Instruction Report HL-94-1
June 1994

US Army Corps
of Engineers
Waterways Experiment
Station

3DSALT: A Three-Dimensional Finite Element Model of Density-Dependent Flow and Transport Through Saturated-Unsaturated Media

by Gour-Tsyh Yeh, Jing-Ru Chang,
Jin-Ping Gwo
Pennsylvania State University

Hsin-Chi Lin, David R. Richards,
William D. Martin

WES

Prepared for U.S. Army Engineer District, Wilmington

Instruction Report HL-94-1
June 1994

3DSALT: A Three-Dimensional Finite Element Model of Density-Dependent Flow and Transport Through Saturated-Unsaturated Media

by Gour-Tsyh Yeh, Jing-Ru Chang, Jin-Ping Gwo

Department of Civil Engineering
Pennsylvania State University
University Park, PA 16802

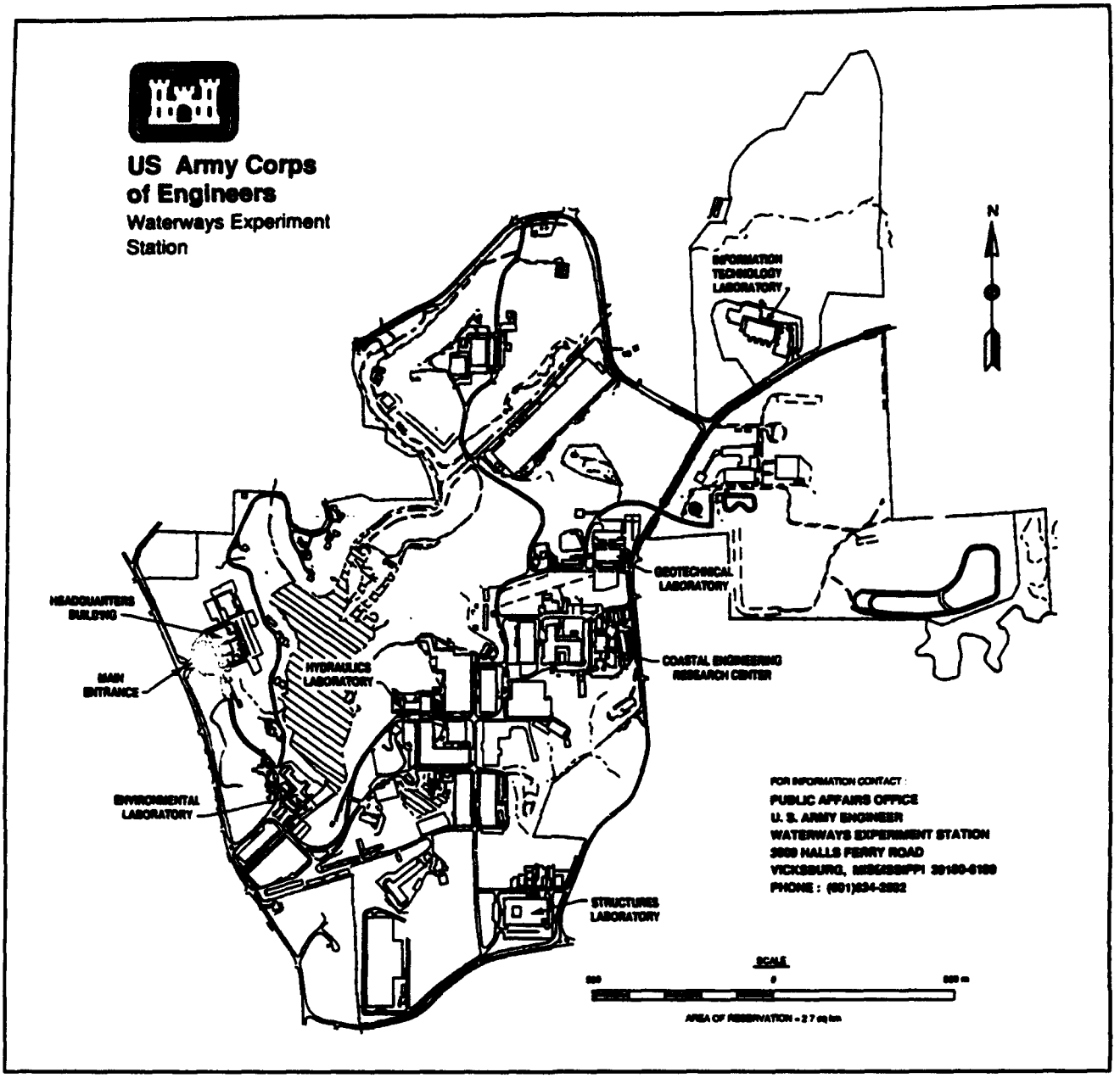
Hsin-Chi Lin, David R. Richards, William D. Martin

U.S. Army Corps of Engineers
Waterways Experiment Station
3909 Halls Ferry Road
Vicksburg, MS 39180-6199

Final report

Approved for public release; distribution is unlimited

Prepared for U.S. Army Engineer District, Wilmington
P.O. Box 1890
Wilmington, NC 28402-1890



Waterways Experiment Station Cataloging-in-Publication Data

3DSALT : A three-dimensional finite element model of density-dependent flow and transport through saturated-unsaturated media / by Gour-Tsyh Yeh ... [et al.] ; prepared for U.S. Army Engineer District, Wilmington.
178 p. : ill. ; 28 cm. — (Instruction report ; HL-94-1)
Includes bibliographic references.
1. Fluids — Migration — Computer programs. 2. 3DSALT (Computer program) 3. Porous materials — Density — Data processing. 4. Groundwater flow — Computer simulation. I. Yeh, G. T. II. United States. Army. Corp of Engineers. Wilmington District. III. U.S. Army Engineer Waterways Experiment Station. IV. Title: A three-dimensional finite element model of density-dependent flow and transport through saturated-unsaturated media. V. Instruction report (U.S. Army Engineer Waterways Experiment Station) ; HL-94-1
TA7 W34i no.HL-94-1

Contents

Preface viii

1—Introduction 1

Background 1
Purpose 11
Scope 11

2—The 3DSALT Program Structure 14

Purpose of 3DSALT 14
Governing flow equation 14
Initial conditions for flow equation 15
Boundary conditions for flow equations 15
Governing equations for transport 17
Description of 3DSALT Subroutines 19
Program MAIN 19
Subroutine RDATIO 23
Subroutine FSSDAT 23
Subroutine TSSDAT 23
Subroutine FBCDAT 23
Subroutine TBCDAT 23
Subroutine GWM3D 24
Subroutine HYDRO 24
Subroutine SURF 25
Subroutine READR 25
Subroutine READN 25
Subroutine PAGEN 25
Subroutine ESSFCT 26
Subroutine WSSFCT 26
Subroutine DBVFCT 26
Subroutine CBVFCT 26
Subroutine NBVFCT 26
Subroutine VBVFCT 27
Subroutine SPROP 27
Subroutine VELT 27
Subroutine FQ8DV 27
Subroutine BCPREP 28

Subroutine FASEMB 29
Subroutine FQ8 29
Subroutine BASE 29
Subroutine FBC 29
Subroutine Q4S 30
Subroutine BLKITR 30
Subroutine SOLVE 30
Subroutine PPCG 31
Subroutine POLYP 31
Subroutine ILUCG 31
Subroutine LLTINV 31
Subroutine FPRINT 31
Subroutine FSTORE 31
Subroutine FSFLOW 32
Subroutine Q8TH 33
Subroutine CHEMI 34
Subroutine AFABTA 34
Subroutine FLUX 34
Subroutine DISPC 34
Subroutine TQ8DV 34
Subroutine TASEMB 35
Subroutine TQ8 35
Subroutine SHAPE 36
Subroutine TBC 36
Subroutine Q4CNVB 37
Subroutine TPRINT 37
Subroutine TSTORE 37
Subroutine TSFLOW 38
Subroutine Q4BB 40
Subroutine Q8R 40
Subroutine THNODE 41
Subroutine ADVBC 41
Subroutine BTGN 42
Subroutine TRACK1H 42
Subroutine TRACK2H 43
Subroutine PLANE 43
Subroutine LOCQ8 43
Subroutine ALGBDY 43
Subroutine BNDRY 44
Subroutine LOCPLN 44
Subroutine BASE2D 44
Subroutine XSI2D 44
Subroutine BASE 45
Subroutine XSI3D 45

3—Adaptation of 3DSALT to Site-Specific Applications 46

Parameter Specifications 46
Soil Property Function Specifications 54

Input and Output Devices 56

4—Sample Problems 57

Problem No. 1: One-Dimensional Column Flow Problem 57
Input for Problem No. 1 60
Problem No. 2: One-Dimensional Column Transport 60
Input for Problem No. 2 63
Problem No. 3: Three-Dimensional Salt Intrusion Problem 63
Input for Problem No. 3 68

References 80

Appendix A: Data Input Guide A1

Title A1
Option Parameters A2
Iteration Parameters A6
Time Control Parameters A6
Material Properties A8
Soil Properties A9
Nodal Coordinate A11
Subregion Data A11
Element Data A13
Material Type Correction A14
Card Input for Initial or Pre-Initial Conditions A14
Element (Distributed) Source/Sink for Flow Simulations A16
Point (Well) Source/Sink Data for Flow Simulation A17
Element (Distributed) Source/Sink
for Transport Simulations A18
Point (Well) Source/Sink Data for Transport Simulation A20
Rainfall/Evaporation-Seepage Boundary Conditions A21
Dirichlet Boundary Conditions for Flow Simulation A24
Cauchy Boundary Conditions for Flow Simulations A26
Neumann Boundary Conditions for Flow Simulations A28
Run-In/Flow-Out (Variable) Boundary Conditions
for Transport Simulations A30
Dirichlet Boundary Conditions for Transport Simulations A32
Cauchy Boundary Conditions for Transport Simulation A33
Neumann Boundary Conditions for Transport Simulations A36
Hydrological Variables A38
End of Job A39

Appendix B: Mathematical Formulation B1

Governing Equations for Flow B1
Governing Equations for Transport B11

Appendix C: Numerical Formulation C1

Numerical Approximation of the Flow Equations C2
Spatial discretization with the Galerkin finite element method C2
Base and weighting functions C5
Numerical integration C6
Mass lumping option C10
Finite difference approximation in time C11
Numerical implementation of boundary conditions C13
Solution of the matrix equations C15
Transport Equation C16
Spatial discretization with the weighted residual finite element method C16
Base and weighting functions C20
Numerical integration C21
Mass lumping option C23
Finite difference approximation in time C23
Numerical Implementation of Boundary Conditions C25
Solution of the Matrix Equations C28

SF 298

List of Figures

Figure 1.1. Common examples of hydrogeological conditions in coastal aquifers 7

Figure 1.2. Idealized cross-sections of a layered constal aquifer 8

Figure 1.3. Illustration of transition zone and circulation 10

Figure 2.1. Program structure of 3DSALT 20

Figure 4.1. Problem definition for the one-dimensional transient flow in a soil column 58

Figure 4.2. Problem definition for the one-dimensional transient transport problem in a soil column 63

Figure 4.3. Problem definition for the three-dimensional transient salt intrusion problem 65

Figure 4.4. Discretization of the region with five planes (four layers) and nine subregions 67

Figure C.1. A hexahedral element in local coordinates C6

Figure C.2. A surface area and its imbedded local coordinate C9

Figure C.3. Weighting factor along a line element C20

Figure C.4. Upstream weighting factors along 12 sides of a hexahedral element C21

Preface

This report on a three-dimensional finite element model of density-dependent flow and transport through saturated-unsaturated media was prepared for the U.S. Army Engineer District, Wilmington.

The study was conducted as part of the Cape Fear Groundwater Modeling Study in the Hydraulics Laboratory (HL) of the U.S. Army Engineer Waterways Experiment Station (WES) during the period November 1992 to April 1993 under the direction of Messrs. F. A. Herrmann, Jr., Director, HL; R. A. Sager, Assistant Director, HL; and W. H. McAnally, Jr., Chief, Estuaries Division (ED), HL.

The report was prepared by Drs. Gour-Tsyh Yeh, Jing-Ru Chang, and Jin-Ping Gwo, Pennsylvania State University; and Dr. Hsin-Chi Lin, Estuarine Engineering Branch (EEB), ED; Mr. David R. Richards, Chief, Estuarine Simulation Branch, ED, and Mr. William D. Martin, Chief, EEB.

At the time of publication of this report, Director of WES was Dr. Robert W. Whalin. Commander was COL Bruce K. Howard, EN.

1 Introduction

Background

Due to population expansion as well as agricultural and industrial growth, pollution of freshwater aquifers is becoming more and more apparent, especially when considering the increasing demand on the quality and quantity of fresh water. Often when a contaminant is introduced into the groundwater system, clear changes in the groundwater density occur that may be sufficiently large to alter the flow dynamics of the system. The pollutant may either displace or mix with the fresh water. The consequence is frequently the degradation or loss of the water resource and the need to seek alternative supplies of fresh water or to purify the polluted water body. The best-known case of such an occurrence is saltwater intrusion. Saltwater intrusion often occurs when, due to the rising demand for fresh water, groundwater is excessively pumped to satisfy this need. The hydraulic gradients that are produced from the excessive pumping may induce a flow of saline water toward the pumping well. Thus, this seawater encroachment can easily upset the long-term natural equilibrium between the fresh water and seawater. Inevitably the seawater wedge moves inland, encroaching on the underground supply of fresh water.

The major causes of saltwater intrusion are overpumping in coastal areas, excessive pumping in noncoastal regions which overlay saline water bodies, advancement of salt water through leaky well casings, and natural sources and processes such as drought or tidal variations (Atkinson et al. 1986). Such encroachment will obviously limit the groundwater for domestic, agricultural, or industrial purposes. Hence, there is a need to predict the location and movement of the saltwater interface in order to be able to protect freshwater aquifers from the possible danger of contamination. Practical management also includes some knowledge of not only the present response, but also of the long-term transient response. For these managerial purposes, a numerical model can easily assist in estimating the location of the salt water for given sets of hydrologic conditions.

In the past, several numerical models have been used to predict the location and movement of the saltwater interface for different types of problems. Depending on the method of treating the interface, these numerical models can

be categorized as the following: (a) sharp interface models and (b) diffused (dispersed) interface models (Contractor and Srivastava 1990). The former type was used to investigate the saltwater interface by a number of researchers (Liu et al. 1981; Henry 1959; Shamir and Dagan 1971; Wilson and Sa da Costa 1982). However, in many cases, the sharp interface assumption is justified only to provide an appropriate simulation under certain conditions, such as when the width of the transition zone is relatively small compared to the thickness of the aquifer. The sharp interface assumption was applied by many investigators since when combined with the Dupuit assumption of horizontal flow, this assumption greatly simplifies the model. Various numerical methods, such as the finite difference methods, finite element methods, and the method of characteristics, have been applied in sharp interface models, some with much success, some with less success. In addition, numerical models based on the boundary integral equation method, assuming an abrupt interface, have been presented (Liggett and Liu 1979; Liu et al. 1981).

Nevertheless, the sharp interface approach can be troublesome when the change in the shape of intrusion is large and/or the aquifer system is complex. This becomes quite apparent when applying the finite element method to the problem of the interface. If, as in the sharp interface approach, the fresh water and salt water are assumed to be immiscible, then certain conditions along the interface boundary must be satisfied. Hence, when the finite element method is applied, the position of the interface must be specified in order to partition each fluid region into individual elements. Needless to say, the finite element method becomes quite difficult in the sharp interface model.

Therefore, a simulation model, such as the diffusive interface model, which accounts for the hydrodynamic effects of dispersion, is much more practical since it gives more details concerning the transition zone, whereas the sharp interface model only represents the overall flow characteristics of the system. Also, the diffusive interface model annihilates the difficulty due to the inner boundary even if the aquifer system is quite complex (Essaid 1990).

As early as 1964, Henry developed the first solution for the steady-state salt distribution in a confined coastal aquifer. He assumed a constant dispersive mechanism in the aquifer and concluded that the steady-state condition is in dynamic equilibrium due to the gravitational forces and dispersion that create a saltwater convection cell. Henry's problem was restated by Lee and Cheng (1974) in terms of stream functions. They formulated a numerical solution which assumed constant dispersion. In 1975, Segol, Pinder, and Gray (1975) developed the first transient solution based on a velocity-dependent dispersion coefficient using the Galerkin finite element method to solve the set of nonlinear partial differential equations describing the movement of a saltwater front in a coastal confined aquifer. Numerous other researchers, such as Pinder and Cooper (1970), Andrews (1981), and more recently Frind (1982a,b), and Huyakorn et al. (1987) have used numerical models for simulation of saltwater intrusion problems using the diffusive interface approach. Some of the numerical diffusive interface models

unfortunately do not consider density-dependent fluid flow and solute transport for mathematical simplification reasons. On the other hand, many models (Pinder and Cooper 1970; Lee and Cheng 1974; Frind 1982a,b; Huyakorn et al. 1987) do. In many cases, however, a steady-state solution in transient simulations was not obtained due to high computing costs.

Adequate knowledge about the physical dynamics of the phenomenon of saltwater encroachment is necessary for the proper management of coastal groundwater resources. Hence, in order to portray the physical complexities and also the temporal and spatial variations involved with saltwater intrusion, the development of numerical models has become quite essential. For this purpose, a Three-Dimensional Finite Element Model for Density-Dependent Flow and Transport Through Saturated-Unsaturated Porous Media (3DSALT) has been developed. This model stems from the combination and modification of two previous codes, a groundwater flow model (FEMWATER, Yeh 1987) and a subsurface contaminant transport model (LEWASTE, Yeh 1992). In the newly combined model, density-dependent effects are accounted for, since according to Reilly and Goodman (1985), it is necessary to consider the sea-water intrusion problem as a density-dependent flow and transport problem in order to account for the dispersed nature of the saltwater-freshwater interface and the associated saltwater circulation zones.

Even though the model, 3DSALT, can be used to investigate saturated-unsaturated flow alone, contaminant transport alone, or combined flow and transport, in this report the code will be used to study seawater intrusion problems, thus using the last option. The code will be verified with similar simulations of other numerical models.

In addition, general facts, such as sources, effects, and control of seawater intrusion, as well as physical and mathematical theory, will be presented to complete this study of saltwater intrusion.

In comparison, sea water is around 2.5 percent heavier than fresh water. Based on the relation, a 12.5-m freshwater column is needed to keep a 12.2-m seawater column in balance. Therefore, within a reasonable distance from the ocean, theoretically every 0.30 m of fresh water above sea level signifies the existence of 12.2 m of fresh water in the aquifer below sea level. To alleviate the endless danger of sea water encroaching inland, the freshwater levels must be maintained as high as practicable above sea level (Atkinson et al. 1986).

Unfortunately, saltwater intrusion in coastal areas occurs all over the world. Investigation of the sources of salt water intrusion is very crucial since saltwater is probably the most common contaminant in fresh water. In the case of coastal aquifers, it arises from a seawater invasion. In all too many cases, human activities are directly or indirectly responsible for saltwater intrusion in coastal environments, which are often heavily urbanized.

According to Atkinson et al. (1986), salt water present in aquifers may derive from the following sources:

a. Seawater, in coastal regions.

b. Seawater that penetrated aquifers during past geological time.

c. Evaporated water residue left over in tidal lagoons, playas, etc.

d. Salt from thin salt beds or salt domes or disseminated in geological formations.

e. Saline wastewaters from human activities.

f. Return flows from irrigated land to stream.

Saltwater intrusion into freshwater aquifers can be influenced in various ways. For example, if the groundwater gradients are reduced or reversed, then denser, saline water can easily take the place of the fresh water. This occurrence is quite common in coastal aquifers, which are hydraulically continuous with the ocean and in which excess well pumping has disturbed the hydrodynamic equilibrium. Another example is when natural barriers separating the fresh water and salt water are removed, or when there is a subsurface disposal of waste salt waters (Atkinson et al. 1986).

Saltwater intrusion can have negative and undesirable effects. Humans may experience health and welfare problems related to decreased water quality. As little as 2 percent of seawater in fresh water can make it undrinkable. Wildlife and fish may also be adversely affected by either high salinity of springs used for watering or high saline runoff. High saltwater content in irrigation waters may decrease crop productivity and make it essential to change to salt-tolerant crops. In addition, salt water can be unacceptable for many industrial purposes (Atkinson et al. 1986).

To control or combat all the possible adverse effects of saltwater intrusion, a control program must be implemented that takes into consideration the type of encroachment, the hydrologic conditions of the region in question, the areal extent of the problem, as well as the specific source(s). The control of saltwater intrusion can be summarized in a general approach of five steps (Atkinson et al. 1986) :

a. Problem definition.

b. Inventory and analysis.

c. Formulation of alternative control plans.

d. Comparative evaluation of control plans.

e. Selection and implementation of controls.

The first and probably the most important step to controlling seawater encroachment is locating and defining the magnitude of the problem. Groundwater monitoring can be used for that purpose. After completion of the first step, an inventory of water users is taken to identify patterns, especially if overpumping is occurring. Also, the development of mathematical or numerical models comes about here to help predict and understand the movement of the salt water. The third step involves the formulation of various alternative seawater intrusion control plans. The fourth step involves the comparative evaluation of control plans, in other words, to investigate if the water quality cannot be brought to the desired levels by other methods than control. The last step involves the formulation of legal and institutional considerations in order to implement the selected method of control (Atkinson et al. 1986).

According to Atkinson et al. (1986), the objective of seawater intrusion control depends on the planned function of water and involves one of the following:

a. Partial or complete avoidance of fresh water migrating seaward.

b. Increasing the rate of flow within the aquifer or the size of the freshwater lens by increasing the freshwater pressures.

c. Preserving a state of seawater intrusion that will not further encroach on the freshwater supply by controlling several methods of freshwater withdrawal in given regions.

In order to meet these objectives, the following methods can be applied in the control of seawater intrusion (Atkinson et al. 1986):

a. Directly recharge the aquifer.

b. Reduce or, in some cases, eliminate pumping.

c. Relocate or disperse pumping wells.

d. Form a hydraulic barrier by recharging fresh water into pumping wells parallel to the coast.

e. Remove encroaching salt water by constructing a trough parallel to the coast.

f. Remove seawater before it reaches the pumping well.

g. Create impermeable subsurface barriers.

h. Combine extraction/injection techniques.

Just which control technique to use in which case can be summarized in Table 1.1 (Bowen 1986).

Table 1.1
Techniques of Saltwater Control

Cause of Intrusion	Control Techniques
Saltwater in a coastal aquifer	Alteration of the pumping pattern Injection freshwater well Injection barrier Extraction barrier Subsurface barrier
Upconing	Alternation of the pumping pattern Saline removing wells
Defective well casing	Plugging defective wells
Saline water zones in freshwater aquifers	Relocating and designing wells
Surface infiltration	Eliminating the surface source
Oil field brine	Injection wells Eliminating surface disposal

There are various hydrogeologic conditions in coastal aquifers. Some of the most common examples are depicted in Figure 1.1 (Essaid 1990). Figures 1.1(a) and (b) portray an unconfined aquifer with an impermeable bottom and an unconfined island aquifer with a free bottom, respectively, whereas Figure 1.1(c) shows a coastal confined aquifer.

Figure 1.2 depicts an idealized cross section of a layered coastal aquifer under steady-state and transient conditions. In the steady-state case (Figure 1.2(a)), there is a stable seaward hydraulic gradient within each aquifer. The location and shape of a stationary "interface" between the fresh water and salt water is determined by the freshwater potential and gradient. As the seawater flows in from the sea within every aquifer layer, a wedge-shaped body of denser salt water settles underneath the lighter fresh water. Fresh water in the lower (confined) aquifers may leak upward through the overlying layers and/or discharge through the outcrop, while fresh water in the top (unconfined) aquifer discharges to the sea via the ocean floor. In a system, such as Figure 1.2(a), the zone of mixed fresh water and salt water will not be static since there might be fresh water leaking vertically upward into an overlying saltwater zone. However, if the system were of a one-layer aquifer configuration, the seawater would be nearly static.

On the other hand, in the transient case (Figure 1.2(b)), salt water may flow into the aquifer system by leaking into the confining layers as well as ocean floor and/or by entering through the outcrop. Gradually the "interface" will move inland and encroach on the freshwater supply. Hence, the dynamics of both the freshwater and saltwater domains must be investigated in order to get a complete picture of the seawater intrusion in coastal aquifers, especially when developing numerical models for saltwater intrusion problems (Essaid 1990).

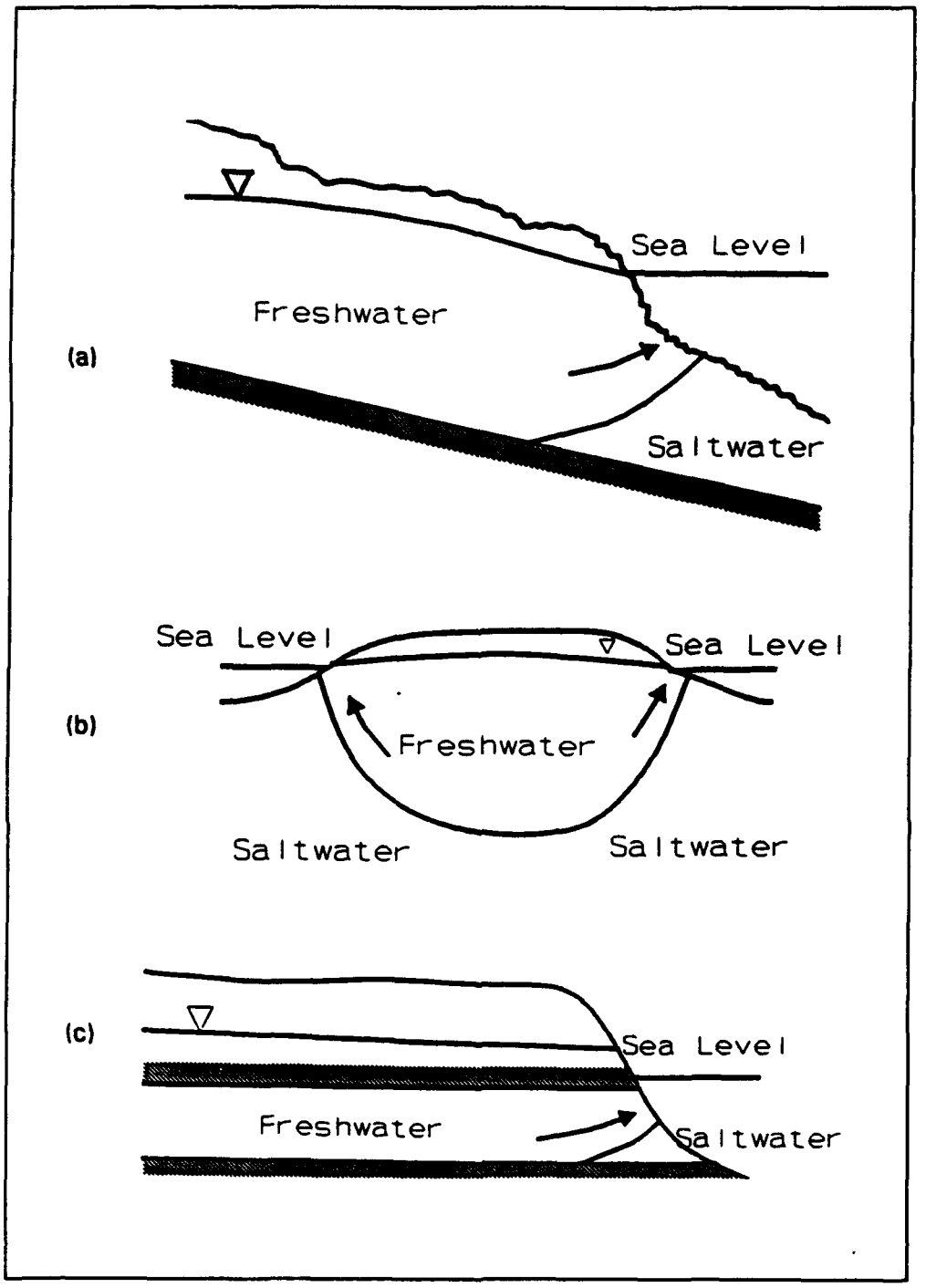


Figure 1.1. Common examples of hydrogeological conditions in coastal aquifers (from Essaid 1990)
(a) Phreatic aquifer with an impermeable bottom
(b) Phreatic island aquifer with a free bottom
(c) Confined aquifer

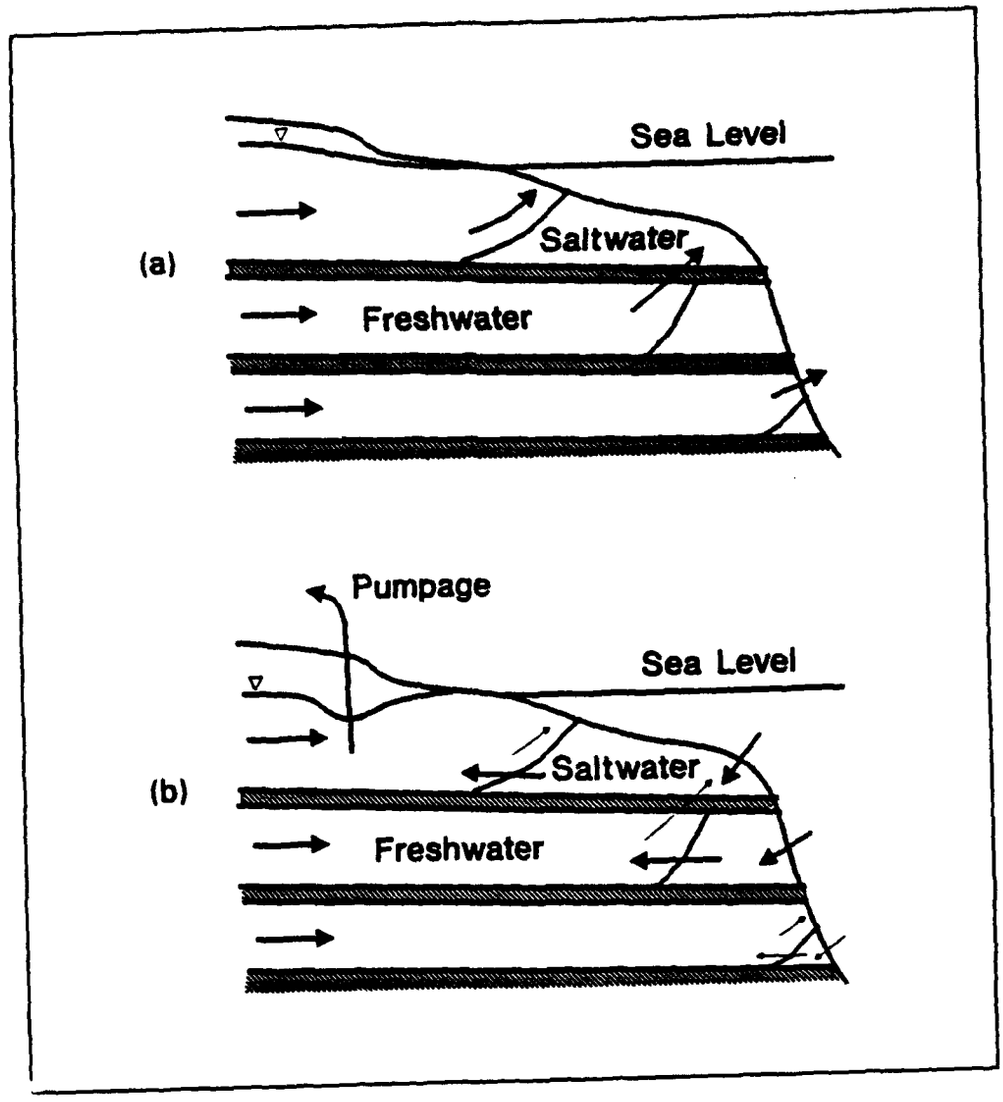


Figure 1.2. Idealized cross-sections of a layered coastal aquifer (from Essaid 1990)
(a) Steady-state case
(b) Transient case

In the freshwater region of coastal aquifers, the flow can sometimes be easily altered by inland changes of discharge or recharge. If the freshwater flow towards the sea is reduced by some means, this may cause the freshwater-saltwater interface to migrate landward, thus resulting in saltwater intrusion into the aquifer. On the other hand, if the freshwater flow towards the sea is increased, the interface may be forced to move towards the sea. Nevertheless, the rate of the interface movement as well as the transient aquifer head response will be determined by the properties of the aquifer and the boundary conditions on both sides of the interface. Generally, the changes in inland freshwater discharge that determine the rate of the interface movement in the aquifer affect the freedom of the salt water to move into or

out of a coastal aquifer system. Hence, it is very important to examine the interface and describe its properties in a realistic fashion (Essaid 1990).

Both disperse and sharp interface approaches have been used by numerous researchers to study saltwater intrusion in coastal aquifers via numerical models. However, as also discussed, the sharp interface approach may be an adequate first approximation in some cases, but in may other cases, the zone of dispersion might be quite extensive, making the sharp interface approach a very poor approximation. In some studies, such as field observations in the Biscayne aquifer, Florida, for example, the error of using a sharp instead of a disperse interface approach can be as much as a few miles seaward. Such an error clearly demonstrates that a sharp interface model cannot fully represent the nature of saltwater intrusion, at least not for some coastal aquifers that experience a thick mixing zone generated by freshwater and saltwater dispersion (Lee and Cheng 1974). In addition, it has been revealed by actual measurements that the dispersion-diffusion phenomena may heavily contribute to notable fluctuations of the water table in coastal aquifers (De Wiest 1965).

One of the early researchers, Beran (1955), investigating the freshwater-saltwater interface, described three cases of flow: (a) when the effects of molecular diffusion are prevailing; (b) when the randomness of the flow pattern is as significant as the molecular diffusion in the mixing process; (c) when the effects of randomness of the flow pattern and molecular diffusion are insignificant (Sherif, Singh, and Amer 1990).

As stated, in reality, where the fresh water and salt water merge, a dispersion zone of finite thickness occurs due to the effects of hydrodynamic dispersion (mechanical dispersion and molecular diffusion). No distinct interface exists since the fresh water and sea water are considered soluble in each other. This transition zone also is influenced by the action of tides and well pumping. Maximum widths of the transition zone occur in extremely permeable coastal aquifers that are exposed to heavy pumping. According to Volker and Rushton (1982), the extent of the dispersion zone is dependent on numerous factors, such as the dispersion parameters of the coastal aquifer and the rate of discharge of the groundwater, as well as the relative densities of the fresh water and salt water. In this dispersion zone, the concentration and thus the fluid density vary. Dispersion results in a change of concentration of the displacing fluid in the transition zone, basically due to the fact that individual fluid particles travel at variable velocities through the irregularly and randomly shaped pore channels of the medium. The flow pattern in the aquifer will obviously be altered by this transition zone. Since the transition zone is moving towards the sea, the saline water coming from underlying sources flows in the same direction. Therefore, due to continuity, the flow is inland in the saltwater region. In addition, the groundwater salinity increases with depth from that of fresh water to that of salt water in the transition zone (Bowen 1986).

Figure 1.3 illustrates the transition zone between the fresh water and salt water in a coastal aquifer. It can be seen that the salt water tends to force

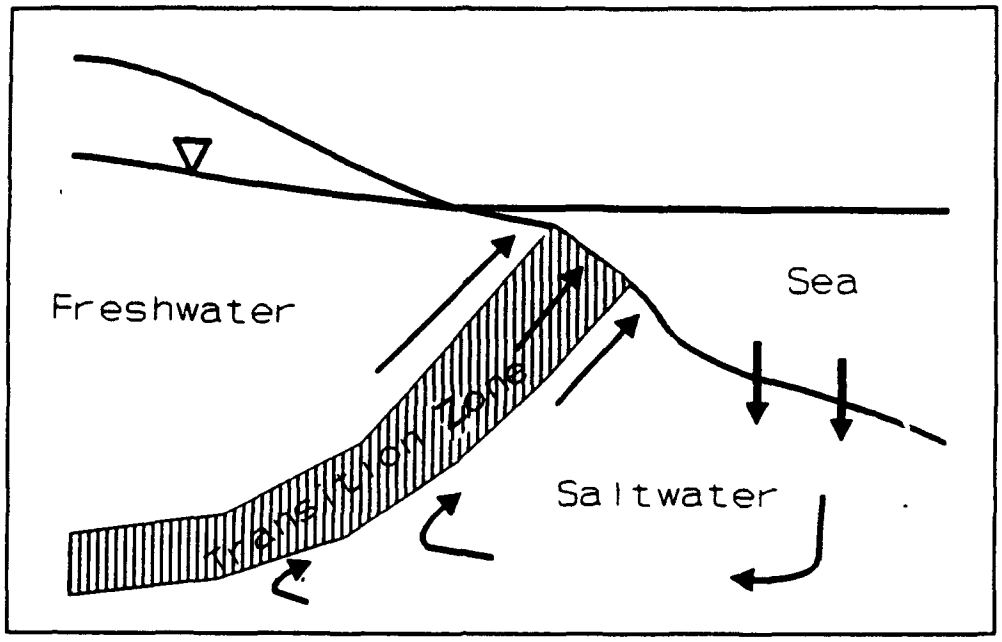


Figure 1.3. Illustration of transition zone and circulation (from Essaid 1990)

itself underneath the fresh water due to the higher density of the salt water. The "diluted" salt water ascends and moves towards the sea along the interface due to the fact that it is less dense than the original seawater. The transfer of salts out of the saltwater environment induces circulation. Due to this movement, there exists a cyclic flow of saltwater originating from the sea, across the ocean floor, to the transition zone, and back to the sea. Even under steady-state conditions, this cyclic flow is evident (Essaid 1990).

The model developed in this research is designed to solve a system of governing equations pertaining both to flow and transport through saturated-unsaturated media. Numerical simulation of contaminant transport in subsurface systems involves the solution of two partial differential equations. The first differential equation is the flow equation that describes the head distribution in the aquifer of interest. For the developed model, the classically used pressure head variable employed in the fluid continuity equation of many flow modules was replaced by the use of equivalent freshwater head that generally results in the elimination of static quantities and the improvement of numerical efficiency (Frind 1982b). If the head distribution is known, then the flow can be calculated via Darcy's law. The other differential equation is the transport (dispersion) equation which is used to describe the chemical concentration. In the specific case of saltwater intrusion, a constitutive equation that relates fluid density to concentration is also needed (Galeati, Gambolati, and Neuman 1992). Furthermore, the two partial differential equations are coupled in such a way that makes, for instance, the seawater intrusion problem nonlinear. Buoyancy effects that cause the upward movement of the fresh water and sea water near the coast primarily affect the degree of nonlinearity (Huyakorn et al. 1987). The coupling is solved in such a way that the groundwater flow and solute transport equations are solved independently and linked through

iterations. In addition, initial and boundary conditions must be accounted for when solving this system of governing equations.

Purpose

The purpose of this report is to provide guidance for users to use the 3DSALT model for site-specific application especially for salt intrusion problems in coastal areas.

3DSALT (A Three-Dimensional Finite Element Model of Density-Dependent Flow And Transport Through Saturated-Unsaturated Media) can be used to investigate saturated-unsaturated flow alone, contaminant transport alone, combined flow and transport, or salt intrusion problems in subsurface media. For the flow module, the Galerkin finite element method is used to discretize the Richards equation; and for the transport module, the hybrid Lagrangian-Eulerian finite element method is used to discretize the transport equation. Using the hybrid Lagrangian-Eulerian approach completely eliminates numerical oscillation due to advection transport. Large time-step sizes can be used to overcome excessive numerical dispersion. The only limitation on the size of time-step is the requirement of accuracy with respect to dispersion transport, which does not pose severe restrictions.

Scope

The scope of this report is to derive and solve the governing equations for density-dependent flow and transport in saturated-unsaturated media. The report also provides the description of a main program and subroutines. Three sample problems were provided to illustrate the application of using the model.

The section "Purpose of 3DSALT," Chapter 2, lists the governing equations and describes initial and boundary conditions for which 3DSALT is designed to provide solutions. The section "Description of 3DSALT Subroutines," Chapter 2, contains the description of all subroutines in 3DSALT. This should facilitate the understanding of the code structure by the users. Since occasions may arise when the users have to modify the code, this section should help them to trace the code so they can make necessary adjustments for their purposes. General information on input parameters required by each subroutine is also provided. The section "Parameter Specifications," Chapter 3, contains the parameter specification. For each application, users must assign 58 maximum control-integers. The section "Soil Property Function Specifications," Chapter 3, describes soil property function specifications so that the users will be able to modify subroutine SPRO for each site-specific application. The section "Input and Output Devices," Chapter 3, describes files required for the execution of 3DSALT. Appendix A contains the data input guide that is essential for any site-specific application.

The users may choose whatever units they want to use provided they are maintained in all the input. Units of mass (M), length (L), and time (T) are indicated in the input description.

The special features of 3DSALT are its flexibility and versatility in modeling a range of real-world problems. The model is designed to do the following:

a. Treat heterogeneous and anisotropic media consisting of as many geologic formations as desired.

b. Consider both distributed and point sources/sinks that are spatially and temporally dependent.

c. Accept the prescribed initial conditions or obtain them by simulating a steady-state version of the system under consideration.

d. Deal with transient Dirichlet boundary conditions.

e. Handle time-dependent fluxes due to the gradient of pressure head or concentration varying along the Neumann boundary.

f. Treat time-dependent total fluxes distributed over the Cauchy boundary.

g. Automatically determine variable boundary conditions of evaporation, infiltration, or seepage on the soil-air interface for the flow module and variable boundary conditions of inflow and outflow for the transport module.

h. Include the off-diagonal hydraulic conductivity components in Richards equation for dealing with cases when the coordinate system does not coincide with the principal directions of the hydraulic conductivity tensor.

i. Give three options for estimating the nonlinear matrix.

j. Include two options (successive subregion block iterations and successive point iterations) for solving the linearized matrix equations.

k. Provide two options of treating the mass matrix — consistent and lumping.

l. Provide three adsorption models in the transport module — linear isotherm and nonlinear Langmuir and Freundlich isotherms.

m. Automatically reset time-step size when boundary conditions or source/sinks change abruptly.

n. Check the mass balance computation over the entire region for every time-step.

Appendix B provides the physical bases and mathematical foundation for describing density-dependent flow and material transport. Appendix C gives the numerical detail in approximating the governing equations. Readers who wish to comprehend salt intrusion problems and understand numerical approaches should read these two appendices. For practitioners they may be skipped.

2 The 3DSALT Program Structure

Purpose of 3DSALT

3DSALT is designed to solve the following system of governing equations along with initial and boundary conditions, which describe flow and transport through saturated-unsaturated media. The governing equations for flow are basically the modified Richards equation, which is derived in Appendix B.

Governing flow equation

$$\frac{\rho}{\rho_o}\frac{d\theta}{dh}\frac{\partial h}{\partial t} = \nabla\cdot\left[\mathbf{K}\cdot\left[\nabla h+\frac{\rho}{\rho_o}\nabla z\right]\right] + \frac{\rho^*}{\rho_o}q \qquad (2.1)$$

where h is the pressure head, t is time, $\mathbf{K}$ is the hydraulic conductivity tensor, z is the potential head, q is the source and/or sink, ρ is the water density at chemical concentration C, ρ_o is the referenced water density at zero chemical concentration, ρ^* is the density of either the injection fluid or the withdrawn water, and θ is the moisture content. The hydraulic conductivity $\mathbf{K}$ is given by

$$\boldsymbol{K} = \frac{\rho g}{\mu}\mathbf{k} = \frac{(\rho/\rho_o)}{(\mu/\mu_o)}\frac{\rho_o g}{\mu_o}\mathbf{k}_s k_r = \frac{\rho/\rho_o}{\mu/\mu_o}\mathbf{K}_{so}k_r \qquad (2.2a)$$

where μ is the dynamic viscosity of water at chemical concentration C; μ_0 is the referenced dynamic viscosity at zero chemical concentration; $\mathbf{k}$ is the permeability tensor; $\mathbf{k}_s$ is the saturated permeability tensor; k_r is the relative permeability or relative hydraulic conductivity; $\mathbf{K}_{so}$ is the referenced saturated hydraulic conductivity tensor. The referenced value is usually taken at zero

chemical concentration. The density and dynamic viscosity of water are functions of chemical concentration and are assumed to take the following form

$$\frac{\rho}{\rho_o} = a_1 + a_2 C + a_3 C^2 + a_4 C^3 \qquad (2.2b)$$

and

$$\frac{\mu}{\mu_o} = a_5 + a_6 C + a_7 C^2 + a_8 C^3 \qquad (2.2c)$$

where a_1, a_2, ..., a_8 are the parameters used to define concentration dependence of water density and viscosity and C is the chemical concentration.

The Darcy velocity is calculated as follows

$$V = -\mathbf{K} \cdot \left[\frac{\rho_o}{\rho} \nabla h + \nabla z \right] \qquad (2.3)$$

Initial conditions for flow equation

$$h = h_i(x,y,z) \quad in \quad R, \qquad (2.4)$$

where R is the region of interest and h_i is the prescribed initial condition, which can be obtained by either field measurements or by solving the steady-state version of Equation 2.1.

Boundary conditions for flow equations

Dirichlet conditions:

$$h = h_d(x_b,y_b,z_b,t) \quad on \quad B_d \qquad (2.5)$$

Neumann conditions:

$$-\boldsymbol{n}\cdot\boldsymbol{K}\cdot\frac{\rho_o}{\rho}\nabla h = q_n(x_b,y_b,z_b,t) \quad on \quad B_n, \tag{2.6}$$

Cauchy conditions:

$$-\boldsymbol{n}\cdot\boldsymbol{K}\cdot\left[\frac{\rho_o}{\rho}\nabla h+\nabla z\right] = q_c(x_b,y_b,z_b,t) \quad on \quad B_c, \tag{2.7}$$

Variable conditions during precipitation period:

$$h = h_p(x_b,y_b,z_b,t) \quad on \quad B_v \tag{2.8a}$$

or

$$-\boldsymbol{n}\cdot\boldsymbol{K}\cdot\left[\frac{\rho_o}{\rho}\nabla h+\nabla z\right] = q_p(x_b,y_b,z_b,t) \quad on \quad B_v, \tag{2.8b}$$

Variable conditions during nonprecipitation period:

$$h = h_p(x_b,y_b,z_b,t) \quad on \quad B_v, \tag{2.8c}$$

or

$$h = h_m(x_b,y_b,z_b,t) \quad on \quad B_v, \tag{2.8d}$$

or

$$-\boldsymbol{n}\cdot\boldsymbol{K}\cdot\left[\frac{\rho_o}{\rho}\nabla h+\nabla z\right] = q_e(x_b,y_b,z_b,t) \quad on \quad B_v, \tag{2.8e}$$

where (x_b,y_b,z_b) is the spatial coordinate on the boundary; **n** is an outward unit vector normal to the boundary; h_d, q_n, and q_c are the prescribed Dirichlet functional value, Neumann flux, and Cauchy flux, respectively; B_d, B_n, and B_c

are the Dirichlet, Neumann, and Cauchy boundaries, respectively; B_v is the variable boundary; h_p is the allowed ponding depth; and q_p is the throughfall of precipitation, respectively, on the variable boundary; h_m is the allowed minimum pressure on the variable boundary; and q_e is the allowed maximum evaporation rate on the variable boundary, which is the potential evaporation. Only one of Equations 2.8a through 2.8e is used at any point on the variable boundary at any time.

The governing equations for transport are derived based on the continuity of mass and flux laws as given in Appendix B. The major processes are advection, dispersion/diffusion, adsorption, decay, and source/sink.

Governing equations for transport

$$\theta\frac{\partial C}{\partial t} + \rho_b\frac{\partial S}{\partial t} + V\cdot\nabla C = \nabla\cdot(\theta D\cdot\nabla C) - \lambda(\theta C + \rho_b S) + QC_{in} - \left[\frac{\rho^*}{\rho}Q - \frac{\rho_o}{\rho}V\cdot\nabla\left[\frac{\rho}{\rho_o}\right]\right]C \tag{2.9}$$

$$S = K_d C \quad \textit{for linear isotherm} \tag{2.10a}$$

$$S = \frac{S_{max}KC}{1 + KC} \quad \textit{for Langmuir isotherm} \tag{2.10b}$$

$$S = KC^n \quad \textit{for Freundlich isotherm} \tag{2.10c}$$

where θ is the moisture concentration, ρ_b is the bulk density of the medium (M/L^3), C is the material concentration in aqueous phase (M/L^3), S is the material concentration in adsorbed phase (M/M), t is time, **V** is the discharge, ∇ is the del operator, **D** is the dispersion coefficient tensor, λ is the decay constant, Q is the source rate of water, C_{in} is the material concentration in the source, K_d is the distribution coefficient, S_{max} is the maximum concentration allowed in the medium in the Langmuir nonlinear isotherm, n is the power index in the Freundlich nonlinear isotherm, and K is the coefficient in the Langmuir or Freundlich nonlinear isotherm.

The dispersion coefficient tensor **D** in Equation 2.9 is given by

$$\theta D = a_T|V|\delta + (a_L - a_T)VV/|V| + \theta a_m\tau\delta \tag{2.11}$$

where $|V|$ is the magnitude of V, δ is the Kronecker delta tensor, a_T is lateral dispersivity, a_L is the longitudinal dispersivity, a_m is the molecular diffusion coefficient, and τ is the tortuosity.

Initial conditions for transport:

$$C = C_i(x,y,z) \quad in \quad R \tag{2.12}$$

Prescribed concentration (Dirichlet) boundary conditions:

$$C = C_d(x_b,y_b,z_b,t) \quad on \quad B_d \tag{2.13}$$

Variable boundary conditions:

$$n\cdot(VC - \theta D\cdot\nabla C) = n\cdot VC_v(x_b,y_b,z_b,t) \; if \; n\cdot V \leq 0 \tag{2.14a}$$

$$n\cdot(-\theta D\cdot\nabla C) = 0 \; if \; n\cdot V > 0 \tag{2.14b}$$

Cauchy boundary conditions:

$$n\cdot(VC - \theta D\cdot\nabla C) = q_c(x_b,y_b,z_b,t) \quad on \quad B_c \tag{2.15}$$

Neumann boundary conditions:

$$n\cdot(-\theta D\cdot\nabla C) = q_n(x_b,y_b,z_b,t) \quad on \quad B_n \tag{2.16}$$

where C_i is initial concentration; R is the region of interest; (x_b,y_b,z_b) is the spatial coordinate on the boundary; n is an outward unit vector normal to the boundary; C_d and C_v are the prescribed concentration on the Dirichlet boundary and the specified concentration of water through the variable boundary, respectively; B_d and B_v are the Dirichlet and variable boundaries, respectively; q_c and q_n are the prescribed total flux and gradient flux through the Cauchy and Neumann boundaries B_c and B_n, respectively.

Since the hybrid Lagrangian-Eulerian approach is used to simulate Equation 2.9, it is written in the Lagrangian-Eulerian form as

$$\left[\theta + \rho_b \frac{ds}{dc}\right] \frac{D_{V_d} C}{Dt} = \nabla\cdot(\theta D\cdot\nabla C) - \lambda(\theta C + \rho_b S) + QC_{in} - \left[\frac{\rho^*}{\rho Q} - \frac{\rho_o}{\rho} V\cdot\nabla\left[\frac{\rho}{\rho_o}\right]\right] C \tag{2.17a}$$

$$V_d = \frac{V}{\theta + \rho_b K_d} \quad \textit{for linear isotherm model} \tag{2.17b}$$

$$\theta \frac{D_{V_f} C}{Dt} + \rho_b \frac{dS}{dC}\frac{\partial C}{\partial t} = \nabla\cdot(\theta D\cdot\nabla C) - \lambda(\theta C + \rho_b S) + QC_{in} - \left[\frac{\rho^*}{\rho} Q - \frac{\rho_o}{\rho} V\cdot\nabla\left[\frac{\rho}{\rho_o}\right]\right] C \tag{2.18a}$$

$$V_f = \frac{V}{\theta} \quad \textit{for Freundlich and Langumir models} \tag{2.18b}$$

where V_d and V_f are the retarded and fluid pore velocities, respectively; and $D_{V_d}()/dt$ and $D_{V_f}()/dt$ denote the material derivative of ( ) with respect to time using the retarded and fluid pore velocities, respectively.

The flow equation (2.1) subject to initial and boundary conditions (Equations 2.5 through 2.8) is solved with the Galerkin finite element method. The transport equations (Equations 2.17 or 2.18) subject to initial and boundary conditions (Equations 2.12 through 2.16) are solved with the hybrid Lagrangian-Eulerian finite element methods. Detail implementation of the numerical approximation of flow and transport problems are given in Appendix C.

Description of 3DSALT Subroutines

3DSALT consists of a MAIN program and 57 subroutines. The program structure of 3DSALT is illustrated in Figure 2.1. The functions of the MAIN program and the subroutines are described below.

Program MAIN

The MAIN is used to specify the sizes of all arrays. The flow of data input for the model is also anchored by the MAIN. The subroutine RDATIO is called to read the geometric and material data. MAIN then calls subroutine PAGEN to generate pointer arrays; SURF to identify the boundary sides and

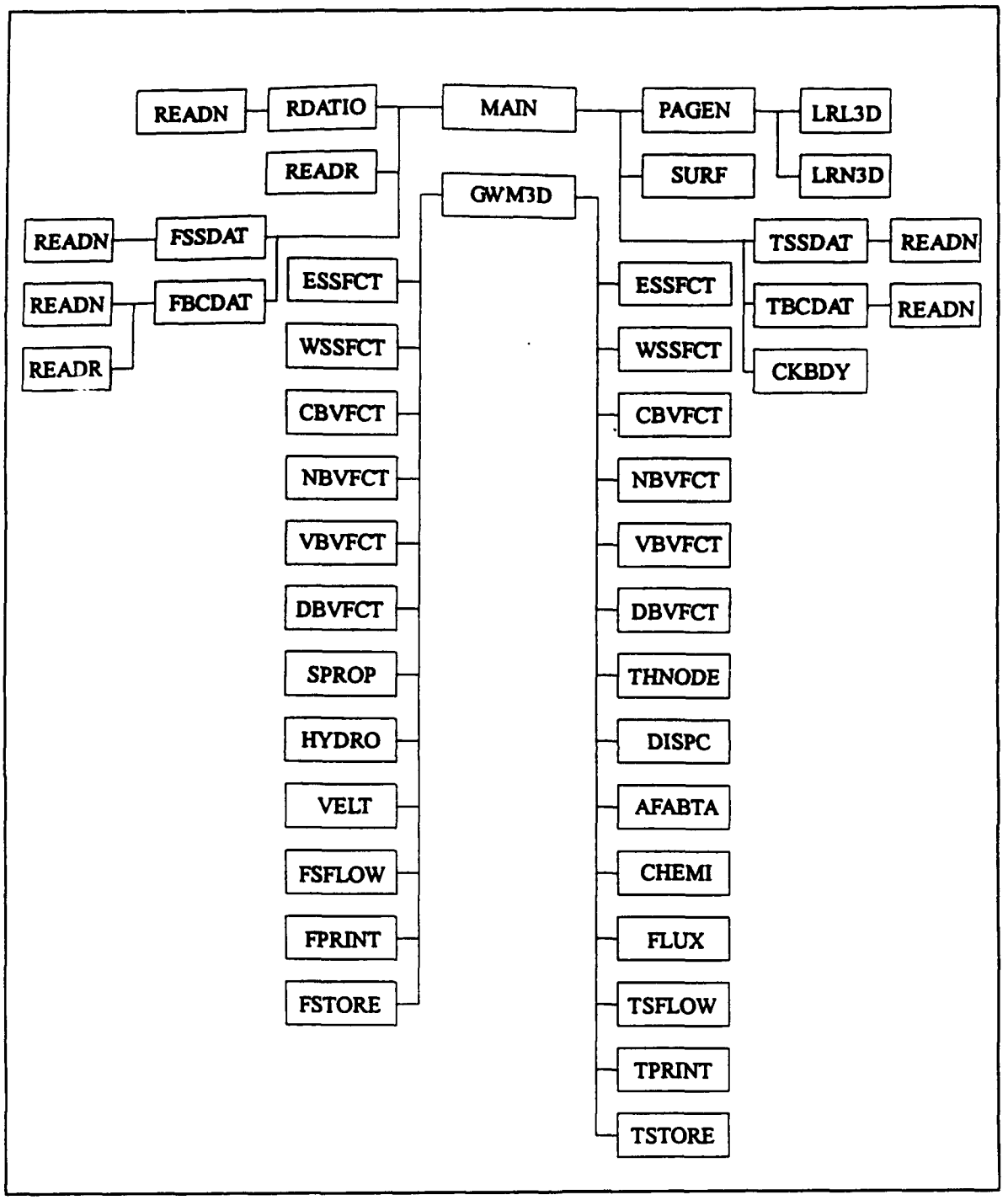


Figure 2.1 Program structure of 3DSALT (Sheet 1 of 3)

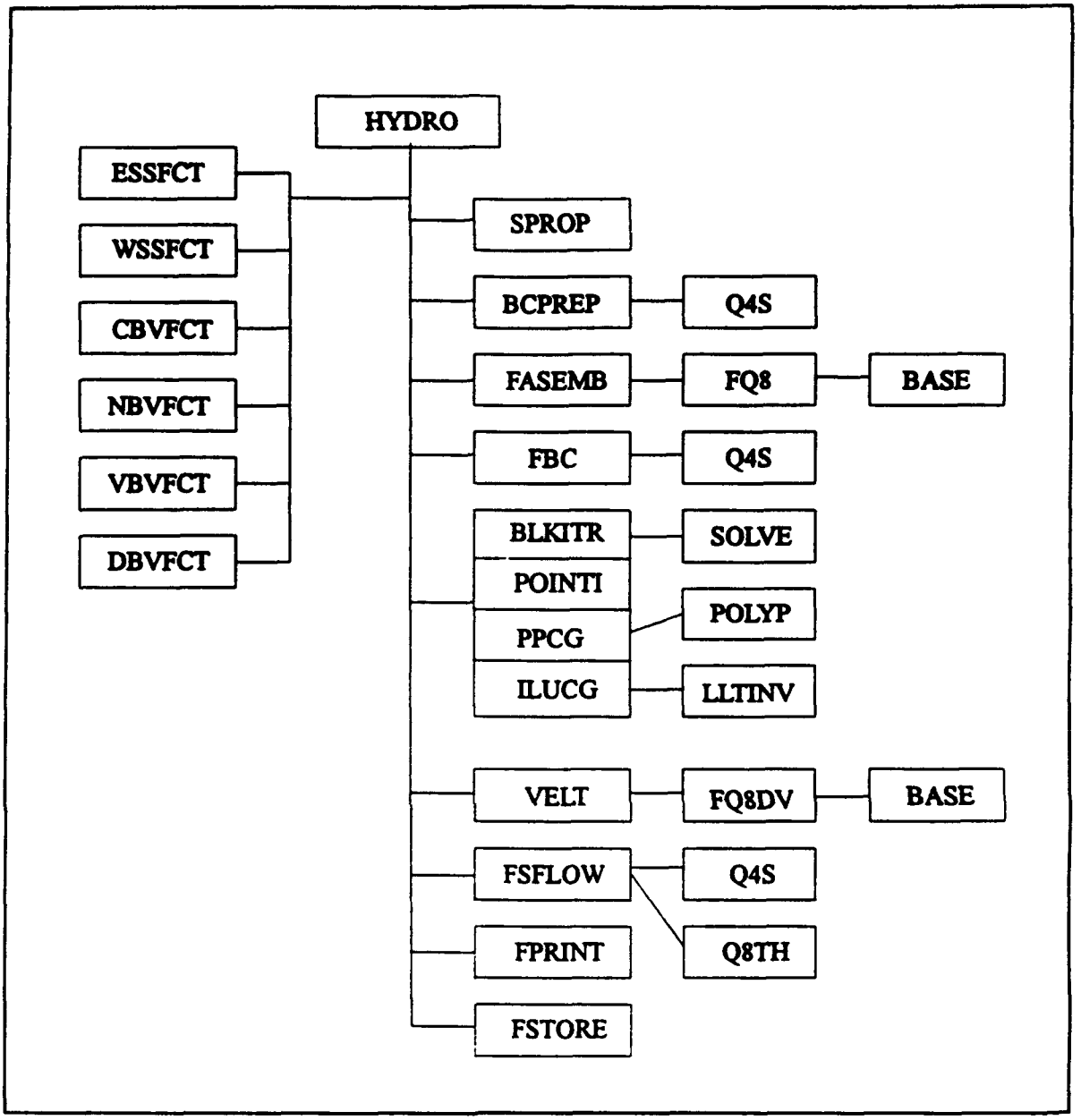


Figure 2.1. (Sheet 2 of 3)

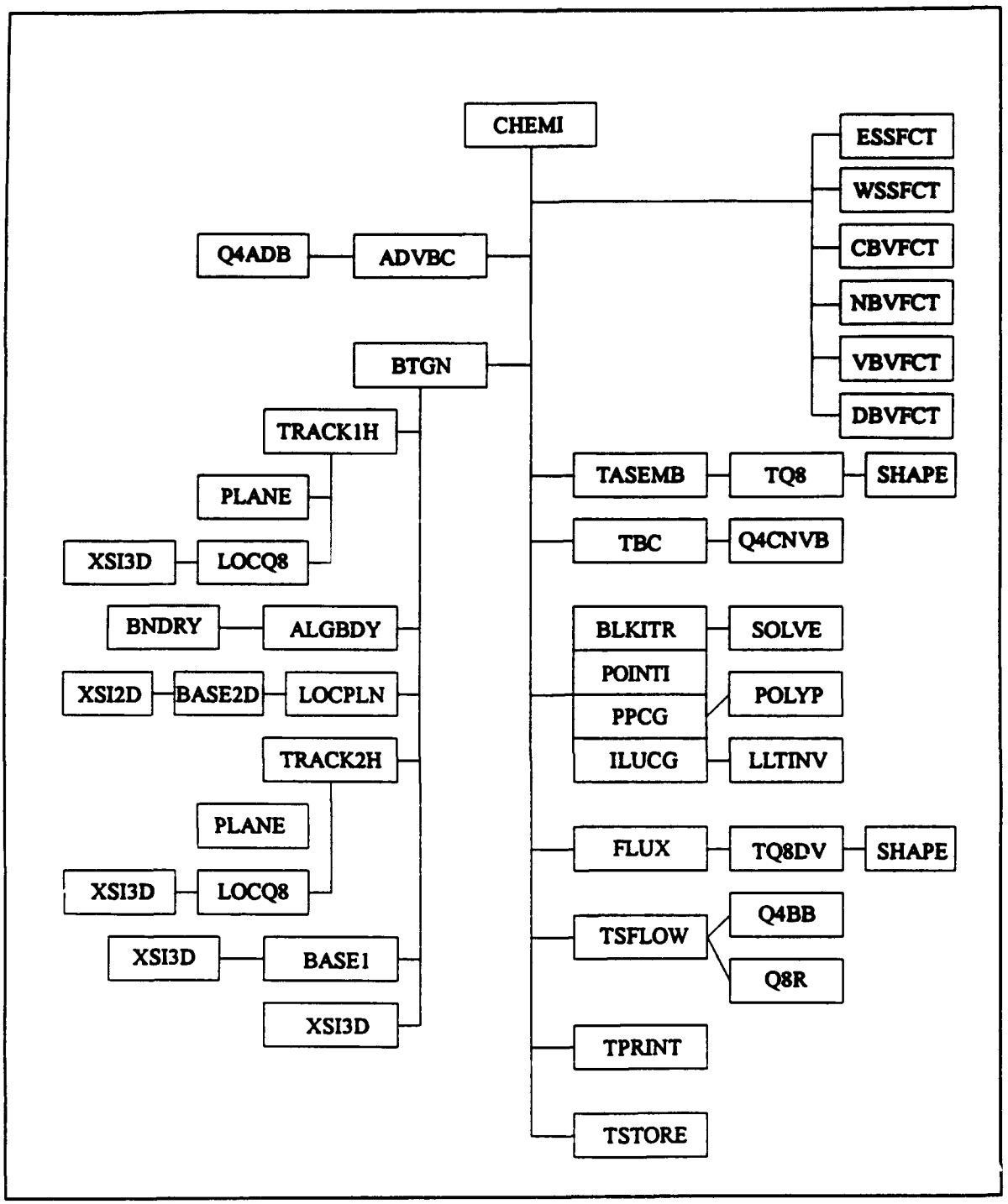


Figure 2.1. (Sheet 3 of 3)

compute the directional cosine. The source/sink data for flow and transport computations are read in by the subroutines FSSDAT and TSSDAT, respectively. The boundary conditions for flow and transport calculations are then read in by the subroutines FBCDAT and TBCDAT, respectively. Control is then passed to subroutine GWM3D to coordinate and perform flow and/or transport computations.

Subroutine RDATIO

The subroutine RDATIO is called by the program MAIN to read in the soil property functions and geometric data for the area of interest.

Subroutine FSSDAT

The subroutine FSSDAT is called by the program MAIN to read in the sources/sinks profiles, nodes, and/or elements for flow simulations. The source/sink type for each node/element is also assigned in this subroutine according to the data given by the users.

Subroutine TSSDAT

The subroutine TSSDAT is called by the program MAIN to read in the sources/sinks profiles, nodes, and/or elements for transport simulations. The source/sink type for each node/element is also assigned in this subroutine according to the data given by the users.

Subroutine FBCDAT

The subroutine FBCDAT controls the input of boundary condition, in time and space, assigned to each boundary node/element for flow simulations. Users need to give the boundary profiles, to specify the global node/element numbers of the boundary, and to assign boundary profile type to each node/element.

Subroutine TBCDAT

The subroutine TBCDAT controls the input of boundary condition, in time and space, assigned to each boundary node/element for transport simulations. Users need to give the boundary profiles, to specify the global node/element numbers of the boundary, and to assign boundary profile type to each node/element.

Subroutine GWM3D

The subroutine GWM3D controls the entire sequence of operations, a function generally performed by the MAIN program. It is, however, preferable to keep a short MAIN and supplement it with several subroutines with variable storage allocation. This makes it possible to place most of the FORTRAN deck on a permanent file and to deal with a site-specific problem without making changes in array dimensions throughout all subroutines.

Depending on the combinations of the parameters KSSf, KSSt, NTI, and IMOD, the subroutine GWM3D will perform either the steady-state flow and/or transport computations only, or the transient-state flow and/or transport computations using the flow and/or transport steady-state solution as the initial conditions, or the transient flow and/or transport computation using user-supplied initial conditions.

GWM3D calls the subroutines ESSFCT, WSSFCT, CBVFCT, NBVFCT, VBVFCT, and DBVFCT to obtain sources/sinks and boundary values; subroutine SPROP to obtain the relative hydraulic conductivity, water capacity, and moisture content from the pressure head; subroutine VELT to compute Darcy's velocity; subroutine FSFLOW to calculate flux through all types of boundaries and water accumulated in the media; subroutine FPRINT to print out the results; subroutine FSTORE to store the flow variables for plotting; subroutine HYDRO to perform the flow computations; subroutine FLUX to compute material flux; subroutine AFABTA to obtain upstream weighting factor based on velocity and dispersivity; subroutine TSFLOW to calculate material flux through all types of boundaries and water accumulated in the media; subroutine TPRINT to print out the transport computation results; subroutine TSTORE to store the transport computation results for plotting; subroutine THNODE to compute the value of moisture content plus bulk density times distribution coefficient in the case of linear isotherm, or the moisture content in the case of nonlinear isotherm at all nodes; subroutine DISPC to compute the dispersion coefficients; and subroutine CHEMI to perform the transport computations.

Subroutine HYDRO

HYDRO calls subroutines ESSFCT, WSSFCT, CBVFCT, NBVFCT, VBVFCT, and DBVFCT to obtain sources/sinks and boundary values; subroutine SPROP to obtain the relative hydraulic conductivity, water capacity, and moisture content from the pressure head; subroutine VELT to compute Darcy's velocity; subroutine BCPREP to determine if a change of boundary conditions is required; subroutine FASEMB to assemble the element matrices over all elements; subroutine FBC to implement the boundary conditions; subroutine BLKITR, POINTI, PPCG, or ILUCG to solve the matrix equations; subroutine FSFLOW to calculate flux through all types of boundaries and water accumulated in the media; subroutine FPRINT to print out the

results; and subroutine FSTORE to store the flow variables in binary format for plotting.

Subroutine SURF

Subroutine SURF identifies the boundary sides, sequences the boundary nodes, and computes the directional cosine of the surface sides. The mappings from boundary nodes to global nodes are stored in NPBB(I) (where NPBB(I) is the global node number of i^{th} boundary node). The boundary node numbers of the four nodes for each boundary side are stored in ISB(I,J) (where ISB(I,J) is the boundary node number of I^{th} node of J^{th} side, I = 1 to 4). There are six sides for each element. Which of these six sides is the boundary side is determined automatically in the subroutine SURF and is stored in ISB(5,J). The global element number, to which the J^{th} boundary side belongs, is also preprocessed in the subroutine SURF and is stored in ISB(6,J). The directional cosines of the J^{th} boundary side are computed and stored in DCOSB(I,J) (where DCOSB(I,J) is the directional cosine of the J^{th} surface with I^{th} coordinate, I = 1 to 3). The information contained in NPBB, ISB, and DOSB, along with the number of boundary nodes and the number of boundary sides, is returned to subroutine DATAIN for other users.

Subroutine READR

This subroutine is called by the subroutines RDATIO, FBCDAT, and TBCDAT to generate real numbers. Automatic generation of regularly patterned data is built into this subroutine.

Subroutine READN

This subroutine is called by the subroutines RDATIO, FSSDAT, TSSDAT, FBCDAT, and TBCDAT to generate integer data automatically.

Subroutine PAGEN

This subroutine is called by the controlling subroutine GWM3D to preprocess pointer arrays that are needed to store the global matrix in compressed form and to construct the subregional block matrices. The pointer arrays automatically generated in this subroutine include the global node connectivity (stencil) GNOJCN(J,N), regional node connectivity LNOJCN(J,I,K), total node number for each subregion NTNPLR(K), bandwidth indicator for each subregion LMAXDF(K), and partial fill-up for the mapping array between global node number and local subregion node number GNPLR(I,k) with I = NNPLR(K) + 1 to NTNPLR(K). Here GNOJCN(J,N) is the global node number of J^{th} node connected to the global node N; LNOJCN(J,I,K) is the local node number of J^{th} node connected to the local node I in K^{th} subregion;

NTNPLR(K) is the total number of nodes in the K^{th} subregion, including the interior nodes, the global boundary nodes, and intraboundary nodes; LMAXDF(K) is the maximum difference between any two nodes of any element in K^{th} subregion; and GNPLR(I,K) is the global node number of I^{th} local-region node in the K^{th} subregion. These pointer arrays are generated based on the element connectivity IE(M,J), the number of node for each subregion NNPLR(K), and the mapping between global node and local-region node GNLR(I,K) with I = 1, NNPLR(K). Here IE(M,J) is the global node number of J^{th} node of element M; NNPLR(K) is the number of nodes in the K^{th} subregion including the interior nodes and the global boundary nodes but not the intraboundary nodes.

Subroutine ESSFCT

This subroutine is called by the subroutines GWM3D, HYDRO, and CHEMI to compute the element source/sink strength. It uses the linear interpolation of the tabular data or uses analytical formulae. If the analytical formulae are used, the users must supply the functions.

Subroutine WSSFCT

This subroutine is called by the subroutines GWM3D, HYDRO, and CHEMI to compute the point source/sink strength. It uses the linear interpolation of the tabular data or uses analytical formulae. If the analytical formulae are used, the users must supply the functions.

Subroutine DBVFCT

This subroutine is called by the subroutines GWM3D, HYDRO, and CHEMI to compute the pressure at the Dirichlet boundary. It uses the linear interpolation of the tabular data or uses analytical formulae. If the analytical formulae are used, the users must supply the functions.

Subroutine CBVFCT

This subroutine is called by the subroutines GWM3D, HYDRO, and CHEMI to compute the total flux at Cauchy boundary. It uses the linear interpolation of the tabular data or uses analytical formulae. If the analytical formulae are used, the users must supply the functions.

Subroutine NBVFCT

This subroutine is called by the subroutines GWM3D, HYDRO, and CHEMI to compute the gradient flux at Neumann boundary. It uses the linear

interpolation of the tabular data or uses analytical formulae. If the analytical formulae are used, the users must supply the functions.

Subroutine VBVFCT

This subroutine is called by the subroutines GWM3D, HYDRO, and CHEMI to compute the infiltration rate or evaporation rate flux at variable boundary. It uses the linear interpolation of the tabular data or uses analytical formulae. If the analytical formulae are used, the users must supply the functions.

Subroutine SPROP

This subroutine calculates the values of moisture content, relative hydraulic conductivity, and the water capacity. Either tabular input or analytical functions can be used to represent soil property function. When analytical functions are used, the users must supply the functional form.

Subroutine VELT

This subroutine calls FQ8DV to evaluate the element matrices and the derivatives of the total head. It then sums over all element matrices to form a matrix equation governing the velocity components at all nodal points. To save computational time, the matrix is diagonalized by lumping. The velocity components can thus be solved point by point. The computed velocity field is then returned to GWM3D or HYDRO through the argument. This velocity field is also passed to subroutine BCPREP to evaluate the Darcy flux across the seepage-infiltration-evapotranspiration surfaces.

Subroutine FQ8DV

Subroutine FQ8DV is called by the subroutine VELT to compute the element matrices given by

$$QR(I,J) = \int_{B_e} N_i^e N_j^e dR \ , \qquad (2.19a)$$

where N_i^e and N_j^e are the basis functions for nodal point i and j of element e, respectively. Subroutine Q8DV also evaluates the element load vector:

$$QRX(I) = -\int_{R_e} N_i^e \mathbf{i}\cdot\mathbf{K}\cdot\frac{\rho_o}{\rho}(\nabla N_j^e)h_j dR - \int_{R_e} N_i^e \mathbf{i}\cdot\mathbf{K}\cdot\nabla z dr \qquad (2.19b)$$

$$QRY(I) = -\int_{R_e} N_i^e \mathbf{j}\cdot\mathbf{K}\cdot\frac{\rho_o}{\rho}(\nabla N_j^e)h_j dR - \int_{R_e} N_i^e \mathbf{j}\cdot\mathbf{K}\cdot\nabla z dr \qquad (2.19c)$$

$$QRZ(I) = -\int_{R_e} N_i^e \mathbf{k}\cdot\mathbf{K}\cdot\frac{\rho_o}{\rho}(\nabla N_j^e)h_j dR - \int_{R_e} N_i^e \mathbf{k}\cdot\mathbf{K}\cdot\nabla z dr \qquad (2.19d)$$

where

h_j = the referenced pressure head at nodal point j

$\mathbf{i}$ = the unit vector along the x-coordinate

$\mathbf{j}$ = the unit vector along the y-coordinate

$\mathbf{k}$ = the unit vector along the z-coordinate

$\mathbf{K}_s$ = the saturated hydraulic conductivity tensor

K_r = the relative hydraulic conductivity

Subroutine BCPREP

This subroutine is called by HYDRO to prepare the infiltration-seepage boundary conditions during a rainfall period or the seepage-evapotranspiration boundary conditions during nonrainfall periods. It decides the number of nodal points on the variable boundary to be considered as Dirichlet or Cauchy points. It computes the number of points that change boundary conditions from ponding depth (Dirichlet types) to infiltration (Cauchy types), or from infiltration to ponding depth, or from minimum pressure (Dirichlet types) to infiltration during rainfall periods. It also computes the number of points that change boundary conditions from potential evapotranspiration (Cauchy types) to minimum pressure, or from ponding depth to potential evapotranspiration, or from minimum pressure to potential evapotranspiration during nonrainfall periods. Upon completion, this subroutine returns the Darcy flux (DCYFLX), infiltration/potential evapotranspiration rate (FLX), the ponding depth nodal index (NPCON), the flux-type nodal index (NPFLX), the minimum pressure nodal index (NPMIN), and the number of nodal points (NCHG) that have changed boundary conditions.

Subroutine FASEMB

This subroutine calls FQ8 to evaluate the element matrices. It then sums over all element matrices to form a global matrix equation governing the pressure head at all nodes.

Subroutine FQ8

This subroutine is called by the subroutine FASEMB to compute the element matrix given by

$$QA(I,J) = \int_{R_e} N_i^e \frac{\rho}{\rho_o} \frac{d\theta}{dh} N_j^e dR \ , \tag{2.20a}$$

$$QB(I,J) = \int_{R_e} (\nabla N_i^e) \cdot \mathbf{K} \cdot (\nabla \mathrm{N_j^e}) \mathrm{dR} \ , \tag{2.20b}$$

where F is the soil property function. Subroutine FQ8 also calculates the element load vector given by

$$RQ(I) = \int_{R_e} [(\nabla N_i^e) \cdot \mathbf{K} \cdot \frac{\rho}{\rho_o} (\nabla z) - \mathrm{N_i^e} \frac{\rho^*}{\rho_o} q] \mathrm{dR} \ , \tag{2.20c}$$

where q is the source/sink.

Subroutine BASE

This subroutine is called by subroutines FQ8DV and FQ8 to evaluate the value of the base function at a Gaussian point. The computation is straightforward.

Subroutine FBC

This subroutine incorporates Dirichlet, Cauchy, Neumann, and variable boundary conditions. For a Dirichlet boundary condition, an identity algebraic equation is generated for each Dirichlet nodal point. Any other equation having this nodal variable is modified accordingly to simplify the computation. For a Cauchy surface, the integration of the surface source is obtained by calling the subroutine Q4S, and the result is added to the load

vector. For a Neumann surface, the integrations of both the gradient and gravity fluxes are obtained by calling the subroutine Q4S. These fluxes are added to the load vector. The subroutine BC also implements the variable boundary conditions. First, it checks all infiltration-evapotranspiration-seepage points, identifying any of them that are Dirichlet points. If there are Dirichlet points, the method of incorporating Dirichlet boundary conditions mentioned above is used. If a given point is not the Dirichlet point, the point is bypassed. Second, it checks all rainfall-evaporation-seepage points again to see if any of them is a Cauchy point. If it is a Cauchy point, then the computed flux by infiltration or potential evapotranspiration is added to the load vector. If a given point is not a Cauchy point, it is bypassed. Because the infiltration-evaporation-seepage points are either Dirichlet or Cauchy points, all points are taken care of in this manner.

Subroutine Q4S

This subroutine is called by the subroutines FBC and FSFLOW to compute the surface node flux of the type

$$RQ(I) = \int_{B_e} N_i^e \frac{\rho}{\rho_o} q dB \ , \tag{2.21}$$

where q is either the Cauchy flux, Neumann flux, or gravity flux.

Subroutine BLKITR

This subroutine is called by the subroutines HYDRO and CHEMI to solve the matrix equation with block iteration methods. For each subregion, a block matrix equation is constructed based on the global matrix equation and two pointer arrays GNPLR and LNOJCN (see subroutine PAGEN), and the resulting block matrix equation is solved with the direct band matrix solver by calling subroutine SOLVE. This is done for all subregions for each iteration until a convergent solution is obtained. This subroutine and the subroutine SOLVE, to be described in the next paragraph, are needed only for the code BLI.

Subroutine SOLVE

This subroutine is called by the subroutine BLKITR to solve for the matrix equation of the type

$$[C]\{x\} = \{y\} \tag{2.22}$$

where [C] is the coefficient matrix and {x} and {y} are two vectors. {x} is the unknown to be solved, and {y} is the known load vector. The computer returns the solution {y} and stores it in {y}. The computation is a standard banded Gaussian direct elimination procedure.

Subroutine PPCG

This subroutine is called by the subroutines HYDRO and CHEMI, if necessary, to solve the linearized matrix equation with the preconditioned conjugate gradient method using the polynomial as a preconditioner. It calls to POLYP to invert the preconditioner.

Subroutine POLYP

This subroutine is called by the subroutine PPCG to solve for a modified residual that will be used in the preconditioned conjugate gradient algorithm.

Subroutine ILUCG

This subroutine is called by the subroutines HYDRO and CHEMI, if necessary, to solve the linearized matrix equation with the preconditioned conjugate gradient method using the incomplete Cholesky decomposition as a preconditioner. It calls to LLTINV to invert the preconditioner.

Subroutine LLTINV

This subroutine is called by the subroutine ILUCG to solve for a modified residual that will be used in the preconditioned conjugate gradient algorithm.

Subroutine FPRINT

This subroutine is used to line-print the flow variables. These include the fluxes through variable boundary surfaces, the pressure head, total head, moisture content, and Darcy's velocity components.

Subroutine FSTORE

This subroutine is used to store the flow variables on Logical Unit 1. It is intended for use for plotting. The information stored includes region

geometry, subregion data, and hydrological variables such as pressure head, total head, moisture content, and Darcy's velocity components.

Subroutine FSFLOW

This subroutine is used to compute the fluxes through various types of boundaries and the increasing rate of water content in the region of interest. The function of FRATE(7) is to store the flux through the whole boundary enclosing the region of interest. It is given by

$$FRATE(7) = \int_B (V_x n_x + V_y n_y + V_z n_z) dB \ , \tag{2.23}$$

where B is the global boundary of the region of interest; V_x, V_y, and V_z are Darcy's velocity components; and n_x, n_y, and n_z are the directional cosines of the outward unit vector normal to the boundary B. FRATE(1) through FRATE(5) store the flux through Dirichlet boundary B_D, Cauchy boundary B_C, Neumann boundary B_N, the seepage/evapotranspiration boundary B_S, and infiltration boundary B_R, respectively, and are given by

$$FRATE(1) = \int_{B_d} (V_x n_x + V_y n_y + V_z n_z) dB \ , \tag{2.24a}$$

$$FRATE(2) = \int_{B_c} (V_x n_x + V_y n_y + V_z n_z) dB \ , \tag{2.24b}$$

$$FRATE(3) = \int_{B_n} (V_x n_x + V_y n_y + V_z n_z) dB \ , \tag{2.24c}$$

$$FRATE(4) = \int_{B_s} (V_x n_x + V_y n_y + V_z n_z) dB \ , \tag{2.24d}$$

$$FRATE(5) = \int_{B_r} (V_x n_x + V_y n_y + V_z n_z) dB \ , \tag{2.24e}$$

FRATE(6), which is related to the numerical loss, is given by

$$FRATE(6) = FRATE(7) - \sum_{I=1}^{5} FRATE(I) \tag{2.25}$$

FRATE(8) and FRATE(9) are used to store the source/sink and increased rate of water within the media, respectively:

$$FRATE(8) = -\int_R \frac{\rho_*}{\rho} q dR \ , \tag{2.26}$$

and

$$FRATE(9) = \int_R \frac{\rho}{\rho_o} \frac{d\theta}{dh} \frac{\partial h}{\partial t} dR \ , \tag{2.27}$$

If there is no numerical error in the computation, the following equation should be satisfied:

$$FRATE(9) = -[FRATE(7) + FRATE(8)] \tag{2.28}$$

and FRATE(6) should be equal to zero. Equation 2.19 simply states that the negative rate of water going out from the region through the entire boundary and due to a source/sink is equal to the rate of water accumulated in the region.

Subroutine Q8TH

This subroutine is used to compute the contribution of the increasing rate of the water content from an element e

$$QTHP = \int_{R_e} \frac{\rho}{\rho_o} \frac{d\theta}{dh} \frac{\partial h}{\partial t} dR \ , \tag{2.29}$$

The computation of the above integration is straightforward.

Subroutine CHEMI

The subroutine CHEMI controls the entire sequence of transport computations. CHEMI calls subroutines ESSFCT, WSSFCT, DBVFCT, VBVFCT, CBVFCT, and NBVFCT to obtain sources/sinks and boundary values; subroutine AFABTA to obtain upstream weighting factor based on velocity and dispersivity; subroutine FLUX to compute material flux; subroutine TASEMB to assemble the element matrices over all elements; subroutine TBC to implement the boundary conditions; subroutine BLKITR, POINTI, PPCG, or ILUCG to solve the resulting matrix equations; subroutine TSFLOW to calculate flux through all types of boundaries and water accumulated in the media; subroutine TPRINT to print out the results; subroutine TSTORE to store the results for plotting; subroutine THNODE to compute the value of moisture content plus bulk density times distribution coefficient in the case of linear isotherm, or the moisture content in the case of nonlinear isotherm at all nodes; subroutine BTGN to compute the Lagrangian concentrations at all node, and subroutine ADVBC to implement boundary conditions in the Lagrangian step.

Subroutine AFABTA

This subroutine calculates the values of upstream weighting factors along 12 sides of all elements.

Subroutine FLUX

This subroutine calls TQ8DV to evaluate the element matrices and the derivatives of concentrations. It then sums over all element matrices to form a matrix equation governing the flux components at all nodal points. To save computational time, the matrix is diagonalized by lumping. The flux components due to dispersion can thus be solved point by point. The flux due to the velocity is then added to the computed flux due to dispersion. The computed total flux field is then returned to GM3D through the argument.

Subroutine DISPC

Subroutine DISPC calculates the dispersion coefficient associated with each Gaussian point of an element.

Subroutine TQ8DV

Subroutine TQ8DV is called by the subroutine FLUX to compute the element matrices given by

$$QB(I,J) = \int_{R_e} N_i^e N_j^e dR \ , \tag{2.30a}$$

where N_i^e and N_j^e are the basis functions for nodal point i and j of element e, respectively. Subroutine Q8DV also evaluates the element load vector:

$$QRX(I) = -\int_{R_e} N_i^e \mathbf{i} \cdot \theta \mathbf{D} \cdot (\nabla N_j^e) C_j dR \ , \tag{2.30b}$$

$$QRY(I) = -\int_{R_e} N_i^e \mathbf{j} \cdot \theta \mathbf{D} \cdot (\nabla N_j^e) C_j dR \ , \tag{2.30c}$$

$$QRZ(I) = -\int_{R_e} N_i^e \mathbf{k} \cdot \theta \mathbf{D} \cdot (\nabla N_j^e) C_j dR \ , \tag{2.30d}$$

where C_j is the concentration at nodal point j, **i** is the unit vector along the x-direction, **j** is the unit vector along the y-coordinate, **k** is the unit vector along the z-coordinate, θ is the moisture content, and **D** is the dispersion coefficient tensor.

Subroutine TASEMB

This subroutine calls TQ8 to evaluate the element matrices. It then sums over all element matrices to form a global matrix equation governing the concentration distribution at all nodes.

Subroutine TQ8

This subroutine is called by the subroutine TASEMB to compute the element matrix given by

$$QA(I,J) = \int_{R_e} N_i^e \theta N_j^e dR \ , \tag{2.31a}$$

$$QAA(I,J) = \int_{R_e} N_j^e \rho_b \frac{dS}{dC} N_j^e dR \ , \tag{2.31b}$$

$$QB(I,J) = \int_{R_e} (\nabla N_i^e) \cdot \theta D \cdot (\nabla N_j^e) dR \ , \tag{2.31c}$$

$$QV(I,J) = \int_{R_e} N_i^e V \cdot (\nabla N_j^e) dR \ , \tag{2.31d}$$

$$QC(I,J) = \int_{R_e} N_i^e \left[\lambda \left[\theta + \rho_b \frac{dS}{dC} \right] + \frac{\rho^*}{\rho} q - \frac{\rho_o}{\rho} V \cdot \nabla \left[\frac{\rho}{\rho_o} \right] \right] N_j^e dR \ , \tag{2.31e}$$

where dS/dC should be evaluated at C_w the dissolved concentration at previous iteration. Subroutines Q8 also calculate the element load vector given by:

$$QR(I) = \int_{R_e} N_i^e \left[-\lambda \rho_b \left[S_w - \frac{dS}{dC} C_w \right] + qC_{in} \right] dR \ , \tag{2.31f}$$

where C_w and S_w are the dissolved and adsorbed concentrations at previous iteration, respectively.

Subroutine SHAPE

This subroutine is called by subroutines TQ8DV and TQ8 to evaluate the value of the base and weighting functions and their derivatives at a Gaussian point. The computation is straightforward.

Subroutine TBC

This subroutine incorporates Dirichlet, variable boundary, Cauchy, and Neumann boundary conditions. For a Dirichlet boundary condition, an

identity algebraic equation is generated for each Dirichlet nodal point. Any other equation having this nodal variable is modified accordingly to simplify the computation. For a variable surface, the integration of the normal velocity times the incoming concentration is added to the load vector and the integration of normal velocity is added to the matrix. For the Cauchy boundaries, the integration of Cauchy flux is added to the load vector and the integration of normal velocity is added to the matrix. For the Neumann boundary, the integration of gradient flux is added to the load vector.

Subroutine Q4CNVB

This subroutine is called by the subroutines TBC to compute the surface node flux of the type

$$RQ(I) = \int_{R_e} N_i^e q dB \ , \tag{2.32a}$$

where q is either the Cauchy flux, Neumann flux, or $\mathbf{n} \cdot \mathrm{VC}_v$. It also computes the boundary element matrices

$$BQ(I,J) = \int_{R} N_i^e V N_j^e dR \tag{2.32b}$$

Subroutine TPRINT

This subroutine is used to line-print the simulation results. These include the fluxes through variable boundary surfaces, the concentration, and vertically integrated material flux components.

Subroutine TSTORE

This subroutine is used to store the simulation results on Logical Unit 12. It is intended for plotting purpose. The information stored includes region geometry, concentrations, and vertically integrated material flux components at all nodes for any desired time-step.

Subroutine TSFLOW

This subroutine is used to compute the flux rates through various types of boundaries and the increasing rate of material in the region of interest. FRATE(7) is to store the flux through the whole boundary

$$FRATE(7) = \int_B (F_x n_x + F_y n_y) dB \ , \tag{2.33}$$

where B is the global boundary of the region of interest; F_x, and F_y are the vertically integrated flux components; and n_x, and n_y are the directional cosines of the outward unit vector normal to the boundary B. FRATE(1) stores the flux rates through Dirichlet boundary B_d. FRATE(2) and FRATE(3) store the flux rate through Cauchy and Neumann boundaries, respectively. FRATE(4) and FRATE(5) store incoming flux and outgoing flux rates, respectively, through the variable boundaries B_v^- and B_v^+, as given by

$$FRATE(1) = \int_{B_d} (F_x n_x + F_y n_y) dB \ , \tag{2.34}$$

$$FRATE(2) = \int_{B_c} (F_x n_x + F_y n_y) dB \ , \tag{2.35}$$

$$FRATE(3) = \int_{B_n} (F_x n_x + F_y n_y) dB \ , \tag{2.36}$$

$$FRATE(4) = \int_{B_v^-} (F_x n_x + F_y n_y) dB \ , \tag{2.37}$$

$$FRATE(5) = \int_{B_v^-} (F_x n_x + F_y n_y) dB \ , \tag{2.38}$$

where B_v^- and B_v^+ are that part of the variable boundary where the fluxes are directed into the region and out from the region, respectively. The integration

of Equations 2.16 through 2.20 is carried out by the subroutine Q4BB.

FRATE(6) stores the flux rate through unspecified boundaries as

$$FRATE(6) = FRATE(7) - \sum_{I=1}^{5} FRATE(I) \tag{2.39}$$

FRATE(8) and FRATE(9) store the accumulated rate in dissolved and adsorbed phases, respectively, as given by

$$FRATE(8) = \int_R \theta C dR \ , \tag{2.40}$$

$$FRATE(9) = \int_R \rho_b S dR \ , \tag{2.41}$$

FRATE(10) stores the rate loss due to decay and FRATE(11) through FRATE(13) are set to zero as given by

$$FRATE(10) = \int_R \lambda(\theta C + \rho_b S) dR \ , \tag{2.42}$$

$$FRATE(11) = FRATE(12) = FRATE(13) = 0 \ , \tag{2.43}$$

FRATE(14) is used to store the source/sink rate as

$$FRATE(14) = \int_R \left\{ \left[QC_{in} - \frac{\rho^*}{\rho} QC \right] [1 + sign(Q)]/2 + \frac{\rho_o}{\rho} \mathbf{V} \cdot \left[\frac{\rho}{\rho_o} \right] \right\} dR \tag{2.44}$$

If there is no numerical error in the computation, the following equation should be satisfied:

$$\sum_{I=7}^{14} FRATE(I) = 0 \tag{2.45}$$

and FRATE(6) should be equal to zero.

Subroutine Q4BB

This subroutine is called by the subroutine SFLOW to perform surface integration of the following type

$$RRQ(I) = \int_{R_s} N_i^s F dB \ , \tag{2.46}$$

where F is the normal flux.

Subroutine Q8R

This subroutine is used to compute the contributions to FRATE(8), FRATE(9), FRATE(1), and FRATE(14):

$$QRM = \int_R \theta C dR \ , \tag{2.47a}$$

$$QDM = \int_R S dR \ , \tag{2.47b}$$

$$SOSM = \int_R \left\{ \frac{\rho^*}{\rho} QC_{in}[1 + sign(Q)] + \frac{\rho^*}{\rho} QC[1 - sign(Q)] \right\} / 2 dR \ , \tag{2.47c}$$

The computation of the above integration is straightforward.

Subroutine THNODE

This subroutine is called by CHEMI to compute the ($\theta + \rho_b dS/dC$) for the linear isotherm model or θ for the Freundlich and Langmuir nonlinear isotherm models.

Subroutine ADVBC

This subroutine is called by CHEMI to implement the boundary conditions. For Dirichlet boundary, the Lagrangian concentration is specified. For variable boundaries, if the flow is directed out of the region, the fictitious particle associated with the boundary node must come from the interior nodes. Hence the Lagrangian concentration for the boundary node has already computed from subroutine BTGN and the implementation for such a boundary segment is bypassed. Thus, care should be taken to ensure that the subroutine BTGN is called before the subroutine ADVBC. For variable boundaries, if the flow is directed into the region, the concentration of incoming fluid is specified. An intermediate concentration C^{**} is calculated according to

$$C_i^{**} = \int_{B_r} N_i V_n C_{in} dB / \int_{B_r} N_i V_n dB \ , \qquad (2.48a)$$

where C_i^{**} is the concentration due to the boundary source at the boundary node i, V_n is the normal vertically integrated Darcy's velocity, and C_{in} is the concentration of incoming fluid.

Cauchy boundary conditions are normally applied to the boundary where flow is directed into the region, where the material flux of incoming fluid is specified. The intermediate concentration is thus calculated according to

$$C_i^{**} = \int_{B_r} N_i q_c dB / \int_{B_r} N_i V_n dB \ , \qquad (2.48b)$$

where C_i^{**} is the concentration due to Cauchy fluxes at the boundary node i, V_n is the normal Darcy's velocity, and q_c is the Cauchy flux of the incoming fluid.

The Lagrangian concentration is obtained by using the value C_i^{**} and C_i^n (the concentration at previous time-step) as follows:

$$C_i^* = \frac{\int_B N_i \theta N_j C_i^{**} dB + \int_B N_i \rho_b K_d N_j C_j^* dB}{\int_B N_i(\theta + \rho_b K_d) dB} \quad \text{for linear isotherm} \qquad (2.49a)$$

$$C_i^* = C_i^{**} \quad \text{for nonlinear isotherm} \qquad (2.49b)$$

Subroutine BTGN

This subroutine is called by CHEMI to control the process of backward particle tracking starting from global nodes. It is designed to get the Lagrangian concentrations of all the particles sitting on the global nodes at the current time-step. In the subroutine, each particle is tracked one element by one element until either the tracking time is completely consumed or the particle encounters a specified boundary side. During the particle tracking, this subroutine calls (a) TRACK1H to track a particle in the element being considered if that particle is standing on a global node of the element, and (b) TRACK2H to track a particle if that particle is not on any nodes of the element. In order to make the particle tracking complete and remedy the given velocity field error on the unspecified boundaries, this subroutine calls ALGBDY to continue tracking particles along the unspecified boundaries. At the end of each particle tracking, this subroutine calls (a) LOCPLN if the target point is on an unspecified boundary side, and (b) BASE or XSI3D if the target point is in a determined element to calculate base functions such that the Lagrangian concentration can be computed by interpolation with those base function values.

Subroutine TRACK1H

This subroutine is called by BTGN to compute the particle tracking in a specified element when the source point coincides with a global node of the element. This subroutine calls PLANE to determine (a) whether the particle would move backwards into the element or not, and (b) which side of the element the particle would head onto if the particle does move backwards into this element. After determining which side the particle is going to move onto, this subroutine calls LOCQ8 to compute the exact location of the target point on the side. For accuracy, using the average velocity of both the source point and the target point to locate the target point is firstly considered in the sub-routine. However, if this average velocity approach is not able to deal with very complex velocity fields, the single velocity of the source point is used to determine the location of the target point.

Subroutine TRACK2H

This subroutine is called by BTGN to compute the particle tracking in a specified element when the source point does not coincide with a global node of the element. This subroutine calls PLANE to determine (a) whether the particle would move backwards into the element or not, and (b) which side of the element the particle would head onto if the particle does move backwards into this element. After determining which side the particle is going to move onto, this subroutine calls LOCQ8 to compute the exact location of the target point on the side. For accuracy, using the average velocity of both the source point and the target point to locate the target point is firstly considered in the subroutine. However, if this average velocity approach is not able to deal with very complex velocity fields, the single velocity of the source point is used to determine the location of the target point.

Subroutine PLANE

This subroutine is called by both TRACK1H and TRACK2H to determine which one of the two sides, separated by a specified plane, the particle would move into. All the computations are made according to the average velocity approach and the single velocity approach, when the index parameter IJUDGE is 1 and 2, respectively.

Subroutine LOCQ8

This subroutine is called by both TRACK1H and TRACK2H to locate the target point of a particle tracking in a specified element. All the computations are made according to the average velocity approach and the single velocity approach, when the index parameter IJUDGE is 1 and 2, respectively. When the average velocity approach is considered, the Newton-Raphson method is used to solve a set of two simultaneous nonlinear algebraic equations such that the local coordinates of the target point on the predetermined element side can be determined. With these local coordinates, one is able to calculate all the information at the target point. On the other hand, when the single velocity approach is considered, the location of the target point can be easily determined based on both the velocity of the source point and the geometrical relationship between the source point and the predetermined element side.

Subroutine ALGBDY

This subroutine is called by BTGN to control the process of backward particle tracking along the unspecified boundaries. In the subroutine, the particle tracking is executed one boundary side by one b' 'ndary side based on the nodal velocity component along the side being considered. The tracking will not be stopped until either the tracking time is completely consumed or the particle encounters a specified boundary side. This subroutine calls

BNDRY to track a particle along a predetermined boundary side. For accuracy, using the average velocity of both the source point and the target point to locate the target point is firstly considered in the subroutine. However, if this average velocity approach is not able to deal with very complex velocity fields, the single velocity of the source point is used to determine the location of the target point.

Subroutine BNDRY

This subroutine is called by ALGBDY to locate the target point of a particle tracking along a specified boundary side. All the computations are made according to the average velocity approach and the single velocity approach, when the index parameter IJUDGE is 1 and 2, respectively. For both approaches, the location of the target point can be analytically determined. However, when the velocity field is very complex, there might be no solution with the average approach. Thus, IJUDGE is originally set to 1 and is changed to 2 if the average approach fails. This control is executed in ALGBDY.

Subroutine LOCPLN

This subroutine is called by BTGN to determine the base function values associated with a specified point on a predetermined plane. Firstly, it transforms all the point coordinates, including that of the point and those of the plane nodes, from three-dimensional space to two-dimensional space. Secondly, it calls BASE2D to compute the base function values based on those transformed two-dimensional coordinates.

Subroutine BASE2D

This subroutine is called by LOCPLN to compute the base function values associated with a specified point based on the given two-dimensional global coordinates. For the cases of quadrilateral elements, it calls XSI2D to calculate the local coordinates, and computes base functions with these determined local coordinates. For the cases of triangular elements, the base functions can be analytically determined based on the given global coordinates.

Subroutine XSI2D

This subroutine is called by BASE2D to compute the local coordinate of a quadrilateral element given the global coordinate within that element

Subroutine BASE

This subroutine is called by ETGN to compute the base functions associated with a specified point based on the given global coordinates. It calls XSI3D to compute the local coordinates associated with the point such that the base function values can be easily calculated.

Subroutine XSI3D

This subroutine is called by BASE to compute the local coordinate of a hexahedral element given the global coordinate within that element.

3 Adaptation of 3DSALT to Site-Specific Applications

The following describes how one should apply the 3DSALT code for site-specific applications and how the data file should be prepared.

Parameter Specifications

For each site-specific problem, the users only need to specify the size of the problem by assigning 58 maximum control-integers with PARAMETER STATEMENT in the MAIN program. The list and definitions of the maximum control-integers required for both flow and transport simulations are given below:

a. Maximum control-integers for the spatial domain

MAXNPK = maximum number of nodes
MAXELK = maximum number of elements
MXBESK = maximum number of boundary-element surfaces
MXBNPK = maximum number of boundary nodal points
MXJBDK = maximum number of nonzero elements in any row for nodewise connectivity
MXKBDK = maximum number of nonzero elements in any row for elementwise connectivity

b. Maximum control-integers for the time domain

MXNTIK = maximum number of time-steps
MXDTCK = maximum number of DELT changes

c. *Maximum control-integers for subregions*

LTMXNK = maximum number of total nodal points in any subregion, including interior nodes, global boundary nodes, and interboundary nodes. LTMXNK = 1 if the block iteration is not used.
LMXNPK = maximum number of nodal points in any subregion, including interior nodes and global boundary nodes. LMXNPK = 1 if the block iteration is not used.
LMXBWK = maximum number of the bandwidth in any subregion. LMXBWK = 1 if the block iteration is not used.
MXRGNK = maximum number of subregions. MXRGNK = 1 if the block iteration is not used.

d. *Maximum control-integers for material and soil properties*

MXMATK = maximum number of material types
MXSPMK = maximum number of soil parameters per material to describe soil characteristic curves
MXMPMK = maximum number of material properties per material
MXRMPK = maximum number of coefficients of the viscosity and density functions.

The maximum control-integers for flow simulations and their definitions are given as the following:

a. *Maximum control-integers for source/sinks, flow*

MXSELh = maximum number of source elements
MXSPRh = maximum number of source profiles
MXSDPh = maximum number of data points on each element source/sink profile
MXWNPh = maximum number of well nodal points
MXWPRh = maximum number of well source/sink profiles
MXWDPh = maximum number of data points on each well source/sink profile

b. *Maximum control-integers for cauchy boundary conditions, flow*

MXCNPh = maximum number of Cauchy nodal points
MXCESh = maximum number of Cauchy element surfaces
MXCPRh = maximum number of Cauchy-flux profiles
MXCDPh = maximum number of data points on each Cauchy-flux profile

c. *Maximum control-integers for Neumann boundary conditions, flow*

MXNNPh = maximum number of Neumann nodal points
MXNESh = maximum number of Neumann element surfaces
MXNPRh = maximum number of Neumann-flux profiles
MXNDPh = maximum number of data points on each Neumann-flux profile

d. *Maximum control-integers for rainfall-seepage boundary conditions, flow*

MXVNPh = maximum number of variable nodal points
MXVESh = maximum number of variable element surfaces
MXVPRh = maximum number of rainfall profiles
MXVDPh = maximum number of data point on each rainfall profile

e. *Maximum control-integers for Dirichlet boundary conditions, flow*

MXDNPh = maximum number of Dirichlet nodal points
MXDPRh = maximum number of Dirichlet total head profiles
MXDDPh = maximum number of data points on each Dirichlet profile

The maximum control-integers for transport simulations and their definitions are given as the following:

a. *Maximum control-integers for source/sinks, transport*

MXSELc = maximum number of source elements
MXSPRc = maximum number of source profiles
MXSDPc = maximum number of data points on each element source/sink profile
MXWNPc = maximum number of well nodal points
MXWPRc = maximum number of well source/sink profiles
MXWDPc = maximum number of data points on each well source/sink profile

b. *Maximum control-integers for Cauchy boundary conditions, transport*

MXCNPc = maximum number of Cauchy nodal points
MXCESc = maximum number of Cauchy element surfaces
MXCPRc = maximum number of Cauchy-flux profiles
MXCDPc = maximum number of data points on each Cauchy-flux profile

c. *Maximum control-integers for Neumann boundary conditions, transport*

MXNNPc = maximum number of Neumann nodal points
MXNESc = maximum number of Neumann element surfaces
MXNPRc = maximum number of Neumann-flux profiles
MXNDPc = maximum number of data points on each Neumann-flux profile

d. *Maximum control-integers for flowin-flowout boundary conditions, transport*

MXVNPc = maximum number of variable nodal points
MXVESc = maximum number of variable element surfaces
MXVPRc = maximum number of rainfall profiles
MXVDPc = maximum number of data point on each rainfall profile

e. *Maximum control-integers for Dirichlet boundary conditions, transport*

MXDNPc = maximum number of Dirichlet nodal points
MXDPRc = maximum number of Dirichlet total head profiles
MXDDPc = maximum number of data points on each Dirichlet profile

For flow simulations only, we demonstrate how to specify the above maximum control-integers with PARAMETER STATEMENT in the MAIN in the following by an example.

Let us assume that a region of interest is discretized by 30 × 20 × 10 nodes and 29 × 19 × 9 elements. In other words, we are discretizing the region with 30 nodes along the longitudinal or x-direction, 20 nodes along the lateral or y-direction, and 10 nodes along the vertical or z-direction. Since we have a total of 30 × 20 × 10 = 6,000 nodes, the maximum number of nodes is MAXNPK = 6000. The total number of elements is 29 × 19 × 9 = 4,959, i.e, MAXELK = 4959. For this simple discretization problem, the maximum connecting number to any of the 6,000 nodes in the region of interest is 27, i.e., MXJBDK = 27, and the maximum connecting number of elements to any of the 6,000 nodes is 8, i.e., MXKBDK = 8. There will be 29 × 19 = 551 element surfaces each on the bottom and top faces of the region, 29 × 9 = 261 element surfaces each on the front and back faces of the region, and 19 × 9 = 171 element surfaces each on the left and right faces of the region. Thus, there will be a total of 1,966 element surfac , i.e., MXBESK = 1966. Similarly, we can compute the surface-boundary nodes to be 1,968, i.e., MXBNPK = 1968.

In order to specify maximum control-integers related to subregion data, we have to know how the region of interest is subdivided into subregions. Let us assume we have subdivided the region of interest into 20 subregions, and each subregion has 30 × 10 nodes. It is seen, in fact, we are taking a vertical slice as a subregion. For this subregionalization, we have MXRGNK = 20. Each subregion has 30 × 10 = 300 nodes, resulting in LMXNPK = 300. It is also seen that there will be 600 interboundary nodes, 300 nodes each on the two neighboring slices of a subregion. Thus, we have LTMXNK = 900. For each subregion, the maximum bandwidth can be computed as LMXBWK = 23 if the nodes are labelled along the z-directions consecutively.

We will assume that there will be a maximum of 11 elements that have the distributed sources/sinks (i.e., MXSELh = 11) and a maximum of 10 nodal points that can be considered as well sources/sinks (i.e., MXWNPh = 10). We will also assume that there will be three different distributed source/sink profiles and five distinct point source/sink profiles. Then we will have MXSPRh = 3 and MXWPRh = 5. Let us further assume that four data points are needed to describe the distributed source/sink profiles as a function of time and that eight data points are required to described point source/sink profiles (i.e., MXSDPh = 4 and MXWDPh = 8).

To specify maximum control-integers for boundary conditions, let us assume that the top face is a variable boundary (i.e., on the air-soil interface, either ponding, infiltration, or evapotranspiration may take place). On the left face, fluxes from the adjacent aquifer are known. On the right face, the total head is assumed known. On the bottom face, natural drainage is assumed to occur (i.e., the gradient of the pressure head can be assumed zero).

There are 20 × 10 = 200 nodes on the left face and 19 × 9 = 171 element surfaces; thus MXCNPh = 200 and MXCESh = 171. It is further assumed that there two different fluxes going into the region through the left face and that each flux can be described by four data points as a function of time (i.e., MXCPFh = 2 and MXCDPh = 4). On the bottom surface, there are 30 × 20 = 600 nodes and 29 × 19 = 551 surface elements. Since the gradient of pressure head on the bottom surface is zero, there is only one Neumann flux profile; and two data points, one at zero time and the other at infinite time, are sufficient to describe the constant value of zero. Hence, we have MXNNPh = 600, MXNESh = 551, MXNPRh = 1, and MXNDPh = 2. On the top face, there will be 30 × 20 = 600 nodes and 29 × 19 = 551 surface elements. Let us assume that there are three different rainfall intensities that might fall on the air-soil interface, and that each rainfall intensity is a function of time and can be described by 24 data points. With these descriptions, we have MXVNPh = 600, MXVESh = 551, MXVPRh = 3, and MXVDPh = 24. On the right face, there are 20 × 10 = 200 nodes. Let us assume that here are twenty different values of the total head, one each on a vertical line of the right face. We further assume that each of these twenty total heads can be described by eight data points as a function of time. We then have MXDNPh = 200, MXDPRh = 20, and MXDDPh = 8.

In this example, we have six material properties (six saturated hydraulic conductivity components) per material. We will assume that the whole region of interest is made of three different kinds of materials. The characteristic curves of each material are assumed to be described by four parameters. We then have MXMATK = 3, MXMPMK = 6, and MXSPMK = 4. If we assume that we will make a 500 time-step simulation and we will reinitiate the change on the time-step size for 20 times during our simulation, then we have MXNTIK = 500 and MXDTCK = 20.

From the above discussion, the following PARAMETER STATEMENTS can be used to specify the maximum control-integers in the MAIN for the problem at hand:

```
PARAMETER(MAXNPK=6000,MAXELK=4959,MXBNPK=1968,
  MXBESK=1966)
PARAMETER(MXJBDK=27,MXKBDK=8,MXNTIK=500,MXDTCK=
  20)
PARAMETER(LTMXNK=900,LMXNPK=300,LMXBWK=23,
  MXRGNK=20)
PARAMETER(MXMATK=4,MXSPMK=4,MXMPMK=6,
  MXRMPK=8)
PARAMETER(MXSELh=11.MXSPRh=3,MXSDPh=4,MXWNPh=10,
  MXWPRh=5,MXWDPh=8)
PARAMETER(MXCNPh=200,MXCESh=171,MXCPRh=2,
  MXCDPh=4)
PARAMETER(MXNNPh=600,MXNESh=551,MXNPRh=1,
  MXNDPh=2)
PARAMETER(MXVNPh=600,MXVESh=551,MXVPRh=3,
  MXVDPh=24)
PARAMETER(MXDNPh=200,MXDPRh=20,MXDDPh=8)
PARAMETER(MXSELc=1,MXSPRc=1,MXSDPc=2,MXWNPc=1,
  MXWPRc=1,MXWDPc=2)
PARAMETER(MXCNPc=1,MXCESc=1,MXCPRc=1,MXCDPc=2)
PARAMETER(MXNNPc=1,MXNESc=1,MXNPRc=1,MXNDPc=2)
PARAMETER(MXVNPc=1,MXVESc=1,MXVPRc=1,MXVDPc=2)
PARAMETER(MXDNPc=1,MXDPRc=1,MXDDPc=2)
```

In the following, for transport simulations only, we demonstrate how to specify the maximum control-integers with PARAMETER statements in the MAIN with an example.

Let us assume that a region of interest is discretized by 30 × 20 × 10 nodes and 29 × 19 × 9 elements. In other words, we are discretizing the region with 30 nodes along the longitudinal or x-direction, 20 nodes along the lateral or y-direction, and 10 nodes along the vertical or z-direction. Since we have a total of 30 × 20 × 10 = 6,000 nodes, the maximum number of nodes is MAXNPK = 6000. The total number of elements is 29 × 19 × 9 = 4,959, i.e, MAXELK = 4959. For this simple discretization problem, the maximum number of connecting nodes to any of the 6,000 nodes in the region

of interest is 27, i.e., MXJBDK = 27. The maximum number of connecting elements to any nodes is 8, i.e., MXKBDK = 8. There will be 29 × 19 = 551 element surfaces each on the bottom and top faces of the region, 29 × 9 = 261 element-surfaces each on the front and back faces of the region, and 19 × 9 = 171 element-surfaces each on the left and right faces of the region. Thus, there will be a total of 1966 element-surfaces, i.e., MXBESK = 1966. Similarly, we can compute the surface-boundary nodes to be 1968, i.e., MXBNPK = 1968.

In order to specify maximum control-integers related to subregion data, we have to know how the region of interest is subdivided into subregions. Let us assume we have subdivided the region of interest into 20 subregions; each subregion has 30 × 10 nodes. It is seen, in fact, we are taking a vertical slice as a subregion. For this subregionalization, we have MXRGNK = 20. Each subregion has 30 × 10 = 300 nodes, resulting in LMXNPK = 300. It is also seen that there will be 600 interboundary nodes, 300 nodes each on the two neighboring slices of a subregion. Thus, we have LTMXNK = 900. For each subregion, the maximum bandwidth can be computed as LMXBWK = 23 if the nodes are labelled along the z-directions consecutively.

We will assume that there will be a maximum of 11 elements that have the distributed sources/sinks (i.e., MXSELc = 11) and a maximum of 10 nodal points that can be considered as well sources/sinks (i.e. MXWNPc = 10). We will also assume that there will be three different distributed source-sink profiles and five distinct point source/sink profiles. Then we will have MXSPRc = 3 and MXWPRc = 5. Let us further assume that four data points are needed to describe the distributed source/sink profiles as a function of time and that eight data points are required to described point source/sink profiles (i.e., MXSDPc = 4 and MXWDPc = 8).

To specify maximum control-integers for boundary conditions, let us assume that the top face is a variable boundary (i.e., on the air-soil interface, either Cauchy condition with known incoming concentration or Neumann condition with zero gradient of concentration). On the left face, fluxes from the adjacent aquifer are known. On the right face, the concentration is assumed known. On the bottom face, Neumann condition is assumed to occur with zero concentration gradient.

There are 20 × 10 = 200 nodes on the left face and 19 × 9 = 171 element surfaces; thus MXCNPc = 200 and MXCESc = 171. It is further assumed that there two different fluxes going into the region through the left face and that each flux can be described by four data points as a function of time (i.e., MXCPRc = 2 and MXCDPc = 4). On the bottom surface, there are 30 × 20 = 600 nodes and 29 × 19 = 551 surface elements. Since the gradient of concentration on the bottom surface is zero, there is only one Neumann flux profile; and two data points, one at zero time and the other at infinite time, are sufficient to describe the constant value of zero. Hence, we have MXNNPc = 600, MXNESc = 551, MXNPRc = 1, and MXNDPc = 2. On the top face, there will be 30 × 20 = 600 nodes and 29 × 19 = 551

surface elements. Let us assume that there are three different concentration profiles that might be imposed on the air-soil interface, and that each concentration profile is a function of time and can be described by 24 data points. With these descriptions, we have MXVNPc = 600, MXVESc = 551, MXVPRc = 3, and MXVDPc = 24. On the right face, there are 20 × 10 = 200 nodes. Let us assume that here are twenty different values of concentration, one each on a vertical line of the right face. We further assume that each of these twenty concentrations can be described by eight data points as a function of time. We then have MXDNPc = 200, MXDPRc = 20, and MXDDPc = 8.

In this example, we have eight material properties per material. We will assume that the whole region of interest is composed of three different kinds of materials. We then have MXMATK = 3 and MXMPMK = 6. If we assume that we will make a 500 time-step simulation and we will reinitiate the change on the time-step size for 20 times during our simulation, then we have MXNTIK = 500 and MXDTCK = 20.

From the above discussion, the following PARAMETER Statements can be used to specify the maximum control-integers in the MAIN for the problem at hand:

```
PARAMETER(MAXNPK=6000,MAXELK=4959,MXBNPK=1968,
  MXBESK=1966)
PARAMETER(MXJBDK=27,MXKBDK=8,MXNTIK=500,
  MXDTCK=20)
PARAMETER(LTMXNK=900,LMXNPK=300,LMXBWK=23,
  MXRGNK=20)
PARAMETER(MXMATK=4,MXSPMK=6,MXMPMK=6,
  MXRMPK=8)

PARAMETER(MXSELh=1,MXSPRh=1,MXSDPh=2,MXWNPh=1,
  MXWPRh=1,MXWDPh=2)
PARAMETER(MXCNPh=1,MXCESh=1,MXCPRh=1,MXCDPh=2)
PARAMETER(MXNNPh=1,MXNESh=1,MXNPRh=1,MXNDPh=2)
PARAMETER(MXVNPh=1,MXVESh=1,MXVPRh=1,MXVDPh=2)
PARAMETER(MXDNPh=1,MXDPRh=1,MXDDPh=2)

PARAMETER(MXSELc=11,MXSPRc=3,MXSDPc=4,MXWNPc=10,
  MXWPRc=5, MXWDPc=8)
PARAMETER(MXCNPc=200,MXCESc=171,MXCPRc=2,MXCDPc=4)
PARAMETER(MXNNPc=600,MXNESc=551,MXNPRc=1,
  MXNDPc=2)
PARAMETER(MXVNPc=600,MXVESc=551,MXVPRc=3,
  MXVDPc=24)
PARAMETER(MXDNPc=200,MXDPRc=20,MXDDPc=8)
```

Soil Property Function Specifications

Analytical functions are used to describe the relationships of water content, water capacity, and relative hydraulic conductivity with pressure head. Therefore, the user must supply three functions to compute the water content, water capacity, and relative hydraulic conductivity based on the current value of pressure head. The parameters needed to specify the functional form are read and stored in SPP. One example is shown in the subroutine SPROP in the source code. In this example, the water content, water capacity, and relative hydraulic conductivity are given by van Genechten (1980):

$$\theta = \theta_r + \frac{\theta_s - \theta_r}{[1 + (\alpha h)^n]^m} \tag{3.1}$$

$$\frac{d\theta}{dh} = \alpha(n - 1)[1 - f(\theta)]^m[f(\theta)](\theta_s - \theta_r) \tag{3.2}$$

$$K_r = [(\theta - \theta_r)/(\theta_s - \theta_r)]^{\frac{1}{2}}\{1 - [1 - f(\theta)]^m\}^2 \tag{3.3}$$

in which

$$f(\theta) = (\theta - \theta_r)/(\theta_s - \theta_r)^{1/m} \tag{3.4}$$

and

$$m = 1 - \frac{1}{n} \tag{3.5}$$

To further demonstrate how we should modify the subroutine SPROP in Appendix A to accommodate the material property functions that are different from those given by Equations 3.4 and 3.5, let us assume that the following Fermi types of functions are used to represent the unsaturated hydraulic properties (Yeh 1987):

$$\theta = \theta_r + (\theta_s - \theta_r)/\{1 + \exp[-\alpha(h - h_\theta)]\} \tag{3.6}$$

$$8d\theta/dh = \alpha(\theta_s - \theta_r) \exp[-\alpha(h - h_\theta)]/\{1 + \exp[-\alpha(h - h_\theta)]\}^2 , \quad (3.7)$$

and

$$\log_{10}(K_r) = \epsilon/\{1 + \exp[-\beta(h - h_k)]\} - \epsilon , \quad (3.8)$$

where θ_s, θ_r, α, and h_θ are the parameters for computing the water content and water capacity; and β, ϵ, and h_k are the parameters for computing the relative hydraulic conductivity. Lines between SPRO 845 and SPRO 1110 in the source code must be changed, for this example, to the following form for computing the moisture content and water capacity

```
      WCR=SPP(1,MTYP,1)
      WCS=SPP(2,MTYP,1)
      ALPHA=SPP(3,MTYP,1)
      HTHETA=SPP(4,MTYP,1)
      EPS=SPP(1,MTYP,2)
      BETA=SPP(2,MTYP,2)
      HSUBK=SPP(3,MTYP,2)
      DO 390 KG=1,8
      NP=IE(M,KG)
      HNP=HKG(KG)
      HNP=-HNP
C
C ------ SATURATED CONDITION
C
      IF(HNP.LE.0.0) THEN
        TH(KG,M)=WCS
        DTH(KG,M)=0.0D0
        USKFCT=1.0D0
C
      ELSE
C
C ------ UNSATURATED CASE
C
      EXPAH=DEXP(-ALPHA*(HNP-HTHETA))
      TH(KG,M)=WCR+(WCS-WCR)/(1.0D0+EXPAH)
      DTH(KG,M)=ALPHA*(WCS-WCR)*EXPAH/(1.0D0+EXPAH)**2
      AKRLOG=EPS/(1.0D0+DEXP(-BETA*(HNP-HSUBK))) - EPS
      USKFCT=10.0D0**AKRLOG
     ENDIF
```

Input and Output Devices

Five logical units are needed to execute 3DSALT. Units 15 and 16 are standard card input and line printer devices, respectively. Unit 11 must be specified to store the flow simulation results, which can be used for plotting purposes. Unit 12 must be specified to store the transport simulation results, which can be used for plotting purposes. Unit 13 is used to store the boundary arrays for later uses, if these arrays are computed for the present job. Unit 14 is used to store pointer arrays for later uses, if these arrays are generated for the present job. For large problems, our experience has indicated that it would take too much time to process the boundary arrays and to generate pointer arrays. Hence, it is advisable that for multijob executions, these boundary and pointer arrays should be computed only once and written on units 13 and 14, respectively. Once they are stored on units 13 and 14, the input data file should be properly identified for the new job so they can be read via units 13 and 14, respectively. Finally, unit 20 is used to print any variable for debugging purposes.

4 Sample Problems

To verify 3DSALT, three illustrative examples are used. The first one is a one-dimensional flow problem through a column. The second one is a one-dimensional transport problem through a column. The third one is a three-dimensional salt intrusion problem.

Problem No. 1: One-Dimensional Column Flow Problem

This example is selected to represent the simulation of a one-dimensional flow problem with 3DSALT. The column is 200 cm long and 50 by 50 cm in cross section (Figure 4.1). The column is assumed to contain soil with a saturated hydraulic conductivity of 10 cm/day, a porosity of 0.45, and a field capacity of 0.1. The unsaturated characteristic hydraulic properties of the soil in the column are given as

$$\theta = \theta_s - \frac{\theta_s - \theta_r}{h_b - h_a}(h - h_a) \tag{4.1}$$

and

$$K_r = \frac{\theta - \theta_r}{\theta_s - \theta_r} \tag{4.2}$$

where h_b and h_a are the parameters used to compute the water content and the relative hydraulic conductivity, respectively.

The initial conditions assumed are a pressure head of -90 cm imposed on the top surface of the column, 0 cm on the bottom surface of the column, and -97 cm elsewhere. The following boundary conditions are given: no flux is imposed on the left, front, right, and back surfaces of the column; pressure head is held at 0 cm on the bottom surface; and variable condition is used on the top surface of the column with a ponding depth of zero, minimum

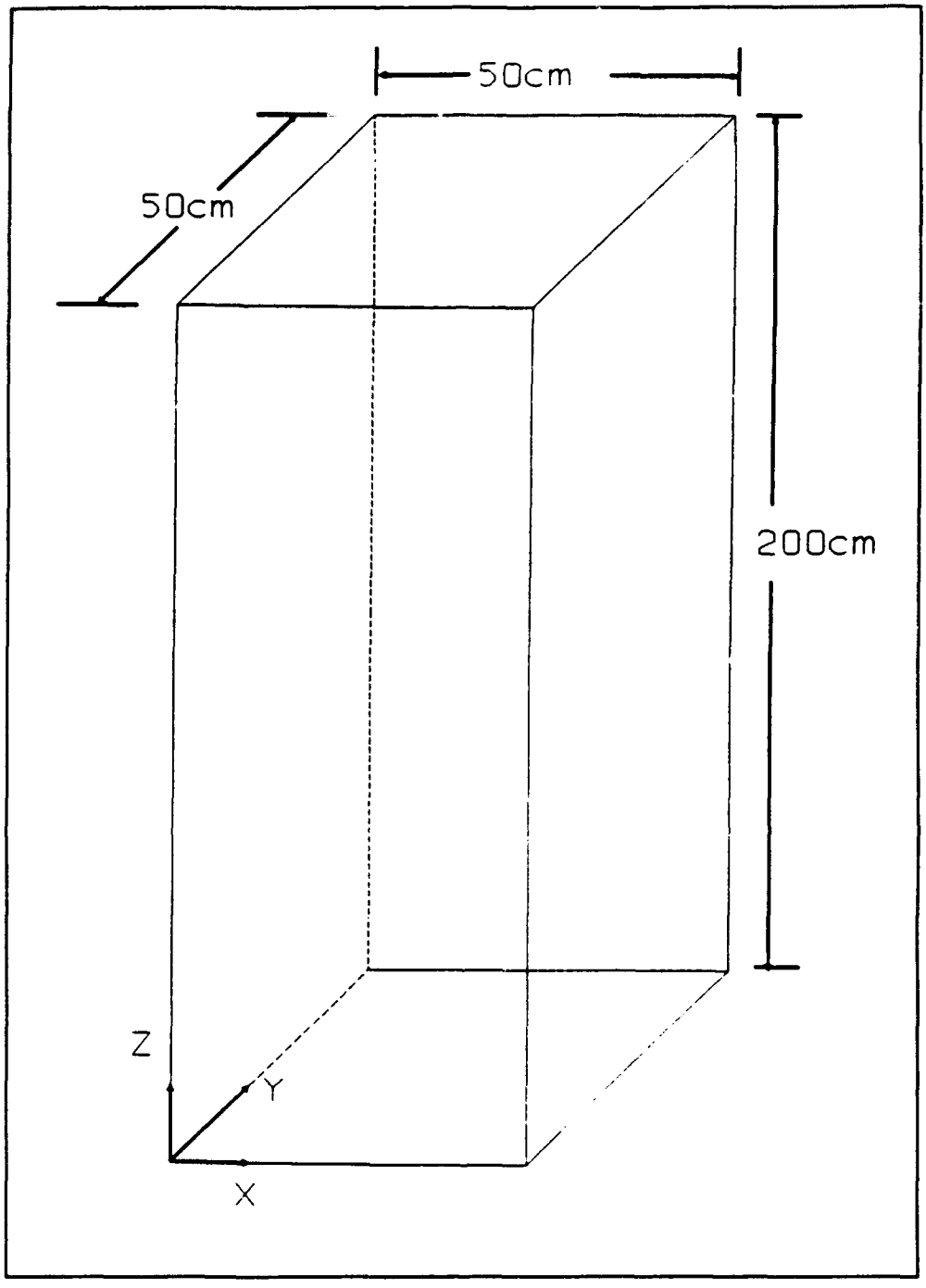


Figure 4.1. Problem definition for the one-dimensional transient flow in a soil column

pressure of -90 cm, and a rainfall of 5 cm/day for the first 10 days and a potential evaporation of 5 cm/day for the second 10 days.

The region of interest, that is, the whole column, will be discretized with $1 \times 1 \times 40 = 40$ elements with element size $= 50 \times 50 \times 5$ cm, resulting in $2 \times 2 \times 41 = 164$ node points. The incomplete Cholesky preconditioned

conjugate gradient method is used to solve the resulting matrix equation. Thus, the subregion data are not needed.

A variable time-step size is used. The initial time-step size is 0.05 day, and each subsequent time-step size is increased by 0.2 times with a maximum time-step size not greater than 1.0 d. Because there is an abrupt change in the flux value from 5 cm/day (infiltration) to -5 cm/day (evaporation) imposed on the top surface at day 10, the time-step size is automatically reset to 0.05 d on the tenth day. A 20-day simulation will be made with 3DSALT. With the time-step size described above, 44 time-steps are needed.

The pressure head tolerance is $2 \cdot 10^{-2}$ cm for nonlinear iteration and $1 \cdot 10^{-2}$ cm for block iteration. The relaxation factor for both the nonlinear iteration and block iteration is 0.5.

To execute the problem, the maximum control-integers in the main program should be specified as follows:

```
PARAMETER(MAXNPK=164,MAXELK=40,MXBNPK=164,
  MXBESK=162)
PARAMETER(MXJBDK=12,MXKBDK=8,MXNTIK=44,MXDTCK=3)
PARAMETER(LTMXNK=1,LMXNPK=1,LMXBWK=1,MXRGNK=1)
PARAMETER(MXMATK=1,MXSPMK=4,MXMPMK=6,
  MXRMPK=8)

PARAMETER(MXSELh=1,MXSPRh=3,MXSDPh=1,MXWNPh=1,
  MXWPRh=1,MXWDPh=1)
PARAMETER(MXCNPh=1,MXCESh=1,MXCPRh=1,MXCDPh=1)
PARAMETER(MXNNPh=1,MXNESh=1,MXNPRh=1,MXNDPh=1)
PARAMETER(MXVNPh=4,MXVESh=1,MXVPRh=1,MXVDPh=4)
PARAMETER(MXDNPh=4,MXDPRh=1,MXDDPh=2)

PARAMETER(MXSELc=1,MXSPRc=1,MXSDPc=1,MXWNPc=1,
  MXWPRc=1,MXWDPc=1)
PARAMETER(MXCNPc=1,MXCESc=1,MXCPRc=1,MXCDPc=1)
PARAMETER(MXNNPc=1,MXNESc=1,MXNPRc=1,MXNDPc=1)
PARAMETER(MXVNPc=1,MXVESc=1,MXVPRc=1,MXVDPc=1)
PARAMETER(MXDNPc=1,MXDPRc=1,MXDDPc=1)
```

To reflect the soil property function given by Equations 4.1 and 4.2, we have to modify the subroutine SPROP in the source code. Lines between SPRO 720 and SPRO 845 in the source code must be changed, for this example, to the following form for computing the moisture content and water capacity:

```
VCR=THPROP(1,MTYP)
WCS=THPROP(2,MTYP)
HAA=THPROP(3,MTYP)
HAB=THPROP(4,MTYP)
```

```
      DO 390 KG=1,8
       HNP=HKG(KG)
       HNP=-HNP
       IF(HNP.LE.0) GO TO THEN
C
C ------- SATURATED CONDITION
C
       TH(KG,M)=WCS
       DTH(KG,M)=0.0D0
       USKFCT=1.0D0
C
       ELSE
C
C ------- UNSATURATED CASE
C
       THMKG=WCS-(WCS-WCR)*(-HNP-HAA)/(HAB-HAA)
       TH(KG,M)=THMKG
       DTH(KG,M)=-(WCS-WCR)/(HAB-HAA)
       USKFCT=(THMKG-WCR)/(WCS-WCR)
      END IF
```

Input for Problem No. 1

With the above descriptions, the input data can be prepared according to the instructions given in Appendix A. The input data are given in Table 4.1.

Problem No. 2: One-Dimensional Column Transport

A simple problem is presented here to illustrate the application of this document. This is a one-dimensional transport problem between x = 0 and x = 200.0 (Figure 4.2). Initially, the concentration is zero throughout the region of interest. The concentration at x = 0.0 is maintained at $C = C_o$ = 1.0 (Fig. 4.2). The natural condition of zero gradient flux is imposed at x = 200.0 (Fig. 4.2). A bulk density of 1.2, a dispersivity of 5.0, and an effective porosity of 0.4 (not used in the program) are assumed. A specific discharge (Darcy velocity) of 2.0 is assumed and a moisture content of 0.4 is used. For numerical simulation the region is divided into 40 elements of equal size with 5.0. A time-step size of 0.5 is used and a 40 time-step simulation is made. No adsorption is allowed. For this discretization, the mesh Peclet number is $P_e = 1$ and the Courant number is $C_r = 0.5$.

Table 4.1 Input Data Set for Example 1

```
1 ONE-D COLUMN INFILTRATION-EVAPORATION; L=CM, T=DAY, M=G, 3dfemft.ex1
10011
= = = = = = = = = data set 2: option parameters
 50  0.5d0
 1  1  0  0  3  3
 1  1.0  0.5d0  0.5d0
 2  0  1  1  1
 1.0d0 0.5d0 1.0d0 1.0d0
  1  1.0d0  1.0d0  1.0d0
= = = = = = = = = data set 3: iteration parameters
 50 20 100 2.0d-2 2.0d-2
  1 100 1.0d-3  1.0d-4
= = = = = = = = = data set 4: time control parameters
 44  3
 0.05d0    0.20d0    1.0d0    22.0d0
333030300030003003000033303030003000300300003
111010100010001001000011101010001000100100001
   1.0D01  2.0000D1    1.0D38
= = = = = = = = = DATA SET 5:  MATERIAL PROPERTIES
1 8 8
    0.0D0     0.0D0    10.0D0     0.0D0     0.0D0     0.0D0    0.0d0  0.0d0
    1.0d0     0.0d0     0.0d0     0.0d0     1.0d0     0.0d0    0.0d0  0.0d0
    0.0d0     1.2d0     5.0d0     0.0d0     0.0d0     1.0d0    0.0d0  0.0d0
= = = = = = = = = DATA SET 6:  soil properties
   0   4   0    1.0d0    1.0d0  1.0d0
  0.150D0  0.450D0   0.00D0   -1.0D2                    THPROP
  0.000D0  0.000D0   0.00D0    0.0D0                    AKPROP
C * * * * * * * DATA SET 7:  NODE COORDINATES
 164
   1  40   1        0.0D0   50.0D0    0.0D0    0.0D0    0.0D0    5.0D0
  42  40   1        0.0D0    0.0D0    0.0D0    0.0D0    0.0D0    5.0D0
  83  40   1       50.0D0    0.0D0    0.0D0    0.0D0    0.0D0    5.0D0
 124  40   1       50.0D0   50.0D0    0.0D0    0.0D0    0.0D0    5.0D0
   0   0   0         0.0     0.0      0.0      0.0      0.0      0.0
C * * * * * * * DATA SET 9:  ELEMENT INCIDENCES
  40
   1  39   1  42  83 124   1  43  84 125   2   1
   0   0   0   0   0   0   0   0   0   0   0   0        END OF IE
C * * * * * * * data set10: material correction
   0
C * * * * * * * DATA SET 11:  INITIAL CONDITIONS
   1   3  41        0.0D0    0.0D0    0.0D0
   2  38   1       -9.70D1    0.0D0    0.0D0
  43  38   1       -9.70D1    0.0D0    0.0D0
  84  38   1       -9.70D1    0.0D0    0.0D0
 125  38   1       -9.70D1    0.0D0    0.0D0
  41   3  41       -9.00D1    0.0D0    0.0D0
   0   0   0        0.0       0.0      0.0          END OF IC, flow
= = = = = = = = = data set 12: element(distributed) source/sink, flow
   0 0 0 0
= = = = = = = = = data set 13: point(well) source/sink, flow
   0 0 0 0
```

(Continued)

Table 4.1 (Concluded)

```
========== data set 16: rainfall/evaporation-seepage boundary conditions
  1  4  1  4  0
  0.0D0    5.0D0    10.0D0    5.0D0  10.001D0   -5.0D0    1.0D38   -5.0D0
  1   0   0   1   0
  0   0   0   0   0       END OF IRTYP
  1   0   0  82  123  164  41   0   0   0   0
  0   0   0   0   0   0   0   0   0   0   0 END OF ISV(J,I) J=1,4
  1   3   1  41  41
  0   0   0   0   0                  END OF NPVB
  1   3   1       0.0D0    0.0D0     0.0
  0   0   0       0.0      0.0       0.0          END OF HCON
  1   3   1     -90.0D0    0.0D0     0.0
  0   0   0       0.0      0.0       0.0          END OF HMIN
C ******* DATA SET 17:  DIRICHLET BOUNDARY CONDITIONS, flow
  1  1  2  0
  0.0D0    0.0D0    1.0D38    0.0D0
  1  42  83  124
  1   3   1   1   0
  0   0   0   0   0                  END OF IDTYP
========== data set 18: cauchy boundary conditions, flow
  0 0 0 0 0
========== data set 19: neumann boundary conditions, flow
  0 0 0 0 0
  0            END OF JOB ----------------------------------------0000
```

To execute the problem, the maximum control-integers in the MAIN must be specified as follows:

```
PARAMETER(MAXNPK=164,MAXELK=40,MXBNPK=164,
  MXBESK=162)
PARAMETER(MXJBDK=12,MXKBDK=2,MXNTIK=40,MXDTCK=1)
PARAMETER(LTMXNK=1,LMXNPK=1,LMXBWK=1,MXRGNK=1)
PARAMETER(MXMATK=3,MXSPMK=6,MXMPMK=8,
  MXRMPK=8)

PARAMETER(MXSELh=1,MXSPRh=1,MXSDPh=1,MXWNPh=1,
  MXWPRh=1,MXWDPh=1)
PARAMETER(MXCNPh=1,MXCESh=1,MXCPRh=1,MXCDPh=1)
PARAMETER(MXNNPh=1,MXNESh=1,MXNPRh=1,MXNDPh=1)
PARAMETER(MXVNPh=1,MXVESh=1,MXVPRh=1,MXVDPh=1)
PARAMETER(MXDNPh=1,MXDPRh=1,MXDDPh=1)

PARAMETER(MXSELc=1,MXSPRc=1,MXSDPc=1,MXWNPc=1,
  MXWPRc=1,MXWDPc=1)
PARAMETER(MXCNPc=1,MXCESc=1,MXCPRc=1,MXCDPc=1)
PARAMETER(MXNNPc=1,MXNESc=1,MXNPRc=1,MXNDPc=1)
PARAMETER(MXVNPc=4,MXVESc=1,MXVPRc=1,MXVDPc=2)
PARAMETER(MXDNPc=4,MXDPRc=1,MXDDPc=2)
```

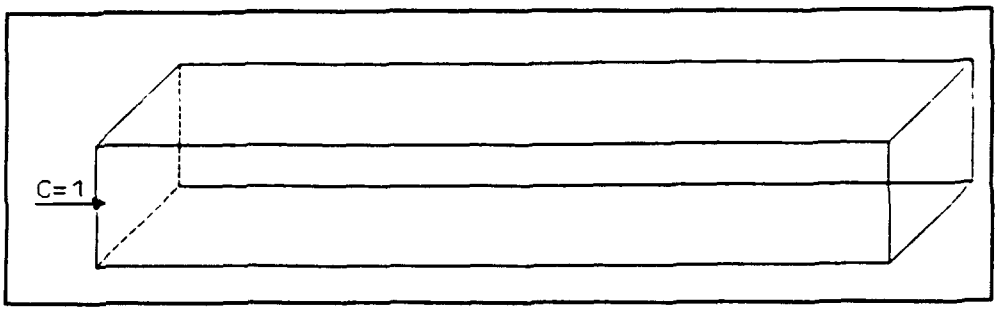


Figure 4.2. Problem definition for the one-dimensional transient transport problem in a soil column

Input for Problem No. 2

Table 4.2 shows the input data set for the sample problem described in the above section.

Problem No. 3: Three-Dimensional Salt Intrusion Problem

This example is selected to represent the simulation with 3DSALT of a three-dimensional salt intrusion problem due to pumping near a coastal area. The problem involves a constant pumping well located near the center of a region (Figure 4.3). The region consists of 80 m in x-direction, 100 m in y-direction, and 11 m in z-direction on the plane y = 50 m and sloping to 10 m at y = 0 and y = 100 m. This region is bounded on the front and back by tidal water bodies. The bottom of the region is bounded by an impervious aquifuges. On the right-hand side and the reversed L-shaped left-hand side the region is bounded by groundwater dividers (Figure 4.3). The pumping well is located at (x,y) = (60,50) (Figure 4.3). The pumping rate of the well is kept constant at 0.001 m^3/hr. Initially, the water in the subsurface is hydrostatic. The tidal fluctuation on the bounding water bodies (front and back faces) is 1.0 m with a mean tidal elevation of z = 10 m. The medium in the region is assumed to be anisotropic and have saturated hydraulic conductivity components K_{xx} = 0.03 m/hr, K_{yy} = 0.03 m/hr, and K_{zz} = 0.01 m/hr. The porosity of the medium is 0.25 and the field capacity is 0.0125. The unsaturated characteristic hydraulic properties of the medium are given as

$$\theta = \theta_r + \frac{\theta_s - \theta_r}{1 + (\alpha|h_a - h|)^{\beta}} \qquad (4.3)$$

Table 4.2
Input Data Set for Example 2

```
  1 ONE-D COLUMN INFILTRATION-EVAPORATION; L=CM, T=DAY, M=G, 3dfemft.ex1
1011
= = = = = = = = = data set 2: option parameters
 50  0.5d0
 1  1  1  0  3  3
 1  1.0  0.5d0  0.5d0
-1  0  1  1  1
 1.0d0 0.5d0 1.0d0 1.0d0
  1  1.0d0  1.0d0  1.0d0
= = = = = = = = = data set 3: iteration parameters
 50 20 100 2.0d-2 2.0d-2
  1 100 1.0d-3  1.0d-4
= = = = = = = = = data set 4: time control parameters
 44   3
 0.5d0    0.0d0    1.0d0    22.0d0
333030300030003003000033303030003000300300003
111010100010001001000011101010001000100100001
   1.0D01  2.0000D1    1.0D38
= = = = = = = = = DATA SET 5:  MATERIAL PROPERTIES
1 8 8
    0.0D0    0.0D0    10.0D0    0.0D0    0.0D0    0.0D0    0.0d0  0.0d0
    1.0d0    0.0d0    0.0d0    0.0d0    1.0d0    0.0d0    0.0d0  0.0d0
    0.0d0    1.2d0    5.0d0    0.0d0    0.0d0    1.0d0    0.0d0  0.0d0
= = = = = = = = = DATA SET 6:  soil properties
   0   4   0   1.0d0   7.316d12   1.1232d2
  0.150D0  0.450D0   0.00D0   -1.0D2                    THPROP
  0.000D0  0.000D0   0.00D0    0.0D0                    AKPROP
C ******* DATA SET 7:  NODE COORDINATES
 164
   1  40   1        0.0D0   50.0D0    0.0D0    0.0D0    0.0D0    5.0D0
  42  40   1        0.0D0    0.0D0    0.0D0    0.0D0    0.0D0    5.0D0
  83  40   1       50.0D0    0.0D0    0.0D0    0.0D0    0.0D0    5.0D0
 124  40   1       50.0D0   50.0D0    0.0D0    0.0D0    0.0D0    5.0D0
   0   0   0         0.0     0.0      0.0      0.0      0.0      0.0
C ******* DATA SET 9:  ELEMENT INCIDENCES
  40
   1  39   1  42  83 124   1  43  84 125   2   1
   0   0   0   0   0   0   0   0   0   0   0   0        END OF IE
C ******* data set10: material correction
   0
C ******* DATA SET 11:  INITIAL CONDITIONS
   1   3  41        1.0d0     0.0d0     0.0
   2  38   1        0.0d0     0.0d0     0.0d0
  43  38   1        0.0d0     0.0d0     0.0d0
  84  38   1        0.0d0     0.0d0     0.0d0
 125  38   1        0.0d0     0.0d0     0.0d0
  41   3  41        0.0d0     0.0d0     0.0d0
   0   0   0       0.0d0     0.0d0     0.0d0        end of ic, transport
= = = = = = = = = data set 14: element(distributed) source/sink, transport
   0 0 0 0
= = = = = = = = = data set 15: point(well) source/sink, transport
   0 0 0 0
= = = = = = = = = data set 20: run-in/seep-out boundary
   1 4 1 2 0
  0.0d0  0.0d0  1.0d38  0.0d0
```

(Continued)

Table 4.2 (Concluded)

```
  1  0  0  1  0
  0  0  0  0  0  end of irtyp
  1  0  0 82 123 164  41  0  0  0  0
  0  0  0  0  0  0  0  0  0  0  0  0  end fof isvt(j,i),j = 1,4
  1  3  1 41  41
  0  0  0  0  0       end of npvbt
= = = = = = = = = data set 21: dirichlet boundary conditions, transport
  4  1  2  0
 0.0d0   1.0d0   1.0d38   1.0d0
  1 42 83 124
  1  3  1  1  0
  0  0  0  0  0   end of idtyp
= = = = = = = = = data set 22: Cauchy boundary condition, transport
  0  0  0  0  0
= = = = = = = = = data set 23: Neumann boundary condition, transport
  0  0  0  0  0
C ******* DATA SET 24:  HYDROLOGICAL BOUNDARY CONDITIONS
**********************
 1 163 1  0.0D0 0.0D0 2.0D0 0.0D0 0.0D0 0.0D0
 0  0 0   0.0   0.0   0.0   0.0   0.0   0.0  END OF VELOCITY
  1  39   1 0.4D0 0.0
  0   0   0   0.0     0.0            END OF TH
  0             END OF JOB ----------------------------------------0000
```

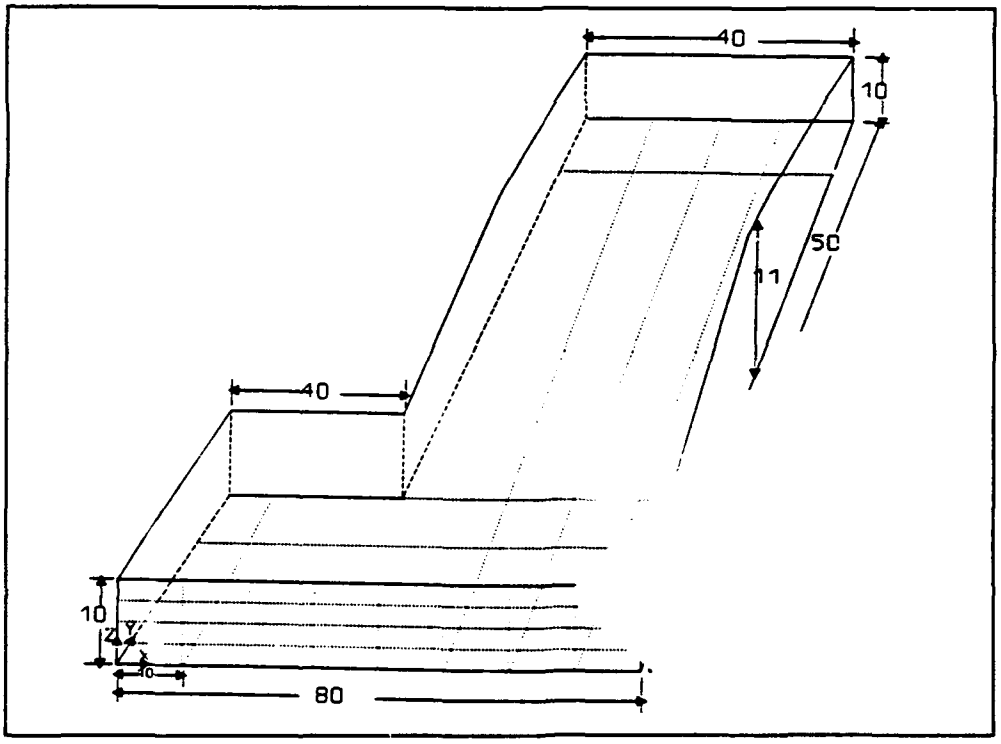


Figure 4.3. Problem definition for the three-dimensional transient salt intrusion problem

and

$$K_r = \left[\frac{\theta - \theta_r}{\theta_s - \theta_r} \right]^2 \qquad (4.4)$$

where h_a, α, and β are the parameters used to compute the water content and the relative hydraulic conductivity.

The following boundary conditions are given: pressure head is assumed hydrostatic and the concentration is assumed 1.0 on two vertical planes located at y = 0 and 100, respectively; no flux is imposed on all other boundaries of the region for both flow and transport. The pumping well is screened at nodes 444, 115, and 186. A total of 960 time-steps with the time-step size = 0.5 hr will be made. Initial conditions for flow are hydrostatic and zero concentration for transport.

For numerical simulations, the region of interest is discretized with five planes or four layers in the vertical direction. Each plane is made of 71 nodes or 52 elements as shown in Figure 4.4. This discretization results in 355 nodes and 208 elements. The numbering system starts from the bottom plane and progresses upward to the fifth plane. In the x-direction, nodes are spaced evenly at Δx = 10 m. In the y-direction, nodes are spaced evenly at Δy = 10 m. In the z-direction, the node space ranges from 2.5 m to 2.75 m. The incomplete Cholesky preconditioned conjugate gradient method is used to solve the matrix equations; thus the subregional input is not needed.

The pressure head tolerance is 10^{-2} m for nonlinear iteration and is $5 \cdot 10^{-3}$m for block iteration. The concentration tolerance is 0.001 for the nonlinear iteration and 0.0005 for block iteration. The relaxation factors for nonlinear iteration and block iteration are set equal to 1.0 and 1.5, respectively, for flow and 1.0 and 1.0, respectively, for transport.

To execute the problem, the maximum control-integers in the MAIN should be specified according to the following:

```
      PARAMETER(MAXNPK=355,MAXELK=208,MXBNPK=250,
     &    MXBESK=248)
      PARAMETER(MXJBDK=27,MXKBDK=8,MXNTIK=960,
     &  MXDTCK=2)
      PARAMETER(LTMXNK=1,LMXNPK=1,LMXBWK=1,MXRGNK=1)
      PARAMETER(MXMATK=1,MXSPMK=5,MXMPMK=8,
     &  MXRMPK=8)
C
      PARAMETER(MXSELh=1,MXSPRh=1,MXSDPh=1,MXWNPh=3,
     &  MXWPRh=1,MXWDPh=2)
      PARAMETER(MXCNPh=1,MXCESh=1,MXCPRh=1,MXCDPh=2)
      PARAMETER(MXNNPh=1,MXNESh=1,MXNPRh=1,MXNDPh=2)
```

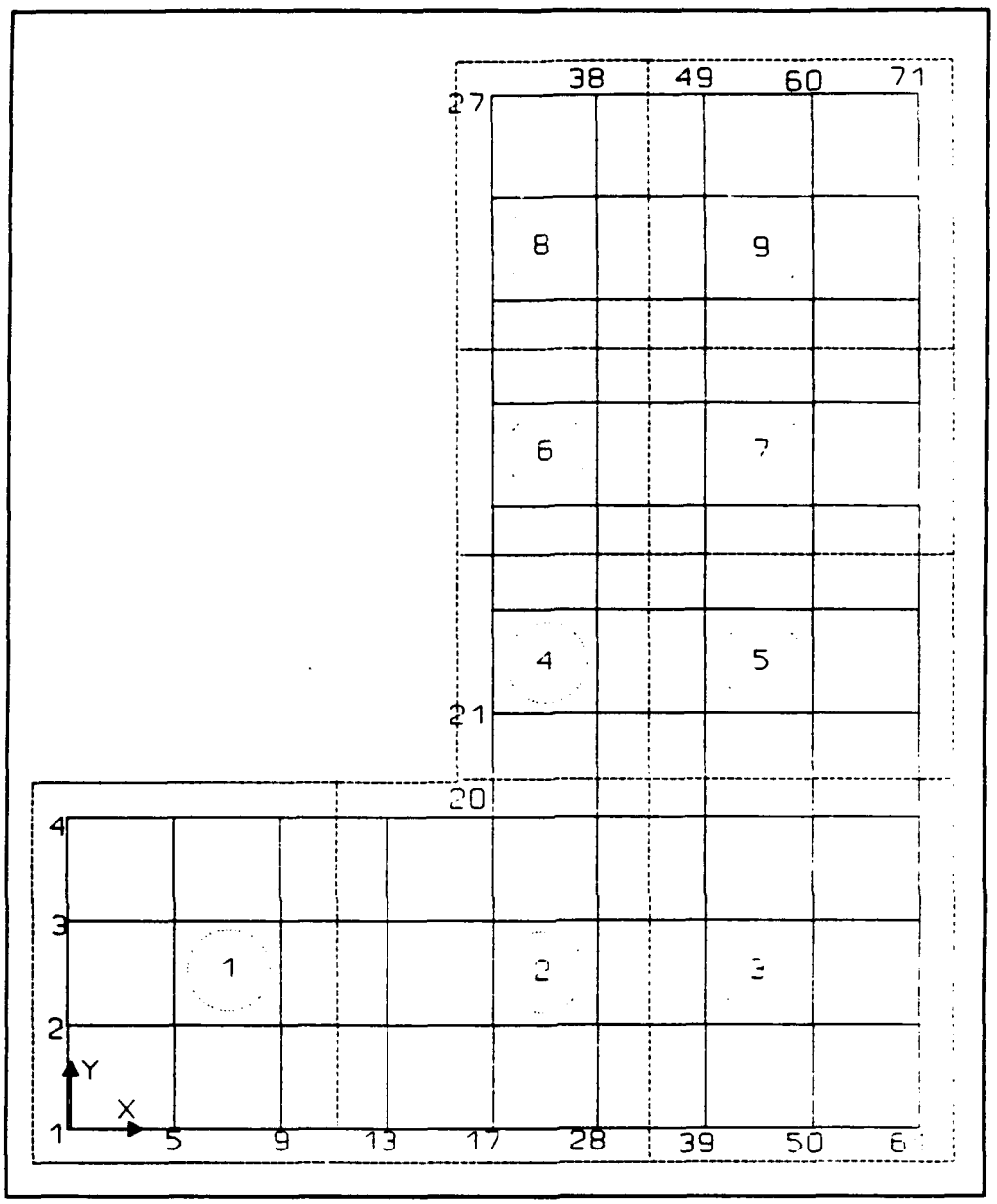


Figure 4.4. Discretization of the region with five planes (four layers) and nine subregions

```
      PARAMETER(MXVNPh=1,MXVESh=1,MXVPRh=1,MXVDPh=2)
      PARAMETER(MXDNPh=70,MXDPRh=1,MXDDPh=3)
C
      PARAMETER(MXSELc=1,MXSPRc=1,MXSDPc=1,MXWNPc=3,
     &    MXWPRc=1,MXWDPc=2)
      PARAMETER(MXCNPc=1,MXCESc=1,MXCPRc=1,MXCDPc=1)
      PARAMETER(MXNNPc=1,MXNESc=1,MXNPRc=1,MXNDPc=1)
      PARAMETER(MXVNPc=1,MXVESc=1,MXVPRc=1,MXVDPc=1)
      PARAMETER(MXDNPc=70,MXDPRc=1,MXDDPc=2)
```

To reflect the soil property function given by Equations 4.3 and 4.4, we have to modify the subroutine SPROP in the source code. Lines between SPRO 845 and SPRO 1100 in the subroutine SPROP must be modified according to

```
   WCR=SPP(1,MTYP,1)
   WCS=SPP(2,MTYP,1)
   HAA=SPP(3,MTYP,1)
   ALPHA=SPP(4,MTYP,1)
   BETA=SPP(5,MTYP,1)
   DO 390 KG=1,8
   HNP=HKG(KG)
   HNP=-HNP
   IF(HNP.LE.0) THEN
C
C ------ SATURATED CONDITION
   TH(KG,M)=WCS
   DTH(KG,M)=0.0D0
   USKFCT=1.0D0
C
   ELSE
C
C ------ UNSATURATED CASE
   THMKG=WCR+(WCS-WCR)/(1.0D0+(ALPHA*DABS
        (-HNP-HAA))**BETA) TH(KG,M)=THMKG
   DNOM=1.0D0+(ALPHA*DABS(-HNP-HAA))**BETA
   DTH(KG,M)=(WCS-WCR)*(ALPHA*DABS(-HNP-THAA))**
        (BETA-1.0D0)/1 DNOM**2
   USKFCT=((THMKG-WCR)/(WCS-WCR))**2
  ENDIF
```

Input for Problem No. 3

Table 4.3 shows the input data set for the sample problem described in the above section.

Table 4.3
Input Data Set for Example 3

```
1    THREE DIMENSIONAL SALT INTRUSION, L=M, T=H, M=KG                11010
DATA SET#2 : OPTION PARAMETER
 50  0.5d0
1 1 1 0 3 3
1 1.0 0.5 1.0
2 0 1 1 1
1.0 1.0 1.0 1.0
1  1.0  1.0  1.0
DATA SET #3 : ITERATION PARAMETERS
20 1 400   0.01 0.01
 1 20 0.001 0.001
DATA SET #4 : TIME CONTROL PARAMETERS
960 4
0.5 0.0 1.0 480
300000000003000000000030000000003000000000300000000003000000000030000000
000030000000000300000000030000000003000000003000000000030000000000300 0
000000030000000000300000000030000000030000000003000000000300000000000
300000000003000000000300000000030000000030000000030000000003000000000
000030000000000300000000030000000003000000003000000000030000000000300 0
000000030000000000300000000030000000030000000030000000003000000000000
300000000003000000000300000000030000000030000000030000000003000000000
000030000000000300000000030000000003000000003000000000030000000000300 0
000000030000000000300000000030000000030000000030000000003000000000000
300000000003000000000300000000030000000030000000030000000003000000000
000030000000000300000000030000000003000000003000000000030000000000300 0
000000030000000000300000000030000000030000000030000000003000000000000
3
000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000C  D00000000000000000000000000000000000000000000
000000000000000000000000000C000000000000000000000000000000000000000000000
```

(Sheet 1 of 11)

Table 4.3 (Continued)

```
0000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000
0
6.0 12.0 18.0 1.0D38
DATA SET #5 : MATERIAL PROPERITIES
1 8 8
3.00D-2 3.00D-2 1.0D-2 0.0D0 0.0D0 0.0D0 0.0D0 0.0D0
1.0D0 0.0D0  0.0D0 0.0D0 1.0D0 0.0D0 0.0D0 0.0D0
0.0D0 1.2D0  5.0D0 5.0D0 0.0D0 1.0D0 0.0D0 0.0D0
DATA SET #6 : SOIL PROOERITIES
0      5      0      1.0D0  1.27D8  4.68D2
0.0125D0 0.250D0 0.00D0 0.5D0  2.0D0
0.000D0 0.000D0 0.00D0  0.0D0  0.0
DATA #7 : NODAL COORDINATES
355
 1 4 71  0.0   0.0 0.0 0.0 0.0 2.50
 5 4 71 10.0   0.0 0.0 0.0 0.0 2.50
 9 4 71 20.0   0.0 0.0 0.0 0.0 2.50
13 4 71 30.0   0.0 0.0 0.0 0.0 2.50
17 4 71 40.0   0.0 0.0 0.0 0.0 2.50
28 4 71 50.0   0.0 0.0 0.0 0.0 2.50
39 4 71 60.0   0.0 0.0 0.0 0.0 2.50
50 4 71 70.0   0.0 0.0 0.0 0.0 2.50
61 4 71 80.0   0.0 0.0 0.0 0.0 2.50
 2 4 71  0.0  10.0 0.0 0.0 0.0 2.55
 6 4 71 10.0  10.0 0.0 0.0 0.0 2.55
10 4 71 20.0  10.0 0.0 0.0 0.0 2.55
14 4 71 30.0  10.0 0.0 0.0 0.0 2.55
18 4 71 40.0  10.0 0.0 0.0 0.0 2.55
29 4 71 50.0  10.0 0.0 0.0 0.0 2.55
```

(Sheet 2 of 11)

Table 4.3 (Continued)

```
40 4 71 60.0  10.0 0.0 0.0 0.0 2.55
51 4 71 70.0  10.0 0.0 0.0 0.0 2.55
62 4 71 80.0  10.0 0.0 0.0 0.0 2.55
 3 4 71  0.0  20.0 0.0 0.0 0.0 2.60
 7 4 71 10.0  20.0 0.0 0.0 0.0 2.60
11 4 71 20.0  20.0 0.0 0.0 0.0 2.60
15 4 71 30.0  20.0 0.0 0.0 0.0 2.60
19 4 71 40.0  20.0 0.0 0.0 0.0 2.60
30 4 71 50.0  20.0 0.0 0.0 0.0 2.60
41 4 71 60.0  20.0 0.0 0.0 0.0 2.60
52 4 71 70.0  20.0 0.0 0.0 0.0 2.60
63 4 71 80.0  20.0 0.0 0.0 0.0 2.60
 4 4 71  0.0  30.0 0.0 0.0 0.0 2.65
 8 4 71 10.0  30.0 0.0 0.0 0.0 2.65
12 4 71 20.0  30.0 0.0 0.0 0.0 2.65
16 4 71 30.0  30.0 0.0 0.0 0.0 2.65
20 4 71 40.0  30.0 0.0 0.0 0.0 2.65
31 4 71 50.0  30.0 0.0 0.0 0.0 2.65
42 4 71 60.0  30.0 0.0 0.0 0.0 2.65
 3 4 71 70.0  30.0 0.0 0.0 0.0 2.65
64 4 71 80.0  30.0 0.0 0.0 0.0 2.65
21 4 71 40.0  40.0 0.0 0.0 0.0 2.70
32 4 71 50.0  40.0 0.0 0.0 0.0 2.70
43 4 71 60.0  40.0 0.0 0.0 0.0 2.70
54 4 71 70.0  40.0 0.0 0.0 0.0 2.70
65 4 71 80.0  40.0 0.0 0.0 0.0 2.70
22 4 71 40.0  50.0 0.0 0.0 0.0 2.75
33 4 71 50.0  50.0 0.0 0.0 0.0 2.75
44 4 71 60.0  50.0 0.0 0.0 0.0 2.75
55 4 71 70.0  50.0 0.0 0.0 0.0 2.75
66 4 71 80.0  50.0 0.0 0.0 0.. 2.75
23 4 71 40.0  60.0 0.0 0.0 0.0 2.70
34 4 71 50.0  60.0 0.0 0.0 0.0 2.70
45 4 71 60.0  60.0 0.0 0.0 0.0 2.70
```

Table 4.3 (Continued)

```
56 4 71 70.0  60.0 0.0 0.0 0.0 2.70
67 4 71 80.0  60.0 0.0 0.0 0.0 2.70
24 4 71 40.0  70.0 0.0 0.0 0.0 2.65
35 4 71 50.0  70.0 0.0 0.0 0.0 2.65
46 4 71 60.0  70.0 0.0 0.0 0.0 2.65
57 4 71 70.0  70.0 0.0 0.0 0.0 2.65
68 4 71 80.0  70.0 0.0 0.0 0.0 2.65
25 4 71 40.0  80.0 0.0 0.0 0.0 2.60
36 4 71 50.0  80.0 0.0 0.0 0.0 2.60
47 4 71 60.0  80.0 0.0 0.0 0.0 2.60
58 4 71 70.0  80.0 0.0 0.0 0.0 2.60
69 4 71 80.0  80.0 0.0 0.0 0.0 2.60
26 4 71 40.0  90.0 0.0 0.0 0.0 2.55
37 4 71 50.0  90.0 0.0 0.0 0.0 2.55
48 4 71 60.0  90.0 0.0 0.0 0.0 2.55
59 4 71 70.0  90.0 0.0 0.0 0.0 2.55
70 4 71 80.0  90.0 0.0 0.0 0.0 2.55
27 4 71 40.0 100.0 0.0 0.0 0.0 2.50
38 4 71 50.0 100.0 0.0 0.0 0.0 2.50
49 4 71 60.0 100.0 0.0 0.0 0.0 2.50
60 4 71 70.0 100.0 0.0 0.0 0.0 2.50
71 4 71 80.0 100.0 0.0 0.0 0.0 2.50
 0 0  0  0.0   0.0 0.0 0.0 0.0 0.0
DATA SET #9 : ELEMENT DATA
208
 1 3 52   1   5   6   2  72  76  77  73 71
 2 3 52   2   6   7   3  73  77  78  74 71
 3 3 52   3   7   8   4  74  78  79  75 71
 4 3 52   5   9  10   6  76  80  81  77 71
 5 3 52   6  10  11   7  77  81  82  78 71
 6 3 52   7  11  12   8  78  82  83  79 71
 7 3 52   9  13  14  10  80  84  85  81 71
 8 3 52  10  14  15  11  81  85  86  82 71
 9 3 52  11  15  16  12  82  86  87  83 71
```

(Sheet 4 of 11)

Table 4.3 (Continued)

```
 10 3 52  13  17  18  14  84  88  89  85 71
 11 3 52  14  18  19  15  85  89  90  86 71
 12 3 52  15  19  20  16  86  90  91  87 71
 13 9  1  17  28  29  18  88  99 100  89  1
 65 9  1  88  99 100  89 159 170 171 160  1
117 9  1 159 170 171 160 230 241 242 231  1
169 9  1 230 241 242 231 301 312 313 302  1
 23 9  1  28  39  40  29  99 110 111 100  1
 75 9  1  99 110 111 100 170 181 182 171  1
127 9  1 170 181 182 171 241 252 253 242  1
179 9  1 241 252 253 242 312 323 324 313  1
 33 9  1  39  50  51  40 110 121 122 111  1
 85 9  1 110 121 122 111 181 192 193 182  1
137 9  1 181 192 193 182 252 263 264 253  1
189 9  1 252 263 264 253 323 334 335 324  1
 43 9  1  50  61  62  51 121 132 133 122  1
 95 9  1 121 132 133 122 192 203 204 193  1
147 9  1 192 203 204 193 263 274 275 264  1
199 9  1 263 274 275 264 334 345 346 335  1
  0 0  0   0   0   0   0   0   0   0   0 0
DATA SET #10 : MATERIAL TYPE CORRECTION
0
DATA SET #11 : INITIAL CONDITION
 1 4 71 10.0 -2.50 0.0
 5 4 71 10.0 -2.50 0.0
 9 4 71 10.0 -2.50 0.0
13 4 71 10.0 -2.50 0.0
17 4 71 10.0 -2.50 0.0
28 4 71 10.0 -2.50 0.0
39 4 71 10.0 -2.50 0.0
50 4 71 10.0 -2.50 0.0
61 4 71 10.0 -2.50 0.0
 2 3 71 10.0 -2.55 0.0
286 0 0 0.0   0.0  0.0
```

Table 4.3 (Continued)

```
 6 3 71 10.0 -2.55 0.0
290 0 0 0.0   0.0  0.0
10 3 71 10.0 -2.55 0.0
294 0 0 0.0   0.0  0.0
14 3 71 10.0 -2.55 0.0
298 0 0 0.0   0.0  0.0
18 3 71 10.0 -2.55 0.0
302 0 0 0.0   0.0  0.0
29 3 71 10.0 -2.55 0.0
313 0 0 0.0   0.0  0.0
40 3 71 10.0 -2.55 0.0
324 0 0 0.0   0.0  0.0
51 3 71 10.0 -2.55 0.0
335 0 0 0.0   0.0  0.0
62 3 71 10.0 -2.55 0.0
346 0 0 0.0   0.0  0.0
 3 3 71 10.0 -2.60 0.0
287 0 0 0.0   0.0  0.0
 7 3 71 10.0 -2.60 0.0
291 0 0 0.0   0.0  0.0
11 3 71 10.0 -2.60 0.0
295 0 0 0.0   0.0  0.0
15 3 71 10.0 -2.60 0.0
299 0 0 0.0   0.0  0.0
19 3 71 10.0 -2.60 0.0
303 0 0 0.0   0.0  0.0
30 3 71 10.0 -2.60 0.0
314 0 0 0.0   0.0  0.0
41 3 71 10.0 -2.60 0.0
325 0 0 0.0   0.0  0.0
52 3 71 10.0 -2.60 0.0
336 0 0 0.0   0.0  0.0
63 3 71 10.0 -2.60 0.0
347 0 0 0.0   0.0  0.0
```

(Sheet 6 of 11)

Table 4.3 (Continued)

```
 4 3 71 10.0 -2.65 0.0
288 0 0 0.0   0.0  0.0
 8 3 71 10.0 -2.65 0.0
292 0 0 0.0   0.0  0.0
12 3 71 10.0 -2.65 0.0
296 0 0 0.0   0.0  0.0
16 3 71 10.0 -2.65 0.0
300 0 0 0.0   0.0  0.0
20 3 71 10.0 -2.65 0.0
304 0 0 0.0   0.0  0.0
31 3 71 10.0 -2.65 0.0
315 0 0 0.0   0.0  0.0
42 3 71 10.0 -2.65 0.0
326 0 0 0.0   0.0  0.0
53 3 71 10.0 -2.65 0.0
337 0 0 0.0   0.0  0.0
64 3 71 10.0 -2.65 0.0
348 0 0 0.0   0.0  0.0
21 3 71 10.0 -2.70 0.0
305 0 0 0.0   0.0  0.0
32 3 71 10.0 -2.70 0.0
316 0 0 0.0   0.0  0.0
43 3 71 10.0 -2.70 0.0
327 0 0 0.0   0.0  0.0
54 3 71 10.0 -2.70 0.0
338 0 0 0.0   0.0  0.0
65 3 71 10.0 -2.70 0.0
349 0 0 0.0   0.0  0.0
22 3 71 10.0 -2.75 0.0
306 0 0 0.0   0.0  0.0
33 3 71 10.0 -2.75 0.0
317 0 0 0.0   0.0  0.0
44 3 71 10.0 -2.75 0.0
328 0 0 0.0   0.0  0.0
```

(Sheet 7 of 11)

Table 4.3 (Continued)

```
55 3 71 10.0 -2.75 0.0
339 0 0 0.0   0.0  0.0
66 3 71 10.0 -2.75 0.0
350 0 0 0.0   0.0  0.0
23 3 71 10.0 -2.70 0.0
307 0 0 0.0   0.0  0.0
34 3 71 10.0 -2.70 0.0
318 0 0 0.0   0.0  0.0
45 3 71 10.0 -2.70 0.0
329 0 0 0.0   0.0  0.0
56 3 71 10.0 -2.70 0.0
340 0 0 0.0   0.0  0.0
67 3 71 10.0 -2.70 0.0
351 0 0 0.0   0.0  0.0
24 3 71 10.0 -2.65 0.0
308 0 0 0.0   0.0  0.0
35 3 71 10.0 -2.65 0.0
319 0 0 0.0   0.0  0.0
46 3 71 10.0 -2.65 0.0
330 0 0 0.0   0.0  0.0
57 3 71 10.0 -2.65 0.0
341 0 0 0.0   0.0  0.0
68 3 71 10.0 -2.65 0.0
352 0 0 0.0   0.0  0.0
25 3 71 10.0 -2.60 0.0
309 0 0 0.0   0.0  0.0
36 3 71 10.0 -2.60 0.0
320 0 0 0.0   0.0  0.0
47 3 71 10.0 -2.60 0.0
331 0 0 0.0   0.0  0.0
58 3 71 10.0 -2.60 0.0
342 0 0 0.0   0.0  0.0
69 3 71 10.0 -2.60 0.0
353 0 0 0.0   0.0  0.0
```

Table 4.3 (Continued)

```
26 3 71 10.0 -2.55 0.0
310 0 0 0.0   0.0  0.0
37 3 71 10.0 -2.55 0.0
321 0 0 0.0   0.0  0.0
48 3 71 10.0 -2.55 0.0
332 0 0 0.0   0.0  0.0
59 3 71 10.0 -2.55 0.0
343 0 0 0.0   0.0  0.0
70 3 71 10.0 -2.55 0.0
354 0 0 0.0   0.0  0.0
27 4 71 10.0 -2.50 0.0
38 4 71 10.0 -2.50 0.0
49 4 71 10.0 -2.50 0.0
60 4 71 10.0 -2.50 0.0
71 4 71 10.0 -2.50 0.0
 0 0  0  0.0  0.0  0.0
1 354 1 0.0 0.0 0.0
0   0 0 0.0 0.0 0.0
DATA SET #12 : ELEMENT(DISTRIBUTED) SOURCE/SINK, FLOW
0 0 0 0
DATA SET #13 : POINT (WELL) SOURCE/SINK DATA, FLOW
3 1 2 0
0.0  -1.0d-2 1.0d38 -1.0d-2
44 115 186
1 2 1 1 0
0 0 0 0 0  end of iwtyp
DATA SET #14 : ELEMENT(DISTRIBUTED) SOURCE/SINK, TRANSPORT
0 1 2 0
DATA SET #15 : POINT(WELL) SOURCE/SINK DATA, TRANSPORT
3 1 2 0
0.0  -1.0d-2  0.0d0  1.0d38  -1.0d-2  0.0d0
44 115 186
1 2 1 1 0
0 0 0 0 0  end of iwtyp
```

(Sheet 9 of 11)

Table 4.3 (Continued)

```
DATA SET #16 : RAINFALL/EVAPORATION-SEEPAGE BOUNDARY CONDITIONS
0 0 0 0 0
DATA SET #17 : DIRICHLET BOUNDARY CONDITIONS, FLOW
70 1 3 1
0.0 10.0  10.0   1.0  1.0d38   6.0d0
  1   5   9  13  17  28  39  50  61
 72  76  80  84  88  99 110 121 132
143 147 151 155 159 170 181 192 203
214 218 222 226 230 241 252 263 274
285 289 293 297 301 312 323 334 345
 27  38  49  60  71
 98 109 120 131 142
169 180 191 202 213
240 251 262 273 284
311 322 333 344 355
1 69 1 1 0
0  0 0 0 0
DATA SET #18 : CAUCHY BOUNDARY CONDITIONS, FLOW
0 0 0 0 0
DATA SET #19 : NEUMANN BOUNDARY CONDITIONS, FLOW
0 0 0 0 0
DATA SET #20 : RUN-IN/SEEP-OUT BOUNDARY
0 0 0 0 0
DATA SET #21 : DIRICHLET BOUNDARY CONDITIONS, TRANSPORT
70 1 2 0
0.0 1.0 1.0D38 1.0
  1   5   9  13  17  28  39  50  61
 72  76  80  84  88  99 110 121 132
143 147 151 155 159 170 181 192 203
214 218 222 226 230 241 252 263 274
285 289 293 297 301 312 323 334 345
 27  38  49  60  71
 98 109 120 131 142
169 180 191 202 213
```

(Sheet 10 of 11)

Table 4.3 (Concluded)

```
240 251 262 273 284
311 322 333 344 355
1 69 1 1 0
0  0 0 0 0
DATA SET #22 : CAUCHY BOUNDARY CONDITIONS, TRANSPORT
0 0 0 0 0
DATA SET #23 : NEUMANN BOUNDARY CONDITIONS, TRANSPORT
0 0 0 0 0
   0 ----------end of job----------0000000
```

(Sheet 11 of 11)

References

Andrews, R. W. (1981). "Salt-water intrusion in the Costa de Hermisillo, Mexico: A numerical analysis of water management proposals," *Ground Water* 19(6), 635-647.

Atkinson, S. F., Miller, G. D., Curry, D. S., and Lee, S. B. (1986). *Salt water intrusion: Status and potential in the contiguous United States.* Lewis Publishers, Inc., Chelsea, MI, 390 pp.

Bear, J. (1979). *Hydraulics of groundwater.* McGraw-Hill, New York, 567 pp.

Beran, M. J. (1955). "Dispersion of soluble matters in slowing moving fluids," Ph.D. Diss., Harvard University, Boston, MA.

Bohn, H. L., McNeal, B. L., and O'Connor, G. A. (1985). *Soil chemistry,* John Wiley & Sons, New York, 341 pp.

Bowen, R. (1986). *Groundwater.* 2nd ed., Elsevier, New York, 427 pp.

Clough, R. W. (1971). "Analysis of structural vibrations and dynamic response." *Recent advances in matrix methods of structural analysis and design;* Proceedings, United States Japan Seminar on Matrix Methods of Structural Analysis and Design. R. H. Gallagher, Y. Yamada, and J. T. Oden, ed., University of Alabama Press, Huntsville, AL.

Contractor, D. N., and Srivastava, R. (1990). "Simulation of saltwater intrusion in the northern Guam lens using a microcomputer," *Journal of Hydrology* 118, 87-106.

De Wiest, R. J. M. (1965). *Geohydrology.* John Wiley & Sons, New York, 366 pp.

Essaid, H. I. (1990). "The computer model SHARP, a quasi-three-dimensional finite-difference model to simulate freshwater and saltwater flow in layered coastal aquifer systems," Water-Resources Investigations

Report 90-4130, U. S. Geological Survey, U. S. Government Printing Office, Washington, DC.

Freeze, R. A. (1972a). "Role of subsurface flow in generating surface runoff: 1. Base flow contribution to channel flow," *Water Resources Research* 8, 609-623.

Freeze, R. A. (1972b). "Role of subsurface flow in generating surface runoff: 2. Upstream source areas. *Water Resources Research* 8, 1272-1283.

Frind, E. O. (1982a). "Seawater intrusion in continuous coastal aquifer-aquitard systems," *Advances in Water Resources* 5(2), 89-97.

Frind, E. O. (1982b). "Simulation of long-term transient density-dependent transport in groundwater," *Advances in Water Resources* 5(2), 73-88.

Galeati, G., Gambolati, G., and Neuman, S. P. (1992). "Coupled and partially coupled Eulerian-Lagrangian model of freshwater-seawater mixing," *Water Resources Research* 28(1), 149-165.

Hager, W. W. (1988). *Applied numerical linear algebra*. Prentice Hall, Englewood Cliffs, NJ, 424 pp.

Henry, H. R. (1959). "Salt intrusion into fresh water aquifers," *Journal of Geophysical Resources* 64(11), 1911-1919.

Henry, H. R. (1964). "Effects of dispersion on salt encroachment in coastal aquifers," *Seawater in coastal aquifers*. H. H. Cooper et al., ed., Water Supply Paper 1613-C, U. S. Geological Survey, U. S. Government Printing Office, Washington, DC, 35-69.

Huyakorn, P. S., Springer, E. P., Guvanasen, V., and Wadsworth, T. D. (1986). "A three-dimensional finite-element model for simulating water flow in variably saturated porous media," *Water Resources Research* 22(13), 1790-1808.

Huyakorn, P. S., Andersen, P. F., Mercer, J. W., and White, H. O., Jr. (1987). "Saltwater intrusion in aquifers: Development and testing of a three-dimensional finite element model," *Water Resources Research* 23(2), 293-312.

Istok, J. D. (1989). *Groundwater modeling by the finite element method*. American Geophysical Union, Washington, DC, 495 pp.

Lee, C. H., and Cheng, R. T. (1974). "On seawater encroachment in coastal aquifers," *Water Resources Research* 10(5), 1039-1043.

Liggett, J. A., and Liu, P. L-F. (1979). "Unsteady flow in confined aquifers: A comparison of two boundary integral methods," *Water Resources Research* 15(4), 861-866.

Liu, P. L-F., Cheng, A. H-D., Liggett, J. A., and Lee, J. H. (1981). "Boundary integral equation solutions to moving interface between two fluids in porous media," *Water Resources Research* 17(5), 1445-1452.

Nguyen, V. V., Gray, W. G., Pinder, G. F., Botha, J. F., and Crefar, D. A. (1982). "A theoretical investigation on the transport of chemicals in reactive porous media," *Water Resources Research* 18(4), 1149-1156.

Owczarek, J. A. (1964). *Fundamentals of gas dynamics*. International Textbook, Scranton, PA.

Pinder, G. F., and Cooper, H. H., Jr. (1970). "A numerical technique for calculating the transient position of the saltwater front," *Water Resources Research* 6(3), 875-880.

Pinder, G. F., and Gray, W. G. (1977). *Finite element simulation in surface and subsurface hydrology*. Academic Press, New York, 295 pp.

Reilly, T. E., and Goodman, A. S. (1985). "Quantitative analysis of saltwater-freshwater relationships in groundwater systems - A historical perspective," *Journal of Hydrology* 80, 125-160.

Segol, G., Pinder, G. F. and Gray, W. G. (1975). "A Galerkin finite element technique for calculating the transient position of the salt water front," *Water Resources Research* 11(2), 343-347.

Shamir, V., and Dagan, G. (1971). "Motion of the seawater interface in coastal aquifers: a numerical solution," *Water Resources Research* 7(3), 644-657.

Sherif, M. M., Singh, V. P., and Amer, A. M. (1990). "A sensitivity analysis of '2D-FED', A model for seawater encroachment in leaky coastal aquifers," *Journal of Hydrology* 118, 343-356.

Shoup, T. E. (1984). *Applied numerical methods for the microcomputer*. Prentice-Hall, Englewood Cliffs, NJ, 262 pp.

van Genechten, M. Th. (1980). "A closed form equation for predicting the hydraulic conductivity of unsaturated soils," *Soil Science Society of America Journal* 44, 892-898.

Volker, R. E., and Rushton, K. R. (1982). "An assessment of the importance of some parameters for seawater intrusion in aquifers and a comparison of dispersive and sharp-interface modelling approaches," *Journal of Hydrology* 56, 239-250.

Wang, J. D., and Connor, J. J. (1975). "Mathematical modeling of near-coastal circulation," Report No. MITSG 75-13, Massachusetts Institute of Technology, Cambridge, MA.

Wilson, J. L., and Sa da Costa, A. A. G. (1982). "Finite element simulation of a saltwater/freshwater interface with indirect toe tracking," *Water Resources Research* 18(4), 1069-1080.

Yeh, G. T. (1981). "On the computation of Darcian velocity and mass balance in the finite element modeling of groundwater flow," *Water Resources Research* 17(5), 1529-1534.

Yeh, G. T. (1985). "Comparison of successive iteration and direct methods to finite element equations of aquifer contaminant transport," *Water Resources Research* 21(3), 272-280.

Yeh, G. T. (1987). "FEMWATER: A finite element model of water flow through saturated-unsaturated porous media," First Revision, Rep. ORNL-5567/R1, Oak Ridge National Laboratory, Oak Ridge, TN.

Yeh, G. T. (1991). "Class notes: CE597D: Computational Subsurface Hydrology Part I," The Pennsylvania State University, University Park, PA.

Yeh, G. T. (1992). "Class notes: CE597C: Computational Subsurface Hydrology Part II," The Pennsylvania State University, University, PA.

Yeh, G. T., and Tripathi, V. S. (1987). "Strategies in modeling hydrogeological transport of reactive multi-chemical components." *Groundwater and the environment; proceedings, International Groundwater Conference.* Faculty of Engineering and Faculty of Physical and Applied Science, Universitti Kebangsaan Malaysis, Kuala Lumpur, Malaysia, E26-E36.

Yeh, G. T., and Ward, D. S. (1980). "FEMWATER: A finite element model of water flow through saturated-unsaturated porous media," ORNL-5567, Oak Ridge National Laboratory, Oak Ridge, TN, 137 pp.

Yeh, G. T., and Ward, D. S. (1981). "FEMWASTE: A finite-element model of waste transport through saturated-unsaturated porous media," Rep. ORNL-5601, Oak Ridge National Laboratory, Oak Ridge, TN.

Appendix A
Data Input Guide

Note: Data sets 2 through 23 must be preceded by a record containing description of the data set.

Title

One record with FORMAT(I5,A70,I2,3I1) per problem. This record contains the following variables:

a. NPROB = Problem number.

b. TITLE = Title of the problem. It may contains up to 65 characters.

c. IMOD = Integer indicating the simulation modes to be carried on.

(1) 0 = Do the initial variable computation ONLY, for both flow and transport simulations. The purpose for this mode is to verify the input data.

(2) 10 = Do the flow simulation ONLY.

(3) 1 = Do the transport simulation only.

(4) 11 = Do both flow and transport simulations.

d. IGEOM = Integer indicating if

(1) The geometry, boundary and pointer arrays are to be printed.

(2) The boundary and pointer arrays are to be computed or read via logical units. If to be computed, they should be written on logical units.

If IGEOM is even number, (1) will not be printed. If IGEOM is odd number, (1) will be printed. If IGEOM is less than or equal to 1, boundary arrays will be computed and written on logical unit, but if IGEOM is greater than 1, boundary arrays will be read via logical unit 3. If IGEOM is less than or equal to 3, pointer arrays will be computed and written on Logical Unit 4, but if IGEOM is greater than 3, pointer arrays will be read via Logical Unit 4.

e. IBUG = Integer indicating if the diagnostic output is desired

(1) 0 = No.

(2) 1 = Yes.

f. ICHNG = Integer control number indicating if the cyclic change of rainfall-seepage nodes is to be printed

(1) = 0, no.

(2) = 1, yes.

Option Parameters

Six lines of free-formatted data records are required for this data set.

Line 1:

NITERHT = Iteration numbers allowed for solving the coupled nonlinear equation

OMEHT = Iteration parameter for solving the coupled nonlinear equations

Line 2:

KSSf = Flow steady-state control

a. 0 = steady-state solution desired.

b. 1 = transient state or transient solutions.

KSSt = Transport steady-state control

a. 0 = steady-state solution desired.

b. 1 = transient state or transient solutions.

ILUMP = Is mass lumping?

a. 0 = no.

b. 1 = yes.

IMID = Is mid-difference?

a. 0 = no.

b. 1 = yes.

IPNTSf = Is pointwise iterative matrix solver to be used for flow simulations?

a. 0 = no.

b. 1 = yes.

IPNTSt = Is pointwise iterative matrix solver to be used for transport simulations?

a. 0 = no.

b. 1 = yes.

Line 3:

KGRAV = Gravity term control

a. 0 = no gravity term.

b. 1 = with gravity term.

Wf = Time derivative weighting factor for flow simulations

a. 0.5 = Crank-Nicholson central.

b. 1.0 = backward difference and/or mid-difference.

OMEf = Iteration parameter for solving the nonlinear flow equation

a. 0.0 - 1.0 = under-relaxation.

b. 1.0 - 1.0 = exact relaxation.

c. 1.0 - 2.0 = over-relaxation.

OMIf = Relaxation parameter for solving the linearized flow matrix equation pointwisely or blockwisely

a. 0.0 - 1.0 = under-relaxation.

b. 1.0 - 1.0 = exact relaxation.

c. 1.0 - 2.0 = over-relaxation.

Line 4:

KVIt = Velocity input control

a. -1 = card input for velocity and moisture content.

b. 1 = steady-state velocity and moisture content will be calculated from steady-state flow simulations.

c. 2 = transient velocity and moisture content will be obtained from transient flow simulations.

IWET = Weighting function control

a. 0 = Galerkin weighting.

b. 1 = Upstream weighting.

IOPTIM = Optimization factor computing indicator

a. 1 = Optimization factor is to be computed.

b. 0 = optimization factor is to be set to -1.0 or 0.0 or 1.0.

KSORP = Sorption model control

a. 1 = linear isotherm.

b. 2 = Freundlich isotherm.

c. 3 = Langmuir isotherm.

LGRN = Lagrangian approach control

a. 0 = no.

b. 1 = yes.

Line 5:

Wt = Time derivative weighting factor for transport simulations

a. 0.5 = Crank-Nicholson central.

b. 1.0 = backward difference and/or mid-difference.

WVt = Integration factor for velocity; should be between 0.0 to 1.0.

OMEt = Iteration parameter for solving the nonlinear transport matrix equation

a. 0.0 - 1.0 = under-relaxation.

b. 1.0 - 1.0 = exact relaxation.

c. 1.0 - 2.0 = over-relaxation.

OMIt = Relaxation parameter for solving the linearized transport matrix equation pointwisely or blockwisely;

a. 0.0 - 1.0 = under-relaxation.

b. 1.0 - 1.0 = exact relaxation.

c. 1.0 - 2.0 = over-relaxation.

Line 6

This line is needed if and only if IPNTSf or IPNTSt is greater than 0.

IEIGEN = signal of parameter estimation for GG in the polynomial preconditioned conjugate gradient method:

a. Zero = not requested.

b. Non-zero = requested.

GG = the upper bound on the maximum eigenvalue of the coefficient matrix used in the polynomial preconditioned conjugate gradient method.

ALPHA = weighting factor for computing the diagonal element of the diagonal matri :d in the modified incomplete Cholesky preconditioned conjugate grad method.

OMEGA = relaxation parameter used in the SSOR PCG method.

Iteration Parameters

Two subsets of free-formatted data records are required for this data set, one for flow simulations, the other for transport simulations.

Subset 1: For flow simulations

NITERf = Number of iterations allowed for solving the nonlinear flow equation.

NCYLf = Number of cycles permitted for iterating rainfall-seepage boundary conditions per time-step.

NPITERf = Number of iterations permitted for solving the linearized flow equation using block or pointwise iterative matrix solver.

TOLAf = Steady-state convergence criteria for flow simulations, (L).

TOLBf = Transient-state convergence criteria for flow simulations, (L).

Subset 2: For transport simulations

NITERt = Number of iterations allowed for solving the nonlinear transport equation.

NPITERt = Number of iterations for block or pointwise solution to solve the linearized transport equation.

TOLAt = Steady-state convergence criteria for transport simulations.

TOLBt = Transient-state convergence criteria for transport simulations.

Time Control Parameters

Five subsets of data records are required for this data set.

Subset 1: free format

NTI = Number of time-steps or time increments.

NDTCHG = Number of times to reset time-step size to initial time-step size.

Subset 2: free format

DELT = Initial time-step size, (T).

CHNG = Percentage of change in time-step size in each of the subsequent time increments (dimensionless in decimal point).

DELMAX = Maximum value of DELT, (T).

TMAX = Maximum simulation time, (T).

Subset 3: format = 8I1

KPR0 = Printer control for steady-state and initial conditions

a. 0 = print nothing.

b. 1 = print FLOW, FRATE, and TFLOW.

c. 2 = print above (1) plus pressure head H.

d. 3 = print above (2) plus total head.

e. 4 = print above (3) plus moisture content.

f. 5 = print above (4) plus Darcy velocity.

KPR(I) = Printer control for the I^{th} (I = 1, 2, ..., NTI) time-step similar to KPR0.

Subset 4: format = 8I1

KDSK0 = Auxiliary storage control for steady-state and initial condition; 0 = no storage, 1 = store on Logical Unit 1.

KDSK(I) = Auxiliary storage control for the I^{th} (I = 1, 2, ..., NTI) time-step similar to KDSK0.

Subset 5: free format

TDTCH(I) = Time when the I^{th} (I = 1, 2, ..., NDTCHG) step-size-resetting is needed.

NOTE: NTI can be computed by NTI = I1 + 1 + I2 + 1, where I1 = largest integer not exceeding Log(DELMAX/DELT)/Log(1+CHNG),

I2 = largest integer not exceeding (RTIME-DELT*((1+CHNG)** (I1+1)-1)/CHNG)/DELMAX, RTIME = real simulation time, DELMAX, DELT,and CHNG are defined in Data Set 3.

Material Properties

Four subsets of free-formatted data records are required for this data set.

Subset 1:

NMAT = Number of material types.

NMPPM = Number of material properties per material = 8 for the present version.

nrmp = Number of constants for computing RHO (fluid density) and MU (fluid dynamic viscosity) as function of concentration.

Subset 2:

A total of NMAT records are needed per problem, one each for one material.

PROPf(1,I) = Saturated xx-conductivity or permeability of the medium I, (L/T or L**2).

PROPf(1,I) = Saturated yy-conductivity or permeability of the medium I, (L/T or L**2).

PROPf(1,I) = Saturated zz-conductivity or permeability of the medium I, (L/T or L**2).

PROPf(1,I) = Saturated xy-conductivity or permeability of the medium I, (L/T or L**2).

PROPf(1,I) = Saturated xz-conductivity or permeability of the medium I, (L/T or L**2).

PROPf(1,I) = Saturated yz-conductivity or permeability of the medium I, (L/T or L**2).

Subset 3:

Parameters a_1, a_2, ..., a_8 used in Equations 2.2b and 2.2c.

RHOMU(1..4) = Coefficients for calculating fluid density as function of concentration. The default setting is four coefficients.

RHOMU(5..nrmp) = Coefficients for calculating fluid dynamic viscosity as function of concentration.

Subset 4:

A total of NMAT records are needed per problem, one each for one material.

PROPt(1,I) = Distribution coefficient (L**3/M) or Freundlich K or Langmuir K for medium I.

PROPt(2,I) = Bulk density, (M/L**3) for medium I.

PROPt(3,I) = Longitudinal dispersivity, (L), for medium I.

PROPt(4,I) = Lateral dispersivity, (L), for medium I.

PROPt(5,I) = Molecular diffusion coefficient, (L**2/T), for medium I.

PROPt(6,I) = Tortuosity (Dimensionless) for medium I.

PROPt(7,I) = Decay constant, (1/L) in medium I.

PROPt(8,I) = Freundlich N or Langmuir SMAX for medium I.

Soil Properties

Three or five subsets of free-formatted data records are required for this data set depending on the forms of the soil property functions given.

Subset 1: Soil property control parameters

KSP = Soil property input control.

a. 0 = analytical input.

b. 1 = tabular data input.

NSPPM = Number of points in tabular soil property functions or number of parameters to specify analytical soil functions per material.

KCP = Permeability input control.

a. 0 = input saturated hydraulic conductivity.

b. 1 = input saturated permeability.

RHO = Referenced density of water, (M/L**3).

GRAV = Acceleration of gravity, (L/T**2).

VISC = Referenced dynamic viscosity of water, (M/L/T).

Subset 2a: Analytical soil parameters

This data subset is needed if and only if KSP is not 0. Two sets of records are required, one for moisture-content parameters and the other for conductivity (permeability) parameters.

SPP(J,I,1) = Analytical moisture-content parameter J of material I, J = 1..NSPPM. NMAT sets of these parameters are required for I = 1..NMAT. That is, if SPP(J,I,1) for J = 1..NSPPM can be put on a single line, we need NMAT consecutive lines for the sets of parameters.

SPP(J,I,2) = Analytical relative conductivity parameter J of material I. Similar input data setting is required for these parameters as for SPP(J,I,1)

Subset 2b: Soil properties in tabular form

This data subset is needed if and only if KSP is 0. Four sets of records are needed: pressure, water content, relative conductivity (or relative permeability), and water capacity, respectively.

SPP(J,I,4) = Tabular value of pressure head of the J^{th} point for material I. NMAT sets of these parameters are required for I = 1..NMAT. That is, if SPP(J,I,4) for J = 1..NSPPM can be put on a single line, we need NMAT consecutive line for the sets of parameters.

SPP(J,I,1) = Tabular value of moisture content of the J^{th} point in material I. Similar input data setting is required for these parameters as for SPP(J,I,4).

SPP(J,I,2) = Tabular value of relative conductivity of the J^{th} point in material I. Similar input data setting is required for these parameters as for SPP(J,I,4).

SPP(J,I,3) = Tabular value of moisture content capacity of the J^{th} point in material I. Similar input data setting is required for these parameters as for SPP(J,I,4)

Nodal Coordinate

Two subsets of free-formatted data records are required.

Subset 1:

NNP = Number of nodes.

Subset 2: nodal coordinates

Coordinates for NNP nodes are needed if KVI .LE. 0. Usually a total of NNP records (KVI records) are required. However, if a group of subsequent nodes appear in regular pattern, automatic generation can be made. Each record contains the following variables and is FREE-FORMATTED.

NI = Node number of the first node in the sequence.

NSEQ = NSEQ subsequent nodes will be automatically generated.

NAD = Increment of node number for each of the NSEQ subsequent nodes.

XNI = x-coordinate of node NI, (L).

YNI = y-coordinate of node NI, (L).

ZNI = z-coordinate of node NI, (L).

XAD = Increment of x-coordinate for each of the NSEQ subsequent nodes, (L).

YAD = Increment of y-coordinate for each of the NSEQ subsequent nodes, (L).

ZAD = Increment of z-coordinate for each of the NSEQ subsequent nodes, (L).

NOTE: A record with nine *0's* must be used to signal the end of this data set.

Subregion Data

This data set is not required if both or either IPNTSf and IPNTSt is 0. Three subsets of free-formatted data records are required.

Subset 1:

One free-format data record is needed for this data subset.

NREGN = Number of subregions.

Subset 2: Number of nodes for each subregion

Normally, NREGN records are required. However, if regular pattern appears, automatic generation can be made. Each record contains the five variables and is FREE-FORMATTED.

NK = Subregion number of the first subregion region in a sequence.

NSEQ = NSEQ subsequent subregions will have their number of nodes automatically generated.

NKAD = Increment of NK in each of the NSEQ subsequent subregions.

NODES = Number of nodes for the subregion NK.

NOAD = Increment of NODES in each of the NSEQ subsequent subregions.

NOTE: A record with five *0's* must be used to end the *input of this* data subset.

Subset 3: Mapping between global nodes and subregion nodes

This data subset should be repeated NREGN times, one for each subregion. For each subregion, normally, LNNP records are needed. However, automatic generation can be made if subregional node number appears in regular pattern. Each record contains five variables and is FREE-FORMATTED.

LI = Local node number of the first node in a sequence.

NSEQ = NSEQ subsequent local nodes will be generated automatically.

LIAD = Increment of LI for each of the NSEQ subsequent nodes.

NI = Global node number of local node LI.

NIAD = Increment of NI for each of the NSEQ subsequent nodes.

NOTE: A record with five *0's* must be used to signal the end of this data subset.

Element Data

Two subsets of free-formatted data records are required for this data set.

Subset 1

NEL = Number of elements.

Subset 2

Element incidences for NEL elements are needed if KVI .LE. 0. Usually, a total of NEL records is needed. However, if a group of elements appear in regular pattern, automatic generation is made. Each record contains the following variables and is FREE-FORMATTED.

MI = Global element number of the first element in a sequence.

NSEQ = NSEQ subsequent elements will be automatically generated.

MIAD = Increment of MI for each of the NSEQ subsequent elements.

IE(MI,1) = Global node number of the first node of element MI.

IE(MI,2) = Global node number of the second node of element MI.

IE(MI,3) = Global node number of the third node of element MI.

IE(MI,4) = Global node number of the fourth node of element MI.

IE(MI,5) = Global node number of the fifth node of element MI.

IE(MI,6) = Global node number of the sixth node of element MI.

IE(MI,7) = Global node number of the seventh node of element MI.

IE(MI,8) = Global node number of the eighth node of element MI.

IEMAD = Increment of IE(MI,1) through IE(MI,8) for each of the NSEQ elements.

NOTE: IE(MI,1)-IE(MI,8) are numbered according the convention shown in Figure C.1. The first four nodes start from the front lower left corner and progress around the bottom element surface in a counterclockwise direction. The other four nodes begin from the front upper left corner and progress around the top element surface in a counterclockwise direction.

Material Type Correction

Two subsets of free-formatted data records are required for this data set.

Subset 1

NCM = Number of elements with material corrections.

Subset 2

This set of data records is required only if NCM > 0. Normally, NCM records are required. However, if a group of elements appear in regular pattern, automatic generation may be made. Each record contains the following variables.

MI = Global element number of the first element in the sequence.

NSEQ = NSEQ subsequent elements will be generated automatically.

MAD = Increment of element number for each of the NSEQ subsequent elements.

MITYP = *Type of material correction for element* MI.

MTYPAD = Increment of MITYP for each of the NSEQ subsequent elements.

NOTE: A line with five *0's* must be used to signal the end of this data set.

Card Input for Initial or Pre-Initial Conditions

Two subsets of free-formatted data records are required for this data set, one for initial pressure head, the other for initial concentration. Generally, for each subset NNP record, one record for each node is needed. However, if a group of nodes appear in regular pattern, autogeneration is made.

Subset 1: Initial pressure head

Each record contains the following variables. This subset is needed if IMOD = 10 or IMOD = 11.

NI = Global node number of the first node in the sequence.

NSEQ = NSEQ subsequent nodes will be generated automatically.

NAD = Increment of node number for each of the NSEQ nodes.

HNI = Initial or pre-initial pressure head of node NI, (L).

HAD = Increment of initial or pre-initial head for each of the NSEQ nodes, (L).

HRD = 0.0

NOTE: A line with six *0's* must be used to signal the end of this data set.

NOTE ON INITIAL CONDITIONS AND RESTARTING: The initial condition for a transient calculation may be obtained in two different ways: from card input or steady-state calculation using time-invariant boundary conditions that are different from those for transient computation. In the latter case a card input of the pre-initial conditions is required as the zeroth order iterate of the steady-state solution.

NOTE ON STEADY-STATE INPUT: Steady-state option may be used to provide either the final state of a system under study or the initial conditions for a transient state calculation. In the former case KSSf = 0, KSSt = 0, and NTI = 0, and in the latter case KSSf = 0 or KSSt = 0 and NTI > 0. If KSSf > 0 and KSSt > 0, there will be no steady-state calculation.

Subset 2: Initial concentration

Each record contains the following variables. This subset is needed if IMOD = 1 or IMOD = 11.

NI = Global node number of the first node in the sequence.

NSEQ = NSEQ subsequent nodes will be generated automatically.

NAD = Increment of node number for each of the NSEQ nodes.

CNI = Initial or pre-initial concentration of node NI, (M/L**3).

CAD = Increment of CNI for each of the NSEQ nodes, (M/L**3).

CRD = Geometrical increment of CNI for each of the NSEQ subsequent nodes.

NOTE: A record with six *0's* must be used to signal the end of this data set.

NOTE ON INITIAL CONDITIONS: The initial condition for a transient calculation may be obtained in two different ways: from card input or steady-state calculation using time-invariant boundary conditions that are

different from those for transient computation. In the latter case a card input of the pre-initial conditions is required as the zeroth order iterate of the steady-state solution.

NOTE ON STEADY-STATE INPUT: Steady-state option may be used to provide either the final state of a system under study or the initial conditions for a transient state calculation. In the former case KSS = 0 and NTI = 0, and in the latter case KSS = 0 and NTI .GT. 0. If KSS .GT. 0, there will be no steady-state calculation.

Element (Distributed) Source/Sink for Flow Simulations

This data set is needed if IMOD = 10 or IMOD = 11. Four subsets of free-formatted data records are required in this data set.

Subset 1: control parameters

NSEL = Number of source/sink elements.

NSPR = Number of source/sink profiles.

NSDP = Number of data points in each of the NSPR source/sink profiles.

KSAI = Is element-source/sink profile to be input analytically?

a. 0 = no.

b. 1 = yes.

Subset 2: source/sink profiles

This group of data is needed if and only if NSEL .GT. 0. For each data subset record, NSDP of the data pair (TSOSF(J,I),SOSF(J,I)) is required. If this data subset record can be fitted in a line. we will need NSPR lines.

TSOSF(J,I) = Time of the J^{th} data point in the I^{th} profile, (T).

SOSF(1,I) = Source/sink value of the J^{th} data point in the I^{th} profile, (L**3/T/L**2/L).

Subset 3: global source/sink element number

This group of data is needed if and only if NSEL .GT. 0. NSEL data points are required for this record.

MSEL(I) = Global element number of the I^{th} compressed distributed source/sink element.

Subset 4: Source type assigned to each element

Usually one record per element. However, automatic generation can be made. For I^{th} (I = 1, 2, ...,) record, it contains the following.

MI = Global element number of the first element in the sequence.

NSEQ = NSEQ elements will be generated automatically.

MAD = Increment of element number for each of the NSEQ elements.

MITYP = Source type in element MI.

MTYPAD = Increment of MITYP for each of the NSEQ elements.

NOTE: A line with five *0's* is used to signal the end of this data set.

Point (Well) Source/Sink Data for Flow Simulation

This data set is needed if IMOD = 10 or IMOD = 11. Four subsets of free-formatted data records are required for this data set.

Subset 1: control parameters

NWNP = Number of well or point source/sink nodal points.

NWPR = Number of well or point source/sink strength profiles.

NWDP = Number of data points in each of the NWPR profiles.

KWAI = Is well-source/sink profile to be input analytically?

a. 0 = no.

b. 1 = yes.

Subset 2: source/sink profiles

This group of data is needed if and only if NWNP .GT. 0. For each data subset record, NWDP of the data pair (TWSSF(J,I),WSSF(J,I)) are required. If this data subset record can be fitted in a line, we will need NWPR lines.

TWSSF(J,I) = Time of the J^{th} data point in the I^{th} profile, (T).

WSSF(J,I) = Source/sink value of the J^{th} data point in the I^{th} profile, (L**3/T/L).

Subset 3: global source/sink element number

This group of data is needed if and only if NWNP .GT. 0. NWNP data points are required for this record.

NPW(I) = Global node number of the I^{th} compressed well source/sink node.

Subset 4: Source type assigned to each well

Usually one record per element. However, automatic generation can be made. For I^{th} (I = 1, 2, ...,) record, it contains the following.

NI = Compressed well node number of the first node in the sequence.

NSEQ = NSEQ nodes will be generated automatically.

NAD = Increment of well node number for each of the NSEQ nodes.

NITYP = Source type in node NI.

NTYPAD = Increment of NITYP for each of the NSEQ nodes.

NOTE: A line with five *0's* is used to signal the end of this data set.

Element (Distributed) Source/Sink for Transport Simulations

This data set is needed if IMOD = 1 or IMOD = 11. Four subsets of free-formatted data records are required in this data set.

Subset 1: control parameters

NSEL = Number of source/sink elements.

NSPR = Number of source profiles, should be .GE. 1.

NSDP = Number of data points in each profile, should be .GE. 2.

KSAI = Is element-source/sink profile to be input analytically?

a. 0 = no.

b. 1 = yes.

Record subset 2: source/sink profile

This data subset is needed if and only if NSEL .GT. 0. For each data subset record, NSDP of the data group (TSOSF(J,I),SOSF(J,I,1),SOSF(J,I,2)) are required. If this data subset record can be fitted in a line, we will need NSPR lines.

TSOSF(J,I) = Time of J^{th} data point in I^{th} profile, (T).

SOSF(J,I,1) = Source/sink flow rate of the J^{th} data point in the I^{th} profile, (L**3/T/L**3); positive for source and negative for sink.

SOSF(J,I,2) = Source/sink concentration of the J^{th} data point in the I^{th} profile, (M/L**3).

Subset 3: global source/sink element number

NSEL data points are required for this record.

LES(I) = Global element number of the I^{th} compressed distributed source/sink element.

Subset 4: Source type assigned to each element

Usually one record per element. However, automatic generation can be made. For I^{th} (I = 1, 2, ...,) record, it contains the following.

MI = Global element number of the first element in the sequence.

NSEQ = NSEQ elements will be generated automatically.

MAD = Increment of element number for each of the NSEQ elements.

MITYP = Source type in element MI.

MTYPAD = Increment of MITYP for each of the NSEQ elements.

NOTE: A line with five *0's* is used to signal the end of this data set.

Point (Well) Source/Sink Data for Transport Simulation

This data set is needed if IMOD = 1 or IMOD = 11. Four subsets of data records are required for this data set.

Subset 1: control parameters

NWNP = Number of well or point source/sink nodes.

NWPR = Number of well or point source/sink strength profiles.

NWDP = Number of data points in each of the NWPR profiles.

KWAI = Is well-source/sink profile to be input analytically?

a. 0 = no.

b. 1 = yes.

Subset 2: source/sink profiles

This group of data is needed if and only if NWNP .GT. 0. For each data subset record, NWDP of the data group (TWSSF(J,I),WSSF(J,I,1),WSSF (J, I,2)) are required. If this data subset record can be fitted in a line, we will need NWPR lines.

TWSSF(J,I) = Time of J^{th} data point in I^{th} profile, (T).

WSSF(J,I,1) = Source/sink flow rate of the J^{th} data point in the I^{th} profile, (L**3/T/L**3); positive for source and negative for sink.

WSSF(J,I,2) = Source/sink concentration of the J^{th} data point in the I^{th} profile, (M/L**3).

Subset 3: global source/sink element number

This group of data is needed if and only if NWNP .GT. 0. NWNP data points are required for this record.

NPW(I) = Global node number of the I^{th} compressed point source/sink node.

Subset 4: Source type assigned to each well

Usually one record per element. However, automatic generation can be made.

NI = Compressed point source/sink node number of the first node in a sequence.

NSEQ = NSEQ nodes will contain the source types that will be automatically generated.

NIAD = Increment of NI for each of the NSEQ nodes.

NITYP = Source type in node NI.

NTYPAD = Increment of NITYP for each of the NSEQ subsequent nodes.

NOTE: A record with five *0's* must be used to signal the end of this data set.

Rainfall/Evaporation-Seepage Boundary Conditions

This data set is needed if IMOD = 10 or IMOD = 11. Seven subsets of data records are required for this data set.

Subset 1: control parameters

NVES = Number of variable boundary element sides.

NVNP = Number of variable boundary nodal points.

NRPR = Number of rainfall profiles.

NRDP = Number of rainfall data points in each of the NRPR rainfall profiles.

KRAI = Is rainfall profile to be input analytically,

a. 0 = no.

b. 1 = yes.

Subset 2: boundary profiles

This subset is required only when NVES is not 0. NRPR profiles are needed. For each profile, NRDP of the data pair (TRF(J,I),RF(J,I)) are required. If these data pairs can fit in a line, we will need NRPR of data lines.

TRF(J,I) = Time of the J^{th} data point in the I^{th} profile, (T).

RF(J,I) = Rainfall/evaporation rate of the J^{th} data point in the I^{th} profile, (L/T).

Subset 3: boundary profile types assigned to each element

At most, NVES records are needed. However, automatic generation can be made. For I^{th} (I = 1, 2, ...,) record, it contains the following variables.

MI = Compressed VB element side of the first side in the sequence.

NSEQ = NSEQ sides will be generated automatically.

MIAD = Increment of NI for each of the NSEQ sides.

MITYP = Type of rainfall/evaporation profiles assigned to side MI.

MTYPAD = Increment of MITYP for each of the NSEQ sides.

NOTE: A line with five *0's* is used to signal the end of this data set.

Subset 4: Specification of rainfall/evaporation-seepage sides

Normally, NVES records are required, one each for a variable boundary (VB) element side. However, if a group of rainfall/evaporation-seepage element sides appears in a regular pattern, automatic generation may be made. For I-th (I = 1, 2, ...,) record, it contains the following variables.

MI = Compressed VB element side number of the first element side in a sequence.

NSEQ = NSEQ subsequent VB element sides will be generated automatically.

MIAD = Increment of MI for each of the NSEQ subsequent VB element sides.

I1 = Global node number of the first node of element side MI.

I2 = Global node number of the second node of element side MI.

I3 = Global node number of the third node of element side MI.

I4 = Global node number of the fourth node of element side MI.

I1AD = Increment of I1 for each of the NSEQ subsequent VB element sides.

I2AD = Increment of I2 for each of the NSEQ subsequent VB element sides.

I3AD = Increment of I3 for each of the NSEQ subsequent VB element sides.

I4AD = Increment of I4 for each of the NSEQ subsequent VB element sides.

NOTE: A blank with eleven *0's* must be used to signal the end of this data subset.

Subset 5: Global Node Number of All Compressed Variable Boundary (VB) Nodes.

At most, NVNP records are needed for this data subset, one each for NVNP variable boundary nodes. For I^{th} (I = 1, 2, ...,) record, it contains the following five variables.

NI = Compressed VB node number of the first node in the sequence.

NSEQ = NSEQ nodes will be generated automatically.

NIAD = Increment of NI for each of the NSEQ nodes.

NODE = Global node number of node NI.

NODEAD = Increment of NODE for each of the NSEQ nodes.

NOTE: A line with five *0's* is used to signal the end of this data set.

Subset 6: Ponding Depth Allowed in Each of NVNP Variable Boundary Nodes

Normally, NVNP records are needed, one for each of the NVNP nodes. However, if a group of nodes has a regular pattern of ponding depth, automatic generation is made. For I^{th} (I = 1, 2, ...,) record, it contains the following variables.

NI = Compressed VB node number of the first node in a sequence.

NSEQ = NSEQ subsequent nodes will be generated automatically.

NIAD = Increment of NI for each of the NSEQ subsequent nodes.

HCONNI = Ponding depth of node NI, (L).

HCONAD = Increment of HCONNI for each of the NSEQ nodes, (L).

NOTE: A line with five *0's* must be used to signal the end of this data subset.

Subset 7: Minimum Pressure Head Allowed in Each NVNP Variable Boundary Nodes

This data subset is read in similar to the above data subset. For I^{th} (I = 1, 2, ...,) record, it contains the following variables.

NI = Compressed VB node number of the first node in a sequence.

NSEQ = NSEQ subsequent nodes will be generated automatically.

NIAD = Increment of NI for each of the NSEQ subsequent nodes.

HMINNI = Minimum pressure head allow for node NI, (L).

HMINAD = Increment of HMINNI for each of the NSEQ nodes, (L).

NOTE: A line with five *0's* must be used to signal the end of this data subset.

Dirichlet Boundary Conditions for Flow Simulation

This data set is needed if IMOD = 10 or IMOD = 11. Four subsets of data records are required for this data set.

Subset 1: control parameters

NDNP = Number of Dirichlet nodal points, should be .GE. 1.

NDPR = Number of total Dirichlet-head profiles, should be .GE. 1.

NDDP = Number of data points in each total head profile, should be .GE. 1.

KDAI = Is Dirichlet boundary value profile to be input analytically?

a. 0 = no.

b. 1 = yes

Subset 2: Dirichlet-head profiles

This data subset is required only if NDNP is not 0. NDPR of profiles are needed. For each profile, NDDP of the data pair (THDBF(J,I),HDBF(J,I)) are needed. If these data pairs can fit in a line, we will need NDPR lines.

THDBF(J,I) = Time of the J^{th} data point in the I^{th} profile, (T).

HDBF(J,I) = Total head of the J^{th} data point in the I^{th} profile, (L).

Subset 3: Dirichlet nodes

One record is needed for this data subset. The number of lines in this record depends on NDNP.

NPDB(I) = Global node number of the I^{th} compressed Dirichlet node.

Subset 4: boundary profile type assigned to each Dirichlet node

Normally one record per Dirichlet node, i.e., a total of NDNP records. However, if the Dirichlet nodes appear in regular pattern, automatic generation may be made. For I^{th} (I = 1, 2, ...,) record, it contains the following variables.

NI = Compressed Dirichlet node number of the first node in the sequence.

NSEQ = NSEQ subsequent Dirichlet nodes will be generated automatically.

NAD = Increment of NI for each of the NSEQ nodes.

NITYP = Type of total head profile for node NI and NSEQ subsequent nodes.

NTYPAD = Increment of NITYP for each of the NSEQ subsequent nodes.

NOTE: A line with five *0's* must be used to signal the end of this data subset.

Cauchy Boundary Conditions for Flow Simulations

This data set is needed if IMOD = 10 or IMOD = 11. Five subsets of data records are required for this data set.

Subset 1: control parameters

NCES = Number of Cauchy boundary element sides.

NCNP = Number of Cauchy nodal points.

NCPR = Number of Cauchy-flux profiles.

NCDP = Number of data points in each of the NCPR Cauchy-flux profiles.

KCAI = Is Cauchy flux profile to be input analytically?

a. 0 = no.

b. 1 = yes

Subset 2: prescribed Cauchy-flux profiles

This data subset is required only if NCES is not 0. NCPR of profiles is needed. For each profile, NCDP of the data pair (TQCBF(J,I),QCBF(J,I)) is needed. If these data pairs can fit in a line, we will need NDPR lines.

TQCBF(J,I) = Time of the J^{th} data point in the I^{th} profile, (T).

QCBF(J,I) = Normal Cauchy flux of the J^{th} data point in the I^{th} profile, (L**3/T/L**2); positive out from the region, negative into the region.

Subset 3: type of Cauchy flux profiles assigned to each of all NCES sides

At most, NCES records are needed. However, automatic generation can be made. For I^{th} (I = 1, 2, ...,) record, it contains the following variables.

MI = Compressed Cauchy side number of the first side in the sequence.

NSEQ = NSEQ sides will be generated automatically.

MIAD = Increment of MI for each of the NSEQ sides.

MITYP = Type of Cauchy flux profile assigned to side MI.

MTYPAD = Increment of MITYP for each of the NSEQ sides.

NOTE: A line with five *0's* is used to signal the end of this data set.

Record subset 4: Cauchy boundary element sides

Normally, NCES records are required, one each for a Cauchy boundary element side. However, if a group of Cauchy boundary element sides appears in a regular pattern, automatic generation may be made. For I^{th} (I = 1, 2, ...,) record, it contains the following variables.

MI = Compressed Cauchy element side number of the first element side in a sequence.

NSEQ = NSEQ subsequent Cauchy element sides will be generated automatically.

MIAD = Increment of MI for each of the NSEQ subsequent sides.

I1 = Global node number of the first node on the Cauchy element side MI.

I2 = Global node number of the second node on the Cauchy element side MI.

I3 = Global node number of the third node on the Cauchy element side MI.

I4 = Global node number of the fourth node on the Cauchy element side MI.

I1AD = Increment of I1 for each of the NSEQ subsequent element sides.

I2AD = Increment of I2 for each of the NSEQ subsequent element sides.

I3AD = Increment of I3 for each of the NSEQ subsequent element sides.

I4AD = Increment of I4 for each of the NSEQ subsequent element sides.

NOTE: A line with eleven *0's* is used to end this data set input.

Subset 5: global node number of all compressed Cauchy nodes

At most, NCNP records are needed for this data subset, one each for NCNP Cauchy nodes.

NPCB(I) = Global node number of the I^{th} compressed Cauchy nodes.

Neumann Boundary Conditions for Flow Simulations

This data set is needed if IMOD = 10 or IMOD = 11. Five subsets of data records are required for this data set.

Subset 1: control parameters

NNES = Number of Neumann boundary element sides.

NNNP = Number of Neumann nodal points.

NNPR = Number of Neumann flux profiles.

NNDP = Number of data points in each of the NNPR Neumann flux profiles.

KNAI = Is Neumann flux profile to be input analytically?

a. 0 = no.

b. 1 = yes

Subset 2: prescribed Neumann-flux profiles

This data subset is required only if NNES is not 0. NNPR of profiles is needed. For each profile, NNDP of the data pair (TQNBF(J,I),QNBF(J,I)) is needed. If these data pairs can fit in a line, we will need NDPR lines.

TQNBF(J,I) = Time of the J^{th} data point in the I^{th} profile, (T).

QNBF(J,I) = Normal Neumann flux of the J^{th} data point in the I^{th} profile, (L**3/T/L**2); positive out from the region, negative into the region.

Subset 3: type of Neumann flux profiles assigned to each of all NNES sides

At most, NNES records are needed. However, automatic generation can be made. For I^{th} (I = 1, 2, ...,) record, it contains the following variables.

MI = Compressed Neumann side number of the first side in the sequence.

NSEQ = NSEQ sides will be generated automatically.

MIAD = Increment of MI for each of the NSEQ sides.

MITYP = Type of Neumann flux profile assigned to side MI.

MTYPAD = Increment of MITYP for each of the NSEQ sides.

NOTE: A line with five *0's* is used to signal the end of this data set.

Subset 4: Neumann boundary element sides

Normally, NNES records are required, one each for a Neumann boundary element side. However, if a group of Neumann boundary element sides appears in a regular pattern, automatic generation may be made. For I^{th} (I = 1, 2, ...,) record, it contains the following variables.

MI = Compressed Neumann side number of the first side in sequence.

NSEQ = NSEQ subsequent Neumann sides will be generated automatically.

MIAD = Increment of MI for each of the NSEQ subsequent sides.

I1 = Global node number of the first node on the Neumann element side MI.

I2 = Global node number of the second node on Neumann element side MI.

I3 = Global node number of the third node on the Neumann element side MI.

I4 = Global node number of the fourth node on the Neumann element side MI.

I1AD = Increment of I1 for each of the NSEQ subsequent element sides.

I2AD = Increment of I2 for each of the NSEQ subsequent element sides.

I3AD = Increment of I3 for each of the NSEQ subsequent element sides.

I4AD = Increment of I4 for each of the NSEQ subsequent element sides.

NOTE: A line with eleven *0's* is used to end this data set input.

Subset 5: global node number of all compressed Neumann nodes

At most, NNNP records are needed for this data subset, one each for NNNP Neumann nodes.

NPNB(I) = Global node number of the I^{th} compressed Neumann nodes.

Run-In/Flow-Out (Variable) Boundary Conditions for Transport Simulations

This data set is needed if IMOD = 1 or IMOD = 11. Five subsets of data records are required for this data set.

Subset 1: control parameters

NVES = Number of variable boundary element sides.

NVNP = Number of variable boundary nodal points.

NRPR = Number of incoming fluid concentration profiles to be applied to variable boundary element sides.

NRDP = Number of data points in each of the NRPR profiles.

KRAI = Is incoming concentration profile to be input analytically?

a. 0 = no.

b. 1 = yes.

Subset 2: variable boundary flux profile

NRPR records are needed. Each record contains NRDP data points and is FREE-FORMATTED. Each data point has two numbers representing the time and run-in/flow-out concentrations, respectively, as follows:

TCRSF(J,I) = Time of the J^{th} data point on the I^{th} run-in concentration profile, (T).

CRSF(J,I) = Concentration of the J^{th} data point on the I^{th} profile, (M/L**3).

Subset 3: Run-in concentration type assigned to each of all NVES sides

Usually one record per variable element side. However, automatic generation can be made. Each record contains five variables and is FREE-FORMATTED.

MI = Compressed VB element side of the first side in a sequence.

NSEQ = NSEQ subsequent sides will be generated automatically.

MIAD = Increment of MI for each of NSEQ subsequent sides.

MITYP = Type of concentration profile assigned to side MI.

MTYPAD = Increment of MITYP for each of the NSEQ subsequent sides.

NOTE: A record with five *0's* must be used to signal the end of this data subset.

Subset 4: Specification of run-in boundary element sides

Normally, NVES records are required, one each for a VB element side. However, if a group of VB element sides appears in a regular pattern, automatic generation may be made. Each record contains 11 variables and is FREE-FORMATTED.

MI = Compressed VB element side number of the first side in a sequence.

NSEQ = NSEQ subsequent VB element sides will be generated automatically.

MIAD = Increment of MI for each of the NSEQ subsequent VB element sides.

I1 = Global node number of the first node of element side MI.

I2 = Global node number of the second node of element side MI.

I3 = Global node number of the third node of element side MI.

I4 = Global node number of the fourth node of element side MI

I1AD = Increment of I1 for each of the NSEQ subsequent element sides.

I2AD = Increment of I2 for each of the NSEQ subsequent element sides.

I3AD = Increment of I3 for each of the NSEQ subsequent element sides.

I4AD = Increment of I4 for each of the NSEQ subsequent element sides.

NOTE: A record with eleven *0's* is used to signal the end of this data subset.

Subset 5: global nodal number of all run-in/flow-out boundary nodes

Usually NVNP records are needed for this data subset. However, automatic generation can be made. Each record contains five variables and is FREE-FORMATTED.

NI = Compressed VB node number of the first node in a sequence.

NSEQ = NSEQ subsequent nodes will be generated automatically.

NIAD = Increment for NI for each of the NSEQ nodes.

NODE = Global nodal number of the node NI.

NODEAD = Increment of NODE for each of the NSEQ subsequent nodes.

NOTE: A record with five *0's* is used to signal end of this data subset.

Dirichlet Boundary Conditions for Transport Simulations

This data set is needed if IMOD = 1 or IMOD = 11. Four subsets of data records are required for this data set.

Subset 1: control parameters

NDNP = Number of Dirichlet nodes, should be .GE. 1.

NDPR = Number of Dirichlet profiles, should be .GE. 1.

NDDP = Number of data points in each profile, should be .GE. 2.

KDAI = Is Dirichlet boundary value profile to be input analytically?

a. 0 = no.

b. 1 = yes.

Subset 2: Dirichlet concentration profiles

NDPR records are needed. Each record contains NDDP data points and is FREE-FORMATTED. Each data point has two numbers representing the time and Dirichlet concentrations, respectively, as follows:

TCDBF(J,I) = Time of J^{th} data point in I^{th} Dirichlet-concentration profile, (T).

CDBF(J,I) = Concentration of J^{th} data point in I^{th} Dirichlet concentration profile, (M/L**3).

Subset 3: global node number of compressed Dirichlet nodes

One record is needed for this data subset, which contains NDNP variables and is FREE-FORMATTED.

NPDB(I) = Global node number of the I^{th} compressed Dirichlet node.

Subset 4: Dirichlet concentration types assigned to Dirichlet nodes

Normally one record per Dirichlet node, i.e., a total of NDNP records, is needed. However, if the Dirichlet nodes appear in regular pattern, automatic generation may be made. Each record contains five variables and is FREE-FORMATTED.

NI = Compressed Dirichlet node number of the first node in the sequence.

NSEQ = NSEQ nodes will contain the Dirichlet concentration types that will be automatically generated.

NIAD = Increment of NI for each of the NSEQ nodes.

NITYP = Dirichlet concentration type in node NI.

NTYPAD = Increment of NITYP for each of the NSEQ subsequent nodes.

NOTE: A record with five *0's* must be used to signal the end of this data set.

Cauchy Boundary Conditions for Transport Simulation

This data set is needed if IMOD = 1 or IMOD = 11. Five subsets of data records are required for this data set.

Subset 1: control parameters

NCES = Number of Cauchy element sides.

NCNP = Number of Cauchy nodal points.

NCPR = Number of Cauchy-flux profiles.

NCDP = Number of data points on each Cauchy-flux profile.

KCAI = Is Cauchy flux profile to be input analytically?

a. 0 = no.

b. 1 = yes.

Subset 2: Cauchy flux profiles

NCPR records are needed. Each record contains NCDP data points and is FREE-FORMATTED. Each data point has two numbers representing the time and Cauchy flux, respectively, as follows:

TQCBF(J,I) = Time of the J^{th} data point in the I^{th} Cauchy flux profile, (T).

QCBF(J,I) = Value of Cauchy flux of the J^{th} data point in the I^{th} Cauchy flux profile, (M/T/L**2).

Subset 3: Cauchy flux type assigned to each of all NCES sides

Usually one record per Cauchy element side. However, automatic generation can be made. Each record contains five variables and is FREE-FORMATTED.

MI = Compressed Cauchy boundary element side of the first side in a sequence.

NSEQ = NSEQ subsequent sides will be generated automatically.

MIAD = Increment of MI for each of NSEQ subsequent sides.

MITYP = Type of Cauchy flux profile assigned to side MI.

MTYPAD = Increment of MITYP for each of the NSEQ subsequent sides.

NOTE: A record with five *0's* must be used to signal the end of this data subset.

Record subset 4: specification of Cauchy boundary element sides

Normally, NCES records are required, one each for a Cauchy boundary element side. However, if a group of Cauchy element sides appears in a regular pattern, automatic generation may be made. Each record contains 11 variables and is FREE-FORMATTED.

MI = Compressed Cauchy boundary element side number of the first element side in a sequence.

NSEQ = NSEQ subsequent Cauchy boundary element sides will be generated automatically.

MIAD = Increment of MI for each of the NSEQ subsequent Cauchy boundary element sides.

I1 = Global node number of the first node of element side MI.

I2 = Global node number of the second node of element side MI.

I3 = Global node number of the third node of element side MI.

I4 = Global node number of the fourth node of element side MI.

I1AD = Increment of I1 for each of the NSEQ subsequent element sides.

I2AD = Increment of I2 for each of the NSEQ subsequent element sides.

I3AD = Increment of I3 for each of the NSEQ subsequent element sides.

I4AD = Increment of I4 for each of the NSEQ subsequent element sides.

NOTE: A record with eleven *0's* is used to signal the end of this data subset.

Subset 5: global nodal number of all Cauchy boundary nodes

Usually NCNP records are needed for this data subset. However, automatic generation can be made. Each record contains five variables and is FREE-FORMATTED.

NI = Compressed Cauchy boundary node number of the first node in a sequence.

NSEQ = NSEQ subsequent nodes will be generated automatically.

NIAD = Increment for NI for each of the NSEQ nodes.

NODE = Global nodal number of the node NI.

NODEAD = Increment of the global nodal number for each of the NSEQ subsequent nodes.

NOTE: A record with five *0's* is used to signal end of this data subset.

Neumann Boundary Conditions for Transport Simulations

This data set is needed if IMOD = 1 or IMOD = 11. Five subsets of data records are required for this data set.

Subset 1: control parameters

NNES = Number of Neumann element sides.

NNNP = Number of Neumann nodal points.

NNPR = Number of Neumann flux profiles.

NNDP = Number of data points on each Neumann flux profile.

KSAI = Is Neumann flux profile to be input analytically?

a. 0 = no.

b. 1 = yes.

Subset 2: Neumann flux profiles

NNPR records are needed. Each record contains NNDP data points and is FREE-FORMATTED. Each data point has two numbers representing the time and Neumann flux, respectively, as follows:

TQNBF(J,I) = Time of the J^{th} data point in the I^{th} Neumann flux profile, (T).

QNBF(J,I) = Value of Neumann flux of the J^{th} data point in the I^{th} Neumann flux profile, (M/T/L**2).

Subset 3: Neumann flux type assigned to each of all NNES sides

Usually one record per Neumann element side. However, automatic generation can be made. Each record contains five variables and is FREE-FORMATTED.

MI = Compressed Neumann boundary element side of the first side in a sequence.

NSEQ = NSEQ subsequent sides will be generated automatically.

MIAD = Increment of MI for each of NSEQ subsequent sides.

MITYP = Type of Neumann flux profile assigned to side MI.

MTYPAD = Increment of MITYP for each of the NSEQ subsequent sides.

NOTE: A record with five *0's* must be used to signal the end of this data subset.

Subset 4: specification of Neumann boundary element sides

Normally, NNES records are required, one each for a Neumann boundary element side. However, if a group of Neumann element sides appears in a regular pattern, automatic generation may be made. Each record contains 11 variables and is FREE-FORMATTED.

MI = Compressed Neumann boundary element side number of the first element side in a sequence.

NSEQ = NSEQ subsequent Neumann boundary element sides will be generated automatically.

MIAD = Increment of MI for each of the NSEQ subsequent sides.

I1 = Global node number of the first node of element side MI.

I2 = Global node number of the second node of element side MI.

I3 = Global node number of the third node of element side MI.

I4 = Global node number of the fourth node of element side MI

I1AD = Increment of I1 for each of the NSEQ subsequent element sides.

I2AD = Increment of I2 for each of the NSEQ subsequent element sides.

I3AD = Increment of I3 for each of the NSEQ subsequent element sides.

I4AD = Increment of I4 for each of the NSEQ subsequent element sides.

NOTE: A record with eleven *0's* is used to signal the end of this data subset.

Subset 5: global nodal number of all Neumann boundary nodes

Usually NNNP records are needed for this data subset. However, automatic generation can be made. Each record contains five variables and is FREE-FORMATTED.

NI = Compressed Neumann boundary node number of the first node in a sequence.

NSEQ = NSEQ subsequent nodes will be generated automatically.

NIAD = Increment for NI for each of the NSEQ nodes.

NODE = Global nodal number of the node NI.

NODEAD = Increment of the global nodal number for each of the NSEQ subsequent nodes.

NOTE: A record with five *0's* is used to signal end of this data subset.

Hydrological Variables

This data set is needed if and only if KVI .LE. 0. When KVI .LE. 0, two groups of data are needed, one group for the velocity field and the other group for the moisture content.

Subset 1: velocity field

Usually NNP records are needed. However, if velocity appears in regular pattern, automatic generation can be made. Each record contains nine variables and is FREE-FORMATTED.

NI = Node number of the first node in a sequence.

NSEQ = NSEQ subsequent nodes will be automatically generated.

NIAD = Increment of node number in each of the NSEQ subsequent nodes.

VXNI = x-velocity component at node NI, (L/T).

VYNI = y-velocity component at node NI, (L/T).

VZNI = z-velocity component at node NI, (L/T).

VXAD = Increment of VXNI for each of the NSEQ subsequent nodes, (L/T).

VYAD = Increment of VYNI for each of the NSEQ subsequent nodes, (L/T).

VZAD = Increment of VZNI for each of the NSEQ subsequent nodes, (L/T).

NOTE: A record with nine *0's* is used to signal the end of this data subset.

Subset 2: moisture content field

Usually, NEL records are needed. However, if moisture content appears in regular pattern, automatic generation can be made. Each record contains five variables and is FREE-FORMATTED.

MI = Element number of the first element in a sequence.

NSEQ = NSEQ subsequent elements will be automatically generated.

MIAD = Increment of MI for each of NSEQ subsequent elements.

THNI = Moisture content of element NI, (Decimal point).

THNIAD = Increment of THNI for NSEQ subsequent elements, (Decimal point).

NOTE: A record with five *0's* is used to signal the end of this data subset.

End of Job

If another problem is to be run, then input begins again with input data set 1. If termination of the job is desired, a blank card must be inserted at the end of the data set.

Appendix B
Mathematical Formulation

Governing Equations for Flow

From the notion for continuity of fluid, continuity of solid, consolidation of the media, and the equation of state (Yeh 1992),[1] we obtain the starting equation for this derivation:

$$\nabla \cdot \left[\frac{\rho \mathbf{k}}{\mu} (\nabla p + \rho g \nabla z) \right] - \nabla (\rho n_e S V_s) + \rho^* q = \frac{\partial (n_e S \rho)}{\partial t} \qquad \text{(B.1.1)}$$

where ρ is the fluid density (M/L^3), k is the intrinsic permeability tensor of the media (L^2), μ is the dynamic viscosity of the fluid (M/L/T), p is the fluid pressure [$(ML/T^2)/L^2$], g is the acceleration of gravity (L/T^2), z is the potential head (L), n_e is the effective porosity (L^3/L^3), S is the degree of saturation (dimensionless), V_s is the velocity of the deformable surface due to consolidation (L/T), ρ^* is the density of the injected fluid (M/L^3), q is the internal source/sink [$(L^3/T)/LL^3$], and t is the time (T).

Expanding the right-hand side of Equation B.1.1:

$$\frac{\partial (n_e S \rho)}{\partial t} = S n_e \frac{\partial \rho}{\partial t} + \rho S \frac{\partial n_e}{\partial t} + n_e \rho \frac{\partial S}{\partial t} \qquad \text{(B.1.2)}$$

Expanding Equation B.1.2 by the chain rule:

[1] References cited in this Appendix are included in the References at the end of the main text.

$$\frac{\partial(n_e S\rho)}{\partial t} = Sn_e \frac{\partial \rho}{\partial p}\frac{\partial p}{\partial t} + Sn_e \frac{\partial \rho}{\partial C}\frac{\partial C}{\partial t} + \rho S \frac{\partial n_e}{\partial t} + n_e \rho \frac{\partial S}{\partial t} \qquad \text{(B.1.3)}$$

where C is chemical concentration (M/L^3). Rearranging Equation B.1.3, we obtain:

$$\frac{\partial(n_e S\rho)}{\partial t} = Sn_e \frac{\partial \rho}{\partial p}\frac{\partial p}{\partial t} + \rho S \frac{\partial n_e}{\partial t} + Sn_e \frac{\partial \rho}{\partial C}\frac{\partial C}{\partial t} + n_e \rho \frac{\partial S}{\partial t} \qquad \text{(B.1.4)}$$

where the first and second terms represent the storage term, the third term is the density-concentration coupling term, and the fourth term is the unsaturated term. Substituting Equation B.1.4 into Equation B.1.1:

$$\nabla \cdot \left[\frac{\rho \mathbf{k}}{\mu} \cdot (\nabla p + \rho g \nabla z) \right] + \rho^* q =$$

$$Sn_e \frac{\partial \rho}{\partial p}\frac{\partial p}{\partial t} + Sn_e \frac{\partial \rho}{\partial C}\frac{\partial C}{\partial t} + n_e \rho \frac{\partial S}{\partial t} \qquad \text{(B.1.5)}$$

$$+ \rho S \frac{\partial n_e}{\partial t} + S\rho \nabla \cdot n_e \mathbf{V}_s + n_e \mathbf{V}_s \cdot \nabla(S\rho)$$

Making the first approximation by neglecting the second-order term:

$$n_e V_s \cdot \nabla(S\rho) \dashrightarrow 0 \qquad \text{(B.1.6)}$$

we have:

$$\nabla \cdot [\frac{\rho k}{\mu} \cdot (\nabla p + \rho g \nabla z)] + \rho^* q =$$

$$Sn_e \frac{\partial \rho}{\partial p}\frac{\partial p}{\partial t} + Sn_e \frac{\partial \rho}{\partial C}\frac{\partial C}{\partial t} + n_e \rho \frac{\partial S}{\partial t} \qquad \text{(B.1.7)}$$

$$+ \rho S \frac{\partial n_e}{\partial t} + S\rho \nabla n_e \mathbf{V}_s$$

Defining the compressibility of the fluid as:

$$\beta = \frac{1}{\rho}\frac{\partial \rho}{\partial p} \tag{B.1.8}$$

where β is the compressibility of the fluid (LT^2/M). Also defining the moisture content as:

$$\theta = Sn_e \tag{B.1.9}$$

where θ is the moisture content (dimensionless). We may substitute Equations B.1.8 and B.1.9 into Equation B.1.7 and rewrite it to obtain:

$$\begin{aligned} \nabla\cdot\left[\frac{\rho k}{\mu}\cdot(\nabla p + \rho g \nabla z)\right] + \rho^* q \quad = \\ \theta\beta\rho\frac{\partial p}{\partial t} + \theta\frac{\partial \rho}{\partial C}\frac{\partial C}{\partial t} + n_e\rho\frac{\partial S}{\partial t} \\ + \rho S\left[\frac{\partial n_e}{\partial t} + \nabla\cdot(n_e V_s)\right] \end{aligned} \tag{B.1.10}$$

Remembering that the continuity statement of incompressible solids is a compressible skeleton (Yeh 1992):

$$\frac{\partial(1-n_e)}{\partial t} + \nabla\cdot(1-n_e)V_s = 0 \tag{B.1.11}$$

Rearranging Equation B.1.11 in the following form:

$$\frac{\partial n_e}{\partial t} + \nabla\cdot n_e V_s = \nabla\cdot V_s \tag{B.1.12}$$

Substituting Equation B.1.12 into Equation B.1.10, we obtain:

$$\nabla\cdot\left[\frac{\rho k}{\mu}(\nabla p + \rho g\nabla z)\right] + \rho^* q \quad =$$
$$\theta\beta\frac{\partial p}{\partial t} + \theta\frac{\partial \rho}{\partial C}\frac{\partial C}{\partial t} + n_e\rho\frac{\partial S}{\partial t} + \rho S\nabla\cdot V_s \tag{B.1.13}$$

Recalling that the flux of solid velocity is the divergence of V_s (Yeh 1992):

$$\nabla\cdot V_s = \alpha\frac{\partial p}{\partial t} \tag{B.1.14}$$

where α is the coefficient of consolidation of the media (LT^2/M). Substituting Equation B.1.14 into Equation B.1.13 and rewriting:

$$\nabla\cdot\left[\frac{\rho k}{\mu}(\nabla p + \rho g\nabla z)\right] + \rho^* q \quad =$$
$$\rho(\theta\beta + S\alpha)\frac{\partial p}{\partial t} + \theta\frac{\partial \rho}{\partial C}\frac{\partial C}{\partial t} + n_e\rho\frac{\partial S}{\partial t} \tag{B.1.15}$$

Remembering Equation B.1.9 and substituting:

$$\nabla\cdot\left[\frac{\rho k}{\mu}(\nabla p + \rho g\nabla z)\right] + \rho^* q \quad =$$
$$\rho\left[\theta\beta + \frac{\theta}{n_e}\alpha\right]\frac{\partial p}{\partial t} + \theta\frac{\partial \rho}{\partial C}\frac{\partial C}{\partial t} + n_e\rho\frac{\partial S}{\partial t} \tag{B.1.16}$$

Recalling the relationship between pressure head ψ and pressure, as well as rewriting:

$$dp = \rho g d\psi = \rho g\frac{d\psi}{d\theta}d\theta \tag{B.1.17}$$

Differentiating Equation B.1.9 and substituting it into Equation B.1.17, we obtain:

$$dp = \rho g \frac{d\psi}{d\theta}(n_e dS + S dn_e) \tag{B.1.18}$$

Making the second approximation assuming that dn_e is much smaller than dS:

$$S dn_e \dashrightarrow 0 \tag{B.1.19}$$

we have:

$$dp = \rho g \frac{d\psi}{d\theta}(n_e dS) \tag{B.1.20}$$

Rearranging Equation B.1.20, we get:

$$\frac{1}{g}\frac{d\theta}{d\psi}\frac{\partial p}{\partial t} = \rho n_e \frac{\partial S}{\partial t} \tag{B.1.21}$$

Substituting Equation B.1.21 into Equation B.1.16:

$$\nabla \cdot \left[\frac{\rho \mathbf{k}}{\mu}(\nabla p + \rho g \nabla z)\right] + \rho^* q = \rho\left[\theta\beta + \frac{\theta}{n_e}\alpha\right]\frac{\partial p}{\partial t} + \theta\frac{\partial \rho}{\partial C}\frac{\partial C}{\partial t} + \frac{1}{g}\frac{d\theta}{d\psi}\frac{\partial p}{\partial t} \tag{B.1.22}$$

Next, we need to define the reference pressure head as:

$$h = \frac{p}{\rho_0 g} \tag{B.1.23}$$

where h is the reference pressure head (L) and ρ_o is the reference water density (M/L^3). From Equations B.1.17 and B.1.23, the relationship between the pressure head and referenced pressure head is

$$\rho_o dh = \rho d\psi \tag{B.1.24}$$

where z is the elevation.

Substituting Equations B.1.23 and B.1.24 into Equation B.1.22, we obtain:

$$\nabla \cdot \left[\frac{\rho \mathbf{k}}{\mu} \cdot (\rho_0 g \nabla h + \rho g \nabla z) \right] + \rho^* q = \rho(\theta\beta + \frac{\theta}{n_e}\alpha)\rho_0 g \frac{\partial h}{\partial t} + \theta \frac{\partial \rho}{\partial C}\frac{\partial C}{\partial t} + \frac{1}{g}\frac{\rho}{\rho_o}\frac{d\theta}{dh}\rho_0 g \frac{\partial h}{\partial t} \tag{B.1.25}$$

Dividing Equation B.1.25 by ρ_0 and rearranging, we get:

$$\nabla \cdot \left[\frac{\rho g \mathbf{k}}{\mu} \cdot \left[\nabla h + \frac{\rho}{\rho_0} \nabla z \right] \right] + \frac{\rho^*}{\rho_0} q = \frac{\rho}{\rho_0} \left[\theta g \rho_0 \beta + \frac{\theta}{n_e} g \rho_0 \alpha \right] \frac{\partial h}{\partial t} + \frac{\theta}{\rho_0}\frac{\partial \rho}{\partial C}\frac{\partial C}{\partial t} + \frac{\rho}{\rho_o}\frac{d\theta}{dh}\frac{\partial h}{\partial t} \tag{B.1.26}$$

Defining the modified compressibilities of the media and water as

$$\alpha' = \alpha \rho_o g \tag{B.1.27}$$

$$\beta' = \beta \rho_o g \tag{B.1.28}$$

where α' is the modified compressibility of the media (1/L) and β' is the modified compressibility of the water (1/L). Substituting Equations B.1.27 and B.1.28 into Equation B.1.26 and rearranging:

$$\nabla \cdot \left[\frac{\rho g k}{\mu} \cdot \left[\nabla h + \frac{\rho}{\rho_o} \nabla z \right] \right] + \frac{\rho^*}{\rho_o} q = \frac{\rho}{\rho_o} \left[\alpha' \frac{\theta}{n_e} + \beta' \theta + \frac{d\theta}{dh} \right] \frac{\partial h}{\partial t} + \frac{\theta}{\rho_o} \frac{\partial \rho}{\partial C} \frac{\partial C}{\partial t} \quad \text{(B.1.29)}$$

Defining the storage coefficient as:

$$F = \alpha' \frac{\theta}{n_e} + \beta' \theta + \frac{d\theta}{dh} \quad \text{(B.1.30)}$$

where F is the storage coefficient. Substituting Equation B.1.30 into Equation B.1.29 and following Frind (1982b) by neglecting the second term on the right-hand side of Equation B.1.29, we get:

$$\nabla \cdot \left[\frac{\rho g k}{\mu} \cdot \left[\nabla h + \frac{\rho}{\rho_o} \nabla z \right] \right] + \frac{\rho^*}{\rho_o} q = \frac{\rho}{\rho_o} F \frac{\partial h}{\partial t} \quad \text{(B.1.31)}$$

Defining the relation:

$$K = \frac{\rho g k}{\mu} \quad \text{(B.1.32)}$$

where K is the hydraulic conductivity tensor. Substituting Equation B.1.32 into Equation B.1.31 and rearranging, we get the density-dependent flow equation:

$$\frac{\rho}{\rho_o} F \frac{\partial h}{\partial t} = \nabla \cdot \left[K \cdot \left[\nabla h + \frac{\rho}{\rho_o} \nabla z \right] \right] + \frac{\rho^*}{\rho_o} q \quad \text{(B.1.33)}$$

From the concept of the motion of fluid (Darcy's law) :

$$V = -\frac{1}{\rho}\frac{\rho \mathbf{k}}{\mu}(\nabla p + \rho g \nabla z) \tag{B.1.34}$$

where V is the Darcy flux (L/T). Recalling Equation B.1.23 and substituting into Equation B.1.34, we obtain:

$$V = -\frac{1}{\rho}\frac{\rho \mathbf{k}}{\mu}(\rho_o g \nabla h + \rho g \nabla z) \tag{B.1.35}$$

Rearranging Equation B.1.35:

$$V = -\frac{\rho g \mathbf{k}}{\mu} \cdot \left[\frac{\rho_o}{\rho}\nabla h + \nabla z\right] \tag{B.1.36}$$

and substituting Equation B.1.32 into Equation B.1.36, we get the Darcy flux equation for density-dependent flow in its final form:

$$V = -\mathbf{K} \cdot \left[\frac{\rho_0}{\rho}\nabla h + \nabla z\right] \tag{B.1.37}$$

The density and dynamic viscosity are functions of chemical concentration and are assumed to take the following form:

$$\frac{\rho}{\rho_0} = a_1 + a_2 C + a_3 C^2 + a_4 C^3 \tag{B.1.38}$$

and

$$\frac{\mu}{\mu_0} = a_5 + a_6 C + a_7 C^2 + a_8 C^3 \tag{B.1.39}$$

where C is the chemical concentration (M/L^3) and a_1, a_2, ..., a_7, a_8 are the parameters (L^3/M) that are used to describe the concentration dependence of water density and dynamic viscosity.

In the specific case of seawater intrusion, the constitutive relation between fluid density and concentration takes the following form:

$$\rho = \rho_0(1 + \epsilon c) \tag{B.1.40}$$

where c is the dimensionless chemical concentration (actually divided by maximum) and ϵ is the dimensionless density reference ratio defined as:

$$\epsilon = \frac{\rho_{max}}{\rho_0} - 1 \tag{B.1.41}$$

where ρ_{max} is the maximum density of the fluid (M/L^3) and ρ_o is the reference (freshwater) density (M/L^3). It should be noted that Equation 3.9 implicitly assumes that the fluid is incompressible and under isothermal conditions (Galeati, Gambolati, and Neuman 1992).

The initial conditions for the flow equations are stated as:

$$h = h_i(x,z) \quad in \ R \tag{B.1.42}$$

where R is the region of interest and h_i is the prescribed initial condition for hydraulic head. H_i can either be obtained by solving the steady-state version of Equation 3.1 or alternatively by defining through field measurements.

The specification of boundary conditions is probably the most critical and complex chore in flow modelling. As explained by Yeh (1987), the boundary conditions of the region of interest can be examined from a dynamic, physical, or mathematical point of view. From a dynamic standpoint, a boundary segment can be either considered as impermeable or flow-through. On the other hand, from a physical point of view, such a segment could be classified as a soil-soil interface, soil-air interface, or soil-water interface. Lastly, from a mathematical point of view, the boundary segment can be classified as one of four types of boundary conditions, namely as (a) Dirichlet, (b) Neumann, (c) Cauchy, or (d) variable boundary conditions. In addition, a good numerical model must be able to handle these boundary conditions when they vary on the boundary and are either abruptly or gradually time-dependent.

The Dirichlet boundary condition is usually applied to soil-water interfaces, such as streams, artificial impoundments, and coastal lines, and involves prescribing the functional value on the boundary. The Neumann boundary condition, on the other hand, involves prescribing the gradient of the function on the boundary and does not occur very often in real-world problems. This condition, however, can be encountered at the base of the media where natural

drainage occurs. The third type of boundary condition, the Cauchy boundary condition, involves prescribing the total normal flux due to the gradient on the boundary. Usually surface water bodies with known infiltration rates through the layers of the bottom of their sediments or liners into the subsurface media are administered by this boundary condition. If there exists a soil-air interface in the region of interest, a variable boundary condition is employed. In such a case, either Dirichlet or Cauchy boundary conditions dominate, mainly depending on the potential evaporation, the conductivity of the media, and the availability of water such as rainfall (Yeh 1987).

From the above discussion, four types of boundary conditions can be specified for the flow equations depending on the physical location of the boundaries. These boundary conditions are stated as:

a. Dirichlet boundary conditions:

$$h = h_d(x_b,y_b,z_b,t) \quad on \quad B_d \tag{B.1.43}$$

b. Neumann boundary conditions:

$$-\boldsymbol{n}\cdot\mathbf{K}\left[\frac{\rho_o}{\rho}\cdot\nabla h\right] = q_n(x_b,y_b,z_b,t) \quad \text{on} \quad B_n \tag{B.1.44}$$

c. Cauchy boundary conditions:

$$-\boldsymbol{n}\cdot\mathbf{K}\cdot\left[\frac{\rho_o}{\rho}\nabla h + \nabla z\right] = q_c(x_b,y_b,z_b,t) \quad \text{on} \quad B_c \tag{B.1.45}$$

d. Variable boundary conditions during precipitation period:

$$h = h_p(x_b,y_b,z_b,t) \quad on \quad B_v \tag{B.1.46a}$$

or

$$-n\cdot K\cdot\left[\frac{\rho_o}{\rho}\nabla h+\nabla z\right]=q_p(x_b,y_b,z_b,t)\ \text{ on }\ B_v \tag{B.1.46b}$$

e. Variable boundary conditions during nonprecipitation period:

$$h=h_p(x_b,y_b,z_b,t)\quad on\quad B_v \tag{B.1.46c}$$

or

$$h=h_m(x_b,y_b,z_b,t)\quad on\quad B_v \tag{B.1.46d}$$

or

$$-n\cdot K\cdot\left[\frac{\rho_o}{\rho}\nabla h+\nabla z\right]=q_e(x_b,y_b,z_b,t)\ \text{ on }\ B_v \tag{B.1.46e}$$

where n is the outward unit vector normal to the boundary; (x_b,y_b,z_b) is the spatial coordinate on the boundary; h_d, q_n, and q_c are the Dirichlet functional value, Neumann flux, and Cauchy flux, respectively; B_d, B_n, B_c, and B_v are the Dirichlet, Neumann, Cauchy, and variable boundaries, respectively; h_p and q_p are the allowed ponding depth and the throughfall of precipitation, respectively, on the variable boundary; h_m is the allowed minimum pressure on the variable boundary; and q_e is the allowed maximum evaporation rate (= potential evaporation) on the variable boundary. Note that only one of Equations B.1.46a through B.1.46e is utilized at any point on the variable boundary at any time.

Governing Equations for Transport

The governing equations for material transport through groundwater systems are derived based on the laws of continuity of mass and flux. The major processes that are included are advection, dispersion/diffusion, decay, adsorption, biodegradation through both liquid and solid phases, the compressibility of media, as well as source(s)/sink(s). Let C be the dissolved concentration and S be the adsorbed concentration. The equation governing the spatial-temporal distribution of dissolved concentrations can be obtained by applying this principle of mass balance in integral form as follows:

$$\frac{D}{Dt}\int_{\nu}(\theta C+\rho_b S)\,d\nu = -\int_{\Gamma} n\cdot(\theta C)V_{fs}\,d\Gamma$$

$$-\int_{\Gamma}\mathbf{n}\cdot\mathbf{J}\,d\Gamma-\int_{\nu}(\theta K_w C+\rho_b K_s S)\,d\nu \qquad \text{(B.2.1)}$$

$$-\int_{\nu}\lambda(\theta K_w C+\rho_b K_s S)\,d\nu+\int_{\nu} m\,d\nu$$

where

ν = material volume containing constant amount of media (L^3)

C = dissolved concentration (M/L^3)

S = adsorbed concentration (M/M)

ρ_b = bulk density of the medium (M/L^3)

Γ = surface enclosing the material volume ν (L^2)

$\mathbf{n}$ = outward unit vector normal to the surface Γ

V_{fs} = fluid velocity relative to the solid (L/T)

$\mathbf{J}$ = surface flux with respect to fluid velocity V_{fs} [(M/T)/L^2]

K_w = first-order biodegradation rate constant through dissolved phase (1/T)

K_s = first-order biodegradation rate constant through adsorbed phase (1/T)

λ = decay constant (1/T)

m = external source/sink rate per medium volume [(M/L^3)/T]

By the Reynolds transport theorem (Owczarek 1964), Equation B.2.1 can be written as

$$\int_v \frac{\partial(\theta C+\rho_b S)}{\partial t}\, dv + \int_\Gamma n\cdot(\theta C+\rho_b S)V_s\, d\Gamma + \int_\Gamma n\cdot(\theta C V_f)\, d\Gamma =$$

$$-\int_\Gamma n\cdot J\, d\Gamma - \int_v (\theta K_w C+\rho_b K_s S)\, dv - \int_v \lambda(\theta C+\rho_b S)\, dv \quad \text{(B.2.2)}$$

$$+ \int_v m\, dv$$

where V_s is the velocity of the solid. The fluid velocity relative to the solid V_{fs} and Darcy velocity V are related to each other by

$$V = \theta V_{fs} \quad \text{(B.2.3)}$$

Applying the Gaussian divergence theorem to Equation B.2.2 and using the fact that v is arbitrary, one can obtain the following continuity in differential form:

$$\frac{\partial(\theta C+\rho_s)}{\partial t} + \nabla\cdot\left[V_s(\theta C+\rho_b S)\right] + \nabla\cdot(VC) = -\nabla\cdot J$$
$$- (\theta K_w C+\rho_b K_s S) - \lambda(\theta C+\rho_b S) + m \quad \text{(B.2.4)}$$

The second term of Equation B.2.4 can be expressed as

$$\nabla\left[V_s(\theta C+\rho_b S)\right] = \nabla(\theta C+\rho_b S)\cdot V_s + (\theta C+\rho_b S)\nabla\cdot V_s \quad \text{(B.2.5)}$$

The first term on the right-hand side of Equation B.2.5 is the product of two small vectors and will be neglected. If all the displacement of the medium is assumed to be vertical (e.g., vertical consolidation), the solid velocity becomes

$$\nabla\cdot V_s = \alpha\frac{\partial p}{\partial p} \quad \text{(B.2.6)}$$

The surface flux J has been postulated to be proportional to the gradient of C (Nguyen et al. 1982) as

$$J = -\theta \mathbf{D} \cdot \nabla C \tag{B.2.7}$$

$$\theta D = a_T |V| \delta + (a_L - a_T) VV/|V| + a_m \theta \tau \delta \tag{B.2.8}$$

where

a_T = transverse diffusivity (L)

δ = Kronecker delta tensor

$|V|$ = the magnitude of the Darcy velocity V (L/T)

a_L = longitudinal diffusivity (L)

a_m = molecular diffusion coefficient (L^2/T)

τ = tortuosity

Substitution of Equations B.2.5 through B.2.8 into Equation B.2.4 yields

$$\frac{\partial(\theta C + \rho_b S)}{\partial t} + \nabla \cdot (VC) - \nabla \cdot (\theta \mathbf{D} \cdot \nabla C) = -\left[\alpha \frac{\partial p}{\partial t} + \lambda\right] (\theta C + \rho_b S) - (\theta K_w C + \rho_b K_s S) + m \tag{B.2.9}$$

Equation B.2.9 is simply the statement of mass balance over a differential volume. The first term represents the rate of mass accumulation, the second term represents the net rate of mass flux due to advection, the third term is the net mass flux due to dispersion and diffusion, the fourth term is the rate of mass production and reduction due to consolidation and radioactive decay, the fifth term is the degradation rate due to microbial transformation through aqueous and adsorbed phases, and the last term is the source/sink term corresponding to artificial injection and or withdrawal.

Equation B.2.9 is written in conservative form. It has been suggested that using the advective form is sometimes more appropriate, especially if the finite element method is used to simulate the chemical transport equation (Huyakorn et al., 1986). More importantly, an advective form of transport equations allows one to use the mixed Lagrangian-Eulerian approach, which can better solve advection-dominant transport problems (Yeh and Tripathi 1987). Therefore, an advective form of the transport equation is derived by expanding the advection term and using the continuity equation for water flow. The water flow equation can be rewritten in the following form

$$\frac{\rho}{\rho_o}F\frac{\partial h}{\partial t} = -\nabla\cdot\left[\frac{\rho}{\rho_o}V\right] + \frac{\rho^*}{\rho_o}q$$

$$= -\frac{\rho}{\rho_o}\nabla\cdot V - V\cdot\nabla\left[\frac{\rho}{\rho_o}\right] + \frac{\rho^*}{\rho_o}q \qquad \text{(B.2.10)}$$

which is conservation of fluid mass. Substituting Equation B.2.10 into B.2.9 and performing the necessary manipulation, we obtain

$$\theta\frac{\partial C}{\partial t} + \rho_b\frac{\partial S}{\partial t} + V\cdot\nabla C - \nabla\cdot(\theta D\cdot\nabla C) = -\left[\alpha'\frac{\partial h}{\partial t} + \lambda\right]$$

$$(\theta C + \rho_b S) - (\theta K_w C + \rho_b K_s S) + m - \frac{\rho^*}{\rho}qC \qquad \text{(B.2.11)}$$

$$+ \left[F\frac{\partial h}{\partial t} + \frac{\rho_o}{\rho}V\cdot\nabla\left[\frac{\rho}{\rho_o}\right] - \frac{\partial\theta}{\partial t}\right]C$$

Because Equation B.2.11 involves two unknowns, C and S, a constitutive relationship must be posed. For the present model, the following empirical relationships are used:

$$S = K_d C \quad \textit{for linear isotherm} \qquad \text{(B.2.12)}$$

$$S = \frac{s_{max}KC}{1 + KC} \quad \textit{for Langmuir isotherm} \qquad \text{(B.2.13)}$$

$$S = KC^n \quad \textit{for Freundlich isotherm} \qquad \text{(B.2.14)}$$

where K_d is the distribution coefficient (L^3/M), s_{max} is the maximum concentration permitted in the medium in the Langmuir nonlinear isotherm, K is the coefficient in the Langmuir or Freundlich nonlinear isotherm, and n is the power index in the Freundlich nonlinear isotherm.

In order to use the Lagrangian-Eulerian approach, Equation B.2.11 is rewritten as:

a. For Linear Isotherm model:

$$(\theta + \rho_b K_d)\frac{D_{v_d}C}{Dt} = \nabla\cdot(\theta \mathbf{D}\cdot\nabla C) - \left[\alpha'\frac{\partial h}{\partial t} + \lambda\right]$$

$$(\theta C + \rho_b S) - (\theta K_w C + \rho_b K_s S) + m - \frac{\rho^*}{\rho} qC \qquad \text{(B.2.15)}$$

$$+ \left[F\frac{\partial h}{\partial t} + \frac{\rho_o}{\rho}\mathbf{V}\cdot\nabla\left(\frac{\rho}{\rho_o}\right) - \frac{\partial\theta}{\partial t}\right]C$$

$$V_d^* = \frac{\mathbf{V}}{\theta + \rho_b K_d} \qquad \text{(B.2.16)}$$

b. For Langmuir and Freundlich models:

$$\theta\frac{D_{v_f}C}{Dt} + \rho_b\frac{dS}{dC}\frac{\partial C}{\partial t} = \nabla\cdot(\theta \mathbf{D}\cdot\nabla C) - \left[\alpha'\frac{\partial h}{\partial t} + \lambda\right]$$

$$(\theta C + \rho_b S) - (\theta K_w C + \rho_b K_s S) + m - \frac{\rho^*}{\rho} qC \qquad \text{(B.2.17)}$$

$$+ \left[F\frac{\partial h}{\partial t} + \frac{\rho_o}{\rho}\mathbf{V}\cdot\nabla\left[\frac{\rho}{\rho_o}\right] - \frac{\partial\theta}{\partial t}\right]C$$

$$V_f^* = \frac{\mathbf{V}}{\theta} \qquad \text{(B.2.18)}$$

where $D_{v_d}()/Dt$ and $D_{v_f}()/Dt$ denote the material derivatives of () with respect to time in reference to the retarded velocity $\mathbf{V}_d$ and fluid velocity $\mathbf{V}_f$, respectively.

To complete the description of the hydrological transport as given by Equations B.2.15 or B.2.17, initial and boundary conditions must be specified in accordance with dynamic and physical consideration. It will be assumed that initially the dissolved concentrations are known throughout the region of interest, that is,

$$C = C_i(x,z) \quad in \quad R \qquad \text{(B.2.19)}$$

where C_i is the initial concentration and R is the region of interest. Initial concentrations for the dissolved concentrations may be obtained from field measurements or by solving the steady-state version of Equation B.2.15 or B.2.17 with time-invariant boundary conditions.

The specification of boundary conditions is a difficult and intricate task in transport modeling. From the dynamic point of view, a boundary segment may be classified as either flow-through or impervious. From the physical point of view, it is a soil-air interface, or soil-soil interface, or soil-water interface. From the mathematical point of view, it may be treated as a Dirichlet boundary on which the total analytical concentration is prescribed, Neumann boundary on which the flux due to the gradient of total analytical concentration is known, or Cauchy boundary on which the total flux is given. An even more difficult mathematical boundary is the variable conditions on which the boundary conditions are not known a priori but are themselves the solution to be sought. In other words, on the mathematically variable boundary, either Neumann or Cauchy conditions may prevail and change with time. Which condition prevails at a particular time can be determined only in the cyclic processes of solving the governing equations (Freeze 1972a, 1972b; Yeh and Ward 1980; Yeh and Ward 1981).

Whatever point of view is chosen, all boundary conditions eventually must be transformed into mathematical equations for quantitative simulations. Thus, we will specify the boundary conditions from the mathematical point of view in concert with dynamic and physical considerations. The boundary conditions imposed on any segment of the boundary are taken to be either Dirichlet, Neumann, Cauchy, or variable. Thus, the global boundary may be split into four parts, B_d, B_n, B_c, and B_v, denoting Dirichlet, Neumann, Cauchy, and variable boundaries, respectively. The conditions imposed on the first three types of boundaries are given as:

a. Prescribed concentration (Dirichlet) boundary conditions:

$$C = C_d(x_b, y_b, z_b, t) \quad on \quad B_d \tag{B.2.20}$$

b. Neumann boundary conditions:

$$n \cdot (-\theta \mathbf{D} \cdot \nabla C) = q_n(x_b, y_b, z_b, t) \quad \text{on} \quad B_n \tag{B.2.21}$$

c. Cauchy boundary conditions:

$$n \cdot (\mathbf{V}C - \theta \mathbf{D} \cdot \nabla C) = q_c(x_b, y_b, z_b, t) \quad \text{on} \quad B_c \tag{B.2.22}$$

where C_d is the prescribed concentration on the Dirichlet boundary B_d, (x_b,y_b,z_b) is the spatial coordinate on the boundary, $\mathbf{n}$ is an outward unit vector normal to the boundary, q_n is the prescribed gradient flux through the Neumann boundary B_n, and q_c is the prescribed total flux through the Cauchy boundary B_c.

The conditions imposed on the variable-type boundary, which is normally the soil-air interface or soil-water interface, are either the Neumann with zero gradient flux or the Cauchy with given total flux. The former is specified when the water flow is directed out of the region from the far boundary, whereas the latter is specified when the water flow is directed into the region. This type of variable condition would normally occur at flow-through boundaries. Written mathematically, the variable boundary condition is given by

$$\begin{aligned} \mathbf{n}\cdot(\mathbf{V}C - \theta\mathbf{D}\cdot\nabla C) &= \mathbf{n}\cdot\mathbf{V}C_v(x_b,z_b,t) \quad \text{on } B_v \quad \text{if } \mathbf{n}\cdot\mathbf{V} \le 0 \\ \mathbf{n}\cdot(-\theta\mathbf{D}\cdot\nabla C) &= 0 \qquad\qquad\qquad\quad \text{on } B_v \quad \text{if } \mathbf{n}\cdot\mathbf{V} > 0 \end{aligned} \tag{B.2.23}$$

where C_v is the specified concentration of water through the variable boundary and B_v is the variable boundary.

Appendix C
Numerical Formulation

The initial-boundary value problem described by the governing equations of the flow and transport modules of 3DSALT along with the boundary conditions cannot, in general, be solved analytically using current applied mathematics. Hence, in order to solve these sets of governing equations, numerical methods are the only mathematical tools capable of handling this task. Even though there are many different numerical approximation methods capable of reducing partial differential equations to simpler systems of algebraic equations, there are only two numerical methods that are most common and that can be employed to the most basic form of the governing equations. These two numerical methods are the finite difference and finite element methods. The basic difference between these two methods is that the finite element method is based upon approximating the function, while the finite difference method is founded upon approximating the derivatives of the function. Therefore, the finite difference method only produces solutions at discrete points, while the finite element method yields spatially continuous solutions. Also, the finite element method offers numerous advantages over the finite difference method, such as (a) anisotropy and heterogeneity of aquifers are easily taken care of, (b) it is not necessary to formulate special formulae to incorporate irregular boundaries, (c) computer storage and computational time can sometimes be saved because often fewer nodal points are needed to portray the region of interest to the same level of accuracy, (d) irregular grids for handling different levels of spatial discretization in different sections of the region of interest can be incorporated, and lastly, (e) the integral formulation used in this method permits the flux types of the boundary conditions to come about naturally (Yeh 1987).[1] Thus, the finite element method is used in this model. The theoretical background as well as numerical procedures of this method can be found in any good finite element method book, such as Istok (1989), and therefore will not be described here. A brief summary of the numerical procedure for applying the finite element method can be found in Yeh (1987).

The flow module of 3DSALT includes four options for solving the finite

[1] References cited in this Appendix are listed in the References at the end of the main text.

element equations. In other words, there are four iteration methods (block iteration, successive point iteration, polynomial preconditioned conjugate gradient, and incomplete Cholesky preconditioned conjugate gradient methods) for solving the linearized matrix equations. Direct elimination methods are not used in this report because it is impractical to deal with large three-dimensional problems. Because the Newton-Raphson method will yield a nonsymmetric matrix, the Picard method is used to linearize the matrix equation.

To handle a large variety of possible problem sets, the flow module for 3DSALT contains 16 optional numerical schemes. Specifically, the mixture of schemes includes the combinations of (a) the four options for solving the resulting matrix equation as mentioned in the above discussion, (b) two options (lumping and consistent) for handling the mass matrix resulting from the storage term, and (c) two options (time-weighted difference and mid-difference) for approximating the time derivatives. The theoretical background for (b) and (c) may also be found in any respectable matrix computation book and in Yeh (1991).

On the other hand, the transport module for 3DSALT also includes these 16 options, plus more. While the Galerkin finite element method is used in the flow module, an option of either two conventional finite element methods, either the Galerkin finite element method or the upstream weighting finite element method, or the alternative option of a hybrid Lagrangian-Eulerian finite element method is provided in the transport module. The main difference between the two conventional finite element methods is that while the Galerkin finite element method uses the base functions as the weighting functions, the upstream weighting finite element method uses weighting functions different from the base functions. The advantages of using the upstream weighting finite element method over the Galerkin finite element method become apparent when the advection terms in the transport governing equation are equally important to the dispersion terms (Yeh and Ward 1981). More details of the two conventional finite element methods may be found in Yeh and Ward (1981).

Numerical Approximation of the Flow Equations

Spatial discretization with the Galerkin finite element method

When using the finite element method, the referenced pressure head is approximated by:

$$h \approx \hat{h} = \sum_{j=1}^{N} h_j(t) N_j(x,y,z) \qquad \text{(C.1.1)}$$

where h_j and N_j are the amplitude of h and the base function, respectively, at nodal point j and N is the total number of nodes. After defining a residual and forcing the weighted residual to zero, the flow equation, Equation B.1.33, is approximated as:

$$\left[\int_R N_i \frac{\rho}{\rho_0} F N_j dR\right] \frac{dh_j}{dt} + \left[\int_R (\nabla N_i)\cdot \boldsymbol{K} \cdot (\nabla N_j) dR\right] h_j = \int_R N_i \frac{\rho^*}{\rho_0} q dR - \int_R (\nabla N_i)\cdot \boldsymbol{K} \cdot \frac{\rho}{\rho_0} \nabla z dR + \int_B \boldsymbol{n}\cdot \boldsymbol{K} \cdot \left[\nabla h + \frac{\rho}{\rho_0} \nabla z\right] N_i dB \quad \text{(C.1.2)}$$

In matrix form, Equation C.1.2 is written as:

$$[M]\{\frac{dh}{dt}\} + [S]\{h\} = \{Q\} + \{G\} + \{B\} \quad \text{(C.1.3)}$$

where {dh/dt} and {h} are the column vectors containing the values of dh/dt and h, respectively, at all nodes; [M] is the mass matrix resulting from the storage term; [S] is the stiff matrix resulting from the action of conductivity; {Q}, {G}, and {B} are the load vectors from the internal source/sink, gravity force, and boundary conditions, respectively. Furthermore, the mass matrix, [M], and stiff matrix, [S], are described as:

$$M_{ij} = \sum_{e \in M_e} \int_{R_e} N_\alpha^e \frac{\rho}{\rho_0} F N_\beta^e dR \quad \text{(C.1.4)}$$

and

$$S_{ij} = \sum_{e \in M_e} \int_{R_e} (\nabla N_\alpha^e)\cdot \boldsymbol{K} \cdot (\nabla N_\beta^e) dR \quad \text{(C.1.5)}$$

where R_e is the region of element e, M_e is the set of elements that have a local side α-β coinciding with the global side i-j, and N_α^e is the α-th local base function of element e.

In addition, the three load vectors, {Q}, {G}, and {B}, are described as:

$$Q_i = \sum_{e \in M_e} \int_{R_e} N_\alpha^e \frac{\rho^*}{\rho_0} q \, dR \tag{C.1.6}$$

$$G_i = -\sum_{e \in M_e} \int_{R_e} (\nabla N_\alpha^e) \cdot \boldsymbol{K} \cdot \frac{\rho}{\rho_0} \nabla z \, dR \tag{C.1.7}$$

and

$$B_i = -\sum_{e \in N_w} \int_{B_e} N_\alpha^e \boldsymbol{n} \cdot \left[-\boldsymbol{K} \cdot \left[\nabla h + \frac{\rho}{\rho_0} \nabla z \right] \right] dB \tag{C.1.8}$$

where N_{we} is the set of boundary segments that have a local node α coinciding with the global node i, and B_e is the length of boundary segment e.

In most finite element work, the Darcy velocity components given in Equation B.1.37 are calculated numerically by taking the derivatives of the simulated h as

$$V = -\boldsymbol{K} \cdot \left[\frac{\rho_o}{\rho} (\nabla N_j) h_j + \nabla z \right] \tag{C.1.9}$$

The above formulation results in a velocity field which is not continuous at element boundaries and nodal points if the variation of h is other than linear or constants. The alternative approach would be to apply the Galerkin finite element method to Equation B.1.37; thus one obtains

$$[T]\{V_x\} = \{D_x\} \tag{C.1.10}$$

$$[T]\{V_y\} = \{D_y\} \tag{C.1.11}$$

$$[T]\{V_z\} = \{D_z\} \tag{C.1.12}$$

where the matrix [T] and the load vectors $\{D_x\}$, $\{D_y\}$, and $\{D_z\}$ are given by

$$T_{ij} = \sum_{e \in M_e} \int_{R_e} N_\alpha^e N_\beta^e \, dR \tag{C.1.13}$$

$$D_{xi} = -\sum_{e \in M_e} \int_{R_e} N_\alpha^e \, \mathbf{i} \cdot \mathbf{K} \cdot \left[\frac{\rho_o}{\rho} \nabla h + \nabla z \right] dB \tag{C.1.14}$$

$$D_{yi} = -\sum_{e \in M_e} \int_{R_e} N_\alpha^e \, \mathbf{j} \cdot \mathbf{K} \cdot \left[\frac{\rho_o}{\rho} \nabla h + \nabla z \right] dB \tag{C.1.15}$$

$$D_{zi} = -\sum_{e \in M_e} \int_{R_e} N_\alpha^e \, \mathbf{k} \cdot \mathbf{K} \cdot \left[\frac{\rho_o}{\rho} \nabla h + \nabla z \right] dB \tag{C.1.16}$$

where V_x, V_y, and V_z are the Darcy velocity components along the x-, y-, and z-directions, respectively, and **i**, **j**, and **k** are the unit vector along the x-, y-, and z-coordinates, respectively.

The reduction of the partial differential equation (Equation B.1.33) to the set of ordinary differential equations (Equation C.1.3) simplifies to the evaluation of integrals on the right-hand side of Equations C.1.4 through C.1.8 for every element for boundary surface e. The major tasks that remain to be done are the specification of base and weighting functions and the performance of integration to yield the element matrices. Linear hexahedral elements are employed in this documentation.

Base and weighting functions

The construction of base functions for hexahedral elements is best accomplished using the local coordinates (ξ,η,ζ). In the local coordinates, the original hexahedral element is mapped into a cubic whose corners are located at $\xi = \pm 1$, $\eta = \pm 1$, and $\zeta = \pm 1$ as shown in Figure C.1. For tri-linear hexahedral elements, the eight base functions are obtained by taking the tensor product of the three base functions for the linear line elements as

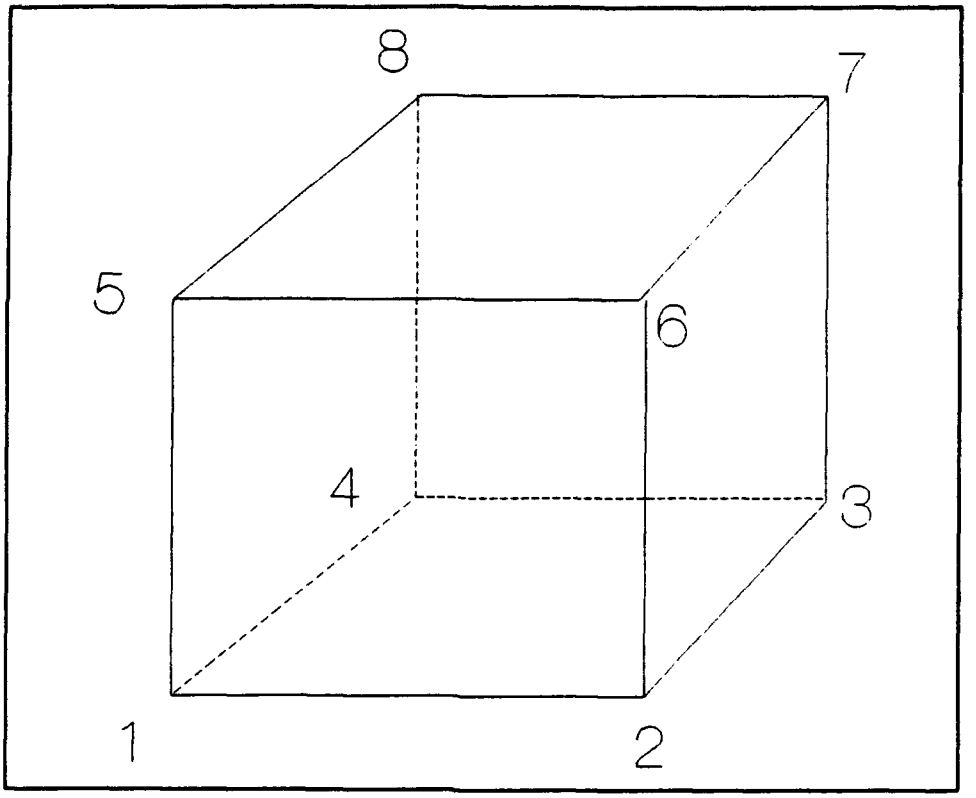


Figure C.1. A hexahedral element in local coordinates

$$N_i(\xi,\eta,\zeta) = \frac{1}{8}(1+\xi\xi_i)(1+\eta\eta_i)(1+\zeta\zeta_i), \qquad i = 1, 2, \ldots, 8 \quad \text{(C.1.17)}$$

Because the Galerkin finite element method is used to solve the flow equations, the set of eight weighting functions is taken as the same set of eight base functions.

Numerical integration

To complete the reduction of the partial differential equation (Equation B.1.33) to the ordinary differential equation (Equation C.1.3), one has to evaluate the integrals on the right sides of Equations C.1.4 through C.1.8 for every element to yield the element mass matrix $[M^e]$ and the stiff element matrix $[S^e]$ as well as the element gravity column vector $\{G^e\}$, the source/sink column vector $\{Q^e\}$, and the boundary column vector $\{B^e\}$ as

$$M^e_{\alpha\beta} = \int_{R_e} N^e_\alpha \frac{\rho}{\rho_0} F N^e_\beta \, dR \qquad \text{(C.1.18)}$$

$$S^e_{\alpha\beta} = \int_{R_e} (\nabla N^e_\alpha) \cdot K \cdot (\nabla N^e_\beta) dR \tag{C.1.19}$$

$$Q^e_\alpha = \int_{R_e} N^e_\alpha \frac{\rho^*}{\rho_0} q dR \tag{C.1.20}$$

$$G^e_\alpha = -\int_{R_e} (\nabla N^e_\alpha) \cdot K \cdot \frac{\rho}{\rho_0} \nabla z dR \tag{C.1.21}$$

and

$$B^e_\alpha = -\int_{B_e} N^e_\alpha n \cdot \left[-K \cdot \left[\nabla h + \frac{\rho}{\rho_0} \nabla z \right] \right] dB \tag{C.1.22}$$

Since Equations C.1.18 through C.1.22 are written in global coordinate and the base functions are defined in local coordinate, a transformation between the global and local coordinate is needed. The required transformation from global coordinate to local coordinate is obtained via the base functions as

$$x = \sum_{j=1}^{8} x_j N_j(\xi, \eta, \zeta) \tag{C.1.23}$$

$$y = \sum_{j=1}^{8} y_j N_j(\xi, \eta, \zeta) \tag{C.1.24}$$

$$z = \sum_{j=1}^{8} z_j N_j(\xi, \eta, \zeta) \tag{C.1.25}$$

Because the coordinate transformation uses the base functions, the element is termed the "isoparametric" element.

Using the transformation in Equations C.1.23 through C.1.25, we convert the differentiation of the base function with respect to the global coordinate to that with respect to local coordinate by

$$\begin{Bmatrix} \frac{\partial N_i}{\partial x} \\ \frac{\partial N_i}{\partial y} \\ \frac{\partial N_i}{\partial z} \end{Bmatrix} = [J]^{-1} \begin{Bmatrix} \frac{\partial N_i}{\partial \xi} \\ \frac{\partial N_i}{\partial \eta} \\ \frac{\partial N_i}{\partial \zeta} \end{Bmatrix} \tag{C.1.26}$$

$$[J] = \begin{bmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial y}{\partial \xi} & \frac{\partial z}{\partial \xi} \\ \frac{\partial x}{\partial \eta} & \frac{\partial y}{\partial \eta} & \frac{\partial z}{\partial \eta} \\ \frac{\partial x}{\partial \zeta} & \frac{\partial y}{\partial \zeta} & \frac{\partial z}{\partial \zeta} \end{bmatrix} \qquad [J]^{-1} = \textit{Inverse of } [J]$$

where [J] is the Jacobian of the transformation. In the mean time, a differential volume written in local coordinate becomes

$$\int_e dR = \int_{-1}^{1}\int_{-1}^{1}\int_{-1}^{1} J d\xi d\eta d\zeta \tag{C.1.27}$$

With Equations C.1.26 and C.1.27, all the integrals in Equations C.1.18 through C.1.21 can be reduced to the following form

$$\int_{-1}^{1}\int_{-1}^{1}\int_{-1}^{1} f(\xi,\eta,\zeta) J d\xi d\eta d\zeta \tag{C.1.28}$$

the integration of which can easily be carried out with a $2 \times 2 \times 2 = 8$-point Gaussian quadrature. The surface integration of Equation C.1.22 in three-dimensional space is not as straightforward as in two-dimensional space. This integration requires further elaboration. Any surface integral of a continuous function F(x,y,z) specified on the surface S (Figure C.2) can be reduced to the area integration. Let I represent the surface integral:

$$I = \int_S F(x,y,z) dS \tag{C.1.29}$$

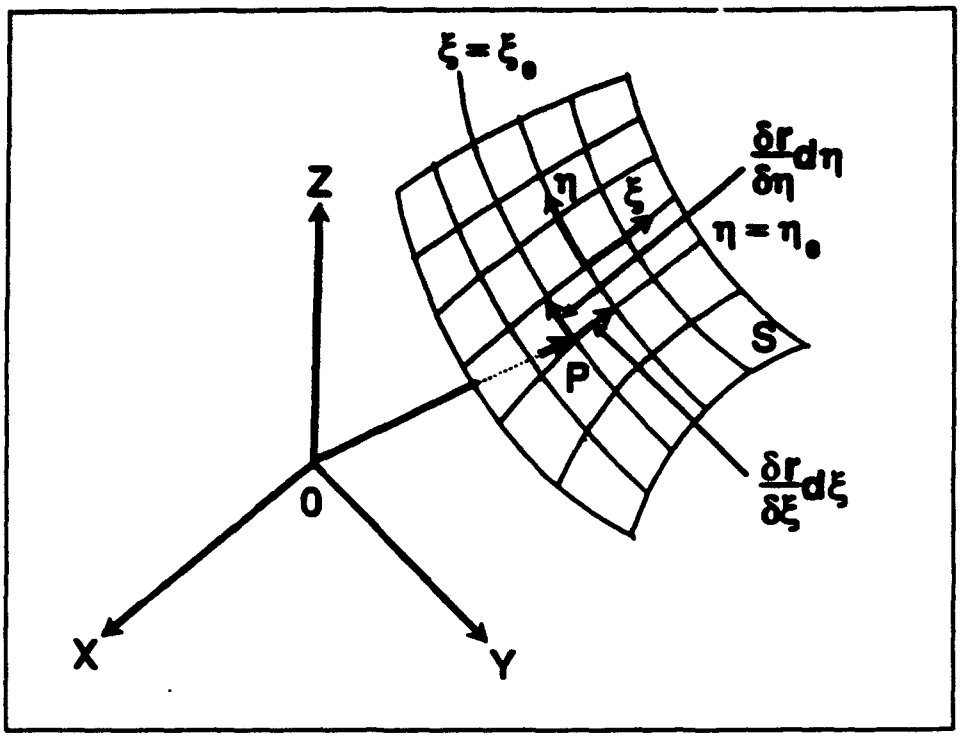


Figure C.2. A surface area and its imbedded local coordinate

where the surface S is given by the following equation

$$z = f(x,y) \tag{C.1.30}$$

Let P be any point on the surface S with coordinates (x,y,z) or (ξ,η) (Figure C.2). Then the vector r from O to P is given by

$$r = xi + yj + zk \tag{C.1.31}$$

The tangent vectors to the coordinate curves $\xi = \xi_o$ and $\eta = \eta_o$ on the surface S are $\partial r/\partial\eta$ and $\partial r/\partial\xi$, respectively. The area dS is given by

$$dS = \left| \frac{\partial r}{\partial \xi} x \frac{\partial r}{\partial \eta} \right| d\xi d\eta \tag{C.1.32}$$

where x represents vector multiplication. But

$$\frac{\partial r}{\partial \xi} x \frac{\partial r}{\partial \eta} = \begin{vmatrix} i & j & k \\ \frac{\partial x}{\partial \xi} & \frac{\partial y}{\partial \xi} & \frac{\partial z}{\partial \xi} \\ \frac{\partial x}{\partial \eta} & \frac{\partial y}{\partial \eta} & \frac{\partial z}{\partial \eta} \end{vmatrix} \tag{C.1.33}$$

so that

$$dS = \sqrt{J_x^2 + J_y^2 + J_z^2}\, d\xi d\eta \tag{C.1.34}$$

where

$$J_z = \begin{vmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial y}{\partial \xi} \\ \frac{\partial x}{\partial \eta} & \frac{\partial y}{\partial \eta} \end{vmatrix}, \quad J_y = \begin{vmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial z}{\partial \xi} \\ \frac{\partial x}{\partial \eta} & \frac{\partial z}{\partial \eta} \end{vmatrix}, \quad J_x = \begin{vmatrix} \frac{\partial z}{\partial \xi} & \frac{\partial y}{\partial \xi} \\ \frac{\partial z}{\partial \eta} & \frac{\partial z}{\partial \eta} \end{vmatrix}. \tag{C.1.35}$$

Substituting Equation C.1.34 into Equation C.1.29, we obtain

$$\int_S F(x,y,z)\, dS = \int_{-1}^{1} \int_{-1}^{1} \phi(\xi,\eta) \sqrt{J_x^2 + J_y^2 + J_z^2}\, d\xi d\eta \tag{C.1.36}$$

where

$$\phi(\xi,\eta) = F(x(\xi,\eta), y(\xi,\eta), z(\xi,\eta)) \tag{C.1.37}$$

Surface integrals (Equation C.1.36) can easily be computed by Gaussian quadrature.

Mass lumping option

Referring to [M], one may recall that this is a unit matrix if the finite difference formulation is used in spatial discretization. Hence, by proper scaling, the mass matrix can be reduced to the finite-difference equivalent by

lumping (Clough 1971). In many cases, the lumped mass matrix would result in better solution, in particular, if it is used in conjunction with the central or backward-difference time marching (Yeh and Ward 1980). Under such circumstances, it is preferred to the consistent mass matrix (mass matrix without lumping). Therefore, options are provided for the lumping of the matrix [M]. More explicitly, [M] will be lumped according to:

$$M_{ij} = \sum_{e \ni M_i} \left[\sum_{\beta=1}^{4} \int_{R_e} N_\alpha^e \frac{\rho}{\rho_o} F N_\beta^e dR \right] \quad if \quad j = i \,, \tag{C.1.38}$$

and

$$M_{ij} = 0 \qquad if \qquad j \neq i \tag{C.1.39}$$

Finite difference approximation in time

Next, we derive a matrix equation by integrating Equation C.1.3. For the time integration of Equation C.1.3, the load vector {B} will be ignored. This load vector will be discussed in the next section on the numerical implementation of boundary conditions. An important advantage in finite element approximation over the finite difference approximation is the inherent ability to handle complex boundaries and obtain the normal derivatives therein. In the time dimension, such advantages are not evident. Thus, finite difference methods are typically used in the approximation of the time derivative. Two time-marching methods are adopted in the present flow model.

The first one is the time weighted method written as:

$$\frac{[M]}{\Delta t}\left(\{h\}_{t+\Delta t} - \{h\}_t\right) + \omega[S]\{h\}_{t+\Delta t} + (1-\omega)\{h\}_t = \{Q\} + \{G\} \tag{C.1.40}$$

where [M], [S], {Q}, and {G} are evaluated at $(t + \omega\Delta t)$. In the Crank-Nicholson centered-in-time approach $\omega = 0.5$, in the backward-difference (implicit difference) $\omega = 1.0$, and in the forward-difference (explicit scheme) $\omega = 0.0$. The central-Nicholson algorithm has a truncation error of $O(\Delta t^2)$, but its propagation-of-error characteristics frequently lead to oscillatory nonlinear instability. Both the backward-difference and forward-difference have a truncation error of $O(\Delta t)$. The backward-difference is quite resistant to oscillatory nonlinear instability. On the other hand, the forward difference is

only conditionally stable even for linear problems, not to mention nonlinear problems.

In the second method, the values of unknown variables are assumed to vary linearly with time during the time interval, Δt. In this mid-difference method, the recurrence formula is written as:

$$\left[\frac{2}{\Delta t}[M] + [S]\right]\{h\}_{t+\Delta t/2} - \frac{2}{\Delta t}[M]\{h\}_t = \{Q\} + \{G\} \qquad \text{(C.1.41)}$$

and

$$\{h\}_{t+\Delta t} = 2\{h\}_{t+\Delta t/2} - \{h\}_t \,, \qquad \text{(C.1.42)}$$

where [M], [S], and {Q} are evaluated at (t+Δt/2).

Equations C.1.40 and C.1.41 can be written as a matrix equation

$$[T]\{h\} = \{Y\} \,, \qquad \text{(C.1.43)}$$

where [T] is the matrix, {h} is the unknown vector to be found and represents the values of discretized pressure field at new time, and {Y} is the load vector. Take for example, Equation C.1.40 with $\omega = 1.0$, [T] and {Y} represent the following:

$$[T] = \frac{[M]}{\Delta t} + [S] \,, \qquad \text{(C.1.44)}$$

and

$$\{Y\} = \frac{[M]}{\Delta t}\{h\}_t + \{Q\} + \{G\} \qquad \text{(C.1.45)}$$

where {h} is the vector of the discretized pressure field at previous time.

Numerical implementation of boundary conditions

The following steps are the incorporation of boundary conditions into matrix equation by finite element method. For the Cauchy boundary condition given by Equation B.1.45, we simply substitute Equation B.1.45 into Equation C.1.22 to yield a boundary-element column vector $\{B_c^e\}$ for a Cauchy segment:

$$\{B_c^e\} = \{q_c^e\} \tag{C.1.46}$$

where $\{q_c^e\}$ is the Cauchy boundary flux vector given by

$$q_{c\alpha}^e = -\int_{B_e} N_\alpha^e \frac{\rho}{\rho_w} q_c dB \ , \qquad \alpha = 1 \ or \ 2 \tag{C.1.47}$$

The Cauchy boundary flux vector represents the normal fluxes through the two nodal points of the segment B_e on B_c. For the Neumann boundary condition given by Equation B.1.44, we substitute Equation B.1.44 into Equation C.1.22 to yield a boundary-element column vector $\{B_n^e\}$ for a Neumann segment:

$$\{B_n^e\} = \{q_n^e\} \tag{C.1.48}$$

where $\{q_{n\alpha}^e\}$ is the Neumann boundary flux vector given by:

$$q_{n\alpha}^e = \int_{B_e} \left[N_\alpha^e \boldsymbol{n} \cdot \boldsymbol{K} \cdot \frac{\rho}{\rho_w} \nabla z - N_\alpha^e q_n \right] dB \ ; \quad \alpha = 1 \ or \ 2 \tag{C.1.49}$$

which is independent of pressure head.

The implementation of variable-type boundary condition is more involved. During the iteration of boundary conditions on the variable boundary, one of Equations B.1.46a-B.1.46e is used at a node. If either Equation B.1.46b or B.1.46e is used, we substitute it into Equation C.1.22 to yield a boundary element column vector $\{B_v^e\}$ for a variable boundary segment:

$$\{B_v^e\} = \{q_v^e\} \tag{C.1.50}$$

where $\{q_v^e\}$ is the variable boundary flux given by:

$$q_{v\alpha}^e = -\int_{B_v} N_\alpha^e \frac{\rho}{\rho_w} q_p dB, \text{ or } q_{v\alpha} = -\int_{B_v} N_\alpha^e \frac{\rho}{\rho_w} q_e dB; \alpha = 1 \text{ or } 2 \qquad \text{(C.1.51)}$$

Assembling over all Neumann, Cauchy, and variable boundary segments, we obtain the global boundary column vector {B} as:

$$\{B\} = \{q\} \qquad \text{(C.1.52)}$$

in which

$$\{q\} = \sum_{e \in N_{ne}} \{q_n^e\} + \sum_{e \in N_{ce}} \{q_c^e\} + \sum_{e \in N_{ve}} \{q_v^e\} \qquad \text{(C.1.53)}$$

where N_{ne}, N_{ce}, and N_{ve} are the number of Neumann boundary segments, Cauchy boundary segments, and variable boundary segments with flux conditions imposed on them, respectively. The boundary flux {B} given by Equations C.1.52 and C.1.53 should be added to the right-hand side of Equation C.1.43.

At nodes where Dirichlet boundary conditions are applied, an identity equation is generated for each node and included in the matrices of Equation C.1.43. The Dirichlet nodes include the nodes on the Dirichlet boundary and the nodes on the variable boundary to which either Equation B.1.46a, B.1.46c, or B.1.46d is applied.

After time discretization of Equation C.1.3 and incorporation of boundary conditions, we obtain the following matrix equation

$$[C]\{h\} = \{R\} \qquad \text{(C.1.54)}$$

where [C] is the coefficient matrix and {R} is the known vector of the right-hand side. For the saturated-unsaturated flow simulation, [C] is a highly nonlinear function of the pressure head {h}.

Solution of the matrix equations

Equation C.1.54 is in general a banded sparse matrix equation. It may be solved numerically by either direct methods or iteration methods. In direct methods, a sequence of operation is performed only once. This would result in each solution except for round-off error. In this method, one is concerned with the efficiency and magnitude of round-off error associated with the sequence of operations. On the other hand, in an iterative method, one attempts the solution by a process of successive approximations. This involves making an initial guess, then improving the guess by some iterative process until an error criterion is obtained. Therefore, in this technique, one must be concerned with convergence and the rate of convergence. The round-off errors tend to be self-corrected.

For practical purposes the advantages of direct method are as follows: (a) the efficient computation when the bandwidth of the matrix [C] is small, and (b) no problem of convergency is encountered when the matrix equation is linear or less severe in convergence than iterative methods even when the matrix equation is nonlinear. The disadvantages of direct methods are the excessive requirements on CPU storage and CPU time when a large number of nodes are needed for discretization. On the other hand, the advantages of iteration methods are efficiencies in terms of CPU storage and CPU time when large problems are encountered. Their disadvantages are the requirements that the matrix [C] must be well conditioned to guarantee a convergent solution. For three-dimensional problems, the bandwidth of the matrix is usually large; thus the direction solution method is not practical. Only the iterative methods are implemented in 3DSALT. Four iteration methods are used in solving the linearized matrix equation: (a) block iteration, (b) successive point iteration, (c) polynomial preconditioned conjugate gradient method, and (d) incomplete Cholesky preconditioned conjugate gradient method.

The matrix equation, Equation C.1.54, is nonlinear because both the hydraulic conductivity and the water capacity are functions of the pressure head h. To solve the nonlinear matrix equation, two approaches can be taken: (a) the Picard method and (b) the Newton-Raphson method. The Newton-Raphson method has a second order of convergent rate and is very robust. However, the Newton-Raphson method would destroy the symmetrical property of the coefficient matrix resulting from the finite element approximation. As a result the solution of the linearized matrix equation requires extra care. Many of the iterative methods will not warrant a convergent solution for the nonsymmetric linearized matrix equation Thus, the Picard method is used in this report to solve the nonlinear problems.

In the Picard method, an initial estimate is made of the unknown {h}. Using this estimate, we then compute the coefficient matrix [C] and solve the linearized matrix equation by the method of linear algebra. The new estimate

is now obtained by the weighted average of the new solution and the previous estimate:

$$\{h^{(k+1)}\} = \omega\{h\} + (1-\omega)\{h^k\} , \qquad \text{(C.1.55)}$$

where $\{h^{(k+1)}\}$ is the new estimate, $\{h^k\}$ is the previous estimate, $\{h\}$ is the new solution, and ω is the iteration parameter. The procedure is repeated until the new solution $\{h\}$ is within a tolerance error. If ω is greater than or equal to 0 but is less than 1, the iteration is under-relaxation. If $\omega = 1$, the method is the exact relaxation. If ω is greater than 1 but less than or equal to 2, the iteration is termed over-relaxation. The under-relaxation should be used to overcome cases when nonconvergency or the slow convergent rate is due to fluctuation rather than due to "blowup" computations. Over-relaxation should be used to speed up convergent rate when it decreases monotonically.

In summary, there are 16 optional numerical schemes here to deal with as wide a range of problems as possible. These are the combinations of (a) two ways of treating the mass matrix (lumping and no-lumping); (b) two ways of approximating the time derivatives (time-weighting and mid-difference), and (c) four ways of solving the linearized matrix equation.

Transport Equation

Spatial discretization with the weighted residual finite element method

Let C_j be approximated by a finite element interpolation as

$$C \approx \hat{C} = \sum_{j=1}^{N} C_j(t) N_j(x,y,z) \qquad \text{(C.2.1)}$$

Neglecting the fluid and medium compressibilities, ignoring the biodegradation, substituting Equation C.2.1 into Equations B.2.11, B.2.15, and B.2.17, and forcing a weighted residual to zero, we obtain the following ordinary differential equations:

a. For the conventional finite element approach

$$\left[\int_R N_i\left(\theta+\rho_b\frac{dS}{dC}\right)N_j dR\right]\left\{\frac{dC}{dt}\right\}+\left[\int_R W_i V\cdot\nabla N_j dR\right]\{C\}+$$

$$\left[\int_R \nabla N_i\cdot\theta D\cdot\nabla N_j dR\right]\{C\}+\left\{\int_R N_i\left[\lambda\left(\theta+\rho_b\frac{dS}{dC}\right)\right]N_j dR\right\}\{C\}+$$

(C.2.2)

$$\left\{\int_R N_i\left[\frac{\rho^*}{\rho}q-\frac{\rho_o}{\rho}V\cdot\nabla\left(\frac{\rho}{\rho_o}\right)\right]\right\}\{C\}=\int_R N_i\left[-\lambda\rho_b\left(S+\frac{dS}{dC}C\right)\right]dR+$$

$$\int_R N_i q C_{in} dR+\int_B N_i n\cdot\theta D\cdot\nabla C dB$$

b. For the Lagrangian-Eulerian approach with a linear isotherm:

$$\left[\int_R N_i\left(\theta+\rho_b\frac{dS}{dC}\right)N_j dR\right]\left\{\frac{D_{V_x}C}{dt}\right\}+$$

$$\left[\int_R \nabla N_i\cdot\theta D\cdot\nabla N_j dR\right]\{C\}+\left\{\int_R N_i\left[\lambda\left(\theta+\rho_b\frac{dS}{dC}\right)\right]N_j dR\right\}\{C\}+$$

(C.2.3)

$$\left\{\int_R N_i\left[\frac{\rho^*}{\rho}q-\frac{\rho_o}{\rho}V\cdot\nabla\left(\frac{\rho}{\rho_o}\right)\right]\right\}\{C\}=$$

$$\int_R N_i\left[-\lambda\rho_b\left(S+\frac{dS}{dC}C\right)\right]dR+\int_R N_i\, qC_{in}dR+\int_B N_i\, n\cdot\theta D\cdot\nabla C dB$$

c. For the Lagrangian-Eulerian approach with a nonlinear isotherm:

$$\left[\int_R N_i \theta N_j dR\right]\left\{\frac{D_v C}{dt}\right\} + \left[\int_R N \rho_b \frac{dS}{dC} N_j dR\right]\frac{dC}{dt} +$$

$$\left[\int_R \nabla N_i \cdot \theta D \cdot \nabla N_j dR\right]\{C\} + \left\{\int_R N_i \left[\lambda\left[\theta + \rho_b \frac{dS}{dC}\right]\right] N_j dR\right\}\{C\} +$$

(C.2.4)

$$\left\{\int_R N_i \left[\frac{\rho^*}{\rho} q - \frac{\rho_o}{\rho} V \cdot \nabla\left[\frac{\rho}{\rho_o}\right]\right]\right\}\{C\} =$$

$$\int_R N_i \left[-\lambda \rho_b \left[S + \frac{dS}{dC} C\right]\right] dR + \int_R N_i q C_{in} dR + \int_B N_i n \cdot \theta D \cdot \nabla C dB$$

The above equations, Equations C.2.2 through C.2.4, are written in matrix form as:

a. For the conventional finite element approach

$$[M]\left\{\frac{dC}{dt}\right\} + ([A] + [D] + [K])\{C\} = \{Q\} + \{B\} \qquad \text{(C.2.5)}$$

b. For the Lagrangian-Eulerian approach with the linear isotherm

$$[M]\left\{\frac{D_v C}{Dt}\right\} + ([D] + [K])\{C\} = \{Q\} + \{B\} \qquad \text{(C.2.6)}$$

c. For the Lagrangian-Eulerian approach with nonlinear isotherms

$$\left[[M_1]\left\{\frac{D_v C}{Dt}\right\} + [M_2]\left\{\frac{dC}{dt}\right\}\right] + ([D] + [K])\{C\} = \{Q\} + \{B\} \qquad \text{(C.2.7)}$$

where {C} is a vector whose components are the concentrations at all nodes, {dC/dt} is the derivative of {C} with respect to time, [M] and [M_1] are the mass matrices associated with the material derivative term, [M_2] is a mass matrix associated with the partial derivative term, [D] is the stiff matrix associated with the dispersion term, [A] is the stiff matrix associated with the advection term, [K] is the stiff matrix associated with all the first-order terms, {Q} is the load vector associated with all zero-order derivative terms, and {B} is the load vector associated with boundary conditions. The above matrices and vectors are given as:

$$M_{ij} = \sum_{e \in M} \int_{R_e} N_\alpha^e \left[\theta + \rho_b \frac{dS}{dC} \right] N_\beta^e \, dR \qquad \text{(C.2.8)}$$

$$M_{1ij} = \sum_{e \in M} \int_{R_e} N_\alpha^e \theta N_\beta^e \, dR \qquad \text{(C.2.9)}$$

$$M_{2ij} = \sum_{e \in M} \int_{R_e} N_\alpha^e \left[\rho_b \frac{dS}{dC} \right] N_\beta^e \, dR \qquad \text{(C.2.10)}$$

$$A_{ij} = \sum_{e \in M} \int_{R_e} N_\alpha^e V \cdot \nabla N_\beta^e \, dR \qquad \text{(C.2.11)}$$

$$D_{ij} = \sum_{e \in M} \int_{R_e} \nabla N_\alpha^e \cdot \theta D \cdot \nabla N_\beta^e \, dR \qquad \text{(C.2.12)}$$

$$K_{ij} = \sum_{e \in M} \int_{R_e} N_\alpha^e \left[\lambda \left[\theta + \rho_b \frac{dS}{dC} \right] + \frac{\rho^*}{\rho} q - \frac{\rho_o}{\rho} V \cdot \nabla \left[\frac{\rho}{\rho_o} \right] \right] N_\beta^e \, dR \qquad \text{(C.2.13)}$$

$$Q_i = \sum_{e \in M} \int_{R_e} N_\alpha^e \left[-\lambda \rho_b \left[S + \frac{dS}{dC} C \right] + q C_{in} \right] dR \qquad \text{(C.2.14)}$$

$$\mathrm{B_i} = -\sum_{e \in B} \int_{B_e} \mathrm{N_\alpha^e n} \cdot (-\theta \mathbf{D} \cdot \nabla \mathrm{C}) \mathrm{dB} \qquad \text{(C.2.15)}$$

where C_{in} is the concentration of the source.

Base and weighting functions

For the case of flow, the weighting functions are taken as the same set as the base functions. However, in transport formulation using finite element methods with the Eulerian approach, sometimes it is advantageous to use the weighting functions which are one or two orders higher than the weighting functions: (N+1) or (N+2) upstream weighting. Here, we will only present the N+1 upstream weighting functions. Recently the N+2 weighting functions have been the subject of several investigations. The success of the N+2 weighting is still under investigation. They will not be included here. First define, for the line element, the following N+1 upstream weighting functions

$$F_1(\xi,\alpha) = N_1(\xi) - \alpha\frac{3}{4}(1 + \xi)(1 - \xi) \qquad \text{(C.2.16)}$$

$$F_2(\xi,\alpha) = N_2(\xi) + \alpha\frac{3}{4}(1 + \xi)(1 - \xi) \qquad \text{(C.2.17)}$$

where α is the weighting factor along the line from node 1 to node 2 (Figure C.3).

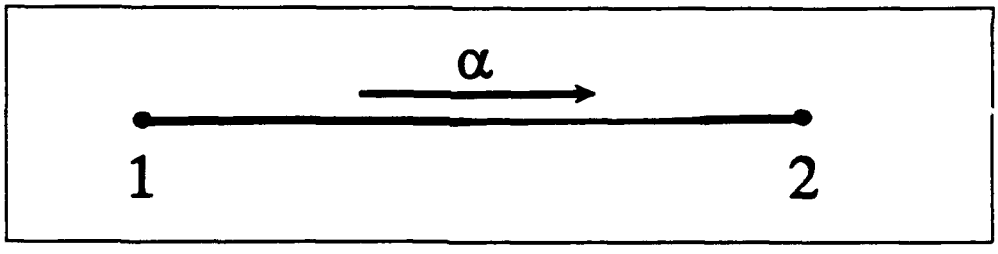


Figure C.3. Weighting factor along a line element

Then the weighting functions are obtained by an appropriate tensor product:

$$W_1 = F_1(\xi,\alpha_1)F_1(\eta,\beta_1)F_1(\zeta,\gamma_1) \qquad \text{(C.2.18)}$$

$$W_2 = F_2(\xi,\alpha_1)F_1(\eta,\beta_2)F_1(\zeta,\gamma_2) \qquad \text{(C.2.19)}$$

$$W_3 = F_2(\xi,\alpha_2)F_2(\eta,\beta_2)F_1(\zeta,\gamma_4) \qquad \text{(C.2.20)}$$

$$W_4 = F_1(\xi,\alpha_2)F_2(\eta,\beta_1)F_1(\zeta,\gamma_3) \qquad \text{(C.2.21)}$$

$$W_5 = F_1(\xi,\alpha_3)F_1(\eta,\beta_3)F_2(\zeta,\gamma_1) \quad \text{(C.2.22)}$$

$$W_6 = F_2(\xi,\alpha_3)F_1(\eta,\beta_4)F_2(\zeta,\gamma_2) \quad \text{(C.2.23)}$$

$$W_7 = F_2(\xi,\alpha_4)F_2(\eta,\beta_4)F_2(\zeta,\gamma_4) \quad \text{(C.2.24)}$$

$$W_8 = F_1(\xi,\alpha_4)F_2(\eta,\beta_3)F_2(\zeta,\gamma_3) \quad \text{(C.2.25)}$$

where α's, β's, and ψ's are the weighting factors along the side given in Figure C.4.

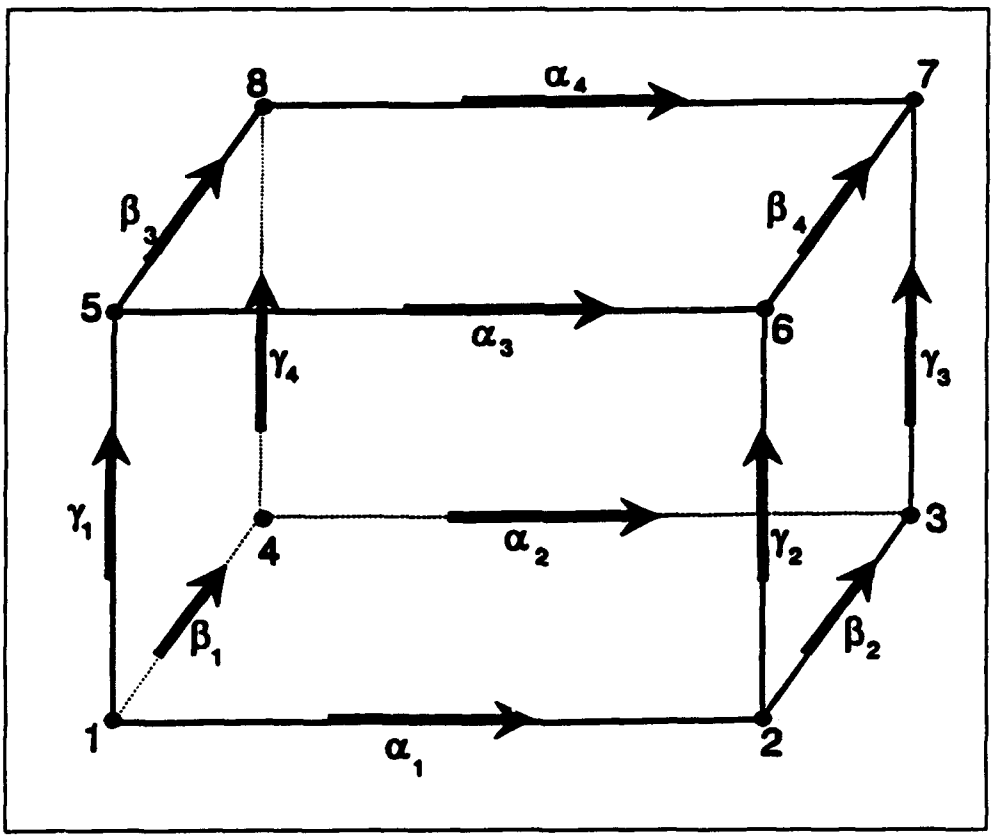


Figure C.4. Upstream weighting factors along 12 sides of a hexahedral element

Numerical integration

To complete the reduction of the partial differential equations (Equations B.2.11, B.2.15, and B.2.17) to the ordinary differential equations

(Equations C.2.5 through C.2.7), one has to evaluate the integrals on the right sides of Equations C.2.8 through C.2.15 for every element to yield the element mass matrices $[M^e]$, $[M_1^e]$, and $[M_2^e]$ and the stiff element matrices $[A^e]$, $[D^e]$, and $[K^e]$ as well as the source/sink column vector $\{Q^e\}$ and the boundary column vector $\{B^e\}$ as

$$M^e_{\alpha\beta} = \int_{R_e} N^e_\alpha \left[\theta + \rho_b \frac{dS}{dC}\right] N^e_\beta \, dR \qquad \text{(C.2.26)}$$

$$M^e_{1\alpha\beta} = \int_{R_e} N^e_\alpha \theta N^e_\beta \, dR \qquad \text{(C.2.27)}$$

$$M^e_{2\alpha\beta} = \int_{R_e} N^e_\alpha \left[\rho_b \frac{dS}{dC}\right] N^e_\beta \, dR \qquad \text{(C.2.28)}$$

$$A^e_{\alpha\beta} = \int_{R_e} N^e_\alpha V \cdot \nabla N^e_\beta \, dR \qquad \text{(C.2.29)}$$

$$D^e_{\alpha\beta} = \int_{R_e} \nabla N^e_\alpha \cdot \theta D \cdot \nabla N^e_\beta \, dR \qquad \text{(C.2.30)}$$

$$K^e_{\alpha\beta} = \int_{R_e} N^e_\alpha \left[\lambda(\theta + \rho_b \frac{dS}{dC}) + \frac{\rho^*}{\rho} q - \frac{\rho_o}{\rho} V \cdot \nabla \left[\frac{\rho}{\rho_o}\right]\right] N^e_\beta \, dR \qquad \text{(C.2.31)}$$

$$Q^e_\alpha = \int_{R_e} N^e_\alpha \left[-\lambda \rho_b \left[S + \frac{dS}{dC} C\right] + q C_{in}\right] dR \qquad \text{(C.2.32)}$$

$$B^e_\alpha = -\int_{B_e} N^e_\alpha n \cdot (-\theta D \cdot \nabla C) dB \qquad \text{(C.2.33)}$$

Following the procedures presented in the section, "Numerical integration," page C6, we first transform Equations C.2.26 through C.2.33 in terms of local coordinate. Then we integrate the resulting equations with the Gaussian quadrature. The transformation between the global and local coordinates is also given by Equations C.1.23 through C.1.25 resulting in

isoparametric elements. The surface integration from the boundary conditions also follows that presented in the section, "Numerical integration."

Mass lumping option

As with the solution of flow equations, a consistent mass matrix or mass lumping option can be used when the Eulerian approach is used. Although a consistent mass matrix option can also be used when the hybrid Lagrangian-Eulerian approach is taken, a mass lumping scheme is more appropriate and easier to implement.

Finite difference approximation in time

When the Eulerian approach is taken in approximating the governing equations, we can use either time-weighted difference or mid-difference as in Equations C.1.40 through C.1.45. However, when the Lagrangian-Eulerian approach is taken, the time integration is different from that for flow problems. Although we still have a choice of time-weighted difference or mid-difference, we prefer to using the time-weighted difference scheme. In the following we demonstrate the time integration for the Lagrangian-Eulerian approach. As in the time integration of flow equations, the boundary load vector will be ignored in the time integration of the transport equations in this section. This load vector will be discussed in the next section.

In the Lagrangian-Eulerian approach, Equations C.2.6 and C.2.7 are integrated along the characteristic lines. We will first integrate Equation C.2.6 for the case of linear isotherms. Then we will integrate Equation C.2.7 for the case of nonlinear isotherms. The time-weighted integration of Equation C.2.6 yields

$$\frac{[M]}{\Delta\tau}\left(\{C^{n+1}\} - \{C^{*}\}\right) + \omega([D]+[K])\{C^{n+1}\} + (1-\omega)([D]+[K])\{C^{*}\} = \{Q\} \qquad \text{(C.2.34)}$$

where $\Delta\tau$ is the time-step size (the determination of $\Delta\tau$ will soon become clear), $\{C^{n+1}\}$ is the concentration vector containing the concentration at all nodes at the new time n+1, and $\{C^{*}\}$ is the Lagrangian concentration vector. The Lagrangian concentration $\{C^{*}\}$ is computed by the backward method of characteristics as follows.

$$x_i^* = x_i - \int_t^{t+\Delta t} V_d dt$$

$$C_i^* = \sum_j C_j(t) N_j(x_i^*) \tag{C.2.35}$$

where x_i^* (the Lagrangian point) is the location of the fictitious particle originating at time t would arrive at the node x_i at time $t+\Delta t$, $C_j(t)$ is the value of concentration at node j at time t and $N_j(x_i^*)$ is the interpolation function associated with node j evaluated at the Lagrangian point x_i^*. If x_i^* is found to locate in the interior of the region of interest, we set $\Delta\tau$ in Equation C.2.34 as

$$\Delta\tau = \Delta t \tag{C.2.36}$$

If Δx^* is found to locate outside the region of interest, we must find a $\Delta\tau(x_i^*)$ as a function of x_i^* such that

$$x_i^* = x_i - \int_t^{t+\Delta\tau(x_i^*)} V_d dt \tag{C.2.37}$$

will locate on the boundary. Thus, it is seen that $\Delta\tau$ is less than or equal to Δt.

For the case of nonlinear isotherm, we integrated Equation C.2.7 to yield

$$\left[\frac{[M_1]}{\Delta\tau} + \frac{[M_2]}{\Delta t}\right]\{C^{n+1}\} + \omega([D]+[K])\{C^{n+1}\}$$

$$\left[\frac{[M_1]}{\Delta\tau}\{C^*\} + \frac{[M_2]}{\Delta t}\{C^n\}\right] - (1-\omega)([D]+[K])\{C^*\} + \{Q\} \tag{C.2.38}$$

The computation of $\Delta\tau$ and the Lagrangian concentrations C_* in Equation C.2.38 follows Equations C.2.35 through C.2.37 but with V_d replaced by V_f.

Numerical Implementation of Boundary Conditions

To incorporate the boundary conditions, we have to evaluate the right-hand side of Equation 3.2.9 for every boundary segment B_e to yield the load vector $\{B^e\}$:

$$B_\alpha^e = -\int_{B_e} N_\alpha^e \boldsymbol{n}\cdot(-\theta \boldsymbol{D}\cdot\nabla C)dB, \quad \alpha = 1, 2 \tag{C.2.39}$$

For the Neumann boundary condition given by Equation B.2.21, we simply substitute Equation B.2.21 into Equation C.2.33 to yield a boundary-element column vector $\{B_n^e\}$ for a Neumann segment:

$$\{B_n^e\} = \{q_n^e\} \tag{C.2.40}$$

where $\{q_n^e\}$ is the Neumann boundary flux vector given by

$$q_{n\alpha}^e = -\int_{B_e} N_\alpha^e q_N dB, \quad \alpha = 1, 2 \tag{C.2.41}$$

This Neumann boundary flux vector represents the normal fluxes through the two nodal points of the segment B_e on B_n.

For the Cauchy boundary condition given by Equation B.2.22, we may rewrite Equation C.2.33 in the following form:

$$B_\alpha^e = -\int_{B_e} N_\alpha^e \boldsymbol{n}\cdot(\boldsymbol{V}C - \theta \boldsymbol{D}\cdot\nabla C)dB + \int_{B_e} N_\alpha^e \boldsymbol{n}\cdot\boldsymbol{V}C dB, \alpha = 1,2 \tag{C.2.42}$$

The concentration on the boundary segment B_e can be approximated by

$$C = \sum_{\beta=1}^{2} C_\beta N_\beta^e \tag{C.2.43}$$

Substituting Equations B.2.22 and C.2.43 into Equation C.2.42, we obtain boundary-element column vector $\{B_c^e\}$ for a Cauchy segment:

$$\{B_c^e\} = \{q_c^e\} + [V_c^e]\{C\} \tag{C.2.44}$$

in which the Cauchy boundary flux vector $\{q_c^e\}$ and the Cauchy boundary matrix $[V_c^e]$ from the normal velocity component are given by

$$q_{c\alpha}^e = -\int_{B_e} N_\alpha^e q_C dB, \quad \alpha = 1, 2 \tag{C.2.45}$$

and

$$V_{c\alpha\beta}^e = \int_{b_e} N_\alpha^e \boldsymbol{n}\cdot\boldsymbol{V} N_\beta^e dB, \quad \alpha = 1, 2 \ \textit{and} \ \beta = 1, 2$$

Segments on which the variable boundary conditions are imposed are the flow-through boundaries on which the flow direction is not known a priori. When the flow is directed into the region, Cauchy boundary conditions will be used. The boundary-element column vector $\{B_v^e\}$ for a variable-boundary segment can be obtained similar to $\{B_c^e\}$:

$$\{B_v^e\} = \{q_v^e\} + [V_v^e]\{C\} \tag{C.2.46}$$

in which the variable-boundary flux vector $\{q_v^e\}$ and the variable-boundary matrix $[V_v^e]$ from the normal velocity component are given by:

$$q_{v\alpha}^e = -\int_{B_e} N_\alpha^e (\boldsymbol{n}\cdot\boldsymbol{V}) C_{in} dB, \quad \alpha = 1, 2 \tag{C.2.47}$$

and

$$V_{v\alpha\beta}^e = \int_{b_e} N_\alpha^e \boldsymbol{n}\cdot\boldsymbol{V} N_\beta^e dB, \quad \alpha = 1, 2 \ \textit{and} \ \beta = 1, 2$$

where C_{in} is the total dissolved concentration of the incoming fluid. When the flow is directed out from the region, both $\{q_v^e\}$ and $[V_v^e]$ are set equal to 0.

Assembling over all Neumann, Cauchy, and variable boundary segments, we obtain the global boundary column vector $\{B\}$ as:

$$\{B\} = \{q\} + [V]\{C\} \tag{C.2.48}$$

in which

$$\{q\} = \sum_{e \in N_{ne}} \{q_n^e\} + \sum_{e \in N_{ce}} \{q_c^e\} + \sum_{e \in N_{ve}} \{q_v^e\}$$

$$[V] = \sum_{e \in N_{ce}} \left[V_c^e\right] + \sum_{e \in N_{ve}} \left[V_v^e\right] \tag{C.2.49}$$

where N_{ne}, N_{ce}, and N_{ve} are the number of Neumann, Cauchy, and variable-boundary segments, respectively.

At nodes where Dirichlet boundary conditions are applied, an identity equation is generated for each node and included in the matrices of Equation C.2.34 for the case of linear isotherms or C.2.38 for the case of nonlinear isotherms. The detailed method of applying this type of boundary condition can be found elsewhere (Wang and Connor 1975).

Boundary conditions need to be implemented in the computation of the Lagrangian concentrations $\{C^*\}$. Neumann boundary conditions are normally applied to the boundary when flow is directed out from the region of interest. On the Neumann boundary, the back tracking would locate x_i^* in the interior of the domain; hence the Lagrangian concentration at the i^{th} Neumann boundary node is simply computed via interpolation. On the Dirichlet boundary nodes, the Lagrangian concentration is simply set to the specified value.

On the variable boundary, boundary conditions need not be implemented if the flow is directed out from the region. If the flow is directed into the region, the concentration of incoming fluid is specified. An intermediate concentration C^{**} is calculated according to

$$C_i^{**} = \int_{B_i} N_i V_n C_{in} dB / \int_{B_i} N_i V_n dB \ , \tag{C.2.50}$$

where C_i^{**} is the concentration due to the boundary source at the boundary node i, V_n is the normal vertically integrated Darcy's velocity, and C_{in} is the concentration of incoming fluid.

Cauchy boundary conditions are normally applied to the boundary where flow is directed into the region, where the material flux of incoming fluid is specified. The intermediate concentration is thus calculated according to

$$C_i^{**} = \int_{B_c} N_i q_c dB / \int_{B_c} N_i V_n dB \ , \qquad \text{(C.2.51)}$$

where C_i^{**} is the concentration due to Cauchy fluxes at the boundary node i, V_n is the normal Darcy's velocity, and q_c is the Cauchy flux of the incoming fluid.

The Lagrangian concentration is obtained by using the value C_i^{**} and C_i^n (the concentration at previous time-step) as follows:

$$C_i^* = \frac{\int_B N_i \theta N_j C_i^{**} dB + \int_B N \rho_b K_d N_j C_j^n dB}{\int_B N_i (\theta + \rho_b K_d) dB} \quad \textit{for linear isotherm} \qquad \text{(C.2.52)}$$

$$C_i^* = C_i^{**} \quad \textit{for nonlinear isotherm} \qquad \text{(C.2.53)}$$

Solution of the Matrix Equations

Because the Lagrangian-Eulerian approach results in a symmetric positive definite matrix, the system of the algebraic equations can be solved by any of the four options: the block iteration, the successive point iteration, the polynomial preconditioned conjugate gradient, and the incomplete Cholesky preconditioned conjugate gradient methods. For the Eulerian approach, however, the block iteration and successive point iteration methods are the preferred choice for solving the matrix equation. Especially when the advection transport is dominant, the two basic iteration methods with under-relaxation are very effective in reducing the number of iterations required for a convergent solution (Yeh 1985).

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	June 1994	Final report

4. TITLE AND SUBTITLE

3DSALT: A Three-Dimensional Finite Element Model of Density-Dependent Flow and Transport Through Saturated-Unsaturated Media

5. FUNDING NUMBERS

6. AUTHOR(S)

Gour-Tsyh Yeh, Jing-Ru Chang, Jin-Ping Gwo,
Hsin-Chi Lin, David R. Richards, William D. Martin

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Department of Civil Engineering, Pennsylvania State University,
University Park, PA 16802, and
U.S. Army Engineer Waterways Experiment Station,
3909 Halls Ferry Road, Vicksburg, MS 39180-6199

8. PERFORMING ORGANIZATION REPORT NUMBER

Instruction Report
HL-94-1

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

U.S. Army Engineer District, Wilmington
P.O. Box 1890
Wilmington, NC 28402-1890

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

Available from National Technical Information Service, 5285 Port Royal Road, Springfield, VA 22161.

12a. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for public release; distribution is unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

This report presents the user's manual of 3DSALT, A Three-Dimensional Finite Element Model of Density-Dependent Flow and Transport Through Saturated-Unsaturated Media. The model is designed for generic application. The model is developed specifically for solving seawater intrusion problems in subsurface media. It consists of a flow module and a transport module. In comparison to conventional finite element (including both Galerkin and upstream finite elements) or finite difference (including both central and upwind finite differences) models, the transport module of 3DSALT offers several advantages: (a) it completely eliminates numerical oscillation due to advection terms, (b) it can be applied to mesh Peclet numbers ranging from 0 to infinity (conventional finite element or finite difference models typically impose unduly severe restriction on the mesh Peclet number), (c) it can use very large time-step sizes to greatly reduce numerical dispersion (in fact, the larger the time-step, the better the solution with respect to advection transport; the time-step size is limited only by the accuracy requirement with respect to diffusion/dispersion transport, which is normally not a very severe restriction), and (d) the hybrid Lagrangian-Eulerian finite element approach is always superior to and will never be worse than its corresponding upstream finite element method. Because of these advantages, 3DSALT is ideal for simulating density-dependent flow and advection-dominant transport.

(Continued)

14. SUBJECT TERMS		15. NUMBER OF PAGES
Density-dependent flow	Lagrangian-Eulerian approach	178
Finite element	Saturated-unsaturated flow	16. PRICE CODE
Groundwater		

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED		

13. (Concluded).

For each site-specific application, 58 control integers must be assigned using the parameter statement in the MAIN program. In addition, if the material properties are specified by analytical functions, the subroutine SPROP must be modified by the users. Sources/sinks and boundary values as functions of time can also be specified by analytical functions. Under such circumstances, the users should provide such functions in subroutines ESSFCT, WSSFCT, DBVFCT, VBVFCT, CBVFCT, and NBVFCT.

Input to the program includes the control indices, properties of the media either in tabular or analytical form, the geometry in the form of elements and nodes, and boundary and initial conditions either in tabular or analytical form. Principal output includes the spatial distribution of pressure head, total head, moisture content, Darcy velocity components, concentration, and material fluxes at any desired time-step. Fluxes through various types of boundaries are output. In addition, diagnostic variables, such as the number of nonconvergent nodes and residuals, may be printed if desired for debugging purposes.

Appendix A presents a data input guide for site-specific application. Appendix B provides the physical bases and mathematical foundation for describing density-dependent flow and material transport. Appendix C gives the numerical detail in approximating the governing equations.