NUREG/CR-0626
LA-7659

# SOLA-LOOP: A Nonequilibrium, Drift-Flux Code for Two-Phase Flow in Networks

University of California

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

8007280 014

Affirmative Action/Equal Opportunity Employer

NOTICE

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, or any of their employees, makes any warranty, expressed or implied, or assumes any legal liability or responsibility for any third party's use, or the results of such use, of any information, apparatus, product or process disclosed in this report, or represents that its use by such third party would not infringe privately owned rights.

The views expressed in this report are not necessarily those of the US Nuclear Regulatory Commission.

NUREG/CR-0626
LA-7659

R-4

# SOLA-LOOP: A Nonequilibrium, Drift-Flux Code for Two-Phase Flow in Networks

C. W. Hirt
T. A. Oliphant
W. C. Rivard
N. C. Romero
M. D. Torrey

Manuscript submitted: January 1979
Date published: June 1979

Prepared for
Division of Reactor Safety Research
Office of Nuclear Regulatory Research
US Nuclear Regulatory Commission
Washington, DC 20555

NRC FIN No. A-FIN-7027-9

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

# CONTENTS

# SOLA-LOOP: A NONEQUILIBRIUM, DRIFT-FLUX CODE FOR TWO-PHASE FLOW IN NETWORKS

by

C. W. Hirt, T. A. Oliphant, W. C. Rivard,
N. C. Romero, and M. D. Torrey

## ABSTRACT

**A new, highly flexible computer code for transient, nonequilibrium, two-phase flow in networks is described. Each component may have a one-dimensional representation with a variable cross-sectional area. The flow dynamics is governed by a set of nonlinear conservation laws based on a generalized drift-flux model for two-phase mixtures. The equations are solved by a partially implicit method that can use different time steps in different components.**

**In addition to being simple and modular, the code can use almost any set of constitutive relations, property tables, or other special purpose features required for different applications.**

**An example problem is provided to verify proper implementation of the code on the user's system. It illustrates the automatic treatment of such phenomena as critical two-phase flow without introducing special assumptions and the use of various input options to initiate a pipe break.**

## I. INTRODUCTION

The increasing demand for more accurate and more detailed predictions of two-phase flow processes has prompted a flurry of activity in recent years. Much of this activity has been generated and supported by efforts to provide better theoretical tools for the analysis of postulated nuclear reactor accidents. Leading this activity are numerous efforts in the US and abroad to make better use of the many sophisticated numerical techniques developed during the last twenty years for single-phase flows. These efforts include several highly detailed models for transient, two-phase flows in two and three dimensions.[1,2] Although such models are essential for understanding complex, two-phase flow processes in localized regions, they cannot be used directly for systems consisting of many coupled flow regions of various sizes, geometries, and other physical characteristics. For large systems, zero- and one-dimensional flow models must be coupled together in, so-called, network codes. Some higher dimensional models may be used for one or two parts of a system, but their use is limited by the higher computing costs that they entail. Thus, for now, multidimensional numerical computations will be confined largely to studies of fundamental two-phase flow processes, to detailed calculations in isolated regions, and to the definition and verification of the simplified lower dimensional models.

To make the best use of the new developments for the practical analysis of full systems, we must develop network codes that are compatible with the advanced multidimensional codes. Advanced network codes should use one-dimensional representations wherever possible to maintain the minimum spatial definition that is necessary for flow transients. Futhermore, advanced network codes should permit finite rate exchanges of mass, momentum, and energy between phases as well as unequal phase velocities to adequately describe the many possible types of two-phase flows.

In this report we describe SOLA-LOOP, an advanced network code derived from SOLA-DF, a transient, two-dimensional code.[3] SOLA-DF is based on a drift-flux approximation for the dynamics of a two-phase mixture.[4] SOLA-LOOP is a relatively simple code that has no flow regime maps, property tables, or other complicating features. Although developed for use in nuclear reactor safety analysis, its simple structure offers a framework that may be used as the basis for developing other types of special-purpose network codes.

Section II presents the differential equations that define the drift-flux model and describes the simple approximations for the equation-of-state and other necessary constitutive relations for water. Section III describes the numerical solution techniques used for one-dimensional components and for coupling components into a network. Section IV lists the input parameters and COMMON storage variables. The example problem results in Sec. V provide a means for verification of the code when implemented at other installations or on other machines. A complete listing of the code is provided in the Appendix.

## II. EQUATIONS, CONSTITUTIVE RELATIONS, AND EXCHANGE RATES

There are various forms for the drift-flux equations that describe the dynamics of two-phase fluid mixtures.[5] For our purpose we chose as dependent variables the mixture density $\rho$, the macroscopic vapor density $\rho_v$ (vapor mass per unit volume of mixture), the center of mass velocity $\underline{u}$, and the mixture specific internal energy I. Important auxiliary variables are the void volume fraction $\theta$, the relative velocity between phases $\underline{u}_r = \underline{u}_v - \underline{u}_\ell$, and the mixture pressure p.

### A. Equations of Motion

In terms of these dependent variables, the basic drift-flux equations used in SOLA-LOOP are

(1) The continuity equations,

$$\frac{\partial \rho}{\partial t} + \frac{1}{A}\frac{\partial A \rho u}{\partial y} = 0 \tag{2.1}$$

and

$$\frac{\partial \rho_v}{\partial t} + \frac{1}{A}\frac{\partial}{\partial y} A \left(\rho_v u + \frac{\rho_v \rho_\ell}{\rho} u_r\right) = \Gamma \; , \tag{2.2}$$

(2) the momentum equation,

$$\frac{\partial \rho u}{\partial t} + \frac{1}{A}\frac{\partial}{\partial y} A \left(\rho u^2 + \frac{\rho_v \rho_\ell}{\rho} u_r^{\,2}\right) = - \frac{\partial p}{\partial y} + \rho g_y + f_{vis} , \tag{2.3}$$

(3) and the internal energy equation,

$$\begin{aligned} &\frac{\partial \rho I}{\partial t} + \frac{1}{A}\frac{\partial}{\partial y} A \left[\rho I u + \frac{\rho_v \rho_\ell}{\rho} (I_v - I_\ell)\, u_r\right] \\ &\quad = - \frac{p}{A}\frac{\partial}{\partial y} \cdot \left[u + \frac{\rho_v \rho_\ell}{\rho}\left(\frac{1}{\rho_v^{\,o}} - \frac{1}{\rho_\ell^{\,o}}\right) u_r\right] + K\, u_r^{\,2} + W_{vis} + Q . \end{aligned} \tag{2.4}$$

In these equations, the independent variables are time t and axial position y. The exchange functions for mass and momentum are $\Gamma$ and K, respectively. The effects of wall heat transfer or bulk heating are described by the heat source function Q. Subscripts v and $\ell$ refer to properties in the vapor and liquid states. Superscript zero on the densities refers to microscopic quantities, whereas densities without superscripts are macroscopic values. The microscopic and macroscopic densities are related through the void fraction as $\rho_v = \theta \rho_v^{\,o}$ and $\rho_\ell = (1 - \theta)\, \rho_\ell^{\,o}$, where $\theta$ is defined as

$$\theta = (\rho_\ell^{\,o} - \rho + \rho_v)/\rho_\ell^{\,o} . \tag{2.5}$$

The axial component of the gravitational acceleration is denoted by $g_y$. The quantity A is the time-independent, cross-sectional area of the flow channel or pipe. In addition to representing variable area ducts, suitably defined A values may be used to represent cylindrical coordinates ($A = r$, the circumferential area per unit azimuthal angle), closed-off pipes ($A = 0$), or approximations for orifices, valves, and abrupt area changes. In the latter cases, local flow losses from rapid area changes are accounted for by adding the necessary pressure loss and energy dissipation to Eqs. (2.3) and (2.4) through the terms $f_{vis}$ and $W_{vis}$. Pipe wall friction is treated similarly.

To complete these equations, constitutive relations and exchange rates must be specified. Considerable care must be exercised when defining these relations. The choices made are governed by the intended use of the code. The best choices are those that can be tested against suitable experimental data. Even with careful testing, however, the prejudices of different researchers often lead to different relations. In the following we describe one set of simple models used in the initial development of the SOLA-DF and SOLA-LOOP codes. These models are not the best possible and, therefore, should not be taken as invariant features of these codes. Instead, the codes are to be regarded as skeletons offering a numerical solution algorithm that will work with various choices.

## B. Constitutive Relations

The equation of state in SOLA-LOOP is a relation that gives pressure as a function of density and internal energy. Although fits to steam table data could be inserted in the equation-of-state subroutines, for developmental purposes we chose a simpler approach. When the void fraction is below a small, predetermined value $\theta_c$ (typically $\theta_c = 0.001$), the fluid is assumed to be a pure liquid with the equation of state

$$p = p_o + a^2 \left(\rho - \rho_\ell^{\,o}\right) ,$$

where a is the speed of sound in the liquid phase and $p_o$ is chosen [see Eq. (2.6)] to ensure pressure continuity between the pure liquid and two-phase states when $\theta = \theta_c$. In the two-phase region $\theta > \theta_c$, the mixture pressure is equal to that of the vapor and is given by the polytropic gas equation

$$p = (\gamma - 1)\rho_v^o I_v \ .$$

These equations are combined into one equation

$$p = (\gamma - 1)\ \rho_v I_v/\theta^* + a^2\ \rho_\ell^{\ o}(\theta^* - \theta)\ , \tag{2.6}$$

where

$$\theta^* = \begin{cases} \theta & \text{if } \theta \geq \theta_c \\ \theta_c & \text{if } \theta < \theta_c \ . \end{cases}$$

For saturated conditions we have found that $\gamma = 1.07$ and $a^2 \approx 10^4\ cm^2/ms^2$ offer reasonable approximations for many reactor safety problems.

In Eqs. (2.4) and (2.6), separate values for vapor and liquid internal energies are required. Because the basic dependent energy variable is the mixture internal energy, a separate prescription must be given for determining the individual phase energies. Two prescriptions are used. In one the phases are considered to be at equal temperatures. In the other the vapor phase is considered to be saturated. For many applications there is little difference between the two because the large heat content of the liquid phase keeps the liquid temperature nearly invariant. For simplicity the vapor and liquid internal energies are specified as functions of the vapor and liquid temperatures as

$$I_v = E_v + C_v(T_v - T_o) - C_{v1}(T_v - T_o)^2$$

and

$$I_\ell = E_\ell + C_\ell(T_\ell - T_o) - C_{\ell 1}(T_\ell - T_o)^2\ , \tag{2.7}$$

where $E_v$ and $E_\ell$ are the saturated internal energies at temperature $T_o$, and $C_v$, $C_\ell$, $C_{v1}$, and $C_{\ell 1}$ are constants chosen to fit steam table data in the temperature range of interest. For example, in the system of units g, cm, K, and ms, the values $E_v = 2.506 \times 10^4$, $E_\ell = 0.4174 \times 10^4$, $C_v = 6.67$, $C_\ell = 44.34$, $C_{v1} = 0.0302$, and $C_{\ell 1} = 0.0129$ are good approximations for temperatures up to about $T = 600$ K.

For equal-phase temperatures, the mixture temperature can be computed from the mixture internal energy as the solution of a quadratic equation

$$\rho I = \rho_v I_v + \rho_\ell I_\ell\ . \tag{2.8}$$

When the vapor is considered to be saturated, its temperature is determined from the mixture pressure by the relation

$$T_v = 255.2 + 117.8\ p^{0.223}, \tag{2.9}$$

where p is in bars. When the vapor temperature is known, one can easily compute the separate liquid and vapor internal energies and liquid temperature using Eqs. (2.7) and (2.8).

### C. Momentum and Mass Exchange

An equation of motion for the relative velocity can be derived from equations that describe a complete two fluid model.

$$\frac{\partial u_r}{\partial t} + \frac{1}{2}\frac{\partial}{\partial y} u_r\left[2u + \frac{u_r}{\rho}(\rho_\ell - \rho_v)\right] = \left(\frac{1}{\rho_\ell^{\,o}} - \frac{1}{\rho_v^{\,o}}\right)\frac{\partial p}{\partial y} - K\frac{\rho}{\rho_v\rho_\ell}u_r \ . \tag{2.10}$$

The quadratic term in $u_r$, on the left side of Eq. (2.10), is neglected because it generally is small compared to the linear term. This significantly simplifies the numerical solution. Assuming the vapor is a dispersed phase of small bubbles when $\theta$ is small, or the liquid is a dispersed phase of small droplets when $\theta$ is large, we can estimate K from the drag on an individual bubble (or droplet) times the number of bubbles (droplets) per unit volume N. The result is

$$K = \frac{\rho S}{8\theta_1}\left(C_d\ |u_r| + \frac{12\nu}{r_o}\right) , \tag{2.11}$$

where $\theta_1$ and $\nu$ are functions of $\theta$ and the kinematic viscosities of the phases,

$$\theta_1 = \theta,\ \nu = \nu_\ell(1 - \theta)^{-2.5} \quad \text{for } \theta \le 0.5$$

$$\theta_1 = 1 - \theta,\ \nu = \nu_v\theta^{-2.5} \quad \text{for } \theta > 0.5 \ .$$

Also, $C_d$ is a drag coefficient (generally of order unity) and S is the surface area per unit volume of bubbles (droplets) with mean radius $r_o$

$$S = \begin{cases} 3\theta/r_o & \text{for } \theta \le 1/2 \\ 3(1 - \theta)/r_o & \text{for } \theta > 1/2 \ . \end{cases} \tag{2.12}$$

The mean radius is related to the number density by the expressions

$$r_o = \left(\frac{3\theta}{4\pi N}\right)^{1/3} \quad \text{for} \quad \theta \le 1/2$$

$$r_o = \left[\frac{3(1 - \theta)}{4\pi N}\right]^{1/3} \quad \text{for} \quad \theta > 1/2 \ . \tag{2.13}$$

The bubble number N often is assumed to be a constant, independent of space and time. Because this is an approximation, it will not work when preferential nucleating sites exist or when significant bubble breakup or coalescence occurs. The following discussion on mass exchange describes how a locally variable N can sometimes be estimated in terms of a critical Weber number.

The form of the phase change model embodied in $\Gamma$ is crucial if nonequilibrium effects are to be predicted correctly. The model described here is still being developed and is not yet sophisticated enough for use as a predictive tool without some adjustment. Nevertheless, it has proved useful in numerous applications and is presented here to illustrate the types of considerations necessary in the development of such models.

If we define q as the interfacial heat flux, a simple energy balance shows that

$$\Gamma = \frac{qS}{\lambda} ,$$

where $\lambda$ is the latent heat of vaporization and S is related to the bubble radius, r, according to S = $3\theta/r$. The heat flux can be further defined as

$$q = k_\ell(T_\ell - T_s)/\ell ,$$

where $T_s$ is the saturation temperature and $k_\ell$ is the thermal conductivity of the liquid whose bulk temperature is $T_\ell$. The length $\ell$ characterizes the thickness of the thermal boundary layer over which the liquid temperature changes from its interior, bulk value $T_\ell$ to the value $T_s$ assumed to exist at the two-phase interface. Thus,

$$\Gamma = \frac{k_\ell(T_\ell - T_s)S}{\lambda\ell} . \tag{2.14}$$

For a single, nontranslating bubble growing in an infinite fluid region, Ref. 6 shows that $\ell = \ell_c$, where

$$\ell_c = r\left[\frac{6}{\pi}\frac{\rho_\ell^o}{\rho_v^o}\frac{C_\ell|T_\ell - T_s|}{\lambda}\right]^{-1} .$$

In this expression r is the instantaneous bubble radius, which we define below.

When the bubbles are translating with respect to the surrounding liquid with speed U, then $\ell = \ell_u$. Moalem and Sideman[7] give the general expression

$$\ell_u = r\left(\frac{\pi}{Re_b Pr}\right)^{\frac{1}{2}} ,$$

where $Re_b = 2rU\rho_\ell^o/\mu_\ell$ is the bubble Reynolds number, $Pr = C_\ell\mu_\ell/k_\ell$ is the liquid Prandtl number, and $\mu_\ell$ is the liquid shear viscosity. As the relative speed U increases, the length $\ell_u$ rapidly decreases below the value of $\ell_c$, which represents stripping away of the thermal boundary layer by relative flow. In an attempt to combine both of these effects, we have defined $\ell$ as the reciprocal average of these limiting characteristic lengths,

$$\frac{1}{\ell} = \frac{1}{\ell_c} + \frac{1}{\ell_u} . \tag{2.15}$$

Equation (2.14), with $\ell$ defined by the above equation, is a vapor generation rate that includes both finite heat conduction and relative velocity effects. However, the model still requires the definition of r and U.

If the number of bubbles per unit volume is known, we can calculate the mean bubble radius by Eq. (2.13) and use $r = r_o$. Unfortunately, the number of bubbles generally does not remain constant in the dynamic flow environment because bubbles larger than a certain size will break up. The maximum stable bubble radius, $r_w$, can be estimated in terms of a critical Weber number $W_c$,

$$r_w = \frac{\sigma W_c}{2\rho_\ell{}^o U^2} , \tag{2.16}$$

where $\sigma$ is the interfacial surface tension. The value of $W_c$ often is taken as 4 for turbulent flow conditions.[8] Thus, we define r as equal to the minimum of $r_o$ and $r_w$ and reserve N as an input parameter that defines the initial number of nucleating sites per unit volume (or more correctly, the minimum number of bubbles).

Finally, the relative speed U could be set equal to the magnitude of the average relative speed $|\underline{u}_r|$ between phases, but this would not account for local turbulent fluctuations that have been averaged out in the definition of $\underline{u}_r$. Fluctuations in $\underline{u}_r$ can locally strip away the individual bubble thermal boundary layers and break up large bubbles. To account for such local effects we define

$$U = |\underline{u}_r| + \beta |\underline{u}| , \tag{2.17}$$

where $\underline{u}$ is the mass averaged mixture velocity and $\beta$ is a parameter that accounts for turbulent fluctuations. We might expect $\beta$ to have a magnitude of 0.1 or less because large turbulent velocity fluctuations often have magnitudes as large as 10% of the mean velocities. In general, the best value of $\beta$ must be determined by comparison with experimental data.

Again we stress that the vapor generation rate described above is preliminary and must be critically tested against various situations before it can be recommended for general use. Nevertheless, this model does include, as special cases, the models used by many other investigators. Also, it has produced good results in several different applications.

## D. Flow Losses

Flow losses affect the momentum and energy of the flow through the terms $f_{vis}$ and $W_{vis}$ in Eqs. (2.3) and (2.4), respectively. The term $f_{vis}$ accounts for both distributed losses, such as pipe wall friction, and local losses that occur at sudden area changes

$$f_{vis} = - \frac{f}{R} \left[ \frac{\rho}{\rho_\ell{}^o} (1 - \psi) \Phi_{TP} \right]^2 \rho_\ell{}^o u^2 - \frac{f_1}{2} \rho_\ell{}^o u^2 . \tag{2.18}$$

The friction coefficient f depends on the relative roughness (k/R) and the Reynolds number $Re = 2uR/\nu_\ell$

$$f = a + b \, Re^{-c} , \tag{2.19}$$

where

$$a = 0.026 \, (k/2R)^{0.225} + 0.133 \, (k/2R) ,$$

$$b = 22.0 \, (k/2R)^{0.44} , \text{ and}$$

$$c = 1.62 \, (k/2R)^{0.134} ,$$

and R is the hydraulic radius. The quantity $\phi_{TP}$ is a two-phase friction multiplier

$$\phi_{TP}^2 = (1 - \theta)^{-1.75} ,$$

and $\psi$ accounts for the relative velocity effects

$$\psi = \rho_v[1 + (\rho - \rho_v)u_r/\rho u]^2/\rho .$$

The coefficient $f_1$ relates to the local losses and is given by

$$f_1 = f(L/2R)/\Delta y , \tag{2.20}$$

where (L/2R) is the number of hydraulic diameters of an equivalent straight channel and $\Delta y$ is the segment length over which the loss occurs. The hydraulic radius can be specified as different from the component's geometric radius to treat flow through components with noncircular cross sections and to model the effects of internal structure. By suitably specifying the hydraulic radius in subroutine DEMXC and the number of flow passages CFRS, a simple model can be made of a steam generator or core that consists of a bundle of many small flow passages. The latter quantity is used to multiply the friction factor determined by Eq. (2.19).

The value of $W_{vis}$ is determined from the rate of change of the fluid kinetic energy associated with the flow loss.

### E. Wall Heat Flux

To represent heat exchanges in a core or steam generator, or to account for heat transfer with pipe walls, the code has a variable designated WT(J) that may be used as a wall temperature. The locally added or removed rate of heat energy Q can be input for segments as QJS(JS) or defined in terms of WT(J) in subroutine WALLT reserved for this purpose. Because the specification of wall heat flux is problem dependent, WALLT has been left blank.

## III. NUMERICAL SOLUTION METHOD

Different schemes can be used to solve numerically the equations given in Sec. II, although each one will vary in accuracy, numerical stability, programming simplicity, flexibility, and computational efficiency. Unfortunately, these desirable traits are often mutually exclusive. For example, the use of implicit difference equations to achieve unconditional numerical stability can result in poor accuracy and generally requires more complex programming and more computer memory. Because different applications require different mixtures of the desirable features, the choice of an optimum solution algorithm rarely can be made. Thus, the choice of a numerical solution procedure generally requires a balance primarily between programming simplicity and the flexibility for future evolution vs stability, accuracy, and computational speed. Inevitably, the choice rests on the developer's experience and prejudices.

In the SOLA-LOOP code, we tried to keep the programming simple and to use a limited implicitness. In all cases, point relaxation methods rather than direct solvers were used for coupled sets of equations. Although point relaxation methods generally are recognized as simple, but inferior to direct methods for linear equation systems, this is not necessarily true for nonlinear equations where iterative methods are used. Point relaxation methods permit considerable latitude for adding new features, changing boundary conditions, varying time steps, and making other substantial changes in the basic code to adapt it to new applications.

Additionally, the code is written in a modular form consisting of numerous subroutines that isolate individual logical and physical processes. This structure makes the code particularly easy to modify and extend for new applications.

The numerical algorithms used naturally separate into two classes: those used to solve the basic flow equations in a single, one-dimensional component and those used to couple the components into a network.

### A. Mesh Construction for Components

The mesh for component I consists of JCEL(I) cells. In many components one typically uses a mesh of cells that have identical properties such as radius, length, etc. Sometimes a component may consist of a few uniform segments, but rarely are the cell properties in a component different in every cell. We took advantage of this typical mesh structure and built SOLA-LOOP so that its cells can be grouped into JSXI(I) segments having uniform properties. Thus, we do not have to store all the cell quantities for all the cells in the mesh. This generally leads to a considerable savings in storage and simplifies problem setups. If necessary, we can define different properties for every cell by making each cell a segment. The first two cells JB0 and JB1 and the last cell JT1 in the component are dummy or fictitious cells used to set boundary conditions and accomplish coupling of components (see Fig. 1). Two cells are needed at the beginning of a component to set boundary conditions because velocities are located at the cell boundaries. All other dependent variables are located at the cell centers. This staggered mesh arrangement is convenient for many of the finite difference approximations. Throughout this report we refer to the cell JB0 as the first or bottom cell and to the cell JT1 as the last or top cell in component I.

### B. Solution Algorithm for Components

A calculation cycle is broken down into four tasks. First, the momentum equation, Eq. (2.3), is advanced explicitly in subroutine TILDE using the previous cycle values for evaluating all contributions. Next, an iteration is made in subroutine PITERP to replace the pressure used in the

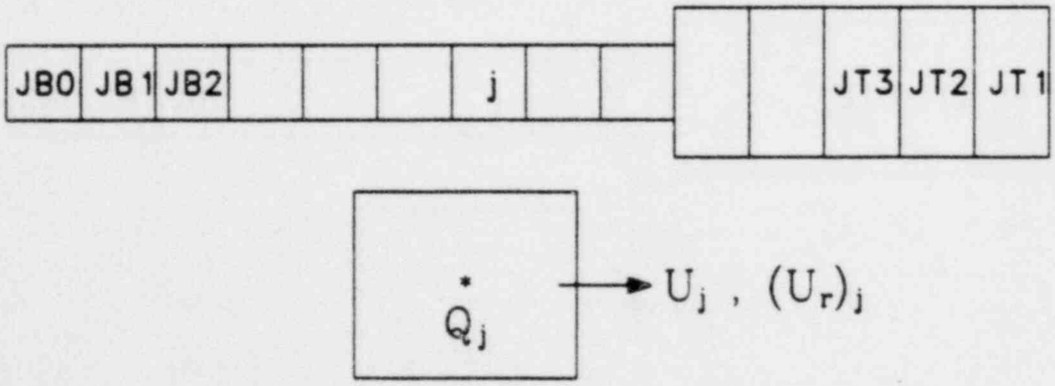

*Fig. 1.*

*Arrangment of fictitious cells at the bottom and top of a component and location of dependent variables in cell j.*

first task with advanced time values. An iteration is needed because the advanced pressures depend on the velocities being calculated. This part of the cycle contains the main implicitness of the numerical scheme. The pressure iteration permits sound waves to propagate more than one mesh cell per cycle. In fact, this scheme is a variant of the ICE technique,[9] which may be used for very low speed (incompressible) flows as well as for high-speed flows that contain shock waves and rarefactions. The third task in a cycle, performed in subroutine UPDATP, is to update all other dependent variables. Finally, the fourth task consists of data output (subroutine DIAG), time step controls (subroutine TIMCT), and bookkeeping operations (subroutine RESET).

For a purely explicit calculation, the iteration making up the second task may be omitted by setting the input number IMP equal to zero. When two or more components are coupled, one iteration pass is made through each component during task two. This is followed by updating the boundary conditions that define the coupling between components (in subroutines JCTPIP and BC) before the next iteration pass is started.

A simple flagging scheme may be used to omit iterations in selected components when such iterations are not needed. For example, every component would be considered during the first iteration; those components satisfying the convergence test would be flagged to indicate that they should be omitted on a successive iteration. If the boundary conditions of any component were changed significantly during the coupling calculations, the flag would have to be set to start the iterations again. Such variations are particularly simple to implement with the point relaxation method.

**1. Explicit Updating of Velocities.** Before introducing finite difference approximations for the momentum equation, Eq. (2.3), it is first written in the equivalent differential form,

$$\frac{\partial u}{\partial t} + u\,\frac{\partial u}{\partial y} + \frac{1}{\rho A}\,\frac{\partial}{\partial y}\left(\frac{A\rho_v\rho_\ell}{\rho}\,u_r^{\,2}\right) = -\,\frac{1}{\rho}\,\frac{\partial p}{\partial y} + g_y + \frac{1}{\rho}\,f_{vis}\,, \tag{3.1}$$

which is a carryover from previous codes in the SOLA series.[10] Its advantage is that $u^{n+1}$, rather than $(\rho u)^{n+1}$, is calculated directly. Its disadvantage is that it is not in conservation form, so we do not get rigorous conservation of momentum in the difference approximation. However, when percentage changes in dependent variables from one cell to the next are not large, the nonconservation form of the momentum equation should not cause any problems. Although it probably would be beneficial to monitor the total momentum of the system to check the accuracy of Eq. (3.1), this has not been included in this version of the code. The difference equation used to approximate Eq. (3.1) is

$$\tilde{u}_j = u_j + \delta t\left[-\,\mathrm{FUX} - \mathrm{FDU} + \frac{2(p_j - p_{j+1})}{\rho_{j+\frac{1}{2}}(\delta y_j + \delta y_{j+1})} + g_y + \left(\frac{1}{\rho}\;f_{vis}\right)_j\right]\;, \tag{3.2}$$

where $\rho_{j+1/2} = (\delta y_{j+1}\,\rho_j + \delta y_j\,\rho_{j+1})/(\delta y_j + \delta y_{j+1})$ and $\tilde{u}_j$ is the explicit estimate for $u_j^{n+1}$. Unless otherwise noted, the boundary value of cell-centered quantities is always defined by linear interpolation, as in $\rho_{j+1/2}$ used above. The indexing in Eq. (3.2) looks uncentered, but recall from Fig. 1 that $u_j$ refers to the velocity at the boundary between cells j and j + 1. All terms on the right side of Eq. (3.2) are evaluated in subroutine TILDE using time level n quantities. The convective fluxes are defined as

$$\mathrm{FUX} = \frac{1}{2}\Big|\,u_j\left[(u_j - u_{j-1})/\delta y_j + (u_{j+1} - u_j)/\delta y_{j+1}\right]$$

$$+\;\alpha|u_j|\left[(u_j - u_{j-1})/\delta y_j - (u_{j+1} - u_j)/\delta y_{j+1}\right]$$

and

$$\begin{aligned} \mathrm{FDU} = {} & \frac{1}{\rho_{j+\frac{1}{2}}(A_j + A_{j+1})(\delta y_j + \delta y_{j+1})} \left( A_{j+1} \left(\frac{\rho_v \rho_\ell}{\rho}\right)_{j+1} \left\{ \left[(u_r)_j + (u_r)_{j+1}\right]^2 \right.\right. \\ & \left. + \alpha \left|(u_r)_j + (u_r)_{j+1}\right| \left[(u_r)_j - (u_r)_{j+1}\right] \right\} - A_j \left(\frac{\rho_v \rho_\ell}{\rho}\right)_j \left\{ \left[(u_r)_{j-1} + (u_r)_j\right]^2 \right. \\ & \left.\left. - \alpha \left|(u_r)_{j-1} + (u_r)_j\right| \left[(u_r)_{j-1} - (u_r)_j\right] \right\} \right) . \end{aligned}$$

The parameter $\alpha$, shown in the convective fluxes, gives a variable amount of upstream differencing. When $\alpha$ is zero, the approximations reduce to the usual centered differenced form; however, this results in an unstable algorithm.[11] When $\alpha$ is unity, the approximations are the so-called donor cell or fully upstream (or upwind) difference expressions, which are stable provided fluid does not convect through more than one mesh cell in one time step. In general, numerical stability is expected (see Ref. 11) when $\alpha$ is chosen such that

$$\alpha > \max \left[ \left(\frac{u \delta t}{\delta y}\right) , \frac{\rho_v \rho_\ell u_r \delta t}{\rho^2 \delta y} \right] .$$

**2. Implicit Pressure Calculation.** In this part of the calculational cycle the n level pressures in Eq. (3.2) are replaced by approximations for the n + 1 level pressures. This is done for components in subroutine PITERP by solving for the pressure in each cell that satisfies the implicit equation

$$F = p - f(\bar{\rho}, \bar{\rho}_v, \bar{I}) = 0 , \tag{3.3}$$

where $f(\bar{\rho}, \bar{\rho}_v, \bar{I})$ is the equation of state that was evaluated using

$$\bar{\rho} = \rho^n/(1 + D) ,$$

$$\bar{\rho}_v = \rho_v{}^n/(1 + D) ,$$

and

$$\bar{I} = I^n - \frac{p^n}{\rho^n} D , \tag{3.4}$$

where D is an approximation to cell volume change per unit volume

$$D = \frac{\delta t}{A} \frac{\partial}{\partial y} Au \approx \frac{1}{A_j} \frac{\delta}{\delta y_j} \left(A_{j+\frac{1}{2}} u_j - A_{j-\frac{1}{2}} u_{j-1}\right) . \tag{3.5}$$

For the cell boundary areas, instead of a linearly interpolated value we prefer to use a combined geometric and arithmetic average

$$A_{j+\frac{1}{2}} = 2A_j A_{j+1}/(A_j + A_{j+1}) \tag{3.6}$$

because $A_{j+1/2}$ vanishes when either $A_j$ or $A_{j+1}$ is zero. The n + 1 level velocities must be used to evaluate D, that is,

$$u_j^{n+1} = \tilde{u}_j - \frac{2\delta t}{\rho_{j+\frac{1}{2}}(\delta y_j + \delta y_{j+1})} \left(p_{j+1} - p_{j+1}^n - p_j + p_j^n\right) . \tag{3.7}$$

Because the n + 1 level velocities depend on p, the implicit nature of Eq. (3.3) is obvious. The pressure that satisfies Eqs. (3.3)-(3.7) is not quite $p^{n+1}$ because convective fluxes are omitted from the estimates of the new densities $\bar{\rho}$, $\bar{\rho}_v$, and energy $\bar{I}$. The pressure would be equal to $p^{n+1}$ if the rest of the equations were in Lagrangian form. The difference is not significant, however, because for every cycle the iteration is always trying to drive p to its equation-of-state value. In this sense p is a stored variable and is not identically equal to the equation-of-state value unless the iterations are omitted and an explicit calculation is used. Note that densities and energies are not actually changed during the iteration, because Eq. (3.4) is used only to estimate the new values. To solve Eqs. (3.3)-(3.7) a local Newton-Raphson procedure is followed, in which an estimate is needed for $\partial F/\partial p$ in each cell. In SOLA-LOOP, estimates for these values are computed at the beginning of each cycle in subroutine RESET by a numerical differentiation and the values are stored. Once the iteration begins, new values are computed and stored after each iteration. In summary, the following steps are performed for a single cell j.

a. Compute D according to Eq. (3.5) using the most updated values of u from Eq. (3.7).
b. Compute $\bar{\rho}$, $\bar{\rho}_v$, and $\bar{I}$ from Eq. (3.4).
c. Evaluate the equation-of-state function and calculate $\delta p = -F/(\partial F/\partial p)$.
d. Replace $p_j$ with $p_j + \delta p$, and $u_j$ and $u_{j-1}$ with

$$u_j = u_j + \frac{2\delta t\delta p}{\rho_{j+\frac{1}{2}}(\delta y_j + \delta y_{j+1})}$$

and

$$u_{j-1} = u_{j-1} - \frac{2\delta t\delta p}{\rho_{j-\frac{1}{2}}(\delta y_{j-1} + \delta y_j)} .$$

Continue this iteration process until all cells satisfy the convergence test

$$\left| \frac{p - f(\bar{\rho}, \bar{\rho}_v, \bar{I})}{p + f(\bar{\rho}, \bar{\rho}_v, \bar{I})} \right| < \varepsilon ,$$

where $\epsilon$ is typically equal to 0.001.

**3. Updating of Remaining Variables.** After completing the implicit portion of the cycle, new time values for the remaining variables are readily computed in subroutine UPDATP. The mixture density changes only by convection,

$$\rho_j^{n+1} = \rho_j - \frac{\delta t}{A_j \delta y_j} \left[A_{j+\frac{1}{2}} (\rho u)_{j+\frac{1}{2}} - A_{j-\frac{1}{2}} (\rho u)_{j-\frac{1}{2}}\right], \tag{3.8}$$

where

$$(\rho u)_{j+\frac{1}{2}} = \frac{1}{2}\left[u_j(\rho_j + \rho_{j+1}) + \alpha|u_j|\ (\rho_j - \rho_{j+1})\right]$$

$$(\rho u)_{j-\frac{1}{2}} = \frac{1}{2}\left[u_{j-1}(\rho_{j-1} + \rho_j) + \alpha|u_{j-1}|\ (\rho_{j-1} - \rho_j)\right] .$$

Quantities on the right side of Eq. (3.8) are evaluated using n level values for $\rho$ and the available n + 1 level values of u.

The mixture energy equation then is approximated by

$$I_j^{n+1} = \frac{1}{\rho_j^{n+1}}\Bigg(\rho_j I_j + \delta t\ \Big\{- \mathrm{FUV} - \mathrm{FUL} - \mathrm{FWK} + \frac{1}{4} K_j\ \left[(u_r)_j + (u_r)_{j-1}\right]^2 + (W_{vis})_j + Q_j\Big\}\Bigg) , \tag{3.9}$$

where

$$\mathrm{FUV} = \frac{1}{A_j \delta y_j}\left[A_{j+\frac{1}{2}}\left(\rho_v I_v u_v\right)_{j+\frac{1}{2}} - A_{j-\frac{1}{2}}\left(\rho_v I_v u_v\right)_{j-\frac{1}{2}}\right]$$

$$\mathrm{FUL} = \frac{1}{A_j \delta y_j}\left[A_{j+\frac{1}{2}}\left(\rho_\ell I_\ell u_\ell\right)_{j+\frac{1}{2}} - A_{j-\frac{1}{2}}\left(\rho_\ell I_\ell u_\ell\right)_{j-\frac{1}{2}}\right] .$$

This formulation separates the convective fluxes into vapor and liquid contributions where

$$\left(\rho_v I_v u_v\right)_{j+\frac{1}{2}} = \frac{1}{2}\Big\{(u_v)_j\ \left[(\rho_v I_v)_j + (\rho_v I_v)_{j+1}\right] + \alpha|u_v|\ \left[(\rho_v I_v)_j - (\rho_v I_v)_{j+1}\right]\Big\} ,$$

and similarly for $(\rho_\ell I_\ell u_\ell)_{j+1/2}$, where the vapor and liquid velocities used in these expressions are defined as

$$(u_v)_j = u_j + (\rho_\ell)_{j+\frac{1}{2}}\ (u_r)_j/\rho_{j+\frac{1}{2}}$$

$$(u_\ell)_j = u_j - (\rho_v)_{j+\frac{1}{2}}\ (u_r)_j/\rho_{j+\frac{1}{2}} . \tag{3.10}$$

Corresponding expressions for the j − 1/2 boundary are obtained by replacing j with j − 1 in the above definitions.

The pressure work term in Eq. (3.9) is approximated as

$$\mathrm{FWK} = \frac{p_j}{A_j \delta y_j} \left( A_{j+\frac{1}{2}} \left\{ u_j + \left[ \frac{(\rho_v)_{j+\frac{1}{2}} - \rho_{j+\frac{1}{2}} + \rho_\ell^{\,o}}{\rho_\ell^{\,o}} \right] - \frac{(\rho_v)_{j+\frac{1}{2}}}{\rho_{j+\frac{1}{2}}} \quad (u_r)_j \right\} \right.$$

$$\left. - A_{j-\frac{1}{2}} \left\{ u_{j-1} + \left[ \frac{(\rho_v)_{j-\frac{1}{2}} - \rho_{j-\frac{1}{2}} + \rho_\ell^{\,o}}{\rho_\ell^{\,o}} - \frac{(\rho_v)_{j-\frac{1}{2}}}{\rho_{j-\frac{1}{2}}} \right] (u_r)_{j-1} \right\} \right) .$$

Quantities on the right side of Eq. (3.9) are evaluated using n level values for $\rho$, $\rho_v$, and I, whereas n + 1 level values are used for $u_j$, $p_j$, and the overall divisor $\rho_j$. Finally, the vapor density is updated as

$$(\rho_v)_j^{n+1} = (\rho_v)_j + \delta t \; (- \mathrm{FRU} + \Gamma_j) \; , \tag{3.11}$$

where

$$\mathrm{FRU} = \frac{1}{A_j \delta y_j} \left[ A_{j+\frac{1}{2}} \left( \rho_v u_v \right)_{j+\frac{1}{2}} - A_{j-\frac{1}{2}} \left( \rho_v u_v \right)_{j-\frac{1}{2}} \right]$$

and

$$\rho_v u_v \,_{j+\frac{1}{2}} = \frac{1}{2} \left\{ (u_v)_j \left[ (\rho_v)_j + (\rho_v)_{j+1} \right] + \alpha |u_v|_j \left[ (\rho_v)_j \quad - (\rho_v)_{j+1} \right] \right\} .$$

The j − 1/2 boundary flux is obtained by replacing j with j − 1 in the above expression.

Care must be taken when approximating the vapor source term Γ. When the mixture is not at equilibrium and the relaxation rate is fast, Γ can be large and Eq. (3.11) may be numerically unstable. To avoid this, the densities in Γ should be evaluated at level n + 1. For general formulations of Γ, use of an iterative technique to solve Eq. (3.11) generally is necessary. Subroutine UPDATP provides such an iteration for the Γ given by Eq. (2.14). There is, however, another more serious problem that can arise when phase transitions are important. Because the effect of Γ is included at the end of a calculation cycle, its influence on the pressure, and hence the dynamics, is not accounted for in the implicit pressure iteration. Therefore, some inaccuracies can be introduced in the propagation of compression and rarefaction waves when a large phase change occurs during a single time step. A large phase change also may drive the equation-of-state pressure far from the value obtained in the pressure iteration; excessive iterations, therefore, may be required to solve the implicit equation in the next time cycle. In extreme cases, the iteration may not even converge. This problem can be eliminated by using sufficiently small time increments, δt, although this sometimes leads to long computing times. A better solution, used recently in the multidimensional code K-FIX (see Ref. 2), is to incorporate Γ into the implicit portion of the cycle. Basically, the idea is to include Γ in Eq. (3.4) for the estimated new time vapor density. Because this more complicated formulation is not in SOLA-LOOP, the user should check his results for time-step dependence (accuracy) by performing a smaller time step calculation when necessary.

Thermodynamic equilibrium calculations can be achieved by using a large phase change rate, or by replacing the vapor density equation with a calculation of the saturated vapor density and using an equilibrium equation of state. The latter procedure often is preferable because it effectively puts $\Gamma$ into the pressure iteration.

### C. Component Boundary Conditions

Various types of boundary conditions may be used at the ends of the one-dimensional component meshes. Prescribed velocities or pressures, together with densities and temperatures, may be used to represent inlet and exit conditions. For example, a guillotine break in a reactor system pipe can be represented by assigning the ambient pressure in the containment structure to the end of the pipe. The closed end of a pipe or a closed valve is a common boundary condition requiring a prescribed velocity of zero. Other boundary conditions that can be specified are uniform or gradient-free outflow and periodic boundaries in which the bottom and top of a component are joined together. As described in Sec. II, abrupt area changes in axially aligned pipes can be approximated through the variable area terms in the equations of motion. However, when the pipes are joined together in elbows or when more than two flow regions are coupled at a common junction, special coupling equations must be solved to get the appropriate boundary conditions for each component.

All component boundary conditions are located in subroutines BC and JCTPIP, which make changes or additions an easy task. The conditions available in these subroutines will be illustrated by considering only the JB0 (bottom) end of a component. The $j$ = JT1 end conditions are analogous.

1. **Prescribed Velocity.** A prescribed zero velocity for u(JB1) at the bottom of component I is obtained by setting LBØT(I) = 1 or 2. A nonzero or time-varying prescription can be obtained easily by a modification to subroutine BC. When u(JB1) is not zero, one must also define values for the pressure PB(M), the void fraction THB(M), and the temperature TEMB(M) as boundary data set M, and assign MBØT(I) = M.

2. **Uniform Outflow.** A uniform or gradient-free outflow boundary condition at the bottom boundary of component I can be specified by setting LBØT(I) = 3. Use of this boundary condition requires the additional specification of PB(M), THB(M), TEMB(M), and the assignment of MBØT(I) = M.

3. **Periodic Coupling.** The top and bottom boundaries of component I can be joined together through the periodic boundary condition specified by setting LBOT(I) = 4.

4. **Prescribed Pressure.** When pressure is prescribed at the JB0 end it must be for cell JB1. The u(JB0) is set equal to u(JB1) so that fluid can flow freely into or out of the specified pressure region. Values must be specified for PB(M), THB(M), and TEMB(M) with MBØT(I) = M. This boundary condition is activated by setting LBØT(I) = 5.

5. **Coupling Two or More Components.** This boundary condition is less well defined than the preceding ones because the multidimensional effects that occur at a junction, where several one-dimensional regions join, cannot be included in detail. The following description outlines the formulation now available in SOLA-LOOP to couple components into a network.

### D. Mesh Construction for a Junction Cell

A junction cell is a three-dimensional rectangular control volume that can join up to four components in a plane, as shown in Fig. 2. The planar dimensions of junction cell K are $\delta x_k$ and $\delta y_k$. Flow in the third dimension is not permitted. This restriction allows the use of a two-dimensional

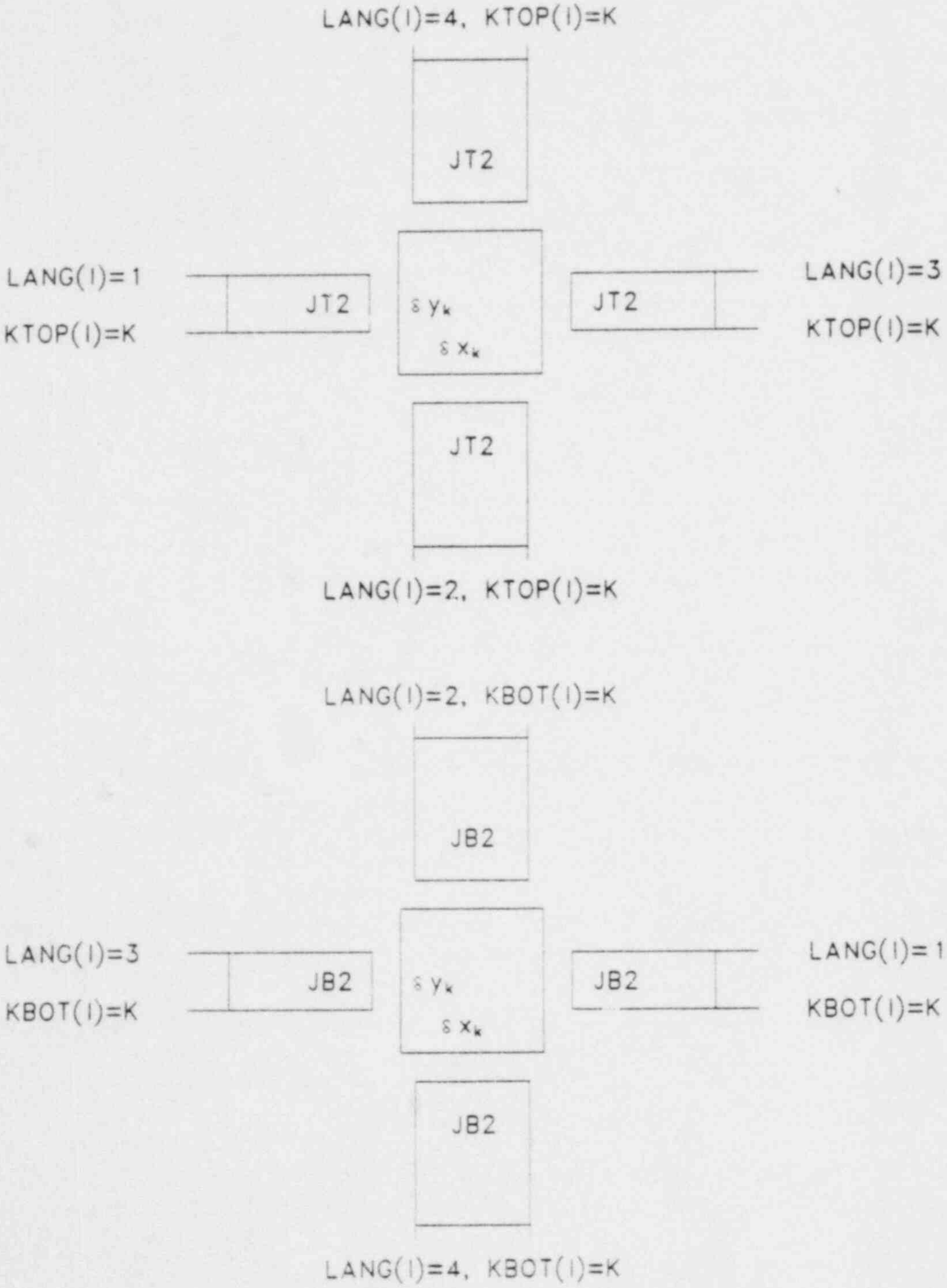

*Fig. 2.*
*Junction cell orientation for component coupling.*

rather than a three-dimensional solution algorithm for the junction cell. A one-dimensional component may be attached perpendicular to any of the four faces of the junction, but only one component is allowed per face and components opposite one another must have the same cross-sectional areas at the junction. If no components are attached to a given junction face, it is treated as a rigid wall boundary. The quantities LANG(I), KBØT(I), and KTØP(I) (Fig. 2) provide the information that identifies which component ends are connected to which junction faces.

The coupling of a junction cell and two or more components is accomplished by making the junction cell the neighboring cell to the JB2 or JT2 cells that belong to the adjoining ends of the components. The JB1 or JT1 fictitious cells at the adjoining ends overlap the junction cell. The centers of the overlapping cells coincide with the center of the junction cell so that cell-centered quantities in the junction cell and cells JB1 and JT1 are identical. Furthermore, the boundaries of the overlapping cells on which velocities are stored coincide with the boundaries of the junction cell so that velocities on the common boundaries are identical.

## E. Solution Algorithm for a Junction Cell

Junction cell values are calculated using the two-dimensional analogues of the one-dimensional equations of motion, Eqs. (2.1)-(2.4). Likewise, the difference equations for junction cells are the two-dimensional counterparts of the one-dimensional finite difference equations. The only instance in which the two-dimensionality is not maintained is in the cross convection of momentum. Because a one-dimensional component can have momentum only in its axial direction, we cannot allow momentum entering the right or left side of a junction to be convected into components connected to the top or bottom of the junction.

The planar dimensions $\delta x_k$ and $\delta y_k$ of junction cell K are calculated automatically by the code from the adjoining component cross-sectional areas and cell sizes. The junction cell size is calculated to provide comparable resolution to that in the adjoining components. To ensure the coincidence between the junction and fictitious cell centers and boundaries, an additional segment is added automatically to the end of the component that adjoins the junction cell. This additional segment contains the required one or two fictitious cells, depending on which end of the component it is added to, and one real cell. Inclusion of this real cell, however, slightly increases the component's overall length. If this is of concern, one can obtain a better approximation of the actual length by specifying input data that are one cell short of the desired length (two cells short if junctions exist at both ends of the components).

The subroutines used to update the junction variables are similar to those used to update components. The implicit pressure iteration is performed in PITERJ, and the remaining variables are updated in UPDATJ. Before entering the junction part of a calculation, junction boundary conditions are computed from the attached components in the subroutine PIPJCT. After the junction computations are finished, their new values are used to get new boundary conditions for the attached components by subroutine JCTPIP.

## F. Variable Time Steps and Subcycling

Network systems often contain low-speed flow with slowly varying properties in one region and high-speed flow or flow that requires a finely detailed description in another region. The variable time stepping and subcycling provisions in SOLA-LOOP are designed specifically for such systems to provide efficient and accurate calculations.

SOLA-LOOP contains provisions that allow the use of different time steps for integration in each component and junction cell in the network. Because the time steps can be significantly different in the various components, one component may be integrated several time steps to keep pace with one time integration in another component. This subcycling feature provides the time integration accuracy only where it is specifically needed, which significantly enhances the overall computational efficiency. The time steps are determined by numerical stability requirements and other user-specified conditions.

For numerical stability the time step in each component and junction is limited by the flux criterion that $u\delta t/\delta y(\delta \mathbf{x}) \leq 1.0/\mathrm{STABC}$, where typically STABC = 4. The time steps determined according to this criterion in subroutine TIMCT are then increased or decreased by 1%. The direction of this adjustment is determined by the relative ease of the previous time integration of the system. If fewer than five system iterations were required, the time steps are increased; otherwise, they are decreased. The time step for the system generally is closely related to the maximum time step determined for the components and junctions. The system time step DELT0 is not a computational time step, but rather provides a time level toward which all the components and junctions are integrated simultaneously. The system time step is increased or decreased by 1%, based on the relative ease of the previous system integration and the number of subcycles used. The system time step is increased if the number of subcycles is fewer than that specified as NSUBDT, otherwise it is decreased.

## IV. INPUT DATA, COMMON VARIABLES, AND SUBROUTINES AND FORTRAN FUNCTIONS

The input data, COMMON variables, and subroutines and FORTRAN functions in SOLA-LOOP are listed and described in this section. The input quantities are tabulated and defined separately; they also appear in the COMMON variable lists and are identified there simply as an input quantity. The descriptions of the input and COMMON variables, although brief, hopefully will assist the user in relating the methodology, described in Sec. III, to its implementation in the code.

### A. Input Data

| Default Value | FORTRAN Symbol | Algebraic Symbol | Definition |
|---|---|---|---|
| 1.0 | ALPHA | $\alpha$ | Parameter that determines the amount of upstream differencing in the convective flux terms. Equal to one gives full donor cell differencing. |
| $1.234 \times 10^4$ | ASQ | $a^2$ | Square of the speed of sound for the liquid phase. |
| $10^4$ | BNUM | N | Representative bubble (or droplet) number density per cubic centimeter used in phase change and interfacial friction model. |

| Default Value | FORTRAN Symbol | Algebraic Symbol | Definition |
|---|---|---|---|
| 0.50 | CDG | $C_d$ | Drag coefficient used in the interfacial friction model. |
| 0.0 | CFRL(I,IS) | $f_l$ | Local flow loss coefficient for component I, segment IS [see Eqs. (2.18) and (2.20)]. |
| 1.0 | CFRS(I,IS) | --- | Number of flow passages through component I, segment IS. |
| 44.34 | CHL | $C_\ell$ | Coefficient of the linear term in the liquid internal energy function. |
| 0.0129 | CHL1 | $C_{\ell 1}$ | Coefficient of the quadratic term in the liquid internal energy function. |
| 6.67 | CHV | $C_v$ | Similar to CHL but for the vapor energy function. |
| 0.0302 | CHV1 | $C_{v1}$ | Similar to CHL1 but for the vapor energy function. |
| 100.0 | DELSTP | --- | Time interval in milliseconds between mass flux prints. |
| $1.0 \times 10^{-4}$ | DELT | $\delta t$ | Starting time step for the calculation. |
| 0.0 | DFVEL | --- | Program control parameter that determines whether the relative velocity is to be calculated (DFVEL = 1.) or set to zero (DFVEL = 0.). |
| 0.0 | DPRST | --- | Print delay interval, output begins at time T = TSTART + DPRST. |
| 500.0 | DTWPRT | --- | Time interval between successive prints (subroutine DIAG). |
| 0.0 | DXS(I,IS) | $\delta x$ | Radius of component I, segment IS. |
| 0.0 | DYS(I,IS) | $\delta y$ | Cell length in component I, segment IS. |

| Default Value | FORTRAN Symbol | Algebraic Symbol | Definition |
|---|---|---|---|
| $4.174 \times 10^8$ | ECL | $E_\ell$ | Constant in the liquid internal energy function. |
| $2.506 \times 10^4$ | ECV | $E_v$ | Similar to ECL but for the vapor energy function. |
| $1.6 \times 10^{-6}$ | EDL | --- | Ratio of the thermal conductivity to the specific heat for the liquid. |
| $1.6 \times 10^{-7}$ | EDV | --- | Ratio of the thermal conductivity to the specific heat for the vapor. |
| $1.76 \times 10^4$ | ELHT | $\lambda$ | Latent heat of vaporization. |
| 0.001 | EPSI | $\epsilon$ | Pressure iteration convergence test parameter. |
| 1.0 | ETEM | --- | Liquid and vapor phases are maintained at equal temperatures if ETEM = 1.0; if ETEM = 0.0, the vapor temperature is maintained equal to the saturation temperature at the local pressure. |
| 0.07 | GAM1 | $(\gamma-1)$ | Parameter in the equation of state. |
| 0.0 | GYS(I,IS) | $g_y$ | Component of gravitational acceleration in component I, segment IS. |
| 1 | IDXS(I,IS) | --- | IDXS(I,IS) = 1 when the hydraulic radius and the geometric radius DXS are the same in component I, segment IS. Otherwise, IDXS (I,IS) = 2 and the user must furnish the appropriate hydraulic radius in function DEMXC. |
| 1.0 | IMP | --- | Program control parameter whose value determines whether an implicit (IMP = 1) or explicit (IMP = 0) solution method is used. |

| Default Value | FORTRAN Symbol | Algebraic Symbol | Definition |
|---|---|---|---|
| 1 | JMHI(I) | --- | Determines the index JMH of the last cell whose edge quantities are to be computed in component I. JMH = JT1 − JMHI(I). |
| 1 | JMLI(I) | --- | Determines the index JML of the first cell whose edge quantities are to be computed in component I. JML = JB0 + JMLI(I). |
| 1 | JPBI(I) | --- | Determines the index JPB of the first cell whose centered quantities are to be computed in component I. JPB = JB1 + JPBI(I). |
| 1 | JPTI(I) | --- | Determines the index JPT of the last cell whose centered quantities are to be computed in component I. JPT = JT1 − JPTI(I). |
| 0 | KBØT(I) | --- | Number of junction connected to the bottom of pipe I. A zero value indicates an isolated end. |
| 0 | KTØP(I) | --- | Number of junction connected to the top of pipe I. A zero value indicates an isolated end. |
| 0 | LANG(I) | --- | Parameter that indicates the orientation of component I (see Fig. 2). |
| 0 | LBØT(I) | --- | Parameter that indicates the type of boundary conditions to be applied at the bottom of component I when it is an isolated end (see Sec. III C). |
| 0 | LTØP(I) | --- | Similar to LBØT(I) except for an isolated top end. |
| 0 | MBØT(I) | --- | Parameter that specifies the boundary data set (if applicable) for the bottom end of component I. |

| Default Value | FORTRAN Symbol | Algebraic Symbol | Definition |
|---|---|---|---|
| 0 | MTØP(I) | --- | Similar to MBØT(I) except for the top end of component I. |
| 0 | NJS(I,IS) | --- | Number of real cells in component I, segment IS. NJS values adjusted in subroutine INPUT automatically account for fictitious cells, as required. |
| 1 | NSUBDT | --- | Desired number of subcycles (see Sec. III.F). |
| 1.0 | ØMG | --- | Relaxation parameter for the calculation of $\delta p$ in the pressure iteration. A value of OMG = 1.7 often improves the rate of convergence for low Mach number flows. |
| 1.0 | PHCH | --- | Parameter whose value determines whether phase change is computed (PHCH = 1) or omitted (PHCH = 0). |
| 0.0 | PB(M) | --- | Pressure associated with boundary data set M. |
| 0.0 | PI(I) | --- | Initial pressure in component I. |
| 0.0 | QJS(I,IS) | Q | Rate of change of internal energy density in component I, segment IS caused by heat addition. |
| 0.0 | RG | k | Pipe wall roughness. |
| 0.958 | RL | $\rho_\ell^\circ$ | Microscopic density of the liquid. |
| 4.0 | STABC | --- | Stability control constant in time step determination, $\Delta t \le 1/\mathrm{STABC}\ \delta y/u$. |
| $10^{20}$ | TBRA(N) | --- | Time at which calculation is to be interrupted and input data changed. |

| Default Value | FORTRAN Symbol | Algebraic Symbol | Definition |
|---|---|---|---|
| 373.0 | TC | $T_o$ | Reference temperature for liquid and vapor internal energy functions. |
| 0.0 | TEMB(M) | --- | Temperature associated with boundary data set M. |
| 0.0 | TEMI(I) | --- | Initial temperature for component I. |
| 0.0 | THB(M) | --- | Void fraction associated with boundary data set M. |
| 0.001 | THC | $\theta_c$ | Void fraction below which the fluid is treated as pure liquid. |
| 0.0 | THI(I) | --- | Initial void fraction for component I. |
| 0.0 | TSTART | --- | Initial time for the problem. |
| $10^4$ | TWFIN | --- | Time when the problem is complete. |
| 0.0 | TWSTP | --- | Time at which mass flux prints begin. |
| 0.0 | VIS(I,IS) | --- | Initial velocity in component I, segment IS. |
| $3.0 \times 10^{-6}$ | VISL | $\nu_\ell$ | Kinematic viscosity of the liquid. |
| $2 \times 10^{-4}$ | VISV | $\nu_v$ | Kinematic viscosity of the vapor. |

## B. COMMON Variables

| FORTRAN Symbol | Algebraic Symbol | Definition |
|---|---|---|
| ALPHA | $\alpha$ | Input quantity |
| ASQ | $a^2$ | Input quantity |
| BNUM | N | Input quantity |
| CDG | $C_d$ | Input quantity |

| FORTRAN Symbol | Algebraic Symbol | Definition |
|---|---|---|
| CHL | $C_\ell$ | Input quantity |
| CHL1 | $C_{\ell 1}$ | Input quantity |
| CHV | $C_v$ | Input quantity |
| CHV1 | $C_{v1}$ | Input quantity |
| CPI | $\pi$ | Constant $\pi$ |
| CYCLE | --- | Counter advanced by one for each minimum time step calculated. |
| DELSTP | --- | Input quantity |
| DELT | $\delta t$ | Input quantity |
| DELT0 | --- | System time step—usually equal to NSUBDT × DTMN. |
| DFVEL | --- | Input quantity |
| DPRST | --- | Input quantity |
| DTMN | --- | Smallest time step for components and junctions. |
| DTWPRT | --- | Input quantity |
| ECL | $E_\ell$ | Input quantity |
| ECV | $E_v$ | Input quantity |
| EDL | --- | Input quantity |
| EDV | --- | Input quantity |
| ELHT | $\lambda$ | Input quantity |
| EPSI | $\epsilon$ | Input quantity |
| ETEM | --- | Input quantity |
| FLG | --- | Iteration logic control flag. |
| GAM1 | $(\gamma\text{-}1)$ | Input quantity |
| IBREAK | --- | Counter for number of interrupts to change input data. |
| IL | --- | Total number of components. |
| IMP | --- | Input data |
| ISTEP | --- | Counter for number of subcycles to advance solution one system time step. |
| ITER | --- | Counter for number of iterations between junctions and components to achieve convergence during one system time step. |
| JB0 | --- | Index of first fictitious cell at bottom of a component. |
| JB1 | --- | Index of second fictitious cell at bottom of a component. |
| JB2 | --- | Index of first real cell at bottom of a component. |
| JMH | --- | Index of last cell whose edge quantities are computed for a given component. |
| JML | --- | Index of first cell whose edge quantities are computed for a given component. |
| JPB | --- | Index of first cell whose centered quantities are computed for a given component. |

| FORTRAN Symbol | Algebraic Symbol | Definition |
|---|---|---|
| JPT | --- | Index of last cell whose centered quantities are computed for a given component. |
| JT1 | --- | Index of fictitious cell at top of a component. |
| JT2 | --- | Index of last real cell at top of a component. |
| KDØ | --- | Index for adjacent cell's edge quantities when top of component I is coupled to the junction and LANG(I) = 1, or when bottom of component I is coupled to the junction and LANG(I) = 3. |
| KMØ | --- | Index for adjacent cell's centered quantities when top of component I is coupled to the junction and LANG(I) = 1, or when bottom of component I is coupled to the junction and LANG(I) = 3. |
| KØD | --- | Index for adjacent cell's edge quantities when top of component I is coupled to the junction and LANG(I) = 2, or when bottom of component I is coupled to the junction and LANG(I) = 4. |
| KØM | --- | Index for adjacent cell's centered quantities when top of component I is coupled to the junction and LANG(I) = 2, or when bottom of component I is coupled to the junction and LANG(I) = 4. |
| KØØ | --- | Index for cell's centered quantities in the junction cell. |
| KØP | --- | Index for adjacent cell's centered quantities when top of component I is coupled to the junction and LANG(I) = 4, or when bottom of component I is coupled to the junction and LANG(I) = 2. |
| KØU | --- | Index for adjacent cell's edge quantities when top of component I is coupled to the junction and LANG(I) = 4, or when bottom of component I is coupled to the junction and LANG(I) = 2. |
| KPØ | --- | Index for adjacent cell's centered quantities when top of component I is coupled to the junction and LANG(I) = 3, or when bottom of component I is coupled to the junction and LANG(I) = 1. |
| KL | --- | Total number of junctions. |
| KTØU | --- | Logic control parameter; KTØU = 1 if there are no junction cells; otherwise, KTØU = 2. |
| KUØ | --- | Index for adjacent cell's edge quantities when top of component I is coupled to the junction and LANG(I) = 3, or when bottom of component I is coupled to the junction and LANG(I) = 1. |
| ML | --- | Total number of boundary data sets. |
| NSUBDT | --- | Input quantity |
| ØMG | --- | Input quantity |

| FORTRAN Symbol | Algebraic Symbol | Definition |
|---|---|---|
| PHCH | --- | Input quantity |
| PNV | --- | $4/3 \pi N$ |
| PRST | --- | Input quantity |
| RG | $k$ | Input quantity |
| R0L | $\rho_\ell^\circ$ | Input quantity |
| SDTC | --- | Internal time step control parameter similar to STABC. |
| T | --- | Problem time |
| TC | $T_\circ$ | Input quantity |
| THC | $\theta_c$ | Input quantity |
| THC1 | --- | $1.0 - \theta_c$ |
| TIMA | --- | Advanced system time toward which time step controller attempts to bring all junction and component times. |
| TSTART | --- | Input quantity |
| TWFIN | --- | Input quantity |
| TWPRT | --- | Problem time for next print. |
| TWSTP | --- | Input quantity |
| VISL | $\nu_\ell$ | Input quantity |
| VISV | $\nu_v$ | Input quantity |
| BETA(J) | $(\partial F/\partial p)_j^{-1}$ | Recriprocal derivative of pressure function given by Eq. (3.3). |
| BETAK(K) | $(\partial F/\partial p)_k^{-1}$ | Similar to BETA(J) but for junctions. |
| DTIM(I) | $\delta t_I$ | Time step for component I. |
| DTIMK(K) | $\delta t_k$ | Time step for junction K. |
| DXK(K) | $\delta x_k$ | Dimension of junction cell K. |
| DXKC(K) | --- | Temporary storage for value of DXK(K). |
| DYK(K) | $\delta y_k$ | Dimension of junction cell K. |
| DYKC(K) | --- | Temporary storage for values of DYK(K). |
| E(J) | $I_j^{n+1}$ | Time n + 1 specific internal energy for component cells. |
| EB(M) | --- | Specific internal energy for boundary data set M. |
| EINC(I) | --- | Initial specific internal energy for component I. |
| EK(K) | $I_k^{n+1}$ | Time n + 1 specific internal energy for junction K. |
| EKC(K) | --- | Initial specific internal energy for junction K. |
| EN(J) | $I_j^n$ | Time n specific internal energy for component cells. |
| ENK(K) | $I_k^n$ | Time n specific internal energy for junction K. |
| GAM(K) | --- | Geometry dependent computational parameter for junctions. |
| IB0T(I) | --- | Logic control parameter for bottom end of component I; IB0T(I) = 1 for an isolated end and 2 if end is coupled to a junction. |
| IT0P(I) | --- | Similar to IB0T(I) but for top end of component I. |
| JCEL(I) | --- | Number of cells in component I, including the three fictitious cells. |

| FORTRAN Symbol | Algebraic Symbol | Definition |
|---|---|---|
| JMHI(I) | --- | Input quantity |
| JMLI(I) | --- | Input quantity |
| JPBI(I) | --- | Input quantity |
| JPTI(I) | --- | Input quantity |
| JREF(I) | --- | Reference value used to calculate value of JB0 for component I. |
| JSF(J) | --- | Single subscript used to obtain doubly subscripted quantities. |
| JSXI(I) | --- | Number of segments in component I. |
| KB0T(I) | --- | Input quantity |
| KT0P(K) | --- | Input quantity |
| LANG(I) | --- | Input quantity |
| LB0T(I) | --- | Input quantity |
| LT0P(I) | --- | Input quantity |
| MB0T(I) | --- | Input quantity |
| MT0P(I) | --- | Input quantity |
| P(J) | $p_J$ | Pressure for component cell J. |
| PB(M) | --- | Input quantity |
| PK(N) | --- | Pressure (five per junction) associated with junction K. |
| PI(I) | --- | Input quantity |
| R0(J) | $\rho_J^{n+1}$ | Time n + 1 mixture density for component cell J. |
| R0B(M) | --- | Mixture density for boundary data set M. |
| R0INC(I) | --- | Initial mixture density for cells in component I. |
| R0K(N) | --- | Time n + 1 mixture densities (five per junction) associated with junction K. |
| R0KC(K) | --- | Temporary storage for initial mixture density in junction cell K. |
| R0N(J) | $\rho_J^n$ | Time n mixture density for component cell J. |
| R0NK(N) | $\rho_K^n$ | Time n mixture densities (five per junction) associated with junction cell K. |
| RV(J) | $(\rho_v)_J^{n+1}$ | Time n + 1 macroscopic vapor density for component J. |
| RVB(M) | --- | Macroscopic vapor density for boundary data set M. |
| RVINC(I) | --- | Initial macroscopic vapor density for cells in component I. |
| RVK(N) | $(\rho_v)_K^{n+1}$ | Time n + 1 macroscopic vapor densities (five per junction) associated with junction cell K. |
| RVKC(K) | --- | Temporary storage for initial macroscopic vapor densities in junction cell K. |
| RVN(J) | $(\rho_v)_J^n$ | Time n macroscopic vapor density for component cell J. |

| FORTRAN Symbol | Algebraic Symbol | Definition |
|---|---|---|
| RVNK(N) | --- | Time n macroscopic vapor densities (five per junction) associated with junction cell K. |
| TBRA(I) | --- | Input quantity |
| TEMB(M) | --- | Input quantity |
| TEMI(I) | --- | Input quantity |
| TEMK(K) | --- | Input quantity |
| THB(M) | --- | Input quantity |
| THI(I) | --- | Input quantity |
| THK(K) | --- | Input quantity |
| TIM(I) | --- | Time level for component I. |
| TIMK(K) | --- | Time level for junction K. |
| V(J) | $u_j^{n+1}$ | Time n + 1 center of mass velocity for component cell J. |
| VD(J) | $(u_r)_j^{n+1}$ | Time n + 1 relative velocity for component cell J. |
| VDK(N) | --- | Time n + 1 relative velocities (four per junction) associated with junction cell K. |
| VK(N) | --- | Time n + 1 center of mass velocities (four per junction) associated with junction cell K. |
| VN(J) | $u_j^n$ | Time n center of mass velocity for component cell J. |
| VNK(N) | --- | Time n center of mass velocities (four per junction) associated with junction cell K. |
| VP(J) | $\Gamma_j$ | Vapor production rate for component cell J. |
| VPK(K) | $\Gamma_k$ | Vapor production rate for junction cell K. |
| WT(J) | --- | Wall temperature for component cell J. |
| CFRS(I,IS) | --- | Input quantity |
| DXS(I,IS) | $\delta x_j$ | Input quantity |
| DYS(I,IS) | $\delta y_j$ | Input quantity |
| GYS(I,IS) | $g_j$ | Input quantity |
| IDXS(I,IS) | --- | Input quantity |
| NJS(I,IS) | --- | Input quantity |
| QJS(I,IS) | $Q_j$ | Input quantity |
| VIS(I,IS) | --- | Input quantity |

## C. Subroutines and FORTRAN Functions

| | |
|---|---|
| ADVANCE | Control program to advance calculation one time step. |
| ARAV | Defines cell center and edge cross-sectional areas. |
| BC | Implements specified boundary conditions. |
| BREAK | Controls calculation interrupt for specification of new input data. |
| DEMXC | Defines hydraulic radius for components. |
| DIAG | Control program for printing component and junction cell solution quantities. |

| | |
|---|---|
| DIAGA | Control program to print component times and time increments. |
| DIAGB | Control program to print mass fluxes at component ends. |
| DRIFT | Calculates relative velocity between phases. |
| ELCAL | Calculates liquid internal energy. |
| EVCAL | Calculates vapor internal energy. |
| FRIC | Calculates effects of distributed and local friction. |
| INFNC | Calculates initial state-equation variables. |
| INPUT | Sets default values, reads and prints input data. |
| JCT0 | Initializes junction cell data storage arrays. |
| JCTPIP | Shifts updated junction cell variables to appropriate component cell data arrays. |
| JSET | Sets indexes required for component cell computations. |
| KSET | Sets indexes required for junction cell computations. |
| L0GCPR | Prints program logic control and storage index parameters. |
| PCAL | Computes pressure from equation of state. |
| PIPJCT | Shifts updated component cell variables to appropriate junction cell arrays. |
| PITERJ | Determines n + 1 pressures and velocities for junction cells by an iterative procedure. |
| PITERP | Determines n + 1 pressures and velocities for component cells by an iterative procedure. |
| PSET | Sets component and junction cell pressure arrays from equations of state—used with explicit solution method. |
| PUMP | Provides capability for specifying a pump model. |
| RBBNCAL | Calculates bubble (or droplet) radius from number density and void fraction. |
| RBWNCAL | Calculates critical bubble (or droplet) radius from Weber number. |
| RESET | Resets cell data arrays for next time step and computes relaxation coefficient $\partial F/\partial p$. |
| RVCAL | Computes vapor density from vapor equation of state. |
| SETUP | Sets program constants, initializes cell variable arrays, and prints processed quantities. |
| SPCAL | Computes saturated pressure from temperature. |
| STCAL | Computes saturated temperature from pressure. |
| TEMC | Computes fluid mixture temperature. |
| THCAL | Computes void fraction from equation of state for $\theta < \theta_c$. |
| TILDE | Computes time n + 1 velocity estimates for all cells. |
| TIMCT | Computes time steps for all components and junctions. |
| VDKCAL | Computes momentum exchange function. |
| VDKPCAL | Computes derivative of momentum exchange function with respect to relative velocity. |
| VPR0 | Computes vapor production rate. |

## V. EXAMPLE PROBLEM

To verify proper implementation of the SOLA-LOOP code on a user's computing system, an example problem is included. A schematic of the problem geometry is shown in Fig. 3. A constant pressure pump ($p = 7.5$ MPa) induces flow of a 500 K liquid through a single branching system into a pressurized vessel ($p = 7.0$ MPa). The flow is calculated until steady state is achieved. A guillotine break is then made in one of the line branches at its vessel connection and a portion of the blowdown transient is calculated. Figure 4 shows the dimensions and computational zoning for the pipe system, which contains three components and one junction. The bottom end of component (pipe) 1, which is connected to the vessel, is treated as an isolated end with a constant pressure outflow boundary condition. At the time of the break the boundary condition will be changed to ambient conditions to reflect the pipe's separation from the vessel. The top end of pipe 1 and the bottom ends of pipes 2 and 3 are connected to the junction. The top end of pipe 2 remains connected to the vessel during the entire calculation and is described by an isolated constant pressure outflow boundary. The top end of pipe 3 is treated as an isolated constant pressure inflow boundary to simulate the pump.

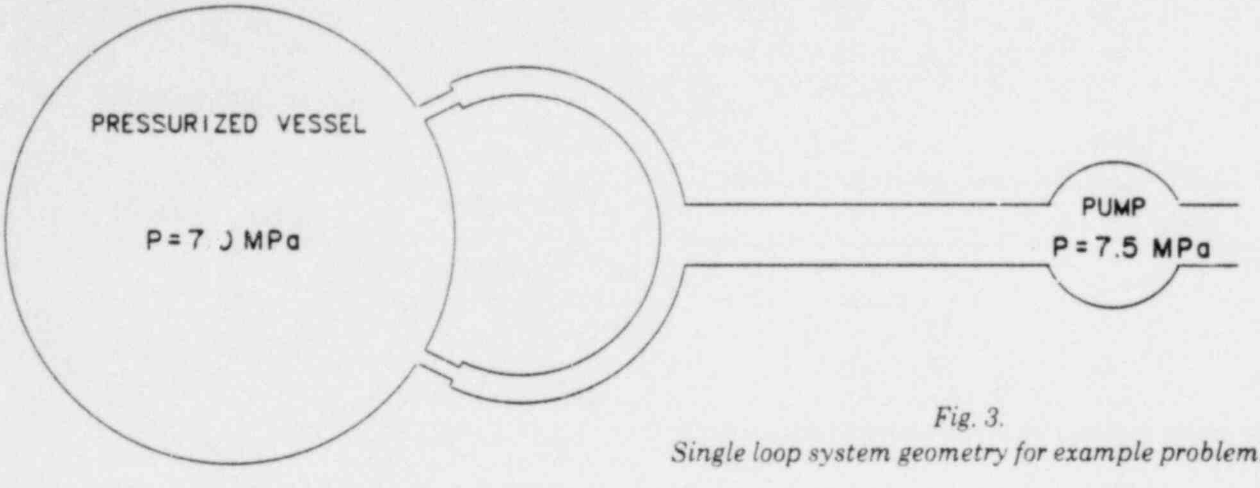

*Fig. 3.*
*Single loop system geometry for example problem.*

The input data that specify the problem are as follows. Recall that the pipe cells adjacent to the junction are furnished automatically by the code to provide a compatible coupling with the automatically sized junction cell.

DXS(1,1) = 3.5
DXS(1,2) = 5.0
DXS(2,1) = 5.0
DXS(2,2) = 3.5
DXS(3,1) = 7.5
DYS(1,1) = 5.25
DYS(1,2) = 22.5
DYS(2,1) = 22.5
DYS(2,2) = 5.25
DYS(3,1) = 23.75
EDL = 1.36 × $10^{-6}$
EDV = 1.67 × $10^{-7}$
ELHT = 1.83 × $10^{6}$
KBØT(2) = 1
KBØT(3) = 1
KTØP(1) = 1
LANG(1) = 2
LANG(2) = 2
LANG(3) = 1
LBØT(1) = 5
LTØP(2) = 5
LTØP(3) = 5
MBØT(1) = 1
MTØP(2) = 1
MTØP(3) = 2
NJS(1,1) = 5
NJS(1,2) = 4
NJS(2,1) = 4
NJS(2,2) = 5
NJS(3,1) = 8
NSUBDT = 5
PB(1) = 70.0
PB(2) = 75.0
PI(1) = 72.0
PI(2) = 72.0
PI(3) = 73.0
RG = 0.05
RØL = 0.83
TBRA(1) = 2000.0
TEMB(1) = 500.
TEMB(2) = 500.
TEMI(1) = 500.
TEMI(2) = 500.
TEMI(3) = 500.
TWFIN = 3000.
VISL = 1.4 × $10^{-6}$
VISV = 1.2 × $10^{-6}$

At 2.0 s, when the break in pipe 1 occurs, the input data are changed to include the following values.

DELT = 1.0 × $10^{-4}$
MBØT(1) = 3
PB(3) = 1.0
TEMB(3) = 373.
THB(3) = 0.995

The input data and other processed data and logic control parameters are listed in Table I as part of the standard output. Note that the size and initial data for the junction cell have been determined automatically by the code. Also note that an additional segment has been added to the top of pipe 1, which contains one real and one fictitious cell. This additional real cell alters the pipe length slightly and the user should be cognizant of this in the input data specification. Additional segments are similarly added to the bottom ends of pipes 2 and 3. These segments each contain two fictitious cells and one real cell. The data that specify the initial states in the

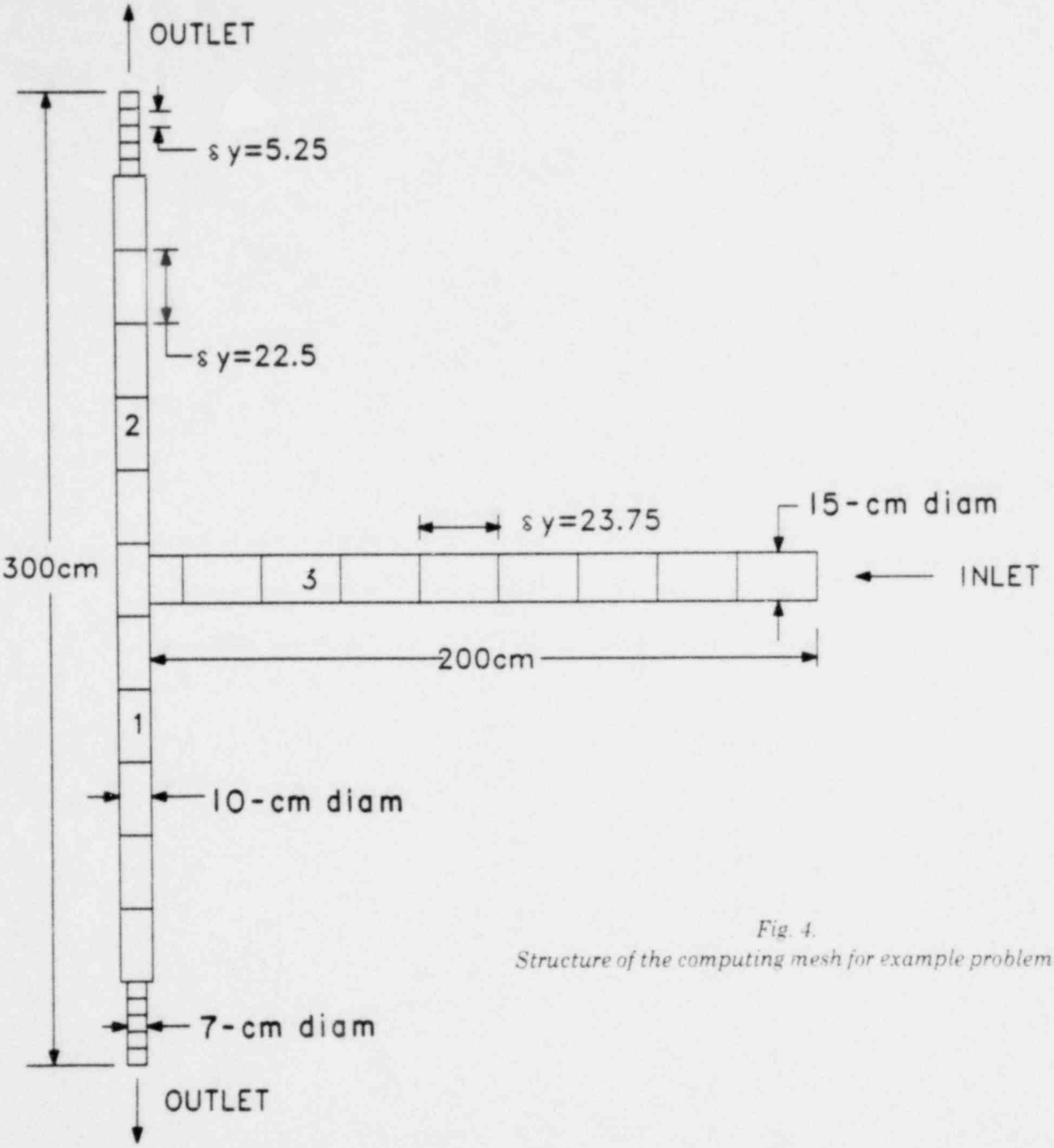

*Fig. 4.*
*Structure of the computing mesh for example problem.*

components and junctions are printed in Table II as the CYCLE = 0, TIME = 0 solution. The solution after one system time step is given in Table III to provide the user with an immediate check. The steady state solution at 2 s just before the break in pipe 1 is given in Table IV. The solution during the blowdown transient at 0.5 s after the break is given in Table V.

## TABLE I

## EXAMPLE PROBLEM INPUT DATA

```
PROBLEM TITLE =  CODE CENTER EXAMPLE    JULY 6,1978       SOLA-LOOP

        INPUT CONSTANTS


          ALPHA =   1.0000000E+00
            ASQ =   1.2340000E+04
           BNUM =   1.0000000E+04
            CDG =   5.0000000E-01
            CHL =   4.4340000E+01
           CHLI =   1.2900000E-02
            CHV =   6.6700000E+00
           CHVI =   3.0200000E-02
            CPI =   3.1415927E+00
           DELT =   1.0000000E-03
         DELSTP =   1.0000000E+02
          DFVEL =   0.
          DPRST =   0.
         DTWPRT =   5.0000000E+02
            ECL =   4.1740000E+03
            ECV =   2.5060000E+04
            EDL =   1.4200000E-06
            EDV =   1.6800000E-07
           ELHT =   1.8730000E+04
           EPSI =   1.0000000E-03
           ETEM =   1.0000000E+00
           GAMI =   7.0000000E-02
            IMP =   1.0000000E+00
         NSUBDT =   2
            OMG =   1.0000000E+00
           PHCH =   1.0000000E+00
             RG =   4.0000000E-03
            ROL =   8.4540000E-01
          STABC =   4.0000000E+00
             TC =   3.7300000E+02
            THC =   1.0000000E-03
         TSTART =   0.
          TWFIN =   2.6000000E+03
          TWSTP =   0.
           VISL =   1.5000000E-06
           VISV =   1.8000000E-05


            BOUNDARY CONDITION INPUT          ML =     3


        M     TEMB            PB              THB


        1  4.8900000E+02  7.2000000E+01  0.
        2  4.8900000E+02  7.5000000E+01  0.
        3  3.7300000E+02  1.0000000E+00  9.9500000E-01
```

## TABLE I (cont)

```
      PIPE INPUT          IL =     3

I  JCEL LANG JSXI     TEMI            PI            THI


1    9    2    2  4.8900000E+02  7.2000000E+01  0.
2    9    2    2  4.8900000E+02  7.2000000E+01  0.
3    8    1    1  4.8900000E+02  7.2000000E+01  0.



      JOINT QUANTITIES           KL =    1

K        DXK            DYK            TEMK           PK           THK


1  1.0000000E+01  2.2500000E+01  4.8900000E+02  7.2000000E+01  0.



      MESH DEPENDENT PROPERTIES

I   JS   NJS IDXS      DXS            DYS          VIS            GYS            CFRL           CFRS          QJS


1    1    7    1  3.500000E+00  5.250000E+00  0.            0.            0.            1.000000E+00  0.
1    2    4    1  5.000000E+00  2.250000E+01  0.            0.            0.            1.000000E+00  0.
1    3    2    1  5.000000E+00  2.250000E+01  0.            0.            0.            1.000000E+00  0.
2    1    3    1  5.000000E+00  2.250000E+01  0.            0.            0.            1.000000E+00  0.
2    2    4    1  5.000000E+00  2.250000E+01  0.            0.            0.            1.000000E+00  0.
2    3    6    1  3.500000E+00  5.250000E+00  0.            0.            0.            1.000000E+00  0.
3    1    3    1  7.500000E+00  1.000000E+01  0.            0.            0.            1.000000E+00  0.
3    2    9    1  7.500000E+00  2.375000E+01  0.            0.            0.            1.000000E+00  0.



      PROCESSED INITIAL STATE VARIABLES


M      TEMB           PB             THB            ROB            RVB            EB


1  4.8900000E+02  7.2000000E+01  0.             8.4865406E-01  1.2114162E-05  9.1440900E+03
2  4.8900000E+02  7.5000000E+01  0.             8.4889717E-01  1.2114162E-05  9.1440900E+03
3  3.7300000E+02  1.0000000E+00  9.9500000E-01  4.7942101E-03  5.6721012E-04  6.6450537E+03



I      TEMI           PI             THI            ROINC          RVINC          EINC          JSXI JCEL


1  4.8900000E+02  7.2000000E+01  0.             8.4865406E-01  1.2114162E-05  9.1440900E+03     3   13
2  4.8900000E+02  7.2000000E+01  0.             8.4865406E-01  1.2114162E-05  9.1440900E+03     3   13
3  4.8900000E+02  7.2000000E+01  0.             8.4865406E-01  1.2114162E-05  9.1440900E+03     2   12



K      TEMK           PK             THK            ROKC           RVKC           EKC


1  4.8900000E+02  7.2000000E+01  0.             8.4865406E-01  1.2114162E-05  9.1440900E+03
```

# TABLE I (cont)

```
      LOGIC PRINT

I  JREF JCEL JMLI JMHI JPBI JPTI KBOT KTOP LBOT LTOP MBOT MTOP  JBI  JTI IBOT ITOP

1     0   13    1    1    1    1    0    1    5    0    1    0    2   13    1    2
2    13   13    1    1    1    1    1    0    0    5    0    1   15   26    2    1
3    26   12    1    1    1    1    1    0    0    5    0    2   28   38    2    1



IP   IS    J  JSF(J)

1     1    1    1
1     1    2    1
1     1    3    1
1     1    4    1
1     1    5    1
1     1    6    1
1     1    7    1
1     2    8   11
1     2    9   11
1     2   10   11
1     2   11   11
1     3   12   21
1     3   13   21
2     1   14    2
2     1   15    2
2     1   16    2
2     2   17   12
2     2   18   12
2     2   19   12
2     2   20   12
2     3   21   22
2     3   22   22
2     3   23   22
2     3   24   22
2     3   25   22
2     3   26   22
3     1   27    3
3     1   28    3
3     1   29    3
3     2   30   13
3     2   31   13
3     2   32   13
3     2   33   13
3     2   34   13
3     2   35   13
3     2   36   13
3     2   37   13
3     2   38   13
```

# TABLE II

## EXAMPLE PROBLEM INITIAL DATA

```
PROBLEM TITLE =  CODE CENTER EXAMPLE    JULY 6,1978       SOLA-LOOP

ITER=  0  TIME= 0.           DELTO= 1.00000E-03  DTMN= 1.00000E-03  CYCLE=      0  ISTEP= 0
  TIM( 1)=-1.00000E-06   DTIM( 1)= 0.              TIM( 2)=-1.00000E-06   DTIM( 2)= 0.
  TIM( 3)=-1.00000E-06   DTIM( 3)= 0.
  TIMK( 1)=-1.00000E-06   DTIMK( 1)= 0.


 I    J          V              VD              P              RO            VP
 1    2    0.             0.             7.2000E+01     8.4865E-01     0.
 1    3    0.             0.             7.2000E+01     8.4865E-01     0.
 1    4    0.             0.             7.2000E+01     8.4865E-01     0.
 1    5    0.             0.             7.2000E+01     8.4865E-01     0.
 1    6    0.             0.             7.2000E+01     8.4865E-01     0.
 1    7    0.             0.             7.2000E+01     8.4865E-01     0.
 1    8    0.             0.             7.2000E+01     8.4865E-01     0.
 1    9    0.             0.             7.2000E+01     8.4865E-01     0.
 1   10    0.             0.             7.2000E+01     8.4865E-01     0.
 1   11    0.             0.             7.2000E+01     8.4865E-01     0.
 1   12    0.             0.             7.2000E+01     8.4865E-01     0.
 1   13    0.             0.             7.2000E+01     8.4865E-01     0.
 I    J         RV              E              TH             TEM            TSAT            WT
 1    2    1.2114E-05     9.1441E+03    -3.8348E-03     4.8900E+02     5.6092E+02     0.
 1    3    1.2114E-05     9.1441E+03    -3.8348E-03     4.8900E+02     5.6092E+02     0.
 1    4    1.2114E-05     9.1441E+03    -3.8348E-03     4.8900E+02     5.6092E+02     0.
 1    5    1.2114E-05     9.1441E+03    -3.8348E-03     4.8900E+02     5.6092E+02     0.
 1    6    1.2114E-05     9.1441E+03    -3.8348E-03     4.8900E+02     5.6092E+02     0.
 1    7    1.2114E-05     9.1441E+03    -3.8348E-03     4.8900E+02     5.6092E+02     0.
 1    8    1.2114E-05     9.1441E+03    -3.8348E-03     4.8900E+02     5.6092E+02     0.
 1    9    1.2114E-05     9.1441E+03    -3.8348E-03     4.8900E+02     5.6092E+02     0.
 1   10    1.2114E-05     9.1441E+03    -3.8348E-03     4.8900E+02     5.6092E+02     0.
 1   11    1.2114E-05     9.1441E+03    -3.8348E-03     4.8900E+02     5.6092E+02     0.
 1   12    1.2114E-05     9.1441E+03    -3.8348E-03     4.8900E+02     5.6092E+02     0.
 1   13    1.2114E-05     9.1441E+03    -3.8348E-03     4.8900E+02     5.6092E+02     0.
 I    J          V              VD              P              RO            VP
 2   15    0.             0.             7.2000E+01     8.4865E-01     0.
 2   16    0.             0.             7.2000E+01     8.4865E-01     0.
 2   17    0.             0.             7.2000E+01     8.4865E-01     0.
 2   18    0.             0.             7.2000E+01     8.4865E-01     0.
 2   19    0.             0.             7.2000E+01     8.4865E-01     0.
 2   20    0.             0.             7.2000E+01     8.4865E-01     0.
 2   21    0.             0.             7.2000E+01     8.4865E-01     0.
 2   22    0.             0.             7.2000E+01     8.4865E-01     0.
 2   23    0.             0.             7.2000E+01     8.4865E-01     0.
 2   24    0.             0.             7.2000E+01     8.4865E-01     0.
 2   25    0.             0.             7.2000E+01     8.4865E-01     0.
 2   26    0.             0.             7.2000E+01     8.4865E-01     0.
 I    J         RV              E              TH             TEM            TSAT            WT
 2   15    1.2114E-05     9.1441E+03    -3.8348E-03     4.8900E+02     5.6092E+02     0.
 2   16    1.2114E-05     9.1441E+03    -3.8348E-03     4.8900E+02     5.6092E+02     0.
 2   17    1.2114E-05     9.1441E+03    -3.8348E-03     4.8900E+02     5.6092E+02     0.
 2   18    1.2114E-05     9.1441E+03    -3.8348E-03     4.8900E+02     5.6092E+02     0.
```

## TABLE II (cont)

```
2  19    1.2114E-05    9.1441E+03   -3.8348E-03    4.8900E+02    5.6092E+02    0.
2  20    1.2114E-05    9.1441E+03   -3.8348E-03    4.8900E+02    5.6092E+02    0.
2  21    1.2114E-05    9.1441E+03   -3.8348E-03    4.8900E+02    5.6092E+02    0.
2  22    1.2114E-05    9.1441E+03   -3.8348E-03    4.8900E+02    5.6092E+02    0.
2  23    1.2114E-05    9.1441E+03   -3.8348E-03    4.8900E+02    5.6092E+02    0.
2  24    1.2114E-05    9.1441E+03   -3.8348E-03    4.8900E+02    5.6092E+02    0.
2  25    1.2114E-05    9.1441E+03   -3.8348E-03    4.8900E+02    5.6092E+02    0.
2  26    1.2114E-05    9.1441E+03   -3.8348E-03    4.8900E+02    5.6092E+02    0.
I   J           V             VD            P            RO          VP

3  28    0.            0.            7.2000E+01    8.4865E-01    0.
3  29    0.            0.            7.2000E+01    8.4865E-01    0.
3  30    0.            0.            7.2000E+01    8.4865E-01    0.
3  31    0.            0.            7.2000E+01    8.4865E-01    0.
3  32    0.            0.            7.2000E+01    8.4865E-01    0.
3  33    0.            0.            7.2000E+01    8.4865E-01    0.
3  34    0.            0.            7.2000E+01    8.4865E-01    0.
3  35    0.            0.            7.2000E+01    8.4865E-01    0.
3  36    0.            0.            7.2000E+01    8.4865E-01    0.
3  37    0.            0.            7.2000E+01    8.4865E-01    0.
3  38    0.            0.            7.5000E+01    8.4890E-01    0.
I   J        RV            E             TH           TEM           TSAT             WT
3  28    1.2114E-05    9.1441E+03   -3.8348E-03    4.8900E+02    5.6092E+02    0.
3  29    1.2114E-05    9.1441E+03   -3.8348E-03    4.8900E+02    5.6092E+02    0.
3  30    1.2114E-05    9.1441E+03   -3.8348E-03    4.8900E+02    5.6092E+02    0.
3  31    1.2114E-05    9.1441E+03   -3.8348E-03    4.8900E+02    5.6092E+02    0.
3  32    1.2114E-05    9.1441E+03   -3.8348E-03    4.8900E+02    5.6092E+02    0.
3  33    1.2114E-05    9.1441E+03   -3.8348E-03    4.8900E+02    5.6092E+02    0.
3  34    1.2114E-05    9.1441E+03   -3.8348E-03    4.8900E+02    5.6092E+02    0.
3  35    1.2114E-05    9.1441E+03   -3.8348E-03    4.8900E+02    5.6092E+02    0.
3  36    1.2114E-05    9.1441E+03   -3.8348E-03    4.8900E+02    5.6092E+02    0.
3  37    1.2114E-05    9.1441E+03   -3.8348E-03    4.8900E+02    5.6092E+02    0.
3  38    1.2114E-05    9.1441E+03   -4.1224E-03    4.8900E+02    5.6372E+02    0.


 K        P               RO              RV               E              VP
 1  7.2000000E+01  8.4865406E-01  1.2114162E-05  9.1440900E+03  0.
```

# TABLE III

## EXAMPLE PROBLEM SOLUTION AFTER ONE CYCLE

```
PROBLEM TITLE =  CODE CENTER EXAMPLE     JULY 6,1978        SOLA-LOOP

ITER=  1  TIME= 1.00000E-03  DELTO= 1.01000E-03  DTMN= 1.01000E-03  CYCLE=      1  ISTEP= 1
  TIM( 1)= 9.99000E-04    DTIM( 1)= 1.01000E-03    TIM( 2)= 9.99000E-04    DTIM( 2)= 1.01000E-03
  TIM( 3)= 9.99000E-04    DTIM( 3)= 1.01000E-03
  TIMK( 1)= 9.99000E-04    DTIMK( 1)= 1.01000E-03


 I   J          V              VD             P             RO            VP
 1   2     5.0012E-15     0.             7.2000E+01    8.4865E-01    0.
 1   3     0.             0.             7.2000E+01    8.4865E-01    0.
 1   4     0.             0.             7.2000E+01    8.4865E-01    0.
 1   5     0.             0.             7.2000E+01    8.4865E-01    0.
 1   6     0.             0.             7.2000E+01    8.4865E-01    0.
 1   7     0.             0.             7.2000E+01    8.4865E-01    0.
 1   8     0.             0.             7.2000E+01    8.4865E-01    0.
 1   9     0.             0.             7.2000E+01    8.4865E-01    0.
 1  10     0.             0.             7.2000E+01    8.4865E-01    0.
 1  11     0.             0.             7.2000E+01    8.4865E-01    0.
 1  12     4.7631E-17     0.             7.2000E+01    8.4865E-01    0.
 1  13     2.2148E-15     0.             7.2000E+01    8.4865E-01    0.
 I   J         RV             E             TH            TEM           TSAT           WT
 1   2     1.2114E-05    9.1441E+03    -3.8348E-03    4.8900E+02    5.6092E+02    0.
 1   3     1.2114E-05    9.1441E+03    -3.8348E-03    4.8900E+02    5.6092E+02    0.
 1   4     1.2114E-05    9.1441E+03    -3.8348E-03    4.8900E+02    5.6092E+02    0.
 1   5     1.2114E-05    9.1441E+03    -3.8348E-03    4.8900E+02    5.6092E+02    0.
 1   6     1.2114E-05    9.1441E+03    -3.8348E-03    4.8900E+02    5.6092E+02    0.
 1   7     1.2114E-05    9.1441E+03    -3.8348E-03    4.8900E+02    5.6092E+02    0.
 1   8     1.2114E-05    9.1441E+03    -3.8348E-03    4.8900E+02    5.6092E+02    0.
 1   9     1.2114E-05    9.1441E+03    -3.8348E-03    4.8900E+02    5.6092E+02    0.
 1  10     1.2114E-05    9.1441E+03    -3.8348E-03    4.8900E+02    5.6092E+02    0.
 1  11     1.2114E-05    9.1441E+03    -3.8348E-03    4.8900E+02    5.6092E+02    0.
 1  12     1.2114E-05    9.1441E+03    -3.8348E-03    4.8900E+02    5.6092E+02    0.
 1  13     1.2114E-05    9.1441E+03    -3.8348E-03    4.8900E+02    5.6092E+02    0.
 I   J          V              VD             P             RO            VP
 2  15     2.2148E-15     0.             7.2000E+01    8.4865E-01    0.
 2  16     0.             0.             7.2000E+01    8.4865E-01    0.
 2  17     0.             0.             7.2000E+01    8.4865E-01    0.
 2  18     0.             0.             7.2000E+01    8.4865E-01    0.
 2  19     0.             0.             7.2000E+01    8.4865E-01    0.
 2  20     0.             0.             7.2000E+01    8.4865E-01    0.
 2  21     0.             0.             7.2000E+01    8.4865E-01    0.
 2  22     0.             0.             7.2000E+01    8.4865E-01    0.
 2  23     0.             0.             7.2000E+01    8.4865E-01    0.
 2  24    -9.6962E-15     0.             7.2000E+01    8.4865E-01    0.
 2  25    -5.0012E-15     0.             7.2000E+01    8.4865E-01    0.
 2  26    -5.0012E-15     0.             7.2000E+01    8.4865E-01    0.
 I   J         RV             E             TH            TEM           TSAT           WT
 2  15     1.2114E-05    9.1441E+03    -3.8348E-03    4.8900E+02    5.6092E+02    0.
 2  16     1.2114E-05    9.1441E+03    -3.8348E-03    4.8900E+02    5.6092E+02    0.
 2  17     1.2114E-05    9.1441E+03    -3.8348E-03    4.8900E+02    5.6092E+02    0.
 2  18     1.2114E-05    9.1441E+03    -3.8348E-03    4.8900E+02    5.6092E+02    0.
```

## TABLE III (cont)

```
2  19    1.2114E-05    9.1441E+03   -3.8348E-03    4.8900E+02    5.6092E+02    0.
2  20    1.2114E-05    9.1441E+03   -3.8348E-03    4.8900E+02    5.6092E+02    0.
2  21    1.2114E-05    9.1441E+03   -3.8348E-03    4.8900E+02    5.6092E+02    0.
2  22    1.2114E-05    9.1441E+03   -3.8348E-03    4.8900E+02    5.6092E+02    0.
2  23    1.2114E-05    9.1441E+03   -3.8348E-03    4.8900E+02    5.6092E+02    0.
2  24    1.2114E-05    9.1441E+03   -3.8348E-03    4.8900E+02    5.6092E+02    0.
2  25    1.2114E-05    9.1441E+03   -3.8348E-03    4.8900E+02    5.6092E+02    0.
2  26    1.2114E-05    9.1441E+03   -3.8348E-03    4.8900E+02    5.6092E+02    0.
I   J           V              VD            P             RO            VP

3  28   4.9834E-15    0.              7.2000E+01   8.4865E-01    0.
3  29   0.            0.              7.2000E+01   8.4865E-01    0.
3  30   0.            0.              7.2000E+01   8.4865E-01    0.
3  31   0.            0.              7.2000E+01   8.4865E-01    0.
3  32   0.            0.              7.2000E+01   8.4865E-01    0.
3  33   0.            0.              7.2000E+01   8.4865E-01    0.
3  34   0.            0.              7.2000E+01   8.4865E-01    0.
3  35   0.            0.              7.2000E+01   8.4865E-01    0.
3  36  -3.1972E-09    0.              7.2000E+01   8.4865E-01    0.
3  37  -1.4882E-04    0.              7.2000E+01   8.4865E-01    0.
3  38  -1.4882E-04    0.              7.5000E+01   8.4890E-01    0.
I   J       RV             E             TH            TEM            TSAT              WT
3  28    1.2114E-05   9.1441E+03   -3.8348E-03    4.8900E+02    5.6092E+02    0.
3  29    1.2114E-05   9.1441E+03   -3.8348E-03    4.8900E+02    5.6092E+02    0.
3  30    1.2114E-05   9.1441E+03   -3.8348E-03    4.8900E+02    5.6092E+02    0.
3  31    1.2114E-05   9.1441E+03   -3.8348E-03    4.8900E+02    5.6092E+02    0.
3  32    1.2114E-05   9.1441E+03   -3.8348E-03    4.8900E+02    5.6092E+02    0.
3  33    1.2114E-05   9.1441E+03   -3.8348E-03    4.8900E+02    5.6092E+02    0.
3  34    1.2114E-05   9.1441E+03   -3.8348E-03    4.8900E+02    5.6092E+02    0.
3  35    1.2114E-05   9.1441E+03   -3.8348E-03    4.8900E+02    5.6092E+02    0.
3  36    1.2114E-05   9.1441E+03   -3.8348E-03    4.8900E+02    5.6092E+02    0.
3  37    1.2114E-05   9.1441E+03   -3.8348E-03    4.8900E+02    5.6092E+02    0.
3  38    1.2114E-05   9.1441E+03   -4.1224E-03    4.8900E+02    5.6372E+02    0.


   K        P                RO                RV                E                VP
   1  7.2000000E+01  8.4865406E-01  1.2114162E-05  9.1440900E+03  0.



TIME =  1.00000E-03
     PIPE  1   BOTTOM MASS FLUX =  1.63340E-13   TOP MASS FLUX =  3.17474E-15
     PIPE  2   BOTTOM MASS FLUX =  1.47625E-13   TOP MASS FLUX = -1.63340E-13
     PIPE  3   BOTTOM MASS FLUX =  7.47353E-13   TOP MASS FLUX = -2.23246E-02
```

# TABLE IV

## EXAMPLE PROBLEM SOLUTION AT INSTANT OF PIPE BREAK

```
PROBLEM TITLE =  CODE CENTER EXAMPLE     JULY 6,1978        SOLA-LOOP

ITER=  1  TIME= 2.00107E+03  DELTO= 1.00000E-02  DTMN= 1.00000E-02  CYCLE=  4213  ISTEP= 2
  TIM( 1)= 2.00038E+03   DTIM( 1)= 6.60176E-01   TIM( 2)= 2.00038E+03   DTIM( 2)= 6.60176E-01
  TIM( 3)= 2.00104E+03   DTIM( 3)= 1.00000E-02
  TIMK( 1)= 2.00104E+03   DTIMK( 1)= 1.00000E-02


 I   J            V              VD              P            RO            VP
 1   2    -1.8785E+00    0.                1.0000E+00    8.4834E-01    0.
 1   3    -1.8818E+00    0.                7.1837E+01    8.4834E-01    0.
 1   4    -1.8862E+00    0.                7.1682E+01    8.4825E-01    0.
 1   5    -1.8919E+00    0.                7.1572E+01    8.4828E-01    0.
 1   6    -1.8965E+00    0.                7.1542E+01    8.4845E-01    0.
 1   7    -1.4136E+00    0.                7.2352E+01    8.4865E-01    0.
 1   8    -9.2938E-01    0.                7.2777E+01    8.4872E-01    0.
 1   9    -9.2942E-01    0.                7.2851E+01    8.4874E-01    0.
 1  10    -9.3044E-01    0.                7.2934E+01    8.4877E-01    0.
 1  11    -9.3303E-01    0.                7.3072E+01    8.4880E-01    0.
 1  12    -9.3738E-01    0.                7.3309E+01    8.4884E-01    0.
 1  13     9.3587E-01    0.                7.4583E+01    8.4893E-01    0.
 I   J          RV            E             TH             TEM            TSAT           WT
 1   2     1.2106E-05    9.1444E+03   -3.4647E-03    4.8901E+02    3.7300E+02    0.
 1   3     1.2106E-05    9.1444E+03   -3.4647E-03    4.8901E+02    5.6077E+02    0.
 1   4     1.2105E-05    9.1443E+03   -3.3616E-03    4.8901E+02    5.6062E+02    0.
 1   5     1.2105E-05    9.1443E+03   -3.3923E-03    4.8900E+02    5.6052E+02    0.
 1   6     1.2108E-05    9.1443E+03   -3.5956E-03    4.8900E+02    5.6049E+02    0.
 1   7     1.2111E-05    9.1443E+03   -3.8291E-03    4.8900E+02    5.6126E+02    0.
 1   8     1.2112E-05    9.1443E+03   -3.9150E-03    4.8900E+02    5.6166E+02    0.
 1   9     1.2112E-05    9.1442E+03   -3.9417E-03    4.8900E+02    5.6173E+02    0.
 1  10     1.2112E-05    9.1442E+03   -3.9714E-03    4.8900E+02    5.6180E+02    0.
 1  11     1.2113E-05    9.1442E+03   -4.0070E-03    4.8900E+02    5.6193E+02    0.
 1  12     1.2113E-05    9.1442E+03   -4.0534E-03    4.8900E+02    5.6216E+02    0.
 1  13     1.2115E-05    9.1442E+03   -4.1649E-03    4.8900E+02    5.6334E+02    0.
 I   J            V              VD              P            RO            VP
 2  15     9.3569E-01    0.                7.4583E+01    8.4893E-01    0.
 2  16     9.3040E-01    0.                7.3412E+01    8.4889E-01    0.
 2  17     9.2938E-01    0.                7.3276E+01    8.4878E-01    0.
 2  18     9.2909E-01    0.                7.3230E+01    8.4877E-01    0.
 2  19     9.2922E-01    0.                7.3205E+01    8.4876E-01    0.
 2  20     1.4134E+00    0.                7.3188E+01    8.4875E-01    0.
 2  21     1.8978E+00    0.                7.2806E+01    8.4870E-01    0.
 2  22     1.8989E+00    0.                7.2010E+01    8.4864E-01    0.
 2  23     1.8998E+00    0.                7.2001E+01    8.4864E-01    0.
 2  24     1.9003E+00    0.                7.1998E+01    8.4865E-01    0.
 2  25     1.9005E+00    0.                7.1998E+01    8.4866E-01    0.
 2  26     1.9005E+00    0.                7.2000E+01    8.4866E-01    0.
 I   J          RV            E             TH             TEM            TSAT           WT
 2  15     1.2115E-05    9.1442E+03   -4.1649E-03    4.8900E+02    5.6334E+02    0.
 2  16     1.2114E-05    9.1442E+03   -4.1114E-03    4.8900E+02    5.6225E+02    0.
 2  17     1.2113E-05    9.1442E+03   -3.9889E-03    4.8900E+02    5.6212E+02    0.
 2  18     1.2112E-05    9.1442E+03   -3.9713E-03    4.8900E+02    5.6208E+02    0.
 2  19     1.2112E-05    9.1442E+03   -3.9587E-03    4.8900E+02    5.6206E+02    0.
```

# TABLE IV (cont)

```
 2  20     1.2112E-05     9.1443E+03    -3.9501E-03     4.8900E+02     5.6204E+02     0.
 2  21     1.2111E-05     9.1443E+03    -3.8913E-03     4.8900E+02     5.6168E+02     0.
 2  22     1.2110E-05     9.1443E+03    -3.8135E-03     4.8901E+02     5.6093E+02     0.
 2  23     1.2110E-05     9.1443E+03    -3.8147E-03     4.8901E+02     5.6093E+02     0.
 2  24     1.2111E-05     9.1444E+03    -3.8292E-03     4.8901E+02     5.6092E+02     0.
 2  25     1.2111E-05     9.1444E+03    -3.8415E-03     4.8901E+02     5.6092E+02     0.
 2  26     1.2111E-05     9.1444E+03    -3.8415E-03     4.8901E+02     5.6092E+02     0.
 I   J          V                VD            P             RO            VP

 3  28    -8.3352E-01    0.                7.4583E+01    8.4893E-01    0.
 3  29    -8.4048E-01    0.                7.4587E+01    8.5011E-01    0.
 3  30    -8.4390E-01    0.                7.4570E+01    8.4900E-01    0.
 3  31    -8.4701E-01    0.                7.4566E+01    8.4898E-01    0.
 3  32    -8.4962E-01    0.                7.4585E+01    8.4896E-01    0.
 3  33    -8.5168E-01    0.                7.4625E+01    8.4895E-01    0.
 3  34    -8.5320E-01    0.                7.4683E+01    8.4893E-01    0.
 3  35    -8.5421E-01    0.                7.4753E+01    8.4892E-01    0.
 3  36    -8.5474E-01    0.                7.4832E+01    8.4890E-01    0.
 3  37    -8.5481E-01    0.                7.4915E+01    8.4889E-01    0.
 3  38    -8.5481E-01    0.                7.5000E+01    8.4890E-01    0.
 I   J        RV              E             TH             TEM            TSAT               WT
 3  28     1.2115E-05     9.1442E+03    -4.1649E-03     4.8900E+02     5.6334E+02     0.
 3  29     1.2131E-05     9.1443E+03    -5.5521E-03     4.8900E+02     5.6334E+02     0.
 3  30     1.2116E-05     9.1442E+03    -4.2410E-03     4.8900E+02     5.6332E+02     0.
 3  31     1.2115E-05     9.1442E+03    -4.2203E-03     4.8900E+02     5.6332E+02     0.
 3  32     1.2115E-05     9.1442E+03    -4.1995E-03     4.8900E+02     5.6334E+02     0.
 3  33     1.2115E-05     9.1441E+03    -4.1798E-03     4.8900E+02     5.6338E+02     0.
 3  34     1.2115E-05     9.1441E+03    -4.1615E-03     4.8900E+02     5.6343E+02     0.
 3  35     1.2114E-05     9.1441E+03    -4.1448E-03     4.8900E+02     5.6349E+02     0.
 3  36     1.2114E-05     9.1441E+03    -4.1293E-03     4.8900E+02     5.6357E+02     0.
 3  37     1.2114E-05     9.1441E+03    -4.1149E-03     4.8900E+02     5.6364E+02     0.
 3  38     1.2114E-05     9.1441E+03    -4.1224E-03     4.8900E+02     5.6372E+02     0.


   K        P                RO                RV                E              VP
   1  7.4582766E+01  8.4893316E-01  1.2114675E-05  9.1441669E+03  0.



TIME =  2.00107E+03
    PIPE  1   BOTTOM MASS FLUX = -6.13294E+01   TOP MASS FLUX = -6.24998E+01
    PIPE  2   BOTTOM MASS FLUX =  6.23873E+01   TOP MASS FLUX =  6.20713E+01
    PIPE  3   BOTTOM MASS FLUX = -1.25216E+02   TOP MASS FLUX = -1.28233E+02
```

## TABLE V

## EXAMPLE PROBLEM SOLUTION AT 0.5 s AFTER PIPE BREAK

```
PROBLEM TITLE =  CODE CENTER EXAMPLE    JULY 6,1978       SOLA-LOOP


ITER=  2  TIME= 2.50010E+03  DELTO= 2.22725E-01  DTMN= 1.11362E-01  CYCLE= 10121  ISTEP= 2
  TIM( 1)= 2.50007E+03   DTIM( 1)= 1.11362E-01   TIM( 2)= 2.50007E+03   DTIM( 2)= 2.22725E-01
  TIM( 3)= 2.50007E+03   DTIM( 3)= 2.22725E-01
  TIMK( 1)= 2.50007E+03   DTIMK( 1)= 2.22725E-01


 I   J         V             VD             P           RO           VP
 1   2   -1.1717E+01    0.             1.0000E+00   7.0110E-01    0.
 1   3   -9.8351E+00    0.             1.7208E+01   7.0110E-01    4.3133E-03
 1   4   -9.7472E+00    0.             1.8467E+01   8.3640E-01    2.0635E-04
 1   5   -9.7350E+00    0.             1.9146E+01   8.4379E-01    2.2800E-05
 1   6   -9.7182E+00    0.             1.9860E+01   8.4480E-01    0.
 1   7   -7.2342E+00    0.             4.0855E+01   8.4627E-01    0.
 1   8   -4.7574E+00    0.             5.0876E+01   8.4695E-01    0.
 1   9   -4.7565E+00    0.             5.1305E+01   8.4698E-01    0.
 1  10   -4.7551E+00    0.             5.1793E+01   8.4702E-01    0.
 1  11   -4.7535E+00    0.             5.2325E+01   8.4707E-01    0.
 1  12   -4.7432E+00    0.             5.2918E+01   8.4715E-01    0.
 1  13    4.7312E-01    0.             7.4434E+01   8.4885E-01    0.
 I   J        RV            E             TH           TEM            TSAT             WT
 1   2    2.0550E-03   9.1467E+03    1.7312E-01   4.8791E+02   3.7300E+02    0.
 1   3    2.0550E-03   9.1467E+03    1.7312E-01   4.8791E+02   4.7738E+02    0.
 1   4    1.3126E-04   9.1498E+03    1.0807E-02   4.8908E+02   4.8091E+02    0.
 1   5    2.3505E-05   9.1493E+03    1.9324E-03   4.8912E+02   4.8274E+02    0.
 1   6    1.2056E-05   9.1486E+03    7.2497E-04   4.8911E+02   4.8460E+02    0.
 1   7    1.2077E-05   9.1480E+03   -1.0171E-03   4.8909E+02   5.2463E+02    0.
 1   8    1.2086E-05   9.1476E+03   -1.8154E-03   4.8909E+02   5.3814E+02    0.
 1   9    1.2087E-05   9.1464E+03   -1.8566E-03   4.8906E+02   5.3867E+02    0.
 1  10    1.2087E-05   9.1459E+03   -1.9033E-03   4.8904E+02   5.3927E+02    0.
 1  11    1.2088E-05   9.1455E+03   -1.9556E-03   4.8903E+02   5.3992E+02    0.
 1  12    1.2089E-05   9.1450E+03   -2.0567E-03   4.8902E+02   5.4064E+02    0.
 1  13    1.2114E-05   9.1447E+03   -4.0686E-03   4.8901E+02   5.6320E+02    0.
 I   J         V             VD             P           RO           VP
 2  15    4.7312E-01    0.             7.4434E+01   8.4885E-01    0.
 2  16    4.7298E-01    0.             7.2365E+01   8.4869E-01    0.
 2  17    4.7259E-01    0.             7.2378E+01   8.4869E-01    0.
 2  18    4.7209E-01    0.             7.2383E+01   8.4869E-01    0.
 2  19    4.7159E-01    0.             7.2382E+01   8.4869E-01    0.
 2  20    7.1632E-01    0.             7.2371E+01   8.4869E-01    0.
 2  21    9.6133E-01    0.             7.2259E+01   8.4869E-01    0.
 2  22    9.6122E-01    0.             7.2048E+01   8.4867E-01    0.
 2  23    9.6113E-01    0.             7.2036E+01   8.4867E-01    0.
 2  24    9.6108E-01    0.             7.2024E+01   8.4866E-01    0.
 2  25    9.6107E-01    0.             7.2012E+01   8.4866E-01    0.
 2  26    9.6107E-01    0.             7.2000E+01   8.4866E-01    0.
 I   J        RV            E             TH           TEM            TSAT             WT
 2  15    1.2114E-05   9.1447E+03   -4.0686E-03   4.8901E+02   5.6320E+02    0.
 2  16    1.2111E-05   9.1447E+03   -3.8728E-03   4.8901E+02   5.6127E+02    0.
 2  17    1.2111E-05   9.1447E+03   -3.8732E-03   4.8901E+02   5.6128E+02    0.
 2  18    1.2111E-05   9.1447E+03   -3.8739E-03   4.8901E+02   5.6129E+02    0.
```

## TABLE V (cont)

```
2  19    1.2111E-05    9.1447E+03   -3.8732E-03    4.8901E+02    5.6129E+02    0.
2  20    1.2111E-05    9.1447E+03   -3.8722E-03    4.8901E+02    5.6128E+02    0.
2  21    1.2111E-05    9.1447E+03   -3.8795E-03    4.8901E+02    5.6117E+02    0.
2  22    1.2111E-05    9.1447E+03   -3.8565E-03    4.8901E+02    5.6097E+02    0.
2  23    1.2111E-05    9.1447E+03   -3.8501E-03    4.8901E+02    5.6096E+02    0.
2  24    1.2111E-05    9.1447E+03   -3.8433E-03    4.8901E+02    5.6095E+02    0.
2  25    1.2111E-05    9.1447E+03   -3.8368E-03    4.8901E+02    5.6094E+02    0
2  26    1.2111E-05    9.1447E+03   -3.8368E-03    4.8901E+02    5.6092E+02    0.
I   J           V               VD            P             RO            VP

3  28   -2.3180E+00    0.                7.4434E+01    8.4885E-01    0.
3  29   -2.3180E+00    0.                7.4466E+01    8.4886E-01    0.
3  30   -2.3183E+00    0.                7.4519E+01    8.4886E-01    0.
3  31   -2.3188E+00    0.                7.4582E+01    8.4886E-01    0.
3  32   -2.3193E+00    0.                7.4639E+01    8.4887E-01    0.
3  33   -2.3199E+00    0.                7.4692E+01    8.4887E-01    0.
3  34   -2.3205E+00    0.                7.4746E+01    8.4888E-01    0.
3  35   -2.3209E+00    0.                7.4802E+01    8.4888E-01    0.
3  36   -2.3212E+00    0.                7.4865E+01    8.4889E-01    0.
3  37   -2.3214E+00    0.                7.4932E+01    8.4889E-01    0.
3  38   -2.3214E+00    0.                7.5000E+01    8.4890E-01    0.
I   J         RV             E              TH            TEM            TSAT              WT
3  28    1.2114E-05    9.1447E+03   -4.0686E-03    4.8901E+02    5.6320E+02    0.
3  29    1.2114E-05    9.1447E+03   -4.0764E-03    4.8901E+02    5.6323E+02    0.
3  30    1.2114E-05    9.1447E+03   -4.0777E-03    4.8901E+02    5.6328E+02    0.
3  31    1.2114E-05    9.1446E+03   -4.0840E-03    4.8901E+02    5.6334E+02    0.
3  32    1.2114E-05    9.1445E+03   -4.0895E-03    4.8901E+02    5.6339E+02    0.
3  33    1.2114E-05    9.1444E+03   -4.0945E-03    4.8901E+02    5.6344E+02    0.
3  34    1.2114E-05    9.1444E+03   -4.0994E-03    4.8901E+02    5.6349E+02    0.
3  35    1.2114E-05    9.1443E+03   -4.1045E-03    4.8901E+02    5.6354E+02    0.
3  36    1.2114E-05    9.1442E+03   -4.1099E-03    4.8900E+02    5.6360E+02    0.
3  37    1.2114E-05    9.1442E+03   -4.1160E-03    4.8900E+02    5.6366E+02    0.
3  38    1.2114E-05    9.1441E+03   -4.1224E-03    4.8900E+02    5.6372E+02    0.


  K        P              RO              RV              E               VP
  1  7.4433858E+01  8.4885173E-01  1.2113513E-05  9.1446877E+03  0.



TIME =  2.50010E+03
    PIPE  1   BOTTOM MASS FLUX = -3.16144E+02   TOP MASS FLUX = -3.16225E+02
    PIPE  2   BOTTOM MASS FLUX =  3.15423E+01   TOP MASS FLUX =  3.13885E-01
    PIPE  3   BOTTOM MASS FLUX = -3.47711E+02   TOP MASS FLUX = -3.48232E+02
```

## VI. SUMMARY

We have described a new computer program, SOLA-LOOP, for the solution of transient, two-phase flow in networks composed of one-dimensional components. The fluid dynamics is described by a nonequilibrium, drift-flux formulation of the fluid conservation laws. We have used relatively simple numerical solution procedures and modular programming to provide a framework that can be easily modified and adapted to different kinds of network flow problems. In addition, we used a limited amount of implicitness to relax excessively restrictive time step limitations encountered in purely explicit integration methods. Even though SOLA-LOOP has a simple structure, its flexibility offers capabilities for treating a wide range of two-phase flow problems.

## ACKNOWLEDGMENTS

We have profited greatly from discussions with our colleagues. We especially thank H. Ruppel for his assistance with the subtleties of high-speed computing in a time sharing environment. In addition, we thank the members of the Analysis Development Branch of the US Nuclear Regulatory Commission, Division of Reactor Safety Research, for their interest, critical comments, and financial support.

## REFERENCES

1. A. A. Amsden and F. H. Harlow, "K-TIF: A Two-Fluid Computer Program for Downcomer Flow Dynamics," Los Alamos Scientific Laboratory report LA-6994 (January 1978).

2. W. C. Rivard and M. D. Torrey, "K-FIX: A Computer Program for Transient, Two-Dimensional, Two-Fluid Flow," Los Alamos Scientific Laboratory report LA-NUREG-6623 (April 1977).

3. C. W. Hirt and N. C. Romero, "Application of a Drift-Flux Model to Flashing in Straight Pipes," Los Alamos Scientific Laboratory report LA-6005-MS (July 1975).

4. N. Zuber, "Flow Excursions and Oscillations in Boiling Two-Phase Flow Systems with Heat Addition," Sym. Two-Phase Flow Dynamics, E.U.R. 4288e, 1071 (1967).

5. M. Ishii, *Thermo-Fluid Dynamic Theory of Two-Phase Flow* (collection de la Direction des Etudes et Recherches d'Electricite de France, Eyrolles, Paris, 1975).

6. T. G. Theofanous, T. Bohrer, M. Chen, and P. D. Patel, "Universal Solutions for Bubble Growth and the Influence of Microlayers," 15th Nat. Heat Transfer Conf., Paper 15, San Francisco, CA (August 1975).

7. D. Moalem and S. Sideman, "The Effect of Motion on Bubble Collapse," Int. J. Heat Mass Transfer **16**, 2321 (1973).

8. G. B. Wallis, H. J. Richter, and J. T. Kuo, "The Separated Flow Model for Two-Phase Flow," Electric Power Research Institute report NP-275, 88 (1976).

9. F. H. Harlow and A. A. Amsden, "A Numerical Fluid Dynamics Calculation Method for all Flow Speeds," J. Comput. Phys. **8**, 197 (1971).

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

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

## APPENDIX

## FORTRAN IV LISTING OF THE SOLA-LOOP CODE

```
*COMDECK,LOOPC
      COMMON
     1 ALPHA,      ASQ,        BNUM,       CDG,        CHL,        CHLI,
     1 CHV,        CHVI,       CPI,        CYCLE,      DELSTP,     DELT,
     1 DELTO,      DFVEL,      DPRST,      DTMN,       DTWPRT,     ECL,
     1 ECV,        EDL,        EDV,        ELHT,       EPSI,       ETEM,
     1 FLG,        GAMI,       IBREAK,     IL,         IMP,        ISTEP,
     1 ITER,       JB0,        JB1,        JB2,        JMH,        JML,
     1 JPB,        JPT,        JT1,        JT2,        KDO,        KL,
     1 KMO,        KOD,        KOM,        KOO,        KOP,        KOU,
     1 KPO,        KTOU,       KUO,        ML,         NSUBDT,     OMG,
     1 PHCH,       PNV,        PRST,       RG,         ROL,        SDTC,
     1 STABC,      T,          TC,         THC,        THC1,       TIMA,
     1 TSTART,     TWFIN,      TWPRT,      TWSTP,      VISL,       VISV
C
      COMMON
     1 CFRL(10, 8), CFRS(10, 8), DXS(10, 8),  DYS(10, 8),  GYS(10, 8),
     1 IDXS(10, 8), NJS(10, 8),  QJS(10, 8),  VIS(10, 8)
C
      COMMON
     1 BETA(200),   BETAK( 6),   DTIM(10),    DTIMK( 6),   DXK(30),
     1 DXKC( 6),    DYK(30),     DYKC( 6),    E(200),      EB( 5),
     1 EINC(10),    EK(30),      EKC( 6),     EN(200),     ENK(30),
     1 GAM(30),     IBOT(10),    ITOP(10),    JCEL(10),    JMHI(10),
     1 JMLI(10),    JPBI(10),    JPTI(10),    JREF(10),    JSF(200),
     1 JSXI(10),    KBOT(10),    KTOP(10),    LANG(10),    LBOT(10),
     1 LTOP(10),    MBOT(10),    MTOP(10),    P(200),      PB( 5),
     1 PI(10),      PK(30),      RO(200),     ROB( 5),     ROINC(10),
     1 ROK(30),     ROKC( 6),    RON(200),    RONK(30),    RV(200),
     1 RVB( 5),     RVINC(10),   RVK(30),     RVKC( 6),    RVN(200),
     1 RVNK(30),    TBRA(10),    TEMB( 5),    TEMI(10),    TEMK( 6),
     1 THB( 5),     THI(10),     THK( 6),     TIM(10),     TIMK( 6),
     1 V(200),      VD(200),     VDK(30),     VK(30),      VN(200),
     1 VNK(30),     VP(200),     VPK( 6),     WT(200)
C
      COMMON TITLE(8),NP,NS,NJ,NK,NM
C
      REAL IMP,NUA,NUC
      INTEGER CYCLE
C
  903 FORMAT(////)
  950 FORMAT(1H1)
  951 FORMAT(18H PROBLEM TITLE =     ,8A10 /)
  953 FORMAT(I5,7E15.7)
C
C
```

```
*COMDECK LOOPCP
C
C     USERS WITH COMPILERS CONTAINING THE PARAMETER OPTION SUBSTITUTE
C     THE FOLLOWING CARDS.
C     PARAMETER(NP=10,NS=8,NJ=200,NK=6,NM=5)
C
C     COMMON
C    1 CFRL(NP,NS), CFRS(NP,NS), DXS(NP,NS),  DYS(NP,NS),  GYS(NP,NS),
C    1 IDXS(NP,NS), NJS(NP,NS),  QJS(NP,NS),  TQJS(NP,NS), VIS(NP,NS)
C
C     COMMON
C    1 BETA(NJ),    BETAK(NK),   DTIM(NP),    DTIMK(NK),   DXK(5*NK),
C    1 DXKC(NK),    DYK(5*NK),   DYKC(NK),    E(NJ),       EB(NM),
C    1 EINC(NP),    EK(5*NK),    EKC(NK),     EN(NJ),      ENK(5*NK),
C    1 GAM(5*NK),   IBOT(NP),    ITOP(NP),    JCEL(NP),    JMHI(NP),
C    1 JMLI(NP),    JPBI(NP),    JPTI(NP),    JREF(NP),    JSF(NJ),
C    1 JSXI(NP),    KBOT(NP),    KTOP(NP),    LANG(NP),    LBOT(NP),
C    1 LTOP(NP),    MBOT(NP),    MTOP(NP),    P(NJ),       PB(NM),
C    1 PI(NP),      PK(5*NK),    RO(NJ),      ROB(NM),     ROINC(NP),
C    1 ROK(5*NK),   ROKC(NK),    RON(NJ),     RONK(5*NK),  RV(NJ),
C    1 RVB(NM),     RVINC(NP),   RVK(5*NK),   RVKC(NK),    RVN(NJ),
C    1 RVNK(5*NK),  TBRA(NP),    TEMB(NM),    TEMI(NP),    TEMK(NK),
C    1 THB(NM),     THI(NP),     THK(NK),     TIM(NP),     TIMK(NK),
C    1 V(NJ),       VD(NJ),      VDK(5*NK),   VK(5*NK),    VN(NJ),
C    1 VNK(5*NK),   VP(NJ),      VPK(NK),     WT(NJ)
C
C
```

```
*DECK,LOOP
      PROGRAM MAIN(INP,OUT,FSET9=OUT,FSET10=INP)
*CALL,LOOPC
C
*CALL LOOPCP
C
      NP=10
      NS=8
      NJ=200
      NK=6
      NM=5
C
C----UNITS-GM-CM-MS-K
C
C---- INITIALIZATION
      CALL INPUT
      CALL SETUP
      CALL JCTO
      CALL BC
      CALL PIPJCT
      CALL JCTPIP
      CALL DIAG
      CALL RESET
C
C     CALCULATIONAL LOOP
    1 TIMA=TIMA+DELTO
      DTMY=1.01
      IF(ITER.GT.5.OR.ISTEP.GT.NSUBDT) DTMY=0.99
      DELTO=DTMY*DELTO
      IF(T.GT.TWFIN) GO TO 4
C
      CALL ADVANCE
C
C     OUTPUT
      IF(T.LT.TWPRT)GO TO 2
C     LONG PRINT
      TWPRT=TWPRT+DTWPRT
      CALL DIAG
      GO TO 3
    2 IF(CYCLE-ISTEP.GT.1)GO TO 3
      CALL DIAG
    3 CONTINUE
      IF(T.LT.TWSTP) GO TO 1
C     SPECIAL PRINT
      CALL DIAGB
      TWSTP=TWSTP+DELSTP
      GO TO 1
    4 CONTINUE
      CALL EXIT(30)
      END
```

= = = = = = = = = = = = = = = = ///// = = = = = = = = = = = = = = = =

```
      SUBROUTINE ADVANCE
*CALL,LOOPC
```

```
C
      ISTEP=0
    1 CONTINUE
      ISTEP=ISTEP+1
C---- CALCULATE ONE TIME STEP
C---- TILDE CALCULATION
      CALL BREAK
      CALL PUMPA
      CALL TILDE
C---- PRESSURE ITERATION LOOP
      ITER=0
      FLG=IMP
    5 IF(FLG.EQ.0.) GO TO 2
      FLG=0.
      ITER=ITER+1
      CALL PITERP
      CALL PIPJCT
      CALL PITERJ
      CALL JCTPIP
      CALL BC
      IF(ITER.GE.505) FLG=0.
      GO TO 5
    2 IF(ITER.LT.505) GO TO 4
      WRITE(9,960)
      NEX=NEX+1
      IF(NEX.GE.5) T=1.E+10
    4 CONTINUE
C---- FINAL UPDATE
      CALL UPDATP
      CALL PIPJCT
      CALL UPDATJ
      CALL JCTPIP
      CALL BC
      IF(IMP.LT.0.5)CALL PSET
      CALL WALLT
      CALL RESET
      IF(T.LT.TIMA-DTMN/2.0)GO TO 1
      RETURN
  960 FORMAT(1X,9HITER =505)
      END
```

= = = = = = = = = = = = = = = = ///// = = = = = = = = = = = = = = = =

```
      FUNCTION ARAV(J1,J1P)
*CALL,LOOPC
C---- AREA AVERAGE
      ARU1=DXS(J1P)**2
      ARD1=DXS(J1)**2
      ARAV=2.*ARU1*ARD1/(ARU1+ARD1)
      RETURN
      END
```

= = = = = = = = = = = = = = = = ///// = = = = = = = = = = = = = = = =

```
      SUBROUTINE BC
*CALL,LOOPC
C---- BOUNDARY CONDITION CONTROL PROGRAM
      DO 32 I=1,IL
      IF(TIM(I).GT.T)GO TO 32
      DELT=DTIM(I)
      CALL JSET(I)
      I1=IBOT(I)
      GO TO (33,44),I1
   33 CONTINUE
      RO(JPB-1)=RO(JPB)
      RV(JPB-1)=RV(JPB)
      P(JPB-1)=P(JPB)
      E(JPB-1)=E(JPB)
      L1=LBOT(I)
      GO TO (172,174,176,178,176),L1
  172 V(JML)=0.
      GO TO 180
  174 V(JML)=0.
      GO TO 180
  176 IF(L1.NE.5) GO TO 177
      M1=MBOT(I)
      P(JPB-1)=PB(M1)
  177 IF(ITER.GT.1.AND.FLG.GT.0) GO TO 180
      M1=MBOT(I)
      IF(V(JML).LT.0.) GO TO 181
      E(JPB-1)=(2.0*EB(M1)-(1.0-ALPHA)*E(JPB))/(1.0+ALPHA)
      RO(JPB-1)=(2.0*ROB(M1)-(1.0-ALPHA)*RO(JPB))/(1.0+ALPHA)
      RV(JPB-1)=(2.0*RVB(M1)-(1.0-ALPHA)*RV(JPB))/(1.0+ALPHA)
  181 CONTINUE
      V(JML-1)=V(JML)
      GO TO 180
  178 V(JML-1)=V(JMH)
      RO(JPB-1)=RO(JPT)
      RV(JPB-1)=RO(JPT)
      P(JPB-1)=P(JPT)
      E(JPB-1)=E(JPT)
  180 CONTINUE
   44 I1=ITOP(I)
      GO TO (35,36),I1
   35 CONTINUE
      RO(JPT+1)=RO(JPT)
      RV(JPT+1)=RV(JPT)
      P(JPT+1)=P(JPT)
      E(JPT+1)=E(JPT)
      L1=LTOP(I)
      GO TO (152,154,156,158,156),L1
  152 V(JMH)=0.
      GO TO 161
  154 V(JMH)=0.
      GO TO 161
  156 M1=MTOP(I)
      IF(L1.NE.5) GO TO 157
      P(JPT+1)=PB(M1)
  157 IF(V(JMH).GT.0.) GO TO 182
      E(JPT+1)=(2.0*EB(M1)-(1.0-ALPHA)*E(JPT))/(1.0+ALPHA)
```

```
      RO(JPT+1)=(2.0*ROB(M1)-(1.0-ALPHA)*RO(JPT))/(1.0+ALPHA)           LOOP     157
      RV(JPT+1)=(2.0*RVB(M1)-(1.0-ALPHA)*RV(JPT))/(1.0+ALPHA)           LOOP     158
  182 CONTINUE                                                          LOOP     159
      V(JMH+1)=V(JMH)                                                   LOOP     160
      GO TO 161                                                         LOOP     161
  158 V(JMH)=V(JML-1)                                                   LOOP     162
      RO(JPT+1)=RO(JPB)                                                 LOOP     163
      RV(JPT+1)=RV(JPB)                                                 LOOP     164
      P(JPT+1)=P(JPB)                                                   LOOP     165
      E(JPT+1)=E(JPB)                                                   LOOP     166
  161 CONTINUE                                                          LOOP     167
   36 CONTINUE                                                          LOOP     168
   32 CONTINUE                                                          LOOP     169
      RETURN                                                            LOOP     170
      END                                                               LOOP     171


          = = = = = = = = = = = = = = = = ///// = = = = = = = = = = = = = = = =


      SUBROUTINE BREAK                                                  LOOP     172
*CALL,LOOPC                                                             LOOP     173
C---- BREAK CONTROL                                                     LOOP     174
      IF(T.LT.TBRA(IBREAK)) GO TO 1                                     LOOP     175
      IBREAK=IBREAK+1                                                   LOOP     176
C----READ INPUT BREAK DATA                                              LOOP     177
      CALL INPBR                                                        LOOP     178
C     CHECK FOR TIME STEP REDUCTION                                     LOOP     179
      IF(DELT.GE.DTMN) GO TO 1                                          LOOP     180
      DO 10 I=1,IL                                                      LOOP     181
   10 DTIM(I)=DELT                                                      LOOP     182
C                                                                       LOOP     183
      IF(KTOU.EQ.1) GO TO 1                                             LOOP     184
      DO 20 K=1,KL                                                      LOOP     185
   20 DTIMK(K)=DELT                                                     LOOP     186
C                                                                       LOOP     187
      DELTO=DELT                                                        LOOP     188
    1 CONTINUE                                                          LOOP     189
      RETURN                                                            LOOP     190
      END                                                               LOOP     191


          = = = = = = = = = = = = = = = = ///// = = = = = = = = = = = = = = = =


      FUNCTION DEMXC(J1,J1P)                                            LOOP     192
*CALL,LOOPC                                                             LOOP     193
C---- RADIUS FOR FRICTION FORMULA                                       LOOP     194
      I1=IDXS(J1)                                                       LOOP     195
      GO TO (1,2),I1                                                    LOOP     196
    1 DEMXC=.5*(DXS(J1)+DXS(J1P))                                       LOOP     197
      GO TO 3                                                           LOOP     198
C     ******** REPLACE NEXT STATEMENT WITH APPLICABLE DEFINITION OF     LOOP     199
C              HYDRAULIC RADIUS IF DIFFERENT THAN STATEMENT 1           LOOP     200
    2 DEMXC=0.                                                          LOOP     201
    3 CONTINUE                                                          LOOP     202
```

```
      RETURN
      END

          = = = = = = = = = = = = = = = = ///// = = = = = = = = = = = = = = = =

      SUBROUTINE DIAG
*CALL,LOOPC
C---- LONG DIAGNOSTIC PRINT
      IF(PRST.GT.T)RETURN
      WRITE(9,950)
      WRITE(9,951) TITLE
      CALL DIAGA
      WRITE(9,904)
      DO 1 I=1,IL
      CALL JSET(I)
      J1=JB1
      J2=JT1
      WRITE(9,47)
      DO 2 J=J1,J2
      WRITE(9,49)I,J,V(J),VD(J),P(J),RO(J),VP(J)
    2 CONTINUE
      WRITF(9,48)
      DO 6  J=J1,J2
      TH=(RUL-RO(J)+RV(J))/ROL
      TEM=TEMC(RO(J),RV(J),E(J),P(J))
      TSAT=STCAL(P(J))
      WRITE(9,50)I,J,RV(J),E(J),TH,TEM,TSAT,WT(J)
    6 CONTINUE
    1 CONTINUE
      GO TO (3,4),KTOU
    4 WRITE(9,904)
      WRITE(9,950)
      DO 5 K=1,KL
      CALL KSET(K)
      WRITE(9,953)K,PK(KOO),ROK(KOO),RVK(KOO),EK(KOO),VPK(K)
    5 CONTINUE
    3 CONTINUE
      WRITE(9,903)
      RETURN
   47 FORMAT(2X,1HI,3X,1HJ,11X,1HV,13X,2HVD,11X,1HP,11X,2HRO,11X,2HVP)
   48 FORMAT(2X,1HI,3X,1HJ,8X,2HRV,13X,1HE,12X,2HTH,12X,3HTEM,12X,
     14HTSAT,12X,2HWT)
   49 FORMAT(1X,I2,1X,I3,3X,1PE11.4,3X,1PE11.4,3X,1PE11.4,3X,1PE11.4,3X,
     11PE11.4)
   50 FORMAT(1X,I2,1X,I3,3X,1PE11.4,3X,1PE11.4,3X,1PE11.4,3X,1PE11.4,3X,
     11PE11.4,3X,1PE11.4)
  904 FORMAT(//)
  960 FORMAT(4X,1HK,7X,1HP,14X,2HRO,13X,2HRV,13X,1HE,13X,2HVP)
      END

          = = = = = = = = = = = = = = = = ///// = = = = = = = = = = = = = = = =

      SUBROUTINE DIAGA
*CALL,LOOPC
C
```

```
C     DIMENSION BFLUX(NP),TFLUX(NP)
      DIMENSION BFLUX(10),TFLUX(10)


C---- SHORT DIAGNOSTIC PRINT
      WRITE(9,50) ITER,T,DELTO,DTMN,CYCLE,ISTEP
      IF(IL.EQ.1) GO TO 5
      I1=IL
      IF(MOD(IL,2).EQ.0) GO TO 4
      I1=IL-1
    4 WRITE(9,55) (I,TIM(I),I,DTIM(I),I=1,I1)
      IF(I1.EQ.IL) GO TO 6
    5 WRITE(9,45) IL,TIM(IL),IL,DTIM(IL)
    6 IF(KL.EQ.0) RETURN
      IF(KL.EQ.1) GO TO 8
      K1=KL
      IF(MOD(KL,2).EQ.0) GO TO 7
      K1=KL-1
    7 WRITE(9,60) (K,TIMK(K),K,DTIMK(K),K=1,K1)
      IF(K1.EQ.KL) RETURN
    8 WRITE(9,65) KL,TIMK(KL),KL,DTIMK(KL)
      RETURN
      ENTRY DIAGB
C     SHORT PRINT OF MASS FLUX AT BOTTOM AND TOP OF PIPES
      DO 10 I=1,IL
      CALL JSET(I)
      JSB=JSF(JB1)
      JST=JSF(JT2)
      J1=JB1
      IF(V(JB1).LT.0.)J1=JB2
      J2=JT2
      IF(V(JT2).LT.0.)J2=JT1
      BFLUX(I)=RO(J1)*V(JB1)*(DXS(JSB)**2)*CPI
      TFLUX(I)=RO(J2)*V(JT2)*(DXS(JST)**2)*CPI
   10 CONTINUE
      WRITE(9,85) T,(I,BFLUX(I),TFLUX(I),I=1,IL)
      RETURN
   45 FORMAT(3X,4HTIM(,I2,2H)=,1PE12.5,3X,5HDTIM(,I2,2H)=,1PE12.5)
   50 FORMAT(1H0,5HITER=,I3,7H  TIME=,1PE12.5,8H  DELTO=,1PE12.5
     17H  DTMN=,1PE12.5,8H  CYCLE=,I6,8H  ISTEP=,I2)
   55 FORMAT(2(3X,4HTIM(,I2,2H)=,1PE12.5,3X,5HDTIM(,I2,2H)=,1PE12.5))
   60 FORMAT(2(3X,5HTIMK(,I2,2H)=,1PE12.5,3X,6HDTIMK(,I2,2H)=,1PE12.5))
   65 FORMAT(3X,5HTIMK(,I2,2H)=,1PE12.5,3X,6HDTIMK(,I2,2H)=,1PE12.5)
   85 FORMAT(1H0,7HTIME =  ,1PE12.5 /(5X,5HPIPE ,I2,3X,
     1 19HBOTTOM MASS FLUX =  ,1PE12.5,3X,16HTOP MASS FLUX = ,1PE12.5))
      END
```

= = = = = = = = = = = = = = = = = ///// = = = = = = = = = = = = = = = =

```
      SUBROUTINE DRIFT
*CALL,LOOPC
      IF(DFVEL.LT..5) GO TO 204
C---- COMPUTE DRIFT VELOCITY
      DO 1 I=1,IL
```

```
      IF(TIM(I).GT.T)GO TO 1
      DELT=DTIM(I)
      CALL JSET(I)
      DO 36 J=JML,JMH
      JC=JSF(J)
      JCP=JSF(J+1)
      DLYU2=DYS(JCP)+DYS(JC)
      ROU=(DYS(JCP)*RO(J)+DYS(JC)*RO(J+1))/DLYU2
      RVU=(DYS(JCP)*RV(J)+DYS(JC)*RV(J+1))/DLYU2
      IF(J.GT.JB1) GO TO 28
      RO1=RO(J+1)
      RV1=RV(J+1)
      GO TO 29
   28 IF(J.LT.JT2) GO TO 40
      RO1=RO(J)
      RV1=RV(J)
      GO TO 29
   40 RO1=ROU
      RV1=RVU
   29 CONTINUE
      TH=(ROL+RV1-RO1)/ROL
      IF(TH.GT.THC.AND.TH.LT.THC1) GO TO 4700
      VD(J)=0.
      GO TO 36
 4700 CONTINUE
      TH1=TH
      IF(TH1.GT.0.5)TH1=1.0-TH1
      RB=RBBNCAL(TH1)
      VDN=VD(J)
      AREA=3.*TH1/RB
      NUC=VISL
      IF(TH.GT..5) NUC=VISV
      NUA=NUC/(1.-TH1)**2.5
      VDO=DELT*(1.0/ROL-TH/RV1)/(0.5*DLYU2)*(P(J+1)-P(J))
      VDM=DELT*RO1/(RV1*(RO1-RV1))
      UDM=ABS(VD(J))+1.0E-10
 4950 UDMT=UDM
      VDKD=VDKCAL(RO1,UDM,TH1,AREA,NUA,RB)
      VDKDP=VDKPCAL(RO1,VD(J),TH1,AREA)
      VD(J)=VD(J)-(VD(J)+VDM*VDKD*VD(J)-VDO-VDN)
     1 /(1.0+VDM*(VDKD+VDKDP*VD(J)))
      UDM=ABS(VD(J))+1.E-10
      IF(ABS((UDMT-UDM)/(UDMT+UDM)).GT..01) GO TO 4950
   36 CONTINUE
    1 CONTINUE
  204 CONTINUE
      RETURN
      END
```

```
= = = = = = = = = = = = = = = = = = = = ///// = = = = = = = = = = = = = = = = = = = =
```

```
      FUNCTION EVCAL(T1)
*CALL.LOOPC
      EVCAL=ECV+CHV*(T1-TC)-CHV1*(T1-TC)**2
      RETURN
```

```
      END                                                               LOOP     353


          = = = = = = = = = = = = = = = = ///// = = = = = = = = = = = = = = = =


      FUNCTION ELCAL(T1)                                                LOOP     354
*CALL,LOOPC                                                             LOOP     355
      ELCAL=ECL+CHL*(T1-TC)-CHL1*(T1-TC)**2                             LOOP     356
      RETURN                                                            LOOP     357
      END                                                               LOOP     358


          = = = = = = = = = = = = = = = = ///// = = = = = = = = = = = = = = = =


      SUBROUTINE FRIC(I,J,V1)                                           LOOP     359
*CALL,LOOPC                                                             LOOP     360
C---- COMPUTE FRICTION EFFECT                                            LOOP     361
      IF(RG.LE.0.) RETURN                                               LOOP     362
      VA1=ABS(V(J))                                                     LOOP     363
      IF(VA1.GE.EPS1)GO TO 11                                           LOOP     364
      V1=V(J)                                                           LOOP     365
      RETURN                                                            LOOP     366
   11 CONTINUE                                                          LOOP     367
      V1=V(J)                                                           LOOP     368
      JC=JSF(J)                                                         LOOP     369
      JCP=JSF(J+1)                                                      LOOP     370
      IF(CFRS(JC).NE.0..OR.CFRL(JC).NE.0.) GO TO 1                      LOOP     371
      RETURN                                                            LOOP     372
    1 CONTINUE                                                          LOOP     373
      DLYU2=DYS(JCP)+DYS(JC)                                            LOOP     374
      ROU=(DYS(JCP)*RO(J)+DYS(JC)*RO(J+1))/DLYU2                        LOOP     375
      RVU=(DYS(JCP)*RV(J)+DYS(JC)*RV(J+1))/DLYU2                        LOOP     376
      IF(J.GT.JB1) GO TO 12                                             LOOP     377
      RV1=RV(J+1)                                                       LOOP     378
      RO1=RO(J+1)                                                       LOOP     379
      GO TO 13                                                          LOOP     380
   12 IF(J.LT.JT2) GO TO 14                                             LOOP     381
      RV1=RV(J)                                                         LOOP     382
      RO1=RO(J)                                                         LOOP     383
      GO TO 13                                                          LOOP     384
   14 RV1=RVU                                                           LOOP     385
      RO1=ROU                                                           LOOP     386
   13 CONTINUE                                                          LOOP.    387
      TH=(ROL+RV1-RO1)/ROL                                              LOOP     388
      IF(TH.LT.THC)TH=THC                                               LOOP     389
      IF(TH.GT.THC1)TH=THC1                                             LOOP     390
      CHI=RV1/RO1*(1.+(RO1-RV1)/RO1*VD(J)/V(J))**2                      LOOP     391
      DEMX=DEMXC(JC,JCP)                                                LOOP     392
      REN=2.*DEMX*VA1/VISL                                              LOOP     393
      RGR=.5*RG/DEMX                                                    LOOP     394
      FLA=.026*RGR**.225+.133*RGR                                       LOOP     395
      FLB=22.*RGR**.44                                                  LOOP     396
      FLC=1.62*RGR**.134                                                LOOP     397
      FLC2=(FLA+FLB/REN**FLC)                                           LOOP     398
```

```
      PHIS=1./(1.-TH)**1.75
      FLC=FLC2*(RO1/ROL)*(1.-CHI)**2/DEMX*PHIS
      IF(TH.EQ.THC1) FLC=FLC2/DEMX
      CFRS1=(DYS(JC)*CFRS(JC)+DYS(JCP)*CFRS(JCP))/DLYU2
      FLCT=2.*DELT*FLC*CFRS1
      CFRL1=(DYS(JC)*CFRL(JC)+DYS(JCP)*CFRL(JCP))/DLYU2
      FLCL=DELT*CFRL1*2.0*ROL/(RO(J)+RO(J+1))
      FLCT=FLCT+FLCL
      V1=SIGN(1.,V(J))*(-1.+(1.+2.*FLCT*VA1)**.5)/FLCT
      EE=.25*(VA1**2-V1**2)
      EN(J)=EN(J)+EE
      EN(J+1)=EN(J+1)+EE
      RETURN
      END
```

= = = = = = = = = = = = = = = = ///// = = = = = = = = = = = = = = = =

```
      SUBROUTINE INFNC(TIN,PIN,THIN,RO1,RV1,E1)
*CALL,LOOPC
C---- SETS UP STATE VARIABLES
      IF(THIN.LT.THC1) GO TO 36
C---- VAPOR STATE (P,T)
      TH=THIN
      EVT=EVCAL(TIN)
      ELT=ELCAL(TIN)
      RV1=RVCAL(EVT,PIN,TH)
      RO1=RV1+(1.-TH)*ROL
      E1=(RV1*EVT+(RO1-RV1) *ELT)/RO1
      GO TO 40
   36 CONTINUE
      TSAT=STCAL(PIN)
      EVSAT=EVCAL(TSAT)
      ELSAT=ELCAL(TSAT)
      IF(THIN.LT.THC) GO TO 38
C---- SATURATED STATE (P,TH)
      RV1=RVCAL(EVSAT,PIN,THIN)
      RO1=RV1+(1.-THIN)*ROL
      E1=(RV1*EVSAT+(1.-THIN)*ROL*ELSAT)/RO1
      TIN=TSAT
      GO TO 40
   38 CONTINUE
C---- LIQUID STATE (P,T)
      ELT=ELCAL(TIN)
      PSAT=SPCAL(TIN)
      EVSAT=EVCAL(TIN)
      RV1=RVCAL(EVSAT,PSAT,THC)
      TH=THCAL(EVSAT,PIN,RV1)
      RO1=RV1+(1.-TH)*ROL
      E1=(RV1*EVSAT+(1.-TH)*ROL*ELT)/RO1
   40 CONTINUE
      RETURN
      END
```

= = = = = = = = = = < = = = = = ///// = = = = = = = = = = = = = = = =

```
      SUBROUTINE INPUT
*CALL,LOOPC



C
      NAMELIST / DATUMS /
     1 ALPHA,      ASQ,        BNUM,       CDG,        CFRL,       CFRS,
     1 CHL,        CHLI,       CHV,        CHVI,       DELSTP,     DELT,
     1 DFVEL,      DPRST,      DTWPRT,     DXS,        DYS,        ECL,
     1 ECV,        EDL,        EDV,        ELHT,       EPSI,       ETEM,
     1 GAMI,       GYS,        IDXS,       IMP,        JMHI,       JMLI,
     1 JPBI,       JPTI,       KBOT,       KTOP,       LANG,       LBOT,
     1 LTOP,       MBOT,       MTOP,       NJS,        NSUBDT,     OMG,
     1 PHCH,       PB,         PI,         QJS,        RG,         ROL,
     1 STABC,      TBRA,       TC,         TEMB,       TEMI,       THB,
     1 THC,        THI,        TITLE,      TSTART,     TWFIN,      TWSTP,
     1 VIS,        VISL,       VISV
C
      N1=NP
      N2=NS
      N3=NJ
      N4=NK
      N5=NM
C
C     ZERO OUT THE ARRAYS  ****     *****     ****
      LENA=9*N1*N2
      LENB=25*N1+76*N4+14*N3+6*N5
C     ALTER IF ADDITONAL VARIABLES ADDED TO COMMON
      DO 100 L=1,LENA
  100 CFRL(L)=0.
      DO 105 L=1,LENB
  105 BETA(L)=0.
C     ******  ******     *****
C---- DEFAULT  VALUES SETUP
      ALPHA=1.
      ASQ=1.234E+4
      BNUM=10000.
      CDG=.5
      CHL=44.34
      CHLI=.0129
      CHV=6.67
      CHVI=.0302
      CPI=3.14159265
      DELSTP=100.
      DELT=.001
      DFVEL=0.
      DPRST=0.
      DTWPRT=500.
      ECL=.4174E+4
      ECV=2.506E+4
      EDL=1.6E-6
      EDV=1.6E-7
      ELHT=1.76E+4
      EPSI=.001
      ETEM=1.
      GAMI=.07
      IMP=1.0
```

```
      NSUBDT=1
      OMG=1.
      PHCH=1.
      RG=0.
      ROL=.958
      STABC=4.0
      TC=373.
      THC=.001
      TITLE(1)=10H
      TITLE(2)=10H
      TITLE(3)=10H
      TITLE(4)=10H
      TITLE(5)=10H
      TITLE(6)=10H
      TITLE(7)=10H
      TITLE(8)=10H
      TSTART=0.
      TWFIN=10000.
      TWSTP=0.
      VISL=3.0E-6
      VISV=2.0E-4
      DO 15 I=1,N1
      JMLI(I)=1
      JMHI(I)=1
      JPBI(I)=1
      JPTI(I)=1
   15 CONTINUE
      DO 18 N=1,N1
      TBRA(N)=1.E+20
      DO 18 M=1,N2
      CFRS(N,M)=1.
      IDXS(N,M)=1
   18 CONTINUE
C
      READ(10,DATUMS)
C
C---- LIST INPUT CONSTANTS
      WRITE(9,950)
      WRITE(9,951) TITLE
      WRITE(9,988)
      WRITE(9,921)ALPHA
      WRITE(9,971)ASQ
      WRITE(9,968)BNUM
      WRITE(9,923)CDG
      WRITE(9,973)CHL
      WRITE(9,935)CHLI
      WRITE(9,924)CHV
      WRITE(9,936)CHVI
      WRITE(9,934)CPI
      WRITE(9,965)DELT
      WRITE(9,938)DELSTP
      WRITE(9,928)DFVEL
      WRITE(9,940)DPRST
      WRITE(9,964)DTWPRT
      WRITE(9,926)ECL
      WRITE(9,925)ECV
      WRITE(9,931)EDL
      WRITE(9,930)EDV
```

```
      WRITE(9,929)ELHT
      WRITE(9,966)EPSI
      WRITE(9,961)ETEM
      WRITE(9,962)GAMI
      WRITE(9,937)IMP
      WRITE(9,941)NSUBDT
      WRITE(9,920)OMG
      WRITE(9,932)PHCH
      WRITE(9,972)RG
      WRITE(9,967)ROL
      WRITE(9,942) STABC
      WRITE(9,927)TC
      WRITE(9,960)THC
      WRITE(9,970)TSTART
      WRITE(9,963)TWFIN
      WRITE(9,939)TWSTP
      WRITE(9,974)VISL
      WRITE(9,975)VISV
      WRITE(9,903)
C
C     CALCULATE  NUMBER OF PIPES,JOINTS,AND BOUNDARY SETS
      DO 30 I=1,N1
      JCEL(I)=0
      DO 25 J=1,N2
      IF(NJS(I,J).EQ.0) GO TO 25
      JSXI(I)=J
      JCEL(I)=JCEL(I)+NJS(I,J)
      IL=I
   25 CONTINUE
   30 CONTINUE
      KL=0
      DO 35 I=1,IL
      KL=MAX0(KL,KBOT(I),KTOP(I))
   35 CONTINUE
      ML=0
      DO 40 M=1,N5
      IF((TEMB(M)+PB(M)+ROB(M)).NE.0.) ML=M
   40 CONTINUE
C
C---- BOUNDARY CONDITION CONSTANTS
      IF(ML.EQ.0) GO TO 13
      WRITE(9,950)
      WRITE(9,976)ML
      WRITE(9,977)
      DO 9 M=1,ML
      WRITE(9,953)M,TEMB(M),PB(M),THB(M)
    9 CONTINUE
   13 CONTINUE
C---- ELEMENT DEPENDENT CONSTANTS
      IF(IL.NE.0) GO TO 14
      WRITE(9,989)
      CALL EXIT(20)
   14 CONTINUE
      WRITE(9,903)
      WRITE(9,978)IL
      WRITE(9,979)
      DO 10 I=1,IL
      WRITE(9,980)I,JCEL(I),LANG(I),JSXI(I),TEMI(I),PI(I),THI(I)
```

```
   10 CONTINUE
C---- JOINT QUANTITIES
      IF(KL.EQ.0) GO TO 3
      DO 50 K=1,KL
      NPIPX=0
      NPIPY=0
      DO 49 I=1,IL
      IF(KBOT(I).NE.K) GO TO 46
      IF(LANG(I).EQ.1.OR.LANG(I).EQ.3) GO TO 45
      NPIPY=NPIPY+1
      IF(NPIPY.EQ.1) DYK(K)=DYS(I,1)
      IF(DYS(I,1).LT.DYK(K)) DYK(K)=DYS(I,1)
      RADY=DXS(I,1)
      GO TO 48
   45 NPIPX=NPIPX+1
      IF(NPIPX.EQ.1) DXK(K)=DYS(I,1)
      IF(DYS(I,1).LT.DXK(K)) DXK(K)=DYS(I,1)
      RADX=DXS(I,1)
      GO TO 48
   46 IF(KTOP(I).NE.K) GO TO 49
      NSEGI=JSXI(I)
      IF(LANG(I).EQ.1.OR.LANG(I).EQ.3) GO TO 47
      NPIPY=NPIPY+1
      IF(NPIPY.EQ.1) DYK(K)=DYS(I,NSEGI)
      IF(DYS(I,NSEGI).LT.DYK(K)) DYK(K)=DYS(I,NSEGI)
      RADY=DXS(I,NSEGI)
      GO TO 48
   47 NPIPX=NPIPX+1
      IF(NPIPX.EQ.1) DXK(K)=DYS(I,NSEGI)
      IF(DYS(I,NSEGI).LT.DXK(K))DXK(K)=DYS(I,NSEGI)
      RADX=DXS(I,NSEGI)
   48 PK(K)=PK(K) + PI(I)
      TEMK(K)=TEMK(K) + TEMI(I)
      THK(K)=THK(K) + THI(I)
   49 CONTINUE
      RNPIP=1.0/FLOAT(NPIPX + NPIPY)
      PK(K)=PK(K)*RNPIP
      TEMK(K)=TEMK(K)*RNPIP
      THK(K)=THK(K)*RNPIP
      IF(NPIPX.EQ.0)RADX=SQRT(RADY*DYK(K))
      IF(NPIPY.EQ.0)RADY=SQRT(RADX*DXK(K))
      IF(NPIPX.EQ.0)DXK(K)=RADY
      IF(NPIPY.EQ.0)DYK(K)=RADX
      DZONE=RADY**2/DXK(K)
      DZTWO=RADX**2/DYK(K)
      DZMAX=AMAX1(DZONE,DZTWO)
      DXK(K)=RADY**2/DZMAX
      DYK(K)=RADX**2/DZMAX
   50 CONTINUE
      WRITE(9,903)
      WRITE(9,984)KL
      WRITE(9,985)
      DO 5 K=1,KL
      WRITE(9,953)K,DXK(K),DYK(K),TEMK(K),PK(K),THK(K)
    5 CONTINUE
    3 CONTINUE
C---- MESH DEPENDENT PROPERTIES
      DO 60 I=1,IL
```

```
      IF(KBOT(I).EQ.0) GO TO 65
      JL=JSXI(I)
      DO 54 J=1,JL
      L=JL+2-J
      NJS(I,L)=NJS(I,L-1)
      IDXS(I,L)=IDXS(I,L-1)
      DXS(I,L)=DXS(I,L-1)
      DYS(I,L)=DYS(I,L-1)
      VIS(I,L)=VIS(I,L-1)
      GYS(I,L)=GYS(I,L-1)
      CFRL(I,L)=CFRL(I,L-1)
      CFRS(I,L)=CFRS(I,L-1)
   54 CONTINUE
      JSXI(I)=JSXI(I)+1
      JCEL(I)=JCEL(I)+3
      NJS(I,1)=3
      K=KBOT(I)
      DYS(I,1)=DXK(K)
      IF(LANG(I).EQ.2.OR.LANG(I).EQ.4) DYS(I,1)=DYK(K)
      GO TO 55
   65 NJS(I,1)=NJS(I,1)+2
      JCEL(I)=JCEL(I)+2
   55 IF(KTOP(I).EQ.0) GO TO 75
      JSXI(I)=JSXI(I)+1
      L=JSXI(I)
      NJS(I,L)=2
      IDXS(I,L)=IDXS(I,L-1)
      DXS(I,L)=DXS(I,L-1)
      VIS(I,L)=VIS(I,L-1)
      GYS(I,L)=GYS(I,L-1)
      CFRL(I,L)=CFRL(I,L-1)
      CFRS(I,L)=CFRS(I,L-1)
      QJS(I,L)=QJS(I,L-1)
      K=KTOP(I)
      DYS(I,L)=DXK(K)
      IF(LANG(I).EQ.2.OR.LANG(I).EQ.4) DYS(I,L)=DYK(K)
      JCEL(I)=JCEL(I)+2
      GO TO 60
   75 L=JSXI(I)
      NJS(I,L)=NJS(I,L)+1
      JCEL(I)=JCEL(I)+1
   60 CONTINUE
      WRITE(9,903)
      WRITE(9,981)
      WRITE(9,983)
      DO 4 I=1,IL
      JL=JSXI(I)
      DO 4 J=1,JL
      JS=I+(J-1)*N1
      WRITE(9,982)I,J,NJS(JS),IDXS(JS),DXS(JS),DYS(JS),VIS(JS),GYS(JS),
     1CFRL(JS),CFRS(JS),QJS(JS)
    4 CONTINUE
      RETURN
C
      ENTRY INPBR
C
      READ(10,DATUMS)
      RETURN
```

```
C
  920 FORMAT(10X, 9H   OMG = ,1PE15.7)
  921 FORMAT(10X, 9H ALPHA = ,1PE15.7)
  923 FORMAT(10X, 9H   CDG = ,1PE15.7)
  924 FORMAT(10X, 9H   CHV = ,1PE15.7)
  925 FORMAT(10X, 9H   ECV = ,1PE15.7)
  926 FORMAT(10X, 9H   ECL = ,1PE15.7)
  927 FORMAT(10X, 9H    TC = ,1PE15.7)
  928 FORMAT(10X, 9H DFVEL = ,1PE15.7)
  929 FORMAT(10X, 9H  ELHT = ,1PE15.7)
  930 FORMAT(10X, 9H   EDV = ,1PE15.7)
  931 FORMAT(10X, 9H   EDL = ,1PE15.7)
  932 FORMAT(10X, 9H  PHCH = ,1PE15.7)
  934 FORMAT(10X, 9H   CPI = ,1PE15.7)
  935 FORMAT(10X, 9H  CHLI = ,1PE15.7)
  936 FORMAT(10X, 9H  CHVI = ,1PE15.7)
  937 FORMAT(10X, 9H   IMP = ,1PE15.7)
  938 FORMAT(10X, 9HDELSTP = ,1PE15.7)
  939 FORMAT(10X, 9H TWSTP = ,1PE15.7)
  940 FORMAT(10X, 9H DPRST = ,1PE15.7)
  941 FORMAT(10X, 9HNSUBDT = ,I3)
  942 FORMAT(10X, 9H STABC = ,1PE15.7)
  960 FORMAT(10X, 9H   THC = ,1PE15.7)
  961 FORMAT(10X, 9H  ETEM = ,1PE15.7)
  962 FORMAT(10X, 9H  GAMI = ,1PE15.7)
  963 FORMAT(10X, 9H TWFIN = ,1PE15.7)
  964 FORMAT(10X, 9HDTWPRT = ,1PE15.7)
  965 FORMAT(10X, 9H  DELT = ,1PE15.7)
  966 FORMAT(10X, 9H  EPSI = ,1PE15.7)
  967 FORMAT(10X, 9H   ROL = ,1PE15.7)
  968 FORMAT(10X, 9H  BNUM = ,1PE15.7)
  970 FORMAT(10X, 9HTSTART = ,1PE15.7)
  971 FORMAT(10X, 9H   ASQ = ,1PE15.7)
  972 FORMAT(10X, 9H    RG = ,1PE15.7)
  973 FORMAT(10X, 9H   CHL = ,1PE15.7)
  974 FORMAT(10X, 9H  VISL = ,1PE15.7)
  975 FORMAT(10X, 9H  VISV = ,1PE15.7)
  976 FORMAT(10X,39HBOUNDARY CONDITION INPUT          ML = ,I5//)
  977 FORMAT(42H    M     TEMB            PB            THB//)
  978 FORMAT(10X,25HPIPE INPUT          IL = ,I5//)
  979 FORMAT(43H    I  JCEL LANG JSXI     TEMI            PI    ,
     114H          THI//)
  980 FORMAT(4I5,6E15.7)
  981 FORMAT(10X,25HMESH DEPENDENT PROPERTIES//)
  982 FORMAT(4I5,7E14.6)
  983 FORMAT(45H    I   JS   NJS IDXS      DXS            DYS   ,
     156H          VIS            GYS            CFRL           CFRS   ,
     2 10X,3HQJS     //)
  984 FORMAT(10X,30HJOINT QUANTITIES          KL = ,I5//)
  985 FORMAT(45H    K      DXK            DYK            TEMK   ,
     125H          PK            THK//)
  988 FORMAT(10X,15HINPUT CONSTANTS//)
  989 FORMAT(38H ERROR *** THIS SYSTEM HAS NO PIPES         )
      END
```

= = = = = = = = = = = = = = ///// = = = = = = = = = = = = = =

```
      SUBROUTINE JCTO
*CALL,LOOPC
C---- INITIALIZATION OF JUNCTION CELL
C---- CREATES AND INITIALIZES ADJACENT STORAGE CELLS
      GO TO (17,18),KTOU
   18 DO 30 K=1,KL
      PK(K)=0.
      DXKC(K)=DXK(K)
      DYKC(K)=DYK(K)
      DYK(K)=0.
      DXK(K)=0.
   30 CONTINUE
      DO 1 K=1,KL
      IF(TIMK(K).GT.T)GO TO 1
      DELT=DTIMK(K)
      CALL KSET(K)
      DXK(KOO)=DXKC(K)
      DXK(KPO)=DXKC(K)
      DXK(KOP)=DXKC(K)
      DXK(KMO)=DXKC(K)
      DXK(KOM)=DXKC(K)
      DYK(KOO)=DYKC(K)
      DYK(KPO)=DYKC(K)
      DYK(KOP)=DYKC(K)
      DYK(KMO)=DYKC(K)
      DYK(KOM)=DYKC(K)
      ROK(KOO)=ROKC(K)
      ROK(KPO)=ROKC(K)
      ROK(KOP)=ROKC(K)
      ROK(KMO)=ROKC(K)
      ROK(KOM)=ROKC(K)
      RVK(KOO)=RVKC(K)
      RVK(KPO)=RVKC(K)
      RVK(KOP)=RVKC(K)
      RVK(KMO)=RVKC(K)
      RVK(KOM)=RVKC(K)
      EK(KOO)=EKC(K)
      EK(KPO)=EKC(K)
      EK(KOP)=EKC(K)
      EK(KMO)=EKC(K)
      EK(KOM)=EKC(K)
      PK(KOO)=PCAL(EK(KOO),ROK(KOO),RVK(KOO))
      PK(KPO)=PK(KOO)
      PK(KOP)=PK(KOO)
      PK(KMO)=PK(KOO)
      PK(KOM)=PK(KOO)
      VK(KUO)=0.
      VK(KOU)=0.
      VK(KDO)=0.
      VK(KOD)=0.
      GAM(KUO)=0.
      GAM(KOU)=0.
      GAM(KDO)=0.
      GAM(KOD)=0.
    1 CONTINUE
      DO 2 I=1,IL
```

```
      IF(TIM(I).GT.T)GO TO 2
      DELT=DTIM(I)
      CALL JSET(I)
      JSB=JSF(JB1)
      JST=JSF(JT1)
      I1=IBOT(I)
      GO TO (3,4),I1
    4 K1=KBOT(I)
      L1=LANG(I)
      CALL KSET(K1)
      GO TO (5,6,7,8),L1
    5 CONTINUE
      DXK(KPO)=DYS(JSB)
      GAM(KUO)=1.
      GO TO 9
    6 CONTINUE
      DYK(KOP)=DYS(JSB)
      GAM(KOU)=1.
      GO TO 9
    7 CONTINUE
      DXK(KMO)=DYS(JSB)
      GAM(KDO)=1.
      GO TO 9
    8 CONTINUE
      DYK(KOM)=DYS(JSB)
      GAM(KOD)=1.
    9 CONTINUE
    3 CONTINUE
      I1=ITOP(I)
      GO TO (10,11),I1
   11 K1=KTOP(I)
      L1=LANG(I)
      CALL KSET(K1)
      GO TO (14,15,12,13),L1
   12 CONTINUE
      DXK(KPO)=DYS(JST)
      GAM(KUO)=1.
      GO TO 16
   13 CONTINUE
      DYK(KOP)=DYS(JST)
      GAM(KOU)=1.
      GO TO 16
   14 CONTINUE
      DXK(KMO)=DYS(JST)
      GAM(KDO)=1.
      GO TO 16
   15 CONTINUE
      DYK(KOM)=DYS(JST)
      GAM(KOD)=1.
   16 CONTINUE
   10 CONTINUE
    2 CONTINUE
   17 CONTINUE
      RETURN
      END
```

= = = = = = = = = = = = = = = = = ///// = = = = = = = = = = = = = = = = =

```
      SUBROUTINE JCTPIP
*CALL,LOOPC


C---- JUNCTION TO PIPE SHIFT
      DO 1 I=1,IL
      CALL JSET(I)
      JSB=JSF(JB1)
      JST=JSF(JT2)
      I1=IBOT(I)
      GO TO (2,3),I1
    3 K1=KBOT(I)
      L1=LANG(I)
      IF(TIMK(K1).GT.T)GO TO 2
      CALL KSET(K1)
      RO(JB1)=ROK(KOO)
      RV(JB1)=RVK(KOO)
      E(JB1)=EK(KOO)
      P(JB1)=PK(KOO)
      VP(JB1)=VPK(K1)
      GO TO (4,5,6,7),L1
    4 V(JB0)=(DYS(JSB)*VK(KDO)+(DXK(KOO)-DYS(JSB))*VK(KUO))/DXK(KOO)
      V(JB1)=VK(KUO)
      GO TO 8
    5 V(JB0)=(DYS(JSB)*VK(KOD)+(DYK(KOO)-DYS(JSB))*VK(KOU))/DYK(KOO)
      V(JB1)=VK(KOU)
      GO TO 8
    6 V(JB0)=-(DYS(JSB)*VK(KUO)+(DXK(KOO)-DYS(JSB))*VK(KDO))/DXK(KOO)
      V(JB1)=-VK(KDO)
      GO TO 8
    7 V(JB0)=-(DYS(JSB)*VK(KOU)+(DYK(KOO)-DYS(JSB))*VK(KOD))/DYK(KUO)
      V(JB1)=-VK(KOD)
    8 CONTINUE
    2 CONTINUE
      I1=ITOP(I)
      GO TO (9,10),I1
   10 K1=KTOP(I)
      L1=LANG(I)
      IF(TIMK(K1).GT.T)GO TO 9
      CALL KSET(K1)
      RO(JT1)=ROK(KOO)
      RV(JT1)=RVK(KOO)
      E(JT1)=EK(KOO)
      P(JT1)=PK(KOO)
      VP(JT1)=VPK(K1)
      GO TO (13,14,11,12),L1
   11 V(JT1)=-(DYS(JST)*VK(KDO)+(DXK(KOO)-DYS(JST))*VK(KUO))/DXK(KOO)
      V(JT2)=-VK(KUO)
      GO TO 15
   12 V(JT1)=-(DYS(JST)*VK(KOD)+(DYK(KOO)-DYS(JST))*VK(KOU))/DYK(KOO)
      V(JT2)=-VK(KOU)
      GO TO 15
   13 V(JT1)=(DYS(JST)*VK(KUO)+(DXK(KOO)-DYS(JST))*VK(KDO))/DXK(KOO)
      V(JT2)=VK(KDO)
      GO TO 15
   14 V(JT1)=(DYS(JST)*VK(KOU)+(DYK(KOO)-DYS(JST))*VK(KOD))/DYK(KOO)
      V(JT2)=VK(KOD)
```

```
   15 CONTINUE
    9 CONTINUE
    1 CONTINUE
      RETURN
      END


          = = = = = = = = = = = = = = = = ///// = = = = = = = = = = = = = = = =


      SUBROUTINE JSET(I)
*CALL,LOOPC
C---- PIPE INDEX ALGEBRA
      JB0=JREF(I)+1
      JB1=JB0+1
      JB2=JB1+1
      JT1=JB0+JCEL(I)-1
      JT2=JT1-1
      JML=JB0+JMLI(I)
      JMH=JT1-JMHI(I)
      JPB=JB1+JPBI(I)
      JPT=JT1-JPTI(I)
      RETURN
      END


          = = = = = = = = = = = = = = = = ///// = = = = = = = = = = = = = = = =


      SUBROUTINE KSET(K)
*CALL,LOOPC
C---- JUNCTION INDEX ALGEBRA
      K1=(K-1)*5
      K00=K1+1
      KP0=K00+1
      K0P=K00+2
      KM0=K00+3
      K0M=K00+4
      KU0=K1+1
      K0U=KU0+1
      KD0=KU0+2
      K0D=KU0+3
      RETURN
      END


          = = = = = = = = = = = = = = = = ///// = = = = = = = = = = = = = = = =


      SUBROUTINE LOGCPR
*CALL,LOOPC
      WRITE(9,950)
      WRITE(9,960)
      WRITE(9,961)
      DO 1 I=1,IL
```

```
      CALL JSET(I)
      WRITE(9,905)I,JREF(I),JCEL(I),JMLI(I),JMHI(I),JPBI(I),JPTI(I)
     1,KBOT(I),KTOP(I),LBOT(I),LTOP(I),MBOT(I),MTOP(I),JB1,JT1,IBOT(I),
     2ITOP(I)
    1 CONTINUE
      WRITE(9,903)
      WRITE(9,962)
      J=0
      DO 40 I=1,IL
      JSL=JSXI(I)
      DO 35 JS=1,JSL
      NI=NJS(I,JS)
      DO 30 N=1,NI
      J=J+1
      WRITE(9,905)I,JS,J,JSF(J)
   30 CONTINUE
   35 CONTINUE
   40 CONTINUE
      RETURN
  905 FORMAT(20I5)
  960 FORMAT(10X,11HLOGIC PRINT//)
  961 FORMAT(52H    I  JREF JCEL JMLI JMHI JPBI JPTI KBOT KTOP LBOT  ,
     135HLTOP MBOT MTOP  JB1  JT1 IBOT ITOP //)
  962 FORMAT(25H   IP   IS    J  JSF(J)   //)
      END
```

= = = = = = = = = = = = = = = = ///// = = = = = = = = = = = = = = =

```
      FUNCTION PCAL(ET,ROT,RVT)
*CALL,LOOPC
C---- PRESSURE CALCULATION
      TH=(ROL+RVT-ROT)/ROL
      THTM=TH
      IF(TH.LT.THC)THTM=THC
      ETEMT=ETEM
      IF(TH.GT.THC1)ETEMT=1.
      TEM=TEMC(ROT,RVT,ET,0.)
      IF(ETEMT.LT..5)TEM=255.2
      EV1=EVCAL(TEM)
      PT=GAM1•RVT•EV1/THTM+ASQ•ROL•(THTM-TH)
      PTT=PT
      IF(ETEMT.GT..5) GO TO 6265
      PCC0=117.8•GAM1•RVT/THTM
      PCC=PCC0•(CHV-CHV1•(510.4-2.•TC))
      PCC1=PCC0•CHV1•117.8
      PT=PTT+2.•(.223•PCC)••1.287
 6260 PTG=PT
      PTA=PCC•PTG••.223-PCC1•PTG••.446
      PT=PTG+(PTT+PTA-PTG)/(1.-.223•PTA/PTG)
      IF(ABS((PT-PTG)/(PT+PTG)).GT..01) GO TO 6260
 6265 CONTINUE
      PCAL=PT
      RETURN
      END
```

```
= = = = = = = = = = = = = = = = ///// = = = = = = = = = = = = = = = =

      SUBROUTINE PIPJCT
*CALL,LOOPC
C---- PIPE TO JUNCTION SHIFT
      DO 1 I=1,IL
      IF(TIM(I).GT.T)GO TO 1
      CALL JSET(I)
      I1=IBOT(I)
      GO TO (2,3),I1
    3 K1=KBOT(I)
      L1=LANG(I)
      CALL KSET(K1)
      GO TO (4,5,6,7),L1
    4 ROK(KPO)=RO(JB2)
      RVK(KPO)=RV(JB2)
      EK(KPO)=E(JB2)
      PK(KPO)=P(JB2)
      VK(KUO)=V(JB1)
      VDK(KUO)=VD(JB1)
      GO TO 8
    5 ROK(KOP)=RO(JB2)
      RVK(KOP)=RV(JB2)
      EK(KOP)=E(JB2)
      PK(KOP)=P(JB2)
      VK(KOU)=V(JB1)
      VDK(KOU)=VD(JB1)
      GO TO 8
    6 ROK(KMO)=RO(JB2)
      RVK(KMO)=RV(JB2)
      EK(KMO)=E(JB2)
      PK(KMO)=P(JB2)
      VK(KDO)=-V(JB1)
      VDK(KDO)=-VD(JB1)
      GO TO 8
    7 ROK(KOM)=RO(JB2)
      RVK(KOM)=RV(JB2)
      EK(KOM)=E(JB2)
      PK(KOM)=P(JB2)
      VK(KOD)=-V(JB1)
      VDK(KOD)=-VD(JB1)
    8 CONTINUE
    2 CONTINUE
      I1=ITOP(I)
      GO TO (9,10),I1
   10 K1=KTOP(I)
      L1=LANG(I)
      CALL KSET(K1)
      GO TO (13,14,11,12),L1
   11 ROK(KPO)=RO(JT2)
      RVK(KPO)=RV(JT2)
      EK(KPO)=E(JT2)
      PK(KPO)=P(JT2)
      VK(KUO)=-V(JT2)
      VDK(KUO)=-VD(JT2)
```

```
      GO TO 15
   12 ROK(KOP)=RO(JT2)
      RVK(KOP)=RV(JT2)
      EK(KOP)=E(JT2)
      PK(KOP)=P(JT2)
      VK(KOU)=-V(JT2)
      VDK(KOU)=-VD(JT2)
      GO TO 15
   13 ROK(KMO)=RO(JT2)
      RVK(KMO)=RV(JT2)
      EK(KMO)=E(JT2)
      PK(KMO)=P(JT2)
      VK(KDO)=V(JT2)
      VDK(KDO)=VD(JT2)
      GO TO 15
   14 ROK(KOM)=RO(JT2)
      RVK(KOM)=RV(JT2)
      EK(KOM)=E(JT2)
      PK(KOM)=P(JT2)
      VK(KOD)=V(JT2)
      VDK(KOD)=VD(JT2)
   15 CONTINUE
    9 CONTINUE
    1 CONTINUE
      RETURN
      END
```

= = = = = = = = = = = = = = = = = ///// = = = = = = = = = = = = = = = = =

```
      SUBROUTINE PITERJ
*CALL,LOOPC
C---- PRESSURE ITERATIONS FOR JUNCTION CELLS
      GO TO(100,200),KTOU
  200 CONTINUE
      DO 1 K=1,KL
      IF(TIMK(K).GT.T)GO TO 1
      DELT=DTIMK(K)
      CALL KSET(K)
      SM=-1.0E+10
      ICT=0
      XMX=1.0E+10
      XMN=0.0
      PBB=0.0
   10 PBAR=PK(KOO)
      D=DELT*((VK(KUO)-VK(KDO))/DXK(KOO)+(VK(KOU)-VK(KOD))/DYK(KOO))
      ROT=RONK(KOO)/(1.+C)
      RVT=RVK(KOO)/(1.+D)
      ET=ENK(KOO)-PK(KOO)*D/ROK(KOO)
      PT=PCAL(ET,ROT,RVT)
      S=PBAR-PT
      IF(ICT.NE.0.AND.S.NE.SM) BETAK(K)=(PBAR-PBB)/(S-SM)
      PK(KOO)=PBAR-BETAK(K)*S
      IF(S.GE.0.0)GO TO 20
      XMN=PBAR
      IF(PK(KOO).GE.XMX)PK(KOO)=0.5*(XMN+XMX)
```

```
      GO TO 30
   20 XMX=PBAR
      IF(PK(KOO).LE.XMN)PK(KOO)=0.5*(XMN+XMX)
   30 CONTINUE
      DELP=PK(KOO)-PBAR
      IF(ABS(DELP).LE.EPSI*PK(KOO))ICT=100
      DLXR2=DXK(KOO)+DXK(KPO)
      DLXL2=DXK(KMO)+DXK(KOO)
      DLYT2=DYK(KOO)+DYK(KOP)
      DLYB2=DYK(KOM)+DYK(KOO)
      ROKR=(DXK(KPO)*ROK(KOO)+DXK(KOO)*ROK(KPO))/DLXR2
      ROKL=(DXK(KMO)*ROK(KOO)+DXK(KOO)*ROK(KMO))/DLXL2
      ROKT=(DYK(KOP)*ROK(KOO)+DYK(KOO)*ROK(KOP))/DLYT2
      ROKB=(DYK(KOM)*ROK(KOO)+DYK(KOO)*ROK(KOM))/DLYB2
      DUR=2.*DELT*DELP/(ROKR*DLXR2)
      DUL=-2.*DELT*DELP/(ROKL*DLXL2)
      DVT=2.*DELT*DELP/(ROKT*DLYT2)
      DVB=-2.*DELT*DELP/(ROKB*DLYB2)
      DUR=DUR*GAM(KUO)
      DUL=DUL*GAM(KDO)
      DVT=DVT*GAM(KOU)
      DVB=DVB*GAM(KOD)
      VK(KUO)=VK(KUO)+DUR
      VK(KDO)=VK(KDO)+DUL
      VK(KOU)=VK(KOU)+DVT
      VK(KOD)=VK(KOD)+DVB
      SM=S
      PBB=PBAR
      ICT=ICT+1
      IF(ICT.GT.10)GO TO 1
      FLG=1.0
      GO TO 10
    1 CONTINUE
  100 CONTINUE
      RETURN
      END


          = = = = = = = = = = = = = = = = ///// = = = = = = = = = = = = = = = =


      SUBROUTINE PITERP
*CALL,LOOPC
C---- PRESSURE ITERATION FOR PIPES
      DO 1 I=1,IL
      IF(TIM(I).GT.T)GO TO 1
      DELT=DTIM(I)
      CALL JSET(I)
      DO 100 J=JPB,JPT
      JC=JSF(J)
      JCP=JSF(J+1)
      JCM=JSF(J-1)
      ARU=ARAV(JC,JCP)
      ARD=ARAV(JCM,JC)
      SM=-1.0E+10
      ICT=0
      XMX=1.0E+10
```

```
      XMN=0.
      PBB=0.0
   10 PBAR=P(J)
      D=(DELT/(DXS(JC)**2*DYS(JC)))*(ARU*V(J)-ARC*V(J-1))
      D=AMAX1(D,-0.99)
      ROT=RON(J)/(1.+D)
      RVT=RVN(J)/(1.+D)
      ET=EN(J)-P(J)*D/RO(J)
      PT=PCAL(ET,ROT,RVT)
      S=PBAR-PT
      IF(ICT.NE.0.AND.S.NE.SM) BETA(J)=(PBAR-PBB)/(S-SM)
      P(J)=PBAR-BETA(J)*S
      IF(S.GE.0.0)GO TO 20
      XMN=PBAR
      IF(P(J).GE.XMX)P(J)=0.5*(XMN+XMX)
      GO TO 30
   20 XMX=PBAR
      IF(P(J).LE.XMN)P(J)=0.5*(XMN+XMX)
   30 CONTINUE
      DELP=P(J)-PBAR
      IF(ABS(DELP).LE.EPSI*P(J))ICT=100
      DLYU2=DYS(JC)+DYS(JCP)
      DLYD2=DYS(JCM)+DYS(JC)
      ROU=(DYS(JCP)*RO(J)+DYS(JC)*RO(J+1))/DLYU2
      ROD=(DYS(JC)*RO(J-1)+DYS(JCM)*RO(J))/DLYD2
      DVU=2.*DELT*DELP/(ROU*DLYU2)
      DVD=-2.*DELT*DELP/(ROD*DLYD2)
      V(J)=V(J)+DVU
      V(J-1)=V(J-1)+DVD
      SM=S
      PBB=PBAR
      ICT=ICT+1
      IF(ICT.GT.10)GO TO 100
      FLG=1.0
      GO TO 10
  100 CONTINUE
    1 CONTINUE
      RETURN
      END
```

= = = = = = = = = = = = = = = = ///// = = = = = = = = = = = = = = = =

```
      SUBROUTINE PSET
*CALL,LOOPC
C---- SET PRESSURE
      DO 1 I=1,IL
      CALL JSET(I)
      DO 1 J=JPB,JPT
      P(J)=PCAL(E(J),RO(J),RV(J))
    1 CONTINUE
      GO TO (2,3),KTOU
    3 DO 4 K=1,KL
      CALL KSET(K)
      PK(KOO)=PCAL(EK(KOO),ROK(KOO),RVK(KOO))
    4 CONTINUE
```

```
    2 CONTINUE
      RETURN
      END


          = = = = = = = = = = = = = = = ///// = = = = = = = = = = = = = = =


      SUBROUTINE PUMP(J,PHEAD)
*CALL,LOOPC
C---- PUMP PACKAGE
      PHEAD=0.
      RETURN
      ENTRY PUMPA
      RETURN
      END


          = = = = = = = = = = = = = = = ///// = = = = = = = = = = = = = = =


      FUNCTION RBBNCAL(TH1)
*CALL,LOOPC
      RBBNCAL=(TH1/PNV)**0.3333
      RETURN
      END


          = = = = = = = = = = = = = = = ///// = = = = = = = = = = = = = = =


      FUNCTION RBWNCAL(J1,RO1,KTRAN)
*CALL,LOOPC
      J=J1
      GO TO (1,2) KTRAN
    1 VAVE=0.5*ABS(V(J)+V(J-1))
      VDAVE=0.5*ABS(VD(J)+VD(J-1))
      GO TO 3
    2 VAVE=0.5*SQRT((VK(KUO)+VK(KDO))**2+(VK(KOU)+VK(KOD))**2)
      VDAVE=0.5*SQRT((VDK(KOU)+VDK(KDO))**2+(VDK(KOU)+VDK(KOD))**2)
    3 VTB=0.1*VAVE+VDAVE
      RBWNCAL=0.0008/(RO1*VTB**2+1.0E-10)
      RETURN
      END


          = = = = = = = = = = = = = = = ///// = = = = = = = = = = = = = = =


      SUBROUTINE RESET
*CALL,LOOPC
      DATA BETAC / .98 /
C---- RESET FOR NEXT TIME STEP
      DO 1 I=1,IL
```

```
      IF(TIM(I).GT.T)GO TO 1
      DELT=DTIM(I)
      CALL JSET(I)
      DO 2 J=JB0,JT1
      VN(J)=V(J)
      RON(J)=RO(J)
      RVN(J)=RV(J)
      EN(J)=E(J)
    2 CONTINUE
    1 CONTINUE
      GO TO (8,9),KTOU
    9 DO 7 K=1,KL
      IF(TIMK(K).GT.T)GO TO 7
      DELT=DTIMK(K)
      CALL KSET(K)
      VNK(KUO)=VK(KUO)
      VNK(KDO)=VK(KDO)
      VNK(KOU)=VK(KOU)
      VNK(KOD)=VK(KOD)
      RONK(KOO)=ROK(KOO)
      RONK(KPO)=ROK(KPO)
      RONK(KMO)=ROK(KMO)
      RONK(KOP)=ROK(KOP)
      RONK(KOM)=ROK(KOM)
      RVNK(KOO)=RVK(KOO)
      RVNK(KPO)=RVK(KPO)
      RVNK(KMO)=RVK(KMO)
      RVNK(KOP)=RVK(KOP)
      RVNK(KOM)=RVK(KOM)
      ENK(KOO)=EK(KOO)
      ENK(KPO)=EK(KPO)
      ENK(KMO)=EK(KMO)
      ENK(KOP)=EK(KOP)
      ENK(KOM)=EK(KOM)
    7 CONTINUE
    8 CONTINUE
C---- ADJUST TIME STEP
      CALL TIMCT
C---- RELAXATION FACTOR
      DO 3 I=1,IL
      DELT=DTIM(I)
      CALL JSET(I)
      DO 3 J=JB2,JT2
      JC=JSF(J)
      JCP=JSF(J+1)
      JCM=JSF(J-1)
      DLYU2=DYS(JCP)+DYS(JC)
      DLYD2=DYS(JCM)+DYS(JC)
      ROU=(DYS(JCP)*RO(J)+DYS(JC)*RO(J+1))/DLYU2
      ROD=(DYS(JC)*RO(J-1)+DYS(JCM)*RO(J))/DLYD2
      IF(J.EQ.JB2) ROD=RO(J)
      IF(J.EQ.JT2) ROU=RO(J)
      PTO=PCAL(E(J),RO(J),RV(J))
      DELP=-EPSI*PTO
      VT=2.*DELT*DELP/(ROU*DLYU2)
      VB=-2.*DELT*DELP/(ROD*DLYD2)
      ARU=ARAV(JC,JCP)
      ARD=ARAV(JCM,JC)
```

```
      DT=(DELT/(DXS(JC)**2*DYS(JC)))*(ARU*VT-ARD*VB)
      ROT=RO(   J)/(1.0+DT)
      RVT=RV(   J)/(1.0+DT)
      ET=E(   J)-PTO/RO(   J)*DT
      PT=PCAL(ET,ROT,RVT)
      BETA(   J)=OMG*DELP/(DELP-(PT-PTO))
      BETA(J)=AMIN1(BETA(J),BETAC)
      IF(BETA(J).LE.0.) BETA(J)=BETAC
    3 CONTINUE
      GO TO (5,6),KTOU
    6 DO 4 K=1,KL
      DELT=DTIMK(K)
      CALL KSET(K)
      ET=EK(KOO)
      ROT=ROK(KOO)
      RVT=RVK(KOO)
      PT=PCAL(ET,ROT,RVT)
      PTO=PT
      DELP=-EPSI*PTO
      UR=2.*DELT*DELP/DXK(KOO)/(ROK(KOO)+ROK(KPO))
      UL=-2.*DELT*DELP/DXK(KOO)/(ROK(KMO)+ROK(KOO))
      VT=2.*DELT*DELP/DYK(KOO)/(ROK(KOO)+ROK(KOP))
      VB=-2.*DELT*DELP/DYK(KOO)/(ROK(KOM)+ROK(KOO))
      UR=UR*GAM(KUO)
      UL=UL*GAM(KDO)
      VT=VT*GAM(KOU)
      VB=VB*GAM(KOD)
      DT=DELT*((UR-UL)/DXK(KOO)+(VT-VB)/DYK(KOO))
      R[illegible]O)/(1.+DT)
      [illegible]J)/(1.+DT)
      [illegible]K(KOO)-PTO*DT/ROK(KOO)
      PT=PCAL(ET,ROT,RVT)
      BETAK(K)=OMG*DELP/(DELP-(PT-PTO))
      BETAK(K)=AMIN1(BETAK(K),BETAC)
      IF(BETAK(K).LE.0.) BETAK(K)=BETAC
    4 CONTINUE
    5 CONTINUE
      RETURN
      END


         = = = = = = = = = = = = = = = = ///// = = = = = = = = = = = = = = = =


      FUNCTION RVCAL(EI,PI,THI)
*CALL,LOOPC
      RVCAL=THI*PI/(GAMI*EI)
      RETURN
      END


         = = = = = = = = = = = = = = = = ///// = = = = = = = = = = = = = = = =


      SUBROUTINE SETUP
*CALL,LOOPC
```

```
C---- INITIALIZE PARAMETERS
      N2=NP
      FLG=1.
      THC1=1.-THC
      PNV=BNUM*4.1888
      NEX=0
      IPLTG=0
      T=TSTART
      TIMA=TSTART-1.0E-6
      DTMN=DELT
      SDTC=3.0
      ITAG=0
      ITER=0
      CYCLE=0
      DELTO=DELT
      PRST=TSTART+DPRST
      TWPRT=PRST
      IBREAK=1
C----- INITIALIZE STATE VARIABLES
      DO 18 I=1,IL
      CALL INFNC(TEMI(I),PI(I),THI(I),ROINC(I),RVINC(I),EINC(I))
   18 CONTINUE
      DO 11 M=1,ML
      CALL INFNC(TEMB(M),PB(M),THB(M),ROB(M),RVB(M),EB(M))
   11 CONTINUE
      IF(KL.EQ.0) GO TO 61
      DO 60 K=1,KL
      CALL INFNC(TEMK(K),PK(K),THK(K),ROKC(K),RVKC(K),EKC(K))
   60 CONTINUE
   61 CONTINUE
      WRITE(9,950)
      WRITE(9,960)
      WRITE(9,961)
      DO 15 M=1,ML
      WRITE(9,953)M,TEMB(M),PB(M),THB(M),ROB(M),RVB(M),EB(M)
   15 CONTINUE
      WRITE(9,903)
      WRITE(9,962)
      DO 14 I=1,IL
      WRITE(9,955)I,TEMI(I),PI(I),THI(I),ROINC(I),RVINC(I),
     1EINC(I),JSXI(I),JCEL(I)
   14 CONTINUE
      IF(KL.EQ.0) GO TO 13
      WRITE(9,903)
      WRITE(9,963)
      DO 10 K=1,KL
      WRITE(9,953)K,TEMK(K),PK(K),THK(K),ROKC(K),RVKC(K),EKC(K)
   10 CONTINUE
   13 CONTINUE
C---- LOGIC SETUP
      JREF(1)=0
      IF(IL.EQ.1) GO TO 8
      DO 7 I=2,IL
      JREF(I)=JREF(I-1)+JCEL(I-1)
    7 CONTINUE
    8 CONTINUE
      DO 4 I=1,IL
```

```
      IBOT(I)=1
      KI=KBOT(I)
      IF(KI.GT.0) IBOT(I)=2
      ITOP(I)=1
      KI=KTOP(I)
      IF(KI.GT.0) ITOP(I)=2
    4 CONTINUE
      KTOU=1
      IF(KL.GT.0) KTOU=2
      J=0
      DO 40 I=1,IL
      JSL=JSXI(I)
      DO 35 JS=1,JSL
      NI=NJS(I,JS)
      DO 30 N=1,NI
      J=J+1
      JSF(J)=I+(JS-1)*N2
   30 CONTINUE
   35 CONTINUE
   40 CONTINUE
      CALL LOGCPR
C---- INITIALIZE MESH VARIABLES
      DO 1 I=1,IL
      CALL JSET(I)
      TIM(I)=TSTART-1.0E-6
      DTIM(I)=0.0
      DO 1 J=JBO,JTI
      JC=JSF(J)
      V(J)=VIS(JC)
      VD(J)=0.
      RO(J)=ROINC(I)
      RV(J)=RVINC(I)
      E(J)=EINC(I)
      WT(J)=0.0
      P(J)=PCAL(E(J),RO(J),RV(J))
    1 CONTINUE
      DO 62 I=1,IL
      CALL JSET(I)
      DO 62 J=JBO,JTI
      VN(J)=V(J)
      RON(J)=RO(J)
      RVN(J)=RV(J)
      EN(J)=E(J)
   62 CONTINUE
      GO TO (2,3),KTOU
    3 DO 5 K=1,KL
      CALL KSET(K)
      TIMK(K)=TSTART-1.0E-6
      DTIMK(K)=0.0
      VK(KUO)=0.
      VK(KDO)=0.
      VK(KOU)=0.
      VK(KOD)=0.
      VDK(KUO)=0.
      VDK(KDO)=0.
      VDK(KOU)=0.
      VDK(KOD)=0.
      ROK(KOO)=ROKC(K)
```

```
      RVK(KOO)=RVKC(K)
      EK(KOO)=EKC(K)
      PK(KOO)=PCAL(EK(KOO),ROK(KOO),RVK(KOO))
      VNK(KUO)=0.
      VNK(KDO)=0.
      VNK(KOU)=0.
      VNK(KOD)=0.
      RONK(KOO)=ROK(KOO)
      RVNK(KOO)=RVK(KOO)
      ENK(KOO)=EK(KOO)
    5 CONTINUE
    2 CONTINUE
      RETURN
  955 FORMAT(I5,6E15.7,I8,I5)
  960 FORMAT(10X,33HPROCESSED INITIAL STATE VARIABLES///)
  961 FORMAT(42H     M        TEMB             PB             THB     .
     142H             ROB              RVB            EB //)
  962 FORMAT(42H     I        TEMI             PI             THI     .
     146H             ROINC             RVINC            EINC    .
     211X,4HJSXI,5H JCEL   //)
  963 FORMAT(42H     K        TEMK             PK             THK     .
     145H              ROKC              RVKC            EKC//)
      END


          = = = = = = = = = = = = = = = ///// = = = = = = = = = = = = = = =


      FUNCTION SPCAL(T1)
*CALL,LOOPC
      SPCAL=((T1-255.2)/117.8)**4.48
      RETURN
      END


          = = = = = = = = = = = = = = = ///// = = = = = = = = = = = = = = =


      FUNCTION STCAL(P1)
*CALL,LOOPC
      STCAL=255.2+117 3*P1**.223
      RETURN
      END


          = = = = = = = = = = = = = = = ///// = = = = = = = = = = = = = = =


      FUNCTION TEMC(RO1,RV1,E1,P1)
*CALL,LOOPC
C---- TEMPERATURE CALCULATION
      ITMT=1
      TMT1=0.0
      TCA=-(RO1-RV1)*CHL1-RV1*CHV1
```

```
      TCB=(RO1-RV1)*CHL+RV1*CHV                                         LOOP    1549
      TCC=RO1*(E1-ECL)+RV1*(ECL-ECV)                                    LOOP    1550
      IF(ETEM.GT.0.5)GO TO 5                                            LOOP    1551
      TSATT=STCAL(P1)                                                   LOOP    1552
      EVAP=EVCAL(TSATT)                                                 LOOP    1553
      TCA=TCA+RV1*CHV1                                                  LOOP    1554
      TCB=TCB-RV1*CHV                                                   LOOP    1555
      TCC=TCC+RV1*(ECV-EVAP)                                            LOOP    1556
    5 CONTINUE                                                          LOOP    1557
      TMT=(TCC+TCA*TMT1**2)/(TCB+2.0*TCA*TMT1)                          LOOP    1558
      IF(ABS(TMT-TMT1).LT.0.005*TMT)GO TO 10                            LOOP    1559
      ITMT=ITMT+1                                                       LOOP    1560
      IF(ITMT.GT.10)GO TO 10                                            LOOP    1561
      TMT1=TMT                                                          LOOP    1562
      GO TO 5                                                           LOOP    1563
   10 CONTINUE                                                          LOOP    1564
      TEMC=TMT+TC                                                       LOOP    1565
      RETURN                                                            LOOP    1566
      END                                                               LOOP    1567


          = = = = = = = = = = = = = = = = ///// = = = = = = = = = = = = = = = =


      FUNCTION THCAL(E1,P1,RV1)                                         LOOP    1568
*CALL,LOOPC                                                             LOOP    1569
C     THETA ASSUMED LESS THAN THC                                       LOOP    1570
      THCAL=(GAM1*RV1*E1/THC-P1)/(ASQ*ROL)+THC                          LOOP    1571
      RETURN                                                            LOOP    1572
      END                                                               LOOP    1573


          = = = = = = = = = = = = = = = = ///// = = = = = = = = = = = = = = = =


      SUBROUTINE TILDE                                                  LOOP    1574
*CALL,LOOPC                                                             LOOP    1575
C---- TILDE CALCULATION                                                 LOOP    1576
      DO 1 I=1,IL                                                       LOOP    1577
      IF(TIM(I).GT.T)GO TO 1                                            LOOP    1578
      DELT=DTIM(I)                                                      LOOP    1579
      CALL JSET(I)                                                      LOOP    1580
      DO 11 J=JML,JMH                                                   LOOP    1581
      JC=JSF(J)                                                         LOOP    1582
      JCP=JSF(J+1)                                                      LOOP    1583
      T1=(VN(J)-VN(J-1))/DYS(JC)                                        LOOP    1584
      T2=(VN(J+1)-VN(J))/DYS(JCP)                                       LOOP    1585
      FVY=.5*(VN(J)*(T1+T2)+ALPHA*ABS(VN(J))*(T1-T2))                   LOOP    1586
      ARU=DXS(JCP)**2                                                   LOOP    1587
      ARD=DXS(JC)**2                                                    LOOP    1588
      DLYB2=DYS(JCP)+DYS(JC)                                            LOOP    1589
      ROBAR=(DYS(JCP)*RO(J)+DYS(JC)*RO(J+1))/DLYB2                      LOOP    1590
      DEN=ROBAR*(ARU+ARD)*DLYB2                                         LOOP    1591
      VDU2=VD(J)+VD(J+1)                                                LOOP    1592
      VDD2=VD(J-1)+VD(J)                                                LOOP    1593
      ROSU=RV(J+1)*(RO(J+1)-RV(J+1))/RO(J+1)                            LOOP    1594
```

```
      ROSD=RV(J)*(RO(J)-RV(J))/RO(J)
      T3=ARU*ROSU*(VDU2**2+ALPHA*ABS(VDU2)*(VD(J)-VD(J+1)))
      T4=ARD*ROSD*(VDD2**2-ALPHA*ABS(VDD2)*(VD(J-1)-VD(J)))
      FVDY=(T3-T4)/DEN
      VISY=0.
      CALL PUMP(J,PHEAD)
      T6=P(J)-P(J+1)
      T3=2.*T6/(ROBAR*DLYB2)
      GYB=(DYS(JCP)*GYS(JC)+DYS(JC)*GYS(JCP))/DLYB2
      T5=T3+GYB-FVY+VISY-FVDY+PHEAD*2./DLYB2
      V(J)=VN(J)+DELT*T5
      CALL FRIC(I,J,V(J))
   11 CONTINUE
    1 CONTINUE
      RETURN
      END
```

= = = = = = = = = = = = = = = = = ///// = = = = = = = = = = = = = = = = =

```
      SUBROUTINE TIMCT
*CALL,LOOPC
      SMLP=0.99
      IF(ITER.GT.5)SMLP=1.01
      SDTC=AMAX1(STABC,SDTC*SMLP)
      DO 1 I=1,IL
      IF(TIM(I).GT.T)GO TO 1
      TIM(I)=TIM(I)+DTIM(I)
      TIMN=TIMA
      IF(TIM(I).GT.TIMA-DTMN/2.0)TIMN=TIMA+DELTO
      DVMX=1.0E-10
      CALL JSET(I)
      DO 2 J=JML,JMH
      JC=JSF(J)
      VDM=ABS(VD(J))
      VMM=ABS(V(J))
      DVMX=AMAX1(DVMX,VDM/DYS(JC),VMM/DYS(JC))
    2 CONTINUE
      DVMX=SDTC*DVMX
      ANDT=AINT(1.0+DVMX*(TIMN-TIM(I)))
      DTIM(I)=(TIMN-TIM(I))/ANDT
    1 CONTINUE
      GO TO (10,20),KTOU
   20 DO 4 K=1,KL
      IF(TIMK(K).GT.T)GO TO 4
      TIMK(K)=TIMK(K)+DTIMK(K)
      TIMN=TIMA
      IF(TIMK(K).GT.TIMA-DTMN/2.0)TIMN=TIMA+DELTO
      DVMX=1.0E-10
      CALL KSET(K)
      V1=ABS(VDK(KUO))
      V2=ABS(VDK(KOU))
      V3=ABS(VDK(KDO))
      V4=ABS(VDK(KOD))
      V5=ABS(VK(KUO))
      V6=ABS(VK(KOU))
```

```
      V7=ABS(VK(KDO))
      V8=ABS(VK(KOD))
      DVMX=AMAX1(V2,V4,V6,V8)/DYK(KOO)
      DUMX=AMAX1(V1,V3,V5,V7)/DXK(KOO)
      DVMX=AMAX1(DVMX,DUMX)
      DVMX=2.0*SDTC*DVMX
      ANDT=AINT(1.0+DVMX*(TIMN-TIMK(K)))
      DTIMK(K)=(TIMN-TIMK(K))/ANDT
    4 CONTINUE
   10 CONTINUE
C     ADVANCE TIME AND CYCLE
      IF((TIM(1)-TSTART).LT.0.01*DELTO) GO TO 25
      T=T+DTMN
      CYCLE=CYCLE+1
   25 CONTINUE
C     COMPUTE NEW DT MIN AND MAX
      DTMN=DELTO
      DO 30 I=1,IL
      DTMN=AMIN1(DTMN,DTIM(I))
   30 CONTINUE
      GO TO(40,50),KTOU
   50 DO 55 K=1,KL
      DTMN=AMIN1(DTMN,DTIMK(K))
   55 CONTINUE
   40 CONTINUE
      IF(NSUBDT.NE.1) RETURN
      DO 60 I=1,IL
   60 DTIM(I)=DTMN
      GO TO (90,70),KTOU
   70 DO 80 K=1,KL
   80 DTIMK(K)=DTMN
   90 CONTINUE
      RETURN
      END


         = = = = = = = = = = = = = = = = ///// = = = = = = = = = = = = = = = =


      SUBROUTINE UPDATJ
*CALL,LOOPC
C---- FINAL UPDATE FOR JUNCTION CELLS
      GO TO(100,200),KTOU
  200 CONTINUE
      DO 1 K=1,KL
      IF(TIMK(K).GT.T)GO TO 1
      DELT=DTIMK(K)
      CALL KSET(K)
      DLXR2=DXK(KOO)+DXK(KPO)
      DLXL2=DXK(KMO)+DXK(KOO)
      DLYT2=DYK(KOO)+DYK(KOP)
      DLYB2=DYK(KOM)+DYK(KOO)
      ROKR=(DXK(KPO)*ROK(KOO)+DXK(KOO)*ROK(KPO))/DLXR2
      ROKL=(DXK(KMO)*ROK(KOO)+DXK(KOO)*ROK(KMO))/DLXL2
      ROKT=(DYK(KOP)*ROK(KOO)+DYK(KOO)*ROK(KOP))/DLYT2
      ROKB=(DYK(KOM)*ROK(KOO)+DYK(KOO)*ROK(KOM))/DLYB2
      RONKR=(DXK(KPO)*RONK(KOO)+DXK(KOO)*RONK(KPO))/DLXR2
```

```
      RONKL=(DXK(KMO)*RONK(KOO)+DXK(KOO)*RONK(KMO))/DLXL2               LOOP    1699
      RONKT=(DYK(KOP)*RONK(KOO)+DYK(KOO)*RONK(KOP))/DLYT2               LOOP    1700
      RONKB=(DYK(KOM)*RONK(KOO)+DYK(KOO)*RONK(KOM))/DLYB2               LOOP    1701
      RVNKR=(DXK(KPO)*RVNK(KOO)+DXK(KOO)*RVNK(KPO))/DLXR2               LOOP    1702
      RVNKL=(DXK(KMO)*RVNK(KOO)+DXK(KOO)*RVNK(KMO))/DLXL2               LOOP    1703
      RVNKT=(DYK(KOP)*RVNK(KOO)+DYK(KOO)*RVNK(KOP))/DLYT2               LOOP    1704
      RVNKB=(DYK(KOM)*RVNK(KOO)+DYK(KOO)*RVNK(KOM))/DLYB2               LOOP    1705
      ULR=VK(KUO)-VDK(KUO)*RVNKR/RONKR                                  LOOP    1706
      ULL=VK(KDO)-VDK(KDO)*RVNKL/RONKL                                  LOOP    1707
      ULT=VK(KOU)-VDK(KOU)*RVNKT/RONKT                                  LOOP    1708
      ULB=VK(KOD)-VDK(KOD)*RVNKB/RONKB                                  LOOP    1709
      UVR=VK(KUO)+VDK(KUO)*(RONKR-RVNKR)/RONKR                          LOOP    1710
      UVL=VK(KDO)+VDK(KDO)*(RONKL-RVNKL)/RONKL                          LOOP    1711
      UVT=VK(KOU)+VDK(KOU)*(RONKT-RVNKT)/RONKT                          LOOP    1712
      UVB=VK(KOD)+VDK(KOD)*(RONKB-RVNKB)/RONKB                          LOOP    1713
      FLR=ULR*(RONK(KOO)-RVNK(KOO)+RONK(KPO)-RVNK(KPO))+ALPHA*ABS(ULR)*  LOOP    1714
     1(RONK(KOO)-RVNK(KOO)-RONK(KPO)+RVNK(KPO))                         LOOP    1715
      FLL=ULL*(RONK(KMO)-RVNK(KMO)+RONK(KOO)-RVNK(KOO))+ALPHA*ABS(ULL)*  LOOP    1716
     1(RONK(KMO)-RVNK(KMO)-RONK(KOO)+RVNK(KOO))                         LOOP    1717
      FLT=ULT*(RONK(KOO)-RVNK(KOO)+RONK(KOP)-RVNK(KOP))+ALPHA*ABS(ULT)*  LOOP    1718
     1(RONK(KOO)-RVNK(KOO)-RONK(KOP)+RVNK(KOP))                         LOOP    1719
      FLB=ULB*(RONK(KOM)-RVNK(KOM)+RONK(KOO)-RVNK(KOO))+ALPHA*ABS(ULB)*  LOOP    1720
     1(RONK(KOM)-RVNK(KOM)-RONK(KOO)+RVNK(KOO))                         LOOP    1721
      FVR=UVR*(RVNK(KOO)+RVNK(KPO))+ALPHA*ABS(UVR)*(RVNK(KOO)-RVNK(KPO))  LOOP    1722
      FVL=UVL*(RVNK(KMO)+RVNK(KOO))+ALPHA*ABS(UVL)*(RVNK(KMO)-RVNK(KOO))  LOOP    1723
      FVT=UVT*(RVNK(KOO)+RVNK(KOP))+ALPHA*ABS(UVT)*(RVNK(KOO)-RVNK(KOP))  LOOP    1724
      FVB=UVB*(RVNK(KOM)+RVNK(KOO))+ALPHA*ABS(UVB)*(RVNK(KOM)-RVNK(KOO))  LOOP    1725
C---- UPDATE RO                                                          LOOP    1726
      ROK(KOO)=RONK(KOO)-.5*DELT*((FLR+FVR-FLL-FVL)/DXK(KOO)             LOOP    1727
     1+(FLT+FVT-FLB-FVB)/DYK(KOO))                                       LOOP    1728
C---- ENERGY EQUATION                                                    LOOP    1729
      DLXR2=DXK(KOO)+DXK(KPO)                                            LOOP    1730
      DLXL2=DXK(KMO)+DXK(KOO)                                            LOOP    1731
      DLYT2=DYK(KOO)+DYK(KOP)                                            LOOP    1732
      DLYB2=DYK(KOM)+DYK(KOO)                                            LOOP    1733
      ROKR=(DXK(KPO)*ROK(KOO)+DXK(KOO)*ROK(KPO))/DLXR2                   LOOP    1734
      ROKL=(DXK(KMO)*ROK(KOO)+DXK(KOO)*ROK(KMO))/DLXL2                   LOOP    1735
      ROKT=(DYK(KOP)*ROK(KOO)+DYK(KOO)*ROK(KOP))/DLYT2                   LOOP    1736
      ROKB=(DYK(KOM)*ROK(KOO)+DYK(KOO)*ROK(KOM))/DLYB2                   LOOP    1737
      RVKR=(DXK(KPO)*RVK(KOO)+DXK(KOO)*RVK(KPO))/DLXR2                   LOOP    1738
      RVKL=(DXK(KMO)*RVK(KOO)+DXK(KOO)*RVK(KMO))/DLXL2                   LOOP    1739
      RVKT=(DYK(KOP)*RVK(KOO)+DYK(KOO)*RVK(KOP))/DLYT2                   LOOP    1740
      RVKB=(DYK(KOM)*RVK(KOO)+DYK(KOO)*RVK(KOM))/DLYB2                   LOOP    1741
      RLKR=ROKR-RVKR                                                     LOOP    1742
      RLKL=ROKL-RVKL                                                     LOOP    1743
      RLKT=ROKT-RVKT                                                     LOOP    1744
      RLKB=ROKB-RVKB                                                     LOOP    1745
      ROEC=RONK(KOO)*ENK(KOO)                                            LOOP    1746
      ROER=RONK(KPO)*ENK(KPO)                                            LOOP    1747
      ROEL=RONK(KMO)*ENK(KMO)                                            LOOP    1748
      ROET=RONK(KOP)*ENK(KOP)                                            LOOP    1749
      ROEB=RONK(KOM)*ENK(KOM)                                            LOOP    1750
      IF(ETEM.GT..5) GO TO 425                                           LOOP    1751
      TVC=STCAL(PK(KOO))                                                 LOOP    1752
      TVR=STCAL(PK(KPO))                                                 LOOP    1753
      TVL=STCAL(PK(KMO))                                                 LOOP    1754
      TVT=STCAL(PK(KOP))                                                 LOOP    1755
      TVB=STCAL(PK(KOM))                                                 LOOP    1756
```

```
      GO TO 430
  425 TVC=TEMC(RONK(KOO),RVNK(KOO),ENK(KOO),PK(KOO))
      TVR=TEMC(RONK(KPO),RVNK(KPO),ENK(KPO),PK(KPO))
      TVL=TEMC(RONK(KMO),RVNK(KMO),ENK(KMO),PK(KMO))
      TVT=TEMC(RONK(KOP),RVNK(KOP),ENK(KOP),PK(KOP))
      TVB=TEMC(RONK(KOM),RVNK(KOM),ENK(KOM),PK(KOM))
  430 REVC=RVNK(KOO)*EVCAL(TVC)
      REVR=RVNK(KPO)*EVCAL(TVR)
      REVL=RVNK(KMO)*EVCAL(TVL)
      REVT=RVNK(KOP)*EVCAL(TVT)
      REVB=RVNK(KOM)*EVCAL(TVB)
      RELC=ROEC-REVC
      RELR=ROER-REVR
      RELL=ROEL-REVL
      RELT=ROET-REVT
      RELB=ROEB-REVB
      FRER=0.5*(UVR*(REVC+REVR)+ALPHA*ABS(UVR)*(REVC-REVR)
     1+ULR*(RELC+RELR)+ALPHA*ABS(ULR)*(RELC-RELR))
      FREL=0.5*(UVL*(REVL+REVC)+ALPHA*ABS(UVL)*(REVL-REVC)+ULL*
     1(RELL+RELC)+ALPHA*ABS(ULL)*(RELL-RELC))
      FRET=0.5*(UVT*(REVC+REVT)+ALPHA*ABS(UVT)*(REVC-REVT)+ULT*(RELC+
     1RELT)+ALPHA*ABS(ULT)*(RELC-RELT))
      FREB=0.5*(UVB*(REVB+REVC)+ALPHA*ABS(UVB)*(REVB-REVC)+ULB*(RELB+
     1RELC)+ALPHA*ABS(ULB)*(RELB-RELC))
      FEC=(FRER-FREL)/DXK(KOO)+(FRET-FREB).DYK(KOO)
      FPW=-PK(KOO)*((VK(KUO)-VK(KDO))/DXK(KOO)+(VK(KOU)-VK(KOD))/DYK(KOO
     1))
      TH=(ROL+RVNK(KOO)-RONK(KOO))/ROL
      TH=AMAX1(TH,THC)
      TH=AMIN1(TH,THC1)
      TH1=TH
      IF(TH1.GT.0.5)TH1=1.0-TH1
      RB=RBBNCAL(TH1)
      CKN=.5*CDG*((VDK(KUO)+VDK(KDO))**2+(VDK(KOU)+VDK(KOD))**2)**.5
     1+12.0*VISV/RB
      FDIS=0.25*CKN*(0.375*TH1/RB)*((VDK(KUO)+VDK(KDO))**2+(VDK(KOU)+
     1VDK(KOD))**2)*ROK(KOO)
      FDP=-PK(KOO)*((((RVNKR-RONKR+ROL)/ROL-(RVNKR/RONKR))*VDK(KUO)
     1-((RVNKL-RONKL+ROL)/ROL- RVNKL/RONKL)*VDK(KDO))/DXK(KOO)
     2+(((RVNKT-RONKT+ROL)/ROL- RVNKT/RONKT)*VDK(KOU)
     3-((RVNKB-RONKB+ROL)/ROL- RVNKB/RONKB)*VDK(KOD))/DYK(KOO))
C---- HEAT CONDUCTION
      TEM=ENK(KOO)
      TEMR=ENK(KPO)
      TEML=ENK(KMO)
      TEMT=ENK(KOP)
      TEMBS=ENK(KOM)
      THR=(ROL+RVNK(KPO)-RONK(KPO))/ROL
      THR=AMAX1(THR,THC)
      THR=AMIN1(THR,THC1)
      THL=(ROL+RVNK(KMO)-RONK(KMO))/ROL
      THL=AMAX1(THL,THC)
      THL=AMIN1(THL,THC1)
      THT=(ROL+RVNK(KOP)-RONK(KOP))/RO
      THT=AMAX1(THT,THC)
      THT=AMIN1(THT,THC1)
      THBS=(ROL+RVNK(KOM)-RONK(KOM))/ROL
      THBS=AMAX1(THBS,THC)
```

```
      THBS=AMIN1(THBS,THC1)
      EKR=0.5*(EDV*(TH+THR)+EDL*(2.0-TH-THR))
      EKT=0.5*(EDV*(TH+THT)+EDL*(2.0-TH-THT))
      EKL=0.5*(EDV*(TH+THL)+EDL*(2.0-TH-THL))
      EKB=0.5*(EDV*(TH+THBS)+EDL*(2.0-TH-THBS))
      DIFE=(EKR*(TEMR-TEM)-EKL*(TEM-TEML))/DXK(KOO)**2
     1+(EKT*(TEMT-TEM)-EKB*(TEM-TEMBS))/DYK(KOO)**2
C---- UPDATE ENERGY
      EK(KOO)=(ROEC+DELT*(-FEC+FDIS+FDP+DIFE+FPW))/ROK(KOO)
C---- VAPOR DENSITY EQUATION
      RVK(KOO)=RVNK(KOO)-.5*DELT*((FVR-FVL)/DXK(KOO)+(FVT-FVB)/DYK(KOO))
      RVK(KOO)=AMIN1(RVK(KOO),ROK(KOO))
      RVK(KOO)=AMAX1(RVK(KOO),0.)
C---- PHASE CHANGE
      IF(PHCH.LT.0.5)GO TO 4500
      VPK(K)=0.
      ROT=ROK(KOO)
      RVO=RVK(KOO)
      RVT=RVO
      ET=EK(KOO)
      TH=(ROL-ROT+RVO)/ROL
      TH1=TH
      IF(TH.LT.THC.OR.TH.GT.THC1)GO TO 4500
      IF(TH.GT.0.5)TH1=1.0-TH
      RB=RBBNCAL(TH1)
      RBW=RBWNCAL(K,ROT,2)
      RB=AMIN1(RB,RBW)
      AREA=3.0*TH1/RB
      TLQ=TEMC(ROT,RVO,ET,P(KOO))
      PSAT=SPCAL(TLQ)
      EVSAT=EVCAL(TLQ)
      RSAT=RVCAL(EVSAT,PSAT,TH)
      IVSL=0
      RBETA=1.0E-5*ROL
      IRVT=0
C---- START ITERATION
 4412 CONTINUE
      RVT=RVK(KOO)
      PT=PCAL(ET,ROT,RVT)
      TVAP=STCAL(PT)
      TLQ=TEMC(ROT,RVT,ET,PT)
      VPK(K)=VPRO(K,TH,RSAT,RB,TLQ,TVAP,AREA,2)
      RATE=VPK(K)*DELT
      RVEQ=RVT-RVO-RATE
      IF(ABS(RVEQ/RSAT).LT.0.001)GO TO 4500
      IF(IRVT.LE.23)GO TO 4420
      WRITE(9,4413)
      WRITE(9,4414)K,IRVT,RSAT,RVO,RVT,RATE,TVAP,TLQ
 4413 FORMAT(6X,1HJ,8X,4HIRVT,10X,4HRSAT,16X,3HRVO,12X,3HRVT,12X,
     14HRATE,12X,4HTVAP,16X,3HTLQ)
 4414 FORMAT(5X,I2,6X,I5,6X,1PE12.5,6X,E12.5,6X,E12.5,6X,E12.5,6X,E12.5,
     16X,E12.5)
 4420 CONTINUE
      IRVT=IRVT+1
      IF(IRVT.GT.25)GO TO 4500
      IF(IRVT.EQ.1)SRH=SIGN(1.0,RVEQ)
      IF(IVSL.EQ.1)GO TO  4418
      IF(RVEQ.GE.0.0)GO TO 4415
```

```
C---- SEEK BOUNDS
      FMN=RVEQ
      RVMN=RVK(KOO)
      IF(SRH.GE.0.0)IVSL=1
      RVK(KOO)=AMIN1(RVT+RBETA,ROT)
      IF(IVSL.EQ.1) RVK(KOO)=.5*(RVMN+RVMX)
      RBETA=2.0*RBETA
      GO TO 4412
 4415 FMX=RVEQ
      RVMX=RVK(KOO)
      IF(SRH.LT.0.0)IVSL=1
      RVK(KOO)=AMAX1(RVT-RBETA,0.)
      IF(IVSL.EQ.1)RVK(KOO)=.5*(RVMN+RVMX)
      RBETA=2.0*RBETA
      GO TO 4412
 4418 CONTINUE
C---- CONVERGE BOUNDS
      IF(RVEQ.LT.0.0)GO TO 4422
      RVTP=RVT-RVEQ*(RVMX-RVT)/(FMX-RVEQ)
      FMX=RVEQ
      RVMX=RVT
      RVK(KOO)=RVTP
      IF(RVTP.LT.RVMN)RVK(KOO)=.5*(RVMN+RVMX)
      GO TO 4412
 4422 RVTP=RVT-RVEQ*(RVT-RVMN)/(RVEQ-FMN)
      FMN=RVEQ
      RVMN=RVT
      RVK(KOO)=RVTP
      IF(RVTP.GT.RVMX)RVK(KOO)=.5*(RVMN+RVMX)
      GO TO 4412
 4500 CONTINUE
    1 CONTINUE
  100 CONTINUE
      RETURN
      END
```

= = = = = = = = = = = = = = = = ///// = = = = = = = = = = = = = = = =

```
      SUBROUTINE UPDATP
*CALL,LOOPC
C---- FINAL UPDATE FOR PIPES
      DO 3 I=1,IL
      IF(TIM(I).GT.T)GO TO 3
      DELT=DTIM(I)
      CALL JSET(I)
      DO 1 J=JPB,JPT
      JC=JSF(J)
      JCP=JSF(J+1)
      JCM=JSF(J-1)
      T1=.5*(V(J)*(RON(J)+RON(J+1))+ALPHA*ABS(V(J))*(RON(J)-RON(J+1)))
      T2=.5*(V(J-1)*(RON(J-1)+RON(J))+ALPHA*ABS(V(J-1))*(RON(J-1)-RON(J
     1))
      ARU=ARAV(JC,JCP)
      ARD=ARAV(JCM,JC)
      RO(J)=RON(J)-(DELT/(DXS(JC)**2*DYS(JC)))*(ARU*T1-ARD*T2)
```

```
    1 CONTINUE
C---- ENERGY EQUATION
      CALL JSF[illegible](I)
      DO 2 J=JPB,JPT
      JC=JSF(J)
      JCP=JSF(J+1)
      JCM=JSF(J-1)
      ARU=ARAV(JC,JCP)
      ARD=ARAV(JCM,JC)
      DLYU2=DYS(JCP)+DYS(JC)
      DLYD2=DYS(JCM)+DYS(JC)
      ROU=(DYS(JCP)*RON(J)+DYS(JC)*RON(J+1))/DLYU2
      ROD=(DYS(JC)*RON(J-1)+DYS(JCM)*RON(J))/DLYD2
      RVU=(DYS(JCP)*RVN(J)+DYS(JC)*RVN(J+1))/DLYU2
      RVD=(DYS(JC)*RVN(J-1)+DYS(JCM)*RVN(J))/DLYD2
      RLU=ROU-RVU
      RLD=ROD-RVD
      VVU=V(J)+RLU*VD(J)/ROU
      VVD=V(J-1)+RLD*VD(J-1)/ROD
      VLU=V(J)-RVU*VD(J)/ROU
      VLD=V(J-1)-RVD*VD(J-1)/ROD
      ROE=RON(J)*EN(J)
      ROEM=RON(J-1)*EN(J-1)
      ROEP=RON(J+1)*EN(J+1)
      IF(ETEM.GT..5) GO TO 425
 1000 CONTINUE
      TEMA=STCAL(P(J))
      IF(J.EQ.JB2) GO TO 1001
      TEMAM=STCAL(P(J-1))
      GO TO 1002
 1001 TEMAM=TEMA
 1002 IF(J.EQ.JT2) GO TO 1003
      TEMAP=STCAL(P(J+1))
      GO TO 1004
 1003 TEMAP=TEMA
 1004 CONTINUE
      GO TO 430
  425 CONTINUE
      TEMA=TEMC(RON(J),RVN(J),EN(J),P(J))
      TEMAM=TEMC(RON(J-1),RVN(J-1),EN(J-1),P(J-1))
      TEMAP=TEMC(RON(J+1),RVN(J+1),EN(J+1),P(J+1))
  430 EEV=EVCAL(TEMA)
      EEVM=EVCAL(TEMAM)
      EEVP=EVCAL(TEMAP)
      REV=RVN(J)*EEV
      REVM=RVN(J-1)*EEVM
      REVP=RVN(J+1)*EEVP
      REL=ROE-REV
      RELM=ROEM-REVM
      RELP=ROEP-REVP
      RIUVU=.5*(VVU*(REV+REVP)+ALPHA*ABS(VVU)*(REV-REVP))
      RIUVD=.5*(VVD*(REVM+REV)+ALPHA*ABS(VVD)*(REVM-REV))
      RIULU=.5*(VLU*(REL+RELP)+ALPHA*ABS(VLU)*(REL-RELP))
      RIULD=.5*(VLD*(RELM+REL)+ALPHA*ABS(VLD)*(RELM-REL))
      DEN=DXS(JC)**2*DYS(JC)
      FVV=(ARU*RIUVU-ARD*RIUVD)/DEN
      FVL=(ARU*RIULU-ARD*RIULD)/DEN
      T3=((RVU-ROU+ROL)/ROL-RVU/ROU)*VD(J)
```

```
      T4=((RVD-ROD+ROL)/ROL-RVD/ROD)*VD(J-1)
      T1=V(J)+T3
      T2=V(J-1)+T4
      FWK=(P(J)/(DXS(JC)**2*DYS(JC)))*(ARU*T1-ARD*T2)
      TH=(ROL+RVN(J)-RON(J))/ROL
      TH=AMAX1(TH,THC)
      TH=AMIN1(TH,THC1)
      TH1=TH
      IF(TH1.GT..5)TH1=1.-TH1
      RB=RBBNCAL(TH1)
      CKN=.5*CDG*ABS(VD(J)+VD(J-1))+12.*VISV/RB
      FDIS=.25*CKN*(.375*TH1/RB)*(VD(J)+VD(J-1))**2
      FQ=QJS(JC)
      T9=-FVV-FVL-FWK+FDIS+FQ
      T10=RON(J)*EN(J)+DELT*T9
      E(J)=T10/RO(J)
C---- DRIFT FLUX TERMS
      T1=.5*(VVU*(RVN(J)+RVN(J+1))
     1+ALPHA*ABS(VVU)*(RVN(J)-RVN(J+1)))
      T2=.5*(VVD*(RVN(J-1)+RVN(J))
     1+ALPHA*ABS(VVD)*(RVN(J-1)-RVN(J)))
      RV(J)=RVN(J)-(DELT/(DXS(JC)**2*DYS(JC)))*(ARU*T1-ARD*T2)
C---- PHASE CHANGE
      IF(PHCH.LT..5) GO TO 4500
      VP(J)=0.
      ROT=RO(J)
      RVO=RV(J)
      RVT=RVO
      ET=E(J)
      TH=(ROL-ROT+RVO)/ROL
      TH1=TH
      IF(TH.LT.THC.OR.TH.GT.THC1) GO TO 4500
      IF(TH.GT..5) TH1=1.-TH
      RB=RBBNCAL(TH1)
      RBW=RBWNCAL(J,ROT,1)
      RB=AMIN1(RB,RBW)
      AREA=3.*TH1/RB
      TLQ=TEMC(ROT,RVO,ET,P(J))
      PSAT=SPCAL(TLQ)
      EVSAT=EVCAL(TLQ)
      RSAT=RVCAL(EVSAT,PSAT,TH)
      IVSL=0
      RBETA=1.E-5*ROL
      IRVT=0
 4412 CONTINUE
      IF(RV(J).LT.0)RV(J)=0.0
      RVT=RV(J)
      PT=PCAL(ET,ROT,RVT)
      TVAP=STCAL(PT)
      TLQ=TEMC(ROT,RVT,ET,PT)
      VP(J)=VPRO(J,TH,RSAT,RB,TLQ,TVAP,AREA,1)
      RATE=VP(J)*DELT
      RVEQ=RVT-RVO-RATE
      IF(ABS(RVEQ/RSAT).LT..001) GO TO 4500
      IF(IRVT.LE.24) GO TO 4420
      WRITE(9,4413)
      WRITE(9,4414)J,IRVT,RSAT,PVO,RVT,RATE,TVAP,TLQ
 4420 CONTINUE
```

```
      IRVT=IRVT+1                                                       LOOP    2041
      IF(IRVT.GT.25) GO TO 4500                                         LOOP    2042
      IF(IRVT.EQ.1) SRH=SIGN(1.,RVEQ)                                   LOOP    2043
      IF(IVSL.EQ.1) GO TO 4418                                          LOOP    2044
      IF(RVEQ.GE.0.) GO TO 4415                                         LOOP    2045
      FMN=RVEQ                                                          LOOP    2046
      RVMN=RV(J)                                                        LOOP    2047
      IF(SRH.GE.0.) IVSL=1                                              LOOP    2048
      RV(J)=RV(J)+RBETA                                                 LOOP    2049
      IF(IVSL.EQ.1) RV(J)=.5*(RVMN+RVMX)                                LOOP    2050
      RBETA=2.*RBETA                                                    LOOP    2051
      GO TO 4412                                                        LOOP    2052
 4415 FMX=RVEQ                                                          LOOP    2053
      RVMX=RV(J)                                                        LOOP    2054
      IF(SRH.LT.0.) IVSL=1                                              LOOP    2055
      RV(J)=RV(J)-RBETA                                                 LOOP    2056
      IF(IVSL.EQ.1) RV(J)=.5*(RVMN+RVMX)                                LOOP    2057
      RBETA=2.*RBETA                                                    LOOP    2058
      GO TO 4412                                                        LOOP    2059
 4418 CONTINUE                                                          LOOP    2060
      IF(RVEQ.LT.0.) GO TO 4422                                         LOOP    2061
      RVTP=RVT-RVEQ*(RVMX-RVT)/(FMX-RVEQ)                               LOOP    2062
      FMX=RVEQ                                                          LOOP    2063
      RVMX=RVT                                                          LOOP    2064
      RV(J)=RVTP                                                        LOOP    2065
      IF(RVTP.LT.RVMN) RV(J)=.5*(RVMN+RVMX)                             LOOP    2066
      GO TO 4412                                                        LOOP    2067
 4422 RVTP=RVT-RVEQ*(RVT-RVMN)/(RVEQ-FMN)                               LOOP    2068
      FMN=RVEQ                                                          LOOP    2069
      RVMN=RVT                                                          LOOP    2070
      RV(J)=RVTP                                                        LOOP    2071
      IF(RVTP.GT.RVMX) RV(J)=.5*(RVMN+RVMX)                             LOOP    2072
      GO TO 4412                                                        LOOP    2073
 4500 CONTINUE                                                          LOOP    2074
    2 CONTINUE                                                          LOOP    2075
    3 CONTINUE                                                          LOOP    2076
      CALL PRINT                                                        LOOP    2077
      RETURN                                                            LOOP    2078
 4413 FORMAT(6X,1HJ,8X,4HIRVT,10X,4HRSAT,16X,3HRVO,12X,3HRVT,12X,       LOOP    2079
     14HRATE,12X,4HTVAP,16X,3HTLQ)                                      LOOP    2080
 4414 FORMAT(5X,I2,6X,I5,6X,1PE12.5,6X,E12.5,6X,E12.5,6X,E12.5,6X,E12.5,LOOP    2081
     16X,E12.5)                                                         LOOP    2082
      END                                                               LOOP    2083
```

= = = = = = = = = = = = = = = = ///// = = = = = = = = = = = = = = = =

```
      FUNCTION VDKCAL(RO1,UD1,TH1,A1,DNU,RB1)                           LOOP    2084
*CALL,LOOPC                                                             LOOP    2085
      TH=TH1+1.E-10                                                     LOOP    2086
      VDKCAL=0.125*RO1*A1/TH*(CDG*UD1+12.*DNU/RB1)                      LOOP    2087
      RETURN                                                            LOOP    2088
      END                                                               LOOP    2089
```

= = = = = = = = = = = = = = = = ///// = = = = = = = = = = = = = = = =

```
      FUNCTION VDKPCAL(RO1,VD1,TH1,A1)
*CALL,LOOPC


      TH=TH1+1.E-11
      VDKPCAL=0.125*RO1*A1/TH*CDG*SIGN(1.,VD1)
      RETURN
      END


          = = = = = = = = = = = = = = = = ///// = = = = = = = = = = = = = = = =


      FUNCTION VPRO(J1,TH1,RST1,RB1,T1,T2,A1,KTRAN)
*CALL,LOOPC
C       PHASE CHANGE RATE
      J=J1
      GO TO (1,2),KTRAN
    1 VAVE=0.5*ABS(V(J)+V(J-1))
      VDAVE=0.5*ABS(VD(J)+VD(J-1))
      GO TO 3
    2 VAVE=0.5*SQRT((VK(KUO)+VK(KDO))**2+(VK(KOU)+VK(KOD))**2)
      VDAVE=0.5*SQRT((VDK(KOU)+VDK(KDO))**2+(VDK(KOU)+VDK(KOD))**2)
    3 VTB1=0.1*VAVE+VDAVE
      E1=(1.91*ROL*TH1/RST1*CHL/ELHT*ABS(T1-T2)+
     1(0.637*VTB1*RB1*ROL/EDL)**0.5)/RB1
      VPRO=E1*A1/ELHT*EDL*CHL*(T1-T2)
      RETURN
      END


          = = = = = = = = = = = = = = = = ///// = = = = = = = = = = = = = = = =


      SUBROUTINE WALLT
*CALL,LOOPC
C----WALL TEMPERATURE EQUATION
      RETURN
      END


          = = = = = = = = = = = = = = = = ///// = = = = = = = = = = = = = = = =
```

## DISTRIBUTION

| | Copies |
|---|---|
| Nuclear Regulatory Commission, Bethesda, Maryland, R-4 | 283 |
| Technical Information Center, Oak Ridge, Tennessee | 2 |
| Los Alamos Scientific Laboratory, Los Alamos, New Mexico | 50 |

☆ U.S. Government Printing Office: 1979 – 677-013/128

Available from
US Nuclear Regulatory Commission
Washington, DC 20555

Available from
National Technical Information Service
Springfield, VA 22161

Microfiche $3.00

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

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