CONTENTS

FIGURES

TABLES

CONVERSION FACTORS

Metric (International System) units in this report may be converted to inch-pound units by the following conversion factors:

Multiply SI units	*By*	*To obtain inch-pound units*
centimeter (cm)	0.3937	inch
centimeter per cubic centimeter (cm/cm^3)	6.542	inch per cubic inch
centimeter per hour (cm/h)	0.3937	inch per hour
centimeter per second (cm/s)	0.03281	foot per second
cubic meter per hour (m^3/h)	35.32	cubic foot per hour
gram (gm)	0.002205	pound
kilopascal (kPa)	0.01450	pound per square inch
liter per hour (L/h)	0.2642	gallon per hour
meter (m)	3.281	foot
meter per hour (m/h)	3.281	foot per hour
meter per second (m/s)	3.201	foot per second
millimeter (mm)	0.03937	inch

SIMULATION OF SOLUTE TRANSPORT IN VARIABLY SATURATED POROUS MEDIA WITH SUPPLEMENTAL INFORMATION ON MODIFICATION TO THE U.S. GEOLOGICAL SURVEY'S COMPUTER PROGRAM VS2D

By R.W. Healy

ABSTRACT

This report documents computer program VS2DT for solving problems of solute transport in variably saturated porous media. The program uses a finite-difference approximation to the advection-dispersion equation. The program is an extension to the computer program VS2D developed by the U.S. Geological Survey, which simulates water movement through variably saturated porous media. Simulated regions can be one-dimensional columns, two-dimensional vertical cross sections, or axially symmetric, three-dimensional cylinders. Program options include: backward or centered approximations for both space and time derivatives, first-order decay, equilibrium adsorption as described by Freundlich or Langmuir isotherms, and ion exchange. Five test problems are used to demonstrate the ability of the computer program to accurately match analytical and previously published simulation results. Additional modifications to computer program VS2D are included as supplemental information.

The computer program is written in standard FORTRAN77. Extensive use of subroutines and function subprograms provides a modular code that can be easily modified for particular applications. A complete listing of data-input requirements and input and output for an example problem are included.

INTRODUCTION

Operations conducted at land surface or within the unsaturated zone may have considerable impact on the quality and quantity of water reaching local ground water reservoirs. Some of the more important of these operations include application of agricultural chemicals, solid-waste disposal, hazardous and radioactive-waste disposal, use of septic tanks, and accidental chemical spills. Understanding the fate of dissolved chemicals within the unsaturated zone can greatly aid in the prediction of the chemistry of the water that reaches aquifers. Such an understanding would also allow for evaluation of different preventative or remedial actions designed to protect our valuable ground-water resources. Computer models of water and solute movement within variably saturated porous media can be useful tools for gaining insight to processes that occur within the unsaturated zone. Computer models are a cost-effective means for predicting the effects of modifications to, or perturbations of, the unsaturated-zone system on the water contained in that system. Through a simple sensitivity analysis, the relative importance of different parameters that affect flow and transport can be investigated.

This report describes computer program VS2DT that simulates solute transport in porous media under variably saturated conditions. The program is an extension to the U.S. Geological Survey's computer program VS2D (Lappala and others, 1987), which simulates water movement through variably saturated porous media. The extension consists of four new subroutines and slight modifications to existing routines. VS2DT may be a useful tool in studies of water quality, ground-water contamination, waste disposal, or ground-water recharge. The program is user oriented and easy to use. However, its use must be accompanied by an awareness of the assumptions and limitations inherent in its development. This report describes theory and numerical implementation of the solute transport model. Details on simulation of water flow are contained in Lappala and others (1987), therefore little additional information on this topic is included in this report. Potential users of VS2DT should obtain a copy of Lappala and others (1987). The program is verified by comparing results to analytical solutions and previously published simulation results. Detailed description of data-input requirements and program structure are also included. Some additional modifications to computer program VS2D are presented as supplemental information.

Computer program VS2DT uses a finite-difference approximation to the advection-dispersion equation as well as the nonlinear water-flow equation (based on total hydraulic head). It can simulate problems in one, two (vertical cross section), or three dimensions (axially symmetric). The porous media may be heterogeneous and anisotropic, but principal directions must coincide with the coordinate axes. Boundary conditions for flow can take the form of fixed pressure heads, infiltration with ponding, evaporation from the soil surface, plant transpiration, or seepage faces. An extension to the program (Healy, 1987) also allows simulation of infiltration from trickle irrigation. Boundary conditions for solute transport include fixed solute concentration and fixed mass flux. Solute source/sink terms include first-order decay, equilibrium partitioning to the solid phase (as described by Langmuir or Freundlich isotherms), and ion exchange. The design of the program is modular, so that programmers can easily modify subroutines and functions in order to apply the model to particular field, laboratory, or hypothetical problems.

THEORY OF SOLUTE TRANSPORT IN VARIABLY SATURATED POROUS MEDIA

For purposes of this report solute transport is assumed to be described by the advection-dispersion equation. Derivation of that equation is based on mass conservation and Fick's law. Details of the derivation are beyond the scope of this report, but are contained in texts such as Bear (1979) or Hillel (1980).

Three mechanisms affect the movement of solutes under variably saturated conditions: (1) advective transport, in which solutes are moving with the flowing water; (2) hydrodynamic dispersion, in which molecular diffusion and variability of fluid velocity cause a spreading of solutes about the average direction of water flow; and (3) sources and sinks--including fluid sources, where a water of a specified chemical concentration is introduced to water of a different concentration, and chemical reactions such as radioactive decay or adsorption to the solid phase. The advection-dispersion equation that describes solute transport under variably saturated conditions can be written as (Bear, 1979, p. 251):

$$\frac{\partial(\theta c)}{\partial t} = \nabla\cdot\theta\ \bar{\bar{D}}_h\cdot\nabla c - \nabla\cdot\theta\bar{v}c + SS \qquad (1)$$

where θ = volumetric moisture content, dimensionless;
c = concentration of chemical constituent, ML^{-3} (mass per unit volume of water);
t = time, T;
∇ = del operator = $\frac{\partial}{\partial x} + \frac{\partial}{\partial y} + \frac{\partial}{\partial z}$, L^{-1};
$\bar{\bar{D}}_h$ = hydrodynamic dispersion tensor, L^2T^{-1};
$\bar{v}$ = fluid velocity vector, LT^{-1}; and
SS = source/sink terms, $ML^{-3}T^{-1}$.

Advection

The second term in the right hand side of equation 1 represents the divergence of the advective flux. This term accounts for changes in solute concentrations due to water moving and carrying solute with it. A simple one-dimensional experiment is shown in figure 1a to illustrate the advective and dispersive components of solute transport. In the experiment, a steady downward flow of solute-free water is obtained through a vertical column. At time t_0 the solute concentration is instantaneously increased to C_0 and maintained at that concentration throughout the remainder of the experiment. Relative concentration of the column outflow over time (commonly called a breakthrough curve) is shown in figure 1c. If advection is the only driving force for transport, then the tracer will move through the column as a plug and the breakthrough curve will simply be a step function, as shown by the dashed line in figure 1c.

Hydrodynamic Dispersion

The first term on the right-hand side of equation 1 represents the divergence of the flux of chemicals due to hydrodynamic dispersion. Hydrodynamic dispersion refers to a spreading process whereby molecules of a solute gradually move in directions different from that of the average ground-water flow. This spreading process is illustrated in the previously described experiment by the solid line in figure 1c. The theory behind dispersion has been reviewed extensively in the literature (see, for example, Bear, 1972, 1979; Scheidegger, 1961; Konikow and Grove, 1977). Two mechanisms comprise this phenomenon. The first is called mechanical dispersion and is caused by variations in the velocity field at the microscopic level. These variations are related to the tortuous nature of flow paths through porous media and the differences in velocity that occur across a single pore. Flow paths are not straight, but must follow the pores (fig. 2). Therefore molecules of solute will also be carried through these paths.

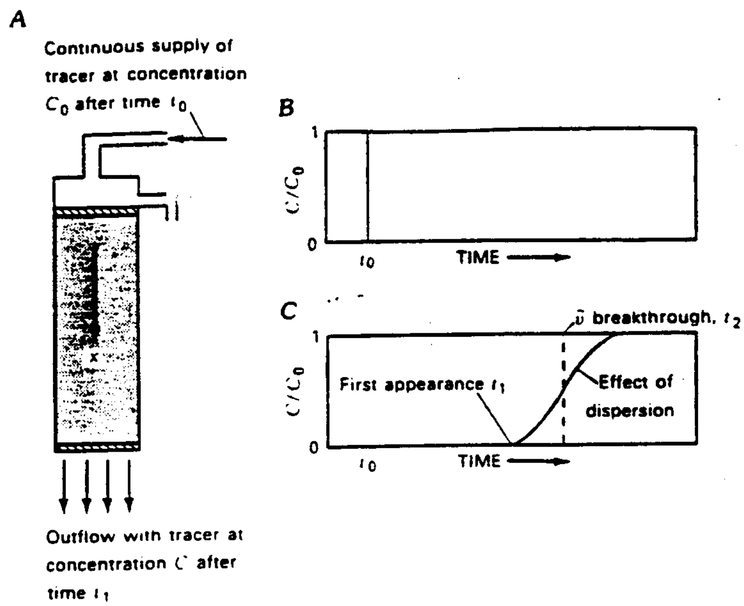


Figure 1.--Diagram showing effects of advection and dispersion of a tracer through a column of porous media: *A*) Column with steady flow and continuous supply of tracer after time t_0; *B*) step-function-type tracer input relation; *C*) relative tracer concentration in outflow from column (dashed line indicates plug flow condition and solid line illustrates effect of mechanical dispersion and molecular diffusion). Reproduced from Freeze and Cherry (1979, p. 390) and published with permission.

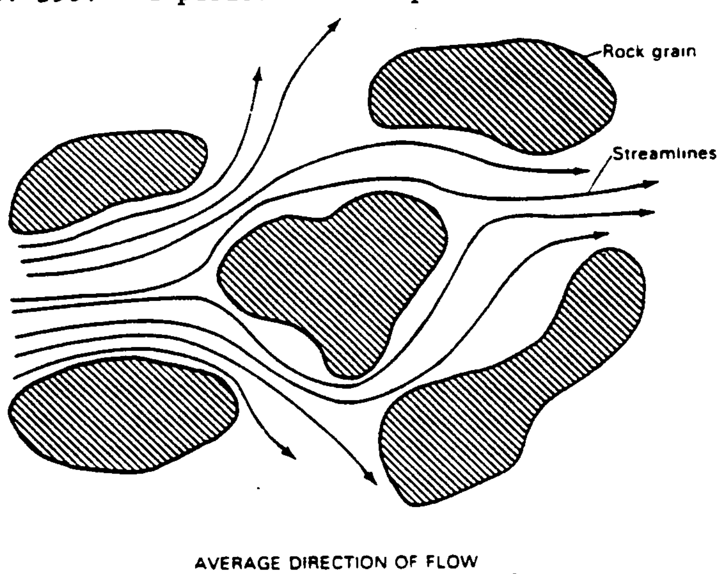


Figure 2.--Diagram showing spreading of flow paths..

The second mechanism contributing to hydrodynamic dispersion is molecular diffusion, which results from variations in solute concentrations. In the absence of water flow, molecules of solute will move from areas of high concentration to areas of low concentrations, in an effort to equalize concentrations everywhere. This mechanism also works when velocities are nonzero, causing lateral solute movement across streamtubes.

Following Bear (1979, p. 238) we can write the hydrodynamic dispersion tensor as the sum of tensors of mechanical dispersion ($\bar{\bar{D}}$) and molecular diffusion ($\bar{\bar{D}}_m$):

$$\bar{\bar{D}}_h = \bar{\bar{D}} + \bar{\bar{D}}_m \tag{2}$$

$$D_{ij} = \alpha_T |v| \delta_{ij} + (\alpha_L - \alpha_T) v_i v_j / |v| \tag{3}$$

$$D_{m_{ij}} = D_d \tau_{ij} \tag{4}$$

where α_T = transverse dispersivity of the porous medium, L;
$|v|$ = magnitude of the velocity vector, LT^{-1};
δ_{ij} = Kronecker delta, dimensionless
= 1 if $i = j$
= 0 if $i \neq j$;
α_L = longitudinal dispersivity of the porous media, L;
v_i = $i^{\underline{th}}$ component of the velocity vector, LT^{-1};
D_d = coefficient of molecular diffusion of solute in water, L^2T^{-1}; and
τ_{ij} = tortuosity, dimensionless.

In saturated porous media, dispersivity is theoretically a property of the geometry of the solid matrix. However, experimental data show a large scale effect, with dispersivities at the lab scale typically on the order of centimeters but at the field scale being on the order of several meters. There also is some question as to whether dispersivity varies as a function of moisture content in unsaturated porous media. In VS2DT, α_L and α_T are treated as constants. For this report, it is assumed that tortuosity is constant and uniformly aligned with the x and z axes so that $\tau_{xx} = \tau_{zz} = \tau$ and $\tau_{xz} = \tau_{zx} = 0$. Then, setting $D_m = D_d\tau$, we have $D_{m_{xx}} = D_{m_{zz}} = D_m$; $D_{m_{xz}} = D_{m_{zx}} = 0$. Therefore, the components of the two-dimensional hydrodynamic dispersion tensor can be written as:

$$D_{h_{xx}} = \alpha_L \frac{v_x^2}{|v|} + \alpha_T \frac{v_z^2}{|v|} + D_m \tag{5}$$

$$D_{h_{zz}} = \alpha_L \frac{v_z^2}{|v|} + \alpha_T \frac{v_x^2}{|v|} + D_m \tag{6}$$

$$D_{h_{zx}} = D_{h_{xz}} = (\alpha_L - \alpha_T)\, v_x v_z / |v| . \qquad (7)$$

Source/Sink Terms

Source/sink terms can be divided into 2 general categories: solute mass introduced to or removed from the domain by fluid sources and sinks; and mass introduced or removed by chemical reactions occurring within the water or between the water and the solid phase.

Fluid Sources and Sinks

Mathematically, the first category of source/sink terms can be represented by:

$$SS = c^* q \qquad (8)$$

where c^* = mass concentration in fluid source/sink, ML^{-3};
q = strength of fluid source/sink, T^{-1}.

When $q > 0$ (flow is into the system), c^* must be specified by the user. When $q < 0$ (flow is out of the system), c^* is set equal to the ambient solute concentration at the location where flow is leaving the system, that is:

$$c^* = c.$$

Decay, Adsorption, and Ion Exchange

For the second category of Source/Sink Terms three types of reactions may be simulated by the program. The first is a linear decay of the solute (such as radioactive decay). This is described by:

$$SS = \lambda \theta c \qquad (9)$$

where λ = the decay constant, T^{-1}.

The second type of reaction that may be simulated with VS2DT is sorption of solute from the water phase to the solid phase through physical or chemical attraction. Sorption may actually be a very complex process, but it is treated simplistically in VS2DT. Since the movement of water in soils is often slow relative to the rate of adsorption, it is assumed, for purposes of this computer program, that adsorption is equilibrium controlled. Therefore, the rate of change of solute mass in the sorped state is given by:

$$SS = \frac{\partial \rho_b \tilde{c}}{\partial t} = \rho_b \frac{\partial \tilde{c}}{\partial c} \frac{\partial c}{\partial t} \qquad (10)$$

where $\tilde{c}$ = concentration of solute mass in solid phase, MM^{-1};
ρ_b = bulk density of solid phase, ML^{-3}.

Experimental data are usually used to describe the relation between c and $\tilde{c}$. Plots of $\tilde{c}$ as a function of c at constant temperature are called isotherms. Often, empirically derived formulae are fit to these isotherms. Two such formulae may be used in VS2DT--the Freundlich or the Langmuir isotherm.

The Freundlich isotherm is given by:

$$\tilde{c} = K_f c^n \tag{11}$$

$$\frac{\partial \tilde{c}}{\partial c} = n\ K_f c^{n-1} \tag{12}$$

where K_f = Freundlich adsorption constant, and
n = Freundlich exponent.

Typical Freundlich isotherms are shown in figure 3. These isotherms are characterized by an unlimited capacity of the solid to adsorb the solute. A special case of the Freundlich isotherm occurs when n = 1. This produces a linear isotherm:

$$\tilde{c} = K_d c \tag{13}$$

$$\frac{\partial \tilde{c}}{\partial c} = K_d \tag{13a}$$

where K_d = equilibrium distribution coefficient, L^3M^{-1}.

Linear isotherms are shown in figure 3. Because of its simplicity, the linear isotherm is probably the most widely used isotherm in solute-transport simulations. For nonionic organic compounds K_d primarily represents adsorption to organic matter in soils. Since organic content of soils can vary greatly among and within individual soil types, the following equation is commonly used to approximate K_d (Jury and others, 1983):

$$K_d = f_{oc} K_{oc} \tag{14}$$

where f_{oc} = fraction of organic carbon in soil, MM^{-1}; and
K_{oc} = organic carbon distribution coefficient, L^3M^{-1}.

This approximation requires knowledge of f_{oc} instead of K_d; f_{oc} is much easier to measure than K_d. Several authors have reported correlations between K_{oc} and K_{ow}, the octanol-water partition coefficient (Karickhoff, 1981; Chiou and others, 1983). Rao and Davidson (1980) developed the following equation:

$$\log(K_{oc}/1000) = 1.029 \log(K_{ow}/1000) - 0.18 \tag{15}$$

where K_{oc} and K_{ow} are in m^3Kg^{-1}.

Values of K_{ow} may be obtained in standard indices such as Corwin and Hansch (1979).

The Langmuir isotherm is given by:

$$\tilde{c} = \frac{K_1 Q c}{1 + K_1 c} , \qquad (16)$$

$$\frac{\partial \tilde{c}}{\partial c} = \frac{K_1 Q}{(1 + K_1 c)^2} \qquad (16a)$$

where K_1 = Langmuir adsorption constant, L^3M^{-1}; and
Q = maximum number of adsorption sites.

Langmuir isotherms are characterized by a fixed number of adsorption sites. Figure 3 shows example Langmuir isotherms.

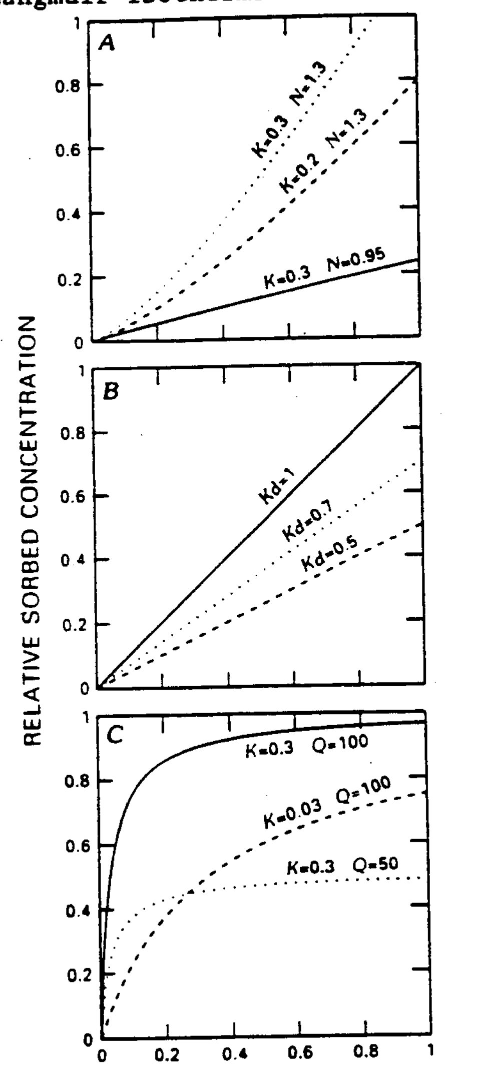


Figure 3.--Graph showing examples of isotherms: A) Freundlich; B) Linear; and C) Langmuir.

The third type of reaction is ion exchange, which is described by:

$$mc_1^{n} + n\tilde{c}_2 \rightleftarrows m\tilde{c}_1 + nc_2^{m} \quad (17)$$

where n is the valence for ion 1, and
m is the valence for ion 2.

The rate of change of ion concentration of solute mass in the solid phase can again be represented by equation 10. Four types of exchange are permitted in VS2DT, monovalent-monovalent exchange (m=n=1), divalent-divalent exchange (m=n=2), monovalent-divalent exchange (m=2, n=1), and divalent-monovalent exchange (m=1, n=2). The ion-exchange selectivity coefficient (K_m) is defined as:

$$K_m = \begin{cases} \dfrac{\tilde{c}_1 c_2}{\tilde{c}_2 c_1}, & \text{if } m = n, \\[2ex] \dfrac{\tilde{c}_1^{m} c_2^{n}}{\tilde{c}_2^{n} c_1^{m}}, & \text{if } m \neq n. \end{cases} \quad (18)$$

If only two ions are involved and C_o and $\hat{Q}$ are constant, where C_o is the total-solution concentration for ions 1 and 2, in terms of equivalents per volume; and $\hat{Q}$ is the ion-exchange capacity, in terms of equivalents per mass; then:

$$nc_1 + mc_2 = C_0 \quad (19)$$

$$n\tilde{c}_1 + m\tilde{c}_2 = \hat{Q}. \quad (20)$$

By combining equations 18, 19, and 20, the second component in the exchange process can be eliminated. For monovalent-monovalent exchange (such as the exchange of sodium and potassium) the following equations are produced:

$$\tilde{c} = \frac{K_m \hat{Q} c}{c(K_m - 1) + C_0} \quad (21)$$

$$\frac{\partial \tilde{c}}{\partial c} = \frac{K_m \hat{Q} C_0}{[c(K_m - 1) + C_0]^2}. \quad (21a)$$

Divalent-divalent exchange (such as the exchange of calcium and strontium) is described by:

$$\tilde{c} = \frac{K_m \hat{Q} c}{2c(K_m - 1) + C_0} \quad (22)$$

$$\frac{\partial \tilde{c}}{\partial c} = \frac{K_m \hat{Q} Co}{[2c(K_m - 1) + Co]^2} \tag{22a}$$

An example of monovalent-divalent exchange is the exchange of sodium with calcium. The following equations are produced for this exchange:

$$\tilde{c}^2(Co-c) + \tilde{c}K_m c^2 - c^2\hat{Q}K_m = 0 \tag{23}$$

$$\frac{\partial \tilde{c}}{\partial c} = \frac{\tilde{c}^2 - \tilde{c}2K_m c + 2c\hat{Q}K_m}{(Co-c)2\tilde{c} + K_m c^2} . \tag{23a}$$

In order to solve equation 23a, equation 23 must first be solved for $\tilde{c}$ by the quadratic formula.

Divalent-monovalent exchange (such as calcium-sodium exchange) is described by

$$\tilde{c}^2 4cK_m + \tilde{c}(-4c\hat{Q}K_m - (Co-2c)^2) + K_m c\hat{Q}^2 = 0 \tag{24}$$

$$\frac{\partial \tilde{c}}{\partial c} = \frac{-\tilde{c}^2 4K_m + \tilde{c}4(\hat{Q}K_m - (Co-2c)) - K_m\hat{Q}^2}{4cK_m(2\tilde{c}-\hat{Q}) - (Co-2c)^2} . \tag{24a}$$

Again, equation 24, which is quadratic in $\tilde{c}$, must be solved prior to solving equation 24a.

Additional information concerning the chemistry of adsorption and ion exchange can be found in texts such as Freeze and Cherry (1979) and Stumm and Morgan (1981). Bear (1972) and Grove and Stollenwerk (1984) present additional details on incorporating adsorption and ion-exchange into ground-water solute transport models.

Selection of adsorption or ion exchange must be made by the user at the time the computer program is compiled by selecting the appropriate version of the subroutine function VTRET. All other versions of that routine must be removed from the program or commented out. If ion exchange is selected, the user must take care to use consistent units for all variables. Ion exchange and adsorption cannot be simulated at the same time.

Boundary Conditions

The distinction between boundary conditions and source/sink terms is somewhat artificial; therefore, this discussion overlaps that in the previous section. Two types of boundaries may be specified for solute transport simulations: fixed concentration and fixed mass flux of solute. In addition, when fluid boundary conditions are such that water flow is into the system then the concentration of the water entering the system also must be specified.

When fluid boundary conditions are such that water flow is out of the system then the program assumes that the concentration of that water is identical to that in the finite-difference cell where the water is departing. An exception to this rule is removal of water from the system by evaporation. That water is assumed to be solute free.

Equation 1 can now be rewritten, assuming linear adsorption and noting that decay of solute mass in the solid phase also must be accounted for, as:

$$\frac{\partial}{\partial t}(\theta+\rho_b K_d)c = \nabla\cdot\theta\bar{\bar{D}}_h\cdot\nabla c - \nabla\cdot\theta\bar{v}c - \lambda(\theta+\rho_b K_d)c + c^*q \;. \tag{25}$$

NUMERICAL IMPLEMENTATION

Following the derivation of the finite difference approximation for the fluid flow equation (Lappala and others, 1987), let us look at the conservation of mass for a finite-difference cell of volume V and surface area $\hat{S}$ (fig. 4). We have

$$\int_V \frac{\partial(\theta+\rho_b K_d)c}{\partial t}\,dV = \int_V \nabla\cdot\theta\bar{\bar{D}}_h\cdot\nabla c\,dV - \int_V \nabla\cdot\bar{v}\theta c\,dV - \int_V \lambda(\theta+\rho_b K_d)c\,dV + \int_V c^*q\;dV \tag{26}$$

We can use the Gauss divergence theorem to transform the first two volume integrals on the right-hand side to surface integrals

$$\int_V \nabla\cdot\theta\bar{\bar{D}}_h\cdot\nabla c\,dV = \int_{\hat{S}} \theta\bar{\bar{D}}_h\cdot\nabla c\cdot\bar{n}\;d\hat{S} \tag{27}$$

$$\int_V \nabla\cdot\bar{v}\theta c\,dV = \int_{\hat{S}} \bar{v}\theta c\cdot\bar{n}\;d\hat{S} \tag{28}$$

where $\bar{n}$ is the outward normal unit vector.

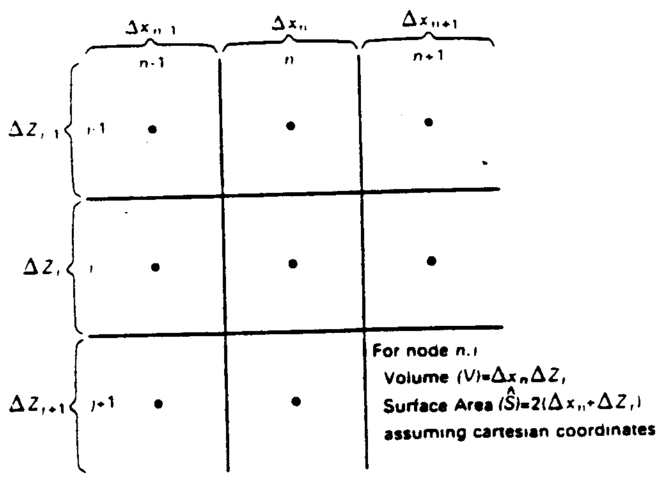


Figure 4.--Sketch showing finite-difference grid.

It is assumed that the volume V is small enough that within V the moisture content, bulk density, equilibrium distribution coefficient, and concentration can be considered constant, so that:

$$\int_V \frac{\partial(\theta+\rho_b Kd)c}{\partial t} = V \frac{\partial(\theta+\rho_b Kd)c}{\partial t} \tag{29}$$

$$\int_V \lambda(\theta+\rho_b K_d)c\,dV = V\lambda(\theta+\rho_b K_d)c; \tag{30}$$

$$\int_V c^* q\,dV = c^* qV = c^* q^* \tag{31}$$

where $q^* = qV$ = volumetric fluid flux, L^3T^{-1}.

We then have

$$V \frac{\partial(\theta+\rho_b Kd)c}{\partial t} = \int_{\hat{S}} \theta\bar{\bar{D}}_h \cdot \nabla c \cdot \bar{n}\,d\hat{S} - \int_{\hat{S}} \bar{v}\theta c \cdot \bar{n}\,d\hat{S} - V\lambda(\theta+\rho_b K_d)c + c^* q^* . \tag{32}$$

Spatial Discretization

The integral describing dispersive flux in equation 32 can be approximated by realizing that the surface of the finite difference cell contains four active faces (this is because of the assumption of two-dimensional flow; if three-dimensional flow were to be considered, then the number of faces would be 6). Referring to figure 4, we can write:

$$\int_{\hat{S}} \theta\bar{\bar{D}}_h \cdot \nabla c \cdot \bar{n}\,d\hat{S} = \sum_{\ell=1}^{4} \int_{\hat{S}_\ell} \theta\bar{\bar{D}}_h \cdot \nabla c \cdot \bar{n}\,d\hat{S}_\ell \tag{33}$$

$$\approx -\left[A\theta(D_{h_{xx}} \frac{\partial c}{\partial x} + D_{h_{xz}} \frac{\partial c}{\partial z})\right]_{n-1/2,j} + \left[A\theta(D_{h_{xx}} \frac{\partial c}{\partial x} + D_{h_{xz}} \frac{\partial c}{\partial z})\right]_{n+1/2,j}$$

$$- \left[A\theta(D_{h_{zz}} \frac{\partial c}{\partial z} + D_{h_{zx}} \frac{\partial c}{\partial x})\right]_{n,j-1/2} + \left[A\theta(D_{h_{zz}} \frac{\partial c}{\partial z} + D_{h_{zx}} \frac{\partial c}{\partial x})\right]_{n,j+1/2} \tag{34}$$

where
ℓ = index to faces of cell n,j;
n = nodal index in x direction;
j = nodal index in z direction;
$n\pm1/2$, $j\pm1/2$ = indices to boundary faces of cell n,j;
A = surface area of cell face normal to flux direction, L^2; and directions are positive from left to right and top to bottom.

AR 020529

Terms along cell boundaries that appear in equation 33 are evaluated in the following manner:

$$\theta_{n-1/2,j} = \frac{1}{2}(\theta_{n-1,j} + \theta_{n,j}) \tag{35}$$

$$\left.\begin{aligned} A_{n-1/2,j} &= A_{n+1/2,j} = \Delta z_j \\ A_{n,j-1/2} &= A_{n,j+1/2} = \Delta x_n \end{aligned}\right\}$$ Note: These equations are for cartesian coordinates. For radial coordinates the areas are given in Lappala and others (1987).

$$\left.\frac{\partial c}{\partial x}\right|_{n-1/2,j} = \frac{c_{n,j} - c_{n-1,j}}{1/2(\Delta x_{n-1} + \Delta x_n)} \tag{36}$$

$$\left.\frac{\partial c}{\partial z}\right|_{n-1/2,j} = 1/2 \, \frac{c_{n,j+1} + c_{n-1,j+1} - c_{n,j-1} - c_{n-1,j-1}}{\Delta z_j + 1/2(\Delta z_{j-1} + \Delta z_{j-1})} \tag{37}$$

Δz_j = height of finite-difference cells in row j, L; and

Δx_n = width of finite-difference cells in column n, L.

Spatial discretization of the advective component in equation 32 can be accomplished with either central or backward differencing. The integral representing the advective flux can be approximated by:

$$\int_{\hat{S}} \bar{v}\theta c \cdot \bar{n} d\hat{S} = \sum_{\ell=1}^{4} \int_{\hat{S}_\ell} \bar{v}\theta c \cdot \bar{n} d\hat{S} \tag{38}$$

$$= -[A\theta v_x c]_{n-1/2,j} + [A\theta v_x c]_{n+1/2,j} - [A\theta v_z c]_{n,j-1/2} + [A\theta v_z c]_{n,j+1/2} \tag{39}$$

where

$v_{x_{n-1/2,j}}$ = velocity in x direction at n-1/2,j, positive from left to right;

$$= -\left[\frac{K_r(h)K}{\theta}\frac{\partial H}{\partial x}\right]_{n-1/2,j} \tag{40}$$

$$= \left[\frac{K_r(h)K}{\theta}\right]_{n-1/2,j} \frac{H_{n-1,j} - H_{n,j}}{1/2(\Delta x_n + \Delta x_{n-1})} \tag{41}$$

$v_{z_{n,j-1/2}}$ = velocity in z direction at n,j-1/2, positive from top to bottom;

H = total hydraulic head, L;
= h - z;
h = pressure head, L;
$K_r(h)$ = relative hydraulic conductivity, dimensionless; and
K = saturated hydraulic conductivity, LT^{-1}.

$$c_{n-1/2,j} = \begin{cases} 1/2(c_{n,j}+c_{n-1,j}), & \text{if central differencing in space is specified by the user;} \\ c_{n-1,j}, & \text{if backward differencing in space is specified and } v_{x_{n-1/2,j}} > 0; \\ c_{n,j}, & \text{if backward differencing in space is specified and } v_{x_{n-1/2,j}} < 0. \end{cases}$$

Temporal Discretization

The time derivative in equation 32 can be approximated by two different methods in the program. Either a fully backward-in-time (fully implicit) or a centered-in-time (Crank-Nicholson) approximation may be selected by the user. For either method we can write

$$\frac{\partial}{\partial t}(\theta+\rho_b K_d)c = c\frac{\partial \theta}{\partial t} + (\theta+\rho_b K_d)\frac{\partial c}{\partial t} \tag{42}$$

$$\approx c^{i+1/2}\,\frac{\theta^{i+1} - \theta^{i}}{\Delta t} + (\theta^{i+1/2}+\rho_b K_d)\frac{c^{i+1} - c^{i}}{\Delta t} \tag{43}$$

where i = index for previous time step;
i+1 = index for current time step;
Δt = length of the i+1st time step, T;
$c^{i+1/2}$ is assumed to be equal to c^{i+1}; and
$\theta^{i+1/2}$ is assumed to be equal to θ^{i+1}.

For the fully implicit formulation, concentrations on the right-hand side of equation 32 are all evaluated at the i+1 time level. For the time centered formulation, the terms on the right-hand side are evaluated as the average between the current time step and the previous time step. The time centered scheme is more accurate than the fully implicit scheme. It is second order correct in Δt while the fully implicit method is first order correct in Δt. However, as will be discussed later, for some problems the fully implicit methods may have some advantages. The final finite-difference form for equation 26 can now be written as:

AR 020531

$$\hat{A}^{i+1}c^{i+1}_{n-1,j} + \hat{B}^{i+1}c^{i+1}_{n,j-1} + \hat{C}^{i+1}c^{i+1}_{n+1,j} + \hat{D}^{i+1}c^{i+1}_{n,j+1} + \hat{E}^{i+1}c^{i+1}_{n,j} = RHS \quad (44)$$

$$\hat{A}^{i+1} = TC\left[(A\theta)_{n-1/2,j}\left[\frac{D_{h_{xx\ n-1/2,j}}}{1/2(\Delta x_n+\Delta x_{n-1})} + \frac{1}{2}v_{x_{n-1/2,j}}\right] + \hat{G} - \hat{H}\right]^{i+1} \quad (45a)$$

$$\hat{B}^{i+1} = TC\left[(A\theta)_{n,j-1/2}\left[\frac{D_{h_{zz\ n,j-1/2}}}{1/2(\Delta z_j+\Delta z_{j-1})} + \frac{1}{2}v_{z_{n,j-1/2}}\right] + \hat{F} - \hat{I}\right]^{i+1} \quad (45b)$$

$$\hat{C}^{i+1} = TC\left[(A\theta)_{n+1/2,j}\left[\frac{D_{h_{xx\ n+1/2,j}}}{1/2(\Delta x_n+\Delta x_{n+1})} - \frac{1}{2}v_{x_{n+1/2,j}}\right] - \hat{G} + \hat{H}\right]^{i+1} \quad (45c)$$

$$\hat{D}^{i+1} = TC\left[(A\theta)_{n,j+1/2}\left[\frac{D_{h_{zz\ n,j+1/2}}}{1/2(\Delta z_j+\Delta z_{j+1})} - \frac{1}{2}v_{z_{n,j+1/2}}\right] - \hat{F} + \hat{I}\right]^{i+1} \quad (45d)$$

$$\hat{E} = -\hat{A}-\hat{B}-\hat{C}-\hat{D} + TC\left[(A\theta v_x)_{n-1/2,j} + (A\theta v_z)_{n,j-1/2} - (A\theta v_x)_{n+1/2,j} - (A\theta v_z)_{n,j+1/2}\right]$$

$$- \frac{V}{\Delta t}(2\theta^{i+1}_{n,j}+\rho_b K_d-\theta^{i}_{n,j}) - V\lambda(\theta^{i+1}_{n,j}+\rho_b K_d) \quad (45e)$$

$$RHS = -\frac{V}{\Delta t}c^{i}_{n,j}(\theta^{i+1}_{n,j}+\rho_b K_d) - 2(1-TC)x[\hat{A}^{i}c^{i}_{n-1,j}+\hat{B}^{i}c^{i}_{n,j-1}+\hat{C}^{i}c^{i}_{n+1,j}+\hat{D}^{i}c^{i}_{n,j+1}+$$

$$\hat{E}^{i}c^{i}_{n,j}] - (\hat{F}^{i+1}+\hat{G}^{i+1})c^{i+1}_{n-1,j-1} + (\hat{H}^{i+1}+\hat{F}^{i+1})c^{i+1}_{n-1,j+1} +$$

$$(\hat{I}^{i+1}+\hat{G}^{i+1})c^{i+1}_{n+1,j-1} - (\hat{H}^{i+1}+\hat{I}^{i+1})c^{i+1}_{n+1,j+1} \quad (45f)$$

where

$$\hat{F} = \frac{1}{2} \frac{(A\theta \, D_{h_{xz}})_{n-1/2,j}}{\Delta z_j + 1/2(\Delta z_{j+1} + \Delta z_{j-1})}$$

$$\hat{G} = \frac{1}{2} \frac{(A\theta \, D_{h_{zx}})_{n,j-1/2}}{\Delta x_n + 1/2(\Delta x_{n-1} + \Delta x_{n+1})}$$

$$\hat{H} = \frac{1}{2} \frac{(A\theta \, D_{h_{zx}})_{n,j+1/2}}{\Delta x_n + 1/2(\Delta x_{n-1} + \Delta x_{n+1})}$$

$$\hat{I} = \frac{1}{2} \frac{(A\theta \, D_{h_{xz}})_{n+1/2,j}}{\Delta z_j + 1/2(\Delta z_{j-1} + \Delta z_{j+1})}$$

$$TC = \begin{cases} 1 & \text{, fully implicit; and} \\ 1/2 & \text{, time centered.} \end{cases}$$

The formulations given in equations 45 are based on central-difference approximations for the spatial derivatives in equation 26. If backward-in-space differences are used, equation 45 needs to be modified only slightly. For example, if $v_{x_{n-1/2,j}} > 0$ then equation 45a would become:

$$\hat{A} = TC\left[(A\theta)_{n-1/2,j}\left[\frac{D_{h_{xx\ n-1/2,j}}}{1/2(\Delta x_n + \Delta x_{n-1})} + v_{x_{n-1/2,j}}\right] + \hat{G} - \hat{H}\right]$$

and the term containing $v_{x_{n-1/2,j}}$ in equation 45e would be eliminated.

If fluid source/sink terms are present then equations 45e and 45f must be modified to account for them in the following manner:

if $q^* > 0$ then
$RHS = RHS + q^*c^*$ (46)

if $q^* < 0$ then
$\hat{E} = \hat{E} - q^*$. (47)

AR 020533

Equation 44 must be solved for each node in the finite difference grid. Thus, we have reduced the problem to that of solving the matrix equation:

$$\bar{\bar{A}}\ \bar{c}^{i+1} = \overline{RHS} \tag{48}$$

where A = a pentadiagonal square coefficient matrix;

$\bar{c}^{i+1}$ = the vector of unknown concentrations at the i+1 time level; and

RHS = the vector defined by equation 45f.

As with the flow equation, VS2DT actually solves the residual form of equation 48 with an iterative matrix solver:

$$\bar{\bar{A}}\ \Delta\bar{c}^{i+1,k+1} = \overline{RHS}^{k} - \bar{\bar{A}}\bar{c}^{i+1,k}$$

where $\Delta\bar{c}^{i+1,k+1} = \bar{c}^{i+1,k+1} - \bar{c}^{i+1,k}$

k = iteration index; and the terms at i+1 time level in equation 45f are assigned values from the k^{th} iteration.

Selection of fully implicit or time-centered differencing is a user option. The optimum method is problem dependent. Although the Crank-Nicholson method is more accurate, it can produce results which oscillate around the true solution. This oscillation is illustrated in the verification problems. Fully implicit time differencing eliminates the oscillations but can introduce numerical dispersion or the smearing of sharp fronts. Numerical dispersion can be controlled by limiting the size of each time step; however, small time steps can add great expense and computation time to each simulation.

Source/Sink Terms

Function subprograms (all named VTRET) have been written and tested for calculation of $\rho_b \frac{\partial\bar{c}}{\partial c}$ for adsorption and ion exchange. Six options are available to the user: Freundlich isotherm, Langmuir isotherm, monovalent-monovalent ion exchange, divalent-divalent ion exchange, monovalent-divalent ion exchange, and divalent-monovalent ion exchange.

As listed under Supplemental Information, the program is set up to use the Langmuir isotherm. The five other versions of VTRET are included as comment cards at the end of the program. To use any of the other options the required version of VTRET should be stripped of comment designation, compiled, and loaded with the compiled version of VS2DT that does not contain the Langmuir isotherm version of VTRET. Only one version of VTRET should be loaded with VS2DT at any one time. Variables required by the isotherm or ion-exchange option may vary with texture class (for example, if a simulation involves multiple soil types, then each soil type may have a different ion-exchange capacity).

Boundary and Initial Conditions

Specification of solute transport boundary conditions cannot be done independently of specification of flow boundaries. Two basic boundary conditions can be specified with regard to concentration: fixed-concentration node and a fixed-mass-flux node. In addition, for constant-head and constant-flux flow boundaries, the concentration of any flow entering the system must be specified. Table 1 lists the permissible combination of flow and transport boundary conditions. While some combinations that are not allowed may still be solved by the model, they are not permitted because no practical application for them exists.

Table 1.--*Summary of permissible combinations of boundary conditions*

[X, permitted; Y, mandatory; --, not allowed]

	Transport boundary conditions			
Flow boundary Conditions	Fixed Concentrations	Fixed mass flux	No specified boundary	Specified Concentration of inflow
Fixed head				
flow into domain	X	--	X	Y
flow out of domain	--	--	X	Y
Fixed flux				
into domain	X	--	X	Y
out of domain	--	--	X	Y
No specified boundary	X	X	X	--
Evaporation	--	--	X	--
Plant transpiration	--	--	X	--
Seepage face	--	--	X	--

For flow boundaries where flow is into the domain, there are two possible options for transport boundary conditions: 1) no specified boundary, for which the mass-flux rate into the domain is calculated as the influx rate times concentration of inflow (this is essentially treated as a fixed-mass-flux or Neumann boundary condition); and 2) fixed concentration or Dirichlet boundary condition, for which the mass flux rate into the domain is calculated as the sum of influx rate times concentrations of inflow plus the rate of dispersive flux from the boundary node. For flow boundaries where flow leaves the domain no transport boundary condition can be specified. Under this condition the rate of solute flux out of the domain is equal to the rate of water flux times the concentration at the exit node--diffusive flux out of the domain is not allowed. The evaporation boundary condition is treated differently from other boundaries where water leaves the domain; evaporating water is assumed to be solute free (no solute is allowed to leave the domain through evaporation). Therefore, solute may become concentrated in evaporation nodes as evaporation proceeds. The fixed-mass-flux boundary condition is used to represent a strictly diffusive flux and can be located only on nodes at which there is no inflow to or outflow from the domain.

Mass Balance

At the completion of every time step, the mass flux into and out of the system, as well as the change in mass stored in the system, is calculated. Printout of mass-balance results is an option in VS2DT. Fluxes into and out of the system are divided into dispersive/diffusive and advective fluxes. The former refers to fluxes dependent upon the concentration gradient between fixed concentration nodes and adjacent nodes. The latter represents changes in mass within the system due to mass entering or leaving the system with flowing water. When water flow is into the system, that water is assumed to have a concentration equal to that specified by the user. When water flows out of the system the concentration of that water is set equal to the concentration of the node from which the water is moving. The gain or loss of mass through source/sink terms also is determined.

The change in mass stored within the system over the last time step is calculated as:

$$\Delta SC^{i+1} = \sum_{n=1}^{NXR}\sum_{j=1}^{NLY} c^{i+1}_{n,j}\theta^{i+1}_{n,j}\left(1+\frac{Ss}{\phi}H^{i+1}_{n,j}\right) - c^{i}_{n,j}\theta^{i}_{n,j}\left(1+\frac{Ss}{\phi}H^{i}_{n,j}\right) V_{n,j} \qquad (50)$$

where ΔSC^{i+1} = change in mass storage between time steps i and i+1, M;
NXR = number of columns in grid, dimensionless;
NLY = number of rows in grid, dimensionless;
Ss = specific storage, L^{-1};
ϕ = porosity, dimensionless.
θ = volumetric moisture content (dimensionless)

The loss of mass due to decay and adsorption is calculated as:

$$\Delta SC_d^{i+1} = -\sum_{n=1}^{NXR}\sum_{j=1}^{NLY} V_{n,j}\ \lambda\Delta t\ c^{i+1}_{n,j}\left(\theta^{i+1}_{n,j}+\rho_b\frac{\partial\bar{c}}{\partial c}\right) + \rho_b\frac{\partial\bar{c}}{\partial c}\left(c^{i+1}_{n,j} - c^{i}_{n,j}\right) \qquad (51)$$

where ΔSC_d^{i+1} = change in mass due to decay and adsorption between time steps i and i+1, M.

COMPUTER PROGRAM

This section contains information on program structure, data input, and considerations on spatial and temporal grid design for model application.

Program Structure

A listing of all new variables added to VS2DT for solute transport simulation is given in Table 2. An effort was made to keep the computer program in modular form so that it could be easily customized for particular applications. Three subroutines and one function routine were added to VS2D to allow simulation of solute transport, these are described below:

1) VTVELO — Subroutine that calculates intercell velocities in the x and z directions.

2) VTDCOEF — Subroutine that calculates the components of the dispersion coefficient tensor.

3) VTSETUP — Subroutine that assembles the matrix equation and calls the matrix solving routine.

4) VTRET — Function subroutine that calculates the adsorption term $\rho_b \frac{\partial \tilde{c}}{\partial c}$.

Six versions of routine VTRET are included in the program listing in Supplemental Information. These versions correspond to the Freundlich and Langmuir adsorption isotherms and monovalent-monova'ent, divalent-divalent, monovalent-divalent, and divalent-monovalent ion exchange. When compiling the computer program, the user must select the appropriate version and be sure that the other versions are deleted or appear as comments. File definitions are similar to those described in Lappala and others (1987). However, when output is requested to Fortran file number 8, both pressure heads and concentrations are printed at the appropriate times. Similarly, concentrations are printed to Fortran file number 11 for selected observation points. The user also may now specify which mass-balance components are printed to file 9 (this option is described under Modifications to Computer Program VS2D in Supplemental Information).

Instructions for Data Input

Input-data formats are described in Table 3. The formats are very similar to the original VS2D input formats described by Lappala and others (1987). Several additional input variables are required for simulation of solute transport. If solute transport is not to be simulated then only two new variables need to be coded (ANG on line A-2, and TRANS on line A-6) in addition to those variables described in the original VS2D documentation. The variable RHOZ on line B-2 is no longer entered by the user. New users of VS2DT should obtain a copy of Lappala and others (1987) for additional information on input variables dealing with simulation of water flow.

Table 2.--*Definitions of new VS2DT program variables*

[NN, number of nodes]

Variable	Definition
DX1(NN)	XX Component of hydrodynamic dispersion tensor at left side of cell times $\Delta x/\Delta z$, L^2T^{-1}.
DX2(NN)	XZ Component of hydrodynamic dispersion tensor at left side of cell times $\Delta x/2\Delta z$, L^2T^{-1}.
DZ1(NN)	ZZ Component of hydrodynamic dispersion tensor at top of cell times $\Delta z/\Delta x$, L^2T^{-1}.
DZ2(NN)	ZX Component of hydrodynamic dispersion tensor at top of cell times $\Delta z/2\Delta x$, L^2T^{-1}.
VX(NN)	X Velocity at left side of cell, LT^{-1}.
VZ(NN)	Z Velocity at top of cell, LT^{-1}.
CC(NN)	Concentration, ML^{-3}.
COLD(NN)	Concentration at previous time step, ML^{-3}.
CS(NN)	Concentration of specified fluid sources, ML^{-3}.
QT(NN)	Fluid flux through constant head nodes, L^3T^{-1}.
NCTYP(NN)	Boundary condition or cell type indicator: 0 = internal node, 1 = specified concentration node, and 2 = specified solute flux node.
RET(NN)	Slope of adsorption isotherm times bulk density, dimensionless.
ANG	Angle at which grid is to be tilted, degrees.
TRANS	If = T, solute transport and flow are to be simulated; if = F, only flow is simulated.
TRANS1	If = T, matrix solver solves for head; if = F, matrix solver solves for concentration.
SSTATE	If = T, steady-state flow has been achieved.
CIS	If = T, centered-in-space differencing is used for transport equation; if = F, backward-in-space differencing is used.
CIT	If = T, centered-in-time differencing is used for transport equation; if = F, backward-in-time differencing is used.
EPS1	Convergence criteria for transport equation, ML^{-3}.
VPNT	If = T, velocities are written to file 6.
SORP	If = T, nonlinear sorption is to be simulated.

Table 3.--*Input data formats*

Card	Variable	Description
		[Line group A read by VSEXEC]
A-1	TITL	80-character problem description (formatted read, 20A4).
A-2	TMAX	Maximum simulation time, T.
	STIM	Initial time (usually set to 0), T.
	ANG	Angle by which grid is to be tilted (Must be between -90 and +90 degrees, ANG = 0 for no tilting, see Supplemental Information for further discussion), degrees.
A-3	ZUNIT	Units used for length (A4).
	TUNIT	Units used for time (A4).
	CUNX	Units used for mass (A4).
Note: Line A-3 is read in 3A4 format, so the unit designations must occur in columns 1-4, 5-8, 9-12, respectively.		
A-4	NXR	Number of cells in horizontal or radial direction.
	NLY	Number of cells in vertical direction.
A-5	NRECH	Number of recharge periods.
	NUMT	Maximum number of time steps.
A-6	RAD	Logical variable = T if radial coordinates are used; otherwise = F.
	ITSTOP	Logical variable = T if simulation is to terminate after ITMAX iterations in one time step; otherwise = F.
	TRANS	Logical variable = T if solute transport is to be simulated.
Line A-6A is present only if TRANS = T.		
A-6A	CIS	Logical variable = T if centered-in-space differencing is to be used; = F if backward-in-space differencing is to be used for transport equation.
	CIT	Logical variable = T if centered-in-time differencing is to be used; = F if backward-in-time or fully implicit differencing is to be used.
	SORP	Logical variable = T if nonlinear sorption or ion exchange is to be simulated. Nonlinear sorption occurs when ion exchange, Langmuir isotherms, or Freundlich isotherms with n not equal to 1 are used.
A-7	F11P	Logical variable = T if head, moisture content, and saturation at selected observation points are to be written to file 11 at end of each time step; otherwise = F.

Table 3.--*Input data formats*--Continued

Card	Variable	Description
A-7--Continued	F7P	Logical variable = T if head changes for each iteration in every time step are to be written in file 7; otherwise = F.
	F8P	Logical variable = T if output of pressure heads (and concentrations if TRANS = T) to file 8 is desired at selected observation times; otherwise = F.
	F9P	Logical variable = T if one-line mass balance summary for each time step to be written to file 9; otherwise = F.
	F6P	Logical variable = T if mass balance is to be written to file 6 for each time step; = F if mass balance is to be written to file 6 only at observation times and ends of recharge periods.
A-8	THPT	Logical variable = T if volumetric moisture contents are to be written to file 6; otherwise = F.
	SPNT	Logical variable = T if saturations are to be written to file 6; otherwise = F.
	PPNT	Logical variable = T if pressure heads are to be written to file 6; otherwise = F.
	HPNT	Logical variable = T if total heads are to be written to file 6; otherwise = F.
	VPNT	Logical variable = T if velocities are to be written to file 6; requires TRANS = T.
A-9	IFAC	= 0 if grid spacing in horizontal (or radial) direction is to be read in for each column and multiplied by FACX. = 1 if all horizontal grid spacing is to be constant and equal to FACX. = 2 if horizontal grid spacing is variable, with spacing for the first two columns equal to FACX and the spacing for each subsequent column equal to XMULT times the spacing of the previous column, until the spacing equals XMAX, whereupon spacing becomes constant at XMAX.

Table 3.--*Input data formats*--Continued

Card	Variable	Description
A-9--Continued	FACX	Constant grid spacing in horizontal (or radial) direction (if IFAC=1); constant multiplier for all spacing (if IFAC=0); or initial spacing (if IFAC=2), L.
Line set A-10 is present if IFAC = 0 or 2.		
If IFAC = 0, A-10	DXR	Grid spacing in horizontal or radial direction. Number of entries must equal NXR, L.
If IFAC = 2, A-10	XMULT	Multiplier by which the width of each node is increased from that of the previous node.
	XMAX	Maximum allowed horizontal or radial spacing, L.
A-11	JFAC	= 0 if grid spacing in vertical direction is to be read in for each row and multiplied by FACZ. = 1 if all vertical grid spacing is to be constant and equal to FACZ. = 2 if vertical grid spacing is variable, with spacing for the first two rows equal to FACZ and the spacing for each subsequent row equal to ZMULT times the spacing at the previous row, until spacing equals ZMAX, whereupon spacing becomes constant at ZMAX.
	FACZ	Constant grid spacing in vertical direction (if JFAC=1); constant multiplier for all spacing (if JFAC =0); or initial vertical spacing (if JFAC=2), L.
Line set A-12 is present only if JFAC = 0 or 2.		
If JFAC = 0, A-12	DELZ	Grid spacing in vertical direction; number of entries must equal NLY, L.
If JFAC = 2, A-12	ZMULT	Multiplier by which each node is increased from that of previous node.
	ZMAX	Maximum allowed vertical spacing, L.
Line sets A-13 to A-14 are present only if F8P = T,		
A-13	NPLT	Number of time steps to write heads and concentrations to file 8 and heads, concentrations, saturations, and/or moisture contents to file 6.

Table 3.--*Input data formats*--Continued

Card	Variable	Description
A-14	PLTIM	Elapsed times at which pressure heads and concentrations are to be written to file 8, and heads, concentrations, saturations, and/or moisture contents to file 6, T.
Line sets A-15 to A-16 are present only if F11P = T,		
A-15	NOBS	Number of observation points for which heads, concentrations, moisture contents, and saturations are to be written to file 11.
A-16	J,N	Row and column of observation points. A double entry is required for each observation point, resulting in 2xNOBS values.
Lines A-17 and A-18 are present only if F9P = T.		
A-17	NMB9	Total number of mass balance components to be written to File 9.
A-18	MB9	The index number of each mass balance component to be written to file 9. (See table 7 in Supplemental Information for index key)
	[Line group B read by subroutine VSREAD]	
B-1	EPS	Closure criteria for iterative solution of flow equation, units used for head, L.
	HMAX	Relaxation parameter for iterative solution. See discussion in Lappala and others (1987) for more detail. Value is generally in the range of 0.4 to 1.2.
	WUS	Weighting option for intercell relative hydraulic conductivity: WUS = 1 for full upstream weighting. WUS = 0.5 for arithmetic mean. WUS = 0.0 for geometric mean.
	EPS1	Closure criteria for iterative solution of transport equation, units used for concentration, ML^{-3}. Present only if TRANS = T.
B-3	MINIT	Minimum number of iterations per time step.
	ITMAX	Maximum number of iterations per time step. Must be less than 200.
B-4	PHRD	Logical variable = T if initial conditions are read in as pressure heads; = F if initial conditions are read in as moisture contents.

Table 3.--*Input data formats*--Continued

Card	Variable	Description
B-5	NTEX	Number of textural classes or lithologies having different values of hydraulic conductivity, specific storage, and/or constants in the functional relations among pressure head, relative conductivity, and moisture content.
	NPROP	Number of flow properties to be read in for each textural class. When using Brooks and Corey or van Genuchten functions, set NPROP = 6, and when using Haverkamp functions, set NPROP = 8. When using tabulated data, set NPROP = 6 plus number of data points in table. [For example, if the number of pressure heads in the table is equal to N1, then set NPROP =3*(N1+1)+3]
	NPROP1	Number of transport properties to be read in for each textural class. For no adsorption set NPROP1 = 6. For a Langmuir or Freundlich isotherm set NPROP1 = 7. For ion exchange set NPROP1 = 8. Present only if TRANS = T.
Line sets B-6, B-7, and B-7A must be repeated NTEX times		
B-6	ITEX	Index to textural class.
B-7	ANIZ(ITEX)	Ratio of hydraulic conductivity in the z-coordinate direction to that in the x-coordinate direction for textural class ITEX.
	HK(ITEX,1)	Saturated hydraulic conductivity (K) in the x-coordinate direction for class ITEX, LT^{-1}.
	HK(ITEX,2)	Specific storage (S_s) for class ITEX, L^{-1}.
	HK(ITEX,3)	Porosity for class ITEX.

Definitions for the remaining sequential values on this line are dependent upon which functional relation is selected to represent the nonlinear coefficients. Four different functional relations are allowed: (1) Brooks and Corey, (2) van Genuchten, (3) Haverkamp, and (4) tabular data. The choice of which of these to use is made when the computer program is compiled, by including only the function subroutine which pertains to the desired relation (see discussion in Lappala and others (1987) for more detail).

Table 3.--*Input data formats*--Continued

Card	Variable	Description
B-7--Continued		

In the following descriptions, definitions for the different functional relations are indexed by the above numbers. For tabular data, all pressure heads are input first (in decreasing order from the largest to the smallest), all relative hydraulic conductivities are then input in the same order, followed by all moisture contents.

HK(ITEX,4)	(1)	h_b, L. (must be less than 0.0).
	(2)	α', L. (must be less than 0.0).
	(3)	A', L. (must be less than 0.0).
	(4)	Largest pressure head in table.
HK(ITEX,5)	(1)	Residual moisture content (θ_r).
	(2)	Residual moisture content (θ_r).
	(3)	Residual moisture content (θ_r).
	(4)	Second largest pressure head in table.
HK(ITEX,6)	(1)	λ, pore-size distribution index.
	(2)	β'.
	(3)	B'.
	(4)	Third largest pressure head in table.
HK(ITEX,7)	(1)	Not used.
	(2)	Not used.
	(3)	α, L. (must be less than 0.0).
	(4)	Fourth largest pressure head in table.
HK(ITEX,8)	(1)	Not used.
	(2)	Not used.
	(3)	β.
	(4)	Fifth largest pressure head in table.

For functional relations (1), (2), and (3) no further values are required on this line for this textural class. For tabular data (4), data input continues as follows:

HK(ITEX,9)	Next largest pressure head in table.
K(ITEX,N1+3)	Minimum pressure head in table. (Here N1 = Number of pressure heads in table; NPROP = 3*(N1+1)+3).
HK(ITEX,N1+4)	Always input a value of 99.
HK(ITEX,N1+5)	Relative hydraulic conductivity corresponding to first pressure head.
HK(ITEX,N1+6)	Relative hydraulic conductivity corresponding to second pressure head.
.	
.	
.	
HK(ITEX,2*N1+4)	Relative hydraulic conductivity corresponding to smallest pressure head.
HK(ITEX,2*N1+5)	Always input a value of 99.
HK(ITEX,2*N1+6)	Moisture content corresponding to first pressure head.

Table 3.--*Input data formats*--Continued

Card	Variable	Description
B-7--Continued		
	HK(ITEX,2*N1+7)	Moisture content corresponding to second pressure head.
	.	
	.	
	.	
	HK(ITEX,3*N1+5)	Moisture content corresponding to smallest pressure head.
	HK(ITEX,3*N1+6)	Always input a value of 99.

Regardless of which functional relation is selected there must be NPROP+1 values on line B-7.

Line B-7A is present only if TRANS = T.

B-7A	HT(ITEX,1)	α_L, L.
	HT(ITEX,2)	α_T, L.
	HT(ITEX,3)	D_m, L^2T^{-1}.
	HT(ITEX,4)	λ, decay constant, T^{-1}.
	HT(ITEX,5)	ρ_b (can be set to 0 for no adsorption or ion exchange), ML^{-3}.
	HT(ITEX,6)	= 0 for no adsorption or ion exchange, = K_d for linear adsorption isotherm, = K_1 for Langmuir isotherm, = K_f for Freundlich isotherm, = K_m for ion exchange.
	HT(ITEX,7)	= Q for Langmuir isotherm, = n for Freundlich isotherm (Note: n is a real, rather than an integer, variable), = $\hat{Q}$ for ion exchange, not used when adsorption is not simulated.
	HT(JTEX,8)	= C_0 for ion exchange, only used for ion exchanged.
B-8	IROW	If IROW = 0, textural classes are read for each row. This option is preferable if many rows differ from the others. IF IROW = 1, textural classes are read in by blocks of rows, each block consisting of all the rows in sequence consisting of uniform properties or uniform properties separated by a vertical interface.

Line set B-9 is present only if IROW = 0.

B-9	JTEX	Indices (ITEX) for textural class for each node, read in row by row. There must be NLY*NXR entries.

Table 3.--*Input data formats*--Continued

Card	Variable	Description
Line set B-10 is present only if IROW = 1.		
As many groups of B-10 variables as are needed to completely cover the grid are required. The final group of variables for this set must have IR = NXR and JBT = NLY.		
B-10	IL	Left hand column for which texture class applies. Must equal 1 or [IR(from previous card)+1].
	IR	Right hand column for which texture class applies. Final IR for sequence of rows must equal NXR.
	JBT	Bottom row of all rows for which the column designations apply. JBT must not be increased from its initial or previous value until IR = NXR.
	JRD	Texture class within block.
Note: As an example, for a column of uniform material; IL = 1, IR = NXR, JBT = NLY, and JRD = texture class designation for the column material. One line will represent the set for this example.		
B-11	IREAD	If IREAD = 0, all initial conditions in terms of pressure head or moisture content as determined by the value of PHRD are set equal to FACTOR. If IREAD = 1, all initial conditions are read from file IU in user-designated format and multiplied by FACTOR. If IREAD = 2 initial conditions are defined in terms of pressure head, and an equilibrium profile is specified above a free-water surface at a depth of DWTX until a pressure head of HMIN is reached. All pressure heads above this are set to HMIN.
	FACTOR	Multiplier or constant value, depending on value of IREAD, for initial conditions, L.
Line B-12 is present only if IREAD = 2,		
B-12	DWTX	Depth to free-water surface above which an equilibrium profile is computed, L.
	HMIN	Minimum pressure head to limit height of equilibrium profile; must be less than zero, L.

AR 020546

Table 3.--*Input data formats*--Continued

Card	Variable	Description
Line B-13 is read only if IREAD = 1,		
B-13	IU	Unit number from which initial head values are to be read.
	IFMT	Format to be used in reading initial head values from unit IU. Must be enclosed in quotation marks, for example '(10X,E10.3)'.
B-14	BCIT	Logical variable = T if evaporation is to be simulated at any time during the simulation; otherwise = F.
	ETSIM	Logical variable = T if evapotranspiration (plant-root extraction) is to be simulated at any time during the simulation; otherwise = F.
Line B-15 is present only if BCIT = T or ETSIM = T.		
B-15	NPV	Number of ET periods to be simulated. NPV values for each variable required for the evaporation and/or evapotranspiration options must be entered on the following lines. If ET variables are to be held constant throughout the simulation code, NPV = 1.
	ETCYC	Length of each ET period, T.

Note: For example, if a yearly cycle of ET is desired and monthly values of PEV, PET, and the other required ET variables are available, then code NPV = 12 and ETCYC = 30 days. Then, 12 values must be entered for PEV, SRES, HA, PET, RTDPTH, RTBOT, RTTOP, and HROOT. Actual values, used in the program, for each variable are determined by linear interpolation based on time.

Card	Variable	Description
Line B-16 to B-18 are present only if BCIT = T.		
B-16	PEVAL	Potential evaporation rate (PEV) at beginning of each ET period. Number of entries must equal NPV, LT^{-1}.

To conform with the sign convention used in most existing equations for potential evaporation, all entries must be greater than or equal to 0. The program multiplies all nonzero entries by -1 so that the evaporative flux is treated as a sink rather than a source.

Table 3.--*Input data formats*--Continued

Card	Variable	Description
B-17	RDC(1,J)	Surface resistance to evaporation (SRES) at beginning of ET period, L^{-1}. For a uniform soil, SRES is equal to the reciprocal of the distance from the top active node to land surface, or 2./DELZ(2). If a surface crust is present, SRES may be decreased to account for the added resistance to water movement through the crust. Number of entries must equal NPV.
B-18	RDC(2,J)	Pressure potential of the atmosphere (HA) at beginning of ET period; may be estimated using equation 6 of Lappala and others (1987), L. Number of entries must equal NPV.
Lines B-19 to B-23 are present only if ETSIM = T.		
B-19	PTVAL	Potential evapotranspiration rate (PET) at beginning of each ET period, LT^{-1}. Number of entries must equal NPV. As with PEV, all values must be greater than or equal to 0.
B-20	RDC(3,J)	Rooting depth at beginning of each ET period, L. Number of entries must equal NPV.
B-21	RDC(4,J)	Root activity at base of root zone at beginning of each ET period, L^{-2}. Number of entries must equal NPV.
B-22	RDC(5,J)	Root activity at top of root zone at beginning of each ET period, L^{-2}. Number of entries must equal NPV.

Note: Values for root activity generally are determined empirically, but typically range from 0 to 3.0 cm/cm3. As programmed, root activity varies linearly from land surface to the base of the root zone, and its distribution with depth at any time is represented by a trapezoid. In general, root activities will be greater at land surface than at the base of the root zone.

Card	Variable	Description
B-23	RDC(6,J)	Pressure head in roots (HROOT) at beginning of each ET period, L. Number of entries must equal NPV.
Lines B-24 and B-25 are present only if TRANS = T.		
B-24	IREAD	If IREAD = 0, all initial concentrations are set equal to FACTOR. If IREAD = 1, all initial concentrations are read from file IU in user designated format and multiplied by FACTOR.

Table 3.--*Input data formats*--Continued

Card	Variable	Description
B-24--Continued	FACTOR	Multiplier or constant value, depending on value of IREAD, for initial concentrations.
Line B-25 is present only if IREAD = 1.		
B-25	IU	Unit number from which initial concentrations are to be read.
	IFMT	Format to be used in reading initial head values from unit IU. Must be enclosed in quotation marks, for example '(10X, E10.3)'.
[Line group C read by subroutine VSTMER, NRECH sets of C lines are required]		
C-1	TPER	Length of this recharge period, T.
	DELT	Length of initial time step for this period, T.
C-2	TMLT	Multiplier for time step length.
	DLTMX	Maximum allowed length of time step, T.
	DLTMIN	Minimum allowed length of time step, T.
	TRED	Factor by which time-step length is reduced if convergence is not obtained in ITMAX iterations. Values usually should be in the range 0.1 to 0.5. If no reduction of time-step length is desired, input a value of 0.0.
C-3	DSMAX	Maximum allowed change in head per time step for this period, L.
	STERR	Steady-state head criterion; when the maximum change in head between successive time steps is less than STERR, the program assumes that steady state has been reached for this period and advances to next recharge period, L.
C-4	POND	Maximum allowed height of ponded water for constant flux nodes. See Lappala ans others (1987) for detailed discussion of POND, L.
C-5	PRNT	Logical variable = T if heads, concentration, moisture contents, and/or saturations are to be printed to file 6 after each time step; = F if they are to be written to file 6 only at observation times and ends of recharge periods.
C-6	BCIT	Logical variable = T if evaporation is to be simulated for this recharge period; otherwise = F.

Table 3.--*Input data formats*--Continued

Card	Variable	Description
C-6--Continued	ETSIM	Logical variable = T if evapotranspiration (plant-root extraction) is to be simulated for this recharge period; otherwise = F.
	SEEP	Logical variable = T if seepage faces are to be simulated for this recharge period; otherwise = F
C-7 to C-9 cards are present only if SEEP = T,		
C-7	NFCS	Number of possible seepage faces. Must be less than or equal to 4.
Line sets C-8 and C-9 must be reported NFCS times		
C-8	JJ	Number of nodes on the possible seepage face.
	JLAST	Number of the node which initially represents the highest node of the seep; value can range from 0 (bottom of the face) up to JJ (top of the face).
C-9	J,N	Row and column of each cell on possible seepage face, in order from the lowest to the highest elevation; JJ pairs of values are required.
C-10	IBC	Code for reading in boundary conditions by individual node (IBC=0) or by row or column (IBC=1). Only one code may be used for each recharge period, and all boundary conditions for period must be input in the sequence for that code.
Line set C-11 is read only if IBC = 0. One line should be present for each node for which new boundary conditions are specified.		
C-11	JJ	Row number of node.
	NN	Column number of node.
	NTX	Node type identifier for boundary conditions. = 0 for no specified boundary (needed for resetting some nodes after intial recharge period); = 1 for specified pressure head; = 2 for specified flux per unit horizontal surface area in units of LT-1; = 3 for possible seepage face; = 4 for specified total head; = 5 for evaporation; = 6 for specified volumetric flow in units of L^3T^{-1}.

Table 3.--*Input data formats*--Continued

Card	Variable	Description
C-11--Continued	PFDUM	Specified head for NTX = 1 or 4 or specified flux for NTX = 2 or 6. If codes 0, 3, or 5 are specified, the line should contain a dummy value for PFDUM or should be terminated after NTX by a blank and a slash.
	NTC	Node type identifier for transport boundary conditions = 0 for no specified boundary; = 1 for specified concentration, ML^{-3}; = 2 for specified mass flux, MT^{-1}. Present only if TRANS = T.
	CF	Specified concentration for NTC = 1 or NTX = 1,2,4, or 6; or specified flux for NTC = 2. Present only if TRANS = T.
C-12 is present only if IBC = 1. One card should be present for each row or column for which new boundary conditions are specified,		
C-12	JJT	Top node of row or column of nodes sharing same boundary condition.
	JJB	Bottom node of row or column of nodes having same boundary condition. Will equal JJT if a boundary row is being read.
	NNL	Left column in row or column of nodes having same boundary condition.
	NNR	Right column of row or column of nodes having same boundary condition. Will equal NNL if a boundary column is being read in.
	NTX	Same as line C-11.
	PFDUM	Same as line C-11.
	NTC	Same as line C-11.
	CF	Same as line C-11.
C-13	Designated end of recharge period. Must be included after line C-12 data for each recharge period. Two C-13 lines must be included after final recharge period. Line must always be entered as 999999 /.	

Considerations in Discretization

Users need to be aware that selection of spatial grid increments and time step sizes can have a large effect upon calculated results for the advection-dispersion equation. Those readers familiar with the flow portion of VS2D are well aware that fine spatial and temporal discretizations are required to accurately solve variably saturated flow problems involving sharp wetting fronts (such as infiltration to dry soil). For such problems the discretizations are probably adequate for solute-transport simulation. However for other problems, solute transport simulations may require finer discretizations than that required for flow simulations in order to obtain accurate results.

Two common problems are encountered in approximating the advection-dispersion equation by the finite-difference method: numerical dispersion and numerical oscillation. Numerical dispersion arises from the use of backward differencing and is illustrated by the smearing of sharp concentration fronts. Backward-in-space differencing is first-order accurate in terms of Δx, while backward-in-time differencing is first-order accurate in terms of Δt. Kipp (1987) makes the following recommendations to insure that numerical dispersion remains small relative to actual physical dispersion:

$$\frac{\Delta x}{2} << \alpha_L \tag{52}$$

and

$$\frac{|v|\Delta t}{2} << \alpha_L \ . \tag{53}$$

Numerical oscillations arise from the use of central differences. It is illustrated by overshoot and undershoot in the vicinity of sharp concentration fronts. Centered-in-space differencing is second order accurate in Δx and hence introduces no numerical dispersion. Numerical oscillations may occur unless:

$$\frac{|v|\Delta v}{|D_{h_{zz}}|} + \frac{|v|\Delta x}{|D_{h_{xx}}|} \leq 2 \tag{54}$$

This can be a very restrictive requirement. In practice a little more leeway is allowed especially for problems that do not involve sharp concentration fronts. Centered-in-time differencing is second order accurate in Δt. It can also cause oscillations, but criteria for determining a maximum Δt to ensure no oscillations are not as developed as for spatial discretization. In general, the differences between centered and backward time differencing are not as great as the differences encountered in spatial differencing.

Regardless of the discretization methods or refinements that are used, it is strongly recommended that the effects of grid size and time-step size be evaluated for any application of this computer program. This can be done with a simple sensitivity test by refining both the space and time grid. The results obtained with the original and refined grids should be compared and a decision made as to the significance of the differences.

MODEL VERIFICATION AND EXAMPLE PROBLEMS

The transport option of VS2DT was verified on five test problems. Three of the problems have analytical solutions. The other two problems are compared with results of other numerical models. No verification problems involve ion exchange. However the ion-exchange options were all tested with the example problems presented by Grove and Stollenwerk (1984). Results obtained with VS2DT were virtually identical to those of Grove and Stollenwerk (1984).

Verification Problem 1

The first test problem involves fluid injection from a well in a fully saturated confined aquifer. Axial symmetry is assumed and radial coordinates are used in the simulation. The solute concentration within the aquifer is initially 0, while the concentration of the injected water is 1.0. This problem has been simulated previously with the finite-element program SUTRA by Voss (1984). Analytical solutions have been developed by Tang and Babu (1979) and Hsieh (1986). Hoopes and Harleman (1967) and Gelhar and Collins (1971) developed approximate analytical solutions. The analytical solution of Hsieh (1986) has the following form:

$$c(r^*,t^*) = C_0 \left(1+\int_0^\infty F(\nu)d\nu\right) \tag{55}$$

$$F(\nu) = \frac{2\exp[-\nu^2 t^*+(r^*-r_w^*)/2][Ai(y)Bi(y_w)-Ai(y_w)Bi(y)]}{\pi\nu([Ai(y_w)]^2+[Bi(y_w)]^2)} \tag{56}$$

where $r^* = r/\alpha_L$;

r = radial distance from injection well ;

t^* = dimensionless time ;

$= Qt/(2\pi\theta_s b\alpha_L^2)$;

Q = injection rate ;

$= 225\ m^3/h$;

θ_s = moisture content at saturation ;

= 0.20 ;

b = thickness of aquifer ;

= 10 m ;

C_0 = concentration of injected water ;

$r_w^* = r_w/\alpha_L$;

r_w = radius of injection well ;

= 0.05 m ;

Ai = Airy function;

Bi = Airy function ;

$y = \frac{1-4r^*\nu}{4\nu^{4/3}}$; and

$y_w = \frac{1-4r_w^*\nu}{4\nu^{4/3}}$.

The spatial grid consisted of 3 rows and 188 columns. Spacing in the vertical direction was 10 m. Spacing in the radial direction increased from 0.05 m at the injection well by a factor of 1.2 until a maximum size of 5 m was reached. The total length of the grid in the radial direction was 847 m. Initial total head was 10.0 m everywhere in the aquifer. The following constants were used:

$K = .36$ m/h;
$\alpha_L = 10.$ m;
$\alpha_T = 0.$ m.

A pumping period of 2,000 h was simulated. The length of the initial time step was 1×10^{-7} h. The time-step size was increased for each subsequent time step by a factor of 1.5 until the maximum allowed time-step size of 2.0 h was reached. A total of 1,043 time steps were used. Flow boundaries consisted of a constant flux of +225 m^3/h at the injection well and a fixed head of 10.0 m at the radial boundary. Centered-in-time and centered-in-space differencing were selected.

Results of VS2DT and the analytical solution are shown in figure 5 for four times. The match between results is very good at all times.

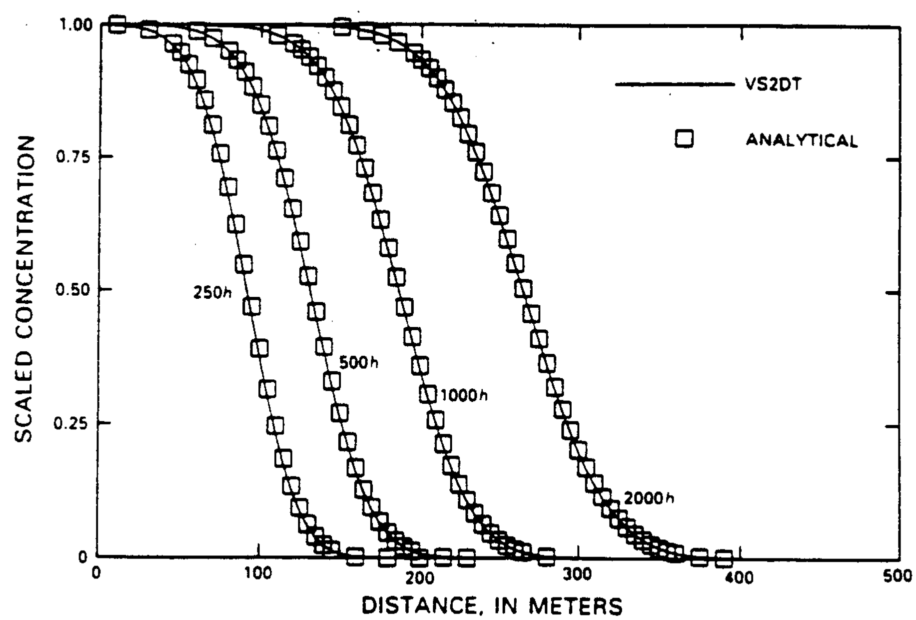


Figure 5.--Graph showing results of first verification problem: Analytical solution of Hsieh (1986) and numerical solution of VS2DT.

<u>Verification Problem 2</u>

In the second test problem, solute transport through a saturated one-dimensional column was simulated for a period of 7,200 s. Initial solute concentration was 0 at all points in the column. A steady-flow field was obtained in the column so that the interstitial velocity was 2.7778×10 m/s.

At time equal 0 the boundary at the top of the vertical column was set to a fixed concentration of 1.0. Ogata and Banks (1961) present an analytical solution to this problem. Kipp (1987) used the program HST to simulate the same problem.

The column was 160 m in length and was represented by 43 nodes. Spacing was set at 0.1 m at the top of the column and allowed to increase by a factor of 1.2 for each subsequent node. The maximum allowed node spacing was 8.0 m. The initial time step length was 1×10^{-7} s. This was increased by a factor of 1.5 for each subsequent step. The maximum allowed time step size was 200 s. A total of 86 time steps was used in the simulation. The following constants were used:

$K = 9.8\times10^{-4}$ m/s;
$\theta_s = 0.50$;
$\alpha_L = 10$ m; and
$D_m = 1\times10^{-10}$ m²/s.

Results are shown in figure 6 at 7,200 s. A good match was obtained between the VS2DT results and the analytical solution.

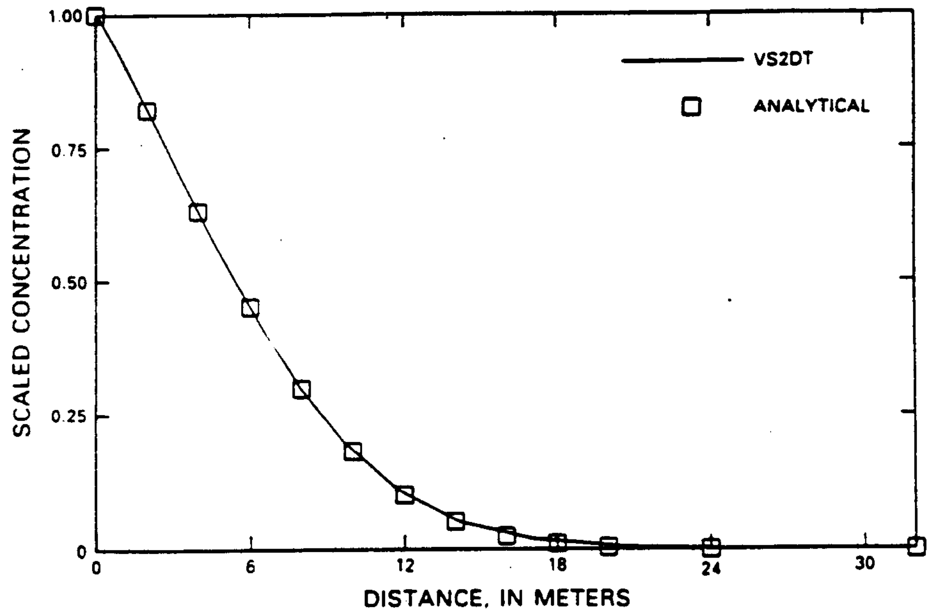


Figure 6.--Graph showing analytical and numerical results of second verification problem at 7,200 seconds.

Verification Problem 3

The third test involves infiltration of water containing a solute into a one-dimensional unsaturated solute-free column of soil. After 2.8 h the infiltrating water is solute free. The problem is based on a field experiment described by Warrick, Biggar, and Nielsen (1971). The problem has been

AR 020555

simulated numerically by van Genuchten (1982), Voss (1984), and Kwicklis (1987). The spatial grid consisted of a column containing 50 rows, each 2.5 cm in height. A period of 9 h was simulated. A constant time step size of .048 h was used. The hydraulic properties of the soil were represented by the following equations:

$$\theta(h) = \begin{cases} 0.6829 - 0.09524 \ln(|h|), & h \leq -29.484 \\ 0.4531 - 0.02732 \ln(|h|), & -29.484 < h \leq -14.495 \end{cases} \quad (57)$$

$$K_r(h) = \begin{cases} 5.1164\times10^4 \; |h|^{-3.4095}, & h \leq -29.484 \\ 13.672 \; |h|^{-.97814}, & -29.484 < h \leq -14.495 \end{cases} \quad (58)$$

where h = pressure head, in centimeters;
K = 37.8 cm/day;
α_L = 1.026 cm; and
D_m = 0.6 cm²/day.

Initial concentrations were 0 everywhere in the column. Initial moisture contents and boundary conditions are:

$$\theta = \begin{cases} .15 + z/1{,}200, & 0 < z \leq 60 \text{ cm} \\ .20, & 60 \text{ cm} < z \end{cases};$$

$$h = -14.495 \qquad z = 0, \qquad t > 0; \text{ and}$$

$$c = \begin{cases} 209 & z = 0, \quad t < .11667 \text{ days} \\ 0 & z = 0, \quad t \geq .11667 \text{ days}. \end{cases}$$

Figures 7 and 8 show VS2DT results using centered-in-time and centered-in-space differencing (CTCS) for 2 and 9 h along with the results of van Genuchten (1982), who assumed that the correct solution was obtained by using a very fine grid. The results are in good agreement at all times, but the simulated concentration peak at 9.0 h lags the true solution slightly. Small oscillations in concentrations at the tail of the plume can be seen at 9 h because of the use of centered-in-space differencing. It is interesting to note that the wetting front is propagated more quickly through the column than is the solute front.

To illustrate the effects of using the various differencing options, the problem was rerun using the following differencing schemes:

1) backward-in-time, centered-in-space (BTCS)
2) backward-in-time, backward-in-space (BTBS)

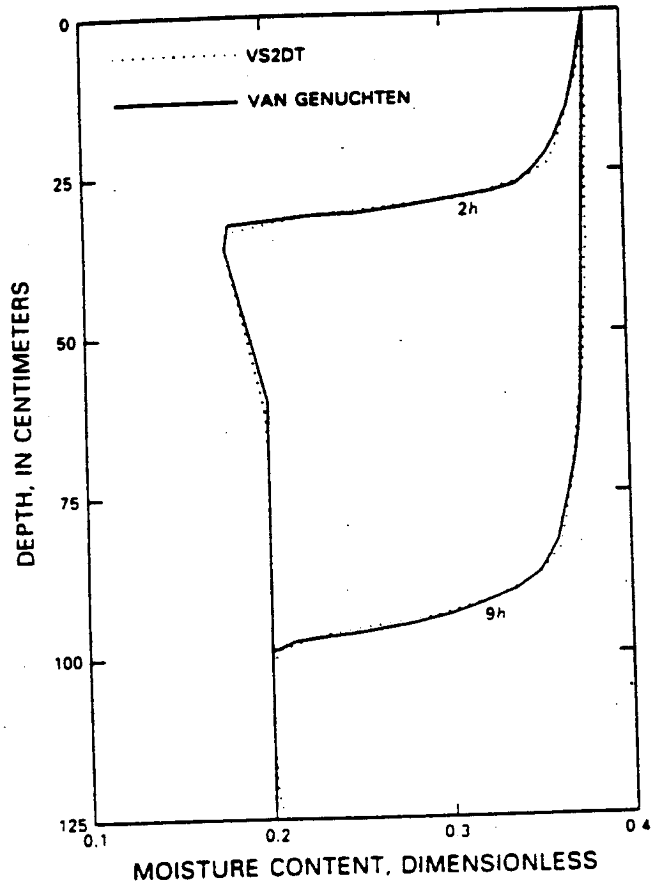


Figure 7.--Graph showing results of third verification problem, moisture content versus depth for VS2DT and van Genuchten (1982).

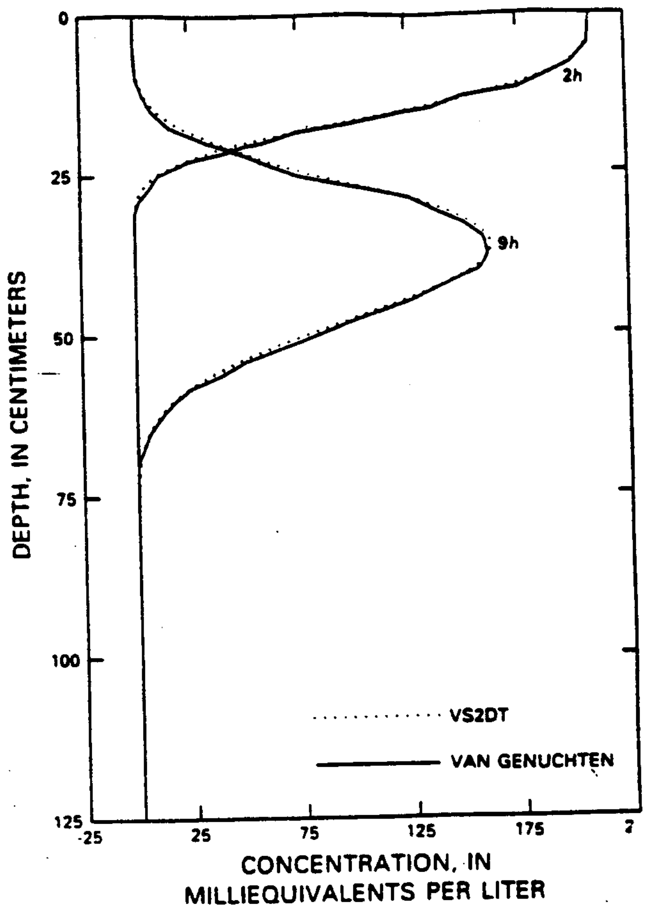


Figure 8.--Graph showing results of third verification problem, concentration versus depth for VS2DT (centered-in-time and centered-in-space differencing) and van Genuchten (1982).

Results for the concentration field are shown in figures 9 and 10. The BTCS simulation (fig. 9) produced concentration profiles similar to those in figure 8, but with slightly more smearing at the leading and trailing edges of the profile. The BTBS simulation (fig. 10) produced quite a different profile, with decreased peak concentrations and drastic smearing on both sides of the peak.

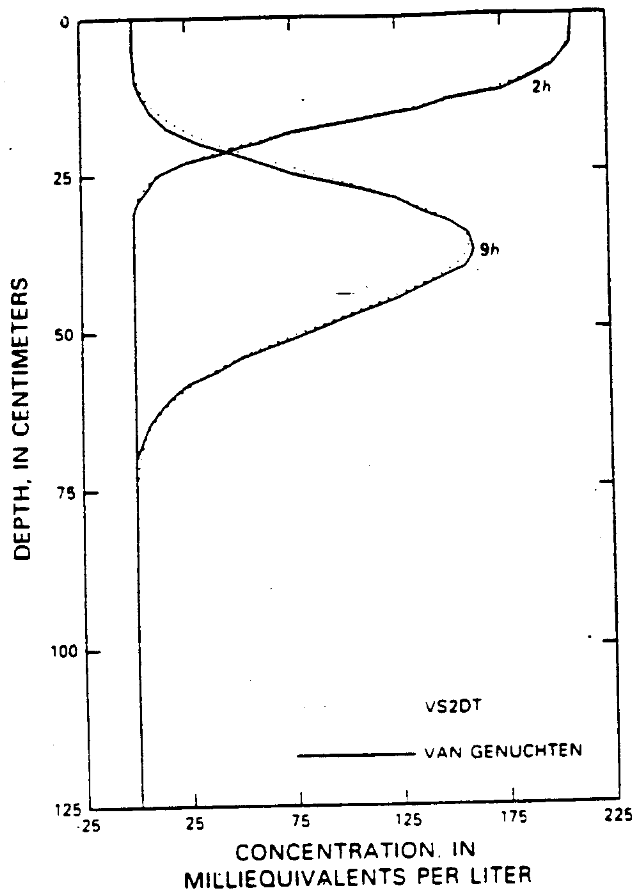


Figure 9.--Graph showing results of third verification problem, concentration versus depth for VS2DT (backward-in-time and centered-in-space differencing) and van Genuchten (1982).

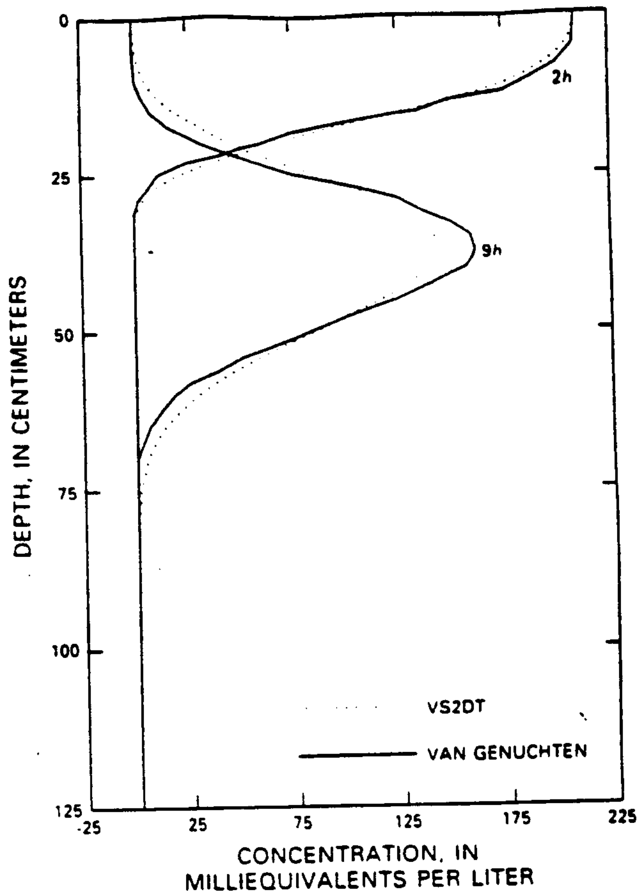


Figure 10.--Graph showing results of third verification problem, concentration versus depth for VS2DT (backward-in-time and backward-in-space differencing) and van Genuchten (1982).

Verification Problem 4

The fourth verification problem involves two-dimensional transport of a nonconservative tracer in a vertical plane. The problem is taken from Huyakorn and others (1985). The vertical section is 10 cm in height and 15 cm in width (fig. 11). Initial pressure head is everywhere -90 cm. The right-hand boundary is maintained at that pressure head. The top and bottom of the section are no-flow boundaries, as is the bottom 6 cm of the left-hand boundary. Water flows into the domain along the top 4 cm of the left-hand side. The pressure heads there are fixed at h = z-4, where z is measured positive downward and both h and z are in cm. Initial solute concentration is 0 in the plane. The inflowing water is given a concentration of 1 mg/L. Grid spacing was uniform with $\Delta x = \Delta z = 1$ cm. The time-step size was initially 0.01 days and was increased by a factor of 1.2 for each subsequent time step, with the maximum allowed time-step size of 0.05 d. The following hydraulic functions and physical properties were used:

$$\theta(h) = .45 + .003h \tag{59}$$

$$K_r(\theta) = 3.33\,\theta - .5 \tag{60}$$

$K = 1$ cm/d;
$\alpha_L = 1$ cm;
$\alpha_T = 0$ cm;
$D_m = 0.01$ cm^2/d;
$\lambda = 0.001$ d^{-1}; and
$\rho_b K_d = \theta$.

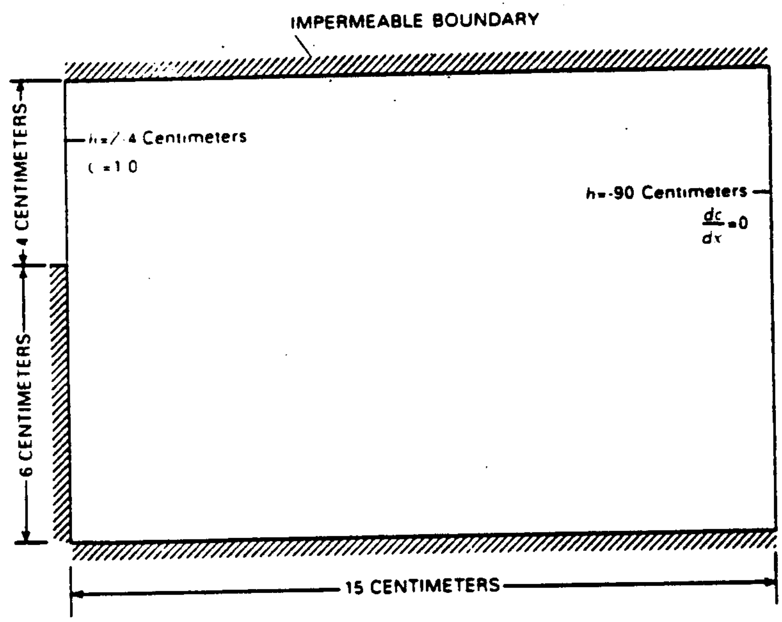


Figure 11.--Sketch showing boundary and initial conditions for verification problem 4.

Results for VS2DT, using backward-in-time and centered-in-space differencing, and the finite-element model of Huyakorn and others (1985) are shown in terms of a horizontal profile (fig. 12) and a vertical profile (fig. 13). In general, the results of the two models are very similar. Because the manner in which nodes are treated is different for finite-element

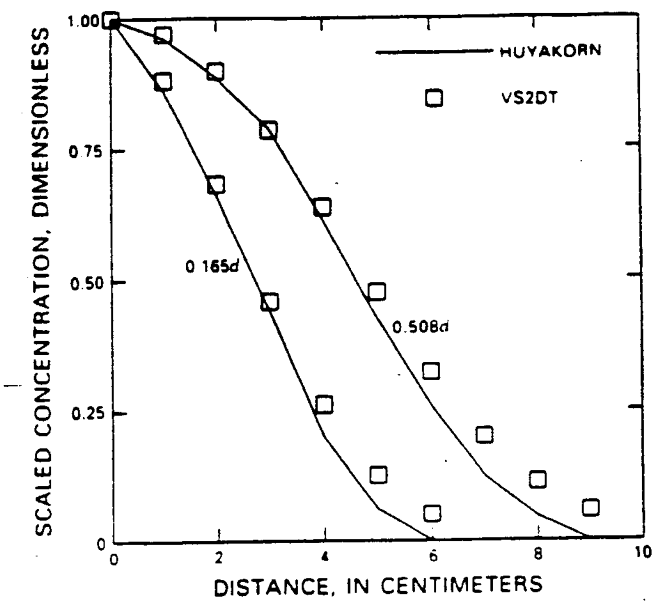


Figure 12.--Graph showing horizontal distribution of solute concentration for verification problem 4 for VS2DT, at a depth of 0.5 centimeter, and Huyakorn and others (1985) at a depth of 0 centimeter.

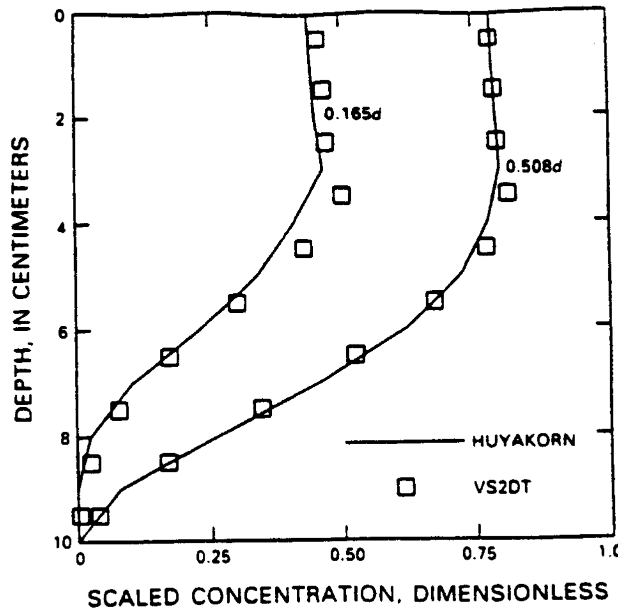


Figure 13.--Graph showing vertical distribution of solute concentration for verification problem 4 at a distance of 3 centimeters from left-hand boundary for VS2DT and Huyakorn and others (1985).

techniques than for block-centered finite-difference techniques, the horizontal profile in figure 12 is for a depth of 0. cm for the Huyakorn and others (1985) results and a depth of 0.5 cm for the VS2DT results. This explains the slightly greater concentrations predicted by VS2DT at larger distances from the left-hand boundary.

Verification Problem 5

The fifth verification problem involves one-dimensional transport through a saturated soil column with steady water flow and both first-order decay and linear sorption. The initial solute concentration in the column is zero. At times greater than zero, the inflowing water has a concentration of c_0. The analytical solution to this problem (assuming a semi-infinite column) is given by Bear (1972, p. 630) as:

$$c(z,t) = \frac{1}{2} c_0 \, e^{(vz/2D)} \{ e^{-z\beta} \, \mathrm{erfc}[\frac{z - (v'^2 + 4\lambda D')^{1/2} t}{2(D't)^{1/2}}] + e^{z\beta} \, \mathrm{erfc}[\frac{z + (v'^2 + 4\lambda D')^{1/2} t}{2(D't)^{1/2}}] \} \qquad (61)$$

where $\beta^2 = (\frac{v}{2D})^2 + \frac{\lambda}{D'}$;

$v' = v/(1+\rho_b K_d/\theta_s)$; and

$D' = \alpha|v|/(1+\rho_b K_d/\theta_s)$.

The following constants were used:

length of column = 35 cm;
v = 0.1 cm/s;
α = 1.0 cm;
D_m = 0;
Δt = 1 s;
ρ_b = 1.587 gm/cm^3;
θ_s = 0.37;
Δz = 0.2 cm;
λ = 0.0, 0.01; and
K_d = 0.0, 0.3.

The water flowing into the column was maintained at a concentration of c_0 for 160 s, after which the concentration was set to zero for an additional 320 s. As shown in figure 14, the numerical results of VS2DT produce a good match with analytical results at a distance of 8 cm for the column inlet at all times for the three different cases.

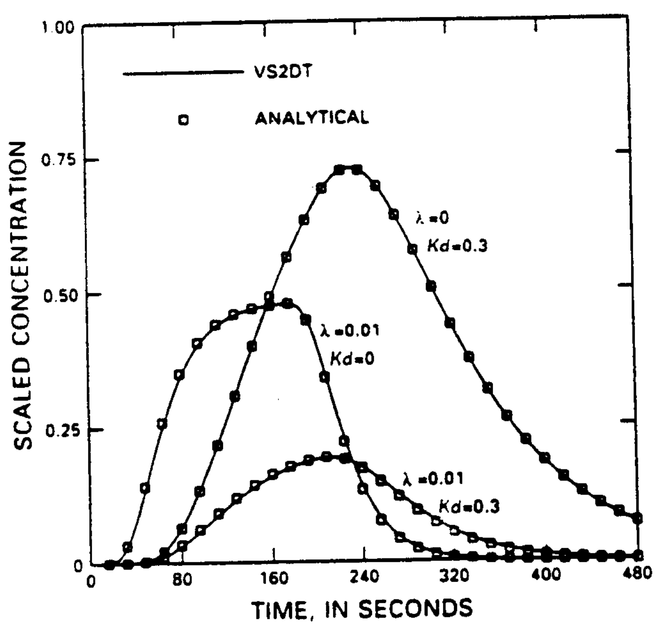


Figure 14.--Graph showing analytical and numerical results at distance of 8 centimeters from column inlet for fifth verification problem.

AR 020561

Example Problem

The purpose of this example is to demonstrate the data-input requirements and output listing for the simulator. The example can also be used as a test of the code after installation on any computer. The problem involves infiltration of water containing a solute concentration of 1.0 into a partially saturated one-dimensional soil column containing solute free water. The soil is a sandy loam with moisture and hydraulic-conductivity curves described by van Genuchten (1980):

$$\theta(h) = \theta_r + (\theta_s - \theta_r)\left[\frac{1}{1 + (\frac{h}{\alpha'})^{\beta'}}\right]^{\gamma}$$

$$K_r(h) = \frac{\{1-(\frac{h}{\alpha'})^{\beta'-1}[1+(\frac{h}{\alpha'})^{\beta'}]^{-\gamma}\}^2}{[1+(\frac{h}{\alpha'})^{\beta'}]^{\gamma/2}}$$

The following constants were used:

θ_s = 0.45;
θ_r = 0.10;
β' = 2.75;
α' = -40 cm;
γ = 0.64;
K = 6.25 cm/h; and
α_L = 10 cm.

The column is 40. cm in height. Uniform spacing (Δz = 1 cm) and time-step sizes (Δt = .005 hr) are used. Initial conditions were h = -120. cm and c = 0. everywhere. A constant flux of 5.5 cm/h was applied to the top of the column for a period of 0.50 h. The infiltrating water had a solute concentration of 1.0. A full listing of input data for the problem is shown in Table 4. Tables 5 and 6 show results printed to Fortran files 6 and 9, respectively.

Table 4.--*Input data for example problem*

EXAMPLE PROBLEM 1-D INFILTRATION	
0.50 0. 0.	A2--IMAX,STIM,ANG
CMHOURGRAM	A3--ZUNIT,TUNIT,CUNX
3 42	A4--NXR,NLY
2 600	A5--NRECH,NUMT
F T T	A6--RAD,ITSTOP,TRANS
T T F	A6A--CIS,CIT,SORP
F T T T F	A7--F11P,F7P,F8P,F9P,F6P
F F T F T	A8--THPT,SPNT,PPNT,HPNT,VPNT
1 1.	A9--IFAC,FACX
1 1.	A11-IFAC,FACZ
1	A13--NPLT
0.50	A14--PLTIM
6	A17--NMB9
31 33 40 42 70 72	A18--MB9
.0005 .90 0.00 0.00001	B1--EPS,HMAX,WUS,EPS1
2 100	B3--MINIT,ITMAX
T	B4--PHRD
1 6 7	B5--NTEX,NPROP,NPROP1
1	B6--ITEX
1. 10.0 0. .45 -40. .10 2.75	B7--ANIZ,HK
10. 0. 0.000001 0. 0. 0. 1.	B7A--HT
1	B8--IROW
1 3 42 1	B10--IL,IR,JBT,JRD
0 -120.	B11--IREAD,FACTOR
F F	B15--NPV,ETCYC
0 0.	B24--IREAD,FACTOR
0.50 .005	C1--TPER,DELT
1.0 0.005 0.005 0.0	C2--TMLT,DLTMX,DLTMIN,TRED
100. 0.	C3--DSMAX,STERR
0.	C4--POND
F	C5--PRNT
F F F	C6--BCIT,ETSIM,SEEP
0	C10--IBC
2 2 2 5.5 0 1.0	C11--JJ,NN,NTX,PFDUM,NTC,CF
999999 /	C13
999999 /	C13

Table 5.--*Output to file 6 for example problem*

```
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
+                          VS2DT                           +
+    SIMULATION OF 2-DIMENSIONAL VARIABLY                  +
+    SATURATED FLOW AND SOLUTE TRANSPORT                   +
+    THROUGH POROUS MEDIA.  VERSION DATED                  +
+                          4-1-90                          +
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++


****************************************************************************************************
    EXAMPLE PROBLEM 1-D INFILTRATION
****************************************************************************************************

         SPACE AND TIME CONSTANTS
         ----------------------
    MAXIMUM SIMULATION TIME =     0.5000 HOUR
    STARTING TIME =      0.0000
    NUMBER OF RECHARGE PERIODS =         2
    MAXIMUM NUMBER OF TIME STEPS =        600
    NUMBER OF ROWS =     42
    NUMBER OF COLUMNS =     3
    AXES TILTED BY ANGLE =     0.00
         SOLUTION OPTIONS
         ---------------
    WRITE ALL PRESSURE HEADS TO FILE 8 AT OBSERVATION TIMES? T
    STOP SOLUTION IF MAXIMUM NO. OF ITERATIONS EXCEEDED IN ANY TIME STEP?,T
    WRITE MAXIMUM CHANGE IN HEAD FOR EACH ITERATION TO FILE 7? T
    WRITE RESULTS AT SELECTED OBSERVATION POINTS TO FILE 11? F
    WRITE MASS BALANCE RATES TO FILE 9? T
    WRITE MASS BALANCE RATES TO FILE 6? F
    WRITE MOISTURE CONTENTS TO FILE 6? F
    WRITE SATURATIONS TO FILE 6? F
    WRITE PRESSURE HEADS TO FILE 6? T
    WRITE TOTAL HEADS TO FILE 6? F
    WRITE VELOCITIES TO FILE 6? T
                                GRID SPACING IN VERTICAL DIRECTION, IN   CM
    1.000     1.000     1.000     1.000     1.000     1.000     1.000     1.000     1.000     1.000
    1.000     1.000     1.000     1.000     1.000     1.000     1.000     1.000     1.000     1.000
    1.000     1.000     1.000     1.000     1.000     1.000     1.000     1.000     1.000     1.000
    1.000     1.000     1.000     1.000     1.000     1.000     1.000     1.000     1.000     1.000
    1.000     1.000
```

Table 5.--*Output to file 6 for example problem*--Continued

```
                                GRID SPACING IN HORIZONTAL OR RADIAL DIRECTION, IN   CM
1.000      1.000      1.000
TIMES AT WHICH H WILL BE WRITTEN TO FILE 08
   0.5000
MASS BALANCE COMPONENTS WRITTEN TO FILE 9
  31  33  40  42  70  72
COORDINATE SYSTEM IS RECTANGULAR
TRANSPORT TO BE SIMULATED
CENTRAL DIFFERENCING IN SPACE USED FOR TRANSPORT EQUATION
CENTRAL DIFFERENCING IN TIME USED FOR TRANSPORT EQUATION
MATRIX EQUATIONS TO BE SOLVED BY SIP
     INITIAL MOISTURE PARAMETERS
     ___________________________

CONVERGENCE CRITERIA FOR SIP FOR FLOW =    5.000E-04   CM
CONVERGENCE CRITERIA FOR SIP FOR TRANSPORT =    1.000E-05
DAMPING FACTOR, HMAX =    9.000E-01
GEOMETRIC MEAN USED FOR INTERCELL CONDUCTIVITY
NUMBER OF SOIL TEXTURAL CLASSES =          1
NUMBER OF SOIL PARAMETERS FOR EACH CLASS =          6
NUMBER OF TRANSPORT PARAMETERS FOR EACH CLASS =          7
MINIMUM PERMITTED NO. OF ITERATIONS/TIME STEP =          2
MAXIMUM PERMITTED NO. OF ITERATIONS/TIME STEP =        100
                             CONSTANTS FOR SOIL TEXTURAL CLASSES

     ANISOTROPY        KSAT       SPECIFIC    POROSITY
                                  STORAGE
       ALPHAL         ALPHAT        DM        LAMBDA    B DENSITY
CLASS # 1
       1.0000D+00   1.0000D+01   0.0000D-01   4.5000D-01  -4.0000D+01   1.0000D-01   2.7500D+00
       1.0000D+01   0.0000D-01   1.0000D-06   0.0000D-01   0.0000D-01   0.0000D-01   1.0000D+00
     TEXTURAL CLASS INDEX MAP


    TEXTURAL CLASSES READ IN BY BLOCK
       1  111
       2  111
       3  111
       4  111
       5  111
       6  111
       7  111
       8  111
```

Table 5.--*Output to file 6 for example problem*--Continued

```
  9  111
 10  111
 11  111
 12  111
 13  111
 14  111
 15  111
 16  111
 17  111
 18  111
 19  111
 20  111
 21  111
 22  111
 23  111
 24  111
 25  111
 26  111
 27  111
 28  111
 29  111
 30  111
 31  111
 32  111
 33  111
 34  111
 35  111
 36  111
 37  111
 38  111
 39  111
 40  111
 41  111
 42  111
INITIAL PRESSURE HEAD OR MOISTURE CONTENT WAS SET TO A CONSTANT VALUE OF  -1.200E+02
INITIAL CONCENTRATION SET TO A CONSTANT VALUE OF    0.000E-01
5SIP ITERATION PARAMETERS:  0.1421085D-13  0.8131982D+00  0.9651051D+00  0.9934816D+00  0.9987823D+00
 EXAMPLE PROBLEM 1-D INFILTRATION
TOTAL ELAPSED TIME =   0.000E-01 HOUR
TIME STEP     0

                                   PRESSURE HEAD
```

Table 5.--*Output to file 6 for example problem*--Continued

```
Z, IN
  CM                    X OR R DISTANCE, IN   CM
             0.50
  0.50-1.20E+02
  1.50-1.20E+02
  2.50-1.20E+02
  3.50-1.20E+02
  4.50-1.20E+02
  5.50-1.20E+02
  6.50-1.20E+02
  7.50-1.20E+02
  8.50-1.20E+02
  9.50-1.20E+02
 10.50-1.20E+02
 11.50-1.20E+02
 12.50-1.20E+02
 13.50-1.20E+02
 14.50-1.20E+02
 15.50-1.20E+02
 16.50-1.20E+02
 17.50-1.20E+02
 18.50-1.20E+02
 19.50-1.20E+02
 20.50-1.20E+02
 21.50-1.20E+02
 22.50-1.20E+02
 23.50-1.20E+02
 24.50-1.20E+02
 25.50-1.20E+02
 26.50-1.20E+02
 27.50-1.20E+02
 28.50-1.20E+02
 29.50-1.20E+02
 30.50-1.20E+02
 31.50-1.20E+02
 32.50-1.20E+02
 33.50-1.20E+02
 34.50-1.20E+02
 35.50-1.20E+02
 36.50-1.20E+02
 37.50-1.20E+02
 38.50-1.20E+02
 39.50-1.20E+02
```

Table 5.--*Output to file 6 for example problem*--Continued

CONCENTRATION

Z, IN CM	X OR R DISTANCE, IN CM 0.50
0.50	0.00E-01
1.50	0.00E-01
2.50	0.00E-01
3.50	0.00E-01
4.50	0.00E-01
5.50	0.00E-01
6.50	0.00E-01
7.50	0.00E-01
8.50	0.00E-01
9.50	0.00E-01
10.50	0.00E-01
11.50	0.00E-01
12.50	0.00E-01
13.50	0.00E-01
14.50	0.00E-01
15.50	0.00E-01
16.50	0.00E-01
17.50	0.00E-01
18.50	0.00E-01
19.50	0.00E-01
20.50	0.00E-01
21.50	0.00E-01
22.50	0.00E-01
23.50	0.00E-01
24.50	0.00E-01
25.50	0.00E-01
26.50	0.00E-01
27.50	0.00E-01
28.50	0.00E-01
29.50	0.00E-01
30.50	0.00E-01
31.50	0.00E-01
32.50	0.00E-01
33.50	0.00E-01
34.50	0.00E-01
35.50	0.00E-01
36.50	0.00E-01
37.50	0.00E-01
38.50	0.00E-01

Table 5.--*Output to file 6 for example problem*--Continued

```
39.50 0.00E-01
 DATA FOR RECHARGE PERIOD     1

     LENGTH OF THIS PERIOD =   5.000E-01 HOUR
     LENGTH OF INITIAL TIME STEP FOR THIS PERIOD = 5.000E-03 HOUR
     MULTIPLIER FOR TIME STEP =  1.000E+00
     MAXIMUM TIME STEP SIZE =  5.000E-03 HOUR
     MINIMUM TIME STEP SIZE =  5.000E-03 HOUR
     TIME STEP REDUCTION FACTOR =  0.000E-01
          MAXIMUM PRESSURE HEAD CHANGE ALLOWED IN ONE TIME STEP = 100.000
          STEADY-STATE CLOSURE CRITERION =  0.000E-01
          MAXIMUM DEPTH OF PONDING =  0.000E-01
          PRINT SOLUTION AFTER EVERY TIME STEP? F
          SIMULATE EVAPORATION? F
          SIMULATE EVAPOTRANSPIRATION? F
          SIMULATE SEEPAGE FACES? F

  NODE TYPE AND INITIAL BOUNDARY CONDITIONS FOR PERIOD    1
  LEGEND:
            0 = INTERIOR CELL
            1 = SPECIFIED PRESSURE HEAD CELL
            2 = SPECIFIED FLUX CELL
            3 = POTENTIAL SEEPAGE FACE NODE
            5 = NODE FOR WHICH EVAPORATION IS PERMITTED


  1      000
  2      020
  3      000
  4      000
  5      000
  6      000
  7      000
  8      000
  9      000
 10      000
 11      000
 12      000
 13      000
 14      000
 15      000
 16      000
 17      000
```

Table 5.--*Output to file 6 for example problem*--Continued

```
  18    000
  19    000
  20    000
  21    000
  22    000
  23    000
  24    000
  25    000
  26    000
  27    000
  28    000
  29    000
  30    000
  31    000
  32    000
  33    000
  34    000
  35    000
  36    000
  37    000
  38    000
  39    000
  40    000
  41    000
  42    000
     MAXIMUM CELL PECLET NUMBER  --  HORIZONTAL     0.00000E+00     ROW    0  COLUMN    0
                                     VERTICAL       0.10000E+00     ROW   41  COLUMN    2
TIME STEP     1  TIME =   0.5000E-02  NIT =  15  NIT1 =   6
     MAXIMUM CELL PECLET NUMBER  --  HORIZONTAL     0.00000E+00     ROW    0  COLUMN    0
                                     VERTICAL       0.10000E+00     ROW   41  COLUMN    2
TIME STEP     2  TIME =   0.1000E-01  NIT =   9  NIT1 =   6
     MAXIMUM CELL PECLET NUMBER  --  HORIZONTAL     0.00000E+00     ROW    0  COLUMN    0
                                     VERTICAL       0.10000E+00     ROW   41  COLUMN    2
TIME STEP     3  TIME =   0.1500E-01  NIT =   9  NIT1 =   6
     MAXIMUM CELL PECLET NUMBER  --  HORIZONTAL     0.00000E+00     ROW    0  COLUMN    0
                                     VERTICAL       0.10000E+00     ROW   41  COLUMN    2
TIME STEP     4  TIME =   0.2000E-01  NIT =   9  NIT1 =   6
     MAXIMUM CELL PECLET NUMBER  --  HORIZONTAL     0.00000E+00     ROW    0  COLUMN    0
                                     VERTICAL       0.10000E+00     ROW   41  COLUMN    2
TIME STEP     5  TIME =   0.2500E-01  NIT =  10  NIT1 =   6
     MAXIMUM CELL PECLET NUMBER  --  HORIZONTAL     0.00000E+00     ROW    0  COLUMN    0
                                     VERTICAL       0.99999E-01     ROW   41  COLUMN    2
TIME STEP     6  TIME =   0.3000E-01  NIT =  10  NIT1 =   6
```

Table 5.--*Output to file 6 for example problem*--Continued

```
    MAXIMUM CELL PECLET NUMBER  --  HORIZONTAL      0.00000E+00     ROW   0  COLUMN    0
                                    VERTICAL        0.99999E-01     ROW  41  COLUMN    2
TIME STEP     7  TIME =   0.3500E-01  NIT =  10  NIT1 =   6
    MAXIMUM CELL PECLET NUMBER  --  HORIZONTAL      0.00000E+00     ROW   0  COLUMN    0
                                    VERTICAL        0.99999E-01     ROW  41  COLUMN    2
TIME STEP     8  TIME =   0.4000E-01  NIT =   9  NIT1 =   6
    MAXIMUM CELL PECLET NUMBER  --  HORIZONTAL      0.00000E+00     ROW   0  COLUMN    0
                                    VERTICAL        0.99999E-01     ROW  41  COLUMN    2
TIME STEP     9  TIME =   0.4500E-01  NIT =   9  NIT1 =   6
    MAXIMUM CELL PECLET NUMBER  --  HORIZONTAL      0.00000E+00     ROW   0  COLUMN    0
                                    VERTICAL        0.99999E-01     ROW  41  COLUMN    2
TIME STEP    10  TIME =   0.5000E-01  NIT =  10  NIT1 =   6
    MAXIMUM CELL PECLET NUMBER  --  HORIZONTAL      0.00000E+00     ROW   0  COLUMN    0
                                    VERTICAL        0.99999E-01     ROW  41  COLUMN    2
              .
              .
              .
TIME STEP    91  TIME =   0.4550E+00  NIT =   8  NIT1 =   5
    MAXIMUM CELL PECLET NUMBER  --  HORIZONTAL      0.00000E+00     ROW   0  COLUMN    0
                                    VERTICAL        0.99999E-01     ROW  41  COLUMN    2
TIME STEP    92  TIME =   0.4600E+00  NIT =   8  NIT1 =   5
    MAXIMUM CELL PECLET NUMBER  --  HORIZONTAL      0.00000E+00     ROW   0  COLUMN    0
                                    VERTICAL        0.99999E-01     ROW  41  COLUMN    2
TIME STEP    93  TIME =   0.4650E+00  NIT =   8  NIT1 =   5
    MAXIMUM CELL PECLET NUMBER  --  HORIZONTAL      0.00000E+00     ROW   0  COLUMN    0
                                    VERTICAL        0.99999E-01     ROW  41  COLUMN    2
TIME STEP    94  TIME =   0.4700E+00  NIT =   8  NIT1 =   5
    MAXIMUM CELL PECLET NUMBER  --  HORIZONTAL      0.00000E+00     ROW   0  COLUMN    0
                                    VERTICAL        0.99999E-01     ROW  41  COLUMN    2
TIME STEP    95  TIME =   0.4750E+00  NIT =   8  NIT1 =   5
    MAXIMUM CELL PECLET NUMBER  --  HORIZONTAL      0.00000E+00     ROW   0  COLUMN    0
                                    VERTICAL        0.99999E-01     ROW  41  COLUMN    2
TIME STEP    96  TIME =   0.4800E+00  NIT =   8  NIT1 =   5
    MAXIMUM CELL PECLET NUMBER  --  HORIZONTAL      0.00000E+00     ROW   0  COLUMN    0
                                    VERTICAL        0.99999E-01     ROW  41  COLUMN    2
TIME STEP    97  TIME =   0.4850E+00  NIT =   8  NIT1 =   5
    MAXIMUM CELL PECLET NUMBER  --  HORIZONTAL      0.00000E+00     ROW   0  COLUMN    0
                                    VERTICAL        0.99999E-01     ROW  41  COLUMN    2
TIME STEP    98  TIME =   0.4900E+00  NIT =   8  NIT1 =   5
    MAXIMUM CELL PECLET NUMBER  --  HORIZONTAL      0.00000E+00     ROW   0  COLUMN    0
                                    VERTICAL        0.99999E-01     ROW  41  COLUMN    2
TIME STEP    99  TIME =   0.4950E+00  NIT =   8  NIT1 =   5
    MAXIMUM CELL PECLET NUMBER  --  HORIZONTAL      0.00000E+00     ROW   0  COLUMN    0
```

Table 5.--*Output to file 6 for example problem*--Continued

```
                                   VERTICAL       0.99999E-01     ROW   41  COLUMN    2
TIME STEP   100  TIME =   0.5000E+00  NIT =   8  NIT1 =   5
     EXAMPLE PROBLEM 1-D INFILTRATION
    TOTAL ELAPSED TIME =   5.000E-01 HOUR
    TIME STEP   100


                                                  PRESSURE HEAD
 Z, IN
   CM                     X OR R DISTANCE, IN   CM
            0.50
   0.50-2.66E+01
   1.50-2.73E+01
   2.50-2.82E+01
   3.50-2.91E+01
   4.50-3.02E+01
   5.50-3.14E+01
   6.50-3.28E+01
   7.50-3.45E+01
   8.50-3.65E+01
   9.50-3.89E+01
  10.50-4.20E+01
  11.50-4.62E+01
  12.50-5.22E+01
  13.50-6.22E+01
  14.50-8.44E+01
  15.50-1.11E+02
  16.50-1.19E+02
  17.50-1.20E+02
  18.50-1.20E+02
  19.50-1.20E+02
  20.50-1.20E+02
  21.50-1.20E+02
  22.50-1.20E+02
  23.50-1.20E+02
  24.50-1.20E+02
  25.50-1.20E+02
  26.50-1.20E+02
  27.50-1.20E+02
  28.50-1.20E+02
  29.50-1.20E+02
  30.50-1.20E+02
  31.50-1.20E+02
```

Table 5.--*Output to file 6 for example problem*--Continued

```
 32.50-1.20E+02
 33.50-1.20E+02
 34.50-1.20E+02
 35.50-1.20E+02
 36.50-1.20E+02
 37.50-1.20E+02
 38.50-1.19E+02
 39.50-1.19E+02
                                        X-VELOCITY
Z, IN
  CM                     X OR R DISTANCE, IN   CM
           0.50
  0.50 0.00E-01
  1.50 0.00E-01
  2.50 0.00E-01
  3.50 0.00E-01
  4.50 0.00E-01
  5.50 0.00E-01
  6.50 0.00E-01
  7.50 0.00E-01
  8.50 0.00E-01
  9.50 0.00E-01
 10.50 0.00E-01
 11.50 0.00E-01
 12.50 0.00E-01
 13.50 0.00E-01
 14.50 0.00E-01
 15.50 0.00E-01
 16.50 0.00E-01
 17.50 0.00E-01
 18.50 0.00E-01
 19.50 0.00E-01
 20.50 0.00E-01
 21.50 0.00E-01
 22.50 0.00E-01
 23.50 0.00E-01
 24.50 0.00E-01
 25.50 0.00E-01
 26.50 0.00E-01
 27.50 0.00E-01
 28.50 0.00E-01
 29.50 0.00E-01
 30.50 0.00E-01
```

Table 5.--*Output to file 6 for example problem*--Continued

31.50	0.00E-01
32.50	0.00E-01
33.50	0.00E-01
34.50	0.00E-01
35.50	0.00E-01
36.50	0.00E-01
37.50	0.00E-01
38.50	0.00E-01
39.50	0.00E-01

Z-VELOCITY

Z, IN CM	X OR R DISTANCE, IN CM
	0.50
0.50	0.00E-01
1.50	1.39E+01
2.50	1.39E+01
3.50	1.38E+01
4.50	1.37E+01
5.50	1.36E+01
6.50	1.35E+01
7.50	1.33E+01
8.50	1.31E+01
9.50	1.28E+01
10.50	1.24E+01
11.50	1.19E+01
12.50	1.10E+01
13.50	9.66E+00
14.50	6.89E+00
15.50	1.96E+00
16.50	2.43E-01
17.50	4.11E-02
18.50	2.39E-02
19.50	2.25E-02
20.50	2.24E-02
21.50	2.24E-02
22.50	2.24E-02
23.50	2.24E-02
24.50	2.24E-02
25.50	2.24E-02
26.50	2.24E-02
27.50	2.24E-02
28.50	2.24E-02
29.50	2.24E-02

Table 5.--*Output to file 6 for example problem*--Continued

30.50	2.24E-02
31.50	2.24E-02
32.50	2.24E-02
33.50	2.23E-02
34.50	2.22E-02
35.50	2.18E-02
36.50	2.09E-02
37.50	1.87E-02
38.50	1.45E-02
39.50	8.11E-03

CONCENTRATION

X OR R DISTANCE, IN CM

Z, IN CM	0.50
0.50	7.31E-01
1.50	7.05E-01
2.50	6.77E-01
3.50	6.49E-01
4.50	6.20E-01
5.50	5.91E-01
6.50	5.61E-01
7.50	5.30E-01
8.50	4.98E-01
9.50	4.65E-01
10.50	4.31E-01
11.50	3.94E-01
12.50	3.54E-01
13.50	3.08E-01
14.50	2.41E-01
15.50	6.43E-02
16.50	1.39E-03
17.50	4.18E-06
18.50	7.15E-09
19.50	1.15E-11
20.50	1.83E-14
21.50	2.90E-17
22.50	4.59E-20
23.50	7.24E-23
24.50	1.14E-25
25.50	1.79E-28
26.50	2.81E-31
27.50	4.39E-34
28.50	6.84E-37

Table 5.--*Output to file 6 for example problem*--Continued

```
29.50 1.06E-39
30.50 1.65E-42
31.50 2.55E-45
32.50 3.92E-48
33.50 6.01E-51
34.50 9.14E-54
35.50 1.36E-56
36.50 1.94E-59
37.50 2.47E-62
38.50 2.45E-65
39.50 1.36E-68
          ---------- MASS BALANCE SUMMARY FOR TIME STEP 100 ----------
              PUMPING PERIOD NUMBER     1
              TOTAL ELAPSED SIMULATION TIME =  5.000E-01 HOUR

+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
+                                                                                                                             +
+                                                                                    TOTAL THIS          RATE THIS            +
+                       VOLUMETRIC FLOW BALANCE                           TOTAL      TIME STEP           TIME STEP            +
+                                                                         CM**3        CM**3             CM**3/HOUR           +
+  FLUX INTO DOMAIN ACROSS SPECIFIED PRESSURE HEAD BOUNDARIES --     0.00000E-01     0.00000E-01       0.00000E-01          +
+ FLUX OUT OF DOMAIN ACROSS SPECIFIED PRESSURE HEAD BOUNDARIES --    0.00000E-01     0.00000E-01       0.00000E-01          +
+          FLUX INTO DOMAIN ACROSS SPECIFIED FLUX BOUNDARIES --      2.75000E+00     2.75000E-02       5.50000E+00          +
+        FLUX OUT OF DOMAIN ACROSS SPECIFIED FLUX BOUNDARIES --      0.00000E-01     0.00000E-01       0.00000E-01          +
+                                     TOTAL FLUX INTO DOMAIN --      2.75000E+00     2.75000E-02       5.50000E+00          +
+                                   TOTAL FLUX OUT OF DOMAIN --      0.00000E-01     0.00000E-01       0.00000E-01          +
+                                                EVAPORATION --      0.00000E-01     0.00000E-01       0.00000E-01          +
+                                              TRANSPIRATION --      0.00000E-01     0.00000E-01       0.00000E-01          +
+                                   TOTAL EVAPOTRANSPIRATION --      0.00000E-01     0.00000E-01       0.00000E-01          +
+                           CHANGE IN FLUID STORED IN DOMAIN --      2.75000E+00     2.75000E-02       5.49999E+00          +
+                                      FLUID VOLUME BALANCE --       3.10821E-06     4.34478E-08       8.68957E-06          +
+                                                                                                                             +
+                                                                                                                             +
+                       SOLUTE MASS BALANCE                                                                                   +
+                                                                         GRAM           GRAM             GRAM/HOUR           +
+  FLUX INTO DOMAIN ACROSS SPECIFIED PRESSURE HEAD BOUNDARIES --     0.00000E-01     0.00000E-01       0.00000E-01          +
+ FLUX OUT OF DOMAIN ACROSS SPECIFIED PRESSURE HEAD BOUNDARIES --    0.00000E-01     0.00000E-01       0.00000E-01          +
+          FLUX INTO DOMAIN ACROSS SPECIFIED FLUX BOUNDARIES --      2.75000E+00     2.75000E-02       5.50000E+00          +
+        FLUX OUT OF DOMAIN ACROSS SPECIFIED FLUX BOUNDARIES --      0.00000E-01     0.00000E-01       0.00000E-01          +
+                       DIFFUSIVE/DISPERSIVE FLUX INTO DOMAIN --     0.00000E-01     0.00000E-01       0.00000E-01          +
+                     DIFFUSIVE/DISPERSIVE FLUX OUT OF DOMAIN --     0.00000E-01     0.00000E-01       0.00000E-01          +
+                                     TOTAL FLUX INTO DOMAIN --      2.75000E+00     2.75000E-02       5.50000E+00          +
+                                   TOTAL FLUX OUT OF DOMAIN --      0.00000E-01     0.00000E-01       0.00000E-01          +
```

Table 5.--*Output to file 6 for example problem*--Continued

```
+                    TOTAL EVAPOTRANSPIRATION --   0.00000E-01    0.00000E-01    0.00000E-01   +
+                          FIRST ORDER DECAY --   0.00000E-01    0.00000E-01    0.00000E-01   +
+                   ADSORPTION/ION EXCHANGE --   0.00000E-01    0.00000E-01    0.00000E-01   +
+          CHANGE IN SOLUTE STORED IN DOMAIN --   2.71061E+00    2.73234E-02    5.46469E+00   +
+                        SOLUTE MASS BALANCE --   3.93940E-02    1.76551E-04    3.53103E-02   +
+                                                                                               +
+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
   ***********************************************************************************
   END OF SIMULATION
   ***********************************************************************************
   TOTAL NUMBER OF ITERATIONS FOR FLOW EQUATION =    842
   TOTAL NUMBER OF ITERATIONS FOR TRANSPORT EQUATION =    543
```

Table 6.--*Output to file 9 for example problem*

5.000E-03	-3.103E-07	-6.205E-05	2.750E-02	5.500E+00	3.428E-03	6.857E-01
1.000E-02	-7.551E-07	-8.897E-05	5.500E-02	5.500E+00	5.719E-03	4.581E-01
1.500E-02	-9.124E-07	-3.145E-05	8.250E-02	5.500E+00	7.266E-03	3.094E-01
2.000E-02	-9.552E-07	-8.564E-06	1.100E-01	5.500E+00	8.465E-03	2.399E-01
2.500E-02	-8.418E-07	2.267E-05	1.375E-01	5.500E+00	9.596E-03	2.262E-01
3.000E-02	-6.997E-07	2.843E-05	1.650E-01	5.500E+00	1.062E-02	2.055E-01
3.500E-02	-6.579E-07	8.364E-06	1.925E-01	5.500E+00	1.147E-02	1.686E-01
4.000E-02	-6.893E-07	-6.282E-06	2.200E-01	5.500E+00	1.219E-02	1.451E-01
4.500E-02	-7.285E-07	-7.832E-06	2.475E-01	5.500E+00	1.291E-02	1.443E-01
5.000E-02	-7.079E-07	4.119E-06	2.750E-01	5.500E+00	1.365E-02	1.477E-01
5.500E-02	-5.442E-07	3.272E-05	3.025E-01	5.500E+00	1.434E-02	1.381E-01
6.000E-02	-5.051E-07	7.821E-06	3.300E-01	5.500E+00	1.494E-02	1.192E-01
6.500E-02	-5.111E-07	-1.191E-06	3.575E-01	5.500E+00	1.547E-02	1.059E-01
7.000E-02	-5.488E-07	-7.545E-06	3.850E-01	5.500E+00	1.599E-02	1.053E-01
7.500E-02	-5.332E-07	3.116E-06	4.125E-01	5.500E+00	1.655E-02	1.107E-01
8.000E-02	-4.952E-07	7.600E-06	4.400E-01	5.500E+00	1.710E-02	1.108E-01
8.500E-02	-4.168E-07	1.569E-05	4.675E-01	5.500E+00	1.761E-02	1.021E-01
9.000E-02	-4.026E-07	2.832E-06	4.950E-01	5.500E+00	1.807E-02	9.058E-02
9.500E-02	-4.041E-07	-3.022E-07	5.225E-01	5.500E+00	1.849E-02	8.437E-02
1.000E-01	-4.217E-07	-3.517E-06	5.500E-01	5.500E+00	1.892E-02	8.574E-02
	⋮					
4.050E-01	2.384E-06	8.108E-06	2.227E+00	5.500E+00	3.581E-02	4.190E-02
4.100E-01	2.436E-06	1.029E-05	2.255E+00	5.500E+00	3.601E-02	4.112E-02
4.150E-01	2.481E-06	9.026E-06	2.282E+00	5.500E+00	3.621E-02	3.947E-02
4.200E-01	2.527E-06	9.248E-06	2.310E+00	5.500E+00	3.640E-02	3.812E-02
4.250E-01	2.560E-06	6.593E-06	2.337E+00	5.500E+00	3.659E-02	3.729E-02
4.300E-01	2.593E-06	6.615E-06	2.365E+00	5.500E+00	3.678E-02	3.765E-02
4.350E-01	2.620E-06	5.368E-06	2.392E+00	5.500E+00	3.697E-02	3.846E-02
4.400E-01	2.656E-06	7.109E-06	2.420E+00	5.500E+00	3.717E-02	3.951E-02
4.450E-01	2.692E-06	7.311E-06	2.447E+00	5.500E+00	3.737E-02	3.971E-02
4.500E-01	2.739E-06	9.387E-06	2.475E+00	5.500E+00	3.756E-02	3.922E-02
4.550E-01	2.781E-06	8.478E-06	2.502E+00	5.500E+00	3.775E-02	3.782E-02
4.600E-01	2.826E-06	8.928E-06	2.530E+00	5.500E+00	3.793E-02	3.655E-02
4.650E-01	2.859E-06	6.594E-06	2.557E+00	5.500E+00	3.811E-02	3.559E-02
4.700E-01	2.892E-06	6.543E-06	2.585E+00	5.500E+00	3.829E-02	3.571E-02
4.750E-01	2.917E-06	5.130E-06	2.612E+00	5.500E+00	3.847E-02	3.631E-02
4.800E-01	2.950E-06	6.471E-06	2.640E+00	5.500E+00	3.866E-02	3.734E-02
4.850E-01	2.982E-06	6.529E-06	2.667E+00	5.500E+00	3.885E-02	3.774E-02
4.900E-01	3.025E-06	8.512E-06	2.695E+00	5.500E+00	3.904E-02	3.758E-02
4.950E-01	3.065E-06	7.964E-06	2.722E+00	5.500E+00	3.922E-02	3.648E-02
5.000E-01	3.108E-06	8.690E-06	2.750E+00	5.500E+00	3.939E-02	3.531E-02

SUMMARY

A computer program, VS2DT, has been developed and tested for simulating solute transport in variably saturated porous media. The program is an extension to the U.S. Geological Survey's computer program VS2D for simulating water movement through variably saturated porous media. The finite-difference method is used to solve the advection-dispersion equation. The user may select either backward or centered approximations for time and space derivatives. The program also allows the following processes to be simulated: first-order decay of the solute, equilibrium adsorption of solute to the solid phase (as described by Freundlich or Langmuir isotherms), and ion exchange. The ability of the program to accurately match analytical results and results of other simulations is demonstrated with five verification problems.

The computer program is written in standard FORTRAN77 and is modular in structure. It can easily be modified or customized for particular applications. Modifications to the original version of VS2D are described as Supplemental Information. A complete listing of VS2DT is given, as well as data input requirements and listings of input and output for an example problem.

REFERENCES

Bear, Jacob, 1972, Dynamics of fluids in porous media: New York, American Elsevier Publishing Co., 764 p.

Bear, Jacob, 1979, Hydraulics of groundwater: New York, McGraw-Hill, 569 p.

Chiou, C.T., Porter, P.E., and Schmedding, D.W., 1983, Partition equilibria of nonionic organic compounds between soil organic matter and water: Environmental Science and Technology, v. 16, p. 4-10.

Corwin, T., and Hansch, C., eds., 1979, Substituents contents for correlation and level analysis in chemistry and biology: New York, John Wiley and Sons.

Gelhar, L. W., and Collins, M.A., 1971, General analysis of longitudinal dispersion in nonuniform flow: Water Resources Research, v. 7, no. 6, p. 1511-1521.

Freeze, R.A., and Cherry, J.A., 1979, Groundwater: Englewood Cliffs, N.J., Prentice-Hall, Inc., 604 p.

Grove, D.B., and Stollenwerk, K.G., 1984, Computer model of one-dimensional equilibrium controlled sorption processes: U.S. Geological Survey Water-Resources Investigations Report 84-4059, 58 p.

Healy, R.W., 1987, Simulation of trickle irrigation, an extension to the U.S. Geological Survey's computer program VS2D: U.S. Geological Survey Water Resources Investigations Report 87-4086, 61 p.

Hillel, Daniel, 1980, Fundamentals of soil physics: New York, Academic Press, 413 p.

Hoopes, J.A., and Harleman, D.R., 1967, Dispersion in radial flow from a recharge well: Journal of Geophysical Research, v. 72, no. 14, p. 3595-3607.

Hsieh, P.A., 1986, A new formula for the analytical solution of the radial dispersion problem: Water Resources Research, v. 22, no. 11, p. 1,597-1,605.

Huyakorn, P.S., Mercer, J.W., and Ward, D.S., 1985, Finite element matrix and mass balance computational schemes for transport in variably saturated porous media: Water Resources Research, v. 21, no. 3, p. 346-358.

Jury, W.A., Spencer, W.F., and Farmer, W.J., 1983, Behavior assessment model for trace organics in soils: I. Model description: Soil Science Society of America Journal, v. 12, p. 558-564.

Karickhoff, S.W., 1981, Semi-empirical estimation of sorption of hydrophobic pollutants on natural sediments: Chemisphere, v. 10, p. 833-846.

Kipp, K.L., 1987, HST3D: A computer code for simulation of heat and solute transport in three-dimensional ground-water flow systems: U.S. Geological Survey Water-Resources Investigations Report 86-4095, 517 p.

Konikow, L.F., and Grove, D.B., 1977, Derivation of equations describing solute transport in ground water: U.S. Geological Survey Water-Resources Investigations Report 77-19, 30 p.

Kwicklis, E.M., 1987, Development of the numerical simulator VSFAST (variably saturated flow and solute transport): A computer code to solve the governing equations in partially or wholly saturated media: Boulder, University of Colorado Master's Thesis, 310 p.

Lappala, E.G., Healy, R.W., and Weeks, E.P., 1987, Documentation of computer program VS2D to solve the equations of fluid flow in variably saturated porous media: U.S. Geological Survey Water-Resources Investigations Report 83-4099, 184 p.

Ogata, Akio, and Banks, R.B., 1961, A solution of the differential equation of longitudinal dispersion in porous media: U.S. Geological Survey Professional Paper 411-A, 7 p.

Rao, P.S.C., and Davidson, J.M., 1980, Estimation of pesticide retention and transformation parameters required in nonpoint source pollution models, in, Overcash, M.R., ed., Environmental impact of nonpoint source pollution: Ann Arbor, Michigan, Ann Arbor Science Publishing, Inc., p. 23-67.

Scheidegger, A.E., 1961, General theory of dispersion in porous media: Journal of Geophysical Research, v. 66, no. 10, p. 3273-3278.

Stumm, Werner, and Morgan, J.J., 1981, Aquatic chemistry: New York, John Wiley and Sons, 780 p.

Tang, D.H., and Babu, D.K., 1979, Analytical solution of a velocity dependent dispersion problem: Water Resources Research, v. 15, no. 6, p. 1,471-1,478.

van Genuchten, M.Th., 1980, A closed-form equation for predicting the hydraulic conductivity of unsaturated soils: Soil Science Society of America Proceedings, v. 44, no. 5, p. 892-898.

van Genuchten, M.Th., 1982, A comparison of numerical solutions of the one-dimensional unsaturated-saturated flow and mass transport equations: Advances in Water Resources, v. 5, no. 1, p. 47-55.

Voss, C.I., 1984, A finite-element simulation model for saturated-unsaturated, fluid-density-dependent ground-water flow with energy transport or chemically reactive single-species solute transport: U.S. Geological Survey Water-Resources Investigations Report 84-4369, 409 p.

Warrick, A.W., Biggar, J.W., and Nielsen, D.R., 1971, Simultaneous solute and water transfer for an unsaturated soil: Water Resources Research, v. 7, no. 5, p. 1216-1225.

SUPPLEMENTAL INFORMATION

Three items are presented in this section. The first is a description of recent modifications to VS2D other than those related to the solute transport option. The second item is a complete listing of the revised version of VS2DT. The final item is a flow chart for VS2DT.

Modifications to Computer Program VS2D

In an effort to improve the efficiency and usefulness of computer program VS2DT, several minor modifications have been incorporated into the original version of the code as listed in Lappala and others (1987). These are detailed below.

(1) The x and z axes may now be tilted for a simulation. This option requires input of the angle of rotation (ANG on card A-2), which is referenced from horizontal. ANG = 0 corresponds to no tilting. Figure 15 illustrates how the finite-difference grid is treated in the program for different rotation angles. ANG must be between -90 and +90 degrees. Because elevation is an important factor in the infiltration/ponding and seepage face boundaries, incorporation of the tilted-axes option required that the subroutines VSPOND and VSFAC be rewritten. The new versions are contained in the following program listing. The algorithms used in these subroutines are still identical to those described in Lappala and others (1987). Because cross-derivative terms are not included in the finite-difference approximation to the flow equation, it is necessary that the principal directions of the hydraulic-conductivity tensor be aligned with the coordinate axes. Therefore, the value for HK(ITEX,1), on input line B-7, must correspond to the saturated hydraulic conductivity in the direction of the tilted x axis. Similarly, the value for ANIZ(ITEX), on the same input line, must represent the ratio of hydraulic conductivity in the direction of the tilted z axis to that in the direction of the tilted x axis.

(2) Selection of mass-balance components for output for file 9 is now a user option. There are 72 components that can be selected. These are listed in table 7, along with the index number that must be included on input card A-18. A maximum of 24 components may be selected for any simulation. The output format for each component is E11.4. The first item in each output line is simulated time. Mass balance information is written to file 9 at the end of every time step. It is anticipated that file 9 results will be used primarily for generating computer plots, therefore no column headings are included in the file.

(3) Fluid mass balance is now given in terms of volume rather than mass. Therefore the variable RHOZ is no longer used in the program and input card B-2 must not appear in the input data stream.

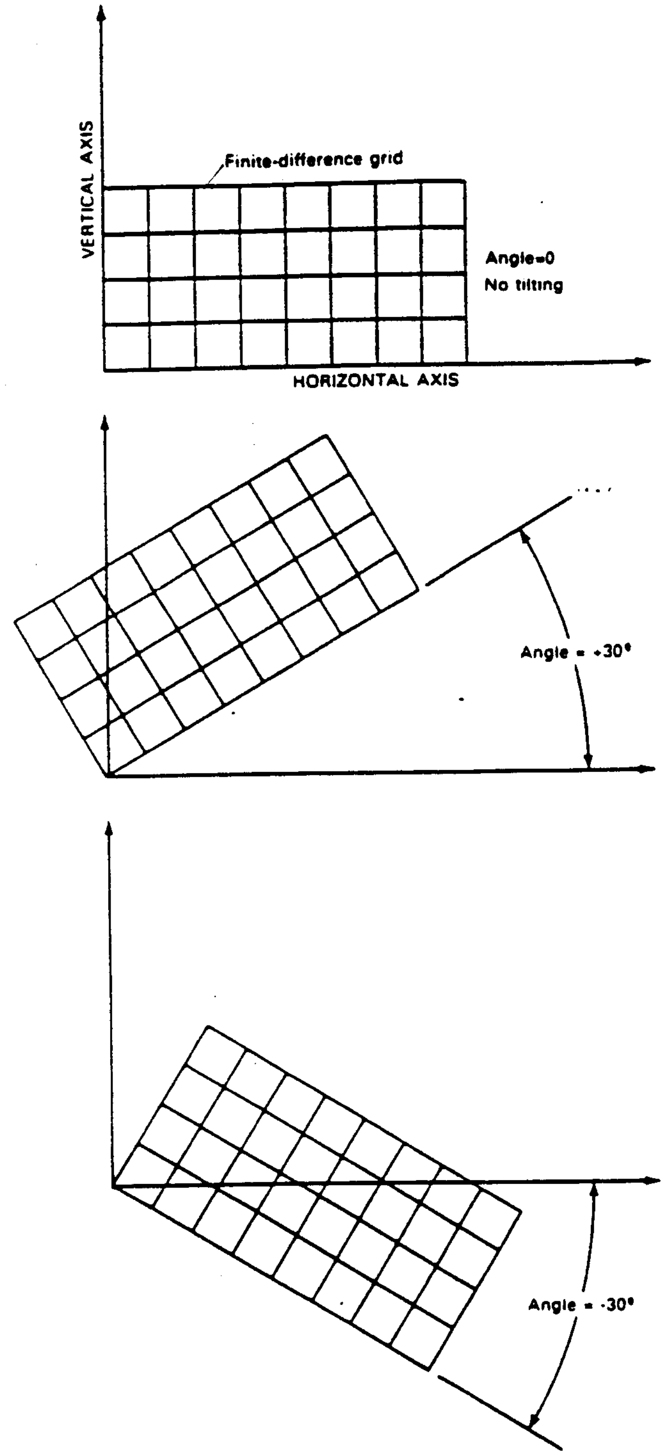


Figure 15.--Sketch showing tilting of finite-difference grid for different angles.

AR 020582

Table 7.--*Index of Mass-Balance Components for Output to File 9*

Index Number	Component	
1	Flow in across specified head boundaries	- total for simulation
2	Flow in across specified head boundaries	- total for time step
3	Flow in across specified head boundaries	- rate for time step
4	Flow out across specified head boundaries	- total for simulation
5	Flow out across specified head boundaries	- total for time step
6	Flow out across specified head boundaries	- rate for time step
7	Flow in across specified flux boundaries	- total for simulation
8	Flow in across specified flux boundaries	- total for time step
9	Flow in across specified flux boundaries	- rate for time step
10	Flow out across specified flux boundaries	- total for simulation
11	Flow out across specified flux boundaries	- total for time step
12	Flow out across specified flux boundaries	- rate for time step
13	Total flow in	- total for simulation
14	Total flow in	- total for time step
15	Total flow in	- rate for time step
16	Total flow out	- total for simulation
17	Total flow out	- total for time step
18	Total flow out	- rate for time step
19	Evaporation	- total for simulation
20	Evaporation	- total for time step
21	Evaporation	- rate for time step
22	Transpiration	- total for simulation
23	Transpiration	- total for time step
24	Transpiration	- rate for time step
25	Evaporation + Transpiration	- total for simulation
26	Evaporation + Transpiration	- total for time step
27	Evaporation + Transpiration	- rate for time step
28	Change in fluid stored in domain	- total for simulation
29	Change in fluid stored in domain	- total for time step
30	Change in fluid stored in domain	- rate for time step
31	Fluid volumetric balance	- total for simulation
32	Fluid volumetric balance	- total for time step
33	Fluid volumetric balance	- rate for time step
34	Solute flux in across specified pressure head boundaries	- total for simulation
35	Solute flux in across specified pressure head boundaries	- total for time step
36	Solute flux in across specified pressure head boundaries	- rate for time step
37	Solute flux out across specified pressure head boundaries	- total for simulation
38	Solute flux out across specified pressure head boundaries	- total for time step
39	Solute flux out across specified pressure head boundaries	- rate for time step
40	Solute flux in across specified flux boundaries	- total for simulation
41	Solute flux in across specified flux boundaries	- total for time step
42	Solute flux in across specified flux boundaries	- rate for time step
43	Solute flux out across specified flux boundaries	- total for simulation
44	Solute flux out across specified flux boundaries	- total for time step
45	Solute flux out across specified flux boundaries	- rate for time step
46	Diffusive/Dispersive flux in across specified flux boundaries	- total for simulation
47	Diffusive/Dispersive flux in across specified flux boundaries	- total for time step
48	Diffusive/Dispersive flux in across specified flux boundaries	- rate for time step
49	Diffusive/Dispersive flux out across specified flux boundaries	- total for simulation
50	Diffusive/Dispersive flux out across specified flux boundaries	- total for time step
51	Diffusive/Dispersive flux out across specified flux boundaries	- rate for time step

Table 7.--*Index of Mass Balance Components for Output to File 9*--Continued

Index Number	Component	
52	Total solute flux in	- total for simulation
53	Total solute flux in	- total for time step
54	Total solute flux in	- rate for time step
55	Total solute flux out	- total for simulation
56	Total solute flux out	- total for time step
57	Total solute flux out	- rate for time step
58	Solute flux out through evapotranspiration	- total for simulation
59	Solute flux out through evapotranspiration	- total for time step
60	Solute flux out through evapotranspiration	- rate for time step
61	First order decay of solute	- total for time step
62	First order decay of solute	- total for time step
63	First order decay of solute	- rate for time step
64	Adsorption or ion exchange of solute	- total for simulation
65	Adsorption or ion exchange of solute	- total for time step
66	Adsorption or ion exchange of solute	- rate for time step
67	Change in solute stored in domain	- total for simulation
68	Change in solute stored in domain	- total for time step
69	Change in solute stored in domain	- rate for time step
70	Solute mass balance	- total for simulation
71	Solute mass balance	- total for time step
72	Solute mass balance	- rate for time step.

```
      SUBROUTINE VSEXEC
C
C******
CVSEXEC
C******
C --------------------------------------------------------------------
C          ************ PROGRAM VS2D ********************
C
C   PROGRAM TO SOLVE FOR:
C      TWO DIMENSIONAL VERTICAL SECTION OR CYLINDRICAL THREE
C          DIMENSIONAL FLUID FLOW AND SOLUTE TRANSPORT UNDER
C          VARIABLY SATURATED CONDITIONS
C
C       FLUID FLOW IS SOLVED FOR BY AN IMPLICIT FINITE DIFFERENCE
C          FORMULATION OF THE COMBINED RICHARDS AND COOPER-JACOB
C          EQUATIONS FOR FLUID CONTINUITY.
C
C  -------      VERSION AS OF APRIL 1, 1990       ----------
C .....................................................................
C
C          DEFINITION OF FUNCTIONAL RELATIONSHIPS REQUIRED
C  VSHKU = RELATIVE HYDRAULIC CONDUCTIVITY AS A FUNCTION OF
C             PRESSURE HEAD
C  VSTHU = VOLUMETRIC MOISTURE CONTENT AS A FUNCTION OF PRESSURE HEAD
C  VSDTHU = FIRST DERIVATIVE OF MOISTURE CONTENT WITH RESPECT
C              TO PRESSURE HEAD
C  VSTHNV = PRESSURE HEAD AS A FUNCTION OF VOLUMETRIC MOISTURE
C              CONTENT
C  VSRDF = ROOT ACTIVITY AS A FUNCTION OF TIME AND DEPTH
C  VTRET = BULK DENSITY TIMES SLOPE OF ADSORPTION ISOTHERM.
C
C --------------------------------------------------------------------
C
C      SPECIFICATIONS FOR ARRAYS AND SCALARS
C
      IMPLICIT DOUBLE PRECISION (A-H,P-Z)
      COMMON/RSPAC/DELZ(600),DZZ(600),DXR(600),RX(600),PI2
      COMMON/ISPAC/NLY,NLYY,NXR,NXRR,NNODES
      COMMON/KCON/HX(1600),NTYP(1600)
      COMMON/RPROP/HK(10,100),HT(10,20),ANIZ(10)
      COMMON/MPROP/THETA(1600),THLST(1600)
      COMMON/PRESS/P(1600),PXXX(1600),CS1,CS2
      COMMON/DISCH/Q(1600),QQ(1600),ETOUT,ETOUT1
      COMMON/HCON/HCND(1600),HKLL(1600),HKTT(1600)
      COMMON/EQUAT/A(1600),B(1600),C(1600),D(1600),E(1600),RHS(1600),
     &XI(1600)
      COMMON/JTXX/JTEX(1600)
      COMMON/DUMM/DUM(1600)
      COMMON/SPFC/JSPX(3,25,8),NFC(8),JLAST(8),NFCS
      COMMON/PTET/DPTH(1600),RT(1600),RDC(6,25),ETCYC,
     &PEVAL(25),PTVAL(25),PET,PEV,HROOT,HA,SRES,RTDPTH,
     &RTBOT,RTTOP,NPV
      COMMON/PND/POND
      COMMON/PLOTT/PLTIM(50),IJOBS(50),JPLT,NPLT,NOBS
      COMMON/WGT/WUS,WDS
      COMMON/SCON/DHMX(201),DELT,HMAX,TMAX,EPS,NUMT,ITMAX,MINIT,ITEST
      COMMON/SCN1/TMPX,TMLT,DLTMX,DLTMIN,TRED
      COMMON/TCON/STIM,DSMAX,KTIM,NIT,NIT1,KP
      COMMON/JCON/JSTOP,JFLAG
      COMMON/TRXX/DX1(1600),DX2(1600),DZ1(1600),DZ2(1600),VX(1600),
     &VZ(1600),CC(1600),COLD(1600),CS(1600),QT(1600),NCTYP(1600),
     &RET(1600)
      COMMON/TRXY1/A0(1600),B0(1600),C0(1600),D0(1600),E0(1600)
      LOGICAL TRANS,TRANS1,SORP,SSTATE
```

```
      COMMON/TRXY/MB9(72),NMB9,EPS1,TRANS,TRANS1,SORP,SSTATE
      LOGICAL RAD,BCIT,ETSIM,SEEP,ITSTOP,CIS,CIT
      LOGICAL F7P,F11P,F8P,F9P,F6P,PRNT
      LOGICAL THPT,SPNT,PPNT,HPNT,VPNT
      COMMON/LOG1/RAD,BCIT,ETSIM,SEEP,ITSTOP,CIS,CIT
      COMMON/LOG2/F7P,F11P,F8P,F9P,F6P,PRNT
      COMMON/LOG4/THPT,SPNT,PPNT,HPNT,VPNT
      CHARACTER*80 TITL
      CHARACTER*4 ZUNIT,TUNIT,CUNX
      COMMON/SCHAR/TITL,ZUNIT,TUNIT,CUNX
      SAVE IFET,IFET1,NITT,NITT1
      DIMENSION KDUM(50,2)
C
C-------------------------------------------------------------------
C
C  ---- READ AND WRITE PROBLEM TITLE AND SPACE AND TIME CONSTANTS
C
      READ (05,4000) TITL
      READ (5,*) TMAX,STIM,ANG
      READ (05,4010) ZUNIT,TUNIT,CUNX
      READ (05,*) NXR,NLY
      READ (05,*) NRECH,NUMT
      WRITE (06,4060)
      WRITE (06,4070) TITL,TMAX,TUNIT,STIM,NRECH,NUMT,NLY,NXR
      WRITE(06,4080) ANG
      IF(ANG.GT.90..OR.ANG.LT.-90.)THEN
      WRITE(06,4090)
      STOP
      END IF
      READ (05,*) RAD,ITSTOP,TRANS
      IF(TRANS) READ(05,*)CIS,CIT,SORP
      READ (05,*) F11P,F7P,F8P,F9P,F6P
      READ (05,*) THPT,SPNT,PPNT,HPNT,VPNT
      WRITE (06,4100) F8P,ITSTOP,F7P,F11P,F9P,F6P
      WRITE (06,4110) THPT,SPNT,PPNT,HPNT,VPNT
      NLYY=NLY-1
      NXRR=NXR-1
      NNODES=NLY*NXR
C
C   IF NUMBER OF NODES IS GREATER THAN ARRAY DIMENSIONS THEN
C   TERMINATE SIMULATION
C
      IF(NNODES.GT.1600.OR.NXR.GT.600.OR.NLY.GT.600) THEN
      WRITE (06,4020) NLY,NXR
      STOP
      END IF
C
C   ESTABLISH HORIZONTAL OR RADIAL SPACING
C
      READ (05,*) IFAC,FACX
      IF(IFAC.GT.0) GO TO 20
C
C   READ IN SPACING FOR EACH COLUMN
C
      READ (05,*) (DXR(K),K=1,NXR)
      DO 10 K=1,NXR
   10 DXR(K)=DXR(K)*FACX
      GO TO 60
   20 IF(IFAC.EQ.2) GO TO 40
      DO 30 K=1,NXR
   30 DXR(K)=FACX
      GO TO 60
C
C  IF IFAC=2, HORIZONTAL NODE SPACING IS INCREMENTED BY A CONSTANT
C    MULTIPLIER UNTIL A USER-SPECIFIED MAXIMUM IS REACHED, WHERE-
C    UPON THE SPACING BECOMES CONSTANT
C
```

```
   40 READ (05,*) XMULT,XMAX
      DXR(1)=FACX
      DXR(2)=FACX
      DO 50 K=3,NXRR
      DXR(K)=DXR(K-1)*XMULT
      IF(DXR(K) .GT. XMAX)DXR(K)=XMAX
   50 CONTINUE
      DXR(NXR)=DXR(NXRR)
C
C  ESTABLISH VERTICAL SPACING
C
   60 READ (05,*) JFAC,FACZ
      IF(JFAC.GT.0) GO TO 80
C
C  READ IN VERTICAL SPACINGS INDIVIDUALLY
C
      READ (05,*) (DELZ(K),K=1,NLY)
      DO 70 K=1,NLY
   70 DELZ(K)=DELZ(K)*FACZ
      GO TO 120
   80 IF(JFAC.EQ.2) GO TO 100
      DO 90 K=1,NLY
   90 DELZ(K)=FACZ
      GO TO 120
C
C  ESTABLISH VERTICAL SPACING BY PROGRESSION, AS ABOVE FOR HORIZ.
C
  100 READ (05,*) ZMULT,ZMAX
      DELZ(1)=FACZ
      DELZ(2)=FACZ
      DO 110 K=3,NLYY
      DELZ(K)=DELZ(K-1)*ZMULT
      IF(DELZ(K) .GT. ZMAX)DELZ(K)=ZMAX
  110 CONTINUE
      DELZ(NLY)=DELZ(NLYY)
  120 CONTINUE
C
C     DETERMINE HORIZONTAL AND VERTICAL COORDINATES
C
      RX(1)=-0.5 *DXR(1)
      DO 130 N=2,NXR
      RX(N)=RX(N-1)+0.5 *(DXR(N-1)+DXR(N))
  130 CONTINUE
      DZZ(1)=-0.5 *DELZ(1)
      DO 140 J=2,NLY
  140 DZZ(J)=DZZ(J-1)+0.5 *(DELZ(J-1)+DELZ(J))
      WRITE (06,4120) ZUNIT,(DELZ(K),K=1,NLY)
      WRITE (06,4130) ZUNIT,(DXR(K),K=1,NXR)
      PI=3.141592654
      PI2=PI+PI
      ANG=ANG/360.
      IF(ANG.EQ.0) THEN
      CS1=1
      CS2=0.
      ELSE
      IF(ANG.EQ.0.25.OR.ANG.EQ.-0.25) THEN
      CS1=0.
      ELSE
      CS1=DCOS(ANG*PI2)
      END IF
      CS2=-DSIN(ANG*PI2)
      END IF
C
C     READ DATA FOR MONITORING TIMES AND POINTS
C
      NPLT=0
      IF(F8P) THEN
```

```
     &35X,1H+,4X,35HSATURATED FLOW AND SOLUTE TRANSPORT,19X,1H+
     &/35X,1H+,4X,41HTHROUGH POROUS MEDIA.  VERSION DATED     ,13X,1H+
     &/35X,1H+,26X,'4-1-90',26X,1H+
     & /35X,60(1H+)//)
 4070 FORMAT(//,1X,100(1H*)/5X,A80/1X,100(1H*)//10X,
     &24HSPACE AND TIME CONSTANTS/10X,23(1H-)/
     & 5X,26HMAXIMUM SIMULATION TIME = ,F10.4,1X,A4/
     &5X,'STARTING TIME = ',F10.4,/
     &5X,28HNUMBER OF RECHARGE PERIODS =,I10/
     &4X,32H MAXIMUM NUMBER OF TIME STEPS = ,I10/
     &5X,17HNUMBER OF ROWS = ,I5/5X,20HNUMBER OF COLUMNS = ,I5)
 4080 FORMAT(5X,'AXES TILTED BY ANGLE = ',F8.2)
 4090 FORMAT(1X,'ANGLE OF AXES TILTING MUST BE BETWEEN -90 AND 90 ',
     &'DEGREES',/,1X,'SIMULATION TERMINATED')
 4100 FORMAT(10X,16HSOLUTION OPTIONS/10X,16(1H-)/
     &5X,'WRITE ALL PRESSURE HEADS TO FILE 8',
     &23H AT OBSERVATION TIMES? ,L1,/
     &5X,28HSTOP SOLUTION IF MAXIMUM NO.,
     &42H OF ITERATIONS EXCEEDED IN ANY TIME STEP?,L1/5X,
     &'WRITE MAXIMUM CHANGE IN HEAD FOR EACH ITERATION TO FILE 7? ',
     &L1/5X,'WRITE RESULTS AT SELECTED OBSERVATION POINTS TO ',
     &9HFILE 11? , L1/,5X,36HWRITE MASS BALANCE RATES TO FILE 9? L1/
     &5X,36HWRITE MASS BALANCE RATES TO FILE 6? ,L1)
 4110 FORMAT(1H ,4X,35HWRITE MOISTURE CONTENTS TO FILE 6? ,L1/
     & 5X,29HWRITE SATURATIONS TO FILE 6? ,L1/
     & 5X,32HWRITE PRESSURE HEADS TO FILE 6? ,L1/
     & 5X,29HWRITE TOTAL HEADS TO FILE 6? ,L1/
     &5X,'WRITE VELOCITIES TO FILE 6? ',L1)
 4120 FORMAT(50X,39HGRID SPACING IN VERTICAL DIRECTION, IN ,A4/
     & (10(F10.3)))
 4130 FORMAT(50X,47HGRID SPACING IN HORIZONTAL OR RADIAL DIRECTION,
     &,3H IN,1X,A4/(10F10.3))
 4140 FORMAT(5X,43HTIMES AT WHICH H WILL BE WRITTEN TO FILE 08
     &/(5X,10F10.4))
 4150 FORMAT(5X,37HROW AND COLUMN OF OBSERVATION POINTS:/
     & 3X,10(2X,2I4))
 4160 FORMAT(5X,'MASS BALANCE COMPONENTS WRITTEN TO FILE 9',
     &/,5X,24I4)
 4170 FORMAT(5X,36HMATRIX EQUATIONS TO BE SOLVED BY SIP)
 4180 FORMAT(5X,100(1H*)/5X,17HEND OF SIMULATION/
     &    5X,100(1H*))
 4190 FORMAT(5X,'TOTAL NUMBER OF ITERATIONS FOR FLOW EQUATION = ',I6
     &/5X,'TOTAL NUMBER OF ITERATIONS FOR TRANSPORT EQUATION = ',I6)
 4200 FORMAT(5X,'CENTRAL DIFFERENCING IN SPACE USED FOR TRANSPORT',
     &' EQUATION')
 4210 FORMAT(4X,' BACKWARD DIFFERENCING IN SPACE USED FOR TRANSPORT',
     &' EQUATION')
 4220 FORMAT(4X,' CENTRAL DIFFERENCING IN TIME USED FOR TRANSPORT',
     &' EQUATION')
 4230 FORMAT(4X,' BACKWARD DIFFERENCING IN TIME USED FOR TRANSPORT',
     &' EQUATION')
 4240 FORMAT(4X,' TRANSPORT TO BE SIMULATED')
 4250 FORMAT(4X,' NONLINEAR SORPTION TO BE SIMULATED')
 4260 FORMAT(5X,'-- WARNING --  INFILTRATION/PONDING BOUNDARY WAS NOT'
     &' SOLVED ACCURATELY FOR THIS TIME STEP')
      END
      BLOCK DATA  DAT1
      IMPLICIT DOUBLE PRECISION (A-H,P-Z)
      COMMON/PRESS/P(1600),PXXX(1600),CS1,CS2
      COMMON/KCON/HX(1600),NTYP(1600)
      COMMON/MPROP/THETA(1600),THLST(1600)
      COMMON/HCON/HCND(1600),HKLL(1600),HKTT(1600)
      COMMON/PTET/DPTH(1600),RT(1600),RDC(6,25),ETCYC,
     &PEVAL(25),PTVAL(25),PET,PEV,HROOT,HA,SRES,RTDPTH,
     &RTBOT,RTTOP,NPV
      COMMON/DISCH/Q(1600),QQ(1600),ETOUT,ETOUT1
      COMMON/EQUAT/A(1600),B(1600),C(1600),D(1600),E(1600),RHS(1600),
```

```
      &XI(1600)
       COMMON/TRXX/DX1(1600),DX2(1600),DZ1(1600),DZ2(1600),VX(1600),
      &VZ(1600),CC(1600),COLD(1600),CS(1600),QT(1600),NCTYP(1600),
      &RET(1600)
       DATA P/1600*0.0/,PXXX/1600*0.0/,HX/1600*0.0/,THETA/1600*0.0/,
      &THLST/1600*0.0/
       DATA HCND/1600*0.0/,HKLL/1600*0.0/,HKTT/1600*0.0/,DPTH/1600*0.0/,
      &RT/1600*0.0/,PTVAL/25*0.0/,PEVAL/25*0.0/
       DATA Q/1600*0.0/,QQ/1600*0.0/
       DATA A/1600*0.0/,B/1600*0.0/,C/1600*0.0/,D/1600*0.0/,
      &E/1600*0.0/,RHS/1600*0.0/,XI/1600*0.0/
       DATA DX1/1600*0.0/,DX2/1600*0.0/,DZ1/1600*0.0/,DZ2/1600*0.0/,
      &VX/1600*0.0/,VZ/1600*0.0/,CC/1600*0.0/,COLD/1600*0.0/,
      &CS/1600*0.0/,QT/1600*0.0/,RET/1600*0.0/
       END
       SUBROUTINE VSREAD
C******
CVSREAD
C******
C
C   PURPOSE: TO READ INITIAL HEAD AND SATURATION DATA
C
C-----------------------------------------------------------------
C
C   SPECIFICATIONS FOR ARRAYS AND SCALARS
C
       IMPLICIT DOUBLE PRECISION (A-H,P-Z)
       COMMON/RSPAC/DELZ(600),DZZ(600),DXR(600),RX(600),PI2
       COMMON/ISPAC/NLY,NLYY,NXR,NXRR,NNODES
       COMMON/KCON/HX(1600),NTYP(1600)
       COMMON/RPROP/HK(10,100),HT(10,20),ANIZ(10)
       COMMON/MPROP/THETA(1600),THLST(1600)
       COMMON/PRESS/P(1600),PXXX(1600),CS1,CS2
       COMMON/DISCH/Q(1600),QQ(1600),ETOUT,ETOUT1
       COMMON/HCON/HCND(1600),HKLL(1600),HKTT(1600)
       COMMON/JTXX/JTEX(1600)
       COMMON/DUMM/DUM(1600)
       COMMON/PTET/DPTH(1600),RT(1600),RDC(6,25),ETCYC,
      &PEVAL(25),PTVAL(25),PET,PEV,HROOT,HA,SRES,RTDPTH,
      &RTBOT,RTTOP,NPV
       COMMON/WGT/WUS,WDS
       COMMON/SCON/DHMX(201),DELT,HMAX,TMAX,EPS,NUMT,ITMAX,MINIT,ITEST
       COMMON/SCN1/TMPX,TMLT,DLTMX,DLTMIN,TRED
       COMMON/TCON/STIM,DSMAX,KTIM,NIT,NIT1,KP
       COMMON/TRXX/DX1(1600),DX2(1600),DZ1(1600),DZ2(1600),VX(1600),
      &VZ(1600),CC(1600),COLD(1600),CS(1600),QT(1600),NCTYP(1600),
      &RET(1600)
       LOGICAL TRANS,TRANS1,SORP,SSTATE
       COMMON/TRXY/MB9(72),NMB9,EPS1,TRANS,TRANS1,SORP,SSTATE
       LOGICAL PHRD
       LOGICAL RAD,BCIT,ETSIM,SEEP,ITSTOP,CIS,CIT
       COMMON/LOG1/RAD,BCIT,ETSIM,SEEP,ITSTOP,CIS,CIT
       CHARACTER*80 TITL
       CHARACTER*36 IFMT
       CHARACTER*4 ZUNIT,TUNIT,CUNX
       COMMON/SCHAR/TITL,ZUNIT,TUNIT,CUNX
       DIMENSION IDUM(0600)
C-----------------------------------------------------------------------
C
C   READ AND WRITE INITIAL DATA FOR SIMULATION
C
       IF (TRANS) THEN
       READ (5,*) EPS,HMAX,WUS,EPS1
       ELSE
       READ(5,*) EPS,HMAX,WUS
       EPS1=0.0
       END IF
```

```
      READ (5,*) MINIT,ITMAX
      READ (05,*) PHRD
      IF(TRANS) THEN
      READ (05,*) NTEX,NPROP,NPROP1
      ELSE
      READ (05,*) NTEX,NPROP
      NPROP1=0
      END IF
C
C    CHECK THAT SUM OF WEIGHTING FACTORS IS EQUAL TO ONE
C
      WRITE (6,4000) EPS,ZUNIT,EPS1,HMAX
      IF(WUS.EQ.1) THEN
      WDS=0.
      WRITE(06,4020)
      ELSE
      IF(WUS.EQ.0.5) THEN
      WDS=0.5
      WRITE(06,4070)
      ELSE
      WUS=0.0
      WRITE(06,4010)
      END IF
      END IF
      WRITE (6,4080) NTEX,NPROP,NPROP1,MINIT,ITMAX
      IF(ITMAX.GT.200) GO TO 210
      WRITE (06,4100)
      IF (TRANS) WRITE(06,4110)
C
C    READ AND WRITE MATERIAL PROPERTIES FOR EACH TEXTURAL CLASS
C
      DO 20J22=1,10
      DO 10J23=1,100
   10 HK(J22,J23)=0.
      DO 20J23=1,20
   20 HT(J22,J23)=0.
      DO 30J22=1,NTEX
      READ (5,*) J
      READ (5,*) ANIZ(J),(HK(J,I),I=1,NPROP)
      WRITE (6,4120) J,ANIZ(J),(HK(J,I),I=1,NPROP)
      IF(TRANS) THEN
      READ(5,*) (HT(J,I),I=1,NPROP1)
      WRITE(6,4130) (HT(J,I),I=1,NPROP1)
      END IF
   30 CONTINUE
      WRITE (06,4140)
C
C     READ TEXTURAL CLASS INDEX MAP
C
      READ (05,*) IROW
      IF(IROW.EQ.0) THEN
      WRITE(06,4090)
      DO 50 J=1,NLY
      READ (05,*) (IDUM(N),N=1,NXR)
      WRITE (06,4150) J,(IDUM(N),N=1,NXR)
      DO 40 N=1,NXR
      IN=NLY*(N-1)+J
      J22=IDUM(N)
      HX(IN)=HK(J22,1)
   40 JTEX(IN)=J22
   50 CONTINUE
      ELSE
C
C     READ TEXTURE CLASSES BY BLOCK--EITHER CONTINUOUS LAYERS OR
C     LAYERS BOUNDED BY VERTICAL DISCONTINUITIES.
C
      WRITE (06,4040)
```

```
      JTP=1
   60 READ (05,*) IL,IR,JBT,JRD
      DO 70 N=IL,IR
      IDUM(N)=JRD
   70 CONTINUE
      IF(IR.LT.NXR) GO TO 60
      DO 80 J=JTP,JBT
   80 WRITE (06,4150) J,(IDUM(N),N=1,NXR)
      DO 90 J=JTP,JBT
      DO 90 N=1,NXR
      IN=NLY*(N-1)+J
      J22=IDUM(N)
      HX(IN)=HK(J22,1)
      JTEX(IN)=J22
   90 CONTINUE
      IF(JBT.EQ.NLY) GO TO 100
      JTP=JBT+1
      GO TO 60
      END IF
  100 CONTINUE
C
C   BORDERS OF DOMAIN ARE ALL SET TO NO FLOW BOUNDARIES
C
      DO 110 I=1,NLY
      I1=NNODES-I+1
      HX(I)=0
  110 HX(I1)=0
      DO 120 I=2,NXR
      I1=(I-1)*NLY
      HX(I1)=0
  120 HX(I1+1)=0
C
C   READ INITIAL HEADS OR MOISTURE CONTENTS
C
      READ (05,*) IREAD,FACTOR
      IF(IREAD.EQ.2) THEN
      READ (05,*) DWTX,HMIN
      WRITE (06,4190) DWTX,ZUNIT,HMIN,ZUNIT,DWTX,ZUNIT
C
C   CALCULATE EQUILIBRIUM INITIAL HEAD PROFILE
C
      DO 130 J=2,NLYY
      DO 130 N=2,NXRR
      IN=NLY*(N-1)+J
      IF(HX(IN).EQ.0.) GO TO 130
      IF(CS1.EQ.1.) THEN
      Z1=DZZ(J)
      ELSE
      Z1=DZZ(J)*CS1+(RX(N))*CS2
      END IF
      P1=Z1-DWTX
      IF(P1.LT.HMIN)P1=HMIN
      P(IN)=P1-Z1
      PXXX(IN)=P(IN)
  130 CONTINUE
      ELSE
      IF(IREAD.NE.1) THEN
      WRITE (6,4170) FACTOR
      ELSE
      READ (05,*) IU,IFMT
      WRITE (06,4180) IU,FACTOR
      END IF
      DO 160 J=1,NLY
      IF(IREAD.NE.0) THEN
C
C   READ INITIAL CONDITIONS FROM FILE IU
C
```

```
      READ (IU,FMT=IFMT) (DUM(N),N=1,NXR)
      ELSE
      DO 140 N=1,NXR
  140 DUM(N)=FACTOR
      END IF
      DO 150 N=1,NXR
      IN=NLY*(N-1)+J
      IF(IREAD.EQ.1)DUM(N)=DUM(N)*FACTOR
      IF(CS1.EQ.1.) THEN
      Z1=DZZ(J)
      ELSE
      Z1=DZZ(J)*CS1+(RX(N))*CS2
      END IF
      IF(.NOT.PHRD) THEN
      IF(DUM(N).LE.0.) THEN
      WRITE(6,4230) J,N
      STOP
      END IF
C
C   CONVERT INITIAL MOISTURE CONTENTS TO HEADS
C
      P(IN)=VSTHNV(DUM(N),JTEX(IN),HK)-Z1
      THETA(IN)=DUM(N)
      ELSE
      P(IN)=DUM(N)-Z1
      END IF
      PXXX(IN)=P(IN)
  150 CONTINUE
  160 CONTINUE
C
C   COMPUTE INITIAL NONLINEAR COEFFICIENT VALUES
C
      END IF
      CALL VSCOEF
C
C   IF ET IS TO BE SIMULATED, ALL VARIABLES MUST BE ENTERED HERE.
C
      READ(05,*) BCIT,ETSIM
      IF(BCIT .OR. ETSIM) THEN
C
C    COMPUTE DEPTHS FOR ET CALCULATIONS
C
      DPTH(1)=-.5 *DELZ(1)
      DO 170 J=2,NLYY
      DO 170 N=2,NXRR
      IN=NLY*(N-1)+J
      JM1=IN-1
      IF(HX(IN).NE.0.) THEN
      IF(HX(JM1).EQ.0.) THEN
      DPTH(IN)=0.0
      ELSE
      DPTH(IN)=DPTH(JM1)+DELZ(J-1)
      END IF
      END IF
  170 CONTINUE
      WRITE (6,4200)
      CALL VSOUT(2,DPTH)
C
C   READ EVAPORATION VARIABLES
C
      READ(05,*)NPV,ETCYC
      WRITE(6,4030) NPV,ETCYC,TUNIT
      IF(BCIT) THEN
      READ (05,*)(PEVAL(I),I=1,NPV)
      READ(05,*)  (RDC(1,I),I=1,NPV)
      READ(05,*)  (RDC(2,I),I=1,NPV)
      WRITE (06,4050)ZUNIT,TUNIT,ZUNIT,ZUNIT,(I,PEVAL(I),RDC(1,I),RDC(2,
```

```
     *I),I=1,NPV)
      END IF
      IF (ETSIM )THEN
C
C   READ TRANSPIRATION VARIABLES
C
      READ(05,*)(PTVAL(I),I=1,NPV)
      READ(05,*) (RDC(3,I),I=1,NPV)
      READ(05,*) (RDC(4,I),I=1,NPV)
      READ(05,*) (RDC(5,I),I=1,NPV)
      READ(05,*) (RDC(6,I),I=1,NPV)
      WRITE(06,4060)ZUNIT,TUNIT,ZUNIT,ZUNIT,ZUNIT,ZUNIT,(I,PTVAL(I),
     *(RDC(J,I),J=3,6),I=1,NPV)
      END IF
      END IF
      DO 180 IN=1,NNODES
      NTYP(IN)=0
      NCTYP(IN)=0
      IF(HX(IN).GT.0) THLST(IN)=THETA(IN)
  180 CONTINUE
C
C   READ INITIAL CONCENTRATIONS IF TRANSPORT EQUATION IS TO
C   BE SOLVED
C
      IF (TRANS) THEN
      READ(05,*) IREAD,FACTOR
      IF(IREAD.EQ.0) THEN
      WRITE(6,4210) FACTOR
      DO 190 N=1,NNODES
      CC(N)=FACTOR
      COLD(N)=FACTOR
  190 CONTINUE
      ELSE
      READ(05,*)IU,IFMT
      WRITE(06,4220) IU,FACTOR
      DO 200 J=1,NLY
      READ(IU,FMT=IFMT) (DUM(N),N=1,NXR)
      DO 200 N=1,NXR
      IN=NLY*(N-1)+J
      CC(IN)=DUM(N)*FACTOR
      COLD(IN)=CC(IN)
  200 CONTINUE
      END IF
C
C   COMPUTE INTERCELL CONDUCTANCES
C
      END IF
      CALL VSHCMP
      RETURN
  210 WRITE (06,4160) ITMAX
      STOP
 4000 FORMAT(10X,27HINITIAL MOISTURE PARAMETERS/10X,27(1H_)//
     &5X,'CONVERGENCE CRITERIA FOR SIP FOR FLOW =',1PE12.3,1X,A4/
     &5X,'CONVERGENCE CRITERIA FOR SIP FOR TRANSPORT =',1PE12.3,1X,/
     &5X,23HDAMPING FACTOR, HMAX = ,1PE12.3)
 4010 FORMAT(5X,46HGEOMETRIC MEAN USED FOR INTERCELL CONDUCTIVITY)
 4020 FORMAT(5X,45HUPSTREAM WEIGHTING USED FOR INTERCELL CONDUCT
     &,5HIVITY)
 4030 FORMAT(//15X,'NUMBER OF EVAPORATION AND/OR EVAPOTRASPIRATION PER'
     &,'IODS = ',I4,/,15X,'LENGTH OF EACH PERIOD = ',F10.4,2X,A4)
 4040 FORMAT(5X,'TEXTURAL CLASSES READ IN BY BLOCK')
 4050 FORMAT(//5X,'EVAPORATION   POTENTIAL      SURFACE    ATMOSHERIC',
     &/'      PERIOD      RATE       RESISTANCE     PRESSURE',
     &/19X,A4,'/',A4,3X,A4,'**(-1)',5X,A4,/,1X,90('-'),
     &25(/,5X,I6,4X,3E14.5))
 4060 FORMAT(/,/,3X,'TRANSPIRATION   POTENTIAL         ROOT       ACTIVIT
     &Y       ACTIVITY         ROOT',
```

```
     &/'        PERIOD          RATE              DEPTH      AT BOTTOM       A
     &T TOP         PRESSURE',/,19X,A4,'/',A4,9X,A4,5X,A4,'**(-2)',4X,A4,
     &'**(-2)',8X,A4,/,1X,90('-'),25(/,5X,I6,4X,5E14.5))
 4070 FORMAT(5X,47HARITHEMTIC MEAN USED FOR INTERCELL CONDUCTIVITY)
 4080 FORMAT(5X,34HNUMBER OF SOIL TEXTURAL CLASSES = ,I10/
     &5X,43HNUMBER OF SOIL PARAMETERS FOR EACH CLASS = ,I10/
     &5X,'NUMBER OF TRANSPORT PARAMETERS FOR EACH CLASS = ',I10/
     &5X,47HMINIMUM PERMITTED NO. OF ITERATIONS/TIME STEP =,I10/
     &5X,47HMAXIMUM PERMITTED NO. OF ITERATIONS/TIME STEP =,I10)
 4090 FORMAT(5X,41HTEXTURAL CLASS TO BE READ IN FOR EACH ROW)
 4100 FORMAT(41X,35HCONSTANTS FOR SOIL TEXTURAL CLASSES//
     &10X,10HANISOTROPY,7X,4HKSAT,5X,8HSPECIFIC,4X,8HPOROSITY,/,
     &36X,7HSTORAGE)
 4110 FORMAT(12X,'ALPHAL',8X,'ALPHAT',6X,'DM',9X,'LAMBDA',
     &4X,'B DENSITY')
 4120 FORMAT(1X,7HCLASS #,I2,/9X,3(1PD12.3),14(7(1PD12.3),/))
 4130 FORMAT(9X,10(1PD12.3))
 4140 FORMAT(6X,24HTEXTURAL CLASS INDEX MAP//   )
 4150 FORMAT(1H ,5X,I4,2X,100I1)
 4160 FORMAT(5X,24H ****** VALUE OF ITMAX =,I5,8HEXCEEDS ,
     &44HDIMENSION OF DHMX, PROGRAM TERMINATED ******)
 4170 FORMAT(5X,48HINITIAL PRESSURE HEAD OR MOISTURE CONTENT WAS SE,
     & 24HT TO A CONSTANT VALUE OF,1PE12.3)
 4180 FORMAT(5X,48HINITIAL PRESSURE HEAD OR MOISTURE CONTENT WAS RE,
     & 12HAD FROM UNIT,I5,
     & 20H A SCALING FACTOR OF,1PE12.3,9H WAS USED)
 4190 FORMAT(5X,'EQUILLIBRIUM PROFILE USED TO INITIALIZE PRESSURE',
     & 27H HEADS ABOVE WATER TABLE AT,F10.2,1X,A4,1X,
     & 12HBELOW ORIGIN/5X,
     & 57HEQUILLIBRIUM PROFILE ONLY USED UNTIL PRESSURE HEADS EQUAL,
     &  F10.2,1X,A4/5X,
     & 20HPRESSURE HEADS BELOW,F10.2,1X,A4,16H ARE HYDROSTATIC)
 4200 FORMAT(1H ,50X,18HDEPTH FROM SURFACE)
 4210 FORMAT('      INITIAL CONCENTRATION SET TO A CONSTANT VALUE OF ',
     &1PE12.3)
 4220 FORMAT('      INITIAL CONCENTRATION WAS READ FROM UNIT',I5,
     &' A SCALING FACTOR OF, ',1PE12.3,' WAS USED')
 4230 FORMAT('  INITIAL MOISTURE CONTENT AT ROW ',I3,' COLUMN ',
     &I3,' IS LESS THAN OR EQUAL TO 0.',/'  PROGRAM TERMINATED')
      END
      SUBROUTINE VSTMER
C******
CVSTMER
C******
C
C     PURPOSE: TO CONTROL THE TIME SEQUENCE OF SIMULATION
C     AND TO READ NEW BOUNDARY CONDITION DATA
C
C ------------------------------------------------------------
C
C   SPECIFICATIONS FOR ARRAYS AND SCALARS
C
      IMPLICIT DOUBLE PRECISION (A-H,P-Z)
      COMMON/RSPAC/DELZ(600),DZZ(600),DXR(600),RX(600),PI2
      COMMON/ISPAC/NLY,NLYY,NXR,NXRR,NNODES
      COMMON/KCON/HX(1600),NTYP(1600)
      COMMON/MPROP/THETA(1600),THLST(1600)
      COMMON/PRESS/P(1600),PXXX(1600),CS1,CS2
      COMMON/DISCH/Q(1600),QQ(1600),ETOUT,ETOUT1
      COMMON/HCON/HCND(1600),HKLL(1600),HKTT(1600)
      COMMON/DUMM/DUM(1600)
      COMMON/SPFC/JSPX(3,25,8),NFC(8),JLAST(8),NFCS
      COMMON/PTET/DPTH(1600),RT(1600),RDC(6,25),ETCYC,
     &PEVAL(25),PTVAL(25),PET,PEV,HROOT,HA,SRES,RTDPTH,
     &RTBOT,RTTOP,NPV
      COMMON/PND/POND
      COMMON/PLOTT/PLTIM(50),IJOBS(50),JPLT,NPLT,NOBS
```


AR 020594

```
      COMMON/SCON/DHMX(201),DELT,HMAX,TMAX,EPS,NUMT,ITMAX,MINIT,ITEST
      COMMON/SCN1/TMPX,TMLT,DLTMX,DLTMIN,TRED
      COMMON/TCON/STIM,DSMAX,KTIM,NIT,NIT1,KP
      COMMON/JCON/JSTOP,JFLAG
      COMMON/TRXX/DX1(1600),DX2(1600),DZ1(1600),DZ2(1600),VX(1600),
     &VZ(1600),CC(1600),COLD(1600),CS(1600),QT(1600),NCTYP(1600),
     &RET(1600)
      LOGICAL TRANS,TRANS1,SORP,SSTATE
      COMMON/TRXY/MB9(72),NMB9,EPS1,TRANS,TRANS1,SORP,SSTATE
      LOGICAL RAD,BCIT,ETSIM,SEEP,ITSTOP,CIS,CIT
      LOGICAL F7P,F11P,F8P,F9P,F6P,PRNT
      COMMON/LOG1/RAD,BCIT,ETSIM,SEEP,ITSTOP,CIS,CIT
      COMMON/LOG2/F7P,F11P,F8P,F9P,F6P,PRNT
      CHARACTER*80 TITL
      CHARACTER*4 ZUNIT,TUNIT,CUNX
      COMMON/SCHAR/TITL,ZUNIT,TUNIT,CUNX
      DIMENSION IDUM(0600)
      SAVE STERR,KPLT
C-----------------------------------------------------------------------
C
C   ADVANCE TO NEXT TIME STEP
C
      KTIM=KTIM+1
      IF (KTIM.NE.1.AND.JSTOP.EQ.1) RETURN
      JSTOP=0
      JPLT=0
      NIT=0
      NIT1=0
      IF(KTIM.EQ.1) KPLT=1
      IF(JFLAG.EQ.1) THEN
C
C ...................................................................
C
C    READ DATA FOR NEW RECHARGE PERIOD
C ...................................................................
C
      READ (05,*) TPER,DELT
C
C   CHECK FOR END OF SIMULATION
C
      IF(TPER.GE.999998.) THEN
      WRITE (06,4070) TMAX,STIM
      STOP
      END IF
      READ (05,*) TMLT,DLTMX,DLTMIN,TRED
      KP=KP+1
      SSTATE=.FALSE.
      WRITE (06,4000) KP,TPER,TUNIT,DELT,TUNIT,TMLT,DLTMX,TUNIT,DLTMIN,
     *TUNIT,TRED
      READ (05,*) DSMAX,STERR
      READ (05,*) POND
      WRITE (06,4020) DSMAX,STERR,POND
      READ (05,*) PRNT
      READ (05,*) BCIT,ETSIM,SEEP
      WRITE (06,4010) PRNT,BCIT,ETSIM,SEEP
      DSMAX=ABS(DSMAX)
      ETOUT=0
      ETOUT1=0
C
C    READ SEEPAGE FACE DATA
C
      IF(SEEP) THEN
      READ (05,*) NFCS
      DO 20 K=1,NFCS
      READ (05,*) JJ,JLAST(K)
      NFC(K)=JJ
      READ (05,*) ((JSPX(L,J,K),L=2,3),J=1,JJ)
```

```
      DO 10 J=1,JJ
      J1=JSPX(2,J,K)
      N1=JSPX(3,J,K)
      N2=NLY*(N1-1)+J1
      JSPX(1,J,K)=N2
      Q(N2)=0.
      QQ(N2)=0.
      NCTYP(N2)=0
      CS(N2)=0
      IF(J.GT.JLAST(K)) THEN
      NTYP(N2)=3
      ELSE
      NTYP(N2)=1
      IF(CS1.EQ.1.) THEN
      Z1=DZZ(J1)
      ELSE
      Z1=DZZ(J1)*CS1+(RX(N1))*CS2
      END IF
      P(N2)=-Z1
      END IF
   10 CONTINUE
   20 CONTINUE
      END IF
C
C    READ IN NEW BOUNDARY CONDITIONS FOR RECHARGE PERIOD
C      IF IBC=0, POINT BOUNDARY CONDITIONS ARE READ IN.
C      IF IBC=1, LINE BOUNDARY CONDITIONS ARE READ IN, AND IT IS NECESSARY
C       TO SPECIFY FOUR POINTS ON THE LINE--THIS ALLOWS VERTICAL OR HORI-
C       ZONTAL LINES TO BE READ IN INDISCRIMINATELY.  THE SEQUENCE IS:
C       TOP ROW, BOTTOM ROW, LEFT COLUMN, RIGHT COLUMN, CODE, AND FLUX OR
C       PRESSURE HEAD FOR BOUNDARY CONDITION.
C
      READ (05,*) IBC
      IF(IBC.GT.0) GO TO 40
   30 IF (TRANS) THEN
      READ(05,*) JJ,NN,NTX,PFDUM,NTC,CF
      ELSE
      READ (05,*) JJ,NN,NTX,PFDUM
      CF=0
      NTC=0
      END IF
      IF(JJ.GE.999998) GO TO 90
      JJT=JJ
      JJB=JJ
      NNL=NN
      NNR=NN
      GO TO 50
   40 IF (TRANS) THEN
      READ(05,*) JJT,JJB,NNL,NNR,NTX,PFDUM,NTC,CF
      ELSE
      READ (05,*) JJT,JJB,NNL,NNR,NTX,PFDUM
      CF=0
      NTC=0
      END IF
      IF(JJT.GE.999) GO TO 90
   50 CONTINUE
      DO 80 JJ=JJT,JJB
      DO 80 NN=NNL,NNR
      IN=NLY*(NN-1)+JJ
      CS(IN)=CF
      IF(NTC.EQ.1) CC(IN)=CF
      NCTYP(IN)=NTC
      IF(NTX.NE.6) GO TO 60
      NTYP(IN)=2
      QQ(IN)=PFDUM
      GO TO 80
   60 NTYP(IN)=NTX
```

```
      IF(NTX .EQ. 4)NTYP(IN)=1
      IF(NTX.EQ.0) WRITE (06,4030) JJ,NN
      IF(CS1.EQ.1.) THEN
      Z1=DZZ(JJ)
      ELSE
      Z1=DZZ(JJ)*CS1+(RX(NN))*CS2
      END IF
      IF(NTX.EQ.1) P(IN)=PFDUM-Z1
      IF(NTX.EQ.4) P(IN)=PFDUM
      IF(NTX.EQ.2) GO TO 70
      QQ(IN)=0
      GO TO 80
   70 CONTINUE
C
C     SET QQ TO RAINFALL RATE
C
      AREA=DXR(NN)
      IF(RAD)AREA=PI2*RX(NN)*DXR(NN)
      QQ(IN)=PFDUM*AREA
   80 CONTINUE
      IF(IBC.EQ.0) GO TO 30
      GO TO 40
   90 CONTINUE
C
C      WRITE INITIAL BOUNDARY CONDITIONS FOR THIS PERIOD
C
      WRITE (06,4040) KP
      DO 110 J=1,NLY
      DO 100 N=1,NXR
      IN=NLY*(N-1)+J
      Q(IN)=0.
  100 IDUM(N)=NTYP(IN)
  110 WRITE (06,4050) J,(IDUM(I),I=1,NXR)
      TMPX=STIM+TPER
      IF(TMPX+0.5*DLTMIN.GT.TMAX) TMPX=TMAX
C
C   CALCULATE NEW COEFFICIENTS
C
      IF(KTIM.NE.1)CALL VSCOEF
      END IF
C
C   INITIALIZE REQUIRED ARRAYS FOR NEW BOUNDARY CONDITION, UPDATE
C   PXXX,THLST.  COMPUTE MAXIMUM HEAD CHANGE DURING LAST TIME STEP
C
      PDIF=0.
      IF(KTIM.NE.1.AND..NOT.SSTATE) THEN
      DO 120 J=2,NLYY
      DO 120 N=2,NXRR
      IN=NLY*(N-1)+J
      IF(HX(IN).EQ.0.) GO TO 120
      P12=P(IN)-PXXX(IN)
      PTMP=ABS(P12)
      IF(PTMP.GT.PDIF)PDIF=PTMP
      PXXX(IN)=P(IN)
      THLST(IN)=THETA(IN)
  120 CONTINUE
C
C    CHECK FOR STEADY STATE
C
      IF(PDIF.LE.STERR) THEN
      SSTATE=.TRUE.
      WRITE(6,4060) STIM,KTIM
      END IF
      END IF
      JFLAG=0
C
C   INITIALIZE DHMX
```

```
C
      DO 130 K=1,201
  130 DHMX(K)=0.
C
C     ADVANCE DELT AND RESET TO PROPER LENGTH IF NECESSARY
C
      DLTOLD=DELT
      DELT= TMLT*DELT
C
C     MAXIMUM PERMISSABLE HEAD CHANGE CHECK
C
      IF(KTIM.GE.2) THEN
      IF((PDIF*DELT/DLTOLD).GT.DSMAX)DELT=DLTOLD*DSMAX*.98/PDIF
      END IF
      IF(ABS(TMPX-PLTIM(KPLT)).LT.DLTMIN) PLTIM(KPLT)=TMPX
      T1=DMIN1(TMPX,PLTIM(KPLT))
      T2=T1-STIM
      IF(DELT.GT.(T2-DLTMIN)) DELT=T2
      IF(DELT.LT.DLTMIN)DELT=DLTMIN
      IF(DELT.GT.DLTMX)DELT=DLTMX
      IF(T1.NE.PLTIM(KPLT).OR.T2-DELT.GT.0.5*DLTMIN) GO TO 140
      KPLT=KPLT+1
      JPLT=1
  140 IF(DELT.LT.DLTMIN)DELT=DLTMIN
      STIM=STIM+DELT
      IF (TMPX-STIM.LT.0.5*DLTMIN) JFLAG=1
      IF(TMAX-STIM.LT.0.5*DLTMIN.OR.KTIM.GT.NUMT) THEN
      JSTOP=1
      JPLT=1
      END IF
      RETURN
 4000 FORMAT(6X,'DATA FOR RECHARGE PERIOD ',I5//10X,
     &23HLENGTH OF THIS PERIOD =,1PE12.3,1X,A4/10X,
     &45HLENGTH OF INITIAL TIME STEP FOR THIS PERIOD =,1PE10.3,1X,A4/
     &10X,27HMULTIPLIER FOR TIME STEP = ,1PE10.3,/10X,
     &25HMAXIMUM TIME STEP SIZE = ,1PE10.3,1X,A4/10X,
     &25HMINIMUM TIME STEP SIZE = ,1PE10.3,1X,A4,
     &/10X,'TIME STEP REDUCTION FACTOR = ',1PE10.3)
 4010 FORMAT(15X,37HPRINT SOLUTION AFTER EVERY TIME STEP?,1X,L1/
     &15X,'SIMULATE EVAPORATION? ',L1/
     &15X,29HSIMULATE EVAPOTRANSPIRATION? ,L1/
     &15X,24HSIMULATE SEEPAGE FACES? ,L1/)
 4020 FORMAT(
     &15X,55HMAXIMUM PRESSURE HEAD CHANGE ALLOWED IN ONE TIME STEP =,
     &F8.3/15X,'STEADY-STATE CLOSURE CRITERION = ',1PE10.3/
     &15X,'MAXIMUM DEPTH OF PONDING = ',1PE10.3)
 4030 FORMAT(1H ,1X,10(1H*),41HWARNING --- NODE TYPE OF 0 ASSIGNED TO BO
     &12HUNDARY NODE ,2I4,43H SPECIFIED FLUX OR PRESSURE HEAD NOT ASSIGN
     &2HED)
 4040 FORMAT(6X,41HNODE TYPE AND INITIAL BOUNDARY CONDITIONS,
     &12H FOR PERIOD ,I4/6X,8HLEGEND: /15X,17H0 = INTERIOR CELL/
     &15X,32H1 = SPECIFIED PRESSURE HEAD CELL/15X,
     &23H2 = SPECIFIED FLUX CELL/
     & 15X,31H3 = POTENTIAL SEEPAGE FACE NODE/
     & 15X,43H5 = NODE FOR WHICH EVAPORATION IS PERMITTED//)
 4050 FORMAT(1H ,I5,5X,80I1)
 4060 FORMAT(6X,100(1H*)/5X,
     &'STEADY STATE REACHED AT TIME = ',E12.4,'   TIME STEP NUMBER = '
     &,I5//)
 4070 FORMAT(6X,100(1H*),/,5X,17HEND OF SIMULATION/,
     &5X,33HMAXIMUM SIMULATION TIME (TMAX) = ,E15.4/,
     &5X,33HELAPSED SIMULATION TIME (STIM) = ,E15.4/,
     &6X,100(1H*))
      END
      SUBROUTINE VSMGEN
C******
CVSMGEN
```

```
C******
C
C    PURPOSE:  TO SET UP COEFFICIENT MATRICES AND CALL
C         SOLUTION ALGORITHM
C
C-----------------------------------------------------------------------
C
C   SPECIFICATIONS FOR ARRAYS AND SCALARS
C
      IMPLICIT DOUBLE PRECISION (A-H,P-Z)
      COMMON/RSPAC/DELZ(600),DZZ(600),DXR(600),RX(600),PI2
      COMMON/ISPAC/NLY,NLYY,NXR,NXRR,NNODES
      COMMON/KCON/HX(1600),NTYP(1600)
      COMMON/RPROP/HK(10,100),HT(10,20),ANIZ(10)
      COMMON/MPROP/THETA(1600),THLST(1600)
      COMMON/PRESS/P(1600),PXXX(1600),CS1,CS2
      COMMON/DISCH/Q(1600),QQ(1600),ETOUT,ETOUT1
      COMMON/HCON/HCND(1600),HKLL(1600),HKTT(1600)
      COMMON/EQUAT/A(1600),B(1600),C(1600),D(1600),E(1600),RHS(1600),
     &XI(1600)
      COMMON/JTXX/JTEX(1600)
      COMMON/PTET/DPTH(1600),RT(1600),RDC(6,25),ETCYC,
     &PEVAL(25),PTVAL(25),PET,PEV,HROOT,HA,SRES,RTDPTH,
     &RTBOT,RTTOP,NPV
      COMMON/WGT/WUS,WDS
      COMMON/SCON/DHMX(201),DELT,HMAX,TMAX,EPS,NUMT,ITMAX,MINIT,ITEST
      COMMON/SCN1/TMPX,TMLT,DLTMX,DLTMIN,TRED
      COMMON/TCON/STIM,DSMAX,KTIM,NIT,NIT1,KP
      COMMON/JCON/JSTOP,JFLAG
      LOGICAL RAD,BCIT,ETSIM,SEEP,ITSTOP,CIS,CIT
      COMMON/LOG1/RAD,BCIT,ETSIM,SEEP,ITSTOP,CIS,CIT
      CHARACTER*80 TITL
      CHARACTER*4 ZUNIT,TUNIT,CUNX
      COMMON/SCHAR/TITL,ZUNIT,TUNIT,CUNX
      DIMENSION PITT(1600)
      SAVE PITT
C
C .....................................................................
C  START OF LINEARIZATION ITERATION LOOP
C. ....................................................................
C
C    UPDATE COEFFICIENTS
C
      I13=0
C
C     ESTABLISH TIME-DEPENDENT PARAMETERS GOVERNING EVAPORATION AND
C     TRANSPIRATION.  DETERMINE ROOT ACTIVITY.
C
   10 IF ( BCIT.OR. ETSIM) THEN
      CALL VSPET
      DO 20 J=2,NLYY
      DO 20 I=2,NXRR
      N=NLY*(I-1)+J
      IF(HX(N).GT.0) THEN
      IF(ETSIM) RT(N)=VSRDF(DPTH(N),DELZ(J))
      Q(N)=0.0
      END IF
   20 CONTINUE
      END IF
   30 IF (NIT.NE.0) CALL VSCOEF
C
C -------------------- UPDATE BOUNDARY AND FLUX CONDITIONS ------------
C
      IF(BCIT)CALL VSEVAP
      IF (ETSIM)CALL VSPLNT
      IF(SEEP) CALL VSSFAC
C
```

```
C ..............................................................
C
C        LOOP TO CALCULATE COEFFICIENT MATRIX
C ..............................................................
C
      DO 40 J=2,NLYY
      DO 40 I=2,NXRR
      N=NLY*(I-1)+J
      IF(HX(N).GT.0.) THEN
      JM1=N-1
      JP1=N+1
      IM1=N-NLY
      IP1=N+NLY
      VOL=DXR(I)*DELZ(J)
      IF(RAD)VOL=PI2*RX(I)*DXR(I)*DELZ(J)
      JJ=JTEX(N)
C
C   CALCULATE STORAGE TERMS
C
      IF(CS1.EQ.1.) THEN
      Z1=DZZ(J)
      ELSE
      Z1=DZZ(J)*CS1+(RX(I))*CS2
      END IF
      PTMP=P(N)+Z1
      SCAP=VSDTHU(PTMP,JJ,HK)
      GSF=VOL*SCAP
      SS=HK(JJ,2)/HK(JJ,3)
      GSS=VOL*THETA(N)*SS
      G1=0
C
C   APPLY NEWTON-RAPHSON LINEARIZATION TO STORAGE TERM.
C   PITT HOLDS STORAGE TERMS FROM PREVIOUS ITERATION.
C
      IF(NIT.GT.0.AND.XI(N).NE.0)G1=(P(N)-PXXX(N))*(GSF+GSS-PITT(N))/
     *XI(N)
      PITT(N)=GSF+GSS
      G1=-G1/DELT
      GSF=-GSF/DELT
      GSS=-GSS/DELT
      IF(WUS.EQ.0.) THEN
C
C   USE GEOMETRIC MEAN OR WEIGHTS FOR INTERCELL K
C
      A(N)=HKLL(N)*DSQRT(HCND(IM1)*HCND(N))
      B(N)=HKTT(N)*DSQRT(HCND(JM1)*HCND(N))
      C(N)=HKLL(IP1)*DSQRT(HCND(IP1)*HCND(N))
      D(N)=HKTT(JP1)*DSQRT(HCND(JP1)*HCND(N))
      ELSE
C
C  CHOOSE UPSTREAM WEIGHTING COEFFICIENTS
C
      IF(P(IM1).LE.P(N).OR.HX(IM1).EQ.0.) THEN
      ALA=WDS
      BTA=WUS
      ELSE
      ALA=WUS
      BTA=WDS
      END IF
      IF(P(JM1).LE.P(N).OR.HX(JM1).EQ.0.) THEN
      ALB=WDS
      BTB=WUS
      ELSE
      ALB=WUS
      BTB=WDS
      END IF
      IF(P(IP1).LE.P(N).OR.HX(IP1).EQ.0.) THEN
```

```
      ALC=WDS
      BTC=WUS
      ELSE
      ALC=WUS
      BTC=WDS
      END IF
      IF(P(JP1).LE.P(N).OR.HX(JP1).EQ.0.) THEN
      ALD=WDS
      BTD=WUS
      ELSE
      ALD=WUS
      BTD=WDS
      END IF
C
C    SET THE PENTA-DIAGNOL COEFFICIENT MATRIX (E IS MAIN DIAGNOL)
C    AND RIGHT HAND SIDE
C
      A(N)=(ALA*HCND(IM1)+BTA*HCND(N))*HKLL(N)
      B(N)=(ALB*HCND(JM1)+BTB*HCND(N))*HKTT(N)
      C(N)=(ALC*HCND(IP1)+BTC*HCND(N))*HKLL(IP1)
      D(N)=(ALD*HCND(JP1)+BTD*HCND(N))*HKTT(JP1)
      END IF
      E(N)=-A(N)-B(N)-C(N)-D(N)
      RHS(N)=VOL*(THETA(N)-THLST(N))/DELT-(Q(N)+QQ(N))-(A(N)*P(IM1)+B(N)
     &*P(JM1)+C(N)*P(IP1)+D(N)*P(JP1)+(E(N)+GSS)*P(N))+GSS*PXXX(N)
      E(N)=E(N)+GSF+GSS+G1
      END IF
   40 CONTINUE
C
C    CALL SOLUTION ALGORITHM
C
      NIT=NIT+1
      CALL SLVSIP
      IF(NIT.LT.MINIT) GO TO 30
C
C    IF SOLUTION HAS BEEN FOUND THEN RETURN
C
      IF(ITEST.EQ.0) RETURN
      IF(NIT.LE.ITMAX) GO TO 30
C
C    MAXIMUM NUMBER OF ITERATIONS EXCEEDED
C
      WRITE (6,4000) NIT,KTIM,STIM,TUNIT
C
C    AUTOMATICALLY REDUCE TIME STEP SIZE, BUT NOT MORE
C    THAN TWICE.
C
      IF(DELT.LE.DLTMIN.OR.I13.GT.2.OR.TRED.LE.0) THEN
      IF(.NOT.ITSTOP)RETURN
C
C    TERMINATE SIMULATION.
C
      JSTOP=1
      JFLAG=1
      RETURN
      ELSE
      I13=I13+1
      DELTT=DELT*TRED
      IF(DELTT.LT.DLTMIN) DELTT=DLTMIN
      WRITE(6,4010) DELTT
      STIM=STIM-DELT+DELTT
      DELT=DELTT
C
C    RESET HEADS TO VALUES AT END OF PREVIOUS TIME STEP.
C
      DO 50 II=1,NNODES
      IF(NTYP(II).NE.1.AND.HX(II).GT.0) P(II)=PXXX(II)
```

```
   50 CONTINUE
      NIT=1
      GO TO 10
      END IF
 4000 FORMAT(5X,100(1H*)/5X,'EXCEEDED PERMITTED NUMBER OF ITERATIONS',
     &'  ( =',I4,')'
     & /5X,'TIME STEP NUMBER',I4/5X,'ELAPSED TIME = ',
     & 1PE12.3,1X,A4  /5X,100(1H*))
 4010 FORMAT(5X,'TIME STEP SIZE REDUCED TO ',E12.4)
      END
      SUBROUTINE VSSIP
C
C*****
CVSSIP
C*****
C
C     PURPOSE: TO SOLVE THE  MATRIX EQUATIONS USING THE
C     STRONGLY IMPLICIT METHOD
C
C -----------------------------------------------------------------
C
C   SPECIFICATIONS FOR ARRAYS AND SCALARS
C
      IMPLICIT DOUBLE PRECISION (A-H,P-Z)
      COMMON/RSPAC/DELZ(600),DZZ(600),DXR(600),RX(600),PI2
      COMMON/ISPAC/NLY,NLYY,NXR,NXRR,NNODES
      COMMON/KCON/HX(1600),NTYP(1600)
      COMMON/RPROP/HK(10,100),HT(10,20),ANIZ(10)
      COMMON/PRESS/P(1600),PXXX(1600),CS1,CS2
      COMMON/EQUAT/A(1600),B(1600),C(1600),D(1600),E(1600),RHS(1600),
     &XI(1600)
      COMMON/JTXX/JTEX(1600)
      COMMON/SCON/DHMX(201),DELT,HMAX,TMAX,EPS,NUMT,ITMAX,MINIT,ITEST
      COMMON/SCN1/TMPX,TMLT,DLTMX,DLTMIN,TRED
      COMMON/TCON/STIM,DSMAX,KTIM,NIT,NIT1,KP
      COMMON/TRXX/DX1(1600),DX2(1600),DZ1(1600),DZ2(1600),VX(1600),
     &VZ(1600),CC(1600),COLD(1600),CS(1600),QT(1600),NCTYP(1600),
     &RET(1600)
      LOGICAL TRANS,TRANS1,SORP,SSTATE
      COMMON/TRXY/MB9(72),NMB9,EPS1,TRANS,TRANS1,SORP,SSTATE
      DIMENSION IORDER(21)
      DIMENSION DEL(1600),ETA(1600),V(1600),TEMP(100),HM(30)
      SAVE HM,W1,W9,L2
C
C------------------------------------------------------------------
C
      DATA IORDER/1,2,3,4,5,1,2,3,4,5,11*1/
C
C     COMPUTE ITERATION PARAMETERS
C
      J2=NXR-2
      I2=NLY-2
      L2=5
      PL2=L2-1
      W=0.
      PIE=0.
      W9=100.
C
C  COMPUTE MAXIMUM PARAMETER
C
      DO 10 I=2,NLYY
      DO 10 J=2,NXRR
      N=NLY*(J-1)+I
      IF(HX(N).GT.0.) THEN
      IM1=JTEX(N)
      PIE=PIE+1.
      DX=DXR(J)/RX(NXR)
```

```
      DY=DELZ(I)/DZZ(NLY)
      DX3=DX*DX
      DY2=DY*DY
      W=W+1-DMIN1((DX3+DX3)/(1.+ANIZ(IM1)*DX3/DY2),(DY2+DY2)/(1+DY2/
     &(ANIZ(IM1)*DX3)))
      END IF
   10 CONTINUE
      W=W/PIE
C
C COMPUTE PARAMETERS IN GEOMETRIC SEQUENCE
C
      PJ=-1.
      DO 20 I=1,L2
      PJ=PJ+1.
   20 TEMP(I)=1. -(1. -W)**(PJ/PL2)
C
C ORDER SEQUENCE OF PARAMETERS
C
      DO 30 J=1,L2
   30 HM(J)=TEMP(IORDER(J))
      WRITE (06,4000) L2,(HM(J),J=1,L2)
      RETURN
C
C STRONGLY IMPLICIT ALGORITHM
C
      ENTRY SLVSIP
      I2=NLY-2
      J2=NXR-2
C
C       SELECT ITERATION PARAMETER.  INITIALIZE ARRAYS
C
      IF(TRANS1) THEN
C
C  IF TRANS1=T  TRANSPORT EQUATION IS SOLVED
C           =F  FLOW EQUATION IS SOLVED
C
      NT=NIT1
      ELSE
      NT=NIT
      END IF
      IF(MOD(NT,L2).EQ.0.OR.NT.EQ.1)NTH=0
      NTH=NTH+1
      W=HM(NTH)
      ITEST=0
      DO 40 I=1,NNODES
      DEL(I)=0.
      ETA(I)=0.
      V(I)=0.
   40 XI(I)=0.
      BIGI=0.
      BIGI1=0.
C
C CHOOSE SIP NORMAL OR REVERSE ALGORITHM
C
      IF(MOD(NT,2)) 50,80,50
C .....................................................................
C ORDER EQUATIONS WITH ROW 1 FIRST  -  3X3 EXAMPLE:
C    1 2 3
C    4 5 6
C    7 8 9
C .....................................................................
   50 DO 60 I=2,NLYY
      DO 60 J=2,NXRR
      N=I+NLY*(J-1)
C
C   ---- SKIP COMPUTATIONS OF NODE IS OUTSIDE OF SOLUTION DOMAIN
C
```

```
      IF(HX(N).EQ.0.) GO TO 60
      IF((NTYP(N).EQ.1.AND.(.NOT.TRANS1)).OR.(TRANS1.AND.(NCTYP(N).EQ.1
     *)))GO TO 60
      NL=N-NLY
      NA=N-1
      NB=N+1
C
C     --- SIP "NORMAL" ALGORITHM-----
C     --- FORWARD SUBSTITUTE, COMPUTING INTERMEDIATE VECTOR V --
C
      CH=DEL(NA)*B(N)/(1. +W*DEL(NA))
      GH=ETA(NL)*A(N)/(1. +W*ETA(NL))
      BH=B(N)-W*CH
      DH=A(N)-W*GH
      EH=E(N)+W*CH+W*GH
      FH=C(N)-W*CH
      HH=D(N)-W*GH
      ALFA=BH
      BETA=DH
      GAMA=EH-ALFA*ETA(NA)-BETA*DEL(NL)
      DEL(N)=FH/GAMA
      ETA(N)=HH/GAMA
      RES=RHS(N)
      V(N)=(HMAX*RES-ALFA*V(NA)-BETA*V(NL))/GAMA
   60 CONTINUE
C
C  ---BACK SUBSTITUTE FOR VECTOR XI
C
      DO 70 I=1,I2
      I3=NLY-I
      DO 70 J=1,J2
      J3=NXR-J
      N=I3+NLY*(J3-1)
      IF(HX(N).EQ.0.) GO TO 70
      IF((NTYP(N).EQ.1.AND.(.NOT.TRANS1)).OR.(TRANS1.AND.(NCTYP(N).EQ.1
     *)))GO TO 70
      XI(N)=V(N)-DEL(N)*XI(N+NLY)-ETA(N)*XI(N+1)
C
C      FIND MAXIMUM HEAD CHANGE
C
      TCHK=ABS(XI(N))
      IF(TCHK.GE.BIGI) THEN
      BIGI=TCHK
      BIGI1=XI(N)
      END IF
   70 CONTINUE
      GO TO 110
C
C.................................................................
C  ---ORDER EQUATIONS WITH THE LAST ROW FIRST  -  3X3 EXAMPLE
C           7 8 9
C           4 5 6
C           1 2 3
C.................................................................
C
   80 DO 90 II=1,I2
      I=NLY-II
      DO 90 J=2,NXRR
      N=I+NLY*(J-1)
      NL=N-NLY
      NA=N-1
      NB=N+1
C
C  -- SKIP COMPUTATIONS IF NODE IS OUTSIDE OF SOLUTION DOMAIN
C
      IF(HX(N).EQ.0.) GO TO 90
      IF((NTYP(N).EQ.1.AND.(.NOT.TRANS1)).OR.(TRANS1.AND.(NCTYP(N).EQ.1
```

```
     *)))GO TO 90
C
C ------ SIP "REVERSE" ALGORITHM
C --- FORWARD SUBSTITUTE, COMPUTING INTERMEDIATE VECTOR V
C
      CH=DEL(NB)*D(N)/(1. +W*DEL(NB))
      GH=ETA(NL)*A(N)/(1. +W*ETA(NL))
      BH=D(N)-W*CH
      DH=A(N)-W*GH
      EH=E(N)+W*CH+W*GH
      FH=C(N)-W*CH
      HH=B(N)-W*GH
      ALFA=BH
      BETA=DH
      GAMA=EH-ALFA*ETA(NB)-BETA*DEL(NL)
      DEL(N)=FH/GAMA
      ETA(N)=HH/GAMA
      RES=RHS(N)
      V(N)=(HMAX*RES-ALFA*V(NB)-BETA*V(NL))/GAMA
   90 CONTINUE
C
C --- BACK SUBSTITUTE FOR VECTOR XI
C
      DO 100 I3=2,NLYY
      DO 100 J=1,J2
      J3=NXR-J
      N=I3+NLY*(J3-1)
      IF(HX(N).EQ.0.) GO TO 100
      IF((NTYP(N).EQ.1.AND.(.NOT.TRANS1)).OR.(TRANS1.AND.(NCTYP(N).EQ.1
     *)))GO TO 100
      XI(N)=V(N)-DEL(N)*XI(N+NLY)-ETA(N)*XI(N-1)
C
C      FIND MAXIMUM HEAD CHANGE
C
      TCHK=ABS(XI(N))
      IF(TCHK.GE.BIGI) THEN
      BIGI=TCHK
      BIGI1=XI(N)
      END IF
  100 CONTINUE
C
C      COMPUTE RELAXATION PARAMETER W FOR HEAD CHANGES.  ALGORITHM
C       IS FROM COOLEY (1983)
C
  110 S=1.
      IF(NT.GT.1.AND.W1.NE.0.0) S=BIGI1/W1
      S1=ABS(S)
      IF(S.LT.-1.) THEN
      W=1/(S1+S1)
      ELSE
      W=(3+S)/(3+S1)
      END IF
      IF(W.EQ.W9) W=.9*W
      W1=W*BIGI
      IF(W1.GT.DSMAX) W=DSMAX/BIGI
      IF(BIGI1.LT.0.) W1=-W1
C
C      ADD CHANGES TO MATRIX.
C
      W9=W
      IF(TRANS1) THEN
      DO 120 N=NLY+1,NNODES
      IF(NCTYP(N).NE.1.AND.HX(N).GT.0.) CC(N)=CC(N)+W*XI(N)
  120 CONTINUE
      IF(BIGI.GT.EPS1) ITEST=1
      ELSE
      DO 130 N=NLY+1,NNODES
```

```
      IF(HX(N).GT.0.AND.NTYP(N).NE.1) P(N)=P(N)+W*XI(N)
  130 CONTINUE
C
C       COMPARE MAXIMUM HEAD CHANGE TO CLOSURE CRITERION.
C
      IF(BIGI.GT.EPS) ITEST=1
      DHMX(NIT)=BIGI
      END IF
      RETURN
 4000 FORMAT(1X,I5,25HSIP ITERATION PARAMETERS:,6D15.7/(28X,6D15.7/))
      END
      SUBROUTINE VSCOEF
C******
CVSCOEF
C******
C     PURPOSE: TO COMPUTE ALL VALUES OF NONLINEAR COEFFICIENTS
C              USING THE MOST RECENT VALUES OF PRESSURE HEAD
C ------------------------------------------------------------
C
C   SPECIFICATIONS FOR ARRAYS AND SCALARS
C
      IMPLICIT DOUBLE PRECISION (A-H,P-Z)
      COMMON/RSPAC/DELZ(600),DZZ(600),DXR(600),RX(600),PI2
      COMMON/ISPAC/NLY,NLYY,NXR,NXRR,NNODES
      COMMON/KCON/HX(1600),NTYP(1600)
      COMMON/RPROP/HK(10,100),HT(10,20),ANIZ(10)
      COMMON/MPROP/THETA(1600),THLST(1600)
      COMMON/PRESS/P(1600),PXXX(1600),CS1,CS2
      COMMON/HCON/HCND(1600),HKLL(1600),HKTT(1600)
      COMMON/JTXX/JTEX(1600)
C
C--------------------------------------------------------------
      DO 10 J=2,NLYY
      DO 10 N=2,NXRR
      IN=NLY*(N-1)+J
      IF(HX(IN).GT.0.) THEN
      J1=JTEX(IN)
      HCND(IN)=0.D0
C
C        COMPUTE PRESSURE HEADS TO USE IN FUNCTIONS
C
      IF(CS1.EQ.1.) THEN
      Z1=DZZ(J)
      ELSE
      Z1=DZZ(J)*CS1+(RX(N))*CS2
      END IF
      PTMP=P(IN)+Z1
      HCND(IN)=VSHKU(PTMP,J1,HK)
      THETA(IN)=VSTHU(PTMP,J1,HK)
      END IF
   10 CONTINUE
      RETURN
      END
      SUBROUTINE VSHCMP
C******
CVSHCMP
C******
C
C   PURPOSE: TO COMPUTE INTERCELL CONDUCTANCES
C
C ------------------------------------------------------------
C
C   SPECIFICATIONS FOR ARRAYS AND SCALARS
C
      IMPLICIT DOUBLE PRECISION (A-H,P-Z)
      COMMON/RSPAC/DELZ(600),DZZ(600),DXR(600),RX(600),PI2
      COMMON/ISPAC/NLY,NLYY,NXR,NXRR,NNODES
```

```
      COMMON/KCON/HX(1600),NTYP(1600)
      COMMON/RPROP/HK(10,100),HT(10,20),ANIZ(10)
      COMMON/HCON/HCND(1600),HKLL(1600),HKTT(1600)
      COMMON/JTXX/JTEX(1600)
      LOGICAL RAD,BCIT,ETSIM,SEEP,ITSTOP,CIS,CIT
      COMMON/LOG1/RAD,BCIT,ETSIM,SEEP,ITSTOP,CIS,CIT
C
C-----------------------------------------------------------------------
C
C     COMPUTE HARMONIC MEANS OF KSAT AND GRID SPACING
C
      DO 10 J=2,NLY
      DO 10 N=2,NXR
      IN=NLY*(N-1)+J
      JM1=IN-1
      NM1=IN-NLY
      A1=ANIZ(JTEX(IN))
      A2=ANIZ(JTEX(JM1))
      IF(HX(IN).EQ.0.) GO TO 10
      AREA=DXR(N)
      IF(RAD)AREA=PI2*RX(N)*DXR(N)
C
C   VERTICAL CONDUCTANCE
C   THROUGH TOP
C
      HKTT(IN)=2.0*A1*A2*AREA*HX(IN)*HX(JM1)/(A2*HX(JM1)*DELZ(J)+
     &A1*HX(IN)*DELZ(J-1))
      AREA=DELZ(J)
      IF(RAD)AREA=PI2*DELZ(J)*(RX(N)-.5 *DXR(N))
C
C   HORIZONTAL OR RADIAL CONDUCTANCE
C   THROUGH LEFT-HAND SIDE
C
      HKLL(IN)=2.0*AREA*HX(IN)*HX(NM1)/(HX(NM1)*DXR(N)+HX(IN)*DXR(N-1))
   10 CONTINUE
      RETURN
      END
      SUBROUTINE VSFLUX
C******
CVSFLUX
C******
C
C       PURPOSE: TO COMPUTE FLUXES AND MASS BALANCE
C
C ---------------------------------------------------------------------
C
C   SPECIFICATIONS FOR ARRAYS AND SCALARS
C
      IMPLICIT DOUBLE PRECISION (A-H,P-Z)
      COMMON/RSPAC/DELZ(600),DZZ(600),DXR(600),RX(600),PI2
      COMMON/ISPAC/NLY,NLYY,NXR,NXRR,NNODES
      COMMON/KCON/HX(1600),NTYP(1600)
      COMMON/RPROP/HK(10,100),HT(10,20),ANIZ(10)
      COMMON/MPROP/THETA(1600),THLST(1600)
      COMMON/PLOTT/PLTIM(50),IJOBS(50),JPLT,NPLT,NOBS
      COMMON/PRESS/P(1600),PXXX(1600),CS1,CS2
      COMMON/DISCH/Q(1600),QQ(1600),ETOUT,ETOUT1
      COMMON/JTXX/JTEX(1600)
      COMMON/SCON/DHMX(201),DELT,HMAX,TMAX,EPS,NUMT,ITMAX,MINIT,ITEST
      COMMON/SCN1/TMPX,TMLT,DLTMX,DLTMIN,TRED
      COMMON/TCON/STIM,DSMAX,KTIM,NIT,NIT1,KP
      COMMON/JCON/JSTOP,JFLAG
      COMMON/TRXX/DX1(1600),DX2(1600),DZ1(1600),DZ2(1600),VX(1600),
     &VZ(1600),CC(1600),COLD(1600),CS(1600),QT(1600),NCTYP(1600),
     &RET(1600)
      LOGICAL TRANS,TRANS1,SORP,SSTATE
      COMMON/TRXY/MB9(72),NMB9,EPS1,TRANS,TRANS1,SORP,SSTATE
```

```
      LOGICAL RAD,BCIT,ETSIM,SEEP,ITSTOP,CIS,CIT
      LOGICAL F7P,F11P,F8P,F9P,F6P,PRNT
      COMMON/LOG1/RAD,BCIT,ETSIM,SEEP,ITSTOP,CIS,CIT
      COMMON/LOG2/F7P,F11P,F8P,F9P,F6P,PRNT
      CHARACTER*80 TITL
      CHARACTER*4 ZUNIT,TUNIT,CUNX
      COMMON/SCHAR/TITL,ZUNIT,TUNIT,CUNX
      DIMENSION BL(72)
      SAVE BL
C----------------------------------------------------------------------
C
C   INITIALIZE MASS BALANCE VARIABLES USED FOR
C   ENTIRE SIMULATION.
C
      IF(KTIM.EQ.1) THEN
      DO 10 I=1,72
      BL(I)=0.
   10 CONTINUE
      END IF
C
C   INITIALIZE MASS BALANCE VARIABLES USED FOR CURRENT
C   TIME STEP
C
      BLTEMP=0
      BL(3)=0.
      BL(6)=0.
      BL(9)=0.
      BL(12)=0.
      BL(27)=0.
      BL(29)=0
      BL(36)=0.
      BL(39)=0.
      BL(42)=0.
      BL(45)=0.
      BL(60)=0.
      BL(68)=0
      BL(62)=0.
      BL(51)=0.
      BL(48)=0.
      DO 20 J=2,NLYY
      DO 20 N=2,NXRR
      IN=NLY*(N-1)+J
      IF(HX(IN).EQ.0.) GO TO 20
      JM1=IN-1
      JP1=IN+1
      NM1=IN-NLY
      NP1=IN+NLY
      VOL=DXR(N)*DELZ(J)
      IF(RAD)VOL=PI2*RX(N)*DXR(N)*DELZ(J)
C
C     SUM CHANGE IN STORAGE
C
      GSF=VOL*(THETA(IN)-THLST(IN))
      JJ=JTEX(IN)
      SS=HK(JJ,2)/HK(JJ,3)
      GSS=VOL*THETA(IN)*SS
      BL(29)=BL(29)+(GSF+GSS*(P(IN)-PXXX(IN)))
      IF(TRANS) THEN
C
C   FOR TRANSPORT SUM CHANGE IN STORAGE AND DIFFUSIVE/DISPERSIVE
C   FLUXES
C
      IF(NCTYP(IN).NE.1) BL(68)=BL(68)+VOL*(
     *CC(IN)*THETA(IN)*(1+SS*P(IN))-COLD(IN)*THLST(IN)*(1+SS*PXXX(IN)))
      SS=-HT(JJ,4)*(THETA(IN)+THETA(IN)*P(IN)*SS+RET(IN))*DELT
      BL(62)=BL(62)+VOL*SS*CC(IN)
      BLTEMP=BLTEMP-RET(IN)*CC(IN)*VOL
```

```
      IF(NCTYP(IN).EQ.2) THEN
      IF(CS(IN).LT.0) THEN
      BL(51)=BL(51)+CS(IN)
      ELSE
      BL(48)=BL(48)+CS(IN)
      END IF
      END IF
      IF(NCTYP(IN).EQ.1) THEN
      IP2=NP1-1
      IM2=NM1+1
      IM3=NM1-1
      IP3=NP1+1
      T5=(DX1(NP1)*(CC(IN)-CC(NP1))-DX2(NP1)*(0.5)*(
     &CC(JP1)-CC(JM1)+CC(IP3)-CC(IP2)))
     &+(DX1(IN)*(CC(IN)-CC(NM1))+DX2(IN)*(0.5)*(
     &CC(JP1)-CC(JM1)+CC(IM2)-CC(IM3)))
     &+(DZ1(JP1)*(CC(IN)-CC(JP1))-DZ2(JP1)*(0.5)*(
     &CC(NP1)-CC(NM1)+CC(IP3)-CC(IM2)))
     &+(DZ1(IN)*(CC(IN)-CC(JM1))+DZ2(IN)*(0.5)*(
     &CC(NP1)-CC(NM1)+CC(IP2)-CC(IM3)))
      IF(T5.LT.0) THEN
      BL(51)=BL(51)+T5
      ELSE
      BL(48)=BL(48)+T5
      END IF
      END IF
      END IF
C
C  FLUX FOR NEUMANN CELLS
C
      IF(NTYP(IN).EQ.2) THEN
      IF(QQ(IN).LE.0) THEN
      BL(12)=BL(12)+QQ(IN)
      IF(TRANS) BL(45)=BL(45)+QQ(IN)*CC(IN)
      ELSE
      BL(9)=BL(9)+QQ(IN)
      IF(TRANS) BL(42)=BL(42)+QQ(IN)*CS(IN)
      END IF
      ELSE
C
C   FLUX FOR DIRICHLET CELLS
C
      IF(NTYP(IN).EQ.1) THEN
      IF(TRANS) THEN
      QX=QT(IN)
      ELSE
      QX=VSFLX1(IN)
      END IF
      IF(QX.LT.0) THEN
      BL(3)=BL(3)-QX
      IF(TRANS) BL(36)=BL(36)-QX*CS(IN)
      ELSE
      BL(6)=BL(6)-QX
      IF(TRANS) BL(39)=BL(39)-QX*CC(IN)
      END IF
      ELSE
C
C    SUM SOURCES AND SINKS
C
      BL(27)=BL(27)+Q(IN)
      IF(TRANS) BL(60)=BL(60)+ETOUT*CC(IN)
      END IF
      END IF
   20 CONTINUE
C
C   ACCUMULATE VALUES FOR TOTAL ELAPSED SIMULATION TIME
C
```

```
      BL(24)=ETOUT
      BL(21)=ETOUT1
      BL(30)=BL(29)/DELT
      BL(15)=BL(3)+BL(9)
      BL(18)=BL(6)+BL(12)
      DO 30 I=2,26,3
      BL(I)=DELT*BL(I+1)
   30 CONTINUE
      BL(19)=BL(19)+BL(20)
      BL(22)=BL(22)+BL(23)
      BL(1)=BL(1)+BL(2)
      BL(4)=BL(4)+BL(5)
      BL(10)=BL(10)+BL(11)
      BL(13)=BL(13)+BL(14)
      BL(7)=BL(7)+BL(8)
      BL(16)=BL(16)+BL(17)
      BL(25)=BL(25)+BL(26)
      BL(28)=BL(28)+BL(29)
      BL(32)=BL(14)+BL(17)+BL(26)-BL(29)
      BL(33)=BL(32)/DELT
      BL(31)=BL(31)+BL(32)
      IF(TRANS) THEN
C
C   TRANSPORT MASS BALANCE COMPONENTS
C
      BL(67)=BL(67)+BL(68)
      BL(69)=BL(68)/DELT
      BL(61)=BL(61)+BL(62)
      BL(65)=BLTEMP-BL(64)
      BL(64)=BLTEMP
      BL(63)=BL(62)/DELT
      BL(66)=BL(65)/DELT
      BL(54)=BL(36)+BL(42)+BL(48)
      DO 40 I=35,59,3
      BL(I)=DELT*BL(I+1)
   40 CONTINUE
      BL(49)=BL(49)+BL(50)
      BL(46)=BL(46)+BL(47)
      BL(57)=BL(39)+BL(45)+BL(51)
      BL(58)=BL(58)+BL(59)
      BL(34)=BL(34)+BL(35)
      BL(37)=BL(37)+BL(38)
      BL(43)=BL(43)+BL(44)
      BL(52)=BL(52)+BL(53)
      BL(40)=BL(40)+BL(41)
      BL(55)=BL(55)+BL(56)
      BL(71)=BL(53)+BL(56)+BL(59)-BL(68)+BL(62)+BL(65)
      BL(72)=BL(71)/DELT
      BL(70)=BL(70)+BL(71)
      END IF
C
C   WRITE RESULTS TO FILE 9
C
      IF(F9P) WRITE(09,4000) STIM,(BL(MB9(IM)),IM=1,NMB9)
      IF(.NOT.F6P.AND.JPLT.NE.1.AND.JSTOP.NE.1.AND.JFLAG.NE.1) GO TO 50
C
C    WRITE RESULTS OF MASS BALANCE TO FILE 6
C
      WRITE (06,4010) KTIM,KP,STIM,TUNIT,ZUNIT,ZUNIT,ZUNIT,TUNIT,(BL(M),
     *M=1,12)
      WRITE(06,4020) (BL(M),M=13,33)
      IF(TRANS) WRITE(06,4030) CUNX,CUNX,CUNX,TUNIT,(BL(M),M=34,72)
      WRITE(06,4040)
   50 CONTINUE
      RETURN
 4000 FORMAT(11(1PE11.3))
 4010 FORMAT(21X,10(1H-),1X,'MASS BALANCE SUMMARY FOR TIME STEP',
```

```
     & I4,1X,10(1H-)/25X,'PUMPING PERIOD NUMBER ',I4/25X,
     &'TOTAL ELAPSED SIMULATION TIME = ',1PE10.3,1X,A4//2X,128(1H+)/
     & 2X,'+',126X,'+'/
     &2X,'+',90X,' TOTAL THIS',11X,'RATE THIS',5X,'+'/2X,'+',
     &33X,'VOLUMETRIC FLOW BALANCE',
     &18X,'TOTAL   ',9X,'TIME STEP',11X,'TIME STEP',6X,'+'/
     &2X,'+',72X,A4,'**3',13X,A4,'**3/',A4,4X,'+'/
     &2X,'+',4X,'FLUX INTO DOMAIN ACROSS SPECIFIED PRESSURE HEAD',
     &1X,'BOUNDARIES --  ',2(1PE15.5,5X),1PE15.5,4X,'+'/
     &2X,'+',2X,'FLUX OUT OF DOMAIN ACROSS SPFCIFIED PRESSURE HEAD',
     &1X,'BOUNDARIES --  ',2(1PE15.5,5X),1PE15.5,4X,'+'/
     &2X,'+',13X,'FLUX INTO DOMAIN ACROSS SPECIFIED FLUX BOUNDARIES',
     &1X,'--  ',2(1PE15.5,5X),1PE15.5,4X,'+'/
     &2X,'+',11X,'FLUX OUT OF DOMAIN ACROSS SPECIFIED FLUX',
     &1X,'BOUNDARIES --  ',2(1PE15.5,5X),1PE15.5,4X,'+')
4020 FORMAT(1H ,1X,'+',40X,'TOTAL FLUX INTO DOMAIN --  ',2(1PE15.5,5X),
     & 1PE15.5,4X,'+'/2X,'+',38X,'TOTAL FLUX OUT OF DOMAIN --  ',
     &2(1PE15.5,5X),1PE15.5,4X,'+'/
     &2X,'+',51X,'EVAPORATION --  ',2(1PE15.5,5X),1PE15.5,4X,'+'/
     &2X,'+',49X,'TRANSPIRATION --  ',2(1PE15.5,5X),1PE15.5,4X,'+'/
     &2X,'+',38X,'TOTAL EVAPOTRANSPIRATION',
     &1X,'--  ',2(1PE15.5,5X),1PE15.5,4X,'+'/
     &2X,'+',30X,'CHANGE IN FLUID STORED IN DOMAIN --  ',
     &2(1PE15.5,5X),1PE15.5,4X,'+'/2X,'+',42X,'FLUID VOLUME BALANCE'
     &,1X,'--  ',2(1PE15.5,5X),1PE15.5,4X,'+'/2X,'+',126X,'+')
4030 FORMAT(2X,'+'126X,'+',/,2X,'+',35X,'SOLUTE MASS BALANCE',
     &72X,'+',/,2X,'+',74X,A4,16X,A4,14X,A4,'/',A4,5X,'+',/,
     &2X,'+',4X,'FLUX INTO DOMAIN ACROSS SPECIFIED PRESSURE HEAD',
     &1X,'BOUNDARIES --  ',2(1PE15.5,5X),1PE15.5,4X,'+'/
     &2X,'+',2X,'FLUX OUT OF DOMAIN ACROSS SPECIFIED PRESSURE HEAD',
     &1X,'BOUNDARIES --  ',2(1PE15.5,5X),1PE15.5,4X,'+'/
     &2X,'+',13X,'FLUX INTO DOMAIN ACROSS SPECIFIED FLUX BOUNDARIES',
     &1X,'--  ',2(1PE15.5,5X),1PE15.5,4X,'+'/
     &2X,'+',11X,'FLUX OUT OF DOMAIN ACROSS SPECIFIED FLUX',
     &1X,'BOUNDARIES --  ',2(1PE15.5,5X),1PE15.5,4X,'+'/,
     &2X,'+',25X,'DIFFUSIVE/DISPERSIVE FLUX INTO DOMAIN --  ',
     &2(1PE15.5,5X),1PE15.5,4X,'+'/2X,
     &'+',23X,'DIFFUSIVE/DISPERSIVE FLUX OUT OF DOMAIN --  ',
     &2(1PE15.5,5X),1PE15.5,4X,'+'/,
     &1H ,1X,'+',40X,'TOTAL FLUX INTO DOMAIN --  ',2(1PE15.5,5X),
     & 1PE15.5,4X,'+'/2X,'+',38X,'TOTAL FLUX OUT OF DOMAIN --  ',
     &2(1PE15.5,5X),1PE15.5,4X,'+'/
     &2X,'+',38X,'TOTAL EVAPOTRANSPIRATION',
     &1X,'--  ',2(1PE15.5,5X),1PE15.5,4X,'+'/
     &2X,'+',45X,'FIRST ORDER DECAY --  ',2(1PE15.5,5X),
     &1PE15.5,4X,'+'/
     &2X,'+',39X,'ADSORPTION/ION EXCHANGE --  ',2(1PE15.5,5X),
     &1PE15.5,4X,'+'/
     &2X,'+',29X,'CHANGE IN SOLUTE STORED IN DOMAIN --  ',
     &2(1PE15.5,5X),1PE15.5,4X,'+'/2X,'+',43X,'SOLUTE MASS BALANCE'
     &,1X,'--  ',2(1PE15.5,5X),1PE15.5,4X,'+'/2X,'+',126X,'+')
4040 FORMAT( 2X,128(1H+))
      END
      DOUBLE PRECISION FUNCTION VSFLX1(IN)
C******
CVSFLX1
C******
C   PURPOSE: TO COMPUTE INTERCELL MASS FLUX RATES FOR DIRICHLET
C   BOUNDARY NODES
C ---------------------------------------------------------------
C
C   SPECIFICATIONS FOR ARRAYS AND SCALARS
C
      IMPLICIT DOUBLE PRECISION (A-H,P-Z)
      COMMON/RSPAC/DELZ(600),DZZ(600),DXR(600),RX(600),PI2
      COMMON/ISPAC/NLY,NLYY,NXR,NXRR,NNODES
      COMMON/KCON/HX(1600),NTYP(1600)
```

```
      COMMON/PRESS/P(1600),PXXX(1600),CS1,CS2
      COMMON/DISCH/Q(1600),QQ(1600),ETOUT,ETOUT1
      COMMON/HCON/HCND(1600),HKLL(1600),HKTT(1600)
      COMMON/EQUAT/A(1600),B(1600),C(1600),D(1600),E(1600),RHS(1600),
     &XI(1600)
      COMMON/WGT/WUS,WDS
      LOGICAL RAD,BCIT,ETSIM,SEEP,ITSTOP,CIS,CIT
      COMMON/LOG1/RAD,BCIT,ETSIM,SEEP,ITSTOP,CIS,CIT
C-----------------------------------------------------------------------
C
C
C   COMPUTE FLUXES ON ALL FOUR SIDES OF EACH CONSTANT HEAD NODE
C
      JM1=IN-1
      JP1=IN+1
      NP1=IN+NLY
      NM1=IN-NLY
C
C   COMPUTE A,B,C,D
C
      IF(WUS.EQ.0.) THEN
      A(IN)=HKLL(IN)*DSQRT(HCND(NM1)*HCND(IN))
      B(IN)=HKTT(IN)*DSQRT(HCND(JM1)*HCND(IN))
      C(IN)=HKLL(NP1)*DSQRT(HCND(NP1)*HCND(IN))
      D(IN)=HKTT(JP1)*DSQRT(HCND(JP1)*HCND(IN))
      ELSE
      IF(P(NM1).GT.P(IN).AND.HX(NM1).NE.0.) THEN
      ALA=WDS
      BTA=WUS
      ELSE
      ALA=WUS
      BTA=WDS
      END IF
      IF(P(JM1).GT.P(IN).AND.HX(JM1).NE.0.) THEN
      ALB=WDS
      BTB=WUS
      ELSE
      ALB=WUS
      BTB=WDS
      END IF
      IF(P(NP1).GT.P(IN).AND.HX(NP1).NE.0.) THEN
      ALC=WDS
      BTC=WUS
      ELSE
      ALC=WUS
      BTC=WDS
      END IF
      IF(P(JP1).GT.P(IN).AND.HX(JP1).NE.0.) THEN
      ALD=WDS
      BTD=WUS
      ELSE
      ALD=WUS
      BTD=WDS
      END IF
C
C   DETERMINE FLUXES
C
      A(IN)=(ALA*HCND(NM1)+BTA*HCND(IN))*HKLL(IN)
      B(IN)=(ALB*HCND(JM1)+BTB*HCND(IN))*HKTT(IN)
      C(IN)=(ALC*HCND(NP1)+BTC*HCND(IN))*HKLL(NP1)
      D(IN)=(ALD*HCND(JP1)+BTD*HCND(IN))*HKTT(JP1)
      END IF
   10 QL=-A(IN)*(P(IN)-P(NM1))
      QA=-B(IN)*(P(IN)-P(JM1))
      QR=-C(IN)*(P(IN)-P(NP1))
      QB=-D(IN)*(P(IN)-P(JP1))
C
```

```
C     COMPUTE NET FLUX IN (+) OR OUT (-)
C
      VSFLX1=QL+QR+QA+QB
      RETURN
      END
      SUBROUTINE VSOUTP
C******
CVSOUTP
C******
C
C    PURPOSE: TO OUTPUT RESULTS AFTER EACH TIME STEP.
C
C-----------------------------------------------------------------------
C
C        SPECIFICATIONS FOR ARRAYS AND SCALARS
C
      IMPLICIT DOUBLE PRECISION(A-H,P-Z)
      COMMON/RSPAC/DELZ(600),DZZ(600),DXR(600),RX(600),PI2
      COMMON/ISPAC/NLY,NLYY,NXR,NXRR,NNODES
      COMMON/KCON/HX(1600),NTYP(1600)
      COMMON/RPROP/HK(10,100),HT(10,20),ANIZ(10)
      COMMON/MPROP/THETA(1600),THLST(1600)
      COMMON/PRESS/P(1600),PXXX(1600),CS1,CS2
      COMMON/DISCH/Q(1600),QQ(1600),ETOUT,ETOUT1
      COMMON/JTXX/JTEX(1600)
      COMMON/DUMM/DUM(1600)
      COMMON/PLOTT/PLTIM(50),IJOBS(50),JPLT,NPLT,NOBS
      COMMON/SCON/DHMX(201),DELT,HMAX,TMAX,EPS,NUMT,ITMAX,MINIT,ITEST
      COMMON/SCN1/TMPX,TMLT,DLTMX,DLTMIN,TRED
      COMMON/TCON/STIM,DSMAX,KTIM,NIT,NIT1,KP
      COMMON/JCON/JSTOP,JFLAG
      COMMON/TRXX/DX1(1600),DX2(1600),DZ1(1600),DZ2(1600),VX(1600),
     &VZ(1600),CC(1600),COLD(1600),CS(1600),QT(1600),NCTYP(1600),
     &RET(1600)
      LOGICAL TRANS,TRANS1,SORP,SSTATE
      COMMON/TRXY/MB9(72),NMB9,EPS1,TRANS,TRANS1,SORP,SSTATE
      LOGICAL F7P,F11P,F8P,F9P,F6P,PRNT
      LOGICAL THPT,SPNT,PPNT,HPNT,VPNT
      COMMON/LOG2/F7P,F11P,F8P,F9P,F6P,PRNT
      COMMON/LOG4/THPT,SPNT,PPNT,HPNT,VPNT
      CHARACTER*80 TITL
      CHARACTER*4 ZUNIT,TUNIT,CUNX
      COMMON/SCHAR/TITL,ZUNIT,TUNIT,CUNX
C
C-----------------------------------------------------------------------
C
C     OUTPUT RESULTS TO FILE 11 AT EACH TIME STEP
C
      IF(F11P) THEN
      DO 10 J=1,NOBS
      N=IJOBS(J)
      I=N/NLY+1
      J1=MOD(N,NLY)
      IF(HX(N).NE.0.) THEN
      PPR=HK(JTEX(N),3)
      IF(PPR.EQ.0.)PPR=1.
      SAT=THETA(N)/PPR
      IF(CS1.EQ.1.) THEN
      Z1=DZZ(J1)
      ELSE
      Z1=DZZ(J1)*CS1+(RX(I))*CS2
      END IF
      PHD=P(N)+Z1
      IF(TRANS) THEN
      WRITE(11,4020) STIM,RX(I),DZZ(J1),P(N),PHD,THETA(N),SAT,CC(N)
      ELSE
      WRITE (11,4020) STIM,RX(I),DZZ(J1),P(N),PHD,THETA(N),SAT
```

```
      END IF
      END IF
   10 CONTINUE
      END IF
      IF(KTIM.EQ.0)  GO TO 20
C
C    WRITE TIME STEP HEADER TO FILE 6
C
C    WRITE MAXIMUM HEAD CHANGE EACH TIME STEP TO FILE 7
C
      IF(F7P) THEN
      WRITE(07,4040) KTIM,STIM,NIT,NIT1
      WRITE(07,4030) (DHMX(M2),M2=1,NIT)
      END IF
      WRITE(06,4040) KTIM,STIM,NIT,NIT1
      IF(JSTOP.EQ.1.OR.JPLT.EQ.1) GO TO 20
      IF(.NOT.PRNT.AND.JFLAG.EQ.0) RETURN
   20 WRITE (6,4050) TITL,STIM,TUNIT,KTIM
C
C    PRINT SOLUTION FOR CURRENT TIME STEP
C
      IF(JPLT.EQ.1) THEN
C
C   WRITE PRESSURE HEADS TO FILE 8 AT OBSERVATION TIMES.
C
      WRITE (8,4000) STIM,TUNIT
      DO 40 J=1,NLY
      DO 30 N=1,NXR
      IN=NLY*(N-1)+J
      IF(CS1.EQ.1.) THEN
      Z1=DZZ(J)
      ELSE
      Z1=DZZ(J)*CS1+(RX(N))*CS2
      END IF
   30 DUM(IN)=P(IN)+Z1
   40 WRITE(8,4010) (DUM(N),N=J,NNODES-NLY+J,NLY)
C
C  WRITE CONCENTRATIONS TO FILE 8
C
      IF(TRANS) THEN
      DO 50 J=1,NLY
      WRITE(08,4010) (CC(N),N=J,NNODES-NLY+J,NLY)
   50 CONTINUE
      END IF
      END IF
C
C    PRINT TOTAL HEADS
C
      IF(HPNT) THEN
      WRITE (6,4060)
      CALL VSOUT(1,P)
      END IF
C
C  PRINT PRESSURE HEADS
C
      IF(PPNT) THEN
      IF(JPLT.NE.1) THEN
      DO 60 J=2,NLYY
      DO 60 N=2,NXRR
      IN=NLY*(N-1)+J
      IF(CS1.EQ.1.) THEN
      Z1=DZZ(J)
      ELSE
      Z1=DZZ(J)*CS1+(RX(N))*CS2
      END IF
      DUM(IN)=P(IN)+Z1
      IF(HX(IN).EQ.0.)DUM(IN)=0.
```

```
   60 CONTINUE
      END IF
      WRITE (6,4070)
      CALL VSOUT(1,DUM)
      END IF
C
C  PRINT SATURATIONS
C
      IF(SPNT) THEN
      DO 70 J=2,NLYY
      DO 70 N=2,NXRR
      IN=NLY*(N-1)+J
      TTX=HK(JTEX(IN),3)
      IF(TTX.EQ.0.) THEN
      DUM(IN)=0.
      ELSE
      DUM(IN)=THETA(IN)/TTX
      END IF
   70 CONTINUE
      WRITE (6,4080)
      CALL VSOUT(2,DUM)
      END IF
C
C  PRINT MOISTURE CONTENTS
C
      IF(THPT) THEN
      WRITE (6,4090)
      CALL VSOUT(2,THETA)
      END IF
C
C  PRINT VELOCITIES
C
      IF(VPNT.AND.KTIM.GT.0) THEN
      WRITE(06,4100)
      CALL VSOUT(1,VX)
      WRITE(06,4110)
      CALL VSOUT(1,VZ)
      END IF
C
C  PRINT CONCENTRATIONS
C
      IF(TRANS) THEN
      WRITE(6,4120)
      CALL VSOUT(1,CC)
      END IF
      CONTINUE
      RETURN
 4000 FORMAT(/,8H TIME = ,E14.4,1X,A4/)
 4010 FORMAT(8(1PE10.3))
 4020 FORMAT(8(1PE12.3))
 4030 FORMAT(7E11.4)
 4040 FORMAT(' TIME STEP ',I5,'  TIME = ',E12.4,'  NIT = ',I3,
     &'  NIT1 = ',I3)
 4050 FORMAT(6X,A80/5X,20HTOTAL ELAPSED TIME =,1PE12.3,1X,A4/5X,
     &10HTIME STEP ,I5,//)
 4060 FORMAT(1H ,50X,10HTOTAL HEAD)
 4070 FORMAT(1H ,50X,13HPRESSURE HEAD)
 4080 FORMAT(1H ,50X,10HSATURATION)
 4090 FORMAT(1H ,50X,16HMOISTURE CONTENT)
 4100 FORMAT(51X,'X-VELOCITY')
 4110 FORMAT(51X,'Z-VELOCITY')
 4120 FORMAT(51X,'CONCENTRATION')
      END
      SUBROUTINE VSOUT(IV,VPRNT)
C*****
CVSOUT
C*****
```

```
C
C     PURPOSE: TO PRINT TWO DIMENSIONAL ARRAYS
C
C
C-----------------------------------------------------------------------
C
C    SPECIFICATIONS FOR ARRAYS AND SCALARS
C
C
      IMPLICIT DOUBLE PRECISION (A-H,P-Z)
      COMMON/RSPAC/DELZ(600),DZZ(600),DXR(600),RX(600),PI2
      COMMON/ISPAC/NLY,NLYY,NXR,NXRR,NNODES
      COMMON/KCON/HX(1600),NTYP(1600)
      COMMON/DUMM/DUM(1600)
      COMMON/PLOTT/PLTIM(50),IJOBS(50),JPLT,NPLT,NOBS
      LOGICAL F7P,F11P,F8P,F9P,F6P,PRNT
      COMMON/LOG2/F7P,F11P,F8P,F9P,F6P,PRNT
      CHARACTER*80 TITL
      CHARACTER*4 ZUNIT,TUNIT,CUNX
      COMMON/SCHAR/TITL,ZUNIT,TUNIT,CUNX
      DIMENSION VPRNT(1),DUM1(600)
C
C-------------------------------------------------------------------
C
      WRITE (06,4000) ZUNIT,ZUNIT
      WRITE (06,4010) (RX(K),K=2,NXRR)
      DO 30 J=2,NLYY
      DO 10 N=2,NXRR
      IN=NLY*(N-1)+J
      DUM1(N)=VPRNT(IN)
      IF(HX(IN).EQ.0. ) DUM1(N)=0.
   10 CONTINUE
      IF(IV.GT.1) GO TO 20
      WRITE (06,4020) DZZ(J),(DUM1(N),N=2,NXRR)
      GO TO 30
   20 WRITE (06,4030) DZZ(J),(DUM1(N),N=2,NXRR)
   30 CONTINUE
      RETURN
 4000 FORMAT(1H ,1X,5HZ, IN/2X,A4,20X,20HX OR R DISTANCE, IN ,A4)
 4010 FORMAT(1H ,8X,13(F9.2)/(9X,13(F9.2)))
 4020 FORMAT(1X,F8.2,13(1PE9.2)/(9X,13(1PE9.2)))
 4030 FORMAT(1X,F8.2,13F9.3/(9X,13F9.3))
      END
      SUBROUTINE VSPOND(IFET,IFET1,IFET2)
C******
CVSPOND
C******
C
C  UPDATED 10-88
C
C     PURPOSE:  TO DETERMINE IF PONDING OR UNPONDING HAS OCCURRED, AND
C               IF SO TO CHANGE BOUNDARY CONDITIONS AT THOSE NODES FROM
C               NEUMAN TO DIRICHLET OR VICE VERSA
C
C ------------------------------------------------------------------
C
C    SPECIFICATIONS FOR ARRAYS AND SCALARS
C
      IMPLICIT DOUBLE PRECISION (A-H,P-Z)
      COMMON/RSPAC/DELZ(600),DZZ(600),DXR(600),RX(600),PI2
      COMMON/ISPAC/NLY,NLYY,NXR,NXRR,NNODES
      COMMON/KCON/HX(1600),NTYP(1600)
      COMMON/PRESS/P(1600),PXXX(1600),CS1,CS2
      COMMON/DISCH/Q(1600),QQ(1600),ETOUT,ETOUT1
      COMMON/EQUAT/A(1600),B(1600),C(1600),D(1600),E(1600),RHS(1600),
     &XI(1600)
      COMMON/PND/POND
```

```
      COMMON/TCON/STIM,DSMAX,KTIM,NIT,NIT1,KP
C
C------------------------------------------------------------------
C
C   IFET1 INDICATES WHETHER THERE ARE ANY NEUMAN BOUNDARIES REMAINING
C   IFET2 INDICATES WHETHER ANY SPECIFIC FLUX NODES HAVE BEEN CONVERTED
C     TO SPECIFIED HEAD NODES.  BECAUSE OF THE CAPILLARY BARRIER
C     EFFECT, THESE NODES MAY NEED TO REVERT TO SPECIFIED FLUX NODES.
C   IFET INDICATES WHETHER PONDING OCCURRED OR DISAPPEARED
C
      IF(IFET1.EQ.0 .AND. IFET2 .EQ. 0) RETURN
      IFET=0
      IFET1=0
      IFET2=0
      IF(CS1.EQ.1.) THEN
      DZ1=DZZ(2)
      ELSE
      IF(CS2.LT.0) THEN
      DZ1=DZZ(2)*CS1+RX(NXRR)*CS2
      ELSE
      DZ1=DZZ(2)*CS1+RX(2)*CS2
      END IF
      END IF
      DO 20 I=2,NXRR
      DO 10 J=2,NLYY
      IN=NLY*(I-1)+J
      IF(HX(IN).NE.0.) THEN
      IF(NTYP(IN).EQ.2.AND.QQ(IN).GT.0.) THEN
      IFET1=1
      IF(CS1.EQ.1.) THEN
      Z1=DZZ(J)
      ELSE
      Z1=DZZ(J)*CS1+RX(I)*CS2
      END IF
      IF(POND.GE.0.) THEN
C
C    DZ2 IS MAXIMUM ALLOWABLE TOTAL HEAD
C
      DZ2=POND-Z1
      ELSE
      DZ2=-DMIN1(Z1,DZ1-POND)
      END IF
      IF(P(IN).GT.DZ2) THEN
C
C    IF COMPUTED HEAD EXCEEDS MAXIMUM THEN SET P=DZ2
C    AND CHANGE BOUNDARY TYPE TO CONSTANT HEAD
C
      P(IN)=DZ2
      NTYP(IN)=1
      IFET=1
      IFET2=1
      WRITE(6,4000) J,I,KTIM,NIT
      END IF
      ELSE
      IF(NTYP(IN).EQ.1.AND.QQ(IN).GT.0.) THEN
      IFET2=1
      JP1=IN+1
      IM1=IN+NLY
      IP1=IN-NLY
      TEST=(P(IN)-P(JP1))*D(IN)
      IF(HX(IM1).NE.0) TEST=TEST+(P(IN)-P(IM1))*C(IN)
      IF(HX(IP1).NE.0)TEST=TEST+(P(IN)-P(IP1))*A(IN)
      TEST=TEST/QQ(IN)
      IF (TEST .GE. 1.01)THEN
C
C  IF FLUX FROM THE CONVERTED NODE IS GREATER THAN THE SPECIFIED
C  FLUX RATE, THE NODE IS RECONVERTED TO A SPECIFIED FLUX NODE.
```

```
C
      NTYP(IN)=2
      IFET=1
      IFET1=1
      WRITE(06,4010)J,I,KTIM,NIT
      END IF
      END IF
      END IF
      GO TO 20
      END IF
   10 CONTINUE
   20 CONTINUE
      RETURN
 4000 FORMAT(//,6X,17H PONDING_AT NODE ,2I4,17H DURING TIME STEP,
     &I4,'  ITERATION ',I4)
 4010 FORMAT(//,6X,' PONDING ENDED AT NODE ',2I4,
     &' DURING TIME STEP ',I4,'  ITERATION ',I4)
      END
      SUBROUTINE VSSFAC
C******
CVSSFAC
C******
C
C  REVISED 10-88
C
C    PURPOSE:  TO COMPUTE POSITION OF SEEPAGE FACE BOUNDARIES
C
C     HEIGHT OF SEEPAGE FACE IS LOWERED IF THERE IS FLUX INTO SYSTEM
C     THRU FACE.
C     HEIGHT IS RAISED IF PRESSURE HEADS ARE POSITIVE ABOVE FACE.
C
C ----------------------------------------------------------------
C
C   SPECIFICATIONS FOR ARRAYS AND SCALARS
C
      IMPLICIT DOUBLE PRECISION (A-H,P-Z)
      COMMON/RSPAC/DELZ(600),DZZ(600),DXR(600),RX(600),PI2
      COMMON/ISPAC/NLY,NLYY,NXR,NXRR,NNODES
      COMMON/KCON/HX(1600),NTYP(1600)
      COMMON/PRESS/P(1600),PXXX(1600),CS1,CS2
      COMMON/HCON/HCND(1600),HKLL(1600),HKTT(1600)
      COMMON/SPFC/JSPX(3,25,8),NFC(8),JLAST(8),NFCS
      COMMON/TCON/STIM,DSMAX,KTIM,NIT,NIT1,KP
C
C------------------------------------------------------------------
C
      DO 90 K=1,NFCS
      NFX=NFC(K)
      JFST=0
      JLST=JLAST(K)
C
C   CHECK FOR POSITIVE PRESSURES ABOVE SEEPAGE FACE
C
      DO 10 J=NFX,1,-1
      IN=JSPX(1,J,K)
      JJ=JSPX(2,J,K)
      NN=JSPX(3,J,K)
      IF(CS1.EQ.1) THEN
      Z1=DZZ(JJ)
      ELSE
      Z1=DZZ(JJ)*CS1+RX(NN)*CS2
      END IF
      PTMP=P(IN)+Z1
      IF(PTMP.LT.0.) GO TO 10
      JFST=J
      GO TO 20
   10 CONTINUE
```

```
   20 CONTINUE
C
C   CHECK FOR FLOW INTO DOMAIN THROUGH SEEPAGE FACE
C
      IF(JFST.GT.JLST) GO TO 60
      DO 40 I=JLST,1,-1
      IN=JSPX(1,I,K)
      IM1=IN-NLY
      JM1=IN-1
      IP1=IN+NLY
      JP1=IN+1
      IF(HX(IM1).EQ.0) THEN
      IF(HX(IP1).NE.0.AND.P(IP1).LT.P(IN)) GO TO 30
      END IF
      IF(HX(JM1).EQ.0) THEN
      IF(HX(JP1).NE.0.AND.P(JP1).LT.P(IN)) GO TO 30
      END IF
      IF(HX(IP1).EQ.0) THEN
      IF(HX(IM1).NE.0.AND.P(IM1).LT.P(IN)) GO TO 30
      END IF
      IF(HX(JP1).EQ.0) THEN
      IF(HX(JM1).NE.0.AND.P(JM1).LT.P(IN)) GO TO 30
      END IF
      GO TO 50
   30 NTYP(IN)=3
   40 CONTINUE
      I=0
   50 IF(I.EQ.JLST) GO TO 60
C
C   RESET SEEPAGE FACE HEIGHT AND BOUNDARIES
C
      JLAST(K)=I
      GO TO 80
   60 IF(JFST.EQ.JLST) GO TO 80
      DO 70 I=1,JFST
      IN=JSPX(1,I,K)
      JJ=JSPX(2,I,K)
      NN=JSPX(3,I,K)
      IF(CS1.EQ.1) THEN
      Z1=DZZ(JJ)
      ELSE
      Z1=DZZ(JJ)*CS1+RX(NN)*CS2
      END IF
      NTYP(IN)=1
      P(IN)=-Z1
   70 CONTINUE
      JLAST(K)=JFST
   80 CONTINUE
   90 CONTINUE
      END
      SUBROUTINE VSEVAP
C******
CVSEVAP
C******
C
C  PURPOSE: TO COMPUTE SURFACE EVAPORATION RATES
C
C
C
C-----------------------------------------------------------------
C
C   SPECIFICATIONS FOR ARRAYS AND SCALARS
C
      IMPLICIT DOUBLE PRECISION (A-H,P-Z)
      COMMON/RSPAC/DELZ(600),DZZ(600),DXR(600),RX(600),PI2
      COMMON/ISPAC/NLY,NLYY,NXR,NXRR,NNODES
      COMMON/KCON/HX(1600),NTYP(1600)
      COMMON/HCON/HCND(1600),HKLL(1600),HKTT(1600)
```

```
      COMMON/PRESS/P(1600),PXXX(1600),CS1,CS2
      COMMON/DISCH/Q(1600),QQ(1600),ETOUT,ETOUT1
      COMMON/PTET/DPTH(1600),RT(1600),RDC(6,25),ETCYC,
     &PEVAL(25),PTVAL(25),PET,PEV,HROOT,HA,SRES,RTDPTH,
     &RTBOT,RTTOP,NPV
      LOGICAL RAD,BCIT,ETSIM,SEEP,ITSTOP,CIS,CIT
      COMMON/LOG1/RAD,BCIT,ETSIM,SEEP,ITSTOP,CIS,CIT
C
C-----------------------------------------------------------------------
C
      ETOUT1=0
      IF(SRES.EQ.0) RETURN
      DO 10 J=2,NLYY
      DO 10 N=2,NXRR
      IN=NLY*(N-1)+J
      IF(NTYP(IN).EQ.5) THEN
C
C     COMPUTE TEMPORARY EVAP RATE, CHECK AGAINST MAX AND
C     CORRECT IF NECESSARY
C
      AREA=DXR(N)
      IF(RAD)AREA=PI2*RX(N)*DXR(N)
      PETT=PEV*AREA
      IF(CS1.EQ.1.) THEN
      Z1=DZZ(J)
      ELSE
      Z1=DZZ(J)*CS1+(RX(N))*CS2
      END IF
      PTMP=P(IN)+Z1
      HKX=HCND(IN)*HX(IN)
      EV=HKX*SRES*(HA-PTMP)*AREA
      IF(EV.GT.0.) EV=0.
      IF(EV.GT.PETT) THEN
      Q(IN)=EV
      ELSE
      Q(IN)=PETT
      END IF
      ETOUT1=ETOUT1+Q(IN)
      END IF
   10 CONTINUE
      RETURN
      END
      SUBROUTINE VSPLNT
C******
CVSPLNT
C******
C
C   THIS SUBROUTINE COMPUTES ACTUAL ET AS A FUNCTION OF A ROOT
C        ACTIVITY FUNCTION, HYDRAULIC CONDUCTIVITY OF THE SOIL,
C        AND THE DIFFERENCE IN PRESSURE HEAD BETWEEN THE ROOTS AND
C        THE SOIL
C
C ----------------------------------------------------------------------
C
C   SPECIFICATIONS FOR ARRAYS AND SCALARS
C
      IMPLICIT DOUBLE PRECISION (A-H,P-Z)
      COMMON/RSPAC/DELZ(600),DZZ(600),DXR(600),RX(600),PI2
      COMMON/ISPAC/NLY,NLYY,NXR,NXRR,NNODES
      COMMON/KCON/HX(1600),NTYP(1600)
      COMMON/PRESS/P(1600),PXXX(1600),CS1,CS2
      COMMON/DISCH/Q(1600),QQ(1600),ETOUT,ETOUT1
      COMMON/HCON/HCND(1600),HKLL(1600),HKTT(1600)
      COMMON/PTET/DPTH(1600),RT(1600),RDC(6,25),ETCYC,
     &PEVAL(25),PTVAL(25),PET,PEV,HROOT,HA,SRES,RTDPTH,
     &RTBOT,RTTOP,NPV
      COMMON/TCON/STIM,DSMAX,KTIM,NIT,NIT1,KP
```


AR 020620

```
      LOGICAL RAD,BCIT,ETSIM,SEEP,ITSTOP,CIS,CIT
      COMMON/LOG1/RAD,BCIT,ETSIM,SEEP,ITSTOP,CIS,CIT
C
C   SUM TRANSPIRATION FOR EACH COLUMN
C
      ETOUT=0
      IF(PET.GE. 0)RETURN
      DO 50 I=2,NXRR
      ETR=0
      AREA=DXR(I)
      IF (RAD) AREA=PI2*RX(I)*DXR(I)
      PETT=AREA*PET
      DO 10 J=2,NLYY
C
C   COMPUTE TRANSPIRATION FOR EACH NODE IN COLUMN
C
      IN=NLY*(I-1)+J
      IF(NTYP(IN).EQ.0.AND.HX(IN).GT.0) THEN
      VOL=AREA*DELZ(J)
      IF(DPTH(IN).GT.RTDPTH) GO TO 20
C
C   TRANSPIRATION IS ZERO IF NTYP IS NOT 0, NODE IS DEEPER
C   THAN RTDPTH, OR PRESSURE IS LESS THAN HROOT
C
      IF(CS1.EQ.1.) THEN
      Z1=DZZ(J)
      ELSE
      Z1=DZZ(J)*CS1+(RX(I))*CS2
      END IF
      PTMP=P(IN)+Z1
      IF(PTMP.LE.HROOT) THEN
      Q(IN)=0
      ELSE
      HXX=HCND(IN)*HX(IN)*RT(IN)*VOL
C
C   Q IS TRANSPIRATION FOR EACH NODE.  ETR IS TOTAL FOR COLUMN
C
      Q(IN)=(HROOT-PTMP)*HXX
      ETR=ETR+Q(IN)
      END IF
      END IF
   10 CONTINUE
   20 IF(ETR.LT.PETT) THEN
C
C   IF TOTAL TRANSPIRATION FOR COLUMN IS GREATER
C   THAN POTENTIAL THEN ADJUST TRANSPIRATION VALUES
C
      R1=PETT/ETR
      ETR=PETT
      DO 30 K=2,J
      IN=NLY*(I-1)+K
      IF(HX(IN).GT.0.AND.NTYP(IN).EQ.0) THEN
      IF(DPTH(IN).GT.RTDPTH) GO TO 40
      Q(IN)=Q(IN)*R1
      END IF
   30 CONTINUE
   40 CONTINUE
      END IF
      ETOUT=ETOUT+ETR
   50 CONTINUE
      RETURN
      END
      SUBROUTINE VSPET
C*****
CVSPET
C*****
C
```

```
C     PURPOSE: TO COMPUTE VALUES OF PEV,SRES,HA,PET,RTDPTH,RTBOT,RTTOP,
C              AND HROOT FOR EVAPORATION AND TRANSPIRATION CALCULATIONS.
C              VALUES ARE DETERMINED BY LINEAR INTERPOLATION IN TIME
C              BETWEEN EVAPOTRANSPIRATION PERIODS.
C
C-----------------------------------------------------------------------
C
C     SPECIFICATIONS FOR ARRAYS AND SCALARS
C
      IMPLICIT DOUBLE PRECISION (A-H,P-Z)
      COMMON/PTET/DPTH(1600),RT(1600),RDC(6,25),ETCYC,
     &PEVAL(25),PTVAL(25),PET,PEV,HROOT,HA,SRES,RTDPTH,
     &RTBOT,RTTOP,NPV
      COMMON/TCON/STIM,DSMAX,KTIM,NIT,NIT1,KP
      LOGICAL RAD,BCIT,ETSIM,SEEP,ITSTOP,CIS,CIT
      COMMON/LOG1/RAD,BCIT,ETSIM,SEEP,ITSTOP,CIS,CIT
C
C-----------------------------------------------------------------------
C
      IF (NPV.EQ.1) THEN
C
C    IF ONLY 1 PERIOD THEN ALL VALUES ARE CONSTANT
C
      IF(BCIT) THEN
      PEV=-PEVAL(1)
      SRES=RDC(1,1)
      HA=RDC(2,1)
      END IF
      IF(ETSIM) THEN
      PET=-PTVAL(1)
      RTDPTH=RDC(3,1)
      RTBOT=RDC(4,1)
      RTTOP=RDC(5,1)
      HROOT=RDC(6,1)
      END IF
      ELSE
C
C    DETERMINE WHICH PERIOD TO USE
C
      ETCYC1=NPV*ETCYC
      SITY=MOD(STIM,ETCYC1)
      I=(SITY/ETCYC)+2
      IF(I.EQ.1) THEN
      K=NPV
      ELSE
      K=I-1
      END IF
C
C    LINEARLY INTERPOLATE
C
      FRPER=(MOD(SITY,ETCYC))/ETCYC
      IF (BCIT) THEN
      PEV=-PEVAL(K)-(PEVAL(I)-PEVAL(K))*FRPER
      SRES=RDC(1,K)+(RDC(1,I)-RDC(1,K))*FRPER
      HA=RDC(2,K)+(RDC(2,I)-RDC(2,K))*FRPER
      END IF
      IF (ETSIM) THEN
      PET=-PTVAL(K)-(PTVAL(I)-PTVAL(K))*FRPER
      RTDPTH=RDC(3,K)+(RDC(3,I)-RDC(3,K))*FRPER
      RTBOT=RDC(4,K)+(RDC(4,I)-RDC(4,K))*FRPER
      RTTOP=RDC(5,K)+(RDC(5,I)-RDC(5,K))*FRPER
      HROOT=RDC(6,K)+(RDC(6,I)-RDC(6,K))*FRPER
      END IF
      END IF
      RETURN
      END
      DOUBLE PRECISION FUNCTION VSRDF(Z1,Z2)
```

```
C*****
CVSRDF
C*****
C
C    PURPOSE: TO DETERMINE THE ROOT ACTIVITY AT EACH NODE WITHIN
C             THE ROOT ZONE FOR EACH TIME STEP
C
C
C-----------------------------------------------------------------
C
      IMPLICIT DOUBLE PRECISION (A-H,P-Z)
      COMMON/PTET/DPTH(1600),RT(1600),RDC(6,25),ETCYC,
     &PEVAL(25),PTVAL(25),PET,PEV,HROOT,HA,SRES,RTDPTH,
     &RTBOT,RTTOP,NPV
C
C-----------------------------------------------------
C
C
C   LINEARLY INTERPOLATE USING DEPTH OF NODE AND MAXIMUM ROOT DEPTH
C
      IF(RTDPTH.GT.Z1.AND.RTDPTH.GT.0)THEN
      IF(RTDPTH.GE.Z1+Z2)THEN
      ZZ=Z1+0.5*Z2
      ZZ1=1.
      ELSE
      ZZ=(Z1+RTDPTH)*0.5
      ZZ1=(RTDPTH-Z1)/Z2
      END IF
      VSRDF=ZZ1*(ZZ*RTBOT+(RTDPTH-ZZ)*RTTOP)/RTDPTH
      ELSE
      VSRDF=0.0
      END IF
      RETURN
      END
      DOUBLE PRECISION FUNCTION VSDTHU(P,I,HK)
C******
CVSDTHU
C******
C
C    FIRST DERIVATIVE OF MOISTURE CONTENT AS A FUNCTION OF PRESSURE HEAD
C
C   VAN GENUCHTEN FUNCTION
C
C          HK(I,1)=SATURATED HYDRAULIC CONDUCTIVITY
C          HK(I,2)=SPECIFIC STORAGE
C          HK(I,3)=POROSITY
C          HK(I,4)=ALPHA PRIME
C          HK(I,5)=RESIDUAL MOISTURE CONTENT
C          HK(I,6)=BETA PRIME
C
      IMPLICIT DOUBLE PRECISION (A-H,P-Z)
      DIMENSION HK(10,100)
      VSDTHU=0.D0
      IF(P.GE.0.0)RETURN
      SE=HK(I,3)-HK(I,5)
      EN=HK(I,6)
      EM=2.-1./EN
      ALPH=HK(I,4)
      A=P/ALPH
      VSDTHU=-(EN-1)*SE*A**(EN-1)/(ALPH*(1+A**EN)**EM)
      RETURN
      END
      DOUBLE PRECISION FUNCTION VSTHNV(V,I,HK)
C******
CVSTHNV
C******
C
```

```
C     INITIAL UNSATURATED PRESSURE HEAD AS A FUNCTION OF VOLUMETRIC
C     MOISTURE CONTENT
C
C     VAN GENUCHTEN FUNCTION
C
      IMPLICIT DOUBLE PRECISION (A-H,P-Z)
      DIMENSION HK(10,100)
      VSTHNV=0.0
      IF(V.GE.HK(I,3)) RETURN
      IF(V.GT.HK(I,5)) GO TO 10
      WRITE(6,4000) V,I
      STOP
   10 SE=(V-HK(I,5))/(HK(I,3)-HK(I,5))
      EN=HK(I,6)
      EM=1.-1./EN
      ALPH=HK(I,4)
      VSTHNV=ALPH*(1/SE**(1/EM)-1)**(1-EM)
      RETURN
 4000 FORMAT(/,28HINITIAL MOISTURE CONTENT OF ,F7.3,49HIS LESS THAN RES
     &IDUAL MOISTURE CONTENT FOR CLASS ,I4,/,
     &14HPROGRAM HALTED)
      END
      DOUBLE PRECISION FUNCTION VSTHU(P,I,HK)
C*****
CVSTHU
C*****
C
C     MOISTURE CONTENT AS A FUNCTION OF PRESSURE HEAD
C
C   VAN GENUCHTEN FUNCTION
C
      IMPLICIT DOUBLE PRECISION (A-H,P-Z)
      DIMENSION HK(10,100)
      VSTHU=HK(I,3)
      IF(P .GE. 0.0)RETURN
      EN=HK(I,6)
      EM=-(1.-1./EN)
      A=HK(I,3)-HK(I,5)
      ALPH=HK(I,4)
      VSTHU=HK(I,5)+A*(1+(P/ALPH)**EN)**EM
      RETURN
      END
      DOUBLE PRECISION FUNCTION VSHKU(P,I,HK)
C*****
CVSHKU
C*****
C
C     RELATIVE HYDRAULIC CONDUCTIVITY WITH RESPECT TO PRESSURE HEAD
C
C    VAN GENUCHTEN FUNCTION
C
C
      IMPLICIT DOUBLE PRECISION (A-H,P-Z)
      DIMENSION HK(10,100)
      VSHKU=1.00
      IF(P.GE.0.0)RETURN
      EN=HK(I,6)
      EM=1.-1./EN
      A=P/HK(I,4)
      TOP=A**EN
      DEN=(1+TOP)**(EM/2.)
      TOP=1-TOP/A*(1+TOP)**(-EM)
      VSHKU=TOP*TOP/DEN
      RETURN
      END
C
C
```

```
C
C   NOTE -- AS LISTED HERE THE PROGRAM USES THE FUNCTIONAL RELATIONS
C           OF THE VAN GENUCHTEN FORM.
C           FUNCTIONS FOR THE THREE ALTERNATIVE RELATIONS ARE LISTED
C           BELOW.  TO USE ONE OF THESE: FIRST PLACE A 'C' (FOR COMMENT)
C           IN THE FIRST COLUMN OF EVERY LINE IN THE VAN GENUCHTEN
C           ROUTINES.  NEXT REMOVE THE COMMENT DESIGNATIONS FOR THE
C           DESIRED SET OF ROUTINES -- 'C&' FOR BROOKS-COREY
C                                       'C$' FOR HAVERKAMP
C                                       'C+' FOR TABULAR DATA
C
C&    DOUBLE PRECISION FUNCTION VSDTHU(P,I,HK)
C******
CVSDTHU
C******
C
C     FIRST DERIVATIVE OF MOISTURE CONTENT AS A FUNCTION OF PRESSURE HEAD
C
C     BROOKS AND COREY, CSU HYDROLOGY PAPER NO. 17 PP.3-4
C
C          HK(I,1)=SATURATED HYDRAULIC CONDUCTIVITY
C          HK(I,2)=SPECIFIC STORAGE
C          HK(I,3)=POROSITY
C          HK(I,4)=BUBBLING PRESSURE
C          HK(I,5)=RESIDUAL MOISTURE CONTENT
C          HK(I,6)=LAMBDA
C
C&    IMPLICIT DOUBLE PRECISION (A-H,P-Z)
C&    DIMENSION HK(10,100)
C&    VSDTHU=0.D0
C&    IF(P.GE.HK(I,4))RETURN
C&    VSDTHU=-((HK(I,3)-HK(I,5))*HK(I,6)*(HK(I,4)/P)**HK(I,6))/P
C&    RETURN
C&    END
C&    DOUBLE PRECISION FUNCTION VSTHNV(V,I,HK)
C******
CVSTHNV
C******
C
C     INITIAL UNSATURATED PRESSURE HEAD AS A FUNCTION OF VOLUMETRIC
C     MOISTURE CONTENT
C
C     BROOKS AND COREY, CSU HYDROLOGY PAPER NO. 17 , PP.3-4
C
C&    IMPLICIT DOUBLE PRECISION (A-H,P-Z)
C&    DIMENSION HK(10,100)
C&    VSTHNV=HK(I,4)
C&    IF(V.GE.HK(I,3)) RETURN
C&    IF(V.GT.HK(I,5)) GO TO 1
C&    WRITE(6,100) V,I
C&100 FORMAT(/,28HINITIAL MOISTURE CONTENT OF ,F7.3,49HIS LESS THAN RES
C&   1IDUAL MOISTURE CONTENT FOR CLASS ,I4,/,
C&   214HPROGRAM HALTED)
C&    STOP
C&1   SE=(V-HK(I,5))/(HK(I,3)-HK(I,5))
C&    VSTHNV=HK(I,4)/(SE**(1.D0/HK(I,6)))
C&    RETURN
C&    END
C&    DOUBLE PRECISION FUNCTION VSTHU(P,I,HK)
C*****
CVSTHU
C*****
C
C     MOISTURE CONTENT AS A FUNCTION OF PRESSURE HEAD BELOW BUBBLING
C     PRESSURE: = POROSITY ELSEWHERE
C
C     BROOKS AND COREY, CSU HYDROLOGY PAPER NO.17, PP.3-4
```

```
C
C&    IMPLICIT DOUBLE PRECISION (A-H,P-Z)
C&    DIMENSION HK(10,100)
C&    VSTHU=HK(I,3)
C&    IF(P.GE.HK(I,4))RETURN
C&    VSTHU=HK(I,5)+(HK(I,3)-HK(I,5))*(HK(I,4)/P)**HK(I,6)
C&    RETURN
C&    END
C&    DOUBLE PRECISION FUNCTION VSHKU(P,I,HK)
C*****
CVSHKU
C*****
C
C    RELATIVE HYDRAULIC CONDUCTIVITY WITH RESPECT TO PRESSURE HEAD
C
C    BROOKS AND COREY, CSU HYDROLOGY PAPER NO. 3
C
C
C&    IMPLICIT DOUBLE PRECISION (A-H,P-Z)
C&    DIMENSION HK(10,100)
C&    VSHKU=1.00
C&    IF(P.GE.HK(I,4))RETURN
C&    VSHKU=(HK(I,4)/P)**(2.+3.*HK(I,6))
C&    IF(VSHKU.LT.1.D-38)VSHKU=0.00
C&    RETURN
C&    END
C$    DOUBLE PRECISION FUNCTION VSDTHU(P,I,HK)
C******
CVSDTHU
C******
C
C    FIRST DERIVATIVE OF MOISTURE CONTENT AS A FUNCTION OF PRESSURE HEAD
C
C   HAVERKAMP FUNCTION
C
C         HK(I,1)=SATURATED HYDRAULIC CONDUCTIVITY
C         HK(I,2)=SPECIFIC STORAGE
C         HK(I,3)=POROSITY
C         HK(I,4)=A PRIME
C         HK(I,5)=RESIDUAL MOISTURE CONTENT
C         HK(I,6)=B PRIME
C         HK(I,7)=ALPHA
C         HK(I,8)=BETA
C
C$    IMPLICIT DOUBLE PRECISION (A-H,P-Z)
C$    DIMENSION HK(10,100)
C$    VSDTHU=0.D0
C$    IF(P.GE.0.0)RETURN
C$    SE=HK(I,3)-HK(I,5)
C$    ALPH=HK(I,7)
C$    EM=HK(I,8)
C$    TOP=P/ALPH
C$    DEN=1+TOP**EM
C$    DEN=DEN*DEN
C$    VSDTHU=SE*EM*TOP**(EM-1)/(ALPH*DEN)
C$    RETURN
C$    END
C$    DOUBLE PRECISION FUNCTION VSTHNV(V,I,HK)
C******
CVSTHNV
C******
C
C    INITIAL UNSATURATED PRESSURE HEAD AS A FUNCTION OF VOLUMETRIC
C    MOISTURE CONTENT
C
C    HAVERKAMP FUNCTION
C
```

```
C$    IMPLICIT DOUBLE PRECISION (A-H,P-Z)
C$    DIMENSION HK(10,100)
C$    VSTHNV=0.0
C$    IF(V.GE.HK(I,3)) RETURN
C$    IF(V.GT.HK(I,5)) GO TO 1
C$    WRITE(6,100) V,I
C$100 FORMAT(/,28HINITIAL MOISTURE CONTENT OF ,F7.3,49HIS LESS THAN RES
C$   1IDUAL MOISTURE CONTENT FOR CLASS ,I4,/,
C$   214HPROGRAM HALTED)
C$    STOP
C$1   SE=(V-HK(I,5))/(HK(I,3)-HK(I,5))
C$    VSTHNV=HK(I,7)*(1.0/SE-1.0)**(1.0/HK(I,8))
C$    RETURN
C$    END
C$    DOUBLE PRECISION FUNCTION VSTHU(P,I,HK)
C******
CVSTHU
C******
C
C     MOISTURE CONTENT AS A FUNCTION OF PRESSURE HEAD
C
C  HAVERKAMP FUNCTION
C
C$    IMPLICIT DOUBLE PRECISION (A-H,P-Z)
C$    DIMENSION HK(10,100)
C$    VSTHU=HK(I,3)
C$    IF(P .GE. 0.0)RETURN
C$    VSTHU=HK(I,5)+(HK(I,3)-HK(I,5))/((P/HK(I,7))**HK(I,8)+1.)
C$    RETURN
C$    END
C$    DOUBLE PRECISION FUNCTION VSHKU(P,I,HK)
C*****
CVSHKU
C*****
C
C     RELATIVE HYDRAULIC CONDUCTIVITY WITH RESPECT TO PRESSURE HEAD
C
C  HAVERKAMP FUNCTION
C
C
C$    IMPLICIT DOUBLE PRECISION (A-H,P-Z)
C$    DIMENSION HK(10,100)
C$    VSHKU=1.00
C$    IF(P.GE.0.0)RETURN
C$    VSHKU=1.0/((P/HK(I,4))**HK(I,6)+1)
C$    RETURN
C$    END
C ***********************************************************************
C ***********************************************************************
C
C+    SUBROUTINE INTERP (P,I,HK)
C******
CINTERP
C******
C
C     THIS SUBROUTINE PERFORMS LINEAR INTERPOLATION OF PRESSURE
C     HEADS FOR RELATIVE HYDRAULIC CONDUCTIVITY (VSHKU), VOLUMETRIC
C     MOISTURE CONTENT (VSTHU), AND MOISTURE CAPACITY (VSDTHU).
C
C
C     TO USE THIS METHOD FOR EVALUATING THE NONLINEAR FUNCTIONS,
C     THE USER MUST ENTER A TABLE OF PRESSURE HEADS
C     AND VALUES OF RELATIVE
C     CONDUCTIVITIES,AND MOISTURE CONTENTS
C     WHICH CORRESPOND TO EACH PRESSURE HEAD INTO ARRAY HK ON
C     B-7 CARDS FOR EACH TEXTURAL CLASS.  SET NPROP (CARD B-5) EQUAL
C     TO 3*(NUMBER OF PRESSURE HEADS IN TABLE + 1).
```

```
C   BEGINNING WITH HK(ITEX,4), ENTER ALL PRESSURE HEADS IN DESCENDING
C   ORDER STARTING WITH THE HIGHEST VALUE.
C   NEXT ENTER THE NUMBER 99.
C   NEXT ENTER THE RELATIVE HYDRAULIC
C   CONDUCTIVITY FOR EACH PRESSURE HEAD.
C   NEXT ENTER THE NUMBER 99.
C   NEXT ENTER THE VOLUMETRIC MOISTURE CONTENT FOR EACH PRESSURE
C   HEAD. FINALLY ENTER THE NUMBER 99.
C
C+     IMPLICIT DOUBLE PRECISION (A-H,P-Z)
C+     DIMENSION HK(10,100)
C+     COMMON I1,I2,I3,I4,I5,I6,DELP
C+     IF (I2.GT.0) GO TO 1
C+     I2=4
C+     DO 2 J=I2,100
C+     IF (HK(I,J).LT.99) GO TO 2
C+     I3=J-I2+1
C+     I1=I3+I3
C+     GO TO 1
C+ 2   CONTINUE
C+ 1   IF(HK(I,I2).LE.P) THEN
C+     DELP=0
C+     I5=I2
C+     I6=I2
C+     ELSE
C+     I4=I2+I3-2
C+     IF(HK(I,I4).GE.P)THEN
C+     I5=I4-1
C+     I6=I4
C+     DELP=0
C+     ELSE
C+     I4=I4-1
C+     DO 3 J=I2+1,I4
C+     IF(HK(I,J).GT.P) GO TO 3
C+     I5=J-1
C+     I6=J
C+     DELP=(P-HK(I,I6))/(HK(I,I5)-HK(I,I6))
C+     RETURN
C+ 3   CONTINUE
C+     END IF
C+     END IF
C+     RETURN
C+     END
C+     DOUBLE PRECISION FUNCTION VSHKU (P,I,HK)
C*****
CVSHKU
C*****
C
C   RELATIVE HYDRAULIC CONDUCTIVITY AS A FUNCTION OF PRESSURE HEAD
C   DETERMINED BY LINEAR INTERPOLATION OF KR VS HP TABLE WHICH IS
C   INPUT BY USER.
C
C+     IMPLICIT DOUBLE PRECISION (A-H,P-Z)
C+     DIMENSION HK(10,100)
C+     COMMON I1,I2,I3,I4,I5,I6,DELP
C+     CALL INTERP (P,I,HK)
C+     IF(I5.EQ.I6)THEN
C+     VSHKU=HK(I,I3+I5)
C+     RETURN
C+     ELSE
C+     VSHKU=HK(I,I3+I6)+(HK(I,I3+I5)-HK(I,I3+I6))*DELP
C+     RETURN
C+     END IF
C+     END
C+     DOUBLE PRECISION FUNCTION VSDTHU(P,I,HK)
C******
CVSDTHU
```

```
C******
C
C   MOISTURE CAPACITY AS A FUNCTION OF PRESSURE HEAD AS
C   DETERMINED FROM TABLE OF THETA VS HP WHICH IS INPUT
C   BY USER.
C
C+      IMPLICIT DOUBLE PRECISION (A-H,P-Z)
C+      DIMENSION HK(10,100)
C+      COMMON I1,I2,I3,I4,I5,I6,DELP
C+      IF (I5.EQ.I6) THEN
C+      VSDTHU=0.
C+      RETURN
C+      ELSE
C+      VSDTHU=(HK(I,I1+I5)-HK(I,I1+I6))/(HK(I,I5)-HK(I,I6))
C+      RETURN
C+      END IF
C+      END
C+      DOUBLE PRECISION FUNCTION VSTHU (P,I,HK)
C*****
CVSTHU
C*****
C
C   VOLUMETRIC MOISTURE CONTENT AS A FUNCTION OF PRESSURE HEAD
C   AS DETERMINED BY LINEAR INTERPOLATION OF THETA VS HP TABLE
C   WHICH IS INPUT BY USER.
C
C+      IMPLICIT DOUBLE PRECISION (A-H,P-Z)
C+      DIMENSION HK(10,100)
C+      COMMON I1,I2,I3,I4,I5,I6,DELP
C+      IF (DELP.EQ.0) THEN
C+      VSTHU=HK(I,I1+I6)
C+      ELSE
C+      VSTHU=HK(I,I1+I6)+(HK(I,I1+I5)-HK(I,I1+I6))*DELP
C+      END IF
C+      RETURN
C+      END
C+      DOUBLE PRECISION FUNCTION VSTHNV(P,I,HK)
C******
CVSTHNV
C******
C
C       NOTE -- THIS FUNCTION IS NOT OPERATIVE WHEN USING INTERPOLATION
C               ROUTINES.  INITIAL CONDITIONS MUST BE INPUT IN TERMS OF
C               PRESSURE HEADS NOT MOISTURE CONTENTS.
C
C+      IMPLICIT DOUBLE PRECISION (A-H,P-Z)
C+      DIMENSION HK(10,100)
C+      WRITE(6,100)
C+      STOP
C+100 FORMAT(5X,'INPUT OF MOISTURE CONTENT FOR INITIAL CONDITIONS IS ',
C+    1'NOT ALLOWED WHEN USING TABULAR DATA '/
C+    25X,'FOR MOISTURE RETENTION AND CONDUCTIVITY CURVES',/
C+    35X,'SIMULATION TERMINATED')
C+      END
        SUBROUTINE VTVELO
C******
CVTVELO
C******
C
C     ROUTINE TO CALCULATE VELOCITIES AT BOUNDARIES OF ADJACENT CELLS
C     VX IS VELOCITY IN X-DIRECTION BETWEEN CURRENT NODE AND NODE TO
C     THE LEFT.
C     VZ IS VELOCITY IN Z-DIRECTION BETWEEN CURRENT NODE AND NODE
C     ABOVE.
C
        IMPLICIT DOUBLE PRECISION (A-H,P-Z)
        COMMON/RSPAC/DELZ(600),DZZ(600),DXR(600),RX(600),PI2
```

```
      COMMON/ISPAC/NLY,NLYY,NXR,NXRR,NNODES
      COMMON/KCON/HX(1600),NTYP(1600)
      COMMON/RPROP/HK(10,100),HT(10,20),ANIZ(10)
      COMMON/MPROP/THETA(1600),THLST(1600)
      COMMON/PRESS/P(1600),PXXX(1600),CS1,CS2
      COMMON/HCON/HCND(1600),HKLL(1600),HKTT(1600)
      COMMON/JTXX/JTEX(1600)
      COMMON/WGT/WUS,WDS
      COMMON/TRXX/DX1(1600),DX2(1600),DZ1(1600),DZ2(1600),VX(1600),
     &VZ(1600),CC(1600),COLD(1600),CS(1600),QT(1600),NCTYP(1600),
     &RET(1600)
      LOGICAL RAD,BCIT,ETSIM,SEEP,ITSTOP,CIS,CIT
      COMMON/LOG1/RAD,BCIT,ETSIM,SEEP,ITSTOP,CIS,CIT
      DO 10 I=2,NXRR
      N1=NLY*(I-1)
      DO 10 J=2,NLYY
      N=N1+J
      VX(N)=0
      VZ(N)=0
      IF(HX(N).NE.0) THEN
      JM1=N-1
      IM1=N-NLY
      IF(HX(JM1).NE.0) THEN
C
C   CALCULATE VERTICAL VELOCITY
C
      AREA=DXR(I)
      IF (RAD) AREA=PI2*RX(I)*DXR(I)
      GRAD=P(JM1)-P(N)
      THETA1=0.5*(THETA(N)+THETA(JM1))*AREA
      IF(WUS.EQ.0.) THEN
      VZ(N)=HKTT(N)*DSQRT(HCND(N)*HCND(JM1))*GRAD/THETA1
      ELSE
      IF(P(JM1).GT.P(N))THEN
      ALA=WUS
      BTA=WDS
      ELSE
      ALA=WDS
      BTA=WUS
      END IF
      VZ(N)=HKTT(N)*(ALA*HCND(JM1)+BTA*HCND(N))*GRAD/THETA1
      END IF
      END IF
      IF(HX(IM1).NE.0) THEN
C
C   CALCULATE HORIZONTAL VELOCITY
C
      GRAD=P(IM1)-P(N)
      AREA=DELZ(J)
      IF (RAD) AREA=PI2*AREA*(RX(I)-0.5*DXR(I))
      THETA1=0.5*(THETA(N)+THETA(IM1))*AREA
      IF(WUS.EQ.0) THEN
      VX(N)=HKLL(N)*DSQRT(HCND(N)*HCND(IM1))*GRAD/THETA1
      ELSE
      IF(P(IM1).GT.P(N)) THEN
      ALA=WUS
      BTA=WDS
      ELSE
      ALA=WDS
      BTA=WUS
      END IF
      VX(N)=HKLL(N)*(ALA*HCND(IM1)+BTA*HCND(N))*GRAD/THETA1
      END IF
      END IF
      END IF
   10 CONTINUE
      RETURN
```

```
      END
      SUBROUTINE VTDCOEF
C*******
CVTDCOEF
C*******
C
C     ROUTINE TO CALCULATE DISPERSION COEFFICIENTS AS FUNCTIONS
C     OF DISPERSIVITIES AND VELOCITIES.  DIAGNOL TERMS ARE
C     CONTAINED IN ARRAYS DX1 AND DZ1.  CROSS PRODUCT TERMS
C     ARE IN DX2 AND DZ2
C
      IMPLICIT DOUBLE PRECISION (A-H,P-Z)
      COMMON/RSPAC/DELZ(600),DZZ(600),DXR(600),RX(600),PI2
      COMMON/ISPAC/NLY,NLYY,NXR,NXRR,NNODES
      COMMON/KCON/HX(1600),NTYP(1600)
      COMMON/RPROP/HK(10,100),HT(10,20),ANIZ(10)
      COMMON/MPROP/THETA(1600),THLST(1600)
      COMMON/JTXX/JTEX(1600)
      COMMON/TRXX/DX1(1600),DX2(1600),DZ1(1600),DZ2(1600),VX(1600),
     &VZ(1600),CC(1600),COLD(1600),CS(1600),QT(1600),NCTYP(1600),
     &RET(1600)
      LOGICAL RAD,BCIT,ETSIM,SEEP,ITSTOP,CIS,CIT
      COMMON/LOG1/RAD,BCIT,ETSIM,SEEP,ITSTOP,CIS,CIT
      DO 10 I=2,NXRR
      N1=NLY*(I-1)
      DO 10 J=2,NLYY
      N=N1+J
      DX1(N)=0
      DX2(N)=0
      DZ1(N)=0
      DZ2(N)=0
      PEX=0.
      PEZ=0.
      IMX=0
      JMX=0
      IMZ=0
      JMZ=0
      IF(HX(N).NE.0) THEN
      N2=JTEX(N)
      AL=HT(N2,1)
      AT=HT(N2,2)
      DM=HT(N2,3)
      V1=VX(N)
      V2=VZ(N)
      JM1=N-1
      IM1=N-NLY
      JP1=N+1
      IP1=N+NLY
      IP2=IP1-1
      IM2=IM1+1
      IF(HX(JM1).NE.0.) THEN
      V3=0.25*(V1+VX(IP1)+VX(IP2)+VX(JM1))
      V32=V3*V3
      V22=V2*V2
      VV2=V32+V22
C
C     CALCULATE DZ1 AND DZ2
C
      N2=JTEX(JM1)
      AL1=DSQRT(AL*HT(N2,1))
      AT1=DSQRT(AT*HT(N2,2))
      DM1=DSQRT(DM*HT(N2,3))
      AREA=DXR(I)
      IF(RAD) AREA=PI2*AREA*RX(I)
      T1=0.5*(THETA(JM1)+THETA(N))
      DD1=(DZZ(J)-DZZ(J-1))/AREA
      T2=T1/DD1
```

```
      IF(VV2.EQ.O.) THEN
      DZ1(N)=DM1
      ELSE
      VAVE=DSQRT(VV2)
      DL=AL1*VAVE
      DT=AT1*VAVE
      DZ1(N)=(DL*V22+DT*V32)/VV2+DM1
      DD1=(RX(I+1)-RX(I-1))/AREA
      DZ2(N)=T1*(DL-DT)*V2*V3/(DD1*VV2)
      END IF
C
C  CALCULATE VERTICAL CELL PECLET NUMBER
C
      PE=DABS(VZ(N))*(DZZ(J)-DZZ(J-1))/DZ1(N)
      DZ1(N)=T2*DZ1(N)
      IF(PE.GT.PEZ) THEN
      PEZ=PE
      IMZ=I
      JMZ=J
      END IF
      END IF
      IF(HX(IM1).NE.O.) THEN
      V3=0.25*(V2+VZ(JP1)+VZ(IM1)+VZ(IM2))
      V32=V3*V3
      V12=V1*V1
      VV2=V12+V32
C
C   CALCULATE DX1 AND DX2
C
      N2=JTEX(IM1)
      AL1=DSQRT(AL*HT(N2,1))
      AT1=DSQRT(AT*HT(N2,2))
      DM1=DSQRT(DM*HT(N2,3))
      AREA=DELZ(J)
      IF(RAD) AREA=PI2*AREA*(RX(I)-0.5*DXR(I))
      DD1=(RX(I)-RX(I-1))/AREA
      T1=0.5*(THETA(IM1)+THETA(N))
      T2=T1/DD1
      IF(VV2.EQ.O.) THEN
      DX1(N)=DM1
      ELSE
      VAVE=DSQRT(VV2)
      DL=AL1*VAVE
      DT=AT1*VAVE
      DX1(N)=(DL*V12+DT*V32)/VV2+DM1
      DD1=(DZZ(J+1)-DZZ(J-1))/AREA
      DX2(N)=T1*(DL-DT)*V1*V3/(VV2*DD1)
      END IF
C
C  CALCULATE HORIZONTAL CELL PECLET NUMBER
C
      PE=DABS(VX(N))*(RX(I)-RX(I-1))/DX1(N)
      DX1(N)=DX1(N)*T2
      IF(PE.GT.PEX) THEN
      PEX=PE
      IMX=I
      JMX=J
      END IF
      END IF
      END IF
   10 CONTINUE
C
C  WRITE MAXIMUM CELL PECLET NUMBERS
C
      WRITE(6,4000) PEX,JMX,IMX,PEZ,JMZ,IMZ
      RETURN
 4000 FORMAT(4X,'  MAXIMUM CELL PECLET NUMBER  --  HORIZONTAL  ',E14.5,
```

```
     &'     ROW ',I4,'  COLUMN ',I4,/,38X,'VERTICAL    ',E14.5,
     &'     ROW ',I4,'  COLUMN ',I4)
      END
      SUBROUTINE VTSETUP
C*******
CVTSETUP
C*******
C
C     ROUTINE TO ASSEMBLE MATRIX EQUATIONS FOR ADVECTION-DISPERSION
C     EQUATIONS AND TO CALL MATRIX SOLVER.
C
      IMPLICIT DOUBLE PRECISION (A-H,P-Z)
      COMMON/PRESS/P(1600),PXXX(1600),CS1,CS2
      COMMON/RSPAC/DELZ(600),DZZ(600),DXR(600),RX(600),PI2
      COMMON/ISPAC/NLY,NLYY,NXR,NXRR,NNODES
      COMMON/KCON/HX(1600),NTYP(1600)
      COMMON/RPROP/HK(10,100),HT(10,20),ANIZ(10)
      COMMON/MPROP/THETA(1600),THLST(1600)
      COMMON/DISCH/Q(1600),QQ(1600),ETOUT,ETOUT1
      COMMON/EQUAT/A(1600),B(1600),C(1600),D(1600),E(1600),RHS(1600),
     &XI(1600)
      COMMON/JTXX/JTEX(1600)
      COMMON/JCON/JSTOP,JFLAG
      COMMON/SCON/DHMX(201),DELT,HMAX,TMAX,EPS,NUMT,ITMAX,MINIT,ITEST
      COMMON/TCON/STIM,DSMAX,KTIM,NIT,NIT1,KP
      COMMON/TRXX/DX1(1600),DX2(1600),DZ1(1600),DZ2(1600),VX(1600),
     &VZ(1600),CC(1600),COLD(1600),CS(1600),QT(1600),NCTYP(1600),
     &RET(1600)
      COMMON/TRXY1/A0(1600),B0(1600),C0(1600),D0(1600),E0(1600)
      LOGICAL TRANS,TRANS1,SORP,SSTATE
      COMMON/TRXY/MB9(72),NMB9,EPS1,TRANS,TRANS1,SORP,SSTATE
      LOGICAL RAD,BCIT,ETSIM,SEEP,ITSTOP,CIS,CIT
      COMMON/LOG1/RAD,BCIT,ETSIM,SEEP,ITSTOP,CIS,CIT
      SAVE JFLAG1
      IF(KTIM.EQ.1) THEN
      JFLAG1=1
      DO 10N=1,NNODES
      A0(N)=0
      B0(N)=0
      C0(N)=0
      D0(N)=0
      E0(N)=0
   10 CONTINUE
      END IF
C
C     INITIALIZE VARIABLES
C
      DO 20 I=2,NXRR
      N1=NLY*(I-1)
      DO 20 J=2,NLYY
      N=N1+J
      A(N)=0
      B(N)=0
      C(N)=0
      D(N)=0
      E(N)=0
      RHS(N)=0
      COLD(N)=CC(N)
      QT(N)=0
      IF(NTYP(N).EQ.1) QT(N)=VSFLX1(N)
      IF(HX(N).NE.0) THEN
      N2=JTEX(N)
      RET(N)=VTRET(CC(N),N2,HT)
      IM1=N-NLY
      JM1=N-1
      JP1=N+1
      IP1=N+NLY
```

```
      IP2=IP1-1
      IM2=IM1+1
      IM3=IM1-1
      IP3=IP1+1
      IF(RAD) THEN
      AREAX=PI2*DELZ(J)*(RX(I)-0.5*DXR(I))
      AREAX1=PI2*DELZ(J)*(RX(I)+0.5*DXR(I))
      AREAZ=PI2*DXR(I)*RX(I)
      ELSE
      AREAX=DELZ(J)
      AREAX1=AREAX
      AREAZ=DXR(I)
      END IF
      VOL=AREAZ*DELZ(J)
      AREAX=AREAX*0.5*(THETA(IM1)+THETA(N))
      AREAX1=AREAX1*0.5*(THETA(IP1)+THETA(N))
      AREAZ1=AREAZ*0.5*(THETA(JP1)+THETA(N))
      AREAZ=AREAZ*0.5*(THETA(JM1)+THETA(N))
C
C   CALCULATE LHS OF MATRIX EQUATION
C
      SS=THETA(N)*P(N)*HK(N2,2)/HK(N2,3)
      E(N)=-DX1(N)-DZ1(N)-DX1(IP1)-DZ1(JP1)
     &-VOL*(HT(N2,4)*(THETA(N)+SS+RET(N)))
      SS=THETA(N)+SS+SS-THLST(N)*(1+PXXX(N)*HK(N2,2)/HK(N2,3))
      IF(HX(IM1).NE.0) THEN
      A(N)=DX1(N)+0.5*(+DZ2(N)-DZ2(JP1))
      IF(.NOT.CIS) THEN
      IF(VX(N).GT.0) THEN
      A(N)=A(N)+AREAX*VX(N)
      ELSE
      E(N)=E(N)+AREAX*VX(N)
      END IF
      ELSE
      VV=AREAX*0.5*VX(N)
      A(N)=A(N)+VV
      E(N)=E(N)+VV
      END IF
      END IF
      IF(HX(JM1).NE.0) THEN
      B(N)=DZ1(N)+0.5*(+DX2(N)-DX2(IP1))
      IF(.NOT.CIS) THEN
      IF(VZ(N).GT.0) THEN
      B(N)=B(N)+AREAZ*VZ(N)
      ELSE
      E(N)=E(N)+AREAZ*VZ(N)
      END IF
      ELSE
      VV=0.5*AREAZ*VZ(N)
      B(N)=B(N)+VV
      E(N)=E(N)+VV
      END IF
      END IF
      IF(HX(IP1).NE.0) THEN
      C(N)=DX1(IP1)+0.5*(-DZ2(N)+DZ2(JP1))
      IF(.NOT.CIS) THEN
      IF(VX(IP1).LT.0) THEN
      C(N)=C(N)-AREAX1*VX(IP1)
      ELSE
      E(N)=E(N)-AREAX1*VX(IP1)
      END IF
      ELSE
      VV=0.5*AREAX1*VX(IP1)
      C(N)=C(N)-VV
      E(N)=E(N)-VV
      END IF
      END IF
```

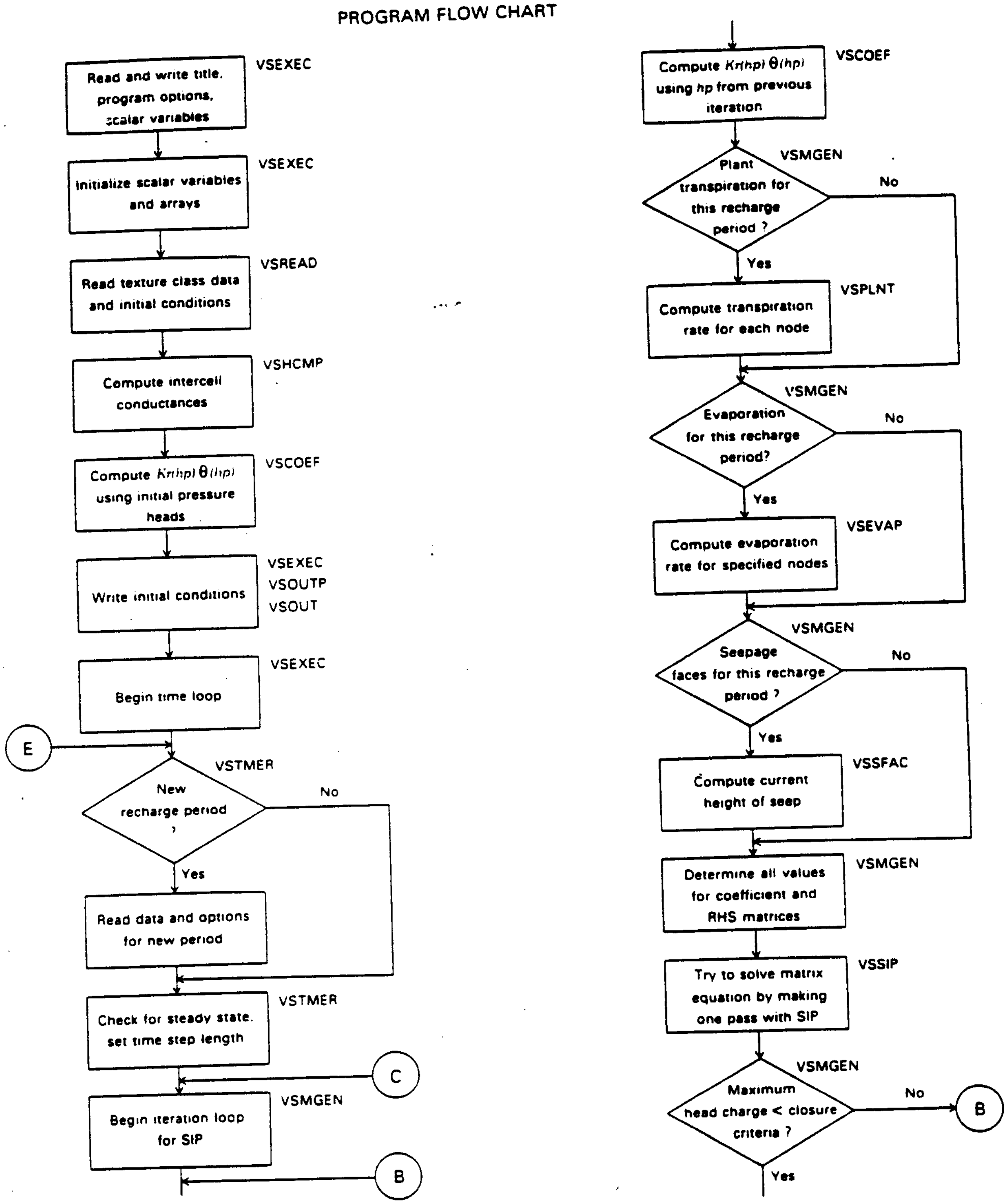
```
      IF(HX(JP1).NE.0) THEN
      D(N)=DZ1(JP1)+0.5*(-DX2(N)+DX2(IP1))
      IF(.NOT.CIS) THEN
      IF(VZ(JP1).LT.0) THEN
      D(N)=D(N)-AREAZ1*VZ(JP1)
      ELSE
      E(N)=E(N)-AREAZ1*VZ(JP1)
      END IF
      ELSE
      VV=0.5*AREAZ1*VZ(JP1)
      D(N)=D(N)-VV
      E(N)=E(N)-VV
      END IF
      END IF
      IF(Q(N).LT.0..AND.NTYP(N).NE.5) E(N)=E(N)+Q(N)
      IF(QQ(N).LT.0.) E(N)=E(N)+QQ(N)
      IF(QT(N).GT.0) E(N)=E(N)-QT(N)
C
C  CENTERED-IN-TIME DIFFERENCING CAN BE USED ONLY AFTER THE
C  FIRST TIME STEP IN ANY RECHARGE PERIOD.
C
      IF(CIT.AND.JFLAG1.NE.1) THEN
      A(N)=0.5*A(N)
      B(N)=0.5*B(N)
      C(N)=0.5*C(N)
      D(N)=0.5*D(N)
      E(N)=0.5*E(N)
      END IF
      E(N)=E(N)-VOL*(THETA(N)+SS+RET(N))/DELT
      END IF
   20 CONTINUE
C
C  BEGIN LOOP TO CALCULATE RHS AND CALL MATRIX SOLVER
C
      DO 50 IT=1,ITMAX
      DO 30 I=2,NXRR
      N1=NLY*(I-1)
      DO 30 J=2,NLYY
      N=N1+J
      IM1=N-NLY
      JM1=N-1
      JP1=N+1
      IP1=N+NLY
      IP2=IP1-1
      IM2=IM1+1
      IM3=IM1-1
      IP3=IP1+1
      IF(RAD) THEN
      VOL=PI2*DELZ(J)*DXR(I)*RX(I)
      ELSE
      VOL=DELZ(J)*DXR(I)
      END IF
      N2=JTEX(N)
      IF(SORP) THEN
      IF(IT.GT.1) THEN
C
C   FOR NONLINEAR SORPTION RECALCULATE RET,E
C
      RET1=RET(N)
      RET(N)=VTRET(CC(N),N2,HT)
      IF(CIT.AND.JFLAG1.NE.1) THEN
      T1=0.5
      ELSE
      T1=1.
      END IF
      E(N)=E(N)+VOL*(RET1-RET(N))*(1./DELT+HT(N2,4)*T1)
      END IF
```

```
      END IF
C
C   CALCULATE RHS OF MATRIX EQUATION
C
      RHS(N)=-VOL*(THETA(N)*(1+P(N)*HK(N2,2)/HK(N2,3))+RET(N))*COLD(N)/
     &DELT+0.5*(DX2(N)*(CC(IM2)-CC(IM3))+
     &DX2(IP1)*(CC(IP2)-CC(IP3))+DZ2(N)*(CC(IP2)-CC(IM3))
     &+DZ2(JP1)*(CC(IM2)-CC(IP3)))-A(N)*CC(IM1)-B(N)*CC(JM1)
     &-C(N)*CC(IP1)-D(N)*CC(JP1)-E(N)*CC(N)
      IF (CIT.AND.JFLAG1.NE.1) RHS(N)=RHS(N)-AO(N)*COLD(IM1)-BO(N)
     **COLD(JM1)-CO(N)*COLD(IP1)-DO(N)*COLD(JP1)-EO(N)*COLD(N)
      IF(QQ(N).GT.0.) RHS(N)=RHS(N)-QQ(N)*CS(N)
      IF(QT(N).LT.0..AND.NCTYP(N).EQ.0) RHS(N)=RHS(N)+QT(N)*CS(N)
      IF(QT(N).LE.0.AND.NCTYP(N).EQ.2) RHS(N)=RHS(N)-CS(N)
   30 CONTINUE
      NIT1=NIT1+1
C
C   CALL MATRIX SOLVER
C
      CALL SLVSIP
      IF(ITEST.EQ.0) THEN
      IF (CIT) THEN
      DO 40 I=2,NXRR
      N1=NLY*(I-1)
      DO 40 J=2,NLYY
      N=N1+J
      IF(HX(N).EQ.0) GO TO 40
      AO(N)=A(N)
      BO(N)=B(N)
      CO(N)=C(N)
      DO(N)=D(N)
      IF(RAD) THEN
      AREAZ=PI2*DXR(I)*RX(I)
      ELSE
      AREAZ=DXR(I)
      END IF
      VOL=AREAZ*DELZ(J)
      N2=JTEX(N)
      SS=HK(N2,2)/HK(N2,3)
      SS=THETA(N)*(1+(SS+SS)*P(N))-THLST(N)*(1+SS*PXXX(N))
      EO(N)=E(N)+VOL*(THETA(N)+SS+RET(N))/DELT
   40 CONTINUE
      END IF
      JFLAG1=JFLAG
      RETURN
      END IF
   50 CONTINUE
      JFLAG1=JFLAG
      WRITE(6,4000)
      IF (.NOT.ITSTOP) RETURN
      JSTOP=1
      JFLAG=1
      RETURN
 4000 FORMAT(' MAXIMUM NUMBER OF ITERATIONS EXCCEDED FOR TRANSPORT'
     &' EQUATION')
      END
      DOUBLE PRECISION FUNCTION VTRET(P,I,HT)
C*****
CVTRET
C*****
C
C  SLOPE OF SORPTION ISOTHERM  --  LANGMUIR
C
      IMPLICIT DOUBLE PRECISION (A-H,P-Z)
      DIMENSION HT(10,20)
      VTRET=HT(I,5)*HT(I,6)*HT(I,7)/(1+HT(I,6)*P)**2
      RETURN
```

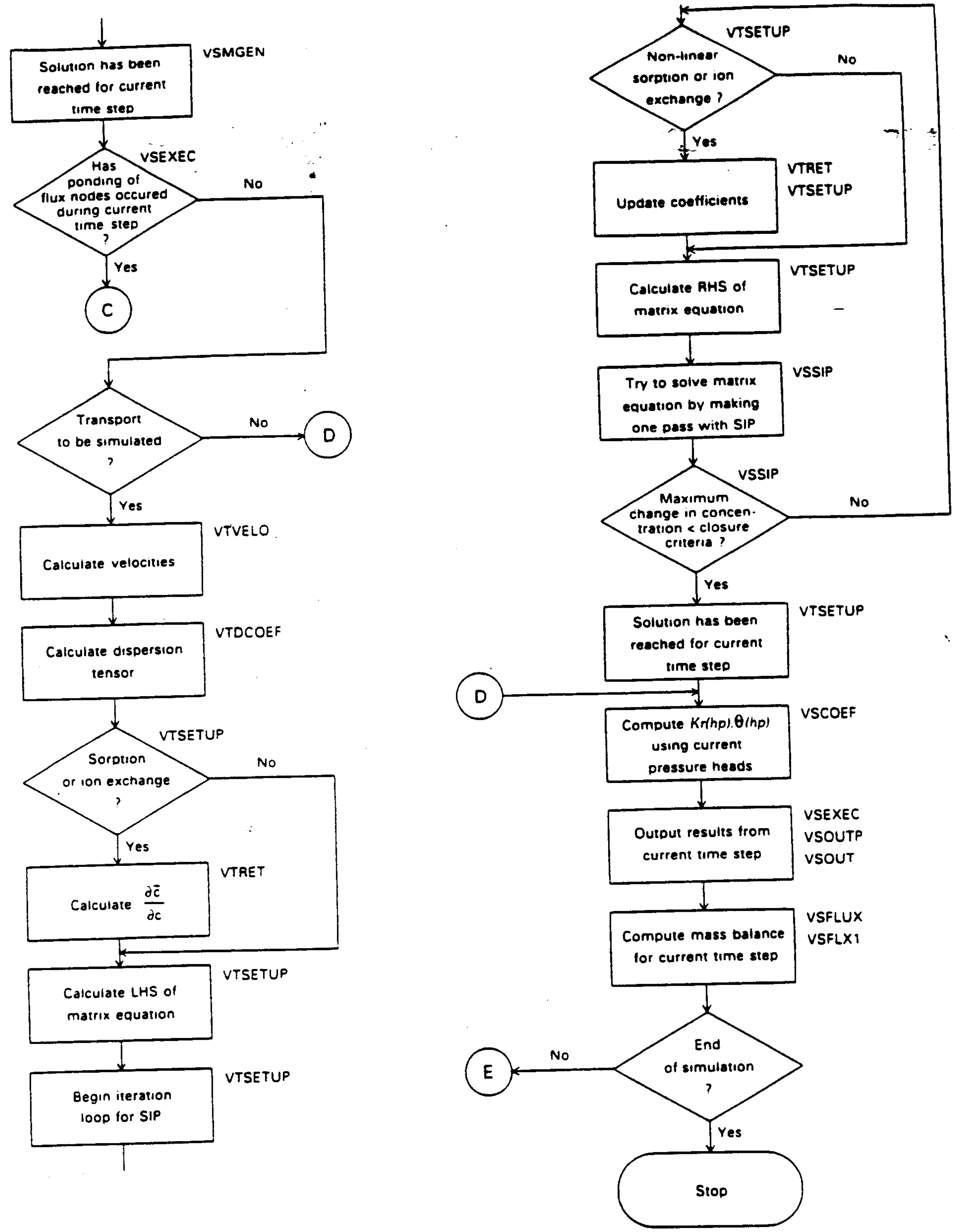
```
      END
C
C
C
C  NOTE -- AS LISTED HERE THE PROGRAM USES THE VTRET FUNCTION
C          ROUTINE FOR THE LANGMUIR ISOTHERM.
C          FUNCTIONS FOR THE 5 ALTERNATIVE VERSIONS OF VTRET ARE
C          LISTED BELOW.  TO USE ONE OF THESE PLACE A 'C' IN
C          COLUMN 1 OF ALL LINES IN THE LANGMUIR VERSION AND
C          REMOVE THE COMMENT DESIGNATIONS FOR THE DESIRED
C          VERSION OF VTRET -- 'CF' FREUNDLICH ISOTHERM
C                              'CM' MONO-MONOVALENT ION EXCHANGE
C                              'CD' DIVALENT-DIVALENT ION EXCHANGE
C                              'CE' MONO-DIVALENT ION EXCHANGE
C                              'CG' DI-MONOVALENT ION EXCHANGE
C
C
C
CF    DOUBLE PRECISION FUNCTION VTRET(P,I,HT)
C*****
CVTRET
C*****
C
C  SLOPE OF SORPTION ISOTHERM  --  FREUNDLICH
C
CF    IMPLICIT DOUBLE PRECISION (A-H,P-Z)
CF    DIMENSION HT(10,20)
CF    IF(HT(I,6).EQ.0.) THEN
CF    VTRET=0.D0
CF    RETURN
CF    END IF
CF    IF(HT(I,7).EQ.1.) THEN
CF    VTRET=HT(I,5)*HT(I,6)
CF    ELSE
CF    IF(P.EQ.0.) THEN
CF    VTRET=0.0
CF    ELSE
CF    VTRET=HT(I,5)*HT(I,6)*HT(I,7)*P**(HT(I,7)-1)
CF    END IF
CF    END IF
CF    RETURN
CF    END
CM    DOUBLE PRECISION FUNCTION VTRET(P,I,HT)
C*****
CVTRET
C*****
C
C  SLOPE OF SORPTION CURVE FOR
C  MONOVALENT-MONOVALENT ION EXCHANGE
C
CM    IMPLICIT DOUBLE PRECISION (A-H,P-Z)
CM    DIMENSION HT(10,20)
CM    VTRET=HT(I,5)*HT(I,6)*HT(I,7)*HT(I,8)/(P*(HT(I,6)-1)
CM   1+HT(I,8))**2
CM    RETURN
CM    END
CD    DOUBLE PRECISION FUNCTION VTRET(P,I,HT)
C*****
CVTRET
C*****
C
C  SLOPE OF SORPTION CURVE FOR
C  DIVALENT-DIVALENT ION EXCHANGE
C
CD    IMPLICIT DOUBLE PRECISION (A-H,P-Z)
CD    DIMENSION HT(10,20)
CD    VTRET=HT(I,5)*HT(I,6)*HT(I,7)*HT(I,8)/((P+P)*
```

```
CD    1(HT(I,6)-1)+HT(I,8))**2
CD     RETURN
CD     END
CE     DOUBLE PRECISION FUNCTION VTRET(P,I,HT)
C*****
CVTRET
C*****
C
C  SLOPE OF SORPTION CURVE FOR
C  MONOVALENT-DIVALENT ION EXCHANGE
C
CE     IMPLICIT DOUBLE PRECISION (A-H,P-Z)
CE     DIMENSION HT(10,20)
CE     IF(P.LE.0) THEN
CE     VTRET=0.
CE     ELSE
CE     P1=P*P
CE     P2=P1*HT(I,6)
CE     P3=HT(I,8)-P
CE     P3=P3+P3
CE     P4=HT(I,6)*(P+P)
CE     CB=(-P2+DSQRT(P2*P2+(P3+P3)*P2*HT(I,7)))/P3
CE     IF(CB.LT.HT(I,7))THEN
CE     VTRET=HT(I,5)*(CB*CB+P4*(HT(I,7)-CB))/(P3*CB+P2)
CE     ELSE
CE     VTRET=HT(I,5)*(HT(I,7)*HT(I,7))/(P3*HT(I,7)+P2)
CE     END IF
CE     END IF
CE     RETURN
CE     END
CG     DOUBLE PRECISION FUNCTION VTRET(P,I,HT)
C*****
CVTRET
C*****
C
C  SLOPE OF SORPTION CURVE FOR
C  DIVALENT-MONOVALENT ION EXCHANGE
C
CG     IMPLICIT DOUBLE PRECISION (A-H,P-Z)
CG     DIMENSION HT(10,20)
CG     IF(P.LE.0.) THEN
CG     VTRET=0.
CG     ELSE
CG     IF((P+P).GE.HT(I,8)) THEN
CG     VTRET=0.00
CG     ELSE
CG     P1=P*HT(I,6)
CG     P2=P1+P1+P1+P1
CG     P4=HT(I,8)-P-P
CG     P5=P4*P4
CG     P6=HT(I,7)**2
CG     P3=-P2*HT(I,7)-P5
CG     P7=P3*P3-4*P2*P1*P6
CG     IF (P7.GT.0) THEN
CG     CB=(-P3-DSQRT(P7))/(P2+P2)
CG     ELSE
CG     CB=0.
CG     END IF
CG     VTRET=HT(I,5)*(-CB*CB*4*HT(I,6)+4*CB*(HT(I,6)*HT(I,7)-P4)-HT(I,6)
CG    1*P6)/(P2*(CB+CB-HT(I,7))-P5)
CG     END IF
CG     END IF
CG     RETURN
CG     END
```

PROGRAM FLOW CHART



AR 020639


Solution has been reached for current time step
VSMGEN
Has ponding of flux nodes occured during current time step ?
VSEXEC
No
Yes
C
Transport to be simulated ?
No
D
Yes
Calculate velocities
VTVELO
Calculate dispersion tensor
VTDCOEF
Sorption or ion exchange ?
VTSETUP
No
Yes
Calculate $\frac{\partial \bar{c}}{\partial c}$
VTRET
Calculate LHS of matrix equation
VTSETUP
Begin iteration loop for SIP
VTSETUP
Non-linear sorption or ion exchange ?
VTSETUP
No
Yes
Update coefficients
VTRET
VTSETUP
Calculate RHS of matrix equation
VTSETUP
Try to solve matrix equation by making one pass with SIP
VSSIP
Maximum change in concentration < closure criteria ?
VSSIP
No
Yes
Solution has been reached for current time step
VTSETUP
D
Compute Kr(hp), θ(hp) using current pressure heads
VSCOEF
Output results from current time step
VSEXEC
VSOUTP
VSOUT
Compute mass balance for current time step
VSFLUX
VSFLX1
End of simulation ?
No
E
Yes
Stop