5-30-75

Dr. 1356

LA-5852

UC-34 and UC-79d
Reporting Date: January 1975
Issued: April 1975

SOLA—A Numerical Solution Algorithm for Transient Fluid Flows

by

C. W. Hirt
B. D. Nichols
N. C. Romero

los alamos
scientific laboratory
of the University of California
LOS ALAMOS, NEW MEXICO 87544

An Affirmative Action/Equal Opportunity Employer

MASTER

UNITED STATES
ENERGY RESEARCH AND DEVELOPMENT ADMINISTRATION
CONTRACT W-7405-ENG. 36

This work was supported by the Office of Naval Research, contract NAonr-6-75, NR 062-455.

Printed in the United States of America. Available from
National Technical Information Service
U.S. Department of Commerce
5285 Port Royal Road
Springfield, VA 22151
Price: Printed Copy $5.45 Microfiche $2.25

SOLA — A NUMERICAL SOLUTION ALGORITHM FOR TRANSIENT FLUID FLOWS

by

C. W. Hirt
B. D. Nichols
N. C. Romero

ABSTRACT

A finite difference technique is presented for solving the Navier-Stokes equations for an incompressible fluid. The technique, based on the Marker-and-Cell method, is simplified to facilitate its use by persons with little or no experience in numerical fluid dynamics. Section I of the report describes the basic algorithm, SOLA, for confined flows; Sec. II describes modifications necessary for free or curved rigid surface boundaries. Each includes a flow chart and a FORTRAN listing. Sample problems show how to incorporate simple modifications into the basic code to adapt it to a variety of problems.

INTRODUCTION

Numerical techniques have been used to solve time-dependent incompressible fluid flow problems for well over a decade.[1] One of the best known techniques, the Marker-and-Cell (MAC) method,[2] uses an Eulerian finite-difference formulation with pressure and velocity as the primary dependent variables. This method, originally developed specifically for problems involving free surfaces, is equally capable of treating flows in confined regions. The basic MAC technique, which has been improved and extended,[3-6] has been used by researchers around the world for many different applications. The essential ideas of the MAC solution procedure are summarized here so that this report may be used as a self-contained guide. The best description of a sophisticated MAC code, including a flow chart and a FORTRAN computer listing, is given in Ref. 7.

This report describes a highly simplified MAC code, SOLA, that does not use marker particles and does not have built-in setups for internal obstacles or other complicating refinements. SOLA is designed for persons with little or no experience in numerical fluid dynamics. In addition to serving as an instructional tool, its purpose is to demonstrate that many useful and difficult problems can be solved without large, complicated computer programs. SOLA also provides a basis for developing many new numerical capabilities.

The basic solution technique in SOLA, for incompressible fluid flows without free surfaces, is presented in Sec. I of this report. The equations solved are the Navier-Stokes equations in two-dimensional plane or axisymmetric coordinates. Boundaries of the rectangular computing region can be chosen (1) as rigid walls with free-slip or no-slip tangential velocities, (2) as specified inflow or outflow boundaries, (3) as continuative outflow boundaries, or (4) as periodic boundaries. Internal walls and obstacles or sources and sinks can be added by inserting additional boundary conditions in a special section of the code reserved for this purpose.

Section II describes a simple extension of the SOLA code that permits a free surface or curved rigid boundary (free-slip) to be located across the top

MASTER

or bottom of the fluid region. These surfaces are defined in terms of their height, H(x,t) for the top surface and HB(x,t) for the bottom surface, with respect to the bottom of the computational mesh. Although the surface must be single-valued functions of the horizontal coordinate x, many useful and interesting problems can be studied by using them in different combinations.

The basic solution algorithm contained in the SOLA code also serves as a good foundation for developing new codes with other capabilities. For example, a scalar transport equation for density (or temperature) can be easily added to investigate buoyancy-driven flows and flows of stratified fluids. With some modifications of the basic equations, the SOLA solution algorithm has been adapted to saturated or unsaturated flow in porous media, to three-dimensional shallow water motions, to a drift flux approximation for two-phase flow, and to almost three-dimensional flow of air or water over variable terrain for pollution dispersal models.[8] Fully three-dimensional, time-dependent calculations have also been make with a straightforward extension of the SOLA code persented here.

For persons interested in performing their own calculations, Sec. I contains a simple flow chart, a descriptive list of input parameters, a FORTRAN computer listing for the basic SOLA code, and output from a sample test problem. Section II has a similar flow chart and FORTRAN computer listing for the code version, SOLA - SURF, which contains the curved surface options.

I. SOLA — BASIC SOLUTION ALGORITHM FOR CONFINED FLOWS

A. Equations of Motion

The differential equations to be solved are written in terms of Cartesian coordinates (x,y). For cylindrical (axisymmetric) coordinates, x is the radial coordinate, y the axial coordinate, and several additional terms must be added to the basic equations. In the following equations, these are included with a coefficient ξ, such that $\xi = 0$ corresponds to plane geometry and $\xi = 1$ corresponds to cylindrical geometry. The SOLA code uses the input parameter CYL instead of ξ.

The mass continuity equation is

$$\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} + \xi \frac{u}{x} = 0 \ . \qquad (1)$$

The equations of motion are the Navier-Stokes equations:

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

$$\frac{\partial v}{\partial t} + \frac{\partial uv}{\partial x} + \frac{\partial v^2}{\partial y} + \xi \frac{uv}{x} = -\frac{\partial p}{\partial y} + g_y + \nu\left[\frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2} + \frac{\xi}{x}\frac{\partial v}{\partial x}\right] \ . \qquad (2)$$

The velocity components (u,v) are in the coordinate directions (x,y), p is the ratio of pressure to constant density, and (g_x, g_y) are body accelerations. The kinematic viscosity coefficient is denoted by the constant ν.

B. Finite Difference Considerations

The finite difference mesh used for numerically solving the above equations consists of rectangular cells of width δx and height δy. The mesh region containing fluid is composed of IBAR cells in the x-direction, labeled with the index i, and JBAR cells in the y-direction, labeled with the index j. The fluid region is surrounded by a single layer of fictitious cells (or phantom or boundary cells) so that the cells in the complete mesh total IMAX = IBAR + 2 by JMAX = JBAR + 2 (see Fig. 1). Fluid velocities and pressures are located at cell positions as shown

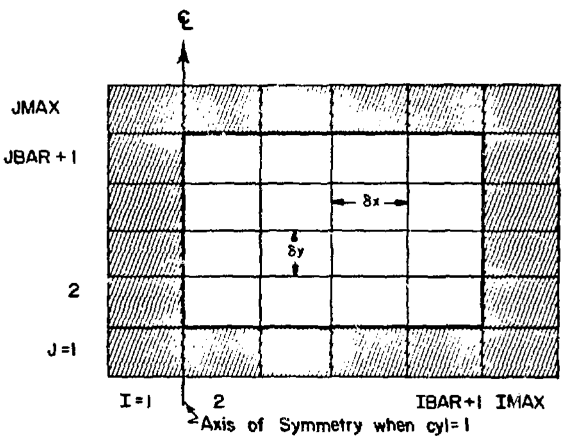


Fig. 1.
General mesh arrangement. Fictitious boundary cells are shaded.

in Fig. 2: u-velocity at the middle of the vertical sides of a cell, v-velocity at the middle of the horizontal sides, and pressure at the cell center.

The finite difference notation used in this report is:

$p^n_{i,j}$ = pressure at center of cell (i,j) at time level n

$u^n_{i,j}$ = x-direction velocity at middle of *right* side of cell (i,j) at time level n

$v^n_{i,j}$ = y-direction velocity at middle of *top* side of cell (i,j) at time level n.

Subscripts are used for the cell location and superscripts for the time level at which quantities are evaluated such that $t = n\delta t$, where δt is the time increment. In most MAC reports fractional indexes were used to represent quantities located at cell edges, e.g., $u_{i+1/2,j}$ to denote the x-direction velocity on the right-hand side of cell (i,j). In a FORTRAN program, however, fractional indexes are not allowed. Therefore, for consistency, all difference equations are written here as they appear in the actual code.

The difference approximation representing the continuity equation, Eq. (1), for a typical cell (i,j) is

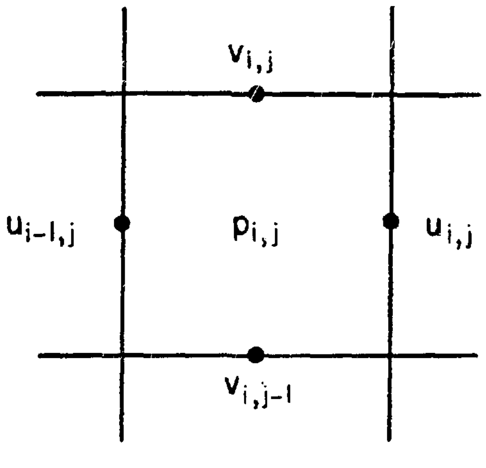


Fig. 2.
Arrangement of finite difference variables in a typical cell.

$$\frac{1}{\delta x}\left(u^{n+1}_{i,j} - u^{n+1}_{i-1,j}\right) + \frac{1}{\delta y}\left(v^{n+1}_{i,j} - v^{n+1}_{i,j-1}\right) + \frac{\xi}{2\delta x(i-1.5)}\left(u^{n+1}_{i,j} + u^{n+1}_{i-1,j}\right) = 0 \quad . \tag{3}$$

The difference equations approximating the Navier-Stokes equations, Eq. (2), are,

$$u^{n+1}_{i,j} = u^n_{i,j} + \delta t\left[\frac{1}{\delta x}\left(p^n_{i,j} - p^n_{i+1,j}\right) + g_x - FUX - FUY - FUC + VISX\right] \quad \text{and}$$

$$v^{n+1}_{i,j} = v^n_{i,j} + \delta t\left[\frac{1}{\delta y}\left(p^n_{i,j} - p^n_{i,j+1}\right) + g_y - FVX - FVY - FVC + VISY\right] \ , \tag{4}$$

where the convective and viscous fluxes are defined as

$$FUX = \frac{1}{4\delta x}\left[\left(u_{i,j} + u_{i+1,j}\right)^2 + \alpha|u_{i,j} + u_{i+1,j}| \cdot \left(u_{i,j} - u_{i+1,j}\right) - \left(u_{i-1,j} + u_{i,j}\right)^2 - \alpha|u_{i-1,j} + u_{i,j}|\left(u_{i-1,j} - u_{i,j}\right)\right] \ ,$$

$$FUY = \frac{1}{4\delta y}\left[\left(v_{i,j} + v_{i+1,j}\right)\left(u_{i,j} + u_{i,j+1}\right) + \alpha|v_{i,j} + v_{i+1,j}|\left(u_{i,j} - u_{i,j+1}\right) - \left(v_{i,j-1} + v_{i+1,j-1}\right)\left(u_{i,j-1} + u_{i,j}\right) - \alpha|v_{i,j-1} + v_{i+1,j-1}|\left(u_{i,j-1} - u_{i,j}\right)\right] \ ,$$

$$FUC = \frac{\xi}{8\delta x(i-1)}\left[\left(u_{i,j} + u_{i+1,j}\right)^2 + \left(u_{i-1,j} + u_{i,j}\right)^2 + \alpha|u_{i,j} + u_{i+1,j}|\left(u_{i,j} - u_{i+1,j}\right) + \alpha|u_{i-1,j} + u_{i,j}|\left(u_{i-1,j} - u_{i,j}\right)\right] \ ,$$

$$FVX = \frac{1}{4\delta x}\left[\left(u_{i,j} + u_{i,j+1}\right)\left(v_{i,j} + v_{i+1,j}\right) + \alpha|u_{i,j} + u_{i,j+1}|\left(v_{i,j} - v_{i+1,j}\right) - \left(u_{i-1,j} + u_{i-1,j+1}\right)\left(v_{i-1,j} + v_{i,j}\right) - \alpha|u_{i-1,j} + u_{i-1,j+1}|\left(v_{i-1,j} - v_{i,j}\right)\right] \ ,$$

$$\mathrm{FVY} = \frac{1}{4\delta y}\left[\left(v_{i,j} + v_{i,j+1}\right)^2 + \alpha\left|v_{i,j} + v_{i,j+1}\right| \cdot \left(v_{i,j} - v_{i,j+1}\right) - \left(v_{i,j-1} + v_{i,j}\right)^2 - \alpha\left|v_{i,j-1} + v_{i,j}\right|\left(v_{i,j-1} - v_{i,j}\right)\right] ,$$

$$\mathrm{FVC} = \frac{\xi}{8\delta x(i-1.5)}\left[\left(u_{i,j} + u_{i,j+1}\right)\left(v_{i,j} + v_{i+1,j}\right) + \left(u_{i-1} + u_{i-1,j+1}\right)\left(v_{i-1,j} + v_{i,j}\right) + \alpha\left|u_{i,j} + u_{i,j+1}\right|\left(v_{i,j} - v_{i+1,j}\right) + \alpha\left|u_{i-1,j} + u_{i-1,j+1}\right|\left(v_{i-1,j} - v_{i,j}\right)\right] ,$$

$$\mathrm{VISX} = \nu\left[\frac{1}{\delta x^2}\left(u_{i+1,j} - 2u_{i,j} + u_{i-1,j}\right) + \frac{1}{\delta y^2}\left(u_{i,j+1} - 2u_{i,j} + u_{i,j-1}\right) + \frac{\xi}{2\delta x^2(i-1)}\left(u_{i+1,j} - u_{i-1,j}\right) - \frac{\xi u_{i,j}}{\delta x^2(i-1)^2}\right] ,$$

and

$$\mathrm{VISY} = \nu\left[\frac{1}{\delta x^2}\left(v_{i+1,j} - 2v_{i,j} + v_{i-1,j}\right) + \frac{1}{\delta y^2}\left(v_{i,j+1} - 2v_{i,j} + v_{i,j-1}\right) + \frac{\xi}{2\delta x^2(i-1.5)}\left(v_{i+1,j} - v_{i-1,j}\right)\right] .$$

All quantities in the above convective and viscous fluxes are to be evaluated at time $n\delta t$. The coefficient α in these expressions gives the desired amount of upstream (donor cell) differencing; that is, when α is zero these difference equations are centered in space and correspond to the original MAC formulation.[2] The centered equations, however, are numerically unstable[9] and generally require some viscosity ν to remain stable. When α is equal to unity the equations reduce to the full upstream or donor cell form, which is stable provided the fluid is not permitted to cross more than one cell in one time step. In general, α should be chosen slightly larger than the maximum value of

$$\left|\frac{u\delta t}{\delta x}\right| \text{ or } \left|\frac{v\delta t}{\delta y}\right|$$

occurring in the mesh.

The velocities computed according to Eqs. (4) will not, in general, satisfy the continuity equation, Eq. (3). In the MAC method this incompressibility constraint is imposed by adjusting the cell pressures. For example, if the divergence of a cell, i.e., the left side of Eq. (3), is negative corresponding to a net flow of mass into the cell, the cell pressure is increased to eliminate the inflow. Likewise, when there is a net flow out of the cell the pressure is decreased to draw it back. Because there is one pressure variable for each cell, the divergence for each cell can be driven to zero in this way. The pressure adjustment must be done iteratively, however, because when one cell is adjusted its neighbors are affected. The iteration in SOLA proceeds by sweeping the mesh rows from left to right starting with the bottom row and working upward. For each cell encountered, the divergence D is computed using the most current velocity values available. The pressure change δp required to make D equal zero is,

$$\delta p = - D\Big/\left[2\delta t\left(\frac{1}{\delta x^2} + \frac{1}{\delta y^2}\right)\right] . \tag{5}$$

The new cell pressure is then $p_{i,j} + \delta p$, and the velocity components at the sides of the cell are adjusted to reflect this change,

$$u_{i,j} \to u_{i,j} + \frac{\delta t\,\delta p}{\delta x}$$

$$u_{i-1,j} \to u_{i-1,j} - \frac{\delta t\,\delta p}{\delta x}$$

$$v_{i,j} \to v_{i,j} + \frac{\delta t\,\delta p}{\delta y}$$

$$v_{i,j-1} \to v_{i,j-1} - \frac{\delta t\,\delta p}{\delta y} . \tag{6}$$

Equation (5) is derived by substituting the right sides of Eqs. (6) into the divergence condition, Eq. (3), and solving for δp.

In some cases, convergence of the iteration can be accelerated by multiplying δp by an over-relaxation factor ω. A value for ω that is often optimium is 1.8, but in no case should ω be larger than 2.0; otherwise an unstable iteration results.

Because the factor multiplying D in Eq. (5) is constant for all cells, its product with ω is denoted by BETA in the code and computed automatically in the setup section,

$$\text{BETA} = \omega \Big/ \left[2\delta t \left(\frac{1}{\delta x^2} + \frac{1}{\delta y^2} \right) \right] . \quad (7)$$

Convergence of the iteration is achieved when all cells have D values satisfying the inequality $|D/D_0| < \epsilon$, where D_0 is some reference value, and ϵ is typically of the order 10^{-3} or smaller. In practice D_0 typically equals unity and ϵ is adjusted to the desired level of accuracy. In this sense D_0 is superfluous, but it serves as a reminder that an acceptable convergence level must be chosen for each problem.

C. Summary of Steps in a Calculational Cycle

The steps involved in completing one calculational cycle are (1) computing guesses for the new velocities for the entire mesh from Eqs. (4), which involve only the previous time values for the contributing pressures and velocities in the various flux contributions (1000 section)* and (2) adjusting these velocities iteratively to satisfy the continuity equation, Eq. (3), by making appropriate changes in the cell pressures (3000 section). In the iteration, each cell is considered successively and is given a pressure change that drives its instantaneous velocity divergence to zero. Finally, when convergence has been achieved, the velocity and pressure fields are at the advanced time level and may be used as starting values for the next cycle.

D. Boundary Conditions

Up to this point, we have avoided applying boundary conditions. However, they are easily imposed by setting appropriate velocities in the fictitious cells surrounding the mesh (2000 section). Consider, for example, the left boundary of the computing mesh. If this boundary is to be a rigid free-slip wall, the normal velocity there must be zero and the tangential velocity, should have no normal gradient, i.e.,

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

If the left boundary is a no-slip rigid wall, then the tangential velocity component at the wall should also be zero and the conditions imposed are,

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

These conditions are imposed on the velocities resulting from applying Eqs. (4), and are imposed 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 it can potentially affect the entire flow field upstream. What is needed is a prescription that permits fluid to flow out of the mesh with a minimum of upstream influence. In this code we have used a continuative boundary condition that involves setting, for the left wall, for example,

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

These conditions, however, are only imposed after applying Eqs. (4) and *not* after each pass through the mesh during the pressure iteration. During the iteration the normal boundary velocities can vary with the changes in pressure, as any interior velocity component.

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 (IBAR-1)δx. Then the boundary conditions for the fictitious cells on the left are

---

*Numbers refer to statements in the SOLA program listed in Sec. G.

$$\left.\begin{array}{l} u_{1,j} = u_{IBAR,j} \\ \\ v_{1,j} = v_{IBAR,j} \end{array}\right\} \text{for all } j \ ,$$

and on the right

$$\left.\begin{array}{l} u_{1,j} = u_{IBAR,j} \\ p_{2,j} = p_{IBAR+1,j} \\ v_{2,j} = v_{IBAR+1,j} \\ v_{1,j} = v_{IBAR,j} \end{array}\right\} \text{for all } j.$$

In this case these conditions are imposed after applying Eqs. (4) and after each pressure iteration.

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 code has been written so that any of the above boundary conditions can be automatically imposed by setting input numbers. The appropriate input number for the left wall is designated WL, where

$$WL = \begin{cases} 1, \text{ rigid free-slip left wall} \\ 2, \text{ rigid no-slip left wall} \\ 3, \text{ continuative outflow left wall} \\ 4, \text{ periodic in x (provided WR} = 4) \ . \end{cases}$$

Similar input numbers are used for the right boundary (WR), top boundary (WT), and bottom boundary (WB). Clearly, when periodic conditions are desired in a given direction, both boundaries in that direction must be assigned wall numbers of 4.

To increase the usefulness of the basic code, specified inflow and outflow boundaries and obstacles inserted within the fluid region are desirable. In the case of obstacles, if restricted to those that can be constructed by blocking out cells of the computing mesh, we can add to the existing boundary conditions additional velocity prescriptions for the interior and boundaries of the obstacles. A place has been reserved for such special boundary conditions (2500 section) at the end of the main boundary condition section (2000 section). Several examples are included in the sample problems in Sec. F.

E. 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 and δy, 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 impossible because of limitations imposed by computing time or memory requirements, special care must be exercised in interpreting calculational results. For example, in 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 two 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

$$\delta t < \min\left\{\frac{\delta x}{|u|}, \frac{\delta y}{|v|}\right\} ,$$

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

$$\nu\delta t < \frac{1}{2}\,\frac{\delta x^2 \delta y^2}{\delta x^2 + \delta y^2} \quad .$$

With δt chosen to satisfy the above two inequalities, the last parameter needed to insure numerical stability is α. We have already noted in Sec. B that the proper choice for α is

$$1 \geq \alpha > \max\left\{\left|\frac{u\delta t}{\delta x}\right| , \left|\frac{v\delta t}{\delta y}\right|\right\} .$$

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 good choice. If α is too large an unnecessary amount of numerical smoothing (diffusion-like truncation errors) may be introduced.[9]

F. Sample Applications

A cross section of calculations done with the SOLA program are briefly described here. In each case the basic code has been supplemented with special boundary conditions to define the specific problem being studied. These changes are inserted into the code (2500 section) at the end of the main boundary condition section.

1. Flow About a Cylindrical Can. To compute the flow about a cylindrical can moving at constant speed in an axial direction, the SOLA program is set for cylindrical coordinates (CYL = 1.0). Figure 3 shows the mesh arrangement, which consists of 20 cells in the radial, x-direction (IBAR = 20), and 40 cells in the axial, y-direction (JBAR = 40). The cylindrical can is composed of 5 by 10 cells ($2 \leq i \leq 6$ and $12 \leq j \leq 21$). In this region, and on its boundary, the fluid velocity is maintained identically zero by inserting the following statements into the special boundary condition section:

$$u_{i,j} = 0 \quad \text{for } i = 1,\ldots,\ 6 \text{ and } j = 12,\ldots,\ 21$$

$$v_{i,j} = 0 \quad \text{for } i = 1,\ldots,\ 6 \text{ and } j = 11,\ldots,\ 21 .$$

The mean flow field is generated by defining, in the special boundary condition section, a constant axial velocity VI across the bottom of the computing mesh (WB = 1),

$$v_{i,1} = \mathrm{VI} \quad \text{for } i = 2,\ldots,\ \mathrm{IM1} ,$$

where IM1 = IMAX-1. A continuative outflow boundary is used across the top (WT=3), and rigid free-slip boundaries are used along the mesh sides (WL = WR = 1). See Fig. 3 for a complete list of input parameters. Definitions of all the code parameters are listed at the beginning of Sec. G, which contains the SOLA FORTRAN listing.

IBAR=	2.00000E+01
JBAR=	4.00000E+01
DELX=	2.00000E-01
DELY=	4.00000E-01
DELT=	1.00000E-01
NU=	0.
CYL=	1.00000E+00
EPSI=	5.00000E-03
DZRO=	1.00000E+00
GX=	0.
GY=	0.
UI=	0.
VI=	1.00000E+00
VELMX=	9.00000E-01
TWFIN=	1.00000E+0'
CWPRT=	2.00000E+01
CWPLT=	2.00000E+01
OMG=	1.70000E+00
ALPHA=	5.00000E-01
WL=	1.00000E+00
WR=	1.00000E+00
WT=	3.00000E+00
WB=	1.00000E+00

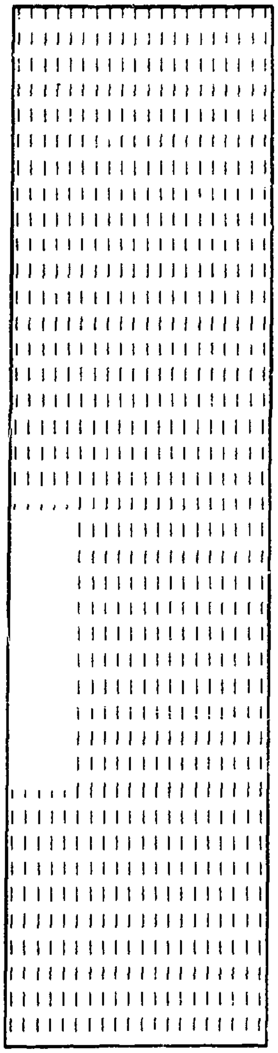


Fig. 3.
Initial velocity field and input parameters for calculating flow about a cylinder. Left edge of mesh is axis of symmetry. Vectors plotted as dots indicate centers of cells within the cylinder.

Figure 4 shows the computed velocity field at time t = 8.1 (81 cycles with δt=0.1). Each vector originates at the center of a computational cell and is drawn with a direction and magnitude proportional to the average of the velocity components located at the cell sides.

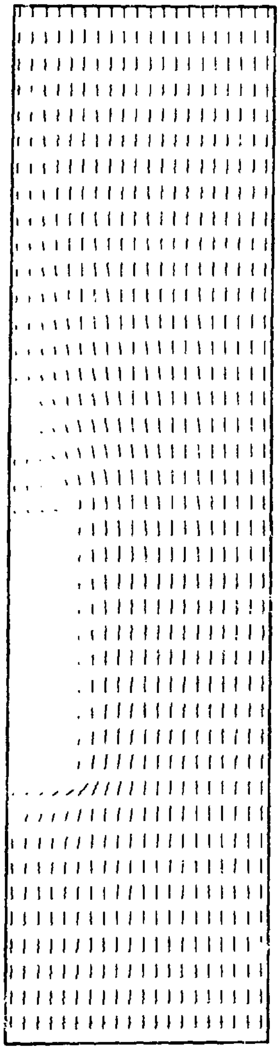


Fig. 4.
Velocity field generated by translating cylinder at $t = 8.1$.

A steady state is not reached in this calculation, because the recirculating wake region continues to grow in length behind the cylinder. A periodic shedding of vortexes (i.e., vortex street) does not develop because of the imposed axial symmetry.

The first cycle of this calculation requires a large number of iterations to achieve convergence, because the initial condition V = VI everywhere outside the cylinder is a poor first guess. Convergence can be improved by defining the outflow velocity at j = JM1 (JM1 = JMAX−1) to be equal to VI for the first few cycles.

An interesting variation of this problem is to move the right boundary nearer to the cylinder, to impose periodic boundaries in the axial direction (WB = WT = 4), and to impose a constant pressure drop across the flow. The resulting calculation then simulates the transport of cans in a pneumatic tube.[10]

2. Flow Over a Recessed Highway. Obstacles with boundaries cutting diagonally across cells can be represented by stepped obstacles. For example, to compute the flow across a notch with sloping sides, as shown in Fig. 5, the following boundary conditions were inserted in the special boundary condition section; for the left slope

$$u_{i,j} = v_{i,j} = 0 \quad \text{for} \quad \begin{cases} i = 1, \ldots, (21-j) \\ j = 2, \ldots, 11 \end{cases}$$

and for the right slope

$$u_{i,j} = 0 \quad \text{for} \quad \begin{cases} i = (26+j), \ldots, \mathrm{IM1} \\ j = 2,\ 11 \end{cases}$$

$$v_{i,j} = 0 \quad \text{for} \quad \begin{cases} i = (27+j), \ldots, \mathrm{IM1} \\ j = 2,\ 11\ . \end{cases}$$

IBAR= 4.50000E+01
JBAR= 2.00000E+01
DELX= 2.00000E-01
DELY= 2.00000E-01
DELT= 5.00000E-02
NU= 0.
CYL= 0.
EPSI= 5.00000E-03
DZRO= 1.00000E+00
GX= 0.
GY= -1.00000E+00
UI= 1.00000E+00
VI= 0.
VELMX= 9.00000E-01
TWFIN= 2.00000E+01
CWPRT= 9.99900E+03
CWPLT= 2.00000E+01
OMG= 1.70000E+00
ALPHA= 5.00000E-01
WL= 1.00000E+00
WR= 3.00000E+00
WT= 1.00000E+00
WB= 1.00000E+00

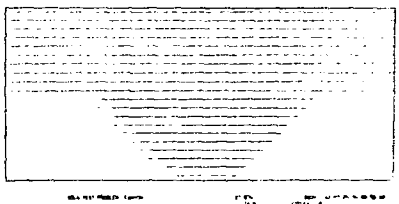


Fig. 5.
Initial velocity field and input parameters for flow over cut highway.

These conditions approximate a no-slip boundary cutting the diagonals of the *obstacle* cells adjacent to the fluid cells. These same conditions also approximate a free-slip boundary cutting the diagonals of the *fluid* cells adjacent to the obstacle cells. In the latter case, however, the tangential stress is not zero; therefore, this free-slip condition only works when the viscous stress terms are omitted from the equations of motion ($\nu=0$). Different slopes can be obtained by appropriately adjusting the ratio $\delta x/\delta y$.

The notched mesh described above has been used to compute the wind field near a sunken highway. A cross flow was generated by inserting into the special boundary condition section.

$$u_{1,j} = \text{UI for } j = 12, \ldots, \text{JM1}.$$

Boundary conditions input for the basic mesh were rigid free-slip walls (WL = WT = WB = 1) except on the right wall, which was a continuative boundary (WT = 3). In all cases, the special boundary conditions override the input conditions because they are located at the end of the basic boundary condition section. Figure 5 gives other input parameters.

The resulting flow field at t = 20 is shown in Fig. 6. A large recirculating eddy is shown in the highway notch. This type of flow structure has a significant effect on the dispersal of automobile pollutants.

3. Water-Cooled Reactor Model. A model simulating the core region of a pressurized water-cooled reactor can be easily set up in the following way. Axisymmetric coordinates are used (CYL = 1.0) with the mesh arrangement shown in Fig. 7. An inflow collar is located at the upper right corner, and is defined by assigning $u_{22,29} = u_{22,30} = u_{22,31} = -1.0$ in the special boundary condition section. The top boundary of the mesh is defined as an outflow boundary, except for the four outermost cells. For these last cells, the conditions set in the special boundary condition section are $v_{i,JM1} = 0.0$ for $i = 18, \cdot \cdot \cdot$, IM1. The cylindrical collar separating the central core region from the outside boundary is defined by inserting into the special boundary condition section,

$$u_{16,j} = u_{17,j} = v_{17,j} = 0.0 \quad \text{for } j = 12, \text{ JM1} ,$$

$$v_{17,11} = 0.0 \quad .$$

A frictional drag was used in the core region to represent the influence of control rods, supports, and other plumbing. This drag was inserted in the region $2 \leq i \leq 16$, $12 \leq j \leq$ JM1 by adding to the right side of the $u_{i,j}$ equation in the temporary velocity calculation (1000 section) a term equal to $-\kappa\delta t\ u^2_{i,j}$, and adding to the $v_{i,j}$ equation a term equal to $-\kappa\delta t\ v^2_{i,j}$. Other drag expressions can just as easily be used and can be defined as functions of space and time. The initial velocity distribution was defined in the setup as v = −1.0 in the outer collar and

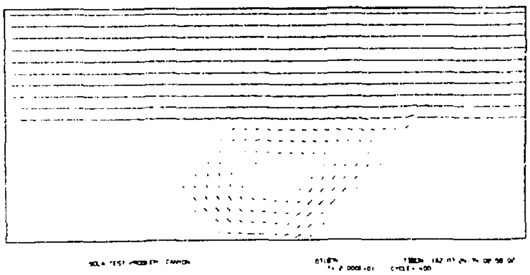


Fig. 6

Velocity field at t = 20.0 shows large recirculation in highway cut. Center of recirculating eddy is shifted downstream from the center of the cut.

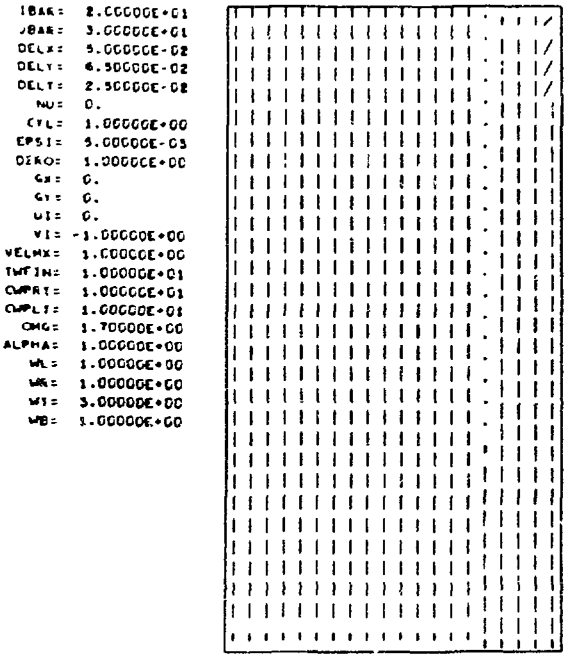


Fig. 7.

Initial velocity field and input parameters for simulation of reactor core and downcomer flow.

$v = +0.64$ in the inner core, which gives the same amount of mass moving upward as is moving downward.

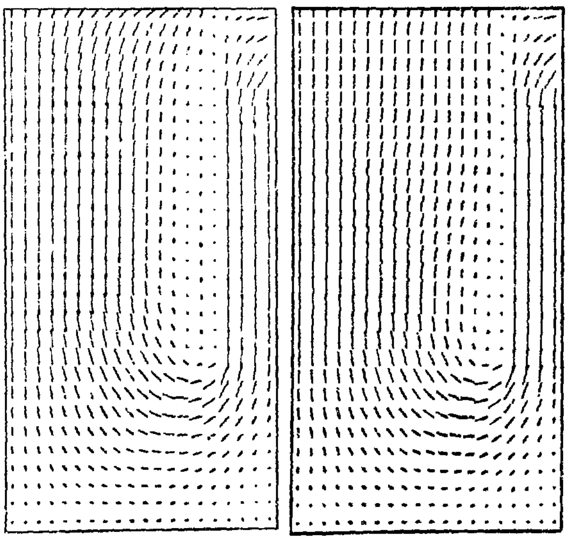
Figure 8 shows a comparison of two calculations, one with the drag coefficient κ equal to zero (A) and the other with κ equal to unity (B). With no drag there is a long narrow recirculation region in the core next to the outer wall. The addition of drag eliminates this recirculation and forces the flow in the core to be nearly uniform. In both cases there is a small recirculation region in the lower right corner of the bottom plenum.

Many variations of this basic setup can be imagined. For example, the inflow and outflow can be defined as arising from a fixed external pressure drop rather than a fixed inflow rate. Also, a rounded bottom for the lower plenum might be approximated by using a stepped boundary in the bottom right corner.

G. Details of the SOLA Program

A conceptual flow chart and FORTRAN listing of the SOLA Program is given in this section. The numbers beside some of the boxes in the flow chart refer to statement numbers in the main program where those instructions appear.

To set up a problem, program in whatever special boundary conditions are desired, if any, in the 2500 section and define any special initial conditions in



(A) (B)

Fig. 8.
Comparison of two calculations of flow in reactor core: (A) with no core drag and (B) with drag.

the 100 section. The basic input parameters that must be defined for every problem are as follows:

IBAR = number of cells in the x-direction (excluding boundary cells)
JBAR = number of cells in the y-direction (excluding boundary cells)
DELX = δx = width of cell in x-direction
DELY = δy = height of cell in y-direction
DELT = δt = time increment
NU = ν = coefficient of kinematic viscosity
CYL = ξ = geometry indicator (1.0 for cylindrical coordinates, 0.0 for plane coordinates)
EPSI = ϵ = pressure interation convergence criterion
DZRO = D_0 scaling factor for convergence test
GX = g_x = body acceleration in positive x-direction
GY = g_y = body acceleration in positive y-direction
UI = x-direction velocity used for initializing mesh and/or setting special boundary conditions
VI = y-direction velocity used for initializing mesh and/or setting special boundary conditions
VELMX = maximum velocity expected in problem, used to scale velocity vector plot
TWFIN = problem time when calculation is to be terminated
CWPRT = number of cycles between long prints output on paper
CWPLT = number of cycles between plots and listings to be output on film
OMG = ω = over-relaxation factor used in pressure iteration
ALPHA = α = controls amount of donor cell fluxing (1.0 for full donor cell differencing and 0.0 for centered differencing.)
WL = indicator for boundary condition to be used along the left side of the mesh (1.0 = rigid free-slip wall, 2.0 = rigid no-slip wall, 3.0 = continuative boundary, and 4.0 = periodic boundary)
WR = indicator for boundary condition along right side of mesh (see WL)
WT = indicator for boundary condition along top of mesh (see WL)
WB = indicator for boundary condition along bottom of mesh (see WL).

The following listing of SOLA is for a CDC-7600 computer at the Los Alamos Scientific Laboratory (LASL). The program, in FORTRAN IV, should be compatible with other machines, except for some of the control cards and some of the subroutine names used for film output.

H. Sample Test Problem

To help debug new SOLA codes, this section contains listings from a simple test problem. The problem is to compute the flow generated in viscous fluid in a square cavity when the top boundary of the cavity is impulsively set into motion parallel to itself.

The sample problem uses a crude 5 x 5 mesh, i.e., IBAR = JBAR = 5. Mesh increments are $\delta x = \delta y = 0.2$ corresponding to a cavity one unit square. All boundaries are no-slip walls (WL = WR = WT = WB = 2.0).

The fluid is initially at rest (UI = VI = 0.0) and has a coefficient of viscosity of $\nu = 0.4$. The sliding of the top boundary is imposed by inserting into the special boundary condition section (2500 section)

$$u_{i,JMAX} = 1.0 \text{ for } i = 1,IMAX.$$

Strictly speaking, the top boundary is located midway between $u_{i,JMAX}$ and $u_{i,JMAX-1}$ so that the average of these two velocities should equal unity, i.e.,

$$u_{i,JMAX} = 2.0 - u_{i,JMAX-1} \quad \text{for } i = 1,IMAX .$$

However, for this test problem the less accurate but simple expression was used. A complete list of input parameters is included in Fig. 9.

IBAR=	5.00000E+00
JBAR=	5.00000E+00
DELX=	2.00000E-01
DELY=	2.00000E-01
DELT=	2.00000E-02
NU=	4.00000E-01
CYL=	0.
EPSI=	5.00000E-03
DZRO=	1.00000E+00
GX=	0.
GY=	0.
UI=	0.
VI=	0.
VELMX=	3.00000E-01
TWFIN=	2.00000E+00
CWPRT=	1.00000E+03
CWPLT=	1.00000E+01
OMG=	1.70000E+00
ALPHA=	1.20000E-01
WL=	2.00000E+00
WR=	2.00000E+00
WT=	2.00000E+00
WB=	2.00000E+00

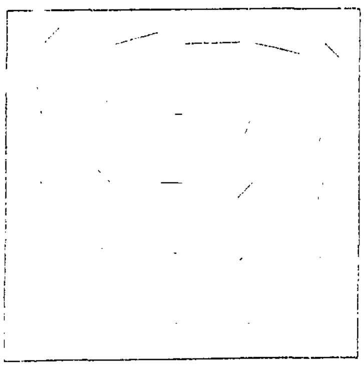


Fig. 9.
Velocity field at t = 1.0 and input parameters for sample test problem of viscous flow in a cavity.

In this problem no velocities are expected to exceed the top boundary's. Therefore, the accuracy and stability condition that fluid not convect more than one cell per cycle is

$$\delta t < 0.2 \ .$$

The diffusion stability condition requires

$$\delta t < \frac{\delta x^2}{4\nu} = 0.025 \ .$$

Thus, this problem is controlled by diffusion. The time step chosen for the calculation was $\delta t = 0.02$.

Table I lists the computed results after 1 cycle, 10 cycles, and 50 cycles. The flow field reaches steady state by approximately the tenth cycle ($t = 0.2$). The velocity field at $t = 1.0$ is shown in Fig. 9. The total calculational time (CP time) for 100 cycles of calculation on a CDC - 7600 computer was approximately 6 s, including film output every 10 cycles.

TABLE I

SOLA TEST PROBLEM

ITER= 15 TIME= 2.00000E-02 CYCLE= 1

I	J	U	V	P
1	1	-0.	0.	0.
1	2	0.	-9.31684E-03	0.
1	3	0.	-2.21067E-02	0.
1	4	0.	-4.47712E-02	0.
1	5	0.	-9.29405E-02	0.
1	6	0.	0.	0.
2	1	8.62282E-03	0.	0.
2	2	-8.62282E-03	9.31684E-03	-4.41928E-01
2	3	-1.27144E-02	2.21067E-02	-5.35036E-01
2	4	-2.20212E-02	4.47712E-02	-7.50184E-01
2	5	-4.01305E-02	9.29405E-02	-1.22582E+00
2	6	9.09722E-02	0.	-2.11322E+00
3	1	1.42036E-02	0.	0.
3	2	-1.42036E-02	5.41972E-03	-3.55703E-01
3	3	-1.95751E-02	1.21995E-02	-4.27097E-01
3	4	-3.17412E-02	2.12559E-02	-5.29092E-01
3	5	-5.29724E-02	2.02407E-02	-7.42451E-01
3	6	1.18990E-01	0.	-1.02292E+00
4	1	1.42026E-02	0.	0.
4	2	-1.42826E-02	4.76872E-05	-2.11070E-01
4	3	-1.96239E-02	3.55725E-05	-2.12146E-01
4	4	-3.17060E-02	2.42076E-05	-2.12480E-01
4	5	-5.29939E-02	9.91607E-06	-2.12722E-01
4	6	1.18971E-01	0.	-2.12621E-01
5	1	9.84245E-03	0.	0.
5	2	-9.84245E-03	-5.29435E-03	-6.08498E-02
5	3	-1.28493E-02	-1.21095E-02	-1.57063E-02
5	4	-2.27259E-02	-2.12726E-02	1.05189E-01
5	5	-4.61952E-02	-2.08247E-02	3.17217E-01
5	6	9.29235E-02	0.	5.97465E-01
6	1	-0.	0.	0.
6	2	0.	-9.18121E-03	2.15746E-02
6	3	0.	-2.19061E-02	1.12587E-01
6	4	0.	-4.46972E-02	3.32248E-01
6	5	0.	-9.29068E-02	7.79169E-01
6	6	0.	0.	1.68823E+00
7	1	-0.	0.	0.
7	2	0.	9.18121E-03	0.
7	3	0.	2.19061E-02	0.
7	4	0.	4.46922E-02	0.
7	5	0.	9.29068E-02	0.
7	6	0.	0.	0.

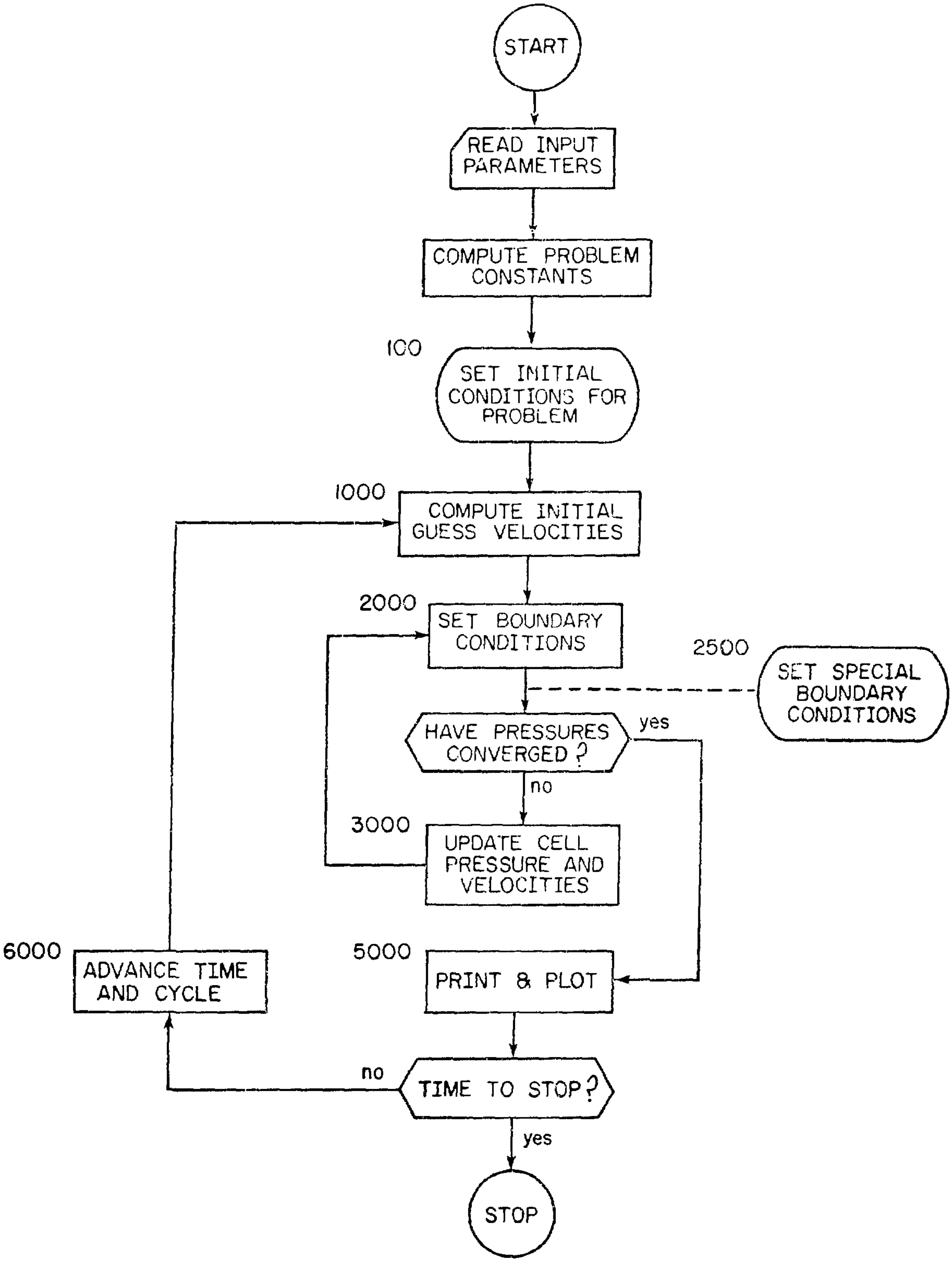
TABLE I (cont)

ITER= 1 TIME= 2.00000E-01 CYCLE= 10

I	J	U	V	P
1	1	-0.	0.	0.
1	2	0.	-1.64082E-02	0.
1	3	0.	-5.57385E-02	0.
1	4	0.	-1.11026E-01	0.
1	5	0.	-1.49229E-01	0.
1	6	0.	0.	0.
2	1	1.65994E-02	0.	0.
2	2	-1.65994E-02	1.64082E-02	-3.73524E-01
2	3	-3.91711E-02	5.57385E-02	-3.22456E-01
2	4	-5.57225E-02	1.11026E-01	-5.17154E-01
2	5	-3.73117E-02	1.49229E-01	-1.04456E+00
2	6	1.49129E-01	0.	-2.11053E+00
3	1	3.57785E-02	0.	0.
3	2	-3.57785E-02	1.95130E-02	-2.66565E-01
3	3	-7.53241E-02	5.55489E-02	-3.01012E-01
3	4	-9.44248E-02	9.42524E-02	-4.15525E-01
3	5	-3.91540E-02	9.59882E-02	-6.51856E-01
3	6	2.45255E-01	0.	-9.26518E-01
4	1	3.60260E-02	0.	0.
4	2	-3.60260E-02	4.66701E-04	-1.62416E-01
4	3	-7.56096E-02	6.37485E-04	-1.84047E-01
4	4	-9.58149E-02	1.58566E-03	-1.91328E-01
4	5	-3.91733E-02	1.27992E-03	-2.02372E-01
4	6	2.46284E-01	0.	-1.98434E-01
5	1	1.68412E-02	0.	0.
5	2	-1.68412E-02	-1.90222E-02	-7.63463E-02
5	3	-3.98025E-02	-5.58194E-02	-6.47406E-02
5	4	-5.67390E-02	-9.33258E-02	4.10255E-02
5	5	-3.73948E-02	-9.51798E-02	2.64831E-01
5	6	1.50961E-01	0.	5.53560E-01
6	1	-0.	0.	0.
6	2	0.	-1.66360E-02	-5.81084E-02
6	3	0.	-5.65414E-02	-4.14782E-02
6	4	0.	-1.13434E-01	1.54354E-01
6	5	0.	-1.50898E-01	6.66089E-01
6	6	0.	0.	1.78625E+00
7	1	-0.	0.	0.
7	2	0.	1.66360E-02	0.
7	3	0.	5.65414E-02	0.
7	4	0.	1.13434E-01	0.
7	5	0.	1.50898E-01	0.
7	6	0.	0.	0.

ITER= 1 TIME= 1.00000E+00 CYCLE= 50

I	J	U	V	P
1	1	-0.	0.	0.
1	2	0.	-1.68871E-02	0.
1	3	0.	-5.68156E-02	0.
1	4	0.	-1.12652E-01	0.
1	5	0.	-1.49692E-01	0.
1	6	0.	0.	0.
2	1	1.68871E-02	0.	0.
2	2	-1.68871E-02	1.68871E-02	-3.24199E-01
2	3	-3.99285E-02	5.68156E-02	-3.22245E-01
2	4	-5.67342E-02	1.12652E-01	-5.3794[illegible]E-01
2	5	-3.68426E-02	1.49692E-01	-1.04602E+00
2	6	1.49694E-01	0.	-2.12750E+00
3	1	3.68231E-02	0.	0.
3	2	-3.68231E-02	1.99360E-02	-2.57037E-01
3	3	-7.69072E-02	5.69945E-02	-3.01556E-01
3	4	-9.49218E-02	9.58827E-02	-4.15059E-01
3	5	-3.79152E-02	9.69567E-02	-6.52160E-01
3	6	2.46647E-01	0.	-9.26002E-01
4	1	3.78747E-02	0.	0.
4	2	-3.78747E-02	2.51663E-04	-1.62736E-01
4	3	-7.74345E-02	6.99216E-04	-1.84394E-01
4	4	-9.54871E-02	1.26459E-03	-1.91382E-01
4	5	-3.79086E-02	1.25796E-03	-2.02043E-01
4	6	2.47985E-01	0.	-1.98622E-01
5	1	1.71865E-02	0.	0.
5	2	-1.71865E-02	-1.98882E-02	-7.49210E-02
5	3	-4.05705E-02	-5.67523E-02	-6.44114E-02
5	4	-5.70320E-02	-9.52067E-02	4.55304E-02
5	5	-3.68684E-02	-9.62469E-02	2.64748E-01
5	6	1.51658E-01	0.	5.53195E-01
6	1	-0.	0.	0.
6	2	0.	-1.71865E-02	-5.61487E-02
6	3	0.	-5.77570E-02	-4.24557E-02
6	4	0.	-1.14790E-01	1.54212E-01
6	5	0.	-1.51658E-01	6.67544E-01
6	6	0.	0.	1.78811E+00
7	1	-0.	0.	0.
7	2	0.	1.71865E-02	0.
7	3	0.	5.77570E-02	0.
7	4	0.	1.14790E-01	0.
7	5	0.	1.51658E-01	0.
7	6	0.	0.	0.



START
READ INPUT PARAMETERS
COMPUTE PROBLEM CONSTANTS
100
SET INITIAL CONDITIONS FOR PROBLEM
1000
COMPUTE INITIAL GUESS VELOCITIES
2000
SET BOUNDARY CONDITIONS
2500
SET SPECIAL BOUNDARY CONDITIONS
HAVE PRESSURES CONVERGED ?
yes
no
3000
UPDATE CELL PRESSURE AND VELOCITIES
6000
ADVANCE TIME AND CYCLE
5000
PRINT & PLOT
no
TIME TO STOP?
yes
STOP

```
    RUN=LCM97  0                        75/02/11     17.42.42     T3BDNZZ3NT    PAGE NO. 1

             PROGRAM SOLA(INP,OUT,FILM,FSET10=INP,FSET9=OUT,FSET12=FILM)
  2          DIMENSION U(152,32),V(152,32),UN(152,32),VN(152,32),P(152,32),
            1XPUT(25),NAME(10)
  2          REAL LONG,NU
  2          INTEGER CYCLE,WL,WR,WT,WB
  2          READ 45, NAME      $      PRINT 35      $      PRINT45, NAME
      *
      C * * READ AND PRINT INITIAL INPUT DATA
      *
 22          READ 25,NUM,(XPUT(I),I=1,NUM)
 33          IBAR=XPUT(1) $ JBAR=XPUT(2) $ DELX=XPUT(3) $ DELY=XPUT(4)
 41          DELT=XPUT(5) $ NU=XPUT(6) $ CYL=XPUT(7)  $ EPSI=XPUT(8)
 47          DZRO=XPUT(9) $ GX=XPUT(10) $ GY=XPUT(11) $ UI=XPUT(12)
 55          VI=XPUT(13) $  VELMX=XPUT(14) $  TWFIN=XPUT(15) $ CWPRT=XPUT(16)
 63          CWPLT=XPUT(17) $ OMG=XPUT(18) $ ALPHA=XPUT(19) $ WL=XPUT(20)
 71          WR=XPUT(21) $ WT=XPUT(22) $ WB=XPUT(23)
 76          PRINT 50,(XPUT(I),I=1,NUM)
          25 FORMAT(6X,I2,/(4(6X,E12.5)))
          27 FORMAT(1H ,18X,10A8,1X,A10,2(1X,A8))
          35 FORMAT(1H1)
          44 FORMAT(6X*CYCLE= *I5,8X*T0= *1PE12.5,8X*T2= *E12.5,9X*ITER=*I5)
          45 FORMAT(10A8)
          46 FORMAT(1H+,80X*T=*,1PE10.3,4X*CYCLE=*,I4)
          47 FORMAT(6X*I*7X*J*12X*U*17X*V*18X*P*)
          48 FORMAT(4X,I3,5X,I3,3(6X,1PE12.5))
          49 FORMAT(6X*ITER= *I5,10X*TIME= *1PE12.5,10X*CYCLE= *I4)
          50 FORMAT(1H ,5X*IBAR= *1PE12.5/6X*JBAR= *E12.5/6X*DELX= *E12.5/
            16X*DELY= *E12.5/6X*DELT= *E12.5/8X*NU= *E12.5/7X*CYL= *E12.5/
            26X*EPSI= *E12.5/6X*DZRO= *E12.5/8X*GX= *E12.5/8X*GY= *E12.5/
            38X*UI= *E12.5/8X*VI= *E12.5/5X*VELMX= *E12.5/5X*TWFIN= *E12.5/
            45X*CWPRT= *E12.5/5X*CWPLT= *E12.5/7X*OMG= *E12.5/5X*ALPHA= *E12.5/
            58X*WL= *E12.5/8X*WR= *E12.5/8X*WT= *E12.5/8X*WB= *E12.5)
      *
      C * * COMPUTE CONSTANT TERMS AND INITIALIZE NECESSARY VARIABLES
      *
104          IMAX=IBAR+2
107          JMAX=JBAR+2
107          IM1=IMAX-1
107          JM1=JMAX-1
107          RDX=1.0/DELX
107          RDY=1.0/DELY
107          JM2=JMAX-2
107          IM2=IMAX-2
121          CALL GETQ(4LKJBN,JNM)
123          CALL DATE1(DAT)
125          CALL CLOCK1(CLK)
127          PRINT 27,NAME,JNM,DAT,CLK
143          T=0.
143          ITER=0
143          CYCLE=0
143          TWPRT=0.
143          TWPLT=0.
143          BETA= OMG/(2.*DELT*(RDX**2+RDY**2))
      *
      C * * SPECIAL INPUT DATA
```

```
      *
      *
      C * * SET CONSTANT TERMS FOR PLOTTING
      *
 143        LONG= FLOAT(IBAR)*DELX
 143        HIGH= FLOAT(JBAR)*DELY
 143        IYB=916
 161        IF(LONG.LE.(1.13556*HIGH))GOTO 300
 164        IXL=0
 164        IXR=1022
 164        IYT= INT(916.-HIGH*1022./LONG)
 172        GOTO 330
 173    300 X=LONG*450./HIGH
 173        IXL= INT(511.-X)
 173        IXR= INT(511.+X)
 173        IYT= 16
 202    330 CONTINUE
 202        VELMX1=  AMIN1(DELX,DELY)/VELMX
      *
      C * * SET INITIAL VELOCITY FIELD INTO U AND V ARRAYS
      *
 207        DO 560 I=2,IM1
 211        DO 560 J=2,JM1
 220        U(I,J)= UI
 220        V(I,J)= VI
 220    560 CONTINUE
 225        ASSIGN 5000 TO KRET
 226        GO TO 2000
      *
      C * * START CYCLE
      *
 226   1000 CONTINUE
 226        ITER=0
 226        FLG=1.
 231        ASSIGN 3000 TO KRET
      *
      C * * COMPUTE TEMPORARY U AND V
      *
 232        DO 1100 I=2,IM1
 233        DO 1100 J=2,JM1
 250        FUX=((UN(I,J)+UN(I+1,J))*(UN(I,J)+UN(I+1,J))+ALPHA*ABS(UN(I,J)+UN(
           1I+1,J))*(UN(I,J)-UN(I+1,J))-(UN(I-1,J)+UN(I,J))*(UN(I-1,J)+UN(I,J)
           2)-ALPHA*ABS(UN(I-1,J)+UN(I,J))*(UN(I-1,J)-UN(I,J)))/(4.*DELX)
 250        FUY=((VN(I,J)+VN(I+1,J))*(UN(I,J)+UN(I,J+1))
           1+ALPHA*ABS(VN(I,J)+VN(I+1,J))*(UN(I,J)-UN(I,J+1))
           2-(VN(I,J-1)+VN(I+1,J-1))*(UN(I,J-1)+UN(I,J))
           3-ALPHA*ABS(VN(I,J-1)+VN(I+1,J-1))*(UN(I,J-1)-UN(I,J)))/(4.*DELY)
 250        FUC=CYL*((UN(I,J)+UN(I+1,J))*(UN(I,J)+UN(I+1,J))+(UN(I-1,J)+UN(I,J
           1))*(UN(I-1,J)+UN(I,J))
           2+ALPHA*ABS(UN(I,J)+UN(I+1,J))*(UN(I,J)-UN(I+1,J))
           3+ALPHA*ABS(UN(I-1,J)+UN(I,J))*(UN(I-1,J)-UN(I,J)))
           4/(8.*DELX*FLOAT(I-1))
 250        FVX=((UN(I,J)+UN(I,J+1))*(VN(I,J)+VN(I+1,J))+ALPHA*ABS(UN(I,J)+UN(
           1I,J+1))*(VN(I,J)-VN(I+1,J))-(UN(I-1,J)+UN(I-1,J+1))*(VN(I-1,J)+VN(
           2I,J))-ALPHA*ABS(UN(I-1,J)+UN(I-1,J+1))*(VN(I-1,J)-VN(I,J)))/(4.*DE
```

```
          3LX)
 250       FVY=((VN(I,J)+VN(I,J+1))*(VN(I,J)+VN(I,J+1))+ALPHA*ABS(VN(I,J)+VN
          1(I,J+1))*(VN(I,J)-VN(I,J+1))-(VN(I,J-1)+VN(I,J))*(VN(I,J-1)+VN(I,J
          2))-ALPHA*ABS(VN(I,J-1)+VN(I,J))*(VN(I,J-1)-VN(I,J)))/(4.*DELY)
 250       FVC=CYL*((UN(I,J)+UN(I,J+1))*(VN(I,J)+VN(I+1,J))+(UN(I-1,J)+UN(I-1
          1,J+1))*(VN(I-1,J)+VN(I,J))+ALPHA*ABS(UN(I,J)+UN(I,J+1))*(VN(I,J)-V
          2N(I+1,J))+ALPHA*ABS(UN(I-1,J)+UN(I-1,J+1))*(VN(I-1,J)-VN(I,J)))
          3/(8.*DELX*(FLOAT(I-1)-0.5))
 250       VISX= NU*((UN(I+1,J)-2.*UN(I,J)+UN(I-1,J))/DELX**2+
          1             (UN(I,J+1)-2.*UN(I,J)+UN(I,J-1))/DELY**2
          2   +CYL*((UN(I+1,J)-UN(I-1,J))/(2.*DELX*DELX*FLOAT(I-1))
          3   -UN(I,J)/(DELX*FLOAT(I-1))**2))
 461       VISY= NU*((VN(I+1,J)-2.*VN(I,J)+VN(I-1,J))/DELX**2+
          1             (VN(I,J+1)-2.*VN(I,J)+VN(I,J-1))/DELY**2
          2    +CYL*(VN(I+1,J)-VN(I-1,J))/(2.*DELX*DELX*FLOAT(I-1.5)))
 461       U(I,J)= UN(I,J)+DELT*((P(I,J)-P(I+1,J))*RDX + GX-FUX-FUY-FUC+VISX)
 461       V(I,J)= VN(I,J)+DELT*((P(I,J)-P(I,J+1))*RDY + GY-FVX-FVY-FVC+VISY)
 527   1100 CONTINUE
      *
      C * * SET BOUNDARY CONDITIONS
      *
 534   2000 CONTINUE
 534        DO 2200 J=1,JMAX
 536        GO TO(2020,2040,2060,2080) WL
 553   2020 U(1,J)=0.0
 553        V(1,J)=V(2,J)
 555        GO TO 2100
 562   2040 U(1,J)=0.0
 562        V(1,J)=-V(2,J)
 564        GO TO 2100
 564   2060 IF(ITER.GT.0)GO TO 2100
 574        U(1,J)=U(2,J)
 574        V(1,J)=V(2,J)
 577        GO TO 2100
 605   2080 U(1,J)= U(IM2,J)     $     V(2,J)= V(IM1,J)
 605        V(1,J)= V(IM2,J)     $     P(2,J)= P(IM1,J)
 610        GO TO 2100
 611   2100 GO TO (2120,2140,2160,2180)WR
 626   2120 U(IM1,J)=0.0
 626        V(IMAX,J)=V(IM1,J)
 630        GO TO 2200
 636   2140 U(IM1,J)=0.0
 636        V(IMAX,J)=-V(IM1,J)
 640        GO TO 2200
 641   2160 IF(ITER.GT.0) GOTO 2200
 652        U(IM1,J)= U(IM2,J)*(IM2/IM1*CYL+(1.0-CYL))
 652        V(IMAX,J)=V(IM1,J)
 660        GO TO 2200
 670   2180 U(IM1,J)=U(2,J)
 670        V(IMAX,J)=V(3,J)
 672   2200 CONTINUE
 675        DO 2500 I=1,IMAX
 676        GOTO (2320,2340,2360,2380) WT
 713   2320 V(I,JMAX)=0.0
 713        U(I,JMAX)=U(I,JM1)
```

```
 715        GO TO 2400
 722   2340 V(I,JM1)=0.0
 722        U(I,JMAX)=-U(I,JM1)
 724        GO TO 2400
 724   2360 IF(ITER.GT.0) GOTO 2400
 736        V(I,JM1)=V(I,JM2)
 736        U(I,JMAX)=U(I,JM1)
 740        GO TO 2400
 745   2380 V(I,JM1)=V(I,2)
 745        U(I,JMAX)=U(I,3)
 747        GO TO 2400
 747   2400 GOTO (2420,2440,2460,2480) WB
 760   2420 V(I,1)=0.0
 760        U(I,1)=U(I,2)
 763        GO TO 2500
 764   2440 V(I,1)=0.0
 764        U(I,1)=-U(I,2)
 767        GO TO 2500
 767   2460 IF(ITER.GT.0) GO TO 2500
 773        V(I,1)=V(I,2)
 773        U(I,1)=U(I,2)
 776        GO TO 2500
1002   2480 V(I,1)= V(I,JM2)     $     U(I,2)= U(I,JM1)
1002        U(I,1)= U(I,JM2)     $     P(I,2)= P(I,JM1)
1005   2500 CONTINUE
      *
      C * * SPECIAL BOUNDARY CONDITIONS
      *
1010        IM2= IMAX-2
1011        DO 2800 I= 1,IM2
1021   2800 U(I,JMAX)=1.0
1023        GO TO KRET
1026   3000 CONTINUE
      *
      C * * HAS CONVERGENCE BEEN REACHED
      *
1026        IF(FLG.EQ.0.)GOTO 4000
1027        ITER=ITER+1
1031        IF(ITER.LT.500) GOTO 3050
1033        IF(CYCLE.LT.10) GO TO 4000
1035        T= 1.E+10
1037        GOTO 5000
1037   3050 FLG=0.0
      *
      C * * COMPUTE UPDATED CELL PRESSURE AND VELOCITIES
      *
1040        DO 3500 J=2,JM1
1042        DO 3500 I=2,IM1
1043        D=RDX*(U(I,J)-U(I-1,J))+RDY*(V(I,J)-V(I,J-1))+CYL*(U(I,J)
           1+U(I-1,J))/(2.*DELX*(FLOAT(I)-1.5))
1065        IF(ABS(D/DZRO).GE.EPSI)FLG=1.0
1102        DELP= -BETA*D
1102        P(I,J)=P(I,J)+DELP
1102        U(I,J)=U(I,J)+DELT*RDX*DELP
1102        U(I-1,J)=U(I-1,J)-DELT*RDX*DELP
```

```
1102          V(I,J)=V(I,J)+DELT*RDY*DELP
1102          V(I,J-1)=V(I,J-1)-DELT*RDY*DELP
1115    3500 CONTINUE
1122          GO TO 2000
1122    4000 CONTINUE
      *
      C * * PRINT AND PLOT
      *
1122    5000 CONTINUE
1122          IF(T.GT.0.)GOTO 5030
1125          WRITE(12,50)(XPUT(I),I=1,NUM)
1133    5030 CONTINUE
1133          PRINT 49,ITER,T,CYCLE
1145          IF(CYCLE.LE.0) GOTO 5100
1147          IF(T+1.E-6 .LT. TWPLT) GO TO 5600
1152          TWPLT=TWPLT+CWPLT*DELT
1154    5100 CONTINUE
1154          CALL ADV(1)
1156          CALL LINCNT(1)
1160          WRITE(12,49) ITER,T,CYCLE
1172          CALL LINCNT(3)
1174          WRITE(12,47)
1200          DO 5250 I=1,IMAX
1202          DO 5250 J=1,JM1
1203          WRITE(12,48)  I,J,U(I,J),V(I,J),P(I,J)
1225    5250 CONTINUE
      *
      C * * VELOCITY VECTOR PLOT
      *
1232          CALL ADV(1)
1234          CALL DGA(IXL,IXR,IYT,IYB,0.,LONG,HIGH,0.)
1244          CALL FRAME(IXL,IXR,IYT,IYB)
1247          CALL FRAME(IXL,IXR,IYT,IYB)
1252          CALL LINCNT(60)
1254          WRITE(12,27) NAME,JNM,DAT,CLK
1270          CALL LINCNT(62)
1272          WRITE(12,46)T,CYCLE
1302          DO 5500 I=2,IM1
1304          DO 5500 J=2,JM1
1311          XCC=DELX*(FLOAT(I)-1.5)
1311          YCC=DELY*(FLOAT(J)-1.5)
1311          UVEC=(U(I-1,J)+U(I,J))*0.5*VELMX1+XCC
1311          VVEC=(V(I,J-1)+V(I,J))*0.5*VELMX1+YCC
1326          CALL CONVRT(UVEC,IUVEC,0.,LONG,IXL,IXR)
1332          CALL CONVRT(VVEC,JVVEC,HIGH,0.,IYT,IYB)
1336          CALL CONVRT(XCC,IXCC,0.,LONG,IXL,IXR)
1342          CALL CONVRT(YCC,JYCC,HIGH,0.,IYT,IYB)
1346          CALL DRV(IXCC,JYCC,IUVEC,JVVEC)
1351    5500 CONTINUE
      *
      C * LIST VELOCITY AND PRESSURE FIELDS
      *
1356    5600 CONTINUE
1356          IF(CYCLE.LE.0) GO TO 5800
1360          IF(T+1.E-6.LT.TWPRT) GO TO 6000
```

```
1363          TWPRT=TWPRT+CWPRT*DELT
1365     5800 CONTINUE
1365          PRINT 35
1371          PRINT 27,NAME,JNM,DAT,CLK
1405          PRINT 49,ITER,T,CYCLE
1417          PRINT 47
1423          DO 5900 I= 1,IMAX
1425          DO 5900 J=1,JM1
1426          PRINT 48, I,J,U(I,J),V(I,J),P(I,J)
1450     5900 CONTINUE
        *
        C * * SET THE ADVANCE TIME VELOCITIES U AND V INTO THE UN AND VN ARRAYS
        *
1455     6000 CONTINUE
1455          DO 6100 I=1,IMAX
1457          DO 6100 J=1,JMAX
1466          UN(I,J)=U(I,J)
1466          VN(I,J)=V(I,J)
1466     6100 CONTINUE
        *
        C * * ADVANCE TIME T=T+DELT
        *
1474          T=T+DELT
1475          IF(T.GT.TWFIN)  GOTO 6500
1501          CYCLE=CYCLE+1
1502          GOTO 1000
1503     6500 STOP
1505          END
```

PROGRAM LENGTH INCLUDING I/O REQUEST TABLES -   SOLA
61576

STATEMENT NUMBER REFERENCES

LOCATION	GEN TAG	STMT NO	REFERENCES		
1516	C00012	25	23		
1522	C00016	27	130	1252	1367
1527	C00023	35	11	1363	
1531	C00025	44	NONE		
1540	C00034	45	3	15	
1542	C00036	46	1270		
1547	C00043	47	1172	1415	
1554	C00050	48	1200	1423	
1560	C00054	49	1131	1156	1403
1566	C00062	50	76	1122	
173	L00111	300	164		
203	L00115	330	172		
226	L00135	1000	1477		
534	L00162	2000	225	1117	
546	L00167	2020	542		
555	L00172	2040	543		
564	L00175	2060	544		
577	L00203	2080	545		

610	L00206	2100	554	563	565	576	607
620	L00211	2120	614				
630	L00214	2140	615				
640	L00217	2160	616				
660	L00225	2180	617				
671	L00227	2200	627	637	641	657	
705	L00236	2320	701				
714	L00241	2340	702				
723	L00244	2360	703				
737	L00252	2380	704				
746	L00255	2400	713	722	724	736	745
756	L00260	2420	752				
762	L00263	2440	753				
766	L00266	2460	754				
775	L00274	2480	755				
1003	L00276	2500	761	765	767	774	
1024	L00311	3000	230				
1035	L00322	3050	1030				
1120	L00346	4000	224	1024	1032	1034	
1120	L00346	5000	224	1024	1032	1034	
1131	L00354	5030	1121				
1152	L00364	5100	1143	1144			
1354	L00463	5600	1147				
1363	L00470	5800	1354	1355			
1453	L00517	6000	1360				
1500	L00541	6500	1475				

VARIABLE REFERENCES

LOCATION	GEN TAG	NAME		REFERENCES					
1730	V00042	ALPHA	\R	70	260	304	331	353	374
				427					
1731	V00062	BETA	\R	160	1077				
1732	V00055	CLK	\R	126	141	1263	1400		
1733	V00040	CWPLT	\R	65	1150				
1734	V00037	CXPRT	\R	64	1361				
1735	V00012	CYCLE	\I	147	1030	1140	1145	1165	1275
				1354	1412	1476			
1736	V00026	CYL	\R	46	335	433	453	477	650
				1057					
1737	V00105	D	\R	1062	1100				
1740	V00054	DAT	\R	124	137	1261	1376		
1741	V00106	DELP	\R	1076					
1742	V00025	DELT	\R	43	145	511	523	1071	1150
				1361	1472				
1743	V00023	DELX	\R	40	111	153	203	237	405
				435	450	462	474	1043	1310
1744	V00024	DELY	\R	42	113	156	205	241	452
				476	1313				
1745	V00030	DZRO	\R	51	1062				
1746	V00027	EPSI	\R	50	1064				
1747	V00074	FLG	\R	230	1024	1035	1070		
1750	V00077	FUC	\R	341	507				
1751	V00075	FUX	\R	272	505				
1752	V00076	FUY	\R	316	506				

1753	V00102	FVC	\R	4 2	521				
1754	V00120	FVX	\R	366	517				
1755	V00101	FVY	\R	410	520				
1756	V00031	GX	\R	52	504				
1757	V00032	GY	\R	54	516				
1760	V00063	HIGH	\R	160	174	1236	1331	1341	
1761	V00020	I	\I	210	211	222	232	233	457
				531	674	725	714	725	737
				756	762	770	775	1003	1010
				1040	1044	1071	1112	1177	1203
				1207	1214	1217	1225	1301	1303
				1351	1422	1426	1432	1437	1442
				1450	1454	1455	1467		
1762	V00021	IBAR	\I	35	105	152			
1763	V00043	IMAX	\I	106	110	117	623	633	643
				662	1003	1226	1451	1467	
1764	V00045	IM1	\I	113	222	531	620	630	542
				650	665	1113	1352		
1765	V00052	IM2	\I	121	577	644	1006	1012	
1766	V00057	ITER	\I	153	226	564	640	723	766
				1025	1134	1161	1406		
1767	V00113	IUVEC	\I	1324	1345				
1770	V00115	IXCC	\I	1334	1344				
1771	V00065	IXL	\I	166	200	1232	1242	1245	1326
				1336					
1772	V00066	IXR	\I	171	201	1232	1242	1245	1326
				1336					
1773	V00064	IYB	\I	162	1233	1243	1246	1332	1342
1774	V00067	IYT	\I	172	202	1233	1243	1246	1332
				1342					
1775	V00072	J	\I	211	233	456	526	535	547
				556	566	600	621	631	642
				660	671	1037	1046	1073	1115
				1200	1205	1210	1223	1302	1311
				1347	1423	1430	1433	1446	1455
1776	V00022	JBAR	\I	36	107	155			
1777	V00044	JMAX	\I	107	671	710	717	727	740
				1010	1457				
2000	V00046	JM1	\I	116	213	527	706	715	725
				742	1115	1223	1347	1446	
2001	V00051	JM2	\I	117	731	776			
2002	V00053	JNM	\I	122	135	1257	1374		
2003	V00114	JVVEC	\I	1330	1345				
2004	V00116	JYCC	\I	1340	1344				
2005	V00073	KRET	\I	225	231	1020			
2006	V00010	LONG	\R	156	163	173	1234	1325	1335
2007	A00007	NAME	\I	6	20	133	1255	1372	
2021	V00011	NU	\R	44	454				
2022	V00017	NUM	\I	26	30	101	1125		
2023	V00041	OMG	\R	66	151				
2024	A00005	P	\R	502	513	1100	514	502	1217
				1442					
13424	V00047	RDX	\R	114	144	503	1055	1073	
13425	V00050	RDY	\R	120	144	516	1056	1072	
13426	V00056	T	\R	146	1034	1120	1136	1144	1163

```
                                            1273        1355        1410        1471
  13427      V00036      TWFIN     \R        62         1473
  13430      V00061      TWPLT     \R       154         1145
  13431      V00060      TWPRT     \R       147         1356
  13432      A00001      U         \R       216          517         553         562         573         574
                                            604          605         626         636         656         711
                                            720          734         761         765         773         774
                                           1001         1002        1015        1103        1314        1462
                                            646          665         707         716         731         737
                                           1050         1207        1432         664         744         760
                                            764         1045        1105        1313
  25032      V00033      UI        \R        56          215
  25033      A00003      UN        \R       244          513        1464         246         246         267
                                            445          247         342         413         422
  36433      V00111      UVEC      \R      1323         1324
  36434      A00002      V         \R       216          526         554         563         574         575
                                            605          606         625         635         654         712
                                            721          735         757         763         771         772
                                           1000         1001        1107        1320        1461         551
                                            560          623         633         645         663         733
                                            741         1052        1214        1437         660         743
                                           1053         1111        1320
  50034      V00035      VELMX     \R        60          206
  50035      V00071      VELMX1    \R       207         1302
  50036      V00034      VI        \R        57          215
  50037      V00103      VISX      \R       456          510
  50040      V00104      VISY      \R       512          522
  50041      A00004      VN        \R       245          467         524        1462         337         411
                                            421          466         473         301         350         415
                                            423          466         472         272         363         470
                                            372          467         273
  61441      V00112      VVEC      \R      1323         1330
  61442      V00016      *B        \I        76          746
  61443      V00013      *L        \I        72          535
  61444      V00014      *R        \I        73          610
  61445      V00015      *T        \I        74          674
  61446      V00070      X         \R       173
  61447      V00107      XCC       \R      1312         1316        1334
  61450      A00006      XPUT      \R        30           34         101        1125          35          37
                                             40           41          43          45          46          47
                                             51           53          54          55          57          61
                                             62           63          65          67          70          71
                                             73           75
  61501      V00110      YCC       \R      1317         1321        1340

EXTERNAL REFERENCES
             GEN TAG     NAME      REFERENCES
             S00200      INPUTC    \I         5            7          10          25          27          32
                                             33
             S00300      OUTPTC    \R        13           14          17          21          22         100
                                            103          104         132         134         136         140
                                            142          143        1124        1127        1130        1133
                                           1135         1137        1141        1142        1160        1162
                                           1164         1166        1167        1174        1175        1202
                                           1204         1206        1213        1216        1221        1222
```

```
                                       1254      1256      1260      1262      1264      1265
                                       1272      1274      1276      1277      1365      1366
                                       1371      1373      1375      1377      1401      1402
                                       1405      1407      1411      1413      1414      1417
                                       1420      1425      1427      1431      1436      1441
                                       1444      1445
          S00400     GETQ      \R       123
          S00500     DATE1     \R       125
          S00600     CLOCK1    \R       127
          S00700     ACGOER    \R       541       613       700       751      1022
          S01000     ADV       \R      1153      1231
          S01100     LINCNT    \I      1155      1171      1251      1267
          S01200     DGA       \R      1235      1241
          S01300     FRAME     \R      1244      1247
          S01400     CONVRT    \R      1327      1333      1337      1343
          S01500     DRV       \R      1346
          S01600     STOP      \R      1501
          S01700     END       \R      1503
          S00100     Q8NTRY            NONE

START OF       CONSTANTS          TEMPORARIES           INDIRECTS          UNUSED COMPILER SPACE
                 1504                1671                  1730                     65700
```

II. SOLA-SURF — BASIC SOLUTION ALGORITHM FOR FLOWS BOUNDED BY CURVED SURFACES

We can easily modify the SOLA code to permit free or curved rigid surfaces across the top and bottom of the computational mesh. The principal restriction is that these surfaces must be definable by single-valued functions, for example, $y = H(x,t)$ for the top surface and $y = HB(x,t)$ for the bottom surface. Also, the slope of the surface must not exceed the cell aspect ratio $\delta y/\delta x$. Several examples illustrating different combinations of the curved surface options are described in Sec. B. The basic modifications required in SOLA to permit these more general boundary conditions are described next.

A. Modifications to Basic SOLA

Let H_i be the height of the top surface above the bottom of the mesh, as measured up the center of the i^{th} column of cells, and let HB_i be the corresponding height of the bottom surface. Dimension statements must be added to SOLA for H_i, HB_i and for their old time values HN_i, HBN_i. In addition, it is convenient to dimension for storage the j index of the cell containing the top surface JT_i and the bottom surface JB_i. An input number TB is 1.0 if the top surface is to be free and is 2.0 for a rigid curved boundary. Likewise, for the bottom boundary the input number BB is 1.0 for a free surface and 2.0 for a rigid surface.

Initial values of H_i, HB_i and corresponding JT_i, JB_i must be defined in the initial condition section (100 section) for each problem. Of course, if curved boundaries are not wanted H_i can be set equal to the height of the mesh and HB_i to zero. The corresponding input numbers TB and BB should then be set to zero, which indicates that those sections of the code used to update these boundaries can be omitted.

For some problems, when the bottom boundary is rigid, it is best to start with a hydrostatic pressure field. This is done in the setup after H_i has been defined.

$$p_{i,j} = -g_v\left[H_i - (j-1.5)\delta y\right] .$$

All DO LOOPS sweeping the mesh are arranged to run up columns starting with the far left and ending with the far right column. In each column the j index runs from the bottom boundary cell JB_i to the top boundary cell JT_i.

In the pressure iteration (3000 section) the top and bottom surface cells must be given special consideration to reflect the new boundary conditions. At a free surface the pressure must be zero (or at some specified value), whereas at a rigid boundary the normal fluid velocity must vanish.

First, consider the free surface condition for the top boundary (TB = 1.0). The surface cell pressure is chosen such that a linear interpolation between it and the pressure in the fluid cell below yields zero or an applied value p_s at the surface,* i.e.,

$$p_{i,JT} = (1-\eta)p_{i,JT-1} + \eta p_s ,$$

where

$$\eta = \delta y \Big/ \left[H_i - (JT-2.5)\delta y\right] .$$

When the bottom surface is free (BB = 1.0) a similar prescription is used for the surface cell pressure $p_{i,JB}$, except the interpolation is with the fluid cell above ($j = JB+1$) and H_i is replaced by HB_i.

$$p_{i,JB} = (1-\eta)p_{i,JB+1} + \eta p_s ,$$

where now

$$\eta = \delta y \Big/ \left[(JB-0.5)\ y - HB_i\right] .$$

If the bottom boundary is a rigid surface (BB = 2.0), the pressure in the surface cell is chosen to make the normal velocity at the surface zero. In difference form the outward normal velocity at HB_i is approximated by

---

*The applied pressure is generally a function of time and location along the surface that must be defined for each specific problem. In the code contained in Sec. C, p_s is assumed equal to zero and does not appear there explicitly. Therefore, when a nonzero p_s is desired the ηp_s term must be added to the surface cell pressure in the 3000 section of the code.

$$u_n = -\frac{1}{4\delta x}\left(u_{i,JB} + u_{i-1,JB}\right)\left(HB_{i+1} - HB_{i-1}\right)$$

$$+ \zeta v_{i,JB} + (1-\zeta) v_{i,JB-1} \quad ,$$

where

$$\zeta = \left[HB_i - (JB-2)\delta y\right]/\delta y \quad .$$

In the code $v_{i,JB}$ has been replaced by its value computed from Eq. (3), which is the boundary condition used at the top surface and the form needed to derive $\delta u_n/\delta p$ below. The velocities appearing in this expression are functions of the cell pressure so that $u_n = 0$ can be considered an implicit equation for $p_{i,JB}$. A Newton-Raphson type solution method is used to obtain a new estimate for $p_{i,JB}$ during each iteration pass. Specifically, the change added to $p_{i,JB}$ in each iteration is

$$\delta p = -u_n \Big/ \frac{\partial u_n}{\partial p} \quad ,$$

where the denominator is given by

$$\frac{\partial u_n}{\partial p} = \frac{\delta t}{\delta y}\left[1 + \frac{2\delta y^2}{\delta x^2}(1-\zeta)\right] \quad .$$

Similar expressions are used in the top cell j = JT for a rigid curved boundary (TB = 2.0).

For both the top and bottom surfaces, velocity boundary conditions are set in the boundary condition section (2000 section), after the regular boundary conditions but before the location reserved for special boundary conditions. These conditions, which are identical for both the free and rigid cases, are set by proceeding from left (i=2) to right (i=IM1). For each top surface cell the u-velocity on its right face is set equal to the u-velocity in the cell below if the cell to the right is empty. Also, the u-velocity in the cell above is set equal to the u-velocity in the surface cell. The v-velocity at the top of the surface cell is chosen to insure that the velocity divergence for the cell is zero. In difference form, for cell JT(i) these conditions are:

$$u_{i,JT} = u_{i,JT-1}, \quad \text{if } JT(i+1) < JT(i)$$

$$u_{i,JT+1} = u_{i,JT}$$

$$v_{i,JT} = v_{i,JT-1} - \frac{\delta y}{\delta x}\left(u_{i,JT} - u_{i-1,JT}\right)$$

$$- \frac{\xi \delta y}{2\delta x(i-1.5)}\left(u_{i,JT} + u_{i-1,JT}\right) \quad .$$

At the bottom surface cell JB(i) the corresponding conditions are,

$$u_{i,JB} = u_{i,JB+1}, \quad \text{if } JB(i+1) > JB(i)$$

$$u_{i,JB-1} = u_{i,JB}$$

$$v_{i,JB-1} = v_{i,JB} + \frac{\delta y}{\delta x}\left(u_{i,JB} - u_{i-1,JB}\right)$$

$$+ \frac{\xi \delta y}{2\delta x(i-1.5)}\left(u_{i,JB} + u_{i-1,JB}\right) \quad .$$

The simplicity of these boundary conditions results from the limitation that surface slopes not exceed the cell aspect ratio $\delta y/\delta x$.

In the case of a free top surface, a new surface configuration must be computed each cycle (4000 section) according to the kinematic equation

$$\frac{\partial H}{\partial t} + u\frac{\partial H}{\partial x} = v \quad ,$$

but only after convergence of the pressure iteration has been obtained. The difference equation used is,

$$H_i^{n+1} = H_i + \delta t \left\{ -\frac{1}{4\delta x}\left[(u_{i,JT} + u_{i-1,JT})\right.\right.$$

$$(H_{i+1} - H_{i-1}) - \gamma|u_{i,JT} + u_{i-1,JT}|$$

$$\left.(H_{i+1} - 2H_i + H_{i-1})\right] + h v_{i,JT}$$

$$\left. + (1-h)\, v_{i,JT-1}\right\} \quad .$$

where h is an interpolation length used to get the v-velocity at the surface position,

$$h = \left[H_i - (JT-2)\delta y\right]/\delta y \quad .$$

All quantities without superscripts are evaluated at time $n\delta t$. The constant γ is used for upstream differencing in analogy with α (it is often chosen equal to α).

When the bottom surface is free it is updated with a similar equation,

$$HB_i^{n+1} = HB_i + \delta t \left\{ -\frac{1}{4\delta x}\left[(u_{i,JB} + u_{i-1,JB})\right.\right.$$

$$(HB_{i+1} - HB_{i-1}) - \gamma|u_{i,JB} + u_{i-1,JB}|$$

$$\left(HB_{i+1} - 2HB_i + HB_{i-1}\right)\Big] + h_b \cdot v_{i,JB}$$

$$+ (1-h_b)\, v_{i,JB-1}\Big\}$$

with

$$h_b = \left[HB_i - (JB-2)\delta y\right]/\delta y \quad .$$

If the surface configurations are changed, the JT(i) and JB(i) indexes and the surface velocity boundary conditions must be reset in preparation for the next cycle.

B. Sample Problems

With the above, relatively minor modifications added to SOLA, many interesting problems can be investigated. The following examples illustrate some of the possibilities.

1. Interaction of Two Solitary Waves. A solitary wave is a single, finite amplitude wave that propagates without a change in shape. According to approximate analytic theories,[11] two solitary waves can interact nonlinearly without losing their identity. To study this phenomenon with a direct numerical simulation, two solitary waves moving toward one another were set up as initial conditions, as shown in Fig. 10. The mesh consisted of 150 cells in the x-direction and 10 cells in the y-direction (IBAR = 150, JBAR = 10). The top surface is to be free (TB = 1.0) with an undisturbed depth of 1.0. The bottom surface is flat and rigid (BB = 0.0 and HB_i = 0.0). The initial wave profiles and velocity distributions were generated using the second-order theory of Laitone.[12] The left wave has an initial amplitude of 0.25 and is moving to the right, whereas the right wave has an amplitude of 0.5 and is moving to the left. Figure 10 gives the remaining input parameters for this calculation.

Subsequent times in the evolution of the two waves are shown in Fig. 11. In the last frame the two waves have separated, are moving apart, and are closely approximating the original two waves, as predicted analytically. Some low-level fluctuation is caused by dispersive errors in the numerical approximations and by the fact that the initial conditions are from an approximate theory. There is also a slight amplitude fluctuation and change after interaction; the parting waves have average heights of 0.241 and 0.522. This change may be the result of nonlinear effects not included in the theoretical

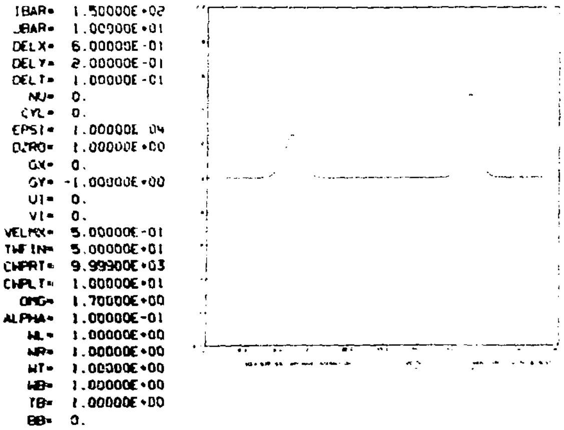


Fig. 10.
Initial free surface profile and input parameters used for interaction of two solitary waves.

analysis. However, more careful and more extensive calculations must be undertaken before this conclusion can be established.

2. Flow Over a Corrugated Bottom. To illustrate the curved wall boundary condition in SOLA-SURF we performed a calculation for a fluid with a free surface flowing over a wavy bottom. For small-amplitude bottom perturbations a linear solution for steady flow conditions can be found in Sec. 246 of Lamb's *Hydrodynamics*.[13] This problem, therefore, serves as a good test case. The calculational mesh was set up as follows: the mesh is 41 cells wide (δx=0.15) and 15 cells high (δy=0.1). The top boundary is a free surface (TB = 1.0) and the bottom boundary is rigid BB = 2.0 with a sinusoidal variation in height.

$$HB_i = H\left[1 + \cos(kx)\right] \quad ,$$

where $k = 2\pi/\lambda$ is the perturbation wave number, corresponding to the wave length λ = (IBAR-1)δx, and H = 0.05. The left and right boundaries of the mesh are periodic (WL = WR = 4). The mean depth of fluid relative to the mean bottom height is 0.90, and the fluid is initially moving with velocity UI = U_0 to the right. Gravity is down, $g_y = -1.0$. The surface profiles at t=10 and remaining input parameters are shown in Fig. 12.

When the flow is subcritical, $U_0^2 k/(-g_y) < 1$, the surface develops corrugations inverted with respect to those of the bottom, but for supercritical flow, $U_0^2 k/(-g_y) > 1$, the corrugations are in phase with those of the bottom.

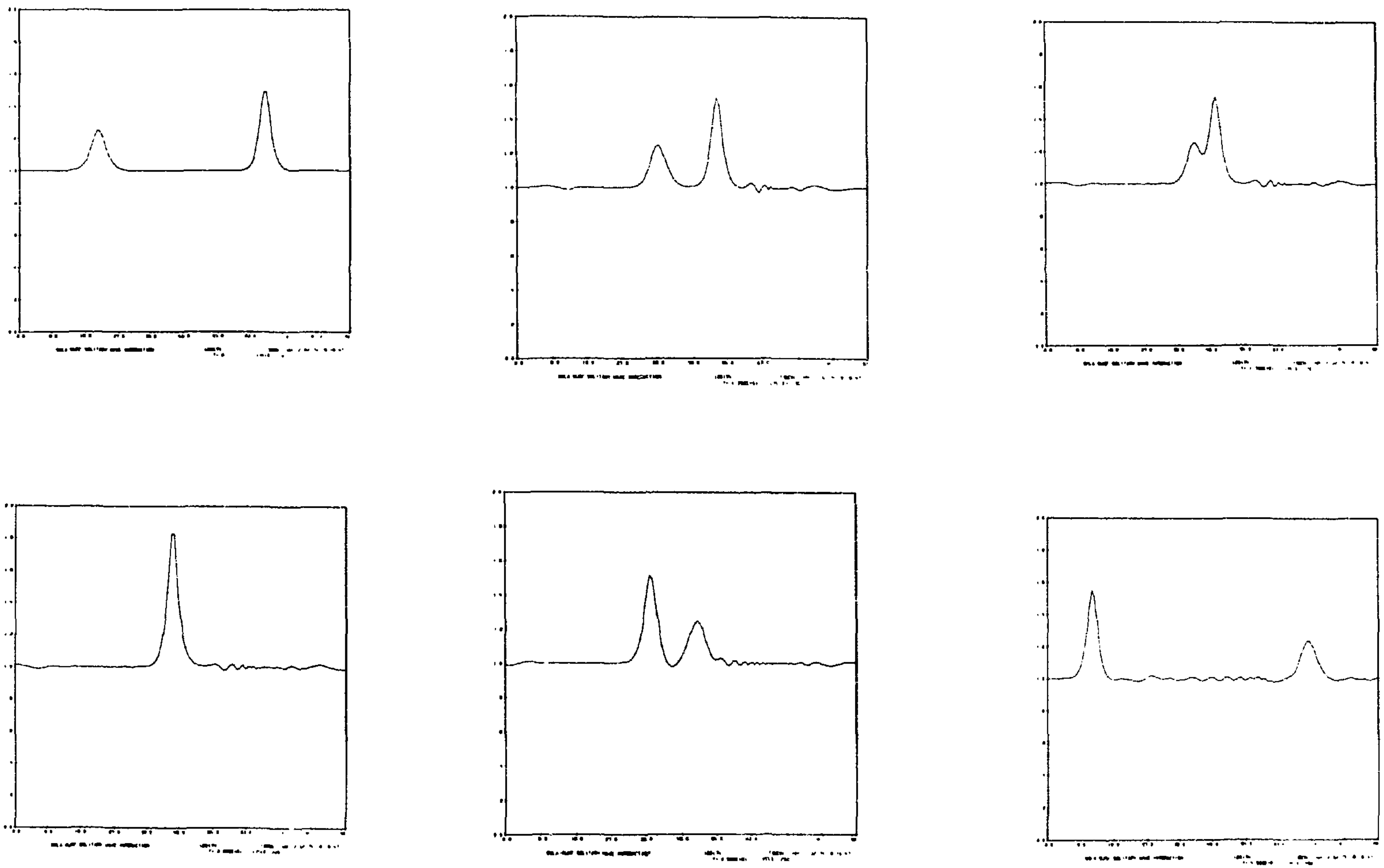


Fig. 11.
Sequence of surface profiles showing the interaction of two solitary waves. Left to right the times of each frame are 0.0, 13.0, 17.0, 20.0, 25.0, and 45.0.

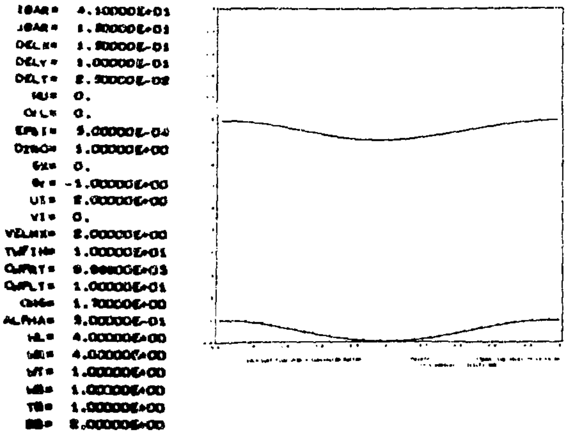


Fig. 12.
Steady-state top and bottom surface profiles and input parameters used in calculation of flow over a corrugated bottom.

The first calculation we attempted used a flat free surface and a uniform horizontal velocity $U_0 = +2.0$ for supercritical flow as initial conditions. The resulting flow developed free surface corrugations that periodically oscillated in phase and amplitude around the correct steady-state values. We believe that these results are reasonable, because any perturbation of the steady flow in an infinitely periodic system without dissipation has no mechanism to return it to the steady state. To check this, the calculation was repeated with the theoretical free surface profile, velocity, and pressure fields input as initial conditions. In this case, the calculated free surface amplitude remained steady at the theoretical value to within 3% of the surface amplitude shown in Fig. 12.

3. Flow About a Planing Surface. An interesting and useful variation of the curved surface treatment in SOLA-SURF is to let only portions of the surface be a rigid boundary. For example, the fluid flowing about a planing body like a surfboard or hydrofoil is confined only underneath the body, but is free elsewhere. To model this type of flow we must introduce a test in the pressure iteration (3000 section) to decide where to apply the free boundary condition (zero pressure) or the confined boundary condition (zero normal velocity at the body).

For a flat planing surface we know that a forward moving jet or splash is usually produced at the leading edge of the flow.[14] This cannot be modeled in SOLA-SURF because of the restriction that the surface must remain a single-valued function of the horizontal coordinate. However, in gravity-dominated situations the principal fluid resistance experienced by a planing body is due to the generation of a train of trailing waves and is not significantly influenced by the forward jet.

A calculation in which the forward jet is ignored can be set up as follows. To avoid a sudden transition in boundary conditions at the free surface, the surface cell pressure condition is modified to be a linear combination of the rigid and free condition; i.e., the pressure is chosen to make the following expression zero.

$$F = \kappa\left[p_{i,JT} - p_{i,JT-1}\left(\frac{h-\frac{1}{2}}{h+\frac{1}{2}}\right)\right] - (1-\kappa)\,\frac{\delta y}{\delta t}\,u_n \quad ,$$

where $h = [H_i - (JT-2)\,\delta y]/\delta y$ and u_n is the normal velocity at the surface. The minus sign for the second term was chosen to insure that F would be a monotonic function of the surface cell pressure. When κ is unity, $F=0$ is the free-surface boundary condition, and when κ is zero, $F=0$ corresponds to the usual rigid boundary condition, $u_n = 0$. In this problem the interpolation factor κ was chosen to be

$$\kappa = \begin{cases} \kappa_o \equiv 2(H - H_i)/\delta y, & \text{if } 0 < \kappa_o < 1 \\ 0, & \text{if } \kappa_o < 0 \\ 1, & \text{if } \kappa_o > 1 \quad , \end{cases}$$

where H is the height of the planing body at the center of the i^{th} column of cells. If this transition between the two kinds of surface boundary conditions is not used, the fluid at the leading edge of the body will periodically bounce off the rigid surface, because of large forward pressure gradients produced every few cycles when the fluid passes from a free surface region with approximately zero pressure to a confined fluid region with large positive pressure.

When inserting the condition $F = 0$ in the pressure iteration scheme, the surface cell pressure is computed using a Newton-Raphson type approximation in which the pressure change δp in the given iteration pass is given by

$$\delta p = -F \Big/ \frac{\partial F}{\partial p}$$

with

$$\frac{\partial F}{\partial p} = \kappa + (1-\kappa)(1 + 2h\,\delta y^2/\delta x^2) \quad .$$

When the top surface profile is advanced in time for this problem, a special test must also be inserted into the 4000 section to prevent H_i from exceeding the height of the planing body. Using the above modified boundary condition, we constructed a mesh with IBAR = 60, JBAR = 15, $\delta x=0.1$, and $\delta y=0.1$. The initial undisturbed fluid height was 1.15 in front of the body. Behind the body, where a flat plate tilted 9.5° with respect to the horizontal, the initial height was 0.95, corresponding to the elevation of the trailing edge of the plate. With respect to the plate the fluid is initially moving uniformly to the left with speed 0.6364. The mesh boundary conditions are WL = 3, WT = WB = WR = 1. Uniform inflow along the right mesh boundary was specified in the special boundary condition section (2500 section). Because of this special right-wall velocity condition we must reset the free-surface-velocity boundary conditions for the last cell (IM1, JT) next to the wall. This had been set in the 2000 section assuming a zero horizontal velocity at the right wall. Calculational parameters are listed in Fig. 13, which also shows the initial top surface profile (exaggerated by a factor of 4 in the vertical direction). Figure 14 shows the development of the surface profile in time. At the plate's leading edge the free surface initially climbs upward, but eventually reaches the steady profile shown in Fig. 14's last frame. The free surface behind the plate also settles down to a steady profile, in this case, without waves. The lack of a trailing wave pattern may be the result of not having a long enough mesh or possibly a result of the continuative outflow condition used at the left boundary.

```
  IBAR=  6.00000E+01
  JBAR=  1.50000E+01
  DELX=  1.00000E-01
  DELY=  1.00000E-01
  DELT=  2.50000E-02
    NU=  0.
   CYL=  0.
  EPSI=  2.00000E-04
  DZRO=  1.00000E+00
    GX=  0.
    GY= -5.00000E-01
    UI= -6.36400E-01
    VI=  0.
 VELMX=  1.50000E+00
 TWFIN=  2.50000E+01
 CWPRT=  2.00000E+01
 CWPLT=  2.00000E+01
   OMG=  1.70000E+00
 ALPHA=  2.50000E-01
    WL=  3.00000E+00
    WR=  2.00000E+00
    WT=  1.00000E+00
    WB=  1.00000E+00
```

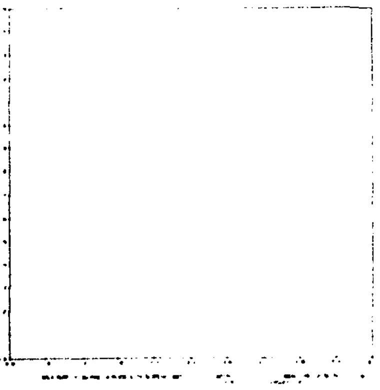


Fig. 13.
Initial top surface profile and input parameters for calculation of flow about a planing surface.

This example shows how simple methods can often be built up in easy steps to perform increasingly complex problems. Many other variations and modifications can also be imagined. In the next problem, for example, this partially rigid and partially free top-boundary method is used to compute the wave field generated by a floating body.

4. Waves Generated by a Floating Body in Forced Heave. In contrast to the previous problem, the partially confined, partially free top-boundary condition is here used in connection with a rigid boundary (the floating body) moving relative to the computing mesh. The principal, new modification required is that the normal velocity at the confining top surface be allowed to assume nonzero values.

The floating body used is a right circular cylinder, with axis oriented vertically, undergoing forced heave. Axisymmetric coordinates are used (CYL=1.0) with the mesh and cylinder arrangement as shown in Fig. 15.

In the SOLA-SURF boundary treatment, surface slopes are not to exceed $\delta y/\delta x$. We eliminate this restraint at the side of the cylinder, however, by aligning the vertical side with an i=constant mesh line and by adding boundary conditions that set to zero all u-velocities along the side of the cylinder. This is done by adding to the special boundary condition section (2500 section) the statement,

$$u_{IR,j} = 0.0 \quad \text{for } j = JR, JMAX ,$$

where IR corresponds to the interior cell along the side of the cylinder and JR = $JT_i + 1$ for i = IR. The forced heaving of the body is periodic and defined by

$$H = H_0 - A \sin(\omega t) ,$$

in which H is the height of the bottom of the body above the mesh floor, H_0 = 0.75, A = 0.10, and $\omega = 2\pi/\tau$ with $\tau=2.0$.

In the pressure iteration (3000 section) the boundary conditions for the top surface are that the surface pressure be zero (free surface condition) for $i > 6$ and that u_n be equal to the body velocity given by

$$\frac{dH}{dt} = - A\omega \cos(\omega t)$$

for $i \leq 6$.

Figure 15 lists the remaining parameters used for the calculation shown in Fig. 16. Note that vorticity is generated at the bottom corner of the body; therefore, this problem could not be solved analytically with a potential flow calculation, even for low-amplitude displacements.

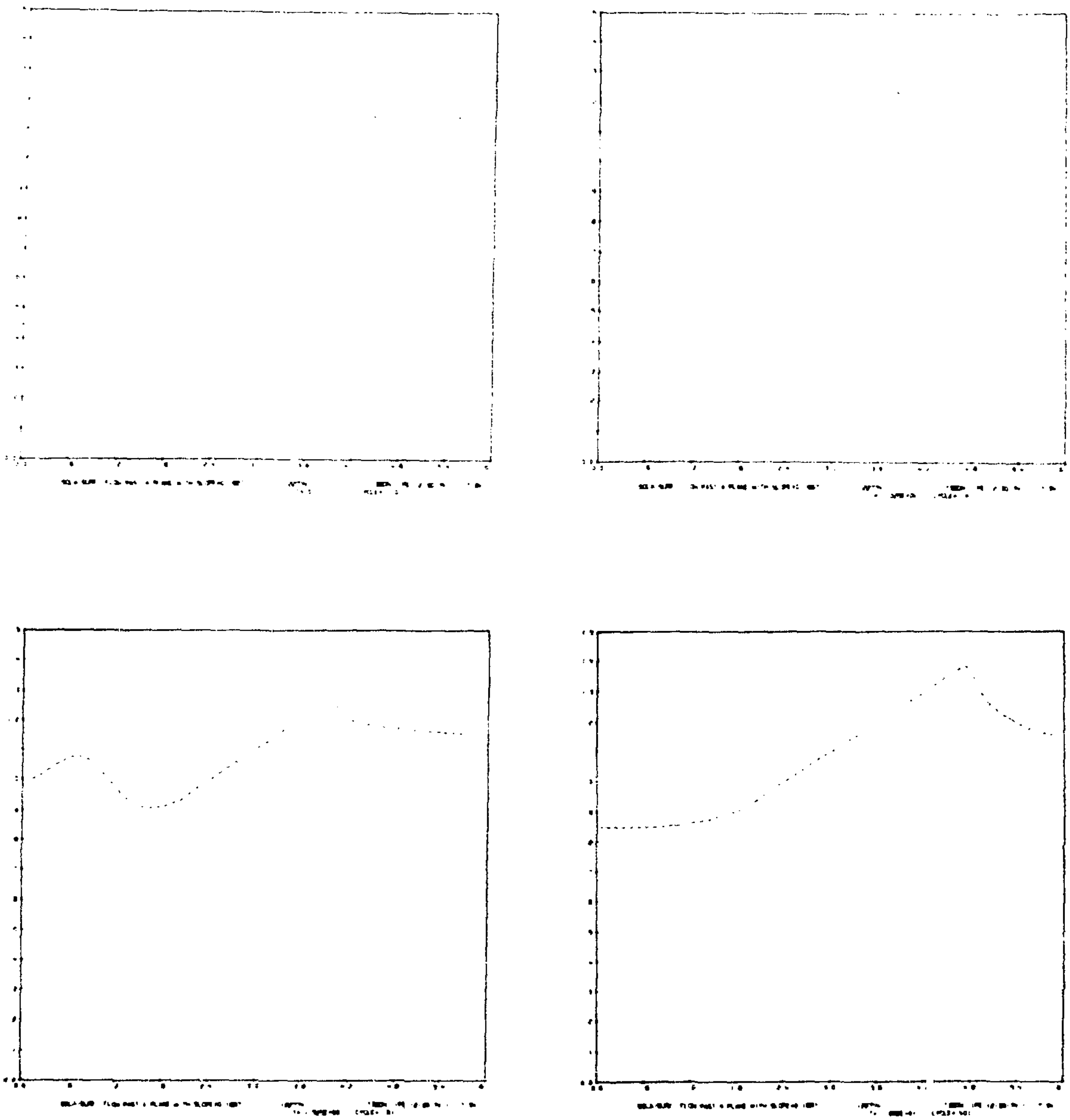


Fig. 14.

Sequence of surface profiles as flow develops about a planing surface. Left to right profiles are at times 0.0, 1.025, 1.525, and 10.025. The flow has reached steady state by the last frame.

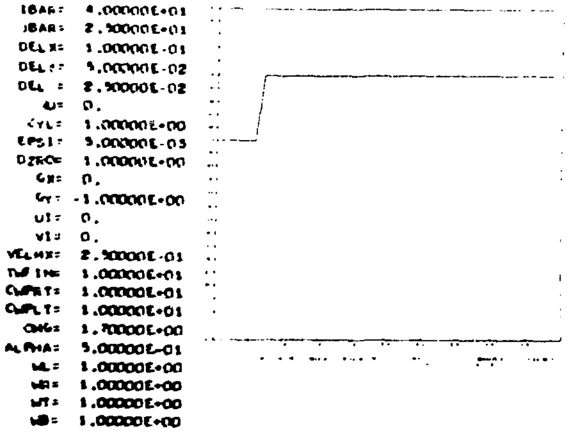


Fig. 15.
Initial free surface profile around cylinder and input parameters for the calculation of a floating cylinder in forced heave. Side of cylinder is vertical but appears sloping because line is drawn between surface height values at neighboring cell centers.

Waves radiating out from the body rapidly lose amplitude because of their radial expansion. In addition to the wave field, calculations of this sort can be used to obtain added mass and damping coefficients for floating bodies.

5. Free-Floating Body. Instead of forcing a periodic body motion, we now wish to have the body dynamically interact with the fluid. The body is to be initially raised above its equilibrium position and allowed to fall, starting from rest. Its subsequent motion will consist of damped oscillations as waves are generated and radiated away. The only difference between this calculation and the previous one is that the velocity of the body is not prescribed *a priori* but determined from the equation of motion

$$\frac{dV_B}{dt} = g_y + f/M ,$$

where f is the total force exerted on the base of the body by the instantaneous fluid pressure and M is the mass of the body. Thus, it seems that we need only replace the previously specified body velocity with that calculated by the above ordinary differential equation. There is, however, one difficulty with this procedure. The pressures computed in the code, when used to accelerate the body, lead to an unstable body motion. To correct this, the body equation of motion must be implicitly coupled with the pressure iteration. This is done by computing a new V_B, for use in the pressure iteration, at the end of

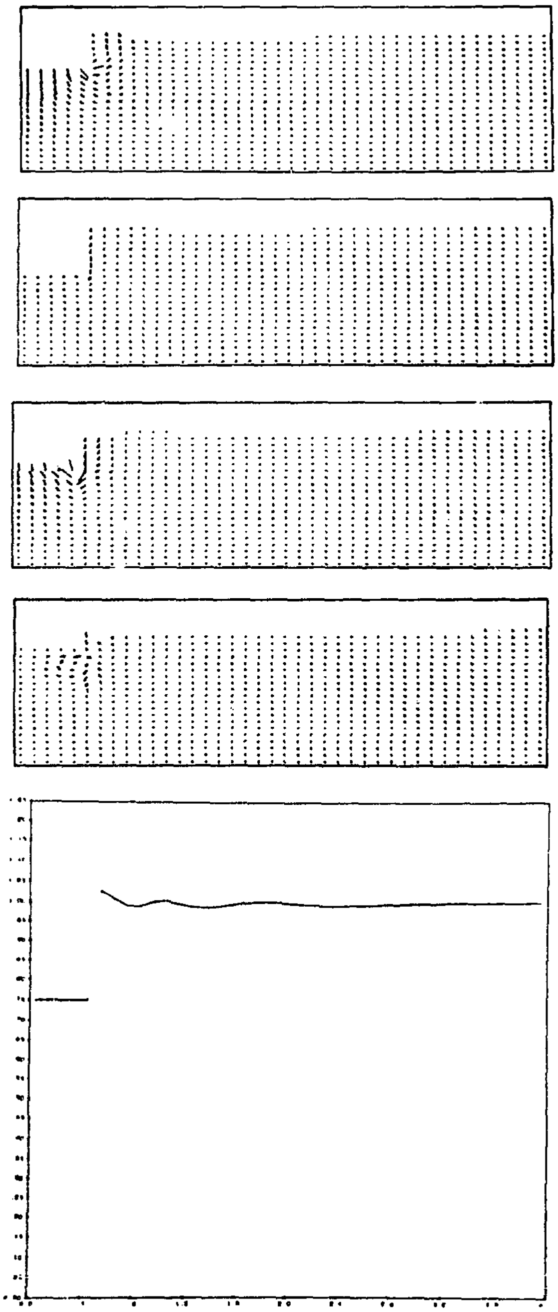


Fig. 16.
Surface profile at t = 8.0 and velocity fields at quarter periods as generated by cylinder undergoing forced heave.

each iteration pass using the most updated pressure values available. If the V_B calculation is inserted into the special boundary condition section (2500 section) this will be done automatically. Because of the dependence of V_B on the new pressures, however, we must also modify the relaxation factor $\partial F/\partial p$ described in subsection 3 to have for each cell (i,JT) under the cylinder the additional term

$$\frac{2\pi}{M}\,\delta x^2(i-1.5)[H-(JT-2.5)\delta y] \quad .$$

Figure 17 shows body height as a function of time computed in this way. The pressures used to accelerate the body may also be used to compute added mass and damping coefficients. Unfortunately, we know of no theoretical or experimental data for this or the previous problem to make comparisons.

C. Details of the SOLA-SURF Program

A conceptual flow chart and FORTRAN listing of the SOLA-SURF code is included here. Comparing this flow chart with SOLA's shows that the only significant differerence is section 4000, which is used to update the free surface position. All input parameters are identical in the two codes, except SOLA-SURF has some additional parameters associated with the surface options. The parameters are defined as:

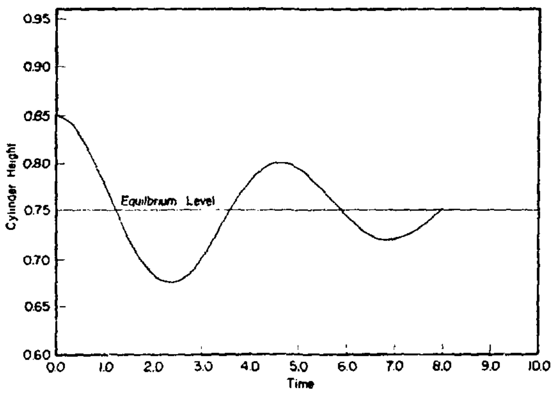


Fig. 17.
Elevation of free-floating cylinder vs time, shows damping of motion as waves are radiated away.

GAMMA = γ = controls the amount of donor cell fluxing in kinematic equations for free surface position (1.0 for full donor cell differencing and 0.0 for centered differencing),

TB = top boundary definition (0.0 for top boundary coincident with the top mesh boundary, 1.0 for a free surface, and 2.0 for a rigid boundary), and

BB = bottom boundary definition (0.0 for bottom boundary coincident with the bottom mesh boundary, 1.0 for a free surface, and 2.0 for a rigid boundary).

REFERENCES

1. F. H. Harlow, "Numerical Methods for Fluid Dynamics — An Annotated Bibliography," Los Alamos Scientific Laboratory report LA-4281 (December 1969).

2. 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).

3. C. W. Hirt and J. P. Shannon, "Free-Surface Stress Conditions for Incompressible-Flow Calculations," J. Comput. Phys. **2**, 403-411 (1968).

4. R. K.-C. Chan and R. L. Street, "A Computer Study of Finite-Amplitude Water Waves," J. Comput. Phys. **6**, 68-94 (1970); R. K.-C. Chan and R. L. Street, "SUMMAC — A Numerical Model for Water Waves," Department of Civil Engineering, Standford University, Stanford, CA, Technical Report No. 135 (1970).

5. B. D. Nichols and C. W. Hirt, "Improved Free Surface Boundary Conditions for Numerical Incompressible-Flow Calculations," J. Comput. Phys. **8**, 434-448 (1971).

6. J. A. Viecelli, "A Method for Including Arbitrary External Boundaries in the MAC Incompressible Fluid Computing Technique," J. Comput.

Phys. 4, 543-551 (1969); "A Computing Method for Incompressible Flows Bounded by Moving Walls," J. Comput. Phys. 8, 119-143 (1971).

7. A. A. Amsden and F. H. Harlow, "The SMAC Method: A Numerical Technique for Calculating Incompressible Fluid Flows," Los Alamos Scientific Laboratory report LA-4370 (May 1970).

8. P. I. Nakayama and N. C. Romero, "Numerical Method for Almost Three-Dimensional Incompressible Fluid Flow and a Simple Internal Obstacle Treatment," J. Comput. Phys. 8, 230-240 (1971).

9. C. W. Hirt, "Heuristic Stability Theory for Finite-Difference Equations," J. Comput. Phys. 2, 339-355 (1968).

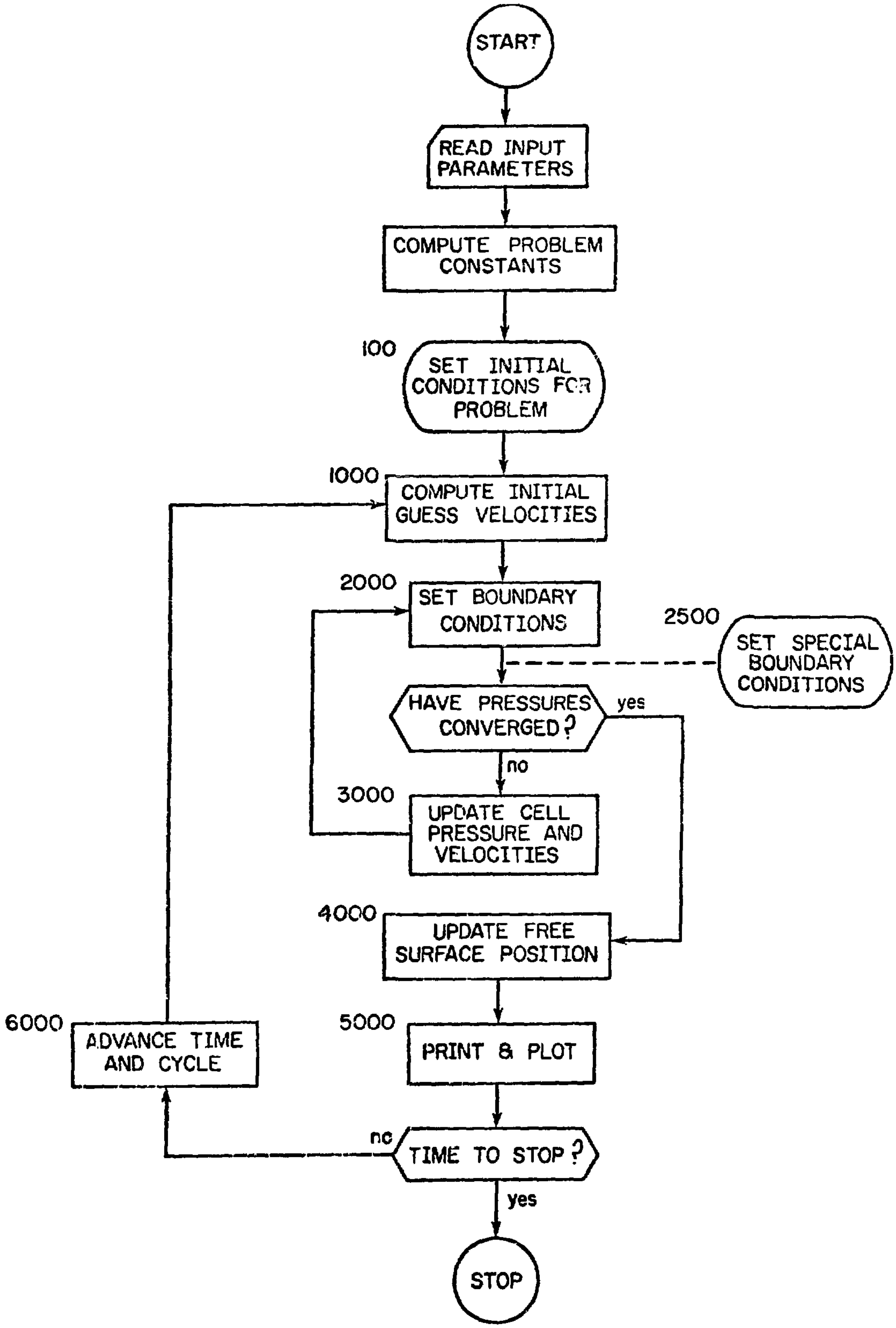
10. A. E. Vardy, M.I.G. Bloor, and J. A. Fox, "Capsular Flow in Pipelines," J. Fluid Mech. 56, 49-59 (1972).

11. N. J. Zabusky and C. J. Galvin, "Shallow-Water Waves, the Korteweg-deVries Equation and Solitons," J. Fluid Mech. 47, 811-824 (1971).

12. R. L. Wiegel, *Oceanographical Engineering* (Prentice-Hall, Inc., Englewood Cliffs, NJ, 1964), pp. 65-67.

13. H. Lamb, *Hydrodynamics* (Dover Publications, New York, 1945), pp. 409-410.

14. L. I. Sedov, *Two-Dimensional Problems in Hydrodynamics and Aerodynamics* (Interscience Publishers, New York, 1965) pp. 238-244.


START
READ INPUT PARAMETERS
COMPUTE PROBLEM CONSTANTS
100
SET INITIAL CONDITIONS FOR PROBLEM
1000
COMPUTE INITIAL GUESS VELOCITIES
2000
SET BOUNDARY CONDITIONS
2500
SET SPECIAL BOUNDARY CONDITIONS
HAVE PRESSURES CONVERGED?
yes
no
3000
UPDATE CELL PRESSURE AND VELOCITIES
4000
UPDATE FREE SURFACE POSITION
5000
PRINT & PLOT
6000
ADVANCE TIME AND CYCLE
TIME TO STOP?
no
yes
STOP

```
          PROGRAM SOLASOR(INP,OUT,FILM,FSET10=INP,FSET9=OUT,FSET12=FILM)
  2       DIMENSION U(202,24),V(202,24),P(202,24),W(202,24),F(202,24),
         1XPUT(25),R(202),RS(202),JT(202),JB(202),RB(202),RBS(202),XX(202),
         2NAME(10)
  2       REAL LONG,NU
  2       INTEGER CYCLE,AL,AR,AT,AB
  2       READ 45, NAME       $      PRINT 35      $      PRINT45, NAME
      *
      C * * READ AND PRINT INITIAL INPUT DATA
      *
 22       READ 25,NUM,(XPUT(I),I=1,NUM)
 33       IBAR=XPUT(1) $ JBAR=XPUT(2) $ DELX=XPUT(3) $ DELY=XPUT(4)
 41       DELT=XPUT(5) $ NU=XPUT(6) $ CYL=XPUT(7)  $ EPSI=XPUT(8)
 47       OZRO=XPUT(9) $ GX=XPUT(10) $ GY=XPUT(11) $ UI=XPUT(12)
 55       VI=XPUT(13) $  VELMX=XPUT(14) $  TWFIN=XPUT(15) $ CWPRT=XPUT(16)
 63       CWPLT=XPUT(17) $ OMG=XPUT(18) $ ALPHA=XPUT(19) $ GAMMA=XPUT(20)
 71       AL=XPUT(21) $ AR=XPUT(22) $ AT=XPUT(23) $ AB=XPUT(24)
 77       TB=XPUT(25) $ RB=XPUT(26)
102       PRINT 50,(XPUT(I),I=1,NUM)
      25 FORMAT(6X,I2,/(4(6X,E12.5)))
      27 FORMAT(1H ,18X,10A8,1X,A10,2(1X,A8))
      35 FORMAT(1H1)
      44 FORMAT(6X*CYCLE= *I5,8X*T= *1PE12.5,8X*T2= *E12.5,9X*ITER=*I5)
      45 FORMAT(10A8)
      46 FORMAT(1H+,82X*T=*,1PE10.3,4X*CYCLE=*,I4)
      47 FORMAT(6X*I*7X*J*12X*U*17X*V*18X*P*16X*H*11X*SUR CELL*9X*BOT CELL*
        1)
      48 FORMAT(4X,I3,5X,I3,4(6X,1PE12.5),2(6X,I6))
      49 FORMAT(6X*ITER= *I5,10X*TIME= *1PE12.5,12X*CYCLE= *I4,5X*FVOL=*
        1 E12.5)
      50 FORMAT(1H ,5X*IBAR= *1PE12.5/6X*JBAR= *E12.5/6X*DELX= *E12.5/
        16X*DELY= *E12.5/6X*DELT= *E12.5/8X*NU= *E12.5/7X*CYL= *E12.5/
        26X*EPSI= *E12.5/6X*OZRO= *E12.5/8X*GX= *E12.5/8X*GY= *E12.5/
        38X*UI= *E12.5/8X*VI= *E12.5/5X*VELMX= *E12.5/5X*TWFIN= *E12.5/
        45X*CWPRT= *E12.5/5X*CWPLT= *E12.5/7X*OMG= *E12.5/5X*ALPHA= *E12.5/
        55X*GAMMA= *E12.5/8X*AL= *E12.5/8X*AR= *E12.5/8X*AT= *E12.5/
        68X*AB= *E12.5/8X*TB= *E12.5/8X*RB= *E12.5)
      *
      C * * COMPUTE CONSTANT TERMS AND INITIALIZE NECESSARY VARIABLES
      *
111       IMAX=IBAR+2
114       JMAX=JBAR+2
114       IM1=IMAX-1
114       JM1=JMAX-1
114       RDX=1.0/DELX
114       RDY=1.0/DELY
114       JM2=JMAX-2
114       IM2=IMAX-2
114       ITB= INT(TB+1.E-10)
114       IRB= INT(RB+1.E-10)
132       CALL SETID(ALKJOB,JNM)
134       CALL DATE1(DAT)
136       CALL CLOCK1(CLK)
140       PRINT 27,NAME,JNM,DAT,CLK
154       T=0.
```

```
154         ITER=0
154         CYCLE=0
154         TWPRT= CWPRT*DELT
154         TWPLT=0.
154         BETA= OMG/(2.*DELT*(RDX**2+RDY**2))
      *
      C * * SPECIAL INPUT DATA
      *
      *
      C * * DETERMINE SLOPED BOUNDARY LOCATION
      *
154         HBHT= 0.0
167         DO 240 I= 2,IM1
175         HB(I)= HBHT
175         JB(I)= INT(HB(I)*RDY+1.E-8) + 2
175     240 CONTINUE
204         HB(1)= HB(2)
204         HB(IMAX)= HB(IM1)
204         JB(1)= JB(2)
204         JB(IMAX)= JB(IM1)
      *
      C * * COMPUTE INITIAL TOP SURFACE CONFIGURATION
      *
210         FLHT= 1.0
214         DO 260 I=2,IM1
216         H(I)= FLHT
216         JT(I)= INT(H(I)*RDY+1.E-8) + 2
223         IF(JT(I).GT.JM1) JT(I)= JM1
227     260 CONTINUE
233         H(1)= H(2)
233         H(IMAX)= H(IM1)
233         JT(1)= JT(2)
233         JT(IMAX)= JT(IM1)
      *
      C * * CALCULATE HYDROSTATIC PRESSURE
      *
241         DO 290 I=2,IM1
244         JT1=JT(I)
244         JB1= JB(I)
246         IF(IBB.EQ.1) GO TO 282
251         DO 280 J= JB1,JT1
262         P(I,J)= -GY*(H(I)-(FLOAT(J)-1.5)*DELY)
262     280 CONTINUE
270         GO TO 290
270     282 CONTINUE
270         DO 285 J= JB1,JT1
302         P(I,J)= GY*((FLOAT(J)-1.5)*DELY-HB(I))
302     285 CONTINUE
307     290 CONTINUE
      *
      C * * CALCULATE POSITION OF THE CENTER OF EACH VERTICAL COLUMN OF CELLS
      *
312         DO 320 I=1,IMAX
320         XX(I)=(FLOAT(I)-1.5)*DELX
320     320 CONTINUE
```

```
*
C * * SET CONSTANT TERMS FOR PLOTTING
*
        LONG= FLOAT(IBAR)*DELX
        HIGH= FLOAT(JBAR)*DELY
        IYB=916
        IF(LONG.LE.(1.13556*HIGH))GOTO 320
        IXL=0
        IXR=1022
        IYT= INT(916.-HIGH*1022./LONG)
        GOTO 330
  320   X= LONG*450./HIGH
        IXL= INT(511.-X)
        IXR= INT(511.+X)
        IYT=16
  330   CONTINUE
        VELMX1=AMIN1(DELX,DELY)/VELMX
*
C * * SET INITIAL VELOCITY FIELD INTO U AND V ARRAYS
*
        DO 560 I=2,IM1
        JT2=JT(I)+1
        JB2=JB(I)-1
        DO 560 J=JB2,JT2
        V(I,J)= VI
        U(I,J)= UI
  560   CONTINUE
        ASSIGN 4260 TO KRET
        GO TO 2000
*
C * * START CYCLE
*
 1000   CONTINUE
        ITER=0
        FLG=1.
        ASSIGN 3000 TO KRET
*
C * * COMPUTE TEMPORARY U AND V
*
        DO 1100 I=2,IM1
        JT1= JT(I)
        JB1= JB(I)
        DO 1100 J= JB1,JT1
        FUX=((UN(I,J)+UN(I+1,J))*(UN(I,J)+UN(I+1,J))+ALPHA*ABS(UN(I,J)+UN(
       1I+1,J))*(UN(I,J)-UN(I+1,J))-(UN(I-1,J)+UN(I,J))*(UN(I-1,J)+UN(I,J)
       2)-ALPHA*ABS(UN(I-1,J)+UN(I,J))*(UN(I-1,J)-UN(I,J)))/(4.*DELX)
        FUY=((VN(I,J)+VN(I+1,J))*(UN(I,J)+UN(I,J+1))
       1+ALPHA*ABS(VN(I,J)+VN(I+1,J))*(UN(I,J)-UN(I,J+1))
       2-(VN(I,J-1)+VN(I+1,J-1))*(UN(I,J-1)+UN(I,J))
       3-ALPHA*ABS(VN(I,J-1)+VN(I+1,J-1))*(UN(I,J-1)-UN(I,J)))/(4.*DELY)
        FUC=CYL*((UN(I,J)+UN(I+1,J))*(UN(I,J)+UN(I+1,J))+(UN(I-1,J)+UN(I,J
       1))*(UN(I-1,J)+UN(I,J))
       2+ALPHA*ABS(UN(I,J)+UN(I+1,J))*(UN(I,J)-UN(I+1,J))
       3+ALPHA*ABS(UN(I-1,J)+UN(I,J))*(UN(I-1,J)-UN(I,J)))
       4/(8.*DELX*FLOAT(I-1))
```

```
427          FVX=((UN(I,J)+UN(I,J+1))*(VN(I,J)+VN(I+1,J))+ALPHA*ABS(UN(I,J)+UN(
            1I,J+1))*(VN(I,J)-VN(I+1,J))-(UN(I-1,J)+UN(I-1,J+1))*(VN(I-1,J)+VN(
            2I,J))-ALPHA*ABS(UN(I-1,J)+UN(I-1,J+1))*(VN(I-1,J)-VN(I,J)))/(4.*DE
            3LX)
427          FVY=((VN(I,J)+VN(I,J+1))*(VN(I,J)+VN(I,J+1))+ALPHA*ABS(VN(I,J)+VN
            1(I,J+1))*(VN(I,J)-VN(I,J+1))-(VN(I,J-1)+VN(I,J))*(VN(I,J-1)+VN(I,J
            2))-ALPHA*ABS(VN(I,J-1)+VN(I,J))*(VN(I,J-1)-VN(I,J)))/(4.*DELY)
427          FVC=CYL*((UN(I,J)+UN(I,J+1))*(VN(I,J)+VN(I+1,J))+(UN(I-1,J)+UN(I-1
            1,J+1))*(VN(I-1,J)+VN(I,J))+ALPHA*ABS(UN(I,J)+UN(I,J+1))*(VN(I,J)-V
            2N(I+1,J))+ALPHA*ABS(UN(I-1,J)+UN(I-1,J+1))*(VN(I-1,J)-VN(I,J)))
            3/(8.*DELX*(FLOAT(I-1)-0.5))
427          VISX= NU*((UN(I+1,J)-2.*UN(I,J)+UN(I-1,J))/DELX**2+
            1             (UN(I,J+1)-2.*UN(I,J)+UN(I,J-1))/DELY**2
            2   +CYL*((UN(I+1,J)-UN(I-1,J))/(2.*DELX*DELX*FLOAT(I-1))
            3   -UN(I,J)/(DELX*FLOAT(I-1))**2))
640          VISY= NU*((VN(I+1,J)-2.*VN(I,J)+VN(I-1,J))/DELX**2+
            1             (VN(I,J+1)-2.*VN(I,J)+VN(I,J-1))/DELY**2
            2    +CYL*(VN(I+1,J)-VN(I-1,J))/(2.*DELX*DELX*FLOAT(I-1.5)))
640          U(I,J)= UN(I,J)+DELT*((P(I,J)-P(I+1,J))*RDX + GX-FUX-FUY-FUC+VISX)
640          V(I,J)= VN(I,J)+DELT*((P(I,J)-P(I,J+1))*RDY + GY-FVX-FVY-FVC+VISY)
706     1100 CONTINUE
      *
      C * * SET BOUNDARY CONDITIONS
      *
713     2030 CONTINUE
715          HN(1)= HN(2)
715          HN(IMAX)= HN(IM1)
715          JT(1)=JT(2)
715          JT(IMAX)=JT(IM1)
715          HBN(1)= HBN(2)
715          HBN(IMAX)= HBN(IM1)
715          JB(1)= JB(2)
715          JB(IMAX)= JB(IM1)
731          DO 2200 J=1,JMAX
733          GO TO(2020,2040,2060,2080) WL
750     2020 U(1,J)=0.0
750          V(1,J)=V(2,J)
752          GO TO 2100
757     2040 U(1,J)=0.0
757          V(1,J)=-V(2,J)
761          GO TO 2100
761     2060 IF(ITER.GT.0)GO TO 2100
771          U(1,J)=U(2,J)
771          V(1,J)=V(2,J)
774          GO TO 2100
1003    2080 U(1,J)= U(IM2,J)      $     V(2,J)= V(IM1,J)
1003         V(1,J)= V(IM2,J)      $     P(2,J)= P(IM1,J)
1003         HN(1)= HN(IM2)
1003         JT(1)= JT(IM2)
1003         HBN(1)= HBN(IM2)
1003         JB(1)= JB(IM2)
1014         GO TO 2100
1014    2100 GO TO (2120,2140,2160,2180)WR
1031    2120 U(IM1,J)=0.0
1031         V(IMAX,J)=V(IM1,J)
```

```
1033          GO TO 2200
1041     2140 U(IM1,J)=0.0
1041          V(IMAX,J)=-V(IM1,J)
1043          GO TO 2200
1044     2160 IF(ITER.GT.0) GOTO 2200
1055          U(IM1,J)= U(IM2,J)*(IM2/IM1*CYL+(1.0-CYL))
1055          V(IMAX,J)=V(IM1,J)
1064          GO TO 2200
1075     2180 U(IM1,J)=U(2,J)
1075          V(IMAX,J)=V(3,J)
1075          HN(IM1)= HN(2)
1075          JT(IM1)= JT(2)
1075          HN(IMAX)= HN(3)
1075          JT(IMAX)= JT(3)
1075          HBN(IM1)= HBN(2)
1075          JB(IM1)= JB(2)
1075          HBN(IMAX)= HBN(3)
1075          JB(IMAX)= JB(3)
1112     2200 CONTINUE
1115          DO 2500 I=1,IMAX
1117          JT1= JT(I)
1117          JB1= JB(I)
1122          GOTO (2320,2340,2360,2380) WT
1137     2320 V(I,JM1)=0.0
1137          U(I,JMAX)=U(I,JM1)
1141          GO TO 2400
1146     2340 V(I,JM1)=0.0
1146          U(I,JMAX)=-U(I,JM1)
1150          GO TO 2400
1150     2360 IF(ITER.GT.0) GOTO 2400
1162          V(I,JM1)=V(I,JM2)
1162          U(I,JMAX)=U(I,JM1)
1164          GO TO 2400
1171     2380 V(I,JM1)=V(I,2)
1171          U(I,JMAX)=U(I,3)
1173          GO TO 2400
1173     2400 GOTO (2420,2440,2460,2480) WB
1204     2420 V(I,1)=0.0
1204          U(I,1)=U(I,2)
1207          GO TO 2500
1210     2440 V(I,1)=0.0
1210          U(I,1)=-U(I,2)
1213          GO TO 2500
1213     2460 IF(ITER.GT.0) GO TO 2500
1217          V(I,1)=V(I,2)
1217          U(I,1)=U(I,2)
1222          GO TO 2500
1226     2480 V(I,1)= V(I,JM2)     $     U(I,2)= U(I,JM1)
1226          U(I,1)= U(I,JM2)     $     P(I,2)= P(I,JM1)
1231     2500 CONTINUE
        *
        C * * FREE SURFACE AND SLOPED BOUNDARY CONDITIONS
        *
1234     2640 CONTINUE
1234          IF(ITH.EQ.0 .AND. IBH.EQ.0) GO TO 2650
```

```
1242          DO 2620 I= 2,IM1
1244          JT1= JT(I)
1244          JB1= JB(I)
1246          IF(ITB.EQ.0) GO TO 2610
1250          IF(JT(I+1).LT.JT(I)) U(I,JT1)= U(I,JT1-1)
1262          V(I,JT1)= V(I,JT1-1)-DELY*RDX*(U(I,JT1)-U(I-1,JT1))
             1 -CYL*DELY*0.5*(U(I,JT1)+U(I-1,JT1))/((FLOAT(I)-1.5)*DELX)
1262          U(I,JT1+1)= U(I,JT1)
1277     2610 CONTINUE
1277          IF(IBB.EQ.0) GO TO 2620
1300          IF(JB(I+1).GT.JB(I)) U(I,JB1)= U(I,JB1+1)
1313          V(I,JB1-1)= V(I,JB1)+DELY*RDX*(U(I,JB1)-U(I-1,JB1))
             1 +CYL*DELY*0.5*(U(I,JB1)+U(I-1,JB1))/((FLOAT(I)-1.5)*DELX)
1313          U(I,JB1-1)= U(I,JB1)
1330     2620 CONTINUE
1333     2650 CONTINUE
        *
        C * * SPECIAL BOUNDARY CONDITIONS
        *
1333          GO TO KRET
1336     3000 CONTINUE
        *
        C * * HAS CONVERGENCE BEEN REACHED
        *
1336          IF(FLG.EQ.0.)GOTO 4000
1337          ITER=ITER+1
1341          IF(ITER.LT.500) GOTO 3050
1343          IF(CYCLE.LT.10) GO TO 4000
1345          T= 1.E+10
1347          GOTO 4000
1347     3050 FLG=0.0
        *
        C * * COMPUTE UPDATED CELL PRESSURE AND VELOCITIES
        *
1350          DO 3500 I= 2,IM1
1353          JT1= JT(I)
1353          JB1= JB(I)
1356          DO 3500 J= JB1,JT1
1357          IF(J.NE.JB1 .AND. J.NE.JT1) GO TO 3200
1366          IF(J.EQ.JT1 .AND. ITB.EQ.1) GO TO 3100
1374          IF(J.EQ.JB1 .AND. IBB.EQ.2) GO TO 3060
1403          IF(J.EQ.JT1 .AND. ITB.EQ.2) GO TO 3070
1412          IF(J.EQ.JB1 .AND. IBB.EQ.1) GO TO 3150
1421          GO TO 3200
1421     3060 CONTINUE
1430          VTM= RDY*(HB(I)-(J-2)*DELY)
1430          VBM= RDY*((J-1)*DELY-HB(I))
1430          F=-0.25*RDX*(HB(I+1)-HB(I-1))*(U(I,J)+U(I-1,J))+V(I,J)*VTM
             1+VBM*(V(I,J)+DELY*RDX*(U(I,J)-U(I-1,J)))
1430          DFDP= DELT*RDY*(VTM+VBM) + 2.*DELY*RDX*RDX*DELT*VBM
1430          DELP= -F/DFDP
1465          GO TO 3300
1465     3070 CONTINUE
1471          VTM= RDY*(H(I)-(J-2)*DELY)
1471          F= -0.25**RDX*(H(I+1)-H(I-1))*(U(I,J)+U(I-1,J))+V(I,J-1)
```

```
          1 =VTM*DELY*RDX*(U(I,J)-U(I-1,J))
1471        DFDP= -DELT*RDY*(1.0+2.0*VTM*DELY**2 * RDX**2)
1471        DELP= -F/DFDP
1525        GO TO 3300
1526   3100 CONTINUE
1531        PETA= DELY/(HN(I)-(FLOAT(JT1)-2.5)*DELY)
1531        DELP= (1.0-PETA)*P(I,JT1-1) - P(I,JT1)
1541        GO TO 3300
1542   3150 CONTINUE
1545        PETA= DELY/((FLOAT(JB1)-1.5)*DELY - HB(I))
1545        DELP= (1.0-PETA)*P(I,JB1+1)-P(I,JB1)
1555        GO TO 3300
1556   3200 CONTINUE
1556        D=RDX*(U(I,J)-U(I-1,J))+RDY*(V(I,J)-V(I,J-1))+CYL*(U(I,J)
          1+U(I-1,J))/(2.*DELX*(FLOAT(I)-1.5))
1600        IF(ABS(D/D2RO).GE.EPSI)FLG=1.0
1606        DELP= -BETA*D
1620   3300 P(I,J)=P(I,J)+DELP
1620        U(I,J)=U(I,J)+DELT*RDX*DELP
1620        U(I-1,J)=U(I-1,J)-DELT*RDX*DELP
1620        V(I,J)=V(I,J)+DELT*RDY*DELP
1620        V(I,J-1)=V(I,J-1)-DELT*RDY*DELP
1627   3500 CONTINUE
1634        GO TO 2000
1635   4000 CONTINUE
     *
     C * * COMPUTE NEW SURFACE POSITION
     *
1635        IF(ITB.NE.1) GO TO 4200
1637        DO 4100 I=2,IM1
1650        JT1= JT(I)
1650        HV= RDY*(HN(I)-FLOAT(JT1-2)*DELY)
1650        UAV= 0.5*(U(I-1,JT1) + U(I,JT1))
1650        H(I)= HN(I)+DELT*(HV*V(I,JT1)+(1.0-HV)*V(I,JT1-1)
          1       -0.5*RDX*(UAV*HN(I+1)+GAMMA*ABS(UAV)*(HN(I)-HN(I+1))
          2       -UAV*HN(I-1)-GAMMA*ABS(UAV)*(HN(I-1)-HN(I))))
1650   4100 CONTINUE
1705   4200 CONTINUE
     *
     C * * COMPUTE NEW POSITION FOR BOTTOM SURFACE
     *
1705        IF(IBM.NE.1) GO TO 4230
1707        DO 4220 I= 2,IM1
1720        JB1= JB(I)
1720        HBV= RDY*(HBN(I)-FLOAT(JB1-2)*DELY)
1720        UAV= 0.5*(U(I-1,JB1)+U(I,JB1))
1720        HB(I)= HBN(I)+DELTA(HBV*V(I,JB1)+(1.0-HBV)*V(I,JB1-1)
          1       -0.5*RDX*(UAV*HBN(I+1)+GAMMA*ABS(UAV)*(HBN(I)-HBN(I+1))
          2       -UAV*HBN(I-1)-GAMMA*ABS(UAV)*(HBN(I-1)-HBN(I))))
1720   4220 CONTINUE
1755   4230 CONTINUE
     *
     C * * CALCULATE CELL IN WHICH SURFACE IS LOCATED AND UPDATE ARRAY
     *
1755        DO 4250 I=2,IM1
```

```
1757          JT(I)= INT(H(I)*RDY+1.0E-8) + 2
1764          IF(JT(I).GT.JM1) JT(I)= JM1
1767          JB(I)= INT(HB(I)*RDY+1.0E-8) + 2
1774    4250 CONTINUE
1777          ASSIGN 4280 TO KRET
2000          GO TO 2600
2002    4280 CONTINUE
       *
       C * * CALCULATE TOTAL FLUID VOLUME
       *
2003          FVOL=0.0
2011          DO 4300 I=2,IM1
2014          ADELX= (CYL*6.28318*(FLOAT(I)-1.5)*DELX + (1.0-CYL))*DELX
2014          FVOL= FVOL + (H(I)-HB(I))*ADELX
2014    4300 CONTINUE
2024          FLX=0.0
2024          IF(WL.LT.3)GO TO 4345
2027          JTF= JT(2)-1
2027          JBF= JB(2)+1
2032          DO 4340 J=JBF,JTF
2043          FLX=FLX+U(1,J)*DELT*DELY
2043    4340 CONTINUE
2050          HDIF= H(1)-FLOAT(JT(1)-2)*DELY + FLOAT(JB(1)-1)*DELY-HB(1)
2050          FLX=FLX+HDIF*U(1,JT1)*DELT
2063    4345 CONTINUE
2063          IF(WR.LT.3) GOTO 4355
2067          JTF= JT(IM1)-1
2067          JBF= JB(IM1)+1
2072          DO 4350 J= JBF,JTF
2103          FLX=FLX-U(IM1,J)*DELT*DELY
2103    4350 CONTINUE
2111          HDIF= H(IM1)-FLOAT(JT(IM1)-2)*DELY + FLOAT(JB(IM1)-1)*DELY-HB(IM1)
2111          FLX=FLX-HDIF*U(IM1,JT1)*DELT
2124    4355 CONTINUE
2124          IF(WT.LT.3) GO TO 4365
2127          DO 4360 I=2,IM1
2143          ADELX= (CYL*6.28318*(FLOAT(I)-1.5)*DELX + (1.0-CYL))*DELX
2143          FLX= FLX - V(I,JM1)*DELT*ADELX
2143    4360 CONTINUE
2153    4365 CONTINUE
2153          IF(WB.LT.3)  GOTO 4375
2156          DO 4370 I=2,IM1
2170          ADELX= (CYL*6.28318*(FLOAT(I)-1.5)*DELX + (1.0-CYL))*DELX
2170          FLX= FLX + V(I,1)*DELT*ADELX
2170    4370 CONTINUE
2200    4375 CONTINUE
2200          FVOL=FVOL+FLX
       *
       C * * PRINT AND PLOT
       *
2202    5020 CONTINUE
2202          IF(T.GT.0.)GOTO 5030
2205          WRITE(12,50)(XPUT(I),I=1,NUM)
2213    5030 CONTINUE
2213          PRINT 49,ITER,T,CYCLE,FVOL
```

```
      IF(CYCLE.LE.0) GOTO 5100
      IF(T+1.E-6 .LT. TWPLT) GO TO 5600
      TWPLT=TWPLT+CWPLT*DELT
 5100 CONTINUE
      CALL ADV(1)
      CALL LINCNT(1)
      WRITE(12,49) ITER,T,CYCLE,FVOL
      CALL LINCNT(3)
      WRITE(12,47)
      DO 5250 I=1,IMAX
      JH1= JH(I)
      JT1= JT(I)
      JT2= JT(I)+1
      JB2= JB(I)-1
      DO 5250 J= JB2,JT2
      WRITE(12,48)  I,J,U(I,J),V(I,J),F(I,J),H(I),JT1,JH1
 5250 CONTINUE
*
C * * FREE SURFACE CONFIGURATION PLOTS
*
      CALL SPLOT(1,IBAR,XX(2),H(2),44,0)
      CALL LINCNT(60)
      WRITE(12,27) NAME,JNM,DAT,CLK
      CALL LINCNT(62)
      WRITE(12,46)T,CYCLE
      CALL SPLOT(1,IBAR,XX(2),HB(2),44,0)
      CALL LINCNT(60)
      WRITE(12,27) NAME,JNM,DAT,CLK
      CALL LINCNT(62)
      WRITE(12,46) T,CYCLE
      CALL ADV(1)
      CALL LINCNT(60)
      WRITE(12,27) NAME,JNM,DAT,CLK
      CALL LINCNT(62)
      WRITE(12,46) T, CYCLE
      CALL FRAME(80,1020,0,910)
      CALL FRAME(80,1020,0,910)
      CALL DGA(80,1020,0,910,0.,LONG,HIGH,0.0)
      CALL SELIN(JBAR,01)
      CALL SBLIN(10,01)
      CALL PLOT(IBAR,XX(2),1,H(2),1,42,1)
      CALL PLOT(IBAR,XX(2),1,HB(2),1,44,1)
*
C * * VELOCITY VECTOR PLOT
*
      CALL ADV(1)
      CALL DGA(IXL,IXR,IYT,IYB,0.,LONG,HIGH,0.)
      CALL FRAME(IXL,IXR,IYT,IYB)
      CALL FRAME(IXL,IXR,IYT,IYB)
      CALL LINCNT(60)
      WRITE(12,27) NAME,JNM,DAT,CLK
      CALL LINCNT(62)
      WRITE(12,46)T,CYCLE
      DO 5500 I=2,IM1
      JT1= JT(I)
```

```
2570         JB1= JB(I)
2573         DO 5500 J= JB1,JT1
2600         XCC=DELX*(FLOAT(I)-1.5)
2603         YCC=DELY*(FLOAT(J)-1.5)
2606         UVEC=(U(I-1,J)+U(I,J))*0.5*VELMX1+XCC
2610         VVEC=(V(I,J-1)+V(I,J))*0.5*VELMX1+YCL
2615         CALL CONVRT(UVEC,IUVEC,0.,LONG,IXL,IXR)
2621         CALL CONVRT(VVEC,JVVEC,HIGH,0.,IYT,IYB)
2625         CALL CONVRT(XCC,IXCC,0.,LONG,IXL,IXR)
2631         CALL CONVRT(YCC,JYCC,HIGH,0.,IYT,IYB)
2635         CALL DRV(IXCC,JYCC,IUVEC,JVVEC)
2640   5500 CONTINUE
      *
      C * * LIST VELOCITY, PRESSURE, AND SURFACE POSITION
      *
2645   5600 CONTINUE
2645         IF(CYCLE.LE.0) GO TO 5800
2647         IF(T+1.E-6.LT.TWPRT) GO TO 6000
2652         TWPRT=TWPRT+CWPRT*DELT
2654   5800 CONTINUE
2654         PRINT 35
2660         PRINT 27,NAME,JNM,DAT,CLK
2674         PRINT 49,ITER,T,CYCLE
2706         PRINT 47
2712         DO 5900 I= 1,IMAX
2715         JT1= JT(I)
2715         JB1= JB(I)
2715         JT2= JT(I)+1
2715         JB2= JB(I)-1
2722         DO 5900 J= JB2,JT2
2723         PRINT 48, I,J,U(I,J),V(I,J),P(I,J),H(I),JT1,JB1
2754   5900 CONTINUE
      *
      C * * SET THE ADVANCE TIME VELOCITIES U AND V INTO THE UN AND VN ARRAYS
      C * * AND THE ADVANCED TIME SURFACE HEIGHT H INTO THE HN ARRAY
      *
2761   6000 CONTINUE
2761         DO 6100 J=1,JMAX
2763         DO 6100 I=1,IMAX
2773         UN(I,J)=U(I,J)
2773         VN(I,J)=V(I,J)
2773         HN(I)=H(I)
2773         HBN(I)= HB(I)
2773   6100 CONTINUE
      *
      C * * ADVANCE TIME T= T+DELT
      *
3003         T=T+DELT
3005         IF(T.GT.TWFIN)  GOTO 6500
3010         CYCLE=CYCLE+1
3011         GOTO 1000
3012   6500 STOP
3014         END
```

PROGRAM LENGTH INCLUDING I/O REQUEST TABLES = SOLASOR
65657

STATEMENT NUMBER REFERENCES

LOCATION	GEN TAG	STMT NO	REFERENCES					
3024	L00112	25	23					
3030	L00116	27	141	2556	2372	2424	2534	2655
3035	[illegible]	35	11	2651				
3037	L00125	44	NONE					
3046	L00134	45	3	15				
3053	L00150	46	2354	2410	2442	2552		
3055	L00143	47	2255	2763				
3063	L00153	48	2271	2717				
3072	L00162	49	2212	2237	2671			
3102	L00272	50	[illegible]	2201				
250	L00153	260	225	226				
271	L00156	262	251					
310	L00167	290	270					
342	L00211	320	333	334				
352	L00215	330	341					
401	L00237	[illegible]	[illegible]					
713	L00260	2020	420	1631				
743	L00303	2020	737					
752	L00306	2040	740					
761	L00311	2060	741					
774	L00317	2080	772					
1015	L00320	2100	751	763	762	773	1012	
1023	L00331	2120	1017					
1033	L00334	2140	1026					
1043	L00337	2160	1021					
1063	L00345	2180	1022					
1111	L00357	2200	1052	1042	1044	1062		
1130	L00370	2320	1124					
1137	L00373	2340	1125					
1146	L00376	2360	1126					
1162	L00404	2380	1127					
1171	L00407	2400	1136	1145	1147	1161	1170	
1201	L00412	2420	1175					
1205	L00415	2440	1176					
1211	L00420	2460	1177					
1224	L00426	2480	1206					
1226	L00430	2500	1204	1210	1212	1217		
1251	L00432	2600	1774					
1274	L00452	2610	1244					
1325	L00461	2620	1274					
1350	L00463	2650	1256					
1333	L00465	3040	403					
1344	L00476	3050	1337					
1410	L00534	3060	1377					
1462	L00542	3070	1406					
1525	L00550	3100	1370					
1537	L00553	3150	1415					
1553	L00556	3200	1362	1415				
1605	L00563	3300	1401	1522	1536	1552		

```
   1632      L00573     4000          1333        1341        1343
   1702      L00607     4200          1633
   1752      L00623     4250          1703
   1775      L00636     4260           377        1773
   2060      L00665     4345          2023
   2121      L00702     4355          2062
   2150      L00714     4365          2123
   2175      L00726     4375          2152
   2177      L00727     5000          NONE
   2210      L00735     5030          2200
   2233      L00745     5140          2224        2225
   2642      L01134     5600          2230
   2651      L01141     5800          2642        2643
   2756      L01174     6000          2646
   3006      L01220     6500          3043
```

VARIABLE REFERENCES

```
 LOCATION   GEN TAG    NAME       REFERENCES
   3272     V00142     ADELX   \R  2005     2134     2161
   3273     V00051     ALPHA   \R    70      437      463      510      532      553
                                    606
   3274     V00054     BB      \R   102      127
   3275     V00076     BETA    \R   106     1603
   3276     V00071     CLK     \R   137      152     2347     2403     2435     2545
                                   2666
   3277     V00047     CWPLT   \R    65     2231
   3300     V00046     CWPRT   \R    64      155     2647
   3301     V00021     CYCLE   \I   161     1337     2217     2224     2246     2361
                                   2415     2447     2557     2642     2700     3004
   3302     V00035     CYL     \R    46      514      612      632      656     1053
                                   1267     1320     1571     1777     2124     2153
   3303     V00135     D       \R  1574
   3304     V00070     DAT     \R   135      150     2345     2401     2433     2543
                                   2604
   3305     V00133     DELP    \R  1461     1521     1535     1551     1604     1610
   3306     V00034     DELT    \R    43      155      670      702     1451     1455
                                   1515     1610     1675     1745     2030     2054
                                   2070     2115     2143     2170     2231     2647
                                   3000
   3307     V00032     DELX    \R    40      116      317      324      352      416
                                    564      614      627      641      653     1257
                                   1310     1556     2012     2141     2166     2576
   3310     V00033     DELY    \R    42      120      264      301      326      352
                                    422      631      655     1263     1314     1420
                                   1466     1504     1527     1543     1646     1716
                                   2431     2047     2071     2110     2601
   3311     V00132     DFDP    \R  1460     1521
   3312     V00037     DZRO    \R    51     1575
   3313     V00036     EPSI    \R    56     1576
   3314     V00131     F       \R  1452     1457     1511     1517
   3315     V00116     FLG     \R   443     1333     1344     1602
   3316     V00100     FIRT    \R   214      216
   3317     V00143     FIX     \R  2021     2035     2055     2075     2116     2145
                                   2172     2175
```

3320	VA0121	FMC	\R	520	666				
3321	VA0117	FMX	\R	451	664				
3322	VA0120	FMY	\R	475	665				
3323	VA0124	FVC	\R	617	700				
3324	VA0141	FVOL	\R	1775	2216	2175	2221	2254	
3325	VA0122	FVX	\R	545	676				
3326	VA0123	FVY	\R	567	677				
3327	VA0052	GAMMA	\R	72	1663	1733			
3330	VA0040	GX	\R	52	663				
3331	VA0041	GY	\R	54	257	277	675		
3332	AA0407	H	\R	226	235	237	1470	1730	2005
				2451	2112	2771	234	1476	2330
				2474	254	1476	1754	2313	2741
3644	AA0213	HB	\R	171	206	210	1427	1454	1750
				2004	2053	2114	2772	205	1441
				2304	2523	274	1442	1544	1764
4155	VA0077	HMAT	\R	167	173				
4157	AA0014	HMV	\R	724	726	1007	1010	1113	1166
				1710	2774	722	1141	1713	1104
				1712					
4471	VA0142	HMV	\R	1726	1740				
4472	VA0146	HOIF	\R	2057	2122				
4473	VA0104	HIGH	\R	330	343	2464	2520	2617	2627
4474	AA0210	HV	\R	716	720	1004	1005	1075	1100
				1042	2773	715	1074	1643	1076
				1532	1642				
5006	VA0136	HV	\R	1656	1670				
5007	VA0027	I	\I	170	215	223	230	243	252
				272	310	313	357	364	375
				405	412	656	710	1113	1130
				1137	1150	1162	1201	1205	1213
				1220	1226	1237	1254	1275	1305
				1325	1346	1417	1462	1523	1537
				1553	1605	1626	1635	1705	1753
				1764	1777	2124	2151	2153	2262
				2274	2300	2305	2310	2313	2325
				2503	2571	2637	2710	2722	2726
				2733	2736	2741	2753	2760	
5010	VA0050	IBAR	\I	35	112	323	2331	2365	2475
				2504					
5011	VA0066	IMS	\I	132	247	1233	1274	1373	1411
				1722					
5012	VA0055	IMAX	\I	113	115	125	204	233	313
				713	1026	1036	1046	1063	1226
				2326	2754	2762			
5013	VA0057	IM1	\I	122	170	201	230	312	375
				710	713	1023	1033	1045	1054
				1064	1325	1627	1636	1706	1771
				2002	2062	2067	2103	2150	2156
				2640					
5014	VA0064	IM2	\I	130	774	1047			
5015	VA0065	IM3	\I	131	1231	1243	1364	1402	1632
5016	VA0073	ITER	\I	157	401	761	1043	1146	1211
				1334	2213	2242	2674		
5017	VA0153	IUVEC	\I	2612	2633				

5020	V00155	IXCC	\I	2622	2632				
5021	V00106	IXL	\I	335	347	2514	2524	2527	2614
				2624					
5022	V00107	IXR	\I	340	350	2514	2524	2527	2614
				2624					
5023	V00105	IYB	\I	331	2515	2525	2530	2620	2630
5024	V00110	IYT	\I	341	351	2515	2525	2530	2620
				2630					
5025	V00103	J	\I	252	272	304	412	655	705
				732	744	753	763	775	1024
				1034	1045	1065	1110	1353	1370
				1416	1424	1431	1462	1500	1606
				1624	2027	2067	2271	2276	2301
				2523	2570	2577	2635	2717	2724
				2727	2751	2757	2760	2775	
5026	A00312	JH	\I	172	211	212	240	362	410
				727	730	1011	1012	1105	1107
				1116	1242	1275	1351	1715	2045
				2065	2126	2263	2560	2713	207
				725	1123	2024	1106	1276	1767
5340	V00051	JHAR	\I	56	114	326	2470		
5341	V00145	JOF	\I	2326	2366				
5342	V00122	JM1	\I	247	271	411	1117	1243	1300
				1345	1352	1353	1371	1407	1537
				1717	1720	2264	2323	2567	2714
				2746					
5343	V00114	JM2	\I	303	2270	2716			
5344	V00056	JMAX	\I	114	1110	1133	1142	1152	1163
				2776					
5345	V00060	JN1	\I	124	224	1131	1140	115[illegible]	1165
				1761	2125				
5346	V00063	JN2	\I	125	1154	1221			
5347	V00067	JNM	\I	133	146	2343	2377	2431	2541
				2602					
5351	A00011	JT	\I	225	240	241	244	300	406
				721	723	1006	1007	1077	1122
				1114	1240	1244	1347	1645	2045
				2063	2166	2265	2564	2711	224
				226	1245	1757	1760	1763	236
				717	1075	2224	1100		
5662	V00144	JTP	\I	2426	2432	2465	2472		
5663	V00101	JT1	\I	245	255	275	407	706	1115
				1241	1254	1350	1356	1523	1624
				1647	1650	2042	2102	2266	2516
				2565	2635	2712	2744		
5664	V00113	JT2	\I	302	356	2267	2323	2715	2751
5665	V02154	JVVEC	\I	2615	2633				
5666	V00156	JVEC	\I	2626	2632				
5667	V00115	KRET	\I	424	474	1330	1774		
5670	V00317	LONG	\R	325	352	342	2462	2516	2613
				2623					
5671	R00016	NAME	\I	6	20	144	2341	2375	2427
				2537	2660				
5703	V00022	[illegible]	\R	44	653				
5704	V00026	NUM	\I	26	30	106	2204		

5705	V00050	OMG	\R	66	165				
5706	A00005	P	\R	256	276	661	672	1534	1550
				1614	673	1547	661	1533	2310
				2736					
17266	V00154	RETA	\R	1533	1547				
17267	V00061	RDX	\R	121	156	662	1263	1314	1426
				1443	1454	1477	1505	1567	1611
				1635	1705				
17270	V00062	RDY	\R	126	156	174	217	675	1425
				1472	1515	1570	1627	1652	1722
				1754	1765				
17271	V00072	T	\R	161	1343	2177	2215	2225	2244
				2357	2413	2445	2555	2643	2676
				3400					
17272	V00053	TH	\R	141	122				
17273	V00045	TAPIN	\R	62	3002				
17274	V00075	TAPLT	\R	162	2226				
17275	V00074	TAPRT	\R	164	2644				
17276	A00001	U	\R	571	676	750	757	770	771
				1001	1022	1031	1041	1061	1134
				1143	1157	1204	1210	1216	1217
				1224	1225	1261	1271	1312	1322
				1435	1471	1615	1654	1724	2034
				2070	2134	2160	2602	2706	1051
				1072	1132	1141	1154	1162	1247
				1360	1562	2104	2300	2726	1072
				1167	1223	1207	1251	1273	1262
				1302	1313	1435	1471	1617	1653
				1723	2601	1324	2244	1560	
34656	V00157	UAV	\R	1601	1731				
34657	V00042	UI	\R	56	370				
34660	A00003	UN	\R	423	672	2767	425	425	446
				624	426	521	572	641	
42240	V00151	UVEC	\R	2611	2612				
42241	A00002	V	\R	371	745	751	760	771	772
				1003	1004	1030	1040	1060	1135
				1144	1160	1202	1206	12.4	1215
				1223	1224	1273	1316	1442	1445
				1620	1672	1742	2606	2764	746
				755	1026	1036	1052	1067	1156
				1164	1564	2305	2733	1065	1166
				1265	1324	1506	1622	1672	1742
				2646	1565				
53621	V00130	VBM	\R	1423					
53622	V00044	VELMX	\R	60	355				
53623	V00112	VELMX1	\R	356	2570				
53624	V00043	VI	\R	57	370				
53625	V00125	VISX	\R	635	667				
53626	V00126	VISY	\R	671	701				
53627	A00004	VN	\R	424	646	703	2765	516	570
				630	645	652	463	527	570
				602	645	651	451	542	647
				551	646	452			
65207	V00127	VTM	\R	1422	1465				
65210	V00152	VVEC	\R	2611	2616				

65211	V00025	WH	VI	100	1171	2150			
65212	V00022	WL	VI	75	732	2021			
65213	V00023	WR	VI	74	1015	2060			
65214	V00024	WT	VI	76	1117	2121			
65215	V00111	X	VR	342					
65216	V00147	XCC	VR	2600	2604	2622			
65217	V00006	XPUT	VR	30	34	106	2204	35	37
				40	41	43	45	46	47
				51	53	54	55	57	61
				62	63	65	67	70	71
				73	75	76	77	101	
65253	A00015	XX	VR	316	2350	2364	2474	2503	
65562	V00150	YCC	VR	2605	2607	2626			

EXTERNAL REFERENCES

GEN TAG	NAME		REFERENCES					
S00200	INPUTC	VI	5	7	10	25	27	32
			33					
S00300	OUTPTC	VR	13	14	17	21	22	105
			110	111	143	145	147	151
			153	154	2203	2206	2207	2212
			2214	2216	2220	2222	2223	2241
			2243	2245	2247	2251	2252	2257
			2260	2273	2275	2277	2304	2307
			2312	2315	2317	2321	2322	2340
			2342	2344	2346	2350	2351	2356
			2360	2362	2363	2374	2376	2400
			2402	2404	2405	2412	2414	2416
			2417	2426	2430	2432	2434	2436
			2437	2444	2446	2450	2451	2536
			2540	2542	2544	2546	2547	2554
			2556	2560	2561	2653	2654	2657
			2661	2663	2665	2667	2670	2673
			2675	2677	2701	2702	2705	2706
			2721	2723	2725	2732	2735	2740
			2743	2745	2747	2750		
S00400	GETJ	VR	134					
S00500	GATE1	VR	136					
S00600	CLOCK1	VR	140					
S00700	ACGDER	VR	736	1016	1125	1174	1331	
S01000	RHAREX	VR	1502					
S01100	ADV	VR	2234	2421	2513			
S01200	LINCNT	VI	2236	2254	2335	2353	2371	2407
			2423	2441	2533	2551		
S01300	SPLOT	VR	2333	2567				
S01400	FRAME	VR	2454	2457	2526	2531		
S01500	OGA	VR	2463	2467	2517	2523		
S01600	SCLIN	VR	2471					
S01700	SHLIN	VR	2473					
S02000	PLOT	VR	2477	2502	2506	2511		
S02100	CONVRT	VR	2615	2621	2625	2631		
S02200	DRV	VR	2634					
S02300	STOP	VR	3007					
S02400	END	VR	3011					
S03100	IBRARY		NONE					

U.S. GOVERNMENT PRINTING OFFICE 1975 —677-180 44