00	276-300	11.00	401-425	13.25	526-550	15.50
051-075	5.25	176-200	9.00	301-325	11.75	426-450	14.00	551-575	16.25
076-100	6.00	201-225	9.25	326-350	12.00	451-475	14.50	576-600	16.50
101-125	6.50	226-250	9.50	351-375	12.50	476-500	15.00	601-up	

Note: Add $2.50 for each additional 100-page increment from 601 pages up.

LASL
REPORT LIBRARY
SEP 22 1980
RECEIVED