

THE USE OF GALERKIN
FINITE ELEMENT METHODS TO SOLVE
MASS TRANSPORT EQUATIONS

By

David B. Grove

392-144

ProQuest Number: 10796100

All rights reserved

INFORMATION TO ALL USERS

The quality of this reproduction is dependent upon the quality of the copy submitted.

In the unlikely event that the author did not send a complete manuscript and there are missing pages, these will be noted. Also, if material had to be removed, a note will indicate the deletion.



ProQuest 10796100

Published by ProQuest LLC (2019). Copyright of the Dissertation is held by the Author.

All rights reserved.

This work is protected against unauthorized copying under Title 17, United States Code
Microform Edition © ProQuest LLC.

ProQuest LLC.
789 East Eisenhower Parkway
P.O. Box 1346
Ann Arbor, MI 48106 – 1346

A Thesis submitted to the Faculty and the Board of Trustees
of the Colorado School of Mines in partial fulfillment of the re-
quirements for the degree of Doctor of Philosophy in Chemical and
Petroleum Refining Engineering

Signed: David B. Grove
Student

Golden, Colorado

Date: April 4, 1976

Approved: John A. Golden
Thesis Advisor

J. A. Gibson
Head of Department

Golden, Colorado

Date: May 4, 1976

ARTHUR LAKES LIBRARY
COLORADO SCHOOL of MINES
GOLDEN, COLORADO 80401

ABSTRACT

The partial differential equation that describes the transport and reaction of chemical solutes in porous media was solved using the Galerkin finite-element technique. These finite elements were superimposed over finite difference cells used to solve the flow equation. Both convection and flow due to hydraulic dispersion were considered. Linear and hermite cubic approximations (basis functions) provided satisfactory results, however the linear functions were found to be computationally more efficient for two dimensional problems. Successive over relaxation (SOR) and iteration techniques using Tchebyschef polynomials were used to solve the sparce matrices generated using the linear and hermite cubic functions respectively. Comparisons of the finite-element methods to the finite difference methods, and with analytical results, indicated that a high degree of accuracy may be obtained using the method outlined. The technique was applied to a field problem involving an aquifer contaminated with chloride, tritium and strontium-90.

TABLE OF CONTENTS

	Page
LIST OF FIGURES-----	vi
LIST OF TABLES-----	vii
ACKNOWLEDGEMENTS-----	1
CHAPTER	
I. Introduction-----	2
A. Purpose and objectives of study-----	2
II. Review of the literature-----	4
A. Development of the equations-----	5
1. The flow equation-----	6
2. The mass transport equation-----	9
B. Analytical solutions-----	11
C. Numerical solutions-----	15
1. Finite difference methods-----	15
2. Method of characteristics-----	19
3. Finite element methods-----	21
a. Variational techniques-----	21
b. Weighted residual techniques-----	22
III. The mass transport equation-----	24
A. Convective and dispersive fluxes-----	24
B. Sinks and Sources-----	26
1. Point and distributed terms-----	26
2. Chemical reactions-----	27
a. Ion Exchange-----	27
b. Rate reactions-----	28

	Page
IV. Development of the Galerkin, finite element-technique-	31
A. Theory-----	32
B. Basis functions-----	35
C. Integral evaluation-----	39
D. Equation solution methods-----	42
1. Time derivative approximations-----	43
2. Matrix solution techniques-----	44
V. Applications of the Galerkin, finite-element technique to the solution of mass transport equations-----	47
A. One-dimensional solutions-----	48
B. Two-dimensional solutions-----	54
VI. Application of the Galerkin, finite-element technique to a field problem-----	64
A. Problem description-----	64
B. Simulation of contamination conditions-----	71
1. Chloride contamination-----	74
2. Tritium contamination-----	74
3. Strontium contamination-----	74
C. Summary and conclusions-----	77
LITERATURE CITED-----	81
APPENDIX A-----	87
APPENDIX B-----	96

LIST OF FIGURES

	Page
Figure 1. Longitudinal dispersion in a column-----	14
2. Plot of Chapeau basis functions with grid spacing h--	36
3. Plot of cubic basis functions with grid spacing h----	40
4. Comparison of finite difference and finite element solutions to the convective-diffusion equation for a Peclet number of 001 and a displaced pore volume of 0.5-----	52
5. Comparison of finite difference and finite element solutions to the convective-diffusion equation for a Peclet number of 0001 and a displaced pore volume of 0.5-----	53
6. A schematic representation of a two-dimensional solute transport problem.-----	61
7. A comparison of numerical techniques for a simulated two-dimensional transport problem for a point directly beneath the well-----	62
8. A comparison of numerical techniques for a simulated two-dimensional transport problem for the intercepting well-----	63
9. Map of Idaho showing the location of the NRTS, The Snake River Plain, and inferred groundwater flow lines (From Robertson, 1974). -----	66
10. The regional ground-water table, May through June 1965, the location of the waste water recharge sites and the modeled area at NRTS, Idaho. -----	69
11. Location of finite difference cells and finite element nodes for the NRTS contamination problem.-----	73
12. Comparison of waste chloride plumes for 1968-1969 based on well sample data and computer model.-----	75
13. Comparison of waste tritium plumes for 1968-1969 based on well sample data and computer model.-----	76
14. Comparison of waste strontium-90 plumes for 1964 based on well sample data and computer model. -----	78

LIST OF TABLES

	Page
Table 1. Numerical diffusion errors caused by various space and time differencing on the convective term in the 1-D mass transport equation-----	18
2. Input rates and concentrations for waste water recharge at NRTS, Idaho-----	67
3. List of routines used in the Galerkin, finite-element program using linear basis functions-----	87
4. List of subroutines used in Galerkin, finite-element programs using cubic basis functions-----	88
5. Definition of selected program variables-----	89
6. Data input formats-----	93

Acknowledgements

This study was sponsored by the United States Geological Survey as a part of the authors ongoing research program. The author is indebted to the many scientists of the USGS who assisted through technical consulting and through encouragement. In particular I wish to thank Dr. John Bredehceft, Deputy Assistant Director of Research of the Water Resources Division, and Dr. Jacob Rubin, Senior Research Scientist of the Division, for their support. Jack Robertson, Research Hydrologist, furnished the results of his study that were used to provide the field test of the model.

Dr. Thomas Manteuffel, Professor of Mathematics, Emory University, provided the results of his then unpublished PhD thesis to solve a difficult problem involving large sparse unsymmetric matrix systems. Professor John Golden offered many helpful comments and critically reviewed the thesis as did members of the PhD advisory committee.

Finally, I would like to acknowledge the encouragement and understanding of my wife Kathleen M. Grove.

I. INTRODUCTION

A. PURPOSE AND OBJECTIVES OF THE STUDY

It has become evident that our social and economic well being is jeopardized by the ever increasing pollution of our water resources. The hydrologic systems that control these resources do not act as separate entities but as vast, complicated, interdependent systems. At one time the water available in streams and lakes was the primary, and in some cases the only, source of quality water available for private and industrial purposes. Due to the stresses placed upon this water resource, the public has become more aware of the vast amounts of high-quality water available underground in aquifers. As more is learned about these sources of water, both use and dependence upon them has increased. This use has both beneficial and detrimental aspects as it includes using the ground-water system for both sources of high quality water and as reservoirs to dispose of unwanted aqueous wastes. This disposal practice coupled with the stress placed on the system and the occasional influx of natural low-quality water has in some instances rendered unfit previously useable ground-water supplies.

These problems have been recognized, and as a result state, federal and local agencies have enacted laws pertaining to the subsurface disposal of wastes and to the use of ground water. Enforcement of such laws and proper planning of ground-water use and subsurface waste disposal practices necessitates a complete understanding of the ground water systems and the effects of the chemical and physical stresses

placed on them. Unfortunately, the movement of water and pollutants through aquifers involves a complicated set of physical laws. Various public and private agencies have therefore developed techniques to compute the effects of various chemical and physical stresses on the ground-water system and the resultant change in water quality and quantity. The solution procedures for pollution simulation studies are termed water-quality models. These techniques include simulation models, usually of a deterministic type, that may take the form of analytical expressions for simplified cases, or involve the use of elaborate computer oriented numerical techniques for the more complicated cases. The complicated nature of the mathematics defining the flow of pollutants through aquifer systems requires sophisticated numerical techniques to solve the equations. Some of the numerical techniques currently available have limited application because of their coarseness of approximation to the actual mathematics or are of such detail and complexity that they prove deficient when it comes to their use and understanding.

This present study develops a numerical model that is rigorous in its solution to the equations yet simple enough that it may be used and understood by a knowledgeable ground-water hydrologist or engineer. The code is proven to be accurate by various checks with known analytical solutions when such are available for the simpler situations. The code is also compared with results from a previously developed numerical technique for more involved systems. The value of this generalized

numerical scheme is demonstrated by solving a ground-water contamination problem. The contamination problem includes the effects of dispersion, both transverse and longitudinal, and sink and source terms with rate and equilibrium controlled chemical reactions for multidimensional mass transport systems.

II. REVIEW OF THE LITERATURE

This section consists of three separate parts that are as follows: The development of equations pertaining to the transport of mass, the analytical solutions of some of these equations, and the numerical solutions of these equations encompassing both finite difference and finite element methods. The experimental studies and case histories are included within their respective solution techniques. The literature in the field of mass transport is great and this review includes only those that are specific to this study. The references are limited to the last 4 or 5 years as the majority of the work previous to this time is well documented in these references.

Prior to the last several years the prediction of the quality of water during its movement through porous media was limited to laboratory studies or to controlled industrial processes. This was due to a lack of ability to accurately define multidimensional flow fields, an incomplete mathematical model of solute transport, inadequate numerical schemes to solve the equations and incomplete descriptions of the relevant chemical reactions taking place.

During the last 4 to 5 years, however, models and modeling techniques have developed at an accelerated rate. The usual water-quality model development procedure has been to simplify the general multi-dimensional transient equations describing mass transport and reactions to one-dimensional situations that can be simulated by controlled laboratory experiments or can be solved in a closed analytical form. The simplifying assumptions are then relaxed and the general model applied to more complex situations. Two-dimensional models for solute transport without chemical reactions but including terms to describe nonideal flow, which we term dispersion, and the existence of sources and sinks within the system were then developed. Investigators refined these models to include the capability to predict and verify chemical reactions for selective field situations. Solution techniques for the equations of transport are by necessity elegant and include finite-difference methods, method of characteristics and finite-element methods, that encompass variational and weighted residual techniques. These methods will be reviewed in more detail to provide a proper framework for the development of the Galerkin, finite-element techniques.

A. DEVELOPMENT OF THE EQUATIONS

The equations of mass transport with chemical reactions have been well documented by physicists, chemists, and chemical engineers as is evidenced by the text of Bird, Stewart and Lightfoot, (1966) and Aris, (1969). Bredehoeft and Pinder, (1973) together with Rendell and Sanada,

(1970) expanded on these basic equations and coupled them with the hydraulic flow equations to present a unified picture of mass transport with or without chemical reactions during flow through saturated porous media.

1. THE FLOW EQUATION

The equation of motion for the flow of ground water has been derived in considerable detail by a variety of workers. Bredehoeft and Pinder, (1973) and Longwell, (1966) present rather complete derivations of the equation. Bredehoeft and Pinder (1973) ground-water flow by the following equation:

$$\nabla \cdot \left[\rho \frac{\bar{k}}{\mu} \cdot (\nabla p - \rho \bar{g}) \right] + \sum_{i=1}^r W_i = \rho \alpha \frac{\partial p}{\partial t}$$

(1)

$$+ \rho_o \epsilon \beta \frac{\partial p}{\partial t} + \frac{\epsilon}{V_o} \sum_{j=1}^n \frac{\partial m_j}{\partial t}$$

where

∇ = vector operator

ρ = fluid density, ML^{-3}

\bar{k} = intrinsic permeability, L^{+2}

μ = dynamic viscosity, $ML^{-1}T^{-1}$

\bar{g} = gravitational acceleration, LT^{-2}

Q_i = source (+) or sink (-) L^3T^{-1}

$$W_i(x, y, z) = \sum_{j=1}^r Q_i(x_j, y_j, z_j) \delta(x-x_j)(y-y_j)(z-z_j)$$

r = number of sources and sinks

α = compressibility of the medium, $M^{-1}LT^{+2}$

ϵ = effective porosity of the medium, L^0

δ = Dirac delta function

β = compressibility coefficient of the fluid, $LM^{-1}T^2$

V_0 = reference volume of the fluid, L^3

m_i = mass of speci i in the reference volume V_0 , M

n = total number of species in the system

Implicit in the derivation of the flow equation is Darcy's Law, which relates specific discharge to hydraulic gradient.

$$\bar{q} = - \frac{\bar{k}}{\mu} \cdot (\nabla p - \rho \bar{g}) \quad (2)$$

where q is the specific discharge of the fluid, LT^{-1} .

The hydraulic flow equation is a mathematical description of the hydraulic potential of the aquifer system. Equation 1 can be simplified

by assuming constant fluid density (Bredehoeft and Pinder, (1973), and written in two dimensions (areal) as

$$S \frac{\partial h}{\partial t} = \frac{\partial}{\partial x} \left(T_{xx} \frac{\partial h}{\partial x} \right) + \frac{\partial}{\partial y} \left(T_{yy} \frac{\partial h}{\partial y} \right) - W(x,y,t) \quad (3)$$

where

S = aquifer storage coefficient, L^0

h = hydraulic head, L

T = transmissivity of the aquifer, $L^2 T^{-1}$

Equation 2 is a definition of Darcy's Law for an anisotropic porous media. Normally the permeability axis can be orientated in such a direction so only the main components of the permeability tensor appear. When this is possible and all of the pressure terms appear in the hydraulic gradient, specific discharge for flow in the x direction is illustrated by.

$$q_x = -k_x \frac{\partial h}{\partial x} \quad (4)$$

A similar equation may be written for the y -direction. The interstitial fluid velocity, as will be used in the mass transport equation, is defined as the specific discharge divided by porosity and is given by

$$v_x = - \frac{k_x}{\epsilon} \frac{\partial h}{\partial x} \quad (5)$$

where

v_x = interstitial velocity, LT^{-1}

ϵ = porosity, L^0 .

2. THE MASS TRANSPORT EQUATION

The mass transport equation for solute speci i as given by Bredehoeft and Pinder (1973) is

$$\begin{aligned} \epsilon \rho_i \alpha \frac{\partial v}{\partial t} + \frac{\partial}{\partial t} (\epsilon \rho_i) = v \cdot \rho \bar{D}_i \cdot v \left(\frac{\rho_i}{\rho} \right) \\ - v \cdot \rho_i \bar{q} + \epsilon \sum_{k=1}^s R_{ik} + W_i \rho_i^* \end{aligned} \quad (6)$$

where

ρ_i = mass per unit volume of speci i, ML^{-3}

\bar{D} = hydrodynamic dispersion coefficient, $L^2 T^{-1}$

R_i = rate of production of speci i in reaction k, $ML^{-3} T^{-1}$

s = number of reactions taking place in system

ρ_i^* = mass per unit volume of sink or source solution, ML^{-3}

The mass transport equation as given by equation 6 can be simplified

and written in two dimension, Bredehoeft and Pinder, (1973) and by Reddell and Sunada, (1970) and is given as equation 7.

$$\begin{aligned} \frac{\partial c_i}{\partial t} = & -v_x \frac{\partial c_i}{\partial x} - v_y \frac{\partial c_i}{\partial y} + \frac{\partial}{\partial x} \left(D_{xx} \frac{\partial c_i}{\partial x} \right) + \frac{\partial}{\partial y} \left(D_{yy} \frac{\partial c_i}{\partial y} \right) + \\ & + \frac{\partial}{\partial y} \left(D_{yx} \frac{\partial c_i}{\partial x} \right) + \frac{\partial}{\partial x} \left(D_{xy} \frac{\partial c_i}{\partial y} \right) - \left(c_{wi} - c_i \right) \frac{W_i}{\epsilon} \\ & + \sum_{k=1}^s R_{ik} \end{aligned} \quad (7)$$

where

c_i = mass concentration of speci i, ML^{-3}

v = interstitial fluid velocity, LT^{-1}

D = dispersion coefficient, L^2T^{-1}

c_{wi} = mass concentration of speci in the well, ML^{-3}

This equation is slightly simplified from that given by Bredehoeft and Pinder (1973), as it assumes that the pressure time derivative in the transport equation is small compared to the other terms and may be neglected. This is actually a good assumption as steady state flow does exist for many problems and for transient conditions this assumption introduces little error.

Equations 3, 5 and 7 represent, for isothermal systems, all that is necessary to completely define most systems. Equation 3 is usually termed the flow equation with hydraulic head calculated as the dependent variable for the space and time field of interest. Equation 7, usually

called the transport equation, is coupled to equation 3 by Darcy's Law (equation 5), which relates ground-water velocity to hydraulic head gradient.

The mass transport equation consists of four terms; the mass accumulation term on the lefthand side of the equation and the convection flux, the dispersion flux term and the sink/source terms on the righthand side of the equation. These terms and the parameters that comprise them will be discussed in more detail in section III.

B. ANALYTICAL SOLUTIONS

In many cases equation (7) can be simplified to the point where analytical solutions are available. Solutions for some of these simpler differential equations representing mass transport are important for several reasons. Several of the contamination problems are well represented by one-dimensional equations and in this case simple analytical solutions can be used to predict the extent and concentration of contaminations. A second reason is that many complicated multidimensional models can be simplified and their accuracy or precision checked with the aid of these simple one-dimensional equations.

Perhaps the equation most widely used to analyze the effects of convection and dispersion in a porous media is the one-dimensional equation with unidirectional flow. This equation results through

simplification of equation 7 and is presented with boundary conditions as given by Ogata and Banks, (1961)

$$\frac{\partial c}{\partial t} = -v \frac{\partial c}{\partial x} + D \frac{\partial^2 c}{\partial x^2} \quad (8)$$

$$c = c_0 \text{ for } x = 0, t \geq 0$$

$$c = 0 \quad x \rightarrow \infty, t \geq 0$$

$$c = 0 \quad x > 0, t = 0$$

These boundary conditions describe the physical situation as illustrated in figure 1. A slug of fluid with concentration c_0 is injected into a column at $x = 0$ and at time = 0. This slug of fluid is then convected and dispersed in the longitudinal direction. The column is assumed of infinite extent and the solution is for the relative concentration of the injected fluid at any time t at location x . Ogata and Banks solution to this equation is given as follows:

$$\frac{C}{C_0} = \frac{1}{2} \left[\operatorname{erfc} \left(\frac{x-vt}{2\sqrt{Dt}} \right) + \exp \left(\frac{vx}{D} \right) \operatorname{erfc} \left(\frac{x+vt}{2\sqrt{Dt}} \right) \right] \quad (9)$$

Equation 9 explains the process of dispersion in a porous media when the dispersion coefficient is relatively small and when one is interested in concentration profiles in the porous media and not at the exist of the column.

Brenner, (1962) analyzed the same physical situation, however, he defined the boundary conditions at the inlet and outlet of the column in a more rigorous and specific manner. Brenner assumed the following boundary conditions;

$$D \frac{\partial c}{\partial x} = -v (c - c_0) \quad \text{for } x = D, t \geq 0$$

$$\frac{\partial c}{\partial x} = 0 \quad \text{for } x = L, t \geq 0$$

which define flux conditions at the inlet and outlet of the column.

Brenner's solution is by necessity more complicated and is given for the relative concentration at the exit of a column of length L:

$$\frac{c}{c_0} = \exp [P(2-T)] \sum_{k=1}^{\infty} \frac{\lambda_k \sin (2\lambda_k)}{(\lambda_k^2 + P^2 + P)} \exp \left(-\lambda_k^2 T/P \right) \quad (10)$$

where

$$T = vt/L$$

$$P = vL/4D$$

λ_n = the positive roots ($k = 1, 2, \dots$) taken in order of increasing magnitude of the equation., and where,

$$\text{Tan } 2\lambda = \frac{2\lambda P}{\lambda^2 - P^2} \quad (11)$$

Brenner (1962) included a tabulation of column exit concentrations for various Peclet numbers. The tabulation is excellent to check the relative concentration exiting from columns in which the flows have large dispersion coefficients associated with them.

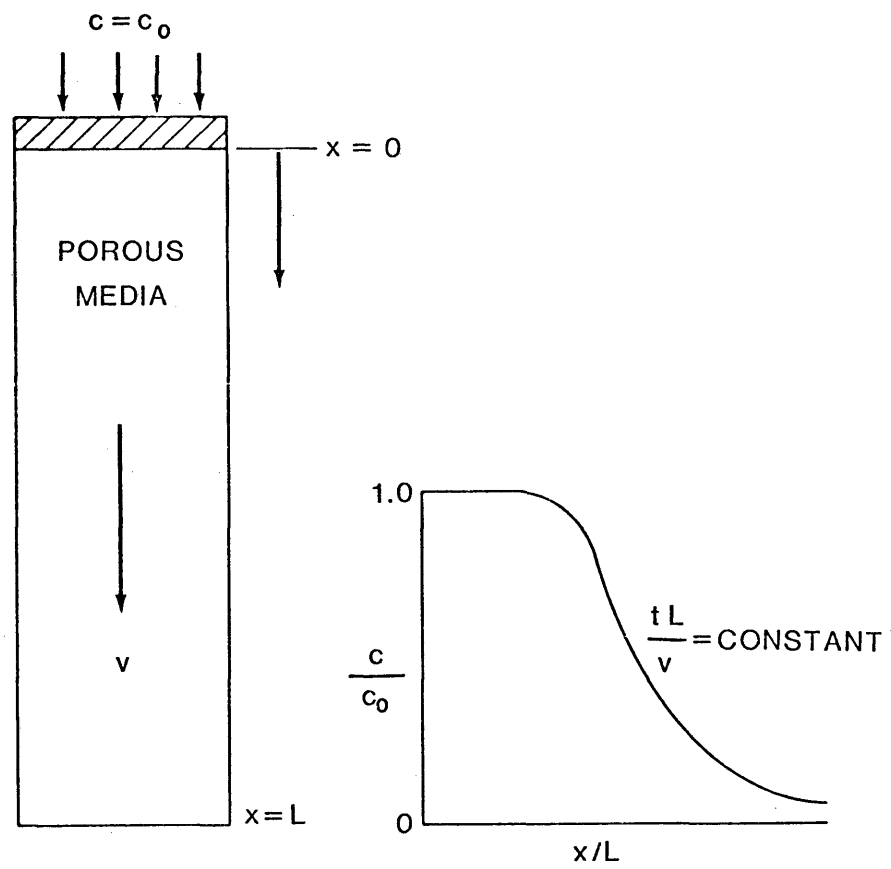


Figure 1.--Longitudinal dispersion in a column.

C. NUMERICAL SOLUTIONS

1. FINITE DIFFERENCE METHODS

Perhaps the simplest numerical technique used to solve both linear and nonlinear differential equations describing mass transport is the finite difference method. Finite difference techniques however, display numerical difficulties when applied to these particular types of equations. These problems can perhaps best be explained in the context of the one-dimensional differential equation as illustrated by Keller (1967):

$$\frac{\partial u}{\partial t} = a(x,t)\frac{\partial^2 u}{\partial x^2} + 2b(x,t)\frac{\partial u}{\partial x} - c(x,t)u + d(x,t) \quad (12)$$

A typical finite difference form to this equation, when variables are centered in space, can be written as follows:

$$u_j^{n+1} = u_j^n + \lambda a^{n+\theta} \left(u_{j+1}^{n+\theta} - 2u_j^{n+\theta} + u_{j-1}^{n+\theta} \right) + \Delta x \lambda b^{n+\theta} \left(u_{j+1}^{n+\theta} - u_{j-1}^{n+\theta} \right) - \Delta t c_j^{n+\theta} u_j^{n+\theta} + \Delta t d_j^{n+\theta} \quad (13)$$

where

$$u^{n+\theta} = \theta u_j^{n+1} + (1-\theta)u_j^n$$

$$\theta = 0 \quad \text{explicit}$$

$$\theta = 1/2 \quad \text{Crank-Nicolson}$$

$$\theta = 1$$

$$\lambda = \Delta t / (\Delta x)^2$$

These typical finite difference forms result in the following system of linear equations

$$\alpha_j u_{j-1}^{n+1} + \beta_j u_j^{n+1} + \gamma_j u_{j+1}^{n+1} = S_j^n \quad (14)$$

that are solved using either iterative or direct matrix techniques. As with most finite difference equations stable solutions occur only with the proper choice of time and space increment sizes.

The following conditions are sufficient to provide stable nonoscillatory solutions to equation 13.

$$1 + \theta \Delta t c(x,t) > 0 \quad (a)$$

$$a(x,t) - \Delta x |b(x,t)| \geq 0 \quad (b) \quad (15)$$

$$1 - (1-\theta) [2\lambda a(x,t) + \Delta t c(x,t)] \geq 0 \quad (c)$$

Condition 15a is usually not a problem as values of concentration are normally positive. Condition 15b results in numerical oscillation if not met and although damped, it may be so large in magnitude to preclude adequate numerical values. For fixed values of a and b, condition 15b requires a small value of the space increment. Condition 15c is met for all fully implicit methods. One must also note that these are "sufficient" and not necessary conditions for stability and in many cases when θ is equal to 1/2 stable solutions result even if condition 15c is not satisfied.

The conditions where finite-difference solutions can be used to solve the differential equations defining mass transport through porous media have been identified. However, the technique that gives stable solutions may result in an additional error caused by the manner in differencing the time and space derivatives. This error is termed a numerical diffusion as its form is identical to the second order diffusion term in the equation. For the one-dimensional equation describing convection-dispersion, Lantz (1970a, 1970b) calculated the magnitude of this numerical diffusion term for various types of spatial and time increments. Table 1 summarizes the results of Lantz's study. Some authors (for example Harlow and Amsden, 1970) reason that the total diffusion term in the equation, both real and numerical, must be positive. Table 1 then indicates that when spatial properties are differenced centrally and time is differenced explicitly, the dispersion coefficient in the equation must be larger than the numerical diffusion term or a negative result appears and unstable solutions result. Table 1 also shows that a spatial central difference and a time Crank-Nicolson difference gives no numerical diffusion error. However, this does not rectify the stability condition given in 15b where the choice of a large spacial increment may cause oscillation in the computed concentration profile. It appears that for all cases small space increments are necessary for adequate solutions of equations with small dispersion coefficients. This does not normally present a problem for one-dimensional equations, however, multidimensional equations result in

large numbers of nodes which produce matrices too large to be solved efficiently with present day computers.

Table 1.--Numerical diffusion errors caused by various space and time differencing of the convective term in the 1-D mass transport equation.

<u>Difference form</u>		<u>Error (second order) X $\frac{\partial^2 c}{\partial x^2}$</u>
<u>Spacial</u>	<u>Time</u>	
BD*	Explicit, $\theta=0$	$(v\Delta x - v^2\Delta t)/2$
CD*	Explicit	$v^2\Delta t/2$
BD	Implicit, $\theta=1$	$(v\Delta x + v^2\Delta t)/2$
CD	Implicit	$v^2\Delta t/2$
BD	C-N*, $\theta=1/2$	$v\Delta x/2$
CD	C-N	0

*BD = Backward difference

*CD = Central difference

*C-N = Crank-Nicolson

Peaceman and Rachford, (1962) presented a finite-difference scheme to solve the two-dimensional transport equation. They illustrated the oscillation problem, caused by small dispersion coefficients, for the one-dimensional case and presented a "transfer of overshoot or undershoot" correction technique. Shamir and Harleman, (1967) studied a steady state flow situations and solved the mass transport equation in terms of streamlines and velocity potentials. Since the velocity is parallel to the streamline one-dimensional flow could be assumed and dispersion tensor cross product terms omitted. Unfortunately both of these approaches have restrictions. The "transfer of overshoot" method of Peaceman and Rachford is not rigorous and the use of streamlines assumes steady state flow conditions and unrealistic one-dimensional flow.

2. METHOD OF CHARACTERISTICS (PARTICLE TRACKING)

Gardner and others, (1964) used a method of characteristics (MOC) to solve the mass transport equation. They essentially modified the MOC method to include a point tracking technique by which particles were given various concentrations and allowed to move with the velocity of water to new points for various time increments. Concentrations were averaged over the various grid domains and the dispersion process calculated by an explicit finite difference method. Reddell and Sunada (1970) and Bredehoeft and Pinder (1973) expanded the one-dimensional

case of Garner's to two-dimensions. Pinder and Cooper (1970) as well as Reddell and Sunada utilized the characteristics method to include density dependence and solved a salt water encroachment problem.

The method and the computer program as developed by Bredehoeft and Pinder has been extensively used within the U.S. Geological Survey to solve field problems involving contamination. Several such studies are as follows: In Georgia, Bredehoeft and Pinder (1973) investigated a contamination problem where saline water upwelled into a fresh water aquifer, and computed the effects of discharge or barrier wells to limit this concentration spread. Hughes and Robson (1973) investigated contamination from sewage lagoons and industrial cleaning areas and predicted the results of various ground-water quality containment practices. Konikow and Bredehoeft (1974) investigated the effects of irrigation and return flow on water quality in the ground-water system and in the adjacent river. Perhaps the best documented use of the method of characteristics has been the application of Robertson and Barraclough (1973) at the National Reactor Testing Station in Idaho Falls, Idaho. They modeled the movement of injected pollutants into a basaltic aquifer and determined the concentration profile for a period of 20 years. Robinson (1974) modified the mass transport equations to include the chemical reaction terms that accounted for ion exchange and radioactive decay.

These studies demonstrated the worth of the method of characteristics for modeling two-dimensional mass transport, but they also showed

it to be cumbersome, expensive and lacking in mathematical rigor. The method of characteristics is perhaps the best technique for hyperbolic equations. The ability to set the dispersion coefficient equal to zero and to model pure convective flow is a distinct advantage for this method. Most processes do involve nonideal flow, characterized by the hydrodynamic dispersion, and simpler numerical techniques should be considered.

Finite element methods

Finite element methods involve integration of some function pertaining to the differential equation over some prescribed element or area. Integration methods, relationship between function and differential equation, and definition of the element over which integration proceeds, all serve to characterize different methods. Most textbooks separate the finite element technique into variational techniques and weighted residual techniques. The variational technique will be discussed first.

A. VARIATIONAL TECHNIQUES

The use of variational calculus to solve partial differential equation that describe transport processes has recently received widespread attention (Forray, 1968 and Schechter, 1967). The coupling of finite elements and variational calculus has lead to a numerical technique that has great promise when the variational calculus principles

apply. The flow equation (3), has been programmed and solved for some time using this method. Remson, Hornberger and Molz's text (1971) on numerical procedures for this type of aquifer equations discusses at some length the techniques and previous work done in this field. Guymon, et al, 1970) perhaps first solved the multidimensional convection-diffusion equations using finite-element variational methods. Guyman's work was expanded by Nalluswami (1971) who improved the numerical techniques, and took into account the tensoral properties of the dispersion coefficient. A recent paper by Smith, Farroday and O'Conner, (1973) compares the variational finite-element technique with the soon to be discussed Galerkin, finite-element technique. The overall consensus of this paper was, unless a definite variational principal exists so that the solution procedures would be identical, the Galerkin, finite-element technique was superior in its efficiency and accuracy in solving the mass transport equation. All variational technique solutions of the mass transport equation were for analytical cases with no application made to the solution of field problems.

B. WEIGHTED RESIDUAL TECHNIQUES

Weighted residual techniques are defined by the method used to weight the residual formed by an approximation to the partial differential equation and the method used to calculate the coefficients used in the approximation. Although there are several weighted residual methods, Finlayson and Scriven (1965) found the Galerkin method best for mass

transport equations. The mathematics associated with the finite element use of the Galerkin weighted residual method will be developed in some detail in the following section. Price, Cavendish and Varga, (1968) first showed the superiority of the Galerkin, finite-element method over the standard finite difference method to solve the one-dimensional mass transport equation. Cavendish, Price and Varga, (1969) utilized this technique to solve nonlinear and multidimensional flow equations. They concluded that for linear equations this method was superior to standard finite-difference methods.

Pinder (1973) used a Galerkin, finite-element technique with the use of isoparametric quadrilateral elements to solve a ground-water contamination problem on Long Island, New York. In this case, elements could take on a variety of configurations and by a mapping procedure be reduced to rectangles. The resulting set of linear equations were solved by a direct matrix technique. This application did not include the introduction of point sources or sinks (as wells) and was conservative in that chemical reactions involving the solute were absent. This technique requires the choice of specific approximating functions somewhat limiting its applicability. The subject of this thesis specifically arranges the nodes in a rectangular form compatible with finite difference techniques. The numerical solution also allows the user to choose a variety of approximating functions, and in this aspect gives more generality to the method.

III. THE MASS TRANSPORT EQUATION

The equations that will be used to define the concentrations of induced chemical species were presented in chapter II. These equations 3, 5, 7, are the flow equation, Darcy's Law and the transport equation respectively. As mentioned previously the transport equation consists of four main terms on the righthand side that account for all changes of concentration. These are the velocity or convective flux, the dispersive flux, physical sink/source terms and chemical reaction terms. These terms and the parameters that make them up will now be discussed in more detail.

A. CONVECTIVE AND DISPERSIVE FLUXES

The convective flux is defined as that mass transport caused by the velocity field. The divergence in this field (the velocity derivatives) is zero in the absence of sink or source terms and is accounted for in the mass transport sink terms when present. The velocities are derived from the potential (flow) equation using Darcy's law. The dispersive flux term is a tensorial quantity and as such deserves more explanation. Scheidegger (1961) and Bear (1972) point out that the dispersion tensor should have 81 components for a three dimensional case, but are able through the use of symmetry to reduce this number to 32 individual components. They are able, for an isotropic media, to reduce the number of components of the tensor to nine for a three dimensional case and four for the two dimensional case.

The general form of the dispersion tensor is given by Scheidegger

(1961) as

$$D_{k1} = \sum_{i=1}^2 \sum_{j=1}^2 a_{ijkl} v_i v_j / |V| \quad \text{for } k = 1, 2 \\ 1 = 1, 2 \quad (16)$$

where 1 = x

2 = y

and

$$a_{ijkl} = \alpha_L \delta_{ij} \delta_{kl} + \frac{(\alpha_L - \alpha_T)}{2} (\delta_{ik} \delta_{jm} + \delta_{im} \delta_{jk}) \quad (17)$$

where δ_{ij} = dirac delta function = 1, $i=j$
= 0, $i \neq j$

α_L = longitudinal dispersivity, L

α_T = transverse dispersivity, L

The four dispersion coefficients of two-dimensional flow can now

be written as

$$D_{xx} = \alpha_L v_x^2 / |V| + \alpha_T v_y^2 / |V| \\ D_{yy} = \alpha_L v_y^2 / |V| + \alpha_T v_x^2 / |V| \\ D_{xy} = D_{yx} = (\alpha_L - \alpha_T) v_x v_y / |V| \quad (18)$$

where $|V| = \sqrt{v_x^2 + v_y^2}$

B. SINKS AND SOURCES

In the mass transport equation there are a variety of sink and source terms that must be considered when solving the equation. The most obvious are be inputs or outputs in the system caused by wells or leakage through the aquifer. Chemical reactions are another means to increase or decrease the concentrations of solute species. After discussing point sources two of the most frequently occurring reactions will be discussed in detail.

1. POINT AND DISTRIBUTED TERMS

Concentration changes caused by the convective flow of solutes in or out of an aquifer can occur through wells or areas normal to the flow field. They may be treated identically or by different techniques depending on the analysis choice of mathematics. The term representing such effects in equation 7 is

$$(c_{iw} - c_i) \frac{W_i}{\epsilon} = \sum_{i=1}^r Q_i(x_i, y_i) (c - c_{iw}) \delta(x - x_i)(y - y_i) \quad (19)$$

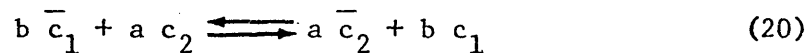
where c_{iw} is the concentration of the fluid recharging or discharging from the system through the well. This term drops out for a pumping well and for a recharging well the effect caused by the term is the difference between injection and input concentrations. The dirac delta function assures values for the term only at nodal points where wells are present.

2. CHEMICAL REACTIONS

Two principal chemical reactions that often occur in contamination problems are equilibrium-controlled ion exchange reactions with a linear adsorption isotherm and irreversible first order rate reactions. The mathematics that describe these reactions are linear and thus allow the use of simpler numerical solution techniques. The addition of linear terms to a differential equation, in most cases, requires no additional mathematical analysis. The use, for example, of a zero order reaction would cause no problem in analytical or numerical analysis.

a. Ion exchange

A typical such exchange reaction might be as follows (Bolt, 1967, Helfferich, 1962)



where 1 and 2 are chemical exchanging species
with a and b valences respectively.

The adsorbed species is given as \bar{c} and the dissolved species as c. The use of a selectivity coefficient may be used to relate the concentration of products and reactants at equilibrium:

$$K_s = \frac{(\bar{c}_2)^a (c_1)^b}{(\bar{c}_1)^b (c_2)^a} \quad (21)$$

where K_s is the ion exchange selectivity coefficient.

Equation 21, when incorporated into the mass transport equation, results in nonlinear terms because of the introduction of more than one dependent variable (chemical species) into the equation. One particular situation that often occurs results in a simplified equation, and is described as follows: when the exchanging ion is very low in concentration relative to the other ions, then exchange process will not materially effect the concentration of this ion, either in solution or adsorbed on the matrix. The adsorbed phase of the major ion is then nearly equal to the cation exchange capacity (CEC), and the solution phase of the major ion equal to the total concentration (C_o). Equation 21 can then be rewritten as,

$$K_s = \frac{(\bar{c}_2)^a (C_o)^b}{(CEC) (c_2)^a} \quad (22)$$

or

$$K_d = \frac{\bar{c}_2}{c_2} = \left[\frac{K_s (CEC)^b}{(C_o)^b} \right]^{1/a} \quad (23)$$

where K_d = ion exchange distribution coefficient.

Equation 23 postulates a linear relationship between the adsorbed species and the solute species where the slope of the equilibrium ion exchange isotherm is the distribution coefficient. Equation 23 thus provides the relationship between the adsorbed species and dissolve species necessary to solve the mass transport equation.

b. Rate reactions

The subsurface disposal of radioactive products is an example where a first order irreversible rate reaction occurs. This reaction is the radioactive decay of the species, adsorbed or in solution. The rate constant can be derived in the following manner. The disappearance of a species by a first order irreversible reaction is given by the equation

$$\frac{dc}{dt} = -kc \quad (24)$$

where k is the rate constant, T^{-1} . This equation can be integrated with integration limits chosen as the time necessary for the initial concentration to decrease by half

$$\int_{c_0}^{c_0/2} \frac{dc}{c} = -k \int_0^{t_{1/2}} dt \quad (25)$$

where c_0 = initial concentration, ML^{-3}

$t_{1/2}$ = half life of species, T

The equation is integrated and solved for k to obtain

$$\lambda = k = \frac{0.693}{t_{1/2}} \quad (26)$$

This is one of the few cases where a rate constant can be obtained without experimentation.

The chemical reactions terms can now be incorporated into the mass transport equation in the following manner: Assume the transport of a chemical species is disappearing by a first order irreversible rate reaction and is being exchanged reversibly by an equilibrium controlled process in which the exchange isotherm is linear. The equation describing this reaction in one dimension is

$$\frac{\partial c}{\partial t} + \frac{\rho_b}{\epsilon} \frac{\partial \bar{c}}{\partial t} = -v \frac{\partial c}{\partial x} + D \frac{\partial^2 c}{\partial x^2} - k \left(c + \frac{\bar{c} \rho_b}{\epsilon} \right) \quad (27)$$

where ρ_b = bulk density of the solid matrix, ML^{-3}

c = dissolved concentration

\bar{c} = adsorbed species concentration

Equation 23 gives the relationship between c and \bar{c} and upon differentiation and substituting into equation 27 gives

$$\frac{\partial c}{\partial t} \left(1 + \frac{K_d \rho_b}{\epsilon} \right) = -v \frac{\partial c}{\partial x} + D \frac{\partial^2 c}{\partial x^2} - kc \left(1 + \frac{K_d \rho_b}{\epsilon} \right) \quad (28)$$

The retardation factor R_f is defined as

$$R_f = 1 + K_d \rho_b / \epsilon \quad (29)$$

Equation 28 is divided by this retardation factor and the dispersion coefficient written proportional to a dispersivity term α_L to obtain

$$\frac{\partial c}{\partial t} = - \frac{v}{R_v} \frac{\partial c}{\partial x} + \frac{\alpha_L v}{R_f} \frac{\partial^2 c}{\partial x^2} - kc \quad (30)$$

The retardation factor thus retards the flow of the solute species creating an average ion velocity term in the equation. This reduced velocity is in many cases, the safety factor which prevents excessive movement of possibly harmful pollutants.

IV DEVELOPMENT OF THE GALERKIN FINITE ELEMENT TECHNIQUE

The Galerkin technique is one of the weighted residual methods discussed in its classical form by numerous authors (Crandall, 1956, Snyder, et. al., 1964, Finalyson, 1969 and 1972). It has shown to be the most efficient of these methods by Finalyson and Scriven (1965) for the solution of mass transport equations. The Galerkin method is an old numerical technique, classically used to solve nonlinear differential equations not amenable to analytical techniques. Recent advance in computing techniques popularized this particular weighted residual method by coupling it with the finite-element technique (See literature review section). The theoretical development of the technique by Douglas and Dupont (1970) lends strong basis to the use of this particular technique for the solution of mass transport equations.

A. Theory

The principle of the Galerkin finite element method is to first choose a set of functions discretized in space but continuous in time to approximate the solution of the differential equation. This set of approximating functions contains time dependent coefficients and basis functions that are real valued and piecewise continuously differentiable over the interval of interest. If $L[c(t,x,y,z)]$ is the differential operator this series approximation is written as

$$c_{\infty}(t,x,y,z) = \sum_{i=1}^{\infty} c_i(t)v_i(x,y,z) \quad (31)$$

where c_{∞} = solution to the differential equation $L[c]=0$

c_i = coefficient in the approximating function

v_i = basis functions.

Since a finite number of coefficients must be evaluated, an approximation to the true solution results. This approximation is written

$$c_n(t,x,y,z) = \sum_{i=1}^n c_i(t)v_i(x,y,z) \quad (32)$$

where c_n = approximation to the solution for n terms. The closeness of this approximation, c_n , to the real solution, c , depends upon three criteria; 1. care in choosing proper basis functions; 2. the number of terms n, in the series and 3. the method used to evaluate the coefficients c_i . It is this third criteria, that of choosing the best

method to evaluate these unknown coefficients, that separates the Galerkin method from the other weighted residual methods. Weighted residual methods use the concept of a residual, R , in their development. This residual is equal to the specific differential equation $L[c]$ and is formed by substituting into the differential equation the previously mentioned series approximation for the dependent variable thus

$$\begin{aligned} L [c_{\infty}(t,x,y,z)] &\approx L [c_n(t,x,y,z)] \\ &= L \left[\sum_{i=1}^n c_i(t)v_i(x,y,z) \right] = R \end{aligned} \quad (33)$$

where L = the differential operator

R = the residual

The residual vanishes, or is identically equal to zero for a true solution to the equation. In general the ideal approximation will not be made, and the residual will not be equal to zero. Galerkin's contribution to the weighted residual methods is how to weight the residual in an optimum manner allowing the approximation to be accurate. Galerkin chose this weighting function identical to the approximating functions (basis functions) used in the original approximation and set this weighted average equal to zero. The weighted average of the residual can be defined and set equal to zero as follows

$$\frac{\int_{dv} R v_k(x,y,z) dv}{\int_{dv} v_k(x,y,z) dv} = 0 \quad (34)$$

where v_k = weighting function

R = residual

Equation 33 and 34 can be combined to give

$$\int_{dV} L \left[\sum_{i=1}^n c_i(t) v_i(x,y,z) \right] v_k(x,y,z) dV = 0 \quad \text{for } k = 1, 2, \dots, n \quad (35)$$

When the series approximation, equation 32, is substituted into the differential equation $L[c]=0$, the approximation may have to be differentiated several times as the equation warrants. The differential equation may contain second order derivatives and unless the basis functions can be differentiated twice, a trivial solution occurs. This problem can be rectified by integrating all second derivatives by parts. Simple integration by parts is defined for the variables u and v as

$$\int_a^b u dv = uv \Big|_a^b - \int_a^b v du \quad (36)$$

This technique can be applied to multiple integrals as well.

After integration of equation 35, a set of n linear differential equations with the follow form results:

$$[\alpha] \frac{dc_i}{dt} + [\beta] \bar{c}_i + [\lambda] = 0 \quad (37)$$

where α , β and λ are coefficient matrices resulting from the Galerkin integration and $\frac{dc_i}{dt}$ and \bar{c}_i are column vectors representing the unknown coefficients. This set of differential equations is then differences with respect to time and the resulting set of linear algebraic equations solved by appropriate techniques, some of which will be discussed later.

B. BASIS FUNCTIONS

Basis functions are usually chosen so that they have analytical properties that conform to the equation, boundary conditions or result in simplified equations for ease of computation during the analysis process. The basis functions are usually defined over a limited number of finite elements. That is they have values at element edges or intersections (nodes) of 0 or 1 thus ameanable to calculation. They are also usually easily integrated by some numerical or analytical technique. Simple polynomials are good choices. Two common sets of basis functions are linear and cubic polynomials.

A linear set of functions for one dimension is described over the preselected intervals as follows (Price, et. al., 1968 and Doherty, 1972).

$$w_i(x) = \begin{cases} (x - (i-1)h)/h & ; & (i-1)h \leq x < ih \\ ((i+1)h - x)/h & ; & ih \leq x \leq (i+1)h \end{cases} \quad \text{for} \quad (38)$$

These particular basis functions form roof or shingle type straight-line segments over the interval of interest. They are hence given the name linear or chapeau basis functions. Figure 2 shows these basis functions over one-dimensional elements of length h . At the node points, each basis function has a value of one and all other basis functions have a value of zero. Each basis function is overlain by the two adjacent basis functions. Integration for a node point i is thus required only over the area of three basis functions, thus, eliminating integration over the entire interval.

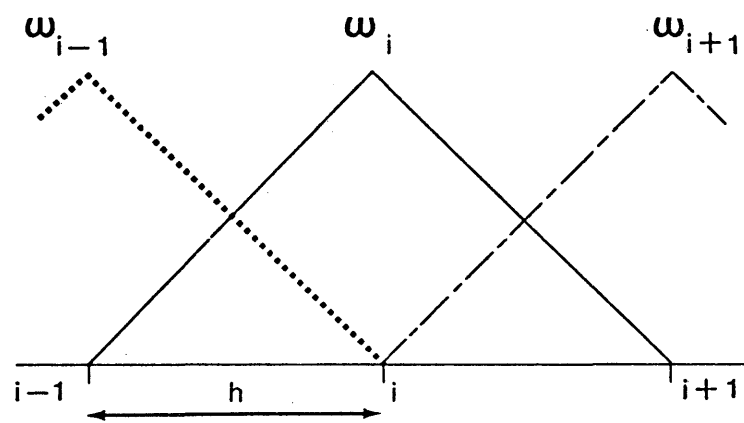


Figure 2.--Plot of Chapeau basis functions with grid spacing h .

The use of linear functions in one-dimensional problems results in a set of linear equations with three unknowns in each equation. The matrix associated with these equations is tridiagonal and therefore simple computational methods can be used for solution. The method is second order accurate with respect to space, as are space centered finite-difference equations, but has been shown (Price, et. al., 1968) to offer advantages in the prevention of solution oscillations not afforded by finite-difference techniques. Oscillations do occur, however, as element intervals become larger. The use of linear functions does not allow the user to obtain derivatives of the final solution at the node points, since the basis functions are discontinuous at the nodes.

Another basis function commonly used is the hermite cubic function. It is composed of two cubic polynomials that offer several advantages not available with the linear function. This particular set of functions is also defined over two elements (for one-dimensional problems), or three nodes but has the property that each set is continuous over the center nodes, thus allowing first derivatives to be computed. One set has a value of zero at each end point and one at the center. Slopes are zero at each node point. The second set has values of zero at each node, slopes of zero at each end point and a slope of unity at the center node point. These sets of functions are described over the preselected intervals as follows (Price, et. al. 1968 and Doherty, 1972).

$$\begin{aligned}
 w_{2i}(x) &= \begin{cases} (-2x + (1 + 2i)h)(x - (i-1)h)^2/h^3 & ; (i-1)h \leq x \leq ih \\ (2x + (1-2i)h)(x - (i+1)h)^2/h^3 & ; ih \leq x \leq (i+1)h \end{cases} \\
 w_{2i+1}(x) &= \begin{cases} (x-ih)(x - (i-1)h)^2/h^2 & ; (i-1)h \leq x \leq ih \\ (x-ih)(x - (i+1)h)^2/h^2 & ; ih \leq x \leq (i+1)h \end{cases}
 \end{aligned} \tag{39}$$

Figure 3 shows these basis functions over elements of length h .

For a one-dimensional problem each basis function is overlain with five adjacent basis functions plus itself. There are six unknown at each central nodal point. For one and multi-dimensional problems a price is paid in computational efficiency for the advantage of differentiability at node points and fourth order accuracy. Solution techniques to methods utilizing band-solve techniques, sparse matrix techniques or iterative methods.

Inspection of figures 2 and 3 and of equation 32 shows that the solution at a particular node involves the sum of basis functions times the time-dependent coefficient c at that node. This means that the solution at that particular point is equal to the value of the time-dependent coefficient. The Galerkin, finite-element method does however, provide, through equation 32, solutions at any point. Finite-difference methods provide solutions only at nodes and solutions are obtained elsewhere by interpolation.

Figures 2 and 3 present basis functions for the one-dimensional case. When more than one dimension is required a product of basis

functions results. The basis function $v_i(x,y)$ for two dimensions can be written as a product of the functions $w_i(x)$ and $w_j(y)$. The function $v_k(x,y)$ can likewise be written $w_k(x) w_l(y)$. The Galerkin formulation for the solution of an equation in two dimensions would then be written

$$\iint_L \left[\sum_{i=1}^n \sum_{j=1}^m c_{ij}(t) w_i(x) w_j(y) \right] w_k(x) w_l(y) dx dy = 0$$

for $k = 1, 2, \dots, n$
 $l = 1, 2, \dots, m$ (40)

This produces $n \times m$ equations with $n \times m$ unknowns; however no new mathematical procedures are required for solution. For linear basis functions nine unknowns are in each equation or row of the matrix and for the cubic functions 36 unknowns appear.

C. INTEGRAL EVALUATION

The solution of equation 35 requires that numerous integrals be evaluated. These integrals may be composed of simple polynomials and in many cases can be evaluated analytically. However, the method is not limited to the use of simple polynomials, such as the linear functions, but can use more complicated functions such as cubics and quintics. Multidimensional equations that result in products of several basis functions should also be considered. Such integrations are certainly

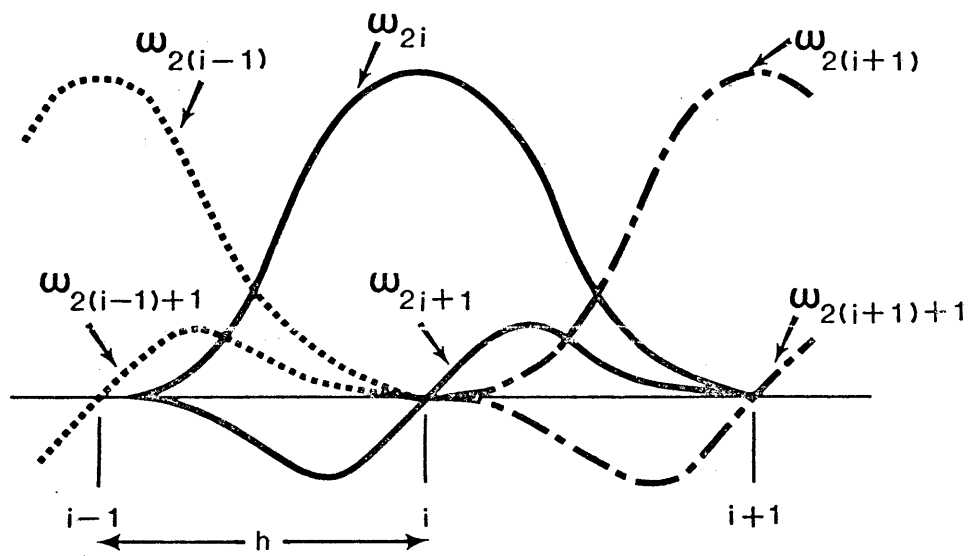


Figure 3.--Plot of cubic basis functions with grid spacing h .

more tedious and some cases more difficult. In this case, it is necessary to resort to numerical integration of the resulting integrals. When cheapeau basis functions were used for the one dimensional problems integration was done explicitly. Gaussian quadrature was used for one-dimensional problems with cubic basis functions and for all two-dimensional problems. Gaussian quadrature gives exact integrations for polynomials of degree $2n - 1$ when at least n gauss points are used, however satisfactory results can be obtained using less gauss points.

As illustrated (IBM, p. 299) Gaussian quadrature formulas can be used to integrate functions in the following manner: To integrate

$$y = \int_a^b f(x) dx$$

let

$$t = \frac{2x - (a+b)}{b-a} \quad (41)$$

where

$$x = \frac{b-a}{2} \cdot t + \frac{b+a}{2}$$

then after suitable algebraic manipulation

$$y_n = (b-a) \sum_{k=1}^n \left\{ \frac{A_k^n}{2} \cdot f \left[(b-a) \frac{t_k^n}{2} + \frac{b+a}{2} \right] \right\} \quad (42)$$

where coefficients A_k^n and nodes t_k^n are given both by the previous reference and in more detail by Krylov (1962). For n equals 3 a polynomial of degree 5 can be integrated exactly. For an integration

interval of 0 to 1 the gauss points are located at 0.113, 0.500 and 0.887.

For two dimensional problems, integration is performed over an area. Procedures for integrating multiple integrals are discussed by Zienkiewicz (1971, p. 148). Integral products of polynomials of degree $(2n-1)$ and $(2m-1)$ require $n \times m$ gauss points for exact integration over the required region. Gauss point locations for $n = 3$ and $m = 3$ over a $(0,1) \times (0,1)$ grid would have 9 locations with respective x and y coordinates of 0.113, 0.5 and 0.887. Position dependent variables within the differential equation may be evaluated at these gauss points during the integration procedure. In most cases these variables are assumed constant within the element and evaluated at its center.

Two dimensional problems with cubic functions result in a sixth degree polynomial when no space derivatives appear. A three point quadrature formula is usually accurate enough for this term and all other terms within the equation with space derivatives are integrated exactly. This formula also results in nine points for a two-dimensional element which is convenient when space dependent variables are important.

D. EQUATION SOLUTION METHODS

It is necessary to select a method to solve the set of linear differential equations illustrated by equation 37. This involves two distinct steps. The first is to approximate the time derivative.

This is usually done using finite-difference techniques. The second is to solve the set of algebraic equations resulting from the particular type of differencing selected. These two steps will be discussed separately.

1. TIME DERIVATIVE APPROXIMATIONS.

Rewrite the matrix equation as

$$[\alpha] \frac{dc_i}{dt} + [\beta] \bar{c}_i = [\alpha] \quad (43)$$

and, as illustrated by equation (13), finite-difference the time derivative in the following manner

$$[\alpha] \frac{\bar{c}_i^{n+1} - \bar{c}_i^n}{\Delta t} + [\beta] \bar{c}_i^{n+\theta} = [\alpha] \Delta t$$

where

$$\bar{c}_i^{n+\theta} = \theta \bar{c}_i^{n+1} + (1-\theta) \bar{c}_i^n \quad (44)$$

The value given to theta depends upon the technique used to increment the time value in the equation. When theta is equal to zero an explicit equation results and when theta is equal to 1/2 an equation centered in time results, (sometimes called a Crank-Nichelson equation). Finally when theta is equal to one, a fully implicit solution results. The most accurate method of time differencing is the Crank-Nichelson approach when theta is equal to 1/2. The more stable method of finite differencing is when theta is equal to one and the method is fully implicit. Although theta can be chosen anywhere between and including zero and one, the usual technique is to choose one of the three. In this analysis theta was chosen equal to 1/2.

An alternate way to write the set of algebraic equations resulting from finite differencing with respect to time would be the use of the "residual" in time method (not to be confused with the use of the residual for the weighted residual methods). Here the difference (residual) in coefficients between time levels at the new and old time level is calculated. This residual in time is defined as

$$\bar{\delta}_i = \bar{c}_i^{n+1} - \bar{c}_i^n \quad (45)$$

By use of this equation \bar{c}_i^{n+1} is removed from the set of equations and $\bar{\delta}_i$ is solved for. This technique allows the user more accuracy if the difference between the coefficient values at the initial and later time values are small. This technique was not necessary for the problems examined and therefore was not used.

A set of algebraic linear equations has now been generated that must be solved for the time dependent coefficients.

2. MATRIX SOLUTION TECHNIQUES

Large systems of linear algebraic equations are normally produced by both finite difference and finite element methods. When five point finite difference schemes are used for two dimensional problems, the resulting system of equations can be solved by convenient iterative methods. Multidimensional finite element programs do not produce simple matrices amenable to such treatment and thus more involved solution methods must be used. Two general methods are available; direct methods

that use a specific number of computational steps and iterative methods that converge to the answer as the number of computational steps increases. Direct and iterative method solution technique efficiencies and accuracies are improved through scaling. The associated matrix was scaled by first dividing each row element by the square root of the diagonal element in it's row. Then each column element was divided by the square root of the diagonal element whose row number corresponded to the column number. This results in a matrix diagonal of all ones. The right hand target vector was divided by the square root of the corresponding diagonal element. After computation the computed value was then rescaled in the same manner as the right hand target vector yielding the answer. For steady-state conditions, scaling of the associated matrix must only be done once but the right hand side vector changes with each time increment and is rescaled each time.

Iteration techniques store only those portions of the matrix with values. Remson, et. al., 1971, Smith, 1965 and Carnahan, 1969 give in detail iteration methods that solve solutions of systems of linear equations. A study of iteration methods brings out some interesting facets. One is that the conditions sufficient and sometimes necessary for convergence of iteration techniques are not always present for matrices developed through Galerkin, finite-element methods. This is not necessarily the case for the simpler one dimensional equation, but for the more complicated two-dimensional situation the matrix may be quite ill conditioned.

The one dimensional problems using cubic basis functions were able to be solved using successive over relaxation (SOR) as was the two dimensional equations using linear basis functions. However the matrix generated by the use of cubic basis functions for two dimensions was so poorly conditioned that the more traditional iterative methods did not suffice. The author was fortunate to obtain the results of a then unpublished PhD thesis in numerical analysis from the University of Illinois Mathematics Department (Manteuffel, pers. comm.) that was able to efficiently solve these sets of equations. The thesis has since been published (Manteuffel, 1975) and the interested reader is referred to the original reference for a complete discussion.

Direct methods, in general, are less susceptible to convergence problems caused by ill-defined matrices than are iteration methods. The problems with these methods are that they may be long and tedious and sometimes require great storage arrays to produce adequate solutions. Typical examples of these methods would be matrix inversion and gaussian elimination.

There are instances, however, when sparse matrices lend themselves to certain techniques of matrix evaluation. This is when the matrix is composed of bands of elements leading down and adjacent to the diagonal. It is sometimes possible when this occurs to use specific linear equation solution techniques that only use storage of, and computation on, these bands of equations. Gaussian elimination is then performed on this reconstructed matrix. The Thomas algorithm for a tridiagonal matrix as

obtained for a one-dimensional problem is an example of this solution technique.

The two dimensional system produced by the cubic basis functions was solved for small sets of equations using an IBM band solve routine DGELB. Manipulation of the associated matrix to conform to the banded structure while preserving low storage was somewhat difficult. The associated matrix and right hand side vector were scaled by dividing each row by its largest absolute value. Columnwise scaling was unnecessary due to pivoting during the gaussian elimination solution procedure. Large systems of equations produced wide bandwidths and excessive storage requirements. Therefore this technique was used for small or moderate sized programs.

V. APPLICATION OF THE GALERKIN FINITE ELEMENT METHOD TO THE SOLUTION OF MASS TRANSPORT EQUATIONS

In this section, solution procedures for the mass transport equation will be constructed. Analytical solutions for one of the simpler cases will be compared with the solutions obtained from the Galerkin finite element technique. For more complex multi-dimensional problems the finite-element method will be compared with finite difference solutions.

Solution of the mass transport equation involving both hyperbolic and parabolic problems will now be discussed in detail. Included will be solution techniques for both one- and two-dimensional mass transport equations with and without chemical reactions.

A. ONE DIMENSIONAL SOLUTIONS

An example of a finite-element solution to a one-dimensional mass transport equation is given by solving the one dimensional diffusion convection equation without chemical reactions.

$$L[c] = \frac{\partial c}{\partial t} + v \frac{\partial c}{\partial x} - D \frac{\partial^2 c}{\partial x^2} = 0 \quad (46)$$

The x dimension is first divided into n-1 finite elements and n nodes. Basis functions are then defined at n nodes that are real valued and piecewise continuously differentiable over the interval of interest. The residual is formed by substituting this approximation into the differential equation for the dependent variable c. Galerkin's technique is applied to the residual by multiplying it by the weighting functions and integrating it over the interval of interest. The following sets of n integral equations with unknowns c(t) and c'(t) results.

$$\int_0^L \sum_{i=1}^n c_i'(t) w_i(x) w_k(x) dx + \int_0^L v \sum_{i=1}^n c_i(t) w_i'(x) w_k(x) dx - \int_0^L D \sum_{i=1}^n c_i(t) w_i''(x) w_k(x) dx = 0 \text{ for } k = 1, 2, \dots, n \quad (47)$$

where

superscript prime (i) indicates differentiation and w_i and w_k denote the approximating and weighting functions respectively.

For every weighting function or value of k there exists a linear differential equation. Equation 47 thus represents a set of n linear differential equations.

The equations that best define the conditions at the boundaries are the flux conditions given by Brenner (1962)

$$D \frac{\partial c}{\partial x} = -v(c - c_0) \quad \text{for } x = 0, t \geq 0$$

$$\frac{\partial c}{\partial x} = 0 \quad \text{for } x = L, t \geq 0$$
(48)

In some situations the boundary conditions are irrelevant such as solid boundaries where the flow is parallel and the normal velocity component zero. Another example is the commonly encountered boundary condition where the flow never reaches the boundary.

When it is important to exactly specify boundary conditions, basis functions and their respective time dependent coefficients are chosen to take on the required characteristics at the boundaries. If for equation 46, boundary conditions are those given by Brenner, basis functions that provide proper solutions at the boundaries must be adaptable to these boundary equations. Substitution of the approximation for the dependent variable into these boundary equations results in the following:

$$D \sum_{i=1}^n c_i(t) w_i'(0) = -v \sum_{i=1}^n [c_i(t) w_i(0) - c_0] \quad \text{for } x = 0 \quad (48a)$$

and

$$\sum_{i=1}^n c_i(t) w_i'(L) = 0 \quad \text{for } x = L \quad (48b)$$

Equation 48b requires that the basis functions at $x = L$ have a slope equal to zero. Cubic functions automatically have this property, however linear functions do not. The $n-1$ and n linear basis functions are therefore modified to give them this property. They are redefined in the following manner

$$w_{n-1}(x) = \begin{cases} (x-(n-2)h)/h & ; (n-2)h \leq x \leq (n-1)h \\ (nh-x)^2/h^2 & ; (n-1)h \leq x \leq nh = 1 \end{cases} \quad (49)$$

$$w_n(x) = ((x-nh)^2-h^2)/h^2 \quad ; (n-1)h \leq x \leq nh = 1$$

The defining equation 46 contains second order derivatives. Unless the basis functions can be differentiated twice, a trivial solution results. This problem is rectified by integrating all second derivatives by parts.

Such integration performed on the last term in equation 47 results in the following:

$$-D \int_0^L \sum_{i=1}^n c_i(t) w_i''(x) w_k(x) dx = - \left\{ \left[D \sum_{i=1}^n c_i(t) w_i'(x) w_k(x) \right]_0^L \right. \\ \left. -D \int_0^L \sum_{i=1}^n c_i(t) w_i'(x) w_k'(x) dx \right\} \text{ for } k = 1, 2, \dots, n \quad (50)$$

After integration by parts the integrand is evaluated at the limits zero and L. Combining 47, 48, and 50 the following equation is obtained.

$$\int_0^L \sum_{i=1}^n c_i'(t) w_i(x) w_k(x) dx + \int_0^L v \sum_{i=1}^n c_i(t) w_i'(x) w_k(x) dx + v [c_1(t) - c_0] w_k(0) + \int_0^L D \sum_{i=1}^n c_i(t) w_i'(x) w_k'(x) dx = 0$$

(51)

for $k = 1, 2, \dots, n$

The next step is to substitute the basis functions into equation 51 and integrate. This results in a set of linear differential equations.

The coefficient matrix that results from the analytical integration of the simple one-dimensional example using linear basis functions is given below.

$$h \begin{bmatrix} 1/3 & 1/6 & & & & \\ & 1/6 & 1/3 & 1/6 & & \\ & & 1/6 & 1/3 & 1/6 & \\ & & & 1/6 & 8/15 & 2/15 \\ & & & & 2/15 & 8/15 \end{bmatrix} \cdot \begin{bmatrix} dc_1/dt \\ dc_2/dt \\ dc_3/dt \\ \vdots \\ dc_{n-1}/dt \\ dc_n/dt \end{bmatrix} + \dots$$

(52)

$$\frac{D}{h} \begin{bmatrix} (1+\alpha) & -(1-\alpha) & & & & \\ -(1+\alpha) & 2 & -(1-\alpha) & & & \\ & -(1+\alpha) & 2 & -(1-\alpha) & & \\ & & & & & \\ & & & -(1+\alpha) & 7/3 & -(4/3 - \alpha) \\ & & & & -(4/3 + \alpha) & (4/3 + \alpha) \end{bmatrix} \cdot \begin{bmatrix} c_1 \\ c_2 \\ c_3 \\ \vdots \\ c_{n-1} \\ c_n \end{bmatrix} = \begin{bmatrix} vc_0 \\ 0 \\ 0 \\ \vdots \\ 0 \\ 0 \end{bmatrix}$$

where $\alpha = \frac{vh}{2D}$

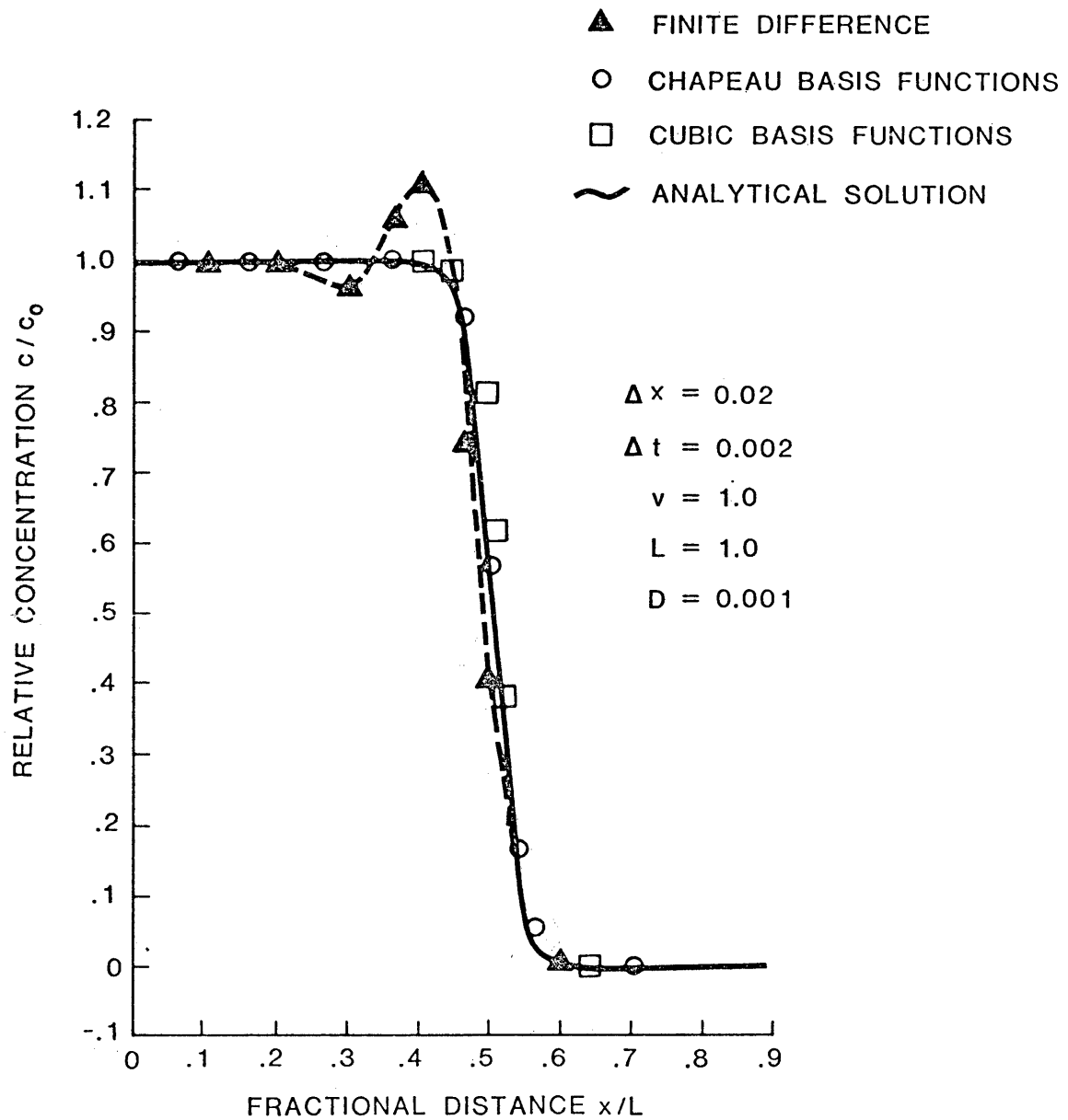


Figure 4.--Comparison of finite difference and finite element solutions to the convective-diffusion equation for a Peclet number of .001 and a displaced pore volume of 0.5.

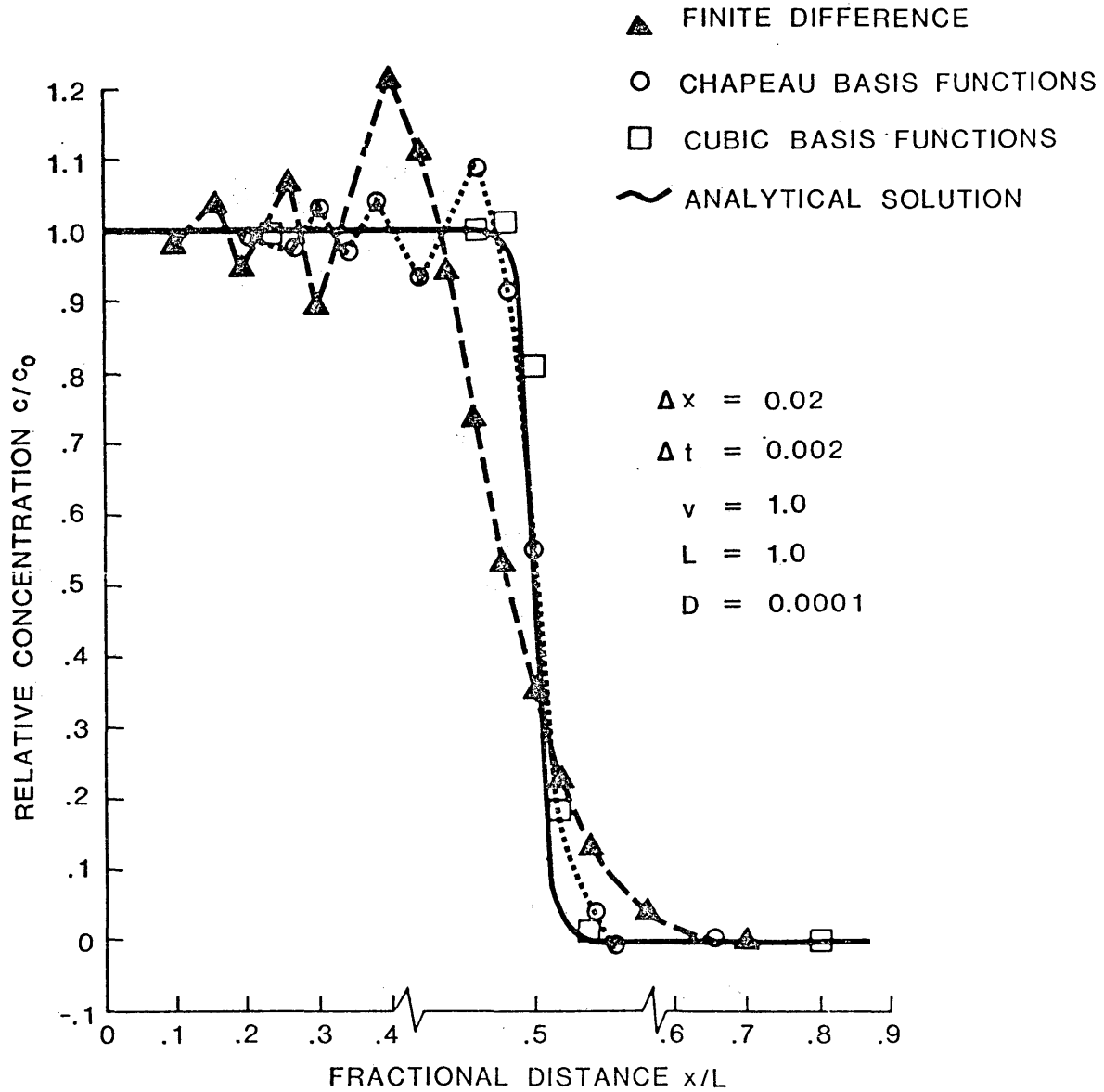


Figure 5.--Comparison of finite difference and finite element solutions to the convective-diffusion equation for a Peclet number of .0001 and a displaced pore volume of 0.5.

This set of equations is solved using the Crank-Nicholson time incrementization. The differential equation describing one-dimensional flow through porous medium is difficult to solve numerically for low values of the dispersion coefficient. Figures 4 and 5 present numerical and analytical solutions respectively, of equation 46 for Peclet (D/vL) numbers of 0.001 and 0.0001. This equation was solved using the Galerkin finite-element method with both linear and cubic basis functions for time steps of 0.002 and a space increment of 0.1. The column length and fluid velocity were both 1.0. A space and time centered finite-difference technique was used to solve this equation. Severe oscillations of the finite-difference solution results for both Peclet numbers. These results demonstrate that sharp fronts, i.e. those with low dispersion coefficients, are difficult to model with standard, finite-difference methods. Note, however, that for figure 4, Galerkin, finite-element techniques using linear and cubic functions closely match the analytical solution. For very low values of the dispersion coefficient (see figure 5) the chapeau functions produce oscillation.

B. TWO-DIMENSIONAL SOLUTIONS

The advantage of finite-element techniques becomes apparent with the solution of the two-dimensional, diffusion-convection equation where small dispersion coefficients predominate. There is basically no difference in the formation of the residual that must be minimized under the integral than was done with the one-dimensional transport equation.

The general transport equation to be solved is equation 7.

As with the one-dimensional equation, second-order differentials appear and must be reduced to first order. As shown previously these terms appear in the description of the dispersion process and are

$$\frac{\partial}{\partial x} (D_{xx} \frac{\partial c}{\partial x}) + \frac{\partial}{\partial y} (D_{yy} \frac{\partial c}{\partial y}) + \frac{\partial}{\partial x} (D_{xy} \frac{\partial c}{\partial y}) + \frac{\partial}{\partial y} (D_{yx} \frac{\partial c}{\partial x})$$

This term is differentiated and the residual formed to obtain

$$\iint \left[D_{xx} \frac{\partial^2 c}{\partial x^2} + \frac{\partial D_{xx}}{\partial x} \frac{\partial c}{\partial x} + D_{yy} \frac{\partial^2 c}{\partial y^2} + \frac{\partial D_{yy}}{\partial y} \frac{\partial c}{\partial y} + D_{xy} \frac{\partial^2 c}{\partial x \partial y} + \frac{\partial D_{xy}}{\partial x} \frac{\partial c}{\partial y} + D_{yx} \frac{\partial^2 c}{\partial y \partial x} + \frac{\partial D_{yx}}{\partial y} \frac{\partial c}{\partial x} \right] w_k(x) w_l(y) dx dy \quad (53)$$

Reduction to first order is accomplished by integration by parts. Integration of this equation can be done in an elegant manner using vector calculus as illustrated generally (Wylie, 1966, p. 572) and specifically (Pinder et. al., 1973) for the flow equation.

Since this integration may produce terms for inclusion into the boundary equations a detailed integration by parts will be performed for the first of the integral terms:

$$u = D_{xx} w_{k\ell}(x,y) \text{ where } w_{k\ell}(x,y) = w_k(x)w_\ell(y)$$

$$dv = \frac{\partial^2 c}{\partial x^2} dx$$

therefore

$$du = D_{xx} \frac{\partial w_{k\ell}(x,y)}{\partial x^2} + w_{k\ell}(x,y) \frac{\partial D_{xx}}{\partial x}$$

$$v = \int \frac{\partial^2 c}{\partial x^2} dx = \frac{\partial c}{\partial x}$$

then the first term of equation 53 becomes

$$\int_{y_0}^{y_L} \left[D_{xx} w_{k\ell} \frac{\partial c}{\partial x} \right]_{x_0}^{x_L} dy - \int_{y_0}^{y_L} \int_{x_0}^{x_L} \left[D_{xx} \frac{\partial w_{k\ell}}{\partial x} \frac{\partial c}{\partial x} + w_{k\ell} \frac{\partial D_{xx}}{\partial x} \frac{\partial c}{\partial x} \right] dx dy$$

Performing this integration on all of the second ordered derivatives results in the following equation

$$\int_{y_0}^{y_L} \left[D_{xx} w_{k\ell} \frac{\partial c}{\partial x} + D_{xy} w_{k\ell} \frac{\partial c}{\partial y} \right]_{x_0}^{x_L} dy + \int_{x_0}^{x_L} \left[D_{yy} w_{k\ell} \frac{\partial c}{\partial y} + D_{yx} w_{k\ell} \frac{\partial c}{\partial x} \right]_{y_0}^{y_L} dx +$$

$$-\int_{y_0}^{y_L} \int_{x_0}^{x_L} \left[D_{xx} \frac{\partial c}{\partial x} \frac{\partial w_{kl}}{\partial x} + D_{yy} \frac{\partial c}{\partial y} \frac{\partial w_{kl}}{\partial y} + D_{xy} \frac{\partial c}{\partial x} \frac{\partial w_{kl}}{\partial y} + D_{yx} \frac{\partial c}{\partial x} \frac{\partial w_{kl}}{\partial x} \right] dx dy \quad (54)$$

The series approximation for the dependent variable (concentration) for the two dimensional, mass transport equation is given as

$$c_{n,m}(t,x,y) = \sum_{i=1}^n \sum_{j=1}^m c_{ij}(t) w_i(x) w_j(y) \quad (55)$$

As mentioned previously integration of two-dimensional problems is carried out over the element area by mathematical operations at positions located at gauss points within the element. When parameters were known to change rapidly with position, selected finite elements (in the area of rapid change) were subdivided into nine subelements, the centers of each closely corresponding to the gauss points. Position dependent variables were evaluated for each of these points. This procedure was used only for elements using cubic basis function and only for elements surrounding nodes that described wells. A complete subdivision of all finite elements in this manner was successfully accomplished but deemed impractical for large scale problems.

The two-dimensional form of the mass transport equation where the chemical reaction terms are assumed to be equilibrium controlled ion exchange with a first order irreversible chemical reaction is now written.

$$\begin{aligned}
& \sum_{i=1}^n \sum_{j=1}^m \int_0^{y_L} \int_0^{x_L} \left\{ c'_{ij}(t) w_i(x) w_j(y) w_k(x) w_\ell(y) + c_{ij}(t) \left[v_x w'_i(x) w_j(y) w_k(x) w_\ell(y) \right. \right. \\
& + v_y w_i(x) w'_j(y) w_k(x) w_\ell(y) + D_{xx} w'_i(x) w_j(y) w_k(x) w_\ell(y) + \\
& + D_{yy} w_i(x) w'_j(y) w_k(x) w_\ell(y) + D_{xy} w'_i(x) w_j(y) w_k(x) w'_\ell(y) + \\
& + D_{yx} w_i(x) w'_j(y) w'_k(x) w_\ell(y) - \frac{Q}{\epsilon} w_i(x) w_j(y) w_k(x) w_\ell(y) + \\
& \left. \left. + k w_i(x) w_j(y) w_k(x) w_\ell(y) \right] \right\} dx dy + \\
& + \sum_{i=1}^n \sum_{j=1}^m c_{ij}(t) \left\{ \int_0^{y_L} \left[(D_{xx} w'_i(x) w_j(y) + D_{xy} w_i(x) w'_j(y)) w_k(x) w_\ell(y) \right]_0^{x_L} dy + \right. \\
& \left. + \int_0^{x_L} \left[(D_{yy} w_i(x) w'_j(y) + D_{yx} w'_i(x) w_j(y)) w_k(x) w_\ell(y) \right]_0^{y_L} dx + \right. \\
& \left. + \int_0^{y_L} \int_0^{x_L} \frac{c_w Q}{\epsilon} w_k(x) w_\ell(y) dx dy = 0 \quad \text{for } k = 1, 2, \dots, n \right. \\
& \qquad \qquad \qquad \qquad \qquad \qquad \qquad \qquad \qquad \qquad \qquad \qquad \qquad \left. \ell = 1, 2, \dots, m \right. \tag{56}
\end{aligned}$$

Integration of equation 56 produces a set of $n \times m$ linear differential equations. The integrals are evaluated using Gaussian numerical quadrature methods as previously discussed. Each algebraic equation will have as many as 9 unknowns for chapeau basis functions and 36 unknowns for cubic basis functions depending on its type and closeness to the boundary. As was the case with the one-dimensional problems the basis functions require that the coefficient value $c_{ij}(t)$ is also the value of the variable $c_{n,m}(x,y,t)$ for linear basis functions and where the product of the basis functions equal one for the cubic functions.

The matrix differential equation that expresses equation 56 after integration can be written

$$[\alpha] \frac{dc_{ij}}{dt} + [\beta] c_{ij} = [\gamma]$$

This equation is then finite differenced with respect to time in the Crank-Nicholson manner and the previously mentioned techniques used to solve the resulting matrix.

The two-dimensional, finite-element solution was compared to an analytical solution as was the one-dimensional case. Unidirectional flow in the x or y direction was imposed on the two-dimensional equation without sinks, sources or chemical reactions. The breakthrough curves, for a problem with boundary conditions similar to those used for the one-dimensional equation, were identical to those obtained for the one-dimensional analytical and numerical solution.

The finite element programs were also compared with a previously developed finite-difference model. The finite difference program has been used previously within the U.S. Geological Survey and checked extensively with analytical solutions to known problems, with satisfactory results. A hypothetical waste contamination problem was used to demonstrate the compatibility of all three models. Two finite element models, using chapeau and cubic basis functions respectively, were used in the simulation study.

The physical system is schematically represented as figure 6. An injection well introduced a conservative solute into an areal flow field directed southward and a pumping well intercepting a portion of the injected fluid during its travel. Figures 7 and 8 give solute concentrations at areal nodal points directly beneath the injection well and at the intercepting well as a function of time. The notation (refined) for one of the simulations using hermite cubic basis indicates more detailed definition of solution parameters at nine gauss points within the finite element. Exact comparisons should not be expected, however the closeness of values for these widely different types of numerical analysis is apparent.

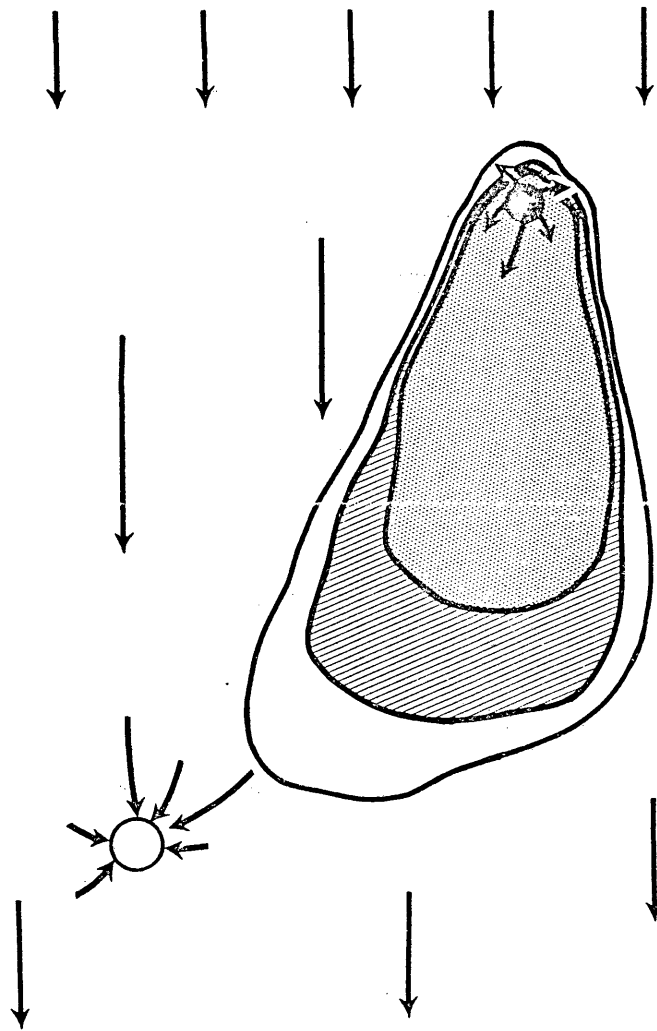


Figure 6.--A schematic representation of a two-dimensional solute transport problem.

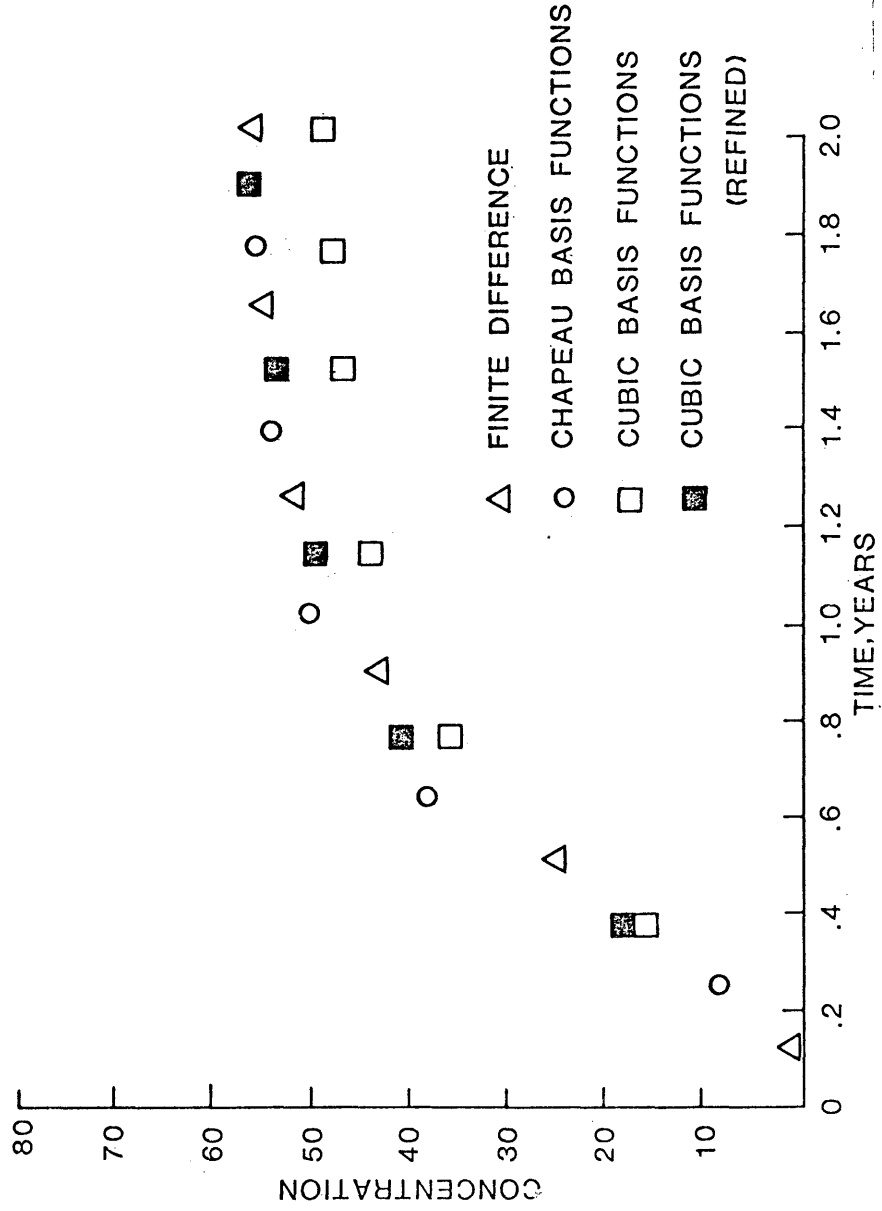


Figure 7.--A comparison of numerical techniques for a simulated two-dimensional transport problem for a point directly beneath the well.

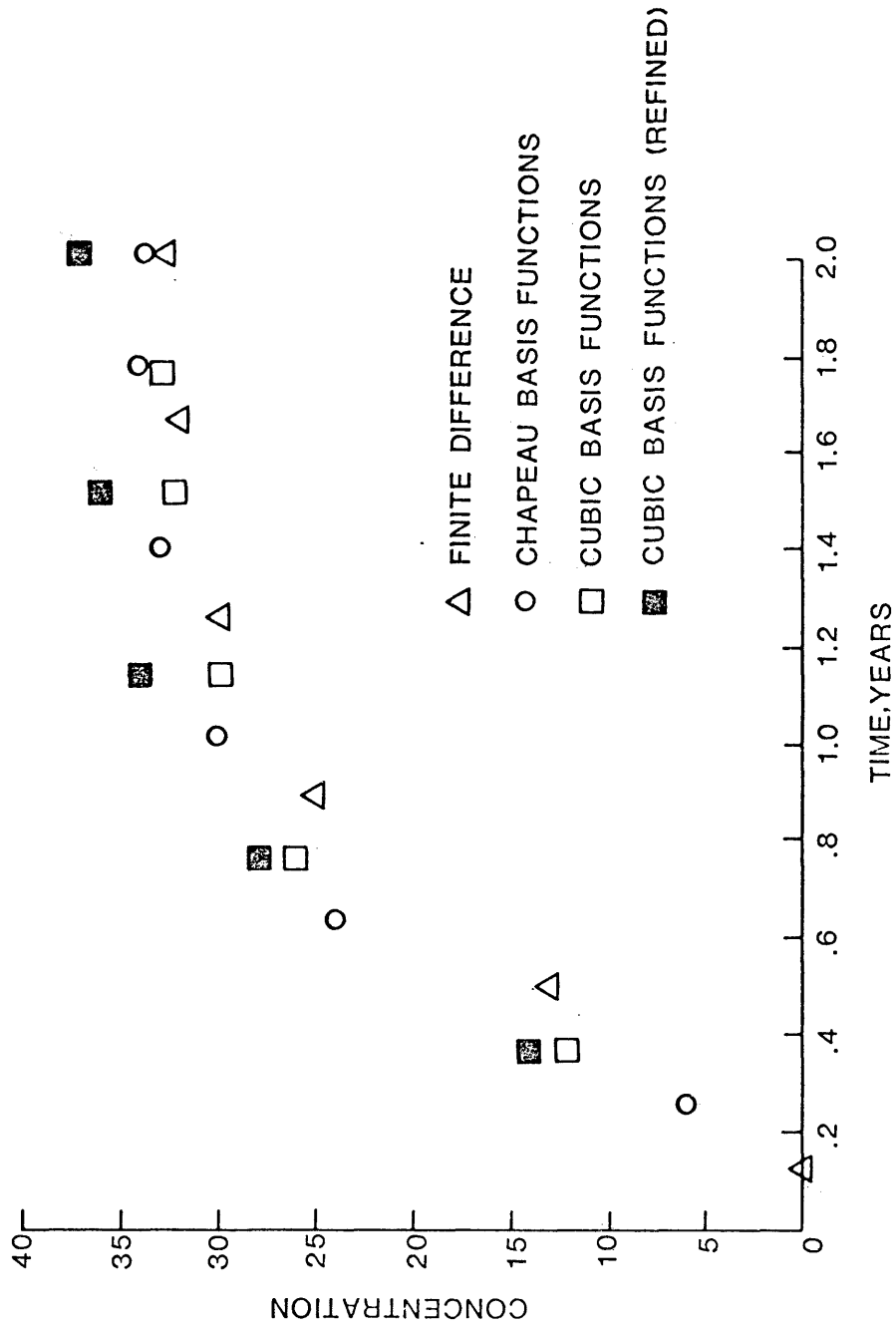


Figure 8.--A comparison of numerical techniques for a simulated two-dimensional transport problem for the intercepting well.

VI. APPLICATION OF THE GALERKIN, FINITE-ELEMENT TECHNIQUE TO

A FIELD PROBLEM:

The Galerkin, finite-element method has been shown in the previous chapters to adequately simulate known analytical solutions to mass transport equations. In this section the method will be applied to a known ground water contamination problem. The area is the Snake River plain aquifer at the National Reactor Testing Station in Idaho. Robertson (1974) modeled the movement of waste through this aquifer using finite difference techniques for the flow model and the method of characteristics for the solute transport model.

The reason that this particular area was chosen is that excellent field data are available and the waste reacts chemically during its movement through the aquifer.

A. PROBLEM DESCRIPTION

Robertson, Schoen and Barraclough (1974) give a detailed geographic, geologic and hydrologic description of the Snake River aquifer underlying the National Reaction Testing Station. The reader interested in such details is referred to reports by these authors. This manuscript deals with one small part of their work and therefore only an abbreviated description of the area and the waste disposal problem will be given here.

The National Reactor Testing Station (NRTS) is located on the Snake River Plain in southeastern Idaho. This plain is underlain by the Snake River aquifer, that is made up of numerous basaltic flows and contains a vast amount of ground water. This ground water is recharged in the higher mountains and flows to the southwest through very heterogeneous basalt discharging through springs into the Snake River. A pictorial map depicting the ground-water flow pattern and salient surface features is shown in figure 9.

Since 1952 chemical and low level radioactive wastes have been disposed of into the ground-water system. These wastes migrate down the hydraulic gradient and, unless removed from the aquifer by physical or chemical means, discharge into the Snake River. The majority of the wastes are injected into the aquifer at two sites; the test reactor area (TRA) and the Idaho Chemical Processing Plant (ICPP). Robertson (1975) gives the input rate for waste water recharge for these two sites and these are shown in table 2.

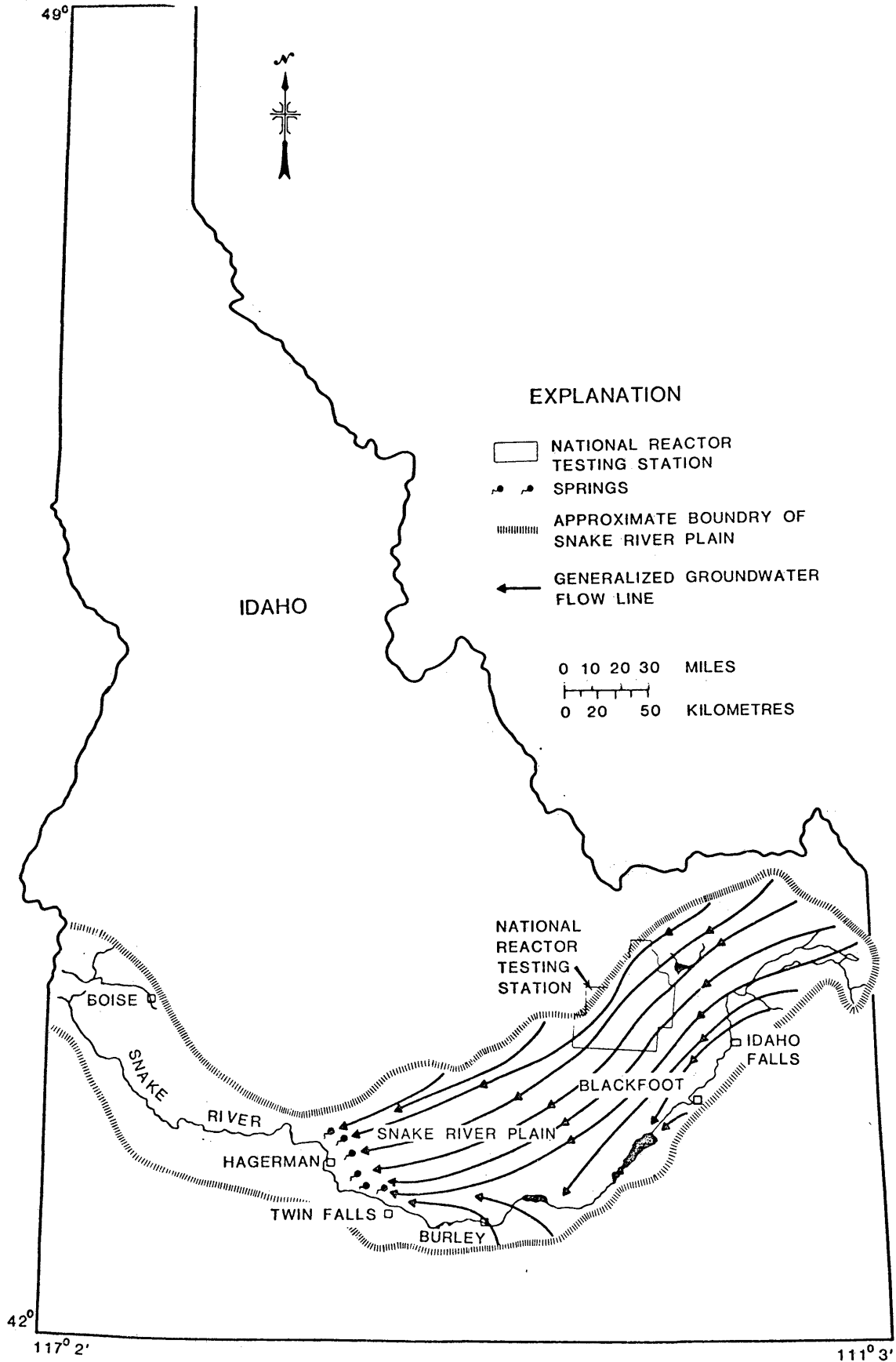


Figure 9.--Map of Idaho showing the location of the NRTS, The Snake River Plain, and inferred groundwater flow lines (From Robertson, 1974).

Table 2. Input rates and concentrations for waste water recharge at NRTS, Idaho

	<u>TRA</u>	<u>ICPP</u>
Chloride (Cl^-)	35 mg/l	245 mg/l
Tritium (H^3)	300 pCi/ml	800 pCi/l
Strontium-90 (Sr^{90})	0.0	1.0 pCi/l
Injection rate	1.9 cfs	1.7 cfs

where:

mg/l is milligrams per litre

pCi/ml is pico curies per millilitre

cfs is cubic feet per second.

Figure 10 illustrates the regional ground-water table, May through June 1965, the location of the waste water recharge sites and the area chosen to be modeled for solute transport.

Sampling of the waste products has been accomplished through monitor wells drilled down gradient from the injection points. The background level for chloride is 10-20 mg/l. The monitoring of tritium and strontium is limited by the detection level of the analysis procedure which is 2 pCi/ml for tritium and 0.005 pCi/ml for strontium.

Movement of these three products through the ground-water system depends on chemical as well as physical factors. The physical factors are the previously discussed effects of convective transport and dispersive transport, the latter being used to approximate non-ideal flow conditions. The chemical factors may include rate reactions and ion

perhaps be lessened by using fewer number but larger sized elements for the cubic equations. The decrease in program sensitivity to parameter values such as areal changes in transmissivity and well locations makes this adjustment presently unattractive.

The model's practical use was demonstrated by solving field contamination problem. The concentration profiles of chloride, tritium and strontium-90 were simulated and compared to field data. The model successfully simulated solute transport for an unreactive conservative solute chloride, a solute with a first order irreversible rate reaction, radioactive decay and a solute with equilibrium controlled ion exchange.

The program incorporated generalized expressions to treat dispersion as a tensor with both longitudinal and transverse components.

Previous studies using finite-difference equations to solve these same sets of transport equations show the finite element methods to be superior for large grid spacings. For the models presented here the linear functions are superior to cubic functions for large two-dimensional problems.

As with any numerical technique further study may reveal shortcuts, improvements; etc. to enhance the technique's features. The most productive research in this area would perhaps be a search for more efficient matrix solution techniques for such sparse unsymmetric systems.

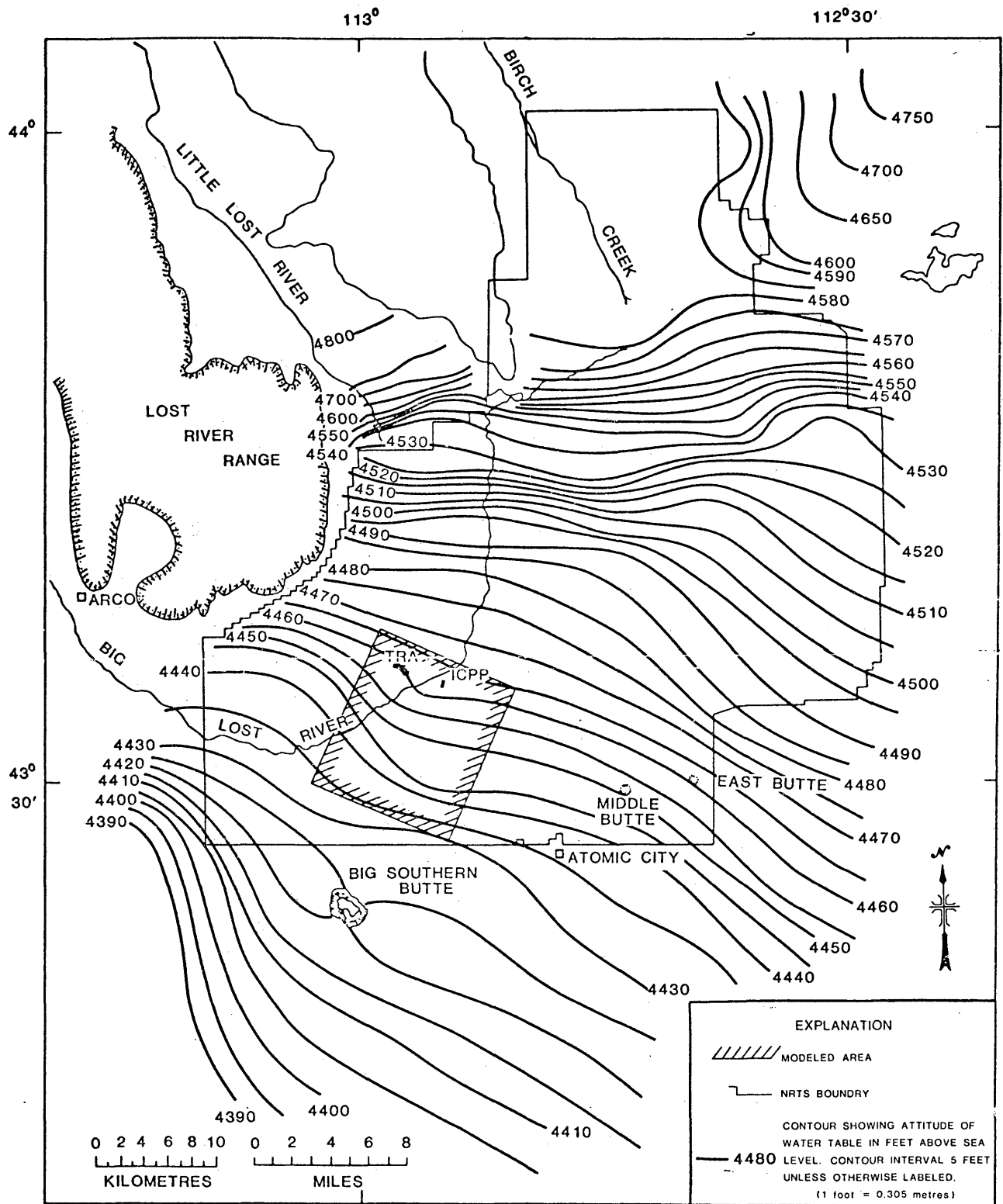


Figure 10.--The regional ground-water table, May through June 1965, the location of the waste water recharge sites and the modeled area at NRTS, Idaho.

exchange. The former is characterized in this instance by radioactive decay of the tritium and the strontium. The latter is characterized solely by the strontium where the small chemical concentrations allow the assumption of linear adsorption isotherms. Since surface controlled ion exchange (as we would expect here) is usually very fast, equilibrium conditions are assumed and the previously developed equations covering this case apply.

Chloride is a non reactive waste speci that is used to characterize the system in terms of the dispersion coefficient. Matching observed with numerically derived data allows determination of dispersivity constants and gives some check on the accuracy of the determined velocities that control the convective transport portion of the transport equation. Using this technique a longitudinal characteristic length of 300 feet and a ratio of $\alpha_L/\alpha_r = \text{one}$, was chosen for this test problem.

The chemical parameters necessary to the solution of the transport equation are now discussed. Equation 26 illustrates the first order rate constant for radioactive decay to be $0.693/\text{half life}$. The half life of tritium is 12.26 years and for strontium-90 is 28.9 years. The respective rate constants for tritium and strontium are then 0.0565 yr^{-1} and 0.0240 yr^{-1} .

Since a linear adsorption isotherm can be used to characterize the exchange process a distribution coefficient is required. The determination of a distribution coefficient that will apply to field situations is difficult. Robertson (1974) reports that of the strontium

injected into the system only 3 percent is present in the aqueous phase. The rest is assumed to be adsorbed on the surface of the aquifer matrix. This information allows calculation of a term related to the distribution coefficient that can be used to calculate the effect of ion exchange. The term $K_d \rho_b / \epsilon$ in equation 28 can be shown for these conditions to be equal to 0.97/0.03 or 32.33. The retardation factor defined by equation 29 is then 33.33. This information can be translated to mean that the average velocity of the strontium ion is traveling 0.03 times the velocity of the ground water.

Initial conditions for contaminants in the ground water system assumed that neither tritium nor strontium were present. As mentioned previously the background level of chloride was assumed to be 10 mg/litre.

B. SIMULATION OF CONTAMINATION CONDITIONS

As illustrated by figure 10 an area of NRTS approximately 8 x 8 miles was chosen to be modeled for the transport of waste solutes. This area was then divided into 20 x 20 finite difference cells. These finite difference cells were used to calculate the detailed head distribution using the finite difference approximation to the flow equation. The finite element net was then overlain on top of the finite difference cells with the finite element node falling on the center of the finite difference cell. The finite element area then comprised quarter portions of four finite difference cells. This program allows the use of existing

finite difference solutions to the simpler equations that define the hydraulic head in the aquifer. The length of the sides of the elements and cells was 2100 feet. Figure 11 illustrates the finite difference and finite element nodes as used for the NRTS contamination problem. The boundary conditions for the mass transport problem assume constant concentrations as the chemical species is assumed not to reach the boundaries of the modeled area. If this does occur to an appreciable extent the modeled area is simply increased to a larger area.

In detail the modeled area contains 4 wells, two discharging (pumping) and two recharging waste water.

All finite element simulation studies were done using linear basis functions. Computer runs for identical problems using the cubic basis functions took an order of magnitude longer for CPU time. When larger size elements were used, to decrease the size of the matrices, the necessary detail for problem solution was lost. The use of cubic basis functions, is recommended for situations where uniform conditions exist throughout the medium. Further research on means to generate more detail within the larger cubic finite elements or more rapid solution techniques for sparse, poorly conditioned matrices, might allow expanded use of this method. Using the input data from table 1 concentration maps were generated for chloride, tritium and strontium. These generated profiles were compared with field data contoured by Robinson (1974).

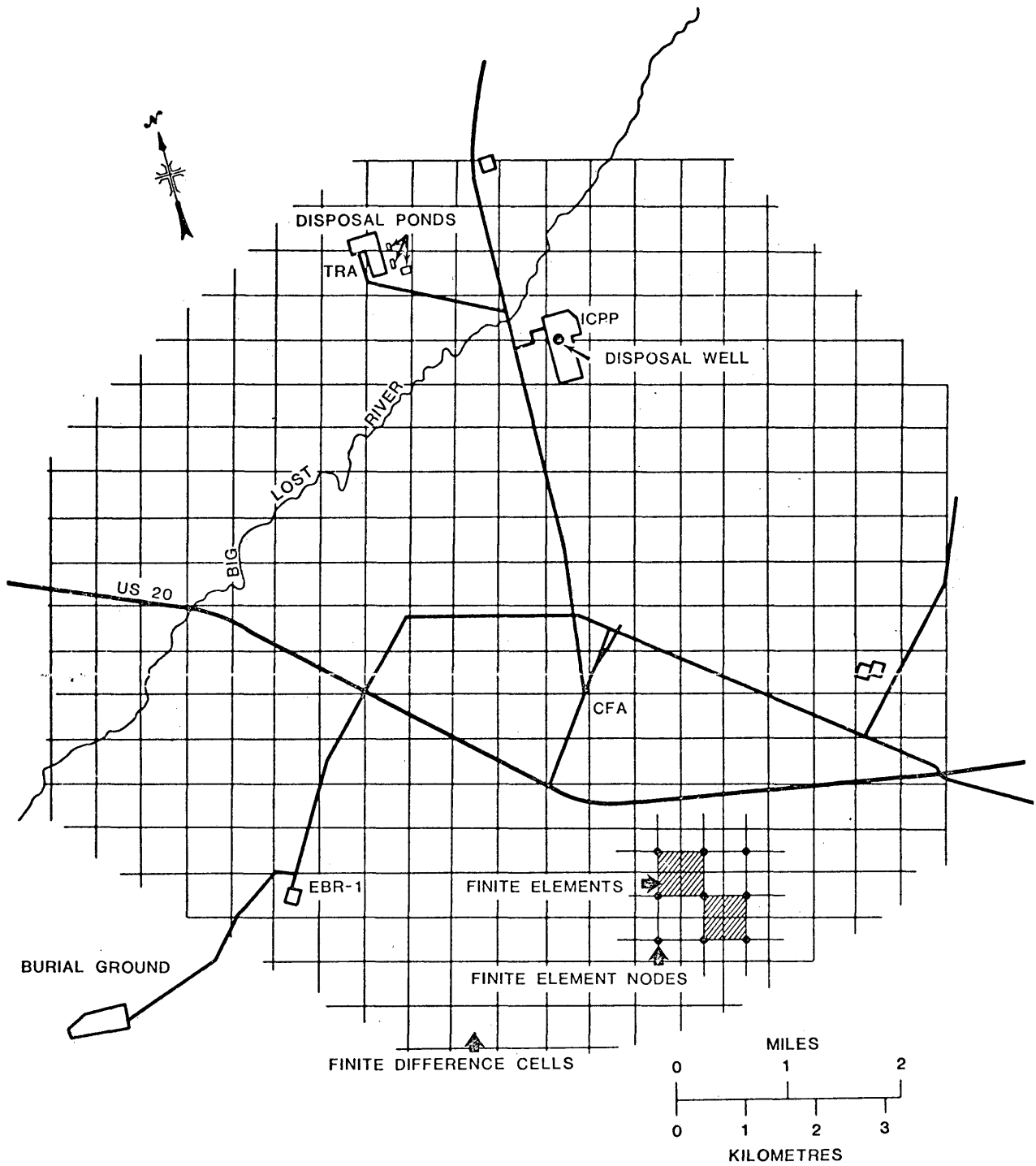


Figure 11.--Location of finite difference cells and finite element nodes for the NRTS contamination problem.

1. CHLORIDE CONTAMINATION SIMULATION

Figure 12 compares the waste chloride plumes for 1968-1969 based on well sample data and the finite element model using the linear basis functions. Comparisons between field and computer simulations are considered good for this type of study. Field generated contours, such as the 15 mg/litre isochlor were in some cases estimates based on one or two sample points. The material balance of the simulation study indicated a 8 percent accuracy for this run of length 17.1 years.

2. TRITIUM CONTAMINATION SIMULATION

Figure 13 compares the waste tritium plumes for 1968-1969 based on well sample data and the finite element model using the linear basis functions. The introduction of an irreversible chemical reaction term to account for radioactive decay distinguishes this simulation from the chloride. Except for the introduction of this term and the different initial and boundary conditions, the mass transport parameters were the same. Excellent correlation between the field data contours and finite element contours again resulted.

3. STRONTIUM CONTAMINATION SIMULATION

Strontium-90 was only injected into the ICPP disposal well at relatively low concentrations. The high degree of ion exchange retards the movement of this dangerous radioactive species through the groundwater system. This chemical reaction adds a term to the equation to

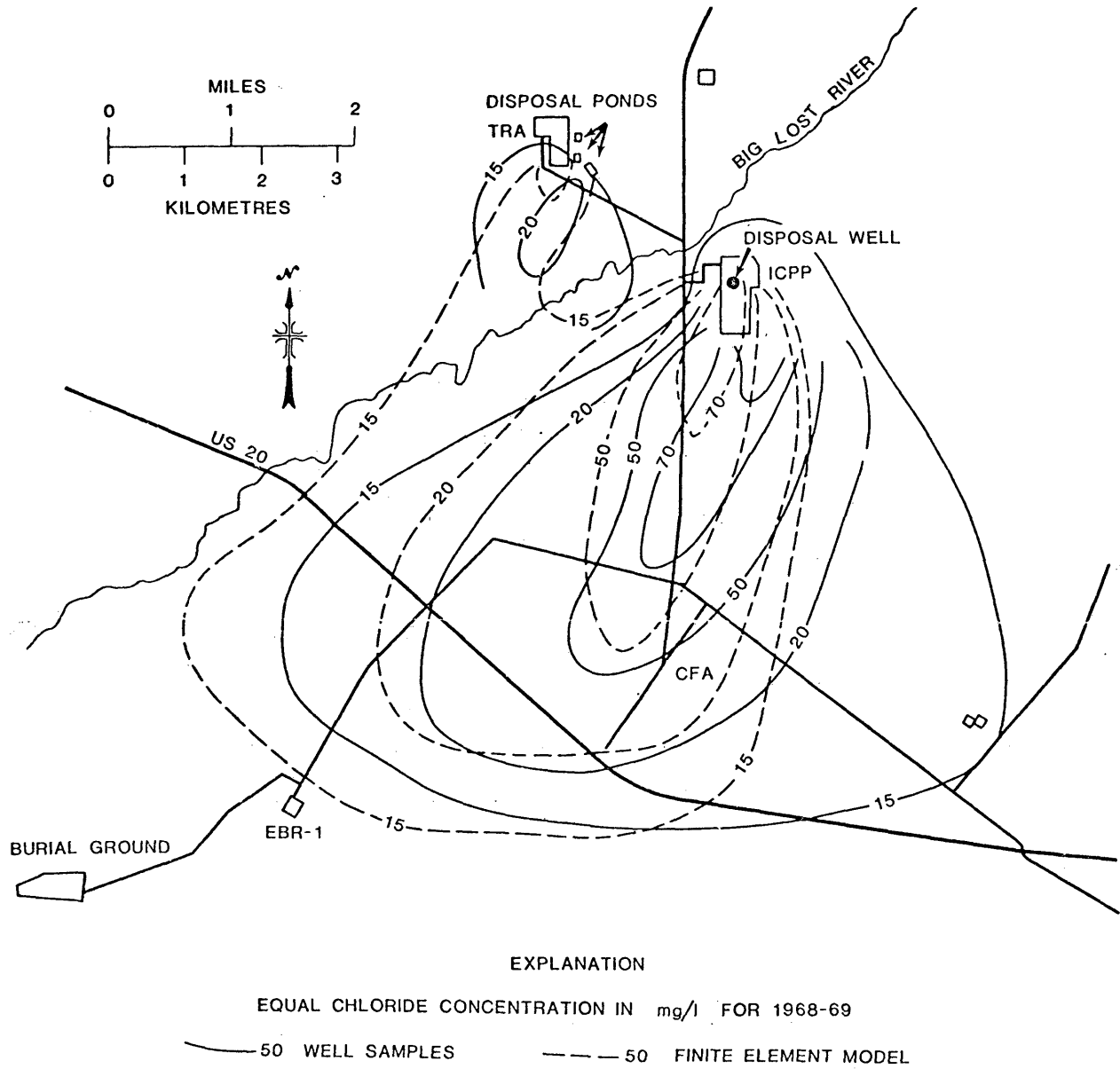


Figure 12.--Comparison of waste chloride plumes for 1968-1969 based on well sample data, and computer model.

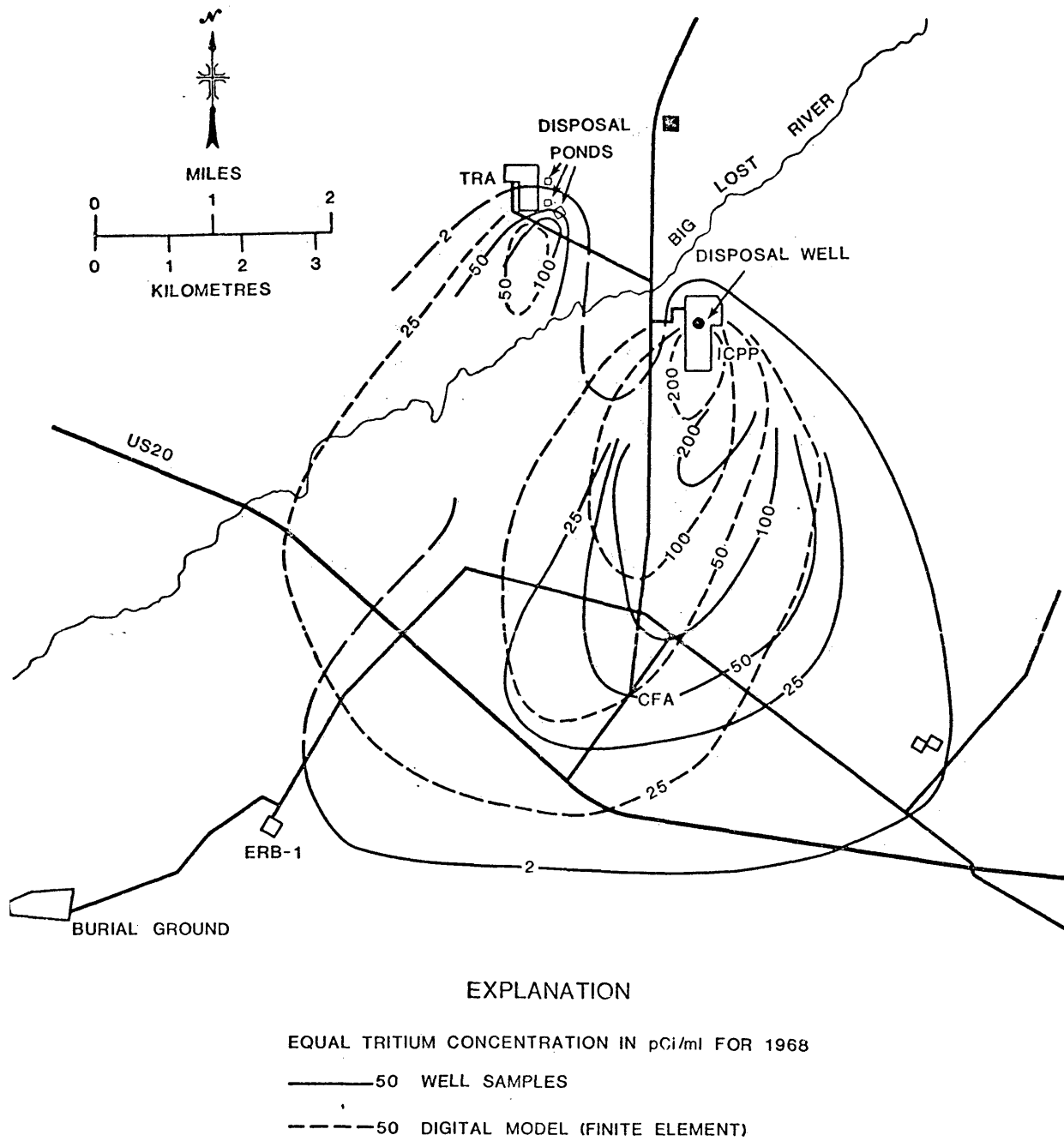


Figure 13.--Comparison of waste tritium plumes for 1968-1969 based on well sample data and computer model.

simulate a retarded movement of the ion. Figure 14 compares numerical calculated versus field observed data for the strontium contamination problem. Although it is difficult to make accurate comparisons for this small movement adequate comparisons of the field and computed data exist.

C. SUMMARY AND CONCLUSIONS

The use of Galerkin, finite-elements to solve the partial-differential equation that describe mass transport was shown to be a feasible approach. The use of linear and cubic basis functions, generated solutions to the convective-diffusion equation that were more accurate and less susceptible to oscillation than finite difference methods. Large scale two-dimensional problems can be solved with these methods, however excessive computer time results with the use of cubic basis functions. This is due to the structure of the coefficient matrix that represents the system of differential equations that must be solved. Linear basis functions form strongly diagonal matrices with no more than nine entries per row. Cubic basis functions form matrices with weak diagonals and as many as 36 elements per row. The use of SOR to solve the matrix formed with linear functions was very efficient. A more involved iterative technique was necessary to solve the matrix generated by the cubic functions. A large sample problem required a factor of 10 times longer CPU time to solve the problem using cubic functions rather than the linear functions. This time difference can

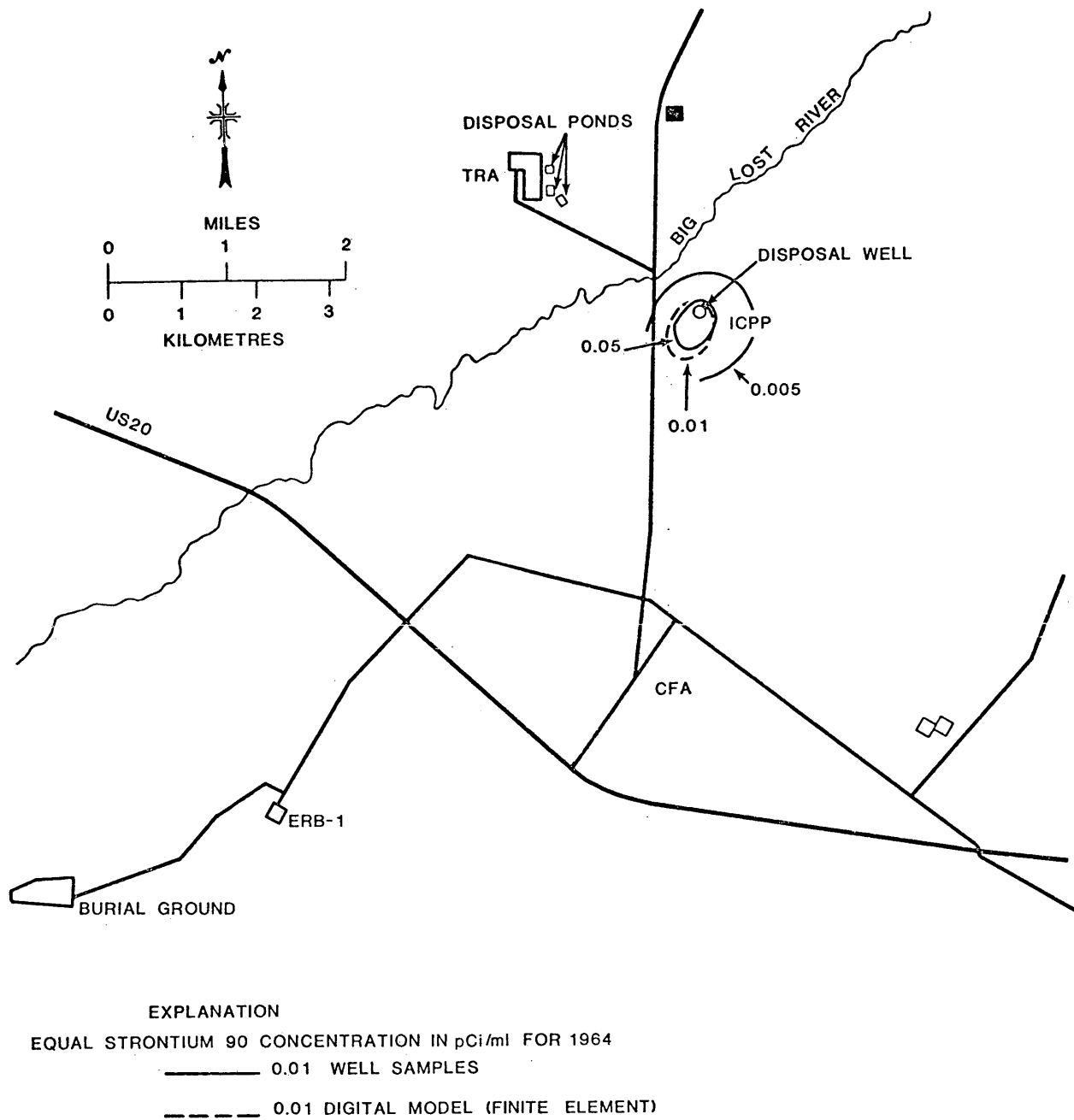


Figure 14.--Comparison of waste strontium-90 plumes for 1964 based on well sample data and computer model.

perhaps be lessened by using fewer number but larger sized elements for the cubic equations. The decrease in program sensitivity to parameter values such as areal changes in transmissivity and well locations makes this adjustment presently unattractive.

The model's practical use was demonstrated by solving field contamination problem. The concentration profiles of chloride, tritium and strontium-90 were simulated and compared to field data. The model successfully simulated solute transport for an unreactive conservative solute chloride, a solute with a first order irreversible rate reaction, radioactive decay and a solute with equilibrium controlled ion exchange.

The program incorporated generalized expressions to treat dispersion as a tensor with both longitudinal and transverse components.

Previous studies using finite-difference equations to solve these same sets of transport equations show the finite element methods to be superior for large grid spacings. For the models presented here the linear functions are superior to cubic functions for large two-dimensional problems. The development of more competitive linear equation solution schemes can of course change this.

As with any numerical technique further study may reveal shortcuts, improvements; etc. to enhance the technique's features. The most productive research in this area would perhaps be a search for more efficient matrix solution techniques for such sparse unsymmetric systems.

Appendix A includes a listing of the two finite element programs using cubic and linear basis functions. The matrix linear solution code TCHEB is documented by Manteuffel (1975). Table 3 and 4 lists the routines for the linear and cubic functions respectively. Table 5 is a list of selected program variables. Table 6 provides information for the data input format. As with any numerical code of this size a complete understanding of the program is prerequisite to efficient use.

LITERATURE CITED

- Aris, Rutherford, 1969, Elementary chemical reactor analysis: Englewood Cliffs, N. J., Prentice-Hall, 352 p.
- Bear, Jacob, 1972, Dynamics of fluids in porous media: New York, American Elsevier, 764 p.
- Bird, R. B., Stewart, W. E., and Lightfoot, E. N., 1966, Transport Phenomena: New York, John Wiley and Sons, 780 p.
- Boldt, G. H., 1967, Cation exchange equations in soil science: A review, Neth. Jour. Agr. Sci., v. 15, p. 81-103.
- Bredehoeft, J. D. and Pinder, G. F., 1973, Mass transport in flowing groundwater: WRR, v. 9, no. 1, p. 194-209.
- Brenner, Howard, 1962, The diffusion model of longitudinal mixing in beds of finite length. Numerical values: Chem. Eng. Sci., v. 17, p. 229-243.
- Carnahan, B., Luther, H. A., and Wilkes, J. O., 1969, Applied Numerical Methods: New York, John Wiley and Sons, 604 p.
- Cavendish, J. C., Price, H. S., and Varga, R. S., 1969, Galerkin methods for the numerical solution of boundary value problems: Soc. of Petroleum Eng. Jour., June, p. 204-220.
- Crandall, S. H., 1956, Engineering analysis, a survey of numerical Procedures: New York, McGraw Hill, p. 147-154.
- Doherty, P. C., 1972, Unsaturated darcian flow by the Galerkin method: Written commun., program no. C938, 70 p.

- Doughlas, J., Jr. and Dupont, T., 1970, Galerkin methods for parabolic equations: SIAM Jour. Numer. Analysis, v. 7, no. 4, p. 575-626.
- Finlayson, B. A., 1969, Applications of the method of weighted residuals and variational methods: British Chem. Eng., v. 14, no. 1, p. 53-57.
- Finlayson, B. A., 1972, The method of weighted residuals and variational principles: New York, Academic Press, 412 p.
- Finlayson, B. A. and Scriven, L. E., 1965, The method of weighted residuals and its relation to certain variational principles for the analysis of transport processes: Chem. Eng. Sci., v. 20, p. 395-404.
- Forray, M. J., 1968, Variational calculus in science and engineering: New York, McGraw Hill, 221 p.
- Garder, A. O., Peaceman, D. W., and Pozzi, A. L., 1964, Numerical calculation of multidimensional displacement by the method of characteristics: Soc. of Petroleum Eng. Jour., v. 4, no. 1, p. 26-36.
- Guymon, G. L., Scott, V. H., and Herrman, L. R., 1970, A general numerical solution of the two-dimensional diffusion-convection equation by the finite element method: Water Resources Research, v. 6, no. 6, p. 1611-1617.
- Harleman, D. R. F. and Rumer, R. R., Jr., 1963, Longitudinal and lateral dispersion in an isotropic porous medium: Jour. of Fluid Mechanics, v. 16, pt. 3, p. 385-394.

- Harlow, F. H. and Amsden, A. A., 1970, Fluid dynamics: an introductory text: LASL U. of Calif., 90 p.
- Helfferrich, Friedrich, 1962, Ion Exchange: New York, McGraw Hill, 624 p.
- Hughes, J. L. and Robson, S. G., 1973, Effects of waste percolation of groundwater in alluvium near Barstow, California in Underground waste management and artificial recharge: v. 1, p. 91-129.
- IBM Application Program - System 1360 Scientific Subroutine Package Version III Programmers Manual, Program Number 360A-CM-03X, 1968, 454 p.
- Keller, Herbert B., 1967, The numerical solution of parabolic partial differential equations, in Mathematical methods for digital computers, v. 1: New York, John Wiley & Sons, 293 p.
- Konikow, L. F., and Bredehoeft, J. D., 1974, Modeling flow and chemical quality changes in an irrigated stream-aquifer system: Water Resources Research, v. 10, no. 3, p. 546-562.
- Kreyszig, Erwin, 1962, Advanced engineering mathematics: New York, John Wiley and Sons, 856 p.
- Krylov, V. I., 1962, Approximate calculation of integrals; translated by Authur H. Stroud, New York, The Macmillian Co., 355 p. (337)
- Lantz, R. B., 1970a, Quantitative evaluation of numerical diffusion (truncation error), paper SPE 2811, presented at SPE second numerical simulation symposium, Dallas, Texas.

- Lantz, R. B.; 1960b, Rigorous calculation of miscible displacement using immiscible reservoir simulators: Soc. Petroleum Eng. Jour., June, p. 192-203.
- Levenspiel, Octave, 1967, Chemical reaction engineering: New York, John Wiley and Sons, 501 p.
- Longwell, P. A., 1966, Mechanics of fluid flow: New York, McGraw-Hill, 433 p.
- Manteuffel, T. A., 1975, An iterative method for solving nonsymmetric linear systems with dynamic estimation of parameters: Dept. of Computer Science, Univ. of Illinois, Urbana, Illinois, 185 p.
- Nalluswami, Marappagounder, 1971, Numerical simulation of general hydrodynamic dispersion in porous medium: Ph.D. dissertation, Colo. State Univ., Fort Collins, 138 p.
- Nalluswami, M., Logenbough, R. A., and Sunada, D. K., 1972, Finite-element method for the hydrodynamic dispersion equation with mixed partial derivatives: Water Resources Research, v. 8, no. 3, p. 1247-1250.
- Ogata, A. and Banks, R. B., 1961, A solution of the differential equation of longitudinal dispersion in porous media: USGS Prof. Paper 411-A, 7 p.
- Peaceman, D. W. and Rachford, H. H., Jr., 1962, Numerical calculation of multidimensional miscible displacement: Soc. of Petroleum Eng. Jour., v. 2, no. 4, p. 327-339.

- Pinder, George F., 1973, A Galerkin finite element simulation of ground-water contamination on Long Island, New York: Water Resources Research, v. 9, no. 6, p. 1657-1670.
- Pinder, G. F. and Cooper, H., 1970, A numerical technique for calculating the transient position of the salt water front: Water Resources Research, v. 6, no. 3, p. 875-882.
- Pinder, G. F., Frind, E. O., and Papadopoulos, S. S., 1973, Functional coefficients in the analysis of groundwater flow: Water Resources Research, v. 9, no. 1, p. 222-226.
- Price, H. S., Cavendish, J. C., and Varga, R. S., 1968, Numerical methods of higher order accuracy for diffusion-convection equations: Soc. of Petroleum Eng. Jour., v. 8, p. 293-303.
- Reddell, D. L. and Sunada, D. K., 1970, Numerical simulation of dispersion in ground-water aquifers: Colorado State Univ. Hydrology Paper 41, Fort Collins, Colo. 79 p.
- Remson, I., Hornberger, G. M., and Molz, F. J., 1971, Numerical methods in subsurface hydrology: New York, Wiley Interscience, 389 p.
- Robertson, J. B. M. and Barraclough, J. T., 1973, Radioactive and chemical-waste transport in groundwater at National Reactor Testing Station, Idaho: 20 year case history, in Underground waste management and artificial recharge: v. 1, p. 291-322.
- Robertson, J. B., 1974, Digital modeling of radioactive and chemical waste in the Snake River Plain Aquifer at the National Reactor Testing Station, Idaho, Springfield, Virginia, NTIS, 41 p.

- Robertson, J. B., Schoen, Robert and Barraclough, J. T., 1974, The influence of liquid waste disposal on the geochemistry of water at the National Reactor Testing Station, Idaho 1952-1970, Springfield, Virginia, NTIS, 231 p.
- Rubin, J. and James, R. V., 1973, Dispersion-affected transport of solutes in porous media; I The Galerkin method applied to equilibrium-controlled exchange in unidirectional, steady, saturated waterflow: Water Resources Research, v. 9, no. 5, p. 1332-1357.
- Schechter, R. S., 1967, The variational method in engineering, New York, McGraw Hill, 287 p.
- Scheidegger, A. E., 1961, General theory of dispersion in porous media: Jour. of Geophys. Research, v. 66, no. 10, p. 3273-3278.
- Shamir, u. Y. and Harleman, D. R. F., 1967, Numerical solutions for dispersion in porous mediums: Water Resources Research, v. 3, no. 2, p. 557-581.
- Smith, G. D., 1965, Numerical Solutions of partial differential equations: New York, Oxford University Press, 179 p.
- Snyder, L. J., Spriggs, T. W., and Stewart, W. E., 1964, Solution of the equations of change by Galerkin's method: Am. Inst. Chem. Eng. Jour., v. 10, no. 4, p. 535-540.
- Wylie, C. R., Jr., 1966, Advanced engineering mathematics: New York, McGraw-Hill, 813 p.
- Zienkewicz, O. C., 1971, The finite element method in engineering science, McGraw Hill, London, 521 p. (148).

APPENDIX A

Table 3.--List of routines used in Galerkin finite element program using linear basis functions.

Subroutine

MAING 7	Controls execution
PARLOD	Data input and initialization
ITERAT	Computes head distribution
OUTPT	Prints head distribution and computes mass balance for flor model
VELYG	Computes hydraulic gradients, velocities and dispersion coefficients
TRSTPG	Computes change in chemical concentration, computes mass balance and prints concentration.

Function

GAUSS5 (in TRSTPG)	Numerically integrates double integrals for linear basis functions
FCT (in TRSTPG)	Calculates the value of linear function over an interval
FCTD (in TRSTPG)	Calculates the derivative of the linear function over an interval

Table 4.--List of subroutines used in Galerkin finite element program using cubic basis functions.

MAING6	Controls execution
PARLOD	Data input and initialization
ITERAT	Computes head distribution
OUTPT	Prints head distribution and prints mass balance for flow model.
VELYG	Computes hydraulic gradients, velocities and dispersion coefficients
NODES	Checks to see which nodes have wells in them and computes more detailed head distribution around these nodes.
TRSPTC	Computes change in chemical concentration and prints concentration at selected intervals.
MATCF2	Calculates the matrix coefficients for the set of linear equations.
INTEG2	Calculates the value of the double integrals for the cubic basis functions.
RSIDE	Calculates the right hand known vector coefficients.
DIMEN	Passes dimensions to matrix solution subroutines
SCALE	Performs scaling operation on matrices that define linear equations.
RSCALE	Scales matrix that defines right hand known vector coefficients and unscales solution vector.
TCHEB	Iterative solution technique (see Manteuffel-reference)

Table 5.--Definition of selected program variables

AAQ	= Area of aquifer in model
ALPHAL	= BETA
AOPT	= Iteration parameters
AREA	= Area of one cell in finite-difference grid
BETA	= Longitudinal dispersivity of porous medium (ALPHAL)
CMAX	= Maximum initial or input concentration to elements
CONK	= A constant concentration for selected nodes (see NODE).
CSTOR	= Solute mass stored in system.
C, CN	= Chemical concentration.
CNRECH	= Concentration in recharging water
DDRW	= Drawdown
DELS	= Change in ground-water storage
DELTA	= Time increment for solute transport (SEC)
DELQ	= Rate of leakage across a confining layer or streambed
DERH	= Change in head with respect to time
DLTRAT	= Ratio of transverse to longitudinal dispersivity
EPS	= Permitted iteration error for SOR solute transport
FCTR	= Multiplication factor
FLMIN	= Mass entering modeled area during time step
FLMOT	= Mass leaving modeled area during time step
GRADX	= Hydraulic gradient in x-direction
GRADY	= Hydraulic gradient in y-direction
FLUX	= Net solute flux through system boundaries

HC = Head from column computation
HI = Head at end of time step
HMIN = Minimum iteration parameter
HR = Head from row computation
INT = Pumping period number
IPRNT = Print control for hydrographs
ITMAX = Maximum permitted number of iterations
KOUNT = Iteration number
N = Time step number
NCA = Number of aquifer nodes in model
NETFL = Total net solute flux
NI = NX
NJ = NJ
NITP = Number of iteration parameters
NMOX = Number of particle movements required to complete time step
NODE = Node identification code for solute transport model
NODEID = Node identification code for flow model
NPMP = Number of pumping periods
NPNT = Number of time steps between printouts for flow model
NREC = Number of pumping wells
NTIM = Number of time steps
NX = Number of nodes in x-direction (NI)
NY = Number of nodes in y-direction (NJ)
PARAM = Iteration parameter

PERM = Hydraulic conductivity (in ft/sec)

PINT = Pumping period in years

POROS = Effective porosity (POR)

PMP = Pumping well output (ft³/sec)

PUMP = Cumulative net pumpage (ft³/sec)

PYR = Pumping period in seconds

QI = Injection well input (ft³/sec)

QSTR = Cumulative change in volume of water in storage

RCMAS = Mass pumped out or recharged to aquifer

REC = Pumpage array; positive for withdrawal, negative for injection
(in ft³/sec)

RECH = Recharge array; negative for recharge, positive for E-T
(in ft/sec)

S = Storage coefficient (or specific yield)

SLEAK = Rate of leakage through confining layer or streambed

STORM = Change in total mass in storage (by summation)

SUMIO = Change in total mass in storage (from inflow - outflow)

SUMT = Total elapsed time (in sec)

SYMBOL = Preselected symbols for concentration printout
(11 required, highest to lowest concentration)

THCK = Saturated thickness of aquifer

TIM = Length of time step

TIMEI = Time increment between printout for solute model

TIMD = Elapsed time in years

TIMM = Elapsed time in minutes

TIMX = Time step multiplier

TINIT = Size of initial time step (in sec)

TITLE = Problem description

TMAX = Maximum time for solute transport run

TMOBS = Elapsed times for observation point records

TMRX = Transmissivity coefficient

TMWL = Computed heads at observation points

TOL = Convergence criteria (ADIP)

TOTLQ = Cumulative net leakage through confining layer on streambed

VPRM = Leakage factor for confining layer or streambed (vertical hydraulic conductivity/thickness)

VX = Velocity in x-direction at a node

VY = Velocity in y-direction at a node

WFAC = Iteration relaxation factor (not used for F.D. model)

WFLUXI = Solute mass injected into system through wells

WFLUXO = Solute mass leaving system through wells

WT = Water-table elevation or head in stream or source bed

XDEL = Grid spacing in x-direction (in ft) (DELX)

YDEL = Grid spacing in y-direction (in ft) (DELY)

XKD = Ion exchange distribution coefficient

XKIN = Kinetic reaction rate constant

XN = Order of reaction rate

XRHO = Soil weight to free volume ratio

Table 6.--Data Input Formats

<u>Card</u>	<u>Columns</u>	<u>Format</u>	<u>Variables</u>	<u>Definition</u>
1	1 -80	10A8	TITLE	Description of problem.
2	1 - 4	14	NTIM	Number of time steps in a pumping period
	5 - 8	14	NPMP	Number of pumping periods.
	9 -12	14	NX	Number of nodes in x-direction.
	13-16	14	NY	Number of nodes in y-direction.
	17-20	14	NPNT	Time-step interval for printing output data.
	21-24	14	NITP	Number of iteration parameters.
	25-28	14	ITMAX	Maximum allowable number of iterations in ADIP.
	29-32	14	NREC	Number of pumping or injection wells to be specified in a following data set.
	33-37	G5.0	TMAX	Maximum time for solute transport.
	38-42	G5.0	TIMEI	Time increment between printout for solute transport.
	43-47	G5.0	CONK	A constant concentration for selected nodes (see NODE).
	48-52	G5.0	WFAC	Iteration relaxation factor.
	53-57	G5.0	EPS	Permitted iteration error.
	58-62	G5.0	TCK	Aquifer thickness (omit if given for each node).
	63-67	G5.0	TTT	Aquifer transmissivity (omit if given for each node).
	68-72	G5.0	WTE	Initial water table elevation (omit if given at each node).
	73-77	G5.0	CONI	Initial concentration (set neg. value if given at each node).
	78	I1	NNODE	Gives the value of NNODE to second outer row & column If NNODE equals 1 see Def. of WT (Data set 5).

Data Input Formats - Continued

<u>Case</u>	<u>Columns</u>	<u>Format</u>	<u>Variables</u>	<u>Definition</u>
	79	I1	IIX	9FD nodes, set equal to 1 if residuals are not printed out for cubic program
	80	I1	NPNCHV	Punches out velocities for plotter output if = 1.
	1 - 5	G5.0	PINT	Pumping period in years.
	6 -10	G5.0	TOL	Convergence criteria in ADIP.
3	11-15	G5.0	POROS	Effective porosity.
	16-20	G5.0	BETA	Characteristic length in feet.
	21-25	G5.0	S	Storage coefficient (set S = 0 for steady flow problem.
	26-30	G5.0	TIMX	Time increment multiplier.
	31-35	G5.0	TINIT	Size of initial time step in seconds.
	36-40	G5.0	XDEL	Width of finite-difference cell in x-direction, in feet.
	41-45	G5.0	YDEL	Width of finite-difference cell in y-direction, in feet.
	46-50	G5.0	DLTRAT	Ratio of transverse to longitudinal dispersivity.
	51-55	G5.0	XKD	Ion exchange distribution coefficient.
	56-60	G5.0	XKIN	Kinetic reaction rate constant.
	61-65	G5.0	XN	Order of reaction rate.
	66-70	G5.0	XRHO	Soil weight to free volume ratio.
	71-75	G5.0	DELT	Time increment.
4	1 -22	I1A2	SYMBOL	Preselected symbols for concentration printout (I1 required, highest to lowest concentration).

Data Input Formats - Continued

<u>Data Set</u>	<u>Number of Cards</u>	<u>Format</u>	<u>Variables</u>	<u>Definition</u>
1	Value of NREC	2I2, 2G8.2	IX, IY, REC, CNRECH	x and y coordinates of pumping (+) or injection (-) wells, rate in cfs, and if an injection well, the concentration of injected water. This data set is eliminated if NREC = 0.
2	Value of . . . NY	20G4.1	VPRM	Array for temporary storage of transmissivity data, in ft ² /sec. (Omit if TTT given)
3	Value of	20G3.0	THCK	Saturated thickness of aquifer, in feet. (Omit if TCK given).
4	Value of	80I1	NODEID	Node identification matrix. (=1 for constant head node). Omit if NNODE given.
5	Value of	20G4.0	WT	Initial water-table or potentiometric elevation, in feet. (Omit if WTE given). If NNODE is given, read 4 cards (Top row, bottom row, left column, right column)
6	Value of	80I1	NODE	Chemical node ID matrix.
7	Value of	20G4.0	C	Initial concentrations matrix. (Omit if CONI given)

APPENDIX B

```

C *****
C
C THIS PROGRAM CONTROLS THE EXECUTION
C *****
C
C SOLUTE TRANSPORT, CHEMICAL REACTION AND DISPERSION IN A POROUS
C MEDIUM -- NUMERICAL SOLUTION-- FLOW EQUATIONS SOLVED BY IADI
C BY J. BREDEHOEFT - 1973 MODIFIED BY D. GROVE - 1976. SOLUTE
C TRANSPORT AND REACTION EQUATION SOLVED BY GLFRKIN FINITE ELEMENT
C METHOD USING LINEAR BASIS FUNCTIONS BY D. GROVE -1976.
C *****
C
C DOUBLE PRECISION DMIN1,DEXP,DLOG,DABS
C REAL *8TMPX,VPRM,HI,HR,HC,HK,WT,REC,RECH,TIM,AOPT,TITLE
C REAL *8XDEL,YDEL,S,AREA,SUMT,PHO,PARAM,TEST,TOL,PINT,HMIN,PYR
C REAL *8 TINT,ALPHA1,ANITP,XSM,YSM
C
C DIMENSION NODE(17,20),DXX(17,20),EYY(17,20),DXY(17,20),
C 1 DYY(17,20),QI(17,20),
C 2 SYMBOL(11),C(17,20),PMP(17,20),VXM(50),VXP(50),VYM(50),
C 3 VYP(50),NWELL(17,20),NODEID(17,20),
C 4 THCK(17,20),PERM(17,20),TMWL(17,20), TMRX(17,20,2)
C DIMENSION VPRM(17,20),HI(17,20),HR(17,20),HC(17,20),HK(17,20),
C 1 WT(17,20),REC(17,20),RECH(17,20),TIM(100),AOPT(7),TITLE(10),
C 2 VX(17,20),VY(17,20),CNRECH(17,20),VYX(17,20),VXY(17,20)
C *****
C LOAD DATA
C NX=17
C NY=20
C 10 CONTINUE
C CALL PARLOU(NODE,XKD,XKIN,XN,XRHO,DXX,DYY,DXY,DYX,QI,SYMBOL,
C 1 DELT,ITMAX,TIMEI,C,CONK,WFAC,EPS,PMP,VXM,VXP,VYM,VYP,
C 2 NWELL,NTIM,NPMP,NPNT,NITP,N,NX,NY,NP,NREC,INT,NNX,NNY,
C 3 ITMAX,IPRNT,NODEID,
C 4 THCK,PERM, TMRX,VPRM,HI,HR,HC,HK,WT,REC,RECH,
C 5 TIM,AOPT,TITLE,XDEL,YDEL,S,AREA,SUMT,PHO,PARAM,TEST,TOL,
C 6 PINT,HMIN,PYR,VX,VY,CNRECH,PORUS,SUMTCH,BETA,TIMV,STORM,RCMAS,
C 7 FLMIN,FLMOT,SUMIO,ULTRAT,VYX,VXY,TOTLO)
C *****
C START COMPUTATIONS
C COMPUTE NPMP PUMPING PERIODS
C DO 100 NINT=1,NPMP
C INT=NINT
C
C COMPUTE NTIM TIME STEPS
C DO 80 NS=1,NTIM
C N=NS
C IPRNT=0
C LOAD NEW DELTA T
C TINT=SUMT-PYR*(INT-1)
C TDEL=DMIN1(TIM(N),PYR-TINT)
C SUMT=SUMT+TDEL
C TIM(N)=TDEL
C REMN=MOD(N,NPNT)
C *****
C 20 CONTINUE
C COMPUTE HEAD DISTRIBUTION
C CALL ITERAT(NITP,NX,NY,ITMAX,THCK,TMRX,VPRM,HR,HC,HK,WT,REC,
C 1 RECH,TIM,AOPT,XDEL,YDEL,S,AREA,TOL,TOTLO,N)
C 30 CONTINUE

```

```

      IF (DEMN.EQ.0.0) CALL OUTPT(N,NX,NY,THCK,VPRM,HI,HK,WT,
1 REC,AREA,SUMT,TUTLO,S)
40 CONTINUE
50 CONTINUE
C   COMPUTE VELOCITY AND DISPERSION COEFFICIENTS
      CALL VELYG(N,NX,NY,DXX,DYY,DXV,DYX,PERM,HK,XDEL,
1 YDEL,VX,VY,POROS,BETA,DLTRAT,NODE,NWELL,VXY,VYX,
2 VYM,VYP,VXM,VXP)
60 CONTINUE
C   COMPUTE CHANGE IN CONCENTRATION
      CALL TRSPTG(NODE,XKD,XKIN,XN,ARHO,DXX,DYY,DXV,DYX,QI,
1 SYMBOL,DELT,IMAX,TIME I,C,CONK,PMP,NX,NY,NNX,NNY,THCK,PECH,
2 XDEL,YDEL,VX,VY,CNPECH,POROS,WFAC,EPS,VXM,VXP,VYM,VYP,NWELL)
C   *****
C   OUTPUT ROUTINES
70 IF (SUMT.GE.(PYR*INT)) GO TO 90
80 CONTINUE
C   *****
C   SUMMARY OUTPUT
90 CONTINUE
      IPRNT=1
C   CALL OUTPT
100 CONTINUE
C   *****
      STOP
C   *****
      END
//

```

```

C *****
C
C THIS SUBROUTINE INPUTS AND INITIALIZES DATA
C
SUBROUTINE PARLOD(NODE,XKD,XKIN,XN,XRHO,DXX,DYY,DXY,DYX,QI,
1 SYMBOL,DELT,TMAX,TIMEI,C,CONK,WFAC,EPS,PMP,VXM,VXP,VYM,VYP,
2 NWELL, NTIM,NPMP,NPNT,NITP,N,NX,NY,NP,NREC,INT,NNX,NNY,
3 ITMAX,IPRNT,NODEID,
4 THCK,PERM, TMRX,VPRM,HI,HR,HC,HK,WT,REC,RECH,
5 TIM, AOPT, TITLE, XDEL, YDEL, S, AREA, SUMT, RHO, PARAM, TEST, TOL,
6 PINT, HMIN, PYR, VX, VY, CNRECH, POROS, SUMTCH, BETA, TIMV, STORM, RCMAS,
7 FLMIN, FLMOT, SUMIO, DLTPAT, VYX, VXY, TOTLO)
C ***** B 20
DOUBLE PRECISION DMINI,DEXP,DLOG,DABS B 30
REAL *8TMRX,VPRM,HI,HR,HC,HK,WT,REC,RECH,TIM, AOPT, TITLE
REAL *8XDEL,YDEL,S,AREA,SUMT,RHO,PARAM,TEST,TOL,PINT,HMIN,PYR B 50
REAL *8FCTR,TIMX,TINIT,PIES,YNS,XNS,PAT,HMX,HMY B 60
REAL *8TINT,ALPHA1,ANITP B 70
DIMENSION NODE(NX,NY),DXX(NX,NY),DYY(NX,NY),DXY(NX,NY),
1 DY(NX,NY),QI(NX,NY),SYMBOL(11),C(NX,NY),PMP(NX,NY),
1 VXM(50),VXP(50),VYM(50),VYP(50),NWELL(NX,NY)
DIMENSION NODEID(NX,NY),IXOBS(5),IYOBS(5),THCK(NX,NY),
1 PERM(NX,NY),TMR(5,50),TMOBS(50),TMPX(NX,NY,2),
2 VPRM(NX,NY),HI(NX,NY),HR(NX,NY),HC(NX,NY),HK(NX,NY),WT(NX,NY),
3 REC(NX,NY),RECH(NX,NY),TIM(100),AOPT(NX),TITLE(10),
4 VX(NX,NY),VY(NX,NY),CNRECH(NX,NY),VYX(NX,NY),VXY(NX,NY)
C ***** B 230
READ(5,490) TITLE B 240
WRITE(6,500) TITLE B 250
C ***** B 260
C INITIALIZE TEST AND CONTROL VARIABLES B 270
TEST=0.0 B 280
TOTLO=0.0 B 290
SUMT=0.0 B 300
SUMTCH=0.0 B 310
INT=0 B 320
IPRNT=0 B 330
KGEN=1 B 340
NCA=0 B 350
N=0 B 360
IMOV=0 B 370
NMOV=0 B 380
C ***** B 390
C LOAD CONTROL PARAMETERS B 400
READ(5,510) NTIM,NPMP,NX,NY,NPNT,NITP,ITMAX,NREC,TMAX,T
1 TIMEI,CONK,WFAC,EPS,TCK,ITT,WT,CONI,NNODE,IIX
READ(5,580) PINT,TOL,POROS,BETA,S,TIMX,TINIT,XDEL,YDEL,DLTRAT B 430
1,XKD,XKIN,XN,XRHO,DELT B 440
READ(5,630) SYMBOL B 450
PYR=PINT*86400.0*365.25 B 460
NNX=NX-1 B 470
NNY=NY-1 B 480
C B 500
WRITE(6,520) B 510
WRITE(6,530) NX,NY,XDEL,YDEL B 520
WRITE(6,540) NTIM,NPMP,PINT,TIMX,TINIT,DELT,TMAX,TIMEI B 530
WRITE(6,550) S,POROS,BETA,DLTRAT,XKD,XKIN,XN,XRHO,EPS,WFAC B 540
WRITE(6,560) NITP,TOL,ITMAX
WRITE(6,570) NPNT,NREC
C ***** B 570
C LIST TIME INCREMENTS B 580

```

```

DO 10 J=1,NTIM
TIM(J)=0.0
10 CONTINUE
TIM(1)=TINIT
IF (S.EQ.0.0) GO TO 30
DO 20 K=2,NTIM
TIM(K)=TIMX*TIM(K-1)
20 CONTINUE
GO TO 40
30 TIM(1)=PYR
C
40 WRITE (6,250)
WRITE (6,260) TIM
*****
C INITIALIZE MATRICES
C
DO 50 IY=1,NY
DO 60 IX=1,NX
VPRM(IX,IY)=0.0
PERM(IX,IY)=0.0
THCM(IX,IY)=0.0
RECH(IX,IY)=0.0
CNRECH(IX,IY)=0.0
REC(IX,IY)=0.0
NOUFD(IX,IY)=0
TMRX(IX,IY,1)=0.0
TMRX(IX,IY,2)=0.0
HI(IX,IY)=0.0
HR(IX,IY)=0.0
HC(IX,IY)=0.0
HK(IX,IY)=0.0
WT(IX,IY)=0.0
VX(IX,IY)=0.0
VY(IX,IY)=0.0
VXY(IX,IY)=0.0
VYX(IX,IY)=0.0
DXX(IX,IY)=0.0
DXY(IX,IY)=0.0
DYY(IX,IY)=0.0
DYX(IX,IY)=0.0
QI(IX,IY)=0.0
C(IX,IY)=0.0
PMP(IX,IY)=0.0
50 CONTINUE
*****
C READ PUMPAGE DATA -- (X-Y COORDINATES AND RATE IN C.F.S.)
C SIGNS ---- WITHDRAWAL = POS.; INJECTION = NEG.
C IF INJECTION WELL. ALSO READ CONCENTRATION OF INJECTED WATER
IF (NREC.LE.0) GO TO 120
WRITE (6,610)
DO 110 I=1,NREC
READ (5,480) IX,IY,FCTR,CNREC
NWELL(IX,IY)=I
IF (FCTR.LT.0.0) CNRECH(IX,IY)=CNREC
REC(IX,IY)=FCTR
IF (FCTR.GT.0.0) GO TO 90
QI(IX,IY)=-FCTR
GO 0 100
90 QI(IX,IY)=0.0
PMP(IX,IY)=FCTR
100 WRITE (6,620) IX,IY,REC(IX,IY),CNRECH(IX,IY),QI(IX,IY)
110 CONTINUE

```

```

B 590
B 600
B 610
B 620
B 630
B 640
B 650
B 660
B 670
B 680
B 690
B 700
B 710
B 720
B 730
B 740
B 750
B 760
B 770
B 780
B 790
B 800
B 810
B 820
B 830
B 840
B 850
B 860
B 870
B 880
B 890
B 900
B 910
B 920
B 930
B 940
B 950
B 960
B 970
B 980
B 990
B1000
B1150
B1160
B1170
B1180
B1190
B1200
B1210
B1220
B1230
B1240
B1250
B1260
B1270
B1280
B1290
B1300
B1310

```

```

120 CONTINUE
C *****
  AREA=XDEL*YDEL
C
  WRITE (6,460) AREA
  WRITE (6,360)
  WRITE (6,370) XDEL
  WRITE (6,370) YDEL
C *****
C READ TRANSMISSIVITY IN FT**2/SEC INTO VPRM MATRIX
C FCTR = TRANSMISSIVITY MULTIPLIER ---> FT**2/SEC
C
  FCTR=1.0
  IF(TTT.LT..1E-10)GO TO 122
  DO 123 IY=1,NY
  DO 123 IX=1,NX
  VPRM(IX,IY)=TTT*FCTR
  IF(IX.EQ.1.OR.IX.EQ.NX)VPRM(IX,IY)=0.0
  IF(IY.EQ.1.OR.IY.EQ.NY)VPRM(IX,IY)=0.0
123 CONTINUE
  GO TO 124
122 CONTINUE
C
  DO 121 IY=1,NY
121 READ (5,320) (VPRM(IX,IY),IX=1,NX)
  DO 130 IX=1,NX
  DO 130 IY=1,NY
  VPRM(IX,IY)=VPRM(IX,IY)*FCTR
  IF (IX.EQ.1.OR.IX.EQ.NX) VPRM(IX,IY)=0.0
  IF (IY.EQ.1.OR.IY.EQ.NY) VPRM(IX,IY)=0.0
130 CONTINUE
124 CONTINUE
C
  WRITE (6,300)
  DO 131 IY=1,NY
131 WRITE (6,290) (VPRM(IX,IY),IX=1,NX)
C *****
C SET UP COEFFICIENT MATRIX
C AVERAGE TRANSMISSIVITY
C BLOCK CENTERED GRID
C GEOMETRIC MEAN
C
  PIES=3.1415927*3.1415927/2.0
  YNS=NY*NY
  XNS=NX*NX
  HMIN=2.0
  DO 140 IY=2,NNY
  DO 140 IX=2,NNX
  IF (VPRM(IX,IY).EQ.0.0) GO TO 140
  TMRX(IX,IY,1)=2.0*VPRM(IX,IY)*VPRM(IX+1,IY)/(VPRM(IX,IY)*XDEL+VPRM
1(IX+1,IY)*XDEL)
  TMRX(IX,IY,2)=2.0*VPRM(IX,IY)*VPRM(IX,IY+1)/(VPRM(IX,IY)*YDEL+VPRM
1(IX,IY+1)*YDEL)
C
C ADJUST COEFFICIENT FOR ANISOTROPY
  FCTR=1.0
C
  TMRX(IX,IY,1)=TMRX(IX,IY,1)*FCTR
  TMRX(IX,IY,2)=TMRX(IX,IY,2)*FCTR
C *****
C COMPUTE MINIMUM ITERATION PARAMETER

```

```

B1320
B1330
B1340
B1350
B1360
B1370
B1380
B1390
B1400
B1410
B1420
B1430
B1440
B1450
B1470
B1480
B1490
B1500
B1510
B1520
B1530
B1540
B1560
B1570
B1580
B1590
B1600
B1610
B1620
B1630
B1640
B1650
B1660
B1670
B1680
B1690
B1700
B1710
B1720
B1730
B1740
B1750
B1760
B1770
B1780
B1790
B1800

```

```

C      HMIN:  MINIMUM ITERATION PARAMETER                                B1810
C
      IF (TMRX(IX,IY,1).EQ.0.0) GO TO 140                               B1820
      IF (TMRX(IX,IY,2).EQ.0.0) GO TO 140                               B1830
      RAT=TMRX(IX,IY,1)*YDEL/(TMRX(IX,IY,2)*XDEL)                       B1840
      HMX=PIES/(XNS*(1.0+RAT))                                           B1850
      HMY=PIES/(YNS*(1.0+(1.0/RAT)))                                     B1860
      IF (HMX.LT.HMIN) HMIN=HMX                                          B1870
      IF (HMY.LT.HMIN) HMIN=HMY                                          B1880
      IF (HMY.LT.HMIN) HMIN=HMY                                          B1890
140  CONTINUE                                                            B1900
C
C      *WRITE (6,23)                                                    B1910
C      WRITE (6,22) (((TMRX(IX,IY,IL),IX=1,NX),IY=1,NY),IL=1,2)       B1920
C      *****                                                           B1930
C      READ AQUIFER THICKNESS                                           B1940
C      IF(TCK.LT..1E-10)GO TO 143                                       B1950
      DO 144 IY=1,NY
      DO 144 IX=1,NX
144  THCK(IX,IY)=TCK
      GO TO 145
143  CONTINUE
      DO 141 IY=1,NY
141  READ (5,310) (THCK(IX,IY),IX=1,NX)
C
C      145 CONTINUE                                                    B1970
      DO 146 IX=1,NX
      DO 146 IY=1,NY
      IF (IX.EQ.1.OR.IY.EQ.1) THCK(IX,IY)=0.0
      IF (IX.EQ.NX.OR.IY.EQ.NY) THCK(IX,IY)=0.0
146  CONTINUE
      WRITE (6,280)
      DO 142 IY=1,NY
142  WRITE (6,270) (THCK(IX,IY),IX=1,NX)
C      *****                                                           B1980
C      COMPUTE PERMEABILITY FROM TRANSMISSIVITY                         B2000
C      COUNT NO. OF CELLS IN AQUIFER                                    B2010
C
C      DO 150 IX=1,NX                                                    B2020
C      DO 150 IY=1,NY                                                    B2030
C      IF (THCK(IX,IY).EQ.0.0) GO TO 150                                B2040
C      PERM(IX,IY)=VPRM(IX,IY)/THCK(IX,IY)                             B2050
C      NCA=NCA+1                                                         B2060
C      VPRM(IX,IY)=0.0                                                  B2070
150  VPRM(IX,IY)=0.0                                                    B2080
      AAQ=NCA*AREA                                                       B2090
C
C      WRITE (6,380)                                                    B2100
C      DO 151 IY=1,NY                                                    B2110
151  WRITE (6,410) (PERM(IX,IY),IX=1,NX)                                B2120
      WRITE (6,390) NCA,AAQ
C      *****                                                           B2140
C      READ NODE IDENTIFICATION CARDS + SET VERTICAL PERMEABILITY..... B2150
C
C      IF (NNODE.LT.1) GO TO 153                                         B2160
      DO 155 IX=2,NNX
      NODEID(IX,2)=1
155  NODEID(IX,NNY)=1
      DO 156 IY=2,NNY
      NODEID(2,IY)=1
156  NODEID(NNX,IY)=1
      GO TO 154
153  CONTINUE
2    NODE IDENTIFICATION CODE:  1 = CONSTANT HEAD BOUNDARY           B2170

```



```

C      READ IN INITIAL CHEMICAL CONCENTRATION                                B2530
      IF (CONI.LT.-.1E-20) GO TO 175
      DO 176 IY=1,NY
      DO 176 IX=1,NX
176    C(IX,IY)=CONI
      GO TO 177
175    CONTINUE
      DO 173 IY=1,NY
173    READ (5,431) (C(IX,IY),IX=1,NX)
177    CONTINUE
      WRITE(6,220)
      DO 174 IY=1,NY
174    WRITE (6,221) (C(IX,IY),IX=1,NX)
      CALL OUTPT(N,NX,NY,THCK,VPRM,HI,HK,WT,REC,AREA,SUMT,TOTLQ,S)
C      *****
C      COMPUTE ITERATION PARAMETERS                                          B2570
C      NITP NUMBER OF ITERATION PARAMETERS                                  B2580
      DO 180 ID=1,NX                                                         B2590
      AOPT(ID)=0.0                                                           B2600
180    CONTINUE                                                             B2610
      ANITP=NITP-1                                                           B2620
      IF (S.EQ.0.0) GO TO 190                                               B2630
      ALPHA=DEXP(DLOG(2.0/HMIN)/ANITP)                                     B2640
      GO TO 200                                                               B2650
190    ALPHA=DEXP(DLOG(1.0/HMIN)/ANITP)                                    B2660
200    AOPT(1)=HMIN                                                         B2670
      DO 210 IP=2,NITP                                                       B2680
      AOPT(IP)=AOPT(IP-1)*ALPHA                                             B2690
210    CONTINUE                                                             B2700
C      WRITE (6,230)                                                         B2710
      WRITE (6,240) AOPT                                                     B2720
C      RETURN                                                                 B2730
C      * * * * *                                                             B2740
C      * * * * *                                                             B2750
C      * * * * *                                                             B2760
C      * * * * *                                                             B2770
C      * * * * *                                                             B2780
C      * * * * *                                                             B2790
220  FORMAT (1H1,5X,'INITIAL CHEMICAL CONCENTRATION '///)
221  FORMAT (1H ,20F6.0)
230  FORMAT (1H1,20HITERATION PARAMETERS)                                  B2810
240  FORMAT (3H ,1G20.6)                                                  B2820
250  FORMAT (1H1,14HTIME INTERVALS)                                       B2830
260  FORMAT (3H ,10G12.5)                                                  B2840
270  FORMAT (3H ,20F5.1)                                                  B2850
280  FORMAT (1H1,17HAQUIFER THICKNESS)                                       B2860
290  FORMAT (3H ,20F5.2)                                                  B2870
300  FORMAT (1H1,30HTRANSMISSIVITY MAP (FT*FT/SEC))                       B2880
310  FORMAT (20G3.0)
320  FORMAT (20G4.1)                                                       B2900
330  FORMAT (1H1,'NODE IDENTIFICATION MAP'///)                             B2910
331  FORMAT (1H1,'CHEMICAL NODE IDENTIFICATION MAP'///)
340  FORMAT (1H ,40I3)
350  FORMAT (1H1,44HVERTICAL PERMEABILITY/THICKNESS(FT/(FT*SEC)))         B2930
360  FORMAT (1H0,12H X-Y SPACING)                                           B2940
370  FORMAT (3H ,10G12.5)                                                  B2950
380  FORMAT (1H1,24HPERMEABILTY MAP (FT/SEC))                               B2960
390  FORMAT (1H0,///10X,'NO. OF FINITE-DIFFERENCE CELLS IN AQUIFER = '   B2970
      1',14//10X,'AREA OF AQUIFER IN MODEL = ',G12.5,' SQ. FT.')
```

```

430 FORMAT (20G4.0)
431 FORMAT(20G4.0)
440 FORMAT (1H1,11HWATER TABLE)
450 FORMAT (1H ,20FS.0)
460 FORMAT (1H0,10X,'AREA = ',G12.4)
470 FORMAT (2I2)
460 FORMAT (2I2,2G8.2)
490 FORMAT (10A8)
500 FORMAT (1H1,10A8////)
510 FORMAT (8I4,9G5.0,3I1)
520 FORMAT (1H0,21X,'I N P U T   D A T A'////)
530 FORMAT (1H0,23X,'GRID DESCRIPTORS'//13X,'NX   (NUMBER OF ROWS)',5
1X,'=',1,16/13X,'NY   (NUMBER OF COLUMNS) = ',1,14/13X,'XDEL (X-DIS
2TANCE IN FEET) = ',F7.1/13X,'YDEL (Y-DISTANCE IN FEET) = ',F7.1//
3)
540 FORMAT (1H0,21X,'TIME PARAMETERS'//12X,' NTIM (MAX. NO. OF TIME
1 STEPS)',7X,'= ',1,15/13X,'NPMP (NO. OF PUMPING PERIODS)',7X,'= '
2,16/13X,'PINT (PUMPING PERIOD IN YEARS)',6X,'=',F9.1/11X,' TIMX
3 (TIME INCREMENT MULTIPLIER) = ',F9.2/13X,'TINIT (INITIAL TI
4ME STEP IN SEC.) = ',F8.0/13X,'DELT',4X,'TIME INCREMENT FOR CHEM
5',7X,'=',5X,E9.2/13X,'TMAX',4X,'MAX TIME FOR CHEM RUN',9X,'=',E14.
66/13X,'TIME1',3X,'INCREMENT BETWEEN CHEM PRINT = ',E9.2//)
550 FORMAT (1H0,14X,'HYDROLOGIC AND CHEMICAL PARAMETERS'//13X,'S',7X,'
1(STORAGE COEFFICIENT)',7X,'=',5X,F9.6/13X,'POROS (EFFECTIVE PORO
2SITY)',8X,'= ',F8.2/13X,'BETA (CHARACTERISTIC LENGTH) = '
3,F7.1/13X,'DLTRAT (RATIO OF TRANSVERSE TO',21X,'LONGITUDINAL DISP
4ERSIVITY) = ',F9.2/13X,'XND',5X,'DISTRIBUTION COEFF',10X,'=',5X,F
59.6/13X,'XKIN',4X,'KINETIC RATE CONSTANT',7X,'=',5X,F9.6/13X,'XN',
66X,'ORDER OF REACTION',11X,'=',5X,F9.6/13X,'XRHO',4X,'SOIL WEIGHT
7TO FREE VOL RATIO = ',5X,F9.6/13X,'EPS',5X,'ITERATION ERROR',13X,'=
8',5X,E14.3/13X,'WFAC',4X,'RELAXATION COEFFICIENT',6X,'=',5X,F9.6)
560 FORMAT (1H0,21X,'EXECUTION PARAMETERS'//13X,'NITP (NO. OF ITERAT
1ION PARAMETERS) = ',1,14/13X,'TOL (CONVERGENCE CRITERIA - ADIP) =
2 ',F9.4/13X,'ITMAX (MAX.NO.OF ITERATIONS - ADIP) = ',1,14//)
570 FORMAT (1H0,23X,'PROGRAM OPTIONS'//13X,'NPNT (PRINT CONTROL IND
1EX) = ',1,14
2/13X,'NREC (NO. OF PUMPING WELLS) = ',1,15//)
580 FORMAT (16G5.0)
590 FORMAT (1H-,10X,'LOCATION OF OBSERVATION WELLS'//17X,'NO.',5X,'X',
15X,'Y'//)
600 FORMAT (1H ,16X,I2,5X,I2,4X,I2)
610 FORMAT (1H-,10X,'LOCATION OF PUMPING WELLS'//10X,'X Y RATE(
1IN CFS) CONC. INJECTION(IN CFS) //)
620 FORMAT (1H ,6X,2I4,3X,F7.2,5X,F7.1,5X,F7.2)
630 FORMAT (11A2)
END

```

B3020
B3030
B3040
B3050
B3060
B3070
B3080
B3090
B3110
B3120
B3130
B3140
B3150
B3160
B3170
B3180
B3190
B3200
B3210
B3220
B3230
B3240
B3250
B3260
B3270
B3280
B3290
B3300
B3310
B3320
B3330
B3370
B3410
B3420
B3430
B3440
B3450
B3460
B3470
B3480
B3490-

```

C      *****
C
C      THIS SUBROUTINE COMPUTES THE HEAD DISTRIBUTION
C
C      SUBROUTINE ITERAT(NITP,NX,NY,ITMAX,THCK,TMRX,VPRM,HR,HC,HK,
1 WT,REC,RECH,TIM,AOPT,XDEL,YDEL,S,AREA,TOL,TOTLC,N)
C
C      *****
C
C      DOUBLE PRECISION DMINI,DEXP,DLOG,DABS
C      REAL *8TMRX,VPRM,HI,HK,HC,HK,WT,REC,RECH,TIM,AOPT,TITLE
C      REAL *8XDEL,YDEL,S,AREA,SUMT,RHO,PARAM,TEST,TOL,PINT,HMIN,PYR
C      REAL *8B,G,W,A,C,E,F,DR,DC,TBAR,TKM,COEF,BLH,BRK,CHK,QL,BRH
C
C      *****
C      DIMENSION THCK(NX,NY),TMRX(NX,NY,2),VPRM(NX,NY),
1 HR(NX,NY),HC(NX,NY),HK(NX,NY),WT(NX,NY),REC(NX,NY),
2 RECH(NX,NY),TIM(100),AOPT(7),w(99),B(99),G(99)
C      KOUNT=0
C      COMPUTE ROW AND COLUMN
C      CALL NEW ITERATION PARAMETER
10 REMN=MOD(KOUNT,NITP)
   IF (REMN.EQ.0) NTH=0
   NTH=NTH+1
   PARAM=AOPT(NTH)
C      *****
C      ROW COMPUTATIONS
C
   TEST=0.0
   RHO=S/TIM(N)
   BRK=-RHO
   DO 50 IY=1,NY
   DO 20 M=1,NX
   W(M)=0.0
   B(M)=0.0
   G(M)=0.0
20 CONTINUE
   DO 30 IX=1,NX
   IF (THCK(IX,IY).EQ.0.0) GO TO 30
   COEF=VPRM(IX,IY)
   QL=-VPRM(IX,IY)*WT(IX,IY)
   A=TMRX(IX-1,IY,1)/XDEL
   C=TMRX(IX,IY,1)/XDEL
   E=TMRX(IX,IY-1,2)/YDEL
   F=TMRX(IX,IY,2)/YDEL
   TBAR=A+C+E+F
   TKM=TEAR*PARAM
   BLH=-A-C-RHO-COEF-TKM
   BRK=E+F-TKM
   DR=BRH*HC(IX,IY)+BRK*HK(IX,IY)-E*HC(IX,IY-1)-F*HC(IX,IY+1)+QL+RECH
1 (IX,IY)+REC(IX,IY)/AREA
   W(IX)=BLH-A*B(IX-1)
   B(IX)=C/W(IX)
   G(IX)=(DR-A*G(IX-1))/W(IX)
30 CONTINUE
C
C      BACK SUBSTITUTION
C      DO 20 J=2,NX
   IJ=J-1
   IS=NX-IJ
   HR(IJ, IY)=G(IS)-B(IS)*HR(IS+1, IY)

```

```

C      CHK=ABS(HR(IS,IY)-HC(IS,IY))
40 CONTINUE
50 CONTINUE

C
C      *****
C      COLUMN COMPUTATIONS
C
      DO 40 IX=1,NX
      DO 40 M=1,NY
      W(M)=0.0
      B(M)=0.0
60 G(M)=0.0
      DO 0 IY=1,NY
      IF (THCK(IX,IY).EQ.0.0) GO TO 70
      COEF=VPRM(IX,IY)
      QL=-VPRM(IX,IY)*WT(IX,IY)
      A=TMRX(IX,IY-1,2)/YDEL
      C=TMRX(IX,IY,2)/YDEL
      E=TMRX(IX-1,IY,1)/XDEL
      F=TMRX(IX,IY,1)/XDEL
      TBAP=A+C+E+F
      TMK=TBAP*PARAM
      BLH=-A-C-RHO-COEF-TMK
      BRH=E+F-TMK
      DC=BRH*HR(IX,IY)+BRK*HK(IX,IY)-E*HR(IX-1,IY)-F*HR(IX+1,IY)+QL+RECH
      I(IX,IY)+REC(IX,IY)/AREA
      W(IY)=BLH-A*B(IY-1)
      B(IY)=C/W(IY)
      G(IY)=(DC-A*G(IY-1))/W(IY)
70 CONTINUE
C      *****
C      BACK SUBSTITUTION
      DO 80 J=2,NY
      IJ=-1
      IB=NY-IJ
      HC(IX,IB)=G(IB)-B(IB)*HC(IX,IB+1)
      CHK=DABS(HC(IX,IB)-HR(IX,IB))
      IF (CHK.GT.TOL) TEST=1.0
80 CONTINUE
90 CONTINUE

C
C      *****
      KOUNT=KOUNT+1

C
      IF (TEST.EQ.0.0) GO TO 100
      IF (KOUNT.GE.ITMAX) GO TO 100
      GO TO 10

C      *****
C      SET NEW HEAD HK
100 DO 110 IY=1,NY
      DO 110 IX=1,NX
      HK(IX,IY)=HK(IX,IY)
      HK(IX,IY)=HC(IX,IY)

C
C      COMPUTE LEAKAGE FOR MASS BALANCE
      DELQ=VPRM(IX,IY)*AREA*(WT(IX,IY)-HK(IX,IY))
      TOTLQ=TOTLQ+DELQ*TIM(N)
110 CONTINUE

C
      WRITE (6,120) N
      WRITE (6,130) KOUNT

```

```
C  
C RETURN  
C * * * * *  
C  
C  
C 120 FORMAT (3X,4HN = ,115)  
C 130 FORMAT (3H ,23HNUMBER OF ITERATIONS = ,115)  
C END
```

```

C      *****
C
C      THIS SUBROUTINE PRINTS THE HEAD DISTRIBUTION AND COMPUTES
C      THE MASS BALANCE FOR THE FLOW MODEL
C
C      SUBROUTINE OUTPT(N,NX,NY,THCK,VPRM,HI,HK,WT,REC,AREA,SUMT,
1  TOTLQ,S)
C      *****
C      REAL *8TMRX,VPRM,HI,HK,HC,HK,WT,REC,RECH,TIM,AOPT,TITLE
C      REAL *8XDEL,YDEL,S,AREA,SUMT,RHO,PARAM,TEST,TOL,PINT,HMIN,PYR
C
C      DIMENSION THCK(NX,NY),VPRM(NX,NY),HI(NX,NY),HK(NX,NY),
1  WT(NX,NY),REC(NX,NY),IH(99)
C
C      TIMM=SUMT/60.0
C      TIMD=SUMT/(86400.0*155.25)
C      WRITE (6,100)
C      WRITE (6,110) N
C      WRITE (6,120) SUMT
C      WRITE (6,130) TIMM
C      WRITE (6,140) TIMD
C      WRITE (6,150)
C      DO 10 IY=1,NY
10  WRITE (6,160) (HK(IX,IY),IX=1,NX)
C      IF (N.EQ.0) GO TO 90
C      *****
C      WRITE (6,100)
C      WRITE (6,110) N
C      WRITE (6,120) SUMT
C      WRITE (6,130) TIMM
C      WRITE (6,140) TIMD
C      WRITE (6,150)
C      DO 20 IY=1,NY
C      DO 20 IX=1,NX
C      IH(JX)=HK(IX,IY)
20  CONTINUE
C      WRITE (6,170) (IH(ID),ID=1,NX)
30  CONTINUE
C      *****
C      COMPUTE WATER BALANCE AND DRAWDOWN
C
C      QSTP=0.0
C      PUMc=0.0
C      TPUM=0.0
C      QIN=0.0
C      QOUT=0.0
C      QNET=0.0
C      DELQ=0.0
C      WRITE (6,260)
C
C      DO 40 IY=1,NY
C      DO 70 IX=1,NX
C      IH(JX)=0.0
C      IF (THCK(IX,IY).EQ.0.0) GO TO 70
C      TPUM=REC(IX,IY)+TPUM
C      IF (VPRM(IX,IY).EQ.0.0) GO TO 60
C      DELQ=VPRM(IX,IY)*AREA*(WT(IX,IY)-HK(IX,IY))
C      IF (DELQ.GT.0.0) GO TO 40
C      QOUT=QOUT+DELQ
C      GO TO 50
40  QIN=QIN+DFLO

```

```

50 CONTINUE
  QNET=QNET+DELO
60 CONTINUE
  DDRW=HI(IX,IY)-HK(IX,IY)
  IH(IX)=DDRW
  QSTR=QSTR+DDRW*AREA*S
70 CONTINUE
  WRITE (6,270) (IH(IX),IX=1,NX)
80 CONTINUE
  PUMP=TPUM*SUMT
  DELS=-QSTR/SUMT
C
  ERRMB=PUMP-TOTLQ-QSTR
  PCTERR=(ERRMB*100.0)/((PUMP+TOTLQ)/2.0)
C
  WRITE (6,220)
  WRITE (6,230) PUMP
  WRITE (6,210) QSTR
  WRITE (6,240) TOTLQ
  WRITE (6,250) ERRMB,PCTERR
  WRITE (6,180) QIN,QOUT,QNET
  WRITE (6,190) TPUM
  WRITE (6,200) DELS
C
90 RETURN
C
C
C
100 FORMAT (1H1,23HHEAD DISTRIBUTION = ROW)
110 FORMAT (1X,23HNUMBER OF TIME STEPS = ,115)
120 FORMAT (8X,16HTIME (SECONDS) = ,1G12.5)
130 FORMAT (8X,16HTIME (MINUTES) = ,1E12.5)
140 FORMAT (8X,16HTIME (YEARS) = ,1E12.5)
150 FORMAT (1H )
160 FORMAT (1H0,20F5.1)
170 FORMAT (1H ,40I3)
180 FORMAT (1H-,19X,'MASS BALANCE'///18X,'RATES IN C.F.S.'//16X,' QIN
1 = ',G12.5/18X,'QOUT = ',G12.5/18X,'QNET = ',G12.5/)
190 FORMAT (1H ,17X,'TPUM = ',G12.5)
200 FORMAT (1H ,17X,'DELS = ',E12.5/)
210 FORMAT (4X,29HWATER RELEASE FROM STORAGE = ,1E12.5)
220 FORMAT (1H-,10X,'CUMULATIVE MASS BALANCE'///)
230 FORMAT (4X,29HCUMULATIVE NET PUMPAGE = ,1E12.5)
240 FORMAT (4X,29HCUMULATIVE NET LEAKAGE = ,1E12.5)
250 FORMAT (1H0,8X,'MASS BALANCE RESIDUAL = ',G12.5/9X,'ERROR (AS PE
IRCENT) = ',G12.5)
260 FORMAT (1H1,8HDRAWDOWN)
270 FORMAT (3H ,20I5)
  END

```

VELYG SUBROUTINE

```

C *****
C THIS SUBROUTINE CALCULATES VELOCITIES AND DISPERSION
C COEFFICIENTS OVER A FINITE ELEMENT AREA
C
SUBROUTINE VELYG(N,NX,NY,DXX,DYY,DXY,DYX,PERM,
1 HK,DELX,DELY,VX,VY,POROS,ALPHAL,DLTRAT,NODE,NWELL,VXY,VYX,
2 VYM,VYP,VXM,VXP)
C *****
REAL *8 HK,DELX,DELY
DIMENSION DXX(NX,NY),DYY(NX,NY),DXY(NX,NY),DYX(NX,NY),
1 NODE(NX,NY),PERM(NX,NY),HK(NX,NY),VX(NX,NY),VY(NX,NY),
2 NWELL(NX,NY),VXY(NX,NY),VYX(NX,NY),VYM(50),
3 VYP(50),VXM(50),VXP(50)
ALPHAT=ALPHAL*DLTRAT
NNX=NX-1
NNY=NY-1
DO 20 IX=2,NNX
DO 20 IY=2,NNY
10 GRADX=((HK(IX+1,IY)+HK(IX+1,IY+1))-(HK(IX,IY)+HK(IX,IY+1)
1 ))/2.
GRADY=((HK(IX,IY+1)+HK(IX+1,IY+1))-(HK(IX,IY)+HK(IX+1,
1 IY)))/2.
PERMX=(PERM(IX,IY)+PERM(IX+1,IY)+PERM(IX,IY+1)+PERM(IX+1,
1 IY+1))/4.
VX(IX,IY)=-PERMX*GRADX/(POROS*DELX)
VY(IX,IY)=-PERMX*GRADY/(POROS*DELX)
VXS=VX(IX,IY)**2
VYS=VY(IX,IY)**2
V=SQRT(VXS+VYS)
DXX(IX,IY)=(ALPHAL*VXS+ALPHAT*VYS)/V
DYY(IX,IY)=(ALPHAL*VYS+ALPHAT*VXS)/V
DXY(IX,IY)=(ALPHAL-ALPHAT)*VX(IX,IY)*VY(IX,IY)/V
DYX(IX,IY)=DXY(IX,IY)
20 CONTINUE
C OUTPUT VELOCITIES AND DISPERSION COEFFICIENTS
IF (N.GT.1) GO TO 90
WRITE (6,100)
DO 30 IY=1,NY
30 WRITE (6,110) (VX(IX,IY),IX=1,NX)
WRITE (6,120)
DO 40 IY=1,NY
40 WRITE (6,110) (VY(IX,IY),IX=1,NX)
WRITE(6,130)
DO 50 IY=1,NY
50 WRITE (6,110) (DXX(IX,IY),IX=1,NX)
WRITE(6,140)

```

VELYG SUBROUTINE

```

      DO 60 IY=1,NY
60  WRITE (6,110) (DXY(IX,IY),IX=1,NX)
      WRITE(6,150)
      DO 70 IY=1,NY
70  WRITE (6,110) (DYY(IX,IY),IX=1,NX)
      WRITE(6,160)
      DO 80 IY=1,NY
80  WRITE (6, 110) (DYX(IX,IY),IX=1,NX)
90  CONTINUE
C   *****
      RETURN
C
100 FORMAT (1H1,12HX VELOCITIES)
110 FORMAT (1H ,12G10.3)
120 FORMAT (1H1,12HY VELOCITIES)
130 FORMAT (1H1,'DXX  DISPERSION COEFFICIENTS 1//)
140 FORMAT (1H1,'DXY  DISPERSION COEFFICIENTS 1//)
150 FORMAT (1H1,'DYY  DISPERSION COEFFICIENTS 1//)
160 FORMAT (1H1,'DYX  DISPERSION COEFFICIENTS 1//)
      END

```

```

C      *****
C
C      THIS SUBROUTINE CALCULATES THE CHEMICAL CONCENTRATION AT
C      THE FINITE ELEMENT NODES
C
C      SUBROUTINE TRSPTG(NODE,XKD,XKIN,XN,XRHO,DXX,DYY,DXY,DYX,QI,
1  SYMBOL,DFLT,IMAX,TIMEI,C,CONK,PMP,NX,NY,NNX,NNY,THCK,RECH,
2  XDFL,YDEL,VX,VY,CNPECH,POROS,WFAC,EPS,VXM,VXP,VYM,VYP,NWELL)
C
C      * * * * *
C      DOUBLE PRECISION DMINI,DEXP,DLOG,DABS
C      REAL *8IMPX,VPRM,HI,HP,HC,HK,WI,REC,RECH,TIM,AOPT,TITLE
C      REAL *8XDEL,YDEL,S,AREA,SUMT,RHO,PARAM,TEST,TOL,PINT,HMIN,PYR
C      REAL *8TINT,ALPHA,ANITP,DIFFX
C
C      DIMENSION NODE(NX,NY),DXX(NX,NY),DYY(NX,NY),DXY(NX,NY),
1  DYX(NX,NY),QI(NX,NY),SYMBOL(11),C(NX,NY),PMP(NX,NY),
2  THCK(NX,NY),RECH(NX,NY),VX(NX,NY),VY(NX,NY),CNRECH(NX,NY)
C      * * * * *
C      THESE DIMENSION STATEMENTS MUST BE CHANGED WITH ARRAY CHANGES
C      DIMENSION IC(20,20),XUXX(3,3,4),XDYY(3,3,4),XDXY(3,3,4),
1  XDYX(3,3,4),XVX(3,3,4),XVY(3,3,4),AM(3,3,4),X(20,20,3,3),
2  Y(20,20,3,3),XDXT(3,3),XDYYT(3,3),XDXYT(3,3),XDYXT(3,3),
3  XVYI(3,3),XVYT(3,3),AMT(3,3),RSIDE(20,20),QT(3,3),GG(20,20)
C      EXTERNAL FCTD,FCT
C      IF (XKIN.LT.1.F-20) XN=1.
C      XKIN=XKIN/.315576F08
C      CSTOR1 = 0.0
C      CHAY=0.0
C      NI=NX
C      NJ=NY
C      POR=POROS
C      DELX=XDEL
C      DELY=YDEL
C      XXX=CONK
C      FLUX=0.0
C      FLUXI=0.0
C      UT=DELT
C      NTOT=50
C      WFLUXI=0.0
C      WFLIXO=0.0
C      CSTOR=0.0
C      NNJ=NNJ-1
C      NNI=NNI-1
C      NNNI=NNI-1
C      NNNJ=NNJ-1
C      TIME=0.0
C      TIMEX=TIMEI
C      WRITE (6,340)
C      N=0
C      RF=1.+XKD*XRHO
C      DO 10 J=2,NNJ
C      DO 10 I=2,NNI
C      CSTOR1 = C(I,J)*DELX*DELY*THCK(I,J)*POR + CSTOR1
10  CONTINUE
C      NTOT=100
C      CALCULATE INTEGRAL VALUES FOR LINEAR BASIS FUNCTIONS
C      DO 20 KX=1,4
C      DO 20 Jx=1,3
C      DO 20 Ix=1,3
C      K=KX

```

```

      J=JX
      I=IX
      XDXY(I,J,K) = GAUSS5(FCTD,FCT,FCTD,FCT,I,J,K)
      XDYY(I,J,K) = GAUSS5(FCT,FCTD,FCT,FCTD,I,J,K)
      XDXY(I,J,K) = GAUSS5(FCTD,FCT,FCT,FCTD,I,J,K)
      XDYY(I,J,K) = GAUSS5(FCT,FCTD,FCT,FCTD,I,J,K)
      XVX(I,J,K) = GAUSS5(FCTD,FCT,FCT,FCT,I,J,K)*DELX
      XVY(I,J,K) = GAUSS5(FCT,FCTD,FCT,FCT,I,J,K)*DELX
      AM(I,J,K) = GAUSS5(FCT,FCT,FCT,FCT,I,J,K) *DELX*DELY
20  CONTINUE
C
C      SUM THE INTEGRAL VALUES FOR THE FOUR ELEMENTS
C
      DO 40 I=2,NNI
      DO 40 J=2,NNJ
        DO 30 IB=1,3
        DO 30 JB=1,3
          AMT(IB,JB)=AM(IB,JB,1)+AM(IB,JB,2)+AM(IB,JB,3)+AM(IB,JB,4)
          QT(IB,JB)=AMT(IB,JB)*PMP(I,J)/(THCK(I,J)*POROS*4.)
          XDXT(IB,JB)=XDXX(IB,JB,1)*DXX(I-1,J-1)+XDXX(IB,JB,2)*DXX(I,J-1)
1          +XDXX(IB,JB,3)*DXX(I-1,J)+XDXX(IB,JB,4)*DXX(I,J)
          XDYYT(IB,JB)=XDYY(IB,JB,1)*DYY(I-1,J-1)+XDYY(IB,JB,2)*DYY(I,J-
1          1)+XDYY(IB,JB,3)*DYY(I-1,J)+XDYY(IB,JB,4)*DYY(I,J)
          XDYXT(IB,JB)=XDYX(IB,JB,1)*DYX(I-1,J-1)+XDYX(IB,JB,2)*
1          DYX(I,J-1)+XDYX(IB,JB,3)*DYX(I-1,J)+
2          XDYX(IB,JB,4)*DYX(I,J)
          XDXYT(IB,JB)=XDXY(IB,JB,1)*DXY(I-1,J-1)+XDXY(IB,JB,2)*DXY(I,
1          J-1)+XDXY(IB,JB,3)*DXY(I-1,J)+XDXY(IB,JB,4)*DXY(I,J)
          XVXT(IB,JB)=XVX(IB,JB,1)*VX(I-1,J-1)+XVX(IB,JB,2)*VX(I,J-1)+
1          XVX(IB,JB,3)*VX(I-1,J)+XVX(IB,JB,4)*VX(I,J)
          XVYT(IB,JB)=XVY(IB,JB,1)*VY(I-1,J-1)+XVY(IB,JB,2)*VY(I,J-1)+
1          XVY(IB,JB,3)*VY(I-1,J)+XVY(IB,JB,4)*VY(I,J)
          AAA=DELT/(2.*RF)*(XDXTT(IB,JB)+XDYYT(IB,JB)+XDYXT(IB,JB)+
1          XDXYT(IB,JB)+XVXTT(IB,JB)+XVYT(IB,JB))
          X(I,J,IB,JB)=AMT(IB,JB)+AAA*XKIN*DELT/2.*AMT(IB,JB)
          Y(I,J,IB,JB)=AMT(IB,JB)-AAA*XKIN*DELT/2.*AMT(IB,JB)
30  CONTINUE
40  CONTINUE
50  CONTINUE
      NIT = 0
      TIME=TIME+DT
      IF(TIME.GT.TMAX) GO TO 330
C
C      CALCULATE THE RIGHT HAND SIDE KNOWN VECTOR
C
      DO 60 I=2,NNI
      DO 60 J = 2,NNJ
        RSIDE(I,J)=Y(I,J,1,1)*C(I-1,J-1)+Y(I,J,1,3)*C(I-1,J+1)+Y(I,J,3,3)*
1C(I+1,J+1) + Y(I,J,2,1)*C(I,J-1) + Y(I,J,3,2)*C(I+1,J) + Y(I,J,2,
2  3) * C(I,
3J+1) + Y(I,J,1,2)*C(I-1,J) + Y(I,J,2,2)*C(I,J)+Y(I,J,3,1)*C(I+1,J
4  -1)+QI(I,J)*DELT/(4.*POROS*THCK(I,J))
5*(4.*CNRECH(I,J)-2.*C(I,J))
60  CONTINUE
C      USE SOR POINT ITERATION TECHNIQUE TO SOLVE EQUATION
70  RES = 0.0
      NIT=NIT+1
      IF(NIT.LT.NTOT) GO TO 90
      WRITE(6,80) NTOT
80  FORMAT(1H , ' NUMBER OF ITERATIONS EXCEEDED ',I5)
      GO TO 330

```

```

90 CONTINUE
   DO 210 J=2,NNJ
   DO 210 I=2,NNI
   K=MODE(I,J)+1
   GO TO(100,110,120,130,140,150,160,170,180,190),K
100  CN=(-(X(I,J,1,1)*C(I-1,J-1)+X(I,J,2,1)*C(I,J-1) + X(I,J,3,1)
1* C(I+1,J-1) + X(I,J,1,2) *C(I-1,J) + X(I,J,3,2) *C(I+1,J) +
2X(I,J,1,3) *C(I-1,J+1) + X(I,J,2,3)*C(I,J+1) + X(I,J,3,3)*C(I+1,J+
31)) + RSICE(I,J) ) / (X(I,J,2,2)+QI(I,J)*2.*DELT/
4(4.*POROS*THCK(I,J))*WFAC-(WFAC-1.)*C(I,J)
   GO TO 200
110 CN = AMAX1(C(I,J+1),CONK)
C     CN=CONK
C 31 CN =C(I,J+2)
   GO TO 200
C 32 CN =C(I,J-2)
120 CN = AMAX1(C(I,J-1),CONK)
C     CN=CONK
   GO TO 200
C 33 CN =C(I+2,J)
130 CN = AMAX1(C(I+1,J),CONK)
C     CN=CONK
   GO TO 200
C 34 CN =C(I-2,J)
140 CN = AMAX1(CONK,C(I-1,J))
C     CN=CONK
   GO TO 200
150 CN =C(I+2,J-2)
   GO TO 200
160 CN =8000.
   GO TO 200
170 CONTINUE
   CN=0.0
   GO TO 200
180 CONTINUE
   GO TO 100
190 CONTINUE
   GO TO 200
200 CONTINUE
   DIFFX=CN-C(I,J)
   RES=DABS(DIFFX)+RES
   C(I,J)=CN
210 CONTINUE
   IF (RES.GT.EPS) GO TO 70
   DO 220 I=1,NI
   DO 220 J=1,NJ
   IC(I,J) = C(I,J) +0.1
220 CONTINUE
C     COMPUTE MASS BALANCE
   DO 230 I=2,NNI
   DO 230 J=2,NNJ
   WFLUXI=QI(I,J)*CNRECH(I,J)*DELT + WFLUXI
230 WFLUXO=PMP(I,J)*C(I,J)*DELT + WFLUXO
C     NET FLUX THROUGH SYSTEM BOUNDRIES
   FLUX=0.
   DO 240 I=3,NNNI
   J=2
   FLUX=(C(I,2)-C(I,3))*DYY(I,J)*THCK(I,J)*POR/DELX+FLUX
   FLUX = VY(I,2) * (C(I,2) ) *THCK(I,J)*POR + FLUX
   J=NNNJ
   FLUX=(C(I,NNJ)-C(I,NNNJ))*DYY(I,J)*THCK(I,J)*POR/DELX+FLUX

```

```

240 FLUX = -(VY(I,NNNJ)*C(I,NNJ) ) *THCK(I,J)*POR+FLUX
DO 250 J=3,NNNJ
  I=2
  FLUX=(C(2,J)-C(3,J))*DXX(I,J)*THCK(I,J)*POR/DELX+FLUX
  FLUX = VX(2,J)* (C(2,J)) *THCK(I,J)*POR+FLUX
  I=NNNI
  FLUX=(C(NNI,J)-C(NNI,J))*DXX(I,J)*THCK(I,J)*POR/DELX+FLUX
250 FLUX = -(VX(NNI,J)* (C(NNI,J)) ) *THCK(I,J)*POR+FLUX
  FLUXI = FLUX*DFLX*DT+FLUXI
  IF (TIME.LT.TIMEX-TIMEI*.00010) GO TO 50
  TIMEX=TIMEX+TIMEI
  CSTOR=0.0
  DO 260 I=2,NNI
  DO 260 J=2,NNJ
260 CSTOR=CSTOR+C(I,J)*DELX*DELY*THCK(I,J)*POR
  FLNET=WFLUXI-WFLUXO-CSTOR+FLUXI+CSTORI
  DAYS = TIME/.86E05
  YEARS = TIME/.315576E08
  WRITE(6,370) WFLUXI,WFLUXO,FLUXI,CSTOP,FLNET,TIME,DAYS,YEARS
  WRITE(6,270) NIT
270  FORMAT(1H ,4X,'NIT = ',I5)
  WRITE(6,380)
  DO 280 J=1,NJ
280  WRITE(6,410) (C(I,J),I=1,NI)
  WRITE(6,340)
  WRITE(6,380)
  DO 290 J=1,NJ
290  WRITE(6,390) (IC(I,J),I=1,NI)
  WRITE(6,340)
  DO 300 I=2,NNI
  DO 300 J=2,NNJ
  YYY=C(I,J)
300  CMAX = AMAX1(XXX,CNRECH(I,J),CMAX,YYY)
  IF (CMAX.LE.1.E-20) CMAX=1.
  DO 310 J=1,NJ
  DO 310 I=1,NI
  K=(1-C(I,J)/CMAX)*10.+1.000001
310  GG(I,J)=SYMBOL(K)
  DO 320 J=1,NJ
320  WRITE(6,400) (GG(I,J),I=1,NI)
  WRITE(6,420)
  WRITE(6,340)
  GO TO 50
330 CONTINUE
  STOP
C
340 FORMAT (1H1)
350 FORMAT (1H ,12E10.3)
360 FORMAT (1H1,15X,'MAXIMUM VALUE FOR TIME INCREMENT FOR EXPLICIT PRO
  1GRAM'///)
370 FORMAT (1H1,3X,'TOTAL WELL INPUT =',E14.5/4X,'TOTAL WELL OUTPUT =',
  1,E14.5/4X,'NET BOUNDRY FLUX =',E14.5/4X,'TOTAL ACCUMULATION = ',E1
  24.5/4X,'TOTAL NET FLUX =',E14.5/4X,'TIME =',E14.5/4X,'DAYS =',E14.
  35/ 4X,'YEARS =',E14.5)
380 FORMAT (1H0,5X,'CHEMICAL CONCENTRATION ',//)
390 FORMAT (1H ,32I4)
400 FORMAT (1H ,60A2)
410 FORMAT (1H ,12E10.3)
420 FORMAT (1H ,///)
  END
  FUNCTION GAUSSS (FCTI,FCTJ,FCTK,FCTL,I,J,K)

```

```

      GO TO (10,20,30,40,50),K
10   M=1
      N=1
      GO TO 50
20   M=2
      N=1
      GO TO 50
30   M=1
      N=2
      GO TO 50
40   M=2
      N=2
50   Y= .2777778*(FCTI(.8872983,1,M)*FCTK(.8872983,2,M)+FCTI(.1127017,1
      1,M)*FCTK(.1127017,2,M))+.4444444*FCTI(.5,1,M)*FCTK(.5,2,M)
      GAUSS5= (.2777778*(FCTJ(.8872983,J,N)*FCTL(.8872983,2,N)+FCTJ(.112
      17017,J,N)*FCTL(.1127017,2,N))+.4444444*FCTJ(.5,J,N)*FCTL(.5,2,N))
      2*Y
      IF (ABS(GAUSS5).LT.1.E-05) GAUSS5=0.0
      RETURN
      END
      FUNCTION FCT(Z,I,NIJ)
      GO TO (10,50),NIJ
10   GO TO (20,30,40),I
20   FCT=1.-Z
      GO TO 60
30   FCT=Z
      GO TO 60
40   FCT=0.0
      GO TO 60
50   GO TO (40,20,30),I
60   CONTINUE
      RETURN
      END
      FUNCTION FCTD(Z,I,NIJ)
      GO TO(10,50),NIJ
10   GO TO(20,30,40),I
20   FCTD=-1.
      GO TO 60
30   FCTD=1.
      GO TO 60
40   FCTD=0.
      GO TO 60
50   GO TO(40,20,30),I
60   CONTINUE
      RETURN
      END

```

```

C      * * * * *
C      THIS ROUTINE CONTROLS EXECUTION
C      SOLUTE TRANSPORT,CHEMICAL REACTION AND DISPERSION IN A POROUS
C      MEDIUM -- NUMERICAL SOLUTION-- FLOW EQUATIONS SOLVED BY IADI
C      BY J. BREDEHOEFT - 1973,MODIFIED BY D. GROVE 1976,
C      SOLUTE TRANSPORT AND REACTION EQUATION SOLVED BY GLERKIN
C      FINITE ELEMENT METHOD USING CUBIC FUNCTIONS BY D. GROVE 1976.
C      * * * * *
C      DOUBLE PRECISION DMIN1,DEXP,DLOG,DABS
C      REAL *8TMRX,VPRM,HI,HR,HC,HK,WT,REC,RECH,TIM,AOPT,TITLE
C      REAL *8XDEL,YDEL,S,AREA,SUMT,RHO,PARAM,TEST,TOL,PINT,HMIN,PYR
C      REAL *8 TINT,ALPHA1,ANITP,XSM,YSM
C
C      DIMENSION NODE(7,8),DXX(11,12),DYY(11,12),DXY(11,12),DYX(11,12),
C      1 QI(11,12),SYMBOL(11),C(7,8),PMP(11,12),VXM(50),VXP(50),VYM(50),
C      2 VYP(50),NWELL(11,12),NODEID(11,12),
C      3 THCK(11,12),PERM(11,12),TMWL(11,12), TMRX(11,12,2)
C      DIMENSION VPRM(11,12),HI(11,12),HR(11,12),HC(11,12),HK(11,12),
C      1 WT(11,12),REC(11,12),RECH(11,12),TIM(100),AOPT(7),TITLE(10),
C      2 VX(11,12),VY(11,12),CNRECH(11,12),VYX(11,12),VXY(11,12)
C      *****
C      DIMENSION NEL(8,9),NE(8,9),VXS(3,3,8),VYS(3,3,8),
C      1 DXS(3,3,8),DYYS(3,3,8),DXYS(3,3,8),DYXS(3,3,8)
C      LOAD DATA
C      NX=11
C      NY=12
C      NGI=NX-4
C      NGJ=NY-4
C      INX=NGI
C      INY=NGJ
C      NGIF=NGI+1
C      NGJF=NGJ+1
10  CONTINUE
C      CALL PARLOD(NODE,XKD,XKIN,XN,XRHO,DXX,DYY,DXY,DYX,QI,SYMBOL,
C      1 DELT,TMAX,TIMEI,C,CONK,WFAC,EPS,PMP,VXM,VXP,VYM,VYP,
C      2 NWELL, NTIM,NPMP,NPNT,NITP,N,NX,NY,NP,NREC,INT,NNX,NNY,
C      3 ITMAX,IPRNT,NODEID,
C      4 THCK,PERM, TMRX,VPRM,HI,HR,HC,HK,WT,REC,RECH,
C      5 TIM,AOPT,TITLE,XDEL,YDEL,S,AREA,SUMT,RHO,PARAM,TEST,TOL,
C      6 PINT,HMIN,PYR,VX,VY,CNRECH,POROS,SUMTCH,BETA,TIMV,STORM,RCMAS,
C      7 FLMIN,FLMOT,SUMIO, DLTRAT,VYX,VXY,TOTLO,INX,INY,IIX)
C      *****
C      START COMPUTATIONS
C      COMPUTE NPMP PUMPING PERIODS
C      DO 100 NINT=1,NPMP
C      INT=NINT
C
C      COMPUTE NTIM TIME STEPS
C      DO 20 NS=1,NTIM
C      N=NS
C      IPRNT=0
C      LOAD NEW DELTA T
C      TINT=SUMT-PYR*(INT-1)
C      TDEL=DMIN1(TIM(N),PYR-TINT)
C      SUMT=SUMT+TDEL
C      TIM(N)=TDEL
C      REMN=MOD(N,NPNT)
C      *****

```

```

20 CONTINUE
C   COMPUTE HEAD DISTRIBUTION
   CALL ITERAT(NITP,NX,NY,ITMAX,THCK,TMRX,VPRM,HR,HC,HK,WT,REC,
1 RECH,TIM,AOPT,XDEL,YDEL,S,AREA,TOL,TOTLQ,N)
30 CONTINUE
   IF(PERMN.EQ.0.0)CALL OUTPT(N,NX,NY,THCK,VPRM,HI,HK,WT,
1 REC,AREA,SUMT,TOTLQ,S)
40 CONTINUE
C   SUBDIVIDE NODES WITH WELLS IN THEM
   CALL NODES(NX,NY,TMRX,RECH,REC,TIM,XDEL,YDEL,TOL,
1 HK,XSM,YSM,NODE,PERM,ITMAX,AOPT,S,TOTLQ,NEL,NE,
2 VXS,VYS,DXXS,DYYS,DXYS,DYXS,POROS,BETA,DLTRAT,NGI,NGJ,NGIE,NGJE)
50 CONTINUE
C   CALCULATE VELOCITIES AND DISPERSION COEFFICIENTS
   CALL VELYG(N,NX,NY,DXX,DYY,DXY,DYX,PERM,HK,XDEL,
1 YDEL,VX,VY,POROS,BETA,DLTRAT)
60 CONTINUE
C   CALCULATE CONCENTRATIONS
   CALL TRSPTC(NODE,XKD,XKIN,XN,XRHO,DXX,DYY,DXY,DYX,QI,
1 SYMBOL,DELT,IMAX,TIMEI,C,CONK,PMP,NX,NY,NNX,NNY,THCK,RECH,
2 XDEL,YDEL,VX,VY,CNRECH,POROS,INX,INY,XSM,VXS,VYS,
3 DXXS,DYYS,DXYS,DYXS,NEL,IIX,NGIE,NGJE)
C   *****
C   OUTPUT ROUTINES
70 IF (SUMT.GE.(PYR*INT)) GO TO 90
80 CONTINUE
C   *****
C   SUMMARY OUTPUT
90 CONTINUE
   IPRNT=1
C   CALL OUTPT
100 CONTINUE
C   *****
   STOP
C   *****
END

```

```

C *****
C
C THIS SUBROUTINE INPUTS AND INITIALIZES DATA
C
SUBROUTINE PARLOD(NODE,XKD,XKIN,XN,XRHO,DXX,DYY,DXY,DYX,QI,
1 SYMBOL,DELTA,TMAX,TIMEI,C,CONK,WFAC,EPS,PMP,VXM,VXP,VYM,VYP,
2 NWFLL,NTIM,NPMP,NPNT,NITP,N,NX,NY,NP,NREC,INT,NNX,NNY,
3 ITMAX,IPRNT,NODEID,
4 THCK,PERM,TMRX,VPRM,HI,HR,HC,HK,WT,REC,RECH,
5 TIM,AOPT,TITLE,XDEL,YDEL,S,AREA,SUMT,RHO,PARAM,TEST,TOL,
6 PINT,HMIN,PYR,VX,VY,CNRECH,POROS,SUMTCH,BETA,TIMV,STORM,RCMAS,
7 FLMIN,FLMOT,SUMIO,DLTRAT,VYX,VXY,TOTLO,INX,INY,IIX)
C *****
DOUBLE PRECISION DMIN1,DEXP,DLOG,DABS
REAL *8TMRX,VPRM,HI,HR,HC,HK,WT,REC,RECH,TIM,AOPT,TITLE
REAL *8XDEL,YDEL,S,AREA,SUMT,RHO,PARAM,TEST,TOL,PINT,HMIN,PYR
REAL *8FCTR,TIMX,TINIT,PIES,YS,XNS,RAT,HMX,HMY
REAL *8TINT,ALPHA1,ANITP
DIMENSION NODE(INX,INY),DXX(NX,NY),DYY(NX,NY),DXY(NX,NY),
1 DYX(NX,NY),QI(NX,NY),SYMBOL(11),C(INX,INY),PMP(NX,NY),
2 VXM(50),VXP(50),VYM(50),VYP(50),NWFLL(NX,NY)
DIMENSION NODEID(NX,NY),THCK(NX,NY),
1 PERM(NX,NY),TMWL(5,50),TMRX(NX,NY,2),
2 VPRM(NX,NY),HI(NX,NY),HR(NX,NY),HC(NX,NY),HK(NX,NY),WT(NX,NY),
3 REC(NX,NY),RECH(NX,NY),TIM(100),AOPT(NX),TITLE(10),
4 VX(NX,NY),VY(NX,NY),CNRECH(NX,NY),VYX(NX,NY),VXY(NX,NY)
C *****
READ(5,830)TITLE
WRITE(6,840)TITLE
C *****
C INITIALIZE TEST AND CONTROL VARIABLES
TEST=0.0
TOTLO=0.0
SUMT=0.0
SUMTCH=0.0
INT=0
IPRNT=0
NCA=0
N=0
C *****
C LOAD CONTROL PARAMETERS
READ(5,850)NTIM,NPMP,NX,NY,NPNT,NITP,ITMAX,NREC,TMAX,T
1 TIMEI,CONK,WFAC,EPS,TCK,TTT,WTE,CONI,NNODE,IIX
READ(5,920)PINT,TOL,POROS,BETA,S,TIMX,TINIT,XDEL,YDEL,DLTRAT
1,XKD,XKIN,XN,XRHO,DELTA
READ(5,970)SYMBOL
PYR=PINT*86400.0*365.25
NNX=NX-1
NNY=NY-1
C
WRITE(6,860)
WRITE(6,870)NX,NY,XDEL,YDEL
WRITE(6,880)NTIM,NPMP,PINT,TIMX,TINIT,DELTA,TMAX,TIMEI
WRITE(6,890)S,POROS,BETA,DLTRAT,XKD,XKIN,XN,XRHO,EPS,WFAC
WRITE(6,900)NITP,TOL,ITMAX
WRITE(6,910)NPNT,NREC
C *****
C LIST TIME INCREMENTS
DO 10 J=1,NTIM
TIM(J)=0.0
10 CONTINUE

```

```

      TIM(1)=TINIT
      IF (S.EQ.0.0) GO TO 30
      DO 20 K=2,NTIM
      TIM(K)=TIMX*TIM(K-1)
20  CONTINUE
      GO TO 40
30  TIM(1)=PYR
C
40  WRITE (6,570)
      WRITE (6,580) TIM
C *****
C INITIALIZE MATRICES
      DO 50 IY=1,NY
      DO 50 IX=1,NX
      VPRM(IX,IY)=0.0
      PERM(IX,IY)=0.0
      THCK(IX,IY)=0.0
      RECH(IX,IY)=0.0
      CNRECH(IX,IY)=0.0
      REC(IX,IY)=0.0
      NODEID(IX,IY)=0
      TMRX(IX,IY,1)=0.0
      TMRX(IX,IY,2)=0.0
      HI(IX,IY)=0.0
      HR(IX,IY)=0.0
      HC(IX,IY)=0.0
      HK(IX,IY)=0.0
      WT(IX,IY)=0.0
      VX(IX,IY)=0.0
      VY(IX,IY)=0.0
      VXY(IX,IY)=0.0
      VYX(IX,IY)=0.0
      DXX(IX,IY)=0.0
      DXY(IX,IY)=0.0
      DYY(IX,IY)=0.0
      DYX(IX,IY)=0.0
      QI(IX,IY)=0.0
      PMP(IX,IY)=0.0
50  CONTINUE
      DO 60 IX=1,INX
      DO 60 IY=1,INY
60  C(IX,IY)=0.0
C *****
C READ PUMPAGE DATA -- (X-Y COORDINATES AND RATE IN C.F.S.)
C SIGNS ---- WITHDRAWAL = POS.; INJECTION = NEG.
C IF INJECTION WELL, ALSO READ CONCENTRATION OF INJECTED WATER
      IF (NREC.LE.0) GO TO 100
      WRITE (6,950)
      DO 90 I=1,NREC
      READ (5,820) IX,IY,FCTR,CNREC
      NWELL(IX,IY)=I
      IF (FCTR.LT.0.0) CNRECH(IX,IY)=CNREC
      REC(IX,IY)=FCTR
      IF (FCTR.GT.0.0) GO TO 70
      QI(IX,IY)=-FCTR
      GO TO 80
70  QI(IX,IY)=0.0
      PMP(IX,IY)=FCTR
80  WRITE (6,960) IX,IY,REC(IX,IY),CNRECH(IX,IY),QI(IX,IY)
90  CONTINUE
100 CONTINUE

```

```

C *****
C AREA=XDEL*YDEL
C
C WRITE (6,800) AREA
C WRITE (6,690)
C WRITE (6,700) XDEL
C WRITE (6,700) YDEL
C *****
C READ TRANSMISSIVITY IN FT**2/SEC INTO VPRM MATRIX
C FCTR = TRANSMISSIVITY MULTIPLIER ---> FT**2/SEC
C
C FCTR=1.0
C IF (TT.LT..1E-10) GO TO 120
C DO 110 IY=1,NY
C DO 110 IX=1,NX
C VPRM (IX,IY)=TTT*FCTR
C IF (IX.EQ.1.OR.IX.EQ.NX) VPRM (IX,IY)=0.0
C IF (IY.EQ.1.OR.IY.EQ.NY) VPRM (IX,IY)=0.0
110 CONTINUE
C GO TO 150
120 CONTINUE
C
C DO 130 IY=1,NY
130 READ (5,640) (VPRM (IX,IY),IX=1,NX)
C DO 140 IX=1,NX
C DO 140 IY=1,NY
C VPRM (IX,IY)=VPRM (IX,IY)*FCTR
C IF (IX.EQ.1.OR.IX.EQ.NX) VPRM (IX,IY)=0.0
C IF (IY.EQ.1.OR.IY.EQ.NY) VPRM (IX,IY)=0.0
140 CONTINUE
150 CONTINUE
C
C WRITE (6,620)
C DO 160 IY=1,NY
160 WRITE (6,610) (VPRM (IX,IY),IX=1,NX)
C *****
C SET UP COEFFICIENT MATRIX
C AVERAGE TRANSMISSIVITY
C BLOCK CENTERED GRID
C GEOMETRIC MEAN
C
C PIES=3.1415927*3.1415927/2.0
C YNS=NY*NY
C XNS=NX*NX
C HMIN=2.0
C DO 170 IY=2,NNY
C DO 170 IX=2,NNX
C IF (VPRM (IX,IY).EQ.0.0) GO TO 170
C TMRX (IX,IY,1)=2.0*VPRM (IX,IY)*VPRM (IX+1,IY)/(VPRM (IX,IY)*XDEL+VPRM
C 1 (IX+1,IY)*XDEL)
C TMRX (IX,IY,2)=2.0*VPRM (IX,IY)*VPRM (IX,IY+1)/(VPRM (IX,IY)*YDEL+VPRM
C 1 (IX,IY+1)*YDEL)
C
C ADJUST COEFFICIENT FOR ANISOTROPY
C FCTR=1.0
C
C TMRX (IX,IY,1)=TMRX (IX,IY,1)*FCTR
C TMRX (IX,IY,2)=TMRX (IX,IY,2)*FCTR
C *****
C COMPUTE MINIMUM ITERATION PARAMETER
C HMIN MINIMUM ITERATION PARAMETER

```

```

C
  IF (TMRX(IX,IY,1).EQ.0.0) GO TO 170
  IF (TMRX(IX,IY,2).EQ.0.0) GO TO 170
  RAT=TMRX(IX,IY,1)*YDEL/(TMRX(IX,IY,2)*XDEL)
  HMX=PIES/(XNS*(1.0+RAT))
  HMY=PIES/(YNS*(1.0+(1.0/RAT)))
  IF (HMX.LT.HMIN) HMIN=HMX
  IF (HMY.LT.HMIN) HMIN=HMY
170 CONTINUE
C
C   WRITE (6,23)
C   WRITE (6,22) (((TMRX(IX,IY,IL),IX=1,NX),IY=1,NY),IL=1,2)
C   *****
C   READ AQUIFER THICKNESS
  IF (TCK.LT,.1E-10) GO TO 190
  DO 180 IY=1,NY
  DO 180 IX=1,NX
180 THCK(IX,IY)=TCK
  GO TO 210
190 CONTINUE
  DO 200 IY=1,NY
200 READ (5,630) (THCK(IX,IY),IX=1,NX)
C
210 CONTINUE
  DO 220 IX=1,NX
  DO 220 IY=1,NY
  IF (IX.EQ.1.OR.IY.EQ.1) THCK(IX,IY)=0.0
  IF (IX.EQ.NX.OR.IY.EQ.NY) THCK(IX,IY)=0.0
220 CONTINUE
  WRITE (6,600)
  DO 230 IY=1,NY
230 WRITE (6,590) (THCK(IX,IY),IX=1,NX)
C   *****
C   COMPUTE PERMEABILITY FROM TRANSMISSIVITY
C   COUNT NO. OF CELLS IN AQUIFER
C
  DO 240 IX=1,NX
  DO 240 IY=1,NY
  IF (THCK(IX,IY).EQ.0.0) GO TO 240
  PERM(IX,IY)=VPRM(IX,IY)/THCK(IX,IY)
  NCA=NCA+1
240 VPRM(IX,IY)=0.0
  AAQ=NCA*AREA
C
  WRITE (6,710)
  DO 250 IY=1,NY
250 WRITE (6,740) (PERM(IX,IY),IX=1,NX)
  WRITE (6,720) NCA,AAQ
C   *****
C   READ NODE IDENTIFICATION CARDS + SET VERTICAL PERMEABILITY.....
  IF (NNODE.LT.1) GO TO 280
  DO 260 IX=2,NNX
  NODEID(IX,2)=1
260 NODFID(IX,NNY)=1
  DO 270 IY=2,NNY
  NODFID(2,IY)=1
270 NODFID(NNX,IY)=1
  GO TO 300
280 CONTINUE
C   NODF IDENTIFICATION CODE: 1 = CONSTANT HEAD BOUNDARY
  DO 290 IY=1,NY

```

```

290 READ (5,730) (NODEID(IX,IY),IX=1,NX)
300 CONTINUE
   DO 310 IX=1,NX
   DO 310 IY=1,NY
   IF (THCK(IX,IY).EQ.0.0) GO TO 310
   IF (NODEID(IX,IY).GT.0) VPRM(IX,IY)=1.0
310 CONTINUE
C
   WRITE (6,650)
   DO 320 IY=1,NY
320 WRITE (6,670) (NODEID(IX,IY),IX=1,NX)
   WRITE (6,680)
   DO 330 IY=1,NY
330 WRITE (6,610) (VPRM(IX,IY),IX=1,NX)
C
*****
C
   READ WATER-TABLE ELEVATION
   IF (NNODE.LT.1) GO TO 340
   READ (5,760) (WT(IX,2),IX=1,NX)
   READ (5,760) (WT(IX,NNY),IX=1,NX)
   READ (5,760) (WT(2,IY),IY=1,NY)
   READ (5,760) (WT(NNX,IY),IY=1,NY)
   GO TO 380
340 CONTINUE
   IF (WTE.LT..1E-10) GO TO 360
   DO 350 IY=1,NY
   DO 350 IX=1,NX
350 WT(IX,IY)=WTE
   GO TO 380
360 CONTINUE
   DO 370 IY=1,NY
370 READ (5,760) (WT(IX,IY),IX=1,NX)
C
380 CONTINUE
   DO 390 IX=1,NX
   DO 390 IY=1,NY
   IF (IX.EQ.1.OR.IY.EQ.1) WT(IX,IY)=0.0
   IF (IX.EQ.NX.OR.IY.EQ.NY) WT(IX,IY)=0.0
390 CONTINUE
   WRITE (6,780)
   DO 400 IY=1,NY
400 WRITE (6,790) (WT(IX,IY),IX=1,NX)
C
*****
C
   SET INITIAL HEAD
C
   SET HC FOR INITIAL CONDITIONS
   DO 410 IX=1,NX
   DO 410 IY=1,NY
   HI(IX,IY)=WT(IX,IY)
   HC(IX,IY)=HI(IX,IY)
   HR(IX,IY)=HI(IX,IY)
   HK(IX,IY)=HI(IX,IY)
410 CONTINUE
   WRITE (6,660)
C
*****
C
   READ CHEMICAL ID NODES
   DO 420 IY=1,INY
420 READ (5,750) (NODE(IX,IY),IX=1,INX)
   DO 430 IY=1,INY
430 WRITE (6,670) (NODE(IX,IY),IX=1,INX)
C
*****
C
   READ IN INITIAL CHEMICAL CONCENTRATION

```

```

      IF (CONI.LT..1E-20) GO TO 450
      DO 440 IY=1,INY
      DO 440 IX=1,INX
440   C(IX,IY)=CONI
      GO TO 470
450   CONTINUE
      DO 460 IY=1,INY
460   READ (5,770) (C(IX,IY),IX=1,INX)
470   CONTINUE
      WRITE(6,530)
      DO 480 IY=1,INY
480   WRITE (6,540) (C(IX,IY),IX=1,INX)
      CALL OUTPT(N,NX,NY,THCK,VPRM,HI,HK,WT,REC,AREA,SUMT,TOTLO,S)
      *****
C     COMPUTE ITERATION PARAMETERS
C     NITP NUMBER OF ITERATION PARAMETERS
      DO 490 ID=1,NX
      AOPT(ID)=0.0
490   CONTINUE
      ANITP=NITP-1
      IF (S.EQ.0.0) GO TO 500
      ALPHA1=DEXP(DLOG(2.0/HMIN)/ANITP)
      GO TO 510
500   ALPHA1=DEXP(DLOG(1.0/HMIN)/ANITP)
510   AOPT(1)=HMIN
      DO 520 IP=2,NITP
      AOPT(IP)=AOPT(IP-1)*ALPHA1
520   CONTINUE
C
      WRITE (6,550)
      WRITE (6,560) AOPT
C
      RETURN
C     * * * * *
C
530   FORMAT (1H1,5X,'INITIAL CHEMICAL CONCENTRATION '//)
540   FORMAT(1H ,20F6.0)
550   FORMAT (1H1,20HITERATION PARAMETERS)
560   FORMAT (3H ,10G20.6)
570   FORMAT (1H1,14HTIME INTERVALS)
580   FORMAT (3H ,10G12.5)
590   FORMAT (3H ,20F5.1)
600   FORMAT (1H1,17HAQUIFER THICKNESS)
610   FORMAT (3H ,20F5.2)
620   FORMAT (1H1,30HTRANSMISSIVITY MAP (FT*FT/SEC))
630   FORMAT (20G3.0)
640   FORMAT (20G4.1)
650   FORMAT (1H1,'NODE IDENTIFICATION MAP'//)
660   FORMAT(1H1,'CHEMICAL NODE IDENTIFICATION MAP'//)
670   FORMAT (1H ,40I3)
680   FORMAT (1H1,44HVERTICAL PERMEABILITY/THICKNESS(FT/(FT*SEC)))
690   FORMAT (1H0,12H X-Y SPACING)
700   FORMAT (3H ,10G12.5)
710   FORMAT (1H1,24HPERMEABILTY MAP (FT/SEC))
720   FORMAT (1H0,///10X,'NO. OF FINITE-DIFFERENCE CELLS IN AQUIFER =
1',I4//10X,'AREA OF AQUIFER IN MODEL = ',G12.5,' SQ. FT.')
```

```

770 FORMAT(20G4.0)
780 FORMAT (1H1,11HWATER TABLE)
790 FORMAT (1H ,20F5.0)
800 FORMAT (1H0,10X,'AREA = ',G12.4)
810 FORMAT (2I2)
820 FORMAT (2I2,2G8.2)
830 FORMAT (10A8)
840 FORMAT (1H1,10A8////)
850 FORMAT(8I4,9G5.0,3I1)
860 FORMAT (1H0,21X,'I N P U T      D A T A'////)
870 FORMAT (1H0,23X,'GRID DESCRIPTORS'//13X,'NX      (NUMBER OF ROWS)',5
1X,'=',16/13X,'NY      (NUMBER OF COLUMNS) = ',14/13X,'XDEL (X-DIS
2TANCE IN FEET) = ',F7.1/13X,'YDEL (Y-DISTANCE IN FEET) = ',F7.1//
3)
880 FORMAT (1H0,21X,'TIME PARAMETERS'//12X,' NTIM (MAX. NO. OF TIME
1 STEPS)',7X,'= ',15/13X,'NPMP (NO. OF PUMPING PERIODS)',7X,'= '
2,16/13X,'PINT (PUMPING PERIOD IN YEARS)',6X,'=',F9.1/11X,' TIMX
3 (TIME INCREMENT MULTIPLIER) = ',F9.2/13X,'TINIT (INITIAL TI
4ME STEP IN SEC.) = ',F8.0/13X,'DELT',4X,'TIME INCREMENT FOR CHEM
5',7X,'=',5X,E9.2/13X,'TMAX',4X,'MAX TIME FOR CHEM RUN',9X,'=',E14.
66/13X,'TIMEI',3X,'INCREMENT BETWEEN CHEM PRINT =',E9.2//)
890 FORMAT (1H0,14X,'HYDROLOGIC AND CHEMICAL PARAMETERS'//13X,'S',7X,'
1(STORAGE COEFFICIENT)',7X,'=',5X,F9.6/13X,'POROS (EFFECTIVE PORO
2SITY)',8X,'= ',F8.2/13X,'BETA (CHARACTERISTIC LENGTH) = '
3,F7.1/13X,'DLTRAT (RATIO OF TRANSVERSE TO',21X,'LONGITUDINAL DISP
4ERSIVITY) = ',F9.2/13X,'XKD',5X,'DISTRIBUTION COEFF',10X,'=',5X,F
59.6/13X,'XKIN',4X,'KINETIC RATE CONSTANT',7X,'=',5X,F9.6/13X,'XN',
66X,'ORDER OF REACTION',11X,'=',5X,F9.6/13X,'XRHO',4X,'SOIL WEIGHT
7TO FREE VOL RATIO =',5X,F9.6/13X,'EPS',5X,'ITERATION ERROR',13X,'=
8',5X,E14.3/13X,'WFAC',4X,'RELAXATION COEFFICIENT',6X,'=',5X,F9.6)
900 FORMAT (1H0,21X,'EXECUTION PARAMETERS'//13X,'NITP (NO. OF ITERAT
1ION PARAMETERS) = ',14/13X,'TOL (CONVERGENCE CRITERIA - ADIP) =
2 ',F9.4/13X,'ITMAX (MAX.NO.OF ITERATIONS - ADIP) = ',14//)
910 FORMAT (1H0,23X,'PROGRAM OPTIONS'//13X,'NPNT (PRINT CONTROL IND
1EX) = ',14
2/13X,'NREC (NO. OF PUMPING WELLS) = ',15//)
920 FORMAT (16G5.0)
930 FORMAT (1H-,10X,'LOCATION OF OBSERVATION WELLS'//17X,'NO.',5X,'X',
15X,'Y'//)
940 FORMAT (1H ,16X,I2,5X,I2,4X,I2)
950 FORMAT (1H-,10X,'LOCATION OF PUMPING WELLS'//10X,'X Y RATE(
1IN CFS) CONC. INJECTION(IN CFS) '//)
960 FORMAT (1H ,6X,2I4,3X,F7.2,5X,F7.1,5X,F7.2)
970 FORMAT (11A2)
      END

```

```

C      ****
C      THIS SUBROUTINE CALCULATES VELOCITIES AND DISPERSION COEFFICIENTS
C      OVR A FINITE ELEMENT AREA FOR CUBIC BASIS FUNCTIOND
C
SUBROUTINE VELYG(N,NX,NY,DXX,DYY,DXY,DYX,PERM,
1 HK,DELX,DELY,VX,VY,POROS,ALPHAL,DLTRAT)
C
C      ****
C
REAL *8 HK,DELX,DELY
DIMENSION DXX(NX,NY),DYY(NX,NY),DXY(NX,NY),DYX(NX,NY),
1 PERM(NX,NY),HK(NX,NY),VX(NX,NY),VY(NX,NY)
ALPHAT=ALPHAL*DLTRAT
NNX=NX-1
NNY=NY-1
DO 20 IX=2,NNX
DO 20 IY = 2,NNY
10 GRADX=((HK(IX+1,IY)+HK(IX+1,IY+1))-(HK(IX,IY)+HK(IX,IY+1)
1 ))/2.
GRADY= ((HK(IX,IY+1)+HK(IX+1,IY+1))-(HK(IX,IY)+HK(IX+1,
1 IY)))/2.
PERMX=(PERM(IX,IY)+PERM(IX+1,IY)+PERM(IX,IY+1)+PERM(IX+1,
1 IY+1))/4.
VX(IX,IY)=-PERMX*GRADX/(POROS*DELX)
VX(1X,IY)=0.0
VY(IX,IY)=-PERMX*GRADY/(POROS*DELX)
VXS=VX(IX,IY)**2
VYS=VY(IX,IY)**2
V=SQRT(VXS+VYS)
DXX(IX,IY)=(ALPHAL*VXS+ALPHAT*VYS)/V
DYY(IX,IY)=(ALPHAL*VYS+ALPHAT*VXS)/V
DXY(IX,IY)=(ALPHAL-ALPHAT)*VX(IX,IY)*VY(IX,IY)/V
DYX(IX,IY)=DXY(IX,IY)
20 CONTINUE
C      OUTPUT VELOCITIES AND DISPERSION COEFFICIENTS
IF (N.GT.1) GO TO 90
WRITE (6,100)
DO 30 IY=1,NY
30 WRITE (6,110) (VX(IX,IY),IX=1,NX)
WRITE (6,120)
DO 40 IY=1,NY
40 WRITE (6,110) (VY(IX,IY),IX=1,NX)
WRITE(6,130)
DO 50 IY=1,NY
50 WRITE (6,110) (DXX(IX,IY),IX=1,NX)
WRITE(6,140)
DO 60 IY=1,NY
60 WRITE (6,110) (DXY(IX,IY),IX=1,NX)
WRITE(6,150)
DO 70 IY=1,NY
70 WRITE (6,110) (DYY(IX,IY),IX=1,NX)
WRITE(6,160)
DO 80 IY=1,NY
80 WRITE (6, 110) (DYX(IX,IY),IX=1,NX)
90 CONTINUE
C      ****
C      RETURN
C
100 FORMAT (1H1,12HX VELOCITIES)
110 FORMAT (1H ,12G10.3)

```

```
120 FORMAT (I1,I2HY VELOCITIES)
130 FORMAT (I1,'DXX DISPERSION COEFFICIENTS '//)
140 FORMAT (I1,'DXY DISPERSION COEFFICIENTS '//)
150 FORMAT (I1,'DYY DISPERSION COEFFICIENTS '//)
160 FORMAT (I1,'DXX DISPERSION COEFFICIENTS '//)
    END
```

```

C      *****
C
C      THIS SUBPROGRAM SUBDIVIDES THE FINITE ELEMENTS AROUND
C      SELECTED WELLS FOR MORE REFINED VELOCITY DISTRIBUTIONS
C      SUBROUTINE NODES(NX,NY,TMRX,RECH,REC,TIM,XDEL,YDEL,TOL,
1 HK,XSM,YSM,NODE,PERM,ITMAX,AOPT,S,TOTLQ,NEL,NE,
2 VXS,VYS,DXXS,DYYS,DXYS,DYXS,POROS,ALPHAL,DLTRAI,NGI,NGJ,NGIE,
3 NGJE)
C
C      *****
C
C      REAL *8TMRX,HK,REC,RECH,TIM,AOPT,S,TOL,XDEL,YDEL
C      DOUBLE PRECISION TMRXN,VPRMN,HR,HC,WTN,RECN,RECHN,AREA
1 ,XSM,YSM,HN
C      DIMENSION THCKN(9,9),HR(9,9),HC(9,9),AOPT(7),NE(NGIE,NGJE),
1 NEL(NGIE,NGJE),DXKN(9,9),DYKN(9,9),DXYN(9,9),DYXN(9,9)
C      DIMENSION HK(NX,NY),WTN(9,9),TMRXN(9,9,2),VPRMN(9,9),TMRX(NX,
1 NY,2),RECH(NX,NY),REC(NX,NY),HN(9,9),NODE(NGI,NGJ),PERMN(9,9)
2 ,PERM(NX,NY),RECN(9,9),TIM(100),RECHN(9,9)
C      DIMENSION VXN(9,9),VYN(9,9),VXS(3,3,8),DXXS(3,3,8),DYYS(3,3,8
1 ),DXXS(3,3,8),DYXS(3,3,8),VYS(3,3,8)
C      XSM=XDEL/3.
C      YSM=YDEL/3.
C      FX=XDEL/XSM
C      FY=YDEL/YSM
C
C      FLAG THE ELEMENTS THAT ARE SUBDIVIDED
C      WRITE(6,10) NX,NY
10 FORMAT(1H,'NODE1',2I5)
C      DO 20 IX=1,NGI
C      DO 20 IY=1,NGJ
C      IF(NODE(IX,IY).NE.9) GO TO 20
C      NE(IX,IY)=1
C      NE(IX+1,IY)=1
C      NE(IX,IY+1)=1
C      NE(IX+1,IY+1)=1
20 CONTINUE
C      RENUMBER THE FINITE ELEMENTS TO CORRESPOND TO THE NW NODE
C      NN=0
C      DO 30 IX=1,NGIE
C      DO 30 IY=1,NGJE
C      NEL(IX,IY)=0
C      IF(NEL(IX,IY).EQ.0) GO TO 30
C      NN=NN+1
C      NEL(IX,IY)=NN
30 CONTINUE
C      CHECK TO SEE IF ANY NODES HAVE WELLS IN THEM AND COMPUTE
C      NEW REFINED VELOCITY DISTRIBUTION AROUND THE NODE.
C      DO 320 IX=1,NGI
C      DO 320 IY=1,NGJ
C      IF(NODE(IX,IY).NE.9) GO TO 320
C      IW=IX+2
C      JW=IY+2
C
C      INITIALIZE ALL VARIABLES
C      DO 40 I=1,9
C      DO 40 J=1,9
C      VXN(I,J)=0.0
C      VYN(I,J)=0.0
C      WTN(I,J)=0.0
C      TMRXN(I,J,1)=0.0
C      RECN(I,J)=0.0
C      VPRMN(I,J)=0.0

```

```

TMRXN(I,J,2)=0.0
PERMN(I,J)=0.0
RECHN(I,J)=0.0
THCKN(I,J)=0.0
40 HN(I,J)=0.0
   RECN(5,5)=REC(IW,JW)
   DO 50 I=2,8
   DO 50 J=2,8
50 THCKN(I,J)=1.
   AREA=XSM*YSM

```

```

C
C   CALCULATE NEW HEADS FOR OUTSIDE EDGE NODES BY LINEAR INTERPOLATION

```

```

WTN(2,2)=HK(IW-1,JW-1)
WTN(8,2)=HK(IW+1,JW-1)
WTN(5,2)=HK(IW,JW-1)
WTN(2,5)=HK(IW-1,JW)
WTN(8,5)=HK(IW+1,JW)
WTN(2,8)=HK(IW-1,JW+1)
WTN(8,8)=HK(IW+1,JW+1)
WTN(5,8)=HK(IW,JW+1)
A6=2./3.
A3=1./3.
DIF=WTN(2,2)-WTN(5,2)
WTN(3,2)=A6*DIF+WTN(5,2)
WTN(4,2)=A3*DIF+WTN(5,2)
DIF=WTN(5,2)-WTN(8,2)
WTN(6,2)=A6*DIF+WTN(8,2)
WTN(7,2)=A3*DIF+WTN(8,2)
DIF=WTN(2,2)-WTN(2,5)
WTN(2,3)=A6*DIF+WTN(2,5)
WTN(2,4)=A3*DIF+WTN(2,5)
DIF=WTN(8,2)-WTN(8,5)
WTN(8,3)=A6*DIF+WTN(8,5)
WTN(8,4)=A3*DIF+WTN(8,5)
DIF=WTN(2,5)-WTN(2,8)
WTN(2,6)=A6*DIF+WTN(2,8)
WTN(2,7)=A3*DIF+WTN(2,8)
DIF=WTN(8,5)-WTN(8,8)
WTN(8,6)=A6*DIF+WTN(8,8)
WTN(8,7)=A3*DIF+WTN(8,8)
DIF=WTN(2,8)-WTN(5,8)
WTN(3,8)=A6*DIF+WTN(5,8)
WTN(4,8)=A3*DIF+WTN(5,8)
DIF=WTN(5,8)-WTN(8,8)
WTN(6,8)=A6*DIF+WTN(8,8)
WTN(7,8)=A3*DIF+WTN(8,8)

```

```

C
C   CALCULATE PARAMETERS FOR NEW NODAL SYSTEM

```

```

DO 60 I=1,5
DO 60 J=1,5
PERMN(I,J)=PERM(IW-1,JW-1)
TMRXN(I,J,1)=TMRX(IW-1,JW-1,1)*FX
60 TMRXN(I,J,2)=TMRX(IW-1,JW-1,2)*FY
DO 70 I=6,9
DO 70 J=1,5
PERMN(I,J)=PERM(IW,JW-1)
TMRXN(I,J,1)=TMRX(IW,JW-1,1)*FX
70 TMRXN(I,J,2)=TMRX(IW,JW-1,2)*FY
DO 80 I=1,5
DO 80 J=6,9
PERMN(I,J)=PERM(IW-1,JW-1)

```

```

      TMRXN(I,J,1)=TMRX(IW-1,JW,1)*FX
80  TMRXN(I,J,2)=TMRX(IW-1,JW,2)*FY
      DO 90 I=6,9
      DO 90 J=6,9
      PERMN(I,J)=PERM(IW,JW)
      TMRXN(I,J,1)=TMRX(IW,JW,1)*FX
90  TMRXN(I,J,2)=TMRX(IW,JW,2)*FY
      DO 100 I=2,8
      VPRMN(I,2)=1.
100  VPRMN(I,8)=1.
      DO 110 J=2,8
      VPRMN(2,J)=1.
110  VPRMN(8,J)=1.
C
      WRITE(6,120) IW,JW,XSM,YSM
120  FORMAT(1H1,'COMPUTED VALUES FOR NODE',I3,',',I3/10X,'XSM = ',
1  G10.4/10X,'YSM = ',G10.4//)
      WRITE(6,130) WTN
130  FORMAT(1H0,5X,'INITIAL WATER TABLE CONDS',//(9G14.5))
      WRITE(6,140) PERMN
140  FORMAT(1H0,5X,'PERMN VALUES'//(9G14.5))
      DO 150 J=1,9
150  WRITE(6,160) (TMRXN(I,J,1),I=1,9)
160  FORMAT(1H0,9G14.5)
      WRITE(6,170) VPRMN
170  FORMAT(1H0,5X,'VPRMN'//(9G14.5))
      DO 180 I=1,9
      DO 180 J=1,9
      HR(I,J)=WTN(I,J)
      HC(I,J)=WTN(I,J)
180  HN(I,J)=WTN(I,J)
C
      CALCULATE THE NEW HEAD DISTRIBUTION
C
C
190  FORMAT(1H , 'NODE2',2I5)
      CALL ITERAT(7,9,9,ITMAX,THCKN,TMRXN,
1  VPRMN,HR,HC,HN,WTN,RECN,RECHN,TIM,AOPT,XSM,YSM,S,AREA,TOL,TOTLQ
2  ,1)
200  FORMAT(1H , 'NODE3',2I5)
C
      CALCULATE VELOCITIES AND DISPERSION COEFFICIENTS AT THE
C
      NEW SUBSPACES
      CALL VELYG(1,9,9,DXXN,DYYN,DXYN,DYXN,PERMN,HN,
1  XSM,YSM,VXN,VYN,POROS,ALPHAL,DLTRAT)
C
      WRITE(6,210) NX,NY
210  FORMAT(1H , 'NODE4',2I5)
C
      RENUMBER THE VELOCITY AND DISPERSION COEFFICIENTS TO
C
      CORRESPOND TO THE FINITE ELEMENTS.
      DO 280 NQ=1,4
      II=0
      JJ=0
      GO TO (220,230,240,250),NQ
220  IS=2
      IE=4
      JS=2
      JE=4
      K=NFL(IW-2,JW-2)
      GO TO 260
230  IS=5
      IE=7
      JS=2

```

```

      JE=4
      K=NFL(IW-1,JW-2)
      GO TO 260
240  IS=2
      IE=4
      JS=5
      JE=7
      K=NFL(IW-2,JW-1)
      GO TO 260
250  IS=5
      IE=7
      JS=5
      JE=7
      K=NFL(IW-1,JW-1)
      GO TO 260
260  CONTINUE
      DO 270 I=IS,IE
          II=I+1
          JJ=0
          DO 270 J=JS,JE
              JJ=J+1
              VXS(II,JJ,K)=VXN(I,J)
              VYS(II,JJ,K)=VYN(I,J)
              DXXS(II,JJ,K)=DXXN(I,J)
              D YYS(II,JJ,K)=DYYN(I,J)
              D XYS(II,JJ,K)=DXYN(I,J)
              D YXS(II,JJ,K)=DYXN(I,J)
270  CONTINUE
280  CONTINUE
C      SET THE OLD COMPUTED NODES EQUAL TO THE NEW SMALLER COMPUTED
C      NODES
      HK(IW,JW)=HN(5,5)
C      PRINT NEW HEAD VALUES AROUND NODE
      WRITE(6,290) IW,JW,HN
290  FORMAT(1H0,'NEW HEAD VALUES FOR NODE ',I3,', ',I3,'/(9G14.5)')
C      PRINT THE NEW VELOCITY AND DISPERSION COEFFICIENTS FOR
C      THE FINITE ELEMENTS
      DO 300 K=1,4
          WRITE(6,310) ((VXS(I,J,K),I=1,3),J=1,3)
          WRITE(6,310) ((DXXS(I,J,K),I=1,3),J=1,3)
          WRITE(6,310) ((D YYS(I,J,K),I=1,3),J=1,3)
          WRITE(6,310) ((D XYS(I,J,K),I=1,3),J=1,3)
          WRITE(6,310) ((VYS(I,J,K),I=1,3),J=1,3)
300  CONTINUE
310  FORMAT(1H0,3G14.5)
320  CONTINUE
      WRITE(6,330) NX,NY
330  FORMAT(1H , 'NODE6',2I5)
340  CONTINUE
      RETURN
      END

```

```

C      *****
C
C      THIS SUBROUTINE CALCULATES THE CHANGES IN CONCENTRATION
C      USING THE GALERKIN FINITE ELEMENT TECHNIQUE WITH CURIC
C      BASIS FUNCTIONS.
C
      SUBROUTINE TRSPTC(NODE,XKD,XKIN,XN,XRHO,DXX,DYY,DXY,DYX,QI,
1  SYMBOL,DELTA,TMAX,TIMEI,C,CONK,PMP,NX,NY,NNX,NNY,THCK,RECH,
2  XDFL,YDEL,VX,VY,CNRECH,POROS,INX,INY,XSM,VXS,VYS,
3  DXS,DYS,DXYS,DYXS,NEL,IIX,NGIE,NGJE)
C      *****
      DOUBLE PRECISION YY,R,X,A
      DOUBLE PRECISION DMIN1,DEXP,DLOG,DABS
      REAL *8 RECH,XDEL,YDEL,XSM,DELX,DELY
C
      DIMENSION NODE(INX,INY),DXX(NX,NY),DYY(NX,NY),DXY(NX,NY),
1  DYX(NX,NY),QI(NX,NY),SYMBOL(11),C(INX,INY),PMP(NX,NY),
2  THCK(NX,NY),RECH(NX,NY),VX(NX,NY),VY(NX,NY),CNRECH(NX,NY)
C      *****
C      THESE DIMENSION STATEMENTS MUST BE CHANGED WITH ARRAY CHANGES
C      DIMENSION B(224),IB( 20,20),R(224),YY(2066),A (224),
1  X( 12,3,224),Y(224,6,6)
C      *****
      DIMENSION AM(6,6,2,2,4),XDXX(6,6,2,2,4),XDYY(6,6,2,2,4),
1  XDXY(6,6,2,2,4),XDYX(6,6,2,2,4),XVX(6,6,2,2,4),XVY(6,6,2,2,4)
      DIMENSION AMT(6,6),XDXT(6,6),XDYTT(6,6),XDXYT(6,6)
1  ,XDYXT(6,6),XVXT(6,6),XVYT(6,6)
      DIMENSION VXS(3,3,8),VYS(3,3,8),DXXS(3,3,8),DYYS(3,3,8),
1  DXYS(3,3,8),DYXS(3,3,8),NEL(NGIE,NGJE)
      WRITE(6,10) NEL
10  FORMAT(1H ,20I4)
      NGJ=INY
      NGI=INX
      NI=NX
      NJ=NY
      POR=POROS
      DELX=XDEL
      DELY=YDEL
      TIMEP=0.
      TIME=0.
      N=0
      KIKJ=4*NGI*NGJ
      Y1=1+KIKJ
      Y2=Y1+KIKJ
      Y3=Y2+KIKJ
      Y4=Y3+KIKJ
      Y5=Y4+KIKJ
      Y6=Y5+4*KIKJ
      NIJ=2*NGJ
      NII=2*NGI
      DO 20 INI=1,KIKJ
      R(INI)=0.
      A(INI)=0.
      DO 20 K=1,6
      DO 20 L=1,6
20  Y(INI,K,L)=0.0
      DO 30 INI=1,KIKJ
      DO 30 K=1,3
      DO 30 I=1,12
30  X(I,K,INI)=0.00

```

```

C      CALCULATE MATRIX COEFFICIENTS FOR EACH NODE
40 CONTINUE
   CALL MATCF2(NGI,NGJ,KIKJ,AM,XDXX,XDYY,XDXY,XDYX,XVX,XVY,DXX,DYY,
   1 DXY,DYX,VX,VY,X,Y,NODE,DELT,QI,THCK,POROS,NI,NJ,DELX,XSM,VXS,
   2 VYS,DXXS,DYYS,DXYS,DYXS,NEL,NGIE,NGJE)
50 CONTINUE
C      PASS DIMENSIONS TO SUBROUTINES
   CALL DIMEN(NGI,NGJ)
C      SCALE THE ASSOCIATED MATRIX
   CALL SCALE(X,YY(1))
   WRITE(6,70)
60 CONTINUE
   WRITE(6,70)
   N=N+1
   TIME=TIME+DELT
   IF(TIME.GT.TMAX) GO TO 140
   WRITE(6,70)
70 FORMAT(1H1)
C      CALCULATE THE RIGHT SIDE OF THE COEFF MATRIX
   CALL RSIDE(NGI,NGJ,NX,NY,KIKJ,NODE,R,A,Y,QI,CNRECH,
   1 DELT,POROS,THCK)
C      SCALE THE RSIDE OF THE MATRIX
   CALL RSCALE(R,YY(1))
C      SOLVE THE SYSTEM OF LINEAR EQUATIONS
   CALL TCHEB(X,A,R,YY(Y1),YY(Y2),YY(Y3),YY(Y4),
   1 YY(Y5),YY(Y6),1.0-1,20,100,2.39,1.596,IIX)
C
C      UNSCALE THE SOLUTION RESULTS
   CALL RSCALE(A,YY(1))
C
C      *****
80 FORMAT(1H ,12E11.4)
   TIMEP = TIMEP+DELT
   IF(TIMEP.LT.TIMEI) GO TO 130
   TIMEP=0.0
   WRITE(6,170) N,TIME
   WRITE(6,70)
   LJ=0
   DO 100 J=1,NGJ
   DO 100 L=1,2
   LJ=LJ+1
   LL=0
   DO 90 I=1,NGI
   DO 90 K=1,2
   LL=LL+1
   INI=(I-1)*NGJ*4+(J-1)*4+(L-1)*2 +K
   B(LL)=A(INI)
   IB(LL,LJ)=B(LL)+.1
90 CONTINUE
   IF(L.EQ.2) GO TO 100
   WRITE(6,80)(B(LL),LL=1,NII,2)
100 CONTINUE
   WRITE(6,70)
   DO 110 J=1,NIJ,2
110 WRITE(6,120)(IB(LL,J),LL=1,NII,2)
120 FORMAT(1H ,32I4)
130 GO TO 60
140 CONTINUE
150 CONTINUE
160 FORMAT(1H ,11E12.4)
170 FORMAT(1H ,4X,'N = ',I5/ 5X,'TIME = ',E10.3//)

```

T-1798

STOP
END

```

C      *****
C
C      THIS SUBROUTINE CALCULATES THE MATRIX COEFFICIENTS FOR CUBIC
C      FINITE ELEMENT SYSTEMS.
C
C      SUBROUTINE MATCF2(NGI,NGJ,KIKJ,AM,XDXX,XDYY, XDXY,XDYX,XVX,XVY,
1 DXX,DYY,DXY,DYX,VX,VY,X,Y,NODE,DELTA,QI,THCK,POFOS,NI,NJ,DELX,
2 XSM,VXS,VYS,DXXS,DYYS,DXYS,DYXS,NEL,NGIE,NGJE)
C
C      *****
C
C      DOUBLE PRECISION DELX,XSM,X
C      DIMENSION VXS(3,3,8),VYS(3,3,8),DXXS(3,3,8),DYYS(3,3,8),
1 DXXS(3,3,8),DYXS(3,3,8),NEL(NGIE,NGJE)
C      DIMENSION AM(6,6,2,2,4),XDXX(6,6,2,2,4),XDYY(6,6,2,2,4),
1 XDXY(6,6,2,2,4),XDYX(6,6,2,2,4),XVX(6,6,2,2,4),XVY(6,6,2,2,4),
2 DXX(NI,NJ),DYY(NI,NJ),DXY(NI,NJ),DYX(NI,NJ),
3 VX(NI,NJ),VY(NI,NJ),AMT(6,6),XDXXT(6,6),XDYYT(6,6),XDXYT(6,6),
4 XDYXT(6,6),XVXT(6,6),XVYT(6,6),X(12,3,KIKJ),Y(KIKJ,6,6),
5 NODE(NGI,NGJ),QI(NI,NJ),THCK(NI,NJ)
C      INI=0
C      DO 210 IKKKI=1,NGI
C      DO 210 IKKKJ=1,NGJ
C      KKKI=IKKKI
C      KKKJ=IKKKJ
C      DO 210 IL=1,2
C      DO 210 IK=1,2
C      L=1L
C      K=1K
C      INI=INI+1
C      KK=NODE(KKKI,KKKJ)+1
C      GO TO(10,20,30,40,50,60,70,80,90,10,190,180),KK
10 CONTINUE
C      IF(KKKJ.EQ.1) GO TO 60
C      IF(KKKJ.EQ.NGJ) GO TO 80
C      IF(KKKI.EQ.1) GO TO 90
C      IF(KKKI.EQ.NGI) GO TO 70
C      IS=1
C      IE=6
C      JS=1
C      JE=6
C      GO TO 100
20 IS=3
C      IE=6
C      JS=3
C      JE=6
C      GO TO 100
30 IS=1
C      IE=4
C      JS=3
C      JE=6
C      GO TO 100
40 IS=1
C      IE=4
C      JS=1
C      JE=4
C      GO TO 100
50 IS=3
C      IE=6
C      JS=1
C      JE=4

```

```

        GO TO 100
60  IS=1
    IE=6
    JS=3
    JE=6
    GO TO 100
70  IS=1
    IE=4
    JS=1
    JE=6
    GO TO 100
80  IS=1
    IE=6
    JS=1
    JE=4
    GO TO 100
90  IS=3
    IE=6
    JS=1
    JE=6
    GO TO 100
100 CONTINUE
C    EVALUATE THE INTEGRALS BY GAUSSIAN QUADRATURE
    CALL INTEG2(AM,XDXX,XDYY,XDXY,XDYX,XVX,
1  XVY,NI,NJ,NGI,NGJ,KKKI,KKKJ,DXX,DYY,DXY,DYX,VX,VY,K,L,VXS,
2  VYS,DXXS,DYYS,DXYS,DYXS,NEL,XSM,DELX,NODE,NGIE,NGJE)
110 CONTINUE
    DO 120 I=1,6
    DO 120 J=JS,JE
    AMT(I,J)=0.0
    XDXXT(I,J)=0.0
    XDYYT(I,J)=0.0
    XDXYT(I,J)=0.0
    XDYXT(I,J)=0.0
    XVXT(I,J)=0.0
    XVYT(I,J)=0.0
120 CONTINUE
C    SUM UP THE INTEGRAL VALUES FOR THE FOUR FINITE ELEMENTS
C    SURROUNDING THE NODE.
    DO 150 I=IS,IE
    DO 150 J=JS,JE
    DO 140 N=1,4
    AMT(I,J)=AMT(I,J)+AM(I,J,K,L,N)
    XDXXT(I,J)=XDXXT(I,J)+XDXX(I,J,K,L,N)
    XDYYT(I,J)=XDYYT(I,J)+XDYY(I,J,K,L,N)
    XDXYT(I,J)=XDXYT(I,J)+XDXY(I,J,K,L,N)
    XDYXT(I,J)=XDYXT(I,J)+XDYX(I,J,K,L,N)
    XVXT(I,J)=XVXT(I,J)+XVX(I,J,K,L,N)
    XVYT(I,J)=XVYT(I,J)+XVY(I,J,K,L,N)
C    WRITE(6,811) I,J,AMT(I,J),XDXXT(I,J),XDYYT(I,J),XDXYT(I,J),
C    1 XDYXT(I,J),XVXT(I,J),XVYT(I,J)
130 FORMAT(1H ,2I3,7E10.3)
140 CONTINUE
150 CONTINUE
C    REORDER THE MATRIX COEFFICIENTS SO THEY CAN BE USED BY
C    TCHEB AND PERFORM CRANK-NICOLSON TIME INCREMENTIZATION
C    ON THEM
160 CONTINUE
    IA=-1
    IIB=0
    DO 170 KKK=1,3

```

```

IA=IA+2
IIB=IIB+2
DO 170 J=JS,JE
DO 170 I=IA,IIB
IC=IIB/2
JC=(J-1)*2+I-IIB+2
AAA=DELTA/2.*(XDXXT(I,J)+XDYYT(
1 I,J)+XDXYT(I,J)+XDYXT(I,J)+XVXT(I,J)+XVYT(I,J))
X(JC,IC,INI)=AMT(I,J)+AAA
Y(INI,I,J)=AMT(I,J)-AAA
170 CONTINUE
IF (CI(KKKI+2,KKKJ+2).LE.0.) GO TO 210
IF (K.EQ.2.OR.L.EQ.2) GO TO 210
XA=CI(KKKI+2,KKKJ+2)*DELTA/(2.*THCK(KKKI+2,KKKJ+2)*PCROS)
Y(INI,K+2,L+2)=Y(INI,K+2,L+2)-XA
X(5,2,INI)=X(5,2,INI)+XA
GO TO 210
180 CONTINUE
IF (K.EQ.2.OR.L.EQ.2) GO TO 10
190 DO 200 KX=1,3
DO 200 J=1,12
200 X(J,KX,INI)=0.00
J=4+(L-1)*2+K
X(J,2,INI)=1.00
210 CONTINUE
RETURN
END

```

```

C *****
C
C THIS SUBROUTINE CALCULATES THE DOUBLE INTEGRALS FOR THE GALERKIN
C FINITE ELEMENTS FOR THE CUBIC FUNCTIONS.
C
SUBROUTINE INTEG2 (AM, XDXX, XDYY, XDXY, XDYX, XVX, XVY, NI, NJ, NGI, NGJ,
1 KKKI, KKKJ, DXX, DYY, DXY, DYX, VX, VY, K, L, VXS, VYS, DXXS, DYYS, DXYS,
2 DYXS, NEL, XSM, XDEL, NODE, NGIE, NGJE)
C *****
C
DOUBLE PRECISION XSM, XDEL
DIMENSION AM(6,6,2,2,4), XDXX(6,6,2,2,4), XDYY(6,6,2,2,4),
1 XDXY(6,6,2,2,4), XDYX(6,6,2,2,4), XVX(6,6,2,2,4), XVY(6,6,2,2,4)
DIMENSION DXX(NI, NJ), DYY(NI, NJ), DXY(NI, NJ), DYX(NI, NJ),
1 VX(NI, NJ), VY(NI, NJ), VXS(3,3,8), VYS(3,3,8), DXXS(3,3,8),
2 DYYS(3,3,8), DXYS(3,3,8), DYXS(3,3,8), NEL(NGIE, NGJE), NODE(NGI, NGJ)
COMMON H, ICH
EXTERNAL FCT, FCTD
H=XDEL
C
C CALCULATE INTEGRAL VALUES
DO 10 M=1,4
DO 10 I=1,6
DO 10 J=1,6
AM(I, J, K, L, M)=0.0
XDXX(I, J, K, L, M)=0.0
XDYY(I, J, K, L, M)=0.0
XDXY(I, J, K, L, M)=0.0
XDYX(I, J, K, L, M)=0.0
XVX(I, J, K, L, M)=0.0
XVY(I, J, K, L, M)=0.0
10 CONTINUE
C
C CALCULATE INTEGRAL VALUES
DO 100 ME=1,4
KK=0
M=ME
GO TO(20,30,40,50),M
20 CONTINUE
IA=KKKI
JA=KKKJ
KK=NEL(IA, JA)
IS=1
IE=4
JS=1
JE=4
IF(KK.GT.0) GO TO 80
GO TO 60
30 CONTINUE
IA=KKKI+1
JA=KKKJ
KK=NEL(IA, JA)
IS=3
IE=6
JS=1
JE=4
IF(KK.GT.0) GO TO 80
GO TO 60
40 CONTINUE
IA=KKKI
JA=KKKJ+1
KK=NEL(IA, JA)

```

```

      IS=1
      IE=4
      JS=3
      JE=6
      IF (KK.GT.0) GO TO 80
      GO TO 60
50  CONTINUE
      IA=KKKI+1
      JA=KKKJ+1
      KK=NEL(IA,JA)
      IS=3
      IE=6
      JS=3
      JE=6
      IF (KK.GT.0) GO TO 80
60  CONTINUE
C    USE THIS INTEGRATION PROCEDURE IF THE PARAMETER IS
C    TAKEN AS THAT AT THE CENTER OF THE NODE.
      JA=JA+1
      IA=IA+1
      H=XDEL
      DO 70 IIE=IS,IE
      I=IIE
      DO 70 JJE=JS,JE
      J=JJE
      AM(I,J,K,L,M)=GAUSS(FCT,FCT,FCT,FCT,I,J,K,L,M)
      XDXx(I,J,K,L,M)=GAUSS(FCTD,FCT,FCTD,FCT,I,J,K,L,M)*DXX(IA,JA)
      XDYY(I,J,K,L,M)=GAUSS(FCT,FCTD,FCT,FCTD,I,J,K,L,M)*DYY(IA,JA)
      XDXY(I,J,K,L,M)=GAUSS(FCTD,FCT,FCT,FCTD,I,J,K,L,M)*DXY(IA,JA)
      XDYx(I,J,K,L,M)=GAUSS(FCT,FCTD,FCTD,FCT,I,J,K,L,M)*DYX(IA,JA)
      XVX(I,J,K,L,M)=GAUSS(FCTD,FCT,FCT,FCT,I,J,K,L,M)*VX(IA,JA)
      XVY(I,J,K,L,M)=GAUSS(FCT,FCTD,FCT,FCT,I,J,K,L,M)*VY(IA,JA)
70  CONTINUE
      GO TO 100
80  CONTINUE
C    USE THIS INTEGRATION PROCEDURE IF PARAMETER VALUES
C    AT EACH GAUSS POINT ARE USED.
      DO 90 IIE=IS,IE
      I=IIE
      DO 90 JJE=JS,JE
      J=JJE
      H=XDEL
      AM(I,J,K,L,M)=GAUSS(FCT,FCT,FCT,FCT,I,J,K,L,M)
      XDXx(I,J,K,L,M)=GAUSSD(FCTD,FCT,FCTD,FCT,I,J,K,L,M,DXXS,KK)
      XDYY(I,J,K,L,M)=GAUSSD(FCT,FCTD,FCT,FCTD,I,J,K,L,M,DYYS,KK)
      XDXY(I,J,K,L,M)=GAUSSD(FCTD,FCT,FCT,FCTD,I,J,K,L,M,DXYS,KK)
      XDYx(I,J,K,L,M)=GAUSSD(FCT,FCTD,FCTD,FCT,I,J,K,L,M,DYXS,KK)
      XVX(I,J,K,L,M)=GAUSSD(FCTD,FCT,FCT,FCT,I,J,K,L,M,VXS,KK)
      XVY(I,J,K,L,M)=GAUSSD(FCT,FCTD,FCT,FCT,I,J,K,L,M,VYS,KK)
      ICH=I+J+M
90  CONTINUE
100 CONTINUE
      RETURN
      END
C    *****
C    THIS FUNCTION CALCULATES THE VALUE OF THE CUBIC
C    POLYNOMIAL OVER THE INTERVAL.
C
      FUNCTION FCT(Z,IJ,N)
      COMMON H,ICH
10  GO TO (20,80,20,80),N

```



```

C      IF(J.EQ.2) STOP
CONTINUE
RETURN
END
C      ****
C
C      THIS FUNCTION PERFORMS GAUSSIAN QUADRATURE OVER THE ELEMENT
C      WITH PARAMETER EVALUATION AT THE GAUSS POINTS.
C
FUNCTION GAUSSD(FCTI,FCTJ,FCTK,FCTL,I,J,K,L,N,PAR,KK)
DIMENSION PAR(3,3,8)
COMMON H
KN=2*K
LN=2*L
A=.5*H
B=H
M=N
IF(M.EQ.2.OR.M.EQ.3) M=M+1
C=.3872983*B
X3=FCTI(A+C,I,N)*FCTK(A+C,KN,N)
X2=FCTI(A,I,N)*FCTK(A,KN,N)
X1=FCTI(A-C,I,N)*FCTK(A-C,KN,N)
Y1=FCTJ(A-C,J,M)*FCTL(A-C,LN,M)
Y2=FCTJ(A,J,M)*FCTL(A,LN,M)
Y3=FCTJ(A+C,J,M)*FCTL(A+C,LN,M)
A1=.2777778
A3=.2777778
A2=.4444444
B2=.4444444
B3=.2777778
B1=.2777778
GAUSSD=B*B*(A1*B1*X1*Y1*PAR(1,1,KK)+A2*B1*X2*Y1*PAR(2,1,KK)
1 +A3*B1*X3*Y1*PAR(3,1,KK)+A1*B2*X1*Y2*PAR(1,2,KK)
2 +A2*B2*X2*Y2*PAR(2,2,KK)+A3*B2*X3*Y2*PAR(3,2,KK)+
3 A1*B3*X1*Y3*PAR(1,3,KK)+A2*B3*X2*Y3*PAR(2,3,KK)+A3*B3*X3*Y3*
4 PAR(3,3,KK))
10 CONTINUE
20 FORMAT(1H ,'PAR1',(3E12.4))
RETURN
END

```

```

C *****
C
C THIS SUBROUTINE CALCULATES THE KNOWN RIGHT HAND
C SIDE VECTOR.
C
C SUBROUTINE RSIDE(NGI,NGJ,NX,NY,KIKJ,NODE,R,A,Y,QI,CNRECH,
1 DELT,POROS,THCK)
C *****
C
C DOUBLE PRECISION A,R
C DIMENSION A(KIKJ),R(KIKJ),Y(KIKJ,6,6),QI(NX,NY),CNRECH(NX,NY),
1 THCK(NX,NY),NODE(NGI,NGJ)
C INI=0
C DO 150 I=1,NGI
C DO 150 J=1,NGJ
C KK=NODE(I,J)+1
C DO 150 L=1,2
C DO 150 K=1,2
C INI=INI+1
C R(INI)=0.0
C GO TO (10,20,30,40,50,60,70,80,90,10,130,140),KK
C THIS IS A CENTER NODE
C 10 CONTINUE
C IF(J.EQ.1) GO TO 60
C ISTART=1
C ISTOP=6
C JSTART=1
C JSTOP=6
C GO TO 100
C THIS IS A NW NODE
C 20 ISTART=3
C ISTOP=6
C JSTART=3
C JSTOP=6
C GO TO 100
C THIS IS A NE NODE
C 30 ISTART=1
C ISTOP=4
C JSTART=3
C JSTOP=6
C GO TO 100
C THIS IS A SE NODE
C 40 ISTART=1
C ISTOP=4
C JSTART=1
C JSTOP=4
C GO TO 100
C THIS IS A SW NODE
C 50 ISTART=3
C ISTOP=6
C JSTART=1
C JSTOP=4
C GO TO 100
C THESE ARE N NODES
C 60 ISTART=1
C ISTOP=6
C JSTART=3
C JSTOP=6
C GO TO 100
C THESE ARE E NODES

```

```

70 ISTART=1
   ISTOP=4
   JSTART=1
   JSTOP=6
   GO TO 100
C   THESE ARE S NODES
80 ISTART=1
   ISTOP=6
   JSTART=1
   JSTOP=4
   GO TO 100
C   THESE ARE W NODES
90 ISTART=3
   ISTOP=6
   JSTART=1
   JSTOP=6
   GO TO 100
100 CONTINUE
    DO 120 II=ISTART,ISTOP
      ICAL=I+(II-1)/2-1
      IIP=1
      IF (II/2*2.EQ.II) IIP=2
      DO 110 JJ=JSTART,JSTOP
        JCAL=J+(JJ-1)/2-1
        JIP=1
        IF (JJ/2*2.EQ.JJ) JIP=2
        IIN=(ICAL-1)*NGJ*4+(JCAL-1)*4+(JIP-1)*2+IIP
        R(INI)=R(INI)+A(IIN)*Y(INI,II,JJ)
110 CONTINUE
120 CONTINUE
    IF (QI(I+2,J+2).LE.0.0) GO TO 150
    IF (K.EQ.2.OR.L.EQ.2) GO TO 150
    R(INI)=R(INI)+CNRECH(I+2,J+2)*QI(I+2,J+2)*DELTA/(POROS*
1 THCK(I+2,J+2))
C   R(INI)=CNRECH(I+2,J+2)*DELTA
    GO TO 150
130 CONTINUE
    R(INI)=0.D0
    GO TO 150
140 CONTINUE
    IF (K.EQ.2.OR.L.EQ.2) GO TO 60
    R(INI)=100.
150 CONTINUE
160 CONTINUE
    RETURN
    END

```

```

      SUBROUTINE DIMEN(NNC,NNR)
C*****
C THIS SUBROUTINE SETS THE DIMENSION PARAMETERS THAT ARE *
C PASSED IN COMMON /DIMCOM/ N,NC,NR,NR4. *
C           N IS THE NUMBER OF UNKNOWNNS *
C           NC THE NUMBER OF COLUMNS *
C           NR THE NUMBER OF ROWS *
C*****
      IMPLICIT REAL*8(A-H,O-Z)
      COMMON /DIMCOM/ N,NC,NR,NR4
      COMMON /AREA1/ NI,ND

C
      NC= NNC
      NR=NNR
      N = 4*NR*NC
      NR4 = 4*NR
      NI = N
      ND = 36

C
      RETURN
      END
      SUBROUTINE SCALE(A,D)
C*****
C THIS SUBROUTINE SCALES THE MATRIX A BY A DIAGONAL TRANS- *
C FORMATION. THE SCALED MATRIX IS DAD WHERE D IS A DIAGONAL *
C MATRIX AND D(I) = 1/SQRT( DIAG OF A). IF ANY DIAGONAL *
C ELEMENT OF A IS NOT POSITIVE, THEN AN ERROR MESSAGE IS *
C PRINTED OUT AND D(I) IS SET EQUAL TO 1. *
C*****
      IMPLICIT REAL*8(A-H,O-Z)
      COMMON /DIMCOM/ N,NC,NR,NR4
      DIMENSION A(12,3,N),D(N)

C
C      INDICIES:      IC IS THE COLUMN
C                   IR IS THE ROW
C                   IU IS THE UNKOWN
C
C      SET DIMENSION PARAMETERS
      NC1 = NC - 1
      NR1 = NR - 1

C
C      SCALE: PART I
C
C      INITIALIZE I
      I=0

C
      IC = 1
      IR = 1
      DO 40 IU=1,4
      I=I+1
      J= MOD((I-1),4)+5
      IF(A(J,2,I).GT.0.D0) GO TO 10
      GO TO 30

C
      THEN:
      10 D(I)=1/DSQRT(A(J,2,I))
      DO 20 K=2,3
      DO 20 KK=5,12
      A(KK,K,I)=A(KK,K,I)*D(I)
      20 CONTINUE
      GO TO 40

C
      ELSE: PRINT WARNING

```

```

30  WRITE(6,540) I,A(J,2,I)
    D(I)=1.00
C
40  CONTINUE
C
    DO 90 IR=2,NR1
DO 80 IU=1,4
    I=I+1
    J= MOD((I-1),4)+5
    IF(A(J,2,I).GT.0.00) GO TO 50
GO TO 70
C
    THEN:
50  D(I)=1/DSQRT(A(J,2,I))
DO 60 K=2,3
    DO 60 KK=1,12
    A(KK,K,I)=A(KK,K,I)*D(I)
60  CONTINUE
GO TO 80
C
    ELSE: PRINT WARNING
70  WRITE(6,540) I,A(J,2,I)
    D(I)=1.00
C
80  CONTINUE
C
90  CONTINUE
C
    IR = NR
DO 130 IU=1,4
    I=I+1
    J= MOD((I-1),4)+5
    IF(A(J,2,I).GT.0.00) GO TO 100
GO TO 120
C
    THEN:
100 D(I)=1/DSQRT(A(J,2,I))
DO 110 K=2,3
    DO 110 KK=1,8
    A(KK,K,I)=A(KK,K,I)*D(I)
110 CONTINUE
GO TO 130
C
    ELSE: PRINT WARNING
120 WRITE(6,540) I,A(J,2,I)
    D(I)=1.00
C
130 CONTINUE
C
DO 270 IC=2,NC1
    IR = 1
DO 170 IU=1,4
    I=I+1
    J= MOD((I-1),4)+5
    IF(A(J,2,I).GT.0.00) GO TO 140
GO TO 160
C
    THEN:
140 D(I)=1/DSQRT(A(J,2,I))
DO 150 K=1,3
    DO 150 KK=5,12
    A(KK,K,I)=A(KK,K,I)*D(I)
150 CONTINUE
GO TO 170
C
    ELSE: PRINT WARNING

```

```

160 WRITE(6,540) I,A(J,2,I)
    D(I)=1.D0
C
170 CONTINUE
C
    DO 220 IR=2,NR1
    DO 210 IU=1,4
        I=I+1
        J= MOD((I-1),4)+5
        IF(A(J,2,I).GT.0.D0) GO TO 180
    GO TO 200
C
        THEN:
180 D(I)=1/DSQRT(A(J,2,I))
    DO 190 K=1,3
        DO 190 KK=1,12
            A(KK,K,I)=A(KK,K,I)*D(I)
190 CONTINUE
    GO TO 210
C
        ELSE: PRINT WARNING
200 WRITE(6,540) I,A(J,2,I)
    D(I)=1.D0
C
210 CONTINUE
C
220 CONTINUE
C
    IR = NR
    DO 260 IU=1,4
        I=I+1
        J= MOD((I-1),4)+5
        IF(A(J,2,I).GT.0.D0) GO TO 230
    GO TO 250
C
        THEN:
230 D(I)=1/DSQRT(A(J,2,I))
    DO 240 K=1,3
        DO 240 KK=1,8
            A(KK,K,I)=A(KK,K,I)*D(I)
240 CONTINUE
    GO TO 260
C
        ELSE: PRINT WARNING
250 WRITE(6,540) I,A(J,2,I)
    D(I)=1.D0
C
260 CONTINUE
C
270 CONTINUE
C
    IC=NC
    IR=1
    DO 310 IU=1,4
        I=I+1
        J= MOD((I-1),4)+5
        IF(A(J,2,I).GT.0.D0) GO TO 280
    GO TO 300
C
        THEN:
280 D(I)=1/DSQRT(A(J,2,I))
    DO 290 K=1,2
        DO 290 KK=5,12
            A(KK,K,I)=A(KK,K,I)*D(I)
290 CONTINUE

```

```

      GO TO 310
C      ELSE: PRINT WARNING
C 300  WRITE(6,540) I,A(J,2,I)
      D(I)=1.00
C
C 310  CONTINUE
C
      DO 360 IR=2,NR1
DO 350 IU=1,4
      I=I+1
      J= MOD((I-1),4)+5
      IF(A(J,2,I).GT.0.D0) GO TO 320
      GO TO 340
C      THEN:
C 320  D(I)=1/DSQRT(A(J,2,I))
      DO 330 K=1,2
      DO 330 KK=1,12
      A(KK,K,I)=A(KK,K,I)*D(I)
C 330  CONTINUE
      GO TO 350
C      ELSE: PRINT WARNING
C 340  WRITE(6,540) I,A(J,2,I)
      D(I)=1.00
C
C 350  CONTINUE
C
C 360  CONTINUE
C
      IR=NR
DO 400 IU=1,4
      I=I+1
      J= MOD((I-1),4)+5
      IF(A(J,2,I).GT.0.D0) GO TO 370
      GO TO 390
C      THEN:
C 370  D(I)=1/DSQRT(A(J,2,I))
      DO 380 K=1,2
      DO 380 KK=1,8
      A(KK,K,I)=A(KK,K,I)*D(I)
C 380  CONTINUE
      GO TO 400
C      ELSE: PRINT WARNING
C 390  WRITE(6,540) I,A(J,2,I)
      D(I)=1.00
C
C 400  CONTINUE
C
C
C
C      SCALE: PART II
C
C      INITIALIZE POINTER
      IP=0
      IPD=-4
C
C  IC = 1
      IR = 1
DO 410 IU=1,4
      IA=IP+IU
      DO 410 J=2,3
      JD=IPD+(J-2)*NR4

```

```

      DO 410 I=5,12
        ID=JD+I
        A(I,J,IA)=A(I,J,IA)*D(ID)
410  CONTINUE
C
      DO 430 IR=2,NR1
        IP0=IP
        IP=IP+4
        DO 420 IU=1,4
          IA=IP+IU
          DO 420 J=2,3
            J0=IP0+(J-2)*NR4
            DO 420 I=1,12
              ID=JD+I
              A(I,J,IA)=A(I,J,IA)*D(ID)
420  CONTINUE
C
430  CONTINUE
C
      IR = NR
      IP0=IP
      IP=IP+4
      DO 440 IU=1,4
        IA=IP+IU
        DO 440 J=2,3
          J0=IP0+(J-2)*NR4
          DO 440 I=1,8
            ID=JD+I
            A(I,J,IA)=A(I,J,IA)*D(ID)
440  CONTINUE
C
C
      DO 490 IC=2,NC1
        IR = 1
        IP0=IP
        IP=IP+4
        DO 450 IU=1,4
          IA=IP+IU
          DO 450 J=1,3
            J0=IP0+(J-2)*NR4
            DO 450 I=5,12
              ID=JD+I
              A(I,J,IA)=A(I,J,IA)*D(ID)
450  CONTINUE
C
      DO 470 IR=2,NR1
        IP0=IP
        IP=IP+4
        DO 460 IU=1,4
          IA=IP+IU
          DO 460 J=1,3
            J0=IP0+(J-2)*NR4
            DO 460 I=1,12
              ID=JD+I
              A(I,J,IA)=A(I,J,IA)*D(ID)
460  CONTINUE
C
470  CONTINUE
C
      IR = NR
      IP0=IP

```

```

      IP=IP+4
      DO 480 IU=1,4
        IA=IP+IU
        DO 480 J=1,3
          JD=IPD+(J-2)*NR4
          DO 480 I=1,8
            ID=JD+I
            A(I,J,IA)=A(I,J,IA)*D(ID)
480    CONTINUE
C
490 CONTINUE
C
C
      IC = NC
      IR = 1
      IPD=IP
      IP=IP+4
      DO 500 IU=1,4
        IA=IP+IU
        DO 500 J=1,2
          JD=IPD+(J-2)*NR4
          DO 500 I=5,12
            ID=JD+I
            A(I,J,IA)=A(I,J,IA)*D(ID)
500    CONTINUE
C
      DO 520 IR=2,NR1
        IPD=IP
        IP=IP+4
        DO 510 IU=1,4
          IA=IP+IU
          DO 510 J=1,2
            JD=IPD+(J-2)*NR4
            DO 510 I=1,12
              ID=JD+I
              A(I,J,IA)=A(I,J,IA)*D(ID)
510    CONTINUE
C
520    CONTINUE
C
      IR = NR
      IPD=IP
      IP=IP+4
      DO 530 IU=1,4
        IA=IP+IU
        DO 530 J=1,2
          JD=IPD+(J-2)*NR4
          DO 530 I=1,8
            ID=JD+I
            A(I,J,IA)=A(I,J,IA)*D(ID)
530    CONTINUE
C
C
540 FORMAT('0 WARNING: THE',IS,'TH DIAGONAL ELEMENT OF THE
1 MATRIX IS',D12.5,'< 0')
C
      RETURN
      END
      SUBROUTINE RSCALE(X,D)
C*****
C THIS SUBROUTINE RESCALES THE VECTOR X, RETURNING THE PRODUCT *

```

```

C OF THE DIAGONAL MATRIX D*X.
C*****
  IMPLICIT REAL*8(A-H,O-Z)
  COMMON /DIMCOM/ N,NC,NR,NR4
  DIMENSION X(N),D(N)
C
  DO 10 I=1,N
    X(I) = D(I)*X(I)
  10 CONTINUE
C
  RETURN
  END
  SUBROUTINE MATMUL(A,X,Y)
C*****
C THIS SUBROUTINE PERFORMS THE MATRIX VECTOR MULTIPLICATION: *
C AX = Y, WHERE A IS A NXN MATRIX AND X AND Y ARE VECTORS. *
C*****
  IMPLICIT REAL*8(A-H,O-Z)
  COMMON /DIMCOM/ N,NC,NR,NR4
  DIMENSION A(12,3,N),X(NR4,NC),Y(N)
C
C BEGIN: IC IS THE COLUMN
C IR IS THE ROW
C IU IS THE UNKNOWN
C
C INITIALIZE Y
  DO 10 I=1,N
    Y(I) = 0.0D0
  10 CONTINUE
C
C SET DIMENSION PARAMETERS
  NC1 = NC - 1
  NR1 = NR - 1
C
  IC = 1
C INITIALIZE POINTERS
  IC2 = IC - 2
  IPX = -4
  IPY = 0
C
  IR = 1
  DO 20 IU=1,4
    IY=IPY+IU
    DO 20 J=2,3
      JX=IC2+J
      DO 20 I=5,12
        IX=IPX+I
        Y(IY)=Y(IY)+A(I,J,IY)*X(IX,JX)
      20 CONTINUE
C
      DO 30 IR=2,NR1
        IPX=IPX+4
        IPY=IPY+4
        DO 30 IU=1,4
          IY=IPY+IU
          DO 30 J=2,3
            JX=IC2+J
            DO 30 I=1,12
              IX=IPX+I
              Y(IY)=Y(IY)+A(I,J,IY)*X(IX,JX)
            30 CONTINUE

```

```

C
      IR = NR
      IPX=IPX+4
      IPY=IPY+4
      DO 40 IU=1,4
          IY=IPY+IU
          DO 40 J=2,3
              JY=IC2+J
              DO 40 I=1,8
                  IX=IPX+I
                  Y(IY)=Y(IY)+A(I,J,IY)*X(IX,JX)
40    CONTINUE
C
C      DO 80 IC=2,NC1
C
C          INITIALIZE POINTERS
      IC2=IC-2
      IPX=-4
C
      IR = 1
      IPY=IPY+4
      DO 50 IU=1,4
          IY=IPY+IU
          DO 50 J=1,3
              JX=IC2+J
              DO 50 I=5,12
                  IX=IPX+I
                  Y(IY)=Y(IY)+A(I,J,IY)*X(IX,JX)
50    CONTINUE
C
      DO 60 IR=2,NR1
      IPX=IPX+4
      IPY=IPY+4
      DO 60 IU=1,4
          IY=IPY+IU
          DO 60 J=1,3
              JX=IC2+J
              DO 60 I=1,12
                  IX=IPX+I
                  Y(IY)=Y(IY)+A(I,J,IY)*X(IX,JX)
60    CONTINUE
C
      IR = NR
      IPX=IPX+4
      IPY=IPY+4
      DO 70 IU=1,4
          IY=IPY+IU
          DO 70 J=1,3
              JX=IC2+J
              DO 70 I=1,8
                  IX=IPX+I
                  Y(IY)=Y(IY)+A(I,J,IY)*X(IX,JX)
70    CONTINUE
C
80    CONTINUE
C
      IC = NC
C
          INITIALIZE POINTER

```

```
      IPX=-4
      IC2=IC-2
C
      IR = 1
      IPY=IPY+4
      DO 90 IU=1,4
         IY=IPY+IU
         DO 90 J=1,2
            JX=IC2+J
            DO 90 I=5,12
               IX=IPX+I
               Y(IY)=Y(IY)+A(I,J,IY)*X(IX,JX)
90      CONTINUE
C
      DO 100 IR=2,NR1
      IPX=IPX+4
      IPY=IPY+4
      DO 100 IU=1,4
         IY=IPY+IU
         DO 100 J=1,2
            JX=IC2+J
            DO 100 I=1,12
               IX=IPX+I
               Y(IY)=Y(IY)+A(I,J,IY)*X(IX,JX)
100     CONTINUE
C
      IR=NR
      IPX=IPX+4
      IPY=IPY+4
      DO 110 IU=1,4
         IY=IPY+IU
         DO 110 J=1,2
            JX=IC2+J
            DO 110 I=1,8
               IX=IPX+I
               Y(IY)=Y(IY)+A(I,J,IY)*X(IX,JX)
110     CONTINUE
C
      RETURN
      END
```