REPORT DOCUMENTATION FORM
WATER RESOURCES RESEARCH CENTER
University of Hawaii at Manoa

[1]Report Number Technical Report No. 125	[2]FCST Category III-C, IV-B, V-E
[3]Title *Numerical Modeling of Liquid Waste Injection into a Two-Phase Fluid System*	[4]Report Date August 1979
	[5]No. of Pages ix + 103
	[6]No. of Tables 2 / [7]No. of Figures 21
[8]Author(s) Stephen W. Wheatcraft, Frank L. Peterson	[9]Grant or Contract Agency Office of Water Researcn and Technology, U.S. Dept. of Interior
	[10]Grant or Contract No. A-071-HI 14-34-0001-7025, 7026, 8013

[11]Keywords *Decriptors:* *Waste disposal, *Injection, Groundwater, Hydrodynamics, Dispersion, Numerical analysis, Hawaii.
Identifiers: *Sandbox model, Fluid transport equation, convective dispersion equation.

[12]Abstract (Purpose, method, results, conclusions)

The injection of liquid wastes into a groundwater environment saturated with density-stratified fluid is simulated by a finite-difference numerical model. The fluid transport equation is simultaneously solved with the convective-dispersion equation for salinity. The migration of the injected liquid waste effluent is then tracked by solving a second convective-dispersion equation for an ideal tracer dissolved in the effluent. The convective-dispersion equation for the ideal tracer is solved with the flow velocities obtained from the simultaneous solution of the fluid transport and the salinity convective-dispersion equations. The equations are solved for the two-dimensional case of a line of injection wells set close together parallel to the coastline. Total length of the line of injection wells is considered to be much longer than the distance to the ocean so that any vertical cross section taken normal to the coastline will appear the same. Results are presented in a time-series of contour maps in the vertical plane: one map for each time-step, with lines of equal concentration for the salinity (isochlors); and the effluent tracer (isopleths). The more concentrated effluent is found to migrate vertically upward around the injection well due to buoyant force, while dilute effluent solutions migrate horizontally, displaying very little buoyant rise.

2540 Dole Street, Holmes Hall 283 • Honolulu, Hawaii • U.S.A.

NUMERICAL MODELING OF LIQUID WASTE INJECTION INTO A TWO-PHASE FLUID SYSTEM

by

Stephen W. Wheatcraft
Frank L. Peterson

Technical Report No. 125

August 1979

Project Completion Report
for
Dynamics of Waste Injection into a Two-Phase Fluid System, Phase II

OWRT Project No. A-071-HI
Grant Agreement Nos.: 14-34-0001-7025, 7026, 8013
Principal Investigator: Frank L. Peterson
Project Period: 1 October 1977 to 30 September 1978

The work upon which this publication is based was supported in part by funds provided by the Office of Water Research and Technology, U.S. Department of the Interior, Washington, D.C., as authorized by the Water Research and Development Act of 1978.

ABSTRACT

The injection of liquid wastes into a groundwater environment saturated with density-stratified fluid is simulated by a finite-difference numerical model. The fluid transport equation is simultaneously solved with the convective-dispersion equation for salinity. The migration of the injected liquid waste effluent is then tracked by solving a second convective-dispersion equation for an ideal tracer dissolved in the effluent. The convective-dispersion equation for the ideal tracer is solved with the flow velocities obtained from the simultaneous solution of the fluid transport and the salinity convective-dispersion equations. The equations are solved for the two-dimensional case of a line of injection wells set close together parallel to the coastline. Total length of the line of injection wells is considered to be much longer than the distance to the ocean so that any vertical cross section taken normal to the coastline will appear the same. Results are presented in a time-series of contour maps in the vertical plane: one map for each time-step, with lines of equal concentration for the salinity (isochlors); and the effluent tracer (isopleths). The more concentrated effluent is found to migrate vertically upward around the injection well due to buoyant force, while dilute effluent solutions migrate horizontally, displaying very little buoyant rise.

CONTENTS

ABSTRACT. . . . . *v*

INTRODUCTION. 1

Background 1

Objectives 2

Conduct of Study 2

MATHEMATICAL FORMULATION. 3

Nature of Problem. 3

Basic Governing Equations. 5

Assumptions. 7

Boundary and Initial Conditions. 10

FORMULATION OF GOVERNING EQUATION AND BOUNDARY
CONDITIONS INTO A NUMERICAL MODEL 15

Solving Coupled Fluid Transport and
Convective-Dispersion Equations. 15

Choice of Methods for This Study 19

Explanation of IADI and ADI Methods. 21

Modifications Necessary for Solving Coupled Equations. 25

DISCUSSION OF NUMERICAL RESULTS 26

Accuracy 26

Errors in Vertical Velocity Calculation. 26

Stability and Convergence. 29

Efficiency 31

RESULTS OF SIMULATIONS. 31

Verification of Numerical Model with
Existing Numerical and Analytic Models 31

Nomenclature Definition of Three Types of Aquifer Fluids 32

Summary of Numerical Simulations 32

Presentation of Results. 34

Description of H-1 Simulation. 37

Description of F-1 Simulation. 39

Comparison of F Series of Simulations. 44

Lower Dispersivity, Anisotropy, and Field-Size Dimension 47

COMPARISON OF NUMERICAL SIMULATIONS TO SANDBOX-MODEL EXPERIMENTS. . . . 48
Differences Between Sandbox Model and Numerical Simulation 48
Determining Relative Importance of Dispersion. 51
Similarities Between Sandbox Model and Numerical Simulation. 53

SUMMARY OF CONCLUSIONS AND RECOMMENDATIONS. 56
Conclusions. 56
Recommendations for Future Research. 57

ACKNOWLEDGMENTS 59

GLOSSARY OF SYMBOLS 60

REFERENCES. 63

APPENDICES. 65

FIGURES

1. Hydrogeology and Resulting Geometric Boundary Conditions for Hypothetical Injection Site 11
2. Initial Salinity Values for Saltwater Intrusion Problem. 13
3. Numerical Dispersion and Overshoot 16
4. Regions of High Salinity Gradient. 28
5. Effects of Truncation Error on Vertical Velocities 28
6. Comparison of Previous with Present Solutions to Sea-Water Intrusion Problem. 31
7. Velocity Vectors for Steady-State Solution of Saltwater Intrusion Problem for H-1 Simulation 35
8. Velocity Vectors for Steady-State Solution of Injection Phases for H=1 and F-1 Simulations 35
9. Time-Series of Salinity and Effluent Tracer Concentrations for H-1 Simulation. 36
10. Relative Proportions of Effluent, Saline, and Fresh Water as Function of Depth at Ocean-Aquifer Interface for H-1 Simulation. . 37
11. Steady-State Preinjection Isochlors for H-1, F-1, F-2, and F-3 Simulations. 40
12. Time-Series of Salinity and Effluent Tracer Concentrations for F-1 Simulation. 41
13. Positions of Stagnation Streamline and 10% Isochlor for F-1 Simulation. 42

14. Relative Proportions of Saline, Effluent, and Fresh Water as Function of Depth at Ocean-Aquifer Interface for F=1 Simulation. . 43

15. Relative Proportions of Saline, Effluent, and Fresh Water as Function of Depth at Ocean-Aquifer Interface for F-1, F-2, and F-3 Simulations 45

16. Time-Series of Salinity and Effluent Tracer Concentrations for F-2 Simulation. 46

17. Time-Series of Salinity and Effluent Tracer Concentrations for F-3 Simulation. 47

18. Difference in Pressure for Saltwater and Mixed Saltwater Heads . . 49

19. Comparison of Ambient Density Effect on Effluent Tracer. 54

20. Downstream Breakthrough Curves for Sandbox-Model Experiments and Numerical Simulations. 55

21. Relative Proportions of Saline, Effluent, and Fresh Water as Function of Depth at Ocean-Aquifer Interface for Sandbox-Model Experiment 56

TABLES

1. Geometrical and Aquifer Constants. 33

2. Parameters Varied in Simulations 34

INTRODUCTION

Background

Over the past several years the practice of waste disposal by subsurface injection has become increasingly widespread in Hawai'i. Peterson and Lau (1974) have described the mechanics and some of the problems associated with subsurface injection into the Hawaiian groundwater body, and Takasaki (1974) has inventoried the extent of the waste injection practices in Hawai'i. Since the publication of Takasaki's inventory in 1974 numerous additional waste injection facilities have gone into operation.

The single most obvious advantage of subsurface injection over other waste disposal alternatives is economics: the cost (at least capital cost) is considerably less than for most other methods of liquid waste disposal. However, there are also serious potential—and in some cases actual—problems associated with injection of liquid wastes. The two basic kinds of problems involve: (1) possible contamination of potable groundwater and shallow near-shore coastal waters, and (2) at the other end of the spectrum, clogging in the immediate vicinity of the injection wells to the extent that injection capacity is severely restricted.

Clogging has been a very common problem at many of the small injection facilities in Hawai'i and currently is being investigated by one of the authors (Petty and Peterson 1979). Contamination of either potable groundwater supplies or near-shore coastal waters is not known to be a problem at the present time, but remains a very real potential problem as more and larger injection facilities continue to be put into operation. Considerable investigation by the authors and others using physical laboratory models has been undertaken (Wheatcraft, Peterson, and Heutmaker 1976; Heutmaker, Peterson, and Wheatcraft 1977; Williams 1977). The laboratory work, which has utilized the sandbox and Hele-Shaw models, has resulted in many interesting findings; however, all of this work has one common limitation. That is, it is impossible to accurately model effects of hydrodynamic dispersion with the sandbox and Hele-Shaw models. Furthermore, the boundary conditions imposed by the Hele-Shaw and especially the sandbox model, considerably restrict the type of injection situations that could be modeled.

Consequently, the present investigation utilizing mathematical modeling was undertaken. This investigation will allow the study of hydrodynamic dispersion as well as enabling a much wider range of injection problems and conditions to be considered.

Objectives

The overall objective of this study was to obtain a better understanding of the process of liquid waste injection in Hawai'i through use of mathematical modeling. To do this, the following specific questions were investigated.

1. What is the distribution of effluent and salinity in space and time?
2. To what extent does hydrodynamic dispersion affect the effluent and salinity distribution?
3. What is the relative significance on the injection process of various aquifer and injection well parameters, such as formation dispersivity and anisotrophy, ambient flowfield strength, formation fluid and effluent densities, and injection rate and depth.

Conduct of Study

In order to achieve the objectives described above, the physical situation must first be stated as a mathematical problem with the appropriate set of governing equations and boundary and initial conditions. Then, the resulting equations must be transformed into a suitable numerical scheme, in this case using the finite difference method. Next, to assure that the resulting finite difference program correctly solves the governing equations, the problem is reduced to simple cases for which an analytic solution or a previous numerical solution is already known, and the results from the simplified program and the known solution are compared.

It is necessary to calibrate the physical parameters of the numerical model to have predictive value for a specified field situation. Ideally, the calibration procedure would involve using extensive field data collected over a period of years, however, for Hawai'i no such data exists. Therefore, the results from the numerical model are compared to the results obtained from previous sandbox modeling studied conducted by the authors (Wheatcraft, Peterson, and Heutmaker 1976; Heutmaker, Peterson, and Wheatcraft 1977). The dispersivity parameters used in comparing the numerical results to the sandbox model are shown to be much higher than those in the sandbox model, but much lower than those commonly experienced in field studies. This is because experience has shown that field values calculated for dispersivity are nearly always higher than laboratory values calculated for the same material.*

Because there is no field calibration procedure applied to this model, it should not be regarded as a *predictive* model, but rather as an *interpretive* model. An interpretive model provides useful information that helps in understanding the various physical processes that contribute to a system but claims no accurate predictive value. A primary value of the model developed here lies in being able to satisfy in a general way the objectives posed earlier in this chapter. A second value is that the model simulations can be used in future field studies to help determine the optimum location, spacing, and depth of injection wells, and in addition can be used to help determine the optimum spatial distribution and depth of monitoring sites around the injection well(s).

MATHEMATICAL FORMULATION

Nature of Problem

In simplest terms, the problem being investigated involves the emplacement of liquid effluent into, and the subsequent migration of the effluent through a Ghyben-Herzberg groundwater body. In this case the effluent contains solute, but in concentrations so small that its density is comparable to the density of fresh water. The ambient groundwater body into which the effluent is injected also contains solute, but in concentrations which increase as a function of depth. Thus near its top the salinity and density of the ambient groundwater body is comparable to that of fresh water; however, with increasing depth the salinity and density of the ambient groundwater body eventually approaches that of sea water.

Several transport mechanisms exert significant influence on a fluid and the solutes contained in it, which move through a porous medium. In this investigation only those mechanisms which play an important role in transporting effluent from an injection well and the resultant interactions with ambient fluid in a Ghyben-Herzberg lens system will be considered. The fluid itself (both injected effluent and ambient groundwater) is strictly

*J. Bear 1977: personal communication.

transported by convection, which in the field of groundwater hydrology is used to describe both horizontal and vertical fluid displacement.

In addition to the fluid, there are two solutes whose transport must also be described: the solute of the injected effluent, and the "salt" or "salinity" of the ambient groundwater. In general, three important mechanisms are involved in the transport of any solute in a porous medium: the primary means is by convection in a moving fluid; other mechanisms are mechanical dispersion and molecular diffusion which are generally lumped together and referred to as hydrodynamic dispersion, or simply dispersion. Unless a reference is made specifically to one or the other, the term dispersion will be used to include both mechanical dispersion and molecular diffusion. For a detailed review of the phenomenon of dispersion, see Bear (1972, chap. 10).

It is recognized that injected effluent contains many different types of dissolved solids as well as suspended solids. However, in this study the assumption is made that the effluent is composed of fluid and dissolved solids that do not react with the porous medium and have no time-dependent decay rates. The injected fluid is therefore treated as containing one solute which behaves in a characteristic manner for all of the solutes dissolved in the effluent. This characteristic solute is assumed to be in low enough concentrations that it does not significantly affect the density of the fluid. In other words, the solution of the fluid transport equation is not dependent on the concentration of effluent solute, therefore the two equations for fluid transport and effluent solute transport can be independently solved. The problem of suspended solids is not treated here because of their tendency to adhere to the porous medium itself and thus cause reduction in porosity and intrinsic permeability. This is the problem of "aquifer clogging" and should be treated separately.

Treatment of the second solute, the salinity of the ambient groundwater, is a much more complicated problem. This second solute is present as a result of the intrusion and mixing of sea water with the resident fresh water of the aquifer. For this study, the sea water can be thought of as fresh water containing one solute with a concentration equal to the total dissolved solids (TDS) present in sea water. An average value for the TDS in sea water is about 35.8‰ (35.8 parts per thousand) (Weast 1972). The salinity contained in the fluid can therefore range from 0 to nearly 36‰.

This range of salinity significantly affects the density of the fluid and thus adds a buoyant term to the convective movement of the fluid. This means that the solution to the fluid transport equation is dependent upon knowing the salinity; but the solution of the salinity solute transport equation in turn depends upon knowing the solution to the fluid transport equation. Thus the two equations are coupled and must be simultaneously solved.

Basic Governing Equations

For the problem of waste injection into porous media containing density-stratified fluid as described above, there are five unknowns, requiring five equations. The unknowns are fluid density (ρ), salinity concentration (C), pressure (P), specific discharge ($\underset{\sim}{q}$), and effluent tracer concentration (T). Two more unknowns ($\underset{\sim}{q}_C$ and $\underset{\sim}{q}_T$, specific discharges of the respective salinity and effluent solutes are briefly discussed and it is shown how they can be eliminated. The fluid transport equation previously referred to is really a combination of two equations which represent conservation of momentum (Darcy's law) and conservation of mass (continuity) for a fluid flowing through porous media. It is necessary to write a conservation of mass equation for the salinity and an equation of state which relates how the salinity concentration affects the density of the fluid. Finally, a mass conservation equation for the effluent tracer must be written. Since the effluent tracer does not affect the fluid density, an equation of state for the tracer is not needed.

The five equations are:

Momentum Equation (Darcy's law),

$$\underset{\sim}{q} = -\underset{\approx}{k}/\mu \; (\nabla P + \rho g \nabla z); \tag{1}$$

Conservation of Mass for the Fluid,

$$\nabla \cdot \rho \underset{\sim}{q} = -\frac{\partial n\rho}{\partial t} - Q(\underset{\sim}{X}^*); \tag{2}$$

Conservation of Mass for the Salinity,

$$\nabla \cdot C\underset{\sim}{q}_c = -\frac{\partial nC}{\partial t}; \tag{3.1}$$

Equation of State,

$$\rho = \rho \quad [1 + \beta_c \; (C - C_o)]; \tag{4}$$

Conservation of Mass for the Effluent Tracer,

$$\nabla \cdot T\underset{\sim}{q}_T = - \frac{\partial nT}{\partial t} - Q_T(\underset{\sim}{X}^*) \; . \tag{5.1}$$

Two new variables have been introduced here: $\underset{\sim}{q}_c$, the velocity of the salinity solute, and $\underset{\sim}{q}_T$, the velocity of the effluent tracer solute. to show that the velocity of a solute is not the same as the average fluid velocity because of hydrodynamic dispersion. These two unknowns can be eliminated through constitutive relationships developed from continuum mechanics. These relationships are discussed by Bear (1972, pp. 102-104) and by Pinder and Gray (1977) and are given by the following equations:

$$C \, \underset{\sim}{q}_c = C \, \underset{\sim}{q} - \underset{\approx}{D} \, \nabla \, C \; , \tag{6}$$

and

$$T \, \underset{\sim}{q}_T = T \, \underset{\sim}{q} - \underset{\approx}{D} \, \nabla \, T \; . \tag{7}$$

Substituting equations (6) and (7) into (3.1) and (5.1), we have

$$\nabla \cdot C \, \underset{\sim}{q} - \nabla \cdot \underset{\approx}{D} \; \nabla \; C - \frac{\partial nC}{\partial t} = 0 \;\; , \tag{3.2}$$

and

$$\nabla \cdot T \, \underset{\sim}{q} - \nabla \cdot \underset{\approx}{D} \; \nabla \; T + \frac{\partial nT}{\partial t} = - \, Q_T(\underset{\sim}{X}^*) \;\; . \tag{5.2}$$

Thus, $\underset{\sim}{q}_c$ and $\underset{\sim}{q}_T$ are eliminated and (3) and (5) are written in more familiar forms, generally referred to as convective-dispersion equations.

The source term, $Q(\underset{\sim}{X}^*)$ in (2) is defined as follows:

$$Q(\underset{\sim}{X}^*) = \left[\frac{\partial}{\partial t} \int_{\mathcal{V}} \rho \; d \; V + \rho_o \int_S \underset{\sim}{q} \cdot \underset{\sim}{n} \; d \; s\right] \; \delta(x - x_w, \; z - z_w) \; . \tag{8}$$

Similarly, the source term in equation (5) is defined as:

$$Q_T(\underset{\sim}{X}^*) = \left[\frac{\partial}{\partial t} \int_{\mathcal{V}} T d\mathcal{V} + T_o \int_S \underset{\sim}{q} \cdot \underset{\sim}{n} \; d \; s\right] \;\; \delta(x - x_w, \; z - z_w) \; . \tag{9}$$

Assumptions

Six assumptions were made in setting up this model and their effects on the governing equations are discussed below.

1. Density-dependent flow is, in general, transient and three-dimensional in nature. In this study, solution is limited to the two-dimensional case. This limitation is largely accepted for logistical reasons. In principle, the three-dimensional case is solved by extension of the methods used to solve the two-dimensional problem, however, it requires much more computer time and storage and, hence, proportionally raises the cost of simulation. The two-dimensional case roughly corresponds physically to a field situation in which a group of wells is spaced closely enough together in a linear array so that they may be considered a horizontal line source. It exactly corresponds to a horizontal injection tunnel that is long, relative to its distance from the coast.

2. In the formulation of the groundwater quality problem, $\rho = \rho(C, P)$ and the term $\partial(n\rho)/\partial t$ on the right-hand side of (2) is properly expanded:

$$\frac{\partial}{\partial t}(np) = \frac{\partial n}{\partial t} + n\frac{\partial \rho}{\partial P}\frac{\partial P}{\partial t} + \frac{\partial \rho}{\partial C}\frac{\partial C}{\partial t} \; ;$$

$$\frac{\partial}{\partial t} = \rho_o \beta_c \tag{10.1}$$

The three terms on the right-hand side of (10.1) represent, respectively, the compressibility of the aquifer, the compressibility of the fluid, and the time rate of density change of the fluid due to change in salinity concentration. The first two of these terms are very small compared to the third and will be ignored in this study. In some parts of the aquifer region under study, the third term is also negligible compared to the left-hand side of (2), but in other regions it is not, especially around the injection well. Furthermore, the areas in which the third term is not negligible change as salinity is convected away from the well. Therefore the third term must in general be included everywhere in the solution space. Hence (10.1) can be simplified to

$$\frac{\partial}{\partial t}(n\rho) = n\frac{\partial \rho}{\partial C}\frac{\partial C}{\partial t} = n\rho_o\beta_C\frac{\partial C}{\partial t} \; . \tag{10.2}$$

3. In Hawai'i, the aquifers are known to be anisotropic in the x-z plane, with $k_x > k_z$. The aquifers are also anisotropic in the x-y plane,

but much less than in the x-z plane. Estimates of the ratio k_x/k_z generally range from 10 to 100. Although no quantitative work has been done to determine this ratio, anisotropy must be taken into account when dealing with the x-z (vertical) plane since it has the potential of affecting both dispersive and convective mass transport. Thus, vertical anisotropy will be assumed for this study.

4. The intrinsic permeability is a second-rank tensor that is constant with respect to space and time, but directional in nature such that

$$\underset{\approx}{k} = \begin{bmatrix} k_x & 0 \\ 0 & k_z \end{bmatrix}$$

where the coordinate axes are taken to be the principal directions. This assumption introduces a slight error for basaltic regions because the lava flows dip at an angle of about 7 to 9° from the horizontal. This causes the principal directions to be tilted at the same angle, 7 to 9° from the coordinate axes. This error is relatively small compared to the precision to which values for the intrinsic permeability are known and is therefore ignored.

5. Although the tensor nature of the intrinsic permeability is well understood for the general anisotropic porous medium, this is not the case for the coefficient of hydrodynamic dispersion ($D_{ij} = \underset{\approx}{D}$). D_{ij} is known to be a second-rank tensor even in isotropic media with D_{xx} colinear with the specific discharge vector, and D_{zz} normal to the specific discharge vector. However, the exact form of the function of D_{ij} for anisotropic porous media is not well known and very little work has been done in this area.* Therefore, for the purposes of this study, it is assumed that the porous media is isotropic with respect to hydrodynamic dispersion, even though it is being treated as anisotropic with respect to permeability. For isotropic media,

$$D_{ij} = D_{m\,ij} + D^*_{d\,ij} \ , \tag{11}$$

where

$$D_{m\,ij} = a_t \bar{V} \delta_{ij} + (a_L - a_t) \frac{\bar{V}_i \bar{V}_j}{\bar{V}} \ , \tag{12}$$

*J. Bear 1977: personal communication.

and

$$D^*_d\ ij = D_d\ T^*_{ij}\ . \tag{13}$$

The tortuosity (T^*_{ij}), which is related to the porous medium's intrinsic permeability and is a second-rank tensor, reduces to a scalar (unlike the dispersion coefficient) in an isotropic medium. Since the tortuosity is being used here in an isotropic treatment of dispersion, it may be considered a scalar.

6. Injection is assumed to take place from a point source and the injection pressure, rather than the injection rate, is specified at the well. The injection rate can of course be calculated during the simulation by a divergence-type relationship, and this was done. The relationship used was

$$\left(\frac{\partial \rho \mu}{\partial \mu} + \frac{\partial \rho V}{\partial z}\right)_{\substack{x=x_w \\ z=z_w}} = Q\ (\underset{\sim}{X}^*)\ . \tag{14}$$

With the above considerations, the governing equations (1) through (5) reduce to

Darcy's Law:

$$U = \frac{-k_x}{\mu}\frac{\partial P}{\partial x}\ , \tag{15.1}$$

$$V = \frac{-k_z}{\mu}\left(\frac{\partial P}{\partial z} + \rho g\right)\ ; \tag{15.2}$$

Conservation of Mass:

$$\rho\left(\frac{\partial U}{\partial x} + \frac{\partial V}{\partial z}\right) + U\ \frac{\partial \rho}{\partial x} + V\ \frac{\partial \rho}{\partial z} = -n\ \rho_o \beta_c\ \frac{C}{t}\ ; \tag{16}$$

Convective-Dispersion Equation for Salinity:

$$U\frac{\partial C}{\partial x} + V\frac{\partial C}{\partial z} + \frac{n\partial C}{\partial t} + \frac{\partial}{\partial x}\left(D_{xx}\frac{\partial C}{\partial x} + D_{xz}\frac{\partial C}{\partial z}\right) + \frac{\partial}{\partial z}\left(D_{zx}\frac{\partial C}{\partial x} + D_{zz}\frac{\partial C}{\partial z}\right)\ . \tag{17}$$

Equation of State:

$$\rho = \rho_o\ [1 + \beta_c\ (C - C_o)]\ ; \tag{18}$$

Convective-Dispersion Equation for Tracer Effluent:

$$U\frac{\partial T}{\partial x} + V\frac{\partial T}{\partial z} + \frac{\partial T}{\partial t} = \frac{\partial}{\partial x}\left(D_{xx}\frac{\partial T}{\partial x} + D_{xz}\frac{\partial T}{\partial z}\right) + \frac{\partial}{\partial z}\left(D_{zx}\frac{\partial T}{\partial x} + D_{zz}\frac{\partial T}{\partial z}\right)\ . \tag{19}$$

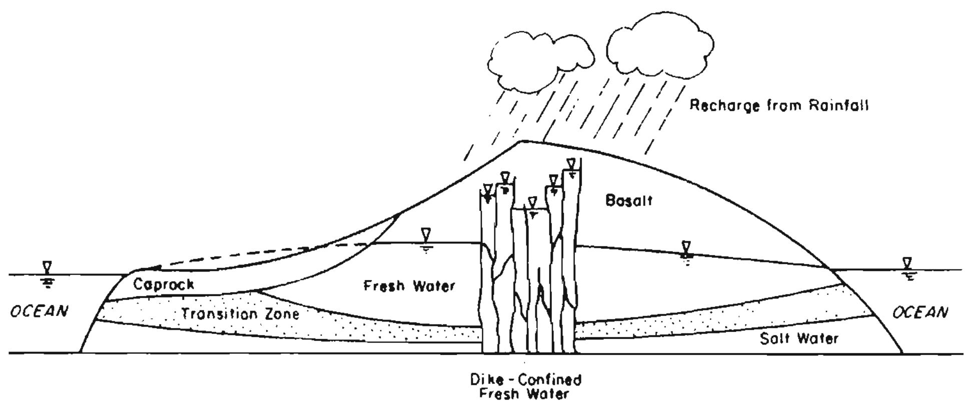
In summary, there are five unknowns ($\underset{\sim}{q}$, P, C, ρ, and T) and five corresponding governing equations ([15] to [19]). Equations (15), (16), and (18) provide the solution to the fluid transport, and (17) provides the solution for the salinity transport. The solution of each of these two sets of equations ([15], [16], and [17], [18]) is dependent upon knowing the solution to the other set. These are the coupled sets of equations which must be simultaneously solved and which completely govern the flow field. Once the flow field has been computed for a given time step, the computed specific discharges are used to solve equation (19), which provides the solution for the effluent solute transport, for the same time step. Although solution of (19) is dependent upon solution of the other four equations, the reverse is not true because the effluent tracer solute, T, is assumed to be an ideal tracer in which the tracer's concentration does not affect the density (and, therefore, does not affect the flow field) of the aquifer fluid.

In equations (15) through (19), we now have a mathematical description of the fluid and salinity transport. To complete the description of the mathematical problem statement, the initial conditions of the system must be defined, and the behavior of the system on the boundaries of the solution domain described.

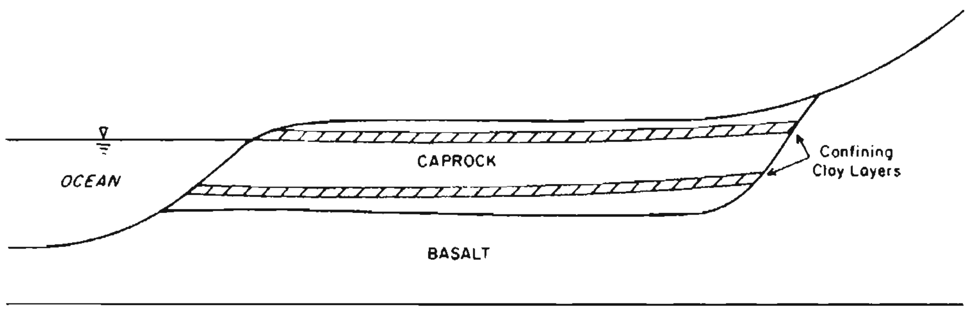
Boundary and Initial Conditions

The geometrical boundary conditions are determined by the geology of the particular injection site under consideration; however, the intent of this study is to produce an interpretive model (not a predictive model for a specific situation) of a hypothetical injection site to understand the various physical processes and their effects on the transport of salinity and effluent tracer. Figure 1.1 is a cross section of a typical volcanic dome (such as O'ahu) and its groundwater occurrence. Most waste water injection takes place in the coastal caprock area which is shown in Figure 1.1 in relation to the entire island hydrogeology, and in Figure 1.2 in greater detail. Figure 1.3 is an idealized version of this caprock which illustrates the geometrical boundary conditions that are employed for this model.

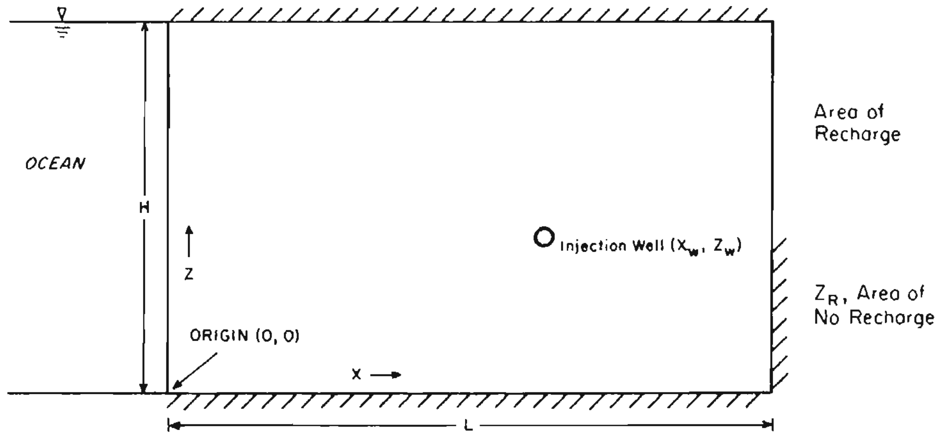
The initial conditions at the start of injection are not straightforward. It is assumed that the caprock has undergone salt-water intrusion and that this intrusion has reached a steady-state condition. The salinity



1.1. Cross Section of a Typical Volcanic Dome Showing Hydrogeology



1.2. Close-up View of Caprock Geometry



1.3. Idealized Illustration of Caprock Geometry

FIGURE 1. HYDROGEOLOGY AND RESULTING GEOMETRIC BOUNDARY CONDITIONS FOR HYPOTHETICAL INJECTION SITE

distributions resulting from the salt-water intrusion are determined by assuming that the caprock starts with an artificial "pristine" condition, with a nearly sharp interface between water with essentially no salinity on top, and water beneath that essentially has a salinity the same as sea water. The simulation is then carried through until steady-state distributions of salinity are achieved; thus, the initial conditions for the injection are provided by the steady-state solution to the salt-water intrusion problem. This first phase of the simulation, which provides the steady-state salinity distribution due to salt-water intrusion, will be referred to as the intrusion phase. The second portion of the simulation begins with the start of injection (after the steady-state salinity distributions have been obtained from the intrusion phase), and is referred to as the injection phase.

The initial conditions for the intrusion phase are

$$C = C(z,\ 0)\,, \tag{20}$$

where

$$C(z,\ 0) = \tfrac{1}{2}\,[A \tan^{-1}(z - H/2) + C_S]\ . \tag{21}$$

This equation is graphed in Figure 2, and closely resembles the shape of a typical salinity concentration curve in a relatively undisturbed Ghyben-Herzberg lens.

The salinity boundary conditions for both the salt-water intrusion and the injection phase (Fig. 1) of the simulation are as follows:

$$\frac{\partial C}{\partial x}(0,\ z,\ t) = 0, \text{ where } U(0,\ z,\ t) < 0\ , \tag{22.1}$$

$$C(0,\ z,\ t) = C_S, \quad \text{where } U(0,\ z,\ t) \geqslant 0\ , \tag{22.2}$$

$$C(L,\ z,\ t) = C_S, \quad \text{where } U(L,\ z,\ t) = 0\ , \tag{22.3}$$

$$C(L,\ z,\ t) = 0, \quad \text{where } U(L,\ z,\ t) = 0\ , \tag{22.4}$$

$$\frac{\partial C}{\partial z}(x,\ 0,\ t) = \frac{\partial C}{\partial z}(x,\ H,\ t) = 0\ ,$$

where

$$V(x,\ 0,\ t) = V(x,\ H,\ t) = 0\ . \tag{22.5}$$

Equations (22.1) and (22.2), respectively, state that as long as fluid is leaving the aquifer through the outflow face into the ocean (negative U), there is a condition of zero diffusive mass flux, but if the fluid is entering the aquifer from the ocean (positive U), then it convects salinity into the aquifer at the same concentration that exists in sea water. Equations (22.3), (22.4), and 22.5) allow no salinity to enter or leave the aquifer from any of the other boundaries.

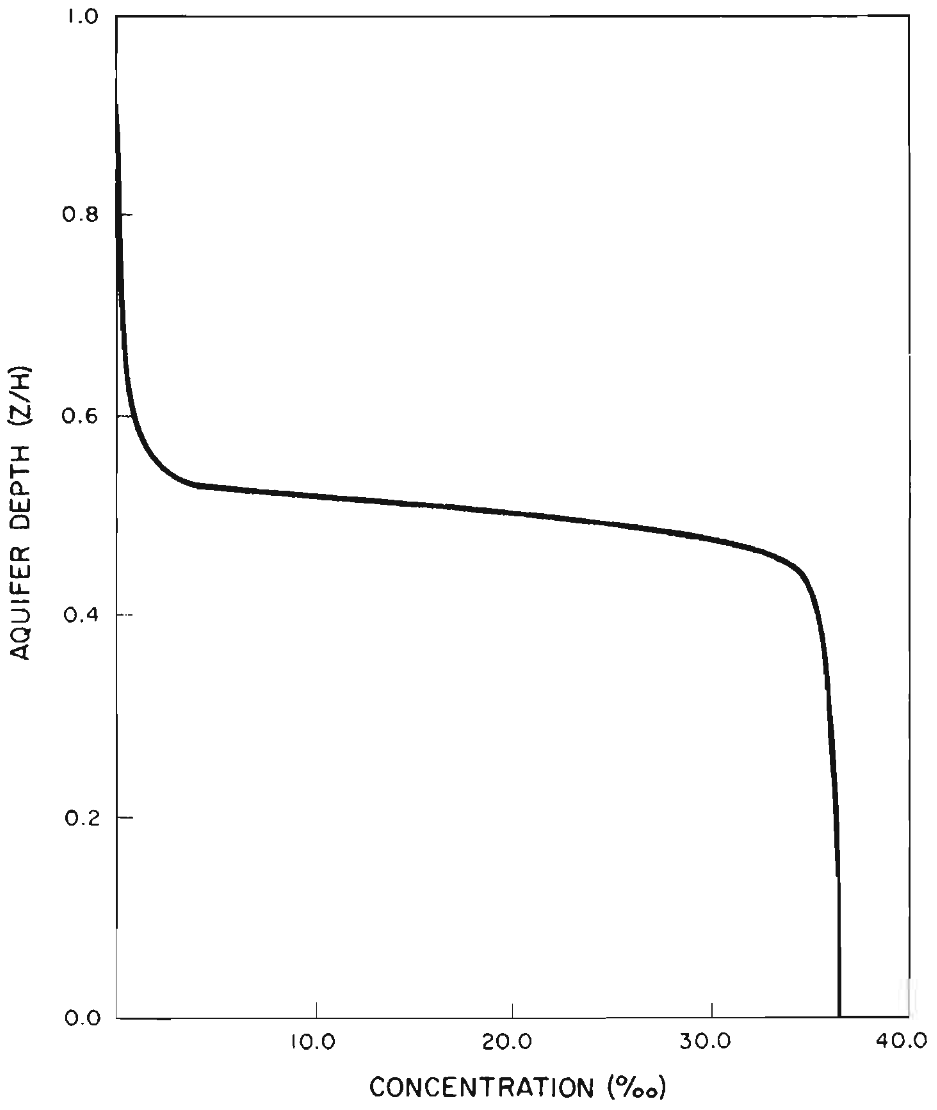


FIGURE 2. INITIAL SALINITY VALUES FOR SALTWATER INTRUSION PROBLEM

Initial conditions for pressure are unnecessary because the solution of equations (15) and (16) are time independent of pressure. This is explained in more detail in a later section of this report. The boundary conditions for pressure for both phases (salt-water intrusion and injection) are as follows:

$$P(0, z, t) = -\rho_s gz + P_{os} , \tag{23.1}$$

$$\frac{\partial P}{\partial x}(L, z, t) = o, c \leqslant z \leqslant z_R , \tag{23.2}$$

$$\frac{\partial P}{\partial x}(L, z, t) = -\frac{\mu}{k_x}U_c , z_r \leqslant z \leqslant H , \tag{23.3}$$

$$\frac{\partial P}{\partial z}(x, 0, t) = -g (x, 0, t) , \tag{23.4}$$

$$\frac{\partial P}{\partial z}(x, H, t) = -g (x, H, t) . \tag{23.5}$$

Equation (23.1) stipulates linear pressure (constant head) along the ocean-side boundary, while (23.2) and (23.3) specify, respectively, zero and constant flux along separate parts of the recharge boundary. Equations (23.4) and (23.5) specify, respectively, zero flux through the bottom and top of the aquifer.

In addition to the above boundary and initial conditions, two more conditions are necessary for pressure and salinity concentrations for the injection phase of the problem.

$$C(x_w, z_w, t) = 0 , \tag{24.1}$$

$$P(x_w, z_w, t) = \rho_o g(Hw - z_w) , \tag{24.2}$$

The effluent tracer (dissolved solids of the effluent) is also introduced into the system at the start of injection and its boundary and initial conditions must be accounted for as follows:

$$T(x, z, 0) = T_o, x = x_w, z = z_w , \tag{25.1}$$

$$T(x, z, 0) = 0, \text{elsewhere}, \tag{25.2}$$

$$T(x_w, z_w, 0) = T_o , \tag{25.3}$$

$$T(x_w \ z_w, t) = T_o , \tag{25.4}$$

$$\frac{\partial T}{\partial x}(0, z, t) = 0, \text{ where } U(0, z, t) < 0 , \tag{25.5}$$

$$T(0, z, t) = 0, \text{ where } U(L, z, t) \leqslant 0 , \tag{25.6}$$

$$T(L, z, t) = 0, \text{ where } U(L, z, t) \leqslant 0 , \tag{25.7}$$

$$\frac{\partial T}{\partial z}(x, 0) = \frac{\partial T}{\partial z}(x, H, t) = 0 , \tag{25.8}$$

$$V(x, 0, t) = V(x, H, t) = 0 . \tag{25.9}$$

Equations (25.1), (25.2), and (25.3) prescribe constant tracer concentrations at the well and zero concentrations everywhere else in the solution domain initially. Equation (25.4) provides for constant effluent tracer at the well throughout the simulation. Equations (25.5) and (25.6) provide for the flux of tracer through the outflow face and into the ocean and (25.7), (25.8), and (25.9) prescribe zero tracer flux through the other boundaries.

A complete mathematical formulation of the problem of waste injection into porous media saturated with density-stratified fluid has now been given.

FORMULATION OF GOVERNING EQUATION AND BOUNDARY CONDITIONS INTO A NUMERICAL MODEL

Solving Coupled Fluid Transport and Convective-Dispersion Equations

Equations of groundwater flow (fluid transport) and of mass transport (convective-dispersion) have been extensively treated. The many problems that have been solved fall into two categories: (a) problems in which the quantity of water is the desired answer, and (b) problems in which the quality of the water (distribution of a dissolved constituent) is the desired answer. In (a) it is only necessary to solve the equations of groundwater flow (eqq. [1], [2]); in (b) it is necessary to solve the equations of groundwater flow, and to use the resulting velocity field as input for the convective-dispersion equation (3).

The use of numerical methods for the solution of fluid transport and convective-dispersion equations has been widespread for many years. Remson, Hornberger, and Moltz (1971) provide a large amount of information on the finite difference method (FDM) solutions to these equations. Trescott, Pinder, and Larson (1976), Cooley (1974), and Pinder and Gray (1977) provide solutions to these equations by the finite element method (FEM).

Two major difficulties have been encountered using the FDM to solve the convective-dispersion equation: numerical dispersion (also known as numerical dissipation, numerical diffusion, and numerical smearing); and numerical overshoot. Numerical dispersion refers to the artificial smearing of the front or breakthrough curve that produces results indicative of a larger dispersion coefficient than has actually been specified. Numerical overshoot is an instability in the solution which produces erroneously high values upon approach to the front on the upstream side. Numerical undershoot is a similar phenomenon, but produces erroneously low concentration values on the downstream side of the front. These two phenomena are referred to collectively as numerical overshoot or simply as overshoot in this study because they are basically the same thing. Examples of numerical dispersion and overshoot are shown in Figure 3.

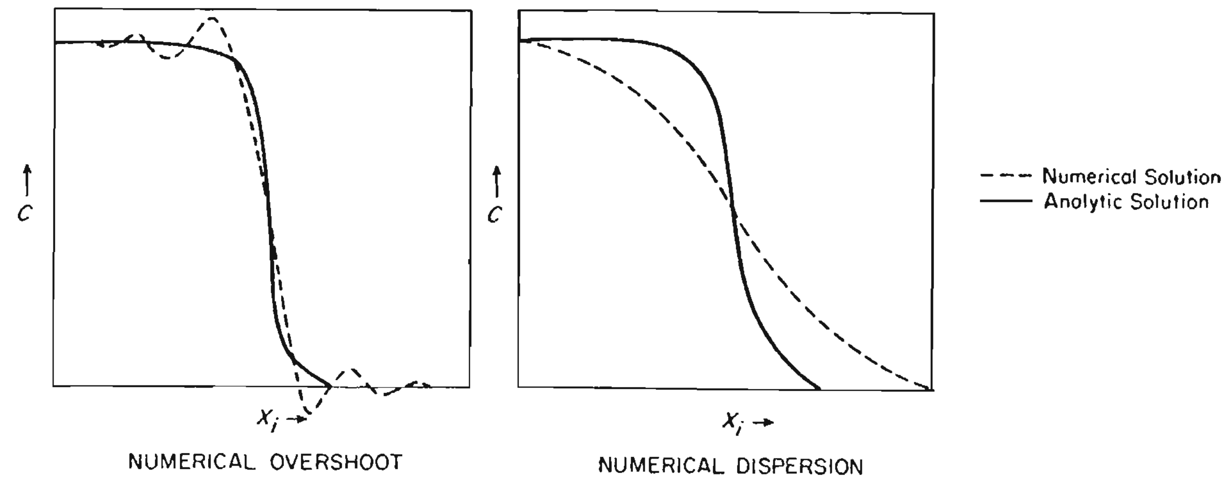


FIGURE 3. NUMERICAL DISPERSION AND OVERSHOOT

Numerical dispersion and overshoot can be especially severe when convective transport dominates over dispersive transport, i.e., when D_{ij} is small compared to $\underset{\sim}{q}$. An excellent review of these two problems is provided by Pinder and Gray (1977, pp. 147-69). Various schemes have been devised to reduce or eliminate numerical dispersion and overshoot for the FDM. Usually, one is reduced or eliminated at the expense of increasing the other. Pinder and Gray (1977) show that keeping the dimensionless parameter

$$\tilde{D} = D_k \frac{\Delta t}{(\Delta x)^2} \tag{26}$$

larger than a certain critical value eliminates the problem of numerical overshoot. The parameter D_k is a dispersion coefficient that is a constant scalar rather than a second-rank tensor and a function of the velocity. It is used in stability analysis of the numerical methods that solve the convective-dispersion equation and is dimensionally the same as D_{ij}. For the nonlinear case, it can be thought of as taking on the highest value of D_{ij} in the solution region. This critical value is much smaller for the FEM than for the FDM. One can interpret this as meaning that the FDM has a limited ability to convect a sharp front and is therefore not very good at simulating systems that closely mimic two-phase flow regimes, in which mass transport takes place as piston-like flow. In recent years, the FEM has been increasingly used to solve problems in convective-dispersion because the severity of such problems as numerical dispersion and overshoot are lessened or entirely overcome by certain types of FEM formulations (Pinder and Gray 1977).

Although numerical dispersion can cause serious errors when using FDM in situations where dispersivity coefficients a_L and a_T (and therefore D_{ij}) are small, these errors may be insignificant for situations in which the nodal spacing is on the same order of magnitude as the values of a_L and a_T. Robertson (1974) reported values for a_L and a_T for fractured-rock aquifers, such as flow basalts in Idaho, as high as 100 m. Lenda and Zuber (1970) presented similar values and Robertson* reports dispersivities as high as 400 m with a_L and a_T having approximately the same value. The grid spacing for many problems of practical value is on the same order of magnitude as dispersivity values; therefore it appears that for these problems there is

*J.B. Robertson 1978: personal communication.

no inherent advantage with respect to accuracy in using FDM or FEM to solve the convective-dispersion equation.

The problem we are addressing in this study falls into a third, much more complicated category for which the flow is coupled. That is, the fluid density is a function of the salinity, hence the fluid transport and convective-dispersion equations must be simultaneously solved. Very little previous research has been done on coupled flow problems, and most of the work that has been done treats problems of sea-water intrusion and upconing. However, this is most useful for the waste injection problem at hand because sea-water intrusion is a subset of the waste injection problem, i.e., a steady-state solution to the problem of sea-water intrusion is used as a set of initial conditions for the problem of waste injection. Salt-water upconing is, in a sense, the reverse problem of waste injection. For waste injection, the values of salinity and effluent tracer are known at the well, whereas for salt-water upconing, the salinity values at the well are unknown and are ultimately what one is seeking to find as an answer. Also, there is no need to introduce a second convective-dispersion equation for salt-water upconing or sea-water intrusion since no fluid from a well is entering the system.

Henry (1964) solved a set of coupled equations for the problem of sea-water intrusion of the form:

$$\underset{\sim}{q} = \frac{k}{\mu} (\nabla P + \rho \underset{\sim}{g}) \tag{27.1}$$

$$\nabla \cdot \rho \underset{\sim}{q} = 0 \tag{27.2}$$

$$\nabla \cdot C \underset{\sim}{q} - D_k \nabla^2 C = 0 \tag{27.3}$$

$$\rho = \rho_o + (1 - E)\, C \tag{27.4}$$

This set of equations is for steady-state and assumes a constant scalar dispersion coefficient. Henry's analytic solution was by a Fourier-Galerkin double-series expansion and was so difficult to evaluate that it could not in practice yield values for field condition parameters.

Pinder and Cooper (1970) solved a similar set of equations by numerical methods. The only difference is that their convective-dispersion equation was transient:

$$D_k \nabla^2 C - \nabla \cdot C\underset{\sim}{q} = \frac{\partial C}{\partial t} \tag{28}$$

Their solution was achieved by using a type of FDM for the fluid-transport equation and the method of characteristics (MOC), which is related to FDMs, for the convective-dispersion equation. Pinder and Cooper showed that the transient numerical solution approached Henry's steady-state analytic solution at long times. They did not, however, extend the solution to an actual field case.

Lee and Cheng (1974) solved the same steady-state set of equations that Henry solved (27) but with the FEM. They showed that their results were in good agreement with Henry's and then extended their model to an actual field problem (the Biscayne aquifer in the Cutler area, near Miami, Florida [Kohout 1964]) for which extensive field data have been produced. Good qualitative agreement with the field study was found.

All of the above studies assumed the dispersion coefficient to be a constant scalar. Pinder and Gray (1977) extended the work of Lee and Cheng (1970) on the Biscayne aquifer by using a dispersion coefficient that is a second-rank tensor and a function of the velocity.

McCracken et al. (1977) provide a FEM equation-solving code for a set of two coupled, three-dimensional, transient, nonlinear partial differential equations. With modification, this code can be used to solve the sea-water intrusion problem and the salt-water upconing problem (Hsieh 1977).

Choice of Methods for this Study

One of the greatest advantages in using the FEM, relative to the FDM, is its ease of formulation for irregularly shaped geometries. However, this advantage is lost in the present study which focusses on the solution of a hypothetical problem with simple rectangular geometry (Fig. 1.1). Therefore, the FDM was chosen for this study for two reasons: (1) the FDM is relatively easy to formulate for problems with rectangular geometry, and (2) its behavior for coupled, nonlinear problems is well understood (Pinder and Cooper 1970; Trescott, Pinder, and Larson 1972; Pinder and Gray 1977).

In order to decide which of the FDM methods to use, equations (15) to (17), and (19) must be classified according to which type of partial differential equation they are. Equations (15) and (16) can be combined in the following way:

$$k_x \frac{\partial^2 P}{\partial x^2} + k_z \frac{\partial^2 P}{\partial z^2} + \frac{k_x}{\rho} \frac{\partial \rho}{\partial x} \frac{\partial P}{\partial x} + \frac{k_z}{\rho} \frac{\partial \rho}{\partial z} \frac{\partial P}{\partial z} = F(x, z, t); \qquad (29.1)$$

$$F(x, z, t) = \frac{n\rho_o \beta_c \mu}{\rho} \frac{C}{\partial t} - \frac{2k_z g}{\mu} \frac{\partial \rho}{\partial z} . \qquad (29.2)$$

Equation (29) will be referred to as the fluid transport equation and is a Poisson-type equation. It is important to note that even though the function F (eq. [29.1]) is a function of time, there is no time derivative of pressure in equation (29.1). This means that once F is known at a given time, the solution of (29) is steady state with respect to pressure; in other words, at each time step the solution of the fluid transport equation is in equilibrium. Thus, an iterative FDM must be chosen to solve the fluid transport equation. It was decided to use the iterative-alternating-direction-implicit (IADI) method because it has been successful in solving the fluid transport portion of other salt-water intrusion problems (Pinder and Cooper 1970). Some authors have successfully used the FEM for similar problems (Lee and Cheng 1974; Pinder and Gray 1977), but the FEM was discarded for reasons presented above.

The convective-dispersion equations (17) and (19) are exactly the same mathematically. They are parabolic equations and, therefore, have a time derivative of the dependent variable (C for salinity, T for effluent tracer), in contrast to the fluid transport equation (29) which has no time derivative of the dependent variable (pressure). Several FDM methods can be used to solve this type of equation. The MOC described by Pinder and Cooper (1970) is particularly good at eliminating numerical dispersion, yet with a grid spacing roughly the same size as the values for dispersivity, numerical dispersion is relatively unimportant. No comparison was found in the literature of the ability of various FDMs to minimize the problem of numerical overshoot. Shamir and Harleman (1967) found that the alternating-direction-implicit (ADI) scheme was more efficient than other FDMs for problems in which dispersion at right angles to the flow is considered, i.e., when the dispersion coefficient is defined as a second-rank tensor. For this reason, and because the ADI method is quite similar in formulation to the IADI method, the ADI method was chosen to solve the convective-dispersion equations for salinity and effluent tracer.

Explanation of IADI and ADI Methods

The IADI and the ADI methods were first described by Peaceman and Ratchford (1955) and a very clear explanation of both methods is provided by Trescott, Pinder, and Larson (1972) along with detailed analyses of stability and convergence criteria for problems relating to fluid transport in aquifer simulation. Therefore, an explanation of both methods will be confined here to showing how they are applied to the particular equations generated for this problem.

The IADI representation of equation (29) can be written as

$$-a\ (A_1\ P_{i-1,j} + A_2\ P_{i,j} + A_3\ P_{i+1,j})^{m+1/2} +$$

$$a(A_4\ P_{i,j-1} + A_5 P_{i,j} + A_6 P_{i,j+1})^{m+b} - F(x,\ z,\ t)^m = 0\ ; \tag{30}$$

where

$$A_1 = \frac{k_x}{(\Delta x)^2} - \frac{k_x}{2\rho\Delta x}\frac{\partial\rho}{\partial x}\ ; \tag{31.1}$$

$$A_4 = \frac{k_z}{(\Delta z)^2} - \frac{k_z}{2\rho\Delta z}\frac{\partial\rho}{\partial z}\ ; \tag{31.2}$$

$$A_2 = M_1 - \frac{2k_x}{(\Delta x)^2}\ ; \tag{31.3}$$

$$A_5 = M_1 - \frac{2k_z}{(\Delta z)^2}\ ; \tag{31.4}$$

$$A_3 = \frac{k_x}{(\Delta x)^2} + \frac{k\ x}{2\rho\Delta x}\ ; \tag{31.5}$$

$$A_6 = \frac{k_z}{(\Delta z)^2} + \frac{k_z}{2\rho\Delta z}\frac{\partial\rho}{z}\ . \tag{31.6}$$

The superscript m refers to the iteration step which requires two steps to achieve a complete solution for each iteration. First, the partial derivatives with respect to z are treated as implicit by setting $a = 1$ and $b = 0$, thus advancing the solution to the $m+1/2$ iteration. Second, the partial derivatives with respect to x are treated as implicit by setting $a = 1$ and $b = 1$, thus advancing the solution to the $m+1$ iteration. The solution at the $m+1$ iteration is then compared to the solution at the m iteration and, if the maximum residual is less than a certain small value (details in Discussion of Numerical Results), a final solution is then considered to have been achieved, otherwise iteration continues.

The term M_1 (in eqq. [31.3], [31.4]) is an iteration parameter which is necessary to force the solution to converge. As long as the numerical value of M_1 is on the same order of magnitude as the other terms in A_2 and A_5, convergence can usually be expected to occur. However, most rapid convergence occurs when a set of iteration parameters are used, the set usually consisting of 5 to 10 separate iteration parameters which are cycled as the iteration proceeds. The cycle is set up by computing an ordered set of, e.g. five, iteration parameters (ordered from smallest to largest) and using M_1 for the first iteration, M_2 for the second iteration, and so on until M_5 is used on the fifth iteration (assuming convergence criteria have not been met). The sixth iteration then uses M_1 and the seventh iteration uses M_2, and thus the cycle repeats until convergence criteria are met (maximum residual becomes less than a certain small value). Trescott, Pinder, and Larson (1972) give equations for the optimum set of iteration parameters, which they have modified from Peaceman and Ratchford (1955), for two-dimensional horizontal flow in aquifers. These optimum iteration parameters are derived from the eigenvalue theory and are related to the eigenvalues of the matrices generated by equation (30). A further modification of these iteration parameters is presented here for equation (30). Let

$$M_1 = H_{p1}\left[\frac{2k_x}{(\Delta x)^2} + \frac{2k_z}{(\Delta z)^2}\right] . \qquad (32)$$

This modification normalizes M_1 so that its numerical value is on the same order of magnitude as the other terms in A_2 and A_5 (eqq. [31.3], [31.4]). The minimum iteration parameter ($H_{p\min}$) is calculated by finding the minimum of the following two terms:

$$H_{p\min} = \text{minimum}\left\{\frac{\pi^2}{2\,N^2\left[1 + \dfrac{k_z(\Delta x)^2}{k_x(\Delta z)^2}\right]} , \frac{\pi^2}{2\,M^2\left[1 + \dfrac{k_x(\Delta z)^2}{k_z(\Delta x)^2}\right]}\right\} , \qquad (33)$$

where $N = (L/\Delta x) + 1$ and $M = (H/\Delta z) + 1$. The incremental iteration parameter (H_{pi}) is calculated by the following equation,

$$H_{pi} = \exp\left[\frac{\ln\left(\frac{2}{H_{p\min}}\right)}{N_{ip} - 1}\right] , \qquad (34)$$

where N_{ip} is the number of iteration parameters to be used. As previously mentioned, the number of iteration parameters (N_{ip}) to be used in the set for cycling is usually between 5 and 10. For a problem where the coefficients in equation (30) are variable (like A_1, A_2, A_4, A_6), the optimum number for N_{ip} is not theoretically known and trial and error provides the only guide to what value of N_{ip} yields the most rapid convergence. A value of N_{ip} = 10 usually provided most rapid convergence for this problem.

The complete set of iteration parameters is computed by a recursive formula,

$$H_{p1} = H_{p\min} , \qquad (35)$$

and

$$H_{pl} = (H_{pl-1})\ H_{pi} \ , \ l = 2,\ N_{ip} \ ; \qquad (36)$$

and thus

$$M_l = (H_{pl-1})\ H_{pi}\ \frac{2k_x}{(\Delta x)^2} + \frac{2k_z}{(\Delta z)^2} \ , \ l = 2,\ N_{ip} \ . \qquad (37)$$

Solutions of the convective-dispersion equations (17) and (19) by the ADI method are quite similar in terms of numerical formulation so that only the solution of the convective-dispersion equation for salinity (17) will be described here. As in the case for the IADI method, the ADI method was first described in Peaceman and Ratchford (1955), and Trescott, Pinder, and Larson (1972) provide a detailed description of the method and its stability and convergence criteria for problems relating to fluid transport in aquifer simulation.

By assuming that mixed derivatives are negligible (compared to other terms), i.e.,

$$\frac{\partial^2 C}{\partial x \partial z} + \frac{\partial^2 C}{\partial z \partial x} = 0 \ , \qquad (38)$$

and that the dispersion tensor is symmetric,

$$D_{xz} = D_{zx} \ , \qquad (39)$$

we can write the ADI representation of equation (17) as:

$$-a\left(B_1C_{i-1,j} + B_2C_{i,j} + B_3C_{i+1,j}\right)^{k+1/2} +$$

$$a\left(B_4C_{i,j-1} + B_5C_{i,j} + B_6C_{i,j+1}\right)^{k+b} = 0 \ , \qquad (40)$$

where

$$B_1 = \frac{D_{xx}}{(\Delta x)^2} - \left[\frac{\left(\frac{\partial D_{xx}}{\partial x} + \frac{\partial D_{xz}}{\partial z} - U\right)}{2\Delta x}\right] , \qquad (41.1)$$

$$B_2 = \frac{2n}{\Delta t} - \frac{2D_{xx}}{(\Delta x)^2} , \qquad (41.2)$$

$$B_3 = \frac{D_{xx}}{(\Delta x)^2} - \left[\frac{\left(\frac{\partial D_{xx}}{\partial x} + \frac{\partial D_{xz}}{\partial z}\right)}{2\Delta x}\right] , \qquad (41.3)$$

$$B_4 = \frac{D_{zz}}{(\Delta x)^2} - \left[\frac{\left(\frac{\partial D_{xz}}{\partial x} + \frac{\partial D_{zz}}{\partial z}\right)}{2\Delta z}\right] , \qquad (41.4)$$

$$B_5 = \frac{2n}{\Delta t} - \frac{D_{zz}}{(\Delta z)} , \qquad (41.5)$$

and

$$B_6 = \frac{D_{zz}}{(\Delta z)^2} + \left[\frac{\left(\frac{\partial D_{xz}}{\partial x} + \frac{\partial D_{zz}}{\partial z}\right)}{2\Delta z}\right] . \qquad (41.6)$$

In equation (40), the superscript k refers to the time step (whereas in [30] the superscript m referred to the iteration step). At the first one-half time step (k+1/2), a = -1 and b = 0 and the partial derivatives with respect to z are implicit. After the solution has been advanced to the k+1/2 time step, a = 1 and b = 1 and the partial derivatives with respect to x are implicit. The solution at the k+1 time step represents the complete solution to equation 17 at the k+1 time step.

The boundary conditions reduce the number of unknowns in equations (30), (40) and the equivalent of (40) for effluent tracer, to two unknowns at the boundaries, so that the matrices can be solved by the Thomas algorithm for solution of tridiagonal matrices. For Dirichlet boundary conditions, the term at the boundary is known and is put on the right-hand side of the equation. For Neumann boundary conditions, the equation is rewritten in terms of the finite difference representation of the known first derivative, which has only two terms.

Modifications Necessary for Solving Coupled Equations

The coefficients A_1, A_2, A_4, and A_6 (hereafter, A_i) are functions of salinity (because density is a function of concentration), thus their exact values are not known at the advance (k+1) time level until a solution to equation (40) is attained. But in equation (40), the coefficients B_1 through B_6 (hereafter, B_i) are all functions of pressure (because the dispersion coefficients are functions of velocity, which is a function of pressure). Therefore, the coefficients B_i are unknowns until equation (30) is solved, thus equations (30) and (40) are coupled. In order to achieve a solution at the advance time level, a procedure which shall be called *iteration across a time step* (IATS) was used.

The IATS procedure involves computing the A_i and B_i coefficients with values of pressure and salinity from the previous (k) time step. Since A_i and B_i are supposed to be computed with values from the k+1 time step, the values from the k time step are considered "guess values." Once a solution for pressure and salinity has been achieved with these guess values for the A_i and B_i coefficients, new guess values for A_i and B_i can be computed and the pressure and salinity values are recomputed for the k+1 time step. The recomputed values for pressure and salinity are compared to the previously computed values and if the maximum residuals for salinity and pressure are less than some small value, IATS is completed and the guess values for A_i and B_i are the true values for the k+1 time step. Otherwise new values for A_i and B_i are computed and the IATS is repeated until the convergence criteria are met. The IATS procedure is summarized in the flow chart in Appendix B, and Appendix C is a listing of the program used to solve the problem.

Once the complete solution for equations (29) and (17) (pressure and salinity) is attained at the k+1 time step, then solution of equation (19) (effluent tracer) is provided by a normal ADI procedure without IATS, because equation (19) is not coupled to any of the other equations (although the velocities obtained from solution [29] are used to calculate dispersion coefficients for [19], i.e., the coupling is only one way, as explained earlier). Thus, obtaining the values for effluent tracer completes the solution at the $k+1$ time step and the entire process is then repeated for the next time step, and the next, and so on until a steady-state solution is achieved for salinity and effluent tracer.

DISCUSSION OF NUMERICAL RESULTS

Accuracy

Since two different methods are used to solve the fluid transport and convective-dispersion equations, the accuracy of these methods are separately treated. The fluid transport equation is a steady-state equation for each time step and the iterative scheme (IADI) discussed in the previous chapter was used to solve it. The iteration procedure was carried out until the maximum residual (εp) was less than 0.01 dyne/cm^2. This value was chosen because decreasing εp by one more order of magnitude (to 0.001) changed the specific discharge by less than 1.0×10^{-6} cm/s. A specific discharge as small as 1.0×10^{-6} cm/s had little effect on the results over the time period of the simulation.

The accuracy of the ADI solution of the convective-dispersion is related to stability and convergence and will be discussed in a following section.

Errors in Vertical Velocity Calculation

The fluid transport equation is solved for the pressure field, hence the velocities are calculated by taking finite difference representations of the pressure derivatives. For certain situations, this method can cause significant errors in the calculation of the vertical velocities which are described below.

The vertical velocity is given by

$$\frac{q_z}{n} = V = \frac{-k_z}{\mu n}\left(\frac{\partial P}{\partial z} + \rho g\right) . \qquad (42.1,\ .2)$$

The finite difference representation of this is given by

$$V \simeq \frac{-k_z}{\mu n} (\delta_z \; P + \rho \;) \; ; \tag{43.1}$$

$$\delta_z \; P = \frac{P_{i,j+1} - P_{i,j-1}}{2\Delta z} \; . \tag{43.2}$$

When $V = 0$,

$$\frac{\partial P}{\partial z} + \rho g = 0 \tag{44}$$

and

$$\delta_z P + \rho g = \varepsilon_T \tag{45}$$

where ε_T is the truncation error in estimating $\partial P/\partial z$. It is shown in Appendix A that in general,

$$\varepsilon_T = \frac{(\Delta z)^2}{3!} \frac{\partial^3 P}{\partial z^3} + \text{(higher order terms)}. \tag{46}$$

For the special case where $V = 0$, (44) holds and (shown in App. A)

$$\varepsilon_T(V = 0) = \frac{g(\Delta z)}{3!} \rho_o \beta_c \frac{\partial^2 C}{\partial z^2} + \text{(higher order terms)}. \tag{47}$$

In most regions of the solution space, V is non-zero and (47) does not hold and/or $\partial^2 C/\partial z^2$ is very small and $\varepsilon_T(V = 0)$ is therefore too small to affect the calculation of V. Figure 4 is a salinity profile similar to Figure 1 and delineates five regions. It can be seen that in regions II and IV, $\partial^2 C/\partial z^2$ is large compared to the other regions. If the condition also exists that $V = 0$ (or very nearly so) then a significant error is made in calculating vertical velocity in these two regions with equation (43) due to the truncation error given by (47).

Figure 5 is a plot of velocity vectors from a typical waste injection simulation and shows the region where this effect is significant. Since the region delineated by Figure 5 is the only area where velocity was significantly affected by truncation error, no attempt was made to add the third-order term in equation (47) to equation (43.2) because the error had little effect on the overall solution.

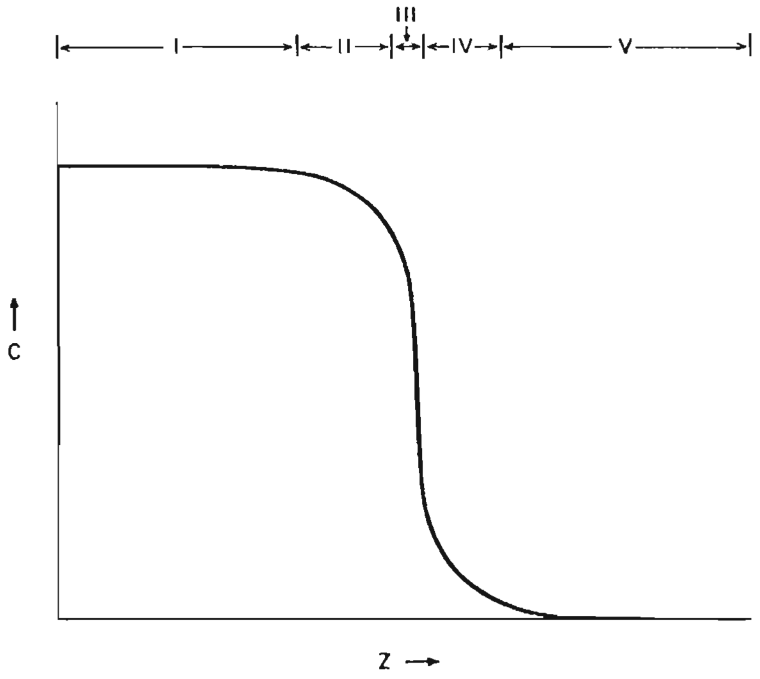


FIGURE 4. REGIONS OF HIGH SALINITY GRADIENT

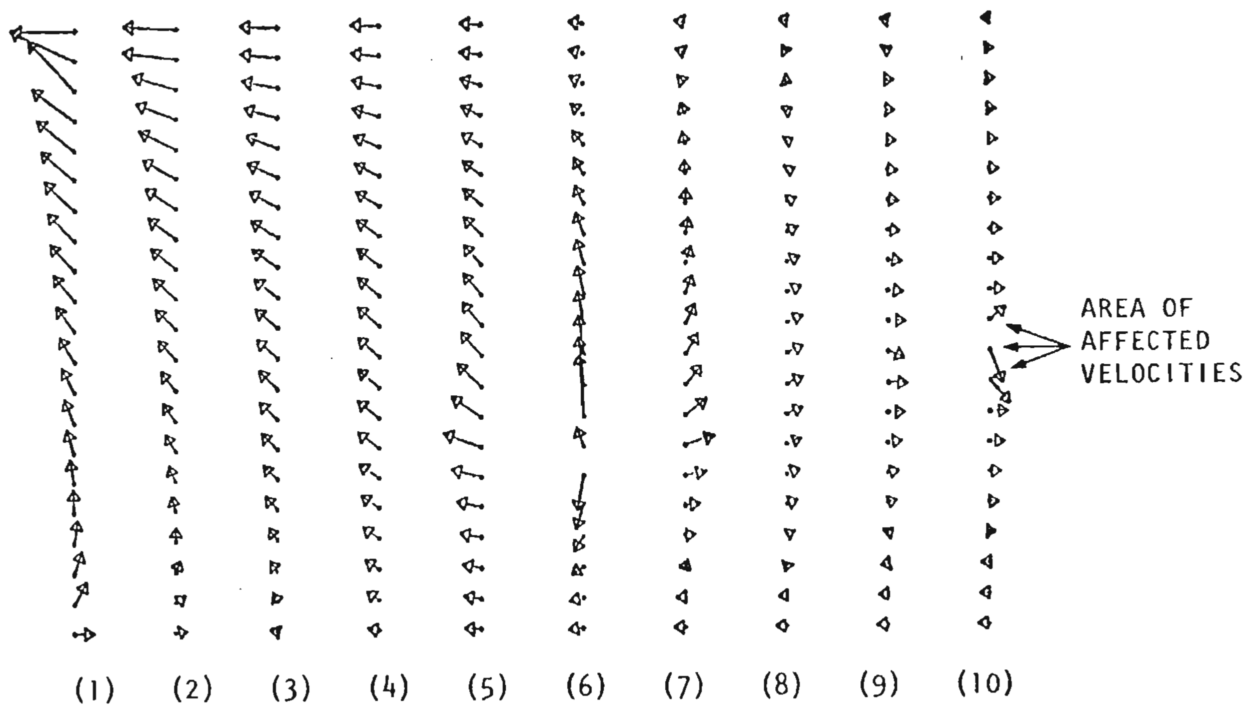


FIGURE 5. EFFECT OF TRUNCATION ERROR ON VERTICAL VELOCITIES

Stability and Convergence

The IADI solution to the fluid transport equation is stable if the maximum residual converges to zero during the iterative process (Peaceman and Rachford 1955); therefore, the iterative solution is often referred to as having convergence. This type of convergence should not be confused with the usual meaning of convergence of a numerical scheme, i.e., that the numerical solution is convergent if it approaches the analytic solution as the grid spacing approaches zero. Unless otherwise noted, the term convergence in this report will refer to iterative convergence.

As discussed previously, convergence is more rapidly achieved when a set of iteration parameters is used. The choice of iteration parameters for the IADI is discussed in detail by Remson, Hornberger, and Moltz (1971), and by Trescott, Pinder, and Larson (1976). A choice of ten iteration parameters provided the most rapid convergence for this problem. For the first time step, approximately 30 iterations were usually necessary because the "guess" values for pressure were not particularly close to the actual values. In succeeding time steps, only about four iterations were required because the "guess" values were taken from the last time step and were therefore much closer to the new pressure values for that time step.

The time step for the fluid transport equation can be much larger than the time step for the convective-dispersion equation because changes in pressure distribution are strictly due to changes in density. Solving the fluid transport equation at every tenth time step was found to be adequate. The vertical velocities were much more sensitive to changes in salinity than to changes in pressure. The vertical velocities were therefore updated at the end of each iteration across a time step and used to solve again the convective-dispersion equation at that time step until convergence was achieved. An adequate convergence criterion for the maximum residual for salinity was found to be 0.5‰. This figure was used because a change of less than 0.5‰ salinity does not significantly affect the density and therefore does not significantly affect the vertical velocity.

For most simulations, rapid changes in salinity near the beginning of the simulation made about six iterations across the time step necessary for the salinity convergence criterion to be met. As time progressed, fewer iterations were necessary. There is a trade-off between the size of the time step and the resultant number of iterations that must be performed to achieve convergence across the time step. In practice, it was found that the optimum time step was one which allowed convergence in two iterations after about 10% of the total simulation time.

It is also of interest to discuss stability in relation to numerical overshoot. Previously, the parameter

$$\tilde{D} = D_k \frac{\Delta t}{(\Delta x)^2} \tag{26}$$

was discussed. This parameter must be kept above a certain minimum value to prevent numerical overshoot. For the nonlinear problem, the analogous equation to (26) is

$$\tilde{D}_{nL} = D_{ij} \frac{\Delta t}{(\Delta x_j)^2} \quad . \tag{48}$$

In general, D_{ij} is a function of a_L, a_T, U, and V. Of these four parameters, only a_L and a_T can be arbitrarily adjusted to eliminate overshoot. If the dispersivity values that are desired for a particular simulation are not large enough to prevent overshoot, then Δt must be made larger and/or Δx_j must be made smaller. The usual choice would be to enlarge Δt, making the solution more efficient (larger time steps bring the solution to steady state with fewer total time steps). However, for a coupled flow problem this is not necessarily true because a larger time step causes more iteration across the time step, thus drastically increasing the amount of computer time necessary to achieve a complete solution at a given time step. In fact, a time step that is too large causes the ADI solution of the convective-dispersion equation to become unstable. The other alternative to increase the size of $\tilde{D}_{nL}$ is to decrease Δx_j which also greatly increases computation time by creating more nodes.

For the simulations using sandbox model parameters, a_L and a_T were respectively specified as 50 and 10 cm. These were the minimum values for Δt = 180 s (for injection), Δx = 16.7 cm, and Δz = 5 cm. For the steady-state salt-water intrusion problem, it was possible to use a Δt of 600 s with the same parameters. The parameters used for the hypothetical field problem are discussed in a later section.

Efficiency

The computer code was written using virtual storage coding practices in keeping with the IBM370/158 OSVS2 machine and operating system on which this work was done.

For virtual storage systems, most of the computer program and its data are kept out of main memory on direct-access devices. The program and its data are "paged in" and "paged out" as needed during execution by the system's hardware, even though the programmer may not be aware that this is being done. It is therefore not necessary to save memory by reading and writing out arrays onto disk; the system essentially does this on its own. Many computer programs are specifically written to conserve memory and this is actually quite inefficient under a virtual programming environment.

RESULTS OF SIMULATIONS

Verification of Numerical Model with Existing Numerical and Analytic Models

Three solutions discussed previously were chosen for comparison: the analytic solution to Henry's (1964) problem; Pinder and Cooper's (1970) numerical solution to the same problem; and Pinder and Gray's (1977) numerical solution to the same problem using the tensor form of the dispersion coefficient. These results are plotted in Figure 6 as the 50% isochlors.

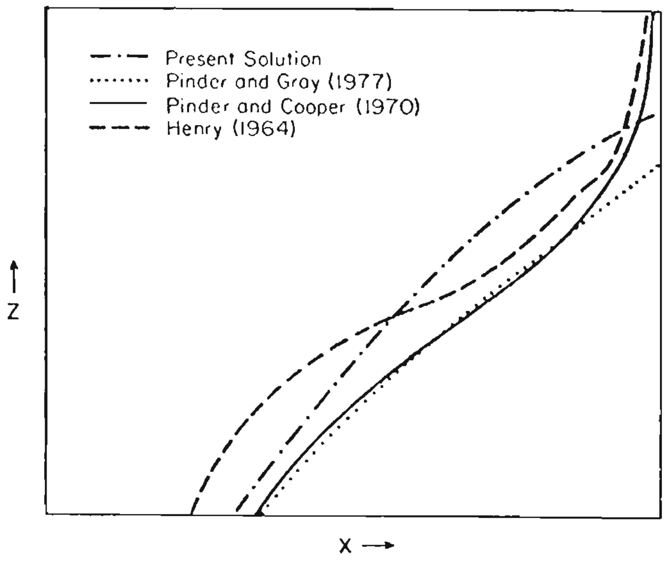


FIGURE 6. COMPARISON OF PREVIOUS AND PRESENT SOLUTIONS TO SEA-WATER INTRUSION PROBLEM

The results of the present study are most similar to those of Pinder and Gray (1977). This is because the present study and Pinder and Gray both employ a second-rank tensor that is a function of the fluid velocity for the dispersion coefficient, while the other two studies use a constant scalar for the dispersion coefficient. The fact that the 50% isochlor from the present study falls somewhat above that of Pinder and Gray (1977) is attributed to the increased numerical dispersion inherent in the FDM (present study) versus the FEM (Pinder and Gray 1977). Nevertheless, the agreement of the present solution with these past solutions is quite good and demonstrates that the program correctly solves the governing equations and boundary conditions for the salt-water intrusion problem. Note that the coordinates for the present problem have been changed for Figure 6 to conform to the coordinate system for the other solutions.

Nomenclature Definition of Three Types of Aquifer Fluids

Although the governing convective-dispersion equations keep track, i.e., conserve mass, of the solutes themselves, one can make the following simple assumption for conceptual purposes. The value $(T/T_o)\times 100$ at any node represents the percent of effluent in a water sample collected hypothetically from that node. Similarly, $(C/C_s)\times 100$ represents the percent of sea water present in the sample. The only other source of fluid is the freshwater recharge R (or simply, fresh water), coming from the right-hand side of the model as ambient flow. Therefore, the percent recharge present at any node can be calculated by the equation:

$$R = 1 - [(T/T_o + C/C_s)] \times 100 \ . \tag{49}$$

For conceptual purposes, the following discussion will sometimes refer to three fluids: fresh water, effluent, and salt or saline water. This is done merely to facilitate discussion and it is recognized that in actual fact, there is only one fluid (fresh water) that contains two solutes (effluent tracer and salinity) as discussed previously.

Summary of Numerical Simulations

Altogether, four simulations were run using geometric and aquifer constants that were identical, except for the dispersivity coefficients, to those of the sandbox model. These values are listed in Table 1. The first

TABLE 1. GEOMETRICAL AND AQUIFER CONSTANTS

Longitudinal dispersivity (a_L), cm	50
Transverse dispersivity (a_T), cm	10
Length (L), cm	150
Height (H), cm	100
Intrinsic permeability, x-direction (k_x), cm^2	8.68×10^{-7}
Intrinsic permeability, z-direction (k_z), cm^2	8.67×10^{-7}; -8.67×10^{-9}
Diffusion coefficient x tortuosity ($D_d T^*$), cm^2/s	9.89×10^{-6}
Viscosity (μ), g/cm-s	0.01
Porosity (n)	0.39
Density (fresh water)(ρ_o), g/cm^3	0.9983
Δx, cm	16
Δz, cm	5

simulation was run with a constant head boundary condition on the recharge ($x = L$) side of the model. The pressure was calculated by:

$$P\ (L,\ z) = \rho_o\ g\ H_r + g \int_H^z \rho(L,\ z)\ d\ z\ , \tag{50}$$

where H_r is the head above sea level on the right-hand ($x = L$) side of the model. Because of the constant head boundary condition, this simulation was called H-1, the H referring to constant head. For reasons that will be discussed later, it was decided that the H-1 run was least similar to the sandbox model experiments, so the boundary condition on the recharge ($x = L$) side was changed to one of constant recharge, or constant flux. Three simulations were run with constant flux and were called F-1, F-2, and F-3 (the F referring to flux). The value of the head, H_r, in H-1 was chosen to correspond with a typical value from the sandbox model experiments. In the F-series, the value of the ambient flow was chosen to correspond to a typical value observed in the sandbox model experiments. Table 2 lists the parameters varied in the simulations.

TABLE 2. PARAMETERS VARIED IN SIMULATIONS

Simulation	Position of Well (x_w,z_w)	Head Above Sea Level at Well	Injection Rate	Ambient Flow*		Ambient Salinity at Start of Injection
	(cm)	(cm)	(cm^2/min)	(cm/s)	(%)	(g/ℓ or %)
H-1	83.3, 30.0	2.6	7.43	(Constant head above sea level at $x = L$ of 1.9 cm)		28.0
F-1	83.3, 30.0	4.0	11.1	1.54×10^{-3},	70	11.2
F-2	100.0, 30.0	4.0	10.1	1.54×10^{-3},	50	18.0
F-3	83.3, 30.0	3.5	4.7	1.54×10^{-3},	50	17.5

*Specific discharge (q) input at right ($x = L$) boundary and percent of right boundary over which q is applied.

Two different boundary conditions were used (constant head and constant flux) because it was not at first apparent which of the two would more closely simulate the conditions observed in the sandbox model experiments. In the sandbox model experiments, a nearly constant flux (recharge) was maintained after the start of injection by maintaining a constant head on the recharge ($x = L$) side of the sandbox model. The three-dimensional nature of the sandbox model allowed a nearly constant ambient flow by using the constant head because the discharge due to the effluent plume was small compared to the total ambient flow. Therefore, even after injection began, the ambient flow remained about the same as before injection. However, in the two-dimensional case of the numerical model, discharge from the effluent plume represented a significant portion of the total discharge; therefore, to achieve a constant ambient flow, a constant flux boundary condition was necessary on the recharge ($x = L$) side of the aquifer.

Presentation of Results

The results of the simulations are presented in three ways. First, the fluid flow is shown as fields of velocity vectors, one for each node (Figs. 7 and 8). Second, the salinity and effluent tracer concentrations are shown in a time-series of plots of salinity and tracer concentrations (Fig. 9). Lines of equal concentration of salinity (isochlors) and effluent tracer (isopleths) are plotted as $(C/C_S)\times100$ and $(T/T_O)\times100$, respectively. Each plot should be viewed with the same orientation as used in Figure 2, i.e.,

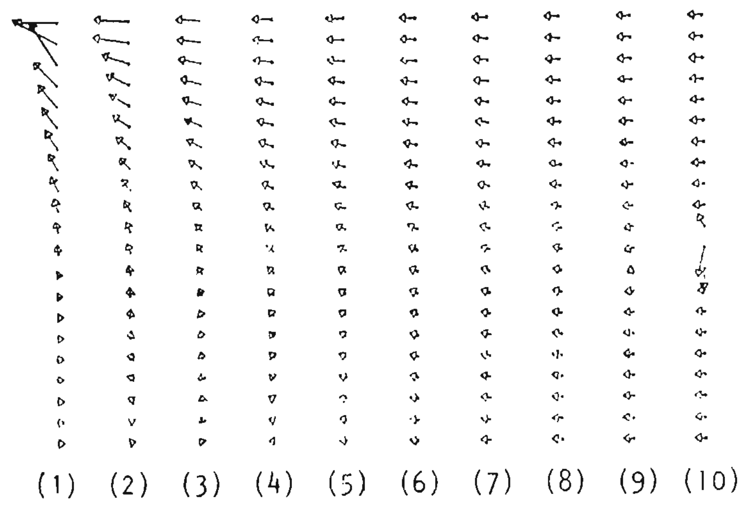
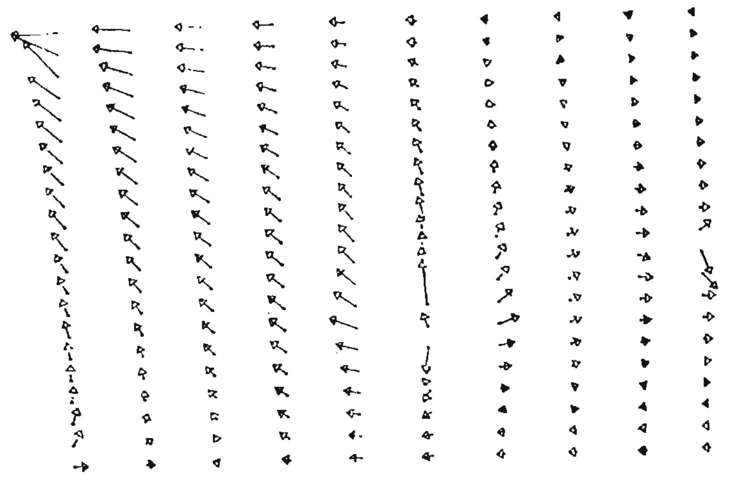
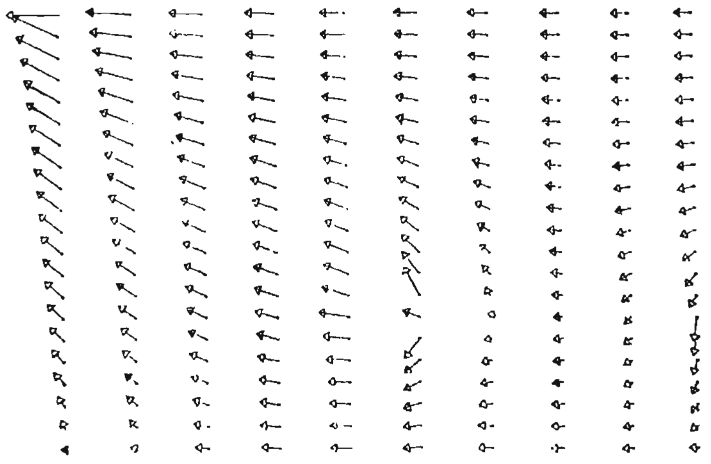


FIGURE 7. VELOCITY VECTORS FOR STEADY-STATE SOLUTION OF SALTWATER INTRUSION PROBLEM FOR H-1 SIMULATION



8.1. H-1 SIMULATION



8.2. F-1 SIMULATION

FIGURE 8. VELOCITY VECTORS FOR STEADY-STATE SOLUTION OF INJECTION PHASES FOR H-1 AND F-1 SIMULATIONS

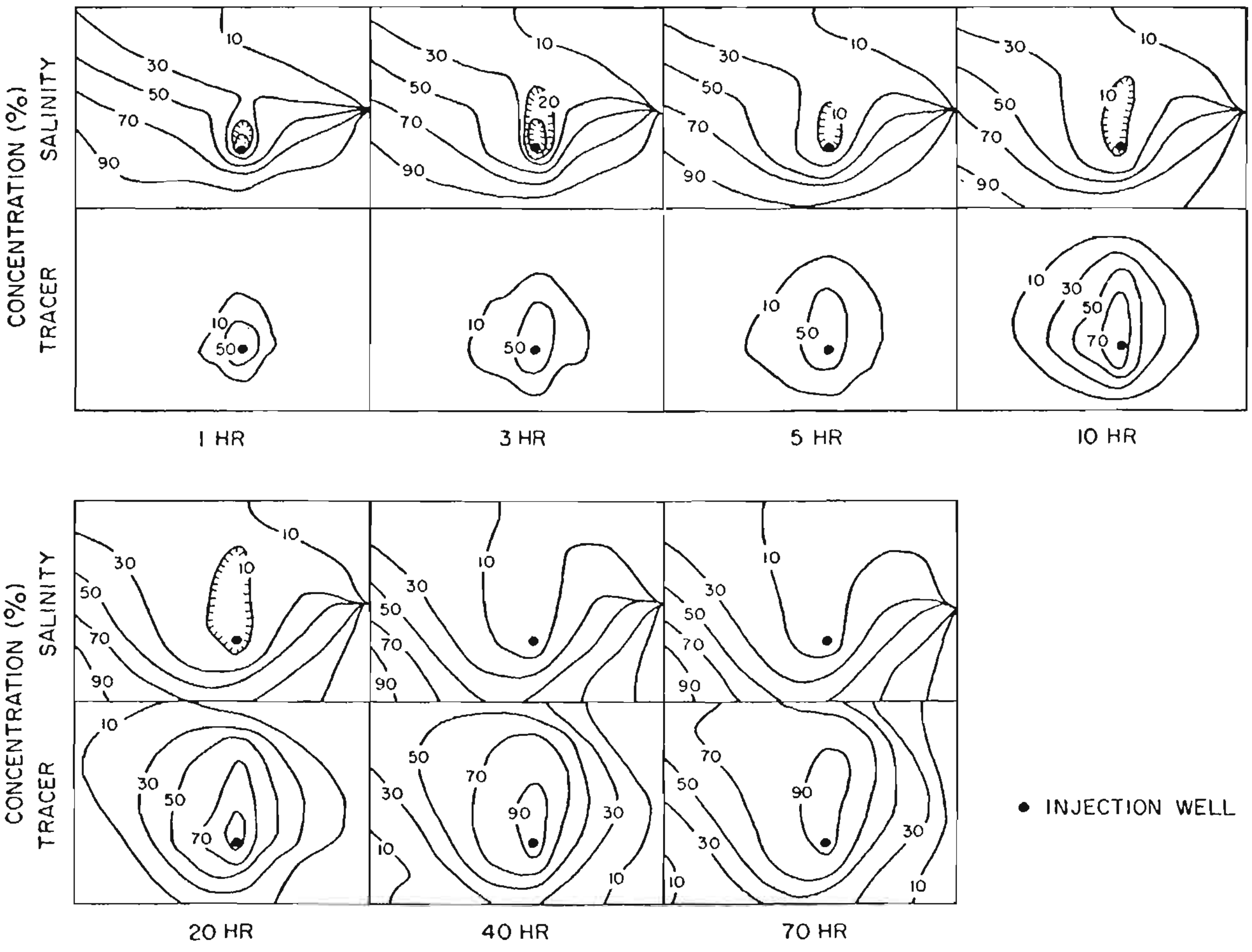


FIGURE 9. TIME-SERIES OF SALINITY AND EFFLUENT TRACER CONCENTRATIONS FOR H-1 SIMULATION

with the ocean on the left and the recharge boundary on the right. By plotting these two sets of data together, one can easily see the effects of the density-dependent flow, dispersion, and the ambient flow regime. The third way in which the results are displayed is shown in Figure 10. This graph (and others like it, shown later) shows the relative proportions of saline, effluent, and fresh (recharge) water that are leaving (or entering) the aquifer at the ocean-aquifer interface as a function of depth. In this graph, the line for salinity is plotted by using $(C/C_s)\times100$, and the line for effluent by using $[(C/C_s) + (T/T_o)]\times100$. Thus, if one were to extract a sample of water just as it left the aquifer right near the top of the aquifer, 18% of the sample would be saline, 67% would be effluent and the remainder (15%) would be fresh (recharge) water. On the other hand, a sample extracted near the bottom of the aquifer would be 100% saline because

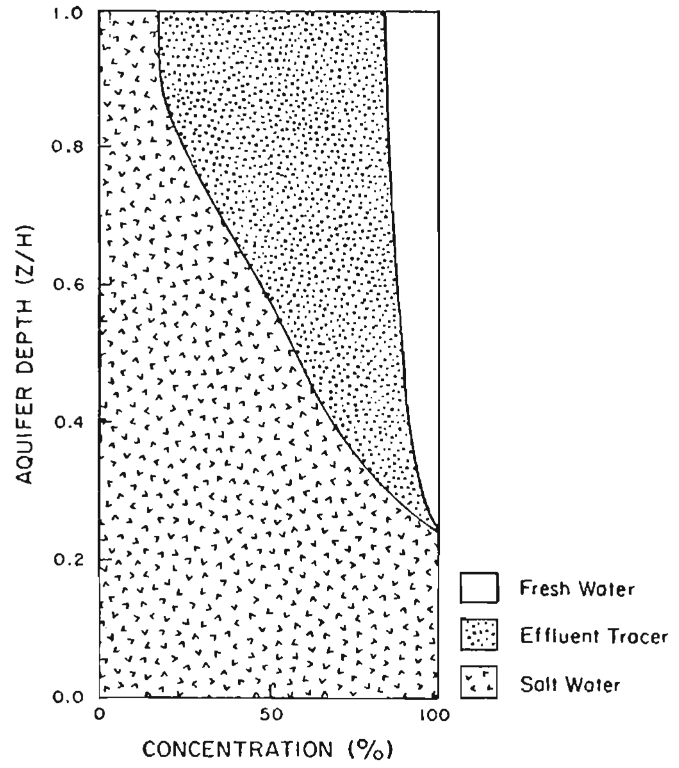


FIGURE 10. RELATIVE PROPORTIONS OF EFFLUENT, SALINE, AND FRESH WATER AS FUNCTION OF DEPTH AT OCEAN-AQUIFER INTERFACE FOR H-1 SIMULATION

salt water is flowing into the bottom 25% of the aquifer (see velocity vectors, Fig. 8.1). Thus, this type of graph helps in understanding how the effluent is distributed with depth as it flows into the ocean.

Description of H-1 Simulation

The velocity vectors for the steady-state solution to the salt-water intrusion phase of H-1 are displayed in Figure 7 (same right-hand boundary condition as injection phase of H-1). Note the recirculation effect of salt-water entering the aquifer on the lower left (ocean side), moving upwards, and exiting with the recharge water at high velocities near the top portion of the exit boundary (upper left-hand side). This same effect has been reported by Henry (1964), Pinder and Cooper (1970), and Pinder and Gray (1977). The steady-state solution to the H-1 salt-water intrusion problem provided ambient salinities around the injection well of about 28‰ (77% isochlor) so that a strong buoyant force was exerted on the fresh-water density fluid exiting the well during injection.

Figure 8.1 is a plot of the steady-state velocity vectors for H-1 (Fig. 8.2 is a similar plot from the F-1 simulation; 8.1 and .2 are plotted together for purposes of comparison). The injection effects on the velocity are apparent. Note that as in Figure 7, continuity considerations force the fluid in the top left area to move faster as it approaches the ocean exit boundary. The velocity vectors above the well have much stronger vertical components than do those below the well due to the buoyancy effects. Because a constant head has been prescribed for the right ($x = L$) boundary, the stagnation line for this flow field is far upstream from the well, and as Figure 8.1 shows, fluid from the well flows out of the right boundary. This indicates that the injection well discharge was large relative to the ambient flow that existed in the intrusion phase of the simulation. Thus, only a very small amount of ambient flow was necessary to maintain the constant head. For this reason it was felt that a constant flux boundary condition on the right side would more correctly simulate a constant ambient flow as observed in the sandbox model experiments. A similar constant ambient flow would also be expected for a field situation.

The migration of salinity and effluent is demonstrated in Figure 9. As the effluent exits the well, it mixes with the denser ambient fluid by the process of hydrodynamic dispersion. As the 10% isopleth propagates away from the injection well, it is so well mixed with the ambient fluid that the buoyant effect is hardly noticeable. After 1 hr, the 50% isopleth demonstrates very little additional upward movement due to buoyancy but by 3 and 5 hr, it has noticeably migrated more in the vertical direction than the horizontal. After 10 hr of simulation time, a strong positive correlation between the percent of effluent and the amount of buoyant rise is obvious. As the effluent enters the region of higher velocities (upper left, Fig. 9), the isopleths propagate faster and disperse more widely (20, 40, 70 hr).

Several important features about this simulation should be emphasized:

1. By comparing Figures 7 and 8.1, it can be seen that the ambient flow field is reduced to only a small portion of its original preinjection amount. The injected fluid displaces most of the ambient flow so that very little ambient flow is necessary to maintain the same preinjection constant head on the right ($x = L$) boundary.

2. Because of hydrodynamic dispersion, some of the effluent mixes well enough with the ambient fluid so that it propagates away from the well almost equally in all directions, thereby displaying very little buoyant effect. However, the fluid near the well is less dense than the surrounding fluid because the effluent is more concentrated near the well. This less dense fluid containing a high percentage of effluent tends to rise more vertically than horizontally due to the buoyant force.
3. By looking at Figure 10, it can be seen that most of the effluent leaves the aquifer near the top, in other words, the effluent exits near the coast in shallow water, rather than in deeper water. This is because very little ambient flow is necessary to maintain constant head on the right ($x = L$) side of the aquifer. With very little ambient flow, the effluent tends to rise into the fresher portion of the aquifer near the top, and this is where it leaves the aquifer at the ocean-aquifer interface.

Description of F-1 Simulation

The initial conditions for the F-1 injection simulation were set up differently than those for H-1. The interface on the right-hand side was set at the point where the specific discharge went to zero ($z/H = 0.35$), thus allowing only fresh water to enter the model (the interface was set at $z/H = 0.5$ for the H-1 simulation). The result is that the 50% isochlor is much lower, i.e., deeper in the aquifer, in the F-1 simulation and injection takes place into a much less dense ambient fluid so that the buoyant force acting on the fluid is much weaker in F-1 than in H-1. For F-1, the ambient salinity at the start of injection was 11.2‰ (31% isochlor), while for H-1 the preinjection ambient salinity was 28‰ (77% isochlor). The F-1 simulation then corresponds to injection into a brackish, highly dispersed transition zone. This is the type of ambient fluid present in many of the coastal caprock sediments into which injection takes place in the Hawaiian Islands, especially O'ahu and Maui. Figure 11 is a series of plots of steady-state preinjection isochlors for the H-1, F-1, and F-2 simulations (F-3 used the same preinjection isochlors for the injection phase initial conditions as F-2 did).

A plot of steady-state velocity vectors for F-1 is shown in Figure 8.2

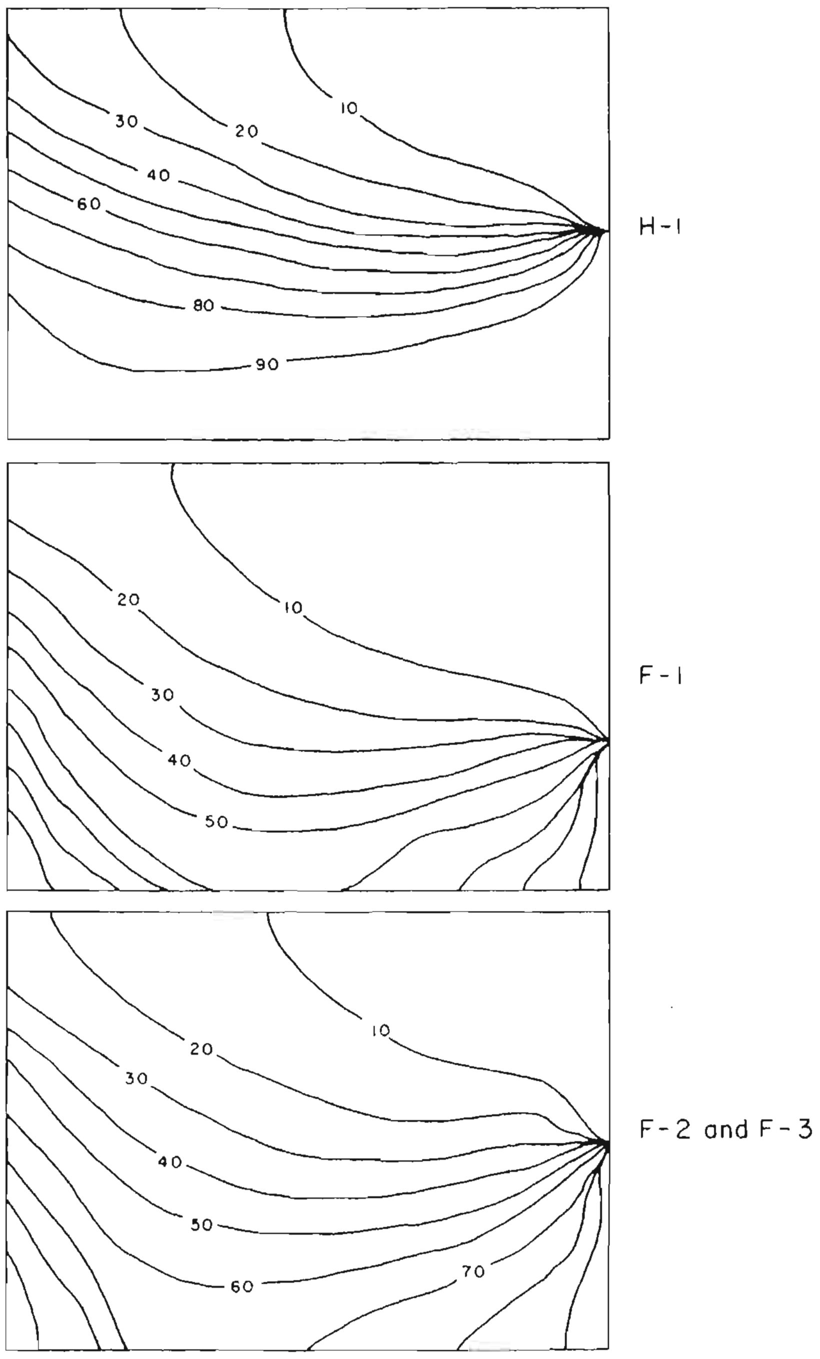


FIGURE 11. STEADY-STATE PREINJECTION ISOCHLORS FOR H-1, F-1, F-2 AND F-3 SIMULATIONS

(those for H-1 are shown in Fig. 8.1 for purposes of comparison). Two major differences are readily apparent. First, F-1 displays an upstream stagnation line (between cols. 7 and 8) as expected for a case with constant recharge, whereas H-1 does not. Second, the difference in buoyant force is readily apparent between the two because of the difference in ambient density (the H-1 simulation showing much greater buoyant effects than the F-1 simulation).

Figure 12 shows the salinity and tracer concentrations as a function of time for F-1. On a gross scale, H-1 and F-1 are quite similar. However, two distinct differences exist. In H-1, isopleths with concentrations less than 50% generally show little sign of buoyant rise and those with concentrations greater than 50% do. Secondly, because of the constant recharge,

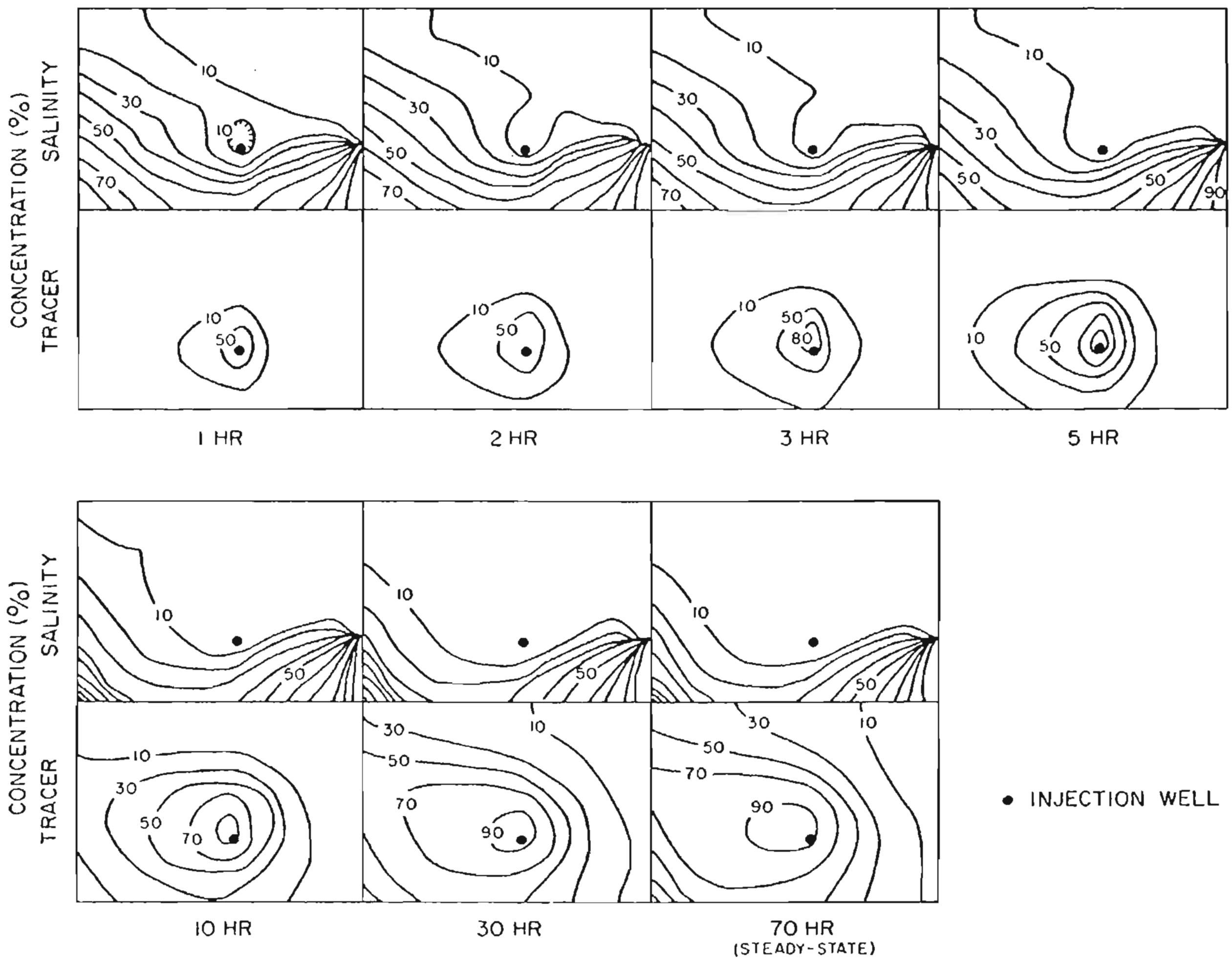
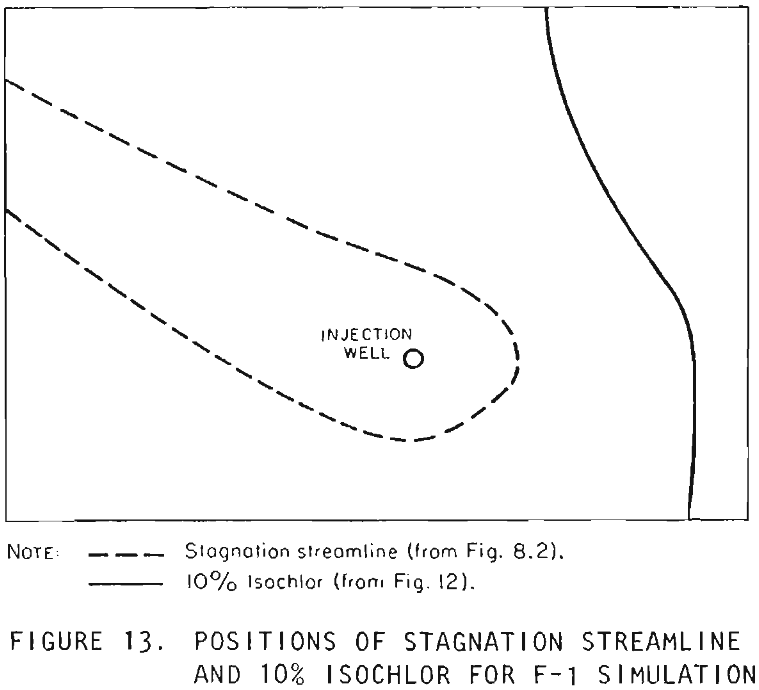


FIGURE 12. TIME-SERIES OF SALINITY AND EFFLUENT TRACER CONCENTRATIONS FOR F-1 SIMULATION

the effluent in F-1 does not have the ability to migrate in concentrated form nearly as far upward into the freshwater zone as it does in H-1. The bulk of the effluent migrates toward the ocean sandwiched between the saline and fresh water.

Several important features should be emphasized about F-1.

1. Owing to the process of hydrodynamic dispersion, the effluent migrates considerably farther upstream than is indicated by the stagnation streamline. This can be seen by comparing Figures 8.2 and 12. The upstream stagnation line falls just about 4% of the aquifer's length to the right of the well in Figure 8.2, while Figure 12 shows that the 10% isopleth extends more than 3 times farther upstream at the same depth as the well. Figure 13 further illustrates this point. The significance of this result will be discussed in a later section.
2. A critical mixing ratio (CMR) can be defined in terms of a particular isopleth which displays little or no buoyant effect. Those isopleths higher in concentration than the CMR do display buoyancy, while those isopleths lower in concentration than the CMR do not display a significant buoyant effect. The CMR isopleth is somewhat variable (both from point to point and from time to time)



NOTE: — — — Stagnation streamline (from Fig. 8.2).
——— 10% Isochlor (from Fig. 12).

FIGURE 13. POSITIONS OF STAGNATION STREAMLINE AND 10% ISOCHLOR FOR F-1 SIMULATION

because the mixing process is (by definition) gradational due to hydrodynamic dispersion. The CMR for H-1 appears to be about 10% or less, while for F-1 it appears to be approximately 50%. The variation between the CMRs for the two experiments reflects the difference in ambient density between the two, as explained earlier.

3. Figure 14 is constructed in the same way as Figure 10, but for the F-1 simulation. At each point along the outflow face at the ocean, there is a mixture of all three fluids and each fluid has its greatest percentage at a specific depth. The ambient flow (fresh-water recharge) has its greatest percentage at the top of the aquifer (58%), while at the same point, 39% of the water emanating into the ocean is effluent and 3% is saline water. The top of the aquifer corresponds to a shallow, near-shore environment. The

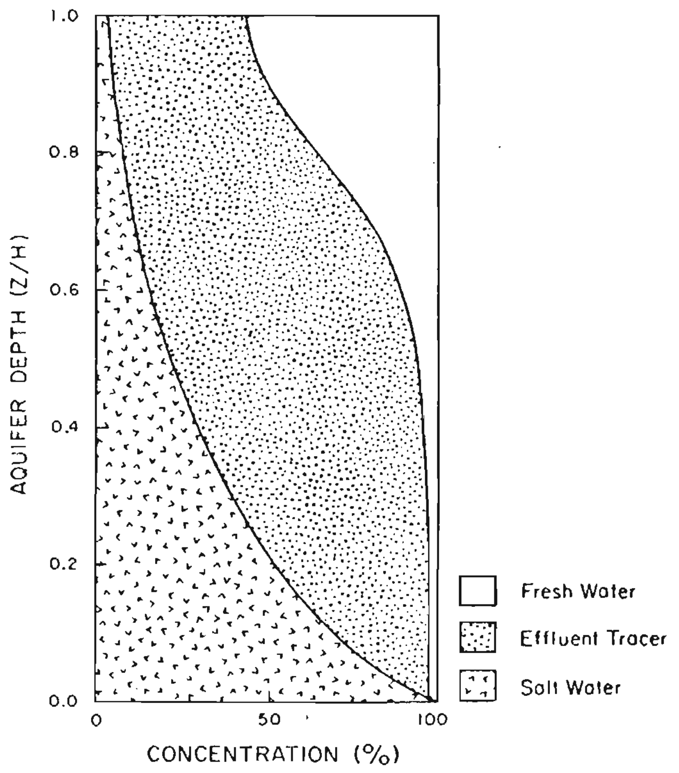


FIGURE 14. RELATIVE PROPORTIONS OF SALINE, EFFLUENT, AND FRESH WATER AS FUNCTION OF DEPTH AT OCEAN-AQUIFER INTERFACE FOR F-1 SIMULATION

effluent has its greatest percentage (73%) of the water flowing into the ocean from the aquifer at a depth of $z/H = 0.55$. Thus, in comparing F-1 with H-1, it can be seen that the largest percentage of effluent flows out of the aquifer near the middle (in the vertical cross section) of the aquifer in F-1, in effect "sandwiched" between fresher water (from ambient flow) on top and more saline water below. On the other hand, in H-1 there was very little ambient flow, so the bulk of the effluent flowed into the ocean near the top of the aquifer, in a shallow, near-shore environment.

Comparison of F Series of Simulations

The differences between simulations F-1, -2, and -3 are best seen by comparing their steady-state distributions of salinity, effluent, and ambient flow (recharge) at the ocean-aquifer interface (left side of aquifer, $x = 0$). Figure 15 is a comparison at the ocean-aquifer interface of the vertical distributions of the three types of waters (saline, effluent, and ambient flow) for simulations F-1, -2, and -3. From Table 2, two differences are apparent between F-1 and F-2. In simulation F-2, the injection well was placed 16.7 cm (20%) farther away from the ocean than in F-1. This was done to see if placing the well farther away from the ocean would disperse (or dilute) the effluent more by traveling through a longer path in the aquifer before reaching the ocean. The highest concentration of effluent tracer in F-1 at the ocean-aquifer interface was 73% (at a height of $z/H = 0.6$), that is, 73% of a water sample hypothetically taken at that depth would be effluent. In F-2, the highest concentration of effluent was 70% (at a height of $z/H = 0.5$); thus, an increase of 20% of the travel distance to the ocean produced only a 4.3% decrease in the maximum effluent concentration entering the ocean from the aquifer.

The other difference between the F-1 and F-2 simulations was that in F-2 the ambient flow was slightly decreased to raise the ambient preinjection salinity around the well to about half that of sea water (Table 2). This decrease in ambient flow is reflected in a decreased amount of fresh water flowing into the ocean and can be seen as a decrease in area of fresh water from F-1 to F-2, as shown in Figure 15. Note that in F-1 the leg of

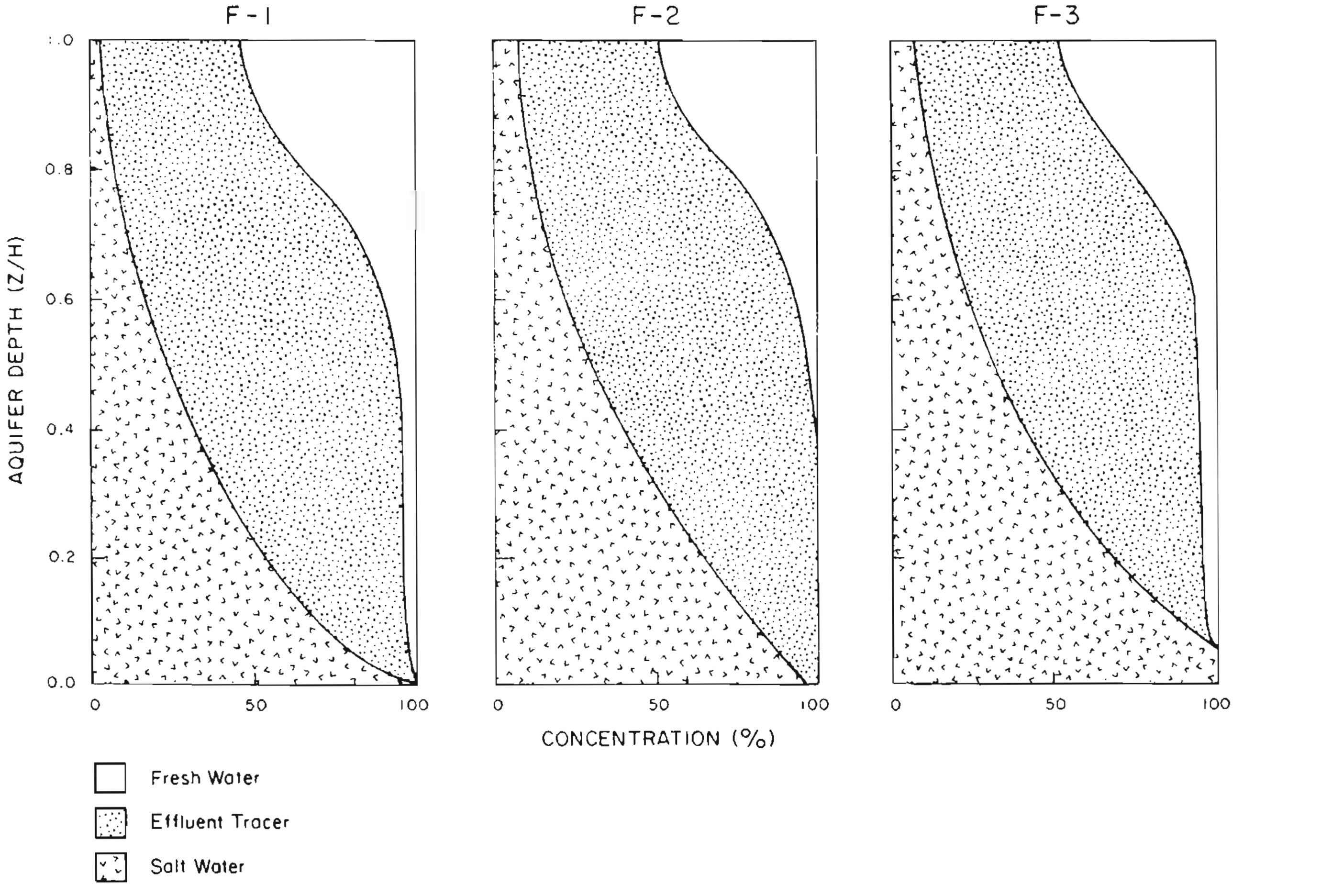


FIGURE 15. RELATIVE PROPORTIONS OF SALINE, EFFLUENT, AND FRESH WATER AS FUNCTION OF DEPTH AT OCEAN-AQUIFER INTERFACE FOR F-1, F-2, AND F-3 SIMULATIONS

fresh water extends all the way to the bottom of the aquifer while in F-2 it pinches out at $z/H = 0.3$. This leg reappears again in F-3, but the area representing the total amount of fresh water escaping to the sea is the same in F-2 and F-3 because in F-3, even though the leg is present, it does not extend to the bottom of the aquifer. The increase in ambient preinjection salinity concentration from F-1 to F-2 causes an increased buoyant affect that results in increased vertical movement of the more concentrated effluent in F-2. This increased vertical movement can be observed by comparing the isopleth movement of F-1 (Fig. 12) to that of F-2 (Fig. 16). In F-2, the isopleths migrate vertically more than those of F-1.

The differences between F-2 and F-3 were that F-3 had a lower injection rate (approximately half as much as F-2), and the F-3 injection well was placed at the same point as the F-1 injection well, i.e., 20% closer to

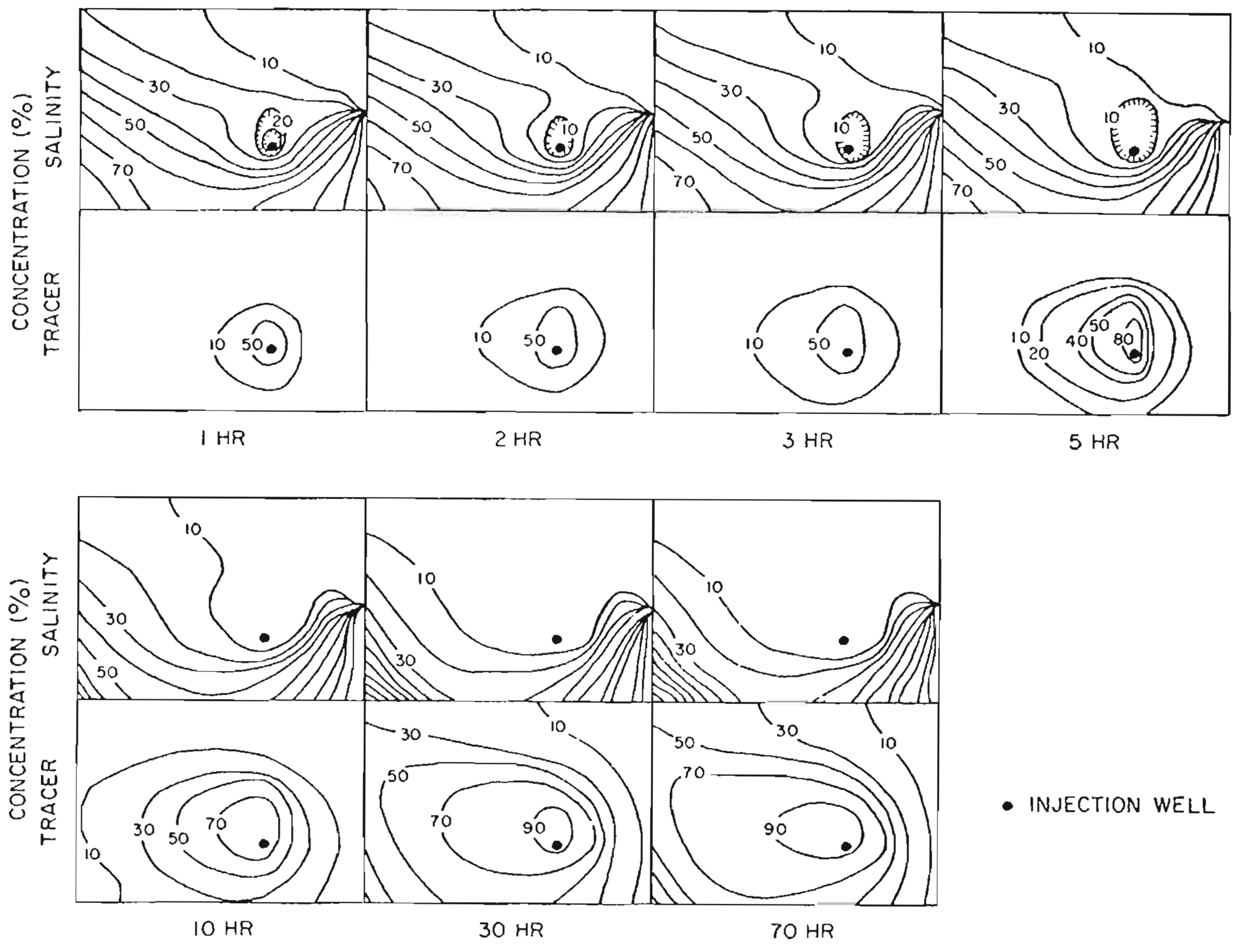


FIGURE 16. TIME-SERIES OF SALINITY AND EFFLUENT TRACER CONCENTRATIONS FOR F-2

the ocean than in F-2. The primary effect of injecting at the significantly lower rate was to allow salt water intrusion into the lower 5% of the aquifer. This effect for F-3 is seen in Figure 15, where the salt water concentration is 100% in the bottom 5% of the aquifer at the ocean-aquifer interface. The increased salinity concentrations due to salt-water intrusion can also be seen in the two-dimensional vertical plane in the lower left of each time-series plot of F-3 in Figure 17.

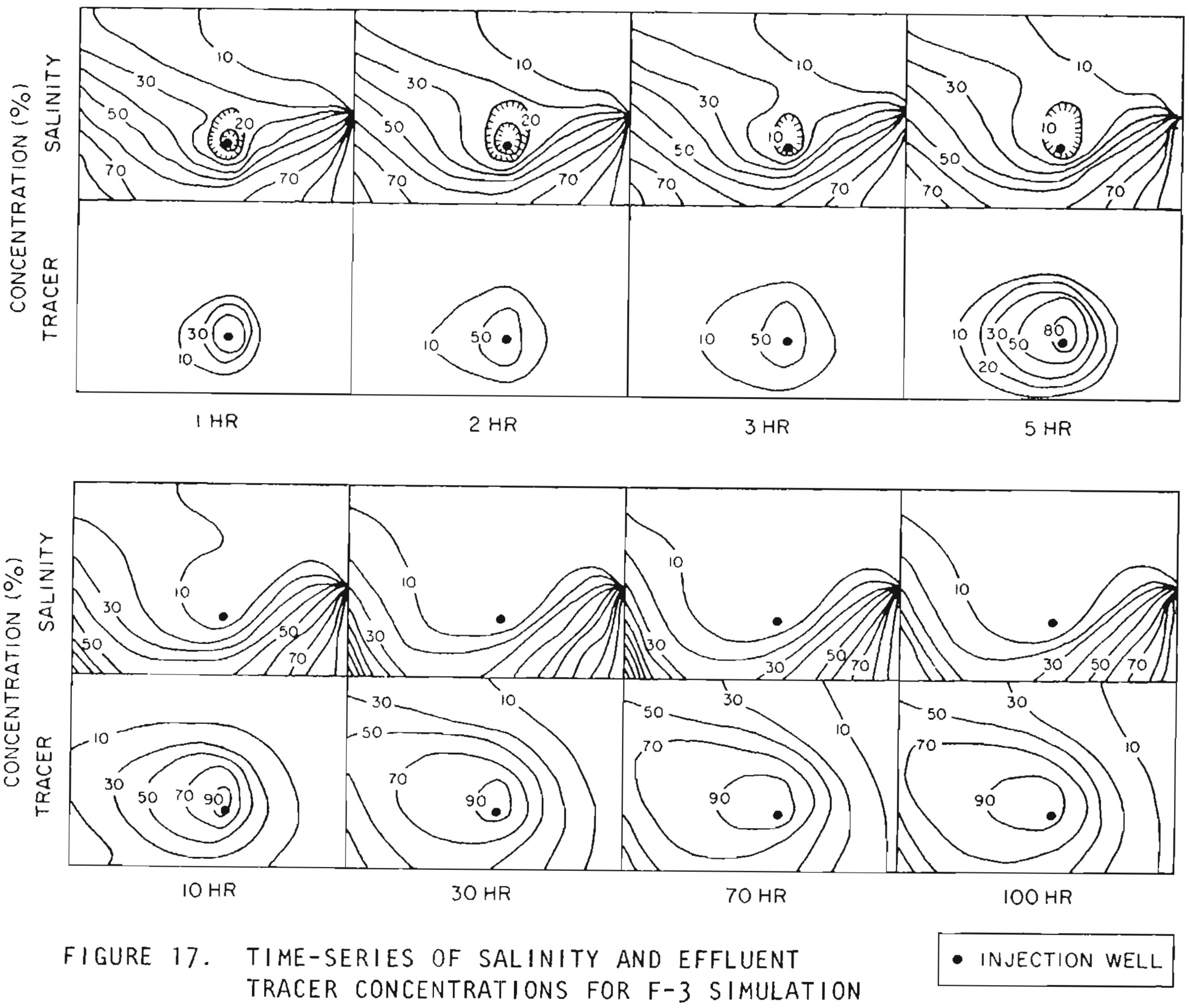


FIGURE 17. TIME-SERIES OF SALINITY AND EFFLUENT TRACER CONCENTRATIONS FOR F-3 SIMULATION

Lower Dispersivity, Anisotropy, and Field-Size Dimension Simulation Results

Attempts to reduce dispersion by decreasing a_L and a_T values were unsuccessful. Values lower than those shown in Table I introduced an unacceptable amount of numerical overshoot. In order to model reduced dispersion (lower values of a_L and a_T), it would have been necessary to reduce Δx and Δz, thus keeping the parameter $\tilde{D}_{nL}$ above a critical minimum value (eq. [48]) so that numerical overshoot would not occur. However, reducing the values of Δx and/or Δz increases the number of nodes, which increases the execution time for the simulation. Attempts to reduce the value of k_z to study anisotropic effects also failed for similar reasons. Reducing k_z had the effect of reducing the vertical velocity (V), which in turn reduced D_{ij} and therefore $\tilde{D}_{nL}$ (eq. [48]), thus, again causing numerical overshoot. The solution to this problem is the same as outlined above for a_L and a_T, to reduce grid spacing, thereby increasing the number of nodes and increasing the cost of the simulation.

Scaling the simulation up to a field-size problem (1 000 m long, 100 m deep) reduced the size of $\tilde{D}_{nL}$ in all of its parameters. D_{ij} was reduced because the velocity was smaller, Δt was larger because of the different time scale, and Δx and Δz were larger because the length scales were larger, but the number of nodes remained the same. Just as in the case of reducing a_L, a_T, and k_z, the smaller values of $\tilde{D}_{nL}$ caused numerical overshoot, thus the solution to simulating a field-size problem is to increase the number of nodes. In order to verify this, the numerical model was run with 1020 nodes (rather than the 210 nodes used for the simulations already discussed) for a few time steps and the numerical overshoot was observed to be eliminated. Only the lack of additional computer funds prevented the completion of this simulation. It is proposed that a further study be conducted to study the effects of lower dispersion, anisotropy, and field-size problems at a time when more computer funds are available.

COMPARISON OF NUMERICAL SIMULATIONS TO SANDBOX-MODEL EXPERIMENTS

Differences Between Sandbox Model and Numerical Simulation

There are three important differences between the sandbox model and the numerical model which make it difficult to make meaningful quantitative comparisons between the two models. These differences are (1) boundary conditions, (2) geometry (three-dimensional in sandbox, two-dimensional in numerical model), and (3) the relative importance of dispersion in each model. The first two points will be discussed in this section while the third will be covered in the following separate section.

To discuss model geometry and boundary conditions, two terms must first be defined. A *saltwater head* refers to the pressure distribution along the ocean-side boundary in the sandbox model that would occur if the entire column of water in the sandbox end chamber were salt water. A *mixed saltwater head* refers to the pressure distribution that would be exerted along the ocean-side boundary of the sandbox if there were a layer of fresh water sitting on top of a layer of salt water in the end chamber. Because fresh water is less dense than salt water, the mixed saltwater head condition produces pressures at each point in the vertical direction along the ocean boundary which are smaller than those produced by a saltwater head (Fig. 18). In the sandbox a mixed saltwater head was always present, whereas in the numerical model a saltwater head was used. In the typical field situation, the saltwater head is usually present (rather than the mixed saltwater head) because regardless of how much fresh water is flowing into the ocean from the aquifer, the fresh water is continually swept away in the ocean. Therefore, the sandbox model would have been a better approximation to the prototype if the ocean-side end chamber had been continually mixed with salt water, thus providing a saltwater head. This was a difficult thing to do physically so it was ignored in the sandbox model experiments, but it is quite simple to do mathematically for the boundary conditions of the numerical model (eq. [23]); thus, the numerical model employs

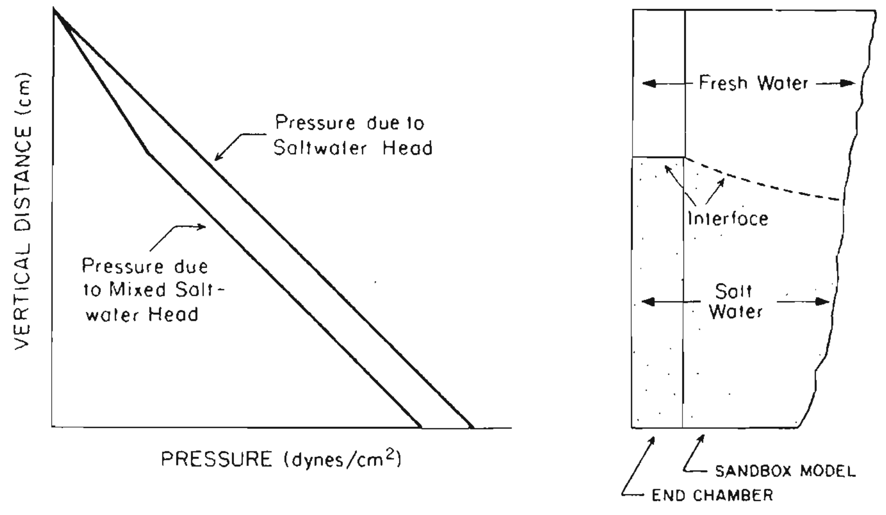


FIGURE 18. DIFFERENCE IN PRESSURE FOR SALTWATER AND MIXED SALTWATER HEADS

a saltwater head and more closely represents the typical prototype situation.

One of the effects of the difference between using saltwater head versus mixed saltwater head was that for the saltwater head, it took a correspondingly larger head on the recharge boundary (in the numerical model) to produce the same specific discharge through the numerical model that was observed for a typical sandbox model experiment with the mixed saltwater head. For instance, in simulation H-1, a head of 1.9 cm was necessary to achieve the same specific discharge rate that a head of 0.75 cm produced in the sandbox model. This value (1.9 cm) was found by adjusting the value of the head in the numerical model until the ambient flow agreed with that in the sandbox model.

The second important difference between the sandbox model experiments and the numerical model is that in the sandbox model, injection is a three-dimensional problem. In the sandbox the injection plume extends back into the model and has a finite horizontal width as it enters the ocean. The numerical model is two-dimensional and corresponds to a situation in which there is an infinite series of closely spaced wells parallel to the ocean. The wells must be close enough to each other so that their effluent plumes coalesce and there is no variation in effluent concentration in the direction parallel to the coastline. In reality, there is never an "infinitely long-strip" of wells, but as long as the total width of the well field, parallel to the coastline, is large compared to the distance to the ocean, then a cross section taken near the center of the finite set of wells, perpendicular to the ocean, will closely resemble a cross section at the center of a two-dimensional infinite strip.

There is of course the question of how many wells would be necessary to provide a good approximation to the infinite strip case. Experience with comparing the results of a two-dimensional Hele-Shaw model and the three-dimensional sandbox model, with one well, has shown that even for just one well, the effluent plume geometry is similar for the two cases on a gross scale (Peterson, Williams, and Wheatcraft 1978). Therefore, even though the sandbox model experiments and the numerical model employ significantly different boundary conditions and flow fields, it is possible to compare the results of the two on a gross qualitative scale.

The last major difference between the sandbox model and the numerical model was that the former modeled water table conditions while the latter modeled confined conditions—a difference that is negligible because the primary difference between the water table and confined conditions is the value of the storage coefficient. The term which includes the storage coefficient is dropped from the continuity equation in this problem (eq. [16]) because it is negligible with respect to other terms in the equations; thus, it is not important whether the aquifer is confined or unconfined.

Determining Relative Importance of Dispersion

Results of the sandbox and Hele-Shaw model studies were quite similar, thus tempting one to ignore the process of hydrodynamic dispersion in theoretical analysis. Yet the results of the present numerical model show that dispersion significantly alters the general nature and appearance of the flow field and mixes salt water, effluent, and recharge in considerable proportions (Figs. 9-12). The values chosen for longitudinal and transverse dispersivities are much larger than the estimated values for the sandbox model. Typical values reported in the literature for a sand with grain size similar to the sandbox range between 1 and 10 cm for longitudinal dispersivity, and approximately one order of magnitude lower than that for the transverse dispersivity. The sand used in the sandbox model was well washed and highly uniform, so it is likely that the dispersivities for the sandbox fall at the lower end of the above range of estimates. The values used in the numerical model were respectively 50 and 10 cm for the longitudinal and transverse dispersivities. These were the smallest values that could be used without causing numerical overshoot.

Because of the difference in dispersion produced by using higher values of the dispersivity in the numerical model than actually occurred in the sandbox model, it is of interest to determine what parameters are important in determining whether the final flow regime will behave more like a two-phase (immiscible) flow field or more like a dispersive flow field, (or like a case between the two extremes). To do this, we will examine the convective-dispersion equation.

For the purposes of this argument, consider the steady-state case and use a coefficient of dispersion that is a constant. With these assumptions, the convective-dispersion equation becomes

$$D_k \nabla^2 C = \nabla \cdot (\underset{\sim}{q}\, C) \ . \tag{51}$$

The term on the right-hand side of the equation becomes

$$\nabla \cdot \underset{\sim}{q}\, C = \underset{\sim}{q} \cdot \nabla C = U\frac{\partial C}{\partial x} + V\frac{\partial C}{\partial z} \ . \qquad (52.1,\ .2)$$

Note that the divergence term has been omitted in (52) because

$$\nabla \cdot \underset{\sim}{q} = 0 \ . \tag{53}$$

Therefore, (51) becomes

$$D_k\ \nabla^2 C = U\frac{\partial C}{\partial x} + V\frac{\partial C}{\partial z} \ . \tag{54}$$

Introducing the following variables into (54):

$$x' = x/H \ , \tag{55.1}$$

$$z' = z/H \ , \tag{55.2}$$

$$U' = U/U_o \ , \tag{55.3}$$

$$V' = V/U_o \ , \tag{55.4}$$

$$C' = C/C_s \ , \tag{55.5}$$

$$Q = U_o\, H \ , \tag{55.6}$$

we have

$$\frac{D_k}{Q}\left[\frac{\partial^2 C'}{\partial (x')^2} + \frac{\partial^2 C'}{\partial (z')^2}\right] = \frac{D_k}{Q}\nabla'^2\ C' = U'\frac{\partial C'}{\partial x'} + V'\frac{\partial C'}{\partial z'} \ . \tag{56}$$

The "primes" will now be dropped from (56) without any loss in clarity.

The dimensionless number Q/D_k is known as the Peclet number, P_e, which can be thought of as the ratio of the convective to dispersive forces. Equation (56) can therefore be written as

$$\frac{1}{P_e}\nabla^2 C = U\frac{\partial C}{\partial x} + V\frac{\partial C}{\partial z} \ . \tag{57}$$

As an approximation, D_k can be assumed to take the form,

$$D_k = a_L\, U_o \ , \tag{58}$$

and the Peclet number becomes

$$P_e = H/a_L \ . \tag{59}$$

For the sandbox model, $P_e = 1000$ (assuming that a_L is on the order of 0.1 cm for the sandbox model). A value for the Peclet number this large renders the dispersive flux term of equation (57), the term on the left-hand side, virtually negligible compared to the convective terms on the right-hand side of equation (57) and the resulting flow field in the sandbox model experiments appears to be nearly immiscible, piston-like flow, i.e., flow by convection only. In the numerical model, a_L was chosen as 50 cm and the resulting value for the Peclet number was on the order of 2, a 500-fold increase over the value in the sandbox model itself.

Thus, there is an inverse relationship between the value of the Peclet number and the importance of dispersion. The larger the Peclet number, the less important hydrodynamic dispersion becomes as a transport mechanism.

A typical field case based on values reported for field dispersivity by other workers (Robertson 1978) might be $H = 100$ m, $a_L = 200$ m. The Peclet number would be 0.5 for this case, making dispersion 4 times more important, as a transport mechanism, than in the numerical model of the sandbox model experiments, and 2000 times more important than in the sandbox experiments themselves. Because of the large discrepancy between the values of the Peclet numbers for the sandbox model and a typical field situation, the sandbox model appears to be a poor representation of the relative importance of hydrodynamic dispersion for a typical field situation.

Similarities Between Sandbox Model and Numerical Simulation

In spite of all the differences between the sandbox model and the numerical model discussed previously in this chapter, there are some interesting similarities which will be discussed here. Three similarities will be discussed in this section: (1) effects due to density-difference between injected and ambient fluids, (2) breakthrough curves for effluent and salinity, and (3) downstream migration of the effluent plume.

The first similarity is illustrated by Figure 19 in which the results of the two sandbox model experiments are shown next to the results of two numerical simulations. For the high ambient density—injection in salt or nearly salt water—a strong vertical migration is seen in the vicinity of the well for both the sandbox and numerical models. Similarly, for the low ambient density, a buoyant migration of effluent is present in both models

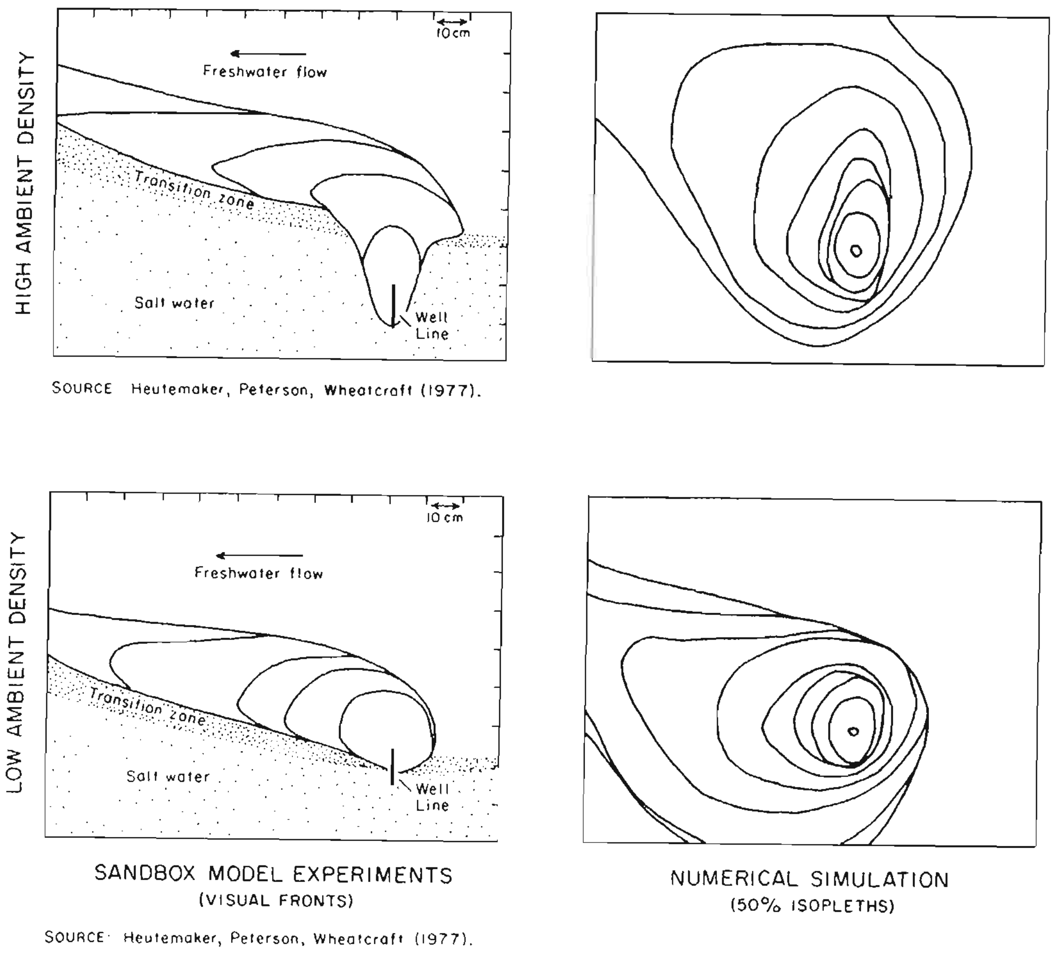
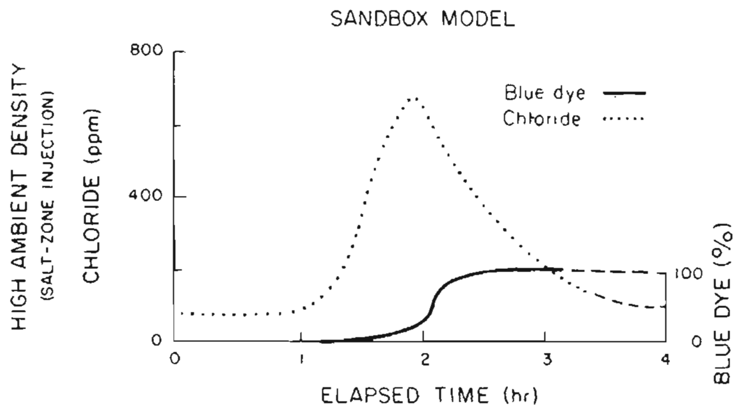


FIGURE 19. COMPARISON OF AMBIENT DENSITY EFFECT ON EFFLUENT TRACER

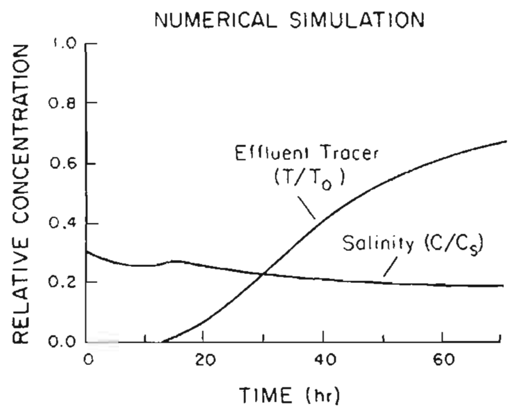
but not nearly as strong as in the case of high ambient density.

Figure 20 illustrates the second similarity. In the sandbox model experiments, a sharp rise in the salinity, as measured by chlorides, was noticed in high ambient density experiments just as the effluent front started to pass by the sample point. For experiments in the transition zone, low ambient density, the same salinity spike did not occur. This same phenomenon is observed in the numerical simulation.

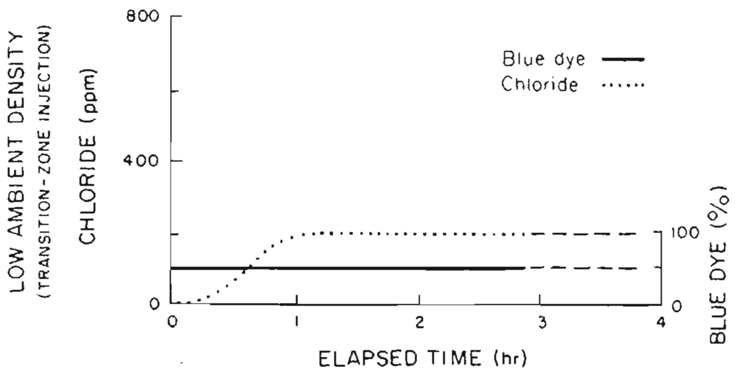
Figure 21 is a plot constructed similar to Figure 15 for an experiment from the sandbox model (the data are taken from the end chamber of the sandbox model interface which is analogous to the ocean-aquifer interface discussed for Fig. 15). It is, in a sense, the limiting effect for the sandbox model which demonstrates almost no dispersion effects. In Figure 21, almost all of the fresh water (from the ambient flow) remains unmixed with the

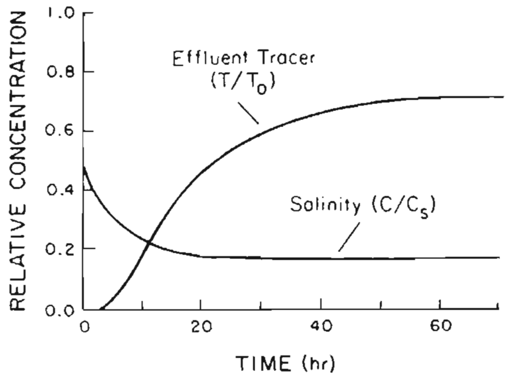


SOURCE: Heutemaker, Peterson, and Wheatcraft (1977).
NOTE: Taken at a point approximately 20 cm downstream from the well.



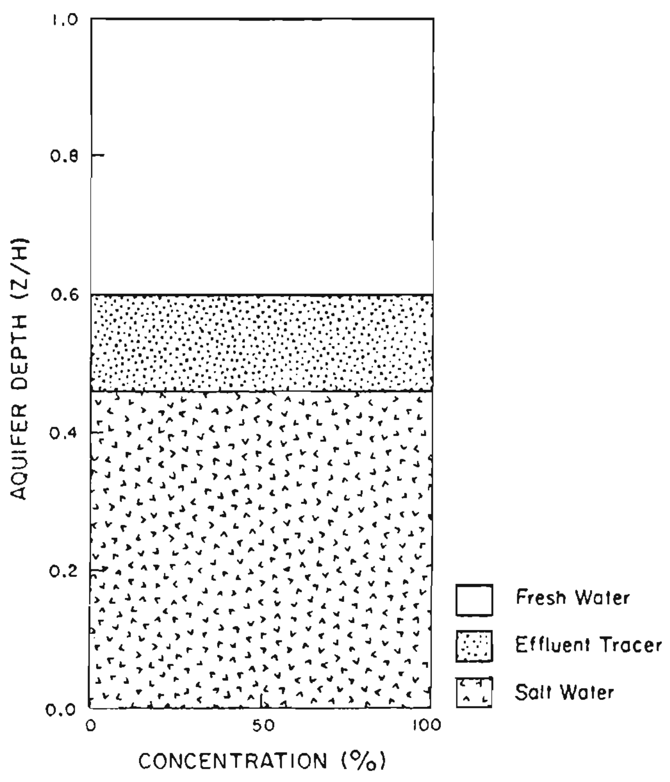
NOTE: Taken at the point X = 0, the ocean boundary.





SOURCE: Heutemaker, Peterson, and Wheatcraft (1977).

FIGURE 20. DOWNSTREAM BREAKTHROUGH CURVES FOR SANDBOX MODEL EXPERIMENTS AND NUMERICAL SIMULATIONS



SOURCE: Heutmaker, Peterson, and Wheatcraft (1977).

FIGURE 21. RELATIVE PROPORTIONS OF SALINE, EFFLUENT, AND FRESH WATER AS FUNCTION OF DEPTH AT OCEAN-AQUIFER INTERFACE, FOR A SANDBOX MODEL EXPERIMENT

effluent and salt water, and flows into the ocean from the top portion of the sandbox model. Similarly, the effluent mixes very little with the fresh water or the salt water and flows out of the sandbox model "sandwiched" between the fresh water (ambient flow) and the salt water. By comparison, in Figure 15 (for numerical simulations F-1, -2, and -3) the three fluids show much greater mixing; however, the water in the lower portion of the aquifer is mostly salt water, in the middle mostly effluent, and in the upper portion mostly fresh water (ambient flow). Thus the largest portion of the effluent flows out of the aquifer "sandwiched" between mostly salt water and mostly fresh (ambient flow) water, in a way very similar to the flow out of the sandbox model.

SUMMARY OF CONCLUSIONS AND RECOMMENDATIONS

Conclusions

The results of the simulations demonstrate several important features which are summarized below.

1. The numerical model developed was shown to correctly solve the governing equations by comparison with other analytic and numerical models. This verification holds only for the saltwater intrusion phase, not the waste injection phase. The waste injection phase uses essentially the same code as the saltwater intrusion phase, with changes only in the boundary conditions, therefore the accuracy of the waste injection phase is reasonably well established by extension, assuming that the boundary conditions are written correctly.
2. The limitations of this model are that it was developed as an interpretive model for a hypothetical situation to study the effects shown by various parameters on the process of waste injection; thus it is an interpretive model, and not a predictive model. The numerical results cannot be applied to a specific field situation until the numerical model has been calibrated with the field data so that it reproduces the history of the field data.
3. The results of the numerical simulations have contributed to the body of knowledge on waste injection. In particular, the simulations indicate that with fairly large dispersivities (similar in value to those which may occur in the field), at least some of the effluent mixes well enough with the ambient brackish water that it no longer has a significant buoyant flow component. This well-mixed effluent is in low concentrations (less than 50%) and propagates away from the well following the same path as the resident fluid. However, much of the effluent remains in high concentrations (greater than 50%) and correspondingly low densities, resulting in a significant buoyant flow component. Thus a strong positive correlation exists between the percent of effluent in the water and the amount of buoyant rise, as displayed by the time-series plots of salinity and effluent tracer. This result is in contrast to the sandbox model experiments in which (due to very small dispersivities) the effluent did not mix at all with resident fluid and rose vertically into the freshwater zone above the salt and brackish water zones.
4. Other parameters varied in the numerical simulations, such as distance from the injection well to the ocean and injection rate, showed only small variations in the results (in keeping with the relatively small variations in the parameters). Moving the injec-

tion well 20% farther from the ocean reduced, i.e., diluted, the maximum concentration of the effluent at the coast by about 5%, whereas a lower injection rate caused the simulation to take longer to go to steady-state.

Recommendations for Future Research

This investigation has concentrated on developing an interpretative numerical model which simulates waste injection into a density-stratified groundwater body. Several recommendations are made for future research which would make use of the basic model developed in this study.

1. It is recommended that a parametric study be undertaken in which selected injection parameters, which are management controllable, be varied. Based on previous work, there appears to be three parameters that are of primary importance in determining the effluent and salinity distribution due to the injection in a density-dependent flow regime, such as the typical sea water intruded, coastal caprock aquifer. The first is the ratio of the injection rate to the recharge rate, or ambient flow field. The second is the ambient density prior to injection, which is directly related to the salinity distribution and the depth of the injection well relative to this distribution. The third parameter of importance is the horizontal injection well location relative to the coastline. In a series of numerical simulations, these three parameters should be systematically varied to determine their effects on the salinity and effluent distributions in the aquifer, and exiting the aquifer into the ocean.
2. Three additional parameters affect the flow field, but are generally not thought of as management controllable. These are longitudinal and transverse dispersivity and anisotrophy. It is suggested that further simulations be run to study the effects of varying these parameters as well.
3. A final suggestion for future research, to take place after the parametric studies recommended above, is to extend the numerical model to a fully three-dimensional system so that the width of the effluent plume as it enters the ocean can be studied with respect to the abovementioned parameters.

ACKNOWLEDGMENTS

The authors wish to express their grateful appreciation to the following persons: L. Stephen Lau for his continued assistance and encouragement throughout the project, John A. Williams, Harold Loomis, William Adams, and Jacob Bear for their many helpful suggestions, and especially the late Gary C. Niemeyer who provided valuable assistance and critical review for the numerical modeling.

GLOSSARY OF SYMBOLS

A	Empirical coefficient used in Eq. (21) to set initial values for $C - A = -23.085$
C	Salinity solute concentration, parts per thousand (dimensionless)
C_o	Initial concentration of salinity in effluent, $C_o = 0$ (dimensionless
C_s	Concentration of TDS, standard sea water (35.8‰)
$\underset{\approx}{D}$	Coefficient of hydrodynamic dispersion, L^2T^{-1}
$\tilde{D}$	Dimensionless dispersion parameter
D_d	Coefficient of molecular diffusion for a solute, L^2T^{-1}
$\underset{\approx}{D}{}_d^*$	Coefficient of molecular diffusion for a porous medium, L^2T^{-1}
D_k	Constant dispersion coefficient used in stability analysis, L^2T^{-1}
$\underset{\approx}{D}{}_m$	Coefficient of mechanical dispersion, L^2T^{-1}
$\tilde{D}_{nL}$	Analogous to D but replacing D_k with d_{ij}, L^2T^{-1}
D_{ij}, D_{mij}, $D^*_{d\;ij}$	Same as previously defined, but using Einstein's summation-convention notation where $i, j = 1, 2$
$\underset{\sim}{g}$	Acceleration of gravity, a vector $= \underset{\sim}{k}\, g$, positive downward, LT^{-2}
H	Depth of aquifer, L
H_r	Head above sea level or right-hand side of model, L
H_w	Total height of water at well, L
$\underset{\approx}{k}$	Intrinsic permeability, a second-rank tensor, L^2
k'	$= k_z/\mu n$, $L^3T\,m^{-1}$
k_x	Intrinsic permeability in x-direction (horizontal), L^2
k_z	Intrinsic permeability in z-direction (vertical), L^2
n	Porosity (dimensionless)
$\underset{\sim}{n}$	Unit normal vector
P	Pressure, $m\,L^{-1}T^{-2}$
P_{os}	Pressure due to weight of a column of salt water, $mL^{-1}T^{-2}$, equal in height to depth, H, of aquifer

$Q(\underset{\sim}{X}^*)$	Integrated fluid injection rate for one cell, $m\ L^{-3}T^{-1}$
$Q_T(\underset{\sim}{X}^*)$	Integrated effluent solute injection rate, $m\ L^{-3}T^{-1}$
$\underset{\sim}{q}$	Specific discharge, a vector, LT^{-1}
$\underset{\sim}{q}_c$	Velocity of salinity solute, LT^{-1}
$\underset{\sim}{q}_T$	Effluent tracer velocity, LT^{-1}
S	Surface, L^2
T	Effluent tracer solute concentration, parts per million (dimensionless)
$\underset{\approx}{T}^*$	Tortuosity (dimensionless)
T^*_{ij}	Same as previously defined, but using Einstein's summation convention notation
T_O	Concentration of effluent solute at well (dimensionless)
t	Time, T
U	Component of specific discharge in x-direction, LT^{-1}
U_O	Specific discharge entering aquifer on right, $x = L$, boundary, LT^{-1}
V	Component of specific discharge in z-direction, LT^{-1}
$\not{V}$	Volume, L^3
x	Independent space variable in horizontal direction, L
x_w	x-coordinate of injection well, L
z	Independent space variable in vertical direction, L
z_R	Point on right boundary below which there is no recharge, L
z_w	z-coordinate of injection well, L
β_c	Empirical multiplication factor to change salinity to density (calculated by using regression analysis on a table of values of density vs. TDS in sea water, from *Handbook of Chemistry and Physics* [1972])
δ	Dirac delta function, = 1 at $\delta(0, 0)$ and = 0 elsewhere
δ_z	Difference operator defined by eq. (43.2)
∇	Vector operator $\underset{\sim}{i}\frac{\partial}{\partial x} + \underset{\sim}{j}\frac{\partial}{\partial y} + \underset{\sim}{k}\frac{\partial}{\partial z}$ (L^{-1})

Δt	Time step size, t
Δx	Grid spacing in x-direction, L
Δz	Grid spacing in z-direction, L
ε_P	Maximum residual error criterion for the iterative solution of the fluid transport equation
ε_T	Truncation error
$\varepsilon(V = 0)$	Truncation error for zero vertical velocity
ρ	Fluid density, mL^{-3}
ρ_o	Density of fresh water, $mL^{-3}0$
‰	Parts per thousand (dimensionless)

REFERENCES

Bear, J. 1972. *Dynamics of fluids in porous media.* New York: American Elsevier.

Cooley, R.L. 1974. *Finite element solutions for the equation of groundwater flow.* Tech. Rep. 18, Center for Water Resources Research, Desert Research Institute, University of Nevada System.

Henry, H.R. 1964. Effects of dispersion of salt encroachment in coastal aquifers. In *Sea Water in Coastal Aquifers*, ed. H.H. Cooper, Jr., F.A. Kohout, H.R. Henry, and R.E. Glover, pp. C70-C84, Water-Supply Paper 1613-C, U.S. Geological Survey.

Heutmaker, D.L.; Peterson, F.L.; and Wheatcraft, S.W. 1977. *A laboratory study of waste injection into a Ghyben-Herzberg groundwater system under dynamic conditions.* Tech. Rep. No. 107, Water Resources Research Center, University of Hawaii.

Hsieh, P. 1977. "Simulation of salt water upconing beneath a well." Water Resources Program, Princeton University, New Jersey.

Kohout, F.A. 1964. The flow of fresh and salt water in the Biscayne aquifer of the Miami area, Florida. In *Sea Water in Coastal Aquifers*, ed. H.H. Cooper, Jr., F.A. Kohout, H.R. Henry, and R.E. Glover, pp. C12-C32, Water-Supply Paper 1613-C, U.S. Geological Survey.

Lee, C.H., and Cheng, R.T. 1974. On seawater encroachment in coastal aquifers. *Water Resources Res.* 10(5):1039-43.

Lenda, A., and Zuber, A. 1970. Tracer dispersion in ground-water experiments. In *Int. Atomic Energy Agency, Isotope Hydrology 1970, Proc. Symp. on Isotope Hydrology*, Vienna, Austria.

McCracken, H.; Voss, C.; Pinder, G.F.; and Ungs, M. 1976. "Block iterative Galerkin finite element package for numerical solution of spatial-transient-nonlinear partial differential equations with one or two dependent variables." Princeton University, New Jersey.

Peaceman, D.W., and Rachford, H.H. 1955. The numerical solution of parabolic and elliptic differential equations. *J. Soc. Ind. Appl. Math.* 3(1):28-41.

Peterson, F.L., and Lau, L.S. 1974. Subsurface waste disposal by injection in Hawaii: A conceptual formulation and physical modeling plan. Tech. Memo. No. 41, Water Resources Research Center, University of Hawaii.

_____; Williams, J.A.; and Wheatcraft, S.W. 1978. Waste injection in a two-phase flow field: Sandbox and Hele-Shaw models study. *Ground Water* 16(6):410-16.

_____, and Petty, S. 1978. "Hawaii waste injection problems. A progress report for the Hawaii State Dept. of Health." Water Resources Research Center, University of Hawaii.

Pinder, G.F., and Cooper, H.H., Jr. 1970. A numerical technique for calculating the transient position of the salt water front. *Water Resources Res.* 6(3):875-82.

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

Remson, I.; Hornberger, G.M.; and Moltz, F.J. 1971. *Numerical methods in subsurface hydrology.* New York: Wiley-Interscience.

Robertson, J.B. 1974. Digital modeling of radioactive and chemical waste transport in the Snake River Plain Aquifer at the National Reactor Testing Station, Idaho.

Shamir, U.Y., and Harleman, D.R.F. 1967. Numerical solutions for dispersion in porous mediums. *Water Resources Res.* 3(2):557-81.

Takasaki, K.J. 1974. Hydrologic conditions related to subsurface and surface disposal of wastes in Hawaii. Open File Rep., U.S. Geological Survey.

Trescott, P.C.; Pinder, G.F.; and Larson, S.P. 1972. *Chapter C1, Finite-difference model for aquifer simulation in two dimensions with results of numerical experiments.* Techniques of Water-Resources Investigations of the United States Geological Survey, Book 7: Automated Data Processing and Computations.

Weast, R.C., ed. 1972. Handbook of chemistry and physics. Cleveland, Ohio: The Chemical Rubber Company.

Wheatcraft, S.W.; Peterson, F.L.; and Heutmaker, D.L. 1976. *Waste injection into a Hawaiian Ghyben-Herzberg aquifer: A laboratory study using a sand-packed hydraulic model.* Tech. Rep. No. 96, Water Resources Research Center, University of Hawaii.

Williams, J.A. 1977. *Well injection into a two-phase flow field: A Hele-Shaw model investigation.* Tech. Rep. No. 108, Water Resources Research Center, University of Hawaii.

APPENDICES

A. ERROR ANALYSIS FOR VERTICAL VELOCITY. 67

B. FLOW CHART FOR FDM COMPUTER PROGRAM 68

C. FDM COMPUTER PROGRAM. 69

APPENDIX A. ERROR ANALYSIS FOR VERTICAL VELOCITY

The truncation error (ε_T) for the central difference representation of the first derivative of pressure can be determined by taking forward and backward the Taylor series expansions of the pressure,

$$P_{i+1,j} = P_{i,j} + (\Delta z)\frac{\partial P}{\partial z}_{i,j} + \frac{(\Delta z)^2}{2!}\frac{\partial^2 P}{\partial z^2}_{i,j} + \frac{(\Delta z)^3}{3!}\frac{\partial^3 P}{\partial z^3}_{i,j} + \ldots \qquad \text{(A.1)}$$

and

$$P_{i-1,j} = P_{i,j} - (\Delta z)\frac{\partial P}{\partial z}_{i,j} + \frac{(\Delta z)^2}{2!}\frac{\partial^2 P}{\partial z^2}_{i,j} - \frac{(\Delta z)^3}{3!}\frac{\partial^3 P}{\partial z^3}_{i,j} + \ldots . \qquad \text{(A.2)}$$

If (A.2) is subtracted from (A.1), we have higher-order terms neglected when rearranged as

$$\frac{\partial P}{\partial z} = \delta_z P - \frac{(\Delta z)^2}{3!}\frac{\partial^3 P}{\partial z^3} , \qquad \text{(A.3.1)}$$

where

$$\delta_z P = \frac{P_{i+1,j} - P_{i-1,j}}{2\Delta z} . \qquad \text{(A.3.2)}$$

Under hydrostatic conditions (where $V = 0$),

$$\frac{\partial P}{\partial z} = - \rho g , \qquad \text{(A.4)}$$

and the second derivative of (A.4) is

$$\frac{\partial^3 P}{\partial z^3} = - g\frac{\partial^2 P}{\partial z^2} . \qquad \text{(A.5)}$$

The second derivative of the equation of state (4) is

$$\frac{\partial^2 \rho}{\partial z^2} = \rho_o \beta_c \frac{\partial^2 C}{\partial z^2} . \qquad \text{(A.6)}$$

Now if (A.6) is substituted into (A.5), and (A.5) in turn into (A.3), the result is

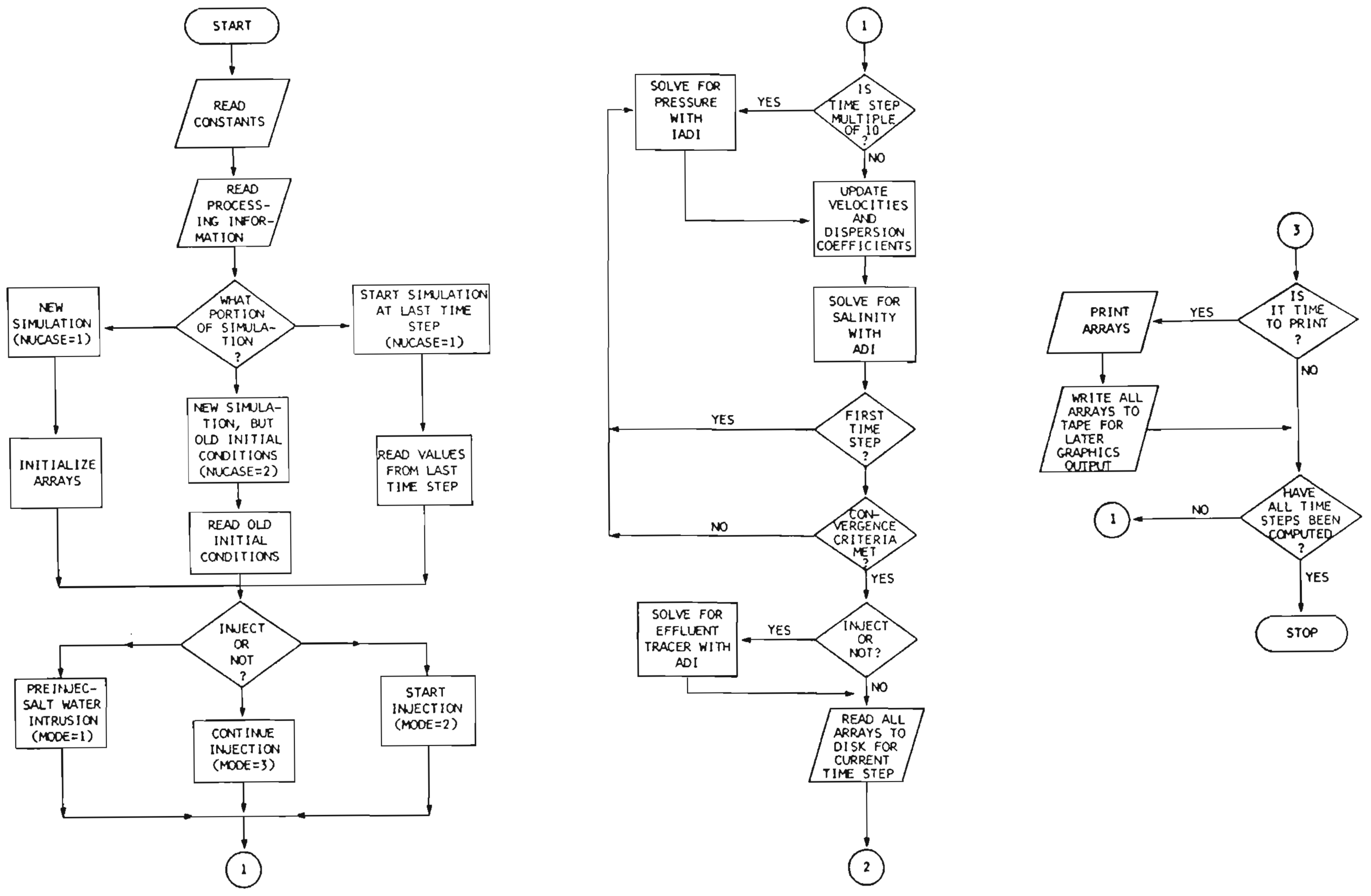
$$\frac{\partial P}{\partial z} = \delta P + \varepsilon_T , \qquad \text{(A.7.1)}$$

where

$$\varepsilon_T = \frac{g(\Delta z)^2}{3!}\rho_o \beta_c \frac{\partial^2 C}{\partial z^2} . \qquad \text{(A.7.2)}$$

Thus, ε_T represents the truncation error (for $V = 0$) in terms of the second derivative of the salinity concentration.

APPENDIX B. FLOW CHART FOR FDM COMPUTER PROGRAM


START
READ CONSTANTS
READ PROCESSING INFORMATION
WHAT PORTION OF SIMULATION ?
NEW SIMULATION (NUCASE=1)
INITIALIZE ARRAYS
NEW SIMULATION, BUT OLD INITIAL CONDITIONS (NUCASE=2)
READ OLD INITIAL CONDITIONS
START SIMULATION AT LAST TIME STEP (NUCASE=1)
READ VALUES FROM LAST TIME STEP
INJECT OR NOT ?
PREINJEC-SALT WATER INTRUSION (MODE=1)
CONTINUE INJECTION (MODE=3)
START INJECTION (MODE=2)
1
1
IS TIME STEP MULTIPLE OF 10 ?
YES
SOLVE FOR PRESSURE WITH IADI
NO
UPDATE VELOCITIES AND DISPERSION COEFFICIENTS
SOLVE FOR SALINITY WITH ADI
FIRST TIME STEP ?
YES
CONVERGENCE CRITERIA MET ?
NO
YES
INJECT OR NOT?
YES
SOLVE FOR EFFLUENT TRACER WITH ADI
NO
READ ALL ARRAYS TO DISK FOR CURRENT TIME STEP
2
3
IS IT TIME TO PRINT ?
YES
PRINT ARRAYS
WRITE ALL ARRAYS TO TAPE FOR LATER GRAPHICS OUTPUT
NO
HAVE ALL TIME STEPS BEEN COMPUTED ?
NO
1
YES
STOP

APPENDIX C. FDM COMPUTER PROGRAM

```
//PLIMP  JOH  (3419.45M,25KI,15KL),'STEVE WHEATCRAFT',MSGLFVFL=(1,1)
//  EXEC SETUP
//SYSIN       DD *
V=UPCONE,T=X21134,ID=UPCONE,SZ=H,RING
/*
//   EXEC   HFORTCG,PARM,COMPILE='OPT=2',PG=700K
//SYSIN  DD  *
      IMPLICIT REAL*8(A-H,O-Z)
      REAL*8 C(10,21,2),P(10,21,2),U(10,21),V(10,21),
     1    ALPHA(21),BETA(21),GAMMA(21),VECKWN(21)
      REAL*8  CT(10,21),CI(10,21),PT(10,21), PI(10,21)
      REAL*8 DXX(10,21),DXZ(10,21),DZZ(10,21)
      REAL*8 ANGLE(10,21),VEL(10,21)
      REAL*8 U0DX(21),U0(21)
      REAL*8 CZRO(10,21)
      REAL*8 RHOP(50)
      REAL*8 T(10,21,2)
      COMMON RHOZRO,BETAC,CZERO,RKX,RKZ,DELTAX,DELTAZ,RMU,G,TODFLX,
     1    TODELZ,DELXSQ,DELZSQ,ALONG,ATRANS,DDT,RKXMU,RKZMU,RKXDXM,
     2    A5,TDX,TDZ,TODDT,ALMAT,TALMAT,A3DDT,A4DDT,
     3    RN,ROBAGZ,ETAP,RMUN,NM1,NM2,MM1,K,KP1
      COMMON/P1/CSEA,IF
      COMMON /P2/ RLENTH,HEIGHT,EPSLON,ACONC,ZZERO,HEAD,UZERO,RECH,EP,
     1    DELTAT,N,M,KD,NTS,NPI
      COMMON /P3/ XL2,ETA,PZROS,PZROF,B1,B2,TB1,TB2,TB1M,TB2M,
     1    B3STAR,B4STAR,B5STAR,EP1M,IRECH
      COMMON /P4/ Q,QDX,QDZ,WX,WZ,DXDZ,DZDX,DXDZG,DZG,IWELL,JWELL,
      COMMON /P6/ RKXDD,RKZDD,DELZ12,MM2
      COMMON /P7/ IWP1,IWM1,JWP1,JWM1
      COMMON /P8/ GDZ2,ROBNDT
      COMMON /P9/ RECHED,ROGRHD
      COMMON /P10/ HPM
      COMMON /P11/ B2GTDZ
      COMMON /P12/ RNDDT
      COMMON /P13/ ITIME
      COMMON /P14/ EFLUNT
      NIPARM = 10
      ISTOP = 1
      CALL RDCTE
      ATRANS = 10.0D0
      ALONG = 50.0D0
      DELTAT = 180.0D0
      RECH = 0.5D0
      READ(5,*) RDUMY,NUCASE
      READ(5,*) RDUMY,N,M,K,KP1,KD,NTS,NPI,NTAPI
      READ(5,*) RDUMY,MODE
      ND = N
      MD = M
      KDD = KD
      CALL CALCK(U0DX,U0,MD)
C
C------------ NUCASE=1; T=0, NEW INITIAL CONDITIONS ------------
C------------ NUCASE=2; T=0, OLD INITIAL CONDITIONS------------
C------------ NUCASE=3; T= LAST TIME STEP ------------
```

```
C
      GO TO (30,40,50),NUCASE
 30   CALL INTL(C,P,CI,PI,CT,PT,U,V,DXX,DXZ,DZZ,U0,U0DX,DZRO,
     .    ND,MD,KDD)
      ENDFILE 12
      WRITE(6,902)
 902  FORMAT(1H1,T53,'T=0, NEW INITIAL CONDITIONS')
      GO TO 60
 40   READ(16) ITS,TIME,C,T,P,CI,PI,CT,PT,U,V,DXX,DXZ,DZZ
      REWIND 16
      WRITE(6,903)
 903  FORMAT(1H1,T53,'T=0, OLD INITIAL CONDITIONS')
      GO TO 60
 50   READ(11) ITS,TIME,C,T,P,CI,PI,CT,PT,U,V,DXX,DXZ,DZZ,CZRO
      WRITE(6,904) ITS
 904  FORMAT(1H1,T51,'T=',I4,', OLD INITIAL CONDITIONS')
 60   CONTINUE
      IWELL = 6
      JWELL = 7
      IF(MODE .GE. 2) C(IWELL,JWELL,KP1) = 0.0D0
      EPSLNP = 0.001D0
      RDXDD = RDX/(12.0D0*DELTAX*RMU)
      MM2 = M-2
      RKZDD = RKZ/RMU
      DELZ12 = 12,0D0*DELTAZ
      WELHED = 103.5D0
      ZWELL = DELTAZ*(DFLOAT(JWELL-1))
      PZERO = RHOZRO*G*(WELHED-ZWELL)
      IF(MODE .GE. 2) P(IWELL,JWELL,KP1) = PZERO
      IWP1 = IWELL+1
      IWM1 = IWELL-1
      JWP1 = JWELL+1
      JWM1 = JWELL-1
      GDZ2 = G*DELTAZ/2.0D0
      EFLUNT = 1.0D0
      ROBNDT = RN*RHOZRO*BETAC*A5
      RECHED = 2.0D0
      ROGRHD = RHOZRO*G*RECHED
      RNDDT = RN*DDT
C
C------------MODE = 1; PRE-INJECTION ------------
C------------- MODE = 2; START INJECTION ------------
C------------ MODE = 3; INJECTION PHASE ------------
C
      IF(ITS .EQ. 0 .AND. MODE .EQ. 1) GO TO 75
      GO TO 76
 75   CONTINUE
      DO 31 J-1,M
      C(1,J,KP1) = 35.8D0
 31   CONTINUE
C------------ PRESSURE INTEGRATION GOES HERE ---------
      CALL INPRES(C,P,1,ND,MD,KDD)
      DO 32 J-1,M
 32   C(1,J,KP1) = CI(1,J)
 76   CONTINUE
```

```
      IF (MODE .EQ. 2) GO TO 70
      GO TO 71
 70   CONTINUE
      DO 72 J=1,M
      DO 72 I=1,N
      CZRO(I,J) = C(I,J,KP1)
 72   CONTINUE
      TIME = 0.0D0
      ITS = 0
 71   CONTINUE
C
C
      B2GTDZ = B2*G*TODELZ
      WRITE(6,950)
 950  FORMAT(1H1,T55,'INITIAL CONCENTRATIONS')
      CALL PRINT3(C,N,M,KD,DELTAX,DELTAZ)
      CALL PRT3DF(P,N,M,KD,DELTAX,DELTAZ)
      CALL COMHPM(RHOP.NIPARM)
      EPSLON = 0.5D0
      NPRESI = 10
      KOUNT = 0
      ITSP1 = ITS + 1
      DO 100 ITIME=ITSP1,NTS
      WRITE(6,977) ITIME
 977  FORMAT(1H ,'K = ',I5)
      IF(ITIME .EQ. 1) GO TO 499
      IF((ITIME/NPRESI)*NPRESI .NE. ITIME) GO TO 501
 499  CONTINUE
      DO 500 ITER = 1,100
      IF(((ITER-1)/NIPARM)*NIPARM .EQ. (ITER-1)) KOUNT = 0
      KOUNT = KOUNT + 1
      HPM = RHOP(KOUNT)*(TB1+TB2)
C
C------------START ITERATION ON COLUMNS (I=COLUMNS) FOR PRESSURE------------
C
      IF = 2
      L = N
      DO 8 J=1,M
      IF(MODE .GE. 2 .AND. J .EQ. JWELL)
     .CALL APIWFX(C,P,U0DX,ALPHA,BETA,GAMMA,VECKWN,ND,MD,KDD,J,&8)
      CALL AIPFLX(C,P,U0DX,ALPHA,BETA,GAMMA,VECKWN,ND,MD,KDD,J)
      CALL TRIDGI(IF,L,ALPHA,BETA,GAMMA,VECKWN,P,N,M,KD,J)
      CALL STUF21(P,ND,MD,KDD)
C
C------------START ITERATION ON ROWS(J=POWS) FOR PRESSURE------------
C
      IF = 1
      L = M
      DO 6 I-2,N
      IF(MODE .GE. 2 .AND. I .EQ. IWELL)
     .CALL APJWEL(C,P,ALPHA,BETA,GAMMA,VECKWN,ND,MD,KDD,I,&6)
      CALL AJPFLX(C,P,U0,U0DX,ALPHA,BETA,GAMMA,VECKWN,ND,MD,KDD,I)
      CALL TRIDAG(IF,L,ALPHA,BETA,GAMMA,VECKWN,P,N,M,KD,I)
 6    CONTINUE
      CALL CHKCON(P,EPSLNP,ICON,ND,MD,KDD)
      CALL STUF21(P,ND,MD,KDD)
      IF (ICON .EQ. 1) GO TO 501
  500 CONTINUE
      STOP 500
  501 CONTINUE
      DO 700 ICOEF=1,100
      CALL VELOC(C,P,U,V,N,M,KD)
      CALL DIJ(DXX,DXZ,DZZ,U,V,N,M)
C
C------------START ITERATION ON COLUMNS (I= COLUMNS)------------
C
      L = NM1
      DO 5 J=1,M
      IF = 1
      IF(MODE .GE. 2 .AND. J .EQ. JWELL)
     .CALL AIWELL(C,U,V,DXX,DXZ,DZZ,ALPHA,BETA,GAMMA,VECKWN,CW,ND,MD,KDD
     .,J,&5)
      CALL ASMBLI(C,U,V,DXX,DXZ,DZZ,ALPHA,BETA,GAMMA,VECKWN,N,M,KD,J)
      CALL TRIDGI(IF,L,ALPHA,BETA,GAMMA,VECKWN,C,N,M,KD,J)
 5    CONTINUE
      CALL STUF21(C,ND,MD,KDD)
C
C------------START  ITERATION  ON ROWS (J=ROWS)------------
C
      L = M
      DO 4 I=1,NM1
      IF = 1
      IF(MODE .GE. 2 .AND. I .EQ. IWELL)
     .CALL AJWELL(C,U,V,DXX,DXZ,DZZ,ALPHA,BETA,GAMMA,VECKWN,CW,ND,MD,KDD
     .,I,&4)
      CALL ASMBLJ(C,U,V,DXX,DXZ,DZZ,ALPHA,BETA,GAMMA,VECKWN,N,M,KD,I)
      CALL TRIDAG(IF,L,ALPHA,BETA,GAMMA,VECKWN,C,N,M,KD,I)
 4    CONTINUE
      IF (ICOEF .EQ. 1) GO TO 699
C
C------------ CHECK VALUES OF CONCENTRATION FOR CONVERGENCE ------------
C
      DIF = 0.0D0
      DO 23 J = 1,M
      DO 23 I=1,N
      DIFNEW = DABS(C(I,J,KP1)-CT(I,J))
      IF (DIFNEW .GT. DIF) DIF=DIFNEW
 23   CONTINUE
      WRITE(6,900) DIF
 900  FORMAT(1H ,'GREATEST DIFFERENCE IN CONCENTRATION =',D20.10)
      IF(DIF .LT. EPSLON) GO TO 701
 699  CONTINUE
      DO 21 J=1,M
      DO 21 I=1,N
      CT(I,J) = C(I,J,KP1)
      PT(I,J) = P(I,J,KP1)
      C(I,J,K) = CI(I,J)
 21   CONTINUE
 700  CONTINUE
      STOP 700
 701  CONTINUE
C
C------------ COMPUTE TRACER EFFLUENT CONCENTRATIONS ---------
C
      IF(MODE .GE. 2)
     .CALL TRACER (T,U,V,DXX,DXZ,DZZ,ALPHA,BETA,GAMMA,VECKWN,ND,MD,KDD)
      TIME = TIME + DELTAT/3600.0D0
      IF((ITIME/NPI)*NPI .EQ. ITIME) GO TO 101
      GO TO 98
 101  CONTINUE
      WRITE(6,919) TIME,ITIME
 919  FORMAT(1H1,'SIMULATION TIME =',F8.2,'   HOURS,',' K = ',I6)
      WRITE(6,905)
 905  FORMAT(1H0,T49,'CONCENTRATIONS AT THE K+1 TIME STEP')
      CALL PRINT3(C,N,M,KD,DELTAX,DELTAZ)
      WRITE(6,991)
      CALL PRTDIF(C,CZRO,ND,MD,KDD)
      IF(MODE .GE. 2) WRITE(6,930)
 930  FORMAT(1H1,T56,'TRACER CONCENTRATIONS')
      IF(MODE .GE. 2) CALL PRINT3(T,N,M,KD,DELTAX,DELTAZ)
      WRITE(6,991)
 991  FORMAT(1H1)
      CALL EXHIB(U,V,ANGLE,VEL,N,M)
 98   CONTINUE
      DO 99 I=1,N
      DO 99 J=1,M
      C(I,J,K) = C(I,J,KP1)
      CI(I,J) = C(I,J,KP1)
      P(I,J,K) = P(I,J,KP1)
 99   PI(I,J) = P(I,J,KP1)
      REWIND 11
      WRITE(11) ITIME,TIME,C,T,P,CI,PI,CT,PT,U,V,DXX,DXZ,DZZ,CZRO
      ENDFILE 11
      IF(MODE .GE. 2 .AND. (ITIME/NTAPI)*NTAPI .EQ. ITIME)
     .WRITE(17) ITIME,TIME,C,T,P,CI,PI,CT,PT,U,V,DXX,DXZ,DZZ,CZRO
 100  CONTINUE
      STOP
      END
//GO.SYSLIB DD DISP=(OLD,KEEP),DSN=B.B3419.NJECT.FORTSUB.NEW
//          DD DSN=SYS1,FORTLBX,DISP=SHR
//          DD DSN=SYS1,FORTSUB,DISP=SHR
//GO.SYSIN DD *
'NUCASE' 3
'N,M,K,KP1,KD,NTS,NPI,NTAPI' 10 21 1 2 2 2000 50 10
'MODE' 3
//GO.FT10F001 DD DISP=SHP,DSN=T034190.GRADR.DATA
//GO.FT11F001 DD DISP=(OLD,KEEP),DSN=B.B3419.INJECT.DATA
//GO.FT12F001 DD DISP=(OLD,KEEP),DSN=B.B3419.INTL.DATA
//GO.FT16F001 DD DUMMY
//GO.FT17F001 DD DISP=(MOD,KEEP),UNIT=XTRK,VOL=SER=UPCONE.
//             LABEL=(8,SL),DSN=FLUX3
//
```

```
      SUBROUTINE RDCTE
      IMPLICIT REAL*8 (A-H,O-Z)
      COMMON RHOZRO,BETAC,CZERO,RKX,RKZ,DELTAX,DELTAZ,RMU,G,TODELX,
     1     TODELZ,DELXSQ,DELZSQ,ALONG,ATRANS,DDT,RKXMU,RKZMU,RKXDXM,
     2     A5,TDX,TDZ,TODDT,ALMAT,TALMAT,A3DDT,A4DDT,
     3     RN,ROBAGZ,ETAP,RMUN,NM1,NM2,MM1,K,KP1
      COMMON/P1/CSEA,IF
      COMMON /P2/ RLENTH,HEIGHT,EPSLON,ACONC,ZZERO,HEAD,UZERO,RECH,EP,
     1     DELTAT,N,M,KD,NTS,NPI
      READ(10,901) RLENTH
      READ(10,901) HEIGHT
      READ(10,901) DELTAT
      READ(10,901) EPSLON
      READ(10,901) BETAC
      READ(10,901) CZERO
      READ(10,901) RHOZRO
      READ(10,901) CSEA
      READ(10,901) RKX
      READ(10,901) RKZ
      READ(10,901) RMU
      READ(10,901) G
      READ(10,901) DDT
      READ(10,901) ATRANS
      READ(10,901) ALONG
      READ(10,901) ACONC
      READ(10.901) ZZERO
      READ(10,901) HEAD
      READ(10,901) RN
      READ(10,901) UZERO
      READ(10,901) RECH
      READ(10,901) EP
  901 FORMAT(10X,D20.10)
      RETURN
      END

      SUBROUTINE CALCK(U0DX,U0,MD)
      IMPLICIT REAL*8(A-H,O-Z)
      REAL*8 U0(MD),U0DX(MD)
      COMMON RHOZRO,BETAC,CZERO,RKX,RKZ,DELTAX,DELTAZ,RMU,G,TODELX,
     1     TODELZ,DELXSQ,DELZSQ,ALONG,ATRANS,DDT,RKXMU,RKZMU,RKXDXM,
     2     A5,TDX,TDZ,TODDT,ALMAT,TALMAT,A3DDT,A4DDT,
     3     RN,ROBAGZ,ETAP,RMUN,NM1,NM2,MM1,K,KP1
      COMMON/P1/CSEA,IF
      COMMON /P2/ RLENTH,HEIGHT,EPSLON,ACONC,ZZERO,HEAD,UZERO,RECH,EP,
     1     DELTAT,N,M,KD,NTS,NPI
      COMMON /P3/ XL2,ETA,PZROS,PZROF,B1,B2,TB1,TB2,TB1M,TB2M,
     1     B3STAR,B4STAR,B5STAR,EP1M,IRECH
      NM1 = N-1
      NM2 = N-2
      MM1 = M-1
      DELTAX = RLENTH/DFLOAT(NM1)
      DELTAZ = HEIGHT/DFLOAT(MM1)
      TODELX = 2.0D0*DELTAX
      TODELZ = 2.0D0*DELTAZ
      DELZSQ = DELTAZ*DELTAZ
```

[redacted]

```
      DELXSQ = DELTAX*DELTAX
      RKXMU = RKX/(TODELX*RMU)
      RKZMU = RKZ/RMU
      RKXDXM = RKX/(DELTAX*RMU)
      A5 = 2.0D0/DELTAT
      XL = HEIGHT
      XL2 = XL/2.0 D 00
      ETA = (ZZERO - XL2)*DATAN(ZZERO-XL2)
     $   -0.5*DLOG(1.0+(ZZERO-XL2)**2)
      Z = XL
      PZROS= - RHOZRO*G*((BETAC*CZERO-1.0)*(Z-ZZERO)
     $   -BETAC*(0.5*ACONC*((Z-XL2)*DATAN(Z-XL2)
     $   -0.5*DLOG(1.0+(Z-XL2)**2)-ETA)+0.5*CSEA
     $   *(Z-ZZERO)))
      Z = XL + HEAD
      PZROF= - RHOZRO*G*((BETAC*CZERO-1.0)*(Z-ZZERO)
     $   -BETAC*(0.5*ACONC*((Z-XL2)*DATAN(Z-XL2)
     $   -0.5*DLOG(1.0+(Z-XL2)**2)-ETA)+0.5*CSEA
     $   *(Z-ZZERO)))
      TODDT = 2.0D0*DDT
      ALMAT = ALONG - ATRANS
      TALMAT = 2.0D0*ALMAT
      A3DDT = TODDT/DELXSQ
      A4DDT = TODDT/DELZSQ
      B1 = RKX/(RMU*DELXSQ)
      B2 = RKZ/(RMU*DELZSQ)
      TB1 = 2.0D0*B1
      TB2 = 2.0D0*B2
      TB1M = -TB1
      TB2M = -TB2
      B3STAR = RHOZRO*BETAC*RKX/(RMU*TODELX)
      B4STAR = RHOZRO*BETAC*RKZ/(RMU*TODELZ)
      B5STAR = -2.0D0*RHOZRO*BETAC*RKZ*G/RMU
      DU0DX = -TODELX*RMU*UZERO/RKX
      DU0 = RMU*UZERO/RKX
      IRECH = (1.0D0-RECH)*(M-1)
      DO 26 J=1,M
      IF(J .LE. IRECH) U0DX(J) = 0.0D0
      IF(J .LE. IRECH) U0(J) = 0.0D0
      IF(J .GT. IRECH) U0DX(J) = DU0DX
      IF (J .GT. IRECH) U0(J) = DU0
26    CONTINUE
      DO 9998 J=1,M
9998  WRITE(6,907) U0DX(J),U0(J)
907   FORMAT(1H ,2(1PD20.10))
      EP1M = 1.0D0-EP
      RETURN
      END

      SUBROUTINE INTL(C,P,CI,PI,CT,PT,U,V,DXX,DXZ,DZZ,U0,U0DX,
     .    CZRO,ND,MD,KDD)
      IMPLICIT REAL*8 (A-H,O-Z)
      REAL*8 C(ND,MD,KDD),P(ND,MD,KDD),U(ND,MD),V(ND,MD),U0(MD),U0DX(MD)
      REAL*8 CI (ND,MD),PI(ND,MD),CT(ND,MD),PT(ND,MD)
      REAL*8 DXX(ND,MD),DXZ(ND,MD),DZZ(ND,MD)
      REAL*8 CZRO(ND,MD)
      COMMON RHOZRO,BETAC,CZERO,RKX,RKZ,DELTAX,DELTAZ,RMU,G,TODELX,
     1    TODELZ,DELXSQ,DELZSQ,ALONG,ATRANS,DDT,RKXMU,RKZMU,RKXDXM,
     2    A5,TDX,TDZ,TODDT,ALMAT,TALMAT,A3DDT,A4DDT,
     3    RN,ROBAGZ,ETAP,RMUN,NM1,NM2,MM1,K,KP1
      COMMON/P1/CSEA,IF
      COMMON /P2/ RLENTH,HEIGHT,EPSLON,ACONC,ZZERO,HEAD,UZERO,RECH,EP,
     1    DELTAT,N,M,KD,NTS,NPI
      COMMON /P3/ XL2,ETA,PZROS,PZROF,B1,B2,TB1,TB2,TB1M,TB2M,
     1    B3STAR,B4STAR,B5STAR,EP1M,IRECH
      N = ND
      M = MD
C
C------------ INITIALIZE CONCENTRATION AND PRESSURE------------
C
      DO 1 J = 1,M
      Z = DELTAZ*DFLOAT(J-1)
      CDUMY = (ACONC*(DATAN(Z-XL2))+CSEA)*0.5
      PDUMYS = RHOZRO*G*((BETAC*CZERO-1.0)*(Z-ZZERO)
     $   -BETAC*(0.5*ACONC*((Z-XL2)*DATAN(Z-XL2)
     $   -0.5*DLOG(1.0+(Z-XL2)**2)-ETA)+0.5*CSEA
     $   *(Z-ZZERO)))+PZROS
      P(1,J,K) = PDUMYS
      P(1,J,KP1) = P(1,J,K)
      C(N,J,KP1) = CDUMY
      DO 3 I = 1,N
      C(I,J,K) = CDUMY
      C(I,J,KP1) = C(I,J,K)
      P(I,J,K) = PDUMYS
      P(I,J,KP1) = P(I,J,K)
      U(I,J) = UZERO
      V(I,J) = UZERO*1.0D-04
 3    CONTINUE
 1    CONTINUE
      DO 2 J = L,M
      Z = DELTAZ*DFLOAT(J-1)
      P(N,J,K) = RHOZRO*G*((BETAC*CZERO-1.0)*(Z-ZZERO)
     $   -BETAC*(0.5*ACONC*((Z-XL2)*DATAN(Z-XL2)
     $   -0.5*DLOG(1.0+(Z-XL2)**2)-ETA)+0.5*CSEA
     $   *(Z-ZZERO)))+PZROF
 2    P(N,J,KP1) = P(N,J,K)
      DO 111 J=1,M
      IF(J .LE. IRECH) C(N,J,K) = CSEA
      IF(J .GT. IRECH) C(N,J,K) = 0.0D0
 111  C(N,J,KP1) = C(N,J,K)
      DO 20 J=1,M
      DO 20 I=1,N
      CT(I,J) = C(I,J,K)
      PT(I,J) = P(I,J,K)
      CI(I,J) = C(I,J,K)
 20   PI(I,J) = P(I,J,K)
      DO 750 J=1,M
      DO 750 I=1,N
 750  CZRO(I,J) = C(I,J,K)
```

```
      CALL DIJ(DXX,DXZ,DZZ,U,V,N,M)
      ITS = 0
      TIME = 0.0D0
      WRITE(12) ITS,TIME,C,P,CI,PI,CT,PT,U,V,DXX,DXZ,DZZ,CZRO
      RETURN
      END

      SUBROUTINE VELOC(C,P,U,V,N,M,KD)
      IMPLICIT REAL*8(A-H,O-Z)
      REAL*8 C(N,M,KD),P(N,M,KD),U(N,M),V(N,M)
      COMMON RHOZRO,BETAC,CZERO,RKX,RKZ,DELTAX,DELTAZ,RMU,G,TODELX,
     1    TODELZ,DELXSQ,DELZSO,ALONG,ATRANS,DDT,RKXMU,RKZMU,RKXDXM,
     2    A5,TDX,TDZ,TODDT,ALMAT,TALMAT,A3DDT,A4DDT,
     3    RN,ROBAGZ,ETAP,RMUN,NM1,NM2,MM1,K,KP1
      L = 2
      DO 100 I=2,NM1
      IP1 = I+1
      IM1 = I-1
      DO 100 J=2,MM1
      JP1 = J+1
      JM1 = J-1
      RHO = RHOZRO*(1.0D0+(BETAC*(C(I,J,L)-CZERO)))
      U(I,J) = -RKXMU*(P(IP1,J,L)-P(IM1,J,L))
100   V(I,J) = -RKZMU*(((P(I,JP1,L)-P(I,JM1,L))/TODELZ)+RHO*G)
      DO 200 I=2,NM1
      IP1 = I+1
      IM1 = I-1
      U(I,1) = -RKXMU*(P(IP1,1,L)-P(IM1,1,L))
      U(I,M) = -RKXMU*(P(IP1,M,L)-P(IM1,M,L))
      RHO = RHOZRO*(1.0D0+(BETAC*(C(I,1,L)-CZERO)))
      V(I,1) = -RKZMU*(((P(I,2,L)-P(I,1,L))/DELTAZ)+RHO*G)
      RHO = RHOZRO*(1.0D0+(BETAC*(C(I,M,L)-CZERO)))
      V(I,M) = -RKZMU*(((P(I,M,L)-P(I,MM1,L))/DELTAZ)+RHO*G)
200   CONTINUE
      DO 300 J=2,MM1
      JP1 = J+1
      JM1 = J-1
      RHO = RHOZRO*(L.0D0+(BETAC*(C(1,J,L)-CZERO)))
      V(1,J) = -RKZMU*(((P(1,JP1,L)-P(1,JM1,L))/TODELZ)+RHO*G)
      RHO = RHOZRO*(1.0D0+(BETAC*(C(N,J,L)-CZERO)))
      V(N,J) = -RKZMU*(((P(N,JP1,L)-P(N,JM1,L))/TODELZ)+RHO*G)
      U(1,J) = -RKXDXM*(P(2,J,L)-P(1,J,K))
      U(N,J) = -RKXDXM*(P(N,J,L)-P(NM1,J,L))
300   CONTINUE
      U(1,1) = -RKXDXM*(P(2,1,L)-P(1,1,L))
      U(1,M) = -RKXDXM*(P(2,M,L)-P(1,M,L))
      U(N,1) = -RKXDXM*(P(N,1,L)-P(NM1,1,L))
      U(N,M) = -RKXDXM*(P(N,M,L)-P(NM1,M,L))
      V(1,1) = 0.0D0
      V(1,M) = 0.0D0
      V(N,1) = 0.0D0
      V(N,M) = 0.0D0
      RETURN
      END
      SUBROUTINE INPRES(C,P,I,ND,MD,KDD)
      IMPLICIT REAL*8(A-H,O-Z)
      REAL*8 C(ND,MD,KDD),P(ND,MD,KDD)
      COMMON RHOZRO,BETAC,CZERO,RKX,RKZ,DELTAX,DELTAZ,RMU,G,TODELX,
     1    TODELZ,DELXSQ,DELZSQ,ALONG,ATRANS,DDT,RKXMU,RKZMU,RKXDXM,
     2    A5,TDX,TDZ,TODDT,ALMAT,TALMAT,A3DDT,A4DDT,
     3    RN,ROBAGZ,ETAP,RMUN,NM1,NM2,MM1,K,KP1
      COMMON/P1/CSEA,IF
      COMMON /P2/ RLENTH,HEIGHT,EPSLON,ACONC,ZZERO,HEAD,UZERO,RECH,EP,
     1    DELTAT,N,M,KD,NTS,NPI
      COMMON /P3/ XL2,ETA,PZROS,PZROF,B1,B2,TB1,TB2,TB1M,TB2M,
     1    B3STAR,B4STAR,B5STAR,EP1M,IRECH
      COMMON /P4/ Q,QDX,QDZ,WX,WZ,DXDZ,DZDX,DXDZG,DZG,IWELL,JWELL
      COMMON /P6/ RKXDD,RKZDD,DELZ12,MM2
      COMMON /P7/ IWP1,IWM1,JWP1,JWM1
      COMMON /P8/ GDZ2,ROBNDT
      COMMON /P9/ RECHED,ROGPHD
      RHOM = RHOZRO*(1.0D0+(BETAC*(C(I,M,KP1)-CZERO)))
      DO 10 J=1,MM2
      RHOJ = RHOZRO*(1.0D0+(BETAC*(C(I,J,KP1)-CZERO)))
      JP2 = J+2
      SUMRHO = 0.0D0
      DO 20 IALPHA=JP2,M
      SUMRHO = SUMRHO + RHOZRO*(1.0D0+(BETAC*(C(I,IALPHA,KP1)-CZERO)))
   20 CONTINUE
      P(I,J,KP1) = GDZ2*(RHOJ + 2.0D0*SUMRHO + RHOM)
      IF(I .EQ. N) P(I,J,KP1) = P(I,J,KP1) + ROGRHD
   10 CONTINUE
      P(1,MM1,KP1) = G*DELTAZ*RHOM
      IF (I .EQ. N) P(I,MM1,KP1) = P(I,MM1,KP1) + ROGRHD
      P(1,M,KP1) = 0.0D0
      IF(I .EQ. N) P(I,M,KP1) = P(I,M,KP1) + ROGRHD
      RETURN
      END

      SUBROUTINE AIPRES(C,P,U0DX,ALPHA,BETA,GAMMA,VECKWN,ND,MD,KDD,J)
      IMPLICIT REAL*8(A-H,O-Z)
      REAL*8 C(ND,MD,KDD),P(ND,MD,KDD),U0DX(MD),ALPHA(ND),BETA(ND),
     1    GAMMA(ND),VECKWN(ND)
      COMMON RHOZRO,BETAC,CZERO,RKX,RKZ,DELTAX,DELTAZ,RMU,G,TODELX,
     1    TODELZ,DELXSQ,DELZSQ,ALONG,ATRANS,DDT,RKXMU,RKZMU,RKXDXM,
     2    A5,TDX,TDZ,TODDT,ALMAT,TALMAT,A3DDT,A4DDT,
     3    RN,ROBAGZ,ETAP,RMUN,NM1,NM2,MM1,K,KP1
      COMMON/P1/CSEA,IF
      COMMON /P2/ RLENTH,HEIGHT,EPSLON,ACONC,ZZERO,HEAD,UZERO,RECH,EP,
     1    DELTAT,N,M,KD,NTS,NPI
      COMMON /P3/ XL2,ETA,PZROS,PZROF,B1,B2,TB1,TB2,TB1M,TB2M
     1    B3STAR,B4STAR,B5STAR,EP1M,IRECH
      COMMON /P4/ Q,QDX,QDZ,WX,WZ,DXDZ,DZDX,DXDZG,DZG,IWELL,JWELL
      COMMON /P8/ GDZ2,ROBNDT
      COMMON /P10/ HPM
      COMMON /P11/ B2GTDZ
      JP1 = J+1
      JM1 = J-1
      IF ( J .EQ. 1) GO TO 20
      IF ( J .EQ. M) GO TO 30
      DO 9 I=2,NM1
      IP1 = I+1
      IM1 = I-1
      IF(I .EQ. N) IP1=N
      RHO = RHOZRO*(1.0D0+(BETAC*(C(I,J,KP1)-CZERO)))
      CWRTX = (C(IP1,J,KP1)-C(IM1,J,KP1))/TODELX
      CWRTZ = (C(I,JP1,KP1)-C(I,JM1,KP1))/TODELZ
      B3 = B3STAR*CWRTX/RHO
      B4 = B4STAR*CWRTZ/RHO
      ALPHA(I) = - (B1-B3)
      BETA(I) = -(TB1M - HPM)
      GAMMA(I) = -(B1+B3)
      VECKWN(I) = B5STAR*CWRTZ+(B4-B2)*P(I,JM1,K)+(TB2-HPM)*P(I,J,K)
     1    -(B4+B2)*P(I,JP1,K)
     .    + ROBNDT*(C(I,J,KP1)-C(I,J,K))
      VECKWN(I) = - VECKWN(I)
 9    CONTINUE
      VECKWN(2) = VECKWN(2)-ALPHA(2)*P(1,J,KP1)
      VECKWN(NM1) = VECKWN(NM1) - GAMMA(NM1)*P(N,J,KP1)
      RETURN
C
C------------ J = 1 ------------
C
 20   CONTINUE
      DO 11 I=2,NM1
      IP1 = I+1
      IM1 = I-1
      RHO = RHOZRO*(1.0D0+(BETAC*(C(I,J,KP1)-CZERO)))
      CWRTX = (C(IP1,J,KP1)-C(IM1,J,KP1))/TODELX
      B3 = B3STAR*CWRTX/RHO
      ALPHA(I) = -(B1-B3)
      BETA(I) = -(TB1M - HPM)
      GAMMA(I) = -(B1+B3)
      VECKWN(I) = RHO*B2GTDZ + TB2*P(I,2,K) - (TB2-HPM)*P(I,1,K)
     .    -ROBNDT*(C(I,J,KP1)-C(I,J,K))
 11   CONTINUE
      VECKWN(2) = VECKWN(2)-ALPHA(2)*P(1,J,KP1)
      VECKWN(NM1) = VECKWN(NM1) - GAMMA(NM1)*P(N,J,KP1)
      RETURN
C
C------------ J = M ------------
C
 30   CONTINUE
      DO 13 I=2,NM1
      IP1 = I+1
      IM1 = I-1
      RHO = RHOZRO*(1.0D0+(BETAC*(C(I,J,KP1)-CZERO)))
      CWRTX = (C(IP1,J,KP1)-C(IM1,J,KP1))/TODELX
      B3 = B3STAR*CWRTX/RHO
      ALPHA(I) = -(B1-B3)
      BETA(I) = -(TB1M - HPM)
      GAMMA(I) = -(B1+B3)
      VECKWN(I) = -RHO*B2GTDZ + TB2*P(I,MM1,K) - (TB2-HPM)*P(I,M,K)
     .     - ROBNDT*(C(I,J,KP1)-C(I,J,K))
 13   CONTINUE
      VECKWN(2) = VECKWN(2)-ALPHA(2)*P(1,J,KP1)
      VECKWN(NM1) = VECKWN(NM1) - GAMMA(NM1)*P(N,J,KP1)
      RETURN
      END

      SUBROUTINE AJPRES(C,P,U0,U0DX,ALPHA,BETA,GAMMA,VECKWN,ND,MD,KDD,I)
      IMPLICIT REAL*8(A-H,O-Z)
      REAL*8 C(ND,MD,KDD),P(ND,MD,KDD),U0(MD),ALPHA(MD),BETA(MD),
     1     GAMMA(MD),VECKWN(MD),U0DX(MD)
      COMMON RHOZRO,BETAC,CZERO,RKX,RKZ,DELTAX,DELTAZ,RMU,G,TODELX,
     1     TODELZ,DELXSQ,DELZSQ,ALONG,ATRANS,DDT,RKXMU,RKZMU,RKXDXM,
     2     A5,TDX,TDZ,TODDT,ALMAT,TALMAT,A3DDT,A4DDT,
     3     RN,ROBAGZ,ETAP,RMUN,NM1,NM2,MM1,K,KP1
      COMMON/P1/CSEA,IF
      COMMON /P2/ RLENTH,HEIGHT,EPSLON,ACONC,ZZERO,HEAD,UZERO,RECH,EP,
     1     DELTAT,N,M,KD,NTS,NPI
      COMMON /P3/ XL2,ETA,PZROS,PZROF,B1,B2,TB1,TB2,TB1M,TB2M,
     1     B3STAR,B4STAR,B5STAR,EP1M,IRECH
      COMMON /P4/ Q,QDX,QDZ,WX,WZ,DXDZ,DZDX,DXDZG,DZG,IWELL,JWELL
      COMMON /P8/ GDZ2,ROBNDT
      COMMON /P10/HPM
      COMMON /P11/ B2GTDZ
      IP1 = I+1
      IM1 = I-1
      DO 7 J=2,MM1
      JP1 = J+1
      JM1 = J-1
      RHO = RHOZRO*(1.0D0+(BETAC*(C(I,J,KP1)-CZERO)))
      CWRTX = (C(IP1,J,KP1)-C(IM1,J,KP1))/TODELX
      CWRTZ = (C(I,JP1,KP1)-C(I,JM1,KP1))/TODELZ
      B3 = B3STAR*CWRTX/RHO
      B4 = B4STAR*CWRTZ/RHO
      ALPHA(J) = -(B2-B4)
      BETA(J) = -(TB2M = HPM)
      GAMMA(J) = -(B2+B4)
      VECKWN(J) = B5STAR*CWRTZ+(B3-B1)*P(IM1,J,K) + (TB1-HPM)*P(I,J,K)
     1     -(B1+B3)*P(IP1,J,K)
     .     + ROBNDT*(C(I,J,KP1)-C(I,J,K))
      VECKWN(J) = - VECKWN(J)
      CONTINUE
      BETA(1) = -(TB2M - HPM)
      GAMMA(1) = TB2M
      RHO = RHOZRO*(1.0D0+(BETAC*(C(I,1,KP1)-CZERO)))
      CWRTX = (C(IP1,1,KP1)-C(IM1,1,KP1))/TODELX
      B3 = B3STAR*CWRTX/RHO
      VECKWN(1) = (-B1+B3)*P(IM1,1,K) + (TB1-HPM)*P(I,1,K)
     .     - (B1+B3)*P(IP1,1,K) - RHO*B2GTDZ
      VECKWN(1) = -VECKWN(1)
      ALPHA(M) = TB2M
      BETA(M) = -(TB2M - HPM)
      RHO = RHOZRO*(1.0D0+(BETAC*(C(I,M,KP1)-CZERO)))
      CWRTX = (C(IP1,M,KP1)-C(IM1,M,KP1))/TODELX
      B3 = B3STAR*CWRTX/RHO
      VECKWN(M) = (-B1+B3)*P(IM1,M,K) + (TB1-HPM)*P(I,M,K)
     .      - (B1+B3)*P(IP1,M,K) + RHO*B2GTDZ
      VECKWN(M) = - VECKWN(M)
      RETURN
      END

      SUBROUTINE APIWEL(C,P,U0DX,ALPHA,BETA,GAMMA,VECKWN,
     1     ND,MD,KDD,J,*)
      IMPLICIT REAL*8(A-H,O-Z)
      REAL*8 C(ND,MD,KDD),P(ND,MD,KDD),U0DX(MD),ALPHA(ND),BETA(ND),
     1     GAMMA(ND),VECKWN(ND)
      COMMON RHOZRO,BETAC,CZERO,RKX,RKZ,DELTAX,DELTAZ,RMU,G,TODELX,
     1     TODELZ,DELXSQ,DELZSQ,ALONG,ATRANS,DDT,RKXMU,RKZMU,RKXDXM,
     2     A5,TDX,TDZ,TODDT,ALMAT,TALMAT,A3DDT,A4DDT,
     3     RN,ROBAGZ,ETAP,RMUN,NM1,NM2,MM1,K,KP1
      COMMON/P1/CSEA,IF
      COMMON /P2/ RLENTH,HEIGHT,EPSLON,ACONC,ZZERO,HEAD,UZERO,RECH,EP,
     1     DELTAT,N,M,KD,NTS,NPI
      COMMON /P3/ XL2,ETA,PZROS,PZROF,B1,B2,TB1,TB2,TB1M,TB2M,
     1     B3STAR,B4STAR,B5STAR,EP1M,IRECH
      COMMON /P4/ Q,QDX,QDZ,WX,WZ,DXDZ,DZDX,DXDZG,DZG,IWELL,JWELL
      COMMON /P6/ RKXDD,RKZDD,DELZ12,MM2
      COMMON /P7/ IWP1,IWM1,JWP1,JWM1
      COMMON /P8/ GDZ2,ROBNDT
      COMMON /P10/ HPM
      JP1 = J+1
      JM1 = J-1
      DO 9 I=2,IWM1
      IP1 = I+1
      IM1 = I-1
      RHO = RHOZRO*(1.0D0+(BETAC*(C(I,J,KP1)-CZERO)))
      CWRTX = (C(IP1,J,KP1)-C(IM1,J,KP1))/TODELX
      CWRTZ = (C(I,JP1,KP1)-C(I,JM1,KP1))/TODELZ
      B3 = B3STAR*CWRTX/RHO
      B4 = B4STAR*CWRTZ/RHO
      ALPHA(I) = -(B1-B3)
      BETA(I) = -(TB1M - HPM)
      GAMMA(I) = -(B1+B3)
      VECKWN(I) = B5STAR*CWRTZ+(B4-B2)*P(I,JM1,K)+(TB2-HPM)*P(I,J,K)
     1     -(B4+B2)*P(I,JP1,K)
     .     + ROBNDT*(C(I,J,KP1)-C(I,J,K))
      VECKWN(I) = - VECKWN(I)
9     CONTINUE
      VECKWN(2) = VECKWN(2) - ALPHA(2)*P(1,J,KP1)
      VECKWN(IWM1) = VECKWN(IWM1) - GAMMA(IWM1)*P(IWELL,JWELL,KP1)
      CALL TRIDGI(2,IWM1,ALPHA,BETA,GAMMA,VECKWN,P,ND,MD,KDD,J)
      DO 10 I=IWP1,NM1
      IP1 = I+1
      IM1 = I-1
      RHO = RHOZRO*(1.0D0+(BETAC*(C(I,J,KP1)-CZERO)))
      CWRTX = (C(IP1,J,KP1)-C(IM1,J,KP1))/TODELX
      CWRTZ = (C(I,JP1,KP1)-C(I,JM1,KP1))/TODELZ
      B3 = B3STAR*CWRTX/RHO
      B4 = B4STAR*CWRTZ/RHO
      ALPHA(I) = -(B1-B3)
      BETA(I) = -(TB1M - HPM)
      GAMMA(I) = -(B1+B3)
      VECKWN(I) = B5STAR*CWRTZ+(B4-B2)*P(I,JM1,K)+(TB2-HPM)*P(I,J,K)
     1    -(B4-B2)*P(I,JP1,K)
     .    + ROBNDT*(C(I,J,KP1)-C(I,J,K))
      VECKWN(I) = - VECKWN(I)
   10 CONTINUE
      VECKWN(IWP1) = VECKWN(IWP1) - ALPHA(IWP1)*P(IWELL,JWELL,KP1)
      VECKWN(NM1) = VECKWN(NM1) - GAMMA(NM1)*P(N,JWELL,KP1)
      CALL TRIDGI(IWP1,NM1,ALPHA,BETA,GAMMA,VECKWN,P,ND,MD,KDD,J)
      RETURN 1
      END

      SUBROUTINE APJWEL(C,P,ALPHA,BETA,GAMMA,VECKWN,ND,MD,KDD,I,*)
      IMPLICIT REAL*8(A-H,O-Z)
      REAL*8 C(ND,MD,KDD),P(ND,MD,KDD),ALPHA(MD),BETA(MD),GAMMA(MD),
     1    VECKWN(MD)
      COMMON RHOZRO,BETAC,CZERO,RKX,RKZ,DELTAX,DELTAZ,RMU,G,TODELX,
     1    TODELZ,DELXSQ,DELZSQ,ALONG,ATRANS,DDT,RKXMU,RKZMU,RKXDXM,
     2    A5,TDX,TDZ,TODDT,ALMAT,TALMAT,A3DDT,A4DDT,
     3    RN,ROBAGZ,ETAP,RMUN,NM1,NM2,MM1,K,KP1
      COMMON/P1/CSEA,IF
      COMMON /P2/ RLENTH,HEIGHT,EPSLON,ACONC,ZZERO,HEAD,UZERO,RECH,EP,
     1    DELTAT,N,M,KD,NTS,NPI
      COMMON /P3/ XL2,ETA,PZROS,PZROF,B1,B2,TB1,TB2,TB1M,TB2M,
     1    B3STAR,B4STAR,B5STAR,EP1M,IRECH
      COMMON /P4/ Q,QDX,QDZ,WX,WZ,DXDZ,DZDX,DXDZG,DZG,IWELL,JWELL
      COMMON /P6/ RKXDD,RKZDD,DELZ12,MM2
      COMMON /P7/ IWP1,IWM1,JWP1,JWM1
      COMMON /P8/ GDZ2,ROBNDT
      COMMON /P10/ HPM
      COMMON /P11/ B2GTDZ
      IP1 = I+1
      IM1 = I-1
      DO 7 J=2,JWM1
      JP1 = J+1
      JM1 = J-1
      RHO = RHOZRO*(1.0D0+(BETAC*(C(I,J,KP1)-CZERO)))
      CWRTX = (C(IP1,J,KP1)-C(IM1,J,KP1))/TODELX
      CWRTZ = (C(I,JP1,KP1)-C(I,JM1,KP1))/TODELZ
      B3 = B3STAR*CWRTX/RHO
      B4 = B4STAR*CWRTZ/RHO
      ALPHA(J) = -(B2-B4)
      BETA(J) = -(TB2M - HPM)
      GAMMA(J) = -(B2+B4)
      VECKWN(J) = B5STAR*CWRTZ + (B3-B1)*P(IM1,J,K)
     .    + (TB1-HPM)*P(I,J,K) - (B1+B3)*P(IP1,J,K)
     .    + ROBNDT*(C(I,J,KP1)-C(I,J,K))
      VECKWN(J) = - VECKWN(J)
[redacted] CONTINUE
      BETA(1) = -(TB2M - HPM)
      GAMMA(1) = -TB2
      RHO = RHOZRO*(1.0D0+(BETAC*(C(I,1,KP1)-CZERO)))
      CWRTX = (C(IP1,1,KP1)-C(IM1,1,KP1))/TODELX
      B3 = B3STAR*CWRTX/RHO
      VECKWN(1) = (-B1+B3)*P(IM1,1,K) + (TB1-HPM)*P(I,1,K)
     .    - (B1+B3)*P(IP1,1,K) - RHO*B2GTDZ
      VECKWN(1) = -VECKWN(1)
      VECKWN(JWM1) = VECKWN(JWM1) - GAMMA(JWM1)*P(I,JWELL,KP1)
      CALL TRIDAG(1.JWM1,ALPHA,BETA,GAMMA,VECKWN,P,ND,MD,KDD,I)
      DO 8 J=JWP1,MM1
      JP1 = J+1
      JM1 = J-1
      RHO = RHOZRO*(1.0D0+(BETAC*(C(I,J,KP1)-CZERO)))
      CWRTX = (C(IP1,J,KP1)-C(IM1,J,KP1))/TODELX
      CWRTZ = (C(I,JP1,KP1)-C(I,JM1,KP1))/TODELZ
      B3 = B3STAR*CWRTX/RHO
      B4 = B4STAR*CWRTZ/RHO
      ALPHA(J) = -(B2-B4)
      BETA(J) = -(TB2M - HPM)
      GAMMA(J) = -(B2+B4)
      VECKWN(J) = B5STAR*CWRTZ + (B3-B1)*P(IM1,J,K)
     .    + (TB1-HPM)*P(I,J,K) - (B1+B3)*P(IP1,J,K)
     .    + ROBNDT*(C(I,J,KP1)-C(I,J,K))
      VECKWN(J) = - VECKWN(J)
 8    CONTINUE
      VECKWN(JWP1) = VECKWN(JWP1) -ALPHA(JWP1)*P(IWELL,JWELL,KP1)
      ALPHA(M) = TB2M
      BETA(M) = -(TB2M - HPM)
      RHO = RHOZRO*(1.0D0+(BETAC*(C(I,M,KP1)-CZERO)))
      CWRTX = (C(IP1,M,KP1)-C(IM1,M,KP1))/TODELX
      B3 = B3STAR*CWRTX/RHO
      VECKWN(M) = (-B1+B3)*P(IM1,M,K) + (TB1-HPM)*P(I,M,K)
     .    - (B1+B3)*P(IP1,M,K) + RHO*B2GTDZ
      VECKWN(M) = - VECKWN(M)
      CALL TRIDAG(JWP1,M,ALPHA,BETA,GAMMA,VECKWN,P,ND,MD,KDD,I)
      RETURN 1
      END

      SUBROUTINE AJPFLX(C,P,U0,U0DX,ALPHA,BETA,GAMMA,VECKWN,ND,MD,KDD,I)
      IMPLICIT REAL*8(A-H,O-Z)
      REAL*8 C(ND,MD,KDD),P(ND,MD,KDD),U0(MD),ALPHA(MD),BETA(MD),
     1    GAMMA(MD),VECKWN(MD),U0DX(MD)
      COMMON RHOZRO,BETAC,CZERO,RKX,RKZ,DELTAX,DELTAZ,RMU,G,TODELX,
     1    TODELZ,DELXSQ,DELZSQ,ALONG,ATRANS,DDT,RKXMU,RKZMU,RKXDXM,
     2    A5,TDX,TDZ,TODDT,ALMAT,TALMAT,A3DDT,A4DDT,
     3    RN,ROBAGZ,ETAP,RMUN,NM1,NM2,MM1,K,KP1
      COMMON/P1/CSEA,IF
      COMMON /P2/ RLENTH,HEIGHT,EPSLON,ACONC,ZZERO,HEAD,UZERO,RECH,EP,
     1    DELTAT,N,M,KD,NTS,NPI
      COMMON /P3/ XL2,ETA,PZROS,PZROF,B1,B2,TB1,TB2,TB1M,TB2M,
     1    B3STAR,B4STAR,B5STAR,EP1M,IRECH
      COMMON /P4/ Q,QDX,QDZ,WX,WZ,DXDZ,DZDX,DXDZG,DZG,IWELL,JWELL
      COMMON /P8/ GDZ2,ROBNDT
      COMMON /P10/ HPM
      COMMON /P11/B2GTDZ
      IP1 = I+1
      IM1 = I-1
      TDX = TODELX
      IF(I .EQ. N) IP1 = N
      IF(I .EQ. N) TDX = DELTAX
      DO 7 J=2,MM1
      JP1 = J+1
      JM1 = J-1
      RHO - RHOZRO*(1.0D0+(BETAC*(C(I,J,KP1)-CZERO)))
      CWRTX = (C(IP1,J,KP1)-C(IM1,J,KP1))/TDX
      CWRTZ = (C(I,JP1,KP1)-C(I,JM1,KP1))/TODELZ
      B3 = B3STAR*CWRTX/RHO
      B4 = B4STAR*CWRTZ/RHO
      ALPHA(J) = -(B2-B4)
      BETA(J) = -(TB2M - HPM)
      GAMMA(J) = -(B2+B4)
      IF(I .EQ. N) VECKWN(J) = B5STAR*CWRTZ - TB1*P(NM1,J,K)
     .    + (TB1-HPM)*P(N,J,K) - (B1+B3)*U0DX(J)
      IF(I .EQ. N) GO TO 6
      VECKWN(J) = B5STAR*CWRTZ+(B3-B1)*P(IM1,J,K) + (TB1-HPM)*P(I,J,K)
     1    -(B1+B3)*P(IP1,J,K)
     .    + ROBNDT*(C(I,J,KP1)-C(I,J,K))
 6    VECKWN(J) = -VECKWN(J)
 7    CONTINUE
      BETA(1) = -(TB2M - HPM)
      GAMMA(1) = TB2M
      RHO = RHOZRO*(1.0D0+(BETAC*(C(I,1,KP1)-CZERO)))
      CWRTX = (C(IP1,1,KP1)-C(IM1,1,KP1))/TDX
      B3 = B3STAR*CWRTX/RHO
      IF(I .EQ. N) VECKWN(1) = -TB1*P(NM1,1,K) + (TB1-HPM)*P(N,1,K)
     .    - (B1+B3)*U0DX(1) - RHO*B2GTDZ
      IF(I .EQ. N) GO TO 8
      VECKWN(1) = (-B1+B3)*P(IM1,1,K) + (TB1-HPM)*P(I,1,K)
     .    - (B1+B3)*P(IP1,1,K) - RHO*B2GTDZ
 8    VECKWN(1) = - VECKWN(1)
      ALPHA(M) = TB2M
      BETA(M) = -(TB2M - HPM)
      RHO = RHOZRO*(1.0D0+(BETAC*(C(I,M,KP1)-CZERO)))
      CWRTX = (C(IP1,M,KP1)-C(IM1,M,KP1))/TDX
      B3 = B3STAR*CWRTX/RHO
      IF(I .EQ. N) VECKWN(M) = -TB1*P(NM1,M,K) + (TB1-HPM)*P(N,M,K)
     .    - (B1+B3)*U0DX(M) + RHO*B2GTDZ
      IF(I .EQ. N) GO TO 9
      VECKWN(M) = (-B1+B3)*P(IM1,M,K) + TB1-HPM)*P(I,M,K)
     .    - (B1+B3)*P(IP1,M,K) + RHO*B2GTDZ
 9    VECKWN(M) = - VECKWN(M)
      RETURN
      END

      SUBROUTINE APIWFX(C,P,U0DX,ALPHA,BETA,GAMMA,VECKWN,
     1    ND,MD,KDD,J,*)
      IMPLICIT REAL*8(A-H,O-Z)
      REAL*8 C(ND,MD,KDD),P(ND,MD,KDD),U0DX(MD),ALPHA(ND),BETA(ND),
     1    GAMMA(ND),VECKWN(ND)
      COMMON RHOZRO,BETAC,CZERO,RKX,RKZ,DELTAX,DELTAZ,RMU,G,TODELX,
     1     TODELZ,DELXSQ,DELZSQ,ALONG,ATRANS,DDT,RKXMU,RKZMU,RKXDXM,
     2     A5,TDX,TDZ,TODDT,ALMAT,TALMAT,A3DDT,A4DDT,
     3     RN,ROBAGZ,ETAP,RMUN,NM1,NM2,MM1,K,KP1
      COMMON/P1/CSEA,IF
      COMMON /P2/ RLENTH,HEIGHT,EPSLON,ACONC,ZZERO,HEAD,UZERO,RECH,EP,
     1     DELTAT,N,M,KD,NTS,NPI
      COMMON /P3/ XL2,ETA,PZROS,PZROF,B1,B2,TB1,TB2,TB1M,TB2M,
     1     B3STAR,B4STAR,B5STAR,EP1M,IRECH
      COMMON /P4/ Q,QDX,QDZ,WX,WZ,DXDZ,DZDX,DXDZG,DZG,IWELL,JWELL
      COMMON /P6/ RKXDD,RKZDD,DELZ12,MM2
      COMMON /P7/ IWP1,IWM1,JWP1,JWM1
      COMMON /P8/ GDZ2,ROBNDT
      COMMON /P10/ HPM
      JP1 = J+1
      IM1 = J-1
      TDX = TODELX
      DO 9 I=2,IWM1
      IP1 = I+1
      IM1 = I-1
      RHO = RHOZRO*(1.0D0+(BETAC*(C(I,J,KP1)-CZERO)))
      CWRTX = (C(IP1,J,KP1)-C(IM1,J,KP1))/TODELX
      CWRTZ = (C(I,JP1,KP1)-C(I,JM1,KP1))/TODELZ
      B3 = B3STAR*CWRTX/RHO
      B4 = B4STAR*CWRTZ/RHO
      ALPHA(I) = -(B1-B3)
      BETA(I) = -(TB1M - HPM)
      GAMMA(I) = -(B1+B3)
      VECKWN(I) = B5STAR*CWRTZ+(B4-B2)*P(I,JM1,K)+(TB2-HPM)*P(I,J,K)
     1     -(B4+B2)*P(I,JP1,*)
     .     + ROBNDT*(C(I,J,KP1)-C(I,J,K))
      VECKWN(I) = - VECKWN(I)
 9    CONTINUE
      VECKWN(2) = VECKWN(2) - ALPHA(2)*P(1,J,KP1)
      VECKWN(IWM1) = VECKWN(IWM1) - GAMMA(IWM1)*P(IWELL,JWELL,KP1)
      CALL TRIDGI(2,IWM1,ALPHA,BETA,GAMMA,VECKWN,P,ND,MD,KDD,J)
      DO 10 I=IWP1,N
      IP1 = I+1
      IM1 = I-1
      IF(I .EQ. N) IP1 = N
      IF(I .EQ. N) TDX = DELTAX
      RHO = RHOZRO*(1.0D0+(BETAC*(C(I,J,KP1)-CZERO)))
      CWRTX = (C(IP1,J,KP1)-C(IM1,J,KP1))/TDX
      CWRTZ = (C(I,JP1,KP1)-C(I,JM1,KP1))/TODELZ
      B3 = B3STAR*CWRTX/RHO
      B4 = B4STAR*CWRTZ/RHO
      ALPHA(I) = -(B1-B3)
      BETA(I) = -(TB1M - HPM)
      GAMMA(I) = -(B1+B3)
      VECKWN(I) = B5STAR*CWRTZ+(B4-B2)*P(I,JM1,K)+(TB2-HPM)*P(I,J,K)
     1     -(B4+B2)*P(I,JP1,K)
     .     + ROBNDT*(C(I,J,KP1)-C(I,J,K))
      VECKWN(I) = - VECKWN(I)
 10   CONTINUE
      VECKWN(IWP1) = VECKWN(IWP1) = ALPHA(IWP1)*P(IWELL,JWELL,KP1)
      ALPHA(N) = TB1M
      VECKWN(N) = VECKWN(N) + (B1+B3)*U0DX(J)
      CALL TRIDGI(IWP1,N,ALPHA,BETA,GAMMA,VECKWN,P,ND,MD,KDD,J)
      RETURN 1
      END

      SUBROUTINE AIPFLX(C,P,U0DX,ALPHA,BETA,GAMMA,VECKWN,ND,MD,KDD,J)
      IMPLICIT REAL*8(A-H,O-Z)
      REAL*8 C(ND,MD,KDD),P(ND,MD,KDD),U0DX(MD),ALPHA(ND),BETA(ND),
     1     GAMMA(ND),VECKWN(ND)
      COMMON RHOZRO,BETAC,CZERO,RKX,RKZ,DELTAX,DELTAZ,RMU,G,TODELX,
     1     TODELZ,DELXSQ,DELZSQ,ALONG,ATRANS,DDT,RKXMU,RKZMU,RKXDXM,
     2     A5,TDX,TDZ,TODDT,ALMAT,TALMAT,A3DDT,A4DDT,
     3     RN,ROBAGZ,ETAP,RMUN,NM1,NM2,MM1,K,KP1
      COMMON/P1/CSEA,IF
      COMMON /P2/ RLENTH,HEIGHT,EPSLON,ACONC,ZZERO,HEAD,UZERO,RECH,EP,
     1     DELTAT,N,M,KD,NTS,NP1
      COMMON /P3/ XL2,ETA,PZROS,PZROF,B1,B2,TB1,TB2,TB1M,TB2M,
     1     B3STAR,B4STAR,B5STAR,EP1M,IRECH
      COMMON /P4/ Q,QDX,QDZ,WX,WZ,DXDZ,DZDX,DXDZG,DZG,IWELL,JWELL
      COMMON /P8/ GDZ2,ROBNDT
      COMMON /P10/ HPM
      COMMON /P11/ B2GTDZ
      JP1 = J+1
      JM1 = J-1
      TDX = TODELX
      IF ( J .EQ. 1) GO TO 20
      IF ( J .EQ. M) GO TO 30
      DO 9 I=2,N
      IP1 = I+1
      IM1 = I-1
      IF(I .EQ. N) IP1=N
      IF(I .EQ. N) TDX = DELTAX
      RHO = RHOZRO*(1.0D0+(BETAC*(C(I,J,KP1)-CZERO)))
      CWRTX = (C(IP1,J,KP1)-C(IM1,J,KP1))/TDX
      CWRTZ = (C(I,JP1,KP1)-C(I,JM1,KP1))/TODELZ
      B3 = B3STAR*CWRTX/RHO
      B4 = B4STAR*CWRTZ/RHO
      ALPHA(I) = - (B1-B3)
      BETA(I) = -(TB1M - HPM)
      GAMMA(I) = -(B1+B3)
      VECKWN(I) = B5STAR*CWRTZ+(B4-B2)*P(I,JM1,K)+(TB2-HPM)*P(I,J,K)
     1     =(B4+B2)*P(I,JP1,K)
     .     + ROBNDT*(C(I,J,KP1)-C(I,J,K)
      VECKWN(I) = - VECKWN(I)
 9    CONTINUE
      VECKWN(2) = VECKWN(2)-ALPHA(2)*P(1,J,KP1)
      ALPHA(N) = TB1M
      VECKWN(N) = VECKWN(N) + (B1+B3)*U0DX(J)
      RETURN
C
C------------ J = 1 ------------
C
 20   CONTINUE
```

```
      DO 11 I=2,N
      IP1 = I+1
      IM1 = I-1
      IF(I .EQ. N) IP1 = N
      IF(I .EQ. N) TDX = DELTAX
      RHO = RHOZRO*(1.0D0+(BETAC*(C(I,J,KP1)-CZERO)))
      CWRTX = (C(IP1,J,KP1)-C(IM1,J,KP1))/TDX
      B3 = B3STAR*CWRTX/RHO
      ALPHA(I) = -(B1-B3)
      BETA(I) = -(TB1M = HPM)
      GAMMA(I) = -(B1+B3)
      VECKWN(I) = RHO*B2GTDZ + TB2*P(I,2,K) - (TB2-HPM)*P(I,1,K)
     .     - ROBNDT*(C(I,J,KP1)-C(I,J,K))
 11   CONTINUE
      VECKWN(2) = VECKWN(2)-ALPHA(2)*P(1,J,KP1)
      ALPHA(N) = TB1M
      VECKWN(N) = VECKWN(N) + (B1+B3)*U0DX(J)
      RETURN
C
C----------- J = M -----------
C
 30   CONTINUE
      DO 13 I=2,N
      IP1 = I+1
      IM1 = I-1
      IF(I .EQ. N) IP1 = N
      IF(I .EQ. N) TDX = DELTAX
      RHO = RHOZRO*(1.0D0+(BETAC*(C(I,J,KP1)-CZERO)))
      CWRTX = (C(IP1,J,KP1)-C(IM1,J,KP1))/TDX
      B3 = B3STAR*CWRTX/RHO
      ALPHA(I) = -(B1-B3)
      BETA(I) = -(TB1M - HPM)
      GAMMA(I) = -(B1+B3)
      VECKWN(I) = -RHO*B2GTDZ + TB2*P(I,MM1,K) - (TB2-HPM)*P(I,M,K)
     .     - ROBNDT*(C(I,J,KP1)-C(I,J,K))
 13   CONTINUE
      VECKWN(2) = VECKWN(2)-ALPHA(2)*P(1,J,KP1)
      ALPHA(N) = TB1M
      VECKWN(N) = VECKWN(N) + (B1+B3)*U0DX(J)
      RETURN
      END

      SUBROUTINE ASMBLI(C,U,V,DXX,DXZ,DZZ,ALPHA,BETA,GAMMA,VECKWN,
     1    N,M,KD,J)
      IMPLICIT REAL*8(A-H,O-Z)
      REAL*8 C(N,M,KD),U(N,M),V(N,M),ALPHA(N),BETA(N),GAMMA(N),VECKWN(N)
      REAL*8 DXX(N,M),DXZ(N,M),DZZ(N,M)
      COMMON RHOZRO,BETAC,CZERO,RKX,RKZ,DELTAX,DELTAZ,RMU,G,TODELX,
     1    TODELZ,DELXSQ,DELZSQ,ALONG,ATRANS,DDT,RKXMU,RKZMU,RKXDXM,
     2    A5,TDX,TDZ,TODDT,ALMAT,TALMAT,A3DDT,A4DDT,
     3    RN,ROBAGZ,ETAP,RMUN,NM1,NM2,MM1,K,KP1
      COMMON/P1/CSEA,IF
      COMMON /P12/ RNDDT
      CSEA = 35.8
      JP1 = J+1
      JM1 = J-1
      DO 100 I=1,NM1
      UD = U(I,J)
      VD = V(I,J)
      CALL ACOEF(DXX,DXZ,DZZ,A1,A2,A3,A4,UD,VD,N,M,I,J)
      ALPHA(I) = -(A1+A3)
      BETA(I) = A5+2.0D0*A3
      GAMMA(I) = A1-A3
      IF(J .EQ. 1) VECKWN(I) = (A5-A4)*C(I,1,K) + A4*C(I,2,K)
      IF (J .EQ. M) VECKWN(I) = 2.0D0*A4*C(I,MM1,K)
     .     + (A5-2.0D0*A4)*C(I,M,K)
      IF ( J .EQ. 1 .OR. J .EQ. M) GO TO 100
      VECKWN(I) = (A2+A4)*C(I,JM1,K1+(A5-2.0D0*A4)*C(I,J,K)
     1     +(A4-A2)*C(I,JP1,K)
100   CONTINUE
      UD = U(1,J)
      IF (UD .GT. 0.0D0) GO TO 10
      VD = V(1,J)
      CALL ACOEF(DXX,DXZ,DZZ,A1,A2,A3,A4,UD,VD,N,M,1,J)
      IF(UD .GT. 0.0D0) GO TO 10
      TOA3 = 2.0D0*A3
      BETA(1) = A5 + TOA3
      GAMMA(1) = -TOA3
      GO TO 20
10    CONTINUE
      IF = 2
      C(1,J,KP1) = CSEA
      VECKWN(2) = VECKWN(2) = ALPHA(2)*CSEA
20    CONTINUE
      VECKWN(NM1) = VECKWN(NM1) -GAMMA(NM1)*C(N,J,KP1)
      RETURN
      END

      SUBROUTINE ASMBLJ(C,U,V,DXX,DXZ,DZZ,ALPHA,BETA,GAMMA,VECKWN,
     1     N,M,KD,I)
      IMPLICIT REAL*8(A-H,O-Z)
      REAL*8 C(N,M,KD),U(N,M),V(N,M),ALPHA(M),BETA(M),
     1     GAMMA(M),VECKWN(M)
      REAL*8 DXX(N,M),DXZ(N,M),DZZ(N,M)
      COMMON RHOZRO,BETAC,CZERO,RKX,RKZ,DELTAX,DELTAZ,RMU,G,TODELX,
     1     TODELZ,DELXSQ,DELZSQ,ALONG,ATRANS,DDT,RKXMU,RKZMU,RKXDXM,
     2     A5,TDX,TDZ,TODDT,ALMAT,TALMAT,A3DDT,A4DDT,
     3     RN,ROBAGZ,ETAP,RMUN,NM1,NM2,MM1,K,KP1
      COMMON/P1/CSEA,IF
      COMMON /P12/RNDDT
      IP1 = I+1
      IM1 = I-1
      IF(I .EQ. 1) GO TO 500
      DO 100 J=2,MM1
      UD = U(I,J)
      VD = V(I,J)
      CALL ACOEF(DXX,DXZ,DZZ,A1,A2,A3,A4,UD,VD,N,M,I,J)
      ALPHA(J) = -(A2+A4)
      BETA(J) = A5 + 2.0D0*A4
      GAMMA(J) = A2 - A4
      VECKWN(J) = (A1+A3)*C(IM1,J,K)+(A5-2.0D0*A3)*C(I,J,K)
     1   +(A3-A1)*C(IP1,J,K)
100   CONTINUE
      UD = U(I,1)
      VD = V(I,1)
      CALL ACOEF(DXX,DXZ,DZZ,A1,A2,A3,A4,UD,VD,N,M,I,1)
      BETA(1) = A5 + 2.0D0*A4
      GAMMA(1) = -2.0D0*A4
      VECKWN(1) =  (A1+A3)*C(IM1,1,K)+(A5-2.0D0*A3)*C(I,1,K)
     1   +(A3-A1)*C(IP1,1,K)
      UD = U(I,M)
      VD = V(I,M)
      CALL ACOEF(DXX,DXZ,DZZ,A1,A2,A3,A4,UD,VD,N,M,I,M)
      ALPHA(M) = -2.0D0*A4
      BETA(M) = 2.0D0*A4 + A5
      VECKWN(M) = (A1+A3)*C(IM1,M,K)+(A5-2.0D0*A3)*C(I,M,K)
     1   +(A3-A1)*C(IP1,M,K)
      RETURN
500   CONTINUE
      DO 200 J=2,MM1
      UD = U(I,J)
      IF(UD .GT. 0.0D0) IF = J+1
      VD = V(I,J)
      CALL ACOEF (DXX,DXZ,DZZ,A1,A2,A3,A4,UD,VD,N,M,I,J)
      ALPHA(J) = -(A2+A4)
      BETA(J) = A5 + 2.0D0*A4
      GAMMA(J) = A2 - A4
      TOA3 = 2.0D0*A3
      VECKWN(J) = (A5-TOA3)*C(1,J,K1 + TOA3*C(2,J,K)
200   CONTINUE
      IF (IF .GT. 1) GO TO 10
      UD = U(I,1)
      VD = V(I,1)
      CALL ACOEF(DXX,DXZ,DZZ,A1,A2,A3,A4,UD,VD,N,M,I,1)
      TOA3 = 2.0D0*A3
      BETA(1) = AT + 2.0D0*A4
      GAMMA(1) = -2.0D0*A4
      VECKWN(1) = (A5-TOA3)*C(1,1,K) + TOA3*C(2,1,K)
      TO TO 20
10    CONTINUE
      VECKWN(IF) = VECKWN(IF) - ALPHA(IF)*CSEA
      IFM1 = IF-1
      DO 30 J=1,IFM1
      C(1,J,KP1) = CSEA
30    CONTINUE
20    CONTINUE
      UD = U(I,M)
      VD = V(I,M)
      CALL ACOEF(DXX,DXZ,DZZ,A1,A2,A3,A4,UD,VD,N,M,I,M)
      TOA3 = 2.0D0*A3
      ALPHA(M) = -2.0D0*A4
      BETA(M) = A5 + 2.0D0*A4
      VECKWN(M) = (A5-TOA3)*C(1,M,K) + TOA3*C(2,M,K)
      IF(U(1,M) .GT. 0.0D0)
     .     VECKWN(M) = (A1+A3)*CSEA+(A5-TOA3)*C(1,M,K)+(A3-A1)*C(2,M,K)
      RETURN
      END

      SUBROUTINE AIWELL(C,U,V,DXX,DXZ,DZZ,ALPHA,BETA,GAMMA,VECKWN,
     1     ND,MD,KDD,J,*)
      IMPLICIT REAL*8(A-H,O-Z)
      REAL*8 C(ND,MD,KDD),U(ND,MD),V(ND,MD),DXX(ND,MD),DXZ(ND,MD),
     1     DZZ(ND,MD),ALPHA(MD),BETA(MD),GAMMA(MD),VECKWN(MD)
      COMMON RHOZRO,BETAC,CZERO,RKX,RKZ,DELTAX,DELTAZ,RMU,G,TODELX,
     1     TODELZ,DELXSQ,DELZSQ,ALONG,ATRANS,DDT,RKXMU,RKZMU,RKXDXM,
     2     A5,TDX,TDZ,TODDT,ALMAT,TALMAT,A3DDT,A4DDT,
     3     RN,ROBAGZ,ETAP,RMUN,NM1,NM2,MM1,K,KP1
      COMMON/P1/CSEA,IF
      COMMON /P2/ RLENTH,HEIGHT,EPSLON,ACONC,ZZERO,HEAD,UZERO,RECH,EP,
     1     DELTAT,N,M,KD,NTS,NPI
      COMMON /P3/ XL2,ETA,PZROS,PZROF,B1,B2,TB1,TB2,TB1M,TB2M,
     1     B3STAR,B4STAR,B5STAR,EP1M,IRECH
      COMMON /P4/ Q,QDX,QDZ,WX,WZ,DXDZ,DZDX,DXDZG,DZG,IWELL,JWELL
      COMMON /P7/ IWP1,IWM1,JWP1,JWM1
      COMMON /P12/ RNDDT
      JP1 = J+1
      JM1 = J-1
      DO 100 I=2,IWM1
      UD = U(I,J)
      VD = V(I,J)
      CALL ACOEF(DXX,DXZ,DZZ,A1,A2,A3,A4,UD,VD,N,M,I,J)
      ALPHA(I) = -(A1+A3)
      BETA(I) = A5+2.0D0*A3
      GAMMA(I) = A1-A3
      VECKWN(I) = (A2+A4)*C(I,JM1,K)+(A5-2.0D0*A4)*C(I,J,K)
     1     +(A4-A2)*C(I,JP1,K)
  100 CONTINUE
      UD = U(1,J)
      IF (UD .GT. 0.0D0) GO TO 10
      VD = V(1,J)
      CALL ACOEF(DXX,DXZ,DZZ,A1,A2,A3,A4,UD,VD,N,M,1,J)
      TOA3 = 2.0D0*A3
      BETA(1) = A5 + TOA3
      GAMMA(1) = -TOA3
      IF = 1
      TO TO 20
   10 CONTINUE
      IF = 2
      C(1,J,KP1) = CSEA
      VECKWN(2) = VECKWN(2) = ALPHA(2)*CSEA
   20 CONTINUE
      VECKWN(IWM1) = VECKWN(IWM1) - GAMMA(IWM1)*C(IWELL,JWELL,KP1)
      CALL TRIDGI(IF,IWM1,ALPHA,BETA,GAMMA,VECKWN,C,ND,MD,KDD,J)
      DO 101 I=IWP1,NM1
      UD = U(I,J)
      VD = V(I,J)
      CALL ACOEF(DXX,DXZ,DZZ,A1,A2,A3,A4,UD,VD,N,M,I,J)
      ALPHA(I) = -(A1+A3)
      BETA(I) = A5+2.0D0*A3
      GAMMA(I) = A1-A3
      VECKWN(I) = (A2+A4)*C(I,JM1,K)+(A5-2.0D0*A4)*C(I,J,K)
     1    +(A4-A2)*C(I,JP1,K)
  101 CONTINUE
      VECKWN(IWP1) = VECKWN(IWP1) - ALPHA(IWP1)*C(IWELL,JWELL,KP1)
      VECKWN(NM1) = VECKWN(NM1) -GAMMA(NM1)*C(N,J,KP1)
      CALL TRIDGI(IWP1,NM1,ALPHA,BETA,GAMMA,VECKWN,C,ND,MD,KDD,J)
      RETURN 1
      END

      SUBROUTINE AJWELL(C,U,V,DXX,DXZ,DZZ,ALPHA,BETA,GAMMA,VECKWN,
     1    ND,MD,KDD,I,*)
      IMPLICIT REAL*8(A-H,O-Z)
      REAL*8 C(ND,MD,KDD),U(ND,MD),V(ND,MD),DXX(ND,MD),DXZ(ND,MD),
     1    DZZ(ND,MD),ALPHA(MD),BETA(MD),GAMMA(MD),VECKWN(MD)
      COMMON RHOZRO,BETAC,CZERO,RKX,RKZ,DELTAX,DELTAZ,RMU,G,TODELX,
     1    TODELZ,DELXSQ,DELZSQ,ALONG,ATRANS,DDT,RKXMU,RKZMU,RKXDXM,
     2    A5,TDX,TDZ,TODDT,ALMAT,TALMAT,A3DDT,A4DDT,
     3    RN,ROBAGZ,ETAP,RMUN,NM1,NM2,MM1,K,KP1
      COMMON/P1/CSEA,IF
      COMMON /P2/ RLENTH,HEIGHT,EPSLON,ACONC,ZZERO,HEAD,UZERO,RECH,EP,
     1    DELTAT,N,M,KD,NTS,NPI
      COMMON /P3/ XL2,ETA,PZROS,PZROF,B1,B2,TB1,TB2,TB1M,TB2M,
     1    B3STAR,B4STAR,B5STAR,EP1M,IRECH
      COMMON /P4/ Q,QDX,QDZ,WX,WZ,DXDZ,DZDX,DXDZG,DZG,IWELL,JWELL
      COMMON /P7/ IWP1,IWM1,JWP1,JWM1
      IP1 = I+1
      IM1 = I-1
      DO 100 J=2,JWM1
      UD = U(I,J)
      VD = V(I,J)
      CALL ACOEF(DXX,DXZ,DZZ,A1,A2,A3,A4,UD,VD,N,M,I,J)
      ALPHA(J) = -(A2+A4)
      BETA(J) = A5 + 2.0D0*A4
      GAMMA(J) = A2 - A4
      VECKWN(J) = (A1+A3)*C(IM1,J,K)+(A5-2.0D0*A3)*C(I,J,K)
     1  +(A3-A1)*C(IP1,J,K)
  100 CONTINUE
      UD = U(I,1)
      VD = V(I,1)
      CALL ACOEF(DXX,DXZ,DZZ,A1,A2,A3,A4,UD,VD,N,M,I,1)
      BETA(1) = A5 + 2.0D0*A4
      GAMMA(1) = -2.0D0*A4
      VECKWN(1) = (A1+A3)*C(IM1,1,K)+(A5-2.0D0*A3)*C(I,1,K)
     1  +(A3-A1)*C(IP1,1,K)
      VECKWN(JWM1) = VECKWN(JWM1) - GAMMA(WM1)*C(IWELL,JWELL,KP1)
      CALL TRIDAG(1,JWM1,ALPHA,BETA,GAMMA,VECKWN,C,ND,MD,K-D,I)
      DO 101 J=JWP1,MM1
      UD = U(I,J)
      VD = V(I,J)
      CALL ACOEF(DXX,DXZ,DZZ,A1,A2,A3,A4,UD,VD,N,M,I,J)
      ALPHA(J) = -(A1+A4)
      BETA(J) = A5 + 2.0D0*A4
      GAMMA(J) = A2 - A4
      VECKWN(J) = (A1+A3)*C(IM1,J,K)+(A5-2.0D0*A3)*C(I,J,K)
     1   +(A3-A1)*C(IP1,J,K)
 101  CONTINUE
      VECKWN(JWP1) = VECKWN(JWP1) - ALPHA(JWP1)*C(IWELL,JWELL,KP1)
      UD = U(I,M)
      VD = V(I,M)
      CALL ACOEF(DXX,DXZ,DZZ,A1,A2,A3,A4,UD,VD,N,M,I,M)
      ALPHA(M) = -2.0D0*A4
      BETA(M) = 2.0D0*A4 + A5
      VECKWN(M) = (A1+A3)*C(IM1,M,K)+(A5-2.0D0*A3)*C(I,M,K)
     1    +(A3-A1)*C(IP1,M,K)
      CALL TRIDAG(JWP1,M,ALPHA,BETA,GAMMA,VECKWN,C,ND,MD,KDD,I)
      RETURN 1
      END

      SUBROUTINE TRACAI(C,U,V,DXX,DXZ,DZZ,ALPHA,BETA,GAMMA,VECKWN,
     1    N,M,KD,J)
      IMPLICIT REAL*8(A-H,O-Z)
      REAL*8 C(N,M,KD),U(N,M),V(N,M),ALPHA(N),BETA(N),GAMMA(N),VECKWN(N)
      REAL*8 DXX(N,M),DXZ(N,M),DZZ(N,M)
      COMMON RHOZRO,BETAC,CZERO,RKX,RKZ,DELTAX,DELTAZ,RMU,G,TODELX,
     1    TODELZ,DELXSQ,DELZSQ,ALONG,ATRANS,DDT,RKXMU,RKZMU,RKXDXM,
     2    A5,TDX,TDZ,TODDT,ALMAT,TALMAT,A3DDT,A4DDT,
     3    RN,ROBAGZ,ETAP,RMUN,NM1,NM2,MM1,K,KP1
      COMMON/P1/CSEA,IF
      COMMON /P12/ RNDDT
      JP1 = J+1
      JM1 = J-1
      DO 100 I=1,NM1
      UD = U(I,J)
      VD = V(I,J)
      CALL ACOEF(DXX,DXZ,DZZ,A1,A2,A3,A4,UD,VD,N,M,I,J)
      ALPHA(I) = -(A1+A3)
      BETA(I) = A5+2.0D0*A3
      GAMMA(I) = A2-A3
      IF(J .EQ. 1) VECKWN(I) = (A5=A4)*C(I,1,K) + A4*C(I,2,K)
      IF (J .EQ. M) VECKWN(I) = 2.0D0*A4*C(I,MM1,K)
     .    + (A5-2.0D0*A4)*C(I,M,K)
      IF ( J .EQ. 1 .OR. J .EQ. M) GO TO 100
      VECKWN(I) = (A1+A4)*C(I,JM1,K) + (A5-2.0D0*A4)*C(I,J,K)
     1    +(A4-A2)*C(I,JP1,K)
 100  CONTINUE
      UD = U(1,J)
      IF (UD .GT. 0.0D0) GO TO 10
      VD = V(1,J)
      CALL ACOEF(DXX,DXZ,DZZ,A1,A2,A3,A4,UD,VD,N,M,1,J)
      TOA3 = 2.0D0*A3
      BETA(1) = A5 + TOA3
      GAMMA(1) = -TOA3
      GO TO 20
 10   CONTINUE
```

```
      IF = 2
      C(1,J,KP1) = 0.0D0
 20   CONTINUE
      VECKWN(NM1) = VECKWN(NM1) -GAMMA(NM1)*C(N,J,KP1)
      RETURN
      END

      SUBROUTINE TRACAJ(C,U,V,DXX,DXZ,DZZ,ALPHA,BETA,GAMMA,VECKWN,
     1    N,M,KD,I)
      IMPLICIT REAL*8(A-H,O-Z)
      REAL*8 C(N,M,KD),U(N,M),V(N,M),ALPHA(M),BETA(M),
     1    GAMMA(M),VECKWN(M)
      REAL*8 DXX(N,M),DXZ(N,M),DZZ(N,M)
      COMMON RHOZRO,BETAC,CZERO,RKX,RKZ,DELTAX,DELTAZ,RMU,G,TODELX,
     1    TODELZ,DELXSQ,DELZSQ,ALONG,ATRANS,DDT,RKXMU,RKZMU,RKXDXM,
     2    A5,TDX,TDZ,TODDT,ALMAT,TALMAT,A3DDT,A4DDT,
     3    RN,ROBAGZ,ETAP,RMUN,NM1,NM2,MM1,K,KP1
      COMMON/P1/CSEA,IF
      COMMON /P12/ RNDDT
      IP1 = I+1
      IM1 = I-1
      IF(I .EQ. 1) GO TO 500
      DO 100 J=2,MM1
      UD = U(I,J)
      VD = V(I,J)
      CALL ACOEF(DXX,DXZ,DZZ,A1,A2,A3,A4,UD,VD,N,M,I,J)
      ALPHA(J) = -(A2+A4)
      BETA(J) = A5 + 2.0D0*A4
      GAMMA(J) = A2 - A4
      VECKWN(J) = (A1+A3)*C(IM1,J,K)+(A5-2.0D0*A3)*C(I,J,K)
     1    +(A3-A1)*C(IP1,J,K)
 100  CONTINUE
      UD = U(I,1)
      VD = V(I,1)
      CALL ACOEF(DXX,DXZ,DZZ,A1,A2,A3,A4,UD,VD,N,M,I,1)
      BETA(1) = A5 + 2.0D0*A4
      GAMMA(1) = -2.0D0*A4
      VECKWN(1) = (A1+A3)*C(IM1,1,K)+(A5-2.0D0*A3)*C(I,1,K)
     1   +(A3-A1)*C(IP1,1,K)
      UD = U(I,M)
      VD = V(I,M)
      CALL ACOEF(DXX,DXZ,DZZ,A1,A2,A3,A4,UD,VD,N,M,I,M)
      ALPHA(M) = -2.0D0*A4
      BETA(M) = 2.0D0*A4 + A5
      VECKWN(M) = (A1+A3)*C(IM1,M,K)+(A5-2.0D0*A3)*C(I,M,K)
     1   +(A3-A1)*C(IP1,M,K)
      RETURN
 500  CONTINUE
      DO 200 J=2,MM1
      UD = U(I,J)
      IF(UD .GT. 0.0D0) IF = J+1
      VD = V(I,J)
      CALL ACOEF(DXX,DXZ,DZZ,A1,A2,A3,A4,UD,VD,N,M,I,J)
      ALPHA(J) = -(A2+A4)
      BETA(J) = A5 + 2.0D0*A4
      GAMMA(J) = A2 - A4
      TOA3 = 2.0D0*A3
      VECKWN(J) = (A5-TOA3)*C(1,J,K) + TOA3*C(2,J,K)
  200 CONTINUE
      IF (IF .GT. 1) GO TO 10
      UD = U(I,1)
      VD = V(I,1)
      CALL ACOEF(DXX,DXZ,DZZ,A1,A2,A3,A4,UD,VD,N,M,I,1)
      TOA3 = 2.0D0*A3
      BETA(1) = A5 + 2.0D0*A4
      GAMMA(1) = -2.0D0*A4
      VECKWN(1) = (A5-TOA3)*C(1,1,K) + TOA3*C(2,1,K)
      GO TO 20
   10 CONTINUE
      IFM1 = IF-1
      DO 30 J=1,IFM1
      C(1,J,KP1) = 0.0D0
   30 CONTINUE
   20 CONTINUE
      UD = U(I,M)
      VD = V(I,M)
      CALL ACOEF(DXX,DXZ,DZZ,A1,A2,A3,A4,UD,VD,N,M,I,M)
      TOA3 = 2.0D0*A3
      ALPHA(M) = -2.0D0*A4
      BETA(M) = A5 + 2.0D0*A4
      VECKWN(M) = (A5-TOA3)*C(1,M,K) + TOA3*C(2,M,K)
      RETURN
      END

      SUBROUTINE TAIWEL(C,U,V,DXX,DXZ,DZZ,ALPHA,BETA,GAMMA,VECKWN,
     1    ND,MD,KDD,J,*)
      IMPLICIT REAL*8(A-H,O-Z)
      REAL*8 C(ND,MD,KDD),U(ND,MD),V(ND,MD),DXX(ND,MD),DXZ(ND,MD),
     1    DZZ(ND,MD),ALPHA(MD),BETA(MD),GAMMA(MD),VECKWN(MD)
      COMMON RHOZRO,BETAC,CZERO,RKX,RKZ,DELTAX,DELTAZ,RMU,G,TODELX,
     1    TODELZ,DELXSQ,DELZSQ,ALONG,ATRANS,DDT,RKXMU,RKZMU,RKXDXM,
     2    A5,TDX,TDZ,TODDT,ALMAT,TALMAT,A3DDT,A4DDT,
     3    RN,ROBAGZ,ETAP,RMUN,NM1,NM2,MM1,K,KP1
      COMMON/P1/CSEA,IF
      COMMON /P2/ RLENTH,HEIGHT,EPSLON,ACONC,ZZERO,HEAD,UZERO,RECH,EP,
     1    DELTAT,N,M,KD,NTS,NPI
      COMMON /P3/ XL2,ETA,PZROS,PZROF,B1,B2,TB1,TB2,TB1M,TB2M,
     1    B3STAR,B4STAR,B5STAR,EP1M,IRECH
      COMMON /P4/ Q,QDX,QDZ,WX,WZ,DXDZ,DZDX,DXDZG,DZG,IWELL,JWELL
      COMMON /P7/ IWP1,IWM1,JWP1,JWM1
      COMMON /P12/ RNDDT
      JP1 = J+1
      JM1 = J-1
      DO 100 I=2,IWM1
      UD = U(I,J)
      VD = V(I,J)
      CALL ACOEF(DXX,DXZ,DZZ,A1,A2,A3,A4,UD,VD,N,M,I,J)
      ALPHA(I) = -(A1+A3)
      BETA(I) = A5+2.0D0*A3
      GAMMA(I) = A1-A3
      VECKWN(I) = (A2+A4)*C(K,JM1,K)+(A5-2.0D0*A4)*C(I,J,K)
     1     +(A4-A2)*C(I,JP1,K)
100   CONTINUE
      UD = U(1,J)
      IF (UD .GT. 0.0D0) GO TO 10
      VD = V(1,J)
      CALL  ACOEF(DXX,DXZ,DZZ,A1,A2,A3,A4,UD,VD,N,M,1,J)
      TOA3 = 2.0D0*A3
      BETA(1) = A5 + TOA3
      GAMMA(1) = -TOA3
      IF = 1
      GO TO 20
10    CONTINUE
      IF = 2
      C(1,J,KP1) = 0.0D0
20    CONTINUE
      VECKWN(IWM1) = VECKWN(IWM1) - GAMMA(IWM1)*C(IWELL,JWELL,KP1)
      CALL TRIDGI(IF,IWM1,ALPHA,BETA,GAMMA,VECKWN,C,ND,MD,KDD,J)
      DO 101 I=IWP1,NM1
      UD = U(I,J)
      VD = V(I,J)
      CALL ACOEF(DXX,DXZ,DZZ,A1,A2,A3,A4,UD,VD,N,M,I,J)
      ALPHA(I) = -(A1+A3)
      BETA(I) = A5+2.0D0*A3
      GAMMA(I) = A1-A3
      VECKWN(I) = (A2+A4)*C(I,JM1,K)+(A5-2.0D0*A4)*C(I,J,K)
     1     +(A4-A2)*C(I,JP1,K)
101   CONTINUE
      VECKWN(IWP1) = VECKWN(IWP1) - ALPHA(IWP1)*C(IWELL,JWELL,KP1)
      VECKWN(NM1) = VECKWN(NM1) -GAMMA(NM1)*C(N,J,KP1)
      CALL TRIDGI(IWP1,NM1,ALPHA,BETA,GAMMA,VECKWN,C,ND,MD,KDD,J)
      RETURN 1
      END

      SUBROUTINE TAJWEL(C,U,V,DXX,DXZ,DZZ,ALPHA,BETA,GAMMA,VECKWN,
     1     ND,MD,KDD,I,*)
      IMPLICIT REAL*8(A-H,O-Z)
      REAL*8 C(ND,MD,KDD),U(ND,MD),V(ND,MD),DXX(ND,MD),DXZ(ND,MD),
     1     DZZ(ND.MD).ALPHA(MD).BETA(MD).GAMMA(MD).VECKWN(MD)
      COMMON RHOZRO.BETAC.CZERO.RKX.RKZ.DELTAX.DELTAZ.RMU.G.TODELX.
     1     TODELZ.DELXSO.DFLZSO.ALONG.ATRANS.DDT.RKXMU.RKZMU.RKXDXM.
     2     A5.TDX.TDZ.TODDT.ALMAT.TALMAT.A3DDT.A4DDT.
     3     RN,ROBAGZ,ETAP,RMUN,NM1,NM2,MM1,K,KP1
      COMMON/P1/CSEA,IF
      COMMON /P2/ RLENTH,HEIGHT,EPSLON,ACONC,ZZERO,HEAD,UZERO,RECH,EP,
     1     DELTAT,N,M,KD,NTS,NPI
      COMMON /P3/ XL2,ETA,PZROS,PZROF,B1,B2,TB1,TB2,TB1M,TB2M,
     1     B3STAR,B4STAR,B5STAR,EP1M,IRECH
      COMMON /P4/ Q,QDX,QDZ,WX,WZ,DXDZ,DZDX,DXDZG,DZG,IWELL,JWELL
      COMMON /P7/ IWP1,IWM1,JWP1,JWM1
      IP1 = I+1
      IM1 = I-1
      DO 100 J=2,JWM1
      UD = U(I,J)
      VD = V(I,J)
      CALL ACOEF(DXX,DXZ,DZZ,A1,A2,A3,A4,UD,VD,N,M,I,J)
      ALPHA(J) = -(A2+A4)
      BETA(J) = A5 + 2.0D0*A4
      GAMMA(J) = A2 - A4
      VECKWN(J) = (A1+A3)*C(IM1,J,K)+(A5-2.0D0*A3)*C(I,J,K)
     1   +(A3-A1)*C(IP1,J,K)
 100  CONTINUE
      UD = U(I,1)
      VD = V(I,1)
      CALL ACOEF(DXX,DXZ,DZZ,A1,A2,A3,A4,UD,VD,N,M,I,1)
      BETA(1) = A5 + 2.0D0*A4
      GAMMA(1) = -2.0D0*A4
      VECKWN(1) = (A1+A3)*C(IM1,1,K)+(A5-2.0D0*A3)*C(I,1,K)
     1   +(A3-A1)*C(IP1,1,K)
      VECKWN(JWM1) = VECKWN(JWM1) - GAMMA(JWM1)*C(IWELL,JWELL,KP1)
      CALL TRIDAG(1.JWM1,ALPHA,BETA,GAMMA,VECKWN,C,ND,MD,KDD,I)
      DO 101 J=JWP1,MM1
      UD = U(I,J)
      VD = V(I,J)
      CALL ACOEF(DXX,DXZ,DZZ,A1,A2,A3,A4,UD,VD,N,M,I,J)
      ALPHA(J) = -(A2+A4)
      BETA(J) = A5 + 2.0D0*A4
      GAMMA(J) = A2 + A4
      VECKWN(J) = (A1+A3)*C(IM1,J,K)+(A5-2.0D0*A3)*C(I,J,K)
     1   +(A3-A1)*C(IP1,J,K)
 101  CONTINUE
      VECKWN(JWP1) = VECKWN(JWP1) - ALPHA(JWP1)*C(IWELL,JWELL,KP1)
      UD - U(I,M)
      VD = V(I,M)
      CALL ACOEF(DXX,DXZ,DZZ,A1,A2,A3,A4,UD,VD,N,M,I,M)
      ALPHA(M) = -2.0D0*A4
      BETA(M) = 2.0D0*A4 + A5
      VECKWN(M) = (A1+A3)*C(IM1,M,K)+(A5-2,0D0*A3)*C(I,M,K)
     1   +(A3-A1)*C(IP1,M,K)
      CALL TRIDAG(JWP1,M,ALPHA,BETA,GAMMA,VECKWN,C,ND,MD,KDD,I)
      RETURN 1
      END

      SUBROUTINE TRACER (C,U,V,DXX,DXZ,DZZ,ALPHA,BETA,GAMMA,VECKWN,
     .    ND,MD,KDD)
      IMPLICIT REAL*8 (A-H,O-Z)
      REAL*8 C(ND,MD,KDD),U(ND,MD),V(ND,MD),DXX(ND,MD),DXZ(ND,MD),
     .    DZZ(ND,MD),ALPHA(MD),BETA(MD),GAMMA(MD),VECKWN(MD)
      COMMON RHOZRO,BETAC,CZERO,RKX,RKZ,DELTAX,DELTAZ,RMU,G,TODELX,
     1    TODELZ,DELXSQ,DELZSQ,ALONG,ATRANS,DDT,RKXMU,RKZMU,RKXDXM,
     2    A5,TDX,TDZ,TODDT,ALMAT,TALMAT,A3DDT,A4DDT,
     3    RN,ROBAGZ,ETAP,RMUN,NM1,NM2,MM1,K,KP1
      COMMON/P1/CSEA.IF
      COMMON /P2/ RLENTH,HEIGHT,EPSLON,ACONC,ZZERO,HEAD,UZERO,RECH,EP,
     1    DELTAT,N,M,KD,NTS,NPI
      COMMON /P3/ XL2,ETA,PZROS,PZROF,B1,B2,TB1,TB2,TB1M,TB2M,
     1    B3STAR,B4STAR,B5STAR,EP1M,IRECH
      COMMON /P4/ Q,QDX,QDZ,WX,WZ,DXDZ,DZDX,DXDZG,DZG,IWELL,JWELL
      COMMON /P6/ RKXDD,RKZDD,DELZ12,MM2
      COMMON /P7/ IWP1,IWM1,JWP1,JWM1
      COMMON /P8/ GDZ2,ROBNDT
      COMMON /P9/ RECHED,ROGRHD
      COMMON /P10/ HPM
      COMMON /P11/ B2GTDZ
      COMMON /P12/ RNDDT
      COMMON /P13/ ITIME
      COMMON /P14/ EFLUNT
      IF (ITIME .EQ. 1) GO TO 1000
 1001 CONTINUE
C
C------------START ITERATION ON COLUMNS (I= COLUMNS)------------
C
      L = NM1
      DO 5 J=1,M
      IF = 1
      IF(J .EQ. JWELL)CALL TAIWEL(C,U,V,DXX,DXZ,DZZ,ALPHA,BETA,
     1    GAMMA,VECKWN,CW,ND,MD,KDD,J,&5)
      CALL TRACAI(C,U,V,DXX,DXZ,DZZ,ALPHA,BETA,GAMMA,VECKWN,N,M,KD,J)
      CALL TRIDGI(IF,L,ALPHA,BETA,GAMMA,VECKWN,C,N,M,KD,J)
 5    CONTINUE
      CALL STUF21(C,ND,MD,KDD)
C
C------------START  ITERATION  ON ROWS (J=ROWS)------------
C
      L = M
      DO 4 I=1,NM1
      IF = 1
      IF(I .EQ. IWELL) CALL TAJWEL(C,U,V,DXX,DXZ,DZZ,ALPHA,BETA,
     1    GAMMA,VECKWN,CW,ND,MD,KDD,I,&4)
      CALL TRACAJ(C,U,V,DXX,DXZ,DZZ,ALPHA,BETA,GAMMA,VECKWN,N,M,KD,I)
      CALL TRIDAG(IF,L,ALPHA,BETA,GAMMA,VECKWN,C,N,M,KD,I)
 4    CONTINUE
      RETURN
 1000 CONTINUE
      DO 300 J=1,M
      DO 300 I=1,N
      C(I,J,K) = 0.0D0
      C(I,J,KP1) = 0.0D0
 300  CONTINUE
      C(IWELL,JWELL,KP1) = EFLUNT
      GO TO 1001
      END

      SUBROUTINE DIJ(DXX,DXZ,DZZ,U,V,N,M)
      IMPLICIT REAL*8(A-H,O-Z)
      REAL*8 DXX(N,M),DXZ(N,M),DZZ(N,M),U(N,M),V(N,M)
      COMMON RHOZRO,BETAC,CZERO,RKX,RKZ,DELTAX,DELTAZ,RMU,G,TODELX,
     1    TODELZ,DELXSQ,DELZSQ,ALONG,ATRANS,DDT,RKXMU,RKZMU,RKXDXM,
     2    A5,TDX,TDZ,TODDT,ALMAT,TALMAT,A3DDT,A4DDT,
     3    RN,ROBAGZ,ETAP,RMUN,NM1,NM2,MM1,K,KP1
      DO 1 J=1,M
      DO 1 I=1,N
      UD = U(I,J)
      VD = V(I,J)
      VMAGD = DSQRT(UD*UD+VD*VD)
      IF(VMAGD .LT. 1.0D-05) DXX(I,J) = DDT
      IF(VMAGD .LT. 1.0D-05) DXZ(I,J) = DDT
      IF(VMAGD .LT. 1.0D-05) DZZ(I,J) = DDT
      IF (VMAGD .LT. 1.0D-05) GO TO 1
      DXX(I,J) = ATRANS*VMAGD + (ALMAT*UD*UD)/VMAGD + DDT
      DXZ(I,J) = (ALMAT*UD*VD/VMAGD) + DDT
      DZZ(I,J) = ATRANS*VMAGD + (ALMAT*VD*VD)/VMAGD + DDT
1     CONTINUE
      RETURN
      END

      SUBROUTINE ACOEF(DXX,DXZ,DZZ,A1,A2,A3,A4,UD,VD,N,M,I,J)
      IMPLICIT REAL*8(A-H,O-Z)
      REAL*8 DXX(N,M),DXZ(N,M),DZZ(N,M)
      COMMON RHOZRO,BETAC,CZERO,RKX,RKZ,DELTAX,DELTAZ,RMU,G,TODELX,
     1    TODELZ,DELXSQ,DELZSQ,ALONG,ATRANS,DDT,RKXMU,RKZMU,RKXDXM,
     2    A5,TDX,TDZ,TODDT,ALMAT,TALMAT,A3DDT,A4DDT,
     3    RN,ROBAGZ,ETAP,RMUN,NM1,NM2,MM1,K,KP1
      IP1 = I+1
      IM1 = I-1
      JP1 = J+1
      JM1 = J-1
      TDZ = TODELZ
      IF (J .EQ. 1) JM1 = J
      IF (J .EQ. M) JP1 = J
      IF (J .EQ. 1 .OR. J .EQ. M) TDZ = DELTAZ
      DXXWRX = (DXX(IP1,J)-DXX(IM1,J))/TODELX
      DXZWRX = (DXZ(IP1,J)-DXZ(IM1,J))/TODELX
      DXZWRZ = (DXZ(I,JP1)-DXZ(I,JM1))/TDZ
      DZZWRZ = (DZZ(I,JP1)-DZZ(I,JM1))/TDZ
      A1 = (UD-DXXWRX-DXZWRZ)/TODELX
      A2 = (VD-DXZWRX-DZZWRZ)/TODELZ
      A3 = (DXX(I,J)+DXZ(I,J))/DELXSQ
      A4 = (DXZ(I,J)+DZZ(I,J))/DELZSQ
      RETURN
      END

      SUBROUTINE DISP(AB1,AB2,UD,VD,VMAGD)
      IMPLICIT REAL*8(A-H,O-Z)
      COMMON RHOZRO,BETAC,CZERO,RKX,RKZ,DELTAX,DELTAZ,RMU,G,TODELX,
     1    TODELZ,DELXSQ,DELZSQ,ALONG,ATRANS,DDT,RKXMU,RKZMU,RKXDXM,
     2    A5,TDX,TDZ,TODDT,ALMAT,TALMAT,A3DDT,A4DDT,
     3    RN,ROBAGZ,ETAP,RMUN,NM1,NM2,MM1,K,KP1
      DXX = ATRANS*VMAGD+(ALMAT*UD*DABS(UD))/VMAGD + DDT
      AB1 = RN*DXX/DELTAX
      DXZ = ALMAT*UD*VD/VMAGD + DDT
      AB2 = RN*DXZ/TODELZ
      RETURN
      END
      SUBROUTINE TRIDAG(IF,L,A,B,C,D,V,N,M,KD,I)
      IMPLICIT REAL*8(A-H,O-Z)
      REAL*8 A(L),B(L),C(L),D(L),V(N,M,KD),BETA(101),GAMMA(101)
      COMMON RHOZRO,BETAC,CZERO,RKX,RKZ,DELTAX,DELTAZ,RMU,G,TODELX,
     1    TODELZ,DELXSQ,DELZSQ,ALONG,ATRANS,DDT,RKXMU,RKZMU,RKXDXM,
     2    A5,TDX,TDZ,TODDT,ALMAT,TALMAT,A3DDT,A4DDT,
     3    RN,ROBAGZ,ETAP,RMUN,NM1,NM2,MM1,K,KP1
      BETA(IF) = B(IF)
      GAMMA(IF) = D(IF)/BETA(IF)
      IFP1 = IF + 1
      DO 1 ID=IFP1,L
      BETA(ID) = B(ID)-A(ID)*C(ID-1)/BETA(ID-1)
1     GAMMA(ID) = (D(ID)-A(ID)*GAMMA(ID-1))/BETA(ID)
      V(I,L,KP1) = GAMMA(L)
      LAST = L-IF
      DO 2 J=1,LAST
      JD = L-J
2     V(I,JD,KP1) = GAMMA(JD)-C(JD)*V(I,(JD+1),KP1)/BETA(JD)
      DO 3 JP=IF,L
3     WRITE(6,900) JP,BETA(JP),GAMMA(JP)
900   FORMAT(1H ,I5,5X,2(1PD20.10
      RETURN
      END

      SUBROUTINE TRIDGI(IF,L,A,B,C,D,V,N,M,KD,I)
      IMPLICIT REAL*8(A-H,O-Z)
      REAL*8 A(L),B(L),C(L),D(L),V(N,M,KD),BETA(101),GAMMA(101)
      COMMON RHOZRO,BETAC,CZERO,RKX,RKZ,DELTAX,DELTAZ,RMU,G,TODELX,
     1    TODELZ,DELXSQ,DELZSQ,ALONG,ATRANS,DDT,RKXMU,RKZMU,RKXDXM,
     2    A5,TDX,TDZ,TODDT,ALMAT,TALMAT,A3DDT,A4DDT,
     3    RN,ROBAGZ,ETAP,RMUN,NM1,NM2,MM1,K,KP1
      BETA(IF) = B(IF)
      GAMMA(IF) = D(IF)/BETA(IF)
      IFP1 = IF + 1
      DO 1 ID=IFP1,L
      BETA(ID) = B(ID)-A(ID)*C(ID-1)/BETA(ID-1)
1     GAMMA(ID) = (D(ID)-A(ID)*GAMMA(ID-1))/BETA(ID)
      V(L,I,KP1) = GAMMA(L)
      LAST = L-IF
      DO 2 J=1,LAST
      JD = L-J
2     V(JD,I,KP1) = GAMMA(JD)-C(JD)*V((JD+1),I,KP1)/BETA(JD)
      RETURN
      END

      SUBROUTINE STUF21(W,ND,MD,KDD)
      IMPLICIT REAL*8 (A-H,O-Z)
      REAL*8 W(ND,MD,KDD)
      COMMON /P2/ RLENTH,HEIGHT,EPSLON,ACONC,ZZERO,HEAD,UZERO,RECH,EP,
     1    DELTAT,N,M,KD,NTS,NPI
      COMMON /P3/ XL2,ETA,PZROS,PZROF,B1,B2,TB1,TB2,TB1M,TB2M,
     1    B3STAR,B4STAR,B5STAR,EP1M,IRECH
      DO 10 J=1,MD
      DO 10 I=1,ND
      W(I,J,1) = W(I,J,2)
 10   CONTINUE
      RETURN
      END

      SUBROUTINE CHKCON(W,EPLN,ICON,ND,MD,KDD)
      IMPLICIT REAL*8 (A-H,O-Z)
      REAL*8 W(ND,MD,KDD)
      ICON = 0
      DIF = 0.0D0
      DO 10 J=1,MD
      DO 10 I=1,ND
      DIFNEW = DABS(W(I,J,2) = W(I,J,1))
      IF (DIFNEW .GT. DIF) DIF=DIFNEW
 10   CONTINUE
      WRITE(6,900) DIF
 900  FORMAT(1H ,'GREATEST DIFFERENCE IS',1PD20.10)
      IF(DIF .LT. EPLN) ICON=1
      RETURN
      END

      SUBROUTINE COMHPM(RHOP,NIPARM)
      IMPLICIT REAL*8 (A-H,O-Z)
      REAL*8 RHOP(NIPARM)
      COMMON RHOZRO,BETAC,CZERO,RKX,RKZ,DELTAX,DELTAZ,RMU,G,TODELX,
     1     TODELZ,DELXSQ,DELZSQ,ALONG,ATRANS,DDT,RKXMU,RKZMU,RKXDXM,
     2     A5,TDX,TDZ,TODDT,ALMAT,TALMAT,A3DDT,A4DDT,
     3     RN,ROBAGZ,ETAP,RMUN,NM1,NM2,MM1,K,KP1
      COMMON/P1/CSEA,IF
      COMMON /P2/ RLENTH,HEIGHT,EPSLON,ACONC,ZZERO,HEAD,UZERO,RECH,EP,
     1     DELTAT,N,M,KD,NTS,NPI
      COMMON /P3/ XL2,ETA,PZROS,PZROF,B1,B2,TB1,TB2,TB1M,TB2M,
     1     B3STAR,B4STAR,B5STAR,EP1M,IRECH
      COMMON /P4/ Q,QDX,QDZ,WX,WZ,DXDZ,DZDX,DXDZG,DZG,IWELL,JWELL
      COMMON /P6/ RKXDD,RKZDD,DELZ12,MM2
      COMMON /P7/ IWP1,IWM1,JWP1,JWM1
      COMMON /P8/ GDZ2,ROBNDT
      COMMON /P9/ RECHED,ROGRHD
      COMMON /P10/ HPM
      PI = 3.141592654D0
      PISQ = PI*PI
      RATIOX = (RKZ*DELXSQ)/(RKX*DELZSQ)
      RNSQ = DFLOAT(N*N)
      XPART = PISQ/((2.0D0*RNSQ)*(1.0D0+RATIOX))
      RATIOZ = 1.0D0/RATIOX
      RMSQ = DFLOAT(M*M)
      ZPART = PISQ / ((1.0D0*RMSQ)*(1.0D0+RATIOZ))
      HMIN = DMINI(XPART,ZPART)
      HMAX = 2.0D0
      HINCRM = DEXP(DLOG(HMAX/HMIN)/(NIPARM-1))
      RHOP(1) = HMIN
      DO 10 LTIME = 2.NIPARM
 10   RHOP(LTIME) = RHOP(LTIME-1)*HINCRM
      WRITE(6,900) NIPARM,(RHOP(JDUM),JDUM=1,NIPARM)
```

```
900  FORMAT(1H0,I5,' ITERATION PARAMETERS;',6(1PD10.3,',',1X)//,27X,6(1)
    .PD10.3))
     RETURN
     END

     SUBROUTINE PRINT2(U,N,M)
     IMPLICIT REAL*8(A-H,O-Z)
     REAL*8 U(N,M)
     ND1 = -9
     ND2 = 0
     DO 1 ITER=1,7
     WRITE(6,990)
990  FORMAT(1H0)
     ND1 = ND1 + 10
     ND2 = ND2 + 10
     DO 2 J = 1,M
     JD = M-J+1
     WRITE(6,900) (U(I,JD),I=ND1,ND2)
2    CONTINUE
1    CONTINUE
     ND2 = ND2 + 1
     WRITE(6,990)
     DO 3 J=1,M
     JD = M-J+1
3    WRITE(6,900)(U(I,JD),I=ND2,N)
900  FORMAT(1H ,10(2X,G11.4))
     RETURN
     END

     SUBROUTINE PRINT3(U,N,M,K,DELX,DELZ)
900  FORMAT(1H ,'Z=',F5.1,10(2X,F10.3))
     SUBROUTINE PRT3DF(U,N,M,K,DELX,DELZ)
     IMPLICIT REAL*8(A-H,O-Z)
     REAL*8 U(N,M,K),DINCRM(10)
     INTEGER*4 BUFFER(10)
     NROWS = N/10
     ND1 = -9
     ND2 = 0
     DO 4 ICOL=1.10
4    BUFFER(ICOL) = ICOL-10
     DO 1 ITER=1,NROWS
     WRITE(6,990)
     DO 5 ICOL=1,10
     BUFFER(ICOL) = BUFFER(ICOL)+10
5    DINCRM(ICOL) = (BUFFER(ICOL)-1) * DELX
     WRITE (6,901)(DINCRM(ICOL),ICOL=1,10)
901  FORMAT(1H0,'X=    ',10(2X,F8.1,2X))
     ND1 = ND1 + 10
     ND2 = ND2 + 10
     DO 2 J = 1,M
     JD = M-J+1
     DXZJD = (JD-1)*DELZ
     WRITE(6,900) DXZJD,(U(I,JD,K),I=ND1,ND2)
2    CONTINUE
```

```
 1    CONTINUE
      WRITE(6,990)
 900  FORMAT(1H ,'Z=',F5.1,10(2X,1PD10.3))
 990  FORMAT(1H0)
      RETURN
      END

      SUBROUTINE PRTDIF(C,CZRO,ND,MD,KDD)
      IMPLICIT REAL*8 (A-H,O-Z)
      REAL*8 C(ND,MD,KDD),CZRO(ND,MD)
      REAL*4 SCR,RDIF
      DIMENSION ICDIF(19),NPRINT(10)
      DATA ICDIF/2H-9,2H-8,2H-7,2H-6,2H-5,2H-4,2H-3,2H-2,2H-1,1H 0,
     .    2H 1,2H 2,2H 3,2H 4,2H 5,2H 6,2H 7,2H 8,2H 9/
      WRITE(6,901)
 901  FORMAT(1H ,T50,'NORMALIZED RELATIVE SALINITY',//)
      DO 10 JDUM = 1,MD
      J = MD-JDUM+1
      DO 20 I=1,ND
      CRELTV = ((C(I,J,2)-CZRO(I,J))/35.8D0)*9.0D0
      IF(CRELTV .GT. 9.0D0) CRELTV = 9.0D0
      IF (CRELTV .LT. -9.0D0) CRELTV = -9.0D0
      SCR = SNGL(CRELTV)
      RDIF = SCR - FLOAT(IFIX(SCR))
      IF (RDIF .GE. 0.5) ICR = IFIX(SCR) + 1
      IF(RDIF .LT. 0.5_ ICR = IFIX(SCR)
      NPRINT(I) = ICDIF(ICR + 10)
 20   CONTINUE
      WRITE(6,900) NPRINT
 10   CONTINUE
 900  FORMAT(1H0,T14,10(A2,7X))
      RETURN
      END

      SUBROUTINE EXHIB(X,Y,ANGLE,VEL,N,M)
C  COMPUTES MAGNITUDE AND DIRECTION OF VELOCITY AT Z-POINTS
      IMPLICIT REAL*8 (A-H,O-Z)
      REAL*8 X(N,M),Y(N,M),ANGLE(N,M),VEL(N,M)
      DIMENSION IPLT1(31),IPLT2(31),IPLT3(31),IPLT4(31),IPLT5(31),
     .    MM(11)
      DATA MM/1H0,1H1,1H2,1H3,1H4,1H5,1H6,1H7,1H8,1H9,1H /
      PI = 3.141592654
      XMAX=-1.E20
      DO 1 J=1,M
      DO 1 I=1,N
      XBAR= X(I,J)
      YBAR= Y(I,J)
      IF(XBAR.EQ.0..AND.YBAR.EQ.0.) GO TO 2
C  ANGLE OF ZERO DEGREES POINTED IN J-DIRECTION
      ANGLE(I,J)=DATAN2(YBAR,XBAR)*180./PI
      IF(ANGLE(I,J).LT.0.) ANGLE(I,J)=ANGLE(I,J)+360.
      VEL(I,J)=DSQRT(YBAR**2+XBAR**2)
      IF(VEL(I,J).GT.XMAX) XMAX=VEL(I,J)
      GO TO 1
      2 ANGLE(I,J)=0.
        VEL(I,J)=0.
      1 CONTINUE
        XFACT = XMAX/99.
        XFACTM = XFACT*864.
        PRINT 3, XFACTM
  3     FORMAT(1H ,'THE SPECIFIC DISCHARGE IN METERS PER DAY =',1PE10.3,
       .' TIMES THE RELATIVE VALUE SHOWN')
        DO 4 JDUM=1,M
        J = M-JDUM+1
        DO 5 I=1,N
        K=VEL(I,J)*99./XMAX
        K10=K/10+1
        K1=K-10*(K/10)+1
        L=ANGLE(I,J)
        L100=L/100+1
        L10=(L=100*(L=100))/10+1
        L1=L-10*(L/10)+1
      7 IPLT1(I)=MM(K10)
        IPLT2(I)=MM(K1)
        IPLT3(I)=MM(L100)
        IPLT4(I)=MM(L10)
      5 IPLT5(I)=MM(L1)
        PRINT 8, (IPLT1(I),IPLT2(I),I=1,N)
  8     FORMAT(1H0,31((2X,2A1,1X),6X))
      4 PRINT 9, (IPLT3(I),IPLT4(I),IPLT5(I),I=1,N)
  9     FORMAT(1H ,1X,31((3A1,2X),6X))
        RETURN
        END

)E
```