AN ABSTRACT OF THE THESIS OF

MICHAEL DENNIS TERRY (Name) for the MASTER OF SCIENCE (Degree)

in MATHEMATICS (Major) presented on May 6, 1968 (Date)

Title: A NUMERICAL STUDY OF VISCOUS, INCOMPRESSIBLE FLUID FLOW PROBLEMS

Abstract approved Redacted for Privacy
Giles W. Maloof

Redacted for Privacy
Larry S. Slotta

The mathematical technique of overrelaxation is used here to speed the convergence of a numerical method for solving viscous, incompressible fluid flow problems. The method, called MAC, involves approximating the complete two-dimensional incompressible Navier-Stokes equations with analogous finite-difference equations. The MAC Method was developed by the Los Alamos Scientific Laboratory of the University of California and is described in detail in "The MAC Method," by J. Eddie Welch, Francis H. Harlow, John P. Shannon, and Bart J. Daly (LA-3425). The most time-consuming part of this method involves iterating on a finite-difference form of Poisson's equation; hence overrelaxation is applied to this iteration process.

Overrelaxation theory is presented and is applied to the above

method to show that overrelaxation does speed convergence. Examples of fluid problem solutions are presented both with and without the use of overrelaxation, showing the amount of computer time saved. In addition, computer plots of several typical solutions are included to show the diversity of the above method.

A Numerical Study of Viscous, Incompressible Fluid Flow Problems

by

Michael Dennis Terry

A THESIS

submitted to

Oregon State University

in partial fulfillment of
the requirements for the
degree of

Master of Science

June 1968

APPROVED:

Redacted for Privacy

Assistant Professor of Mathematics

in charge of major

Redacted for Privacy

Associate Professor of Civil Engineering

in charge of research

Redacted for Privacy

Acting Chairman of Department of Mathematics

Redacted for Privacy

Dean of Graduate School

Date thesis is presented May 6, 1968

Typed by Clover Redfern for Michael Dennis Terry

ACKNOWLEDGMENTS

The author wishes to express his gratitude to the following persons: Dr. Giles W. Maloof, major professor, whose numerous suggestions were very helpful in the preparation of this paper; and Dr. Larry S. Slotta, research supervisor, who originated the research, and who, through a grant from the Federal Water Pollution Control Administration, provided funds for the research (Density Stratified Reservoir Currents, W 00983-02). Appreciation is expressed to John D. Hwang and Howard T. Mercier, colleagues on the research project, who provided help with the mathematical and physical aspects of the research, respectively.

The author also wishes to thank the computer center personnel at the National Center for Atmospheric Research in Boulder, Colorado, and the Lawrence Radiation Laboratory of the University of California in Berkeley, California, for their valuable assistance in times of need.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

A NUMERICAL STUDY OF VISCOUS, INCOMPRESSIBLE FLUID FLOW PROBLEMS

I. INTRODUCTION

In the field of fluid mechanics, the governing equations of motion are very complex non-linear partial differential equations. Because of this complexity, analytical solutions can be obtained only for highly-simplified, in effect non-physical, flow patterns. In order to solve the equations of motion for more sophisticated problems, various numerical methods have been successfully applied. These methods include: (1) reduction to ordinary differential equations so that numerical integration techniques may be used; (2) linearization techniques to reduce the equations to the point where analytical solutions may be obtained; and (3) finite-difference methods to reduce the equations to a set of algebraic equations which are solved by trial-and-error, or iteration, techniques.

The first two of these methods are limited in application because they are restrictive and involve much detailed analytical work. A stringent restriction placed on fluid problems by these techniques is that of steady flow; i.e., time derivatives of variables must vanish. The third method mentioned above--finite differences--allows the user to solve most types of fluid problems, including those involving unsteady flow.

Finite-difference methods are by far the most widely used numerical methods for solving fluid flow problems. These methods involve a minimal amount of analytical work and are further facilitated by today's high-speed digital computers. They involve replacing a homogeneous fluid region with a discrete one and approximating partial derivatives with difference quotients centered at each of the points of the discrete mesh. There are two basic approaches in applying finite-difference methods: the Eulerian approach, where an observer is stationary with respect to the fluid flow; and the Lagrangian approach, where the observer travels with the fluid. Of these, the Eulerian is the more popular.

Recently, scientists at the Los Alamos Scientific Laboratory employed finite differences in developing a method for solving unsteady, viscous, incompressible fluid flow problems. This method is based on the Eulerian approach and is very general; almost any type of incompressible fluid problem may be examined up to the onset of turbulence. The method is called the Marker-and-Cell (MAC) Method and is developed in a report from the Laboratory (Welch et al., 1966).

The MAC Method solves the two-dimensional incompressible form of the equations of motion. It is an algorithm which computes values of the field variables, at a certain point in time, from (1) quantities at the previous time and (2) appropriate boundary conditions.

Although the method is effective for unsteady flow problems, the calculations become tedious and time-consuming during heavy transient periods, i.e., when the fluid velocity changes speed or direction rapidly. This is due to the necessity of solving one of the equations involved implicitly. Since problems must be solved on expensive computers, a reduction in the number of steps, or iterations, needed for this implicit solution would be helpful. One way to accomplish this time-reduction is to employ a mathematical technique called overrelaxation.

Overrelaxation and its application to the MAC Method are the subjects of this paper. In Chapter II, the MAC Method is described briefly, while overrelaxation techniques are presented in Chapter III. Chapter IV consists of the actual application of overrelaxation to MAC, including studies made of the time saved on two problems involving different types of transients. Examples of the diversity of fluid flow problems which can be solved by this method are included in Chapter V. Other examples can be found in papers by Hwang (1968) and Mercier (1968).

II. THE MARKER-AND-CELL METHOD

In this chapter, the Marker-and-Cell (MAC) Method for solving viscous, incompressible, time-dependent fluid flow problems is presented. (For a more detailed treatment, see Welch, 1966.) As mentioned in Chapter I, MAC is an algorithm for solving a system of finite-differenced forms of the partial differential equations governing the motion of a fluid. These equations are presented here along with the initial and boundary conditions necessary for their solution. Next, the algorithm itself is described, and finally some extensions of the MAC Method are mentioned.

Equations

There are two physical laws which govern the motion of a fluid: (1) the Law of Conservation of Mass and (2) Newton's Second Law of Motion. In terms of the field variables, these laws may be written (assuming incompressible flow), respectively, as (Schlichting, 1960):

$$\nabla \cdot \vec{V} = 0 \tag{2.1}$$

and

$$\frac{\partial \vec{V}}{\partial t} = - (\vec{V} \cdot \nabla)\vec{V} - \nabla\phi + \nu \nabla^2 \vec{V} + \vec{g} \, . \tag{2.2}$$

The variables used here are

$\vec{V}$ = fluid velocity,

t = time,

ϕ = ratio of pressure to density,

ν = kinematic viscosity, and

$\vec{g}$ = acceleration due to gravity.

The independent variables involved are time (t), and the space coordinates (x, y in two-dimensional Cartesian coordinates). The dependent variables are $\vec{V} = u\vec{i} + v\vec{j}$ (where $\vec{i}$ and $\vec{j}$ are the unit vectors in the x and y directions, respectively) and ϕ, while ν and $\vec{g} = g_x\vec{i} + g_y\vec{j}$ are considered known quantities. Thus, expanding (2.1) and (2.2) in two dimensions, the problem involves three equations in three unknowns:

$$\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} = 0, \tag{2.3}$$

$$\frac{\partial u}{\partial t} = -\left(u\frac{\partial u}{\partial x} + v\frac{\partial u}{\partial y}\right) - \frac{\partial \phi}{\partial x} + \nu\left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}\right) + g_x , \tag{2.4}$$

and

$$\frac{\partial v}{\partial t} = -\left(u\frac{\partial v}{\partial x} + v\frac{\partial v}{\partial y}\right) - \frac{\partial \phi}{\partial y} + \nu\left(\frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2}\right) + g_y . \tag{2.5}$$

Using the vector identity

$$\nabla \cdot (\vec{A}\,\vec{B}) \equiv (\vec{A} \cdot \nabla)\vec{B} + (\nabla \cdot \vec{A})\vec{B}$$

and Equation (2. 1), it may be seen that

$$\frac{\partial \vec{V}}{\partial t} = - \nabla \cdot (\vec{V}\ \vec{V}) - \nabla \phi + \nu \nabla^2 \vec{V} + \vec{g} \qquad (2.6)$$

is equivalent to Equation (2. 2). Although the partial differential equations are equivalent, it is clear that the finite difference forms of (2. 2) and (2. 6) are not. Harlow (Welch et al. , 1966) has shown that the finite difference analogy of (2. 6) satisfies Newton's Second Law more precisely than the analogous form of (2. 2). Hence the two-dimensional expansions of Equation (2. 6),

$$\frac{\partial u}{\partial t} = - \left[\frac{\partial (u^2)}{\partial x} + \frac{\partial (uv)}{\partial y}\right] - \frac{\partial \phi}{\partial x} + \nu \left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}\right) + g_x \qquad (2.7)$$

and

$$\frac{\partial v}{\partial t} = - \left[\frac{\partial (v^2)}{\partial y} + \frac{\partial (uv)}{\partial x}\right] - \frac{\partial \phi}{\partial y} + \nu \left(\frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2}\right) + g_y \ , \qquad (2.8)$$

are used in place of Equations (2. 4) and (2. 5).

In order to solve Equations (2. 3), (2. 7), and (2. 8) using the calculus of finite differences, the fluid region being examined must be approximated with a finite number of points. This is done by defining a grid, or mesh, which covers the region, as shown in Figure 1. Although there are placements of the field variables relative to the mesh different from that shown in Figure 1, Harlow (Welch et al. , 1966)

reports that this is the only one which satisfies the physical laws.

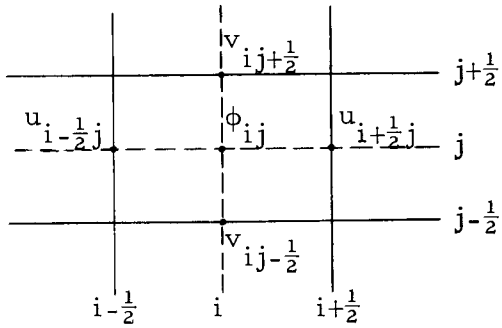


Figure 1. The mesh and variable placement.

Hereafter the rectangle defined by the solid lines $i-\frac{1}{2}$, $i+\frac{1}{2}$, $j-\frac{1}{2}$, and $j+\frac{1}{2}$ in Figure 1 will be designated <u>cell i, j</u> of the computational mesh.

With the fluid region being defined as in Figure 1, and with the cells having dimensions δx by δy, the finite difference forms of Equations (2.3), (2.7), and (2.8) become, respectively:

$$\frac{1}{\partial x}(u_{i+\frac{1}{2}j}-u_{i-\frac{1}{2}j}) + \frac{1}{\delta y}(v_{ij+\frac{1}{2}}-v_{ij-\frac{1}{2}}) = 0 \ , \tag{2.9}$$

$$\begin{aligned}\frac{1}{\delta t}(u^{n+1}_{i+\frac{1}{2}j}-u_{i+\frac{1}{2}j}) = {} & \frac{1}{\delta x}[(u_{ij})^2-(u_{i+1j})^2] \\ & + \frac{1}{\delta y}[(uv)_{i+\frac{1}{2}j-\frac{1}{2}}-(uv)_{i+\frac{1}{2}j+\frac{1}{2}}] \\ & + g_x + \frac{1}{\delta x}(\phi_{ij}-\phi_{i+1j}) \\ & + \nu\Big[\frac{1}{\delta x^2}(u_{i+\frac{3}{2}j}+u_{i-\frac{1}{2}j}-2u_{i+\frac{1}{2}j}) \\ & + \frac{1}{\delta y^2}(u_{i+\frac{1}{2}j+1}+u_{i+\frac{1}{2}j-1}-2u_{i+\frac{1}{2}j})\Big] \ ,\end{aligned} \tag{2.10}$$

and

$$\frac{1}{\delta t}(v^{n+1}_{ij+\frac{1}{2}} - v_{ij+\frac{1}{2}}) = \frac{1}{\delta y}[(v_{ij})^2 - (v_{ij+1})^2]$$

$$+ \frac{1}{\delta x}[(uv)_{i-\frac{1}{2}j+\frac{1}{2}} - (uv)_{i+\frac{1}{2}j+\frac{1}{2}}]$$

$$+ g_y + \frac{1}{\delta y}(\phi_{ij} - \phi_{ij+1})$$

$$+ \nu[\frac{1}{\delta x^2}(v_{i+1j+\frac{1}{2}} + v_{i-1j+\frac{1}{2}} - 2v_{ij+\frac{1}{2}})$$

$$+ \frac{1}{\delta y^2}(v_{ij+\frac{3}{2}} + v_{ij-\frac{1}{2}} - 2v_{ij+\frac{1}{2}})] \quad . \qquad (2.11)$$

The superscript $n+1$ here refers to the value at time $(n+1)\delta t$ and the absence of a superscript denotes the value at time $n\delta t$. It may be noted that there are some quantities in these equations which are not defined in Figure 1. In such a case, a simple average of defined quantities is used. For example,

$$u_{ij} \equiv \frac{1}{2}(u_{i+\frac{1}{2}j} + u_{i-\frac{1}{2}j})$$

and

$$(uv)_{i+\frac{1}{2}j-\frac{1}{2}} \equiv \frac{1}{4}(u_{i+\frac{1}{2}j} + u_{i+\frac{1}{2}j-1})(v_{ij-\frac{1}{2}} + v_{i+1j-\frac{1}{2}}) \quad .$$

Now, since Equations (2.10) and (2.11) can be used to obtain values for u and v, only an equation relating ϕ to u and v is needed. To get this relation, a new variable is first defined:

$$D_{ij} \equiv \frac{1}{\delta x}(u_{i+\frac{1}{2}j} - u_{i-\frac{1}{2}j}) + \frac{1}{\delta y}(v_{ij+\frac{1}{2}} - v_{ij-\frac{1}{2}}) \ . \tag{2.12}$$

Using this definition, Equation (2.9) obviously reduces to:

$$D_{ij} = 0 \ . \tag{2.13}$$

It should be noted that due in part to truncation errors resulting from the process of finite differencing, values of u and v obtained from Equations (2.10) and (2.11) will not, in general, satisfy Equation (2.13). For this reason, (2.13) is used mainly as a check for stability of the solution. If values of D_{ij} exceed some predefined tolerance, then the solution process is unstable and any resulting answers will not be reliable.

Using Equations (2.10), (2.11), and (2.12), an expression for the time rate of change of D_{ij} is found:

$$\begin{aligned}\frac{1}{\delta t}(D_{ij}^{n+1} - D_{ij}) = -\Bigg\{ & \frac{1}{\delta x^2}\left[(u_{i+1j})^2 + (u_{i-1j})^2 - 2(u_{ij})^2\right] \\ & + \frac{1}{\delta y^2}\left[(v_{ij+1})^2 + (v_{ij-1})^2 - 2(v_{ij})^2\right] \\ & + \frac{2}{\delta x \delta y}\left[(uv)_{i+\frac{1}{2}j+\frac{1}{2}} + (uv)_{i-\frac{1}{2}j-\frac{1}{2}} - (uv)_{i+\frac{1}{2}j-\frac{1}{2}} - (uv)_{i-\frac{1}{2}j+\frac{1}{2}}\right] \\ & + \frac{1}{\delta x^2}(\phi_{i+1j} + \phi_{i-1j} - 2\phi_{ij}) + \frac{1}{\delta y^2}(\phi_{ij+1} + \phi_{ij-1} - 2\phi_{ij})\Bigg\} \\ & + \nu\left[\frac{1}{\delta x^2}(D_{i+1j} + D_{i-1j} - 2D_{ij}) + \frac{1}{\delta y^2}(D_{ij+1} + D_{ij-1} - 2D_{ij})\right] \ .\end{aligned} \tag{2.14}$$

Now, by setting $D_{ij}^{n+1} = 0$ in (2.14), the desired relation for ϕ is obtained:

$$\phi_{ij} = \frac{1}{2\left(\frac{1}{\delta x^2} + \frac{1}{\delta y^2}\right)} \left(\frac{\phi_{i+1j}+\phi_{i-1j}}{\delta x^2} + \frac{\phi_{ij+1}+\phi_{ij-1}}{\delta y^2} + S_{ij}\right), \tag{2.15}$$

where

$$\begin{aligned} S_{ij} = {} & \frac{1}{\delta x^2}\left[(u_{i+1j})^2+(u_{i-1j})^2-2(u_{ij})^2\right] \\ & + \frac{1}{\delta y^2}\left[(v_{ij+1})^2+(v_{ij-1})^2-2(v_{ij})^2\right] \\ & + \frac{2}{\delta x \delta y}\left[(uv)_{i+\frac{1}{2}j+\frac{1}{2}}+(uv)_{i-\frac{1}{2}j-\frac{1}{2}}-(uv)_{i+\frac{1}{2}j-\frac{1}{2}}-(uv)_{i-\frac{1}{2}j+\frac{1}{2}}\right] \\ & - \frac{D_{ij}}{\delta t} - \nu\left[\frac{1}{\delta x^2}(D_{i+1j}+D_{i-1j}-2D_{ij})+\frac{1}{\delta y^2}(D_{ij+1}+D_{ij-1}-2D_{ij})\right], \end{aligned} \tag{2.16}$$

called the source term of Equation (2.15). Thus, with appropriate initial and boundary conditions, an algorithm may be developed to solve Equations (2.10), (2.11), and (2.15) for each point in the mesh at each point in time.

Initial and Boundary Conditions

When solving for u^{n+1} and v^{n+1} in Equations (2.10) and (2.11), if values for u and v at time $n\delta t$ are known, they may be substituted into (2.12), (2.16), and (2.15), respectively, to obtain

values of ϕ, thus making the solution of (2.10) and (2.11) a mere matter of substitution. In other words, initial (time = 0) values of u and v are needed to start the solution process. Hence functions f_1 and f_2 are required such that

$$u_{ij}^0 = f_1(i,j) , \tag{2.17}$$

and

$$v_{ij}^0 = f_2(i,j) . \tag{2.18}$$

In addition to the initial conditions, values for u and v are needed at the boundaries of the fluid region; similar values are also required for ϕ. These boundary conditions will now be developed.

There are five types of boundaries which are allowed in the MAC Method: (1) input; (2) output; (3) no-slip wall; (4) free-slip wall; and (5) free surface. In the sequel, each of the first four boundary types is treated as a bottom boundary; i.e., the fluid region lies immediately above. Similar results can be obtained for the other three configurations with suitable manipulation of the subscripts.

Before proceeding to develop the boundary conditions for a specific type of boundary, one should note that the boundary conditions for ϕ are derived from Equations (2.10) and (2.11). Thus, at a bottom boundary,

$$\phi_{ij-1} = \phi_{ij} + \frac{\delta y}{\delta t}\left(v^{n+1}_{ij-\frac{1}{2}} - v_{ij-\frac{1}{2}}\right) - \left\{\left(v_{ij-1}\right)^2 - \left(v_{ij}\right)^2 + \delta y\left[\frac{(uv)_{i-\frac{1}{2}j-\frac{1}{2}} - (uv)_{i+\frac{1}{2}j-\frac{1}{2}}}{\delta x} + g_y + \nu\left(\frac{v_{i+1j-\frac{1}{2}} + v_{i-1j-\frac{1}{2}} - 2v_{ij-\frac{1}{2}}}{\delta x^2} + \frac{v_{ij+\frac{1}{2}} + v_{ij-\frac{3}{2}} - 2v_{ij-\frac{1}{2}}}{\delta y^2}\right)\right]\right\}, \qquad (2.19)$$

where the subscripts are defined in Figure 2.

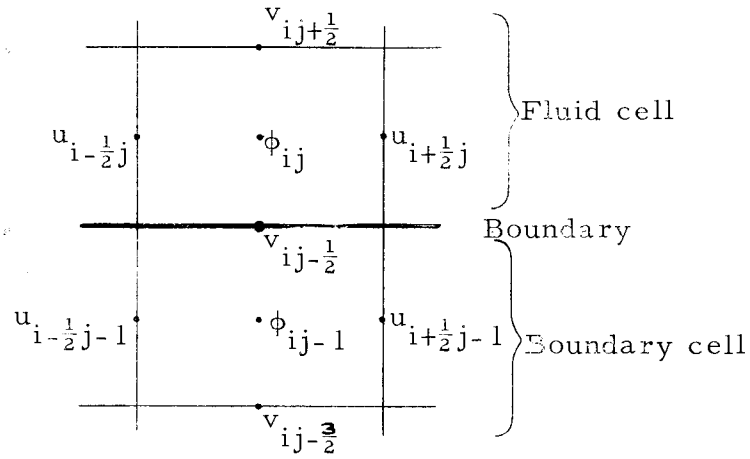


Figure 2. Bottom boundary.

Note that, as shown in Figure 2, an extra layer of cells is required outside the fluid region and that the subscripts are still in general form. The last observation is due to the possible occurrence of a boundary anywhere within the i, j mesh. An example of this is shown in Figure 3, where a typical fluid problem, that of flow through a stepped channel, is depicted physically in Figure 3a and as a finite computational mesh in Figure 3b. Note that there are two bottom boundaries and two right boundaries, thus showing how boundaries

may occur anywhere within the grid. Cells marked with an X are termed null cells since they do not enter into the calculations. Also shown in Figure 3b is a cell marked with a C. This is termed a corner cell and becomes a sixth case for which boundary conditions are needed, since two boundaries overlap at one of these cells. The six cases are outlined below.

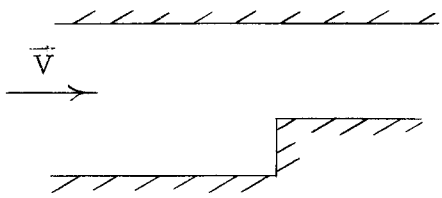
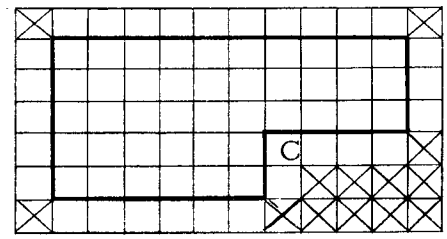


Figure 3a. Flow through a stepped channel.



Null cell C Corner cell

Figure 3b. The computational mesh for a stepped channel.

Case 1: Input. At an input boundary, it is assumed, for simplicity, that the incoming fluid velocity is constant along the boundary, constant with respect to time, and normal to the boundary. These conditions result in the following, respectively (referring to Figure 2):

$$v_{i-1j-\frac{1}{2}} = v_{ij-\frac{1}{2}} = v_{i+1j-\frac{1}{2}} , \tag{2.20}$$

$$v^{n+1}_{ij-\frac{1}{2}} = v_{ij-\frac{1}{2}} = v_{input} , \tag{2.21}$$

and

$$u_{i-\frac{1}{2}j-1} = -u_{i-\frac{1}{2}j}, \quad u_{i+\frac{1}{2}j-1} = -u_{i+\frac{1}{2}j}. \tag{2.22}$$

From Equation (2.9),

$$v_{ij-\frac{1}{2}} = v_{ij+\frac{1}{2}} + \frac{\delta y}{\delta x}(u_{i+\frac{1}{2}j} - u_{i-\frac{1}{2}j}) \tag{2.23}$$

and

$$v_{ij-\frac{1}{2}} = v_{ij-\frac{3}{2}} + \frac{\delta y}{\delta x}(u_{i-\frac{1}{2}j-1} - u_{i+\frac{1}{2}j-1}).$$

Combining these with (2.22) gives

$$v_{ij-\frac{3}{2}} = v_{ij+\frac{1}{2}}. \tag{2.24}$$

Now, by substituting (2.20), (2.21), (2.22), and (2.24) into (2.19), the condition for ϕ is found:

$$\phi_{ij-1} = \phi_{ij} - g_y \delta_y - \frac{2\nu}{\delta y}(v_{ij+\frac{1}{2}} - v_{ij-\frac{1}{2}}). \tag{2.25}$$

Thus the boundary conditions for an input boundary are given by (2.21), (2.22), and (2.25).

Case 2: Output. The velocity boundary conditions at an output boundary are given by (2.23) and the following:

$$u_{i-\frac{1}{2}j-1} = u_{i-\frac{1}{2}j}, \quad u_{i+\frac{1}{2}j-1} = u_{i+\frac{1}{2}j}. \tag{2.26}$$

The boundary condition for ϕ is not derived directly from Equation

(2.19), but rather is formulated to agree with experimental results. It was learned from Harlow (1967) that many conditions were tried, with the following agreeing most closely with experiment:

$$\phi_{ij-1} = \phi_{ij} - \frac{\delta y}{\delta x}\left[(uv)_{i-\frac{1}{2}j-\frac{1}{2}} - (uv)_{i+\frac{1}{2}j-\frac{1}{2}}\right] . \tag{2.27}$$

Case 3: No-slip wall. At a no-slip wall, the tangential component of velocity vanishes, while at a free-slip wall, the fluid is allowed to move freely past it. (No intermediate type of solid wall was considered, and the criterion for choosing one or the other is the expected depth of the boundary layer along the wall. If the depth of viscous interaction of the fluid with the wall is expected to be less than δy, then a free-slip wall is used; otherwise a no-slip wall is assumed.) With these remarks in mind, one takes the boundary conditions at a no-slip wall to be

$$v_{ij-\frac{1}{2}} = 0 , \tag{2.28}$$

$$u_{i-\frac{1}{2}j-1} = - u_{i-\frac{1}{2}j} , \quad u_{i+\frac{1}{2}j-1} = - u_{i+\frac{1}{2}j} , \tag{2.22}$$

and

$$\phi_{ij-1} = \phi_{ij} - g_y \delta y - \frac{2\nu}{\delta y}(v_{ij+\frac{1}{2}} - v_{ij-\frac{1}{2}}) . \tag{2.25}$$

Case 4: Free-slip wall. Again using the above remarks, the boundary conditions at a free-slip wall are given by (2.28), (2.26), and

$$\phi_{ij-1} = \phi_{ij} - g_y \delta y . \tag{2.29}$$

Case 5: Free surface. A free surface is defined to be a boundary of the fluid region which is in contact with another fluid (usually air) whose effects on the fluid being examined are negligible. In this paper the pressure immediately below the free surface (referring still to Figure 2) is assumed to be negligible compared to that above; thus $\phi_{ij-1} \simeq 0$; whereas in the report by Welch (1966), the fluid below may apply some known additional pressure on the free surface. Under this assumption, the boundary conditions at a free surface are given by the following (Welch, 1966).

If only one empty cell, i.e., a cell outside the fluid region, touches the free surface cell (cell i, j), then the velocity at the surface is computed from Equation (2.9). If this empty cell is above or below cell i, j, then

$$\phi_{ij} = \frac{2\nu}{\delta y}(v_{ij+\frac{1}{2}} - v_{ij-\frac{1}{2}}) . \tag{2.30}$$

If it is to the left or right, then

$$\phi_{ij} = \frac{2\nu}{\delta x}(u_{i+\frac{1}{2}j} - u_{i-\frac{1}{2}j}) . \tag{2.31}$$

If two empty cells contact the free surface, then the velocities at the surfaces are set equal to those on the opposite sides of the cell. If the two empty cells are i, j+1 and i+1, j, then

$$\phi_{ij} = \frac{\nu}{2}\left[\frac{1}{\delta y}\left(u_{i-\frac{1}{2}j}+u_{i+\frac{1}{2}j}-u_{i-\frac{1}{2}j-1}-u_{i+\frac{1}{2}j-1}\right)+\frac{1}{\delta x}\left(v_{ij+\frac{1}{2}}+v_{ij-\frac{1}{2}}-v_{i-1j+\frac{1}{2}}-v_{i-1j-\frac{1}{2}}\right)\right]. \tag{2.32}$$

Similar conditions apply to ϕ for the other three configurations of diagonally-adjacent empty cells. If the two empty cells are not diagonally adjacent, then

$$\phi_{ij} = 0 . \tag{2.33}$$

If three empty cells surround the free surface cell, then the velocity opposite the fluid side is set equal to that of the fluid side, and the other two surface velocities are changed only by gravity, e.g.,

$$v^{n+1} = v + g_y \delta t . \tag{2.34}$$

For the case of three empty cells, (2.33) holds for ϕ.

Finally, if the free surface cell is alone, i.e., has four empty cells surrounding it, then it follows a free fall trajectory. That is, all four velocities are changed only by gravity. Again, condition (2.33) applies for ϕ.

Case 6: Corners. Since, as explained later in the section covering the algorithm, the pressure of a cell of fluid is computed immediately after any boundary conditions needed for that cell, corners do not change the conditions on ϕ. This is true even though the pressure in the corner cell will be different depending on which

adjacent fluid cell is being considered.

Velocity boundary conditions, however, are different. Referring to Figure 4, the following boundary conditions are necessary (Welch, 1966):

In the calculation of $v^{n+1}_{ij+\frac{1}{2}}$,

$$v_{i+1j+\frac{1}{2}} = v_{ij+\frac{1}{2}} \tag{2.35}$$

for a free-slip corner, and

$$v_{i+1j+\frac{1}{2}} = - v_{ij+\frac{1}{2}} \tag{2.36}$$

for a no-slip corner.

In the calculation of S_{ij} (only),

$$u_{i+\frac{1}{2}j+1} = 0 \, . \tag{2.37}$$

In the calculation of S_{ij+1} (only),

$$u_{i+\frac{1}{2}j} = u_{i+\frac{1}{2}j+1} \tag{2.38}$$

for free-slip, and

$$u_{i+\frac{1}{2}j} = - u_{i+\frac{1}{2}j+1} \tag{2.39}$$

for no-slip.

Similar conditions hold for the other three orientations of a corner cell.

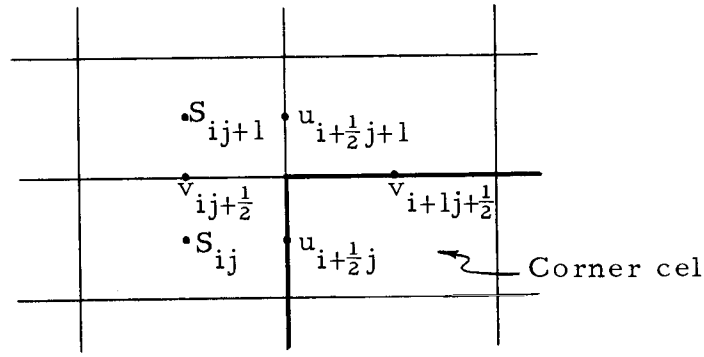


Figure 4. Typical corner cell.

To summarize the first two sections of this chapter, the following tables have been prepared. Table 2.1 lists the unknowns, equations, and basic assumptions pertaining to the MAC Method equations; Table 2.2 shows the initial conditions, boundary conditions, and assumptions used in the derivation of the boundary conditions.

Algorithm

With the equations and associated boundary conditions listed in the previous sections, all that remains is to generate a series of steps to use these results. It is this algorithm, along with the boundary conditions, which makes the MAC Method both unique and powerful.

At the heart of the algorithm lies a concept which has yet to be introduced. This idea is the "existence" of imaginary markers, or

Table 2.1. Unknowns, equations, and basic assumptions of the MAC Method.

Unknowns	Equations	Equation numbers	Assumptions
u	$$u^{n+1}_{i+\frac{1}{2}j} = u_{i+\frac{1}{2}j} + \delta t\left[\frac{(u_{ij})^2-(u_{i+1j})^2}{\delta x} + \frac{(uv)_{i+\frac{1}{2}j-\frac{1}{2}}-(uv)_{i+\frac{1}{2}j+\frac{1}{2}}}{\delta y} + g_x + \frac{\phi_{ij}-\phi_{i+1j}}{\delta x} + \nu\left(\frac{u_{i+\frac{3}{2}j}+u_{i-\frac{1}{2}j}-2u_{i+\frac{1}{2}j}}{\delta x^2} + \frac{u_{i+\frac{1}{2}j+1}+u_{i+\frac{1}{2}j-1}-2u_{i+\frac{1}{2}j}}{\delta y^2}\right)\right].$$	(2.10)	
v	$$v^{n+1}_{ij+\frac{1}{2}} = v_{ij+\frac{1}{2}} + \delta t\left[\frac{(v_{ij})^2-(v_{ij+1})^2}{\delta y} + \frac{(uv)_{i-\frac{1}{2}j+\frac{1}{2}}-(uv)_{i+\frac{1}{2}j+\frac{1}{2}}}{\delta x} + g_y + \frac{\phi_{ij}-\phi_{ij+1}}{\delta y} + \nu\left(\frac{v_{ij+\frac{3}{2}}+v_{ij-\frac{1}{2}}-2v_{ij+\frac{1}{2}}}{\delta y^2} + \frac{v_{i+1j+\frac{1}{2}}+v_{i-1j+\frac{1}{2}}-2v_{ij+\frac{1}{2}}}{\delta x^2}\right)\right].$$	(2.11)	
ϕ	$$\phi_{ij} = \frac{1}{2\left(\frac{1}{\delta x^2}+\frac{1}{\delta y^2}\right)}\left(\frac{\phi_{i+1j}+\phi_{i-1j}}{\delta x^2} + \frac{\phi_{ij+1}+\phi_{ij-1}}{\delta y^2} + S_{ij}\right).$$	(2.15)	

Table 2. 1. Continued.

Unknowns	Equations	Equation numbers	Assumptions
	$S_{ij} = \frac{(u_{i+1j})^2+(u_{i-1j})^2-2(u_{ij})^2}{\delta x^2} + \frac{(v_{ij+1})^2+(v_{ij-1})^2-2(v_{ij})^2}{\delta y^2}$		1. Incompressible flow.
	$+\frac{2}{\delta x \delta y}[(uv)_{i+\frac{1}{2}j+\frac{1}{2}}+(uv)_{i-\frac{1}{2}j-\frac{1}{2}}-(uv)_{i+\frac{1}{2}j-\frac{1}{2}}-(uv)_{i-\frac{1}{2}j+\frac{1}{2}}]$		2. Constant density.
	$-\frac{D_{ij}}{\delta t}-\nu\left(\frac{D_{i+1j}+D_{i-1j}-2D_{ij}}{\delta x^2}+\frac{D_{ij+1}+D_{ij-1}-2D_{ij}}{\delta y^2}\right)$.	(2. 16)	3. Two-dimensional space. 4. Constant kinematic viscosity. 5. Constant gravity.
	$D_{ij} = \frac{u_{i+\frac{1}{2}j}-u_{i-\frac{1}{2}j}}{\delta x} + \frac{v_{ij+\frac{1}{2}}-v_{ij-\frac{1}{2}}}{\delta y}$.	(2. 12)	6. Discrete fluid region. 7. $D_{ij}^{n+1} = 0$, all i, j.

Table 2.2. Initial conditions, boundary conditions, and boundary condition assumptions.

Initial conditions	Numbers	Boundary conditions for bottom boundary (Figure 2)			
		Type of boundary	Conditions	Numbers	Assumptions
$u^0_{ij} = f_1(i,j)$.	(2.17)	Input	$v^{n+1}_{ij-\frac{1}{2}} = v_{ij-\frac{1}{2}} = v_{input}$.	(2.21)	Input velocity constant with respect to time.
$v^0_{ij} = f_2(i,j)$.	(2.18)		$u_{i-\frac{1}{2}j-1} = -u_{i-\frac{1}{2}j}$, $u_{i+\frac{1}{2}j-1} = -u_{i+\frac{1}{2}j}$.	(2.22)	Input velocity normal to wall.
			$\phi_{ij-1} = \phi_{ij} - g_y\delta y - \frac{2\nu}{\delta y}(v_{ij+\frac{1}{2}} - v_{ij-\frac{1}{2}})$.	(2.25)	Input velocity constant along wall.
		Output	$v_{ij-\frac{1}{2}} = v_{ij+\frac{1}{2}} + \frac{\delta y}{\delta x}(u_{i+\frac{1}{2}j} - u_{ij-\frac{1}{2}})$.	(2.23)	
			$u_{i-\frac{1}{2}j-1} = u_{i-\frac{1}{2}j}$, $u_{i+\frac{1}{2}j-1} = u_{i+\frac{1}{2}j}$.	(2.26)	
			$\phi_{ij-1} = \phi_{ij} - \frac{\delta y}{\delta x}[(uv)_{i-\frac{1}{2}j-\frac{1}{2}} - (uv)_{i+\frac{1}{2}j-\frac{1}{2}}]$.	(2.27)	Formulated to agree with experiment.
		No-slip wall	$v_{ij-\frac{1}{2}} = 0$.	(2.28)	Solid wall.
			$u_{i-\frac{1}{2}j-1} = -u_{i-\frac{1}{2}j}$, $u_{i+\frac{1}{2}j-1} = -u_{i+\frac{1}{2}j}$.	(2.22)	$u\|_{wall} = 0$.
			$\phi_{ij-1} = \phi_{ij} - g_y\delta y - \frac{2\nu}{\delta y}(v_{ij+\frac{1}{2}} - v_{ij-\frac{1}{2}})$.	(2.25)	
		Free-slip wall	$v_{ij-\frac{1}{2}} = 0$.	(2.28)	Solid wall.
			$u_{i-\frac{1}{2}j-1} = u_{i-\frac{1}{2}j}$, $u_{i+\frac{1}{2}j-1} = u_{i+\frac{1}{2}j}$.	(2.26)	$u\|_{wall} = u\|_{(wall + \delta y/2)}$.
			$\phi_{ij-1} = \phi_{ij} - g_y\delta y$.	(2.29)	$\nu\|_{wall} = 0$.

Table 2.2. Continued.

Free surface boundary conditions (Figure 2)*			Corner boundary conditions (Figure 4)	
Number of empty cells surrounding	Conditions	Numbers	Condition	Number
One	$v\|_{surface}$ computed from $D_{ij} = 0.$	(2.9)	In calculation of $v^{n+1}_{ij+\frac{1}{2}}$,	
	If empty cell is		free-slip:	
	above or below: $\phi_{ij} = \frac{2\nu}{\delta y}(v_{ij+\frac{1}{2}} - v_{ij-\frac{1}{2}})$;	(2.30)	$v_{i+1j+\frac{1}{2}} = v_{ij+\frac{1}{2}}$;	(2.35)
	to left or right: $\phi_{ij} = \frac{2\nu}{\delta x}(u_{i+\frac{1}{2}j} - u_{i-\frac{1}{2}j})$.	(2.31)	no-slip:	
Two	$v\|_{surface} = v\|_{opposite}$.		$v_{i+1j+\frac{1}{2}} = - v_{ij+\frac{1}{2}}$.	(2.36)
	If empty cells are		In calculation of S_{ij} (only),	
	diagonally adjacent: $\phi_{ij} = f(\nu, u, v)$;	(2.32)	$u_{i+\frac{1}{2}j+1} = 0.$	(2.37)
	otherwise: $\phi_{ij} = 0.$	(2.33)	In calculation of S_{ij+1} (only),	
Three	v opposite fluid side = v fluid side;		free-slip:	
	otherwise $v^{n+1} = v + g\delta t$.	(2.34)	$u_{i+\frac{1}{2}j} = u_{i+\frac{1}{2}j+1}$;	(2.38)
	$\phi_{ij} = 0.$	(2.33)	no-slip:	
Four	$v^{n+1} = v + g\delta t$, all sides.	(2.34)	$u_{i+\frac{1}{2}j} = - u_{i+\frac{1}{2}j+1}$.	(2.39)
	$\phi_{ij} = 0.$	(2.33)		

*Assuming $\phi_{ij} = 0$.

particles of fluid, which move through the fluid region with velocities associated with the cell velocities. If these particles were to enter into the calculations, they would form, in effect, a Lagrangian coordinate system. However, they do not affect the numerical results per se. Instead they are used only to locate a free surface, if present, as it moves about the system. An excellent secondary application for these markers is to allow the investigator to watch the movement of the fluid, as if the markers were particles of dye in the corresponding physical problem. With the introduction of these particles, the algorithm can now be described.

The calculations begin with the problem setup, in which the fluid region is described and initial conditions are applied. After this, the algorithm consists of a series of time cycles, each of length δt. Each time cycle is broken down into the following steps (see Figure 5).

(1) The free surface, if present, is located by counting the number of particles in each cell. (A cell is "empty" if it contains no particles, thus explaining the terminology used in the previous section.) At this time, the free surface boundary conditions on ϕ are applied. Also, velocities are updated for those cells which have changed from the previous time cycle; e.g., a free surface cell may become empty, in which case its velocities are set equal to zero.

(2) Using velocities from the previous cycle, D is computed

for each non-empty cell from Equation (2. 12), and its absolute value is checked against a tolerance. If any D is too large, the calculations must be stopped and the stability criteria should be examined. (The stability requirements are mentioned later in this section.)

(3) The values of S for each full (non-empty and not free surface) cell are now obtained from Equation (2. 16).

(4) ϕ is calculated for all full cells by using Equation (2. 15). This step proves to be the major part of the problem. It may be seen that in order to solve (2. 15) for ϕ, an iteration, or relaxation, process is needed. In the original MAC Method (Welch, 1966), the process used is Seidel's Method, or simple relaxation (see Chapter III). In this paper, the MAC Method is strengthened by using a much faster process called overrelaxation (Chapters III and IV). Since this step (calculating ϕ) usually involves more than double the amount of calculations as the rest of a time cycle, a method for reducing these calculations would be valuable.

The MAC Method procedure for solving (2. 15) is as follows: starting with the lower left-hand full cell of the mesh, any boundary conditions on ϕ necessary for this cell are computed, and then ϕ is computed from (2. 15), always using the latest possible values of the surrounding ϕ's; if a value of ϕ for a surrounding cell is not available from this time cycle, the last value from the previous cycle is used. This process is repeated for each full cell in the mesh, thus

making one iteration, or sweep. Ten such iterations are completed before a check is made to determine if the pressure field is still changing between iterations. If the change is sufficiently small (using a predefined tolerance), the iterations are stopped and step (5) in the time cycle is taken; if the field is still changing, ten more iterations are run and another check is made. This process is repeated until the pressure field converges.

(5) During this step, u^{n+1} and v^{n+1} are computed from Equations (2.10) and (2.11), respectively, using old velocities, the new pressures, and the velocity boundary conditions.

(6) The particles are now moved with a velocity equal to the weighted average of the four nearest cell velocities. This procedure is described in the report by Welch (1966).

Steps (1) through (6) comprise one time cycle and can be repeated as many times as needed for the solution of a fluid problem.

The stability criteria for this procedure are reported to be (Welch, 1966):

$$C\delta t < \frac{2\delta x \delta y}{\delta x + \delta y}, \tag{2.40}$$

where C is the wave speed of the fluid, and

$$2\nu\delta t < \frac{\delta x^2 \delta y^2}{\delta x^2 + \delta y^2}. \tag{2.41}$$

In addition, Shannon (1967) reports that the following criteria should also be met:

$$\delta t < \frac{\delta x^2}{4\nu} , \tag{2.42}$$

$$\delta t^2 < \frac{\delta x^2}{u_{max}^2} , \tag{2.43}$$

$$\delta t < \frac{\delta x}{5u_{input}} , \tag{2.44}$$

and

$$\frac{1}{2}\delta t\, u_{max}^2 + \frac{1}{4}\delta x^2 \frac{\partial(u_{max})}{\delta x} < \nu . \tag{2.45}$$

Similar inequalities hold in the y direction.

The algorithm as described above has been made into a computer program, called SPLIT, and used to examine several typical fluid flow problems. A flow chart for this program is included on the next page and a listing can be found in Appendix II. In addition, some examples of problem solutions are the subject of Chapter V.

Extensions

Although the MAC Method as so far described is a powerful tool for solving fluid flow problems, its true strength lies in the fact that it can be extended in many ways to solve any number of more sophisticated fluid problems.

As reported by Harlow (Welch et al., 1966), some extensions

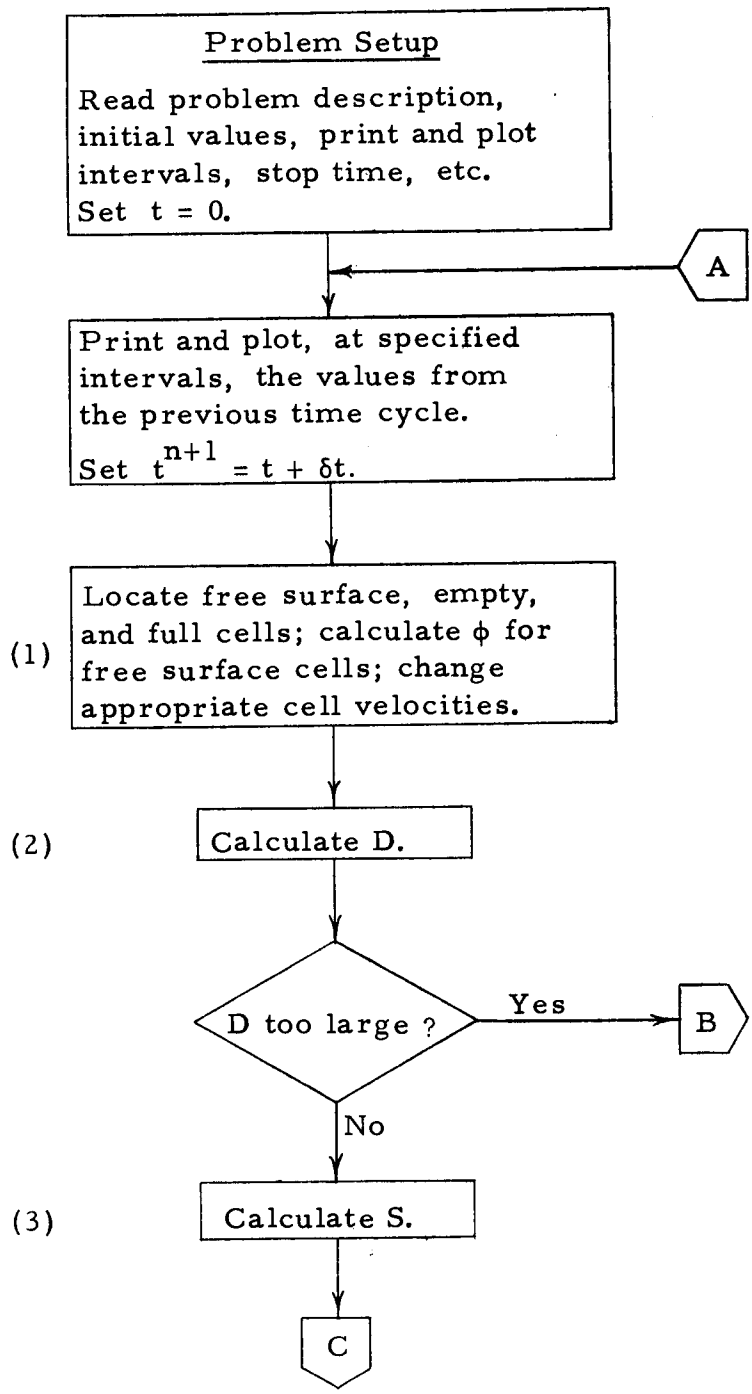


Figure 5. SPLIT flow chart.

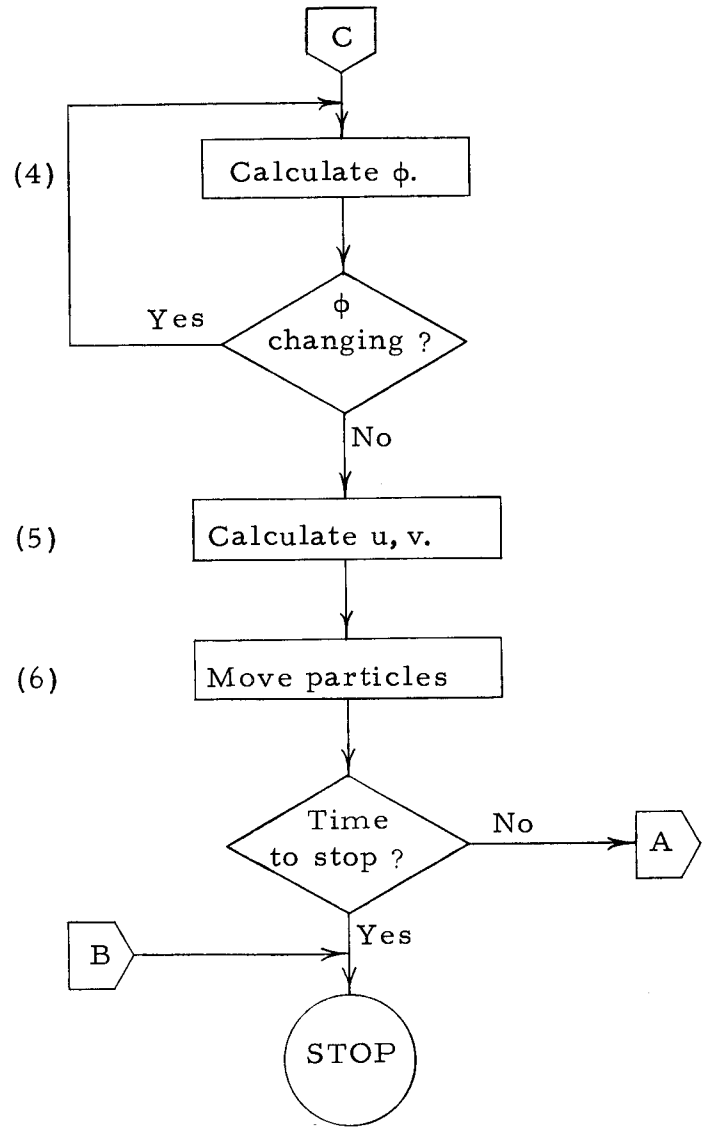


Figure 5. Continued.

include (1) cylindrical coordinates, (2) accelerating systems, (3) three-dimensional coordinates, (4) turbulence, and (5) heat transport. The heat transport problem has been extended even further by Mercier (1968) to include density variations. By adding more equations (energy equation, incompressibility relation), Mercier's analysis can accomodate problems involving density- and temperature-stratified flows, such as reservoir problems.

Another useful addition to the MAC Method is described by Daly (Welch et al., 1966). This admits the solution of problems with two confined (no inflow, outflow, or free surface) incompressible fluids of different density. Using Daly's extension, the interface between the two fluids may be followed throughout the solution. A computer program, called SPLASH, has been written for this method and a sample problem solution is presented in Chapter V. The listing of the program appears in Appendix III.

Hwang (1968) has further strengthened this two-fluid analysis by adding inflow, outflow, and free surfaces. His dissertation also includes error and stability analyses.

III. THE TECHNIQUE OF OVERRELAXATION

When stated in their most generalized form, the physical laws governing the nature of the universe are usually expressed as integral equations. In all but highly-simplified cases, however, these integral equations are next to impossible to solve analytically; likewise, the few numerical integral-equation-solving techniques available are very cumbersome. However, a system of integral equations may be replaced with an equivalent system of partial differential equations either by making a differential analysis or by employing well-known theorems by Green and Stokes. Examples of both of these techniques applied to the equations of motion for fluids may be found in Shapiro (1953). Once a system of partial differential equations is obtained, it may be solved in many different ways, the most common method being the scheme of finite differencing.

Finite difference solutions are usually accomplished in the manner described in Chapter II. A system of partial differential equations is approximated with difference equations, using difference quotients in place of partial derivatives. One of these difference equations is solved implicitly; then the solution of the rest of the equations follows by substitution. There are several schemes available for the implicit solution of a difference equation; one efficient method is an iteration process called overrelaxation. A description of this technique is

contained in the sequel.

Description

The following discussion of overrelaxation is based on a treatment by Forsythe and Wasow (1960), although the original work was done by Young (1954). The theory presented covers the special case of elliptic partial differential equations of functions of two variables.

Given a function of two variables, $w = w(x, y)$, and a partial differential equation of the form

$$a \frac{\partial^2 w}{\partial x^2} + 2b \frac{\partial^2 w}{\partial x \partial y} + c \frac{\partial^2 w}{\partial y^2} + d \frac{\partial w}{\partial x} + e \frac{\partial w}{\partial y} + fw + g = 0 , \qquad (3.1)$$

the equation is said to be elliptic if

$$ac - b^2 > 0 . \qquad (3.2)$$

A typical set of problems involving elliptic equations is the following. Let R denote a region in the x, y plane. The problem is to solve for w in the equation

$$\frac{\partial^2 w(x, y)}{\partial x^2} + \frac{\partial^2 w(x, y)}{\partial y^2} = f(x, y) , \qquad (3.3)$$

for (x, y) in R, subject to the boundary condition

$$g_1(x', y')\frac{\partial w(x', y')}{\partial n} + g_2(x', y')w(x', y') + g_3(x', y') = 0 , \qquad (3.4)$$

for (x', y') on the boundary of R. Here, $\frac{\partial w(x', y')}{\partial n}$ is the derivative (at x', y') in the direction normal to the boundary. Only problems of this type will be considered here.

Let δx and δy be the spacing in the x and y directions, respectively, of a grid which covers R. (It is assumed that the boundary of R is the union of a finite number of straight lines, each of which is either horizontal or vertical, although the technique described below will work for other configurations either by changing the coordinate system or by interpolating at the boundary.) Then the partial derivatives in (3.3) may be written

$$\frac{\partial^2 w_0}{\partial x^2} \simeq \frac{w_1 - 2w_0 + w_2}{\delta x^2} \qquad (3.5)$$

and

$$\frac{\partial^2 w_0}{\partial y^2} \simeq \frac{w_3 - 2w_0 + w_4}{\delta y^2} , \qquad (3.6)$$

where w_0, w_1, w_2, w_3, and w_4 are defined in Figure 6. Thus, using (3.5) and (3.6) and rearranging, Equation (3.3) may by approximated with

$$w_0 = \frac{1}{2\left(\frac{1}{\delta x^2} + \frac{1}{\delta y^2}\right)}\left[\frac{w_1 + w_2}{\delta x^2} + \frac{w_3 + w_4}{\delta y^2} - f(x_0, y_0)\right] , \qquad (3.7)$$

where $w_0 = w(x_0, y_0)$. Hence, by using the boundary condition (3.4), one may solve (3.7) implicitly to obtain values of w at a finite number of points in R; the number of points is dependent on the relation of δx and δy to the total dimensions of R. There are several schemes available for solving (3.7), the easiest of which is an iteration process called simple iteration. This technique consists of the following steps.

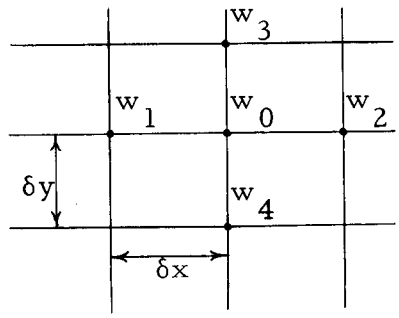


Figure 6. Illustration of the grid covering R.

(1) An initial guess (usually zero) is made for the function w at each point (x_i, y_j) of the mesh (except, of course, at the boundaries). Call these initial values $w_{ij}^{(0)}$.

(2) Using Equation (3.7), new values, called $w_{ij}^{(1)}$, are computed using $w_{ij}^{(0)}$ and boundary values.

(3) The difference between the new values and the old values is checked against a tolerance. If the values of $w_{ij}^{(1)}$ are too

far different from those of $w_{ij}^{(0)}$, new values, $w_{ij}^{(2)}$, are computed from $w_{ij}^{(1)}$ as in step (2).

(4) Steps (2) and (3) are repeated until, for some k, the $w_{ij}^{(k)}$'s are sufficiently close to the $w_{ij}^{(k-1)}$'s. At this point the iterations are stopped, the solution is said to have converged, and the process of simple iteration is said to "work" for this problem.

A more efficient scheme, called Seidel's Method or simple relaxation, is the same as simple iteration except for one refinement. Instead of using only quantities from the previous iteration to compute new ones, simple relaxation uses new values, as soon as they have been determined, in the calculation of other new values. It is clear that the time saved by using this process will be dependent on the order in which the points are taken during an iteration. An order which takes maximum advantage of the refinement over simple iteration is called a consistent order. One such ordering (Forsythe and Wasow, 1960) is to start with the lower left-hand point, work across to the right, then left-to-right on the next higher row of points. This is continued throughout the mesh, ending with the upper right-hand point. It can be shown (Forsythe and Wasow, 1960) that if simple iteration "works" whenever a consistent order is used, then simple relaxation "works" exactly twice as fast.

An even more efficient method for solving Equation (3.7) exists. This scheme, called overrelaxation, speeds the convergence of simple relaxation by multiplying the changes between iterations by a fixed number greater than one. The following discussion will help to clarify this.

Define a new quantity, $R_0^{(k)}$, called the $\underline{k^{th}}$ $\underline{\text{residual of}}$ $\underline{w_0}$, in the following way:

$$R_0^{(k)} \equiv \left\{ \frac{1}{2\left(\frac{1}{\delta x^2}+\frac{1}{\delta y^2}\right)} \left[\frac{w_1^{(\ell)}+w_2^{(\ell)}}{\delta x^2} + \frac{w_3^{(\ell)}+w_4^{(\ell)}}{\delta y^2} - f(x_0, y_0) \right] \right\}^{(k)} - w_0^{(k-1)}, \tag{3.8}$$

where the superscript ℓ has the value of either k or $k-1$.

It is obvious that Equation (3.7) can now be written:

$$w_0^{(k)} = w_0^{(k-1)} + R_0^{(k)} . \tag{3.9}$$

This is the equation which is used in simple relaxation. A more general equation can be written to cover all types of relaxation processes:

$$w_0^{(k)} = w_0^{(k-1)} + qR_0^{(k)} . \tag{3.10}$$

When $q < 1$, the process is termed underrelaxation; when $q = 1$, the process is the simple relaxation already discussed; and when

$q > 1$, it is called overrelaxation.

Forsythe and Wasow (1960) have shown that when a consistent order is used, if simple relaxation "works," then overrelaxation "works." The amount of time saved by using overrelaxation will, of course, depend on the overrelaxation factor, q. Using a matrix analysis of the operations involved, a relation between the rate of convergence and the overrelaxation parameter may be obtained. This relation is depicted in Figure 7; several observations can be made from this curve.

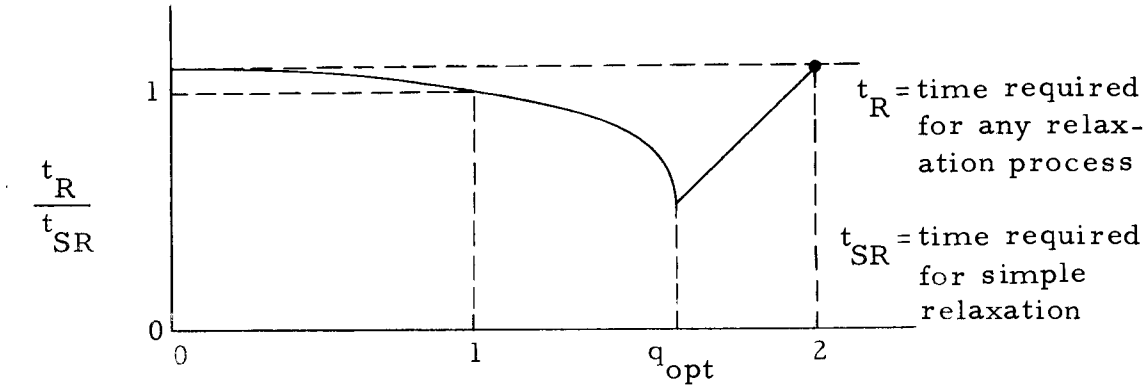


Figure 7. Overrelaxation factor curve.

(1) Underrelaxation is not profitable; it requires more time than any other relaxation method.

(2) An optimum overrelaxation factor, q_{opt}, exists.

(3) Although q_{opt} depends on the problem being considered, its value lies in the interval $1 < q_{opt} < 2$.

(4) Approaching q_{opt} from the left, the curve has an infinite slope, while the slope is one for $q \geq q_{opt} + 0$. Thus it is better to use $q = q_{opt} + \epsilon$ than to use $q = q_{opt} - \epsilon$, for some small $\epsilon > 0$.

In general, determination of the best overrelaxation factor to use cannot be done exactly. The next section describes a method for finding q_{opt} approximately, and also shows how to obtain q_{opt} exactly for the special case when R is a rectangle.

Determination of Overrelaxation Factor

As was noted above, the best overrelaxation factor cannot, in general, be computed exactly. However, a method, shown below, does exist for estimating q_{opt} (Forsythe and Wasow, 1960).

If a problem is started by using simple relaxation ($q = 1$), then an estimate of the rate of convergence, r, will be given by

$$\frac{\| R^{(k)} \|}{\| R^{(k-1)} \|} \rightarrow r \quad \text{as} \quad k \rightarrow \infty . \tag{3.11}$$

Any matrix norm will suffice for this estimate. Fortunately, a relation exists between r and q_{opt}:

$$q_{opt} = \frac{2}{1+\sqrt{1-r}} . \tag{3.12}$$

Thus, if a computer program is being used, one may run for, say, ten iterations using $q = 1$, form the quotient $\|R^{(10)}\|/\|R^{(9)}\|$, compute a new q from (3.12), and continue by using overrelaxation. It should be pointed out that the quotients in (3.11) will behave in a random manner when $q \neq 1$, while with $q = 1$ they will steadily decrease until r is reached.

The optimum overrelaxation parameter may be computed exactly for certain special cases. One of these is when the region R is a rectangle.

When R is a rectangle, the value of r in (3.11) may be computed exactly from (Davis, 1967):

$$r = \left(\frac{\cos \frac{\pi \delta x}{W} + \cos \frac{\pi \delta y}{H}}{2} \right)^2 , \qquad (3.13)$$

where δx and δy are as defined in the previous section, W is the width of the rectangle, and H is the height of the rectangle. As an example of this method, let $\delta x = \delta y = 0.1$, and $W = H = 1.0$. Substituting these values into Equation (3.13), we obtain

$$r = \cos^2 \frac{\pi}{10} .$$

Upon evaluating this cosine and substituting the resulting r into Equation (3.12), we have

$$q_{opt} = 1.53 .$$

IV. APPLICATION OF OVERRELAXATION TO MAC

The MAC Method, as described in Chapter II, can be used to simulate a great variety of fluid flow formations. Because of this diversity, the associated computer program is very large and relatively slow, owing to the complexity of its logic. As an example, a typical problem might require 50, 000 words of storage and one-half hour of computer time on the fastest of machines. Since computers with the required size and speed are very expensive to operate, it seems that any idea which could reduce the running time would be in order.

As pointed out earlier, the most time-consuming portion of the MAC Method is the solution of the pressure equation:

$$\phi_{ij} = \frac{1}{2\left(\frac{1}{\delta x^2}+\frac{1}{\delta y^2}\right)}\left(\frac{\phi_{i+1j}+\phi_{i-1j}}{\delta x^2}+\frac{\phi_{ij+1}+\phi_{ij-1}}{\delta y^2}+S_{ij}\right). \qquad (2.15)$$

Recall that this equation must be solved, i.e., the pressure field relaxed, at every time cycle. For most problems, this can be done in less than the minimum ten iterations (see page 26) for the majority of the time of interest. However, many interesting problems have heavy transient periods, during which the solution of (2.15) takes many times more than ten iterations when the process of simple relaxation is used. Thus the application of overrelaxation would accomplish the time-reduction mentioned above. The remainder of this

chapter is concerned with this application.

Application

Recalling the theory presented in Chapter III, overrelaxation will speed the convergence of the solution of the following problem.

Consider a function $w = w(x, y)$ and a region R in the x, y plane for which

$$\frac{\partial^2 w}{\partial x^2} + \frac{\partial^2 w}{\partial y^2} = f(x, y) \tag{3.3}$$

holds on the interior of R.

Suppose

$$g_1(x, y) \frac{\partial w}{\partial n} + g_2(x, y)w + g_3(x, y) = 0 \tag{3.4}$$

is satisfied on the boundary of R. The problem now is to solve for the function $w(x, y)$.

Hence, in order to justify the application of overrelaxation to the solution of Equation (2. 15), it must be shown that the associated boundary problem of (2. 15) is equivalent to the above problem. (It was shown in Chapter III that overrelaxation applies to the problem above.) This can be done very easily.

First, recalling that the equation

$$w_0 = \frac{1}{2\left(\frac{1}{\delta x^2}+\frac{1}{\delta y^2}\right)}\left[\frac{w_1+w_2}{\delta x^2}+\frac{w_3+w_4}{\delta y^2} - f(x_0, y_0)\right] \tag{3.7}$$

was shown to be the finite-difference form of (3. 3) and noting that the grid system for this equation is similar to that for Equation (2. 15), appropriate substitutions can be made to find that

$$\frac{\partial^2 \phi}{\partial x^2} + \frac{\partial^2 \phi}{\partial y^2} = - S'(x, y) \tag{4.1}$$

is the partial differential equation associated with (2. 15). Here, the function $S'(x, y)$ is the analog of S_{ij}. It is obvious that this equation is equivalent to (3. 3).

It remains to be seen that the boundary conditions for ϕ are in the form of Equation (3. 4). From Chapter II, the ϕ boundary conditions for input, output, no-slip wall, and free-slip wall boundaries are all of the form

$$\phi_{ij-1} = \phi_{ij} + \xi_{ij-\frac{1}{2}} , \tag{4.2}$$

where the boundary in question is the dividing line between cell i, j and cell $i, j-1$, and the term $\xi_{ij-\frac{1}{2}}$ is dependent on the type of boundary. Likewise, at a free surface the conditions are of the form

$$\phi_{ij} = \xi_{ij-\frac{1}{2}} ;$$

or, since $\phi_{ij-1} = 0$,

$$\phi_{ij-1} = \phi_{ij} - \xi_{ij-\frac{1}{2}} . \tag{4.3}$$

Now, recognizing that δy is a constant and that

$$\frac{\phi_{ij}-\phi_{ij-1}}{\delta y} \simeq \frac{\partial \phi}{\partial y} ,$$

it is seen that Equations (4. 2) and (4. 3) are the finite-difference forms of

$$\frac{\partial \phi}{\partial y} = \xi'(x, y) , \tag{4.4}$$

for (x, y) on the boundary.

However, since (4. 2) and (4. 3) apply at a bottom boundary, i. e., one which runs in the x direction, Equation (4. 4) is just

$$\frac{\partial \phi}{\partial n} = \xi'(x, y) . \tag{4.5}$$

But this is equivalent to Equation (3. 4) with $g_1(x, y) \equiv 1$, $g_2(x, y) \equiv 0$, and $g_3(x, y) = \xi'(x, y)$.

The two problems in question prove to be equivalent; it is therefore correct to assume that overrelaxation will result in a decrease in computation time when applied to the MAC Method. This has been done; the ensuing section will show the savings for problems involving

two types of transients.

Timing Studies

Two different types of problems have been investigated to determine the value of overrelaxation techniques when applied to the MAC Method. A short outline of these problems follows.

(1) This problem was devised to attempt to obtain an empirical relation for the savings produced by using overrelaxation. This was done by defining a problem with several transients, whose effects caused a varying amount of iterations required for convergence.

(2) The second problem was designed to illustrate the effects of under- and overestimating q_{opt} in a specific example (see Chapter III). This problem also serves as a check for problem (1).

Problem (1) consists of a channel with six steps in it (see Figure 8). Fluid begins entering from the left at time zero, flows over the steps, and eventually reaches point A. It was expected that as the fluid reached each step, more iterations would be required than when it reached the previous step. This proved to be true, as shown in Figure 9. The peaks of the curve in this figure correspond to the fluid encountering each step. The reader might wonder why there are seven peaks on this curve, while the channel has only six steps. This

can be explained by the following argument.

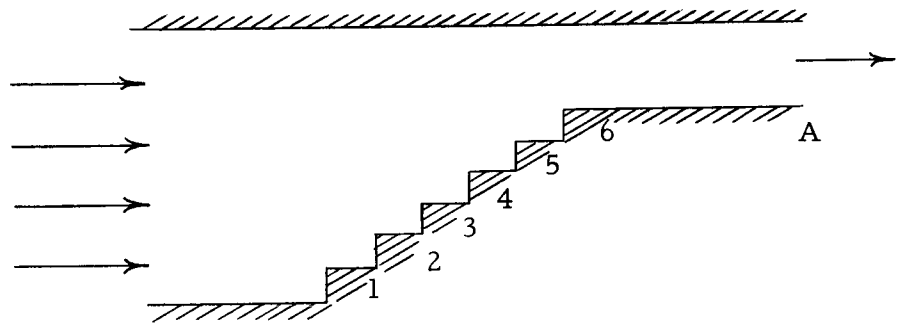


Figure 8. Problem with several transients.

In practice it is observed that as a fluid flows over an obstruction it accelerates. (Note that the peaks get closer together as time goes on, thus showing that the fluid does indeed accelerate.) After passing the sixth step, the fluid accelerates so rapidly that a "hole" is created above the step. Some of the stability criteria exhibited in Chapter II are based on the requirement that the fluid must not travel farther than one cell width in one time cycle (Welch, 1966). If this requirement is not met, then numerical instabilities will result and the solution will diverge. It follows that if the above-mentioned hole is large enough, then the solution will diverge. Thus, Figure 9 shows the divergence of the solution after the fluid passes the last step.

Problem (2) consists merely of a box full of fluid. Actually, because of the finite nature of the representation, the fluid is a small distance above the bottom of the box. A large gravitational force is

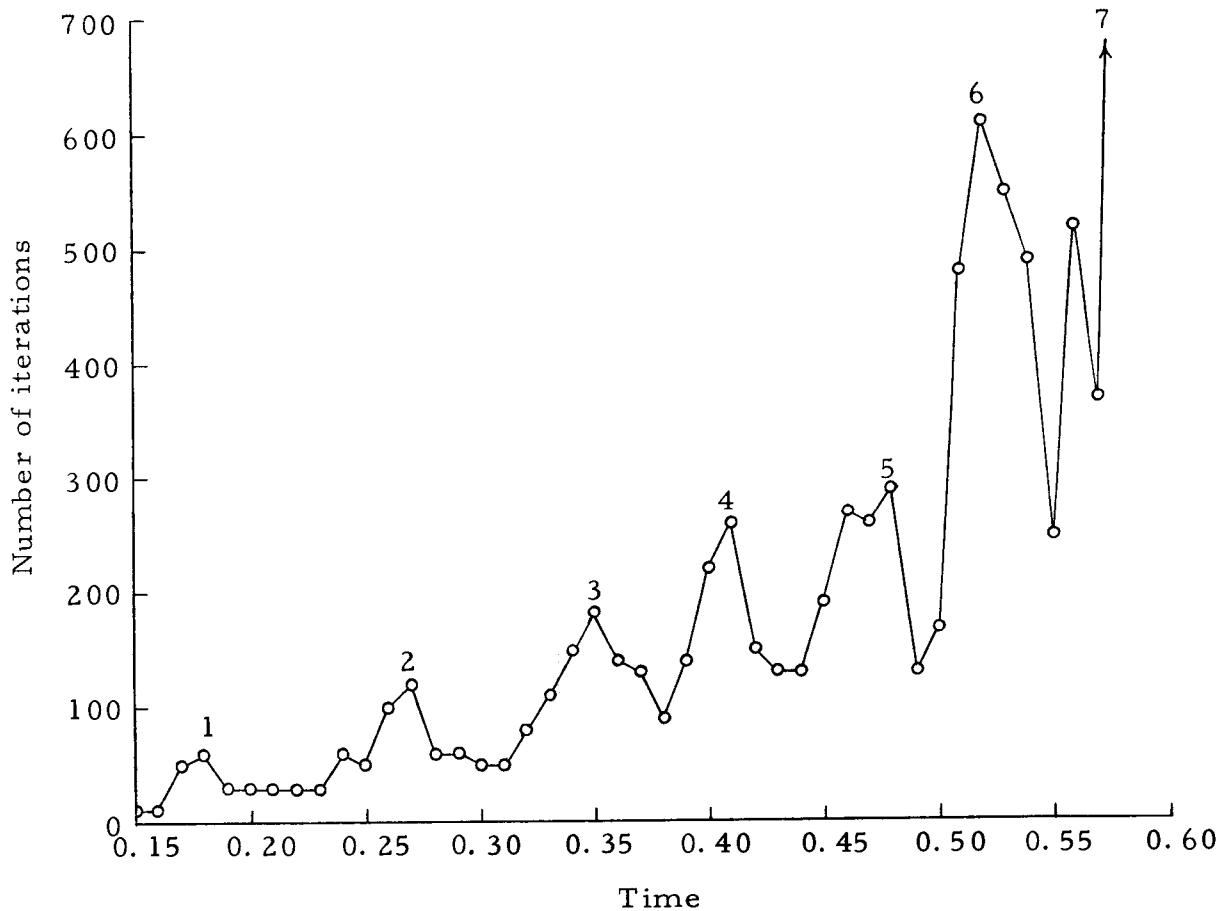


Figure 9. Effect of transients on number of iterations (problem of Figure 8).

allowed to pull the fluid down at time zero, thus causing a heavy transient during the first time cycle. Using simple relaxation ($q = 1$), the solution required 120 iterations. The exact optimum overrelaxation factor for this problem was computed in the example on page 39 and proved to be $q_{opt} = 1.53$. By using this q, the number of iterations was reduced to 50. The program was then asked to estimate q_{opt} after ten iterations, in the manner of Chapter III. It computed $q_{opt} = 1.61$, immediately used this value, and took a total of 60 iterations. A value of $q = 1.45$ was then tried, requiring 70 iterations, thus illustrating that it is better to overestimate q_{opt} than to underestimate, as implied in the discussion on page 38. (The number of iterations is always a multiple of ten since the convergence test is made only every tenth sweep.)

Results seem to indicate that the empirical relation sought in problem (1) does exist (Slotta et al., 1968). Figure 10 shows the number of iterations saved by using overrelaxation plotted against the number required by simple relaxation. Also, the corresponding point from problem (2) is plotted on this graph, thus serving as a check for problem (1). It can be seen that the relation is approximately linear, with a slope of $\simeq \frac{1}{2}$; i. e., the computing time is about halved. Remembering that these values were obtained with an estimated q_{opt}, the resulting 100% savings over simple relaxation show the superiority of the technique of overrelaxation.

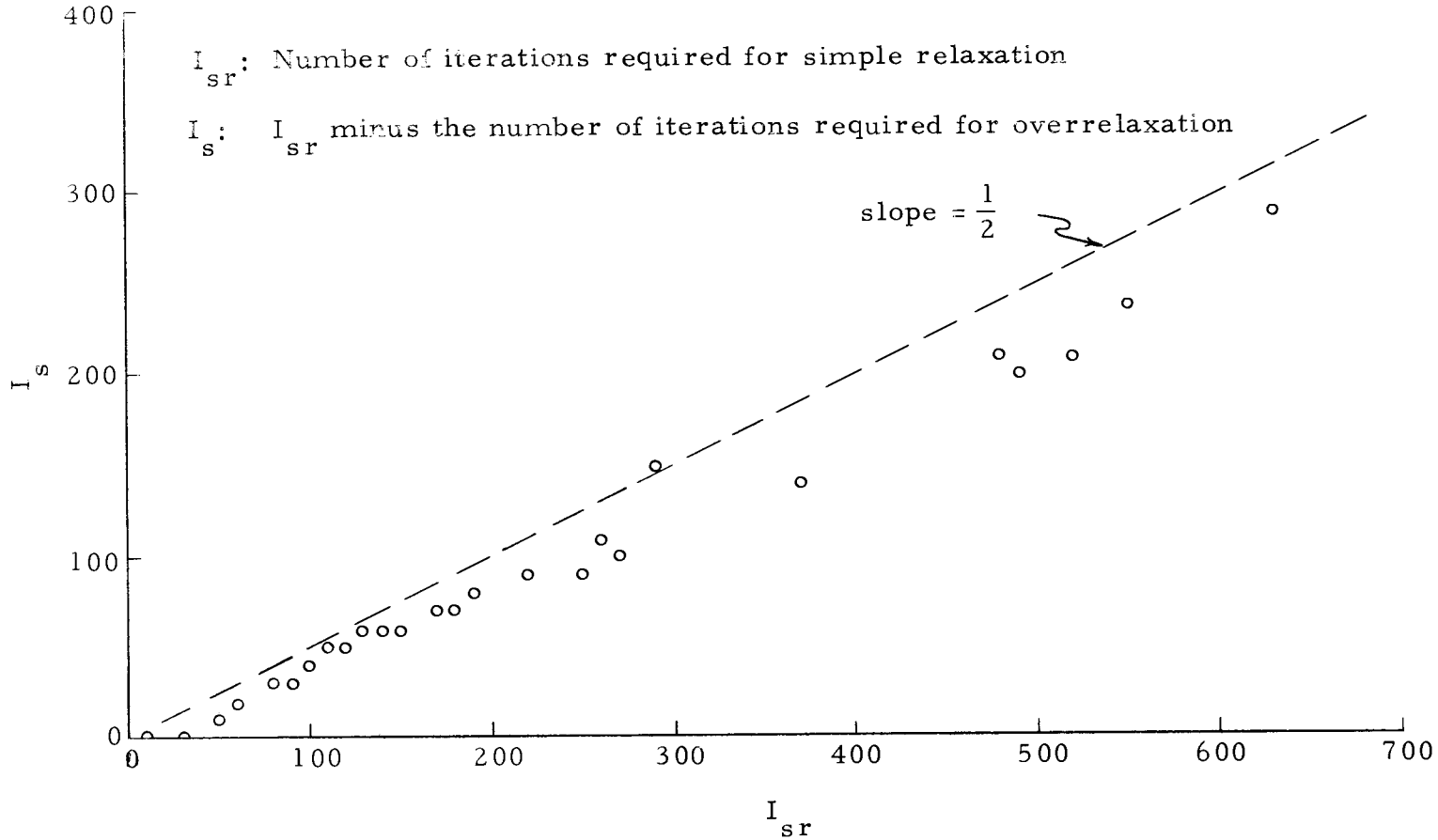


Figure 10. Overrelaxation savings.

V. EXAMPLES

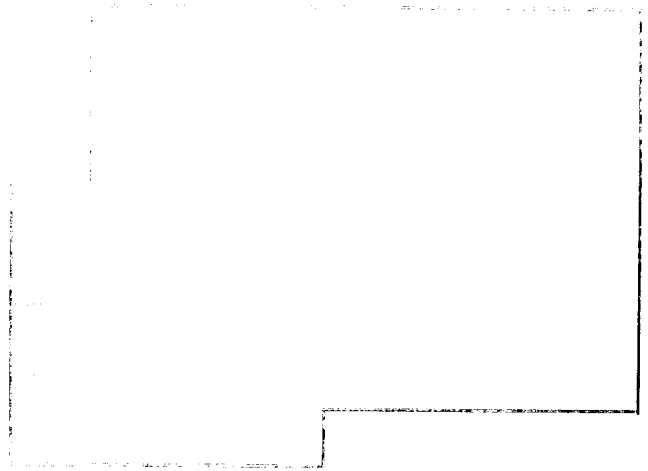
On the following pages are three examples of fluid flow problems which were simulated on a CDC 6600 computer with computer programs embodying the principles of the MAC Method.

The first two problems were solved by the FORTRAN IV program SPLIT, which uses the one-fluid MAC algorithm; the third problem solution was accomplished by the FORTRAN IV program SPLASH, which incorporates the two-fluid extension of the MAC Method.

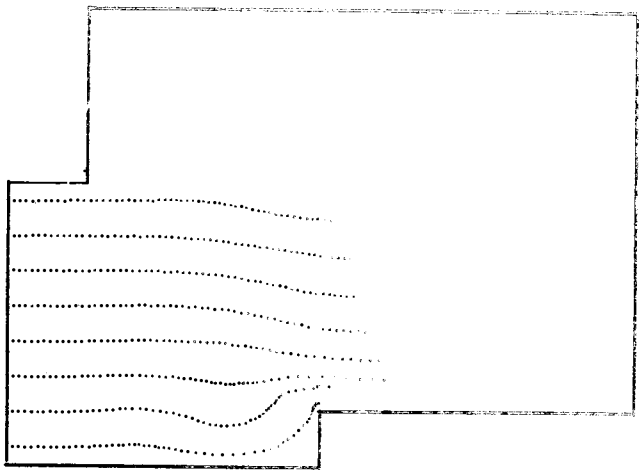
The first page of each example briefly explains the problem involved and lists values used for some of the variables. This is followed by a page of computer plots showing the fluid configuration at various points in the time period of interest. These plots are the representation of the marker particles mentioned in Chapter II, and were generated by the computer program on an SC-4020 microfilm recorder.

Stepped Channel with Free Surface

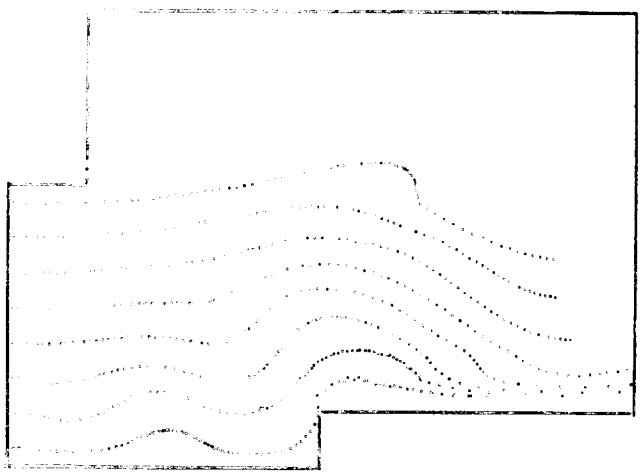
Fluid is allowed to enter a channel with a step in it (Figure 11). The fluid does not fill the channel; thus a free surface exists at the top. As the fluid passes over the step, a wave is formed at the free surface and eventually leaves the system. Further running of the problem shows that following waves get larger and hit the upper



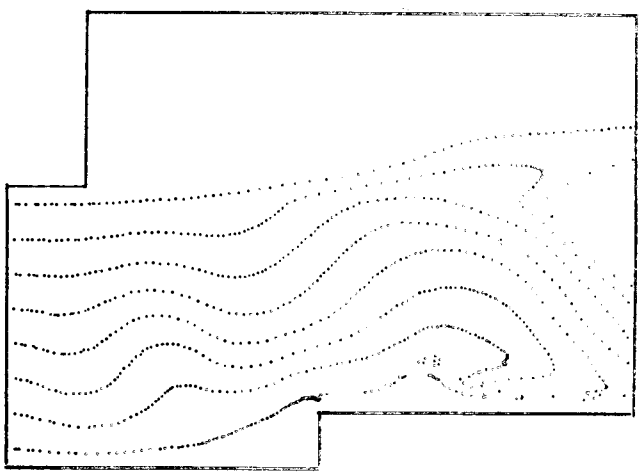
T = .200



T = 1.160



T = 1.800



T = 2.360

Figure 11. Stepped channel with free surface.

boundary, thus causing numerical instabilities which the program is not equipped to handle.

Number of cells = 20 x 15
$\delta x = 0.125$
$\delta y = 0.125$
Depth of fluid = 1.0
Height of step = 0.2
$\nu = 0.2$
$\delta t = 0.02$
$u^0 = 1.0$
$v^0 = 0.0$
$g_x = 0.0$
$g_y = -0.1$

Computer time: $\simeq$ 20 minutes

Fountain Flow

Fluid enters the box from below and proceeds upwards between retaining walls (Figure 12). As the fluid passes the tops of the walls, gravity starts pulling it out and down. Inertia then carries the fluid further upward. Eventually the fluid falls back out of the system.

Number of cells = 32 x 18
$\delta x = 0.1$
$\delta y = 0.1$
Height of retaining walls = 0.3
Width of fountain = 0.6
$\nu = 0.01$
$\delta t = 0.025$
$u^0 = 0.0$
$v^0 = 1.0$
$g_x = 0.0$
$g_y = -1.0$

Computer time: $\simeq$ 30 minutes

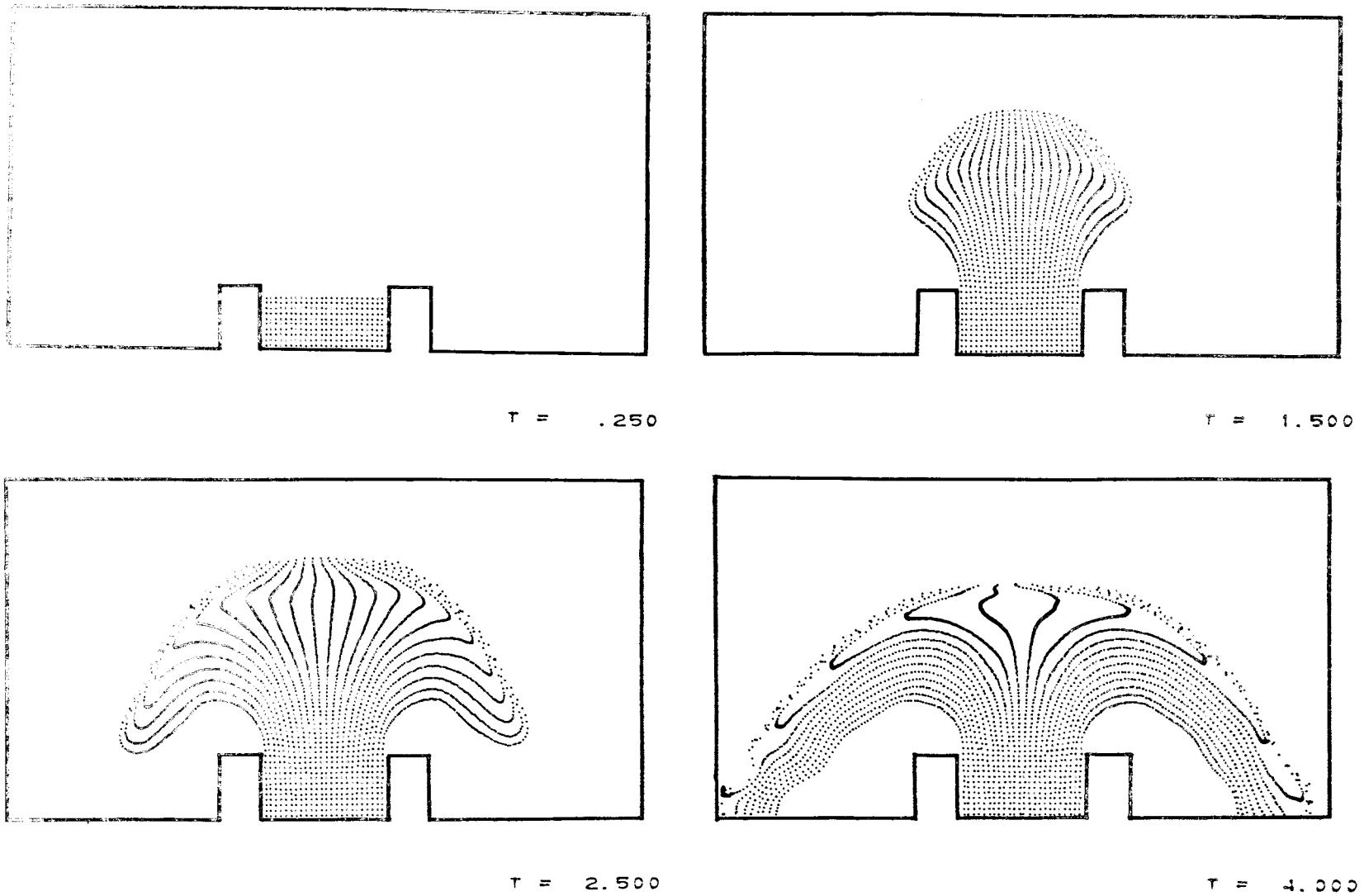


Figure 12. Fountain flow.

Mixing of Two Fluids

Two fluids lie side by side in a box, separated by a thin elastic diaphragm; the heavier is on the right (Figure 13). At time zero, the diaphragm is ruptured, allowing the heavier fluid to fall under the force of gravity.

Number of cells = 30 x 20

Number of particles = 2400

$\delta x = 0.1$

$\delta y = 0.1$

Density of heavy fluid = 1.0

Density of light fluid = 0.5

Viscosity (both fluids) = 0.0

$\delta t = 0.01$

$u^0 = 0.0$

$v^0 = 0.0$

$g_x = 0.0$

$g_y = -1.0$

Computer time: $\simeq$ 1 hour

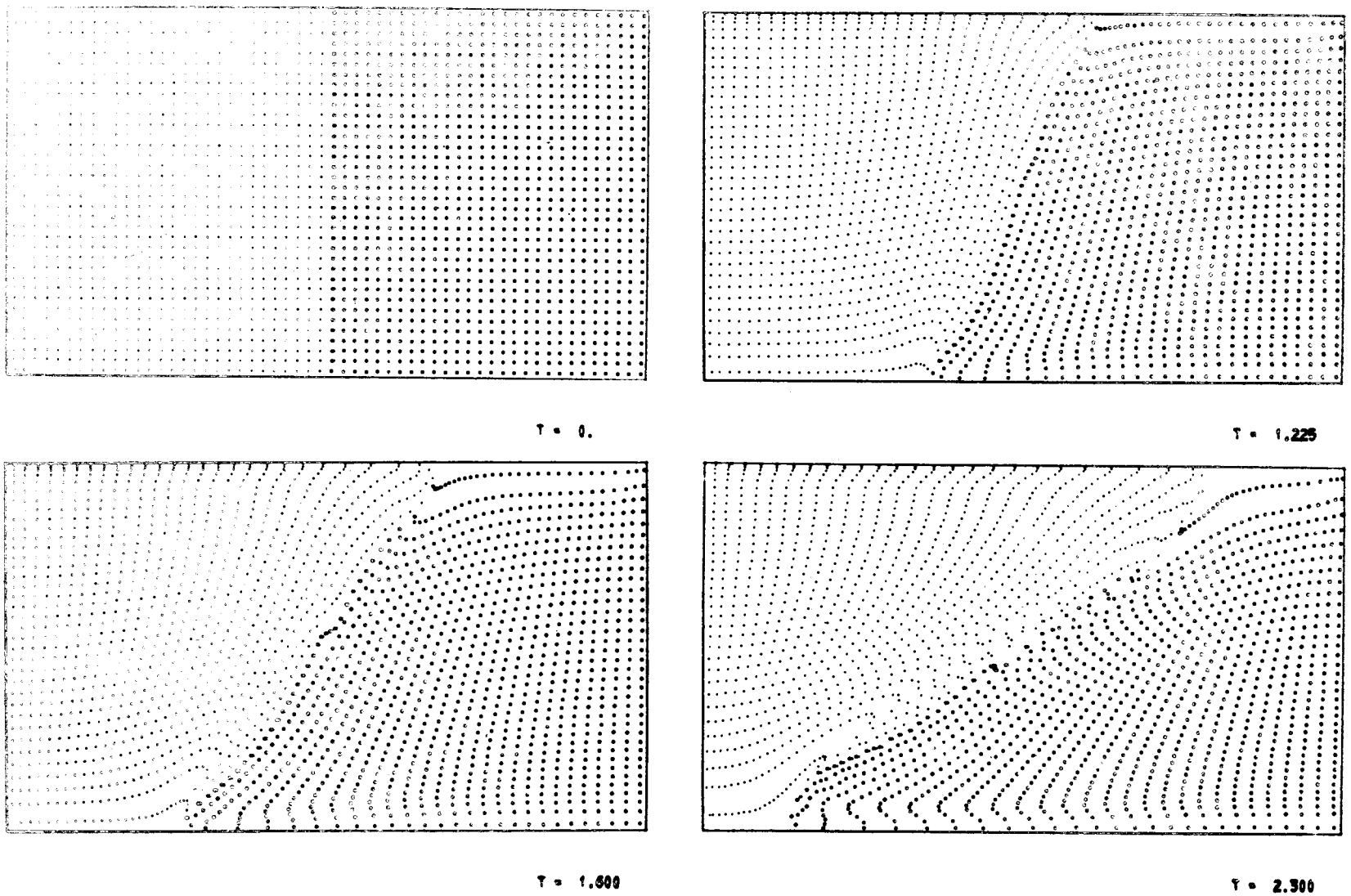


Figure 13. Mixing of two fluids.

VI. CONCLUSIONS

This paper has presented a method for finding transient solutions to the complete two-dimensional incompressible Navier-Stokes equations of motion for fluids. This MAC Method description was intended to be brief; the interested reader may seek details by consulting the excellent report by the Los Alamos research group (Welch, 1966).

In addition, a special case of the theory of overrelaxation methods has been developed. The development was not rigorous; however care was taken in proving that this convergence-speeding technique could be applied to the MAC Method and produce a more efficient algorithm. Results of test cases were introduced to validate the conclusions reached in theory.

Finally, examples of the capability of the MAC Method were presented, showing how it could be used for the simulation of a great variety of incompressible fluid flow problems.

In conclusion, the author would like to point out some problem areas of the MAC Method and suggest paths for future research.

One deficiency of this method is the absence of proofs of existence and uniqueness of solutions. Of course, to do this one must first make the same analysis on the Navier-Stokes equations. Very little work has been carried out in this area.

Secondly, an error analysis of this algorithm, including truncation and round-off errors, would substantiate its value. Hwang (1968) has presented such an analysis for his extended version of the two-fluid MAC Method. An investigation should also be made into why the theoretical outflow boundary conditions do not agree with experimental results. Perhaps an order of magnitude analysis would help here.

Finally, future applications for this method might include: the effect of atmospheric conditions on ocean waves and currents; a study of density currents, both in air and water; and the effect of density stratification on water quality. The latter subject could also be extended to the problem of selective withdrawal from a reservoir; papers by Hwang (1968) and Mercier (1968) might be used as a basis for such an investigation. Particularly needed in this last area is a method of scaling the problems. Mercier has shown that scaling difficulties exist because of the large difference in the horizontal and vertical dimensions of reservoirs.

BIBLIOGRAPHY

Davis, Joel. 1967. Assistant Professor, Oregon State University, Dept. of Mathematics. Lecture in Numerical Calculus. Corvallis. May 5.

Forsythe, George E. and Wolfgang R. Wasow. 1960. Finite-difference methods for partial differential equations. New York, Wiley. 444 p.

Harlow, Francis H. 1967. Los Alamos Scientific Laboratory. Personal communication. Los Alamos, New Mexico. Aug. 14.

Hwang, John Dzen. 1968. On numerical solutions of the general Navier-Stokes equations for two-layered stratified flows. Ph.D. thesis. Corvallis, Oregon State University. 134 numb. leaves.

Mercier, Howard Thomas. 1968. A predictor-corrector method for the transient motion of a nonhomogeneous, incompressible, viscous fluid. Master's thesis. Corvallis, Oregon State University. 56 numb. leaves.

Schlichting, Hermann. 1960. Boundary layer theory, tr. by J. Kestin. 4th ed. New York, McGraw-Hill. 647 p.

Shannon, John P. 1967. Los Alamos Scientific Laboratory. Personal communication. Los Alamos, New Mexico. Aug. 14.

Shapiro, Ascher H. 1953. The dynamics and thermodynamics of compressible fluid flow. Vol. 1. New York, Ronald. 647 p.

Slotta, Larry S., John D. Hwang, Michael D. Terry and Howard T. Mercier. 1968. Digital simulation of nonhomogeneous fluid flow. Paper presented at Northwest Simulation Council Meeting, Corvallis, Oregon State University, Jan. 12.

Welch, J. Eddie, Francis H. Harlow, John P. Shannon and Bart J. Daly. 1966. The MAC Method. 146 p. (U.S. Atomic Energy Commission. LA-3425)

Young, David. 1954. Iterative methods for solving partial difference equations of elliptic type. Transactions of the American Mathematical Society 76:92-111.

APPENDICES

Appendix I. Nomenclature

The following table contains a list of the symbols used in this paper. The symbols are listed in alphabetical order and are identified with the chapter(s) in which they appear.

Symbol	Meaning	Chapter
a, b, c, d, e, f, g	Coefficients of the general second-order partial differential equation	III
$\vec{A}, \vec{B}$	Arbitrary vectors	II
C	Wave speed of a fluid	II
D	Finite difference form of $\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y}$	II
f	Arbitrary function in an elliptic partial differential equation	III, IV
f_1, f_2	Arbitrary functions defining u^0, v^0, respectively	II
$\vec{g}$	Acceleration due to gravity	II
g_x, g_y	Components of $\vec{g}$ in x and y directions, respectively	II, V
g_1, g_2, g_3	Arbitrary functions in elliptic partial differential equation boundary condition	III, IV
H	Height of rectangle	III
i, j	Subscripts denoting position in discrete mesh	II, III, IV
$\vec{i}, \vec{j}$	Unit vectors in x and y directions, respectively	II
k	Superscript denoting k^{th} iteration	III
n	Superscript denoting n^{th} time cycle	II

Symbol	Meaning	Chapter
q	Overrelaxation factor	III, IV
q_{opt}	Optimum overrelaxation factor	III, IV
r	Rate of convergence of simple relaxation	III
R	A region in the x, y plane	III, IV
$R_0^{(k)}$	k^{th} residual at w_0	III
S	Source term in ϕ equation	II, IV
S'	Partial differential equation analogy of S	IV
t	Time	II
u, v	Components of velocity in x and y directions, respectively	II
u^0, v^0	Initial velocities in x and y directions, respectively	II, V
$\vec{V}$	Velocity vector	II
w	Function satisfying an elliptic partial differential equation	III, IV
W	Width of rectangle	III
x, y	Two-dimensional Cartesian space coordinates	II, III, IV
x', y'	Coordinates of a point on the boundary of R	III
δt	Time increment	II, V
$\delta x, \delta y$	Increments in x and y directions, respectively	II, III, IV, V
ϵ	Arbitrary small number	III
ν	Kinematic viscosity	II, V

Symbol	Meaning	Chapter
ξ, ξ'	Arbitrary functions in ϕ boundary conditions	IV
π	Pi, area of the unit circle	III
ϕ	Ratio of pressure to density	II, IV
$\frac{\partial}{\partial n}$	Partial derivative in normal direction	III, IV
$\frac{\partial}{\partial x}, \frac{\partial}{\partial y}$	Partial derivatives in x and y directions, respectively	II, III, IV
$\frac{\partial}{\partial t}$	Partial derivative with respect to time	II
∇	Del, divergence operator, $= \frac{\partial}{\partial x}\vec{i} + \frac{\partial}{\partial y}\vec{j}$	II
$\|R^{(k)}\|$	Norm of matrix $R^{(k)}$	III

Appendix II. SPLIT Listing

```
      PROGRAM SPLIT
      COMMON/TVPOOL/XMIN,XMAX,YMIN,YMAX,TXMIN,TXMAX,TYMIN,TYMAX
      COMMON/TVGUIDE/TMODE,TEXT,ITV
      COMMON/TVFACT/FACT
      COMMON/TVTUNE/LPEN,LPEF,ITAL,IWINK,INTS,IRT,IUP
      COMMON QQ,LQQ,TNUX4,TNUY4
      COMMON DTVP,DELTAS,TVP
      COMMON MSK7,MSK8,MKC7,MKC8,IMP7,IMP8,DV7,DV8
      COMMON AGXL,AGYH,DT,DTCP,DTP,DTPP,DXC,DXCD2,DXCSQ,DYC,DYCD2,DYCSQ,
     $  DXIN,GX,GY,H,NI,NIM1,NJ,NJM1,NP,NTP,NU,T,TCP,TL,TP,TPP,W,Z,DYIN
      COMMON KD,DXY,DYX,NIP1,NJP1,TDXY,GXD,GYD,TNUX,TNUY,DXDY2,DYDX2,
     $       DXY4,DYX4,ZR,ZDXR,ZDYR,GXT,GYT,DX4,DY4,MI,MO,UP,VP,XP,YP
      COMMON CS,D,P,PS,R,U,V,XC,YC
      COMMON MSK(6),MKC(6),IMP(6),DV(6)
      DIMENSION CS(25,100),D(25,100),P(25,100),PS(3000),R(25,100),
     $          U(26,101),V(26,101),XC(25),YC(100)
      DIMENSION UP(3000),VP(3000),XP(3000),YP(3000)
      INTEGER CS,FLAG,PS
      REAL NU
      COMMON KNBP,NBP2,XB2(18),YB2(18)
      COMMON NBP,XB(18),YB(18)
C
C
C     SPLIT SOLVES THE NAVIER-STOKES EQUATIONS FOR TWO-DIMENSIONAL, IN-
C     COMPRESSIBLE FLOW USING THE MAC (MARKER AND CELL) METHOD, AS DE-
C     SCRIBED IN ..THE MAC METHOD... A REPORT WRITTEN FOR LOS ALAMOS
C     SCIENTIFIC LABORATORY BY J. EDDIE WELCH, FRANCIS H. HARLOW, JOHN P.
C     SHANNON, AND BART J. DALY.
C
C     THIS PROGRAM WAS WRITTEN IN MAY OF 1967 BY MICHAEL D. TERRY IN
C     CONNECTION WITH A RESEARCH PROJECT, ENTITLED ..STRATIFIED RESERVOIR
C     CURRENTS... AT OREGON STATE UNIVERSITY.
C
C
      MI=5
      MO=6
      KD=1
      CALL OV1
  100 CALL OV2
      GO TO (101,102),KD
  101 CALL OV3
      GO TO 100
  102 CALL TVEND
      STOP
      END


      SUBROUTINE OV1
      COMMON/TVPOOL/XMIN,XMAX,YMIN,YMAX,TXMIN,TXMAX,TYMIN,TYMAX
      COMMON/TVGUIDE/TMODE,TEXT,ITV
      COMMON/TVFACT/FACT
      COMMON/TVTUNE/LPEN,LPEF,ITAL,IWINK,INTS,IRT,IUP
      COMMON QQ,LQQ,TNUX4,TNUY4
      COMMON DTVP,DELTAS,TVP
      COMMON MSK7,MSK8,MKC7,MKC8,IMP7,IMP8,DV7,DV8
      COMMON AGXL,AGYH,DT,DTCP,DTP,DTPP,DXC,DXCD2,DXCSQ,DYC,DYCD2,DYCSQ,
     $  DXIN,GX,GY,H,NI,NIM1,NJ,NJM1,NP,NTP,NU,T,TCP,TL,TP,TPP,W,Z,DYIN
      COMMON KD,DXY,DYX,NIP1,NJP1,TDXY,GXD,GYD,TNUX,TNUY,DXDY2,DYDX2,
     $       DXY4,DYX4,ZR,ZDXR,ZDYR,GXT,GYT,DX4,DY4,MI,MO,UP,VP,XP,YP
      COMMON CS,D,P,PS,R,U,V,XC,YC
      COMMON MSK(6),MKC(6),IMP(6),DV(6)
      DIMENSION CS(25,100),D(25,100),P(25,100),PS(3000),R(25,100),
     $          U(26,101),V(26,101),XC(25),YC(100)
      DIMENSION UP(3000),VP(3000),XP(3000),YP(3000)
      INTEGER CS,FLAG,PS
      REAL NU
      DIMENSION TYPE(16),XB(18),YB(18),IA(500)
      DIMENSION XPO(20),YPO(20),XPL(20),YPL(20),UPO(20),VPO(20)
      INTEGER TYPE
      COMMON KNBP,NBP2,XB2(18),YB2(18)
      COMMON NBP,XB,YB
      READ (MI,1001) NI,NJ,NB1,NB2,NPR,NP,LQQ
      READ (MI,1000) U0,V0,GX,GY,DTP,DTPP,DTCP
      READ (MI,1000) W,H,NU,TL,DXP,DYP,DT
      READ (MI,1000) DTVP,DELTAS,QQ
      READ (MI,1010) (XPO(I),YPO(I),XPL(I),YPL(I),UPO(I),VPO(I),I=1,NPR)
      NIM1=NI-1
      NIP1=NI+1
      NJM1=NJ-1
      NJP1=NJ+1
      ANI=NI
      ANJ=NJ
      DXC=W/(ANI-2.)
      DYC=H/(ANJ-2.)
      GXT=GX*DT
      GYT=GY*DT
      DXY=DXC/DYC
      DYX=DYC/DXC
      DXY4=DXY*.25
      DYX4=DYX*.25
      DX4=.25/DXC
      DY4=.25/DYC
      TDXY=.5/(DXC*DYC)
      GXD=GX*DXC
      GYD=GY*DYC
      TNU=2.*NU
      TNUX=TNU/DXC
      TNUY=TNU/DYC
      TNUX4=TNUX*.25
      TNUY4=TNUY*.25
      DXCD2=.5*DXC
      DYCD2=.5*DYC
      DXCSQ=DXC*DXC
      DYCSQ=DYC*DYC
      Z=(1./DXCSQ+1./DYCSQ)*2.
      DYDX2=DYC*NU/DXCSQ
      DXDY2=DXC*NU/DYCSQ
      ZR=1./Z
      ZDXR=ZR/DXCSQ
      ZDYR=ZR/DYCSQ
      AGXL=ABS(GX*W)
      AGYH=ABS(GY*H)
      MSK(1)=7B
      MSK(2)=70B
      MSK(3)=700B
      MSK(4)=7000B
      MSK(5)=70000B
      MSK(6)=700000B
      MSK7=7000000B
      MSK8=70000000B
      DO 900 I=1,6
  900 MKC(I)=.NOT.MSK(I)
```

```
      MKC7=.NOT.MSK7
      MKC8=.NOT.MSK8
      IMP(2)=10B
      IMP(3)=100B
      IMP(4)=1000B
      IMP(5)=10000B
      IMP(6)=100000B
      IMP7=1000000B
      IMP8=10000000B
      DV(2)=1./8.
      DV(3)=1./64.
      DV(4)=1./512.
      DV(5)=1./4096.
      DV(6)=1./32768.
      DV7=1./262144.
      DV8=1./2097152.
      T=0.
      TT=0.
      DO 100 J=1,NJ
      DO 100 I=1,NI
      CS(I,J)=0
      P(I,J)=0.
      R(I,J)=0.
  100 D(I,J)=0.
      DO 101 J=1,NJP1
      DO 101 I=1,NIP1
      U(I,J)=0.
  101 V(I,J)=0.
      DO 2 K=1,NP
      PS(K)=0
      XP(K)=0.
      YP(K)=0.
      UP(K)=0.
    2 VP(K)=0.
      XC(1)=-DXCD2
      YC(1)=-DYCD2
      DO 102 I=2,NI
  102 XC(I)=XC(I-1)+DXC
      DO 103 J=2,NJ
  103 YC(J)=YC(J-1)+DYC
      DO 105 J=1,NJ
      DO 105 I=1,NI
      DO 104 K=3,6
  104 CS(I,J)=IMP(K).OR.(CS(I,J).AND.MKC(K))
      CS(I,J)=6.OR.(CS(I,J).AND.MKC)
      KT=5*IMP(2)
  105 CS(I,J)=KT.OR.(CS(I,J).AND.MKC(2))
      NB=NB1
      KNBP=1
      PLMAX=AMAX1(XC(NI)+DXC,YC(NJ)+DYC)
      PLMIN=AMIN1(XC(1)-DXC,YC(1)-DYC)
      XMIN=PLMIN
      YMIN=PLMIN
      XMAX=PLMAX
      YMAX=PLMAX
  106 NBP=NB+1
      READ (MI,1000) (XB(I),YB(I),I=1,NBP)
      READ (MI,1001) (TYPE(I),I=1,NB)
      XB(NBP+1)=XB(2)
      YB(NBP+1)=YB(2)
      DO 142 M=1,NB
      IF (XB(M)-XB(M+1)) 107,121,114
  107 I=XB(M)/DXC+2.001
      J=YB(M)/DYC+1.999
  108 CS(I,J)=2.OR.(CS(I,J).AND.MKC)
      KT=TYPE(M)*IMP(2)
      CS(I,J)=KT.OR.(CS(I,J).AND.MKC(2))
      KT=2*IMP(4)
      CS(I,J+1)=KT.OR.(CS(I,J+1).AND.MKC(4))
      KT=3*IMP7
      CS(I,J)=KT.OR.(CS(I,J).AND.MKC7)
      KT=3*IMP8
      CS(I,J+1)=KT.OR.(CS(I,J+1).AND.MKC8)
      K2=CS(I,J).AND.MSK(2)
      K2=K2*DV(2)
      IF (K2.EQ.2) GO TO 110
  109 KT=2*IMP(6)
      CS(I,J)=KT.OR.(CS(I,J).AND.MKC(6))
  110 I=I+1
      IF (XC(I) .LE. XB(M+1)) 108,111
  111 IF (YB(M+2)-YB(M+1)) 112,142,113
  112 KT=2*IMP(5)
      CS(I-1,J)=KT.OR.(CS(I-1,J).AND.MKC(5))
      GO TO 142
  113 CS(I,J)=1.OR.(CS(I,J).AND.MKC)
      GO TO 137
  114 I=XB(M)/DXC+1.999
      J=YB(M)/DYC+2.001
  115 CS(I,J)=2.OR.(CS(I,J).AND.MKC)
      KT=TYPE(M)*IMP(2)
      CS(I,J)=KT.OR.(CS(I,J).AND.MKC(2))
      KT=2*IMP(4)
      CS(I,J-1)=KT.OR.(CS(I,J-1).AND.MKC(4))
      KT=4*IMP7
      CS(I,J)=KT.OR.(CS(I,J).AND.MKC7)
      KT=4*IMP8
      CS(I,J-1)=KT.OR.(CS(I,J-1).AND.MKC8)
      K2=CS(I,J).AND.MSK(2)
      K2=K2*DV(2)
      IF (K2.EQ.2) GO TO 117
  116 KT=2*IMP(6)
      CS(I,J-1)=KT.OR.(CS(I,J-1).AND.MKC(6))
  117 I=I-1
      IF (XC(I) .GE. XB(M+1)) 115,118
  118 IF (YB(M+2)-YB(M+1)) 120,142,119
  119 KT=2*IMP(5)
      CS(I+1,J)=KT.OR.(CS(I+1,J).AND.MKC(5))
      KT=2*IMP(6)
      CS(I,J-1)=KT.OR.(CS(I,J-1).AND.MKC(6))
      GO TO 142
  120 CS(I,J)=1.OR.(CS(I,J).AND.MKC)
      GO TO 137
  121 IF (YB(M) .LT. YB(M+1)) 122,129
  122 I=XB(M)/DXC+2.001
      J=YB(M)/DYC+2.001
  123 CS(I,J)=2.OR.(CS(I,J).AND.MKC)
      KT=TYPE(M)*IMP(2)
      CS(I,J)=KT.OR.(CS(I,J).AND.MKC(2))
      KT=2*IMP(4)
      CS(I-1,J)=KT.OR.(CS(I-1,J).AND.MKC(4))
      KT=2*IMP7
      CS(I,J)=KT.OR.(CS(I,J).AND.MKC7)
```

```
      KT=2*IMP8
      CS(I-1,J)=KT.OR.(CS(I-1,J).AND.MKC8)
      K2=CS(I,J).AND.MSK(2)
      K2=K2*DV(2)
      IF (K2.EQ.2) GO TO 125
  124 KT=2*IMP(6)
      CS(I-1,J)=KT.OR.(CS(I-1,J).AND.MKC(6))
  125 J=J+1
      IF (YC(J) .LE. YB(M+1)) 123,126
  126 IF (XB(M+2)-XB(M+1)) 128,142,127
  127 KT=2*IMP(5)
      CS(I,J-1)=KT.OR.(CS(I,J-1).AND.MKC(5))
      GO TO 142
  128 CS(I,J)=1.OR.(CS(I,J).AND.MKC)
      GO TO 137
  129 I=XB(M)/DXC+1.999
      J=YB(M)/DYC+1.999
  130 CS(I,J)=2.OR.(CS(I,J).AND.MKC)
      KT=TYPE(M)*IMP(2)
      CS(I,J)=KT.OR.(CS(I,J).AND.MKC(2))
      KT=2*IMP(4)
      CS(I+1,J)=KT.OR.(CS(I+1,J).AND.MKC(4))
      CS(I,J)=IMP7.OR.(CS(I,J).AND.MKC7)
      CS(I+1,J)=IMP8.OR.(CS(I+1,J).AND.MKC8)
      K2=CS(I,J).AND.MSK(2)
      K2=K2*DV(2)
      IF (K2.EQ.2) GO TO 132
  131 KT=2*IMP(6)
      CS(I,J)=KT.OR.(CS(I,J).AND.MKC(6))
  132 J=J-1
      IF (YC(J) .GE. YB(M+1)) 130,133
  133 IF (M .LT. NB) 134,142
  134 IF (XB(M+2)-XB(M+1)) 135,142,136
  135 KT=2*IMP(5)
      CS(I,J+1)=KT.OR.(CS(I,J+1).AND.MKC(5))
      GO TO 142
  136 CS(I,J)=1.OR.(CS(I,J).AND.MKC)
      KT=2*IMP(6)
      CS(I,J)=KT.OR.(CS(I,J).AND.MKC(6))
  137 IF (M .LT. NB) 138,139
  138 KMP1=M+1
      GO TO 140
  139 KMP1=1
  140 IF (TYPE(M) .EQ. 3 .OR. TYPE(KMP1) .EQ. 3) 141,142
  141 KT=3*IMP(2)
      CS(I,J)=KT.OR.(CS(I,J).AND.MKC(2))
  142 CONTINUE
      IF (NB2 .NE. 0) 143,144
  143 NB=NB2
      NB2=0
      KNBP=2
      NBP2=NBP
      DO 1 I=1,NBP2
      XB2(I)=XB(I)
    1 YB2(I)=YB(I)
      GO TO 106
  144 KODE=1
      DO 170 J=1,NJ
      DO 170 I=1,NI
      K1=CS(I,J).AND.MSK
      K2=CS(I,J).AND.MSK(2)
      K2=K2*DV(2)
      K4=CS(I,J).AND.MSK(4)
      K4=K4*DV(4)
      IF (K2.NE.1) GO TO 154
  145 IF (I .EQ. 1) 146,149
  146 L=1
  147 LL=1
      K21=CS(I+1,J-1).AND.MSK(2)
      K21=K21*DV(2)
      IF (K21.EQ.4) GO TO 161
  148 K21=CS(I+1,J+1).AND.MSK(2)
      K21=K21*DV(2)
      GO TO 159
  149 IF (I .EQ. NI) 150,153
  150 L=1
  151 LL=2
      K21=CS(I-1,J-1).AND.MSK(2)
      K21=K21*DV(2)
      IF (K21.EQ.4) GO TO 161
  152 K21=CS(I-1,J+1).AND.MSK(2)
      K21=K21*DV(2)
      GO TO 159
  153 IF (J .EQ. 1) 154,156
  154 L=1
      K21=CS(I-1,J+1).AND.MSK(2)
      K21=K21*DV(2)
      IF (K21.EQ.4) GO TO 161
  155 K21=CS(I+1,J+1).AND.MSK(2)
      K21=K21*DV(2)
      GO TO 159
  156 IF (J .EQ. NJ) 157,160
  157 L=1
      K21=CS(I-1,J-1).AND.MSK(2)
      K21=K21*DV(2)
      IF (K21.EQ.4) GO TO 161
  158 K21=CS(I+1,J-1).AND.MSK(2)
      K21=K21*DV(2)
  159 IF (K21.EQ.4) GO TO 161
      GO TO 162
  160 L=2
      GO TO 147
  161 KT=2*IMP(3)
      CS(I,J)=KT.OR.(CS(I,J).AND.MKC(3))
  162 GO TO (164,163),L
  163 GO TO (151,164),LL
  164 GO TO (165,168),KODE
  165 GO TO (166,168),K4
  166 KODE=1
      IF (K1.EQ.2) GO TO 170
  167 CS(I,J)=1.OR.(CS(I,J).AND.MKC)
      GO TO 170
  168 KODE=1
      IF (K1.EQ.2) GO TO 170
  169 KODE=2
      CS(I,J)=3.OR.(CS(I,J).AND.MKC)
  170 CONTINUE
      DO 902 J=1,NJ
      DO 902 I=1,NI
      K5=CS(I,J).AND.MSK(5)
      K5=K5*DV(5)
      IF (K5.EQ.1) GO TO 902
```

```
      K41=CS(I-1,J).AND.MSK(4)
      K41=K41*DV(4)
      IF (K41.EQ.2) GO TO 901
      CS(I,J)=IMP7.OR.(CS(I,J).AND.MKC7)
      GO TO 902
  901 KT=2*IMP7
      CS(I,J)=KT.OR.(CS(I,J).AND.MKC7)
  902 CONTINUE
      KT=2*IMP(6)
      CS=KT.OR.(CS.AND.MKC(6))
      DO 173 K=1,NP
  173 PS(K)=3.OR.(PS(K).AND.MKC)
      K=1
      DO 191 II=1,NPR
      Y=YP0(II)
  174 X=XP0(II)
  175 I=X/DXC+2.
      J=Y/DYC+2.
      K1=CS(I,J).AND.MSK
      K2=CS(I,J).AND.MSK(2)
      K2=K2*DV(2)
      IF (K1.EQ.1) GO TO 178
      IF (K1.EQ.2) GO TO 180
  176 CS(I,J)=4.OR.(CS(I,J).AND.MKC)
      PS(K)=1.OR.(PS(K).AND.MKC)
      U(I,J)=UP0(II)
      U(I+1,J)=UP0(II)
      V(I,J)=VP0(II)
      V(I,J+1)=VP0(II)
  177 XP(K)=X
      YP(K)=Y
      K=K+1
  178 X=X+DXP
      IF (X .GT. XPL(II)) 179,175
  179 Y=Y+DYP
      IF (Y .GT. YPL(II)) 191,174
  180 IF (K2.NE.1) GO TO 178
  181 PS(K)=2.OR.(PS(K).AND.MKC)
      K7=CS(I,J).AND.MSK7
      K7=K7*DV7
      IF (K7.EQ.1) GO TO 183
      IF (K7.EQ.2) GO TO 186
      IF (K7.EQ.3) GO TO 189
      GO TO 189
  183 U(I+1,J)=U0
      GO TO 177
  186 U(I,J)=U0
      GO TO 177
  189 V(I,J)=V0
      GO TO 177
  190 V(I,J+1)=V0
      GO TO 177
  191 CONTINUE
      CALL BNDVEL
      WRITE (MO,1002)
      WRITE (MO,1003) NI,NJ,NB1,NB2,NPR,LQ0
      WRITE (MO,1006)
      WRITE (MO,1005) U0,V0,GX,GY,DTP,DTPP,DTCP
      WRITE (MO,1007)
      WRITE (MO,1005) W,H,NU,TL,DXP,DYP,DT
      WRITE (MO,1011)
      WRITE (MO,1005) DTVP,DELTAS,Q0
      DELTAS=DELTAS*.5
      WRITE (MO,1004)
      DO 192 I=1,NPR
  192 WRITE (MO,1005) XP0(I),YP0(I),XPL(I),YPL(I)
      WRITE (MO,1008)
      WRITE (MO,1005) DXC,DYC
      WRITE (MO,1009)
      WRITE (MO,1003) NP
      WRITE (MO,1013)
      WRITE (MO,1005) PLMIN,PLMAX
      RETURN
 1000 FORMAT(7F10.0)
 1001 FORMAT(7I10)
 1002 FORMAT(1H1,10X52HMAC METHOD SOLUTION OF TWO-DIMENSIONAL FLUID PROB
     $LEM///1H-,10X5HINPUT/1H-,23X2HNI,13X2HNJ,12X3HNB1,12X3HNB2,12X3HNP
     $R,12X3HLQ0)
 1003 FORMAT(1H ,10X,7I15)
 1004 FORMAT(1H0,22X3HXP0,12X3HYP0,12X3HXPL,12X3HYPL)
 1005 FORMAT(1H ,10X,7E15.8)
 1006 FORMAT(1H0,23X2HU0,13X2HV0,13X2HGX,13X2HGY,12X3HDTP,11X4HDTPP,11X
     $4HDTCP)
 1007 FORMAT(1H0,24X1HW,14X1HH,13X2HNU,13X2HTL,12X3HDXP,12X3HDYP,13X2HDT
     $)
 1008 FORMAT(1H-/1H ,10X17HCALCULATED VALUES/1H-,22X3HDXC,12X3HDYC)
 1009 FORMAT(1H0,23X2HNP)
 1010 FORMAT(6F10.0)
 1011 FORMAT(1H0,21X4HDTVP,9X6HDELTAS,13X2HQ0)
 1012 FORMAT(2F10.0)
 1013 FORMAT(1H0,20X5HPLMIN,10X5HPLMAX)
      END


      SUBROUTINE BNDVEL
      COMMON/TVPOOL/XMIN,XMAX,YMIN,YMAX,TXMIN,TXMAX,TYMIN,TYMAX
      COMMON/TVGUIDE/TMODE,TEXT,ITV
      COMMON/TVFACT/FACT
      COMMON/TVTUNE/LPEN,LPEF,ITAL,IWINK,INTS,IRT,IUP
      COMMON Q0,LQ0,TNUX4,TNUY4
      COMMON DTVP,DELTAS,TVP
      COMMON MSK7,MSK8,MKC7,MKC8,IMP7,IMP8,DV7,DV8
      COMMON AGXL,AGYH,DT,DTCP,DTP,DTPP,DXC,DXCD2,DXCSQ,DYC,DYCD2,DYCSQ,
     $  DXIN,GX,GY,H,NI,NIM1,NJ,NJM1,NP,NTP,NU,T,TCP,TL,TP,TPP,W,Z,DYIN
      COMMON KD,DXY,DYX,NIP1,NJP1,TDXY,GXD,GYD,TNUX,TNUY,DXDY2,DYDX2,
     $       DXY4,DYX4,ZR,ZDXR,ZDYR,GXT,GYT,DX4,DY4,MI,MO,UP,VP,XP,YP
      COMMON CS,D,P,PS,R,U,V,XC,YC
      COMMON MSK(6),MKC(6),IMP(6),DV(6)
      DIMENSION CS(25,100),D(25,100),P(25,100),PS(3000),R(25,100),
     $          U(26,101),V(26,101),XC(25),YC(100)
      DIMENSION UP(3000),VP(3000),XP(3000),YP(3000)
      INTEGER CS,FLAG,PS
      REAL NU
      DO 126 J=1,NJ
      JM=J-1
      JP=J+1
      DO 126 I=1,NI
      IM=I-1
      IP=I+1
      TK1=1.
      TK2=1.
      K1=CS(I,J).AND.MSK
```

```
      IF (K1.NE.2) GO TO 126
  100 K5=CS(I,J).AND.MSK(5)
      K5=K5*DV(5)
      K7=CS(I,J).AND.MSK7
      K7=K7*DV7
      IF (K7.EQ.1) GO TO 117
      IF (K7.EQ.2) GO TO 118
      IF (K7.EQ.3) GO TO 108
      GO TO 109
  107 K43=CS(I,JM).AND.MSK(4)
      K43=K43*DV(4)
      IF (K43.EQ.2) GO TO 109
  108 J1=JP
      J2=JP
      GO TO 110
  109 J1=J
      J2=JM
  110 K2=CS(I,J).AND.MSK(2)
      K2=K2*DV(2)
      GO TO (111,112,113,115,126),K2
  111 S=1.
      GO TO 116
  112 S=1.
      GO TO 116
  113 S=1.
  114 V(I,J1)=0.
      GO TO 116
  115 S=-1.
      GO TO 114
  116 U(IP,J)=S*U(IP,J2)*TK1
      U(I,J)=S*U(I,J2)*TK2
      GO TO 126
  117 I1=IP
      I2=IP
      IF (K5.EQ.2) TK1=0.
      GO TO 119
  118 I1=I
      I2=IM
      IF (K5.EQ.2) TK2=0.
  119 K2=CS(I,J).AND.MSK(2)
      K2=K2*DV(2)
      GO TO (120,121,122,124,126),K2
  120 S=-1.
      GO TO 125
  121 S=1.
      GO TO 125
  122 S=1.
  123 U(I1,J)=0.
      GO TO 125
  124 S=-1.
      GO TO 123
  125 V(I,JP)=S*V(I2,JP)
      V(I,J)=S*V(I2,J)
      IF (K5.EQ.2) GO TO 107
  126 CONTINUE
      RETURN
      END


      SUBROUTINE OV2
      COMMON/TVPOOL/XMIN,XMAX,YMIN,YMAX,TXMIN,TXMAX,TYMIN,TYMAX
      COMMON/TVGUIDE/TMODE,TEXT,ITV
      COMMON/TVFACT/FACT
      COMMON/TVTUNE/LPEN,LPEF,ITAL,IWINK,INTS,IRT,IUP
      COMMON QQ,LQQ,TNUX4,TNUY4
      COMMON DTVP,DELTAS,TVP
      COMMON MSK7,MSK8,MKC7,MKC8,IMP7,IMP8,DV7,DV8
      COMMON AGXL,AGYH,DT,DTCP,DTP,DTPP,DXC,DXCQ2,DXCSQ,DYC,DYCQ2,DYCSQ,
     $  DXIN,GX,GY,H,NI,NIM1,NJ,NJM1,NP,NTP,NU,T,TCP,TL,TP,TPP,W,Z,DYIN
      COMMON KD,DXY,DYX,NIP1,NJP1,TDXY,GXD,GYD,TNUX,TNUY,DXDY2,DYDX2,
     $       DXY4,DYX4,ZR,ZDXR,ZDYR,GXT,GYT,DX4,DY4,MI,MO,UP,VP,XP,YP
      COMMON CS,D,P,PS,R,U,V,XC,YC
      COMMON MSK(6),MKC(6),IMP(6),DV(6)
      DIMENSION CS(25,100),D(25,100),P(25,100),PS(3000),R(25,100),
     $          U(26,101),V(26,101),XC(25),YC(100)
      DIMENSION UP(3000),VP(3000),XP(3000),YP(3000)
      INTEGER CS,FLAG,PS
      REAL NU
      IF (KD.EQ.2) GO TO 200
   99 IF (T .EQ. 0.) 100,101
  100 CALL SG1
      GO TO 202
  200 CALL SG2
      CALL SG3
      IF (KD.EQ.1) GO TO 202
  201 WRITE (MO,1000)
      GO TO 110
  202 TP=DTP
      TPP=DTPP
      TCP=DTCP
      TVP=DTVP
      GO TO 109
  101 IF (T .GE. TP) 102,103
  102 CALL SG1
      TP=T+DTP
  103 IF (T .GE. TCP) 104,105
  104 CALL SG2
      TCP=T+DTCP
  105 IF (T.GE.TPP) 106,400
  106 CALL SG3
      TPP=T+DTPP
  400 IF (T.GE.TVP) 401,107
  401 CALL PLOTV
      TVP=T+DTVP
  107 IF (T .GE. TL) 108,109
  108 KD=2
      GO TO 110
  109 T1=T
      T=T+DT
      CALL SECOND(X)
      WRITE (MO,1001) X,T1,T
  110 RETURN
 1000 FORMAT(1H1,41HABNORMAL STOP -- LOOK FOR ANOTHER MESSAGE)
 1001 FORMAT(1H ,10X,E10.3,17H SECONDS BETWEEN ,F6.3,5H AND ,F6.3)
      END


      SUBROUTINE SG1
      COMMON/TVPOOL/XMIN,XMAX,YMIN,YMAX,TXMIN,TXMAX,TYMIN,TYMAX
      COMMON/TVGUIDE/TMODE,TEXT,ITV
      COMMON/TVFACT/FACT
      COMMON/TVTUNE/LPEN,LPEF,ITAL,IWINK,INTS,IRT,IUP
```

```
      COMMON QO,LQO,TNUX4,TNUY4
      COMMON DTVP,DELTAS,TVP
      COMMON MSK7,MSK8,MKC7,MKC8,IMP7,IMP8,DV7,DV8
      COMMON AGXL,AGYH,DT,DTCP,DTP,DTPP,DXC,DXCD2,DXCSQ,DYC,DYCD2,DYCSQ,
     $  DXIN,GX,GY,H,NI,NIM1,NJ,NJM1,NP,NTP,NU,T,TCP,TL,TP,TPP,W,Z,DYIN
      COMMON KD,DXY,DYX,NIP1,NJP1,TDXY,GXD,GYD,TNUX,TNUY,DXDY2,DYDX2,
     $       DXY4,DYX4,ZR,ZDXR,ZDYR,GXT,GYT,DX4,DY4,MI,MO,UP,VP,XP,YP
      COMMON CS,D,P,PS,R,U,V,XC,YC
      COMMON MSK(6),MKC(6),IMP(6),DV(6)
      DIMENSION CS(25,100),D(25,100),P(25,100),PS(3000),R(25,100),
     $          U(26,101),V(26,101),XC(25),YC(100)
      DIMENSION UP(3000),VP(3000),XP(3000),YP(3000)
      INTEGER CS,FLAG,PS
      REAL NU
      DIMENSION XB2(18),YB2(18),XP1(3000),YP1(3000)
      COMMON KNBP,NBP2,XB2,YB2
      COMMON NBP,XB(18),YB(18)
      TEXT=0.
      CALL TVPLOT(XB,YB,NBP)
      IF (KNBP.EQ.2) CALL TVPLOT(XB2,YB2,NBP2)
      WRITE (98,1000) T
      CALL TVLTR(824.,72.,0,3)
      TEXT=1.
      KOUNT=0
      DO 102 K=1,NP
      KP=PS(K).AND.MSK
      IF (KP.EQ.3) GO TO 102
  101 I=XP(K)/DXC+2.
      J=YP(K)/DYC+2.
      K1=CS(I,J).AND.MSK
      IF, (K1.EQ.2) GO TO 102
      IF (K1.EQ.1) GO TO 102
      KOUNT=KOUNT+1
      XP1(KOUNT)=XP(K)
      YP1(KOUNT)=YP(K)
  102 CONTINUE
      CALL TVPLOT(XP1,YP1,KOUNT)
      CALL TVNEXT
      RETURN
 1000 FORMAT(4HT = ,F6.3)
      END


      SUBROUTINE SG2
      COMMON/TVPOOL/XMIN,XMAX,YMIN,YMAX,TXMIN,TXMAX,TYMIN,TYMAX
      COMMON/TVGUIDE/TMODE,TEXT,ITV
      COMMON/TVFACT/FACT
      COMMON/TVTUNE/LPEN,LPEF,ITAL,IWINK,INTS,IRT,IUP
      COMMON QO,LQO,TNUX4,TNUY4
      COMMON DTVP,DELTAS,TVP
      COMMON MSK7,MSK8,MKC7,MKC8,IMP7,IMP8,DV7,DV8
      COMMON AGXL,AGYH,DT,DTCP,DTP,DTPP,DXC,DXCD2,DXCSQ,DYC,DYCD2,DYCSQ,
     $  DXIN,GX,GY,H,NI,NIM1,NJ,NJM1,NP,NTP,NU,T,TCP,TL,TP,TPP,W,Z,DYIN
      COMMON KD,DXY,DYX,NIP1,NJP1,TDXY,GXD,GYD,TNUX,TNUY,DXDY2,DYDX2,
     $       DXY4,DYX4,ZR,ZDXR,ZDYR,GXT,GYT,DX4,DY4,MI,MO,UP,VP,XP,YP
      COMMON CS,D,P,PS,R,U,V,XC,YC
      COMMON MSK(6),MKC(6),IMP(6),DV(6)
      DIMENSION CS(25,100),D(25,100),P(25,100),PS(3000),R(25,100),
     $          U(26,101),V(26,101),XC(25),YC(100)
      DIMENSION UP(3000),VP(3000),XP(3000),YP(3000)
      INTEGER CS,FLAG,PS
      REAL NU
      LINE =0
      WRITE (MO,1000) T
      WRITE (MO,1001)
      DO 104 J=2,NJM1
      JP=J+1
      DO 104 I=2,NIM1
      IP=I+1
      K1=CS(I,J).AND.MSK
      IF (K1.EQ.4) GO TO 100
      IF (K1.NE.5) GO TO 101
  100 D(I,J)=(U(IP,J)-U(I,J))/DXC+(V(I,JP)-V(I,J))/DYC
      GO TO 102
  101 D(I,J)=0.
  102 U1=(U(IP,J)+U(I,J))*.5
      V1=(V(I,JP)+V(I,J))*.5
      WRITE (MO,1002) I,J,XC(I),YC(J),U1,V1,R(I,J),P(I,J),D(I,J),CS(I,J)
      LINE=LINE+1
      IF (LINE .EQ. 50) 103,104
  103 LINE=0
      WRITE (MO,1003)
      WRITE (MO,1001)
  104 CONTINUE
      WRITE (MO,1004) T
      RETURN
 1000 FORMAT(1H1,10X22HCELL PRINT FOR TIME = ,F6.3///)
 1001 FORMAT(1H+,123X2HCS/1H ,2X1HI,3X1HJ,12X1HX,14X1HY,14X1HU,14X1HV,
     $14X1HR,12X3HPHI,14X1HD,9X8H87654321/)
 1002 FORMAT(1H ,1X,I2,I4,3X,7E15.6,3X,09)
 1003 FORMAT(1H1)
 1004 FORMAT(1H0,20X36H*****  END OF CELL PRINT FOR TIME = ,F6.3,7H  ***
     $**///)
      END


      SUBROUTINE SG3
      COMMON/TVPOOL/XMIN,XMAX,YMIN,YMAX,TXMIN,TXMAX,TYMIN,TYMAX
      COMMON/TVGUIDE/TMODE,TEXT,ITV
      COMMON/TVFACT/FACT
      COMMON/TVTUNE/LPEN,LPEF,ITAL,IWINK,INTS,IRT,IUP
      COMMON QO,LQO,TNUX4,TNUY4
      COMMON DTVP,DELTAS,TVP
      COMMON MSK7,MSK8,MKC7,MKC8,IMP7,IMP8,DV7,DV8
      COMMON AGXL,AGYH,DT,DTCP,DTP,DTPP,DXC,DXCD2,DXCSQ,DYC,DYCD2,DYCSQ,
     $  DXIN,GX,GY,H,NI,NIM1,NJ,NJM1,NP,NTP,NU,T,TCP,TL,TP,TPP,W,Z,DYIN
      COMMON KD,DXY,DYX,NIP1,NJP1,TDXY,GXD,GYD,TNUX,TNUY,DXDY2,DYDX2,
     $       DXY4,DYX4,ZR,ZDXR,ZDYR,GXT,GYT,DX4,DY4,MI,MO,UP,VP,XP,YP
      COMMON CS,D,P,PS,R,U,V,XC,YC
      COMMON MSK(6),MKC(6),IMP(6),DV(6)
      DIMENSION CS(25,100),D(25,100),P(25,100),PS(3000),R(25,100),
     $          U(26,101),V(26,101),XC(25),YC(100)
      DIMENSION UP(3000),VP(3000),XP(3000),YP(3000)
      INTEGER CS,FLAG,PS
      REAL NU
      LINE=0
      WRITE (MO,1000) T
      WRITE (MO,1001)
      DO 102 K=1,NP
      KP=PS(K).AND.MSK
      IF (KP.EQ.3) GO TO 102
  100 WRITE (MO,1002) K,XP(K),YP(K),UP(K),VP(K)
```

```
      LINE=LINE+1
      IF (LINE .EQ. 50) 101,102
  101 LINE=0
      WRITE (MO,1003)
      WRITE (MO,1001)
  102 CONTINUE
      WRITE (MO,1004) T
      RETURN
 1000 FORMAT(1H1,10X26HPARTICLE PRINT FOR TIME = ,F6.3///)
 1001 FORMAT(1H+,83X2HPS/1H ,13X1HK,14X1HX,14X1HY,14X1HU,14X1HV/)
 1002 FORMAT(1H ,10X,I4,5X,4E15.6)
 1003 FORMAT(1H1)
 1004 FORMAT(1H0,20X40H*****  END OF PARTICLE PRINT FOR TIME = ,F6.3,7H
     $ *****///)
      END


      SUBROUTINE PLOTV
      COMMON/TVPOOL/XMIN,XMAX,YMIN,YMAX,TXMIN,TXMAX,TYMIN,TYMAX
      COMMON/TVGUIDE/TMODE,TEXT,ITV
      COMMON/TVFACT/FACT
      COMMON/TVTUNE/LPEN,LPEF,ITAL,IWINK,INTS,IRT,IUP
      COMMON QO,LQO,TNUX4,TNUY4
      COMMON DTVP,DELTAS,TVP
      COMMON MSK7,MSK8,MKC7,MKC8,IMP7,IMP8,DV7,DV8
      COMMON AGXL,AGYH,DT,DTCP,DTP,DTPP,DXC,DXCD2,DXCSQ,DYC,DYCD2,DYCSQ,
     $  DXIN,GX,GY,H,NI,NIM1,NJ,NJM1,NP,NTP,NU,T,TCP,TL,TP,TPP,W,Z,DYIN
      COMMON KD,DXY,DYX,NIP1,NJP1,TDXY,GXD,GYD,TNUX,TNUY,DXDY2,DYDX2,
     $       DXY4,DYX4,ZR,ZDXR,ZDYR,GXT,GYT,DX4,DY4,MI,MO,UP,VP,XP,YP
      COMMON CS,D,P,PS,R,U,V,XC,YC
      COMMON MSK(6),MKC(6),IMP(6),DV(6)
      DIMENSION CS(25,100),D(25,100),P(25,100),PS(3000),R(25,100),
     $          U(26,101),V(26,101),XC(25),YC(100)
      DIMENSION UP(3000),VP(3000),XP(3000),YP(3000)
      INTEGER CS,FLAG,PS
      REAL NU
      A=A
      RETURN
      END


      SUBROUTINE OV3
      COMMON/TVPOOL/XMIN,XMAX,YMIN,YMAX,TXMIN,TXMAX,TYMIN,TYMAX
      COMMON/TVGUIDE/TMODE,TEXT,ITV
      COMMON/TVFACT/FACT
      COMMON/TVTUNE/LPEN,LPEF,ITAL,IWINK,INTS,IRT,IUP
      COMMON QO,LQO,TNUX4,TNUY4
      COMMON DTVP,DELTAS,TVP
      COMMON MSK7,MSK8,MKC7,MKC8,IMP7,IMP8,DV7,DV8
      COMMON AGXL,AGYH,DT,DTCP,DTP,DTPP,DXC,DXCD2,DXCSQ,DYC,DYCD2,DYCSQ,
     $  DXIN,GX,GY,H,NI,NIM1,NJ,NJM1,NP,NTP,NU,T,TCP,TL,TP,TPP,W,Z,DYIN
      COMMON KD,DXY,DYX,NIP1,NJP1,TDXY,GXD,GYD,TNUX,TNUY,DXDY2,DYDX2,
     $       DXY4,DYX4,ZR,ZDXR,ZDYR,GXT,GYT,DX4,DY4,MI,MO,UP,VP,XP,YP
      COMMON CS,D,P,PS,R,U,V,XC,YC
      COMMON MSK(6),MKC(6),IMP(6),DV(6)
      DIMENSION CS(25,100),D(25,100),P(25,100),PS(3000),R(25,100),
     $          U(26,101),V(26,101),XC(25),YC(100)
      DIMENSION UP(3000),VP(3000),XP(3000),YP(3000)
      INTEGER CS,FLAG,PS
      REAL NU
      CALL SG4
      IF (KD.EQ.2) GO TO 102
  100 CALL SG5
      IF (KD.EQ.2) GO TO 102
  101 CALL SG6
  102 RETURN
      END


      SUBROUTINE SG4
      COMMON/TVPOOL/XMIN,XMAX,YMIN,YMAX,TXMIN,TXMAX,TYMIN,TYMAX
      COMMON/TVGUIDE/TMODE,TEXT,ITV
      COMMON/TVFACT/FACT
      COMMON/TVTUNE/LPEN,LPEF,ITAL,IWINK,INTS,IRT,IUP
      COMMON QO,LQO,TNUX4,TNUY4
      COMMON DTVP,DELTAS,TVP
      COMMON MSK7,MSK8,MKC7,MKC8,IMP7,IMP8,DV7,DV8
      COMMON AGXL,AGYH,DT,DTCP,DTP,DTPP,DXC,DXCD2,DXCSQ,DYC,DYCD2,DYCSQ,
     $  DXIN,GX,GY,H,NI,NIM1,NJ,NJM1,NP,NTP,NU,T,TCP,TL,TP,TPP,W,Z,DYIN
      COMMON KD,DXY,DYX,NIP1,NJP1,TDXY,GXD,GYD,TNUX,TNUY,DXDY2,DYDX2,
     $       DXY4,DYX4,ZR,ZDXR,ZDYR,GXT,GYT,DX4,DY4,MI,MO,UP,VP,XP,YP
      COMMON CS,D,P,PS,R,U,V,XC,YC
      COMMON MSK(6),MKC(6),IMP(6),DV(6)
      DIMENSION CS(25,100),D(25,100),P(25,100),PS(3000),R(25,100),
     $          U(26,101),V(26,101),XC(25),YC(100)
      DIMENSION UP(3000),VP(3000),XP(3000),YP(3000)
      DIMENSION SU(25,100),SV(25,100),NK(25,100)
      EQUIVALENCE (D,SU),(R,SV)
      INTEGER CS,FLAG,PS
      REAL NU
      DO 114 K=1,NP
      KP=PS(K).AND.MSK
      IF (KP.EQ.3) GO TO 114
  100 I=XP(K)/DXC+2.
      J=YP(K)/DYC+2.
      IF (KP.EQ.2) GO TO 103
      IF (KP.EQ.3) GO TO 114
  101 K1=CS(I,J).AND.MSK
      IF (K1.EQ.1) GO TO 102
      IF (K1.NE.2) GO TO 114
  102 PS(K)=3.OR.(PS(K).AND.MKC)
      XP(K)=0.
      YP(K)=0.
      UP(K)=0.
      VP(K)=0.
      GO TO 114
  103 K2=CS(I,J).AND.MSK(2)
      K2=K2*DV(2)
      IF (K2.EQ.1) GO TO 114
  104 PS(K)=1.OR.(PS(K).AND.MKC)
      DO 105 L=1,NP
      LP=PS(L).AND.MSK
      IF (LP .EQ. 3) 106,105
  105 CONTINUE
      CALL NOAVL(I,J)
      GO TO 114
  106 KB=PS(K).AND.MSK(2)
      KB=KB*DV(2)
      GO TO (107,108,111,110),KB
  107 XP(L)=XP(K)-DXIN
      GO TO 109
  108 XP(L)=XP(K)+DXIN
```

```
109 YP(L)=YP(K)
    GO TO 113
110 YP(L)=YP(K)+DYIN
    GO TO 112
111 YP(L)=YP(K)-DYIN
112 XP(L)=XP(K)
113 PS(L)=2.OR.(PS(L).AND.MKC)
114 CONTINUE
    DO 700 J=1,NJ
    DO 700 I=1,NI
700 SU(I,J)=0.
    DO 701 J=1,NJ
    DO 701 I=1,NI
701 SV(I,J)=0.
    DO 702 J=1,NJ
    DO 702 I=1,NI
702 NK(I,J)=0
    DO 703 K=1,NP
    KP=PS(K).AND.MSK
    IF (KP.EQ.3) GO TO 703
    I=XP(K)/DXC+2.
    J=YP(K)/DYC+2.
    NK(I,J)=NK(I,J)+1
    SU(I,J)=SU(I,J)+UP(K)
    SV(I,J)=SV(I,J)+VP(K)
703 CONTINUE
    DO 135 J=2,NJM1
    DO 135 I=2,NIM1
    K1=CS(I,J).AND.MSK
    K11=CS(I+1,J).AND.MSK
    K12=CS(I-1,J).AND.MSK
    K13=CS(I,J+1).AND.MSK
    K14=CS(I,J-1).AND.MSK
    IF (K1.EQ.5) GO TO 126
    IF (K1.NE.3) GO TO 135
115 P(I,J)=0.
    IF (NK(I,J).EQ.0) GO TO 135
116 CS(I,J)=5.OR.(CS(I,J).AND.MKC)
    K21=CS(I+1,J).AND.MSK(2)
    K21=K21*DV(2)
    K22=CS(I-1,J).AND.MSK(2)
    K22=K22*DV(2)
    K23=CS(I,J+1).AND.MSK(2)
    K23=K23*DV(2)
    K24=CS(I,J-1).AND.MSK(2)
    K24=K24*DV(2)
    IF (K11.EQ.3) GO TO 118
117 IF (K21.NE.2) GO TO 119
118 U(I+1,J)=SU(I,J)/NK(I,J)
119 IF (K12.EQ.3) GO TO 121
120 IF (K22.NE.2) GO TO 122
121 U(I,J)=SU(I,J)/NK(I,J)
122 IF (K13.EQ.3) GO TO 123
    IF (K23.NE.2) GO TO 124
123 V(I,J+1)=SV(I,J)/NK(I,J)
124 IF (K14.EQ.3) GO TO 125
    IF (K24.NE.2) GO TO 135
125 V(I,J)=SV(I,J)/NK(I,J)
    GO TO 135
126 IF (NK(I,J).NE.0) GO TO 135
127 CS(I,J)=3.OR.(CS(I,J).AND.MKC)
    P(I,J)=0.
    IF (K11.NE.3) GO TO 129
128 U(I+1,J)=0.
129 IF (K12.NE.3) GO TO 131
130 U(I,J)=0.
131 IF (K13.NE.3) GO TO 133
132 V(I,J+1)=0.
133 IF (K14.NE.3) GO TO 135
134 V(I,J)=0.
135 CONTINUE
    DO 156 J=2,NJM1
    DO 156 I=2,NIM1
    IF (NK(I,J).EQ.0) GO TO 156
136 K11=CS(I+1,J).AND.MSK
    K12=CS(I-1,J).AND.MSK
    K13=CS(I,J+1).AND.MSK
    K14=CS(I,J-1).AND.MSK
    K1=CS(I,J).AND.MSK
    IF (K1.NE.4) GO TO 142
137 IF (K11.EQ.3) GO TO 141
138 IF (K12.EQ.3) GO TO 141
139 IF (K13.EQ.3) GO TO 141
140 IF (K14.NE.3) GO TO 156
141 CS(I,J)=5.OR.(CS(I,J).AND.MKC)
    P(I,J)=0.
    GO TO 156
142 P(I,J)=0.
    IF (K1.NE.5) GO TO 156
143 SP=0.
    ANP=0.
    IF (K11.EQ.3) GO TO 156
144 IF (K11.EQ.4) GO TO 151
145 IF (K12.EQ.3) GO TO 156
146 IF (K12.EQ.4) GO TO 152
147 IF (K13.EQ.3) GO TO 156
148 IF (K13.EQ.4) GO TO 153
149 IF (K14.EQ.3) GO TO 156
150 IF (K14.EQ.4) GO TO 154
    GO TO 155
151 SP=SP+P(I+1,J)
    ANP=ANP+1.
    GO TO 145
152 SP=SP+P(I-1,J)
    ANP=ANP+1.
    GO TO 147
153 SP=SP+P(I,J+1)
    ANP=ANP+1.
    GO TO 149
154 SP=SP+P(I,J-1)
    ANP=ANP+1.
155 CS(I,J)=4.OR.(CS(I,J).AND.MKC)
    IF (ANP .EQ. 0.) 156,300
300 P(I,J)=SP/ANP
156 CONTINUE
    CALL VELSUR
    DO 200 J=2,NJM1
    JM=J-1
    JP=J+1
    DO 200 I=2,NIM1
    IM=I-1
    IP=I+1
```

```
      K1=CS(I,J).AND.MSK
      IF (K1.NE.5) GO TO 200
      K11=CS(IM,J).AND.MSK
      K12=CS(IP,J).AND.MSK
      K13=CS(I,JM).AND.MSK
      K14=CS(I,JP).AND.MSK
      IF (K11.EQ.3.AND.K12.NE.3.AND.K13.NE.3.AND.K14.NE.3) GO TO 157
      IF (K11.NE.3.AND.K12.EQ.3.AND.K13.NE.3.AND.K14.NE.3) GO TO 157
      IF (K11.NE.3.AND.K12.NE.3.AND.K13.EQ.3.AND.K14.NE.3) GO TO 158
      IF (K11.NE.3.AND.K12.NE.3.AND.K13.NE.3.AND.K14.EQ.3) GO TO 158
      IF (K12.EQ.3.AND.K14.EQ.3.AND.K11.NE.3.AND.K13.NE.3) GO TO 159
      IF (K12.EQ.3.AND.K13.EQ.3.AND.K11.NE.3.AND.K14.NE.3) GO TO 160
      IF (K11.EQ.3.AND.K13.EQ.3.AND.K12.NE.3.AND.K14.NE.3) GO TO 161
      IF (K11.EQ.3.AND.K14.EQ.3.AND.K12.NE.3.AND.K13.NE.3) GO TO 162
      P(I,J)=0.
      GO TO 200
  157 P(I,J)=TNUX*(U(IP,J)-U(I,J))
      GO TO 200
  158 P(I,J)=TNUY*(V(I,JP)-V(I,J))
      GO TO 200
  159 P(I,J)=TNUY4*(U(I,J)+U(IP,J)-U(I,JM)-U(IP,JM))+TNUX4*(V(I,JP)+
     $        V(I,J)-V(IM,JP)-V(IM,J))
      GO TO 200
  160 P(I,J)=-TNUY4*(U(IP,JP)+U(I,JP)-U(IP,J)-U(I,J))-TNUX4*(V(I,JP)+
     $        V(I,J)-V(IM,JP)-V(IM,J))
      GO TO 200
  161 P(I,J)=TNUY4*(U(IP,JP)+U(I,JP)-U(IP,J)-U(I,J))+TNUY4*(V(IP,JP)+
     $        V(IP,J)-V(I,JP)-V(I,J))
      GO TO 200
  162 P(I,J)=-TNUY4*(U(I,J)+U(IP,J)-U(I,JM)-U(IP,JM))-TNUX4*(V(IP,JP)+
     $        V(IP,J)-V(I,JP)-V(I,J))
  200 CONTINUE
  163 DO 166 J=2,NJM1
      DO 166 I=2,NIM1
      K1=CS(I,J).AND.MSK
      IF (K1.EQ.4) GO TO 164
      IF (K1.NE.5) GO TO 166
  164 D(I,J)=(U(I+1,J)-U(I,J))/DXC+(V(I,J+1)-V(I,J))/DYC
  166 CONTINUE
      RETURN
      END


      SUBROUTINE NOAVL(I,J)
      COMMON/TVPOOL/XMIN,XMAX,YMIN,YMAX,TXMIN,TXMAX,TYMIN,TYMAX
      COMMON/TVGUIDE/TMODE,TEXT,ITV
      COMMON/TVFACT/FACT
      COMMON/TVTUNE/LPEN,LPEF,ITAL,IWINK,INTS,IRT,IUP
      COMMON Q0,LQ0,TNUX4,TNUY4
      COMMON DTVP,DELTAS,TVP
      COMMON MSK7,MSK8,MKC7,MKC8,IMP7,IMP8,DV7,DV8
      COMMON AGXL,AGYH,DT,DTCP,DTP,DTPP,DXC,DXCD2,DXCSQ,DYC,DYCD2,DYCSQ,
     $  DXIN,GX,GY,H,NI,NIM1,NJ,NJM1,NP,NTP,NU,T,TCP,TL,TP,TPP,W,Z,DYIN
      COMMON KD,DXY,DYX,NIP1,NJP1,TDXY,GXD,GYD,TNUX,TNUY,DXDY2,DYDX2,
     $        DXY4,DYX4,ZR,ZDXR,ZDYR,GXT,GYT,DX4,DY4,MI,MO,UP,VP,XP,YP
      COMMON CS,D,P,PS,R,U,V,XC,YC
      COMMON MSK(6),MKC(6),IMP(6),DV(6)
      DIMENSION CS(25,100),D(25,100),P(25,100),PS(3000),R(25,100),
     $          U(26,101),V(26,101),XC(25),YC(100)
      DIMENSION UP(3000),VP(3000),XP(3000),YP(3000)
      INTEGER CS,FLAG,PS
      REAL NU
      WRITE (MO,1000) T,I,J
      RETURN
 1000 FORMAT(1H ,10X40HNO PARTICLE STORAGE AVAILABLE AT TIME = ,F6.3,41H
     $  --  CANNOT CREATE PARTICLE FOR IN CELL ,I3,1H,,I2)
      END


      SUBROUTINE VELSUR
      COMMON/TVPOOL/XMIN,XMAX,YMIN,YMAX,TXMIN,TXMAX,TYMIN,TYMAX
      COMMON/TVGUIDE/TMODE,TEXT,ITV
      COMMON/TVFACT/FACT
      COMMON/TVTUNE/LPEN,LPEF,ITAL,IWINK,INTS,IRT,IUP
      COMMON Q0,LQ0,TNUX4,TNUY4
      COMMON DTVP,DELTAS,TVP
      COMMON MSK7,MSK8,MKC7,MKC8,IMP7,IMP8,DV7,DV8
      COMMON AGXL,AGYH,DT,DTCP,DTP,DTPP,DXC,DXCD2,DXCSQ,DYC,DYCD2,DYCSQ,
     $  DXIN,GX,GY,H,NI,NIM1,NJ,NJM1,NP,NTP,NU,T,TCP,TL,TP,TPP,W,Z,DYIN
      COMMON KD,DXY,DYX,NIP1,NJP1,TDXY,GXD,GYD,TNUX,TNUY,DXDY2,DYDX2,
     $        DXY4,DYX4,ZR,ZDXR,ZDYR,GXT,GYT,DX4,DY4,MI,MO,UP,VP,XP,YP
      COMMON CS,D,P,PS,R,U,V,XC,YC
      COMMON MSK(6),MKC(6),IMP(6),DV(6)
      DIMENSION CS(25,100),D(25,100),P(25,100),PS(3000),R(25,100),
     $          U(26,101),V(26,101),XC(25),YC(100)
      DIMENSION UP(3000),VP(3000),XP(3000),YP(3000)
      INTEGER CS,FLAG,PS
      REAL NU
      DO 108 J=2,NJM1
      DO 108 I=2,NIM1
      K1=CS(I,J).AND.MSK
      K11=CS(I+1,J).AND.MSK
      K12=CS(I-1,J).AND.MSK
      K13=CS(I,J+1).AND.MSK
      K14=CS(I,J-1).AND.MSK
      IF (K1.NE.5) GO TO 108
      IF (K11.EQ.2.OR.K12.EQ.2.OR.K13.EQ.2.OR.K14.EQ.2) GO TO 108
      IF (K11.EQ.3.AND.K12.NE.3.AND.K13.NE.3.AND.K14.NE.3) GO TO 104
      IF (K12.EQ.3.AND.K11.NE.3.AND.K13.NE.3.AND.K14.NE.3) GO TO 105
      IF (K13.EQ.3.AND.K11.NE.3.AND.K12.NE.3.AND.K14.NE.3) GO TO 106
      IF (K14.NE.3.OR.K11.EQ.3.OR.K12.EQ.3.OR.K13.EQ.3) GO TO 108
      V(I,J)=V(I,J+1)+(U(I+1,J)-U(I,J))*DYX
      GO TO 108
  104 U(I+1,J)=U(I,J)+(V(I,J)-V(I,J+1))*DXY
      GO TO 108
  105 U(I,J)=U(I+1,J)+(V(I,J+1)-V(I,J))*DXY
      GO TO 108
  106 V(I,J+1)=V(I,J)+(U(I,J)-U(I+1,J))*DYX
  108 CONTINUE
      RETURN
      END


      SUBROUTINE SG5
      COMMON/TVPOOL/XMIN,XMAX,YMIN,YMAX,TXMIN,TXMAX,TYMIN,TYMAX
      COMMON/TVGUIDE/TMODE,TEXT,ITV
      COMMON/TVFACT/FACT
      COMMON/TVTUNE/LPEN,LPEF,ITAL,IWINK,INTS,IRT,IUP
      COMMON Q0,LQ0,TNUX4,TNUY4
      COMMON DTVP,DELTAS,TVP
      COMMON MSK7,MSK8,MKC7,MKC8,IMP7,IMP8,DV7,DV8
      COMMON AGXL,AGYH,DT,DTCP,DTP,DTPP,DXC,DXCD2,DXCSQ,DYC,DYCD2,DYCSQ,
```

```
     $  DXIN,GX,GY,H,NI,NIM1,NJ,NJM1,NP,NTP,NU,T,TCP,TL,TP,TPP,W,Z,DYIN
      COMMON KD,DXY,DYX,NIP1,NJP1,TDXY,GXD,GYD,TNUX,TNUY,DXDY2,DYDX2,
     $       DXY4,DYX4,ZR,ZDXR,ZDYR,GXT,GYT,DX4,DY4,MI,MO,UP,VP,XP,YP
      COMMON CS,D,P,PS,R,U,V,XC,YC
      COMMON MSK(6),MKC(6),IMP(6),DV(6)
      DIMENSION CS(25,100),D(25,100),P(25,100),PS(3000),R(25,100),
     $          U(26,101),V(26,101),XC(25),YC(100)
      DIMENSION UP(3000),VP(3000),XP(3000),YP(3000)
      INTEGER CS,FLAG,PS
      REAL NU
      DIMENSION P1(25,100),U1(26,101),V1(26,101)
      INTEGER COUNT
      IF (T.NE.DT) GO TO 99
      DO 98 J=1,NJP1
      DO 98 I=1,NIP1
      U1(I,J)=0.
   98 V1(I,J)=0.
   99 SR=0.
      DO 150 J=2,NJM1
      JM=J-1
      JP=J+1
      DO 150 I=2,NIM1
      IM=I-1
      IP=I+1
      K1=CS(I,J).AND.MSK
      IF (K1.NE.4) GO TO 150
  100 S1=0.
      S2=0.
      S3=0.
      S4=0.
      S5=0.
      K51=CS(IP,J).AND.MSK(5)
      K51=K51*DV(5)
      IF (K51.EQ.1) GO TO 104
  101 K41=CS(IP,JP).AND.MSK(4)
      K41=K41*DV(4)
      IF (K41.EQ.1) GO TO 103
  102 S2=S2-U(IP,JP)/DXC
      GO TO 104
  103 S2=S2-U(IP,JM)/DXC
  104 K52=CS(IM,J).AND.MSK(5)
      K52=K52*DV(5)
      IF (K52.EQ.1) GO TO 108
  105 K42=CS(IM,JP).AND.MSK(4)
      K42=K42*DV(4)
      IF (K42.EQ.1) GO TO 107
  106 S2=S2+U(I,JP)/DXC
      GO TO 108
  107 S2=S2+U(I,JM)/DXC
  108 K53=CS(I,JP).AND.MSK(5)
      K53=K53*DV(5)
      IF (K53.EQ.1) GO TO 112
  109 K43=CS(IP,JP).AND.MSK(4)
      K43=K43*DV(4)
      IF (K43.EQ.1) GO TO 111
  110 S1=S1-V(IP,JP)/DYC
      GO TO 112
  111 S1=S1-V(IM,JP)/DYC
  112 K54=CS(I,JM).AND.MSK(5)
      K54=K54*DV(5)
      IF (K54.EQ.1) GO TO 116
  113 K44=CS(IP,JM).AND.MSK(4)
      K44=K44*DV(4)
      IF (K44.EQ.1) GO TO 115
  114 S1=S1+V(IP,J)/DYC
      GO TO 116
  115 S1=S1+V(IM,J)/DYC
  116 K4=CS(I,J).AND.MSK(4)
      K4=K4*DV(4)
      IF (K4.EQ.2) GO TO 133
  117 K55=CS(IP,JM).AND.MSK(5)
      K55=K55*DV(5)
      IF (K55.EQ.1) GO TO 121
  118 K21=CS(IP,JM).AND.MSK(2)
      K21=K21*DV(2)
      IF (K21.NE.3) GO TO 120
  119 S1=S1-V(I,J)/DYC
      S2=S2+U(IP,J)/DXC
      GO TO 121
  120 S1=S1+V(I,J)/DYC
      S2=S2-U(IP,J)/DXC
  121 K56=CS(IM,JM).AND.MSK(5)
      K56=K56*DV(5)
      IF (K56.EQ.1) GO TO 125
  122 K22=CS(IM,JM).AND.MSK(2)
      K22=K22*DV(2)
      IF (K22.NE.3) GO TO 124
  123 S1=S1-V(I,J)/DYC
      S2=S2-U(I,J)/DXC
      GO TO 125
  124 S1=S1+V(I,J)/DYC
      S2=S2+U(I,J)/DXC
  125 K57=CS(IP,JP).AND.MSK(5)
      K57=K57*DV(5)
      IF (K57.EQ.1) GO TO 129
  126 K23=CS(IP,JP).AND.MSK(2)
      K23=K23*DV(2)
      IF (K23.NE.3) GO TO 128
  127 S1=S1+V(I,JP)/DYC
      S2=S2+U(IP,J)/DXC
      GO TO 129
  128 S1=S1-V(I,JP)/DYC
      S2=S2-U(IP,J)/DXC
  129 K58=CS(IM,JP).AND.MSK(5)
      K58=K58*DV(5)
      IF (K58.EQ.1) GO TO 133
  130 K24=CS(IM,JP).AND.MSK(2)
      K24=K24*DV(2)
      IF (K24.NE.3) GO TO 132
  131 S1=S1+V(I,JP)/DYC
      S2=S2-U(I,J)/DXC
      GO TO 133
  132 S1=S1-V(I,JP)/DYC
      S2=S2+U(I,J)/DXC
  133 T1=(U(IP,J)+U(I,J))*.5
      UIJ=T1*T1
      T2=(V(I,JP)+V(I,J))*.5
      VIJ=T2*T2
      K11=CS(IP,J).AND.MSK
      IF (K11.EQ.2) GO TO 135
  134 S1=S1+D(IP,J)-D(I,J)
      T3=(U(I+2,J)+U(IP,J))*.5
```

```
      S3=S3+T3*T3-UIJ
  135 K12=CS(IM,J).AND.MSK
      IF (K12.EQ.2) GO TO 137
  136 S1=S1+D(IM,J)-D(I,J)
      T4=(U(I,J)+U(IM,J))*.5
      S3=S3+T4*T4-UIJ
  137 S1=S1/DXCSQ
      S3=S3/DXCSQ
      K13=CS(I,JP).AND.MSK
      IF (K13.EQ.2) GO TO 139
  138 S2=S2+D(I,JP)-D(I,J)
      T5=(V(I,J+2)+V(I,JP))*.5
      S4=S4+T5*T5-VIJ
  139 K14=CS(I,JM).AND.MSK
      IF (K14.EQ.2) GO TO 141
  140 S2=S2+D(I,JM)-D(I,J)
      T6=(V(I,J)+V(I,JM))*.5
      S4=S4+T6*T6-VIJ
  141 S2=S2/DYCSQ
      S4=S4/DYCSQ
      S1=-NU*(S1+S2)
      S2=S3+S4
      K6=CS(I,J).AND.MSK(6)
      K6=K6*DV(6)
      IF (K6.EQ.2) GO TO 143
  142 S5=(U(IP,J)+U(IP,JP))*(V(I,JP)+V(IP,JP))
  143 K61=CS(IM,JM).AND.MSK(6)
      K61=K61*DV(6)
      IF (K61.EQ.2) GO TO 145
  144 S5=S5+(U(I,J)+U(I,JM))*(V(I,J)+V(IM,J))
  145 K62=CS(I,JM).AND.MSK(6)
      K62=K62*DV(6)
      IF (K62.EQ.2) GO TO 147
  146 S5=S5-(U(IP,J)+U(IP,JM))*(V(I,J)+V(IP,J))
  147 K63=CS(IM,J).AND.MSK(6)
      K63=K63*DV(6)
      IF (K63.EQ.2) GO TO 149
  148 S5=S5-(U(I,J)+U(I,JP))*(V(IM,JP)+V(I,JP))
  149 R(I,J)=S1+S2+S5*TDXY-D(I,J)/DT
      SR=SR+ABS(R(I,J))
  150 CONTINUE
      IF (QQ.EQ.0.) GO TO 496
      Q=QQ
      GO TO 497
  496 Q=1.
  497 LQ=LQQ
      DO 151 J=1,NJ
      DO 151 I=1,NI
  151 P1(I,J)=P(I,J)
      COUNT=0
  152 IF (COUNT .EQ. 4000) 498,499
  498 WRITE (MO,1002) CONMAX,T
      GO TO 501
  499 IF (COUNT .GT. 4999) 500,501
  500 KD=2
      WRITE (MO,1001) T,CONMAX
      RETURN
  501 CONMAX=0.
      KF=1
      DO 182 L=1,10
      COUNT=COUNT+1

      REM=0.
      DO 181 J=2,NJM1
      JM=J-1
      JP=J+1
      DO 181 I=2,NIM1
      IM=I-1
      IP=I+1
      K1=CS(I,J).AND.MSK
      IF (K1.NE.4) GO TO 181
  154 K4=CS(I,J).AND.MSK(4)
      K4=K4*DV(4)
      IF (K4.EQ.1) GO TO 179
  155 K11=CS(I,JP).AND.MSK
      IF (K11.NE.2) GO TO 161
  156 K21=CS(I,JP).AND.MSK(2)
      K21=K21*DV(2)
      GO TO (700,160,159,157,181),K21
  700 P1(I,JP)=P1(I,J)+GYD+DYDX2*(V(IP,JP)+V(IM,JP)-2.*V(I,JP))
      GO TO 161
  157 P1(I,JP)=P1(I,J)+GYD+(V(I,J)-V(I,JP))*TNUY
      K31=CS(I,JP).AND.MSK(3)
      K31=K31*DV(3)
      IF (K31.EQ.1) GO TO 161
  158 P1(I,JP)=P1(I,JP)-(V(IP,JP)+V(IM,JP))*DYDX2
      GO TO 161
  159 P1(I,JP)=P1(I,J)+GYD
      GO TO 161
  160 P1(I,JP)=P1(I,J)+((U(I,J)+U(I,JP))*(V(IM,JP)+V(I,JP))-(U(IP,J)+
     $          U(IP,JP))*(V(I,JP)+V(IP,JP)))*DYX4
  161 K12=CS(I,JM).AND.MSK
      IF (K12.NE.2) GO TO 167
  162 K22=CS(I,JM).AND.MSK(2)
      K22=K22*DV(2)
      GO TO (163,166,165,163,181),K22
  163 P1(I,JM)=P1(I,J)-GYD-(V(I,JP)-V(I,J))*TNUY
      K32=CS(I,JM).AND.MSK(3)
      K32=K32*DV(3)
      IF (K32.EQ.1) GO TO 167
  164 P1(I,JM)=P1(I,JM)+(V(IP,J)+V(IM,J))*DYDX2
      GO TO 167
  165 P1(I,JM)=P1(I,J)-GYD
      GO TO 167
  166 P1(I,JM)=P1(I,J)-((U(I,J)+U(I,JM))*(V(I,J)+V(IM,J))-(U(IP,J)+
     $          U(IP,JM))*(V(IP,J)+V(I,J)))*DYX4
  167 K13=CS(IP,J).AND.MSK
      IF (K13.NE.2) GO TO 173
  168 K23=CS(IP,J).AND.MSK(2)
      K23=K23*DV(2)
      GO TO (169,172,171,169,181),K23
  169 P1(IP,J)=P1(I,J)+GXD+(U(I,J)-U(IP,J))*TNUX
      K33=CS(IP,J).AND.MSK(3)
      K33=K33*DV(3)
      IF (K33.EQ.1) GO TO 173
  170 P1(IP,J)=P1(IP,J)-(U(IP,JP)+U(IP,JM))*DXDY2
      GO TO 173
  171 P1(IP,J)=P1(I,J)+GXD
      GO TO 173
  172 P1(IP,J)=P1(I,J)+((U(IP,J)+U(IP,JM))*(V(IP,J)+V(I,J))-(U(IP,J)+
     $          U(IP,JP))*(V(I,JP)+V(IP,JP)))*DXY4
  173 K14=CS(IM,J).AND.MSK
      IF (K14.NE.2) GO TO 179
```

```
174 K24=CS(IM,J).AND.MSK(2)
    K24=K24*DV(2)
    GO TO (175,178,177,175,181),K24
175 P1(IM,J)=P1(I,J)-GXD-(U(IP,J)-U(I,J))*TNUX
    K34=CS(IM,J).AND.MSK(3)
    K34=K34*DV(3)
    IF (K34.EQ.1) GO TO 179
176 P1(IM,J)=P1(IM,J)+(U(I,JP)+U(I,JM))*DXDY2
    GO TO 179
177 P1(IM,J)=P1(I,J)-GXD
    GO TO 179
178 P1(IM,J)=P1(I,J)-((U(I,J)+U(I,JM))*(V(I,J)+V(IM,J))-(U(I,J)+
   $          U(I,JP))*(V(IM,JP)+V(I,JP)))*DXY4
179 RES=(P1(IP,J)+P1(IM,J))*ZDXR+(P1(I,JP)+P1(I,JM))*ZDYR+R(I,J)*ZR-
   $     P1(I,J)
    P1(I,J)=P1(I,J)+Q*RES
    REM=AMAX1(ABS(RES),REM)
    IF (L.EQ.9) RES1=REM
    IF (L .EQ. 10) 180,181
180 CONIJ=ABS(P1(I,J)-P(I,J))/(ABS(P1(I,J))+ABS(P(I,J))+(U(IP,J)
   $       +U(I,J))*(U(IP,J)+U(I,J))*.25+(V(I,JP)+V(I,J))*(V(I,JP)
   $       +V(I,J))*.25+AGYH+AGXL)
    CONMAX=AMAX1(CONMAX,CONIJ)
181 CONTINUE
182 CONTINUE
    DO 183 J=1,NJ
    DO 183 I=1,NI
183 P(I,J)=P1(I,J)
    IF (CONMAX.LT.2.E-4) GO TO 184
    IF (LQ.EQ.0) GO TO 152
    Q=2./(1.+SQRT(1.-REM/RES1))
    LQ=0
    GO TO 152
184 WRITE (MO,1000) COUNT,T,Q
    DO 258 J=2,NJM1
    JM=J-1
    JP=J+1
    DO 258 I=2,NIM1
    IM=I-1
    IP=I+1
    K11=CS(IP,J).AND.MSK
    K12=CS(I,JP).AND.MSK
    K13=CS(IP,JP).AND.MSK
    K14=CS(IM,J).AND.MSK
    K15=CS(I,JM).AND.MSK
    K21=CS(IP,JP).AND.MSK(2)
    K21=K21*DV(2)
    K22=CS(I,JP).AND.MSK(2)
    K22=K22*DV(2)
    K23=CS(IP,JM).AND.MSK(2)
    K23=K23*DV(2)
    K24=CS(I,JM).AND.MSK(2)
    K24=K24*DV(2)
    K25=CS(IP,J).AND.MSK(2)
    K25=K25*DV(2)
    K26=CS(IM,J).AND.MSK(2)
    K26=K26*DV(2)
    K27=CS(IM,JP).AND.MSK(2)
    K27=K27*DV(2)
    K51=CS(IP,JP).AND.MSK(5)
    K51=K51*DV(5)
    K52=CS(I,JP).AND.MSK(5)
    K52=K52*DV(5)
    K53=CS(IP,JM).AND.MSK(5)
    K53=K53*DV(5)
    K54=CS(I,JM).AND.MSK(5)
    K54=K54*DV(5)
    K55=CS(IP,J).AND.MSK(5)
    K55=K55*DV(5)
    K56=CS(IM,J).AND.MSK(5)
    K56=K56*DV(5)
    K57=CS(IM,JP).AND.MSK(5)
    K57=K57*DV(5)
    K6=CS(I,J).AND.MSK(6)
    K6=K6*DV(6)
    K1=CS(I,J).AND.MSK
    IF (K1.EQ.4) GO TO 185
    IF (K1.NE.5) GO TO 258
    IF (K11.NE.3) GO TO 600
    V(I+1,J+1)=V(I,J+1)
    V(I+1,J)=V(I,J)
600 IF (K14.NE.3) GO TO 601
    V(I-1,J+1)=V(I,J+1)
    V(I-1,J)=V(I,J)
601 IF (K12.NE.3) GO TO 602
    U(I,J+1)=U(I,J)
    U(I+1,J+1)=U(I+1,J)
602 IF (K15.NE.3) GO TO 186
    U(I,J-1)=U(I,J)
    U(I+1,J-1)=U(I+1,J)
185 IF (K11.EQ.2) GO TO 187
    IF (K11.NE.3) GO TO 188
186 U1(IP,J)=U(IP,J)+GXT
    GO TO 221
187 U1(IP,J)=U(IP,J)
    GO TO 221
188 IF (U(IP,JP) .EQ. 0.) 189,198
189 IF (K21.NE.4) GO TO 191
190 IF (K51.EQ.2) GO TO 193
191 IF (K22.NE.4) GO TO 194
192 IF (K52.EQ.1) GO TO 194
193 UP1=-U(IP,J)
    GO TO 202
194 IF (K21.NE.3) GO TO 196
195 IF (K51.EQ.2) GO TO 201
196 IF (K22.NE.3) GO TO 198
197 IF (K52.EQ.2) GO TO 201
198 IF (K12.NE.3) GO TO 200
199 IF (K13.EQ.3) GO TO 201
200 UP1=U(IP,JP)
    GO TO 202
201 UP1=U(IP,J)
202 IF (U(IP,JM) .EQ. 0.) 203,212
203 IF (K23.NE.4) GO TO 205
204 IF (K53.EQ.2) GO TO 207
205 IF (K24.NE.4) GO TO 208
206 IF (K54.EQ.1) GO TO 208
207 UM=-U(IP,J)
    GO TO 216
208 IF (K23.NE.3) GO TO 210
209 IF (K53.EQ.2) GO TO 215
210 IF (K24.NE.3) GO TO 212
```

```
211 IF (K54.EQ.2) GO TO 215
212 K14=CS(I,JM).AND.MSK
    IF (K14.NE.3) GO TO 214
213 K15=CS(IP,JM).AND.MSK
    IF (K15.EQ.3) GO TO 215
214 UM=U(IP,JM)
    GO TO 216
215 UM=U(IP,J)
216 S1=NU*((U(I+2,J)+U(I,J)-2.*U(IP,J))/DXCSQ+(UP1+UM-2.*U(IP,J))
   $  /DYCSQ)+(P(I,J)-P(IP,J))/DXC+GX
    S2=0.
    IF (K6.EQ.2) GO TO 218
217 S2=-(U(IP,J)+U(IP,JP))*(V(I,JP)+V(IP,JP))*.25
218 K61=CS(I,JM).AND.MSK(6)
    K61=K61*DV(6)
    IF (K61.EQ.2) GO TO 220
219 S2=S2+(U(IP,J)+U(IP,JM))*(V(IP,J)+V(I,J))*.25
220 T6=U(IP,J)+U(I,J)
    T7=U(I+2,J)+U(IP,J)
    U1(IP,J)=U(IP,J)+DT*(S1+S2/DYC+(T6*T6-T7*T7)*DX4)
221 IF (K12.NE.3) GO TO 223
222 V1(I,JP)=V(I,JP)+GYT
    GO TO 258
223 IF (K12.NE.2) GO TO 225
224 V1(I,JP)=V(I,JP)
    GO TO 258
225 IF (V(IP,JP) .EQ. 0.) 226,235
226 IF (K21.NE.4) GO TO 228
227 IF (K51.EQ.2) GO TO 230
228 IF (K25.NE.4) GO TO 231
229 IF (K55.EQ.1) GO TO 231
230 VP1=-V(I,JP)
    GO TO 239
231 IF (K21.NE.3) GO TO 233
232 IF (K51.EQ.2) GO TO 238
233 IF (K25.NE.3) GO TO 235
234 IF (K55.EQ.2) GO TO 238
235 IF (K11.NE.3) GO TO 237
236 IF (K13.EQ.3) GO TO 238
237 VP1=V(IP,JP)
    GO TO 239
238 VP1=V(I,JP)
239 IF (V(IM,JP) .EQ. 0.) 240,249
240 IF (K26.NE.4) GO TO 242
241 IF (K56.EQ.2) GO TO 244
242 IF (K27.NE.4) GO TO 245
243 IF (K57.EQ.1) GO TO 245
244 VM=-V(I,JP)
    GO TO 253
245 IF (K26.NE.3) GO TO 247
246 IF (K56.EQ.2) GO TO 252
247 IF (K27.NE.3) GO TO 249
248 IF (K57.EQ.2) GO TO 252
249 K16=CS(IM,J).AND.MSK
    IF (K16.NE.3) GO TO 251
250 K17=CS(IM,JP).AND.MSK
    IF (K17.EQ.3) GO TO 252
251 VM=V(IM,JP)
    GO TO 253
252 VM=V(I,JP)
253 S1=NU*((V(I,J+2)+V(I,J)-2.*V(I,JP))/DYCSQ+(VP1+VM-2.*V(I,JP))
   $  /DXCSQ)+(P(I,J)-P(I,JP))/DYC+GY
    S2=0.
    IF (K6.EQ.2) GO TO 255
254 S2=-(U(IP,J)+U(IP,JP))*(V(I,JP)+V(IP,JP))*.25
255 K62=CS(IM,J).AND.MSK(6)
    K62=K62*DV(6)
    IF (K62.EQ.2) GO TO 257
256 S2=S2+(U(I,J)+U(I,JP))*(V(IM,JP)+V(I,JP))*.25
257 T6=V(I,JP)+V(I,J)
    T7=V(I,J+2)+V(I,JP)
    V1(I,JP)=V(I,JP)+DT*(S1+S2/DXC+(T6*T6-T7*T7)*DY4)
258 CONTINUE
    DO 271 J=2,NJM1
    JP=J+1
    DO 271 I=2,NIM1
    IP=I+1
    K1=CS(I,J).AND.MSK
    IF (K1.EQ.5) GO TO 263
    IF (K1.NE.2) GO TO 271
259 K41=CS(IP,J).AND.MSK(4)
    K41=K41*DV(4)
    IF (K41.EQ.1) GO TO 261
260 U1(IP,J)=U(IP,J)
261 K42=CS(I,JP).AND.MSK(4)
    K42=K42*DV(4)
    IF (K42.EQ.1) GO TO 271
262 V1(I,JP)=V(I,JP)
    GO TO 271
263 K11=CS(I-1,J).AND.MSK
    IF (K11.NE.3) GO TO 265
264 U1(I,J)=U1(IP,J)
    GO TO 267
265 K12=CS(IP,J).AND.MSK
    IF (K12.NE.3) GO TO 267
266 U1(IP,J)=U1(I,J)
267 K13=CS(I,J-1).AND.MSK
    IF (K13.NE.3) GO TO 269
268 V1(I,J)=V1(I,JP)
    GO TO 271
269 K14=CS(I,JP).AND.MSK
    IF (K14.NE.3) GO TO 271
270 V1(I,JP)=V1(I,J)
271 CONTINUE
    DO 289 J=2,NJM1
    JM=J-1
    JP=J+1
    DO 289 I=2,NIM1
    IM=I-1
    IP=I+1
    K4=CS(I,J).AND.MSK(4)
    K4=K4*DV(4)
    IF (K4.EQ.1) GO TO 289
272 K1=CS(I,J).AND.MSK
    IF (K1.EQ.3) GO TO 289
273 K11=CS(IP,J).AND.MSK
    IF (K11.NE.2) GO TO 277
274 K21=CS(IP,J).AND.MSK(2)
    K21=K21*DV(2)
    IF (K21.NE.2) GO TO 276
275 U1(IP,J)=U1(I,J)-(V1(I,JP)-V1(I,J))*DXY
    GO TO 277
```

```
276 U1(IP,J)=U(IP,J)
277 K12=CS(IM,J).AND.MSK
    IF (K12.NE.2) GO TO 281
278 K22=CS(IM,J).AND.MSK(2)
    K22=K22*DV(2)
    IF (K22.NE.2) GO TO 280
279 U1(I,J)=U1(IP,J)+(V1(I,JP)-V1(I,J))*DXY
    GO TO 281
280 U1(I,J)=U(I,J)
281 K13=CS(I,JP).AND.MSK
    IF (K13.NE.2) GO TO 285
282 K23=CS(I,JP).AND.MSK(2)
    K23=K23*DV(2)
    IF (K23.NE.2) GO TO 701
283 V1(I,JP)=V1(I,J)-(U1(IP,J)-U1(I,J))*DYX
    GO TO 285
701 IF (K23.NE.1) GO TO 284
    V1(I,JP)=V1(I,J)
    GO TO 285
284 V1(I,JP)=V(I,JP)
285 K14=CS(I,JM).AND.MSK
    IF (K14.NE.2) GO TO 289
286 K24=CS(I,JM).AND.MSK(2)
    K24=K24*DV(2)
    IF (K24.NE.2) GO TO 288
287 V1(I,J)=V1(I,JP)+(U1(IP,J)-U1(I,J))*DYX
    GO TO 289
288 V1(I,J)=V(I,J)
289 CONTINUE
    DO 290 J=1,NJM1
    JP=J+1
    DO 290 I=1,NIM1
    IP=I+1
    U(IP,J)=U1(IP,J)
290 V(I,JP)=V1(I,JP)
    CALL VELSUR
    CALL BNDVEL
    RETURN
1000 FORMAT(1H ,10X,I5,22H ITERATIONS AT TIME = ,F6.3,10X4HQ = ,F10.6//
   $/)
1001 FORMAT(1H-,45H**********      TOO MANY ITERATIONS AT TIME = ,F6.3,
   $15H  --  CONMAX = ,E13.6,15H     **********)
1002 FORMAT(1H-,24H**********      CONMAX = ,E13.6,34H  AFTER 4000 ITERA
   $TIONS AT TIME = ,F6.3,15H      **********)
    END


    SUBROUTINE SG6
    COMMON/TVPOOL/XMIN,XMAX,YMIN,YMAX,TXMIN,TXMAX,TYMIN,TYMAX
    COMMON/TVGUIDE/TMODE,TEXT,ITV
    COMMON/TVFACT/FACT
    COMMON/TVTUNE/LPEN,LPEF,ITAL,IWINK,INTS,IRT,IUP
    COMMON QQ,LQQ,TNUX4,TNUY4
    COMMON DTVP,DELTAS,TVP
    COMMON MSK7,MSK8,MKC7,MKC8,IMP7,IMP8,DV7,DV8
    COMMON AGXL,AGYH,DT,DTCP,DTP,DTPP,DXC,DXCD2,DXCSQ,DYC,DYCD2,DYCSQ,
   $  DXIN,GX,GY,H,NI,NIM1,NJ,NJM1,NP,NTP,NU,T,TCP,TL,TP,TPP,W,Z,DYIN
    COMMON KD,DXY,DYX,NIP1,NJP1,TDXY,GXD,GYD,TNUX,TNUY,DXDY2,DYDX2,
   $       DXY4,DYX4,ZR,ZDXR,ZDYR,GXT,GYT,DX4,DY4,MI,MO,UP,VP,XP,YP
    COMMON CS,D,P,PS,R,U,V,XC,YC
    COMMON MSK(6),MKC(6),IMP(6),DV(6)
    DIMENSION CS(25,100),D(25,100),P(25,100),PS(3000),R(25,100),
   $          U(26,101),V(26,101),XC(25),YC(100)
    DIMENSION UP(3000),VP(3000),XP(3000),YP(3000)
    INTEGER CS,FLAG,PS
    REAL NU
    KDD=1
    DO 150 K=1,NP
    KP=PS(K).AND.MSK
    IF (KP.EQ.3) GO TO 150
100 T1=YP(K)/DYC+2.
    T2=XP(K)/DXC+2.
    KDD=2
    J=T1
    I=T2
    FY=T1-FLOAT(J)
    FX=T2-FLOAT(I)
    IM=I-1
    IP=I+1
    JM=J-1
    JP=J+1
    K1=CS(I,J).AND.MSK
    IF (K1.EQ.4) GO TO 103
    IF (K1.EQ.5) GO TO 103
101 K2=CS(I,J).AND.MSK(2)
    K2=K2*DV(2)
    IF (K2.NE.1) GO TO 150
102 K7=CS(I,J).AND.MSK7
    K7=K7*DV7
    IF (K7.EQ.1) GO TO 403
    IF (K7.EQ.2) GO TO 402
    IF (K7.EQ.3) GO TO 400
    T2=V(I,J)*DT
    DYIN=-T2
    KT=4*IMP(2)
    PS(K)=KT.OR.(PS(K).AND.MKC(2))
    GO TO 401
400 T2=V(I,J+1)*DT
    DYIN=T2
    KT=3*IMP(2)
    PS(K)=KT.OR.(PS(K).AND.MKC(2))
401 YP(K)=YP(K)+T2
    GO TO 150
402 T1=U(I,J)*DT
    DXIN=-T1
    KT=2*IMP(2)
    PS(K)=KT.OR.(PS(K).AND.MKC(2))
    GO TO 404
403 T1=U(I+1,J)*DT
    DXIN=T1
    PS(K)=IMP(2).OR.(PS(K).AND.MKC(2))
404 XP(K)=XP(K)+T1
    GO TO 150
103 K51=CS(IM,J).AND.MSK(5)
    K51=K51*DV(5)
    K52=CS(IP,J).AND.MSK(5)
    K52=K52*DV(5)
    K53=CS(I,JP).AND.MSK(5)
    K53=K53*DV(5)
    K54=CS(I,JM).AND.MSK(5)
    K54=K54*DV(5)
    IF (FY .LT. .5) 104,105
```

```
104 JPR=JM
    GO TO 106
105 JPR=J
106 JPRP=JPR+1
    HPSX=(XC(I)-XP(K))/DXC+.5
    HPSY=(YC(JPR)+DYCD2-YP(K))/DYC+.5
    HMSX=1.-HPSX
    HMSY=1.-HPSY
    IF (U(I,JPRP) .EQ. 0.) 107,108
107 U11=U(I,J)
    GO TO 111
108 IF (K51.EQ.1) GO TO 110
109 U11=0.
    GO TO 111
110 U11=U(I,JPRP)
111 IF (U(IP,JPRP) .EQ. 0.) 112,113
112 U2=U(IP,J)
    GO TO 116
113 IF (K52.EQ.1) GO TO 115
114 U2=0.
    GO TO 116
115 U2=U(IP,JPRP)
116 IF (U(I,JPR) .EQ. 0.) 117,118
117 U3=U(I,J)
    GO TO 121
118 IF (K51.EQ.1) GO TO 120
119 U3=0.
    GO TO 121
120 U3=U(I,JPR)
121 IF (U(IP,JPR) .EQ. 0.) 122,123
122 U4=U(IP,J)
    GO TO 126
123 IF (K52.EQ.1) GO TO 125
124 U4=0.
    GO TO 126
125 U4=U(IP,JPR)
126 UP(K)=HPSX*HMSY*U11+HMSX*HMSY*U2+HPSX*HPSY*U3+HMSX*HPSY*U4
    XP(K)=XP(K)+UP(K)*DT
    I9=XP(K)/DXC+2.
    IF (I9.GT.NI.OR.I9.LT.1) GO TO 200
    IF (FX .LT. .5) 127,128
127 IPR=IM
    GO TO 129
128 IPR=I
129 IPRP=IPR+1
    HPSX=(XC(IPR)+DXCD2-XP(K))/DXC+.5
    HPSY=(YC(J)-YP(K))/DYC+.5
    HMSX=1.-HPSX
    HMSY=1.-HPSY
    IF (V(IPR,JP) .EQ. 0.) 130,131
130 V11=V(I,JP)
    GO TO 134
131 IF (K53.EQ.1) GO TO 133
132 V11=0.
    GO TO 134
133 V11=V(IPR,JP)
134 IF (V(IPRP,JP) .EQ. 0.) 135,136
135 V2=V(I,JP)
    GO TO 139
136 IF (K53.EQ.1) GO TO 138
137 V2=0.
    GO TO 139
138 V2=V(IPRP,JP)
139 IF (V(IPR,J) .EQ. 0.) 140,141
140 V3=V(I,J)
    GO TO 144
141 IF (K54.EQ.1) GO TO 143
142 V3=0.
    GO TO 144
143 V3=V(IPR,J)
144 IF (V(IPRP,J) .EQ. 0.) 145,146
145 V4=V(I,J)
    GO TO 149
146 IF (K54.EQ.1) GO TO 148
147 V4=0.
    GO TO 149
148 V4=V(IPRP,J)
149 VP(K)=HPSX*HMSY*V11+HMSX*HMSY*V2+HPSX*HPSY*V3+HMSX*HPSY*V4
    YP(K)=YP(K)+VP(K)*DT
    J9=YP(K)/DYC+2.
    IF (J9.GT.NJ.OR.J9.LT.1) GO TO 200
    GO TO 150
200 PS(K)=3.OR.(PS(K).AND.MKC)
    XP(K)=0.
    YP(K)=0.
    UP(K)=0.
    VP(K)=0.
150 CONTINUE
    IF (KDD.EQ.2) GO TO 152
151 KD=2
    WRITE (MO,1000) T
152 RETURN
1000 FORMAT(1H-,48H**********      NO PARTICLES IN SYSTEM AT TIME = ,
   $  F6.3,15H      **********)
    END
```

Appendix III. SPLASH Listing

```
      PROGRAM SPLASH
      COMMON DELTAS,DT,DT2,DT4DX,DT4DY,DTCP,DTDX,DTDY,DTDXS,DTDYS,DTP,
     $       DTPP,DTVP,DXC,DXCD2,DXI4,DXIN,DXP,DYC,DYCD2,DYI4,DYIN,DYP
      COMMON EPS,G,GH,GX,GXD,GXDT,GY,GYD,GYDT,H,ICNTR,ITEST,KD,KKK,LL,
     $       MI,MO,MU1,MU2,NB1,NB2,NI,NIM1,NIP1,NJ,NJM1,NJP1,NP,NPR,NTP,
     $       ODX,ODX2,ODXS,ODY,ODY2,ODYS,R1,R2,T,TCP,TL,TP,TPP,TVP,U0,
     $       V0,W
      COMMON A(50,20),B1(50,20),B2(50,20),B3(50,20),B4(50,20),CS(50,20),
     $       DV(13),IMP(13),MKC(13),MSK(13),MU(50,20),NF(30),NK(50,20),
     $       NKT(50,20),P(50,20),PS(4500),PSI(51,21),R(50,20),SR(50,20),
     $       SRT(50,20),U(51,21),UP(4500),UT(51,21),V(51,21),VP(4500),
     $       VT(51,21),XC(50),XP(4500),XP0(30),XPL(30),YC(20),YP(4500),
     $       YP0(30),YPL(30),ZET(51,21)
      REAL MU,MU1,MU2,NK,NKT
      INTEGER CS,PS


C
C     PROGRAM SPLASH CONTAINS THE TWO-FLUID EXTENSION OF THE MAC METHOD.
C
C     THIS PROGRAM WAS WRITTEN IN JULY OF 1967 BY MICHAEL D. TERRY.
C


      NTP=11
      REWIND NTP
      MI=5
      MO=6
      KD=1
      READ (MI,1000) NB1,NB2,NI,NJ,NP,NPR
      READ (MI,1001) W,H,GX,GY,U0,V0,DT
      READ (MI,1001) MU1,MU2,R1,R2,TL,DELTAS,DTP
      READ (MI,1001) DTCP,DTPP,DTVP,DXP,DYP
      READ (MI,1002) (NF(I),XP0(I),YP0(I),XPL(I),YPL(I),I=1,NPR)
      DXC=W/(NI-2)
      DYC=H/(NJ-2)
      WRITE (MO,1003)
      WRITE (MO,1004) NB1,NB2,NI,NJ,NP,NPR
      WRITE (MO,1005)
      WRITE (MO,1006) W,H,GX,GY,U0,V0,DT
      WRITE (MO,1007)
      WRITE (MO,1006) MU1,MU2,R1,R2,TL,DELTAS,DTP
      WRITE (MO,1008)
      WRITE (MO,1006) DTCP,DTPP,DTVP,DXP,DYP,DXC,DYC
      WRITE (MO,1009)
      DO 105 I=1,NPR
  105 WRITE (MO,1010) NF(I),XP0(I),YP0(I),XPL(I),YPL(I)
      NIM1=NI-1
      NIP1=NI+1
      NJM1=NJ-1
      NJP1=NJ+1
      DELTAS=DELTAS*.5
      G=SQRT(GX*GX+GY*GY)
      GXD=GX*DXC
      GYD=GY*DYC
      ODX=1./DXC
      ODY=1./DYC
      DXCD2=DXC*.5
      DYCD2=DYC*.5
      DXI4=4.*ODX
      DYI4=4.*ODY
      GH=1./(G*H)
      DTDX=DT*ODX
      DTDY=DT*ODY
      DT2=2.*DT
      ODXS=ODX*ODX
      ODYS=ODY*ODY
      ODX2=2.*ODX
      ODY2=2.*ODY
      DTDXS=DT2*ODXS
      DTDYS=DT2*ODYS
      GXDT=GX*DT
      GYDT=GY*DT
      DT4DX=DTDX*.25
      DT4DY=DTDY*.25
      MSK(1)=7
      IMP(1)=1
      DV(1)=1
      MKC(1)=.NOT.MSK(1)
      DO 100 I=2,13
      MSK(I)=MSK(I-1)*8
      MKC(I)=.NOT.MSK(I)
      IMP(I)=IMP(I-1)*8
  100 DV(I)=DV(I-1)/8
      T=0.
      XC(1)=-DXCD2
      YC(1)=-DYCD2
      DO 101 I=2,NI
  101 XC(I)=XC(I-1)+DXC
      DO 102 J=2,NJ
  102 YC(J)=YC(J-1)+DYC
      DO 104 J=1,NJ
      DO 104 I=1,NI
      DO 103 K=3,6
  103 CS(I,J)=IMP(K).OR.(CS(I,J).AND.MKC(K))
      CS(I,J)=6.OR.(CS(I,J).AND.MKC)
      KT=5*IMP(2)
  104 CS(I,J)=KT.OR.(CS(I,J).AND.MKC(2))
      CALL CELSET
      CALL PARSET
      CALL DENVIS
      CALL PLTSET
      CALL BNDCND
  106 CALL CNTROL
      IF (KD.EQ.2) GO TO 113
      ICNTR=0
      LLL=0
      CALL QNTPRS
  107 DO 108 J=1,NJ
      DO 108 I=1,NI
  108 CS(I,J)=IMP(13).OR.(CS(I,J).AND.MKC(13))
      ITER=0
  109 ITER=ITER+1
      ICNTR=ICNTR+1
      ITEST=4
      EPS=.0008
      CALL PRSITN
      IF (KD.EQ.2) GO TO 106
      CALL PARTRA
      IF (KD.EQ.2) GO TO 106
      CALL DENCHG
      IF (LL.EQ.0) GO TO 112
      IF (LL.GT.5) GO TO 202
      IF (LL.EQ.LLL) GO TO 200
```

```
      IC=0
      GO TO 201
  200 IC=IC+1
      IF (IC.GT.10) GO TO 112
  201 LLL=LL
  202 CALL NEWBA
      IF (ITER.NE.3) GO TO 109
      DO 111 J=2,NJM1
      DO 111 I=2,NIM1
      K11=CS(I,J).AND.MSK(11)
      K11=K11*DV(11)
      IF (K11.EQ.2) GO TO 110
      K10=CS(I,J).AND.MSK(10)
      K10=K10*DV(10)
      IF (K10.NE.2) GO TO 111
      K13=CS(I,J).AND.MSK(13)
      K13=K13*DV(13)
      IF (K13.NE.1) GO TO 111
      CS(I,J)=IMP(10).OR.(CS(I,J).AND.MKC(10))
      GO TO 111
  110 K13=CS(I,J).AND.MSK(13)
      K13=K13*DV(13)
      IF (K13.NE.1) GO TO 111
      CS(I,J)=IMP(11).OR.(CS(I,J).AND.MKC(11))
  111 CONTINUE
      GO TO 107
  112 ITEST=13
      EPS=.0002
      WRITE (MO,1011)
      CALL PRSITN
      IF (KD.EQ.2) GO TO 106
      CALL VELCTS
      KKK=2
      CALL MOVPAR
      IF (KD.EQ.2) GO TO 106
      CALL DENVIS
      IF (KD.EQ.2) GO TO 106
      CALL REFCEL
      GO TO 106
  113 END FILE NTP
      REWIND NTP
      STOP
 1000 FORMAT(7I10)
 1001 FORMAT(7F10.0)
 1002 FORMAT(I10,4F10.0)
 1003 FORMAT(1H1,10X49HMAC METHOD SOLUTION OF TWO-MATERIAL FLUID PROBLEM
     $///1H-,10X5HINPUT/1H-,22X3HNB1,12X3HNB2,13X2HN1,13X2HNJ,13X2HNP,12
     $X3HNPR)
 1004 FORMAT(1H ,10X,7I15)
 1005 FORMAT(1H0,24X1HW,14X1HH,13X2HGX,13X2HGY,13X2HUO,13X2HVO,13X2HDT)
 1006 FORMAT(1H ,10X,7E15.8)
 1007 FORMAT(1H0,22X3HMU1,12X3HMU2,11X4HRHO1,11X4HRHO2,13X2HTL,9X6HDELTA
     $S,12X3HDTP)
 1008 FORMAT(1H0,21X4HDTCP,11X4HDTPP,11X4HDTVP,12X3HDXP,12X3HDYP,12X3HDX
     $C,12X3HDYC)
 1009 FORMAT(1H0,23X2HNF,12X3HXPO,12X3HYPO,12X3HXPL,12X3HYPL)
 1010 FORMAT(1H ,10X,I15,4E15.8)
 1011 FORMAT(1H0)
      END
```

```
      SUBROUTINE CELSET
      COMMON DELTAS,DT,DT2,DT4DX,DT4DY,DTCP,DTDX,DTDY,DTDXS,DTDYS,DTP,
     $       DTPP,DTVP,DXC,DXCD2,DXI4,DXIN,DXP,DYC,DYCD2,DYI4,DYIN,DYP
      COMMON EPS,G,GH,GX,GXD,GXDT,GY,GYD,GYDT,H,ICNTR,ITEST,KD,KKK,LL,
     $       MI,MO,MU1,MU2,NB1,NB2,NI,NIM1,NIP1,NJ,NJM1,NJP1,NP,NPR,NTP,
     $       ODX,ODX2,ODXS,ODY,ODY2,ODYS,R1,R2,T,TCP,TL,TP,TPP,TVP,UO,
     $       VO,W
      COMMON A(50,20),B1(50,20),B2(50,20),B3(50,20),B4(50,20),CS(50,20),
     $       DV(13),IMP(13),MKC(13),MSK(13),MU(50,20),NF(30),NK(50,20),
     $       NKT(50,20),P(50,20),PS(4500),PSI(51,21),R(50,20),SR(50,20),
     $       SRT(50,20),U(51,21),UP(4500),UT(51,21),V(51,21),VP(4500),
     $       VT(51,21),XC(50),XP(4500),XPO(30),XPL(30),YC(20),YP(4500),
     $       YPO(30),YPL(30),ZET(51,21)
      REAL MU,MU1,MU2,NK,NKT
      INTEGER CS,PS
      DIMENSION IA(100),TYPE(20),XB(22),YB(22)
      INTEGER TYPE
      PLMAX=AMAX1(XC(NI)+DXC,YC(NJ)+DYC)
      PLMIN=AMIN1(XC-DXC,YC-DYC)
      CALL SET(0.,1.,.1,1.,PLMIN,PLMAX,PLMIN,PLMAX,1)
      CALL LABMOD(6H(F6.3),1H ,6,1,1,0,0,30,0)
      CALL FLASH1(IA,100)
      CALL FRAME
      NB=NB1
  200 NBP=NB+1
      READ (MI,2000) (XB(I),YB(I),I=1,NBP)
      READ (MI,2001) (TYPE(I),I=1,NB)
      XB(NBP+1)=XB(2)
      YB(NBP+1)=YB(2)
      DO 236 M=1,NB
      IF (XB(M)-XB(M+1)) 201,215,208
  201 I=XB(M)*ODX+2.001
      J=YB(M)*ODY+1.999
  202 CS(I,J)=2.OR.(CS(I,J).AND.MKC)
      KT=TYPE(M)*IMP(2)
      CS(I,J)=KT.OR.(CS(I,J).AND.MKC(2))
      KT=2*IMP(4)
      CS(I,J+1)=KT.OR.(CS(I,J+1).AND.MKC(4))
      KT=3*IMP(7)
      CS(I,J)=KT.OR.(CS(I,J).AND.MKC(7))
      KT=3*IMP(8)
      CS(I,J+1)=KT.OR.(CS(I,J+1).AND.MKC(8))
      K2=CS(I,J).AND.MSK(2)
      K2=K2*DV(2)
      IF (K2.EQ.2) GO TO 204
  203 KT=2*IMP(6)
      CS(I,J)=KT.OR.(CS(I,J).AND.MKC(6))
  204 I=I+1
      IF (XC(I) .LE. XB(M+1)) 202,205
  205 IF (YB(M+2)-YB(M+1)) 206,236,207
  206 KT=2*IMP(5)
      CS(I-1,J)=KT.OR.(CS(I-1,J).AND.MKC(5))
      GO TO 236
  207 CS(I,J)=1.OR.(CS(I,J).AND.MKC)
      KT=3*IMP(7)
      CS(I,J)=KT.OR.(CS(I,J).AND.MKC(7))
      GO TO 231
  208 I=XB(M)*ODX+1.999
      J=YB(M)*ODY+2.001
  209 CS(I,J)=2.OR.(CS(I,J).AND.MKC)
      KT=TYPE(M)*IMP(2)
```

```
      CS(I,J)=KT.OR.(CS(I,J).AND.MKC(2))
      KT=2*IMP(4)
      CS(I,J-1)=KT.OR.(CS(I,J-1).AND.MKC(4))
      KT=4*IMP(7)
      CS(I,J)=KT.OR.(CS(I,J).AND.MKC(7))
      KT=4*IMP(8)
      CS(I,J-1)=KT.OR.(CS(I,J-1).AND.MKC(8))
      K2=CS(I,J).AND.MSK(2)
      K2=K2*DV(2)
      IF (K2.EQ.2) GO TO 211
  210 KT=2*IMP(6)
      CS(I,J-1)=KT.OR.(CS(I,J-1).AND.MKC(6))
  211 I=I-1
      IF (XC(I).GE.XB(M+1)) 209,212
  212 IF (YB(M+2)-YB(M+1)) 214,236,213
  213 KT=2*IMP(5)
      CS(I+1,J)=KT.OR.(CS(I+1,J).AND.MKC(5))
      KT=2*IMP(6)
      CS(I,J-1)=KT.OR.(CS(I,J-1).AND.MKC(6))
      GO TO 236
  214 CS(I,J)=1.OR.(CS(I,J).AND.MKC)
      KT=4*IMP(7)
      CS(I,J)=KT.OR.(CS(I,J).AND.MKC(7))
      GO TO 231
  215 IF (YB(M).LT.YB(M+1)) 216,223
  216 I=XB(M)*ODX+2.001
      J=YB(M)*ODY+2.001
  217 CS(I,J)=2.OR.(CS(I,J).AND.MKC)
      KT=TYPE(M)*IMP(2)
      CS(I,J)=KT.OR.(CS(I,J).AND.MKC(2))
      KT=2*IMP(4)
      CS(I-1,J)=KT.OR.(CS(I-1,J).AND.MKC(4))
      KT=2*IMP(7)
      CS(I,J)=KT.OR.(CS(I,J).AND.MKC(7))
      KT=2*IMP(8)
      CS(I-1,J)=KT.OR.(CS(I-1,J).AND.MKC(8))
      K2=CS(I,J).AND.MSK(2)
      K2=K2*DV(2)
      IF (K2.EQ.2) GO TO 219
  218 KT=2*IMP(6)
      CS(I-1,J)=KT.OR.(CS(I-1,J).AND.MKC(6))
  219 J=J+1
      IF (YC(J).LE.YB(M+1)) 217,220
  220 IF (XB(M+2)-XB(M+1)) 222,236,221
  221 KT=2*IMP(5)
      CS(I,J-1)=KT.OR.(CS(I,J-1).AND.MKC(5))
      GO TO 236
  222 CS(I,J)=1.OR.(CS(I,J).AND.MKC)
      KT=2*IMP(7)
      CS(I,J)=KT.OR.(CS(I,J).AND.MKC(7))
      GO TO 231
  223 I=XB(M)*ODX+1.999
      J=YB(M)*ODY+1.999
  224 CS(I,J)=2.OR.(CS(I,J).AND.MKC)
      KT=TYPE(M)*IMP(2)
      CS(I,J1=KT.OR.(CS(I,J).AND.MKC(2))
      KT=2*IMP(4)
      CS(I+1,J)=KT.OR.(CS(I+1,J).AND.MKC(4))
      CS(I,J)=IMP(7).OR.(CS(I,J).AND.MKC(7))
      CS(I+1,J)=IMP(8).OR.(CS(I+1,J).AND.MKC(8))
      K2=CS(I,J).AND.MSK(2)
      K2=K2*DV(2)
      IF(K2.EQ.2) GO TO 226
  225 KT=2*IMP(6)
      CS(I,J)=KT.OR.(CS(I,J).AND.MKC(6))
  226 J=J-1
      IF (YC(J).GE.YB(M+1)) 224,227
  227 IF (M.LT.NB) 228,236
  228 IF (XB(M+2)-XB(M+1)) 229,236,230
  229 KT=2*IMP(5)
      CS(I,J+1)=KT.OR.(CS(I,J+1).AND.MKC(5))
      GO TO 236
  230 CS(I,J)=1.OR.(CS(I,J).AND.MKC)
      CS(I,J)=IMP(7).OR.(CS(I,J).AND.MKC(7))
      KT=2*IMP(6)
      CS(I,J)=KT.OR.(CS(I,J).AND.MKC(6))
  231 IF (M.LT.NB) 232,233
  232 KMP1=M+1
      GO TO 234
  233 KMP1=1
  234 IF (TYPE(M).EQ.3.OR.TYPE(KMP1).EQ.3) 235,236
  235 KT=3*IMP(2)
      CS(I,J)=KT.OR.(CS(I,J).AND.MKC(2))
  236 CONTINUE
      IF (NB2.NE.0) 237,238
  237 NB=NB2
      NB2=0
      CALL CURVE(XB,YB,NBP)
      GO TO 200
  238 CALL CURVE(XB,YB,NBP)
      CALL FLASH2(1,LENGTH)
      WRITE (NTP) LENGTH,(IA(I),I=1,LENGTH)
      WRITE (NTP) NIM1,NJM1,PLMIN,PLMAX
      KODE=1
      DO 264 J=1,NJ
      DO 264 I=1,NI
      K1=CS(I,J).AND.MSK
      K2=CS(I,J).AND.MSK(2)
      K2=K2*DV(2)
      K4=CS(I,J).AND.MSK(4)
      K4=K4*DV(4)
      IF (K2.NE.1) GO TO 258
  239 IF (I.EQ.1) 240,243
  240 L=1
  241 L1=1
      K2A=CS(I+1,J-1).AND.MSK(2)
      K2A=K2A*DV(2)
      IF (K2A.EQ.4) GO TO 255
  242 K2B=CS(I+1,J+1).AND.MSK(2)
      K2B=K2B*DV(2)
      GO TO 253
  243 IF (I.EQ.NI) 244,247
  244 L=1
  245 L1=2
      K2C=CS(I-1,J-1).AND.MSK(2)
      K2C=K2C*DV(2)
      IF (K2C.EQ.4) GO TO 255
  246 K2B=CS(I-1,J+1).AND.MSK(2)
      K2B=K2B*DV(2)
      GO TO 253
  247 IF(J.EQ.1) 248,250
  248 L=1
```

```
      K2D=CS(I-1,J+1).AND.MSK(2)
      K2D=K2D*DV(2)
      IF (K2D.EQ.4) GO TO 255
  249 K2B=CS(I+1,J+1).AND.MKC(2)
      K2B=K2B*DV(2)
      GO TO 253
  250 IF (J.EQ.NJ) 251,254
  251 L=1
      K2E=CS(I-1,J-1).AND.MSK(2)
      K2E=K2E*DV(2)
      IF (K2E.EQ.4) GO TO 255
  252 K2B=CS(I+1,J-1).AND.MSK(2)
      K2B=K2B*DV(2)
  253 IF (K2B.EQ.4) GO TO 255
      GO TO 256
  254 L1=2
      GO TO 241
  255 KT=2*IMP(3)
      CS(I,J)=KT.OR.(CS(I,J).AND.MKC(3))
  256 IF (L.EQ.1) GO TO 258
  257 IF (L1.EQ.1) GO TO 245
  258 IF (KODE.EQ.2) GO TO 262
  259 IF (K4.EQ.2) GO TO 262
  260 KODE=1
      IF (K1.EQ.2) GO TO 264
  261 CS(I,J)=1.OR.(CS(I,J).AND.MKC)
      GO TO 264
  262 KODE=1
      IF (K1.EQ.2) GO TO 264
  263 KODE=2
      CS(I,J)=3.OR.(CS(I,J).AND.MKC)
  264 CONTINUE
      KT=2*IMP(6)
      CS=KT.OR.(CS.AND.MKC(6))
      RETURN
 2000 FORMAT(7F10.0)
 2001 FORMAT(7I10)
      END


      SUBROUTINE PARSET
      COMMON DELTAS,DT,DT2,DT4DX,DT4DY,DTCP,DTDX,DTDY,DTDXS,DTDYS,DTP,
     $       DTPP,DTVP,DXC,DXCD2,DXI4,DXIN,DXP,DYC,DYCD2,DYI4,DYIN,DYP
      COMMON EPS,G,GH,GX,GXD,GXDT,GY,GYD,GYDT,H,ICNTR,ITEST,KD,KKK,LL,
     $       MI,MO,MU1,MU2,NB1,NB2,NI,NIM1,NIP1,NJ,NJM1,NJP1,NP,NPR,NTP,
     $       ODX,ODX2,ODXS,ODY,ODY2,ODYS,R1,R2,T,TCP,TL,TP,TPP,TVP,U0,
     $       VC,W
      COMMON A(50,20),B1(50,20),B2(50,20),B3(50,20),B4(50,20),CS(50,20),
     $       DV(13),IMP(13),MKC(13),MSK(13),MU(50,20),NF(30),NK(50,20),
     $       NKT(50,20),P(50,20),PS(4500),PSI(51,21),R(50,20),SR(50,20),
     $       SRT(50,20),U(51,21),UP(4500),UT(51,21),V(51,21),VP(4500),
     $       VT(51,21),XC(50),XP(4500),XPO(30),XPL(30),YC(20),YP(4500),
     $       YPO(30),YPL(30),ZET(51,21)
      REAL MU,MU1,MU2,NK,NKT
      INTEGER CS,PS
      DO 300 K=1,NP
  300 PS(K)=3.OR.(PS(K).AND.MKC)
      K=1
      DO 301 J=1,NJ
      DO 301 I=1,NI
      KT=4*IMP(9)
  301 CS(I,J)=KT.OR.(CS(I,J).AND.MKC(9))
      DO 315 II=1,NPR
      Y=YPO(II)
  302 X=XPO(II)
  303 I=X*ODX+2.
      J=Y*ODY+2.
      K1=CS(I,J).AND.MSK
      K2=CS(I,J).AND.MSK(2)
      K2=K2*DV(2)
      IF (K1.EQ.1) GO TO 308
      IF (K1.EQ.2) GO TO 310
  304 CS(I,J)=4.OR.(CS(I,J).AND.MKC)
      PS(K)=1.OR.(PS(K).AND.MKC)
      KT=NF(II)*IMP(3)
      PS(K)=KT.OR.(PS(K).AND.MKC(3))
      K9=CS(I,J).AND.MSK(9)
      K9=K9*DV(9)
      IF (K9.NE.4) GO TO 305
      KT=NF(II)*IMP(9)
      CS(I,J)=KT.OR.(CS(I,J).AND.MKC(9))
      GO TO 306
  305 IF (NF(II).EQ.K9) GO TO 306
      KT=3*IMP(9)
      CS(I,J)=KT.OR.(CS(I,J).AND.MKC(9))
  306 U(I,J)=U0
      V(I,J)=V0
      U(I+1,J)=U0
      V(I,J+1)=V0
  307 XP(K)=X
      YP(K)=Y
      K=K+1
  308 X=X+DXP
      IF (X.GT.XPL(II)) 309,303
  309 Y=Y+DYP
      IF (Y.GT.YPL(II)) 315,302
  310 IF (K2.NE.1) GO TO 308
  311 PS(K)=2.OR.(PS(K).AND.MKC)
      K7=CS(I,J).AND.MSK(7)
      K7=K7*DV(7)
      IF (K7.EQ.1) GO TO 312
      IF(K7.EQ.2) GO TO 313
      IF (K7.EQ.3) GO TO 314
      V(I,J)=V0
      GO TO 307
  312 U(I+1,J)=U0
      GO TO 307
  313 U(I,J)=U0
      GO TO 307
  314 V(I,J+1)=V0
      GO TO 307
  315 CONTINUE
      DO 316 J=1,NJ
      DO 316 I=1,NI
      CS(I,J)=IMP(10).OR.(CS(I,J).AND.MKC(10))
  316 CS(I,J)=IMP(11).OR.(CS(I,J).AND.MKC(11))
      CALL FLGCEL
      RETURN
      END


      SUBROUTINE FLGCEL
```

```
      COMMON DELTAS,DT,DT2,DT4DX,DT4DY,DTCP,DTDX,DTDY,DTDXS,DTDYS,DTP,
     $       DTPP,DTVP,DXC,DXCD2,DXI4,DXIN,DXP,DYC,DYCD2,DYI4,DYIN,DYP
      COMMON EPS,G,GH,GX,GXD,GXDT,GY,GYD,GYDT,H,ICNTR,ITEST,KD,KKK,LL,
     $       MI,MO,MU1,MU2,NB1,NB2,NI,NIM1,NIP1,NJ,NJM1,NJP1,NP,NPR,NTP,
     $       ODX,ODX2,ODXS,ODY,ODY2,ODYS,R1,R2,T,TCP,TL,TP,TPP,TVP,U0,
     $       V0,W
      COMMON A(50,20),B1(50,20),B2(50,20),B3(50,20),B4(50,20),CS(50,20),
     $       DV(13),IMP(13),MKC(13),MSK(13),MJ(50,20),NF(30),NK(50,20),
     $       NKT(50,20),P(50,20),PS(4500),PSI(51,21),R(50,20),SR(50,20),
     $       SRT(50,20),U(51,21),UP(4500),UT(51,21),V(51,21),VP(4500),
     $       VT(51,21),XC(50),XP(4500),XP0(30),XPL(30),YC(20),YP(4500),
     $       YP0(30),YPL(30),ZET(51,21)
      REAL MU,MU1,MU2,NK,NKT
      INTEGER CS,PS
      DO 818 J=2,NJM1
      JM1=J-1
      JP1=J+1
      DO 818 I=2,NIM1
      IM1=I-1
      IP1=I+1
      K1=CS(I,J).AND.MSK
      IF (K1.EQ.4) GO TO 800
      IF (K1.NE.5) GO TO 818
  800 K9=CS(I,J).AND.MSK(9)
      K9=K9*DV(9)
      IF (K9.NE.3) GO TO 815
    1 KT1=2*IMP(11)
      KT2=2*IMP(10)
      CS(I,J)=KT1.OR.(CS(I,J).AND.MKC(11))
      CS(I,JP1)=KT1.OR.(CS(I,JP1).AND.MKC(11))
      CS(IP1,JP1)=KT1.OR.(CS(IP1,JP1).AND.MKC(11))
      CS(IP1,J)=KT1.OR.(CS(IP1,J).AND.MKC(11))
      CS(IP1,JM1)=KT1.OR.(CS(IP1,JM1).AND.MKC(11))
      CS(I,JM1)=KT1.OR.(CS(I,JM1).AND.MKC(11))
      CS(IM1,JM1)=KT1.OR.(CS(IM1,JM1).AND.MKC(11))
      CS(IM1,J)=KT1.OR.(CS(IM1,J).AND.MKC(11))
      CS(IM1,JP1)=KT1.OR.(CS(IM1,JP1).AND.MKC(11))
      CS(I,J)=KT2.OR.(CS(I,J).AND.MKC(10))
      CS(I,JP1)=KT2.OR.(CS(I,JP1).AND.MKC(10))
      CS(IP1,JP1)=KT2.OR.(CS(IP1,JP1).AND.MKC(10))
      CS(IP1,J)=KT2.OR.(CS(IP1,J).AND.MKC(10))
      CS(IP1,JM1)=KT2.OR.(CS(IP1,JM1).AND.MKC(10))
      CS(I,JM1)=KT2.OR.(CS(I,JM1).AND.MKC(10))
      CS(IM1,JM1)=KT2.OR.(CS(IM1,JM1).AND.MKC(10))
      CS(IM1,J)=KT2.OR.(CS(IM1,J).AND.MKC(10))
      CS(IM1,JP1)=KT2.OR.(CS(IM1,JP1).AND.MKC(10))
      ML=1
      MR=1
      MB=1
      MT=1
      K1A=CS(IM1,J).AND.MSK
      IF (K1A.EQ.2) ML=2
      K1B=CS(IP1,J).AND.MSK
      IF (K1B.EQ.2) MR=2
      K1C=CS(I,JM1).AND.MSK
      IF (K1C.EQ.2) MB=2
      K1D=CS(I,JP1).AND.MSK
      IF (K1D.EQ.2) MT=2
      IF (ML+MR+MB+MT.NE.4) GO TO 801
      JM2=J-2
      JP2=J+2
      IM2=I-2
      IP2=I+2
    2 KT2=2*IMP(10)
      CS(I,JP2)=KT2.OR.(CS(I,JP2).AND.MKC(10))
      CS(IP1,JP2)=KT2.OR.(CS(IP1,JP2).AND.MKC(10))
      CS(IP2,JP2)=KT2.OR.(CS(IP2,JP2).AND.MKC(10))
      CS(IP2,JP1)=KT2.OR.(CS(IP2,JP1).AND.MKC(10))
      CS(IP2,J)=KT2.OR.(CS(IP2,J).AND.MKC(10))
      CS(IP2,JM1)=KT2.OR.(CS(IP2,JM1).AND.MKC(10))
      CS(IP2,JM2)=KT2.OR.(CS(IP2,JM2).AND.MKC(10))
      CS(IP1,JM2)=KT2.OR.(CS(IP1,JM2).AND.MKC(10))
      CS(I,JM2)=KT2.OR.(CS(I,JM2).AND.MKC(10))
      CS(IM1,JM2)=KT2.OR.(CS(IM1,JM2).AND.MKC(10))
      CS(IM2,JM2)=KT2.OR.(CS(IM2,JM2).AND.MKC(10))
      CS(IM2,JM1)=KT2.OR.(CS(IM2,JM1).AND.MKC(10))
      CS(IM2,J)=KT2.OR.(CS(IM2,J).AND.MKC(10))
      CS(IM2,JP1)=KT2.OR.(CS(IM2,JP1).AND.MKC(10))
      CS(IM2,JP2)=KT2.OR.(CS(IM2,JP2).AND.MKC(10))
      CS(IM1,JP2)=KT2.OR.(CS(IM1,JP2).AND.MKC(10))
      GO TO 818
  801 IF (ML+MR+MB.NE.6) GO TO 802
      JP2=J+2
    3 KT2=2*IMP(10)
      CS(I,JP2)=KT2.OR.(CS(I,JP2).AND.MKC(10))
      CS(IP1,JP2)=KT2.OR.(CS(IP1,JP2).AND.MKC(10))
      CS(IM1,JP2)=KT2.OR.(CS(IM1,JP2).AND.MKC(10))
      GO TO 818
  802 IF (ML+MR+MT.NE.6) GO TO 803
      JM2=J-2
    4 KT2=2*IMP(10)
      CS(I,JM2)=KT2.OR.(CS(I,JM2).AND.MKC(10))
      CS(IP1,JM2)=KT2.OR.(CS(IP1,JM2).AND.MKC(10))
      CS(IM1,JM2)=KT2.OR.(CS(IM1,JM2).AND.MKC(10))
      GO TO 818
  803 IF (ML+MB+MT.NE.6) GO TO 804
      IP2=I+2
    5 KT2=2*IMP(10)
      CS(IP2,JM1)=KT2.OR.(CS(IP2,JM1).AND.MKC(10))
      CS(IP2,J)=KT2.OR.(CS(IP2,J).AND.MKC(10))
      CS(IP2,JP1)=KT2.OR.(CS(IP2,JP1).AND.MKC(10))
      GO TO 818
  804 IF (MR+MB+MT.NE.6) GO TO 805
      IM2=I-2
    6 KT2=2*IMP(10)
      CS(IM2,JM1)=KT2.OR.(CS(IM2,JM1).AND.MKC(10))
      CS(IM2,J)=KT2.OR.(CS(IM2,J).AND.MKC(10))
      CS(IM2,JP1)=KT2.OR.(CS(IM2,JP1).AND.MKC(10))
      GO TO 818
  805 IF (ML+MR.NE.4) GO TO 806
      JM2=J-2
      JP2=J+2
    7 KT2=2*IMP(10)
      CS(IM1,JP2)=KT2.OR.(CS(IM1,JP2).AND.MKC(10))
      CS(I,JP2)=KT2.OR.(CS(I,JP2).AND.MKC(10))
      CS(IP1,JP2)=KT2.OR.(CS(IP1,JP2).AND.MKC(10))
      CS(IM1,JM2)=KT2.OR.(CS(IM1,JM2).AND.MKC(10))
      CS(I,JM2)=KT2.OR.(CS(I,JM2).AND.MKC(10))
      CS(IP1,JM2)=KT2.OR.(CS(IP1,JM2).AND.MKC(10))
      GO TO 818
  806 IF (ML+MB.NE.4) GO TO 807
      JP2=J+2
```

```
      IP2=I+2
    8 KT2=2*IMP(10)
      CS(IM1,JP2)=KT2.OR.(CS(IM1,JP2).AND.MKC(10))
      CS(I,JP2)=KT2.OR.(CS(I,JP2).AND.MKC(10))
      CS(IP1,JP2)=KT2.OR.(CS(IP1,JP2).AND.MKC(10))
      CS(IP2,JP2)=KT2.OR.(CS(IP2,JP2).AND.MKC(10))
      CS(IP2,JP1)=KT2.OR.(CS(IP2,JP1).AND.MKC(10))
      CS(IP2,J)=KT2.OR.(CS(IP2,J).AND.MKC(10))
      CS(IP2,JM1)=KT2.OR.(CS(IP2,JM1).AND.MKC(10))
      GO TO 818
  807 IF (ML+MT.NE.4) GO TO 808
      JM2=J-2
      IP2=I+2
    9 KT2=2*IMP(10)
      CS(IP2,JP1)=KT2.OR.(CS(IP2,JP1).AND.MKC(10))
      CS(IP2,J)=KT2.OR.(CS(IP2,J).AND.MKC(10))
      CS(IP2,JM1)=KT2.OR.(CS(IP2,JM1).AND.MKC(10))
      CS(IP2,JM2)=KT2.OR.(CS(IP2,JM2).AND.MKC(10))
      CS(IP1,JM2)=KT2.OR.(CS(IP1,JM2).AND.MKC(10))
      CS(I,JM2)=KT2.OR.(CS(I,JM2).AND.MKC(10))
      CS(IM1,JM2)=KT2.OR.(CS(IM1,JM2).AND.MKC(10))
      GO TO 818
  808 IF (MR+MB.NE.4) GO TO 809
      JP2=J+2
      IM2=I-2
   10 KT2=2*IMP(10)
      CS(IM2,JM1)=KT2.OR.(CS(IM2,JM1).AND.MKC(10))
      CS(IM2,J)=KT2.OR.(CS(IM2,J).AND.MKC(10))
      CS(IM2,JP1)=KT2.OR.(CS(IM2,JP1).AND.MKC(10))
      CS(IM2,JP1)=KT2.OR.(CS(IM2,JP1).AND.MKC(10))
      CS(IM1,JP2)=KT2.OR.(CS(IM1,JP2).AND.MKC(10))
      CS(I,JP2)=KT2.OR.(CS(I,JP2).AND.MKC(10))
      CS(IP1,JP2)=KT2.OR.(CS(IP1,JP2).AND.MKC(10))
      GO TO 818
  809 IF (MR+MT.NE.4) GO TO 810
      JM2=J-2
      IM2=I-2
   11 KT2=2*IMP(10)
      CS(IP1,JM2)=KT2.OR.(CS(IP1,JM2).AND.MKC(10))
      CS(I,JM2)=KT2.OR.(CS(I,JM2).AND.MKC(10))
      CS(IM1,JM1)=KT2.OR.(CS(IM1,JM1).AND.MKC(10))
      CS(IM2,JM2)=KT2.OR.(CS(IM2,JM2).AND.MKC(10))
      CS(IM2,JM1)=KT2.OR.(CS(IM2,JM1).AND.MKC(10))
      CS(IM2,J)=KT2.OR.(CS(IM2,J).AND.MKC(10))
      CS(IM2,JP1)=KT2.OR.(CS(IM2,JP1).AND.MKC(10))
      GO TO 818
  810 IF (MB+MT.NE.4) GO TO 811
      IM2=I-2
      IP2=I+2
   12 KT2=2*IMP(10)
      CS(IM2,JP1)=KT2.OR.(CS(IM2,JP1).AND.MKC(10))
      CS(IM2,J)=KT2.OR.(CS(IM2,J).AND.MKC(10))
      CS(IM2,JM1)=KT2.OR.(CS(IM2,JM1).AND.MKC(10))
      CS(IP2,JP1)=KT2.OR.(CS(IP1,JP1).AND.MKC(10))
      CS(IP2,J)=KT2.OR.(CS(IP2,J).AND.MKC(10))
      CS(IP2,JM1)=KT2.OR.(CS(IP2,JM1).AND.MKC(10))
      GO TO 818
  811 IF (ML.NE.2) GO TO 812
      JM2=J-2
      JP2=J+2
      IP2=I+2
   13 KT2=2*IMP(10)
      CS(IM1,JP2)=KT2.OR.(CS(IM1,JP2).AND.MKC(10))
      CS(I,JP2)=KT2.OR.(CS(I,JP2).AND.MKC(10))
      CS(IP1,JP2)=KT2.OR.(CS(IP1,JP2).AND.MKC(10))
      CS(IP2,JP2)=KT2.OR.(CS(IP2,JP2).AND.MKC(10))
      CS(IP2,JP1)=KT2.OR.(CS(IP2,JP1).AND.MKC(10))
      CS(IP2,J)=KT2.OR.(CS(IP2,J).AND.MKC(10))
      CS(IP2,JM1)=KT2.OR.(CS(IP2,JM1).AND.MKC(10))
      CS(IP2,JM2)=KT2.OR.(CS(IP2,JM2).AND.MKC(10))
      CS(IP1,JM2)=KT2.OR.(CS(IP1,JM2).AND.MKC(10))
      CS(I,JM2)=KT2.OR.(CS(I,JM2).AND.MKC(10))
      CS(IM1,JM2)=KT2.OR.(CS(IM1,JM2).AND.MKC(10))
      GO TO 818
  812 IF (MR.NE.2) GO TO 813
      JM2=J-2
      JP2=J+2
      IM2=I-2
   14 KT2=2*IMP(10)
      CS(IP1,JM2)=KT2.OR.(CS(IP1,JM2).AND.MKC(10))
      CS(I,JM2)=KT2.OR.(CS(I,JM2).AND.MKC(10))
      CS(IM1,JM2)=KT2.OR.(CS(IM1,JM2).AND.MKC(10))
      CS(IM2,JM2)=KT2.OR.(CS(IM2,JM2).AND.MKC(10))
      CS(IM2,JM1)=KT2.OR.(CS(IM2,JM1).AND.MKC(10))
      CS(IM2,J)=KT2.OR.(CS(IM2,J).AND.MKC(10))
      CS(IM2,JP1)=KT2.OR.(CS(IM2,JP1).AND.MKC(10))
      CS(IM2,JP2)=KT2.OR.(CS(IM2,JP2).AND.MKC(10))
      CS(IM1,JP2)=KT2.OR.(CS(IM1,JP2).AND.MKC(10))
      CS(I,JP2)=KT2.OR.(CS(I,JP2).AND.MKC(10))
      CS(IP1,JP2)=KT2.OR.(CS(IP1,JP2).AND.MKC(10))
      GO TO 818
  813 IF (MB.NE.2) GO TO 814
      JP2=J+2
      IM2=I-2
      IP2=I+2
   15 KT2=2*IMP(10)
      CS(IM2,JM1)=KT2.OR.(CS(IM2,JM1).AND.MKC(10))
      CS(IM2,J)=KT2.OR.(CS(IM2,J).AND.MKC(10))
      CS(IM2,JP1)=KT2.OR.(CS(IM2,JP1).AND.MKC(10))
      CS(IM2,JP2)=KT2.OR.(CS(IM2,JP2).AND.MKC(10))
      CS(IM1,JP2)=KT2.OR.(CS(IM1,JP2).AND.MKC(10))
      CS(I,JP2)=KT2.OR.(CS(I,JP2).AND.MKC(10))
      CS(IP1,JP2)=KT2.OR.(CS(IP1,JP2).AND.MKC(10))
      CS(IP2,JP2)=KT2.OR.(CS(IP2,JP2).AND.MKC(10))
      CS(IP2,JP1)=KT2.OR.(CS(IP2,JP1).AND.MKC(10))
      CS(IP2,J)=KT2.OR.(CS(IP2,J).AND.MKC(10))
      CS(IP2,JM1)=KT2.OR.(CS(IP2,JM1).AND.MKC(10))
      GO TO 818
  814 IF (MT.NE.2) GO TO 815
      JM2=J-2
      IM2=I-2
      IP2=I+2
   16 KT2=2*IMP(10)
      CS(IP2,JP1)=KT2.OR.(CS(IP2,JP1).AND.MKC(10))
      CS(IP2,J)=KT2.OR.(CS(IP2,J).AND.MKC(10))
      CS(IP2,JM1)=KT2.OR.(CS(IP2,JM1).AND.MKC(10))
      CS(IP2,JM2)=KT2.OR.(CS(IP2,JM2).AND.MKC(10))
      CS(IP1,JM2)=KT2.OR.(CS(IP1,JM2).AND.MKC(10))
      CS(I,JM2)=KT2.OR.(CS(I,JM2).AND.MKC(10))
      CS(IM1,JM2)=KT2.OR.(CS(IM1,JM2).AND.MKC(10))
      CS(IM2,JM2)=KT2.OR.(CS(IM2,JM2).AND.MKC(10))
      CS(IM2,JM1)=KT2.OR.(CS(IM2,JM1).AND.MKC(10))
```

```
      CS(IM2,J)=KT2.OR.(CS(IM2,J).AND.MKC(10))
      CS(IM2,JP1)=KT2.OR.(CS(IM2,JP1).AND.MKC(10))
      GO TO 818
  815 K9A=CS(I,JP1).AND.MSK(9)
      K9A=K9A*DV(9)
      IF (K9.EQ.K9A) GO TO 816
      K1A=CS(I,JP1).AND.MSK
      IF (K1A.EQ.2) GO TO 818
      JP2=J+2
   17 KT1=2*IMP(11)
      CS(I,J)=KT1.OR.(CS(I,J).AND.MKC(11))
      CS(I,JP1)=KT1.OR.(CS(I,JP1).AND.MKC(11))
      KT2=2*IMP(10)
      CS(I,J)=KT2.OR.(CS(I,J).AND.MKC(10))
      CS(I,JP1)=KT2.DR.(CS(I,JP1).AND.MKC(10))
      CS(I,JP2)=KT2.OR.(CS(I,JP2).AND.MKC(10))
      CS(IP1,JP2)=KT2.OR.(CS(IP1,JP2).AND.MKC(10))
      CS(IP1,JP1)=KT2.OR.(CS(IP1,JP1).AND.MKC(10))
      CS(IP1,J)=KT2.OR.(CS(IP1,J).AND.MKC(10))
      CS(IP1,JM1)=KT2.OR.(CS(IP1,JM1).AND.MKC(10))
      CS(I,JM1)=KT2.OR.(CS(I,JM1).AND.MKC(10))
      CS(IM1,J)=KT2.OR.(CS(IM1,J).AND.MKC(10))
      CS(IM1,JM1)=KT2.OR.(CS(IM1,JM1).AND.MKC(10))
      CS(IM1,JP1)=KT2.OR.(CS(IM1,JP1).AND.MKC(10))
      CS(IM1,JP2)=KT2.OR.(CS(IM1,JP2).AND.MKC(10))
  816 K9B=CS(IP1,JP1).AND.MSK(9)
      K9B=K9B*DV(9)
      IF (K9.EQ.K9B) GO TO 817
      K1B=CS(IP1,JP1).AND.MSK
      IF (K1B.EQ.1.OR.K1B.EQ.2) GO TO 818
      JP2=J+2
      IP2=I+2
   18 KT1=2*IMP(11)
      CS(I,J)=KT1.OR.(CS(I,J).AND.MKC(11))
      CS(IP1,JP1)=KT1.OR.(CS(IP1,JP1).AND.MKC(11))
      KT2=2*IMP(10)
      CS(I,J)=KT2.OR.(CS(I,J).AND.MKC(10))
      CS(IP1,JP1)=KT2.OR.(CS(IP1,JP1).AND.MKC(10))
      CS(IP1,JP2)=KT2.OR.(CS(IP1,JP2).AND.MKC(10))
      CS(IP2,JP2)=KT2.OR.(CS(IP2,JP2).AND.MKC(10))
      CS(IP2,JP1)=KT2.OR.(CS(IP2,JP1).AND.MKC(10))
      CS(IP2,J)=KT2.OR.(CS(IP2,J).AND.MKC(10))
      CS(IP1,J)=KT2.OR.(CS(IP1,J).AND.MKC(10))
      CS(IP1,JM1)=KT2.OR.(CS(IP1,JM1).AND.MKC(10))
      CS(I,JM1)=KT2.OR.(CS(I,JM1).AND.MKC(10))
      CS(IM1,JM1)=KT2.OR.(CS(IM1,JM1).AND.MKC(10))
      CS(IM1,J)=KT2.OR.(CS(IM1,J).AND.MKC(10))
      CS(IM1,JP1)=KT2.OR.(CS(IM1,JP1).AND.MKC(10))
      CS(I,JP1)=KT2.OR.(CS(I,JP1).AND.MKC(10))
      CS(I,JP2)=KT2.OR.(CS(I,JP2).AND.MKC(10))
  817 K9C=CS(IP1,J).AND.MSK(9)
      K9C=K9C*DV(9)
      IF (K9.EQ.K9C) GO TO 818
      K1C=CS(IP1,J).AND.MSK
      IF (K1C.EQ.2) GO TO 818
      IP2=I+2
   19 KT1=2*IMP(11)
      CS(I,J)=KT1.OR.(CS(I,J).AND.MKC(11))
      CS(IP1,J)=KT1.OR.(CS(IP1,J).AND.MKC(11))
      KT2=2*IMP(10)
      CS(I,J)=KT2.OR.(CS(I,J).AND.MKC(10))
      CS(IP1,J)=KT2.OR.(CS(IP1,J).AND.MKC(10))
      CS(IP2,J)=KT2.OR.(CS(IP2,J).AND.MKC(10))
      CS(IP2,JM1)=KT2.OR.(CS(IP2,JM1).AND.MKC(10))
      CS(IP1,JM1)=KT2.OR.(CS(IP1,JM1).AND.MKC(10))
      CS(I,JM1)=KT2.OR.(CS(I,JM1).AND.MKC(10))
      CS(IM1,JM1)=KT2.OR.(CS(IM1,JM1).AND.MKC(10))
      CS(IM1,J)=KT2.OR.(CS(IM1,J).AND.MKC(10))
      CS(IM1,JP1)=KT2.OR.(CS(IM1,JP1).AND.MKC(10))
      CS(I,JP1)=KT2.OR.(CS(I,JP1).AND.MKC(10))
      CS(IP1,JP1)=KT2.OR.(CS(IP1,JP1).AND.MKC(10))
      CS(IP2,JP1)=KT2.OR.(CS(IP2,JP1).AND.MKC(10))
  818 CONTINUE
      RETURN
      END


      SUBROUTINE PLTSET
      COMMON DELTAS,DT,DT2,DT4DX,DT4DY,DTCP,DTDX,DTDY,DTDXS,DTDYS,DTP,
     $       DTPP,DTVP,DXC,DXCD2,DXI4,DXIN,DXP,DYC,DYCD2,DYI4,DYIN,DYP
      COMMON EPS,G,GH,GX,GXD,GXDT,GY,GYD,GYDT,H,ICNTR,ITEST,KD,KKK,LL,
     $       MI,MO,MU1,MU2,NB1,NB2,NI,NIM1,NIP1,NJ,NJM1,NJP1,NP,NPR,NTP,
     $       ODX,ODX2,ODXS,ODY,ODY2,ODYS,R1,R2,T,TCP,TL,TP,TPP,TVP,UO,
     $       VO,W
      COMMON A(50,20),B1(50,20),B2(50,20),B3(50,20),B4(50,20),C3(50,20),
     $       DV(13),IMP(13),MKC(13),MSK(13),MU(50,20),NF(30),NK(50,20),
     $       NKT(50,20),P(50,20),PS(4500),PSI(51,21),R(50,20),SR(50,20),
     $       SRT(50,20),U(51,21),UP(4500),UT(51,21),V(51,21),VP(4500),
     $       VT(51,21),XC(50),XP(4500),XP0(30),XPL(30),YC(20),YP(4500),
     $       YP0(30),YPL(30),ZET(51,21)
      REAL MU,MU1,MU2,NK,NKT
      INTEGER CS,PS
      CALL FRAME
      DO 412 J=1,NJ
      DO 412 I=1,NI
      K1=CS(I,J).AND.MSK
      K2=CS(I,J).AND.MSK(2)
      K2=K2*DV(2)
      K4=CS(I,J).AND.MSK(4)
      K4=K4*DV(4)
      K5=CS(I,J).AND.MSK(5)
      K5=K5*DV(5)
      K6=CS(I,J).AND.MSK(6)
      K6=K6*DV(6)
      GO TO (412,400,406,405,406,412),K1
  400 GO TO (401,402,403,404,412),K2
  401 CALL PSYM(XC(I),YC(J),1H+,0,0,1)
      GO TO 408
  402 CALL PSYM(XC(I),YC(J),1H-,0,0,1)
      GO TO 408
  403 CALL PSYM(XC(I),YC(J),1H*,0,0,1)
      GO TO 408
  404 CALL PSYM(XC(I),YC(J),1HX,0,1,1)
      GO TO 408
  405 CALL PSYM(XC(I),YC(J),1H/,0,0,1)
  406 IF (K4.EQ.1) GO TO 408
  407 CALL PSYM(XC(I),YC(J),1HO,0,0,1)
  408 IF (K5.EQ.1) GO TO 410
  409 CALL PSYM(XC(I),YC(J),1HQ,0,0,1)
  410 IF (K6.EQ.1) GO TO 412
  411 CALL PSYM(XC(I)+DXCD2,YC(J)+DYCD2,1H.,0,0,1)
  412 CONTINUE
```

```
      CALL FLUSH
      RETURN
      END


      SUBROUTINE BNDCND
      COMMON DELTAS,DT,DT2,DT4DX,DT4DY,DTCP,DTDX,DTDY,DTDXS,DTDYS,DTP,
     $       DTPP,DTVP,DXC,DXCD2,DXI4,DXIN,DXP,DYC,DYCD2,DYI4,DYIN,DYP
      COMMON EPS,G,GH,GX,GXD,GXDT,GY,GYD,GYDT,H,ICNTR,ITEST,KD,KKK,LL,
     $       MI,MO,MU1,MU2,NB1,NB2,NI,NIM1,NIP1,NJ,NJM1,NJP1,NP,NPR,NTP,
     $       ODX,ODX2,ODXS,ODY,ODY2,ODYS,R1,R2,T,TCP,TL,TP,TPP,TVP,U0,
     $       V0,W
      COMMON A(50,20),B1(50,20),B2(50,20),B3(50,20),B4(50,20),CS(50,20),
     $       DV(13),IMP(13),MKC(13),MSK(13),MU(50,20),NF(30),NK(50,20),
     $       NKT(50,20),P(50,20),PS(4500),PSI(51,21),R(50,20),SR(50,20),
     $       SRT(50,20),U(51,21),UP(4500),UT(51,21),V(51,21),VP(4500),
     $       VT(51,21),XC(50),XP(4500),XP0(30),XPL(30),YC(20),YP(4500),
     $       YP0(30),YPL(30),ZET(51,21)
      REAL MU,MU1,MU2,NK,NKT
      INTEGER CS,PS
      DO 511 J=1,NJ
      DO 511 I=1,NI
      K1=CS(I,J).AND.MSK
      IF (K1.NE.2) GO TO 511
      K2=CS(I,J).AND.MSK(2)
      K2=K2*DV(2)
      K7=CS(I,J).AND.MSK(7)
      K7=K7*DV(7)
      IF (K7.NE.1) GO TO 502
      IF (K2.NE.4) GO TO 500
      U(I,J)=U(I+2,J)
      V(I,J+1)=-V(I+1,J+1)
      V(I,J)=-V(I+1,J)
      GO TO 501
  500 IF (K2.NE.3) GO TO 511
      U(I,J)=-U(I+2,J)
      V(I,J+1)=V(I+1,J+1)
      V(I,J)=V(I+1,J)
  501 U(I+1,J)=0.
      GO TO 511
  502 IF (K7.NE.2) GO TO 505
      IF (K2.NE.4) GO TO 503
      U(I+1,J)=U(I-1,J)
      V(I,J+1)=-V(I-1,J+1)
      V(I,J)=-V(I-1,J)
      GO TO 504
  503 IF (K2.NE.3) GO TO 511
      U(I+1,J)=-U(I-1,J)
      V(I,J+1)=V(I-1,J+1)
      V(I,J)=V(I-1,J)
  504 U(I,J)=0.
      GO TO 511
  505 IF (K7.NE.3) GO TO 508
      IF (K2.NE.4) GO TO 506
      V(I,J)=V(I,J+2)
      U(I+1,J)=-U(I+1,J+1)
      U(I,J)=-U(I,J+1)
      GO TO 507
  506 IF (K2.NE.3) GO TO 511
      V(I,J)=-V(I,J+2)
      U(I+1,J)=U(I+1,J+1)
      U(I,J)=U(I,J+1)
  507 V(I,J+1)=0.
      GO TO 511
  508 IF (K2.NE.4) GO TO 509
      V(I,J+1)=V(I,J-1)
      U(I+1,J)=-U(I+1,J-1)
      U(I,J)=-U(I,J-1)
      GO TO 510
  509 IF (K2.NE.3) GO TO 511
      V(I,J+1)=-V(I,J-1)
      U(I+1,J)=U(I+1,J-1)
      U(I,J)=U(I,J-1)
  510 V(I,J)=0.
  511 CONTINUE
      RETURN
      END


      SUBROUTINE CNTROL
      COMMON DELTAS,DT,DT2,DT4DX,DT4DY,DTCP,DTDX,DTDY,DTDXS,DTDYS,DTP,
     $       DTPP,DTVP,DXC,DXCD2,DXI4,DXIN,DXP,DYC,DYCD2,DYI4,DYIN,DYP
      COMMON EPS,G,GH,GX,GXD,GXDT,GY,GYD,GYDT,H,ICNTR,ITEST,KD,KKK,LL,
     $       MI,MO,MU1,MU2,NB1,NB2,NI,NIM1,NIP1,NJ,NJM1,NJP1,NP,NPR,NTP,
     $       ODX,ODX2,ODXS,ODY,ODY2,ODYS,R1,R2,T,TCP,TL,TP,TPP,TVP,U0,
     $       V0,W
      COMMON A(50,20),B1(50,20),B2(50,20),B3(50,20),B4(50,20),CS(50,20),
     $       DV(13),IMP(13),MKC(13),MSK(13),MU(50,20),NF(30),NK(50,20),
     $       NKT(50,20),P(50,20),PS(4500),PSI(51,21),R(50,20),SR(50,20),
     $       SRT(50,20),U(51,21),UP(4500),UT(51,21),V(51,21),VP(4500),
     $       VT(51,21),XC(50),XP(4500),XP0(30),XPL(30),YC(20),YP(4500),
     $       YP0(30),YPL(30),ZET(51,21)
      REAL MU,MU1,MU2,NK,NKT
      INTEGER CS,PS
      IF (T.NE.0.) GO TO 600
      CALL PLTPAR
      CALL CELPRT
      CALL PARPRT
      CALL PLTVEL
      TP=DTP
      TCP=DTCP
      TPP=DTPP
      TVP=DTVP
      GO TO 605
  600 IF (KD.EQ.2) GO TO 606
      IF (SENSE SWITCH 6) 596,598
  596 FARG=SAVEF(1)
      IF (FARG.EQ.0.) GO TO 598
      IS=NN+2
      DO 597 II=1,IS
  597 READ (NTP)
  598 CONTINUE
      IF (T.LT.TP-.00000001) GO TO 601
      CALL PLTPAR
      TP=T+DTP
  601 IF (T.LT.TCP-.00000001) GO TO 602
      CALL CELPRT
      TCP=T+DTCP
  602 IF (T.LT.TPP-.00000001) GO TO 603
      CALL PARPRT
      TPP=T+DTPP
  603 IF (T.LT.TVP-.00000001) GO TO 604
```

```
      CALL PLTVEL
      TVP=T+DTVP
  604 IF (T.GE.TL-.00000001) KD=2
  605 T=T+DT
      RETURN
  606 CALL CELPRT
      CALL PARPRT
      WRITE (MO,6000)
      GO TO 605
 6000 FORMAT(1H1,10X,41HABNORMAL STOP -- LOOK FOR ANOTHER MESSAGE)
      END


      SUBROUTINE PLTPAR
      COMMON DELTAS,DT,DT2,DT4DX,DT4DY,DTCP,DTDX,DTDY,DTDXS,DTDYS,DTP,
     $       DTPP,DTVP,DXC,DXCD2,DXI4,DXIN,DXP,DYC,DYCD2,DYI4,DYIN,DYP
      COMMON EPS,G,GH,GX,GXD,GXDT,GY,GYD,GYDT,H,ICNTR,ITEST,KD,KKK,LL,
     $       MI,MO,MU1,MU2,NB1,NB2,NI,NIM1,NIP1,NJ,NJM1,NJP1,NP,NPR,NTP,
     $       ODX,ODX2,ODXS,ODY,ODY2,ODYS,R1,R2,T,TCP,TL,TP,TPP,TVP,U0,
     $       V0,W
      COMMON A(50,20),B1(50,20),B2(50,20),B3(50,20),B4(50,20),CS(50,20),
     $       DV(13),IMP(13),MKC(13),MSK(13),MU(50,20),NF(30),NK(50,20),
     $       NKT(50,20),P(50,20),PS(4500),PSI(51,21),R(50,20),SR(50,20),
     $       SRT(50,20),U(51,21),UP(4500),UT(51,21),V(51,21),VP(4500),
     $       VT(51,21),XC(50),XP(4500),XP0(30),XPL(30),YC(20),YP(4500),
     $       YP0(30),YPL(30),ZET(51,21)
      REAL MU,MU1,MU2,NK,NKT
      INTEGER CS,PS
      CALL FLASH3(1)
      ENCODE (10,7000,AA) T
      CALL PWRT(824,72,AA,10,1,0)
      DO 702 K=1,NP
      KP=PS(K).AND.MSK
      IF (KP.EQ.3) GO TO 702
      I=XP(K)*ODX+2.
      J=YP(K)*ODY+2.
      K1=CS(I,J).AND.MSK
      IF (K1.EQ.2) GO TO 702
      IF (K1.EQ.1) GO TO 702
      KPA=PS(K).AND.MSK(3)
      KPA=KPA*DV(3)
      IF (KPA.EQ.2) GO TO 700
      CALL OPTION(0,0,0,0)
      CALL POINT(XP(K),YP(K))
      GO TO 702
  700 CALL OPTION(0,1,0,0)
      DO 701 L=1,5
  701 CALL POINT(XP(K),YP(K))
  702 CONTINUE
      CALL OPTION(0,0,0,0)
      CALL FLUSH
      RETURN
 7000 FORMAT(4HT = ,F6.3)
      END


      SUBROUTINE CELPRT
      COMMON DELTAS,DT,DT2,DT4DX,DT4DY,DTCP,DTDX,DTDY,DTDXS,DTDYS,DTP,
     $       DTPP,DTVP,DXC,DXCD2,DXI4,DXIN,DXP,DYC,DYCD2,DYI4,DYIN,DYP
      COMMON EPS,G,GH,GX,GXD,GXDT,GY,GYD,GYDT,H,ICNTR,ITEST,KD,KKK,LL,
     $       MI,MO,MU1,MU2,NB1,NB2,NI,NIM1,NIP1,NJ,NJM1,NJP1,NP,NPR,NTP,
     $       ODX,ODX2,ODXS,ODY,ODY2,ODYS,R1,R2,T,TCP,TL,TP,TPP,TVP,U0,
     $       V0,W
      COMMON A(50,20),B1(50,20),B2(50,20),B3(50,20),B4(50,20),CS(50,20),
     $       DV(13),IMP(13),MKC(13),MSK(13),MU(50,20),NF(30),NK(50,20),
     $       NKT(50,20),P(50,20),PS(4500),PSI(51,21),R(50,20),SR(50,20),
     $       SRT(50,20),U(51,21),UP(4500),UT(51,21),V(51,21),VP(4500),
     $       VT(51,21),XC(50),XP(4500),XP0(30),XPL(30),YC(20),YP(4500),
     $       YP0(30),YPL(30),ZET(51,21)
      REAL MU,MU1,MU2,NK,NKT
      INTEGER CS,PS
      LINE=0
      WRITE (MO,8000) T
      WRITE (MO,8001)
      DO 800 J=1,NJ
      DO 800 I=1,NI
      U1=(U(I+1,J)+U(I,J))*.5
      V1=(V(I,J+1)+V(I,J))*.5
      WRITE (MO,8002) I,J,U1,V1,P(I,J),R(I,J),MU(I,J),CS(I,J)
      LINE=LINE+1
      IF (LINE.NE.50) GO TO 800
      LINE=0
      WRITE (MO,8003)
      WRITE (MO,8001)
  800 CONTINUE
      WRITE (MO,8004) T
      RETURN
 8000 FORMAT(1H1,10X22HCELL PRINT FOR TIME = ,F6.3///)
 8001 FORMAT(1H ,5X1HI,3X1HJ,14X1HU,14X1HV,14X1HP,12X3HRHO,13X2HMU,13X2H
     $CS/)
 8002 FORMAT(1H ,4X,I2,I4,5E15.6,2X,O15)
 8003 FORMAT(1H1)
 8004 FORMAT(1HO,20X36H****  END OF CELL PRINT FOR TIME = ,F6.3,7H  ***
     $**/1H1)
      END


      SUBROUTINE PARPRT
      COMMON DELTAS,DT,DT2,DT4DX,DT4DY,DTCP,DTDX,DTDY,DTDXS,DTDYS,DTP,
     $       DTPP,DTVP,DXC,DXCD2,DXI4,DXIN,DXP,DYC,DYCD2,DYI4,DYIN,DYP
      COMMON EPS,G,GH,GX,GXD,GXDT,GY,GYD,GYDT,H,ICNTR,ITEST,KD,KKK,LL,
     $       MI,MO,MU1,MU2,NB1,NB2,NI,NIM1,NIP1,NJ,NJM1,NJP1,NP,NPR,NTP,
     $       ODX,ODX2,ODXS,ODY,ODY2,ODYS,R1,R2,T,TCP,TL,TP,TPP,TVP,U0,
     $       V0,W
      COMMON A(50,20),B1(50,20),B2(50,20),B3(50,20),B4(50,20),CS(50,20),
     $       DV(13),IMP(13),MKC(13),MSK(13),MU(50,20),NF(30),NK(50,20),
     $       NKT(50,20),P(50,20),PS(4500),PSI(51,21),R(50,20),SR(50,20),
     $       SRT(50,20),U(51,21),UP(4500),UT(51,21),V(51,21),VP(4500),
     $       VT(51,21),XC(50),XP(4500),XP0(30),XPL(30),YC(20),YP(4500),
     $       YP0(30),YPL(30),ZET(51,21)
      REAL MU,MU1,MU2,NK,NKT
      INTEGER CS,PS
      LINE=0
      WRITE (MO,9000) T
      WRITE (MO,9001)
      DO 900 K=1,NP
      KP=PS(K).AND.MSK
      IF (KP.EQ.3) GO TO 900
      WRITE (MO,9002) K,XP(K),YP(K),UP(K),VP(K),PS(K)
      LINE=LINE+1
      IF (LINE.NE.50) GO TO 900
      LINE=0
```

```
      WRITE (MO,9003)
      WRITE (MO,9001)
  900 CONTINUE
      WRITE (MO,9004) T
      RETURN
 9000 FORMAT(1H1,10X26HPARTICLE PRINT FOR TIME = ,F6.3///)
 9001 FORMAT(1H ,9X1HK,14X1HX,14X1HY,14X1HU,14X1HV,5X2HPS)
 9002 FORMAT(1H ,I10,4E15.6,2X,O5)
 9003 FORMAT(1H1)
 9004 FORMAT(1HO,20X40H*****  END OF PARTICLE PRINT FOR TIME = ,F6.3,7H
     $ *****/1H1)
      END


      SUBROUTINE PLTVEL
      COMMON DELTAS,DT,DT2,DT4DX,DT4DY,DTCP,DTDX,DTDY,DTDXS,DTDYS,DTP,
     $       DTPP,DTVP,DXC,DXCD2,DXI4,DXIN,DXP,DYC,DYCD2,DYI4,DYIN,DYP
      COMMON EPS,G,GH,GX,GXD,GXDT,GY,GYD,GYDT,H,ICNTR,ITEST,KD,KKK,LL,
     $       MI,MO,MU1,MU2,NB1,NB2,NI,NIM1,NIP1,NJ,NJM1,NJP1,NP,NPR,NTP,
     $       ODX,ODX2,ODXS,ODY,ODY2,ODYS,R1,R2,T,TCP,TL,TP,TPP,TVP,U0,
     $       V0,W
      COMMON A(50,20),B1(50,20),B2(50,20),B3(50,20),B4(50,20),CS(50,20),
     $       DV(13),IMP(13),MKC(13),MSK(13),MU(50,20),NF(30),NK(50,20),
     $       NKT(50,20),P(50,20),PS(4500),PSI(51,21),R(50,20),SR(50,20),
     $       SRT(50,20),U(51,21),UP(4500),UT(51,21),V(51,21),VP(4500),
     $       VT(51,21),XC(50),XP(4500),XPO(30),XPL(30),YC(20),YP(4500),
     $       YPO(30),YPL(30),ZET(51,21)
      REAL MU,MU1,MU2,NK,NKT
      INTEGER CS,PS
      NN=NN+1
      WRITE (NTP) T
      WRITE (MO,1000) NN
      DO 102 J=2,NJM1
      DO 102 I=2,NIM1
      K1=CS(I,J).AND.MSK
      IF (K1.EQ.4) GO TO 100
      IF (K1.NE.5) GO TO 101
  100 X1=XC(I)
      Y1=YC(J)
      X2=X1+DELTAS*(U(I+1,J)+U(I,J))
      Y2=Y1+DELTAS*(V(I,J+1)+V(I,J))
      GO TO 102
  101 X1=7777777777777777777B
  102 WRITE (NTP) X1,Y1,X2,Y2
      RETURN
 1000 FORMAT(1H-,18H//////////    NN = ,I4,13H    //////////,///)
      END


      SUBROUTINE QNTPRS
      COMMON DELTAS,DT,DT2,DT4DX,DT4DY,DTCP,DTDX,DTDY,DTDXS,DTDYS,DTP,
     $       DTPP,DTVP,DXC,DXCD2,DXI4,DXIN,DXP,DYC,DYCD2,DYI4,DYIN,DYP
      COMMON EPS,G,GH,GX,GXD,GXDT,GY,GYD,GYDT,H,ICNTR,ITEST,KD,KKK,LL,
     $       MI,MO,MU1,MU2,NB1,NB2,NI,NIM1,NIP1,NJ,NJM1,NJP1,NP,NPR,NTP,
     $       ODX,ODX2,ODXS,ODY,ODY2,ODYS,R1,R2,T,TCP,TL,TP,TPP,TVP,U0,
     $       V0,W
      COMMON A(50,20),B1(50,20),B2(50,20),B3(50,20),B4(50,20),CS(50,20),
     $       DV(13),IMP(13),MKC(13),MSK(13),MU(50,20),NF(30),NK(50,20),
     $       NKT(50,20),P(50,20),PS(4500),PSI(51,21),R(50,20),SR(50,20),
     $       SRT(50,20),U(51,21),UP(4500),UT(51,21),V(51,21),VP(4500),
     $       VT(51,21),XC(50),XP(4500),XPO(30),XPL(30),YC(20),YP(4500),
     $       YPO(30),YPL(30),ZET(51,21)
      REAL MU,MU1,MU2,NK,NKT
      INTEGER CS,PS
      DO 517 J=2,NJM1
      JM=J-1
      JP=J+1
      DO 517 I=2,NIM1
      IM=I-1
      IP=I+1
      K1=CS(I,J).AND.MSK
      IF (K1.EQ.4) GO TO 500
      IF (K1.NE.5) GO TO 517
  500 VAVE1=.5*(V(I,J)+V(IP,J))
      VAVE2=.5*(V(I,JP)+V(IP,JP))
      UAVE1=.5*(U(I,J)+U(I,JP))
      UAVE2=.5*(U(IP,J)+U(IP,JP))
      AR1=.5*(R(I,J)+R(IP,J))
      AR2=.5*(R(I,J)+R(I,JP))
      IF (VAVE1) 501,502,503
  501 RV1=AR1*U(IP,J)*VAVE1
      GO TO 504
  502 RV1=0.
      GO TO 504
  503 RV1=.5*(R(I,JM)+R(IP,JM))*U(IP,JM)*VAVE1
  504 IF (VAVE2) 505,506,507
  505 RV2=.5*(R(I,JP)+R(IP,JP))*U(IP,JP)*VAVE2
      GO TO 508
  506 RV2=0.
      GO TO 508
  507 RV2=AR1*U(IP,J)*VAVE2
  508 IF (UAVE1) 509,510,511
  509 RU1=AR2*V(I,JP)*UAVE1
      GO TO 512
  510 RU1=0.
      GO TO 512
  511 RU1=.5*(R(IM,J)+R(IM,JP))*V(IM,JP)*UAVE1
  512 IF (UAVE2) 513,514,515
  513 RU2=.5*(R(IP,J)+R(IP,JP))*V(IP,JP)*UAVE2
      GO TO 516
  514 RU2=0.
      GO TO 516
  515 RU2=AR2*V(I,JP)*UAVE2
  516 U1=U(I,J)+U(IP,J)
      U2=U(IP,J)+U(I+2,J)
      V1=V(I,J)+V(I,JP)
      V2=V(I,JP)+V(I,J+2)
      AM1=MU(I,J)+MU(I,JP)+MU(IP,JP)+MU(IP,J)
      PSI(IP,J)=AR1*U(IP,J)+DT4DX*(R(I,J)*U1*U1-R(IP,J)*U2*U2)+DTDY*(RV1
     $          -RV2)+DTDXS*(MU(IP,J)*(U(I+2,J)-U(IP,J))-MU(I,J)*(U(IP,J
     $          )-U(I,J)))+DT4DY*(AM1*(ODY*(U(IP,JP)-U(IP,J))+ODX*(V(IP,
     $          JP)-V(I,JP)))-(MU(I,JM)+MU(I,J)+MU(IP,J)+MU(IP,JM))*(ODY
     $          *(U(IP,J)-U(IP,JM))+ODX*(V(IP,J)-V(I,J))))+AR1*GXDT
      ZET(I,JP)=AR2*V(I,JP)+DTDX*(RU1-RU2)+DT4DY*(R(I,J)*V1*V1-R(I,JP)*V
     $          2*V2)+DTDYS*(MU(I,JP)*(V(I,J+2)-V(I,JP))-MU(I,J)*(V(I,JP
     $          )-V(I,J)))+DT4DX*(AM1*(ODY*(U(IP,JP)-U(IP,J))+ODX*(V(IP,
     $          JP)-V(I,JP)))-(MU(IM,J)+MU(IM,JP)+MU(I,JP)+MU(I,J))*(ODY
     $          *(U(I,JP)-U(I,J))+ODX*(V(I,JP)-V(IM,JP))))+AR2*GYDT
  517 CONTINUE
      DO 519 J=2,NJM1
      JM=J-1
      JP=J+1
```

```
      DO 519 I=2,NIM1
      IM=I-1
      IP=I+1
      K1=CS(I,J).AND.MSK
      IF (K1.EQ.4) GO TO 518
      IF (K1.NE.5) GO TO 519
  518 AR1=1./(R(I,J)+R(IP,J))
      AR2=1./(R(I,J)+R(IM,J))
      AR3=1./(R(I,J)+R(I,JP))
      AR4=1./(R(I,J)+R(I,JM))
      CIJ=1./(DT2*(ODXS*(AR1+AR2)+ODYS*(AR3+AR4)))
      B1(I,J)=CIJ*DTDXS*AR1
      B2(I,J)=CIJ*DTDXS*AR2
      B3(I,J)=CIJ*DTDYS*AR3
      B4(I,J)=CIJ*DTDYS*AR4
      A(I,J)=CIJ*(ODX2*(PSI(I,J)*AR2-PSI(IP,J)*AR1)+ODY2*(ZET(I,J)*AR4-
     $     ZET(I,JP)*AR3))
  519 CONTINUE
      RETURN
      END


      SUBROUTINE PRSITN
      COMMON DELTAS,DT,DT2,DT4DX,DT4DY,DTCP,DTDX,DTDY,DTDXS,DTDYS,DTP,
     $     DTPP,DTVP,DXC,DXCD2,DXI4,DXIN,DXP,DYC,DYCD2,DYI4,DYIN,DYP
      COMMON EPS,G,GH,GX,GXD,GXDT,GY,GYD,GYDT,H,ICNTR,ITEST,KD,KKK,LL,
     $     MI,MO,MU1,MU2,NB1,NB2,NI,NIM1,NIP1,NJ,NJM1,NJP1,NP,NPR,NTP,
     $     ODX,ODX2,ODXS,ODY,ODY2,ODYS,R1,R2,T,TCP,TL,TP,TPP,TVP,U0,
     $     V0,W
      COMMON A(50,20),B1(50,20),B2(50,20),B3(50,20),B4(50,20),CS(50,20),
     $     DV(13),IMP(13),MKC(13),MSK(13),MU(50,20),NF(30),NK(50,20),
     $     NKT(50,20),P(50,20),PS(4500),PSI(51,21),R(50,20),SR(50,20),
     $     SRT(50,20),U(51,21),UP(4500),UT(51,21),V(51,21),VP(4500),
     $     VT(51,21),XC(50),XP(4500),XP0(30),XPL(30),YC(20),YP(4500),
     $     YP0(30),YPL(30),ZET(51,21)
      REAL MU,MU1,MU2,NK,NKT
      INTEGER CS,PS
      DIMENSION PN(50,20)
      EQUIVALENCE (PN,SRT)
      DO 200 J=1,NJ
      DO 200 I=1,NI
  200 PN(I,J)=P(I,J)
      ICNTP=0
  201 ERR=0.
      DO 213 L=1,ITEST
      ICNTP=ICNTP+1
      DO 211 J=2,NJM1
      JM=J-1
      JP=J+1
      DO 211 I=2,NIM1
      IM=I-1
      IP=I+1
      K1=CS(I,J).AND.MSK
      IF (K1.EQ.4) GO TO 202
      IF (K1.NE.5) GO TO 211
  202 K1A=CS(IM,J).AND.MSK
      IF (K1A.NE.2) GO TO 204
      K2A=CS(IM,J).AND.MSK(2)
      K2A=K2A*DV(2)
      IF (K2A.NE.3) GO TO 203
      PN(IM,J)=PN(I,J)-R(I,J)*GXD
      GO TO 204
  203 IF (K2A.NE.4) GO TO 204
      PN(IM,J)=PN(I,J)-R(I,J)*GXD-DXI4*MU(I,J)*U(IP,J)-ODY*(V(I,JP)*
     $          (MU(I,J)+MU(I,JP))-V(I,J)*(MU(I,J)+MU(I,JM)))
  204 K1B=CS(IP,J).AND.MSK
      IF (K1B.NE.2) GO TO 206
      K2B=CS(IP,J).AND.MSK(2)
      K2B=K2B*DV(2)
      IF (K2B.NE.3) GO TO 205
      PN(IP,J)=PN(I,J)+R(I,J)*GXD
      GO TO 206
  205 IF (K2B.NE.4) GO TO 206
      PN(IP,J)=PN(I,J)+R(I,J)*GXD+DXI4*MU(I,J)*U(I,J)-ODY*(V(I,JP)*
     $          (MU(I,J)+MU(I,JP))-V(I,J)*(MU(I,J)+MU(I,JM)))
  206 K1C=CS(I,JM).AND.MSK
      IF (K1C.NE.2) GO TO 208
      K2C=CS(I,JM).AND.MSK(2)
      K2C=K2C*DV(2)
      IF (K2C.NE.3) GO TO 207
      PN(I,JM)=PN(I,J)-R(I,J)*GYD
      GO TO 208
  207 IF (K2C.NE.4) GO TO 208
      PN(I,JM)=PN(I,J)-R(I,J)*GYD-DYI4*MU(I,J)*V(I,JP)-ODX*(U(IP,J)*
     $          (MU(I,J)+MU(IP,J))-U(I,J)*(MU(I,J)+MU(IM,J)))
  208 K1D=CS(IP,J).AND.MSK
      IF (K1D.NE.2) GO TO 210
      K2D=CS(I,JP).AND.MSK(2)
      K2D=K2D*DV(2)
      IF (K2D.NE.3) GO TO 209
      PN(I,JP)=PN(I,J)+R(I,J)*GYD
      GO TO 210
  209 IF (K2D.NE.4) GO TO 210
      PN(I,JP)=PN(I,J)+R(I,J)*GYD+DYI4*MU(I,J)*V(I,J)-ODX*(U(IP,J)*
     $          (MU(I,J)+MU(IP,J))-U(I,J)*(MU(I,J)+MU(IM,J)))
  210 PN(I,J)=B1(I,J)*PN(IP,J)+B2(I,J)*PN(IM,J)+B3(I,J)*PN(I,JP)+B4(I,J)
     $         *PN(I,JM)+A(I,J)
      IF (L.EQ.ITEST) ERR=AMAX1(ERR,ABS(PN(I,J)-P(I,J))*GH/ABS(R(I,J)))
  211 CONTINUE
      DO 212 J=1,NJ
      DO 212 I=1,NI
  212 P(I,J)=PN(I,J)
  213 CONTINUE
      IF (ERR.LT.EPS) GO TO 214
      IF (ICNTP.LT.300*ITEST) GO TO 201
      KD=2
      WRITE (MO,2000) T,ICNTR,LL
      GO TO 215
  214 WRITE (MO,2001) ICNTP,T,ICNTR,LL
  215 RETURN
 2000 FORMAT(1H-/1H-,20X30HTOO MANY ITERATIONS AT TIME = ,F6.3,10X8HICNT
     $R = ,I4,10X5HLL = ,I4)
 2001 FORMAT(1H ,10X,I4,23H  ITERATIONS AT TIME = ,F6.3,10X8HICNTR = ,I4
     $,10X5HLL = ,I4)
      END


      SUBROUTINE PARTRA
      COMMON DELTAS,DT,DT2,DT4DX,DT4DY,DTCP,DTDX,DTDY,DTDXS,DTDYS,DTP,
     $     DTPP,DTVP,DXC,DXCD2,DXI4,DXIN,DXP,DYC,DYCD2,DYI4,DYIN,DYP
      COMMON EPS,G,GH,GX,GXD,GXDT,GY,GYD,GYDT,H,ICNTR,ITEST,KD,KKK,LL,
     $     MI,MO,MU1,MU2,NB1,NB2,NI,NIM1,NIP1,NJ,NJM1,NJP1,NP,NPR,NTP,
```

```
     $        ODX,ODX2,ODXS,ODY,ODY2,ODYS,R1,R2,T,TCP,TL,TP,TPP,TVP,UQ,
     $        VQ,W
      COMMON A(50,20),B1(50,20),B2(50,20),B3(50,20),B4(50,20),CS(50,20),
     $        DV(13),IMP(13),MKC(13),MSK(13),MU(50,20),NF(30),NK(50,20),
     $        NKT(50,20),P(50,20),PS(4500),PSI(51,21),R(50,20),SR(50,20),
     $        SRT(50,20),U(51,21),UP(4500),UT(51,21),V(51,21),VP(4500),
     $        VT(51,21),XC(50),XP(4500),XP0(30),XPL(30),YC(20),YP(4500),
     $        YP0(30),YPL(30),ZET(51,21)
      REAL MU,MU1,MU2,NK,NKT
      INTEGER CS,PS
      DO 900 J=1,NJ
      DO 900 I=1,NI
      NKT(I,J)=0.
  900 SRT(I,J)=0.
      DO 902 J=2,NJM1
      JM=J-1
      JP=J+1
      DO 902 I=2,NIM1
      IM=I-1
      IP=I+1
      K1=CS(I,J).AND.MSK
      IF (K1.EQ.4) GO TO 901
      IF (K1.NE.5) GO TO 902
  901 K10=CS(I,J).AND.MSK(10)
      K10=K10*DV(10)
      IF (K10.EQ.1) GO TO 902
      UT(I,JP)=2.*(PSI(I,JP)+DTDX*(P(IM,JP)-P(I,JP)))/(R(IM,JP)+R(I,JP))
      UT(I,J)=2.*(PSI(I,J)+DTDX*(P(IM,J)-P(I,J)))/(R(IM,J)+R(I,J))
      UT(I,JM)=2.*(PSI(I,JM)+DTDX*(P(IM,JM)-P(I,JM)))/(R(IM,JM)+R(I,JM))
      UT(IP,JP)=2.*(PSI(IP,JP)+DTDX*(P(I,JP)-P(IP,JP)))/(R(I,JP)+R(IP,JP
     $           ))
      UT(IP,J)=2.*(PSI(IP,J)+DTDX*(P(I,J)-P(IP,J)))/(R(I,J)+R(IP,J))
      UT(IP,JM)=2.*(PSI(IP,JM)+DTDX*(P(I,JM)-P(IP,JM)))/(R(I,JM)+R(IP,JM
     $           ))
      VT(IP,J)=2.*(ZET(IP,J)+DTDY*(P(IP,JM)-P(IP,J)))/(R(IP,JM)+R(IP,J))
      VT(I,J)=2.*(ZET(I,J)+DTDY*(P(I,JM)-P(I,J)))/(R(I,JM)+R(I,J))
      VT(IM,J)=2.*(ZET(IM,J)+DTDY*(P(IM,JM)-P(IM,J)))/(R(IM,JM)+R(IM,J))
      VT(IP,JP)=2.*(ZET(IP,JP)+DTDY*(P(IP,J)-P(IP,JP)))/(R(IP,J)+R(IP,JP
     $           ))
      VT(I,JP)=2.*(ZET(I,JP)+DTDY*(P(I,J)-P(I,JP)))/(R(I,J)+R(I,JP))
      VT(IM,JP)=2.*(ZET(IM,JP)+DTDY*(P(IM,J)-P(IM,JP)))/(R(IM,J)+R(IM,JP
     $           ))
      CALL TBDCND
  902 CONTINUE
      KKK=1
      CALL MOVPAR
      RETURN
      END


      SUBROUTINE TBDCND
      COMMON DELTAS,DT,DT2,DT4DX,DT4DY,DTCP,DTDX,DTDY,DTDXS,DTDYS,DTP,
     $        DTPP,DTVP,DXC,DXCD2,DXI4,DXIN,DXP,DYC,DYCD2,DYI4,DYIN,DYP
      COMMON EPS,G,GH,GX,GXD,GXDT,GY,GYD,GYDT,H,ICNTR,ITEST,KD,KKK,LL,
     $        MI,MO,MU1,MU2,NB1,NB2,NI,NIM1,NIP1,NJ,NJM1,NJP1,NP,NPR,NTP,
     $        ODX,ODX2,ODXS,ODY,ODY2,ODYS,R1,R2,T,TCP,TL,TPP,TVP,UQ,
     $        VQ,W
      COMMON A(50,20),B1(50,20),B2(50,20),B3(50,20),B4(50,20),CS(50,20),
     $        DV(13),IMP(13),MKC(13),MSK(13),MU(50,20),NF(30),NK(50,20),
     $        NKT(50,20),P(50,20),PS(4500),PSI(51,21),R(50,20),SR(50,20),
     $        SRT(50,20),U(51,21),UP(4500),UT(51,21),V(51,21),VP(4500),
     $        VT(51,21),XC(50),XP(4500),XP0(30),XPL(30),YC(20),YP(4500),
     $        YP0(30),YPL(30),ZET(51,21)
      REAL MU,MU1,MU2,NK,NKT
      INTEGER CS,PS
      DO 811 J=1,NJ
      JM=J-1
      JP=J+1
      DO 811 I=1,NI
      IM=I-1
      IP=I+1
      K1=CS(I,J).AND.MSK
      IF (K1.NE.2) GO TO 811
      K2=CS(I,J).AND.MSK(2)
      K2=K2*DV(2)
      K7=CS(I,J).AND.MSK(7)
      K7=K7*DV(7)
      IF (K7.EQ.4) GO TO 808
      IF (K7.EQ.3) GO TO 805
      IF (K7.EQ.2) GO TO 802
      IF (K2.NE.4) GO TO 800
      S=1.
      GO TO 801
  800 IF (K2.NE.3) GO TO 811
      S=-1.
  801 UT(I,J)=S*UT(I+2,J)
      UT(IP,J)=0.
      VT(I,J)=-S*VT(IP,J)
      VT(I,JP)=-S*VT(IP,JP)
      GO TO 811
  802 IF (K2.NE.4) GO TO 803
      S=1.
      GO TO 804
  803 IF (K2.NE.3) GO TO 811
      S=-1.
  804 UT(I,J)=0.
      UT(IP,J)=S*UT(IM,J)
      VT(I,J)=-S*VT(IM,J)
      VT(I,JP)=-S*VT(IM,JP)
      GO TO 811
  805 IF (K2.NE.4) GO TO 806
      S=-1.
      GO TO 807
  806 IF (K2.NE.3) GO TO 811
      S=1.
  807 UT(I,J)=S*UT(I,JP)
      UT(IP,J)=S*UT(IP,JP)
      VT(I,J)=-S*VT(I,J+2)
      VT(I,JP)=0.
      GO TO 811
  808 IF (K2.NE.4) GO TO 809
      S=-1.
      GO TO 810
  809 IF (K2.NE.3) GO TO 811
      S=1.
  810 UT(I,J)=S*UT(I,JM)
      UT(IP,J)=S*UT(IP,JM)
      VT(I,J)=0.
      VT(I,JP)=-S*VT(I,JM)
  811 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE DENCHG
      COMMON DELTAS,DT,DT2,DT4DX,DT4DY,DTCP,DTDX,DTDY,DTDXS,DTDYS,DTP,
     $       DTPP,DTVP,DXC,DXCD2,DXI4,DXIN,DXP,DYC,DYCD2,DYI4,DYIN,DYP
      COMMON EPS,G,GH,GX,GXD,GXDT,GY,GYD,GYDT,H,ICNTR,ITEST,KD,KKK,LL,
     $       MI,MO,MU1,MU2,NB1,NB2,NI,NIM1,NIP1,NJ,NJM1,NJP1,NP,NPR,NTP,
     $       ODX,ODX2,ODXS,ODY,ODY2,ODYS,R1,R2,T,TCP,TL,TP,TPP,TVP,UO,
     $       VO,W
      COMMON A(50,20),B1(50,20),B2(50,20),B3(50,20),B4(50,20),CS(50,20),
     $       DV(13),IMP(13),MKC(13),MSK(13),MU(50,20),NF(30),NK(50,20),
     $       NKT(50,20),P(50,20),PS(4500),PSI(51,21),R(50,20),SR(50,20),
     $       SRT(50,20),U(51,21),UP(4500),UT(51,21),V(51,21),VP(4500),
     $       VT(51,21),XC(50),XP(4500),XPO(30),XPL(30),YC(20),YP(4500),
     $       YPO(30),YPL(30),ZET(51,21)
      REAL MU,MU1,MU2,NK,NKT
      INTEGER CS,PS
      LL=0
      DO 600 J=1,NJ
      DO 600 I=1,NI
600   CS(I,J)=IMP(12).OR.(CS(I,J).AND.MKC(12))
      DO 602 J=2,NJM1
      DO 602 I=2,NIM1
      K1=CS(I,J).AND.MSK
      IF (K1.EQ.4) GO TO 601
      IF (K1.NE.5) GO TO 602
601   K11=CS(I,J).AND.MSK(11)
      K11=K11*DV(11)
      IF (K11.EQ.1) GO TO 602
      IF (NK(I,J)+NKT(I,J).EQ.0.) GO TO 602
      RHO=(SR(I,J)+SRT(I,J))/(NK(I,J)+NKT(I,J))
699   IF (RHO.EQ.R(I,J)) GO TO 602
      R(I,J)=RHO
      LL=LL+1
      KT=2*IMP(12)
      CS(I,J)=KT.OR.(CS(I,J).AND.MKC(12))
      KT=2*IMP(13)
      CS(I,J)=KT.OR.(CS(I,J).AND.MKC(13))
602   CONTINUE
      RETURN
      END


      SUBROUTINE NEWBA
      COMMON DELTAS,DT,DT2,DT4DX,DT4DY,DTCP,DTDX,DTDY,DTDXS,DTDYS,DTP,
     $       DTPP,DTVP,DXC,DXCD2,DXI4,DXIN,DXP,DYC,DYCD2,DYI4,DYIN,DYP
      COMMON EPS,G,GH,GX,GXD,GXDT,GY,GYD,GYDT,H,ICNTR,ITEST,KD,KKK,LL,
     $       MI,MO,MU1,MU2,NB1,NB2,NI,NIM1,NIP1,NJ,NJM1,NJP1,NP,NPR,NTP,
     $       ODX,ODX2,ODXS,ODY,ODY2,ODYS,R1,R2,T,TCP,TL,TP,TPP,TVP,UO,
     $       VO,W
      COMMON A(50,20),B1(50,20),B2(50,20),B3(50,20),B4(50,20),CS(50,20),
     $       DV(13),IMP(13),MKC(13),MSK(13),MU(50,20),NF(30),NK(50,20),
     $       NKT(50,20),P(50,20),PS(4500),PSI(51,21),R(50,20),SR(50,20),
     $       SRT(50,20),U(51,21),UP(4500),UT(51,21),V(51,21),VP(4500),
     $       VT(51,21),XC(50),XP(4500),XPO(30),XPL(30),YC(20),YP(4500),
     $       YPO(30),YPL(30),ZET(51,21)
      REAL MU,MU1,MU2,NK,NKT
      INTEGER CS,PS
      DO 219 J=2,NJM1
      JM=J-1
      JP=J+1
      DO 219 I=2,NIM1
      IM=I-1
      IP=I+1
      K12=CS(I,J).AND.MSK(12)
      K12=K12*DV(12)
      IF (K12.EQ.1) GO TO 219
      ML=1
      MR=1
      MB=1
      K1A=CS(IM,J).AND.MSK
      IF (K1A.NE.2) GO TO 200
      ML=2
200   K1B=CS(IP,J).AND.MSK
      IF (K1B.NE.2) GO TO 201
      MR=2
201   K1C=CS(I,JM).AND.MSK
      IF (K1C.NE.2) GO TO 202
      MB=2,
202   K1D=CS(I,JP).AND.MSK
      IF (K1D.NE.2) GO TO 203
      MT=2
203   IF (ML+MR+MT+MB.NE.4) GO TO 204
      RJM=R(I,J-2)
      RJP=R(I,J+2)
      RIM=R(I-2,J)
      RIP=R(I+2,J)
      GO TO 218
204   IF (ML+MR+MB.NE.6) GO TO 205
      RIM=0.
      RIP=0.
      RJM=0.
      RJP=R(I,J+2)
      GO TO 218
205   IF (ML+MR+MT.NE.6) GO TO 206
      RIM=0.
      RIP=0.
      RJM=R(I,J-2)
      RJP=0.
      GO TO 218
206   IF (ML+MB+MT.NE.6) GO TO 207
      RIM=0.
      RIP=R(I+2,J)
      RJM=0.
      RJP=0.
      GO TO 218
207   IF (MR+MB+MT.NE.6) GO TO 208
      RIM=R(I-2,J)
      RIP=0.
      RJM=0.
      RJP=0.
      GO TO 218
208   IF (ML+MR.NE.4) GO TO 209
      RIM=0.
      RIP=0.
      RJM=R(I,J-2)
      RJP=R(I,J+2)
      GO TO 218
209   IF (ML+MB.NE.4) GO TO 210
      RIM=0.
      RIP=R(I+2,J)
      RJM=0.
```

```
      RJP=R(I,J+2)
      GO TO 218
  210 IF (ML+MT.NE.4) GO TO 211
      RIM=0.
      RIP=R(I+2,J)
      RJM=R(I,J-2)
      RJP=0.
      GO TO 218
  211 IF (MR+MB.NE.4) GO TO 212
      RIM=R(I-2,J)
      RIP=0.
      RJM=0.
      RJP=R(I,J+2)
      GO TO 218
  212 IF (MR+MT.NE.4) GO TO 213
      RIM=R(I-2,J)
      RIP=0.
      RJM=R(I,J-2)
      RJP=0.
      GO TO 218
  213 IF (MB+MT.NE.4) GO TO 214
      RIM=R(I-2,J)
      RIP=R(I+2,J)
      RJM=0.
      RJP=0.
      GO TO 218
  214 IF (ML.NE.2) GO TO 215
      RIM=0.
      RIP=R(I+2,J)
      RJM=R(I,J-2)
      RJP=R(I,J+2)
      GO TO 218
  215 IF (MR.NE.2) GO TO 216
      RIM=R(I-2,J)
      RIP=0.
      RJM=R(I,J-2)
      RJP=R(I,J+2)
      GO TO 218
  216 IF (MB.NE.2) GO TO 217
      RIM=R(I-2,J)
      RIP=R(I+2,J)
      RJM=0.
      RJP=R(I,J+2)
      GO TO 218
  217 IF (MT.NE.2) GO TO 218
      RIM=R(I-2,J)
      RIP=R(I+2,J)
      RJM=R(I,J-2)
      RJP=0.
  218 AR1=1./(R(I,J)+R(IP,J))
      AR2=1./(R(I,J)+R(IM,J))
      AR3=1./(R(I,J)+R(I,JP))
      AR4=1./(R(I,J)+R(I,JM))
      AR5=1./(R(I,JM)+R(IP,JM))
      AR6=1./(R(I,JM)+R(IM,JM))
      AR7=1./(R(I,JM)+RJM)
      AR8=1./(R(I,JP)+R(IP,JP))
      AR9=1./(R(I,JP)+R(IM,JP))
      AR10=1./(R(I,JP)+RJP)
      AR11=1./(R(IM,J)+RIM)
      AR12=1./(R(IM,J)+R(IM,JP))
      AR13=1./(R(IM,J)+R(IM,JM))
      AR14=1./(R(IP,J)+RIP)
      AR15=1./(R(IP,J)+R(IP,JP))
      AR16=1./(R(IP,J)+R(IP,JM))
      CIJ=1./(DT2*(ODXS*(AR1+AR2)+ODYS*(AR3+AR4)))
      CIJM=1./(DT2*(ODXS*(AR5+AR6)+ODYS*(AR4+AR7)))
      CIJP=1./(DT2*(ODXS*(AR8+AR9)+ODYS*(AR10+AR3)))
      CIMJ=1./(DT2*(ODXS*(AR2+AR11)+ODYS*(AR12+AR13)))
      CIPJ=1./(DT2*(ODXS*(AR14+AR1)+ODYS*(AR15+AR16)))
      B1(I,J)=CIJ*DTDXS*AR1
      B1(I,JM)=CIJM*DTDXS*AR5
      B1(I,JP)=CIJP*DTDXS*AR8
      B1(IM,J)=CIMJ*DTDXS*AR2
      B1(IP,J)=CIPJ*DTDXS*AR14
      B2(I,J)=CIJ*DTDXS*AR2
      B2(I,JM)=CIJM*DTDXS*AR6
      B2(I,JP)=CIJP*DTDXS*AR9
      B2(IM,J)=CIMJ*DTDXS*AR11
      B2(IP,J)=CIPJ*DTDXS*AR1
      B3(I,J)=CIJ*DTDYS*AR3
      B3(I,JM)=CIMJ*DTDYS*AR4
      B3(I,JP)=CIJP*DTDYS*AR10
      B3(IM,J)=CIMJ*DTDYS*AR12
      B3(IP,J)=CIPJ*DTDYS*AR15
      B4(I,J)=CIJ*DTDYS*AR4
      B4(I,JM)=CIJM*DTDYS*AR7
      B4(I,JP)=CIJP*DTDYS*AR3
      B4(IM,J)=CIMJ*DTDYS*AR13
      B4(IP,J)=CIPJ*DTDYS*AR16
      A(I,J)=CIJ*(ODX2*(PSI(I,J)*AR2-PSI(IP,J)*AR1)+ODY2*(ZET(I,J)*AR4-
     $       ZET(I,JP)*AR3))
      A(I,JM)=CIJM*(ODX2*(PSI(I,JM)*AR6-PSI(IP,JM)*AR5)+ODY2*(ZET(I,JM)*
     $        AR7 -ZET(I,J)*AR4))
      A(I,JP)=CIJP*(ODX2*(PSI(I,JP)*AR9-PSI(IP,JP)*AR8)+ODY2*(ZET(I,JP)*
     $        AR3 -ZET(I,J+2)*AR10))
      A(IM,J)=CIMJ*(ODX2*(PSI(IM,J)*AR11-PSI(I,J)*AR2)+ODY2*(ZET(IM,J)*
     $        AR13 -ZET(IM,JP)*AR12))
      A(IP,J)=CIPJ*(ODX2*(PSI(IP,J)*AR1-PSI(I+2,J)*AR14)+ODY2*(ZET(IP,J)
     $        *AR16 -ZET(IP,JP)*AR15))
  219 CONTINUE
      RETURN
      END

      SUBROUTINE MOVPAR
      COMMON DELTAS,DT,DT2,DT4DX,DT4DY,DTCP,DTDX,DTDY,DTDXS,DTDYS,DTP,
     $       DTPP,DTVP,DXC,DXCD2,DXI4,DXIN,DXP,DYC,DYCD2,DYI4,DYIN,DYP
      COMMON EPS,G,GH,GX,GXD,GXDT,GY,GYD,GYDT,H,ICNTR,ITEST,KD,KKK,LL,
     $       MI,MO,MU1,MU2,NB1,NB2,NI,NIM1,NIP1,NJ,NJM1,NJP1,NP,NPR,NTP,
     $       ODX,ODX2,ODXS,ODY,ODY2,ODYS,R1,R2,T,TCP,TL,TP,TPP,TVP,U0,
     $       V0,W
      COMMON A(50,20),B1(50,20),B2(50,20),B3(50,20),B4(50,20),CS(50,20),
     $       DV(13),IMP(13),MKC(13),MSK(13),MU(50,20),NF(30),NK(50,20),
     $       NKT(50,20),P(50,20),PS(4500),PSI(51,21),R(50,20),SR(50,20),
     $       SRT(50,20),U(51,21),UP(4500),UT(51,21),V(51,21),VP(4500),
     $       VT(51,21),XC(50),XP(4500),XPO(30),XPL(30),YC(20),YP(4500),
     $       YPO(30),YPL(30),ZET(51,21)
      REAL MU,MU1,MU2,NK,NKT
      INTEGER CS,PS
      DO 148 K=1,NP
      KP=PS(K).AND.MSK
```

```
      IF (KP.EQ.3) GO TO 148
      O=XP(K)*ODX+2.
      Q=YP(K)*ODY+2.
      I=O
      J=Q
      FX=O-I
      FY=Q-J
      K1=CS(I,J).AND.MSK
      IF (K1.EQ.4) GO TO 99
      IF (K1.NE.5) GO TO 138
   99 IF (KKK.EQ.2) GO TO 100
      K10=CS(I,J).AND.MSK(10)
      K10=K10*DV(10)
      IF (K10.EQ.1) GO TO 148
  100 IF (FY.LT..5) GO TO 101
      JPR=J
      GO TO 102
  101 JPR=J-1
  102 IP=I+1
      JPRP=JPR+1
      HPSX=.5+ODX*(XC(I)-XP(K))
      HPSY=.5+ODY*(YC(JPR)+DYCD2-YP(K))
      HMSX=1.-HPSX
      HMSY=1.-HPSY
      IF (KKK.EQ.2) GO TO 103
      UT1=UT(I,JPRP)
      UT2=UT(I,J)
      UT3=UT(IP,JPRP)
      UT4=UT(IP,J)
      UT5=UT(I,JPR)
      UT6=UT(IP,JPR)
      GO TO 104
  103 UT1=U(I,JPRP)
      UT2=U(I,J)
      UT3=U(IP,JPRP)
      UT4=U(IP,J)
      UT5=U(I,JPR)
      UT6=U(IP,JPR)
  104 IF (UT1.NE.0.) GO TO 105
      U1=UT2
      GO TO 107
  105 K5A=CS(I-1,J).AND.MSK(5)
      K5A=K5A*DV(5)
      IF (K5A.EQ.1) GO TO 106
      U1=0.
      GO TO 107
  106 U1=UT1
  107 IF (UT3.NE.0.) GO TO 108
      U2=UT4
      GO TO 110
  108 K5B=CS(IP,J).AND.MSK(5)
      K5B=K5B*DV(5)
      IF (K5B.EQ.1) GO TO 109
      U2=0.
      GO TO 110
  109 U2=UT3
  110 IF (UT5.NE.0.) GO TO 111
      U3=UT2
      GO TO 113
  111 K5A=CS(I-1,J).AND.MSK(5)
      K5A=K5A*DV(5)
      IF (K5A.EQ.1) GO TO 112
      U3=0.
      GO TO 113
  112 U3=UT5
  113 IF (UT6.NE.0.) GO TO 114
      U4=UT4
      GO TO 116
  114 K5B=CS(IP,J).AND.MSK(5)
      K5B=K5B*DV(5)
      IF (K5B.EQ.1) GO TO 115
      U4=0.
      GO TO 116
  115 U4=UT6
  116 UPT=HPSX*HMSY*U1+HMSX*HMSY*U2+HPSX*HPSY*U3+HMSX*HPSY*U4
      XPT=XP(K)+UPT*DT
      IF (FX.LT..5) GO TO 117
      IPR=I
      GO TO 118
  117 IPR=I-1
  118 JP=J+1
      IPRP=IPR+1
      HPSX=.5+ODX*(XC(IPR)+DXCD2-XP(K))
      HPSY=.5+ODY*(YC(J)-YP(K))
      HMSX=1.-HPSX
      HMSY=1.-HPSY
      IF (KKK.EQ.2) GO TO 119
      VT1=VT(IPR,JP)
      VT2=VT(I,JP)
      VT3=VT(IPRP,JP)
      VT4=VT(IPR,J)
      VT5=VT(I,J)
      VT6=VT(IPRP,J)
      GO TO 120
  119 VT1=V(IPR,JP)
      VT2=V(I,JP)
      VT3=V(IPRP,JP)
      VT4=V(IPR,J)
      VT5=V(I,J)
      VT6=V(IPRP,J)
  120 IF (VT1.NE.0.) GO TO 121
      V1=VT2
      GO TO 123
  121 K5C=CS(I,JP).AND.MSK(5)
      K5C=K5C*DV(5)
      IF (K5C.EQ.1) GO TO 122
      V1=0.
      GO TO 123
  122 V1=VT1
  123 IF (VT3.NE.0.) GO TO 124
      V2=VT2
      GO TO 126
  124 K5C=CS(I,JP).AND.MSK(5)
      K5C=K5C*DV(5)
      IF (K5C.EQ.1) GO TO 125
      V2=0.
      GO TO 126
  125 V2=VT3
  126 IF (VT4.NE.0.) GO TO 127
      V3=VT5
      GO TO 129
  127 K5D=CS(I,J-1).AND.MSK(5)
```

```
      K5D=K5D*DV(5)
      IF (K5D.EQ.1) GO TO 128
      V3=0.
      GO TO 129
  128 V3=VT4
  129 IF (VT6.NE.0.) GO TO 130
      V4=VT5
      GO TO 132
  130 K5D=CS(I,J-1).AND.MSK(5)
      K5D=K5D*DV(5)
      IF (K5D.EQ.1) GO TO 131
      V4=0.
      GO TO 132
  131 V4=VT6
  132 VPT=HPSX*HMSY*V1+HMSX*HMSY*V2+HPSX*HPSY*V3+HMSX*HPSY*V4
      YPT=YP(K)+VPT*DT
      I1=XPT*ODX+2.
      J1=YPT*ODY+2.
      IF (I1.LE.NI.AND.J1.LE.NJ) GO TO 133
      KD=2
      WRITE (MO,1000) K,I,J,KKK
      GO TO 149
  133 IF (KKK.EQ.1) GO TO 134
      UP(K)=UPT
      XP(K)=XPT
      VP(K)=VPT
      YP(K)=YPT
      GO TO 148
  134 IF (J.NE.J1) GO TO 135
      IF (I.EQ.I1) GO TO 148
  135 KPA=PS(K).AND.MSK(3)
      KPA=KPA*DV(3)
      IF (KPA.EQ.2) GO TO 136
      SRT(I,J)=SRT(I,J)-R1
      SRT(I1,J1)=SRT(I1,J1)+R1
      GO TO 137
  136 SRT(I,J)=SRT(I,J)-R2
      SRT(I1,J1)=SRT(I1,J1)+R2
  137 NKT(I,J)=NKT(I,J)-1.
      NKT(I1,J1)=NKT(I1,J1)+1.
      GO TO 148
  138 K2=CS(I,J).AND.MSK(2)
      K2=K2*DV(2)
      IF (K2.NE.1) GO TO 148
      IF (KKK.EQ.2) GO TO 139
      UT7=UT(I,J)
      UT8=UT(I+1,J)
      VT7=VT(I,J)
      VT8=VT(I,J+1)
      GO TO 140
  139 UT7=U(I,J)
      UT8=U(I+1,J)
      VT7=V(I,J)
      VT8=V(I,J+1)
  140 K7=CS(I,J).AND.MSK(7)
      K7=K7*DV(7)
      IF (K7.EQ.1) GO TO 144
      IF (K7.EQ.2) GO TO 143
      IF (K7.EQ.3) GO TO 141
      T2=VT7*DT
      DYIN=-T2
      KT=4*IMP(2)
      PS(K)=KT.OR.(PS(K).AND.MKC(2))
      GO TO 142
  141 T2=VT8*DT
      DYIN=T2
      KT=3*IMP(2)
      PS(K)=KT.OR.(PS(K).AND.MKC(2))
  142 YPT=YP(K)+T2
      XPT=XP(K)
      GO TO 146
  143 T1=UT7*DT
      DXIN=-T1
      KT=2*IMP(2)
      PS(K)=KT.OR.(PS(K).AND.MKC(2))
      GO TO 145
  144 T1=UT8*DT
      DXIN=T1
      PS(K)=IMP(2).OR.(PS(K).AND.MKC(2))
  145 XPT=XP(K)+T1
      YPT=YP(K)
  146 IF (KKK.EQ.2) GO TO 147
      I1=XPT*ODX+2.
      J1=YPT*ODY+2.
      GO TO 134
  147 XP(K)=XPT
      YP(K)=YPT
  148 CONTINUE
  149 RETURN
 1000 FORMAT(1H-/1H-,20X13HPARTICLE NO. ,I4,13H,  FROM CELL ,I2,1H,,I2,2
     $2H,  CROSSED A BOUNDARY.,10X6HKKK = ,I1)
      END


      SUBROUTINE VELCTS
      COMMON DELTAS,DT,DT2,DT4DX,DT4DY,DTCP,DTDX,DTDY,DTDXS,DTDYS,DTP,
     $       DTPP,DTVP,DXC,DXCD2,DXI4,DXIN,DXP,DYC,DYCD2,DYI4,DYIN,DYP
      COMMON EPS,G,GH,GX,GXD,GXDT,GY,GYD,GYDT,H,ICNTR,ITEST,KD,KKK,LL,
     $       MI,MO,MU1,MU2,NB1,NB2,NI,NIM1,NIP1,NJ,NJM1,NJP1,NP,NPR,NTP,
     $       ODX,ODX2,ODXS,ODY,ODY2,ODYS,R1,R2,T,TCP,TL,TP,TPP,TVP,UO,
     $       VO,W
      COMMON A(50,20),B1(50,20),B2(50,20),B3(50,20),B4(50,20),CS(50,20),
     $       DV(13),IMP(13),MKC(13),MSK(13),MU(50,20),NF(30),NK(50,20),
     $       NKT(50,20),P(50,20),PS(4500),PSI(51,21),R(50,20),SR(50,20),
     $       SRT(50,20),U(51,21),UP(4500),UT(51,21),V(51,21),VP(4500),
     $       VT(51,21),XC(50),XP(4500),XP0(30),XPL(30),YC(20),YP(4500),
     $       YP0(30),YPL(30),ZET(51,21)
      REAL MU,MU1,MU2,NK,NKT
      INTEGER CS,PS
      DO 401 J=2,NJM1
      JP=J+1
      DO 401 I=2,NIM1
      IP=I+1
      K1=CS(I,J).AND.MSK
      IF (K1.EQ.4) GO TO 400
      IF (K1.NE.5) GO TO 401
  400 T1=(PSI(IP,J)+DTDX*(P(I,J)-P(IP,J)))/(R(I,J)+R(IP,J))
      T2=(ZET(I,JP)+DTDY*(P(I,J)-P(I,JP)))/(R(I,J)+R(I,JP))
      U(IP,J)=T1+T1
      V(I,JP)=T2+T2
  401 CONTINUE
      CALL BND(2ND
```

```
      RETURN
      END


      SUBROUTINE DENVIS
      COMMON DELTAS,DT,DT2,DT4DX,DT4DY,DTCP,DTDX,DTDY,DTDXS,DTDYS,DTP,
     $       DTPP,DTVP,DXC,DXCD2,DXI4,DXIN,DXP,DYC,DYCD2,DYI4,DYIN,DYP
      COMMON EPS,G,GH,GX,GXD,GXDT,GY,GYD,GYDT,H,ICNTR,ITEST,KD,KKK,LL,
     $       MI,MO,MU1,MU2,NB1,NB2,NI,NIM1,NIP1,NJ,NJM1,NJP1,NP,NPR,NTP,
     $       ODX,ODX2,ODXS,ODY,ODY2,ODYS,R1,R2,T,TCP,TL,TP,TPP,TVP,U0,
     $       V0,W
      COMMON A(50,20),B1(50,20),B2(50,20),B3(50,20),B4(50,20),CS(50,20),
     $       DV(13),IMP(13),MKC(13),MSK(13),MU(50,20),NF(30),NK(50,20),
     $       NKT(50,20),P(50,20),PS(4500),PSI(51,21),R(50,20),SR(50,20),
     $       SRT(50,20),U(51,21),UP(4500),UT(51,21),V(51,21),VP(4500),
     $       VT(51,21),XC(50),XP(4500),XP0(30),XPL(30),YC(20),YP(4500),
     $       YP0(30),YPL(30),ZET(51,21)
      REAL MU,MU1,MU2,NK,NKT
      INTEGER CS,PS
      DIMENSION SM(50,20)
      EQUIVALENCE (SM,NKT)
      DO 300 J=1,NJ
      DO 300 I=1,NI
      NK(I,J)=0.
      SR(I,J)=0.
  300 SM(I,J)=0.
      KK=1
      DO 303 K=1,NP
      KP=PS(K).AND.MSK
      IF (KP.EQ.3) GO TO 303
      KK=2
      I=XP(K)*ODX+2.
      J=YP(K)*ODY+2.
      KPA=PS(K).AND.MSK(3)
      KPA=KPA*DV(3)
      IF (KPA.EQ.2) GO TO 301
      SR(I,J)=SR(I,J)+R1
      SM(I,J)=SM(I,J)+MU1
      GO TO 302
  301 SR(I,J)=SR(I,J)+R2
      SM(I,J)=SM(I,J)+MU2
  302 NK(I,J)=NK(I,J)+1.
  303 CONTINUE
      IF (KK.EQ.1) GO TO 306
      DO 305 J=1,NJ
      DO 305 I=1,NI
      IF (NK(I,J).EQ.0.) GO TO 305
      R(I,J)=SR(I,J)/NK(I,J)
      MU(I,J)=SM(I,J)/NK(I,J)
  305 CONTINUE
      DO 357 J=1,NJ
      JM=J-1
      JP=J+1
      DO 357 I=1,NI
      IM=I-1
      IP=I+1
      K1=CS(I,J).AND.MSK
      IF (K1.NE.2) GO TO 353
      K7=CS(I,J).AND.MSK(7)
      K7=K7*DV(7)
      IF (K7.EQ.4) GO TO 352
      IF (K7.EQ.3) GO TO 351
      IF (K7.EQ.2) GO TO 350
      R(I,J)=R(IP,J)
      MU(I,J)=MU(IP,J)
      GO TO 357
  350 R(I,J)=R(IM,J)
      MU(I,J)=MU(IM,J)
      GO TO 357
  351 R(I,J)=R(I,JP)
      MU(I,J)=MU(I,JP)
      GO TO 357
  352 R(I,J)=R(I,JM)
      MU(I,J)=MU(I,JM)
      GO TO 357
  353 IF (K1.NE.1) GO TO 357
      K7=CS(I,J).AND.MSK(7)
      K7=K7*DV(7)
      IF (K7.EQ.4) GO TO 356
      IF (K7.EQ.3) GO TO 355
      IF (K7.EQ.2) GO TO 354
      IF (K7.NE.1) GO TO 357
      R(I,J)=R(IP,JP)
      MU(I,J)=MU(IP,JP)
      GO TO 357
  354 R(I,J)=R(IM,JM)
      MU(I,J)=MU(IM,JM)
      GO TO 357
  355 R(I,J)=R(IM,JP)
      MU(I,J)=MU(IM,JP)
      GO TO 357
  356 R(I,J)=R(IP,JM)
      MU(I,J)=MU(IP,JM)
  357 CONTINUE
      GO TO 307
  306 KD=2
      WRITE (MO,3000) T
  307 RETURN
 3000 FORMAT(1H-/1H-,20X33HNO PARTICLES IN SYSTEM AT TIME = ,F6.3)
      END


      SUBROUTINE REFCEL
      COMMON DELTAS,DT,DT2,DT4DX,DT4DY,DTCP,DTDX,DTDY,DTDXS,DTDYS,DTP,
     $       DTPP,DTVP,DXC,DXCD2,DXI4,DXIN,DXP,DYC,DYCD2,DYI4,DYIN,DYP
      COMMON EPS,G,GH,GX,GXD,GXDT,GY,GYD,GYDT,H,ICNTR,ITEST,KD,KKK,LL,
     $       MI,MO,MU1,MU2,NB1,NB2,NI,NIM1,NIP1,NJ,NJM1,NJP1,NP,NPR,NTP,
     $       ODX,ODX2,ODXS,ODY,ODY2,ODYS,R1,R2,T,TCP,TL,TP,TPP,TVP,U0,
     $       V0,W
      COMMON A(50,20),B1(50,20),B2(50,20),B3(50,20),B4(50,20),CS(50,20),
     $       DV(13),IMP(13),MKC(13),MSK(13),MU(50,20),NF(30),NK(50,20),
     $       NKT(50,20),P(50,20),PS(4500),PSI(51,21),R(50,20),SR(50,20),
     $       SRT(50,20),U(51,21),UP(4500),UT(51,21),V(51,21),VP(4500),
     $       VT(51,21),XC(50),XP(4500),XP0(30),XPL(30),YC(20),YP(4500),
     $       YP0(30),YPL(30),ZET(51,21)
      REAL MU,MU1,MU2,NK,NKT
      INTEGER CS,PS
      DIMENSION SU(51,21),SV(51,21)
      EQUIVALENCE (SU,UT),(SV,VT)
      DO 710 K=1,NP
      KP=PS(K).AND.MSK
      IF (KP.EQ.3) GO TO 710
```

```
      I=XP(K)*ODX+2.
      J=YP(K)*ODY+2.
      IF (KP.EQ.2) GO TO 701
      K1=CS(I,J).AND.MSK
      IF (K1.EQ.1) GO TO 700
      IF (K1.NE.2) GO TO 710
  700 PS(K)=3.OR.(PS(K).AND.MKC)
      XP(K)=0.
      YP(K)=0.
      UP(K)=0.
      VP(K)=0.
      GO TO 710
  701 K2=CS(I,J).AND.MSK(2)
      K2=K2*DV(2)
      IF (K2.EQ.1) GO TO 710
      PS(K)=1.OR.(PS(K).AND.MKC)
      DO 702 L=1,NP
      LP=PS(L).AND.MSK
      IF (LP.EQ.3) GO TO 703
  702 CONTINUE
      GO TO 710
  703 KB=PS(K).AND.MSK(2)
      KB=KB*DV(2)
      IF (KB.NE.4) GO TO 704
      YP(L)=YP(K)+DYIN
      GO TO 705
  704 IF (KB.NE.3) GO TO 706
      YP(L)=YP(K)-DYIN
  705 XP(L)=XP(K)
      GO TO 709
  706 IF (KB.NE.1) GO TO 707
      XP(L)=XP(K)-DXIN
      GO TO 708
  707 XP(L)=XP(K)+DXIN
  708 YP(L)=YP(K)
  709 PS(L)=2.OR.(PS(L).AND.MKC)
  710 CONTINUE
      DO 711 J=1,NJ
      DO 711 I=1,NI
      KT=4*IMP(9)
  711 CS(I,J)=KT.OR.(CS(I,J).AND.MKC(9))
      DO 712 J=1,NJ
      DO 712 I=1,NI
      SU(I,J)=0.
  712 SV(I,J)=0.
      DO 714 K=1,NP
      KP=PS(K).AND.MSK
      IF (KP.EQ.3) GO TO 714
      I=XP(K)*ODX+2.
      J=YP(K)*ODY+2.
      SU(I,J)=SU(I,J)+UP(K)
      SV(I,J)=SV(I,J)+VP(K)
      K9=CS(I,J).AND.MSK(9)
      K9=K9*DV(9)
      KPA=PS(K).AND.MSK(3)
      KPA=KPA*DV(3)
      IF (K9.NE.4) GO TO 713
      KT=KPA*IMP(9)
      CS(I,J)=KT.OR.(CS(I,J).AND.MKC(9))
      GO TO 714
  713 IF (KPA.EQ.K9) GO TO 714
      KT=3*IMP(9)
      CS(I,J)=KT.OR.(CS(I,J).AND.MKC(9))
  714 CONTINUE
      CALL FLGCEL
      RETURN
      END
```